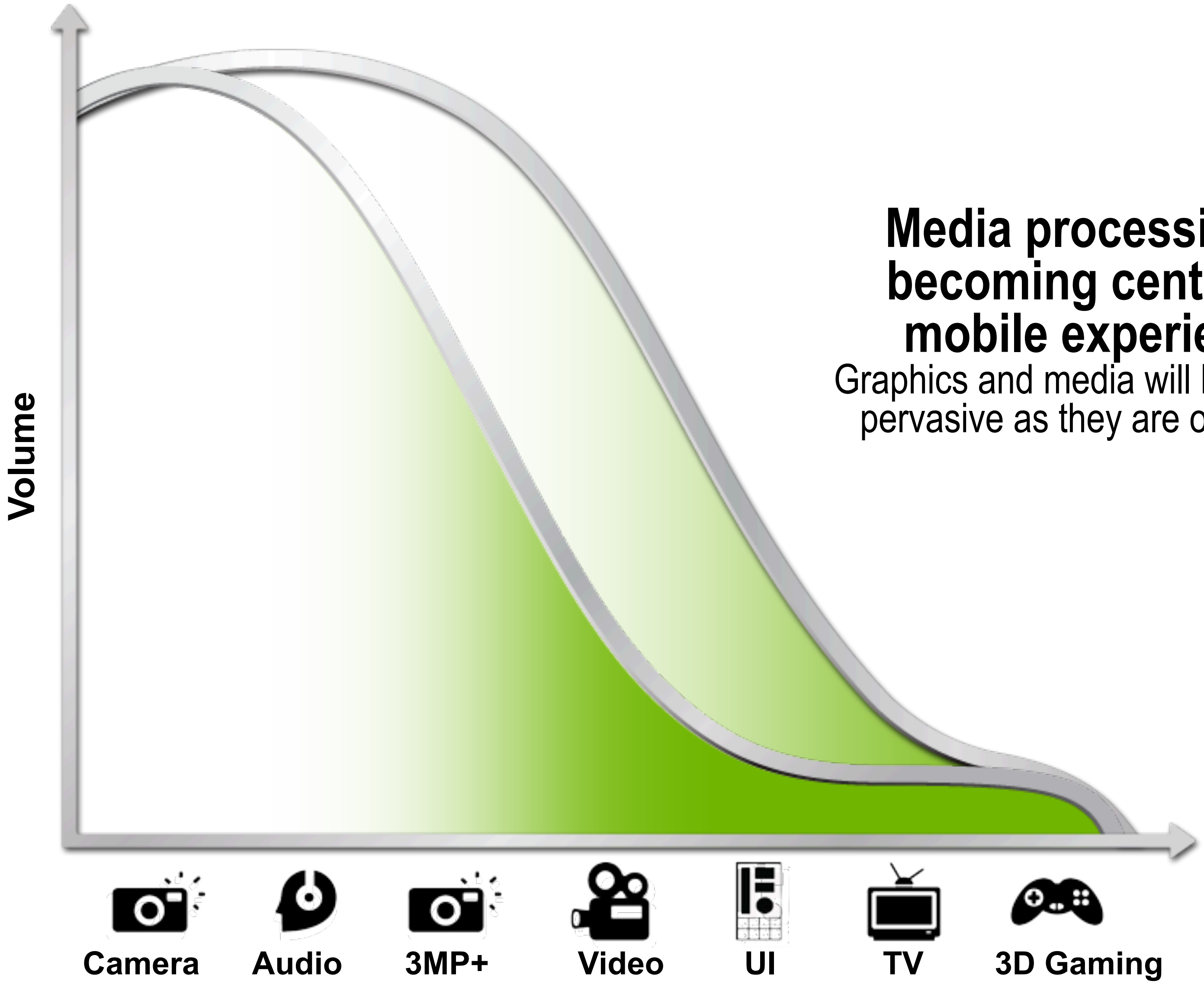
KHRONOS™
GROUP

Khronos Mobile Graphics and Media Ecosystem

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

Mobile Platform Media Evolution



Media processing is becoming central to mobile experience

Graphics and media will become as pervasive as they are on the PC

Mobile Media Fragmentation

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

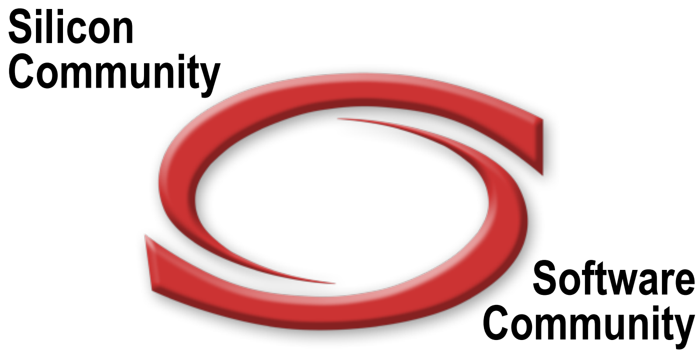
Symbian 7, 8, 9, UIQ, S60
PocketPC / Windows Mobile / WinCE
Linux variants – Android, Limo
Java MIDP-1, MIDP-2, JSR fragmentation
RTOS – Nucleus, Synergy
Brew, WIPI, MAC OS X

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

Who is the Khronos Group?

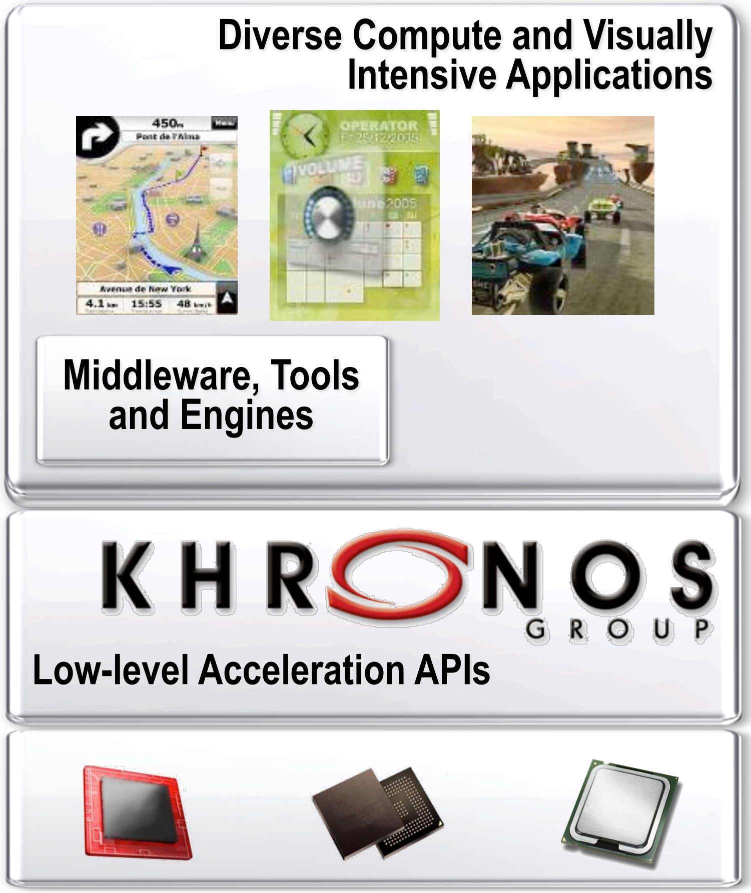
- **Industry consortium to create open API standards**
  - By the industry, for the industry - founded eight years ago - any company is welcome to join
- **Focused on enabling software to efficiently leverage silicon acceleration**
  - Reducing software fragmentation and exposing cutting-edge silicon functionality
- **Strong commercial focus – we are in this to make money**
  - Aim is to enable members and the wider industry to develop and grow market opportunities
- **Commitment to royalty-free standards**
  - Making money from enabled products – not from the standards themselves

Enabling strategic business development through open standards that enable silicon and software to interoperate



What Standards Does Khronos Create?

- “Foundation-Level” acceleration APIs
  - Needed on every platform to support an ecosystem of middleware and applications
- Low-level access to processor silicon
  - Designed with strong silicon vendor participation
- Cross-vendor software portability
  - API abstractions just high enough to hide implementation specifics
- Established focus on graphics, media and parallel compute acceleration
  - 3D, vector 2D, video, imaging, audio, heterogenous parallel programming APIs…



Diverse Compute and Visually Intensive Applications

Middleware, Tools and Engines

Low-level Acceleration APIs

Khronos APIs create the foundation of an ecosystem that enable applications to be PORTABLE and ACCELERATED on diverse silicon platforms

3D Inc. ABB Acrodea Activision Blizzard Adobe ALT Software Inc. Analog Devices Antix Aplix Corporation ARC ArcSoft

Ardites austriamicrosystems Autodesk Beatnik BitFlash Broadcom codeplay Conexant

Core Logic Creative Daimler DAZ Dell digia DMP Dolby EA france telecom

AMD Apple ARM intel IBM Sony Computer Entertainment Samsung Imagination Technologies

KHRONOS GROUP

Over 100 companies creating authoring and acceleration standards

Ericsson freescale semiconductor Motorola Nokia nVidia Sun microsystems Texas Instruments

Board of Promoters

FhG Fujitsu Futuremark Corporation GE GiQuila Google graphicREMEDY Hantro Hi Corp. HiSilicon hooked wireless

HUONE Innovation Partner inoras ideaworks3D imec SK innoace LG matrox Marvell Mentor Graphics Micron

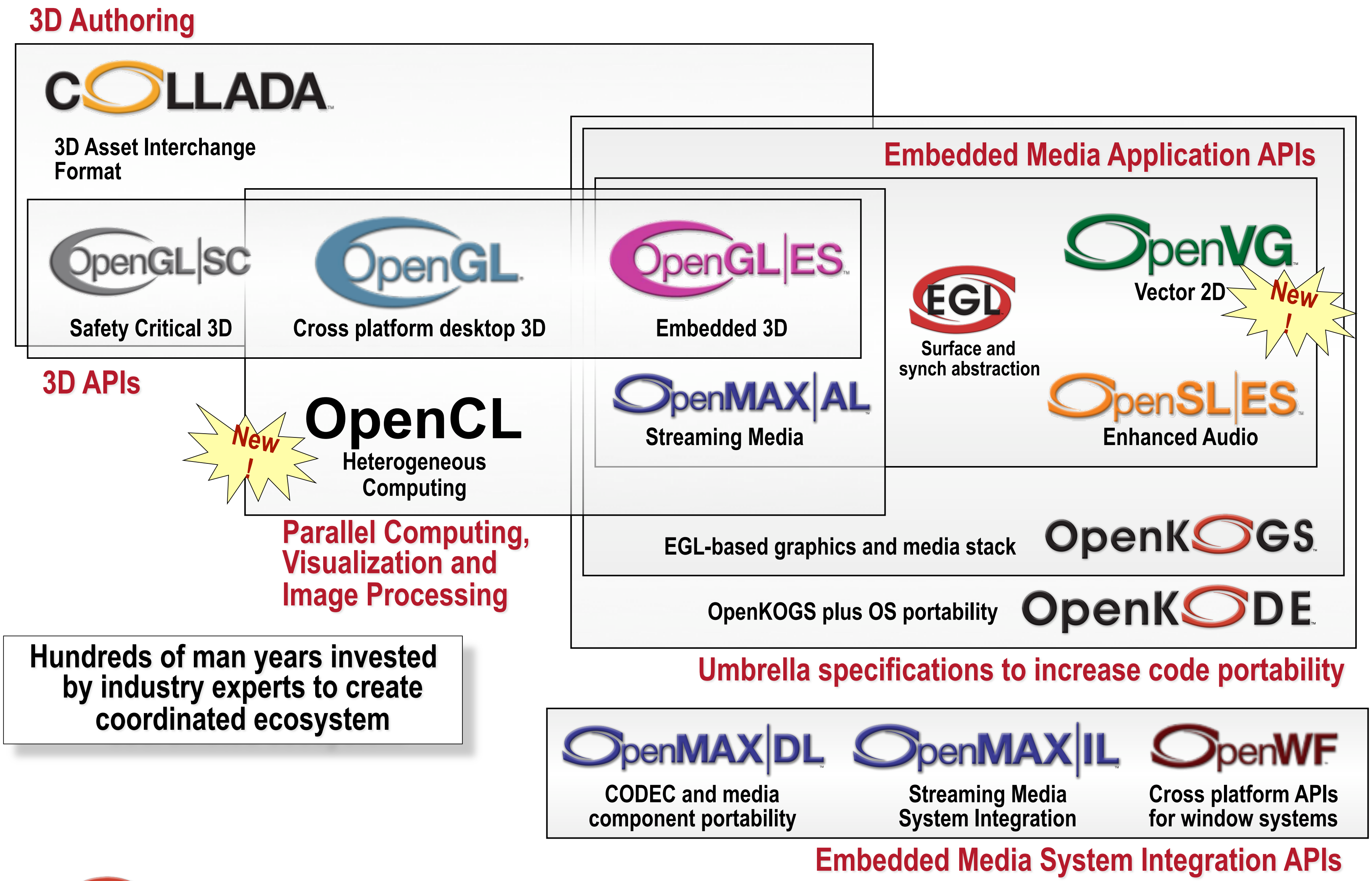
Mitsubishi Electric Monotype movidia NDS NEC Nexus Chips NXP Panasonic QNX Software Systems

Qualcomm RapidMind Renesas S3 Graphics sasken Scaleform Seaweed Systems Siemens SiRF

SK telecom SmithMicro Software SoftBank Sony Ericsson ST Sunplus symbian Takumi tat TES Electronic Solutions

Toshiba Transgaming Technologies TTPCOM Tungsten Graphics Universität Karlsruhe (TH) Umeå Universitet Vincent Vivante Yamaha Yumetech

Khronos Standards Ecosystem



3D Authoring

COLLADA

3D Asset Interchange Format

OpenGL|SC

Safety Critical 3D

OpenGL

Cross platform desktop 3D

OpenGL|ES

Embedded 3D

3D APIs

New !

OpenCL

Heterogeneous Computing

Parallel Computing, Visualization and Image Processing

Embedded Media Application APIs

EGL

Surface and synch abstraction

OpenVG

Vector 2D

New !

OpenMAX|AL

Streaming Media

OpenSL|ES

Enhanced Audio

EGL-based graphics and media stack

OpenKOGS

OpenKOGS plus OS portability

OpenKODE

Umbrella specifications to increase code portability

Hundreds of man years invested by industry experts to create coordinated ecosystem

OpenMAX|DL

CODEC and media component portability

OpenMAX|IL

Streaming Media System Integration

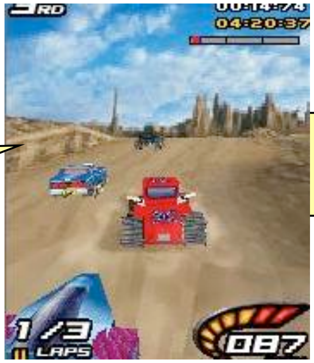
OpenWF

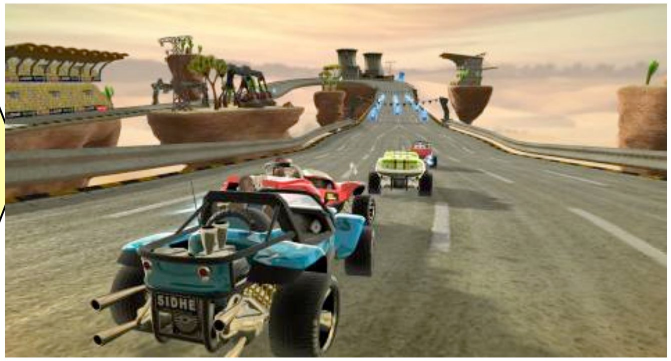
Cross platform APIs for window systems

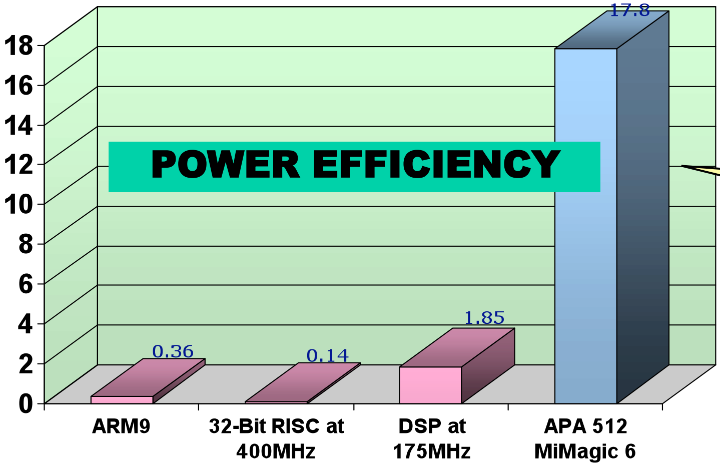
Embedded Media System Integration APIs

Advantages of Mobile Graphics Acceleration

Faster Performance at Higher Quality
Hardware delivers smoother interaction with much better looking graphics


Software 3D


Accelerated 3D

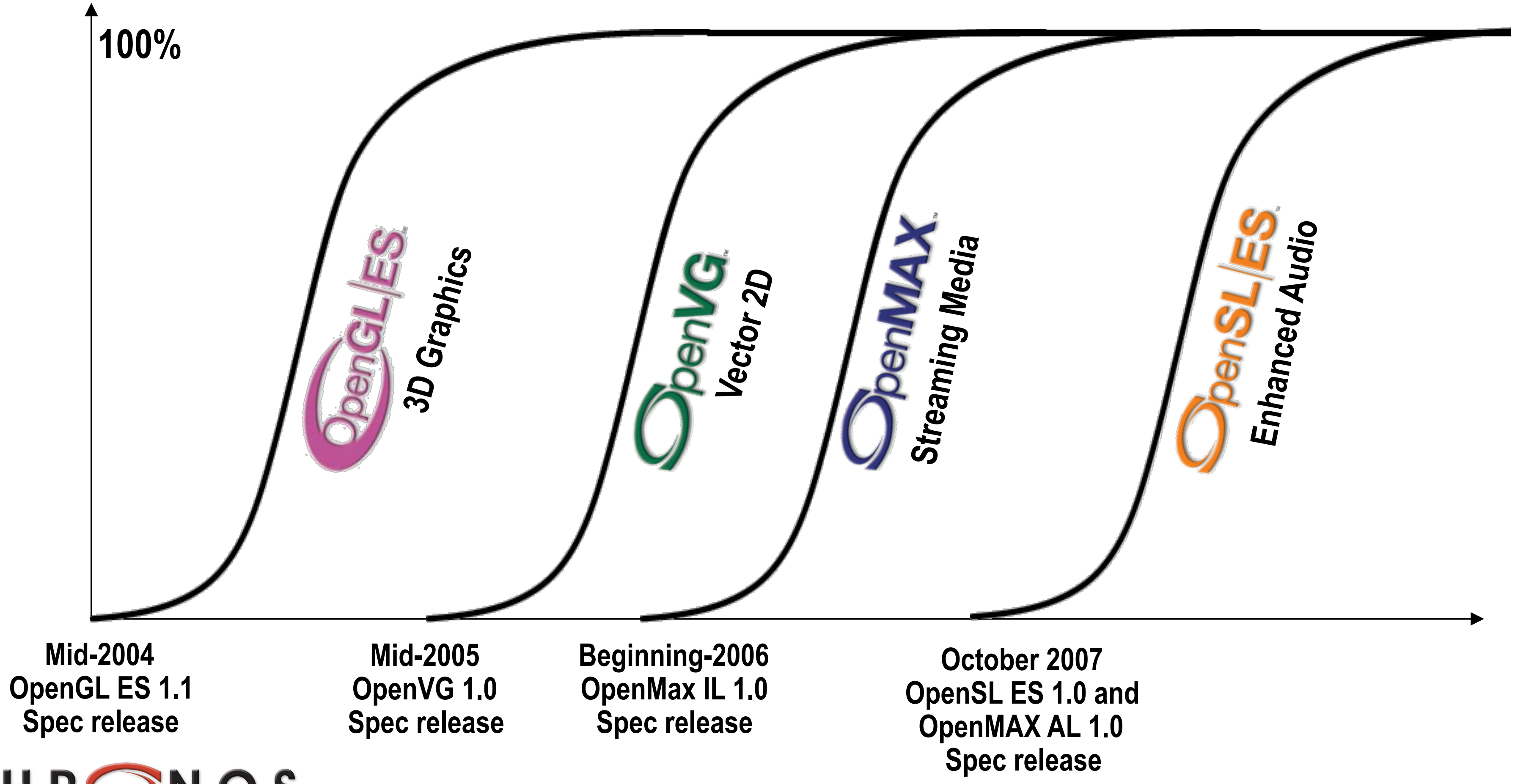


Less Power
Hardware accelerators exploit media pipeline parallelism and caching for a x10 increase in power efficiency over software

Better User Experience
Smaller screens need more advanced graphics processing per pixel

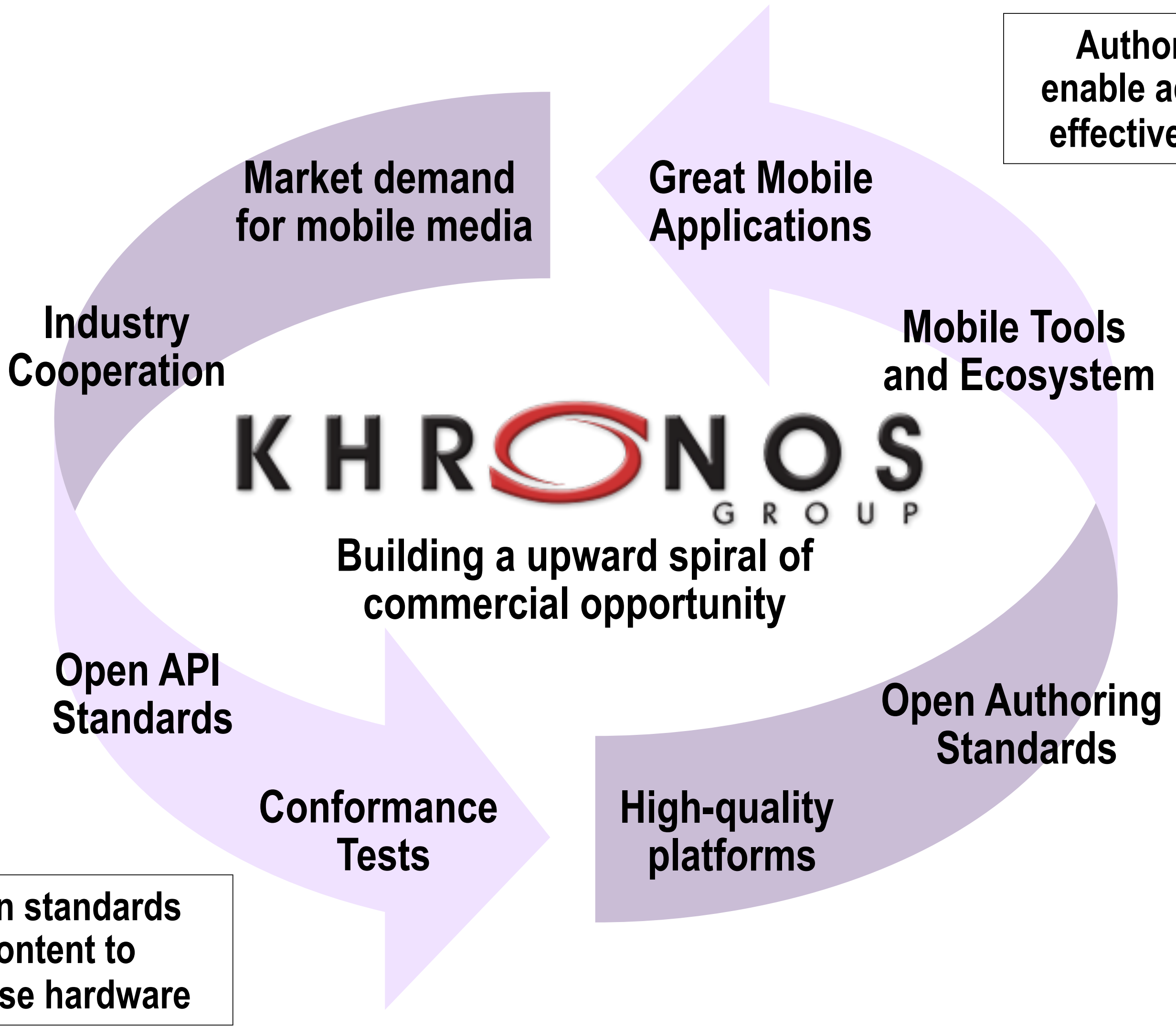
Adoption of Khronos Mobile APIs

- OpenGL ES 1.1 is becoming widespread, OpenGL ES 2.0 silicon has arrived
-OpenVG & OpenMAX IL – becoming widely implemented by silicon vendors
- OpenMAX AL and OpenSL ES – preliminary specs available – implementations started



Khronos Mobile Ecosystem

Authoring standards enable advanced tools to effectively build content



Market demand for mobile media

Great Mobile Applications

Industry Cooperation

Mobile Tools and Ecosystem

KHRONOS GROUP

Building a upward spiral of commercial opportunity

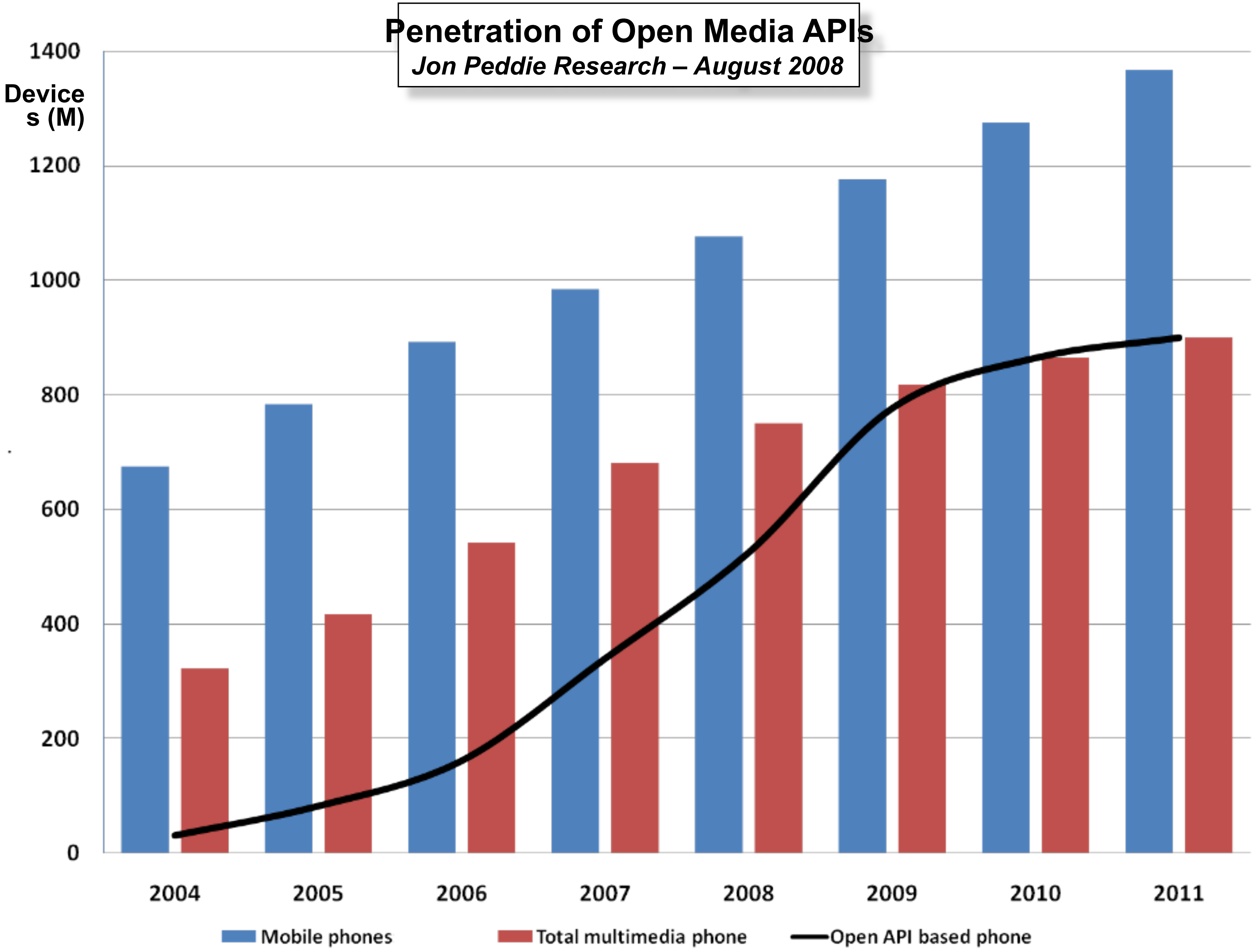
Open API Standards

Open Authoring Standards

Conformance Tests

High-quality platforms

Acceleration standards enable content to effectively use hardware



Penetration of Open Media APIs

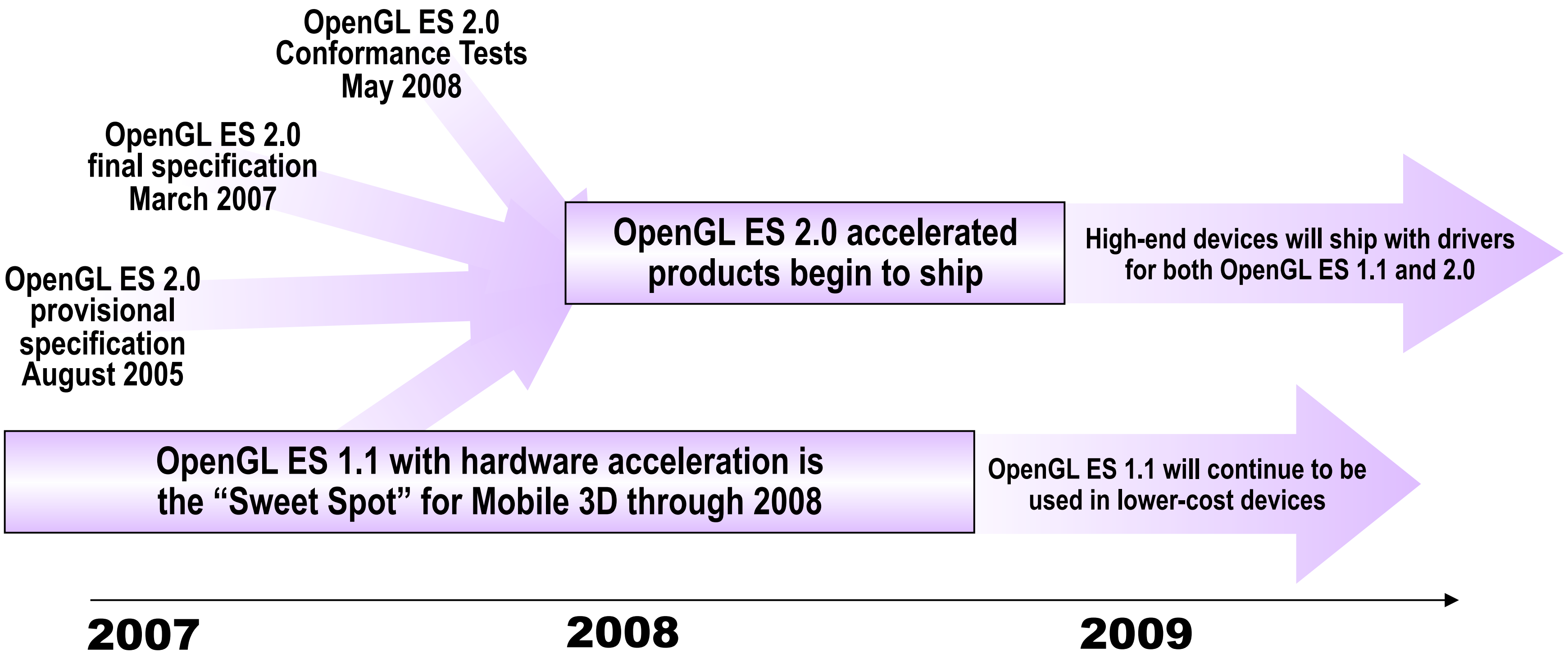
Jon Peddie Research – August 2008

OpenGL ES

- **The leading 3D rendering API for mobile devices**
  - Based on desktop OpenGL – but optimized for mobile / handheld devices
  - Removes redundancy & rarely used features - adds mobile-friendly data types
  - The power of OpenGL distilled into a much smaller package
- **A smashing success!**
  - Widely used in mobile phone handsets from every major handset manufacturer
  - Personal Navigation, Personal Media Player, Automotive, Set-Top Box, Mobile Internet Device
  - Brew, Windows Mobile, Symbian, Android, iPhone OS, Limo
- **OpenGL ES has become the most widely deployed 3D API**
  - Used in diverse applications, devices and markets

OpenGL ES Roadmap

- OpenGL ES 2.0 silicon implementations now shipping
  - Shader-based graphics comes to mobile
  - Conformance tests shipping in May 2008
- Listening carefully to implementation and developer feedback
  - The determine next-generation requirements



OpenGL ES 2.0 Conformance Tests May 2008

OpenGL ES 2.0 final specification March 2007

OpenGL ES 2.0 provisional specification August 2005

OpenGL ES 2.0 accelerated products begin to ship

High-end devices will ship with drivers for both OpenGL ES 1.1 and 2.0

OpenGL ES 1.1 with hardware acceleration is the “Sweet Spot” for Mobile 3D through 2008

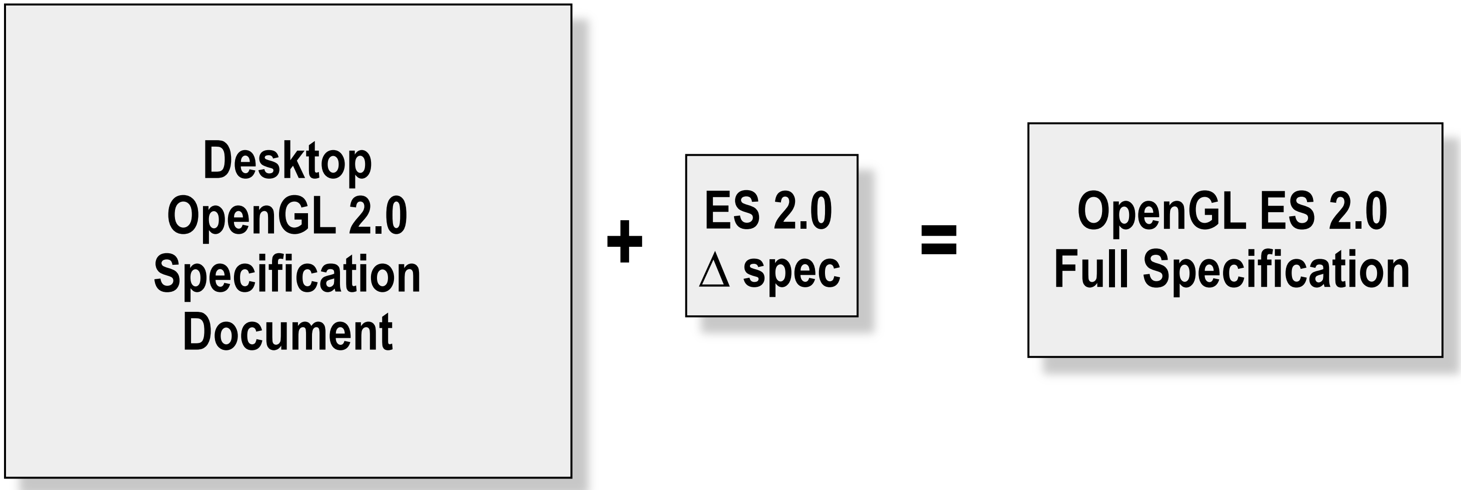
OpenGL ES 1.1 will continue to be used in lower-cost devices

OpenGL ES 2.0 Conformance Tests

- **Launched May 2008**
  - The best OpenGL conformance tests ever!
  - About 120,000 lines of code, 1637 shader programs
  - Submission site went live on 12 May 2008
  - A team effort, led by Maurice Ribble (AMD)
- **What this means**
  - Guarantee of a quality OpenGL ES 2.0 implementation
  - Only conformant implementations can use the trademark name and logo
- **Conformant Products so far**
  - AMD Z430 Development Kit
  - ARM Mali 200
  - Imagination Technologies SGX530
  - NVIDIA Tegra APX 2500
  - Vivante GC500

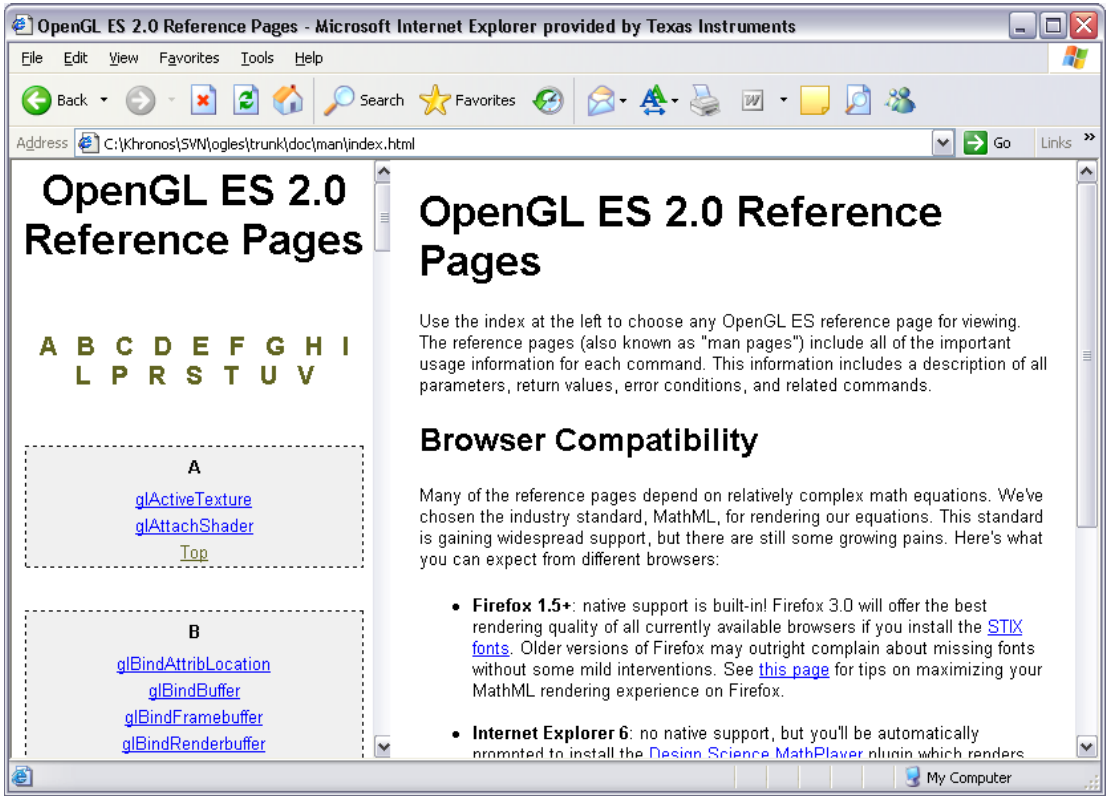
OpenGL ES Full Specification

- **OpenGL ES was originally defined relative to desktop GL**
  - Makes the OpenGL ES spec very compact
  - Nice for people who know the desktop spec well BUT Confusing for people who don't
- **Now have the OpenGL ES 2.0 Full Specification**
  - Authored by Jon Leech for Khronos; now the official definition
  - The original diff spec will be maintained for reference
  - See http://www.khronos.org/registry/gles/


Desktop OpenGL 2.0 Specification Document
+
ES 2.0 Δ spec
=
OpenGL ES 2.0 Full Specification

OpenGL ES Man Pages

- OpenGL ES 2.0 pages are on the web now
  - OpenGL ES 1.1 pages coming soon
- Created by Benj Lipchak based on OpenGL 2.1 pages
  - Thanks to the Khronos Promoters for supporting the effort!


OpenGL ES 2.0 Reference Pages - Microsoft Internet Explorer provided by Texas Instruments
File
Edit
View
Favorites
Tools
Help
Back
Search
Favorites
Go
Links
My Computer

Address C:\Khronos\SVN\ogles\trunk\doc\man\index.html

OpenGL ES 2.0 Reference Pages

A B C D E F G H I L P R S T U V

A

glActiveTexture
glAttachShader
Top

B

glBindAttribLocation
glBindBuffer
glBindFramebuffer
glBindRenderbuffer

OpenGL ES 2.0 Reference Pages

Use the index at the left to choose any OpenGL ES reference page for viewing. The reference pages (also known as "man pages") include all of the important usage information for each command. This information includes a description of all parameters, return values, error conditions, and related commands.

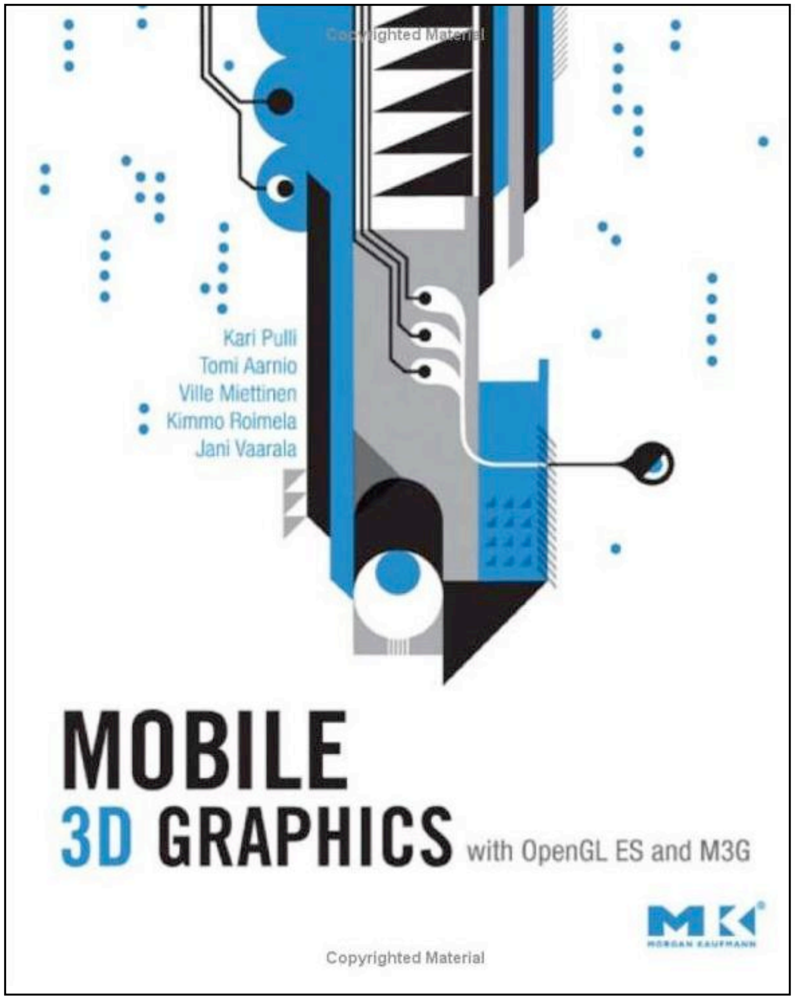
Browser Compatibility

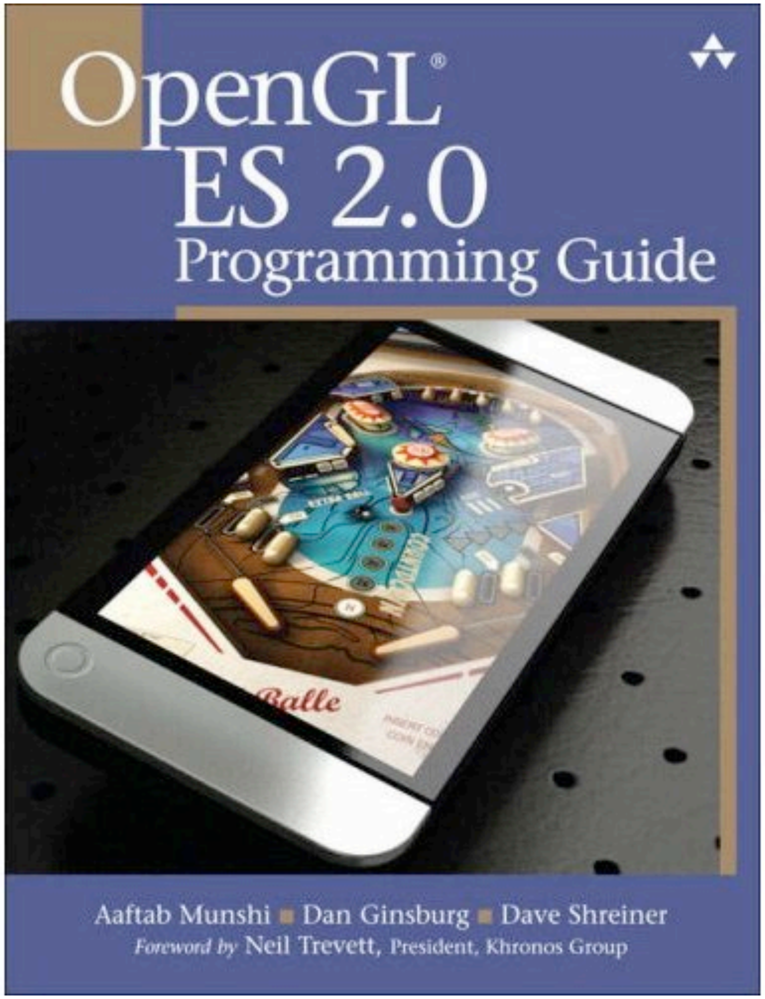
Many of the reference pages depend on relatively complex math equations. We've chosen the industry standard, MathML, for rendering our equations. This standard is gaining widespread support, but there are still some growing pains. Here's what you can expect from different browsers:

- **Firefox 1.5+**: native support is built-in! Firefox 3.0 will offer the best rendering quality of all currently available browsers if you install the STIX fonts. Older versions of Firefox may outright complain about missing fonts without some mild interventions. See this page for tips on maximizing your MathML rendering experience on Firefox.
- **Internet Explorer 6**: no native support, but you'll be automatically prompted to install the Design Science MathPlayer plugin which renders

http://www.khronos.org/opengles/sdk/docs/man

OpenGL ES Books




OpenGL®
ES 2.0
Programming Guide
Aaftab Munshi ■ Dan Ginsburg ■ Dave Shreiner
Foreword by Neil Trevett, President, Khronos Group

OpenGL ES Working Group Priorities

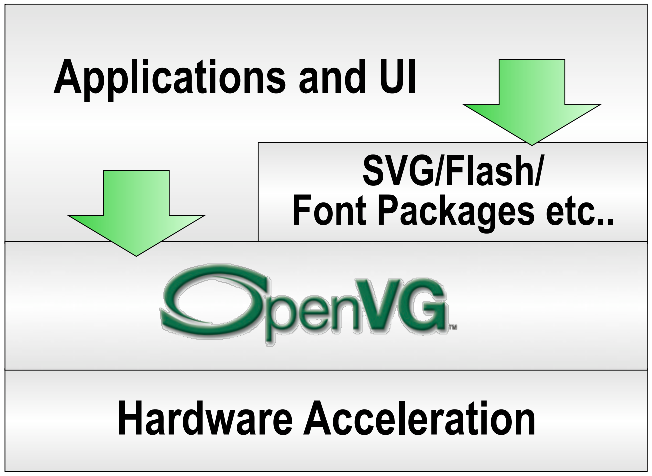
- **Ecosystem Support**
  - Complete OpenGL ES 1.1 man pages
  - ETC1 file format and open-source loader
    - Effort led by Jacob Ström (Ericsson)
  - SDK site
- **Conformance Tests**
  - Updates to both OpenGL ES 2.0 and ES 1.1 tests expected in 4Q 2008
    - ES 1.1 lead – Tom McReynolds (NVidia)
    - ES 2.0 lead – Maurice Ribble (AMD)
- **OpenGL ES 2.0 Extensions**
  - To expose additional hardware capabilities

OpenVG - Accelerated Vector Graphics

- **OpenVG is the industry's first native Bezier rendering API**
  - Enables a new class of hardware acceleration – Bezier primitives – not polygons
- **Primarily used to accelerate higher-level engines such as Flash and SVG**
  - OpenVG is an OpenGL-style, low-level API – but many artists prefer tools over programming

OpenGL ES
Accurately represents
PERSPECTIVE and
LIGHTING
using 3D polygons

OpenVG
Accurately represents
SHAPE and COLOR
using 2D Bezier
Curves



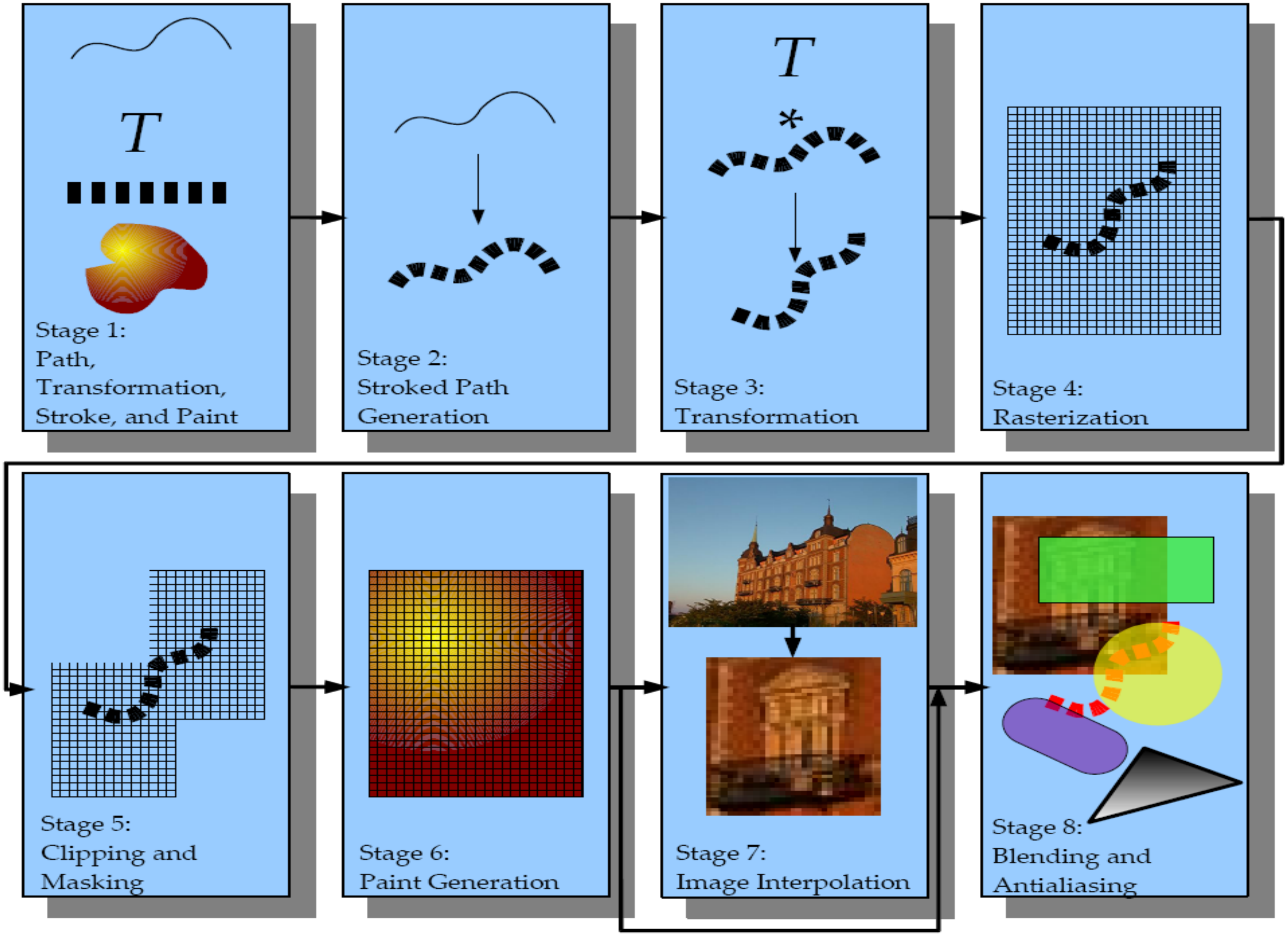
Applications and UI

SVG/Flash/
Font Packages etc..

OpenVG

Hardware Acceleration

OpenVG Pipeline



OpenVG Adoption

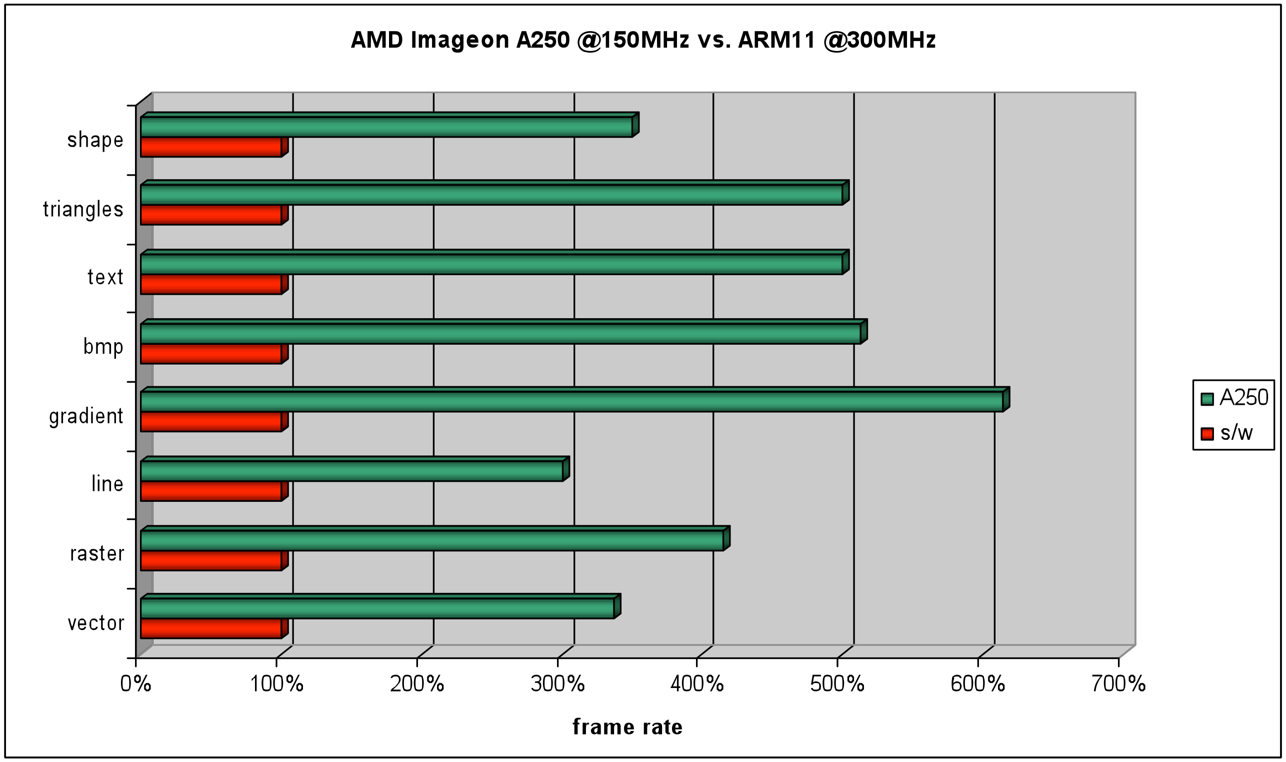
- **OpenVG 1.0 specification shipped in August 2005**
  - Open source sample implementation and conformance tests in March 2007
- **OpenVG developer momentum is growing strongly**
  - Cairo renderer in GTK+ widget toolkit, WebKit Browser engine, SVG acceleration
  - Native UI, mapping applications
- **OpenVG silicon acceleration is shipping**
  - Many graphics silicon vendors have announced support
  - 8 conformance submissions this year
- **Active work in OpenVG extensions**
  - NDS vendor extensions for advanced blending and filtering in HDTV market

OpenVG 1.1

- **OpenVG 1.1 specification publicly released at SIGGRAPH Asia 2008**
  - Together with open source sample implementation and full Conformance Test Suite
- **OpenVG 1.1 Conformance Test**
  - Comprehensive set of tests (~900 test cases)
  - Conformant implementation has to pass all of them
- **Major features**
  - Adobe Flash 7 / Flash Lite 3 support
  - Glyph API for hardware accelerated text rendering
  - Multi-sampled anti-aliasing

Flash Acceleration with OpenVG 1.1

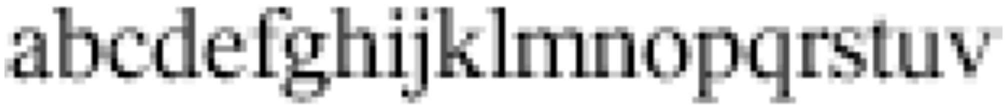
- **Adobe is a strong participant in the OpenVG working group**
  - Helping to ensure that OpenVG 1.1 provides effective Flash acceleration
- **OpenVG support is now included in Flash Lite 3.1 engine**
  - Provides approximately 5x speed-up over software at 1/10 the power



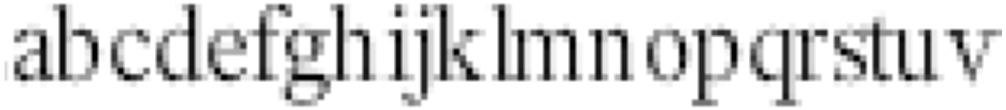
Glyph API

- Many applications require high-quality scalable text
  - User interfaces, SVG, PDF or E-book readers, GPS and mapping, web-browsers
- Glyphs are complex 2D shapes
  - Can be scaled down to very small sizes and minor variations in pixel coverage and / or glyph positioning may significantly affect legibility and perceived text quality
- CPU and bandwidth-efficient commands to use cached font data
  - Complex text layout support, including kerning, glyph variants and positional adjustments
  - Use both vector outlines and bitmap images to display characters
- Glyph API enables text-specific rendering optimizations
  - Auto-hinting of glyph outlines, modified filter kernels for anti-aliased text rendering
  - Missing pixel recovery for monochrome rendering mode

Unoptimized

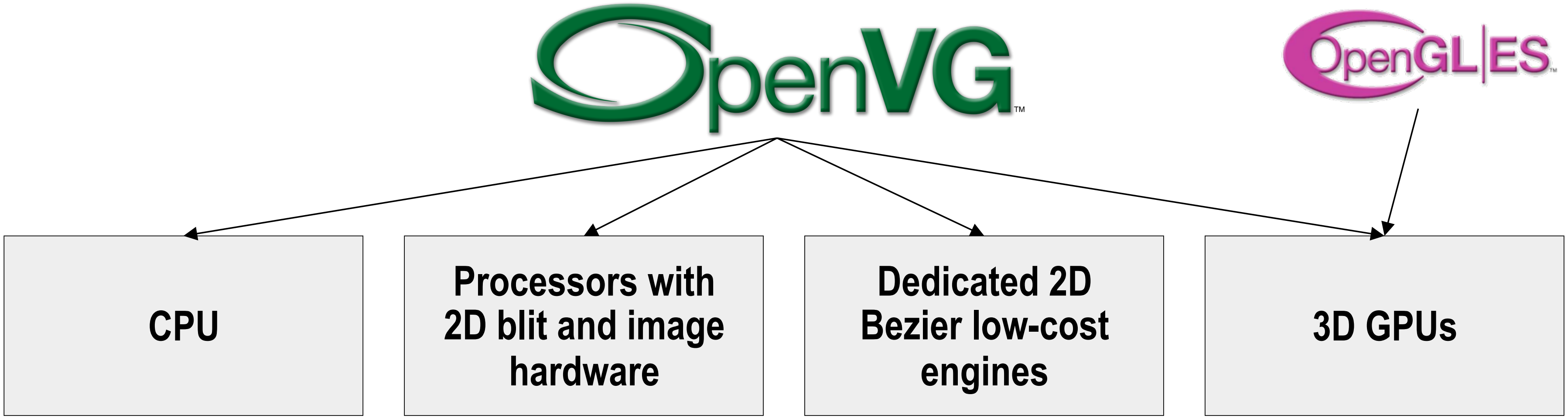


Optimized using Glyph API



OpenVG Implementations

- **There are a wide range of OpenVG implementations possible**
  - Depending on the price point and target device architecture
- **3D GPU can accelerate both OpenVG and OpenGL ES APIs**
  - With interoperability through EGL



CPU

Processors with 2D blit and image hardware

Dedicated 2D Bezier low-cost engines

3D GPUs

100% Software

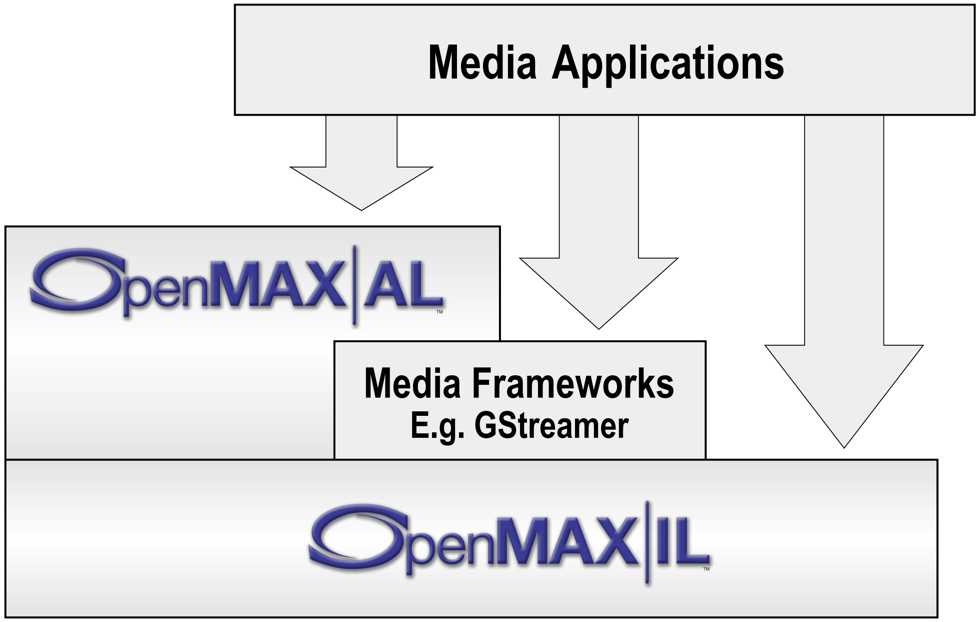
Software Bezier rendering with some acceleration for imaging pipeline

Fully accelerated Bezier rendering and imaging pipeline

GPU accelerates Bezier rendering / imaging pipeline through OpenVG API - and 3D acceleration through OpenGL ES API

OpenMAX Applications

- **OpenMAX AL provides fastest development and enhanced portability**
  - High-level, object-oriented abstractions for record/play applications
- **OpenMAX IL provides more flexibility and low-level hardware access**
  - For expert developers, no high-level or convenience functions
- **OpenMAX IL can be used by the native media framework**
  - Framework can be used to implement and extend OpenMAX AL

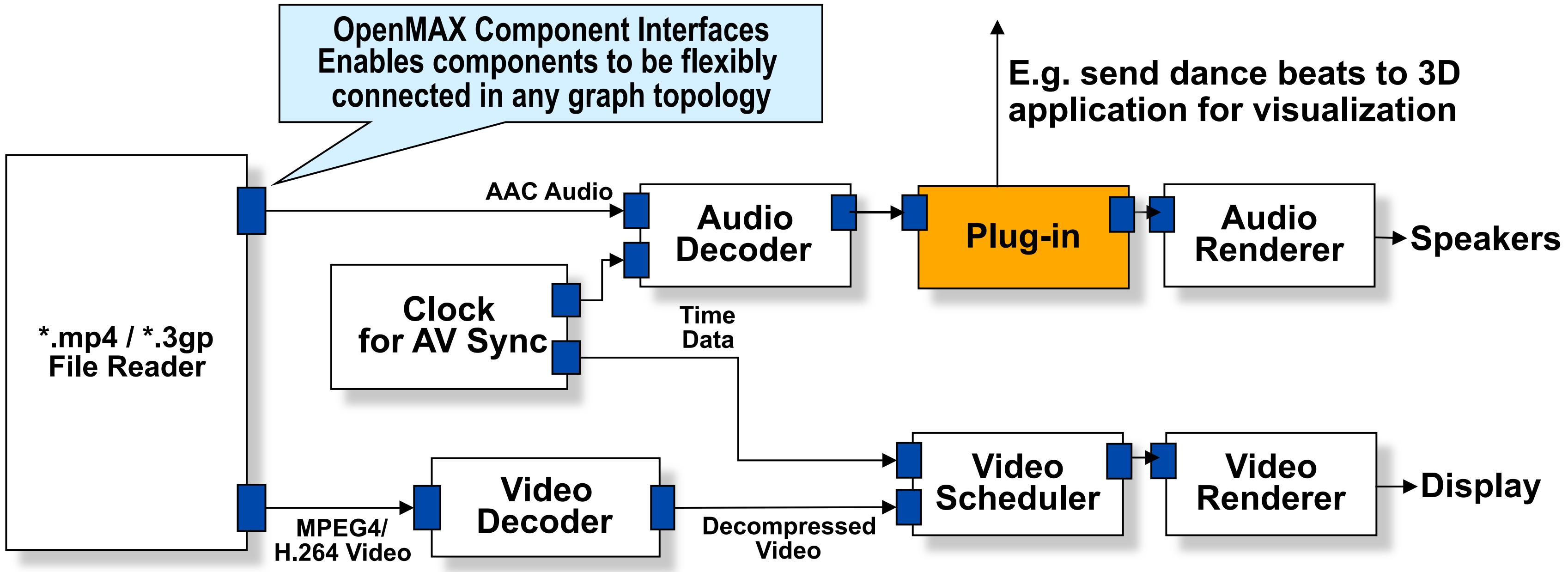


Applications may use OpenMAX AL, OpenMAX IL or the native media framework

OpenMAX AL may be implemented independently and not layered over OpenMAX IL

OpenMAX IL Example Graph

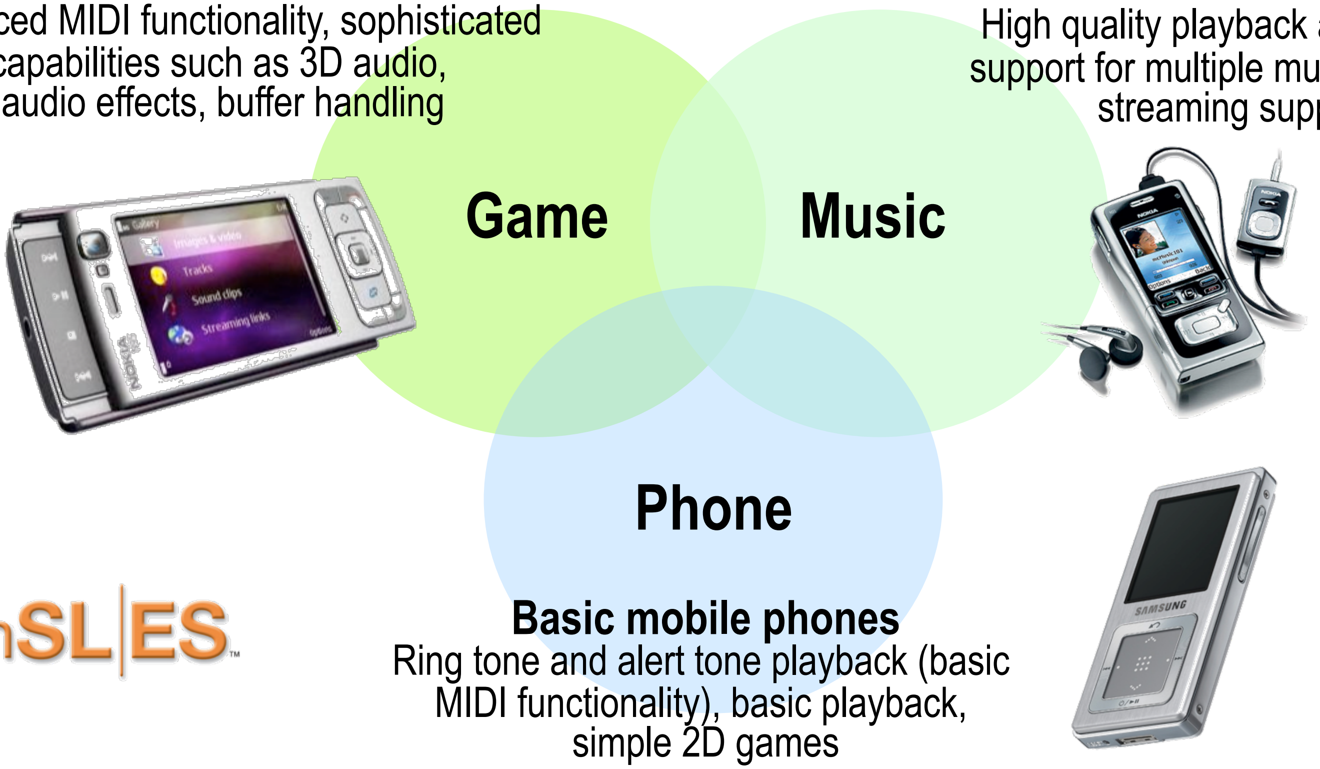
- Standardized component interfaces enable flexible media graphs
- Includes multi-stream synchronization
- Allows for custom plug-ins


OpenMAX Component Interfaces Enables components to be flexibly connected in any graph topology
E.g. send dance beats to 3D application for visualization
*.mp4 / *.3gp File Reader
AAC Audio
Audio Decoder
Plug-in
Audio Renderer
Speakers
Clock for AV Sync
Time Data
Video Scheduler
Video Renderer
Display
MPEG4/ H.264 Video
Video Decoder
Decompressed Video

Example: MPEG-4 video synchronized with AAC audio decode

Three OpenSL ES Profiles

- **Phone, Music and Game**
  - Sufficient to cover all the market segments of importance to OpenSL ES
- **Some profile overlap**
  - Mostly in basic audio functionality



Game-centric mobile devices
Advanced MIDI functionality, sophisticated capabilities such as 3D audio, audio effects, buffer handling

Music-centric mobile devices
High quality playback and record, support for multiple music codecs, streaming support

Basic mobile phones
Ring tone and alert tone playback (basic MIDI functionality), basic playback, simple 2D games

OpenMAX and OpenSL ES Priorities

- **OpenMAX IL 1.1.2 – released in September 2008**
  - Corrections and clarifications
  - Extension mechanism
- **Several extensions to IL 1.1 in discussion**
  - Imaging, 3D audio, content demuxer, content pipes
- **OpenMAX IL graph-level conformance tests**
  - Due for release early 2009
- **OpenSL ES and OpenMAX AL final specifications and conformance tests**
  - Due for release in early 2009

Mobile Umbrella Specifications

- **Individual APIs define domain specific media acceleration**
  - OpenGL ES for 3D, OpenMAX for video and images etc.
- **Latest mobile applications want to MIX media types**
  - E.g. route live video into a composited 3D user interface
- **For portability of mixed-media need to define how the APIs work *together***
  - E.g. how to transfer video data from OpenMAX into OpenGL ES
- **Umbrella specs define and CONFORMANCE TEST trans-API operation**
  - Creates a reliable, cross-vendor media-stack definition

OpenKOGS™

OpenKODE™

EGL

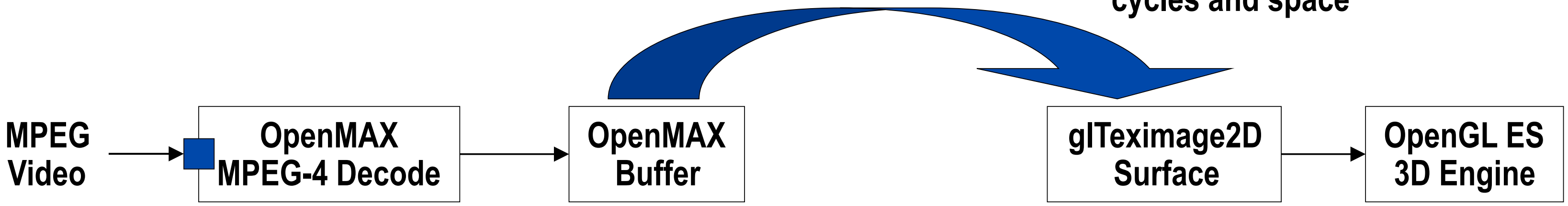
- **EGL abstracts access to rendering surfaces**
  - Interfaces Khronos rendering APIs to native platform window system
  - A derivative of the WGL Windows abstraction API
- **Emerging role as a communication hub between handheld APIs**
  - Sharing images via EGLImage extensions
  - EGLSync objects for inter-API fences and other signalling
- **Can create rendering surfaces into which multiple client APIs can draw**
  - Enables high-performance, accelerated, mixed-mode 2D and 3D rendering
  - Using OpenGL ES and OpenVG
- **EGL 1.3 was released in December 2006 – supports OpenGL ES**
  - OpenKODE 1.0 uses EGL 1.3 PLUS EGL extensions to integrate OpenVG PLUS
  - Lock Surface EGL extension for direct blitting of software rendering applications to the screen
- **EGL 1.4 integrated OpenVG and Lock Surface into core EGL**
  - In spring 2008

Directions for EGL 1.5

- Create EGLImage within EGL
  - Pre-declare uses => guaranteed image compatibility
- Share images with OpenMAX
  - Add EGLImage video data formats (YUV)
- Stream images between APIs
  - Queue of images with producer/consumer operations
- EGLSync objects
  - Inter-API fences and other signalling

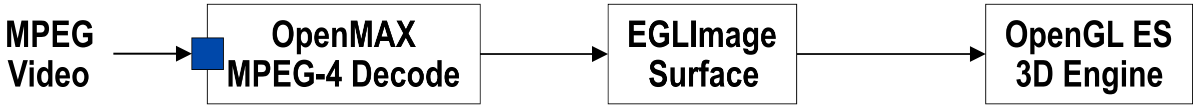
Without EGLImage the game must copy
the OpenMAX buffer into an OpenGL ES texture

Data copy wastes
cycles and space



MPEG Video → OpenMAX MPEG-4 Decode → OpenMAX Buffer → glTeximage2D Surface → OpenGL ES 3D Engine

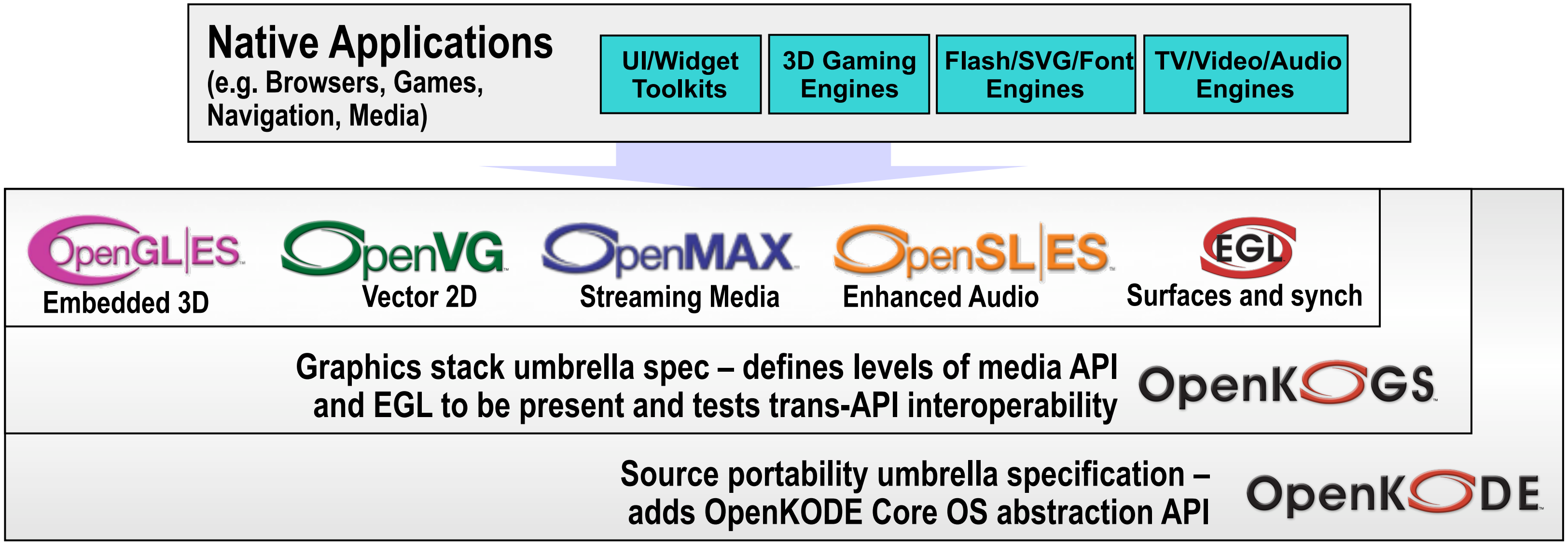
An EGLImage surface can be used as both the destination of
the decode and a texture without copying the data



MPEG Video → OpenMAX MPEG-4 Decode → EGLImage Surface → OpenGL ES 3D Engine

Khronos Mobile Umbrella Specs

- **Platform vendors can choose to ship more than just individual APIs**
  - Provide *conformance tested* multi-API programming platforms
- **1. OpenKOGS – integrated media stack – due in 2009**
  - Defines reliable trans-API interoperability through EGL for OpenGL ES, OpenVG, OpenMAX
- **2. OpenKODE = OpenKOGS *plus* OS abstraction API**
  - OpenKODE Core is Posix-like API for application portability across mobile operating systems
  - Threads, events, utilities, math, timers, files, strings, networking, IO, windowing
  - A lot of mobile OS knowledge encapsulated in the API



Native Applications
(e.g. Browsers, Games, Navigation, Media)

UI/Widget Toolkits

3D Gaming Engines

Flash/SVG/Font Engines

TV/Video/Audio Engines

OpenGL ES – Embedded 3D

OpenVG – Vector 2D

OpenMAX – Streaming Media

OpenSL ES – Enhanced Audio

EGL – Surfaces and synch

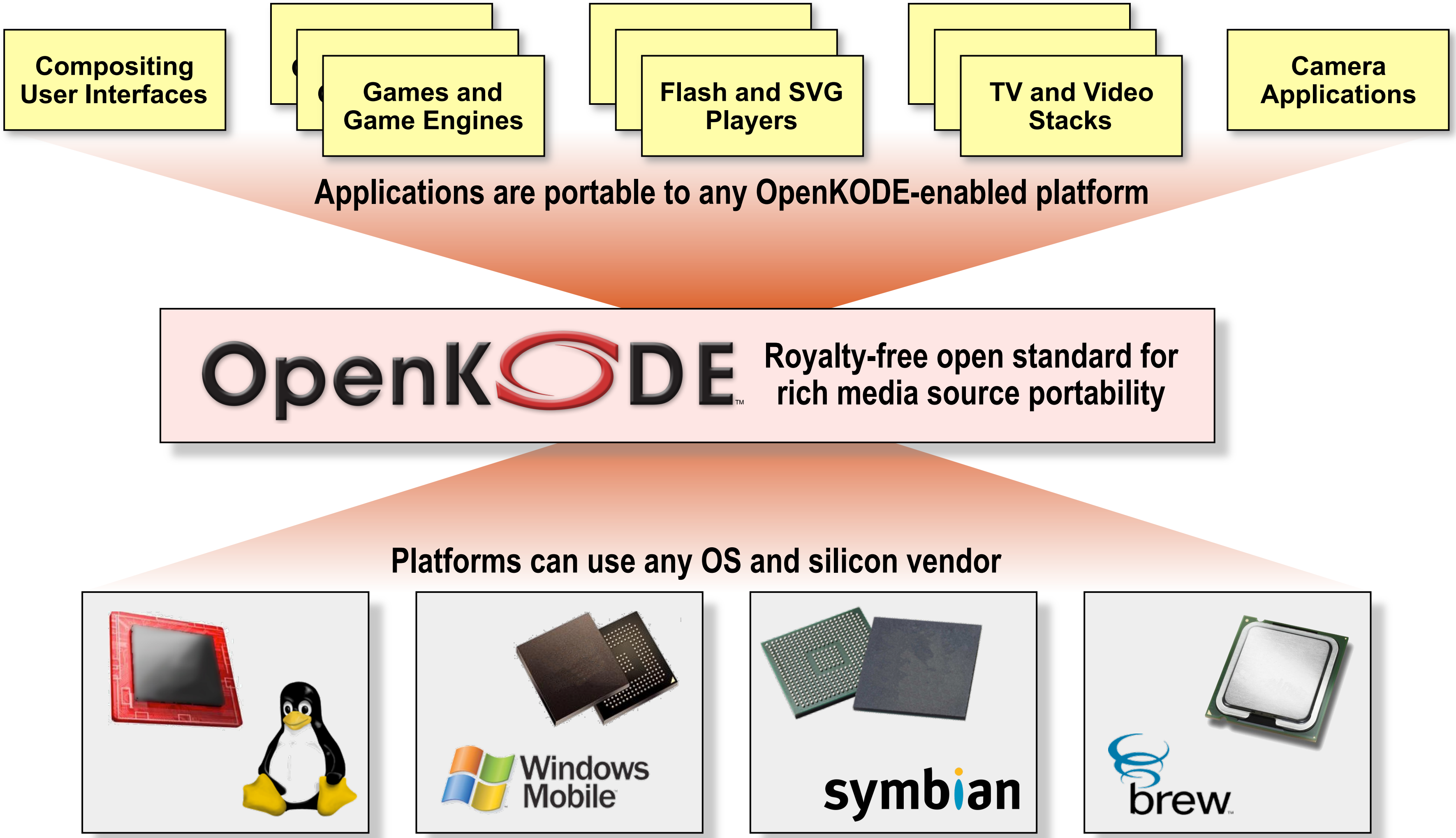
Graphics stack umbrella spec – defines levels of media API and EGL to be present and tests trans-API interoperability – OpenKOGS

Source portability umbrella specification – adds OpenKODE Core OS abstraction API – OpenKODE

OpenKODE Core

- OS Abstraction layer - familiar to POSIX and C Programmers
  - "POSIX ES" – drawing on C89, C99
  - Some added functionality – events and IO functions
- Genuinely implementable across all major mobile operating systems
  - No small feat!
  - A lot of mobile OS knowledge encapsulated in the API
- Threads, Events, Utilities, Math, Time and Timers, File system, Strings, Networking, Input/Output, Windowing
  - Full functionally for rich-media applications
- Lightweight – typically less than 100KB
  - Adds no significant performance overhead
  - OpenKODE Core is NOT a layer between application and media APIs

OpenKODE – Source Portability



Compositing User Interfaces

Games and Game Engines

Flash and SVG Players

TV and Video Stacks

Camera Applications

Applications are portable to any OpenKODE-enabled platform

OpenKODE™

Royalty-free open standard for rich media source portability

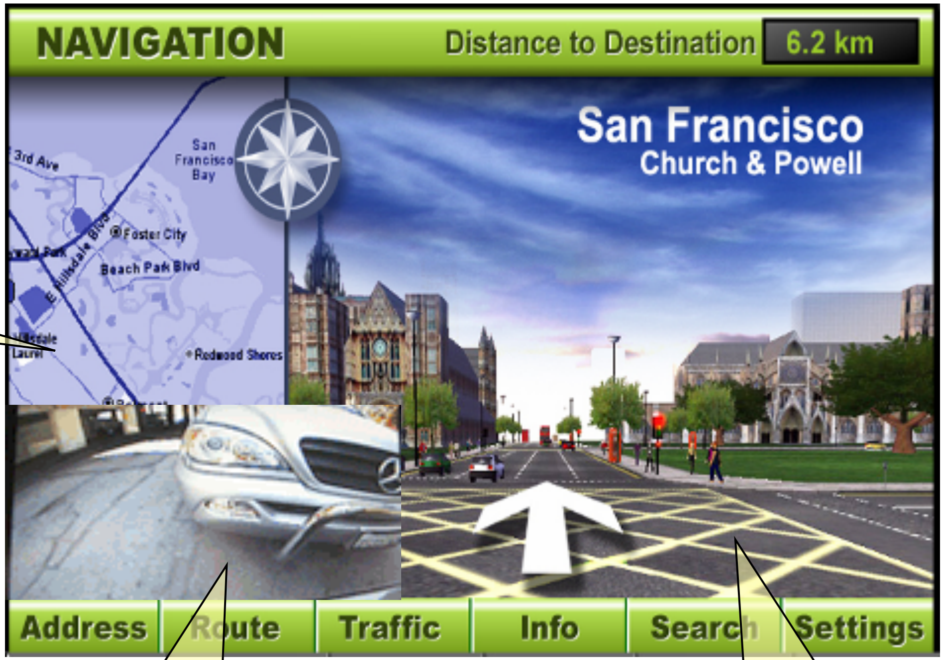
Platforms can use any OS and silicon vendor

Raising 2D and 3D Visual Quality

State-of-the-art APIs enable compelling consumer displays

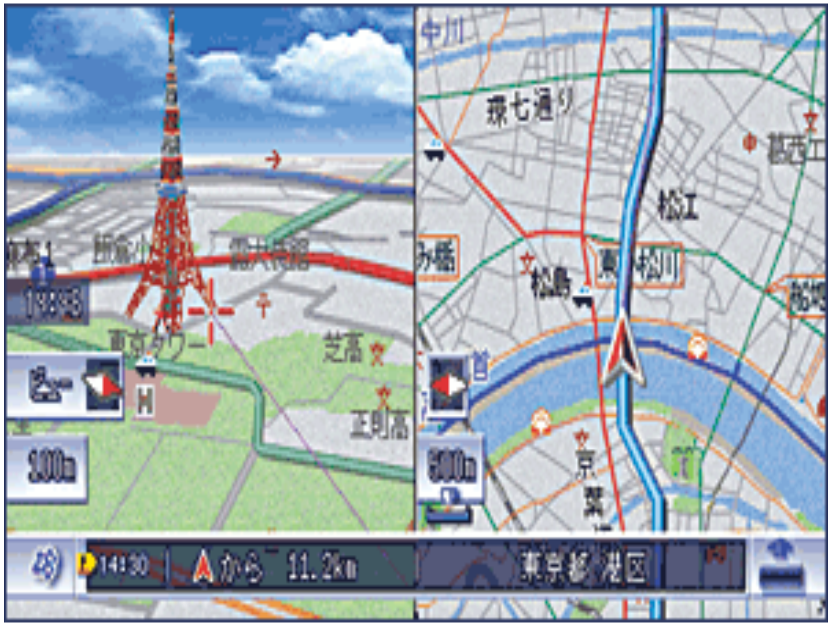
Advanced functionality, fast interactivity and extremely high quality

High-quality 2D graphics and test using OpenVG



Video and image processing with OpenMAX

High-quality 3D displays using OpenGL ES



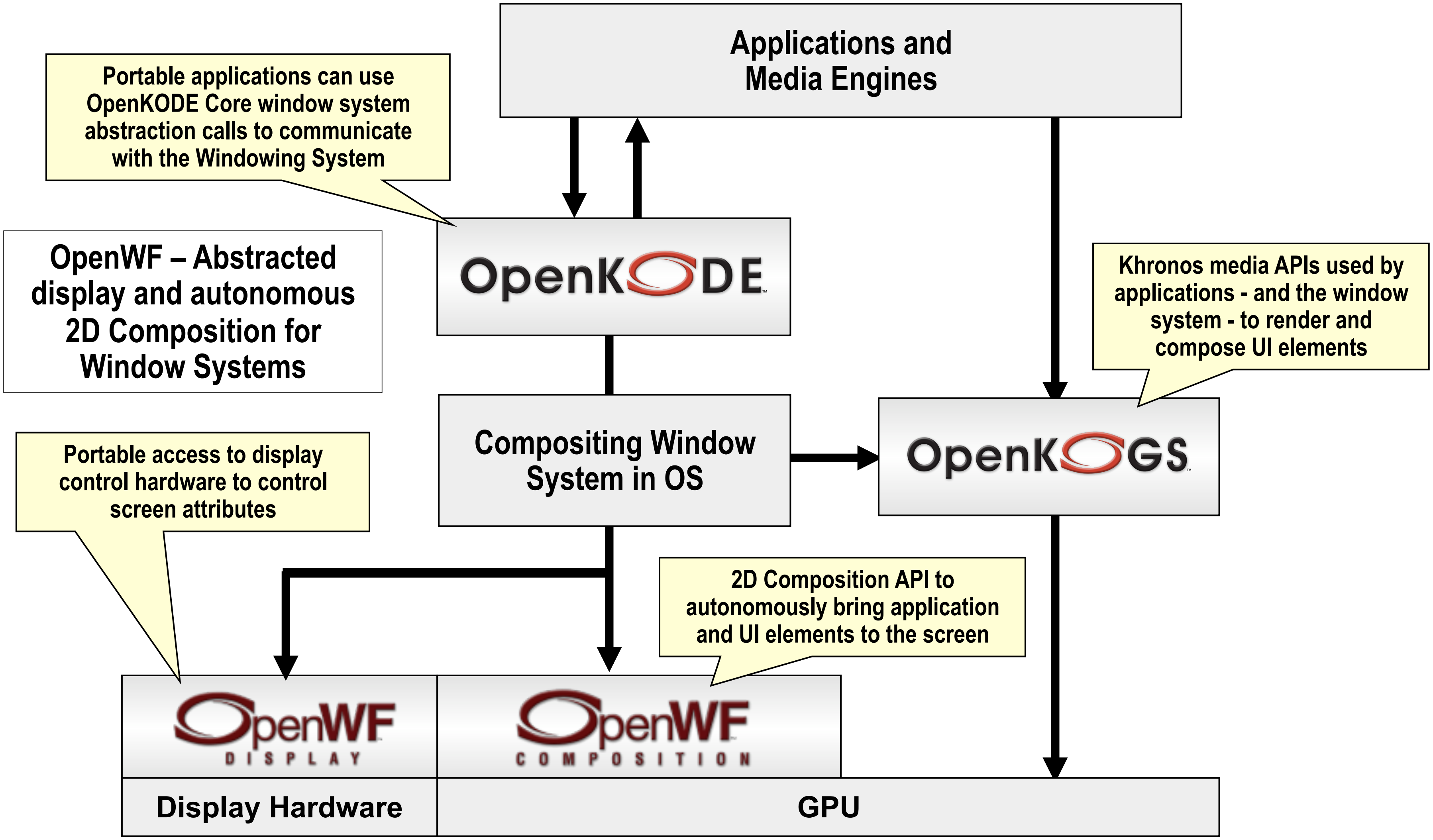
Older generation APIs

Provide rudimentary graphics functionality and quality

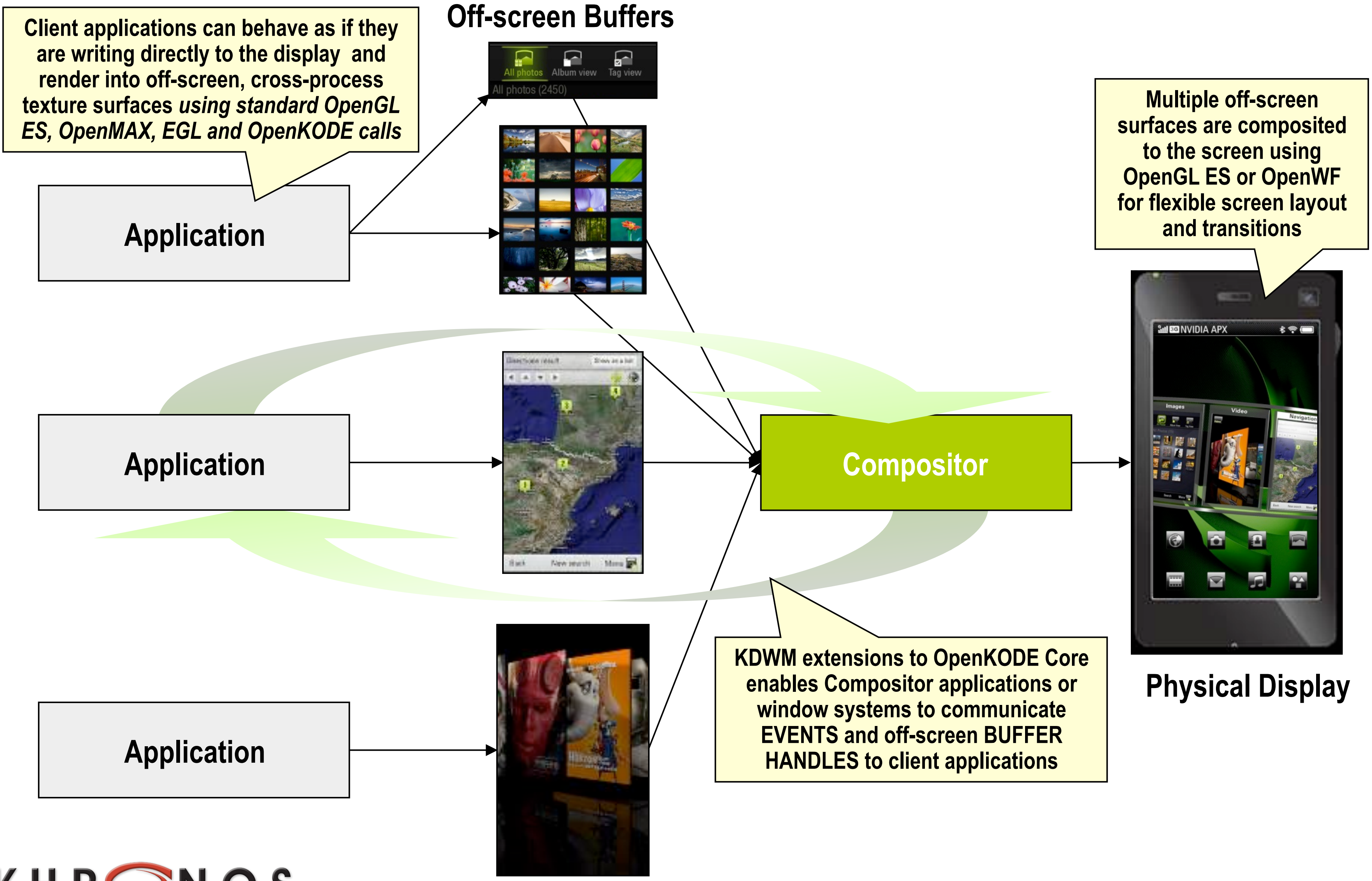
Composition and User Interface

- **The iPhone's success has demonstrates the importance of good UI**
  - Ease of discovery and use of device capabilities and resources drives revenue
- **Strong focus on using mobile graphics acceleration for advanced UI**
  - Accelerated rendering, window management and composition
- **Accelerated composition**
  - Bringing multiple screen elements together with complete flexibility on the display
- **Khronos ecosystem enables 3D composition architectures**
  - KDWM – OpenKODE extensions to enable 3D composition using OpenGL ES

OpenWF – Windowing Foundation


Applications and Media Engines
Portable applications can use OpenKODE Core window system abstraction calls to communicate with the Windowing System
OpenKODE
OpenWF – Abstracted display and autonomous 2D Composition for Window Systems
Khronos media APIs used by applications - and the window system - to render and compose UI elements
Compositing Window System in OS
OpenKOGS
Portable access to display control hardware to control screen attributes
2D Composition API to autonomously bring application and UI elements to the screen
OpenWF DISPLAY
OpenWF COMPOSITION
Display Hardware
GPU

OpenKODE Screen Composition



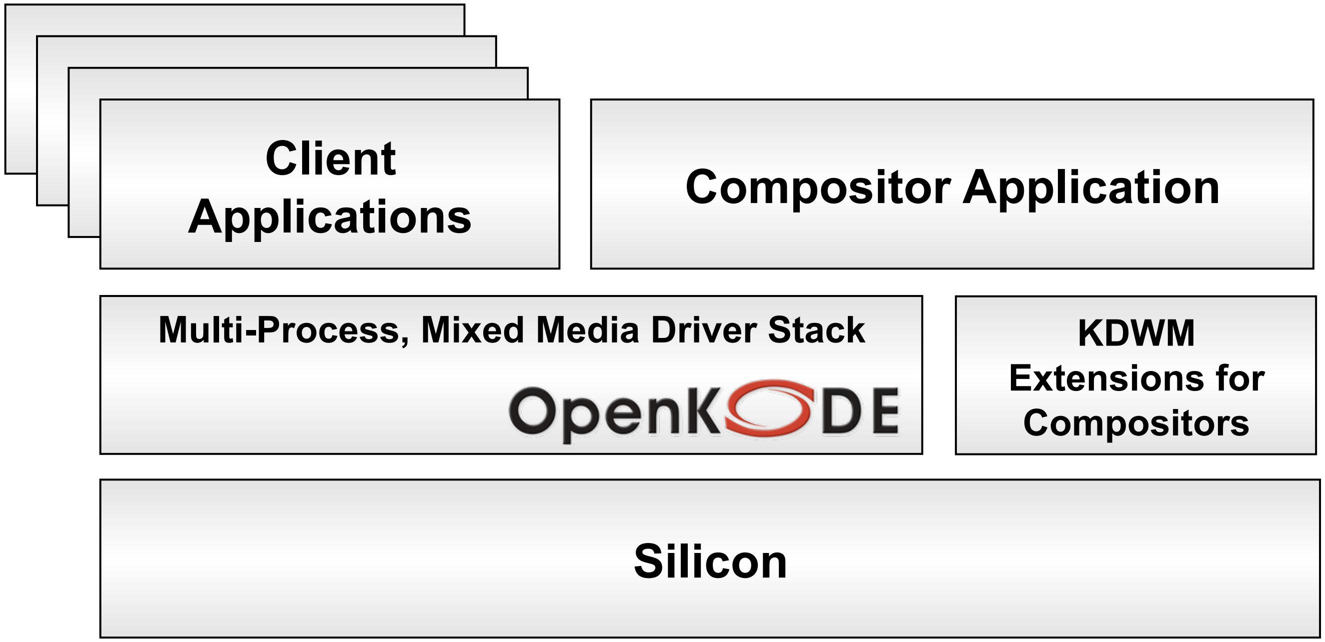
Client applications can behave as if they are writing directly to the display and render into off-screen, cross-process texture surfaces *using standard OpenGL ES, OpenMAX, EGL and OpenKODE calls*

Multiple off-screen surfaces are composited to the screen using OpenGL ES or OpenWF for flexible screen layout and transitions

KDWM extensions to OpenKODE Core enables Compositor applications or window systems to communicate EVENTS and off-screen BUFFER HANDLES to client applications

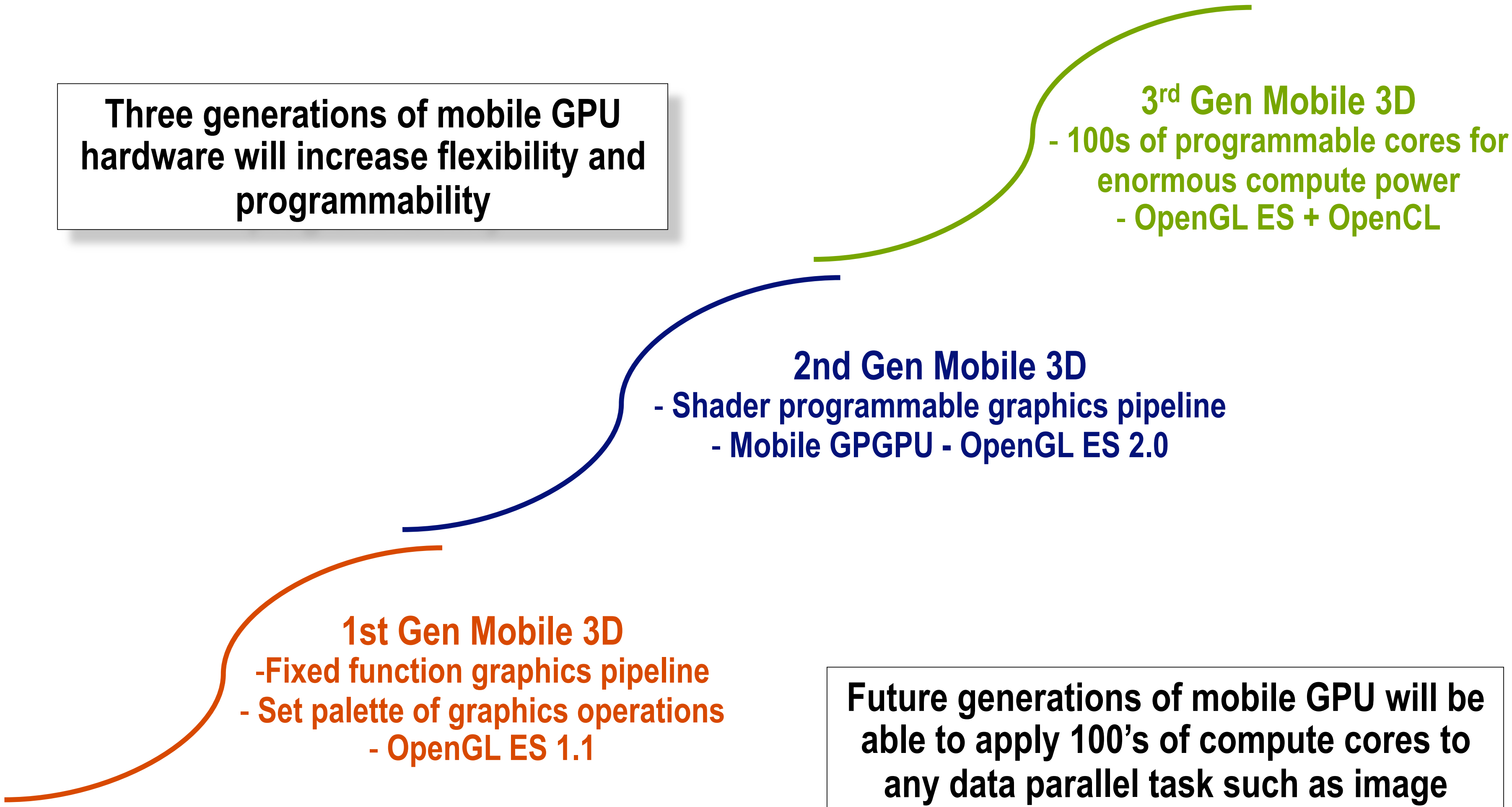
3D UI Composition Initiative

- KDWM – “the Compositors OpenKODE”
  - Extensions to OpenKODE Core - based on existing EGL and OpenKODE mechanisms
  - Off-screen surface allocation and cross process surface handle communication
  - Cross-process event dispatch
  - Proposed by NVIDIA as OpenKODE Core extension


Client Applications
Compositor Application
Multi-Process, Mixed Media Driver Stack
OpenKODE
KDWM Extensions for Compositors
Silicon

Tomorrow:
A Supercomputer in the Palm of your Hand

Three generations of mobile GPU hardware will increase flexibility and programmability



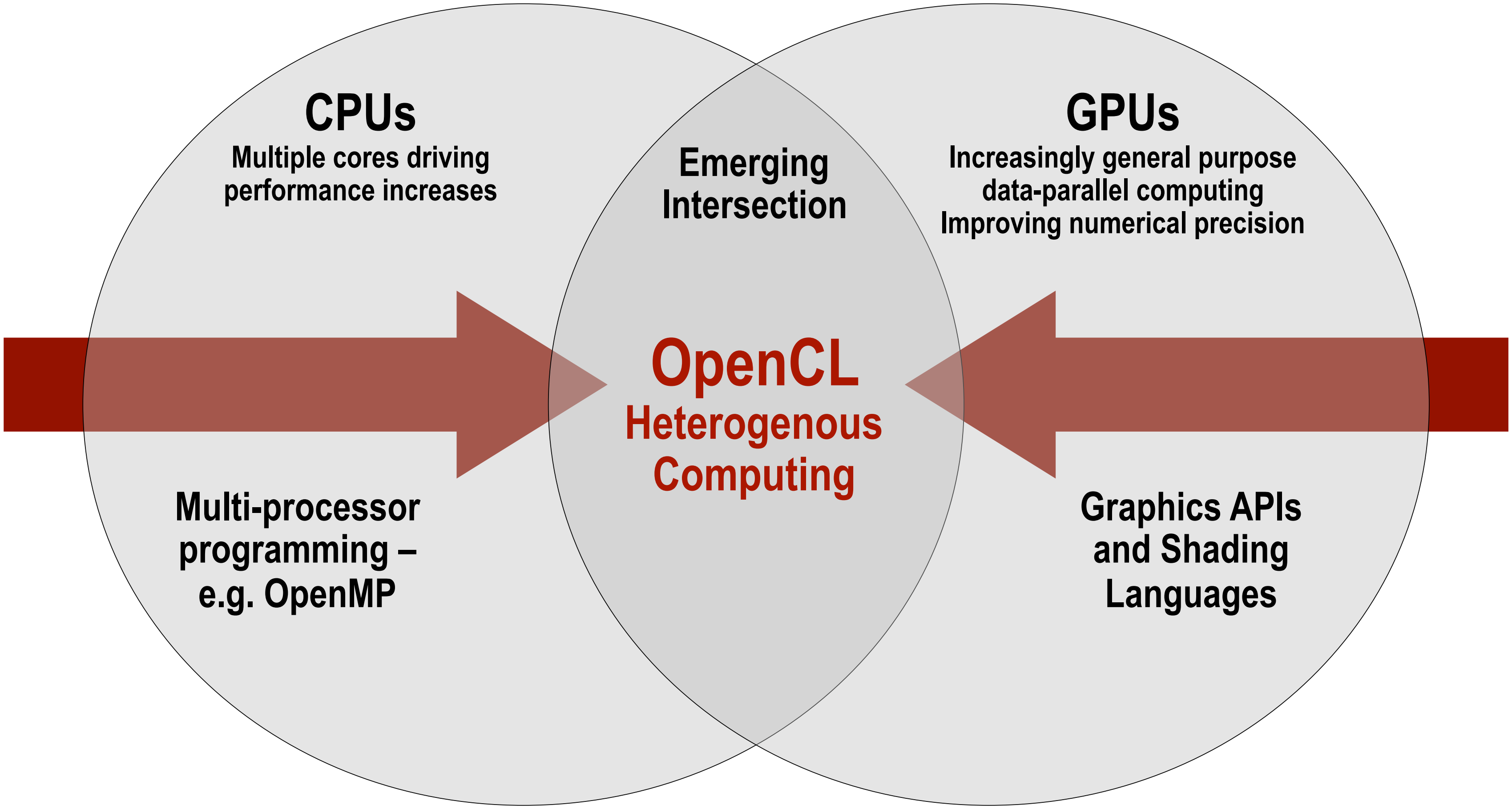
1st Gen Mobile 3D
-Fixed function graphics pipeline
- Set palette of graphics operations
- OpenGL ES 1.1

2nd Gen Mobile 3D
- Shader programmable graphics pipeline
- Mobile GPGPU - OpenGL ES 2.0

3rd Gen Mobile 3D
- 100s of programmable cores for enormous compute power
- OpenGL ES + OpenCL

Future generations of mobile GPU will be able to apply 100's of compute cores to any data parallel task such as image processing and augmented reality

Processor Parallelism



CPUs

Multiple cores driving performance increases

Emerging Intersection

GPUs

Increasingly general purpose data-parallel computing
Improving numerical precision

OpenCL
Heterogenous Computing

Multi-processor programming – e.g. OpenMP

Graphics APIs and Shading Languages

OpenCL – Open Computing Language

Open, royalty-free standard for portable, parallel programming of heterogeneous parallel computing CPUs, GPUs, and other processors

OpenCL Objectives

- **Grow the market for parallel computing**
  - For vendors of systems, silicon, middleware, tools and applications

- **Open, royalty-free standard for heterogeneous parallel computing**
  - Unified programming model for CPUs, GPUs, Cell, DSP and other processors in a system
- **Cross-vendor software portability to a wide range of silicon and systems**
  - HPC servers, desktop systems and handheld devices covered in one specification
- **Support for a wide diversity of applications**
  - From embedded and mobile software through consumer applications to HPC solutions
- **Create a foundation layer for a parallel computing ecosystem**
  - Close-to-the-metal interface to support a rich diversity of middleware and applications
- **Rapid deployment in the market**
  - Designed to run on current latest generations of GPU hardware

OpenCL Working Group

- Diverse industry participation
  - Processor vendors, system OEMs, middleware vendors, application developers
- Many industry-leading experts involved in OpenCL's design
  - A healthy diversity of industry perspectives
- Apple initially proposed the working group
  - And served as specification editor
- Here are some of the other companies in the OpenCL working group

3DLABS SEMICONDUCTOR
ACTIVISION
BLIZZARD
AMD
ARM
BROADCOM
EA
codeplay
ERICSSON
freescale semiconductor
HI CORP.
IBM
intel
Imagination TECHNOLOGIES
MOTOROLA
movidia
NOKIA
NVIDIA
QNX SOFTWARE SYSTEMS
RAPIDMIND
SAMSUNG
Seaweed SYSTEMS
TAKUMI
TEXAS INSTRUMENTS
UMEA UNIVERSITET

Using OpenCL with OpenGL (ES)

- **Both standards under one IP framework**
  - Enables very close collaborative design
- **Efficient, inter-API communication**
  - While still allowing both APIs to handle the types of workloads for which they were designed
- **OpenCL can efficiently share resources with OpenGL (ES)**
  - Textures, Buffer Objects and Renderbuffers
  - Data is shared, not copied
- **Applications can select compute device(s) to run OpenGL and OpenCL**
  - Efficient queuing of OpenCL and OpenGL commands into the hardware
  - Flexible scheduling and synchronization
- **Examples**
  - Vertex and image data generated with OpenCL and then rendered with OpenGL
  - Images rendered with OpenGL and post-processed with OpenCL kernels
  - etc. etc.

OpenCL 1.0 Embedded Profile

- **Enables OpenCL on mobile and embedded silicon**
  - Relaxes some data type and precision requirements
  - Avoids the need for a separate “ES” specification
- **OpenCL will be tightly integrated with OpenGL ES and OpenMAX**
  - Enabling advanced applications e.g. augmented reality

A GPS phone processes images to recognize buildings and landmarks and uses the internet to supply relevant data

Why Are Khronos Standards Key?

- **"Foundation Level" APIs to enable software to effectively use silicon**
  - State-of-the-art, fundamental functionality needed on every platform
  - Non-proprietary – application portability across many platforms
- **Architected to be cost and power efficient**
  - Ideal for embedded markets
- **Hundreds of man years invested**
  - Beyond any single company now to produce specifications of this breadth and depth
  - Leverage the investment of the silicon industry in creating standards-based drivers sets
- **Royalty-free**
  - Khronos is committed to generating market opportunities for its members and the industry

- **More information and slides at www.khronos.org**