United Business Media

REVIEWED MICROSOFT PROJECT 2007

AUGUST 2008

gamedeveloper

THE LEADING GAME INDUSTRY MAGAZINE

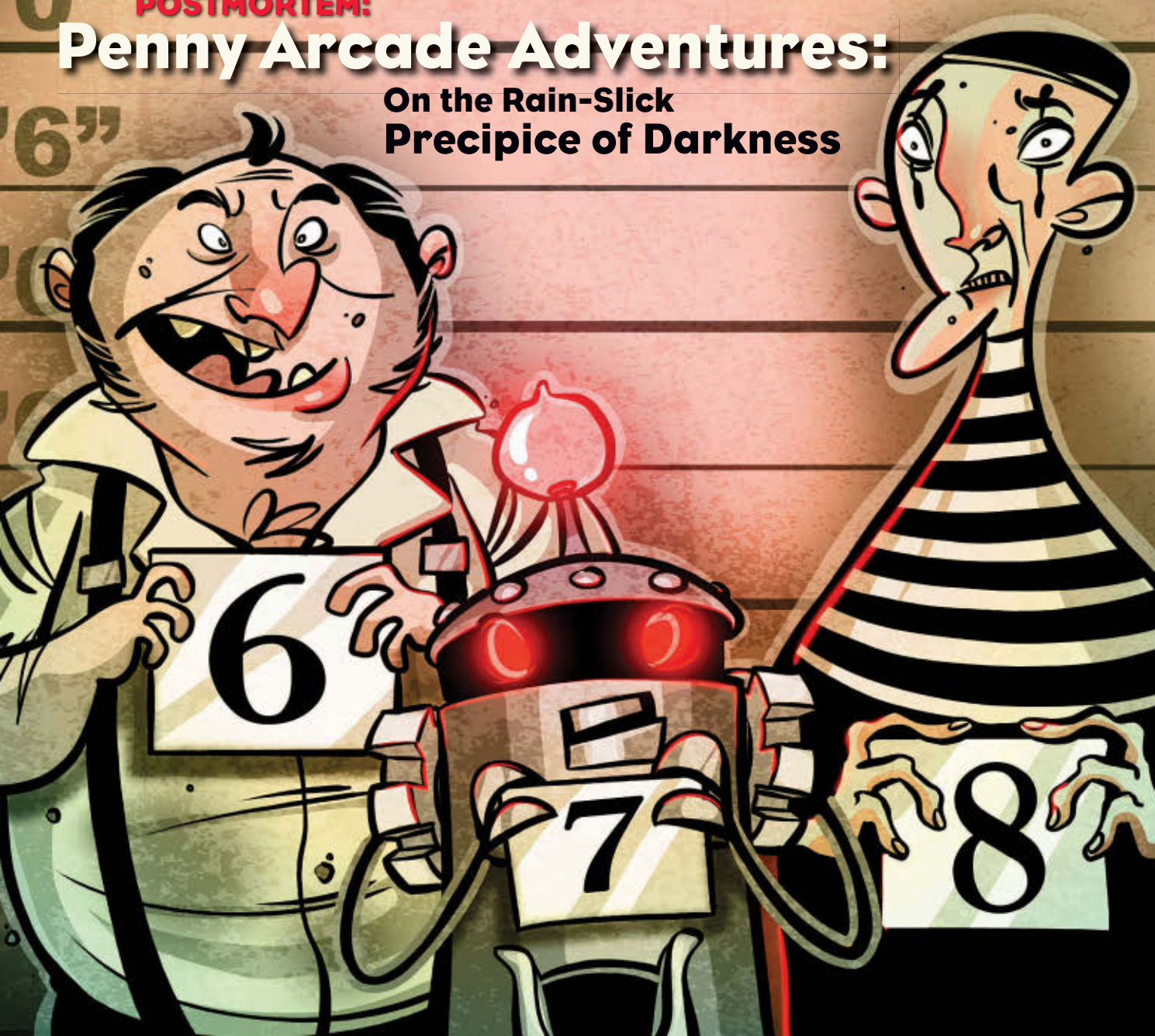
AI! AI! AI! AI!
ARTIFICIAL INTELLIGENCE MIDDLEWARE ROUNDUP

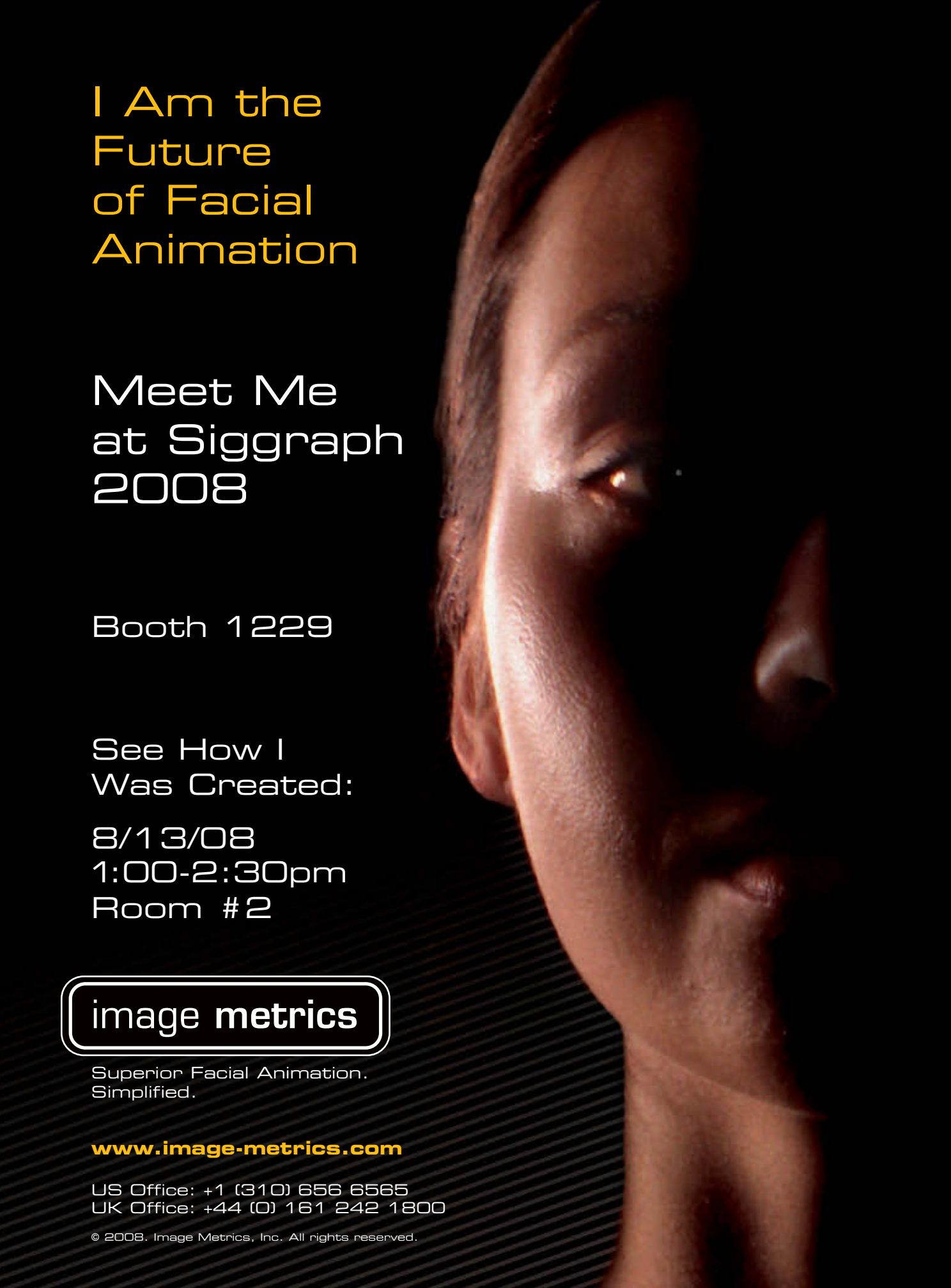
ALL FIRED UP
THE SHOOTER AND SHOOTEE DISCONNECT

INTERVIEW
HIROKAZU YASUHARA ON GAME DESIGN METHODS

POSTMORTEM:

Penny Arcade Adventures:
On the Rain-Slick Precipice of Darkness





I Am the Future of Facial Animation

Meet Me at Siggraph 2008

Booth 1229

See How I Was Created:

8/13/08
1:00-2:30pm
Room #2

image **metrics**

Superior Facial Animation.
Simplified.

www.image-metrics.com

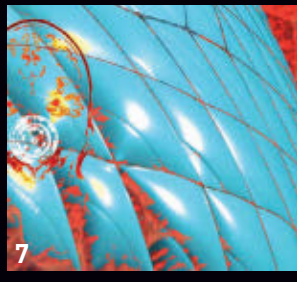
US Office: +1 (310) 656 6565
UK Office: +44 (0) 161 242 1800

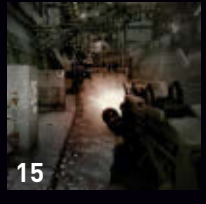
gamedeveloper

AUGUST 2008
VOLUME 15, NUMBER 7

CONTENTS

FEATURES



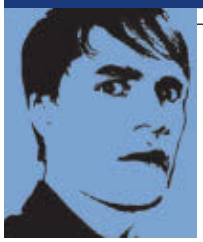


POSTMORTEM

DEPARTMENTS

COVER ART: MIKE KRAHULIK

COLUMNS



THE GRAPHICS PLATEAU

DEVELOPERS AND JOURNALISTS ALIKE HAVE talked about the inevitable point past which graphics do not matter, and the focus turns to gameplay. The question I pose to you is—has that already happened? Were we just not listening?

I was considering this when confronted with a few facts from Japan. The most popular "modern" consoles out there are the DS, the Wii, and the PSP. None of these consoles have the graphics push of the big boys, as we know. Further, Koei recently released DYNASTY WARRIORS 6, "exclusively" for PS3, but (rumor has it) due to fan reaction, subsequently ported it to PS2. Atlus is still releasing its largest product, PERSONA 4, on the PS2.

In the U.S., the DS is doing famously, the Wii has sold like gangbusters, and PSP hardware (though perhaps not official software) is doing quite well. The PS2 still has the largest installed base in the country. Now does that mean the Xbox 360 and PS3 are doomed? Certainly not. But I propose the possibility that PS2 and Wii-level graphics were and are enough for the average gamer.

WHITHER ART THOU, EXCLUSIVES?

METAL GEAR SOLID 4 was a big console mover for Sony, and the game sold over a million copies worldwide. But how integral were the graphics and tech to that experience, really? If MGS 4 had been released on PS2, how many copies would it have sold? Certainly not less, and quite possibly more.

This comes to mind now that Square Enix announced during Microsoft's E3 press conference that it would be bringing FINAL FANTASY XIII to the Xbox 360 day and date with the PS3 version in North America. The hype of exclusivity, or even special graphics hardware features has essentially come to naught. The phrase "only on PS3" means even less now than it did then, before we really knew whether SPUs were magical fairies that could handle all our various processes. Now that we know we can make essentially equivalent products across both of the high-end consoles, and indeed on PC, that rhetoric starts to fall by the wayside.

Although I am not proclaiming the death of the next-gen console, I do think that the era of graphics wars is gone, for your average consumer. Would anyone have complained if GTA IV had been released on PS2 or a machine with similar graphic fidelity? I doubt it—everyone would have been able to like or dislike it as much as they did the next-gen versions. Graphics don't make that game fun, and it is not nearly the best looking game on either the 360 or PS3—yet nobody minds. The fact that Rockstar is releasing a GTA on the DS only pushes this idea further.

WHAT OF BLU-RAY?

Blu-Ray won the high-definition media wars—but what does that really mean? A recent comment posted on Gamasutra noted that traditional media formats, from music to movies, are all quickly shuffling online. Sony has effectively won a war that is no longer being fought. It's been demonstrated time and again that the mainstream user is willing to watch streamed videos of movies on YouTube, or torrent them on The Pirate Bay, or even download them at only slightly lower quality from legitimate portals like the Xbox 360 or Netflix. The high end isn't going to be supplanted by the low—there are people who want the highest definition everything. But there are a lot of people for whom it's just not the largest concern anymore, especially as the market broadens.

AND THEN THE PC?

I've made much ado about the potential of the PC to retake the mainstream market, and I will reiterate the fact that the casual PC market is booming, while developers like Crytek feel they can no longer play to the high-end PC consumers, as the market simply isn't there (or when it is, it's through piracy). The PC is the place where this postulate holds the most water. The majority of gamers on PC these days do not need the highest-level graphics. WORLD OF WARCRAFT is a great example, and the multitudinous casual games only put mortar on the bricks.

TECH JUNKIES

Who is pushing this graphics and tech thing anyway? Isn't it just the people who want to sell tech? I don't think it's the average consumer. The average consumer doesn't complain about the graphics on the Wii, because they know what to expect, and understand the approach. The best innovations today are coming in terms of gameplay implementation. Cover mechanics, intuitive UI and HUDs, natural in-game tutorials, and persistent worlds are just a few examples. All of these things can be done on the most recent generations. While the tech of the PS3 and 360 can make streaming and seamless worlds much easier, a lot of this can be done to users' satisfaction on the lower-end.

It turns out the average consumer of today does not necessarily want a Ferrari hooked up to his or her entertainment system, to paraphrase our production editor Jeffrey Fleming. The average consumer is content with the Toyota Corolla of videogame systems, and for that reason, I propose that the war of bigger and badder graphics can safely end, and we can finally focus on pushing gameplay to the fore.

— Brandon Sheffield

gamedeveloper
www.gdmag.com

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

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

EDITORIAL
PUBLISHER
Simon Carless scarless@gdmag.com
EDITOR-IN-CHIEF
Brandon Sheffield bsheffield@gdmag.com
SENIOR CONTRIBUTING EDITOR
Jill Duffy jduffy@gdmag.com
ART DIRECTOR
Gin Gravette ggravette@gdmag.com
PRODUCTION EDITOR
Jeffrey Fleming jfleming@gdmag.com
CONTRIBUTING COPY EDITOR
Della Wyser
CONTRIBUTING EDITORS
Jesse Harlin jharlin@gdmag.com
Steve Theodore stheodore@gdmag.com
Noel Llopis nllopis@gdmag.com
Soren Johnson sjohnson@gdmag.com
Damion Schubert dschubert@gdmag.com
ADVISORY BOARD
Hal Barwood Designer-at-Large
Ellen Guon Beeman Microsoft
Brad Bulkley Neversoft
Clinton Keith High Moon Studios
Ryan Lesser Harmonix
Mark DeLoura Ubisoft

ADVERTISING SALES
MEDIA ACCOUNT MANAGER
John Malik Watson e: jmwatson@think-services.com t: 415.947.6224
GLOBAL SALES MANAGER, RECRUITMENT & EDUCATION
Aaron Murawski e: amurawski@think-services.com t: 415.947.6227
ACCOUNT MANAGER, EDUCATION AND RECRUITMENT
Gina Gross e: ggross@think-services.com t: 415.947.6241
SR. EVENTS ACCOUNT MANAGER, SOUTHWEST
Jasmin Davé
ACCOUNT MANAGER, WESTERN CANADA, INDIA, AUSTRALIA, & ASIA
Amanda Mae Miller

ADVERTISING PRODUCTION
PRODUCTION MANAGER Pete C. Scibilia e: pscibili@ubm-us.com

REPRINTS
PARS INTERNATIONAL
Joe Nunziata t: 212.221.9595 e: reprints@parsintl.com

THINK SERVICES
CEO THINK SERVICES Philip Chapnick
GROUP DIRECTOR Kathy Schoback
CREATIVE DIRECTOR Cliff Scorso

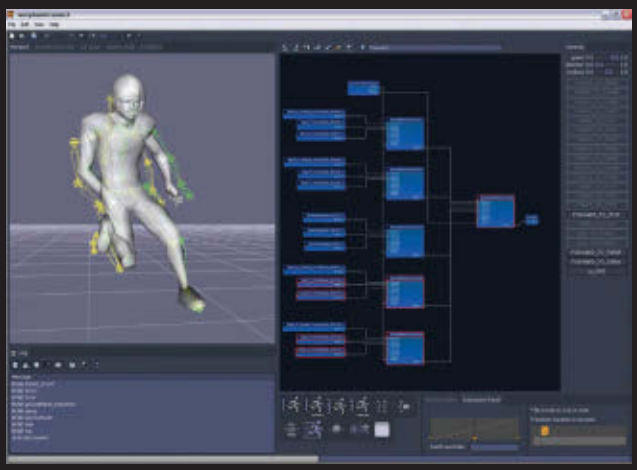
AUDIENCE DEVELOPMENT
GROUP DIRECTOR Kathy Henry e: khenry@techinsights.com
DIRECTOR Kristi Cunningham e: kcunningham@techinsights.com
LIST RENTAL Merit Direct LLC t: 914.368.1000

MARKETING
SERVICES MARKETING DIRECTOR Karen Tom e: ktom@think-services.com
SERVICES MARKETING COORDINATOR Laura Robison e: lrobison@think-services.com

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

United Business Media

morpheme™
advanced animation system



morpheme is the industry's first graphically authorable animation engine. morpheme consists of morpheme:runtime: an advanced runtime animation engine for PLAYSTATION®3, Xbox 360™, Wii™ and PC. morpheme:connect: a highly-customizable 3D authoring application.

morpheme gives animators and developers unprecedented control over the look and feel of their animations in-game: blends, transitions, compression, etc. can all be previewed and modified graphically in morpheme:connect and live on the target platform.

morpheme:runtime ships with full source code and integrates seamlessly with euphoria, NaturalMotion's Dynamic Motion Synthesis technology.

For more information, visit www.naturalmotion.com

blend tree

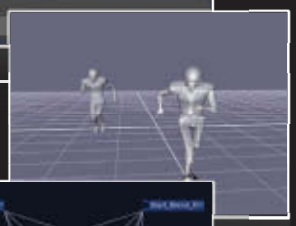
Advanced graphical tools for building complex blend trees. Real-time visualization of animation source contribution through node highlighting

blending

Graphical control of transition blending between states in the transition graph

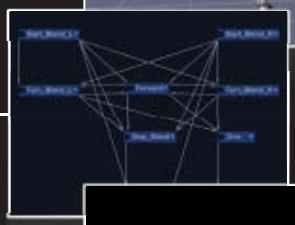
multiple characters

Visualization of multiple runtime characters in morpheme:connect for easy authoring and analysis of character interaction



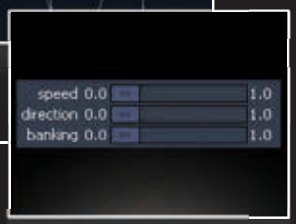
network

Advanced graphical tools for creating and visualizing transition networks through drag-and-drop



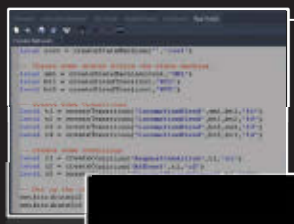
control parameters

Exposure of custom high-level controls for entire animation system. Real-time manipulation through sliders or game pad controller



scripting

Full Lua scripting for automating tasks, adding AI logic or polling game pads for real-time input

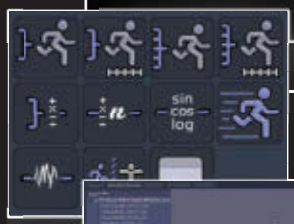


timeline

Graphical mark up of animation data to add one-shot and duration events, for highlighting footfalls, sound effects, etc.

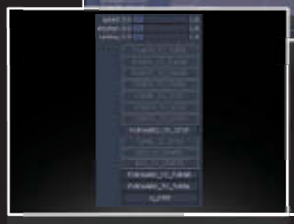
node palette

Advanced blend notes for dragging and dropping into transition network. Fully customizable node types through C++ and scripting



animation browser

Easy browsing and selection (drag & drop) of source animation. Animation list is automatically updated to reflect changed source files



transition requests

Exposure of custom transition messages. In-tool emulation of interaction between morpheme:runtime and game AI system

NATURALMOTION

OVERHEARD AT PARIS GDC

OLIVIER LEJADE

CREATIVE DIRECTOR OF SOUL BUBBLES AND FOUNDER OF MEKENSLEEP

ON THE TROUBLE WITH REVIEWS: "We've tried to basically bridge the gap between casual gamers and hardcore gamers, and the interesting part is that even though the reviewers found it too easy, they still enjoyed it very much. And that's what they say. I'm talking about the most critical reviews. They say that they liked the game a lot, but they mark us down two to three points sometimes just because they found the game too easy, and I think that's a problem. Because if the only thing you have to say about a game—if you say I had a really great time, but it was too short, or a little too easy, I don't think that's a big enough criticism to mark down a game the way they do it.

"Another thing that skews reviews the wrong way is plus and minus points, because if you use plus and minus points, you have to find minus points, and you're going to be honing in on something that might be totally minor to the actual gaming experience, and putting that in the minus points, which makes a big deal out of it when it (may not be) really."

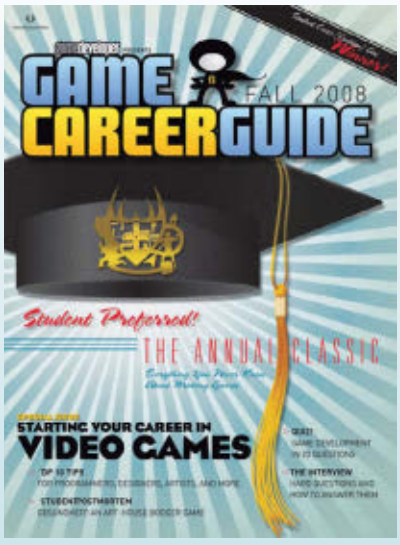
DAVID CAGE

FOUNDER OF QUANTIC DREAM (INDIGO PROPHECY)

ON THE SUBJECT OF REALISM VERSUS ABSTRACTION IN GAMES: "I think it's easier for the major part of an audience to relate to something that looks real, as opposed to something that is totally out of their ... I wouldn't say this is my personal opinion, because as an educated gamer I can relate to anything basically as long as it's done with taste and talent. But I think that a lot of games explore realism, and it's easier for a lot of players to relate to something that looks close to what they know.

"I think the cinema would be incredibly limited if only *The Incredibles* were available, if you see what I mean. And it's the same with games, there's room for different styles, and different stories to be told. I think that rendering is not an end in itself, and when you're a developer or a creator, you don't wake up in the morning and say 'I'm a creator of non-realistic worlds,' or 'realistic worlds.' It all depends on your vision. 'What do I have to tell to the world?' Then you think, 'What is the best way of telling this story? Is it realistic, is it non-realistic?' And I'm quite agnostic about that. I'm not saying everything should be realistic or non-realistic, it all depends on what you have to say."

GAME DEVELOPER'S CAREER GUIDE 2008 FREE TO DOWNLOAD



FOR THE FIRST TIME, *Game Developer's* annual *Game Career Guide* is being given away for free. The special magazine, which is devoted to helping aspiring video game creators and guided by the editors of the GameCareerGuide.com website, is now available as a digital version at http://gamedeveloper.texterity.com/gamedeveloper/2008careerguide with both web-readable and PDF-downloadable versions to choose from. The *Game Career Guide* issue includes a version of *Game Developer*'s 7th annual salary report for entry-level jobs in video game development, as well as numerous articles with tips on breaking into the industry, penned by former or current staffers from Secret Level, High Moon, Vicarious Visions, and Linden Lab. Also included in the 2008 edition is a postmortem of Matt Hammill's notable student game GESUNDHEIT!, as well as detailed advice on how to answer ten frequently asked questions in game development job interviews.

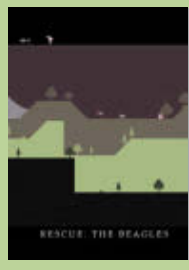
New this year is the Student Survey, which asked students what they like and don't like about their game-related scholastic experiences. This provides a unique aggregate impression of the game school—from traditional full-time university, to specialized game programs within those universities, through to dedicated game or art schools. There's also a description of the major events and organizations within the game industry, the different job titles developers may have, a lighthearted game development quiz testing how much you know about video games from a developer's perspective, and an index of more than 100 colleges and universities offering programs and degrees in game-related studies.

While the *Game Career Guide* is clearly aimed at students, professional game developers may take an interest in it to learn what is guiding the developers of tomorrow. The *Game Career Guide* is now available for digital download, and physical versions of the magazine will be available for free at major game-related events over the next few months, including SIGGRAPH, Penny Arcade Expo, E For All, Austin GDC, Game Developers Conference 2009, and more.

—Staff

Top Ten Indie Freeware Downloads 2008: The Best So Far

Most of you probably make games that cost money but there's a universe of great free games out there. Here, Wee Tim Boon, editor of Think Services' Indiegames.com web site, takes us through some of the top releases so far this year.

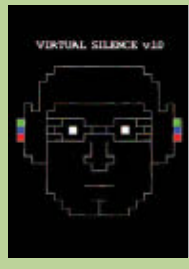


RESCUE: THE BEAGLES
Nenad Jalsovec
www.16x16.org/category/rescue-the-beagles
RESCUE: THE BEAGLES is one of many retro-styled arcade games in this list. The action involves stepping into the tiny shoes of Edwin or Nicole as they embark on a mission to save cute little pooches from the hands of evil researchers and their henchmen.

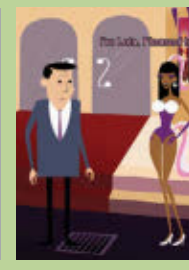
PSYCHOSOMNIUM
Cactus
http://64digits.com/users/cactus/psycho1.zip
This subversive platformer places players in a sparse but colorful dream world filled with a cast of surreal characters.
(also check out: SEIZUREDOME, SHOTGUN NINJA)

KAROSHI 2
Jesse Venbrux
www.venbrux.com/index.php?page=games&content=karoshi
Killing the hero instead of saving the world or rescuing the princess? Sign us up! Plenty of challenging puzzles to solve in each stage.

ROM CHECK FAIL
Farbs
www.farbs.org/games.html
Lots of ripped sprites and game play rules from arcade classics mashed together to produce amusing results. Plus the game itself is actually quite fun. Available for both Windows and Linux.



VIRTUAL SILENCE
Erkka Virtanen and Tuukka Virtanen
http://koti.mbnet.fi/erkkavir/VirtualSilence.zip
VIRTUAL SILENCE is an action game where the player takes control of a young boy named Jason. He must overcome a series of experimental tests in a virtual world under the watchful eyes of his caring mother and an unnamed doctor.
Comes with an epilepsy warning.



DIRTY SPLIT
Uwe Sittig
www.dreamagination.org
A point and click adventure game with gorgeous artwork, a catchy soundtrack, and professional voice acting. A private investigator named Baxter attempts to solve a criminal case that will require journeys to three different cities in order to gather new clues and evidence.

SAMURAI RAILROAD MANSION
Lurk
http://host-a.net/lurk/SRM1_02.zip
An action game with more than a passing resemblance to Nintendo's light gun classic HOGAN'S ALLEY. Slash your way through eight stages' worth of ninjas, shoguns, daimyos, shape shifters, and even a couple of evil spirits.

DYSON
Alex May, Rudolf Kremers, Brian Grainger
www.deadrock-game.com/files/Dyson.rar
DYSON is a real-time strategy game that is not unlike Phil Hassey's GALCON (www.imitationpickles.org/galcon/index.html). Take control of an entire asteroid belt by conquering other asteroids with your seedlings.

MIGHTY JILL OFF
dessgeega
http://mightyjilloff.dessgeega.com
Dessgeega's tribute to the fantastic BOMB JACK series, created with collaborative efforts from James Harvey and Andrew Toups. Not for kids!

HARPOONED
Conor O'Kane
http://harpooned.org
Agitprop pretending to be a Cetacean Research Simulator disguised as a vertical shooter. Play the role of a Japanese scientist performing explosive and bloody "research" on whales around Antarctica.

—Wee Tim Boon

CALENDAR

G03 Electronic & Entertainment Expo 2008
Perth Convention Exhibition Centre
Perth
Western Australia
August 1–3, 2008
Price: $193 USD
www.go3.com.au

SIGGRAPH 2008
Los Angeles Convention Center
Los Angeles, CA
August 11–15, 2008
Price: $450–$1,125
www.siggraph.org/s2008

3rd ACM International Conference on Digital Interactive Media in Entertainment and Arts
Athens Information Technology, Athens
Greece
September 10–12, 2008
Price: $795 USD
www.ait.edu.gr

Austin GDC
Austin Convention Center
Austin, TX
September 15–17, 2008
Price: $695–$995
www.austingdc.net

Games Convention Asia 2008
Suntec International Convention & Exhibition Centre
Singapore
September 18–20, 2008
Price: $162 USD
www.gc-asia.sg

Unreal Technology News

by Mark Rein, Epic Games, Inc.

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

Epic's Unreal Engine 3 has won Game Developer Magazine's Best Engine Front Line Award for the past three years, and "Gears of War," the 2006 Game of the Year, sold 5 million units on Xbox 360 and PC.

Epic recently shipped "Unreal Tournament 3" for PC, PlayStation 3 and Xbox 360. "Gears of War 2" for Xbox 360 is scheduled for release in November.

Upcoming Epic Attended Events:

SIGGRAPH
Los Angeles, CA
August 11-14, 2008

GC Developers Conference
Leipzig, Germany
August 18-20, 2008

NVISION
San Jose, CA
August 25-27, 2008

CEDEC
Tokyo, Japan
September 9-11, 2008

Please email:
mrein@epicgames.com
for appointments.

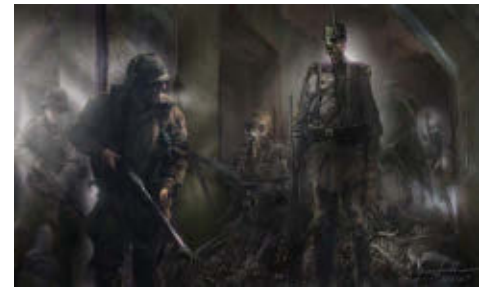
POWERED BY UNREAL TECHNOLOGY

BLOCKADE ENTERTAINMENT PAVES A 'SACRED ROAD' WITH UNREAL ENGINE 3

Blockade Entertainment is an innovative studio conceived to create compelling animated experiences for television, online and film using modern game engine technology and video game assets. Founded by Brad Foxhoven, with partner Gearbox Software in tow, licensing Unreal Engine 3 has been a natural step in Blockade's quest to realizing its creative vision.

With a successful proof of concept and a national commercial already under its belt, Blockade is now creating its first animated series, "Sacred Road," using assets from Gearbox Software's *Brothers in Arms* series.

"One of the big advantages that Unreal Engine 3 brings to developing material for video or any other linear medium is the extremely fast iteration time," said Foxhoven. "You can see the effects of changes you make almost instantly, which allows for very fast fine-tuning of many aspects of our work, particularly when it comes to camera work, post-process effects, and lighting. Being able to replay your work in real time and adjust lighting, animation, or post-process accordingly saves tons of time over a more traditional renderer."



Concept art for Blockade Entertainment's "Sacred Road"

Foxhoven's team has relied on many Unreal Engine 3 tools, including Matinee, which has been used for all of the scene setup and control. Blockade leverages the HDR rendering technology to create a smoother, more efficient pipeline. The team also utilizes the Cascade visual editor and particle system as well as the engine's post-process tools for incorporating motion blur, depth of field, and various lighting effects in the show.

"Unreal Engine 3 allows us to be more efficient in various stages of our production, giving us more time and resources to be experimental on how we approach each episode," explained Foxhoven. "We can stage and edit various aspects of the production, and not lose out on our timeline."

Foxhoven said the biggest advantage to adapting Unreal Engine 3 for Hollywood is its speed.

"UE3 allows directors to create multiple takes, and edit those takes on the fly," he said. "They can see their changes when they want them, allowing the production to continue to move ahead, and not wait for extended rendering to occur. Plus, the cost benefit allows for more shows to be produced in this format, giving a broader basis of opportunity with multiple distribution partners."

Despite the lower production cost, Foxhoven said "Sacred Road" will have the appearance of the best cinematics in today's most technologically advanced games. He added that the show looks like a fully rendered CG production, and the visual quality is spectacular.

The Blockade production team has already ramped up from 12 people to 50, and in addition to this first series, additional 3D shows will be created using Unreal Engine technology down the line.

"We have now created a very robust pipeline that allows us to scale up for additional projects relatively quickly," added Foxhoven. "The engine enables us to keep a consistent pipeline on what we bring into the company, and how we export the shows and their multiple formats. We are currently targeting six series with multiple episode orders for each."

CAPCOM PURCHASES SECOND UNREAL ENGINE 3 LICENSE

Capcom Co., Ltd. has entered into its second agreement with Epic Games to license Unreal Engine 3 for an unannounced project.

"Unreal Engine 3 is a perfect fit for this project," said Keiji Inafune, Managing Corporate Officer, R&D Management Group, Capcom. "Not only does the development team have thorough knowledge of Unreal Engine 3, the general versatility of Unreal Engine 3 will fully meet the requirements particular to this project."

"One of the major advantages is the well-established support system," he said. "We are delighted to enter into this license agreement, and have strong backup by Epic Games. With Unreal Engine 3, we can expect high development efficiency as well as high creativity within the development team."

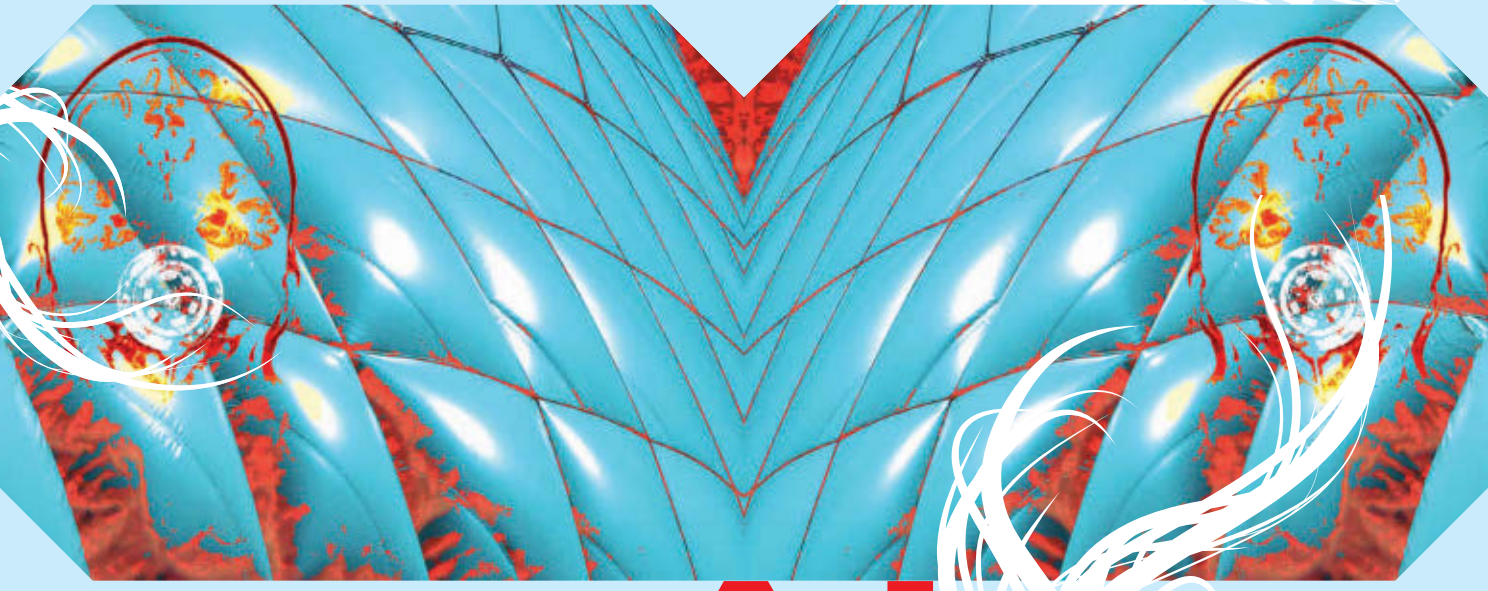
EPIC GAMES

For UE3 licensing inquiries email:
licensing@epicgames.com

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

WWW.EPICGAMES.COM

jeffrey fleming



GAME BRAINS

A SURVEY OF AI MIDDLEWARE

MIDDLEWARE HAS PROVEN ITSELF TO BE AN INDISPENSABLE PART OF MODERN game development. Off-the-shelf engine, animation, and audio solutions have enjoyed success for many years, and physics is an area that is becoming increasingly important. But what about artificial intelligence, which is so intrinsic to the game experience? Here it can be argued that middleware has had less of an impact. Perhaps it is because designing interesting AI behavior is situated so deeply within the process of game creation itself that developers have been comfortable devising their own computer opponents rather than relying on "prepackaged" solutions. However, ever-accelerating hardware capabilities have lead to a demand for vast and dynamic game worlds populated by naturally behaving entities.

While this can be a strain on developers, both in engineering and content creation, it has driven innovative solutions from AI middleware providers. The venerable A* heuristic pathfinding algorithm is being supplemented or supplanted by novel approaches that use the power of multi-core chips to deliver intelligent NPC movement for the new generation of densely populated games. Also, the development of learning AI that can be implemented at low CPU cost and operates in real-time opens up intriguing new game play possibilities.

Here we take a look at eight of the leading AI middleware companies (listed in alphabetical order) and give an overview of their products. It is healthy mix of mature, integrated solutions along with promising new technologies. While academic research and the training simulation market have contributed much to the progress of artificial intelligence we believe that many of the consumer-facing developments will come from the video game industry.

JEFFREY FLEMING *is production editor at* Game Developer *and a member of the Sharon Apple fan club. Email him at* **jfleming@gdmag.com**.

GAME BRAIN'S

AiLive

LiveCombat

Features: Real-time behavior capture

Platforms: PlayStation 2, PlayStation 3, Xbox 360, Wii

Integration With Other Technologies: None

Cost: Available on request

Released Games Include: TBA

Games in Development Include: TBA

www.ailive.net

DERIVED FROM THE SAME Context Learning engine that drives LiveMove Pro's motion recognition, LiveCombat is a game-dependent implementation focused on providing trainable AIs for hand-to-hand or squad-level combat in real time. It allows users to create Context Learning AI "brains" that can be assigned to game characters. The AI brains are then taught by example in real time by keying them to player actions. Old behaviors are gradually over-written as the AI brain acquires new layers of instruction. AiLive suggests a variety of game play possibilities for LiveCombat, including AI tournaments, celebrity created AIs, and specially trained AIs that could be traded or sold from player to player.

Aiseek

Intia SDK

Features: Real-time graph generation for dynamic worlds, dynamic pathfinding with post processing, support for multiple world representations, agent layer, designed for parallel processors and asynchronous operation

Platforms: PC (currently supports multi-core CPUs with support for leading GPUs due in late 2008), contact for Xbox360 and PlayStation 3 support

Integration With Other Technologies: PhysX, HeroEngine, Torque, Gamebryo integration under development

Cost: Available on request

Released Games Include: None

Games in Development Include: MERCENARIES ALL: TRENCH WARS (Playware Studios)

www.aiseek.com

MOST SOFTWARE AI solutions employ heuristic, or "best guess" algorithms to get characters from point A to point B. Because these algorithms have to be fast they carry a certain amount of uncertainty, sometimes generating undesired behaviors that ruin the illusion of intelligence.

To solve the problem Alseek took a unique hardware-based approach to AI pathfinding that dispensed with heuristics. Instead, low-level AI routines such as pathfinding, line-of-site, and terrain analysis can be off-loaded to Alseek's Intia processor, which handles the complex processing without resorting to heuristic short cuts. All these routines rely on search-intensive, branched graphs to arrive at a decision, and Alseek's chip aims to take the brunt of that load. The company is looking to broaden its market, and has now produced the Intia SDK for AI acceleration, which supports multi-core CPUs and GPUs, as well as Alseek's own processor.

KYNOGON'S KYNAPSE TECHNOLOGY TAKES ADVANTAGE OF DATA STREAMING ON multiprocessor and multithreaded hardware to implement large-scale AI that supports agent perception, thinking logic/high-level actions, and dynamic 3D pathfinding. Kynapse uses a hierarchal concept for handling topological data that is similar to the idea of level of detail. Initially the path data for a game's terrain is described in concrete detail. This data is then split into increasingly abstract hierarchal sub graphs that can be rapidly streamed at runtime allowing the AI to make pathfinding decisions on a moment-by-moment basis rather than get bogged down in computing all possible routes to a destination.

Kynogon's recent acquisition by Autodesk gives it the room to continue refining Kynapse and potentially integrate its runtime technology with Autodesk's content creation tools—although no definite plans have been announced.

Kynapse 5

Features: Dynamic 3D pathfinding, runtime topology analysis, open architecture with reusable code, automatic generation of navigation and perception data, automatic hierarchal level generation, information sharing between agents, multi-thread and multi-core architecture with an asynchronous framework, distributed path generation

Platforms: PlayStation 2, PlayStation 3, PSP, Xbox 360, Wii, PC

Integration With Other Technologies: Unreal Engine 3, Vision Engine, Virtools Dev

Cost: Available on request

Released Games Include: Alone in the Dark 5 (Eden Studios), Crackdown (Realtime Worlds), Lord of the Rings: Shadows of Angmar (Turbine), Medal of Honor: Airborne (Electronic Arts)

Games in Development Include: Fable 2 (Lionhead Studios), Destroy All Humans! Path of the Furon (Sandblast Games), TBA (Sony Online Entertainment)

www.kynogon.com

PATHENGINE HAS CARVED OUT A SUCCESSFUL NICHE BY FOCUSING ITS middleware exclusively on solving pathfinding problems. While game developers will still need to create their own higher-level AI, they can leave the complexities of agent movement to PathEngine. The middleware's continuous space and polygonal boundary model uses points of visibility to navigate, allowing designers the freedom to create large, realistic landscapes without the need for tile-based scenery. PathEngine can handle characters of varying size as well as self-overlapping geometries such as bridges, tunnels, and multi-story buildings.

The middleware also features a collision model that can direct movement through dynamically changing obstacles and is designed to provide collision context to the game AI, resulting in more interesting behaviors from agents. According to the company, MMO developers have used PathEngine to direct thousands of simultaneous characters across complex online environments.

PathEngine 5.16

Features: Continuous space, "ground-meshes plus detail obstacles" pathfinding model, automatic expansion by agent shape, integrated dynamic obstacle management, pathfinding and collision against a single world model, content-side automation with automatic ground mesh generation from arbitrary 3D content, support for regions with cost to traverse, and off-mesh connections

Platforms: PC, Xbox 360, PlayStation 3, Linux & FreeBSD

Integration With Other Technologies: 3DS Max, Maya

Cost: Binaries only: 4,500 EUR, Full source code: 13,000 EUR

Released Games Include: Mobile Suit Gundam: Operation Troy (Dimps), Florensia (NetTimeSoft), Titan Quest (Ironlore Entertainment), Pirates of the Burning Sea (Flying Lab Software), Sword of the New World: Granado Espada (IMC Games), Wildlife Park 2 (B-Alive), Alpha Prime (Black Element), Perry Rhodan (3d-io), MayPan (Enium)

Games in Development Include: Just Cause 2 (Avalanche Studios), Guild Wars 2 (ArenaNet), Ragnesis Online (Intelligent Soft), Metro 2033: The Last Refuge (4A Games), TBA (The Creative Assembly), TBA (Rare), TBA (Lionhead)

www.pathengine.com/index.php

GAME BRAIN'S

PRESAGIS

AI.implant 5.3

Features: Lua scripting, geometry tools, stats monitor, voxelization, open and extensible API, source code availability, multi-processor and multi-threaded
Platforms: Xbox 360, PlayStation 3, Windows, Linux
Integration With Other Technologies: Unreal Engine 3, 3DS Max, Maya
Cost: Available on request
Released Games Include: STRANGLEHOLD (Midway), THE BOURNE CONSPIRACY (Highmoon Studios)
Games in Development Include: TBA (BioWare), TBA (Kuju), AMERICAN MCGEE'S GRIMM (Spicy Horse), WHEELMAN (Midway)

www.ai-implant.com

ALTHOUGH MUCH OF PRESAGIS' EMPHASIS IS ON PROVIDING SIMULATION software for the aerospace and defense industries, its AI.implant pathfinding middleware has much to offer game developers as well. Taking a dynamic area-based approach to pathfinding, AI.implant uses terrain data to create a navigation mesh from the polygon environment. The software also allows developers to create non-scripted, rule-based behaviors for their agent movements. The latest version of AI.implant introduced Lua as its scripting language as a replacement for it's own AI.script.

Engenuity originally created AI.implant and the company was subsequently acquired by CAE and re-branded as part of the Presagis group. Partially as a result of these shifts, the software has gone through some changes. Last year it was announced that the middleware's source code would be available for game development at no cost and it appeared that Presagis was withdrawing from the game market. However, Presagis recently revised its plans for AI.implant and has now resumed sale of the software to the game industry.

SPIROPS artificial intelligence

SpirOps AI

Features: Open architecture, graphical development environment, non-interpreted AI compiled in native language, remote debugger, multi-thread compatible

Platforms: PlayStation 2, PlayStation 3, Xbox 360, Wii, Nintendo DS, PC, Linux, mobile

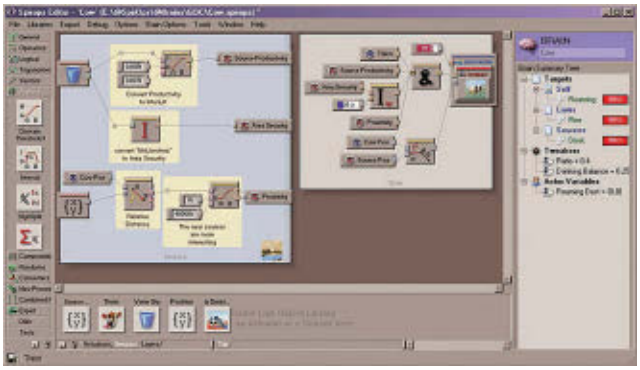
Integration With Other Technologies: Maya, Unreal Engine, CryEngine, Trinigy, Virtools

Cost: Licensed by game title per platform. Approximately 75,000 EUR for SpirOps AI, Path, and Crowd combined, including two years of support.

Released Games Include: TOM CLANCY'S SPLINTER CELL: DOUBLE AGENT (Ubisoft)

Games in Development Include: TBA (Ubisoft)

www.spirops.com



SPIROPS PRODUCES THREE INTERRELATED AI PRODUCTS, SPIROPS AI, SPIROPS Path Generator, and SpirOps Crowd. SpirOps AI is a graphical SDK for producing AI behaviors that are cross-platform and cross-user, allowing both engineers and designers to generate C++ code for integration in a game. The AI brains created in SpirOps are drive oriented, enabling them to follow multiple objectives and exhibit complex behaviors. SpirOps Path Generator is a topographical analysis tool for creating pathfinding for large crowds over wide environments and SpirOps Crowd offers libraries of behaviors for crowds ranging from a few people to thousands.

GAME BRAIN'S

Artificial Contender

Features: instance-based active learning

Platforms: PC, PlayStation 2, PlayStation 3, Xbox 360, Wii

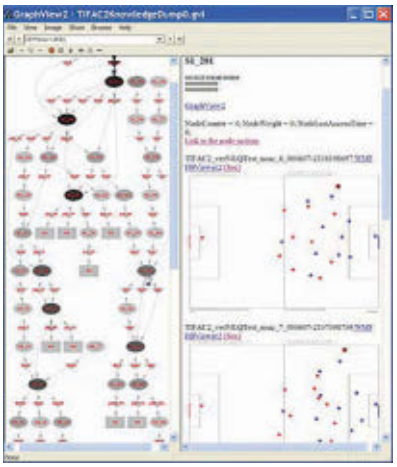
Integration With Other Technologies: None

Cost: Available on request

Released Games Include: This Is Football 2005 (SCEE London)

Games in Development Include: TBA

www.trusoft.com



TRUSOFT'S ARTIFICIAL Contender technology enables game developers to create AI agents that employ behavior capture to learn and adapt in real time. The Artificial Contender AI agents make their decisions based on reasoning by example and can be trained automatically by playing them. This enables developers to generate trained agents without the need for coding and learned play styles can be fine-tuned through self-learning. The technology can also be used to give end-users the ability to train game characters opening up new game play possibilities. The ability to train Artificial Contender agents to play in specific styles could make it a useful tool to assist in game testing as well.

xaitEngine

Features: Supports Windows (.dll) and Linux (shared library), C++ interface to the library with callbacks for memory and file management, multithreading support

Platforms: PlayStation 3, Xbox 360, Wii, PC

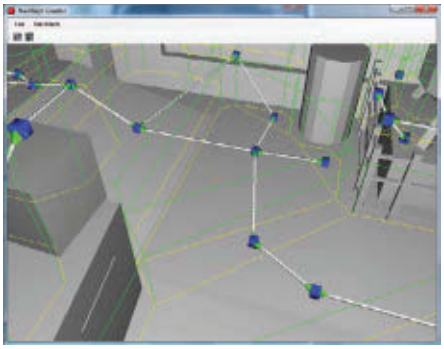
Integration With Other Technologies: OpenMP multithreading API, TBA

Cost: Available on request. Free version for demo development

Released Games Include: Alarm für Cobra 11—Crash Time (Synetic)

Games in Development Include: TBA (Factor 5), TBA (Coreplay)

www.xaitment.com



XAITMENT PROVIDES FIVE AI MODULES CALLED XAITMOVE2, XAITMAP, xaitControl, xaitThink, and xaitKnow that can be used separately to address specific AI needs or work together as the unified xaitEngine. The xaitMove2 module provides movement behaviors such as move, flee, or seek, as well as custom behaviors for AI agents. It supports single unit and group movement with dynamic obstacle avoidance. XaitMap is a navigation mesh tool that can automatically generate navigation meshes from arbitrary geometry. XaitMap also includes xaitMap Creator for importing geometry and editing. XaitControl is a library for building hierarchal and probabilistic finite state machine AI and its graphical editor allows designers to create AI behavior without coding. If rule modeling is needed, xaitThink's graphic interface can be used to design behaviors in an abstracted, rule-based language that is capable of inference reasoning. The xaitKnow module allows designers to define their game world's object concepts using a graphic interface that creates description logic for use by high-level AI. The results can also be exported as an xml file for use in other projects. The company says that it is currently at work on xaitTraffic and xaitRace modules for vehicle games.

Need A Performance Boost?

Shift productivity into high gear with The Archive™, the world's largest single source of high-quality 3D content. The Archive contains 16,504 production-proven 3D models, including over a thousand fully digitized vehicles, landmarks, accurate military models, animals, human anatomy, 3D geography and more. Save modeling time, reduce project costs, and leave your competition in the dust.

Ideal for any game developer's toolbox.

Each model has been carefully crafted in industry-standard OBJ format for compatibility with virtually any production pipeline. The Archive comes wrapped in a game developer friendly site license, available for only **$4,995**.

Find free sample models and a demo at:

www.digimation.com

Digimation THE ARCHIVE

HOW DOES ONE PUT 55 HOURS PACKED WITH GAMING, DEVELOPERS, CONCERTS, SPEAKERS, WORKSHOPS, DEMOSCENERS AND STUNNING VISUAL ART INTO AN AD ON A PIECE OF PAPER?

ONE DOESN'T.

NVISION 08

THE GREAT BIG VISUAL COMPUTING SHOW

AUGUST 25-27, SAN JOSE

REGISTER NOW

NVISION2008.COM

adam hunter

OPTIMIZING WEAPONRY PROJECTILES IN FIRST-PERSON SHOOTERS

IN THEORY, WEAPONS FIRE IS A FAIRLY TRIVIAL OPERATION to simulate convincingly. Click, shoot, fly straight, impact—instant gratification. In fact that may contribute to its popularity in games—it is immediate, intuitive, and straightforward to implement.

There are subtleties that lie in wait, though, and this article focuses on one of those: where, exactly, does the bullet come from? The end of the gun? What if there is no gun, or the gun is invisible/assumed? The eyes of the character? Some invisible point behind the mouse cursor? And where does the bullet go? Straight to the cursor? Straight ahead of the player? We'll talk about some of these options, and the dangers that hide within.

We will make some simplifying assumptions in order to focus on this user interface problem; the conclusions are no less valid in the general case. We will assume for now, as many games do, that bullets fly straight and have an infinite (or at least arbitrarily long) flight distance.

COLLINEAR MODEL

Given this assumption, we can formulate a first model of weapons fire, the collinear model (see Figure 1). In this model, which is both simple and effective, the bullet comes from an invisible point behind the cursor and travels directly towards the cursor. The main issue is that we can't draw an immersive weapon model (or, if we do, it's not where the bullet is coming from, which can be pretty confusing).

It's worth considering the real-world analogue of this model, which is a perfect sharpshooter using long-range weaponry with mirrored scopes to provide a view exactly along the bullet's trajectory. It's somewhat unrealistic and thus invites disbelief when used arbitrarily, but is useful in zoom mode or other careful targeting modes. Of all the models we'll discuss, it is the easiest to implement and the best at allowing users to express and affect their intents.

Some older games—most famously DOOM—attempted to dodge the issue by allowing the player only to choose rotation in the x-y plane. This method might be used when the game really is essentially 2D (CASTLE WOLFENSTEIN), or it could be that the vertical aim is determined automatically as in DOOM. This kind of firing system is no longer used, and for good reasons, due to the advent of true 3D gameplay. The results may not be what the user intends, and the projectile does not impact on the crosshair. Since in modern games almost any object can be the "target" of a bullet, determining the user's intent would be difficult. However, this core idea lives on in some auto-targeting modes such as the one seen in STAR WARS: REPUBLIC COMMANDO.

FIRST-PERSON WEAPON MODEL

The most common firing mode in use today is the standard first-person shooter mode with an immersive weapon model. In this model, the camera is embedded in the avatar's personal space (the "eyes"), and some amount of the weapon and its

ADAM HUNTER

has been a professional programmer for almost a decade, and a graphics engineer for most of that time. He's currently at Page 44 Studios. His website at www.adamhunter.net is perpetually 10 man-hours from being legible or useful. Email him at **ahunter@gdmag.com**.

Photograph by Dave Dyet

READY, AIM, FIRE!

FIGURE 1

projectile exit point or points is visible. The projectile exit points are not along the path traced by the cursor, as seen in Figure 2.

PARALLEL APPROACH

This presents a challenge. Players may (quite rightly) expect that the cursor represents exactly where they intend to shoot. This is a problem though, because in most games, the motion of the mouse can only express a single rotation relative to a fixed camera, located at the eyes (not the weapon). In the most naive implementation, such as the standard "instant fire" of the Unreal Engine, this rotation is inherited by the weapon, as shown in Figure 3.

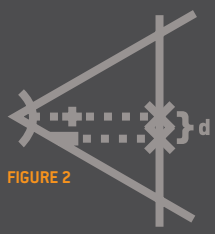


FIGURE 2

We'll call this the parallel approach. This model has some clear advantages. It is easy and intuitive to implement. It only requires one ray trace through the world to determine the path of the bullet, and that trace can be asynchronous, as in an MMO or other server-verified game.

It also may correspond to what a novice would do with a weapon—shoot from the hip. However, it results in a constant deviation of the bullet path (refer to *d* in Figure 2) from the crosshair. That deviation will be predictable, not random. It will also be more obvious the closer the target becomes. Still, this model has been used in many games (such as UNREAL itself) and will probably continue to be used.

One way to help hide the constant deviation, if it's appropriate for the weapons in the game, is to have a "spread" or randomness integrated into the weapon fire.

FIGURE 3

Limiting the player to x-y rotation made the DOOM firing model simple and intuitive.

PREDICTED INTENT APPROACH

A common improvement to this model is to predict the player's intent and attempt to produce the desired result, what we'll call a predicted intent model. In the aforementioned diagrams, *x* indicates where the player's projectile intersects the world. If we pre-compute position *x*, we can use the position of the gun to compute the adjusted rotation the gun would require to hit that target, as shown in Figure 4.

The major benefit to this approach is that, in most common cases, the projectile will now impact the scene roughly at the crosshair, and the user will be happy and none the wiser.

But there are some issues to contend with. First, this model requires computing two ray casts through the world. Second, it has no frame-to-frame coherence; a one-pixel deviation can result in a large change in the angle of the gun as the crosshair moves from target to target. This can make it hard for players to track a target by incrementally correcting their aim, which is a common technique players use on small objects or moving objects.

In online games there will be client-server issues as well. The first ray cast must be synchronous because we must have the result in order to fire the weapon. For client-server games, that requires verifying all firing (to catch cheaters). We can compute the first cast on the client and send up the desired firing rotation.

Unfortunately, that solution poses a security leak. Many games require that the server generate the firing vector to ensure that the client isn't cheating and shooting in a direction they really wouldn't be able to shoot. When the firing vector is not just the player's rotation, it can make doing this cheat detection more difficult.

A common way to get around this is to check the angle between the firing vector the client suggests and the player's true rotation—but this is then subject to lag. It can be difficult to eliminate all the little edge cases. Also, it assumes that the first cast through the eye vector actually hits something. If it does not, we can arbitrarily stop it at an invisible shell around the level, or we could just fail out and return to the parallel model.

COINCIDENT OBJECT PROBLEM

All these problems are real, but a more serious problem is depicted in Figure 5.

Let's call this the coincident object problem. Because the eye ray and the weapon ray are not collinear, they may not impact the same target. In systems with instant projectiles or at least the ability to do synchronous client-side ray casts, this problem is relatively easy to detect, but what do we do about it?

One valid answer is nothing, and many successful games (and I think one could argue QUAKE 2 was a successful game) have stopped there. When precision is required, typically to hit a distant target, many games have a zoom mode that is collinear and thus eliminates this problem. But what if we want to do better?

The problem is that we feel compelled to obey two constraints simultaneously. First, we want to have the final impact position render on top of the crosshair, and second to realistically simulate the path of the projectile out of the gun. We will have to relax one or the other. For games that do not draw the path of the bullet, the choice is clear: simulate the bullet coming out of the gun. If you do draw the path of the bullet, but only loosely, you may be able to get away with simulating the real bullet from the eye, while drawing a tracer from the gun.

This tracer may interpenetrate world geometry, but the flaw might not be noticed. Muzzle flashes can help cover this up. For sci-fi weapons like lasers, though, or any weapons that leave a persistent trail, this will look bad up-close.

A second option is to allow the issue to occur, but let the player know it's about to happen by

Muzzle flashes can mask potential targeting mismatch for the player.

showing players not only the crosshair, but also a separate indicator of their actual target. This is, after all, akin to a real-world laser sight. A downside of this plan is the requirement to do continual client-side ray casts. (One possible optimization is to make them pretty short, and use the eye vector if nothing is detected.) Another downside is that, while players may now know what's going on, they still may not like it, and if you don't have a zoom mode, they won't have much choice.

Another approach is to solve the problem through animation. When the problem is detected, it is often the case that a close-up obstacle between shoulder and waist height is blocking the view. What would a real person do? He would raise the weapon over the obstacle.

This solution requires recognizing the event, producing custom animations, potentially modeling more of the first-person model than you would otherwise have intended, and tracking the end of the weapon barrel. The effect, however, would be potentially outstanding.

A key aspect is moving the projectile exit as close as possible to be collinear with the eye vector, or at least ensuring that the coincident obstacle is avoided, while making the motion feel natural. Inklings of this approach have already started to pop up in FPS games such as CRYSIS, in which the AI will adopt a rest pose with the gun elevated to the shoulder when near an obstacle.

A final idea is to implement a "situational zoom." Assuming your game has a zoom mode, when a close-up object is detected in front of the player, activate the zoom mode automatically. Using an animation blend in this situation is probably advisable. The solution is similar to that of a third-person game in which backing up against a rear wall causes the camera to force itself inside the player model, except in this case the gun is being "pushed inside" the player model so that firing can be more accurate.

WHAT'S LEFT

We've obviously only discussed the tip of the iceberg. For example, we don't have space to adequately cover third-person or over the shoulder camera modes, in which the problem may be just as bad, or worse, depending on how far over the shoulder the camera is.

Some of the techniques outlined in this article are still appropriate. Placing the camera lower also helps, but it may cause the model itself to obscure the target. One ameliorating factor may be that in third-person mode, the expectation of accuracy may decrease. Often, third-person mode is used with vehicles or other area-effect weapons where pinpoint accuracy may not be required. If pinpoint accuracy is required, and relocating the camera is not helpful, consider implementing a first-person zoom mode.

FIGURE 4

CONCLUSION

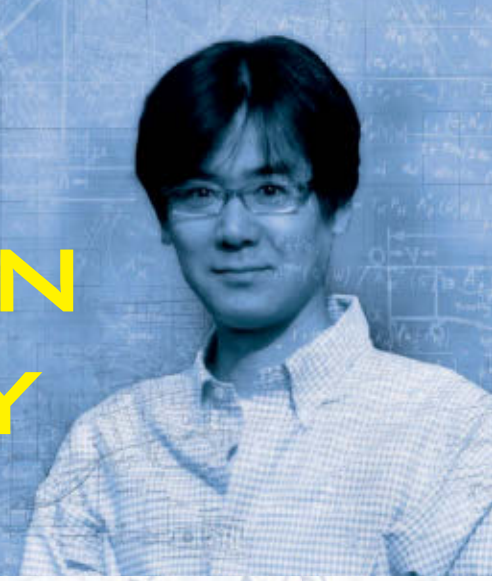
What's the ideal scenario then? The right answer depends, of course, on your game. Do you have a first-person model? Do you use an over-the-shoulder camera? If you don't have a first-person model—either at all or in certain modes—then going with a collinear approach is the best choice. If you do, it's time to consider how much effort you want to put in. The parallel approach is nice and simple. Putting spread on the weapons fire helps masks the inaccuracy. So does keeping the weapon's exit point close to the mouse cursor, but then it can obscure the action. With a little more effort, a hybrid/predicted intent model works well, especially if the client/server issues aren't such a big deal. If your game already has some sort of auto-targeting functionality it can be an obvious fit. But the best option may be the dynamic model used to some extent in CRYSIS; move the gun like a real player would—keep it at the ready, and raise it either when firing, or when near an obstruction. It's more work, but more realistic.

All this may seem like a lot to go through, but remember that for most shooting games, the core interactivity between the players and their weapons is central to their positive experience. Optimizing the firing experience can make a good game into a great one.

FIGURE 5

GAME DESIGN PSYCHOLOGY

An interview with Hirokazu Yasuhara



HIROKAZU YASUHARA IS ONE OF THE GREAT UNSUNG HEROES of game design. He is currently senior design director at Namco Bandai Games America, and before that he held the unassuming title, "game designer," at Naughty Dog, having most recently shipped UNCHARTED: DRAKE'S FORTUNE—but his history is inexorably intertwined with the history of modern character game development. Yasuhara was the level designer on the original SONIC THE HEDGEHOG, as the third person to join that team after Yuji Naka and Naoto Ohshima. He was responsible for the first 3D SONIC game, SONIC R, and was involved in the JAK series for Naughty Dog since the first sequel.

In this extensive interview, Yasuhara outlines his carefully constructed theories of fun and game design, including the differences between American and Japanese audiences, with illustrated documents. After conducting this interview, I was convinced that he should write a book based on his theories. Until then, consider these words to be sketches—a preamble to that necessary work.

Brandon Sheffield: *I have heard that you still use graph paper for all your level designs. What is your process for designing at this point?*

Hirokazu Yasuhara: Actually, I stopped using graph paper to make the level. [pointing out some paper materials] I use this to work out all the gimmicks [ie. the unique features to each level], but I come up with small, easy ideas about events that are happening; how the player acts at each stage. What kind of results happen once you perform this or that gimmick in each level. [shows some examples on paper]

BS: *So these are small bits of design concept, like moments that you could use?*

HY: Yes. So I come up with some ideas for the programmer to work with, and they decide what's good and what's impossible to implement, based on schedule or programming difficulty.

BS: *It's kind of high-concept design, and then they narrow it down more?*

HY: Yeah. So this is the idea for a section, and I make a picture or a scan of what kind of image I have going, add some simple comments, and make a document that I bring to the artists and programmers. These are all concepts. I make a lot of ideas and inserts. And this is what I just created. I don't write the maps by hand anymore; I use Illustrator instead to do the map. It has about five layers.

BS: *In your process, after you think of ideas and then put down whatever comes to mind, do you shrink down the scope of the actual design, or is it other designers?*

HY: I shrink it down by myself, actually. It's up to the schedule, so ... If the artists or programmers say "no," then that's the answer. So it's kind of a mix. I always try to push a can-do attitude with them, you know? [laughs] For the programmer. But sometimes, you know ...

BS: *How do your ideas come from these individual moments into the full art of game design?*

HY: OK, let's start with my ... it's really hard to explain. Yesterday I tried to explain, but ... this is it. [see Figure 1] So I always think about all the different elements of what makes something fun. This formula is made by a sociologist from France who did some thinking into what it is that makes something fun, or interesting, for people to experience. One of the things is competition. The next is happy coincidences; a gamble that pays off, that kind of thing. Following that is dizziness or exhilaration, and the final thing is imitating, or copying.

For example, let's say we go to a theme park one day. There are two slides there: a regular metal slide, and one shaped like an elephant. Which one is more attractive to a child? [see Figure 2] It'll usually be the one with the elephant, because the form of "imitation" that it represents is more interesting to the eye. That, in itself, is enough to make it fun. So what happens when you put all of these factors together? Well, if your park's trying to improve its business, then maybe it'd try to make the slide a longer or faster ride, or maybe make it bigger and shaped like a dinosaur so it'll be more fun for the kids. Maybe they'll make it a dual slide so kids can compete with each other to get to the bottom faster—add a competitive element. If they keep going with it, it'll get big enough that it winds up becoming a log-flume ride or something—but there's still more you can do, like maybe put wheels on the logs and make it look like a car. It's a continual process to make it more fun. So the more you think about the externals of something, the more grandiose it'll wind up being. You'll wind up with a roller coaster eventually—and then you'll make it rotate or something, if you think it'll improve business. That is one of my basic principles.

Another important thing is to consider the basic desires of people, even if all you're thinking about is a simple game. For example, you have active desires—"Freedom From Fear," as they say, the way people actively want to avoid fear in their lives. And one way they deal with that is by engaging in a sorting process. Let's say that you have a flat surface with some bumps

BRANDON SHEFFIELD *is editor-in-chief of* Game Developer *magazine. He played the 3D game in* SONIC JAM *through to completion. Email him at* **bsheffield@gdmag.com**.

FIGURE 1 Yasuhara's theory of fun is shown in Japanese.

sticking up out of it. Most people would want to see those bumps removed, as a sort of equalizing or "beautification" process. Also—you know the game Othello, right? A lot of the fun in that game is the exhilaration you get when you flip a lot of pieces and make more of the board your color. Tidying up things, in a way. It's the same thing even in business—it's nicer when you have a well-organized Excel spreadsheet than a cluttered one. It's a continual process of actively sorting and bringing things under control, and the reason why people do this is because it helps make life simpler for them—the process itself is fun, too.

As for how this goes back into video games, one thing you see a lot of in games is the act of "erasing," or "destruction." For example, in PAC-MAN, you're eating dots—wocka-wocka-wocka-wocka. That is erasing, and it's also a form of destruction. You're destroying everything in your path, and you're leveling out the entire playfield. This is something that I think is vital for any interactive experience—that sort of proactive desire in motion. This manifests itself in a lot of ways; the player can satisfy this desire a lot of ways in a lot of different games. But there's something else involved here: creation. Some people get what they want via destruction, but others do it via creation instead. For example, if I am feeling vulnerable, then I get more friends or party members, if you will, and make myself more protected—or I go to town and interact with people to get that same feeling. By the same token, some people think in the opposite way—if I kill every enemy in the area, then that logically means I'll be more secure. "Fear" at play. It's different ways of arriving at the same emotion.

BS: *Maybe like PIKMIN versus GEARS OF WAR.*

HY: Yes, exactly. And this process keeps repeating itself. You see some culture differences come to the surface with this, too. For example, a lot of Japanese people attain a feeling of security via creation, or making themselves look nice, or saving money. Not that Americans or Europeans aren't like that, but Americans may be more likely to take a more "destructive" process toward feeling safe. I think a lot of that is because the things that you "fear" can be very different between nations—not real, palpable fear, but more the lack of feeling "at ease" with yourself. Something you don't like very much; something that stresses you out—another word for "stress," really. And since sources of stress can be different between Americans and Japanese, it follows that the methods both populations take to relax would be different, too.

One more important thing I want to bring up is that when people achieve freedom from fear, that in itself makes them feel happy. You aren't stressed out anymore, and that cheers you up. I use that a lot in my game design, because it's very basic and important. Another thing is that, putting it simply here, I usually come up with three goals when I'm making a level: a short-distance, middle-distance, and long-distance goal. (see Figures 3 and 4) For example, going to see Cinderella's Castle in Disneyland would, for us, be a long-distance goal; if it was a game, we'd need to keep reminding the player where he's going if we actually want him to remember it. A more short-distance goal, meanwhile, would be if you're in a baseball game; your goal is to get on base, and there are any number of simple, linear ways to achieve that goal. An example of a middle-distance goal would be if you run into a bridge in the forest that you can't gain access to—something I do a lot in games. Maybe you have to do a sequence of jumps to reach it, but it's visible, at least. It's a

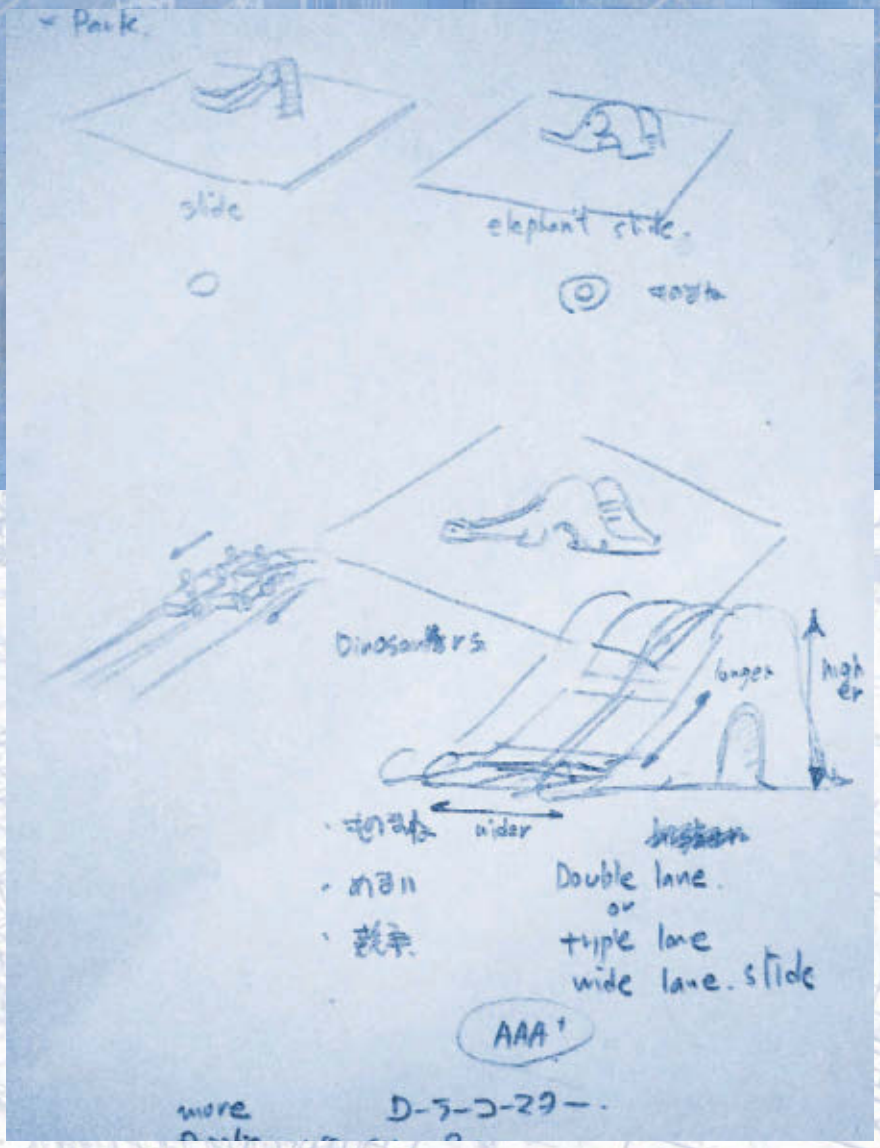


FIGURE 2 Yasuhara uses the example of a children's slide to illustrate the evolution of fun.

FIGURE 3 Short-, middle-, and long-distance goals are shown.

FIGURE 4 Details of the range-based goals are shown with examples.

constant cycle of "fear" and "relief". If you're in an enclosed area, then completing a middle-distance goal to escape it makes you relieved; it makes you think "Oh, that's how I get out of there!" I'm always thinking about that kind of thing.

BS: *Short, middle, and long distance—how do you manage those goals? What is the critical difference between short and long distance?*

HY: It depends on the player's moving speed. For normal gamers, these short-distance goals would be around 30 seconds each.

BS: *In terms of the kind of goal ... how do you judge when the player should feel he's accomplished a small goal, and then a larger one? How do you gate those—what kind of feedback should the player receive in order to know "I've accomplished a bigger goal now?"*

HY: The important thing here is that the player always feels like he's in control of his own fate—that he's got a full understanding of the world around him and what's going on. That has to be a constant process.

BS: *And how can you do that in an open-world environment? In SONIC games, it was 2D and you can see everything around you, but in an open world, there's so much more.*

HY: Well, for example, if there's a house in an otherwise completely empty area, then the player will probably try to go in, since there's nothing else to check out. If you're in a shooter and the enemy is shooting at you, then you know that you have to avoid the bullets and come up with countermeasures. No matter what the environment, if something special is nearby—whether it's hostile or not—it will grab the player's interest. There are lots of ways this can manifest itself.

BS: *What about an open world like GRAND THEFT AUTO where the player can go anywhere at any time, and just do anything? Many American games are moving toward that, and so you must also have to be thinking that way somehow. In that kind of scenario, what do you think is important—do you think it's important to keep a player on the designer's goal, or can they be doing whatever goal they may set for themselves?*

HY: That really depends on the game. Even in GTA, you're still always reminded of the really important things that you should do. If you're lazing out on a mission, you'll get a call asking you what's up. That's the way the game motivates you to continue. Really, "freedom" is not what you get in a game. In SECOND LIFE, they say you can do anything you want, but really, there's nothing to do there! That's not a game. In a game, the designer is a "game master," and he has to be thinking about you.

BS: *Do you think it's possible to give the player too many goals, all at once? Like, again in GTA, or in the open parts of UNCHARTED, do you think it's possible to have too many things the player could do, so they get overwhelmed?*

HY: Hmm, that's difficult ... Well, you have individual goals, like in GTA where you're trying to kill X enemies within one minute, but I don't think that is "the" goal—the real goal of the game. There's a difference between making a game and making a virtual world and putting it in a package. It's the job of the game master to take that world and give you the motivation to move through it. If you don't, then that won't leave the player satisfied. If you're just trying to keep the player playing as long as possible, then that's like an online game, where the focus is much more on communication—"Hey, how are you, let's go kill that enemy," you know. That communication aspect is part of the game.

BS: *From my perspective, sometimes when I start up a game, they say "you can do this, and this, and you can customize your character, and then you change his color, his name, then you set up your party ...", and it's just too much. They give me too many options, and then I don't want to play. How do you avoid that? How do you decide what's the most important for the player at a certain time?*

HY: For example, you could make the setup process the same every time and have it so you can start it right away; I like games like that. Then, after that, you could go buy your own equipment to customize if you like, or make your own designs. People who want to do that could, and those who don't aren't forced to. Keep the basic experience simple, and allow players to explore it at their pace.

BS: *Yeah, sometimes the problem with optional things like that is that if the hardcore player will want to try it, but then they'll burn themselves out because it's too much.*

HY: I definitely understand that. When you begin the game, you're on a high; it's like "Aaahh, what am I going to do!"

BS: *And this seems to happen a lot in MMOs, for example. You start at level 1 and everything else is at level 60 or 70.*

HY: Yeah, and then you just say "Forget it" at the start. (laughs)

Planting Seeds for the Next Gaming Breakthrough

SIMULATING REAL-WORLD PHYSICS

Physics—that fascinating branch of science that traditionally puts high school students into a REM sleep state—has also given birth to innovations as modest as the roulette wheel and the unicycle, and as awe inspiring as the hydrogen bomb and the Apollo space missions. Depicting real-world events realistically in 3-D computer games requires a physics engine that is capable of applying the rules of Newtonian physics to the interactions of characters and objects. This objective guided the successful rise of Havok, from its origins in Dublin, Ireland, in 1998 to its current position as the preeminent provider of physics software for triple-A computer games. Havok continues to gain stature and recognition in the technology industry for its expertise in producing physics, animation, and software-development tools for game developers and the movie industry.

The Havok Physics* engine simulates the nature of motion, particle systems, and collisions in the environment and provides them algorithmically for use in computer gaming. Combined with Havok Animation*, which offers an extensive slate of animation capabilities to game developers, the functionality makes it possible to recreate scores of physical events, from the collision of two billiard balls to an out-of-control motorcycle careening through a traffic-filled intersection. Characters also gain new realism in gameplay through a motion pipeline that helps determine their responses to one another. Packaged together as Havok Complete*, these physics and animation capabilities are now available as a product for downloading and use at no cost.

Intel's acquisition of Havok in September 2007 set the stage for some very high-profile advances in gaming. Long-time partners, Intel and Havok engineering teams worked closely to refine and improve Havok's HydraCore* technology, which optimizes game physics behaviors on platforms based on Intel® Core™ microarchitecture. Intel has pledged that Havok will continue with its cross-platform philosophy and Intel will employ a hands-off management approach, giving Dublin-based Havok the opportunity to take maximum advantage of Intel engineering resources and software tools, while enhancing its popular middleware for the strongly competitive game market.

I talked with David O'Meara, the managing director of Havok, about product announcements at the Game Developers Conference (GDC), the nature of cross-platform game development, rising industry costs, and the benefits of Intel and Havok working together.

Havok Cloth provides cloth simulation technology and tools that dramatically increase the realism of game characters and environments.*

Game Developers Conference: Stunning Response to New Havok Products

I understand you received a very strong response to your new product introductions at the 2008 GDC.

O'Meara: *Havok launched two new products at GDC this year and both of those products have been exceptionally well received. One of them is called Havok Destruction* and the other is called Havok Cloth*. Together, they are going to significantly improve the experience for gamers. The Cloth video demo had 370,000 downloads in 24 hours. That is phenomenal.*

These new technologies, Destruction and Cloth, on top of our existing products, will give games that come out in a year or two another breakthrough in terms of the look and feel and the gameplay. They are also very important for the type of thing that Intel wants to achieve with future hardware for visual computing, with the capability of doing things with Destruction and Cloth that you might not get with smaller platforms.

Is the nature of game development changing?

O'Meara: *In the games industry, the cost of developing these games has risen dramatically: it costs around 30- to 40-million, USD, to develop a game. Five years ago it was around two to five million. Costs have increased substantially for a number of reasons. One is that the consumer expects a compelling story and a compelling video experience, which requires a lot more thought, effort, technology, and movie-like appearances in the game. So, that has obviously increased the cost significantly. Secondly, there is the increase in the cost of developing across these really advanced technical platforms, like PlayStation* 3 and Xbox* 360, compared to the type of platforms we had five years ago. Thirdly, in game studios today there will be over 100 people working on a game.*

At a business level, Havok is providing one piece of software to work across a huge range of hardware that varies greatly in complexity: sophisticated PCs, sophisticated PlayStation 3 consoles with lots of special processing units, and then right down to mobile devices with limited capabilities at the moment.

Is your HydraCore technology adaptable to the full range of game consoles and computers? How easily can you take operations such as multi-threading and work them across a huge range of platforms effectively?

O'Meara: *The heart of why Havok is the most successful provider of software in this space right now is that no one else is as good as we are at supporting multiple platforms. No one has come close to being as good as we are at doing that.*

I think the secret of this gets down to being no great secret in the sense that we are the most battle-tested software company in this space. There are over 200 triple-A video games released using Havok components. There are 100 in production this year due for release that will use Havok. And, it is the experience that we have gained with those publishers on those triple-A games that gives us the insights we need to ensure that our software is as elegant as it can possibly be across a range of platforms.

It is not easy to operate across a range of platforms. We optimize for the most complex piece of hardware out there. Currently, we have teams working, optimizing for hardware that is not yet on the market (including upcoming hardware from Intel).

The insights that we have gained from working on hundreds of games and understanding how the game developer will want to utilize the upcoming platforms give us an edge in providing tools that developers need.

A Freebie for PC Game Development: Havok Complete*

You also announced a free downloadable version of Havok Complete for the PC at the 2008 GDC. What was the reasoning behind making the software freely available?

"Havok is a proven leader in physics technology for gaming and digital content, and will become a key element of Intel's visual computing and graphics efforts."

Renee James,
Intel vice president and general manager, Software and Solutions Group

O'Meara: *We are hoping to see a lot of the universities and a lot of the kids getting very familiar with the technology and we are hoping to see those who have the initiative to then develop things themselves. They will be able to do this without incurring the significant upfront costs of licensing the technology.*

We expect widespread experience of the Havok software and we expect to see breakthroughs coming from some of these people, many of whom may develop small games or develop ideas that they can then show to a publisher and say, "look what I am able to do with this game. Are you interested?"

What is the importance of using the PC platform for development compared to game consoles?

O'Meara: *PCs are often the platform on which critical breakthroughs are made in gaming. A platform like PlayStation 2 can be two or three years old; the PC can have advanced much more than that PlayStation 2 has after those two or three years. The newer PC is capable of providing game experiences that wouldn't have been possible on the PlayStation 2. Typically, the PC is technologically ahead of the capabilities of the consoles. The PC is very important for critical breakthroughs in gaming.*

Secondly, the PC is very important for online gaming. World of Warcraft, Second Life*, and other games where people are using the platform to access an online game are very important uses. Important gaming milestones—from id Software's Doom* to the cinematic storylines of Valve Software's Half-Life 2*—were launched on the PC platform. Because technological advances in computing typically emerge first on the PC, game-development studios often target these high-end platforms for their more ambitious coding efforts and scale down to achieve maximum cross-platform support.*

Havok and Intel Working Together

Whenever one successful company buys another successful company, everyone wonders what the ultimate outcome will be. Is Havok's operational independence important to ongoing business prospects?

THE DRAPE, FLOW, AND MOTION OF GARMENTS IS SIMULATED BY HAVOK CLOTH.

O'Meara: *I think our independence is hugely important to Intel for a couple of reasons. One is that the credibility that Havok has with the gaming industry spills over onto Intel. But if Havok isn't independent, it won't have that credibility because it would be seen as partisan. For Intel to make a much bigger play in gaming itself on the hardware side, it is really helpful to Intel that Havok is independent—because of the credibility Havok has. And Intel is going to gain hugely from the industry if the industry sees that Intel continues—as it has been doing—to allow Havok to be independent.*

How has the relationship with Intel benefited Havok?

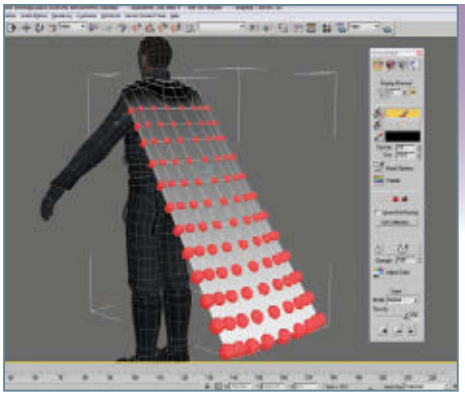
O'Meara: *There have been a number of things that have been very good already in the six months since we were acquired by Intel. For example, the PC initiative underwritten by Intel [offering Havok Complete for free] is really great, because this initiative has the capability of significantly expanding the market beyond just the triple-A developers.*

Broadening our capabilities beyond the entertainment industry is another anticipated development we expect to gain with Intel. In other words, we will go into serious gaming. Serious gaming includes non-entertainment applications, such as industrial and military simulations and training. Intel offers Havok a lot of capabilities and resources to enable us to go beyond the entertainment space in a couple of year's time.

Finally, I think there are technologies within Intel that are developed by Intel (without Intel looking for a commercial application from them). It would be very nice if we could create some commercial applications for some of the technologies that are being developed—take them out of Intel and put them into Havok. And that is part of the concept for Intel and Havok.

"Havok's commitment to being cross-platform and serving all customers continues and it is both wise and visionary of Intel to keep that in place. Working with Intel is a great fit for Havok products, customers, and employees. Intel's scale of technology investment and customer reach should enable Havok to grow more quickly into new market segments with new products than we could have done organically. We believe the winning combination is Havok's philosophy, focus, technology, and customer know-how with Intel's scale. I am excited to be part of this next phase of Havok's growth."

—David O'Meara, Havok managing director



SNAPPY DRESSING GAME CHARACTERS ARE WEARING THE LATEST FASHION STATEMENTS FROM HAVOK CLOTH.

New Technologies, Future Breakthroughs

Gaming platforms will continue to advance and incorporate new technologies, with parallelism, new caching techniques, and other technologies expanding the realism and richness of the gaming experience. As long as there are multiple audiences running games on platforms of varying capabilities, Havok will be providing tools to developers that bring the best in physics and animation to each platform.

Working together, Intel and Havok will certainly play a key role in exploring future options and extending the possibilities for achieving fast, efficient, realistic animation within 3-D worlds. These efforts should also help open new avenues for less-experienced developers to create both 3-D objects and object behaviors based on sophisticated physics that will be the foundation of online environments such as Second Life. The prospects for breakthrough animation advances for gaming and movies are extensive and a worldwide audience awaits the results of the collaborative activities of Intel and Havok. END

ABOUT HAVOK

Havok, an Intel company, was founded in Dublin, Ireland, in 1998 and is the premier provider of interactive software and services for digital media creators in the games and movie industries, with world-leading expertise in physics, animation, and tools. Havok's business is to turn customer's creative aspirations into technical realities. Havok's modular suite of tools gives power to the creator, making sure that clients can reach new standards of realism and interactivity, while mitigating the overall cost and risks associated with creating today's leading video games and movies.

For their technology advances, Havok has earned an impressive collection of industry awards, including the National Academy of Television, Arts & Sciences Award, Game Developer Frontline Award, and a Develop Industry Excellence Award.

Havok has offices in Dublin, San Francisco, San Antonio, Calcutta, Munich, and Tokyo.

David O'Meara, managing director, Havok— David O'Meara is responsible for defining corporate strategic direction and managing the company to achieve its ambitious technical, financial, and commercial goals. David has more than 20 years senior management experience in the software, services, and telecommunications industries. David holds a degree in Economics from Trinity College Dublin.

ABOUT THE AUTHOR

Based in Arlington, VT, Lee Purcell is the executive director of Lightspeed Publishing LLC specializing in digital content creation. He blogs on alternative energy topics at **lightspeedpub.blogspot.com**.

Evolve at SIGGRAPH2008

Image: Dream 191.14533 © 2006 Scott Draves and the Electric Sheep

Sponsored by ACMSIGGRAPH

A Shock Wave of Technology & Innovation

CONFERENCE: MONDAY, 11 AUGUST - FRIDAY, 15 AUGUST 2008 EXHIBITION: TUESDAY, 12 AUGUST - THURSDAY, 14 AUGUST 2008

The highest quality, most timely, multi-sensory educational experiences presented by the most powerful and engaging leaders in computer graphics and interactive techniques. SIGGRAPH brings together learners, legends, and luminaries, all in one pulsating place. This August come to SIGGRAPH and brave the shock wave of technology and innovation.

Register Early and Save!

SIGGRAPH2008 | Evolve

WWW.SIGGRAPH.ORG/S2008

The 35th International Conference and Exhibition on Computer Graphics and Interactive Techniques

LOS ANGELES CONVENTION CENTER LOS ANGELES, CALIFORNIA USA

HOTHEAD GAMES' PENNY ARCADE ADVENTURES

ON THE RAIN-SLICK PRECIPICE OF DARKNESS EPISODE ONE

EVER SINCE WE STARTED WORKING ON Penny Arcade Adventures: On the Rain-Slick Precipice of Darkness, the most common question I get (besides, "Why the ridiculously long name?") is, "What is it really like to work with Mike and Jerry?"

Mike "Gabe" Krahulik and Jerry "Tycho" Holkins of Penny Arcade fame have built an impressive following among video game fans, and we at Hothead were very excited to work with them on their first video game. Once the initial enthusiasm wore off, however, we realized we had some serious challenges to surmount.

For starters, the criticism, sarcasm, and opinionated rants Penny Arcade had dished out over the years made our game a potential target of the same. In addition, we understood that most ardent Penny Arcade fans are pretty particular about their expectations. We knew it was critical to meet those expectations and make the game feel authentically Penny Arcade in every way.

On top of these challenges, we were starting Hothead as a different kind of game company, bringing back classic adventure games, embracing the episodic format, and focusing on selling our games exclusively online. All those endeavors added risk to the project. So, while this article is a postmortem on making the first episode of On the Rain-Slick Precipice of Darkness (mercifully abbreviated to Precipice hereafter), it is also a reflection on our experiences founding a start-up company with a different approach from that of a typical work-for-hire developer.

For this postmortem, it was hard to fit something into one "right" or "wrong" category. The things that went well all seem to have aspects that we would have done differently,

GAME DATA

DEVELOPER
Hothead Games

PUBLISHER
Hothead Games

RELEASE DATE
May 21, 2008

PLATFORMS
Xbox Live Arcade, Windows, Mac, Linux

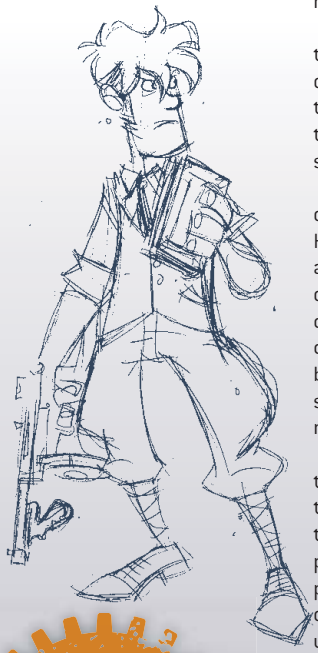
JOEL DEYOUNG *has more than 10 years experience in game development, including work as a lead programmer, technical director, and producer. He is currently COO of Hothead Games and producer for the Penny Arcade Adventures: On the Rain-Slick Precipice of Darkness episodic series. Email him at* jdeyoung@gdmag.com.

and it's hard to regret the difficult parts because we learned so much along the way. I hope these thoughts will shine a light on the process of starting an independent development company and on making a game with two of the most colorful characters in the industry.

WHAT WENT RIGHT

1 **BEING INDEPENDENT.** We started Hothead back in 2006 with an eye to remaining fiercely independent when, with services like Steam and Xbox Live Arcade growing in popularity and success, it became clear to us that digital distribution's time had come.

Focusing our company on selling games online and funding the games ourselves meant that we could remain independent of publishers. This created a genuine indie feel on the team throughout production. Team members felt empowered to make the kind of game we wanted, which gave everyone an extra sense of ownership and motivation to make a great game.



This model was a big change for everyone at Hothead, with virtually all our veteran staff coming from work-for-hire developers. The change in dynamic was most evident because we were no longer shipping off monthly milestones to a publisher.

It is important to note that despite the appeal of this, we realized midway that monthly milestones provide a convenient pressure to keep a project on track. We had to come up with other ways to motivate ourselves to hit interim milestones—a surprisingly difficult task when they are not tied directly to revenue. Despite that, it was a refreshing change to be able to make a game from beginning to end, answering only to ourselves.

Lots of iteration on the combat system made it a fun, polished experience, but the rework was so late in the process that the changes were inefficient to make.

2 **VETERAN TEAM.** We deliberately chose to build our team exclusively with veteran talent. Everyone came into Hothead with between five and 15 years experience making games. Many of us had even worked together on the same team in the past, so getting everyone to gel right from the start was simple.

Hiring experienced staff turned out to be an excellent choice. We were already taking on significant risk by doing a number of new things: making a game in episodic format, relying heavily on outsourcing, and launching our own digital distribution platform, Greenhouse. Having a veteran team meant that we could avoid the additional difficulties caused by beginner mistakes.

An experienced crew is no silver bullet, though. Nothing substitutes for good project management, being thoughtful about putting the right people on the right tasks, and having someone lead the team with a specific game vision. While we experienced our fair share of challenges on the project, taking this approach did help us reach our project goals sooner, while making fewer mistakes along the way.

3 **PICKING A GREAT IP PARTNER.** Making games based on licensed IP is a double-edged sword. Working with a successful IP can translate into significant sales success for a title, but working with the license holder on the creation of the game can be fraught with peril. Difficult approvals, costly re-working of content, and licensor politics are just a few issues that can make you wonder if taking on a license is even worth the hassle.

When we went into the PRECIPICE project, we approached the relationship less like a licensing deal and more like a true collaboration and partnership.

At the beginning, we agreed we would co-develop the game with Penny Arcade. Mike would handle the concept art for the environments and characters to set the tone of the game. Jerry would write everything in the game from the interactive dialogue sequences to descriptions of items in the world. Hothead would handle everything else: creation of the game's assets, design of the gameplay, and general guidance of the process.

The arrangement worked very well, with some notable exceptions that I'll talk about later. Mike and Jerry's direct involvement was essential to making PRECIPICE feel authentically Penny Arcade. For example, Jerry's particular writing style and brand of humor is something his fans will expect from the game's copious dialogue. Plus, they were also just a lot of fun to work with. The days we spent with them sitting around a whiteboard brainstorming content for the game are now among the highlights of my game-making career.

4 **OUTSOURCING.** Much has been made in the last few years about the growth of outsourcing. As development costs rise, outsourcing is an attractive way for studios to manage costs. When we started Hothead, we decided to be open to using external partners in development.

Overall, this worked out very well on PRECIPICE. We worked with other companies around the world for art production, Q/A

Austin GameDevelopers Conference

INDEPENDENT GAMES FESTIVAL

GAME CAREERSEMINAR presented by GameDevelopers Conference

WORLDS IN MOTIONSUMMIT

September 15-17, 2008

Austin Convention Center | Austin, TX

KEYNOTE ANNOUNCED!

Bruce Sterling

American Author

THINK SERVICES

"Computer Entertainment 35 Years From Today"

Preview Guide and Keynote information is now available on **AustinGDC.net**

myGDC

Connect, Engage and Interact **with over 5,000 designers, programmers, publishers, and game professionals** currently in the myGDC community.

Austin GDC Features Sessions Covering:

Audio • Online Games • Writing for Games

Business & Marketing • Design
Social Networking & Community • Technology & Services

DON'T PAY ~~$995~~ ON-SITE.

REGISTER AT AUSTINGDC.NET BY 8/27 AND SAVE UP TO 30% ON A CONFERENCE PASS

testing, and even some programming. Key to our approach was understanding the strengths of all the suppliers we talked to, and matching the specific needs of a particular asset or service to those strengths to maintain the same high quality we expected from our internal team. To achieve this, we never stopped looking for new suppliers.

At the end of the day, we only worked with a small handful of companies on the game, but we have met and developed relationships with dozens. Knowing many providers has given us access to a broader range of specialized talents that we can draw upon on short notice.

Looking back on the project, it turns out the biggest benefit from outsourcing was not direct cost—in other words, using an external person versus internal created little savings on the project. The main advantage it offered was flexibility of team size—convenient, since staffing requirements vary a lot between different phases of production. Being able to effectively grow and shrink our team almost instantly through outsourcing really helped us keep the size of our internal team small and efficient. This had many benefits, allowing us to achieve the results of a larger multi-project studio without losing the small team vibe.

Determining how the story translated into game length was difficult throughout the entire production.

5 **PROJECT MANAGEMENT FLEXIBILITY.** There is a lot of interest among game developers in new project management methodologies such as Agile development and Scrum. When starting on PRECIPICE, we recognized the value of these new approaches, but we also made a conscious decision to avoid following the rules of these new paradigms to the letter. Rather, we took a "buffet" approach, using what worked and being willing to ignore what was not appropriate.

Most importantly, we continually reassessed our practices so that we could add new things as necessary and drop whatever was not working. For example, when it was appropriate to add a quick morning meeting with the programmers similar to the Scrum meeting, we did. When we felt we should bring in other members of the development team, we did that too. But when we decided that meeting so frequently was no longer working, we dropped it. The key was to remain constantly vigilant and not continue doing something just because we had done it up to that point.

WHAT WENT WRONG

1 **MANAGING SCOPE.** Old habits die hard. We started Hothead on the principle of making smaller games, experiences that could be digested in a reasonable amount of time. The episodic format is a great way to achieve this, so we planned the game series' story arc appropriately and scoped each episode to have several hours of gameplay. And yet, numerous times during the project, we noticed the scope creeping larger, as the team fell into old habits we all had from making larger games destined for retail.

At one point, the issue grew to such an extent that we had to step back and rethink how big we were making the game. We realized we were setting a trend with the first episode that we would need to maintain with future releases. These adjustments were not easy to make and cost us extra time.

Another old habit manifested when we completed the content for the first episode. As a large portion of the team rolled onto the second installment, I started hearing discussions on the team that made it clear we were thinking about EPISODE TWO as a sequel rather than an installment of a series. With episodic delivery, it is imperative to think of each release as an iteration on story and content using a static engine. This principle is important when we plan to deliver new episodes every four months and perhaps even essential if we consider the episode-per-month schedule adopted by Telltale Games on its SAM & MAX series.

Although we decided to embark on something different, we did not decide in advance what practices we would use to get there. Looking ahead, we will have to ask ourselves if we are sticking to those practices or if we are sliding back into older, more familiar habits that will prevent us from reaching our goals.

2 **MIKE AND JERRY'S FIRST GAME.** Earlier, I described the benefits of directly collaborating with Penny Arcade. I would be remiss to omit the less pretty side of working so closely on a game with two guys who had no prior direct experience in game development.

Apart from coming to terms with the sheer amount of work they had to do, probably the biggest shock for the Penny Arcade guys was seeing a game that was half finished.

As developers, we took for granted that we could look at the game when it was midway through production and envision

where it would end up as additional layers of polish were added. For them, it was much more of a shock to see their characters and world in a half-finished state. We needed to be more diligent in helping them understand what was still first-pass, and how we were going to make it complete.

The game length problems I mentioned were exacerbated by their lack of experience as well. Story and dialogue are the key factors that determine the length of an episode, and those items came directly from Penny Arcade. Since virtually no one at Hothead had worked on adventure games before, it was like the blind leading the blind with no one having intuition about how the current story translated into game length. This was one of the factors that led us to work with LucasArts adventure game veteran Ron Gilbert.

A final issue in collaborating with Penny Arcade was managing who was in charge of certain decisions. As we progressed, it became clear that the dividing line between their responsibilities (art concepts, story, and dialogue) and Hothead's (gameplay and everything else) was blurry. This is especially true because of the story-based nature of the gameplay. For example, the non-linear dialogue trees in the game are all script: Jerry wrote every word. But finding your way through the dialogue is gameplay, and the type of information that gets revealed, reiterated, or emphasized during each dialogue encounter is key to making a solid adventure game and to ensuring that the player is always aware of what she or he needs to be doing.

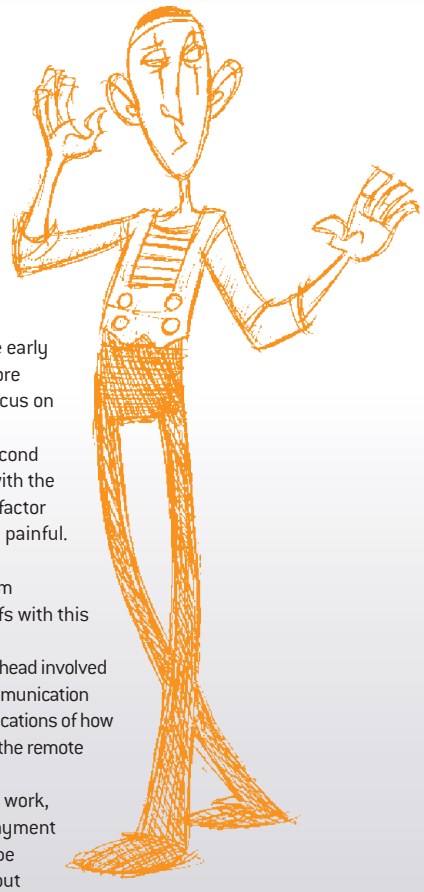
Once we got into production, we realized that a tighter collaboration and communication loop was needed to ensure decisions did not fall into the white spaces between responsibilities.

3 STICK TO WHAT YOU KNOW. Being an experienced team meant a lot of us had developed certain "best practices" that we used to maximize the quality of the final product. It was sobering near the end of the project to realize that despite knowing these things, we had simply failed to employ them.

One such practice is to prototype the newest or riskiest aspects of core gameplay early. For PRECIPICE, this core gameplay was our initiative-based combat system. The goals for the combat were to expose the player to the strategic play styles required by a turn-based system, while keeping him continually engaged with the more frantic feeling of real-time combat. This meant designing things like blocking and special attack mini-games. All together, it was enough new stuff to need lots of prototyping and iteration.

We chose a third-party engine that allowed us to get the game up and running quickly, and we did get to work on the combat system straightaway. However, we could have done more to boil down this part of the game to its essential elements so that iteration was fast and much less programmer-intensive. The optimal approach would have been to implement a throwaway combat prototype and decouple this work from the early work on engine integration and creation of other core systems, allowing a small subset of the team to focus on the "fun factor" of this essential part of the game.

Late in production, we ended up taking a hard second look at the combat system. We are quite pleased with the way it all turned out in the end, but because the refactor came so late in the process, it was both costly and painful.



4 OUTSOURCING. We realized many benefits from outsourcing, but there were definitely tradeoffs with this approach that came as a surprise as well.

First, it's easy to underestimate the amount of overhead involved in managing the relationship, flow of assets, and communication with an outsource partner. It's critical to create specifications of how the work should be done and communicate this with the remote team early in the process.

Clearly-established protocols for requesting new work, approving completed work, and even billing and payment need to be worked out and understood by everyone involved. Creating schedules and tracking how well they are being met is hard enough with an internal team, and it's that much harder when the outsource company is remote, whether just down the street or on the other side of the planet.

On PRECIPICE, we underestimated how much management this would be and ended up having to allocate additional internal staff to manage the process midway through production.

Second, we definitely learned the hard way that there is no simple rule when determining whether to create an asset in-house or to outsource it. In fact, that decision is complex and nuanced and once you make the outsourcing leap, you have to make that call on virtually every aspect in your game. For

example, all the 2D cutscenes in PRECIPICE were drawn, colored, and animated by an external studio. They did a fantastic job, and because several of the mini-games in the first episode were done in the same 2D format, we decided to have the same company make those mini-games—a bad decision.

While the vendor did a competent job, making a functioning mini-game is not the same as producing a cutscene, and engine compatibility problems on our side caused a host of functional and performance bugs with the mini-games. While in the thick of solving this, it was particularly frustrating to realize we were working so much on something the player would see for only a few minutes, and it all could have been avoided had we kept the work in-house. In the end, it took several months of rework by our internal team to get those mini-games to an acceptable level of quality.

5 **STARTUP OVERHEAD.** For most of us, Hothead has been our first time start-up. There's no one thing I can point to that demonstrates the massive distraction of starting a company but, as with any large undertaking, the devil is in the details. After having worked in established studios, it is easy to take for granted the effort that goes on behind the scenes to make the entire operation run smoothly. Someone has to set up and maintain the servers, payroll, and other HR matters, and someone has to make sure there is enough pop in the fridge. To itemize an exhaustive list would take pages.

When first starting out with just a few people, we rolled up our sleeves and did all these things ourselves. To be honest, it was

JERRY HOLKINS: ON WORKING IN GAMES FOR THE FIRST TIME

IF WE HAD KNOWN WHAT WE WERE GETTING INTO, WE WOULD NEVER HAVE DONE IT. GAME DEVELOPMENT IS AN ENDLESS SISYPHEAN nightmare warren of terrible nightmares. We wish we could go back in time, to our first meeting with Hothead, and shake our past selves, crying out: "Run, fools! Run for your very lives! Game development is a nightmare warren," et cetera. We would spend a lot of time driving home this nightmare warren concept.

Because of physics (and the fear of creating a temporal loop scenario), we can't do that. What we can do is hug our knees close to our chest, and rock slowly back and forth. We've put two years into something that—assuming people like it at all—they'll only be able to like for maybe six hours, maybe, because that's all there is! That's all we could make in two years! And seriously, we had help! There were a lot of us! Professional people! People who had made games before! And for some reason, they still wanted to make another one! At any point, they could have warned us, like an old man in a cave. Specifically, they could have described the warren.

There are a few things we wish we had known beforehand. First, not to make video games—but we covered that. Second, coming in as people who ordinarily just buy entertainment software, we didn't understand that a project doesn't actually look like anything until the very end. We had resigned ourselves to the fact that our game would be about grey blocks stumbling around a featureless world. We thought it was a bold visual style, and we applauded it. We also didn't understand that a lot of the ideas that you start with are just wrong, and need to be thrown away. More than anything else, we learned that if you want to criticize this industry, you might want to actually spend some time in it.

—Jerry Holkins, writer, Penny Arcade

kind of fun to take such a pioneering spirit. It becomes easy to convince yourself you are being a great leader by seeing what needs doing and just doing it. But as the company grew and the project got busier, these items quickly became distractions from more important things and delegating to others was not always a simple matter.

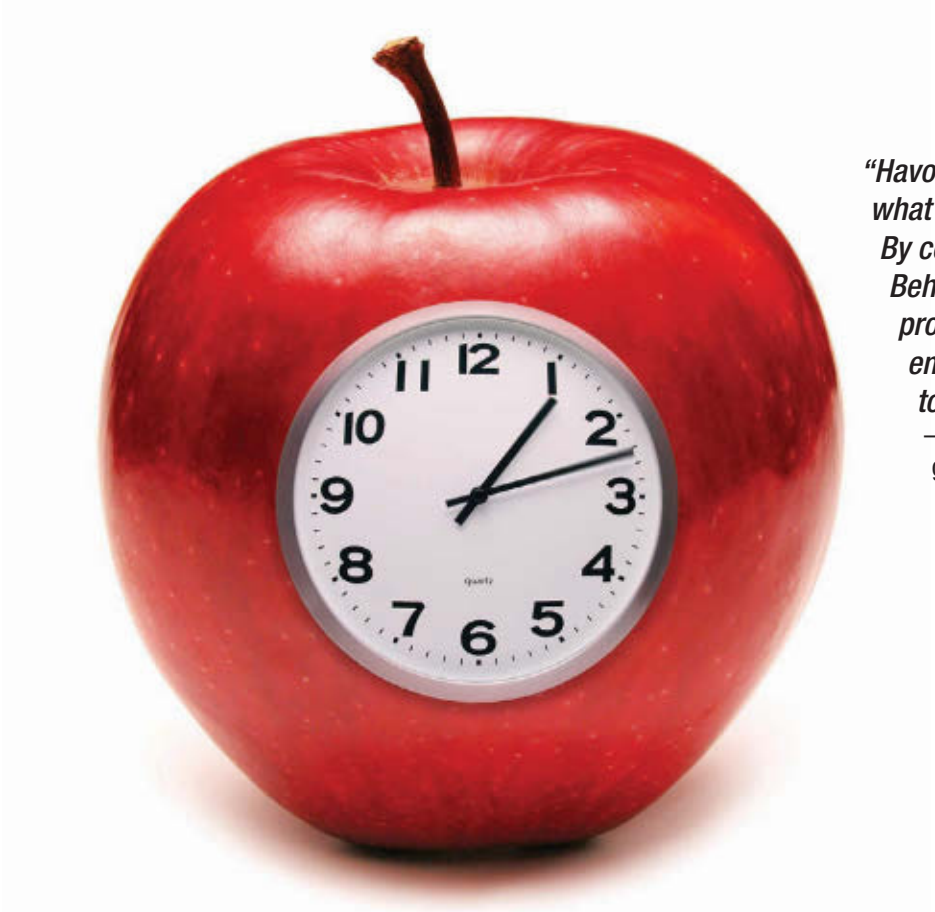
For office administration and IT work in particular, we found ourselves in this strange middle ground: There was enough work to be a distraction, but not so much that we could simply hire someone full-time. It was frustrating when I felt like I should be working more directly with the team on the game, but was pulled away by more mundane responsibilities.

IN CONCLUSION

PENNY ARCADE ADVENTURES: ON THE RAIN-SLICK PRECIPICE OF DARKNESS, EPISODE ONE was the first test of many of the principles on which we founded the company: Create a fun gameplay experience that is loyal to the IP, remain independent by focusing on digital distribution, and create a game that is another shining example of what episodic can be. We feel like we have succeeded, but there were definitely learning experiences along the way.

It's an exciting time to be making games. We would not have been able to do what we are doing if we had started a mere handful of years earlier. The ongoing changes in the development landscape mean new challenges, but also opportunities for independent developers that create exciting new possibilities.

Take a bite out of your development time with Havok Behavior™.



"Havok is an essential ingredient in what makes our games great. By consistently integrating the Behavior 5 tools across our projects, our teams are instantly empowered to take their creativity to the next level."

– Josh Resnick, co-founder and general manager, Pandemic Studios

Havok Behavior™ will save your team time in the creation of character assets and behaviors. Created with the Technical Artist in mind, it is an innovative SDK for game programmers with an intuitive composition tool for artists and designers. **Havok Behavior** allows your characters to physically interact with the game environment during game development.

- Out of the box integration with our award winning Havok Physics™ and Havok Animation™
- Over 15 "triple A" titles currently in development using Havok Behavior
- Backed by our first class, industry leading Developer Support
- Optimized for leading platforms including Wii™, PLAYSTATION®3, Xbox 360™ & the PC

havok™

Dublin • San Francisco • Tokyo

Contact us today for more information and to see if you qualify for a free evaluation copy of Havok Behavior.

www.havok.com

OUR RATING SYSTEM :

★★★★★ EXCEPTIONAL ★★★★ GREAT ★★★ FAIR ★★ POOR ★ UNFORTUNATE

MICROSOFT PROJECT 2007

By Shekhar Dhupelia

THE MICROSOFT PROJECT SERIES HAS always been based on the traditional "waterfall" style of management, closely following the methods backed by the Project Management Institute (www.pmi.org). Having used Microsoft Project 2003 since its release, I've always found it easy to lay out linear and parallel tasks, and manage the day-to-day dependencies flowing back-and-forth between programmers and the many disciplines of art.

However, I have also always considered Project to be clunky and counter intuitive, making it difficult sometimes to access commonly used features and views. In addition, the classic Work Breakdown Structure (WBS) method of tasking doesn't always gel very well with the recent push towards Agile and Scrum project management we're seeing more and more of lately. As Project tends to see fewer upgrades and iterations than many other Microsoft products, I was eager to dig into the new Microsoft Project 2007, mainly to see if either of these areas had been addressed.

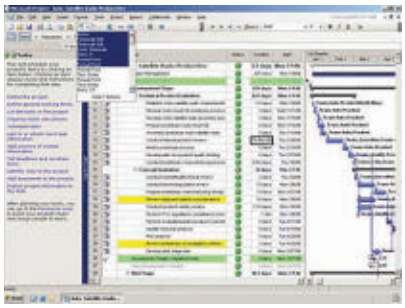
USABILITY

Usability was, right away, the biggest area in which I was disappointed with Project '07. As soon as I opened it for the first time, I realized that none of the common changes across the rest of the Microsoft Office suite had made their way into the new version of Project. Where products like Word 2007 and Excel 2007 have drastically simplified menus and user interface to manage their dizzying number of options, Project still has the same basic rows of toolbar icon docks and traditional "File," "Edit," etc. drop-down menus across the top. The good news is that previous Project users will know where to find their favorite commands right away. The bad news is that many important features remain buried under nested menus.

Further, there are a lot of indistinguishable views and features. For example, if you want to see a summary of all of the tasks you've assigned to a particular animator, you will probably start by choosing a view from options such as "Resource Allocation," "Resource Graph," "Resource Sheet," and "Resource Usage." I'd rather see the product based around the types of things project managers typically need to know, rather than big lists of the various views and forms and features available in Project.

Another issue I found unaddressed in the latest Project was levelling. I've always had good luck with breaking down all of the tasks, assigning resources, and then setting up levelling to help prioritize work and spot key dependency or resource roadblocks well before they come up. However, for some reason automatic levelling (where Project re-levels on the fly as you make changes) has never seemed to work for me. If I make a bunch of priority or resource assignment changes, it comes up with errors such as "this resource needs to be levelled on a day-by-day basis," even though the automatic levelling was already set up to do so. And then, as soon as I go into the levelling dialog box, select the exact same options as I had before, and then click "Level Now," everything comes up properly again. This all seemed unnecessary, since Project is smart enough to lay out the whole schedule if I ask it to, manually, so it should be able to do the same automatically, as I make changes.

This brings up my favorite change in Project '07: multiple levels of undo. Project 2003 suffered from only being



Microsoft Project 2007 now supports multiple levels of undo.

able to undo the last change you made to the schedule, which is downright archaic software design in this day and age. The new version allows you to revert across many levels of changes (although not infinite—it felt like the history only tracked the last 30 or so changes). But Microsoft is smart to be advertising this as a major new feature, because it basically unlocks the option of playing "what-if" scenarios. On my current project, I already applied this feature to try what-ifs like "what if this animator rolled onto the project two weeks early", and "what if we pushed the end date by one week", and "what if this environment dependency didn't come online for another three weeks?" Then, I could revert out of those changes and very quickly understand how the small changes I was making early on in the project would truly trickle down through the rest of the schedule.

Another great improvement is the highlighted changes. Any time something shifts as the result of one of your changes, the changes to start/end dates or durations will be highlighted very clearly, so you know exactly which tasks or features or phases were affected by your change. For example, if I change a resource allocation to reflect that a programmer will be on vacation for two weeks, and then re-level the schedule, the tasks that moved out as a result will light up blue, so I can quickly see the impact. This will happen to tasks the programmer is assigned to directly as well as tasks that are dependent on those prior tasks being completed first. Previous versions of Project would just put in all the changes, and you'd notice that a bunch of text had updated, but it wasn't obvious which dates had actually changed. Now you know the exact ramifications of your edits, which is incredibly helpful.

A couple of small bugs and annoyances did crop up while using Project. For example, Project will try to be smart about dimming and showing the toolbar dock icons, so only icons relevant to what you're currently doing are available. But if you're like me and click back and forth between a lot of views, and alt-tab between 10 documents and browser windows at once, you'll probably notice a bug in which the dock icons remain dimmed, making it impossible to access a feature that should be available. My solution was to quit and restart Project. Another minor annoyance from past versions of Project that still exists is that you are given a save prompt every time you close your Project file regardless whether you made any changes. This can make you a little paranoid, when you're used to other Microsoft Office products that only prompt you if you've changed something.

NO AGILE-SPECIFIC SUPPORT

Iterative development has really taken a foothold in the game industry, and with good reason. It allows for game development around end-user actions and smaller definable goals, rather than months-long feature breakdowns, and it is a really nice way to structure work that isn't totally defined from front-to-back.

Unfortunately, this is the biggest weakness in Microsoft Project 2007. Despite the number of game projects (and traditional software projects) now using Agile-style methods, this new version of Project makes no specific strides towards improving the experience for an Agile project manager. The core of the product is the same—"waterfall" development, with work broken down for months or years in advance.

A quick Google search does turn up a lot of ideas and templates on how to wrangle Project into doing some of the work needed for a purely Agile project manager, but many of the ideas are less than optimal, and more importantly, none of them are inherent parts of how Project works. Where products such as Hansoft have embraced Agile and developed features specifically geared toward managers and developers in an Agile project, Project seems strictly designed toward waterfall development.

VISUAL REPORTS

One very nice enhancement to Project 2007 is the report generation tools. While the underlying engine seems to be creating similar reports as previous versions of Project, the interface for creating reports is much more intuitive than ever before. You can very quickly spit out Excel or Visio documents and completely customize which fields are used and the granularity of the data you want to see (i.e. hours/days/weeks). The interface isn't quite as logical as the graphing tools in Microsoft Excel, but it's much easier than before.

THE BOTTOM LINE

Many producers in the game industry, as well as technical directors, art directors, and other team leads have learned to use Microsoft Project to help organize and drive themselves and their team. Project has a large following of supporters—and a lot of detractors as well. Some teams quickly adapt to traditional "waterfall" scheduling, following the Work Breakdown Structure (WBS) method of drilling down to individual tasks. Many teams, however, find this process and the software to be unwieldy and rigid. This is particularly true for the many game teams now running on Agile/Scrum.

The bottom line is that Microsoft Project 2007 won't do much to change your opinion of the software. If you consider it difficult or unhelpful, the new version doesn't tread much new ground in terms of usability, and doesn't do much to embrace more non-traditional styles of project management. It seems unfortunate that Microsoft chose not to integrate all the great improvements they've made to the rest of the Office suite, or do more to improve ease-of-use for some of the most daunting parts of Project. But, for those large teams that already schedule and track their work this way, the enhancements are a nice iteration on what you're used to working with, and the multiple levels of undo alone will drastically improve your productivity.

SHEKHAR DHUPELIA *has been in the game industry since 2000, first as a programmer and now as a producer. He most recently shipped* NBA Ballers: Chosen One *for Xbox 360 and PlayStation3. Email him at* **sdhupelia@gdmag.com.**

MICROSOFT PROJECT 2007

★★★½

STATS

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
http://office.microsoft.com/en-us/project/default.aspx

PRICE

Standard: $599.95
Professional: $999.95

PROS:

1. Reporting and Data Management simpler than ever.
2. Multiple levels of undo opens up "what-if?" scenarios.
3. Instantly accessible to prior Project users.

CONS:

1. No specific improvements to ease Agile/Scrum project management.
2. Didn't get the visual and usability improvements of the rest of the Office 2007 suite.
3. More of an incremental upgrade than a major improvement over Project 2003 at the price of a major new version.

STEVE THEODORE

THE DEATH OF RIGGING?

NO MATTER HOW LONG YOU'VE BEEN out of school, August always makes you think about reinventing yourself. All those back-to-school wish lists—new clothes, new backpacks, a chance to improve your grades or upgrade your social status—leave a mark that never quite fades. It's a good time of year to break out of a rut.

If you're looking for a good rut to start with, think about animation. Animation is the absentminded professor of the game art disciplines: highly technical, but rather stuck in its ways. Certainly, nothing has come along to revolutionize animation in the way subdivision sculpting has changed modeling, or Shader Model 3.0 has changed texturing and effects. Animators are still keyframing away with only incremental improvements to the tools that debuted with 3ds Max and Maya, back when the Dreamcast was the hot gaming machine and gas cost $1.27 a gallon.

THE ANIMATOR'S FRENEMY

The guardian of the status quo in animation is the character rig. The animation rig is really the lynchpin of a studio's entire animation effort. It's the animators' main UI element. It provides the engine with specialized markup. And it is the backbone of asset control. Most importantly, the things your rig does well (or does poorly) subtly flavor every aspect of your work.

Unfortunately rigs are—let's be frank—a pain in the butt. Even the best rigs are complex, touchy, and hard to maintain. Keeping track of the rig through its evolutions and keeping different files in synch is a nightmare. No single rig can satisfy all animation needs equally well. Some rigs bury themselves in layer upon layer of features until they feel (and perform) like Rube Goldberg machines, while others, hoping to stay lean, become virtual straightjackets. If we're talking about reinventing things, this might be a good place to start.

Only masochists want to go back to animating directly on FK bones all day. But what if you could keep the good parts of a complex modern rig—helpful UI, the right control spaces, and efficiency—without the management overhead or 60 million control nodes cluttering up your scenes? Would you be interested?

Well, there is an alternative to complex rigs, one that tackles many of the problems we ask rigs to help with from a very different standpoint.

LIKE AN ONION

Animation layering is a workflow that builds animations up out of layers in much the same way Photoshop assembles a bitmap image out of bits and pieces. Instead of forcing you to plan ahead for every possible contingency in the design of an omni-competent do-everything rig, layering lets you slap together whatever techniques make sense for a given shot. It can be a very compelling alternative to the standard way of doing things.

Layering evolved quietly outside of conventional animation pipelines. It's a key tool for teams that rely heavily on motion-capture or simulated data, but it's also starting to find favor among traditional animators. The eternal struggle between the proponents of mocap and of hand animation is written deep into games industry lore, so we don't need to revisit it here. And in any case, the fact that layering evolved in response to dealing with mocap doesn't mean that's all it's good for. It is, though, a good way to understand the essence of the approach.

Whatever you think about mocap, everybody agrees that it's a slog to work with. Conventional animators' eyes roll back in their heads when they open a graph editor and see, instead of cheerful colored lines, densely packed keys marching like army ants across their screens. The drawbacks to such dense data are all too obvious: It's slow to work with, and fixing it in place is almost as bad as returning to the bad old days of animating one frame at a time. And of course, the data is all FK.

Aesthetics aside, very few animators really embrace mocap for the sheer joy of working with thousands of keys. The need to tame that mess is what gave rise to animation layering.

The arms race between mocap and traditional keying is almost a replay of the old battle between bitmap painting and vector illustration in the 2D art world. Back in the days of the Super Nintendo Entertainment System and the Macintosh II, bitmaps were for photos or scanned clipart; large format illustration was more often done with vector drawing programs like Illustrator or CorelDraw. Vector art made reshuffling, resizing, and replacing pieces infinitely simpler than chopping up bitmaps—until Photoshop 3.0 introduced a generation of artists to layers.

Photoshop layers didn't miraculously end the seesaw battle between bitmap and vector illustration programs, but they gave bitmap packages a gigantic boost in flexibility and freedom. Being

STEVE THEODORE *has been pushing pixels for more than a dozen years. His credits include* Mech Commander, Half-Life, Team Fortress, *and* Counter-Strike. *He's been a modeler, animator, and technical artist, as well as a frequent speaker at industry conferences. He's currently content-side technical director at Bungie Studios. Email him at* ***stheodore@gdmag.com****.*

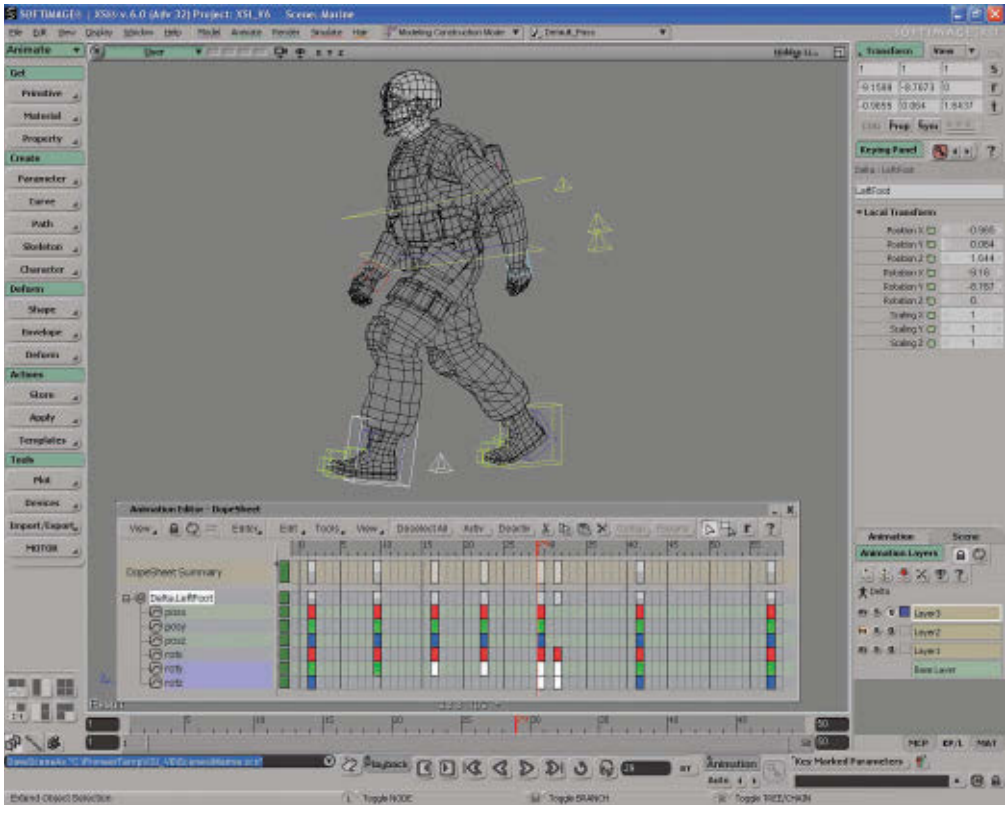


FIGURE 1 Animation layers in Softimage XSI are shown.

able to move pieces non-destructively, to transform parts of the image while leaving the background intact, and to experiment turned Photoshop into the all-purpose juggernaut we know and love today, while vector programs were gradually relegated to technical illustration and graphic design. When layers first came out, they were a geeky curiosity, but it didn't take long for them to become a solid pillar of the illustrator's toolkit. Will the same thing happen in animation?

PHOTOSHOP JOBS

Animation, of course, is vastly more complicated than bitmaps. Even so, animation layering offers modern animators some of the same benefits that bitmap layers gave to painters—freedom to experiment, ease of working with dense data, and the ability to keep the complex texture and nuance of the original source material. Layers let you apply targeted fixes on top of dense data using the kind of simple and elegant curves you'd want from a traditional rig.

For example, if you like the base motion from a mocap file but want to exaggerate the arc of a swinging arm, you could key just the offsets into a new layer to exaggerate the motion without having to touch all the keys in the original arc. You'd only need a handful of keys to provide the extra oomph, and you wouldn't be responsible for touching the zillions of keys in the underlying move. And just as in Photoshop, you could keep your fix around to make it easy to tweak, maybe dial it up or down with a weighting value, or, once you were satisfied, collapse it down onto the original animation for a simpler scene.

COMMITMENT ISSUES

This all sounds pretty attractive. Unfortunately, you have to commit to some serious study if you want to access this kind of power. Animation tools are opaque at the best of times, and the additional complication of non-linear animation blending on top of the familiar hassles of curve management and rigging makes for an intimidating mix. It doesn't help much that 3ds Max's and Maya's animation layering tools are both afterthoughts, tacked onto existing systems rather than designed from the ground up. They both leave a lot to be desired in the UI department.

(Softimage XSI's approach, thankfully, is more neatly integrated. See Figure 1.)

Despite the differences in UI, the basic strategy is the same in all three packages. You start with some dense source data. Usually it's mocap data, but it could be a baked simulation, a ragdoll sequence imported from Endorphin, or even conventional animation that's been collapsed down to pure FK keys for convenient sharing. The "layers" of animation you add are either adding to or just replacing the translation and rotation numbers baked into your original data. It's not wizardry at all, although like everything in the animation world, it does take some doing.

LAYERED APPROACHES

For example, if your source animation is a bit anemic, you could add exaggerations to the key poses by keying offsets (add a bit to the X rotation here, take a bit off the Z translation here ...). In other cases, you might need to completely replace the underlying animation. You can do this by adjusting the relative weights of the new and old animation layers, or by brute force by going into the original keys and zeroing out the bad bits then adding better keys on a new layer. It's conceptually pretty simple. The complexity arises from managing all the different fixes and layers without going crazy.

Of course, straight layering is still old-school FK animation. The skeptics are probably wondering why you'd want to go to all this trouble if you also have to give up goodies like constraints and IK—but thankfully, you don't have to forego the finer things. You can simply add your IK or constraints into new layers. Well, "simply" might be overstating it, thanks to the Neanderthal UI, but it's definitely possible to drive the final character with a combination of baked FK keys, constraints, IKs, and expressions.

However, if you go down that road very far, you'll face a dilemma that has haunted riggers ever since the first caveman IKed a club to the head of a giant ground sloth. In the conventional rigging world, rigs that allow for possible tweaks and offsets are sometimes too complex to use effectively, but rigs that are too stripped down are sometimes too limiting.

Organizing a layered animation presents the same challenges. If there are too many layers and too many detailed tweaks, the scene becomes incomprehensible—but too few, and all the cons of working with dense baked data start to hold you back. You can override your baked data with IKs and constraints, but if you start to think about layering the animation on those as well, you'll run the risk of making your head explode. When do you stop?

There's no single right answer, but there are a couple of good questions to consider when organizing your layers.

1. Will you ever touch this again? If you've made a tweak and you're withholding artistic judgment to see how it plays out, you probably want to keep that layer around. If you've just fixed a technical glitch, though, you might want to merge your fix down to clean up your workspace.

2. Are you planning on retiming? One limitation of working with baked data that layering doesn't really eliminate is the fact that baked keys don't stretch as well as sparse keys with nicely tuned tangents. If you think you'll need to lengthen your animation, you'll probably want to keep your layers around longer to make sure you don't throw away higher quality data.

3. Are you sequencing or sketching? If you're using your non-linear animation the way it was originally designed, you'll be stringing together multiple clips with blends and sometimes overlays. You'll want to keep your clips separate until you're really happy with the gross timings and blocking of your scene. On the other hand, if you're using layers to rework a single sequence–for example if you're trying to lend some life to a lackluster captured cycle–you'll probably want to add and delete layers frequently as you sketch out the effects you want. In this case, baking frequently has the positive side effect of helping you spot hitches, since it forces all the overlapping clips into a single set of f-curves, where it's easy to iron out small irregularities by deleting a few offending keys from one place.

CURVY MEANS FAT

Perhaps after all this, you're still unconvinced. The animator's horror of over-keyed curves is a hard thing to shake. If that's the case, you ought to consider using some of the tools we've talked about to do blockings and establish key poses for conventional animations.

The talk that Jeremy Yates and Judd Simantov of Naughty Dog presented at GDC 2008 gave a detailed and very illuminating look at how the animators of UNCHARTED: DRAKE'S FORTUNE used mocap data as a foundation for more traditional animation. They built tools for grabbing poses and rough timings from the motion capture data but heavily tweaked the results so that the overall flavor was more like hand animation than straight captures.

Without committing completely to an all-layered approach, they were able to simplify and bulletproof their rigs by treating IK and constraints as add-ons rather than permanent features of a monolithic rig. They got a lot of the virtues of a layered approach—flexibility, creative freedom, and speed—without giving up on the virtues of careful hand keying.

Even though your animation needs are almost certainly quite different, their experience illustrates how much power can be had when you give up on the quest for perfect rig and settle instead for a pretty good one combined with pretty good layering tools.

NEW TRICKS

We must admit that reports of the death of rigging are a trifle exaggerated. Rigging might not be dead, but it's not quite the only game in town anymore—and anything that busts up the logjam in animation has got to be a good thing. So take a break and try to teach yourself the Maya Trax Editor or Max's Animation Layers next weekend. You'll probably be a bit baffled to start with, but once you see what new tricks you can do, you'll be glad you spent the time.

Ready for GAME Time!

TOKYO GAME SHOW 2008

10/9 THU. 10 FRI. 11 SAT. 12 SUN

Business Day | Open to the Public

http://tgs.cesa.or.jp/english/

Venue:Makuhari Messe

Organizer: Computer Entertainment Supplier's Association(CESA)
Co-organizer: Nikkei Business Publications,Inc.(Nikkei BP)
Supporter: Ministry of Economy, Trade and Industry (METI)
Special Sponsor: NTT DoCoMo, Inc.

CESA Nikkei Business Publications, Inc. docomo CoFesta 2008

NOEL LLOPIS

THE HEARTBEAT OF THE PROJECT

HAVE YOU EVER GIVEN SOME THOUGHT to why you decided to become a game programmer? I'm pretty sure it wasn't to do mundane, repetitive tasks. Yet sometimes we find ourselves spending a significant portion of our time making sure that the code compiles for all platforms, or that there are no potential bugs lurking in the depths of the game, or even building the assets for each level and running them to make sure they load correctly.

Clearly, those are all things that need to be done, but if they are so repetitive and mindless, couldn't we put some of the computers around us to good use and have them do the job for us?

A build server will do all that and more, much faster and more reliably than we could, and it will free us to work on the thing that made us fall in love with this industry in the first place: the game.

GETTING OFF THE GROUND

Before we can start thinking about setting up a build server, we need to be able to build the game with a single command from the command line. No clicking around, no GUI apps, no multiple steps, no magical incantations that only work during a full moon. Just type a command and the build for the game and all its libraries starts.

This is not just a necessary step to set up a build server; it's a very good engineering practice. So if you're not there, spend some time on it right away and you'll be glad you did when candidate submission time comes around.

Building the game with a single command should be fairly easy, but the specifics will depend on your environment and build system. If you're using Visual Studio, you can put the game and all the libraries in a single solution with the correct dependencies. Then you can invoke the `devenv.com` command line program specifying the solution and configuration you want to build: `devenv.com mygame.sln /build Debug`. You can wrap that up in a single batch file `buildgame.bat` for extra convenience and you're done.

If you're using another build system, such as make or jam, you can probably already build it with a single command. If you're using a bunch of mode-made scripts, at least wrap them all up in a single script file so they can be run with a single command.

Just building the game with a single command isn't enough. We must have a way to automatically detect whether the build succeeded or failed. Fortunately, most build systems (including `devenv.com` and `make`) return an error code when the build fails. If you're rolling your own build script file, make sure to capture build failure and return an error code as well.

SETTING UP THE BUILD SERVER

A build server should be a dedicated machine with access to version control. Whenever a new build is needed, the server syncs to the latest version in version control, starts a build, and notifies the team in case of any errors.

That's a fine start, but we could make things much better. For example, we could include the error message in the notification email so programmers can see right there what the problem was instead of being forced to sync and build the game themselves. We could also trigger a build in different circumstances (for example, code checked-in, forced by a person, or some other event) instead of only at fixed intervals. We might want to distribute the build across multiple machines, or keep logs and make them available on a web page, or format emails better, or use more direct notification methods ...

Put away those Python reference manuals because fortunately, someone has already done all the work for us: Cruise Control (and CruiseControl.Net). It's a free, open source build server program with all the bells and whistles that you could possibly want (see "Cruise Control" in Resources). And did I mention it's free?

There are three main parts to it:

1. *The build server.* It runs as an application or a Windows service. It's configured through a very simple XML file that tells it when to sync, where to sync, what to build, and how to report it.
2. *The web front end.* CruiseControl features a pretty, web-based dashboard showing all the builds, their status, past logs, and other pertinent data.
3. *The system tray notificator.* This is a little app that runs in the system tray and shows the status of all the builds and notifies you of any changes right away with a message and by playing some sounds. This is my favorite way to keep up to date with the build status.

You'll be up and running in about 10 minutes. The most complicated part is

NOEL LLOPIS *recently gave up a steady paycheck, and decided to follow his lifelong dream of being an indie game developer. He keeps busy by pretending to do everything from programming and design to business and IT at Power of Two Games. When he was still getting paid, he worked on* THE BOURNE CONSPIRACY, DARKWATCH, *and the* MECHASSAULT *series. Email him at* **nllopis@gdmag.com**.

probably installing a web server (if you don't already have one) and getting the web dashboard running. You'll spend a few more hours tinkering with it to get it "just right," and then you're done. The only time I have to mess with it is to upgrade to a new version every so often. Other than that, it's virtually maintenance free.

At this point you'll have a fully featured build server in place. It verifies that the game can be built from the latest checked-in version of the code. It notifies developers of failed and successful builds right away. It increases version numbers, keeps a build history and statistics, archives executables, and emails logs.

CruiseControl and CruiseControl.Net are the two build servers I have most experience with. There are other build servers out there, with slightly different features, integrations with different environments, and so forth. Some of them are commercial and come with full support in case you're more comfortable with that model.

It's important to stress that a build server is not intended to be the only machine that builds the game. Every programmer (and maybe every member of the team) should be able to build the game in his or her own machine from scratch. The build server is there to verify that all the checked-in changes build correctly on a clean machine, and to make sure that all platforms and configurations are building successfully.

Any official builds should be created exclusively from the build server, though. Especially any builds distributed externally to publishers or manufacturers. This ensures that the build is clean, was created in a repeatable manner, and is free of any idiosyncrasies from a particular machine.

HOW OFTEN?

Once the build server is in place and is producing successful builds reliably, the question arises of how often to make builds of the game.

It used to be considered good practice to do a weekly build. The team would start ramping things up on Thursday to try and get a build out the door by the end of the day on Friday. Anybody who has done that knows how stressful it can be and how it can easily become a bottleneck.

Why wait a week if you can do one every night? More teams started switching to the daily build, which is much less stressful because there are fewer changes in each new build. It also gives the team a chance to fix anything that was found broken in the previous day's build. Soon, teams took it beyond the daily build and started making two builds a day, or even one every hour.

The build server has been very appropriately described as the heartbeat of the project. A "green build" is one heartbeat and one small step forward. A "red build" is done when something is wrong and needs to get fixed as soon as possible. If you have a red build several days in a row, the project is in serious trouble. The more often you make a successful build, the better. You'll find fewer surprises and stay more on course that way.

My favorite approach is *continuous integration*. With continuous integration, the build server starts a new build as soon as there's a new check-in. If multiple check-ins come in while the build is in progress, another build starts right after it's done, with all the new changes queued during that time. When following this practice, programmers sync to the latest version often, make small changes, and check-in code frequently, rather than batching many changes. Very conveniently, Cruise Control has a setting to start builds whenever anything changes.

The main benefit of continuous integration is that you are notified as soon as a check-in breaks the build—not a day later, or even an hour later, but minutes later. It tells you, "The last build was good. This one is not." You can look through the last couple of check-ins that happened during that short time period and quickly narrow down the problem and fix it. Imagine trying to narrow down an elusive crash bug from all the check-ins for a full day or two!

Another benefit is that all programmers are working on a version very close to the latest one. This means that there are fewer source code conflicts when checking-in code, and fewer surprises lurking in the code. The flip side of that is that working on the latest version is living in the proverbial bleeding edge. It's not unusual for someone to check-in code that has some accidental bad side effects. As long as those bad check-ins are limited, and that whenever they happen they are fixed right away, I have found the benefits to outweigh some instability in the main branch.

Some of the ways to minimize disruptions when working with continuous integration are:

- make sure that any code compiles before checking it in (that should go without saying!)

- execute a fast set of unit tests to verify that basic functionality is working correctly, and
- have the build server notify everybody as soon as there's a broken build so it can be fixed and so that nobody else syncs or checks-in any code while the build is broken.

NEED FOR SPEED

Ideally, I'd like to check in some code and see whether the build server found any problems right away. In the real world, things can be much slower. After all, the build server needs to sync to the latest code, kick off builds for multiple platforms and multiple configurations, and perform some other time-consuming steps.

Even so, there is work we can do to get feedback as soon as possible.

Perform incremental builds during the day, so only the affected sections of the code need to be built. It's still a good idea to do a full build at least every night to make sure that everything can be built from scratch.

Set up each platform and configuration as separate builds. That way you get feedback as soon as one of them completes. The only downside is if an error makes it through that causes all the builds to fail, get ready for lots and lots of broken build sounds playing all over the company.

Speed up build times through good physical dependencies, modularity, precompiled headers, and good use of forward declarations (see "Games From Within" in Resources).

Split up different builds and configurations in different machines. The easiest way is to set up one machine per platform and configuration (or maybe do a couple of configurations per machine). Cruise Control lets you easily integrate several build servers into the same web dashboard and system tray application, so this is a very easy solution.

DON'T SKIMP ON HARDWARE

Get the beefiest computers you can afford. Throw fast CPUs, disk access, and gigabit ethernet. Get multiprocessor cores and make sure your build system takes advantage of them. Does it sound like a lot of money? Not when you take into account how few servers you'll have and how much time you'll save all the members of the team.

I have tried several distributed build systems, and even though they can sometimes be beneficial for some codebases, I'm still not a huge fan. I find that you can often achieve the same (or better) results by using multiple processors and good build architectures, and you avoid the complexity and overhead of a distributed build system.

One "gotcha" we ran into when we scaled our build farm beyond about 15 build servers was that each of them was hitting our version control repository every few seconds to see if anything had changed. That wasn't a trivial operation, and so many servers doing it so frequently definitely slowed things down to a crawl.

To remedy that, instead of having the build servers poll the overtaxed source control server, we had the source control server push out a notification. Whenever there was a check-in, the source control server changed a timestamp in a file located on an internal web server. We changed the build servers to constantly monitor the internal web server for changes in that file, and whenever it changed it triggered a build, which completely eliminated the overhead on the version control server.

BEYOND THE BUILD

So far, we've only been talking about building the game. But the build server is a great tool that we can put to good use for many other purposes.

Why restrict ourselves to just the game? All the in-house tools would also benefit from getting the same treatment. We can even take it a step further and deploy the freshly-built copies of all the tools on a network drive or web page so they're available to the whole team.

The build server can also double up as a symbol server. That makes it much more convenient for programmers to debug an earlier version of the game and libraries and have all the debugging information available without having to rebuild everything locally.

There's no reason to limit the build server to just building source code. One of the most useful things you can do with it is use it to build game assets as well. Building assets is usually a slow process. Having a fast asset build system that can correctly perform incremental builds is crucial to keep asset build times down.

Build servers are general enough to perform just about any task. Running both unit tests (small tests on each class or function) and functional tests (tests that exercise a larger module or even the whole game) are perfect uses for the build server. Functional tests can be pretty slow, so make sure that they're treated as a separate build and not as the last step in building the game. Nobody wants to wait for hours for all the functional tests to complete before they can see the successful build status after a check-in.

The sky is the limit with what the build server can do. We use it to run static analysis of our source code, checking for spots in the code that can lead to subtle and dangerous bugs (uninitialized variables, implicit type conversions, and the like).

Another great use is to run through the different levels of the game, recording frame rate at different points of each level, logging the results, and failing the build if it ever drops below a certain threshold. Having the performance history for specific levels can be really useful to narrow down why a particular section is chugging at 20fps but was running at a solid 60 a couple of weeks ago. For bonus points, integrate all the collected data into easy-to-visualize graphs available through the web front end.

The build server is definitely the heartbeat of a project. Keep those check-ins coming and those builds green, and you know you're heading in the right direction.

RESOURCES

CRUISE CONTROL
http://ccnet.thoughtworks.com

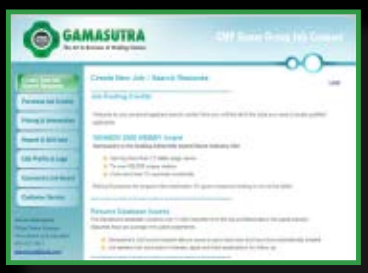
GAMES FROM WITHIN
www.gamesfromwithin.com/articles/0403/000013.html

Jobs on GAMASUTRA

The Art & Business of Making Games

The Leading Professional Game Source for Job Seekers and Employers

NEW! Job Seeker Center



NEW! Employer Job Manager

NEW! GDC Events Job Section

The Web's Largest Game Industry Resume Database and Job Board

Log onto Gamasutra.com/jobs

THINK SERVICES
A DIVISION OF UNITED BUSINESS MEDIA LLC

DAMION SCHUBERT

HUMOR ME

DARKER! DEEPER! MORE SERIOUS! These were the marching orders given to the SHADOWBANE writing team.

The world of Aerynth was a brutal one, appropriate for our PvP-oriented gameplay, with a backstory of politics and treachery spanning centuries. To be honest, even as a developer I would be hard pressed to remember the names of any of the major NPCs. What I do remember is that we had a combat ability called "hammer time."

Beyond the LEISURE SUIT LARRY series, there is no substantial comedy genre in video games—at least not like in film or TV. And with good reason: Funny is hard, especially in a genre in which you don't control the rhythm of the narrative. It's almost impossible to sustain humor over the course of a 10-hour game.

Still, you don't have to be in it just for the yuks to add moments of levity to your otherwise serious games and virtual worlds. Proof that it works can be found in our mega hits. How many times did WORLD OF WARCRAFT, GUITAR HERO, and GRAND THEFT AUTO make you laugh? These aren't explicitly comedies, but they are games with hardcore audiences that used comedy with surgical precision to enhance the experience.

APPETITE FOR AMUSEMENT

Do players really want levity in their gameplay experiences? I'd say yes, judging from what players do with games once they leave our hands. A lot of people might be watching videos of raid strategies and speed runs, but those are hardcore flicks for hardcore audiences. A lot more are watching fan-made comedy shorts like "The Internet is for Porn," "Red vs. Blue," and "Halo Human Pyramid."

Players are investing significant amounts of time when they play these games, so much so that the games have become a meaningful part of their lives and identities. When players create and consume humor in game worlds, it validates that investment and assures them that there are others like them. When they share these moments with people who aren't playing the game in question, they are assuming that laughter is a game principle that will transcend the boundaries of the play experience. Game designers and developers should foster this as much as they can.

LEISURE SUIT LARRY IN THE LAND OF THE LOUNGE LIZARDS, 1987.

IMMERSION AND CADENCE

Too many designers are far too worried about breaking immersion. They want to create dark and dangerous worlds, where every moment is a life-and-death struggle and one wrong decision can be the end of everything. Oftentimes, this tension heightens the perceived consequences of failure, hiding the fact that the player is only one reload away from being back at the last save point. As a result, designers often make experiences that are overly grim and oppressive. A 10-hour game without a joke is 10 very heavy hours for players to endure.

Filmmakers have a much more nuanced view of immersion, and they leverage humor to make it work much better than we do. Comedy in cinema provides an important emotional release, even in tragedies and especially in action films. Good directors carefully map out the emotional cadence, almost like a roller coaster ride, alternating between tension and release. The problem facing the comic relief character and the one-liner delivered by the story's hero are instrumental moments in keeping the audience engaged.

Filmmakers can utilize moments of levity easier than game creators because they have total control over the rhythm of the thing they're making. Some designers claim that games will never do this as well. Interactivity by its very nature puts pacing in the hands of the player.

To counter this argument, I point to PORTAL. Valve did a masterful job of delivering moments of humor, and did it just at the right times, with most jokes being a reward for success, offering cathartic release and a real sense of progress. Play the game with the sound off, and it's almost not worth playing.

WHAT A CHARACTER

Bruce Willis blowing up elevator shafts in *Die Hard* was pretty cool, but it's the "Yippee-kay-aye, mother—" that made him a badass. Cheesy one-liners exist not just because they're fun; they also humanize the character, making him easier to empathize with. Action movies with a silent hero are oddly antiseptic—and kind of creepy.

With DUKE NUKEM being a very successful exception, most games avoid giving the main character a personality so the player can more easily project him or herself into the hero's shoes. But even if you buy into the blank slate hero theory, there's no excuse for most of the other characters in the game world—completely generic macho marines, completely generic psychopathic

DAMION SCHUBERT *is the lead combat designer of an unannounced online title at BioWare Austin. He has spent nearly a decade working on the design of games, with experience on MERIDIAN59 and SHADOWBANE as well as other virtual worlds. Damion also is responsible for Zen of Design, a blog devoted to game design issues. Email him at* ***dschubert@gdmag.com****.*

mercenaries, and completely generic and offensively stupid damsels in distress.

Designers don't need to turn their games into comedies. No one wants *Lost* to be a Moe, Larry, and Curly routine, but how tedious and unwatchable would the show be without the rotund, laid-back Hurley commenting, "Dude ... " every once in a while? Similarly, HK-47 and Minsc are probably the two most popular characters in BioWare's storytelling-based RPGs. Both roles act as comic relief, offering levity as a pause—and a refreshing reward—between long periods of combat and more serious storytelling.

Characters should never upset the tone of a game, but designers should also challenge that tone from time to time. In my career, I've seen many web sites for post-apocalyptic MMOs with harsh and brutal worlds where society barely scrapes by to survive. The concept art is occasionally beautiful, but oh so depressing. If you have standard MMO play patterns, your players will spend 10 to 20 hours a week in your world, until they quit or kill themselves in an emo-enduced fit of pathos. The irony is that FALLOUT succeeded not because it is grim, despairing or brutal (which it is) but because it is also funny, quirky, nostalgic, and introspective.

HANGING THE LAMPSHADE

In *Roseanne*, Alicia Goranson played the eldest daughter, Becky, until she quit her role to go to college. Another actress took over—and one of Roseanne's first lines to her "new" daughter on the show was, "Watch it, young lady. You can be replaced."

Four years later, when Goranson returned, Roseanne says, "Where the hell have you been?"

Hollywood writers call this "hanging a lampshade." Instead of hiding some absurd part of the premise or story, call attention to it. Another example appears frequently in shows like *C.S.I.* and *House* when a character points out how odd it is that the crime-solving team is faced with one hugely mysterious case a week.

These sorts of jokes act as rewards for long-term viewers. In a roundabout way, they acknowledge the intellect of the viewers, as if they're saying, "Hey, we know this is absurd, but work with us here ..."

Every now and again, you'll see lampshade hanging in games. Once, while playing WORLD OF WARCRAFT my quest was to go kill 30 of something-or-other. The reason: "I don't need to give you a reason, do I?"

An even better example is from the victory screen of the original GUITAR HERO. If you beat the game on the expert setting, the game declares you a "legend," and then points out how impressive that is, given you were just playing covers.

Both jokes occur late in the gameplay experience. If either had occurred too early, they would have run the risk of turning the player off by pointing out flaws in the premise or mechanics. But because they happen so late, the player sees it as a reward.

FINDING THE RIGHT TONE

In the old days of *Magic the Gathering*, the flavor text was almost all serious, sometimes even with quotes from Shakespeare or Edgar Allen Poe thrown in. Over time, the game creators began experimenting with humor in the expansion packs. For example, the flavor for Lava Axe was originally, "Meant to cut through the body, and burn straight to the soul." Today it reads, "Catch!"

About six years ago, the *Magic* team at Wizards of the Coast did some market research to determine the most and least popular flavor text. Of the top five chosen, four are jokes. (One example: "An army of squirrels is still an army.") However, two of the bottom three were also attempts at humor that had fallen flat, such as the werebear, who "exercises his right to bear arms." And to make matters more complex, some cards appeared on both lists—you can't please everyone, it would seem.

When adding humor, finding the right tone is critical. *Magic* is a game world that appreciates wit. WORLD OF WARCRAFT isn't afraid to be silly. John McClane, on the other hand, doesn't tell knock-knock jokes. And the jokes that Shakespeare put in his tragedies are all appropriate in a world of intrigue, incest, and murder.

Still, you can color outside the lines a little bit. The AGE OF CONAN team has done a masterful job of creating the grim and brutal adult world that Robert E. Howard envisioned. One particular quest stood out.

A prissy noble asked me to go find his "princess." After slogging through a sewer for half an hour, I discover, sitting on a pedestal, surrounded by a very large armed guard, a Chihuahua named Princess. I cracked up. Despite the absurdity of it, I could imagine being Conan and discovering what sort of fool errand I had been sent on. I couldn't wait to get back to the quest NPC to hear his justification. Was it out of tone? Possibly, but it really stood out from the rest of the pack. Even the people that hated the joke and felt it didn't belong would probably talk about it—and giving people something to talk about in a social space is usually not a bad thing.

HK-47 from BioWare's STAR WARS: KNIGHTS OF THE OLD REPUBLIC series.

THE UNBEARABLE LIGHTNESS OF HUMOR

We game developers have a tendency to take ourselves too seriously, and more tragically, to make games that are themselves entirely too sinister and pretentious. Perhaps it's a misguided pursuit of total immersion combined with a desperate desire for our craft to be taken seriously.

Nonetheless, I would argue the opposite is true. I don't think we'll be taken seriously as a craft until we learn to blend moments of comedy into our dramas, to master the rhythms built into the play experiences we're building, and to laugh at ourselves from time to time.

JESSE HARLIN

SURVIVING THE OVERFLOW

THE GUIDING PHILOSOPHY OF ALL freelance audio work—whether it's sound design, music, or voice work—can be summed up with the simple mantra, "Always say yes." Unfortunately, saying "yes" to everything sometimes means finding oneself with more work than a single person can actually tackle by all given deadlines. As everyone in the game industry knows, schedules slip, milestones push out, and deadlines have a way of floating around the calendar, anchorless, creating all manner of scheduling headaches.

Few things in the game industry are more stressful than juggling multiple creative projects simultaneously. Unfortunately for freelancers, it's an all too common occurrence and a risk that comes from successful networking. This flood of overlapping work can seem insurmountable at times.

Thankfully, two different lifejackets exist to help content creators stay afloat and survive the overflow.

GHOST OF A CHANCE

Ghostwriting has had nearly as long a history as art itself. Even Mozart is known to have been a ghostwriter for Austrian nobility. These days, ghostwriting is a staple of freelance creative work for games, television, and film. However, because secrecy is the cornerstone of the gig, it's difficult to find information on the subject.

After some networking of my own, I found six different contractors—those who have been ghostwriters and those who have hired ghostwriters—willing to speak to me anonymously in order to give me a sense of the ins and outs of the work.

I started by asking about compensation. As with all freelance contractor work, fees are negotiable. Some ghostwriters charge a typical "per minute of music" fee. Others can charge by the day, week, or even month. Another approach is to hire a ghostwriter for a percentage of the total creative fee, the percentage being proportional to the amount of work the ghostwriter is doing for the project.

Ghostwriting is a gig born from stress and, as such, some ghostwriters recognize this and can take it as a means to negotiate higher fees. Taking too great an advantage of the situation, however, will likely destroy all hope at a second ghostwriting opportunity.

Giving people credit for their ghostwriting is a tricky situation. As one contractor stated, "Contracts generally grant complete ownership and control of the music to the publisher, yet saddle all of the liability on the contractor. In order to make the liability as binding as possible, there is usually legal language asking the contractor to warrant that all of the content delivered has been created entirely by the contract signatory."

Some contractors will offer anything from "additional music/sound design by" down to a simple "special thanks" in the credits.

Others aren't so generous. Some contractors hire ghostwriters as a matter of survival and regard crediting their ghostwriters as a potential threat to their professional personas. Even in these situations, permission to list the game on resumes, demos, or web sites may be given in lieu of actual in-game credit.

BOWING OUT GRACEFULLY

If the contracts haven't been signed yet, another option exists: backing out of the gig altogether. Of the freelance contractors I spoke with, none had ever taken a gig and then backed out of it—or none would admit to it.

Declining a gig can be a treacherous professional hazard, especially if you want to maintain the professional relationship and foster future work. After networking your way into a job by convincing the client that you're the right person to hire, turning around and informing them that you can't handle their project is a waste of both their time and trust.

According to music industry author and networking guru Dan Kimpel, author of *Networking Strategies for the New Music Business* (ArtistPro/Thomson PTR), there are graceful ways to pass on a gig without damaging the professional relationship. "What the freelancer needs to be aware of is not to disparage the gig, or to make it feel inconsequential."

According to Kimpel, it's possible to absorb the impact of declining the work with statements such as "I'm so deluged by commitments right now that I don't feel I'd be able to devote the necessary bandwidth and give your project the attention it deserves."

He also stressed to me the importance of making sure the response to the potential employer is laced with emotional language. Words such as "feel" or "for me" have a psychological effect that implies empathy and a connection to their project, even while trying to deflect the work from your already busy schedule.

Lastly, one final approach to declining work without saying "no" is to price your services out of the running for the gig. By asking for a rate that is higher than the project has budgeted or by insisting on residuals, the team looking to hire you may be forced to decline your bid. The risk here is that they won't decline and will instead agree to your higher rates leaving you with the overflow of work you were looking to avoid.

But then again, with an increased rate you could always hire a ghostwriter ...

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**.

THINK SERVICES

GAME CAREERGUIDE.COM

NEW! DOWNLOAD AND PLAY THE LATEST STUDENT GAMES!

GET YOUR GAME ON

EVERYTHING YOU NEED TO KNOW TO GET INTO THE GAME INDUSTRY!

- News and features for students and educators
- Getting Started section – an invaluable how-to guide
- Faculty & Student Research – Salary Survey
- Directory of Game Schools
- Message Boards

Want to get your game exposure?
Gamecareerguide.com student games are made available by participating schools only.
Visit the website for details on how your school can register today!

www.gamecareerguide.com

The 1st ACM SIGGRAPH Conference and Exhibition in Asia

CONFERENCE 10-13 DECEMBER 2008 EXHIBITION 11-13 DECEMBER 2008

COMING SOON. SIGGRAPH ASIA PREMIERES IN SINGAPORE IN DECEMBER 2008!

Be there and become part of the world's fastest growing digital-media region.

EXHIBITION

Reserve your exhibit space now for this year's best opportunity in Asia to reconnect with customers, meet new buyers, recruit talent, and network with all the leading experts who are growing Asia's vibrant computer graphics community.

CONFERENCE

Experience the highest quality, multi-sensory computer graphics and interactive techniques presented by leading industry luminaries from around the world in a variety of SIGGRAPH programs:

Art Gallery
Computer Animation Festival
Courses
Educators Programme
Emerging Technologies
Sketches & Posters
Technical Papers

Online registration starts from 14th July

www.siggraph.org/asia2008

For more information:
SIGGRAPH Asia 2008
Koelnmesse Pte Ltd
+65.6500.6700
+65.6296.2771 fax
asia2008@siggraph.org

SIGGRAPHASIA2008
NEW HORIZONS

HELD IN:

UNIQUELY Singapore
visitsingapore.com

Join the PLAYSTATION Team!

Sony Computer Entertainment America, Inc. (SCEA) markets the PlayStation family of products and develops, publishes, markets, and distributes software for the **PlayStation Portable®**, **PlayStation®2** and **PlayStation®3** computer entertainment systems, for the North American market.

We are currently looking for talented and experienced game development professionals for the following areas:

Foster City • California

Software Engineer
Product Marketing
Technical Manager
Programmer
Producer

San Diego • California

Web Developer
Software Engineer
Animator
Technical Support

Santa Monica • California

Designer

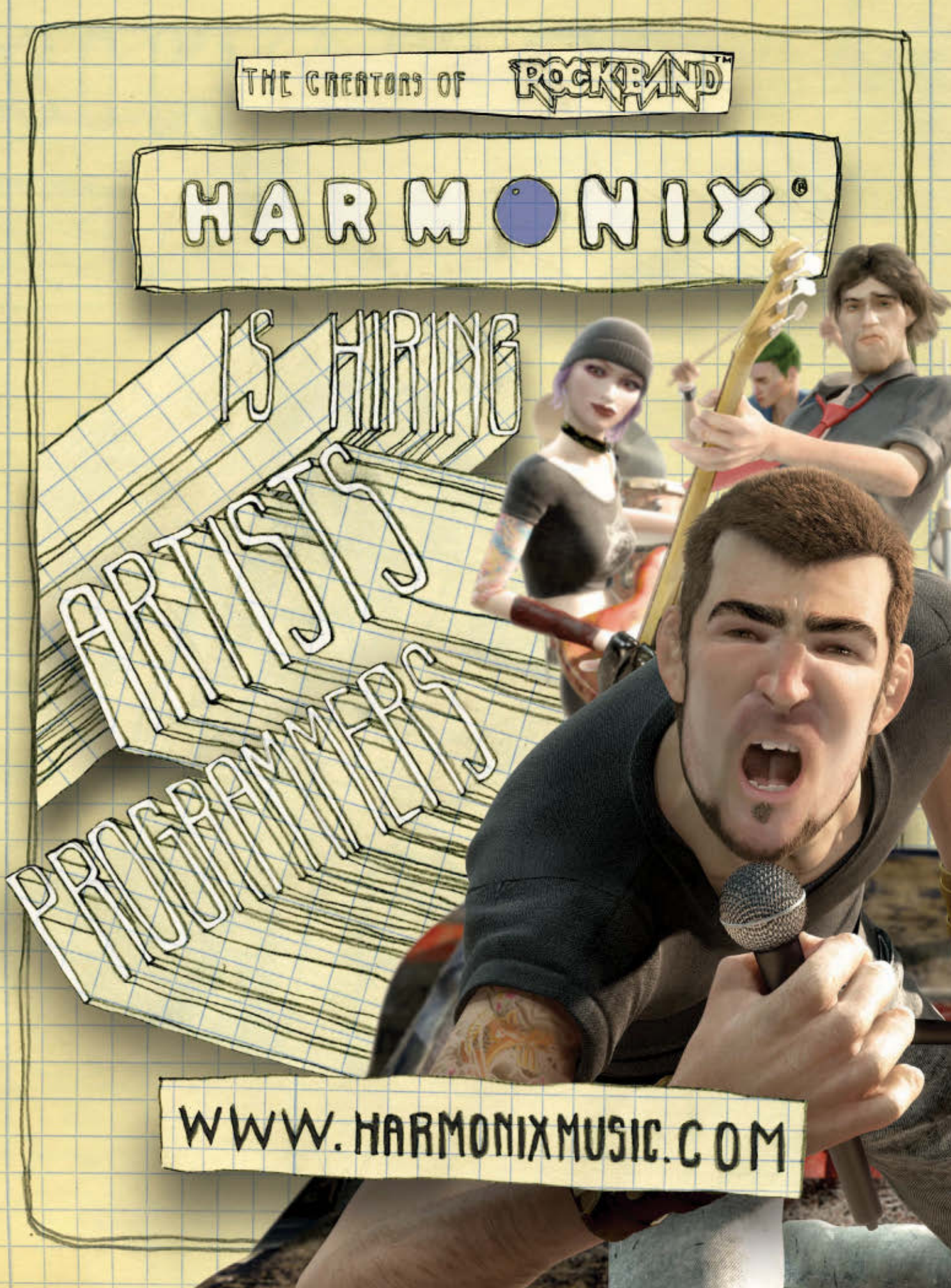
At **Sony Computer Entertainment America, Inc.**, we offer our employees exceptional benefits, an open and creative working environment, and the opportunity to be apart of a team that performs at **1.8 trillion calculations per second.**

For additional information, ability to send your resume, add updates, and establish a new business relationship, please visit our website at:

www.us.playstation.com/jobs

PLAYSTATION 3
PlayStation 2
PSP
PlayStation Portable

SONY COMPUTER ENTERTAINMENT


THE CREATORS OF ROCK BAND™
HARMONIX®
IS HIRING
ARTISTS
PROGRAMMERS
WWW.HARMONIXMUSIC.COM

MOM SAID PLAYING VIDEO GAMES WOULD NEVER PAY OFF.

START PROVING HER WRONG TODAY.

THQ

Check out our career opportunities now at: careers.thq.com

rainbow STUDIOS · sandblast · HEAVY IRON STUDIOS · THQ studio australia · BIG HUGE GAMES · relic ENTERTAINMENT

KAOS STUDIOS · volition inc · bluetongue · juice games · VIGIL GAMES · locomotive games

Helixe · XDG · mass • media · ValuSoft · PARADIGM ENTERTAINMENT · INCINERATOR STUDIOS

THQ WIRELESS

THQ

www.thq.com

WHO the DEVIL is LOU?

LOU is...

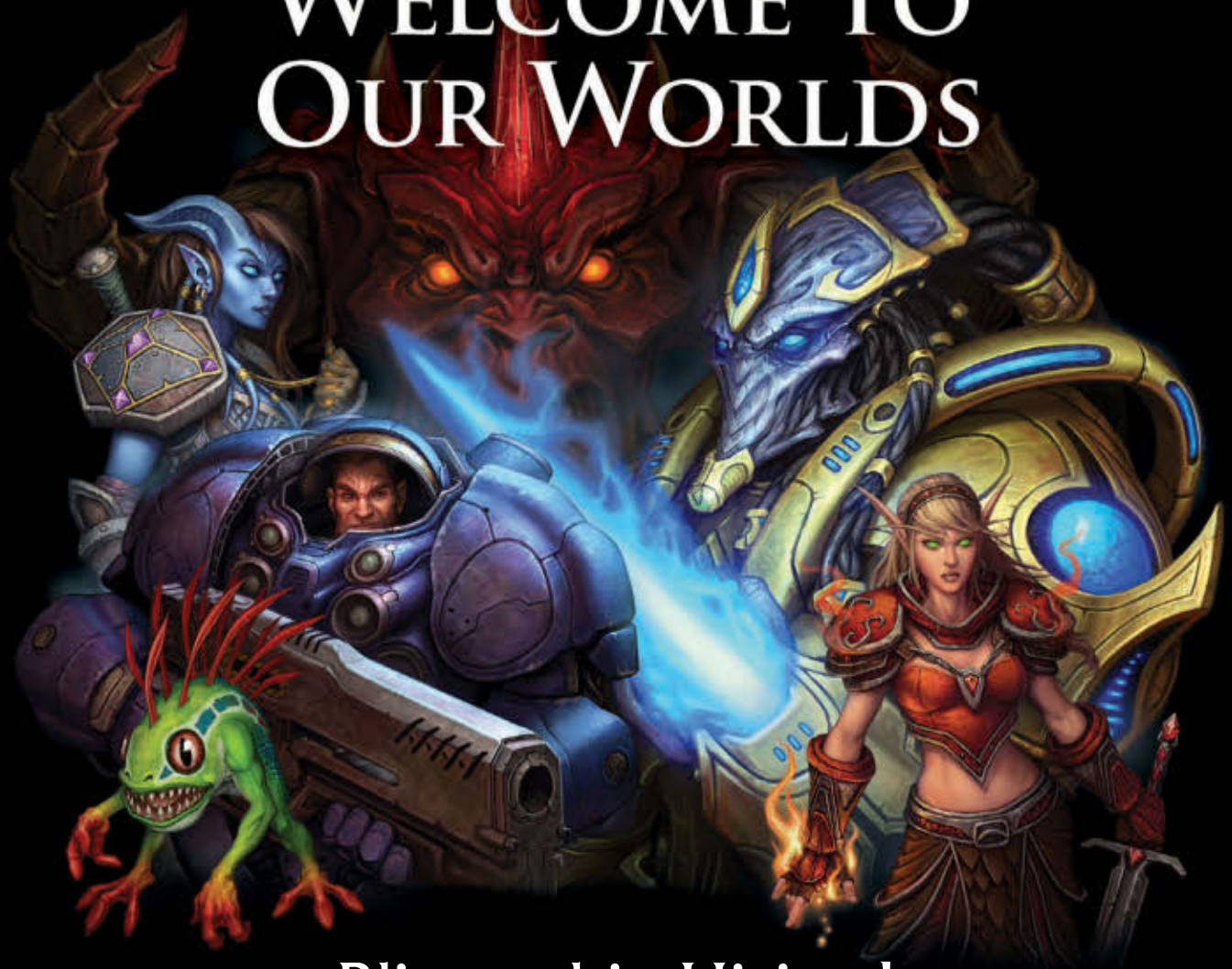
a concept artist, 3D modeler/texture artist, animator, lighter, tech artist, producer, and programmer! These positions and more at www.neversoft.com!

NEVERSOFT

LOU won't be at SIGGRAPH but Neversoft will... Activision Booth #217, Main Pavilion.

BLIZZARD ENTERTAINMENT

WELCOME TO OUR WORLDS



Blizzard is Hiring!

We are actively recruiting across all disciplines for locations worldwide.

For a full list of openings, please visit us online at

www.blizzard.com/jobs

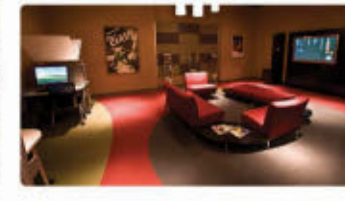
Florida Interactive Entertainment Academy

OUR MASTER'S DEGREE HAS ONE THING ON ITS MIND.

YOUR NEXT JOB. At FIEA everything from our industry-based curriculum to our new MOCAP studio is geared toward teaching what you need to know to become a successful video game producer, programmer or artist.

Earn an accredited Master's degree in 16 months while being mentored by industry veterans who have shipped more than 40 games and films. So tip your cap to our 95% placement rate and learn more at www.fiea.ucf.edu.

FIEA
active. enter. learning.



FIEA is a part of the University of Central Florida, a leader in modeling and simulation, optics and photonics, computer science and many other disciplines.

WE SPEAK G33K

Geeks transform into Alpha-Geeks by becoming fluent in:

LEARN:
DIGITAL ANIMATION
DIGITAL ART AND DESIGN
DIGITAL VIDEO
GAME DESIGN
ARTIFICIAL LIFE PROGRAMMING
COMPUTER FORENSICS

GAME PROGRAMMING
NETWORK ENGINEERING
NETWORK SECURITY
SOFTWARE ENGINEERING
WEB ARCHITECTURE
ROBOTICS & EMBEDDED SYSTEMS

University of Advancing Technology
UAT
Learn. Experience. Innovate.

www.uat.edu > 877.UAT.GEEK
877.828.4335

Make More Enemies

Game Design at Vancouver Film School

The Leader.
Dave Warfield, 15-year veteran game designer for over two dozen titles.

The Know-How.
In just one year, you'll learn every aspect of game design. Your portfolio project is a playable, industry-ready video game.

The Results.
Our graduates work at top game companies including Backbone Entertainment, BioWare, EA Black Box, Next Level Games, and Radical Entertainment.

vfs.com/enemies

VFS™

VFS student work by Jason Martin

ADVERTISER INDEX

Game Developer (ISSN 1073-922X) is published monthly by United Business Media LLC, 600 Harrison St., 6th Fl., San Francisco, CA 94107, (415) 947-6000. Please direct advertising and editorial inquiries to this address. Canadian Registered for GST as United Business Media 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 *Game Developer*, P.O. Box 1274, Skokie, IL 60076-8274. For back issues write to *Game Developer*, 4601 W. 6th St. Suite B, Lawrence, KS 66049. 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 *Game Developer* on any correspondence.

ARRESTED DEVELOPMENT

CLIMBING THE LICENSE LADDER

JANUARY 3RD: **MET WITH THE AGENT** again today and reminded him of our desire to find cult classics that aren't too pricey but fondly remembered by a certain audience. Not sure if they totally understood us, because the biggest news they had was that they'd expanded their sitcom deal to include both *Mr. Belvedere* and *Out of this World*. To be fair, crossover titles do seem all the rage right now, so maybe that could yield something interesting— especially if they also threw *Small Wonder* into the mix. I'll get back to them next week after going over it with my creative team.

January 5th: Wow, so I guess not everyone has the same positive memories of 80's sitcoms that I do ... what bunch of jerks. Anyway, it seems like we're still searching! I'm sure we'll turn up something eventually.

January 19th: After telling my team they should brainstorm some licenses that might be both cool and attainable, we got into a big argument about *GoBots*. Just because there is an extensive Wikipedia article on them does NOT necessarily mean they are "hella" popular. Where do they get these ridiculous ideas? I'm going to clamp down on this, we shouldn't waste any more time talking about it.

February 10th: Asked for more "modern" options from the agent, since we are sick of being stuck in the 80's and in response was asked how I feel about *The Six Million Dollar Man*. We are going back in time!! I don't get these people!

February 12th: I just love moving backwards. This time it's stuff from the 70's nobody has ever even heard of ... *Tubby the Tuba*? Give me a break, dude. We're professionals here, not bottom-feeding, cut-rate hacks. We don't pick up junk like that just because it's work. Unlike some developers out there, we have thing called pride and self-respect. We are seriously considering firing this useless agent and just going it alone. It couldn't possibly be any worse than this.

March 2nd: Had another big argument with my team today. *GoBots* would NOT make a good MMO ... first of all, who cares about them? Secondly, looking at Wikipedia, it seems there is basically zero complexity in the story, and close to nothing in the way of supporting information. Isn't that an important part of building a large and complex virtual world? All of this is a moot point anyway since we just don't have the resources to support such a huge undertaking in any case, with or without the *GoBots* license. Our great ambitions will just have to wait until we're more established.

March 23rd: Vintage *GoBots* toys seem to do decently enough on eBay, I may acquire some for research purposes. They are kinda cool ...

April 17th: The designers have come up with something they call the "dual-class" system for a potential *GoBots* MMO. 12 of each kinds of class would mean up to 144 possible Bot/Transform combinations. Despite myself, I'm starting to get excited about the idea of this ... what if we really do get the license and the funding? Man, that would be cool.

July 11th: Well. It looks like we're in a bit of a tight spot here. We're running low on operating cash, and I suppose it was a bit foolhardy of me to pin my hopes on the whole *GoBots* thing. Much as I hate to admit it, the *Tubby the Tuba* deal is looking better all the time. It's work, for one thing. And I'm sure we could kick it out super easily, just to keep us afloat for the time being. I've been telling the team that if we took the job, it would be a kind of a stopgap measure ... in which case, the next project would be the one to really look forward to. We all have to start somewhere, right?

July 19th – 9:00 AM: Very exciting news today! We are close to a deal with an author of a big series of fantasy books. I hadn't heard of them before, but was told they have a pretty significant cult following. This is great. It could give us a huge fictional base to draw from to create a rich world. The designers were already talking about possibilities for an MMO, and for once, I'm actually agreeing with them. We'd really have to staff up to make that a reality, of course. But our huge dreams might just not be out of reach this time: the license owner has stated he can actually bring some money to the table on his own! Finally ... after endless months of wallowing around in the dregs of the licensing world. There's a fantastic opportunity here. This could be the real thing! I'm about to read the first book now. I guess it's called "*Gor*."

July 19th – 4:00 PM: Abort! Abort! These people are the furries of the fantasy world!

December 1st: After a long, hard crunch, *Tubby the Tuba* finally hit the shelves. I can't lie ... it was definitely painful trying to get this thing out the door. But, when all is said and done I think we all agree it was worth it. Our studio is on the map now, our publisher seems pleased, and we've just signed a deal for the sequel.

MATTHEW WASTELAND *is a pseudonymous game developer who has a fairly common first name. Email him at* ***mwasteland@gdmag.com****.*

gamedeveloper FRONTLINE FINALIST '07

"ReplayDIRECTOR rocks. I doubt we'd have found it otherwise. It turned out to be an occasional array overwrite that would cause random memory corruption…"

***Meilin Wong**, Developer, Crystal Dynamics*

BUGS. PETRIFIED.

RECORD. REPLAY. FIXED.

ReplayDIRECTOR™ gives you **Deep Recording**. This is much **more than just video capture**. Replay records every line of code that you execute and makes certain that it will Replay with the same path of execution through your code. Every time. Instantly Replay any bug you can find. Seriously.

DEEP RECORDING. NO SOURCE MODS.

download today at www.replaysolutions.com
email us at info@replaysolutions.com

REPLAY SOLUTIONS
accelerating you to market

1600 Seaport Blvd., Suite 310, Redwood City, CA, 94063 - Tel: 650-472-2208 Fax: 650-240-0403

EA ELECTRONIC ARTS™ CRYSTAL DYNAMICS eidos XBOX 360 nVIDIA. XBOX

BINK

BINK HAS SHIPPED IN 2,900 GAMES FOR A REASON.

IT'S THE STANDARD FOR VIDEO IN GAMES - UP TO 4 TIMES FASTER AND USING AS MUCH AS 18 MB LESS MEMORY THAN OTHER CODECS.

IT HAS A BUILT-IN AUDIO CODEC WITH MULTITRACK CAPABILITY, SUPPORTS EVERY PLATFORM, AND CAN HANDLE AN ARBITRARY NUMBER OF VIDEO CHANNELS, LETTING YOU ADD Z-DEPTH, NORMALS, OR UV PER-PIXEL DATA, AND MORE!

FLAWLESS HD VIDEO WITH 5.1 AUDIO ON WINDOWS, XBOX AND PS3!

RAD GAME TOOLS

Now playing on the Wii, PSP, PS3, and Xbox 360

425-893-4300 www.radgametools.com

GRANNY 3D BINK VIDEO PIXOMATIC MILES