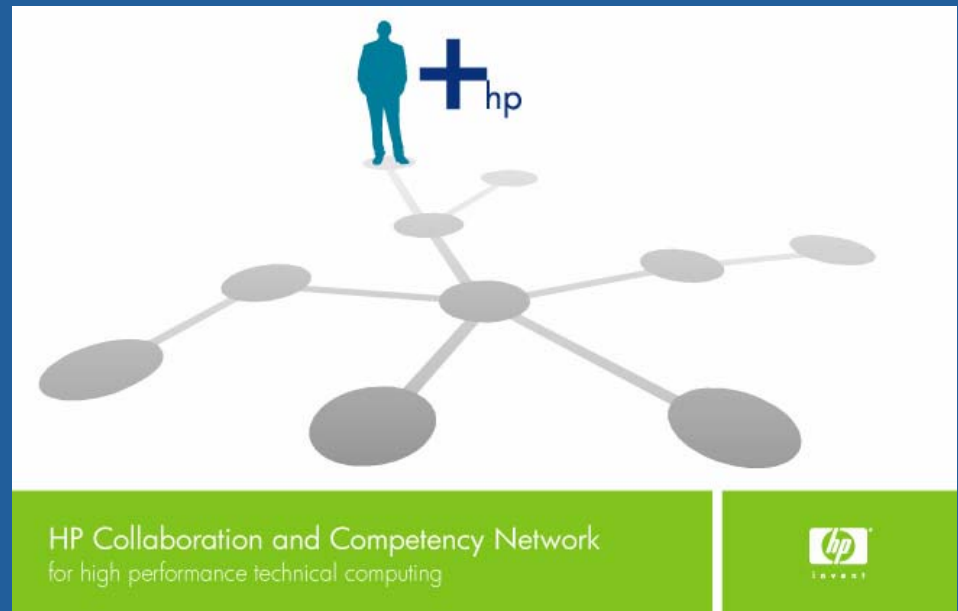


HP CCN Visualization July Meeting Slides



July 27, 2005

The information contained in this presentation is subject to change and is not a commitment by HP to provide any program or product described herein. This information is not considered confidential information.

HP makes no warranties regarding the accuracy of this information. HP does not warrant or represent that it will introduce any product or program to which the information relates. It is presented only for evaluation by the recipient and to assist HP in defining direction.

(c) Copyright 2005 Hewlett-Packard Development Company, L.P.

Background on Presentation



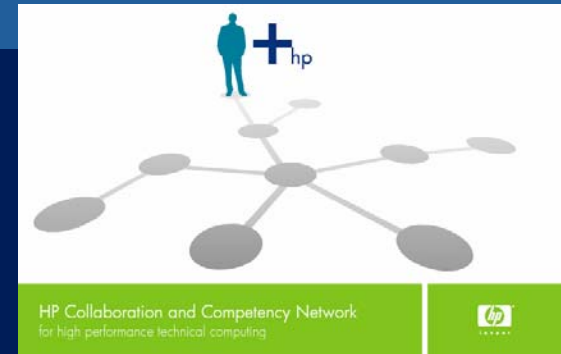
HP is interested in collaborating with users and ISVs on an API and implementation for image / pixel compositing.

The following slides contain a proposal on this subject that was shared with a number of HP collaborators. As a recipient, you may freely distribute the presentation to others, provided you send the presentation in its entirety.

If you'd like to know more about HP's efforts in this area, please contact:

Steve Greenwood (steve.greenwood@hp.com)

Agenda For the Meeting



Introductions

HP's Proposed Compositing Strategy

- The Numbers (15 minutes)
 - A set of measured numbers that lead to our strategy
- The Requirements (20 minutes)
 - What we see as the goals and requirements
- The Framework (10 minutes)
 - How we propose to meet the requirements

Feedback & Next Steps

Numbers from Image Compositing Prototype



- Image compositing using COTS hardware can be:
 - CPU-based – use the CPU to compute resulting image
 - GPU-based – use the graphics card to compute resulting image
- Compositing on a workstation cluster is not a new idea
 - The well-known Binary-Swap method was described in Parallel Volume Rendering Using Binary-Swap Image Composition [Ma et al. 1994]

Image Compositing on a Cluster



- Main operations
 - Reading the image data from the graphics card
 - Receiving image data from another node
 - Combining the local and remote images
 - Sending the result to another node
- Combining the images may be done:
 - Using the CPU
 - Read the local image data from the graphics card
 - Use the CPU to compute the result-image
 - Using the GPU
 - Copy the remote image-data to the graphics card
 - Use the GPU to compute the result-image
 - Read the result-image from the graphics card

Image Compositing Using COTS Hardware



- Traditionally has been limited by:
 - Slow image data readback times
 - Limited affordable network bandwidth
- Traditional Example: 1280x1024 RGBA image
 - Readback - 200-250MB/sec : 21 to 26 msec
 - Significant amount of time when attempting to render at 15-20fps or 50-67 msec per frame
 - Receiving and sending image – GigE 85MB/sec bidirectional
 - Cluster node must receive and send
 - 85MB/sec through the node = 16 fps

Improvements in Graphics Card Readback



- Over the past year, measured readback times for 1280x1024 RGBA have improved from:
 - 235 MB/sec, to
 - 760 MB/sec with the latest driver
- These are using the same workstation model and graphics card:
 - HP xw8200 with Nvidia Quadro FX 3400 (PCI-E 16x)
- Readback of 1280x1024 RGBA can now be done in 6.9 msec vs. 22 msec
- Graphics hardware will get faster
 - Readback only running at 25% of PCI-E 16x bus speed

Improvements in COTS Networking

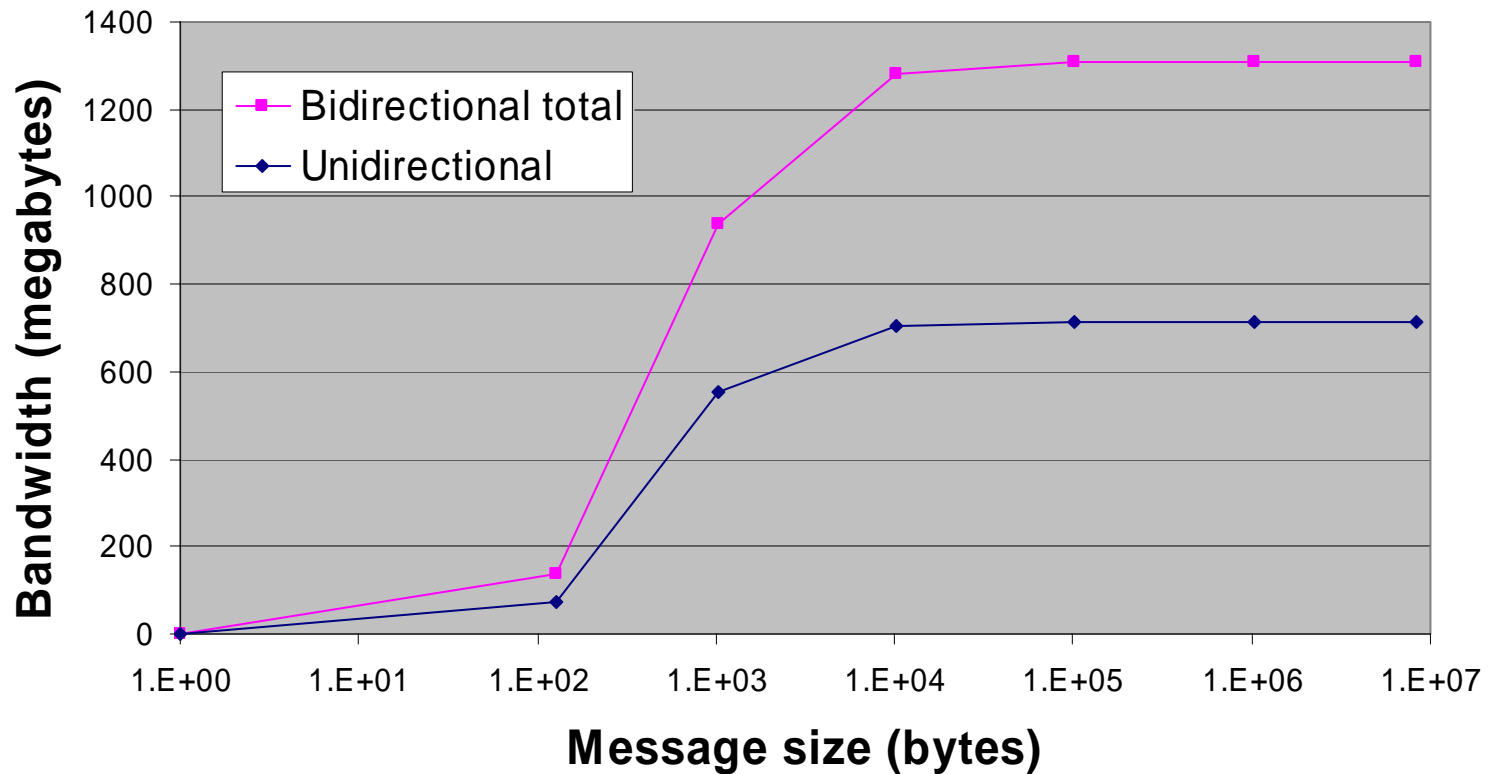


- GigE
 - 85 MB/sec bidirectional (170 MB/sec total)
 - 115 MB/sec bidirectional with parameter tuning and 2.6 Linux kernel
- Infiniband
 - 650 MB/sec bidirectional (1300 MB/sec total) using PCI-E 4x slot
 - Image data (1280x1024 RGBA) can theoretically flow through a node at 124 fps
- Networking hardware will get faster
 - Infiniband Double data rate (4x => 8x)
 - PCI-E 8x slots available in machines

Infiniband Bandwidth



Infiniband Bandwidth on xw8200 PCI-E 4x slot



Compositing Time



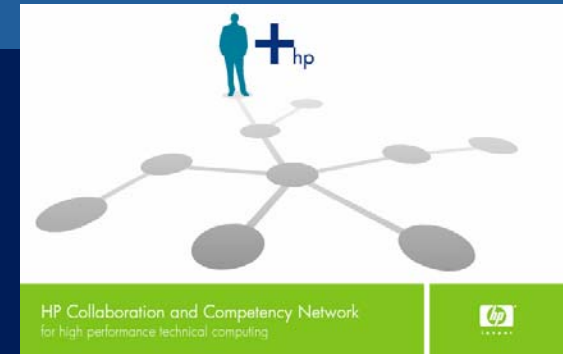
- Compositing a 1280x1024 image using HP xw8200 with Nvidia Quadro FX 3400
- Compositing time includes readback
- Using GPU-based compositing
 - Serially with application rendering
 - Depth compositing – 30.3 msec
 - Alpha compositing – 15.0 msec
- Using CPU-based compositing
 - In parallel with application rendering, except for readback
 - Depth compositing – 18.8 msec (incl 13.8msec readback)
 - Alpha compositing – 11.3 msec (incl 6.9msec readback)

Prototype Demo Application Results



- HP xw8200, FX 3400, IB
- Free running animation vs. mouse driven rendering
- Volume rendering – uses CPU-based alpha blending
 - 512x512x512 model
 - 1280x1024 image
 - 8 nodes
 - Result: 28.2 fps
- Polygon rendering – uses CPU-based depth compositing
 - 28M triangle model
 - 1280x1024 image
 - 8 nodes
 - Result: 20.6 fps (577 Mtris/sec)

Agenda



Introductions

HP's Proposed Compositing Strategy

- The Numbers (15 minutes)
 - A set of measured numbers that lead to our strategy
- The Requirements (20 minutes)
 - What we see as the goals and requirements
- The Framework (10 minutes)
 - How we propose to meet the requirements

Feedback & Next Steps

Compositing API Goals



- Goals
 - Provide an image compositing API whose
 - Interface is callable from applications and frameworks of varying architectures
 - Implementation is optimized for HP HPC Linux cluster platforms
 - Performance meets the needs of scientists and engineers
 - Internal design allows for incremental enhancement and modular development
 - Serve as the basis of a compositing API standard
 - Can be implemented on a variety of HW/SW platforms
- Non-goals
 - Force adoption of a framework such as VTK or Chromium

General API Design Requirements



- Easily called from a variety of applications (C/C++, pure OpenGL, scene graph)
- Does not place restrictions on the application implementation, for example
 - Does not force application to be an MPI application
 - Does not force application code to be thread safe
- Uses a Push Model
 - Push – the application renders an image and calls the API to perform compositing
 - Pull – the API calls the application to render the image it needs
- Allows for a variety of implementation techniques, for example, how nodes communication or how pixels are compositing
- Provides a mechanism to get diagnostics and errors
- Provides a mechanism to get timing information on operations performed by the API

Compositing Requirements



- Provides depth compositing, alpha blending, spatial compositing, temporal compositing
- Sends results to display device or memory
- Allows for displaying compositing images on multi-tile displays with or without the desktop user interface
- Provides a mechanism to specify compositing order per frame
- Provides choice of precision (8-bit, 16-bit, 32-bit)
- Allows for user-defined compositing operators
- Allows for multi-pass compositing techniques, for example
 - Depth compositing followed by alpha blending
 - Shadow rendering
- Allows for flexible pixel routing to minimize data replication for multi-tile displays

Application Control Requirements



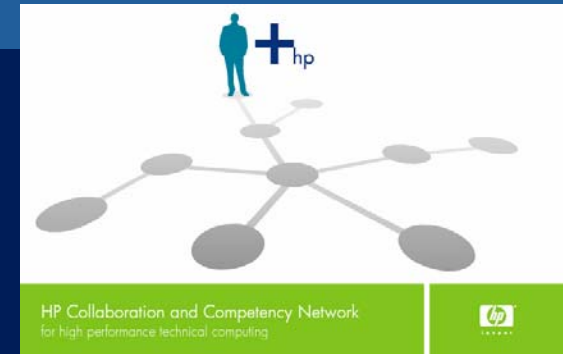
- Allows for specifying buffer attributes (e.g. depth buffer, alpha, stencil)
- Allows for hints
 - To suggest implementation techniques
 - To pass information to the implementation such as bounding box
- Allows for multiple compositing contexts
 - Saves state so the application can easily switch between different sets of settings
- Allows for dynamically changing settings, such as changing resolution to support window resize, and making trade-offs between quality and performance
- Allows the application to do load balancing

API Implementation Requirements



- Abstracts the network layer to support
 - Less expensive, lower performing network (GigE)
 - Higher performing, more expensive network (IB)
 - Other networks as needed
- Allows for dedicated network
- Allows for various compositing models
 - Binary-swap, pipeline, or other compositing strategy
- Can take advantage of multiple graphics cards per node

Agenda



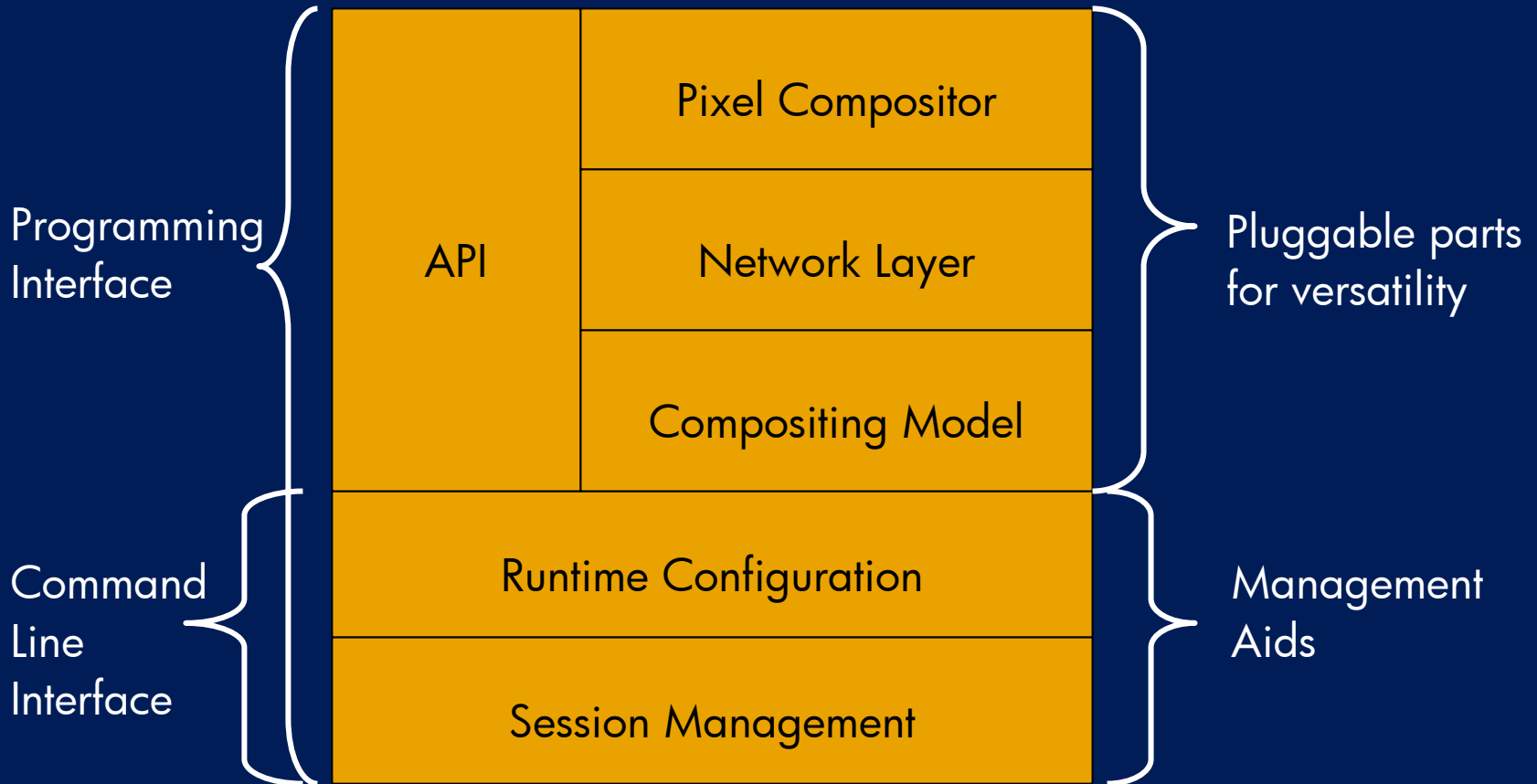
Introductions

HP's Proposed Compositing Strategy

- The Numbers (15 minutes)
 - A set of measured numbers that lead to our strategy
- The Requirements (20 minutes)
 - What we see as the goals and requirements
- The Framework (10 minutes)
 - How we propose to meet the requirements

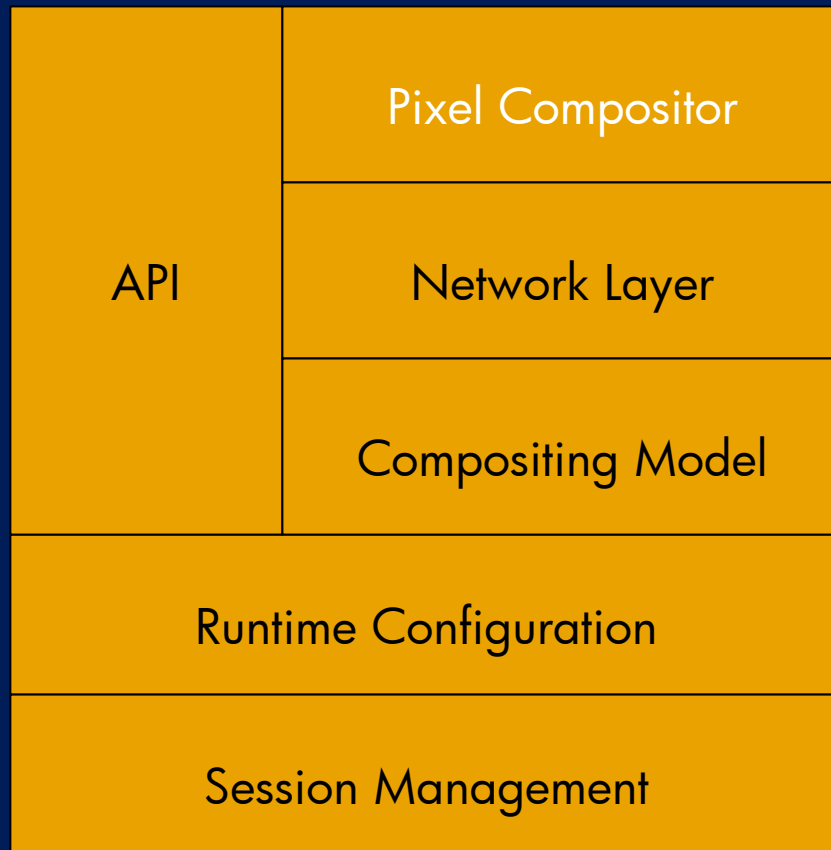
Feedback & Next Steps

The Framework



The API is the primary interface used by an application. The API provides the overall structure for using the pluggable parts and interfaces with the management aids.

The Framework – Pixel Compositor



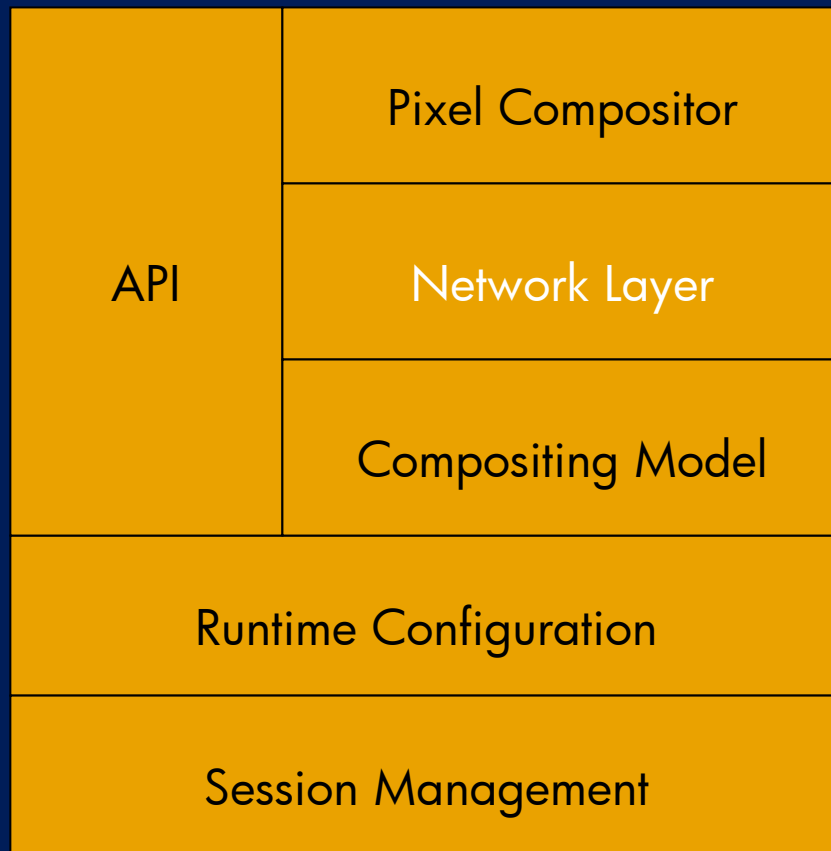
Pixel Compositor

A pixel compositor takes 2 buffers of pixels and combines them to produce the composited result

Pixel Compositors could come in multiple flavors

- A GPU-based compositor
- A CPU-based compositor
- Tuned for a specific graphics card

The Framework – Network Layer



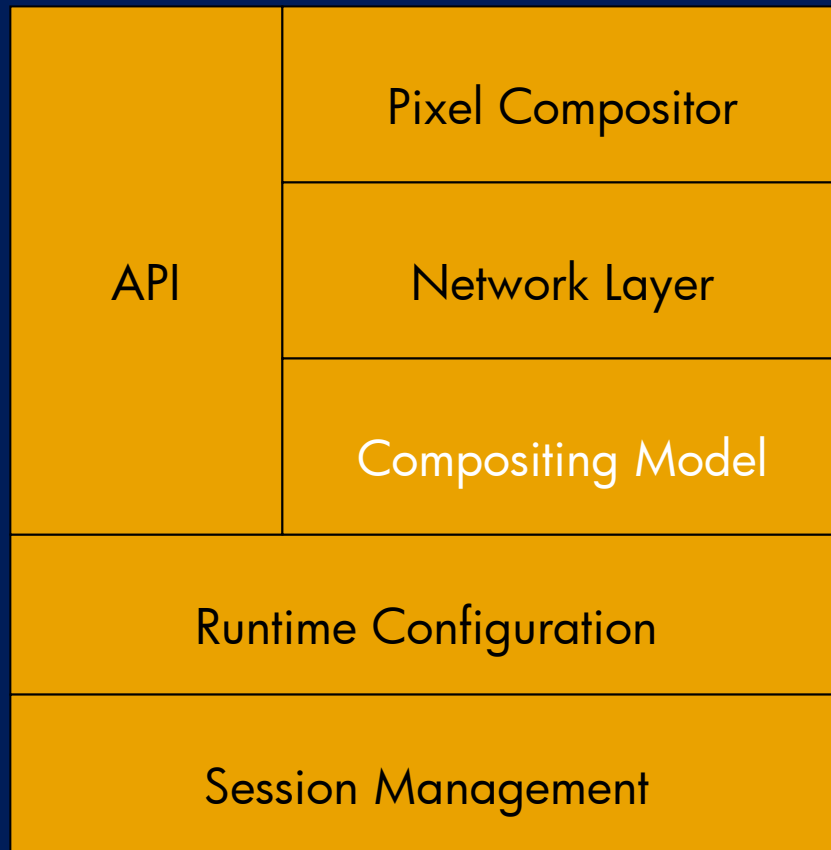
Network Layer

The network layer abstracts how pixels are moved among the nodes

The network layer needs to accommodate

- TCP / IP network (GigE)
- VIA style networks (IB)
- Dedicated network

The Framework – Compositing Model



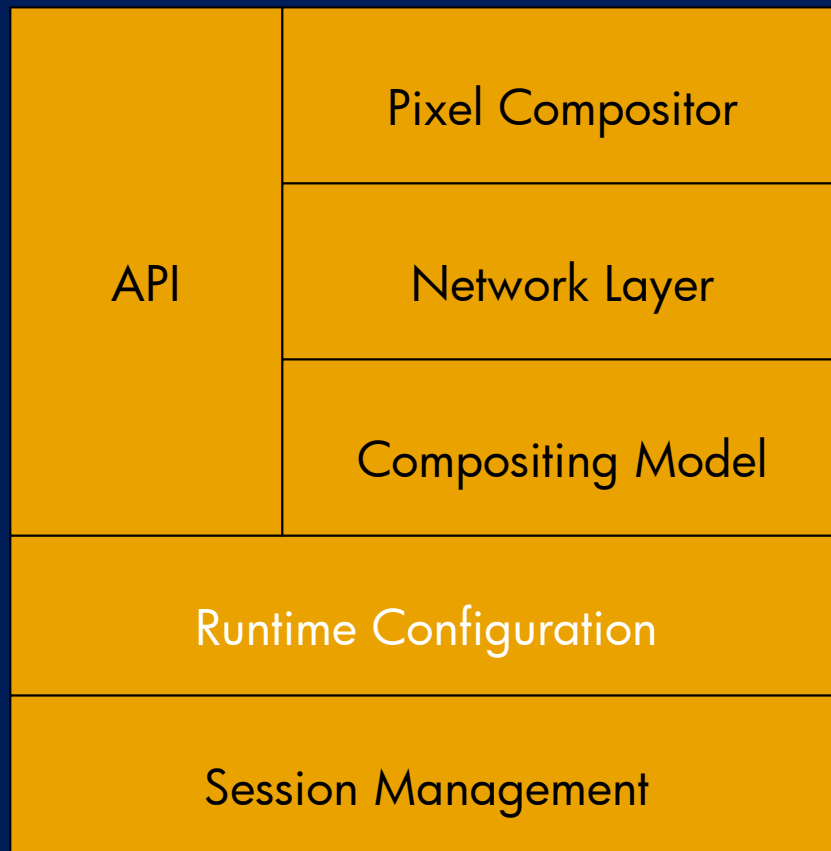
Compositing Model

The Compositing Model is a set of rules for driving the network layer and pixel compositor

The Compositing Model should support

- Binary-Swap
- Sepia Pipeline
- Spatial
- Temporal
- Multi-pass

The Framework – Runtime Configuration



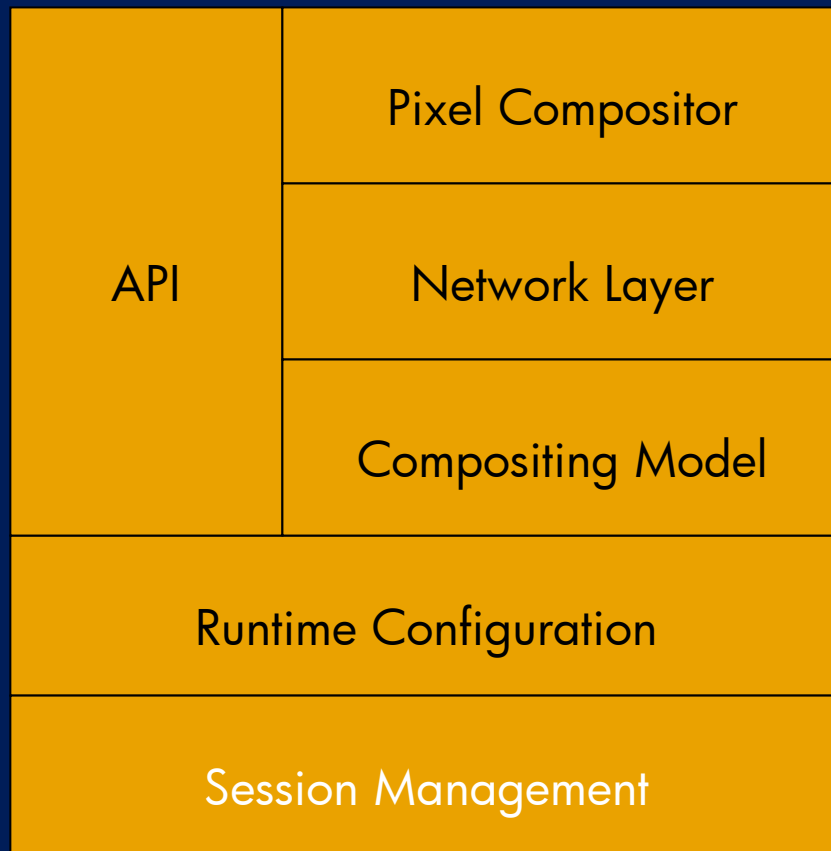
Runtime Configuration

Runtime Configuration abstracts the program from user and system changes

Runtime Configuration is an optional component that should support

- Working with different size displays
- Changing the pixel compositor used
- Changing the compositing model used
- Varying the number of nodes used to composite

The Framework – Session Management



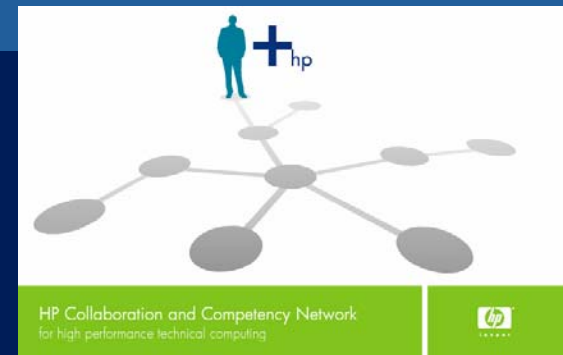
Session Management

Session Management is an optional component that manages a set of nodes

The Session Manager could

- Coordinate the nodes so they processes the correct frame
- Let a node discover the other nodes in a session
- Support exchanging parameters on start-up or on a frame by frame basis
- Do work to optimize the compositing model

Agenda



Introductions

HP's Proposed Compositing Strategy

- The Numbers (15 minutes)
 - A set of measured numbers that lead to our strategy
- The Requirements (20 minutes)
 - What we see as the goals and requirements
- The Framework (10 minutes)
 - How we propose to meet the requirements

Feedback & Next Steps

