Mobile Media APIs
ETF 2007

Neil Trevett
Vice President Embedded Content, NVIDIA
President, Khronos Group
Speakers

• Neil Trevett, VP Embedded Content - NVIDIA, Khronos President
  - Khronos, OpenGL ES, OpenVG, OpenKODE

• Shiv Ramamurthi, Software Technology Manager - TI
  - OpenMAX and OpenSL ES

• Oliver Baltuch, VP Marketing and Sales NA – Futuremark
  - Ecosystem and benchmarking
Pervasive Mobile Computing

• Handsets are becoming personal computing platform - not “just” phones
  - A real computer in your hand – mobility, connectedness and numerous sensors

• Sophisticated media processing will be central to this handheld revolution
  - Graphics and media will become as pervasive as it is on the PC
Media APIs Enable Market Growth

- Khronos APIs typically define interface between silicon and software
  - Enabling both communities - everyone wins

- ISVs see reduced fragmentation across multiple platforms
  - More software can reach market faster at a better level of functionality and quality

- Hardware vendors can accelerate many applications
  - Adding value to their platform
Khronos - Open Media Standards

Open Membership
Any company is welcome
Funded by membership dues - $6K / year

Open Standards
Publicly available on web-site
Royalty-free

Open Standards for Media Authoring and Acceleration

Khronos typically develops “Foundation-Level” APIs
High-performance “Close-to-the-metal” access to hardware acceleration
Good foundation for higher-level engines and middleware
Over 100 companies creating media authoring and acceleration standards
Khronos Dynamic Media Ecosystem

Cross-platform graphics authoring/acceleration Ecosystem

Cross platform 2D/3D

Safety Critical 2D/3D

Dynamic Media Authoring Standards

Dynamic Media Authoring

3D Authoring

Embedded Media Acceleration APIs

“DirectX-like” set of native APIs
Includes mixed media acceleration and OS portability APIs

OpenGL
OpenGL SC

Khronos Dynamic Media Ecosystem

Cross-platform graphics authoring/acceleration Ecosystem

Cross platform 2D/3D

Safety Critical 2D/3D

Dynamic Media Authoring Standards

Dynamic Media Authoring

3D Authoring

Embedded Media Acceleration APIs

“DirectX-like” set of native APIs
Includes mixed media acceleration and OS portability APIs

OpenGL
OpenGL SC

Khronos Dynamic Media Ecosystem

Cross-platform graphics authoring/acceleration Ecosystem

Cross platform 2D/3D

Safety Critical 2D/3D

Dynamic Media Authoring Standards

Dynamic Media Authoring

3D Authoring

Embedded Media Acceleration APIs

“DirectX-like” set of native APIs
Includes mixed media acceleration and OS portability APIs

OpenGL
OpenGL SC
Adoption of Embedded Khronos APIs

We Are Here
- OpenGL ES 1.1 is widespread
- OpenVG is in rapid adoption
- OpenMAX is being implemented
- OpenSL ES is being designed

Market adoption in media-accelerated handsets

100%

Mid-2004
OpenGL ES 1.1 Spec release

Mid-2005
OpenVG 1.0 Spec release

Beginning-2006
OpenMax IL 1.0 Spec release

Mid-2007
OpenSL ES 1.0 Spec release

End-2007

KHRONOS GROUP
© Copyright Khronos Group, 2007 - Page 8
Advantages of Media Acceleration

Faster Performance at Higher Quality
Hardware delivers at least 10 times the performance of software – even on low-cost systems with low-end CPUs

POWER EFFICIENCY

Less Power
Hardware accelerators exploit parallelism in a media pipeline to give a x10 increase in power efficiency over software

Better User Experience
Smaller screens need more advanced graphics processing per pixel
Penetration of Native Media APIs

Millions of units
Jon Peddie Research
Handheld Multimedia Devices report

- MM phones
- Non 3D MM phones
- Native API based phones
OpenGL ES API Standard

• Small-footprint subset of OpenGL
  - Powerful, low-level API with full functionality for 3D applications and games

• Standard in several mobile operating systems
  - E.g. Brew, Symbian, Linux

• Handsets are now shipping in volume
  - Over 50M OpenGL ES 1.1 silicon devices shipped
OpenGL ES – Central to Mobile 3D

- Brings advanced 2D/3D graphics to a wide range of OS platforms
- "Close to the hardware" API provides portability AND flexibility
- Usable directly by applications
- Usable by higher-level graphics libraries
- JSR 239: Defining official Java Bindings to OpenGL ES

Java Applications

C/C++ Applications

Scenegraph APIs
M3G (JSR 184)
Games
Engines
Middleware
Libraries

Hardware OpenGL ES Engines

Software OpenGL ES Engines

C/C++ Applications

Games
Engines
Middleware
Libraries

Scenegraph APIs
M3G (JSR 184)

Java Applications

C/C++ Applications

Software OpenGL ES Engines

Hardware OpenGL ES Engines

"Close to the hardware" API provides portability AND flexibility

Usable directly by applications

Usable by higher-level graphics libraries

JSR 239: Defining official Java Bindings to OpenGL ES

Brings advanced 2D/3D graphics to a wide range of OS platforms

© Copyright Khronos Group, 2007 - Page 12
OpenGL ES – Shader Revolution

- OpenGL ES 1.1 is pervasive today
  - High-performance fixed function 3D graphics
  - Delivered on devices from mobile handsets including iPhone to Playstation 3

- The Shader Revolution comes to mobile with OpenGL ES 2.0
  - Programmable GPUs have revolutionized desktop graphics
  - Shader programs enable amazing graphical effects ...
  - ... coming to mobile handsets in 2008

Reflections, lighting effects
Parallax mapping
OpenGL ES – Two Track Standard

• Two tracks - manage mobile graphics through programmable transition
  - With maximized portability and minimized platform costs

• OpenGL ES 2.0 ruthlessly eliminates redundancy – just like 1.X
  - Deprecates all fixed functionality that can be replaced by shaders
  - Significant reduction in engine cost and driver complexity

• Platforms can ship either or both 1.X and 2.X libraries
  - Cheaper, more flexible than one large driver with both fixed and programmable functions
  - With full backwards compatibility maintained in each track

• OpenGL ES 2.X does NOT replace OpenGL ES 1.X
  - Will always need lowest cost, non-programmable hardware for certain high-volume devices

OpenGL ES 1.X – Fixed Function Acceleration
OpenGL ES 1.1
- For software and fixed functionality hardware
- All 1.X specifications are backwards compatible

OpenGL ES 2.X – Programmable Acceleration
OpenGL ES 2.0
- Vertex & pixel shaders through GLSL ES shading language
- All 2.X specifications will be backwards compatible
OpenGL ES Roadmap

- Stability and reducing fragmentation is currently the key concern
  - More important than new functionality in the current phase of market development
  - Industry is still absorbing and implementing current OpenGL ES specifications

 OpenGL ES 2.0 specification release March 2007
 OpenGL ES 2.0 accelerated products begin to ship
 OpenGL ES 2.1 IF NEEDED to address market needs or 2.0 shortcomings
 OpenGL ES Next Gen New functionality to meet PROVEN market needs
 OpenGL ES 2.x products ship in volume
 OpenGL ES 1.1 with hardware acceleration remains the “Sweet Spot” at least through 2008
 OpenGL ES 1.1 will continue to be used in lower-cost devices

Shipping Products 2007 2008 2009-10?
Announcing glFX!

- An “effects framework” API for OpenGL and OpenGL ES
  - Similar to CgFX for Cg from NVIDIA

- An visual effect is made up of shaders, passes and state
  - Used in applications at runtime to control the visual appearance of objects

- Work based around COLLADA FX file format
  - COLLADA, OpenGL and OpenGL ES working groups – working together

DCC tools create and export effects via Collada FX

Applications use the glFX API to create and apply effects at runtime

glFX can be layered on top of the graphics driver, or embedded for efficiency
glFX in Game Development

Autodesk®

COLLADA
File format for 3D asset interchange – widely adopted by DCC tools vendors and Google, Adobe, Epic etc.

COLLADA FX Effects
Textures, shader programs, geometry, control and pass information

Optionally create data representations for delivery to target devices

Application traverses scene data, uses the glFX Runtime API to extract effects information to setup the rendering pipeline

Application

Optional Processing for Delivery

glFX Runtime API

OpenGL 2.0 / OpenGL ES 2.0

© Copyright Khronos Group, 2007 - Page 17
OpenVG - Accelerated Vector Graphics

• Vector graphics - basis of popular formats such as Flash and SVG
  - User interfaces, screen savers, 2D Games, mapping and GPS, E-book readers

• Scalable Bezier curves
  - Can be scaled and positioned at full quality - not polygon based

• But 2D vector graphics historically run un-accelerated!
  - Not effective on low-powered handset CPUs

• OpenVG ACCELERATES Flash and SVG for the first time
  - Better quality, more interactivity - at less power
The OpenVG Pipeline

- OpenVG defines a hardware pipeline for paths and images
  - Path Definition
    - Lines, quads, arcs, cubics
  - Stroking
    - Line width, joins & caps, dashing, etc.
  - Transformation
    - 2x3 and 3x3 transformations
  - Rasterization
    - Conversion to pixels
    - Image Filters
  - Clipping & Masking
    - Scissor rectangles and pixel masks
  - Paint Generation
    - Flat color, gradient, or pattern paint
  - Blending
    - Sophisticated blend modes
**OpenVG Milestones**

- **OpenVG is an open, royalty-free standard**
  - OpenVG 1.0 was released mid 2005

- **OpenVG 1.0 conformance tests released 1Q07**
  - Immediately available – following standard Khronos process

- **OpenVG 1.0 sample open source sample implementation released 1Q07**
  - Reference implementation coded by NVIDIA now released under MIT license as sample
  - Coded for accuracy not performance – runs on a PC
  - Excellent reference for implementers and developers

- **OpenVG 1.1 is being defined now**
  - Enhanced text rendering – in-driver glyph positioning
  - 100% seamless acceleration for Flash Lite
OpenVG Deployment

- OpenVG 1.0 beginning to be widely deployed
  - A range of software and hardware implementations are coming to market

Vector Graphics Performance

- Software Implementations
  - X1

- Layered Implementations over OpenGL ES hardware
  - X3-7

- Dedicated OpenVG Cores – low power
  - X20

© Copyright Khronos Group, 2007 - Page 21
Innovative Mixed Media Applications

• Breakthrough mobile games and applications will not treat mobile devices as small consoles or PCs
• They will take advantage of the unique capabilities of mobile devices:
  • Use of handset sensors
    - Cameras for video and images, sound
  • Mobility
    - With GPS location awareness
  • Connectedness
    - Unique information flow and social interactions
• An advanced media stack enables innovative applications that mix many kinds of media acceleration
  
A GPS phone processes OpenMAX video to recognize buildings and landmarks
Issue: Fragmentation

- Every handset is unique from the programmers perspective
  - Differences in operating system functions, Java implementations and media functionality

- Severe platform fragmentation exists today
  - ISVs need to port to and support 100s (even 1000s) of source variants of each title

---

**Global Handset Sales by Device Type**

- Smartphones
- Wireless PDAs
- Feature phones
- Browserphones / Basic Terminals

Source: Strategy Analytics, Oct. 2005

- Symbian 7, 8, 9, UIQ, S60
- PocketPC / Windows Mobile / WinCE
- Many Linux variants (and growing)
- Java MIDP-1, MIDP-2, JSR fragmentation

- Numerous RTOS with no consistently defined OS abstraction or media portability – Nucleus, Synergy etc.
Issue: Low Performance

- Java doesn’t provide optimal media performance
  - Native acceleration can provide X5 higher performance for 3D graphics

<table>
<thead>
<tr>
<th>Measured by Kishonti</th>
<th>Java (fps)</th>
<th>OpenGL ES (fps)</th>
<th>Native Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nokia N93</td>
<td>18.6</td>
<td>77.8</td>
<td>x4.2</td>
</tr>
<tr>
<td>Sony Ericsson M600</td>
<td>6.0</td>
<td>30.6</td>
<td>X5.1</td>
</tr>
</tbody>
</table>

GLBenchmark – direct port of identical Java benchmark
Solution: Native Media API Set

Fragmentation

Need Consistent APIs on Every Handset
Reliably available functionality

Lack of Performance

C Native Programming Environment
Familiarity for developers
Eases ports from PCs, handhelds and consoles

“Native Media Acceleration APIs”
A cross-platform set of native APIs for media application development

OpenKODE
OpenKODE

Its like DirectX™ for mobile phones!
Except its an open standard, cross-platform, royalty-free and streamlined for handheld devices

OpenKODE is a set of C-native APIs for handheld games and media applications

2D, 3D, video and audio media types are all seamlessly accelerated by OpenKODE

OpenKODE minimizes source changes when porting games and applications from phone to phone
Native Fragmentation

Software Platform (e.g. Brew, Symbian UIQ, S60, WIPI)

Native Applications

User Interface
3D Game Engines
Flash/SVG Players
TV/Video/Audio Players

Media Accelerator Silicon
GPU / DSPs / CPUs

Kernel Operating System
(e.g. Rex, Symbian OS, Windows Mobile, Linux, Nucleus, Synergy)

CPU

JAR File
Carrier OTA Provisioning
Java JVM
Java Applications
Bindings

Different OS APIs on every handset
Proprietary media APIs, single-threaded, not tested for mixed-media operation

Handheld devices increasingly need advanced graphics and media acceleration for user interfaces, 3D gaming, Flash, TV...

Severe API fragmentation to access OS resources and media acceleration = HUNDREDS of source variants per game

No Native OTA Deployment
The OpenKODE Solution

Software Platform (e.g. Brew, Symbian UIQ, S60, WIPI)

Native Applications

User Interface 3D Game Engines Flash/SVG Players TV/Video/Audio Players

Java JVM

Java Applications

Bindings

System Abstraction (OpenKODE Core)

Media APIs

Trans-API communication (EGL)

Media Accelerator Silicon

GPU / DSPs / CPUs

Kernel Operating System
(e.g. Rex, Symbian OS, Windows Mobile, Linux, Nucleus, Synergy)

CPU

OpenKODE

= Selected Khronos media APIs for state-of-the-art media acceleration + Trans-API data and event coordination – WITH full trans-API Conformance Tests (EGL) + System Abstraction APIs for portable access to operating system resources, input devices and displays (OpenKODE Core) = A coherent, cross-platform API set for NATIVE portable media applications
OpenKODE Participation

- 40 companies participating in OpenKODE working group
  - Core participants include Acrodea, Aplix, ARM, Ericsson, Freescale, Futuremark, Ideaworks 3D, Intel, Nokia, NVIDIA, Samsung, SUN, Symbian, Tao, TI
- Working for industry adoption on many platforms
  - Symbian, WIPI, Linux, Windows, Brew AND RTOS
  - Any platform can use royalty-free, open-standard, vendor-neutral, media acceleration API layer
  - Incorporates significant design expertise from leading industry media companies
OpenKODE provides foundation-level acceleration for advanced user interfaces and media applications that mix multiple media types.
Leveraging and Accelerating Java

Existing provisioning, billing and security systems can be used to OTA provision native OpenKODE applications.

Native applications will need certification – as Brew does today.

CHAPI (Content Handler API) - Java standard to register handlers for media types. A native API enables mixing Java and native applications and services.
OpenKODE Milestones

- **OpenKODE 1.0 Provisional released at 3GSM 2007**
  - Twelve months from kick-off meeting to specification on web-site
  - Encouraging developer feedback before spec finalization mid-2007

- **Full conformance tests will be released in 2Q07**
  - Written by Futuremark – tests OpenKODE Core AND trans-API operation

---

**Call for participation and contributions**

**Scope Agreed June F2F**

**OpenKODE 1.0 Provisional Release**

**OpenKODE 1.0 Final Release Target**

**1Q06 2Q06 3Q06 4Q06 1Q07 2Q07 3Q07**

**Kickoff F2F meeting**

**Strawman specs complete – November F2F**

**Conformance Tests to be Released**

---

If you are a developer – download the spec and give us your feedback!
Spec at [www.khronos.org/openkode/](http://www.khronos.org/openkode/)
Feedback forums at [www.khronos.org/message_boards/](http://www.khronos.org/message_boards/)
**OpenKODE Roll-Out**

- Simple build out of existing OpenGL ES adoption
  - Expecting rapid industry momentum

- OpenKODE 1.0 allows media API selection
  - To enable widespread early adoption

- OpenKODE does NOT preclude other media APIs
  - OEMs can pace their own transition from proprietary APIs
  - E.g. can ship alongside DirectShow

---

- Full DirectX functional equivalence. Robust trans-API functionality and testing

- OpenGL ES is everywhere today

- Now

- OpenKODE Core API has no silicon dependency and takes <50KB. OpenKODE provides enhanced 2D/3D mixed mode operation

---

© Copyright Khronos Group, 2007 - Page 33
Implementation Announcements

OpenKODE 1.0 announced in Feb07
Momentum is already building quickly

Futuremark developing Conformance Tests for OpenKODE. Graphic Remedy developing OpenKODE profiling tools

OpenKODE Core can “drop-in” alongside OpenKODE media stacks. Typically middleware solutions provide native OTA provisioning over Java networks – sometimes with portable binary capability

Announced OpenKODE media stacks with OpenGL ES and increasingly OpenVG

Tools

OpenKODE Core and Middleware

Silicon Acceleration
OpenKODE: Acceleration Foundation

- **Advanced UI – OpenVG**
  - OpenGL ES for special effects

- **Accelerated Java Applications**
  - OpenGL ES – 184/239, OpenVG - 287

- **Augmented Reality – OpenMAX AL**
  - Video processed & rotoscoped with OpenGL ES 3D

- **Advanced Navigation - OpenVG**
  - 3D terrain/satellite data using OpenGL ES

- **Video Telephony – OpenMAX AL**
  - OpenGL ES lip-synch 3D avatar

- **PVR Application – OpenMAX AL**
  - OpenVG-based UI and subtitles

- **Music Visualizer - OpenSL ES**
  - OpenGL ES 3D synchronized to music beat

- **3D Game – OpenGL ES**
  - OpenSL ES positional audio & OpenVG HUD/menus

- **3D TV Channel Selector – OpenMAX AL**
  - OpenGL ES 3D multi-channel visualization

OpenKODE creates the opportunity to create new applications and user interfaces that use multiple media types.
OpenKODE Native Ecosystem

Carriers

Compelling media applications increase revenue through existing provisioning, billing, DRM certification infrastructure

Native OTA provisioning uses existing certification, security, billing, lifecycle management infrastructure

Open standards to simply specify a native, cross-platform media API stack

Proven media stack architecture based on multi-vendor open-standards - reduces risk

Cross-platform native APIs reduce source fragmentation and increase performance

Handset OEMs

Native, cross-platform media API stack

Content Providers
Industry Call to Action

- We now have a forward-looking, multi-threaded media stack architecture
  - Open and royalty-free to encourage industry innovation
- Foundation for new-generation interfaces and mixed-media applications
  - Driving new markets and revenue opportunities
- OpenKODE is a open-standard, multi-vendor native media-stack
  - Helping to solve industry fragmentation and performance issues

- If you are an handset OEM.. Embed an OpenKODE 1.0 media stack!
- If you are a carrier.. Specify OpenKODE 1.0 for your native applications!
- If you are a developer.. Demand OpenKODE 1.0 for development!

OpenKODE is a significant opportunity for the handheld industry to help evolve handheld devices into the most pervasive media-capable computing devices the world has yet seen
If you are a developer – download the spec and give us your feedback!
Specification at [www.khronos.org/openkode/](http://www.khronos.org/openkode/)
Feedback forums at [www.khronos.org/message_boards/](http://www.khronos.org/message_boards/)

These slides and Khronos membership details at [www.khronos.org](http://www.khronos.org)