Developing Open Standards for Safety Critical Technologies

Erik Noreke
Independent Technology Consultant
Khronos Chair Safety Critical Working Group

4th Scandinavian Conference - System and Software Safety | March 2016

Agenda

- **Introduction to Khronos and open standards**
  - Who is the Khronos Group
  - What are royalty-free open standards
  - Industry collaboration
- **Developing safety critical technology standards**
  - Case study: Developing OpenGL SC
  - Lessons Learned
- **Ways forward**
  - New ways of working
  - Safety Critical road map

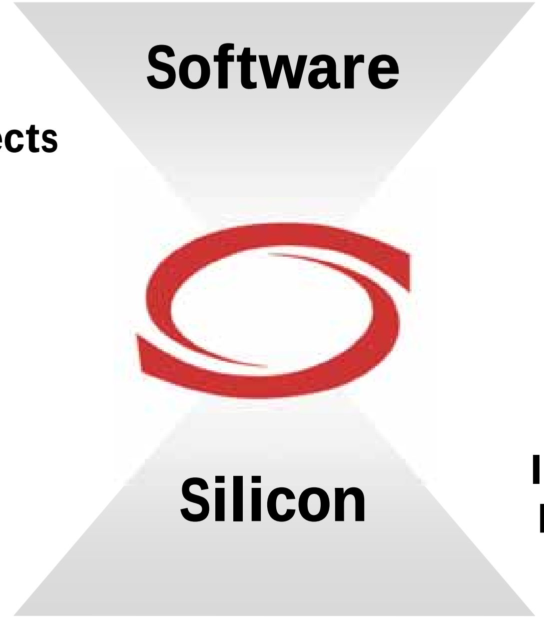
Agenda

- **Introduction to Khronos and open standards**
  - Who is the Khronos Group
  - What are royalty-free open standards
  - Industry collaboration
- **Developing safety critical technology standards**
  - Case study: Developing OpenGL SC
  - Lessons Learned
- **Ways forward**
  - New ways of working
  - Safety Critical roadmap

Khronos Connects Software to Silicon

Industry Consortium creating OPEN STANDARD APIs for hardware acceleration
Any company is welcome – one company one vote

ROYALTY-FREE specifications
State-of-the art IP framework protects members AND the standards



Conformance Tests and Adopters Programs for specification integrity and cross-vendor portability

Low-level silicon APIs needed on almost every platform: graphics, parallel compute, rich media, vision, sensor and camera processing

International, non-profit organization
Membership and Adopters fees cover operating and engineering expenses

Strong industry momentum
100s of man years invested by industry experts
Well over a *BILLION* people use Khronos APIs *Every Day…*

BOARD OF PROMOTERS

Over 100 members worldwide

any company is welcome to join

Khronos Standards for Advanced Processing

COLLADA glTF

3D File formats for
AUTHORING and TRANSMISSION
of 3D runtime assets

OpenVX

Low-power vision processing
for tracking, odometry and
scene analysis

Image: ScreenMedia

WebGL OpenGL|ES Vulkan OpenGL

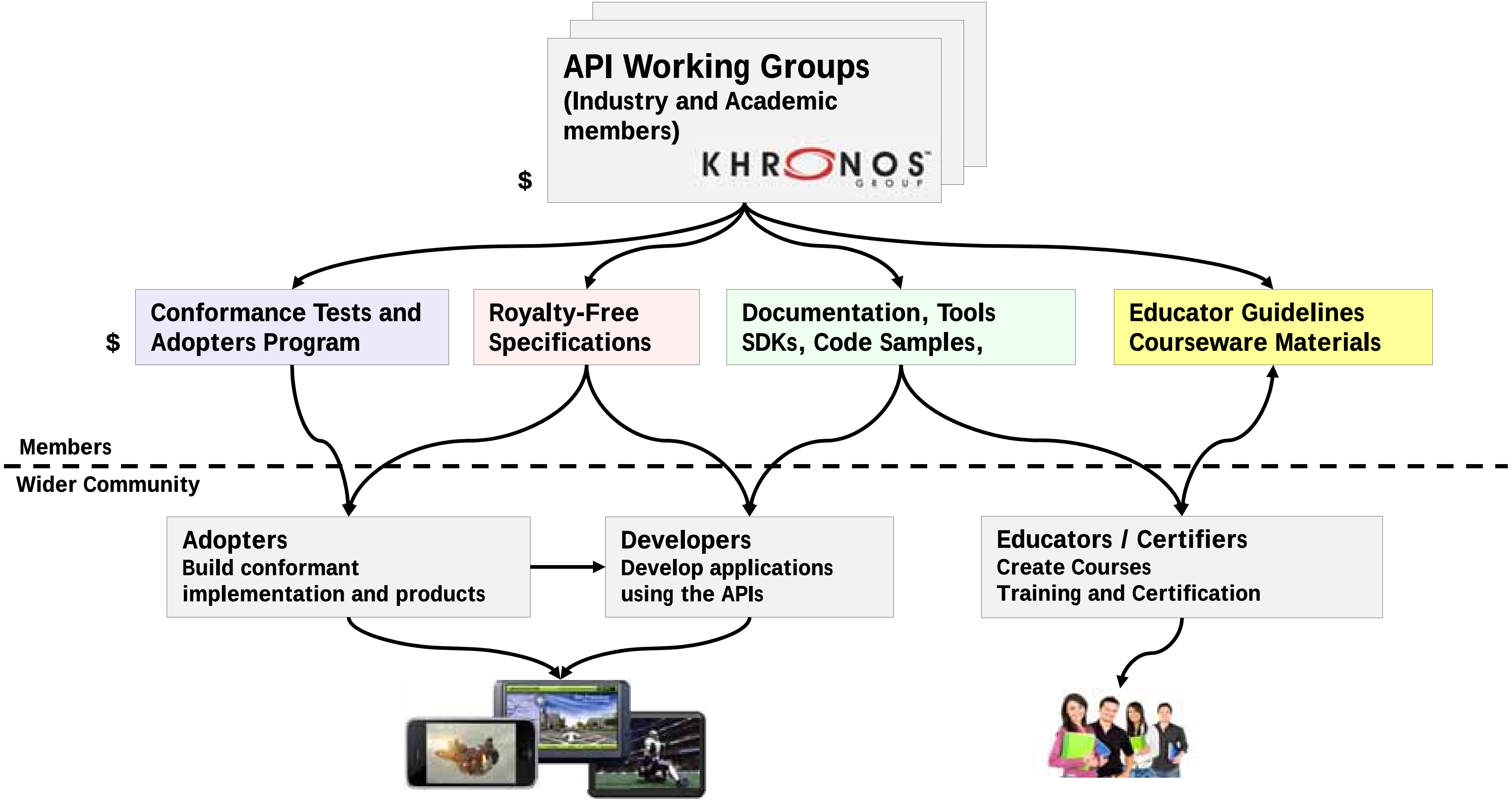
3D Graphics for
Portable display of augmentations
and visualizations on every platform

SPIR OpenCL

Heterogeneous Processing Acceleration
e.g. Neural Net Processing for scene understanding

http://accelerateyourworld.org/

Khronos Cooperative Framework



The Value of Khronos Participation

See an early window into the future of the industry technology roadmap before products are developed

Have a voice in how key standards evolve to suit *your* business needs

Develop products *in parallel* with spec drafting for faster time to market

Products are aligned with global market needs and trends

Members can ship products faster than non-members

Gather Industry Requirements for future silicon acceleration → Draft Specifications Confidential to Khronos members → Publicly Release Specifications and Conformance Tests → Non-members Release Products

The Khronos standardization process is proven to RAPIDLY generate industry consensus on future hardware acceleration functionality to EFFICIENTLY create new market opportunities

KHRONOS GROUP

Agenda

- Introduction to Khronos and open standards
  - Who is the Khronos Group
  - What are royalty-free open standards
  - Industry collaboration
- **Developing safety critical technology standards**
  - Case study: Developing OpenGL SC
  - Lessons Learned
- Ways forward
  - New ways of working

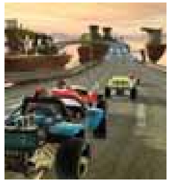
Standards for Safety Critical Technologies - Why?

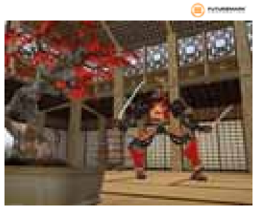
- **Allows use COTS instead of proprietary hardware**
  - Take advantage of mainstream technologies
  - Reduce time to market
- **Existing mobile hardware is well suited to safety critical systems**
  - High performance & Low power usage
  - Cost effective
- **Safety Critical standards optimize for the validation and certification processes**
  - Reduce certification costs while retaining system performance

Announcement of new Pope in St. Peters Square

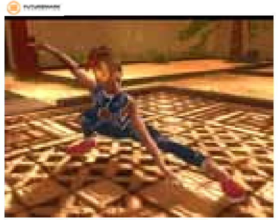
OpenGL ES Roadmap

Fixed function Pipeline





Programmable Vertex and fragment shaders



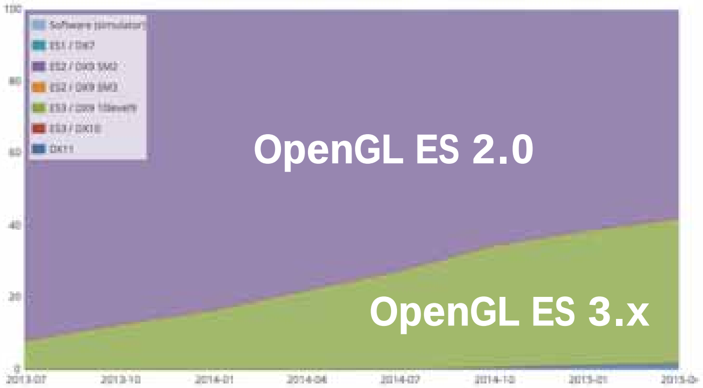
32-bit integers and floats
NPOT, 3D/depth textures
Texture arrays
Multiple Render Targets

Compute Shaders

Tessellation and geometry shaders
ASTC Texture Compression
Floating point render targets
Debug and robustness for security

Epic's Rivalry demo using full Unreal Engine 4
https://www.youtube.com/watch?v=jRr-G95GdaM

2003	Driver Update	2004	Silicon Update	2007	Silicon Update	2012	Driver Update	2014	Silicon Update	2015
1.0		1.1		2.0		3.0		3.1		3.2



http://hwstats.unity3d.com/mobile/gpu.html

The most widely deployed 3D graphics API in history
Industry shipped >1.9 billion OpenGL ES-enabled devices in 2015

Safety Critical Working Group

2003
OpenGL ES 1.0
Fixed function graphics

2005
OpenGL SC 1.0
Fixed function graphics subset

2007
OpenGL ES 2/3
Programmable shader pipeline

2016 (planned)
OpenGL SC 2.0
Programmable shader pipeline subset

New Generation API for safety certifiable graphics, vision AND compute

Many future safety critical use cases involve vision and compute acceleration (e.g. neural nets)

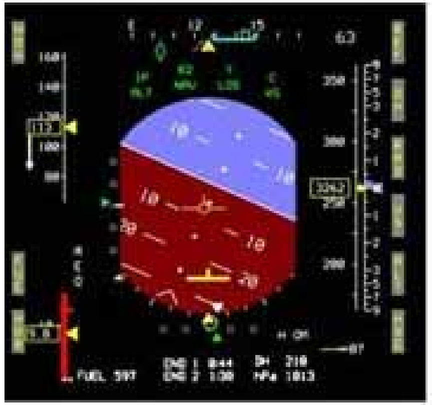
Develop Standards for Safety Critical Technologies - How?

- **The Challenge: How do you develop an open standard for a safety critical technology?**
  - Not what safety requirements the technology has to fulfill, but rather how do you develop a technology specifically adapted to the needs of the safety critical community
- **The Case Study: The implementation of an open, royalty-free 3D-graphics standard specifically designed for use in Safety Critical systems**
  - OpenGL SC 2.0

Getting Started

- **Khronos released OpenGL SC 1.0 in 2005**
  - OpenGL SC was created specifically for use in Safety Critical systems
    - Target industries avionics and automotive

2005(!)

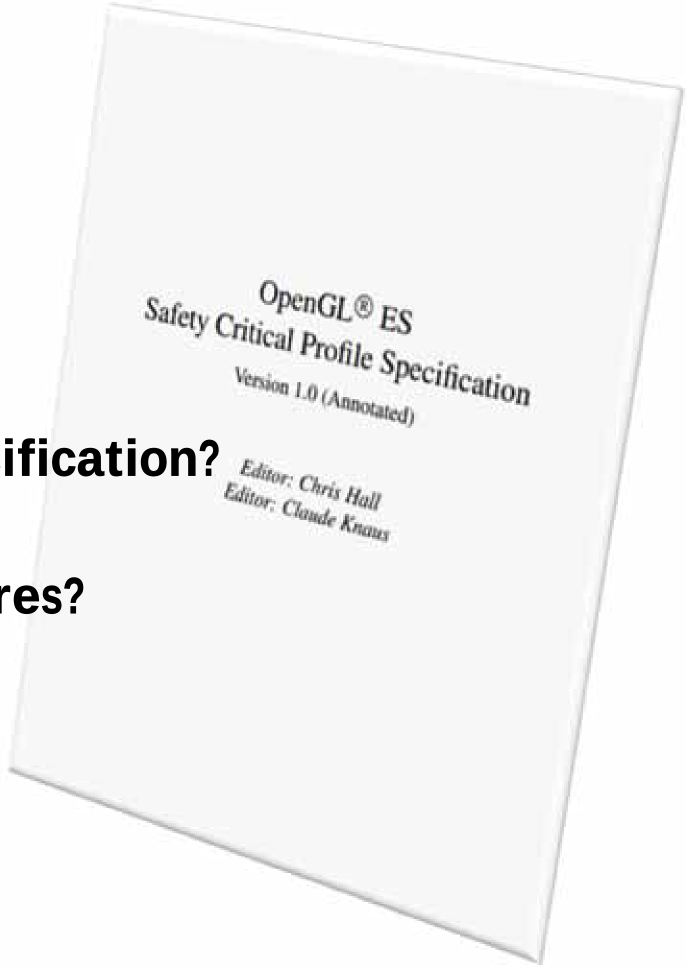


- **Industry demand prompted the start of development of Open GL SC 2.0 in 2015**
  - Target industries include avionics and automotive, and is designed for use in any safety critical application

- **Other Khronos working groups are looking at Safety Critical versions of technologies**
  - Experiencing the same questions and difficulties that the Safety Critical working group has encountered

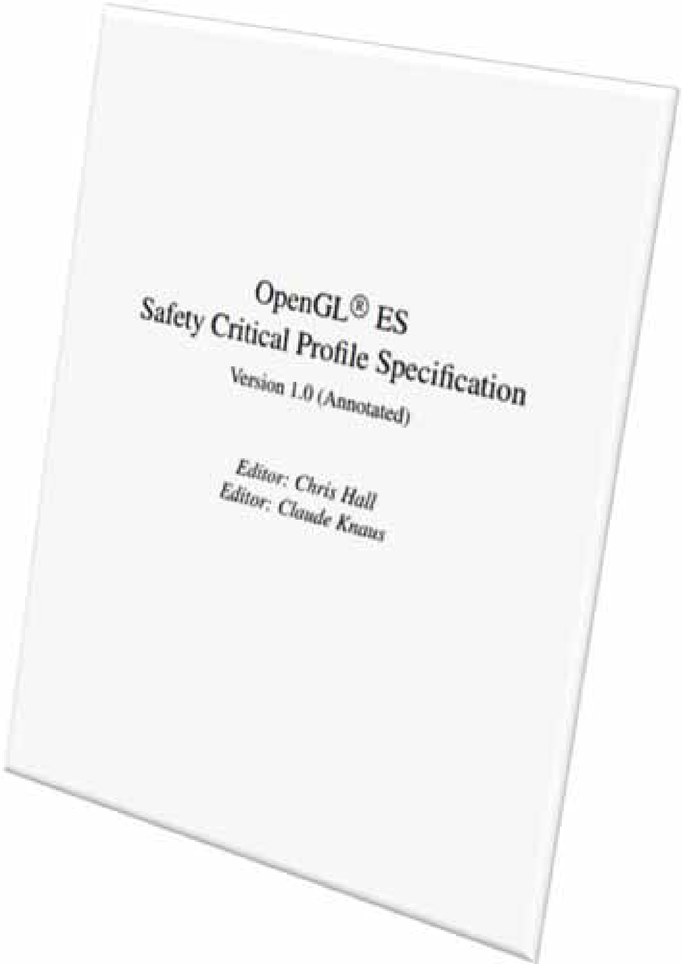
Lessons Learned from OpenGL SC 1.0 / 2.0

- New technology or based on existing standard?
- How do you relate to industry standards?
- Differential or full spec?
- Write specification as a behavioral or functional specification?
- Backwards compatible or change behavior or signatures?
- What input from stakeholders?
- Conformance tests for specification?



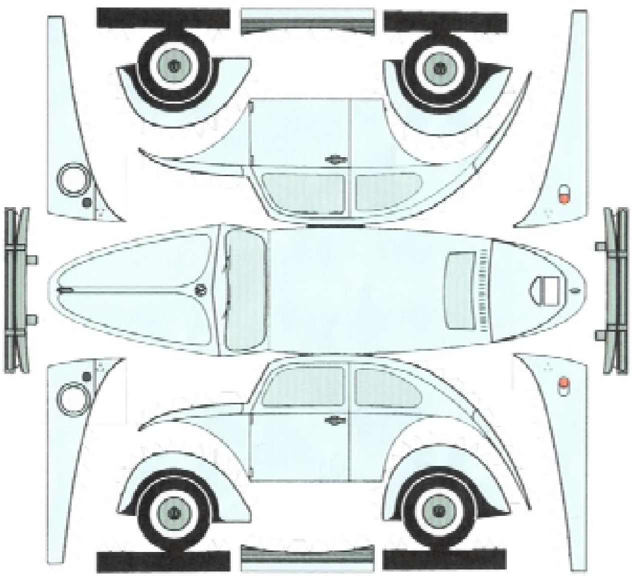
Lessons Learned from OpenGL SC 1.0 / 2.0

- **New technology or based on existing standard?**
  - Questionable if market can bear the development costs of a new technology specifically developed for Safety Critical systems
  - Basing SC standard on existing standard significantly reduces time to market

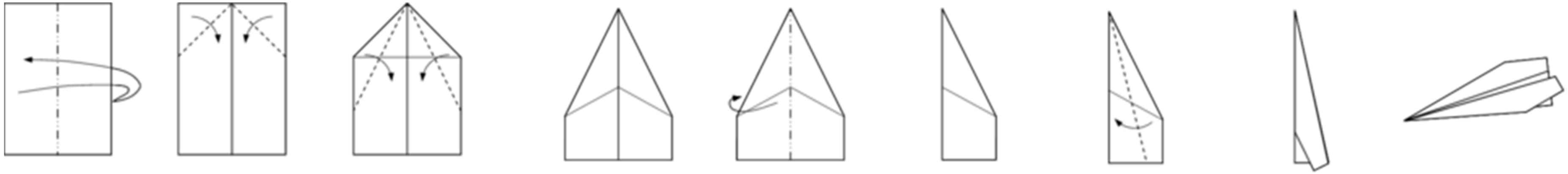


Lessons Learned from OpenGL SC 1.0 / 2.0

- **How do you relate to industry standards?**
  - ISO 26262 – Automotive
    - Industry recommended
  - DO 178 – Avionics
    - Legally mandated



  - Specification work has to take both types into account



Lessons Learned from OpenGL SC 1.0 / 2.0

- **Differential vs. full spec?**
  - OpenGL SC 1.0 was based on OpenGL ES 1.0
    - OpenGL SC 1.0 was created as a differential specification
  - OpenGL SC 2.0 is based on OpenGL ES 2.0
    - OpenGL SC 2.0 is being created as a full specification
  - Decision was made that more people would use a full specification
    - Easier to use during validation and certification

Lessons Learned from OpenGL SC 1.0 / 2.0

- **Specification written as a behavioral or functional specification?**
  - A behavioral specification is good for understanding system
    - Necessary for developers and system designers
  - A functional specification is good for requirements
    - Reduces certification and validation costs
    - Common set of requirements across systems
  - OpenGL ES 1.1 was written as a behavioral specification
    - As is OpenGL SC

Lessons Learned from OpenGL SC 1.0 / 2.0

- **Backwards compatible or change behavior or signatures?**
  - Tempting to tighten up loose behavior and signatures when building Safety Critical specification
    - But can wreak havoc on an implementation
  - Backwards compatible makes it easier to develop systems
    - Reduces need to spot modify drivers – only modify to remove additional functionality
  - Non-backwards compatible behavior *could* impact certain hardware designs
    - Limiting use of specification

Lessons Learned from OpenGL SC 1.0 / 2.0

- **What input from stakeholders?**
  - Important to get early input from both SC and non-SC experts
    - Design and constraint input
  - Industry regulatory and certification hurdles
    - Can be facilitated in specification
  - Validation and certification requirements
    - Sometimes more important than functionality
  - Functionality requirements
    - Need to be evaluated based on certification requirements

Lessons Learned from OpenGL SC 1.0 / 2.0

- **Conformance tests for specification?**
  - What should they test and what should they accomplish?
    - Should they evaluate the compliance with the specification?
    - Should they assist in validation and certification process?
  - What kind of systems should they run on?
    - What are the constraints of those systems?
  - Who writes the tests?
    - A single company writing tests may favor their implementation
  - Who validates the results of the tests?
    - Need to leave industry competition out of the process

Lessons Learned from OpenGL SC 1.0 / 2.0

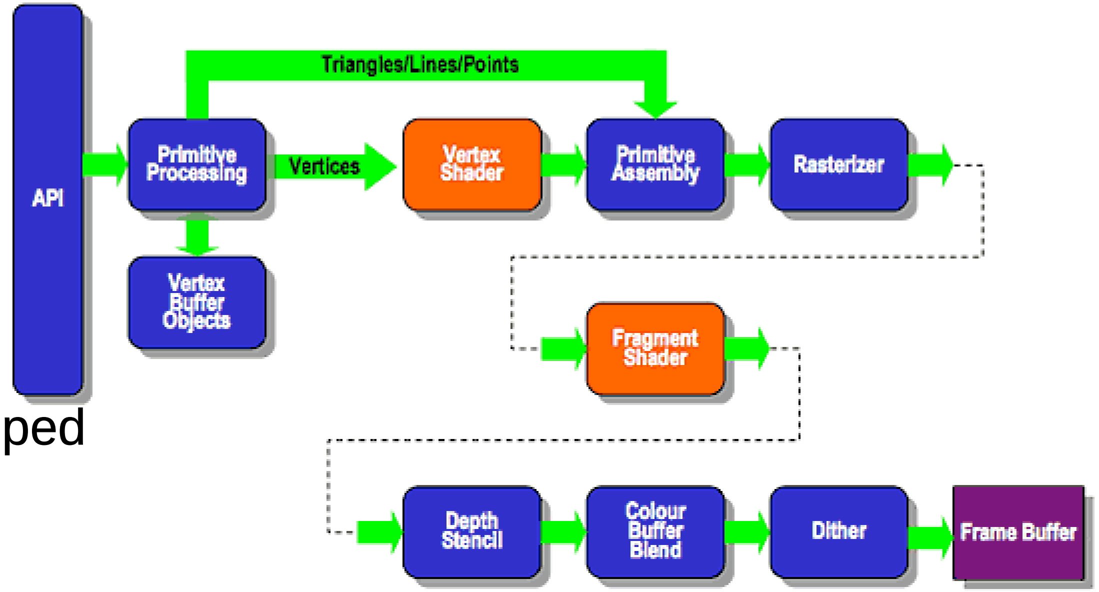
- **New technology or based on existing standard?**
- **How do you relate to industry standards?**
- **Differential or full spec?**
- **Write specification as a behavioral or functional specification?**
- **Backwards compatible or change behavior or signatures?**
- **What input from stakeholders?**
- **Conformance tests for specification?**

Note:
No discussion about technical content

The result: OpenGL SC 2.0 (planned 2016)

- **Contains most of the functionality of OpenGL ES 2.0**
  - Programmable pipeline
  - Robustness extension incorporated into core specification
  - Debug functionality removed

ES2.0 Programmable Pipeline



- **Backwards compatible with OpenGL ES 2.0**
  - Allows use of existing OpenGL ES 2.0 hardware

- **Conformance tests**
  - A full suite of tests is being developed
  - Adoption program for conformant implementations

Agenda

- **Introduction to Khronos and open standards**
  - Who is the Khronos Group
  - What are royalty-free open standards
  - Industry collaboration
- **Developing safety critical technology standards**
  - Case study: Developing OpenGL SC
  - Lessons Learned
- **Ways forward**
  - New ways of working
  - Safety Critical Roadmap

Safety Critical Advisory Panel

- **The Safety Critical working group found that there was a need for industry input that was not related to technology**
  - Get input not related to a specific technology
- **The Khronos Group created the Safety Critical Advisory Panel**
  - Open to participation without IP commitment
  - Goal is to develop a set of guidelines to aid in the development of open technology standards for Safety Critical systems
  - Discussion forum for Lessons Learned

Safety Critical Advisory Panel

- **The Safety Critical Advisory Panel provides feedback to main working groups**
  - Also in an IP-free arena
  - Facilitates the creation of the next generation of Safety Critical standards

- **The Safety Critical Advisory Panel bridges safety critical community with mainstream industry**
  - Makes companies not involved in Safety Critical aware of issues of Safety Critical development

Roadmap **Possibilities**

SPIR-V Ingestion for OpenGL and OpenGL ES for shading language flexibility

Thin and predictable graphics and compute for safety critical systems

1. C++ Shading Language
2. Single source C++ Programming from SYCL
3. OpenCL-class Heterogeneous Compute to Vulkan runtime

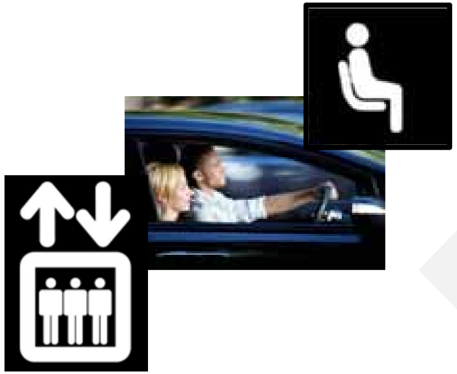
Safety Critical Roadmap

- **Roadmap driven by industry need**
  - Members decide what to develop and when

- **2016**
  - Expect to release of 3 Safety Critical standards
  - Including OpenGL SC 2.0
- **2017**
  - Expect to release at least 2 Safety Critical standards

Low-level Sensor Abstraction API

Apps Need Sophisticated Access to Sensor Data
Without coding to specific sensor hardware

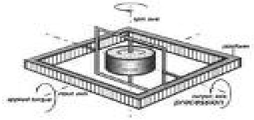


Apps request semantic sensor information
StreamInput defines possible requests, e.g.
Read Physical or Virtual Sensors e.g. “Game Quaternion”
Context detection e.g. “Am I in an elevator?”

StreamInput™

Sensor Discoverability
Sensor Code Portability

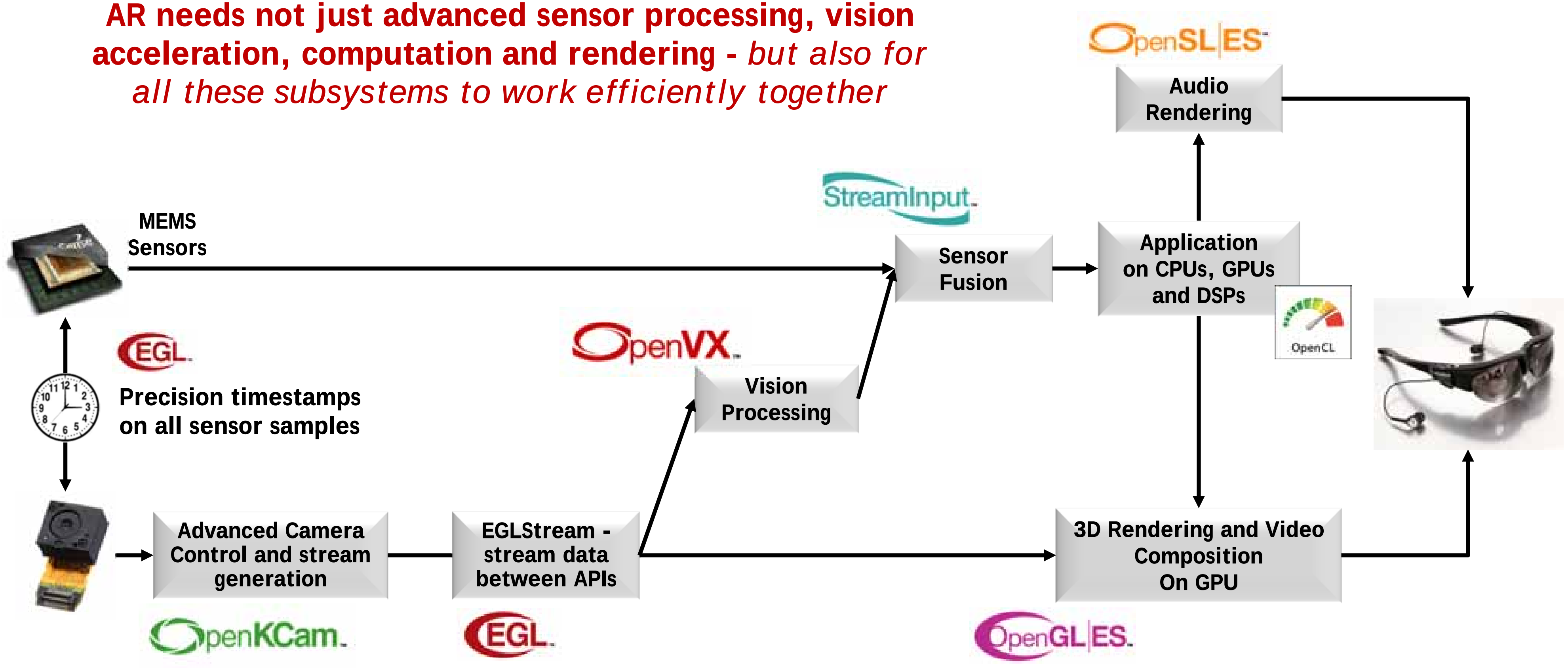
StreamInput processing graph provides optimized sensor data stream
High-value, smart sensor fusion middleware can connect to apps in a portable way
Apps can gain ‘magical’ situational awareness



Advanced Sensors Everywhere
Multi-axis motion/position, quaternions, context-awareness, gestures, activity monitoring, health and environmental sensors

Khronos APIs for Augmented Reality

AR needs not just advanced sensor processing, vision acceleration, computation and rendering - ***but also for all these subsystems to work efficiently together***



Finally...

- **Khronos is creating cutting-edge royalty-free open standards for Safety Critical**
  - For graphics, vision and parallel computation
  - Adoption programs to facilitate system conformance to specifications
- **Khronos standards are key to many safety critical markets such as avionics, automotive and automation**
  - Advanced next generation capabilities for ALL safety critical platforms
- **Any company is welcome to join Khronos influence the direction of these important international standards**
  - $15K annual membership fee for access to all Khronos API working groups
  - Well-defined IP framework protects your IP and conformant implementations
- **More Information**
  - www.khronos.org
  - erik@noreke.se

Questions, Comments or Coffee?