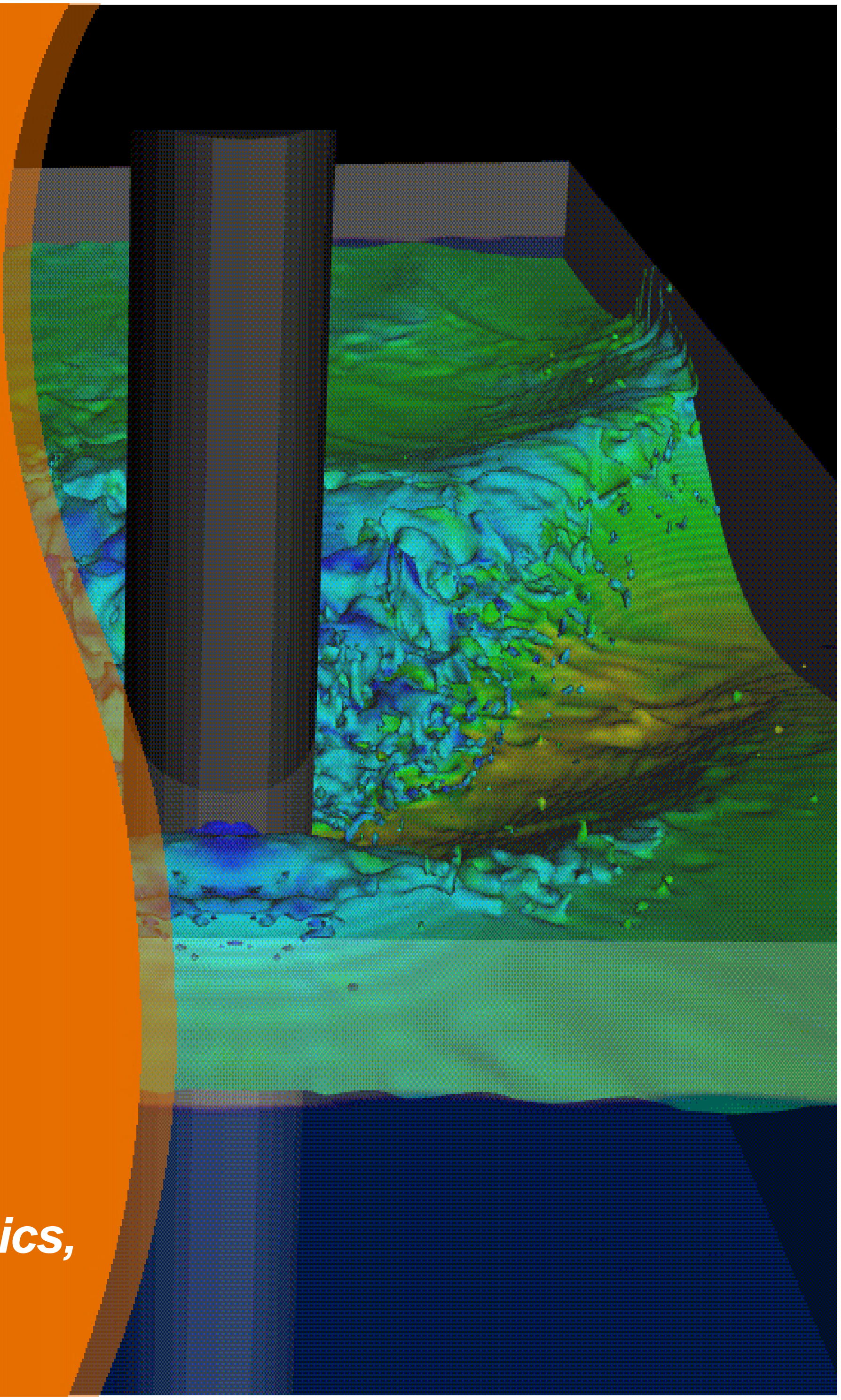
HPC Visualization On the Grid



W. Dean Stanton

*Sr. Staff Engineer, Advanced Visualization,
Sun Microsystems*

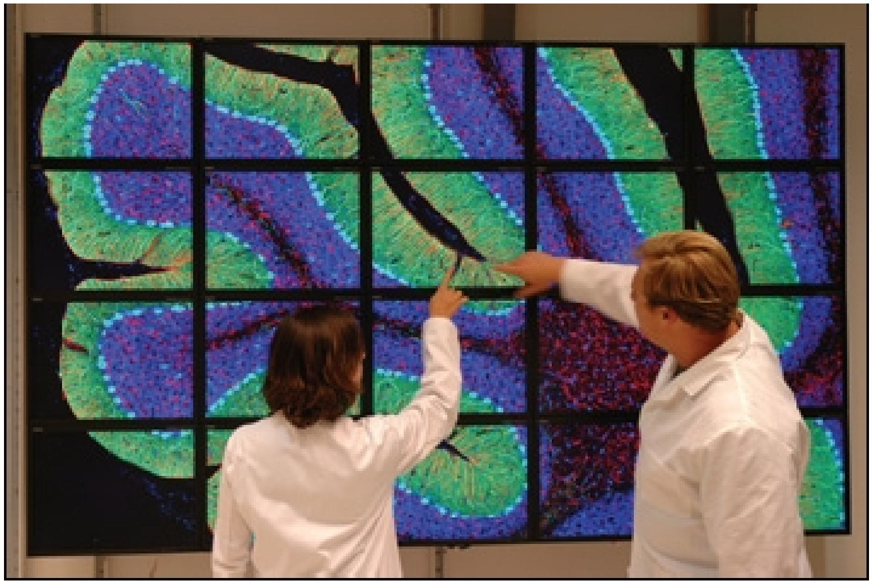
Linda Fellingham, Ph. D.

*Manager, Advanced Visualization and Graphics,
Sun Microsystems*

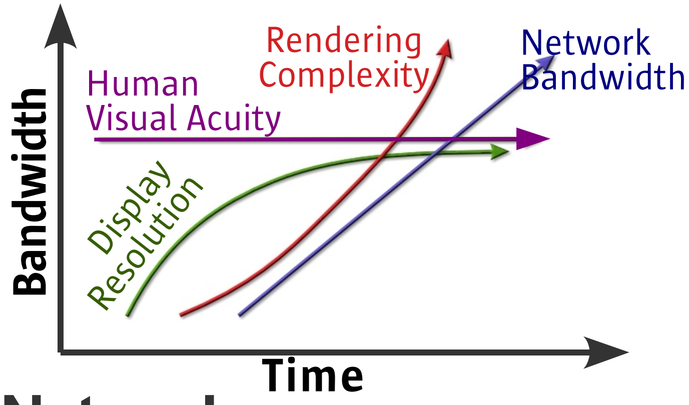
Understanding Huge Data Sets Requires Interactive Visualization

But visualization is hardware intensive

- Big data clogs networks
- Requires lots of memory
- Requires lots of CPU power
- Graphics accelerators need lots of power and cooling
- Workstations inadequate?
- And who wants to work near the heat and noise?

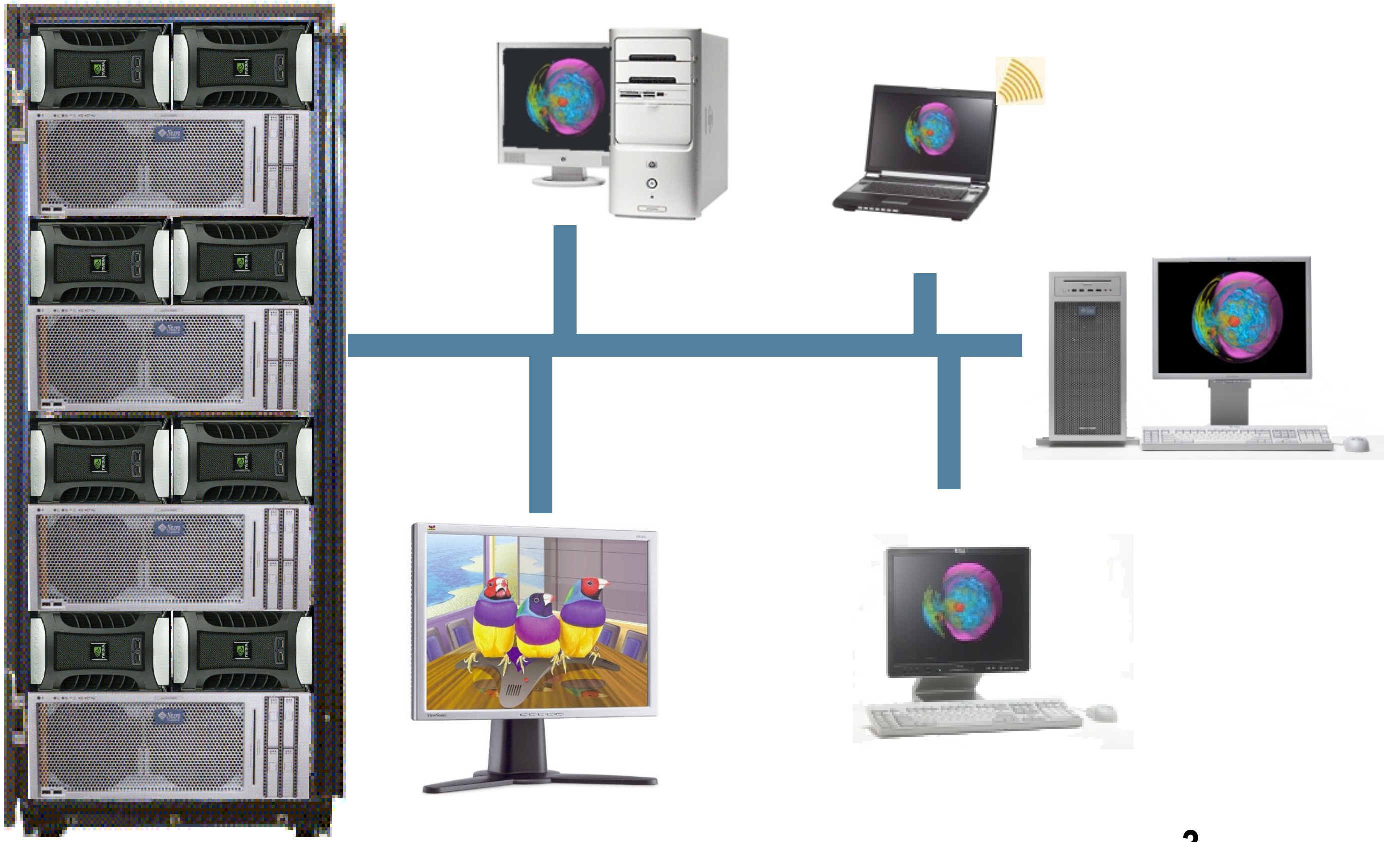


Change the Model



Share Visualization on the Grid over the Network

- Big Data
- Big Memory
- Secure
- Graphics Power
- CPU Power
- Keep heat and noise in the server room
- Send images over network
- Share the cost



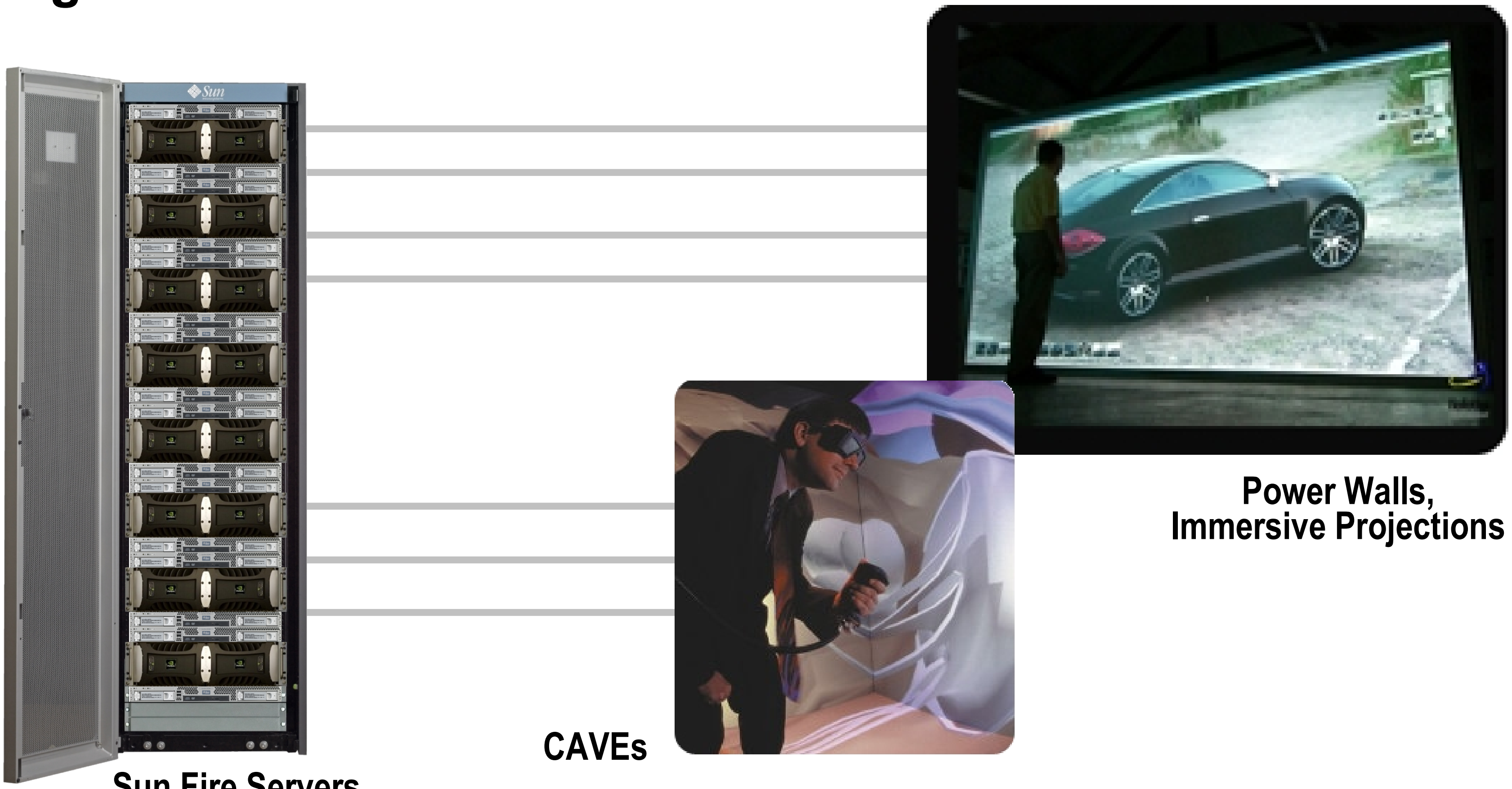
Agenda:

- Scalable Visualization Solutions
- Shared Visualization Software
- Integration with Sun Grid Engine

Sun Scalable Visualization Solutions

Graphics servers support multiple graphics devices to drive:

- Higher performance
- Higher image quality
- Higher resolution



Power Walls, Immersive Projections

CAVEs

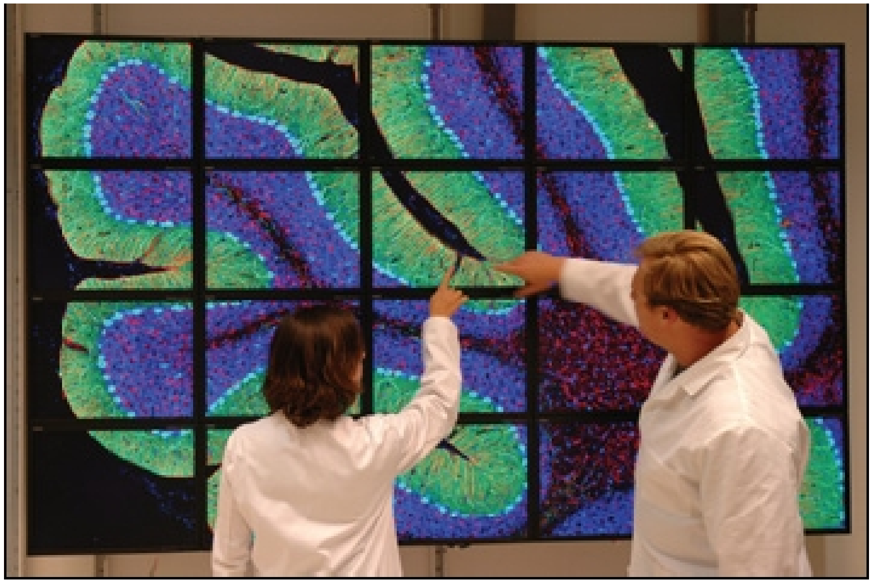
Sun Fire Servers

Scalability Problem 1: Servers

- Servers have lots of processors and memory, but do not have adequate space, power, or cooling for a high-end 3D graphics accelerator
- Solution 1:
  > Get the graphics card(s) out of the system
- Hardware Technology
  > NVidia's Quadro® Plex Visual Computing Systems

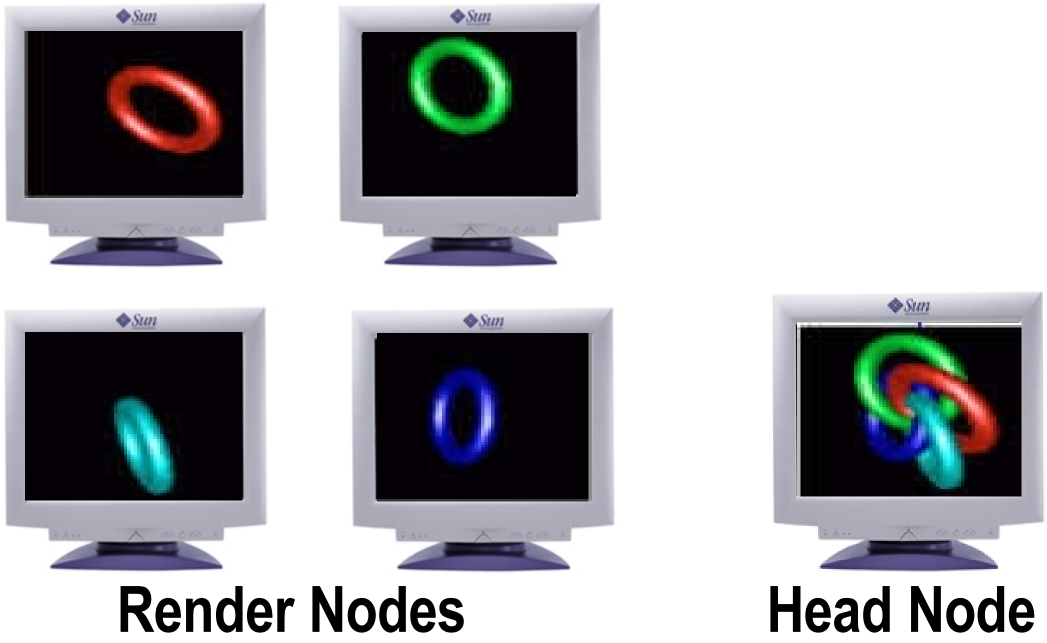
Scalability Problem 2: Lots of Screens

- Need to view applications across many screens, in order to view adequate detail
- Solution 2:
  - Distribute the rendering across many systems (sort first)
  - High-bandwidth, low-latency interconnects (InfiniBand or 10gigE)
- Open Source Software: Chromium or OpenSceneGraph



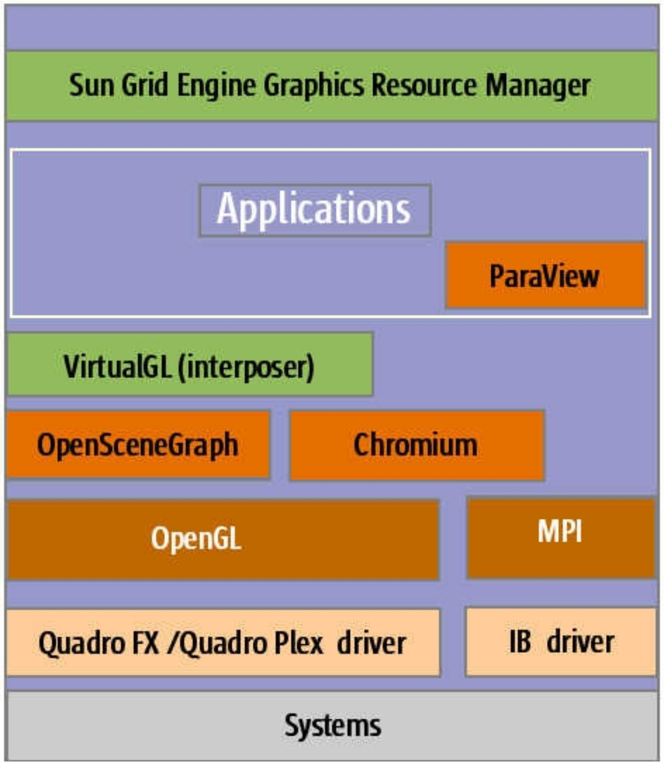
Scalability Problem 3: Performance

- Performance can be too slow for very large data sets
- Solution 3
  - > Break the problem up and distribute the rendering to multiple render nodes, and reassemble on the head node (sort last)
- Open Source Software Technologies:
  - > Paraview (Parallel Visualization Appplication) or Chromium (with work)

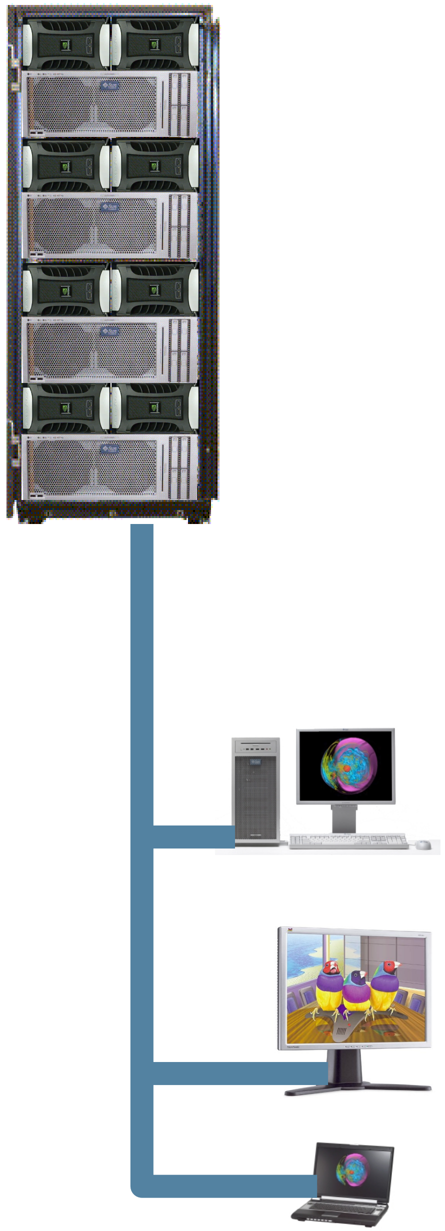


Software Details

- Scalable Visualization 1.1 software supports:
  - > Red Hat Linux (RHEL 4U2-5 and RHEL 5/5.1)
  - > SLES 10
  - > Solaris 10 Update 3 or 4
- Complete Open Source software stack
  - > Chromium, MVAPICH2, OFED, OpenSceneGraph, Paraview
  - > Sun added value:
    - > Pre-built binaries, tested for interoperability
    - > Installation scripts and configuration files
    - > Wrappers for greater ease of use
    - > MPI protocol added as a Chromium interconnect
    - > Supported on Sun hardware
    - > Free download for Solaris at OpenSolaris.org/os/project/visualization-hpc/



Sun Shared Visualization Software

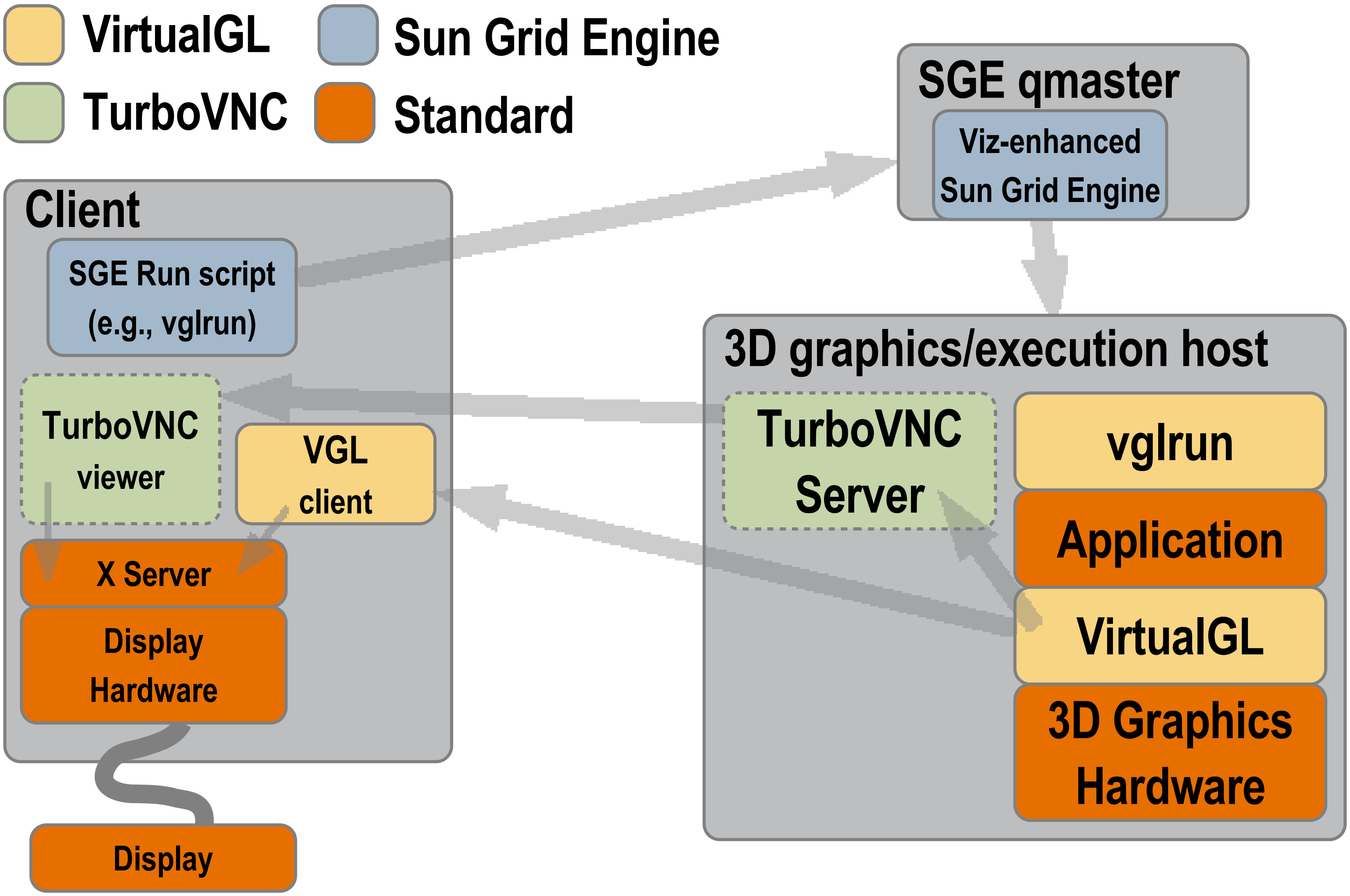


The Shared Visualization Model

Feature	Benefit
Secure	Data stays on the server. Control data access, even among 3D application users.
Performant	Optimized compression and decompression transfers visually lossless images at interactive speeds (more than 20 frames/sec).
Interoperable	Client only needs enough network performance and a display. Interoperable with a variety of devices
Sharable	Average CPU, memory, and graphics needs over many users. Reduce total cost of ownership
Scalable	A single user can access lots of CPU, memory, and attached graphics. Get more resources than possible in ANY workstation
Flexible	Graphics computation and display technology are separated. Display on what you already have, upgrade graphics separately
Load Balancing	Better utilization of compute and graphics resources. Grid software helps to find and manage resources.

Sun Shared Visualization Software

Transparent Remote Access to 3D Applications



- 🟨 VirtualGL
- 🟦 Sun Grid Engine
- 🟩 TurboVNC
- 🟧 Standard

SGE qmaster

Viz-enhanced Sun Grid Engine

Client

SGE Run script (e.g., vglrun)

TurboVNC viewer

VGL client

X Server

Display Hardware

Display

3D graphics/execution host

TurboVNC Server

vglrun

Application

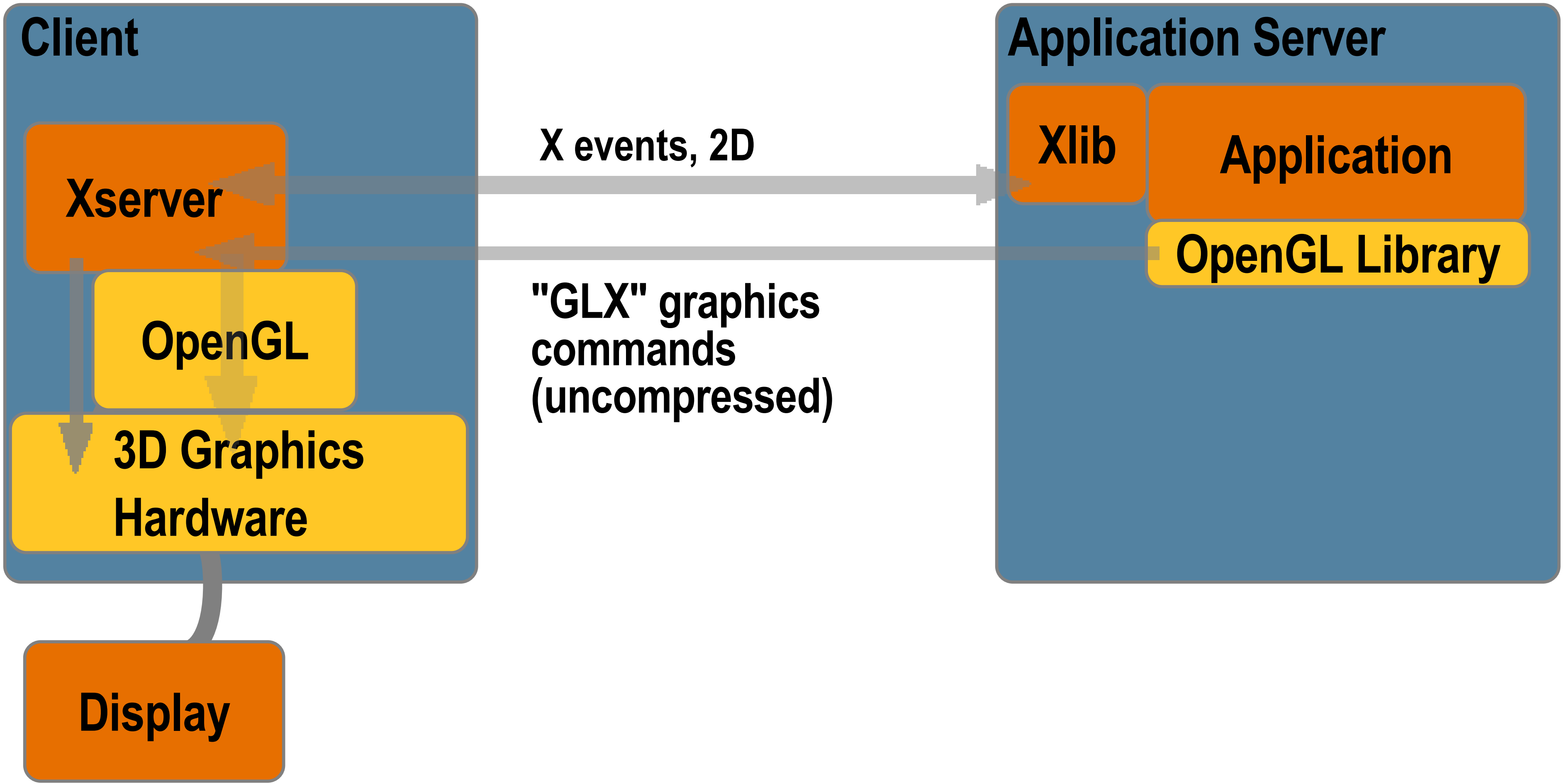
VirtualGL

3D Graphics Hardware

VirtualGL

- Purpose
  - > Allows OpenGL applications which were designed to run and display on the same system to efficiently and transparently run on one system and display on another.
- Components
  - > "Middleware" software for Linux and Solaris servers
  - > Platform-optimized Image compression technology
  - > Various data transport methods
  - > vglclient program decompresses and displays the images
    - > Clients for Solaris, Linux, Windows, Mac OS X
  - > Sun Ray ultra thin clients
- Open source software project sponsored by Sun

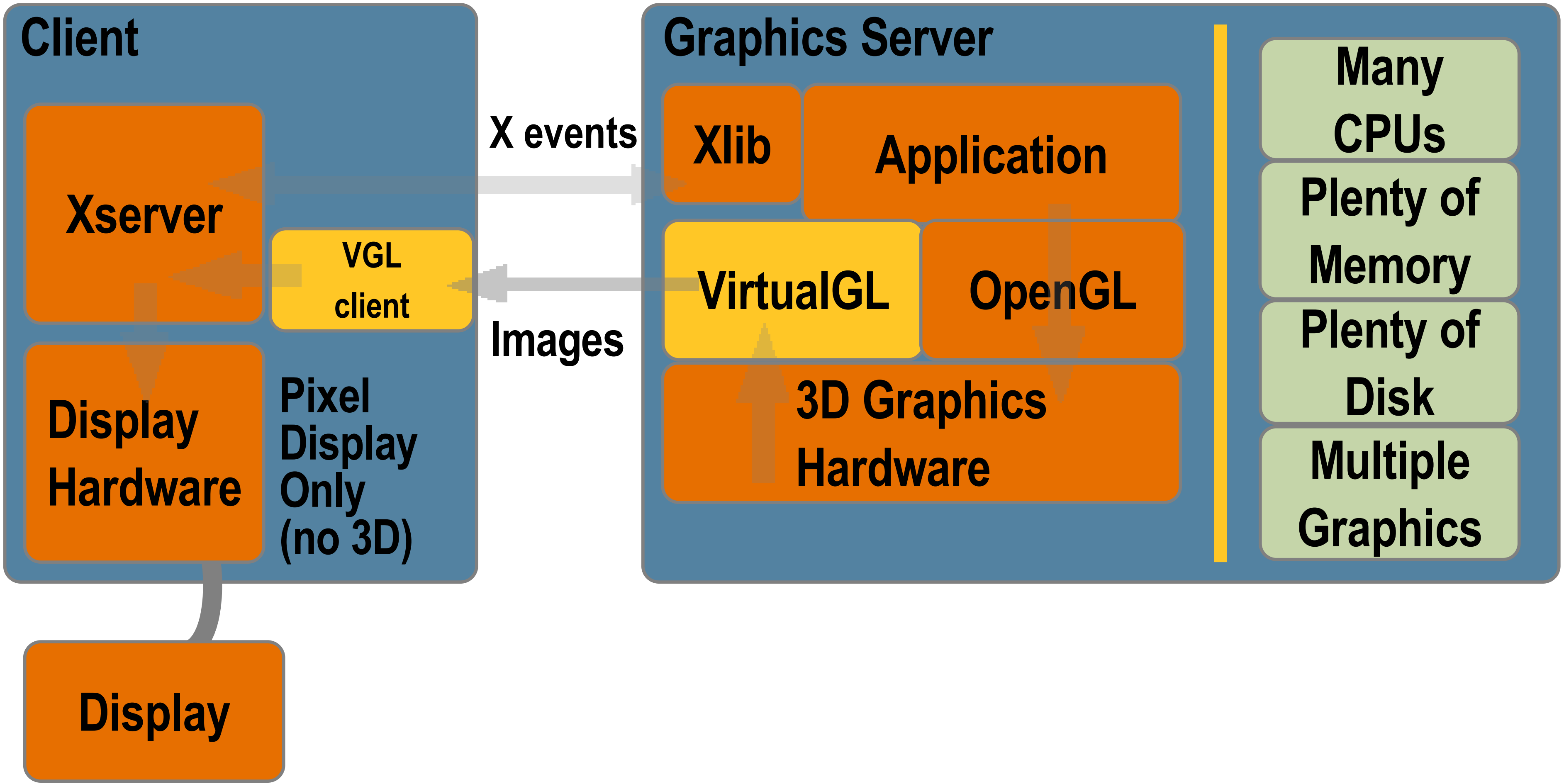
Remote Graphics In the Past



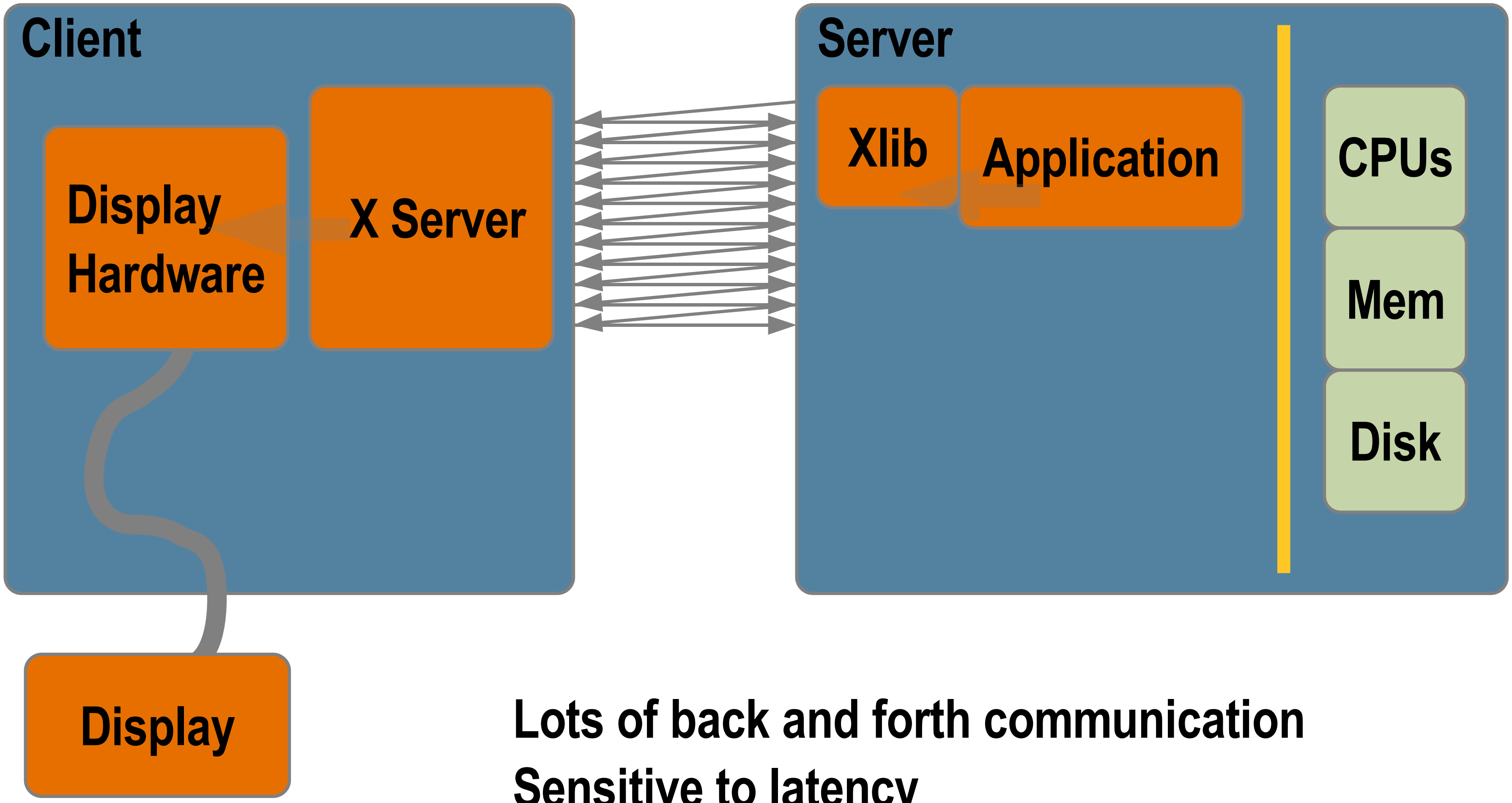
Remote Graphics Using VirtualGL

```
client% vglconnect my_server
```

```
server% vglrun my_application
```



Standard Remote X



Lots of back and forth communication

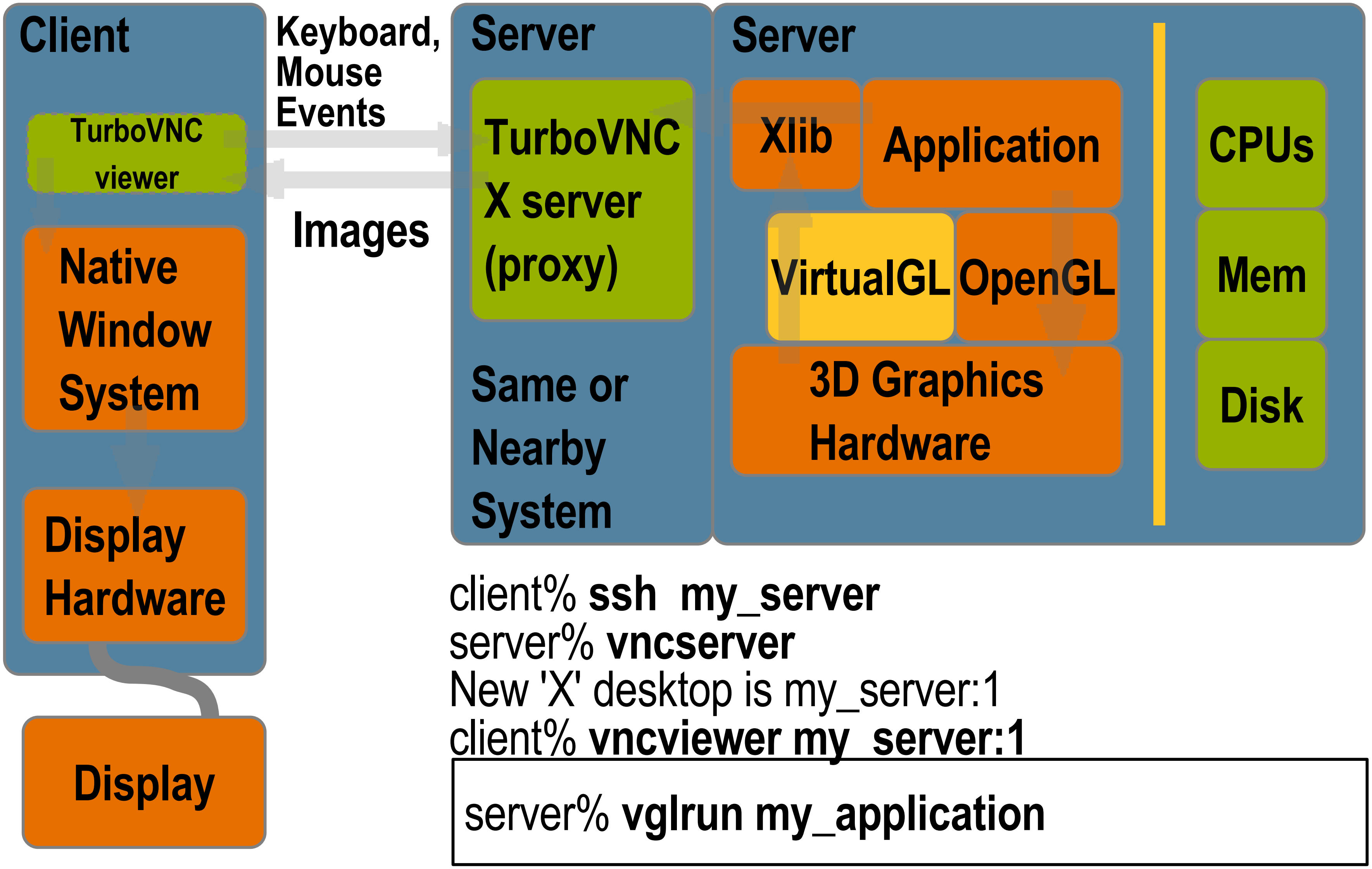
Sensitive to latency

TurboVNC

Latency Tolerant Remote Visualization with Collaboration

- Purpose
  - Allows X applications which were designed to run and display on the same system (or on low latency networks) to transparently run on one system and display on another.
  - Also enables collaboration by allowing more than one client system to display the X session.
- Open source project sponsored by Sun
  - Derived from TightVNC but uses same optimized image compression technology as VirtualGL.
  - Interoperable with other VNC viewers, including Java-based WebVNC

VirtualGL With TurboVNC



```
client% ssh  my_server
server% vncserver
New 'X' desktop is my_server:1
client% vncviewer my_server:1
```

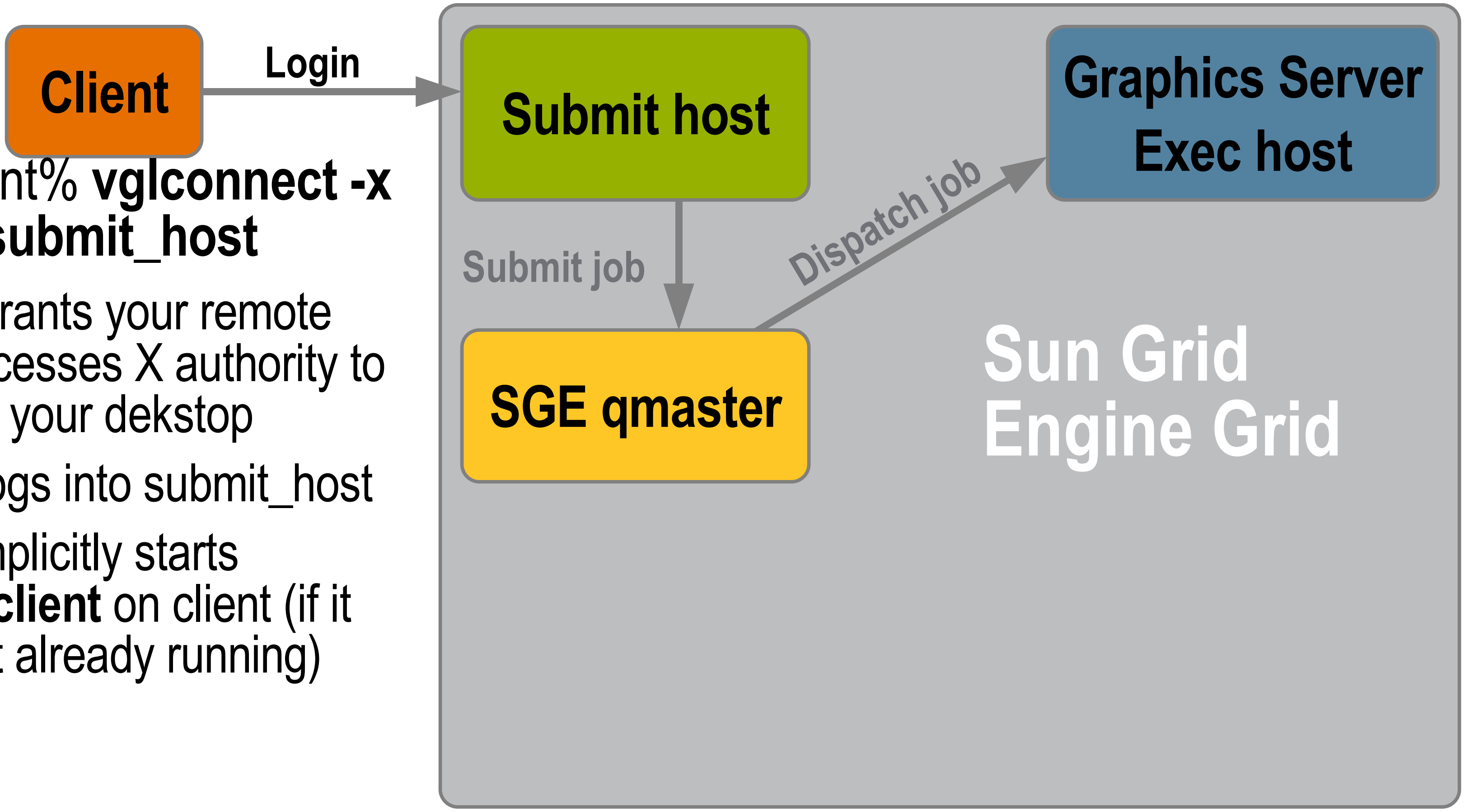
```
server% vglrun my_application
```

Integration with Sun Grid Engine

Sun Grid Engine (With Shared Viz Enhancements)

- Purpose
  - Lets users share graphics servers
    - SGE assures available CPU, memory, and graphics resources
- Components
  - Standard SGE provides management and allocation of regular compute resources (CPUs, memory, OS, software licenses).
  - Enhancements allow SGE to manage graphics resources
    - provides “user-transparent” connection between the allocated graphics device and the user's display on the remote client.
  - Advance Reservation system allows resources to be reserved for a specific time in the future.
- Open source software developed by Sun

Sun Grid Engine With Shared Visualization



client% **vglconnect -x submit_host**

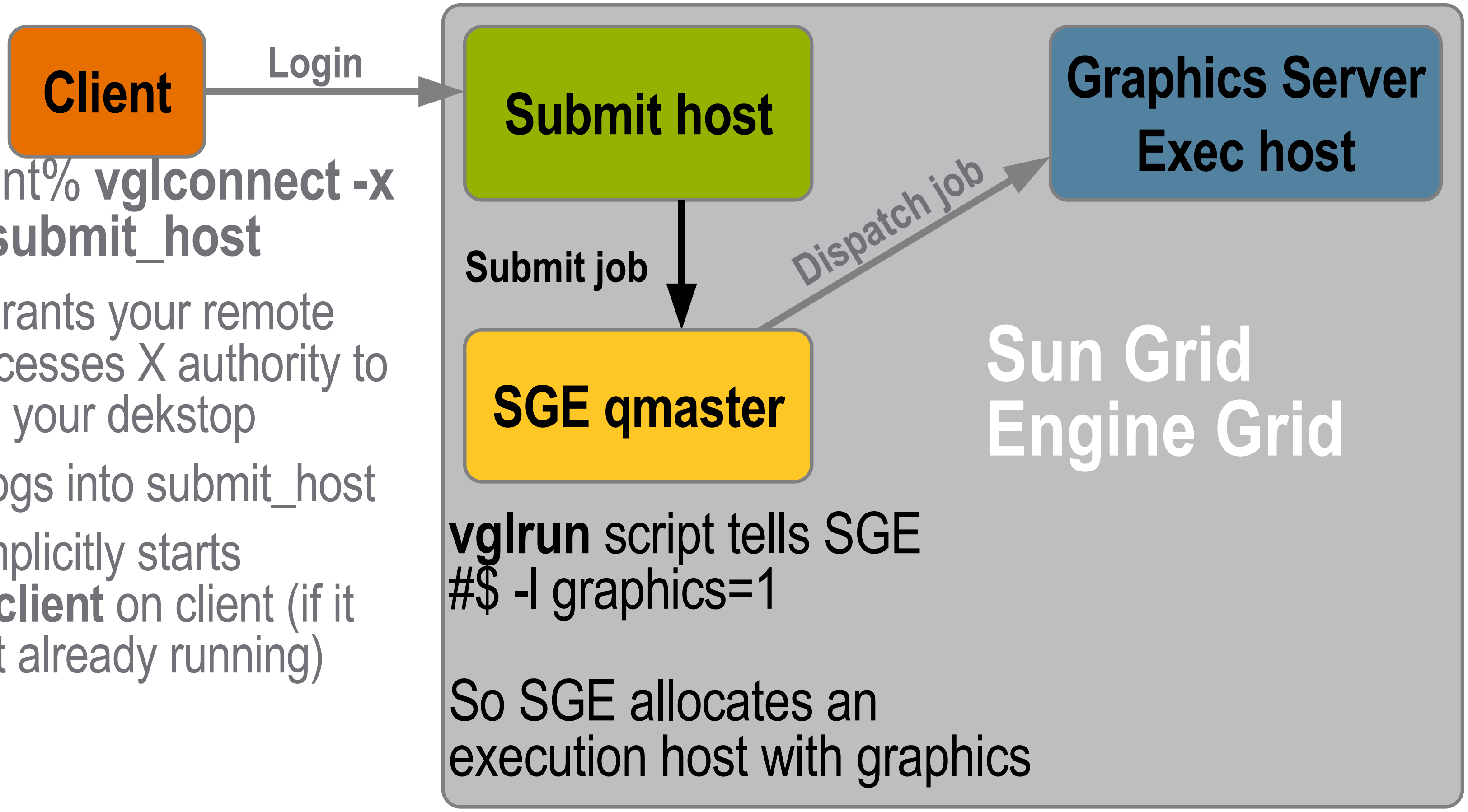
- Grants your remote processes X authority to use your dekstop
- Logs into submit_host
- Implicitly starts **vglclient** on client (if it isn't already running)

Sun Grid Engine With Shared Visualization

submit_host% **qrsh -b no /opt/VirtualGL/bin/vglrun my_app**

client% **vglconnect -x submit_host**

- Grants your remote processes X authority to use your dekstop
- Logs into submit_host
- Implicitly starts **vglclient** on client (if it isn't already running)

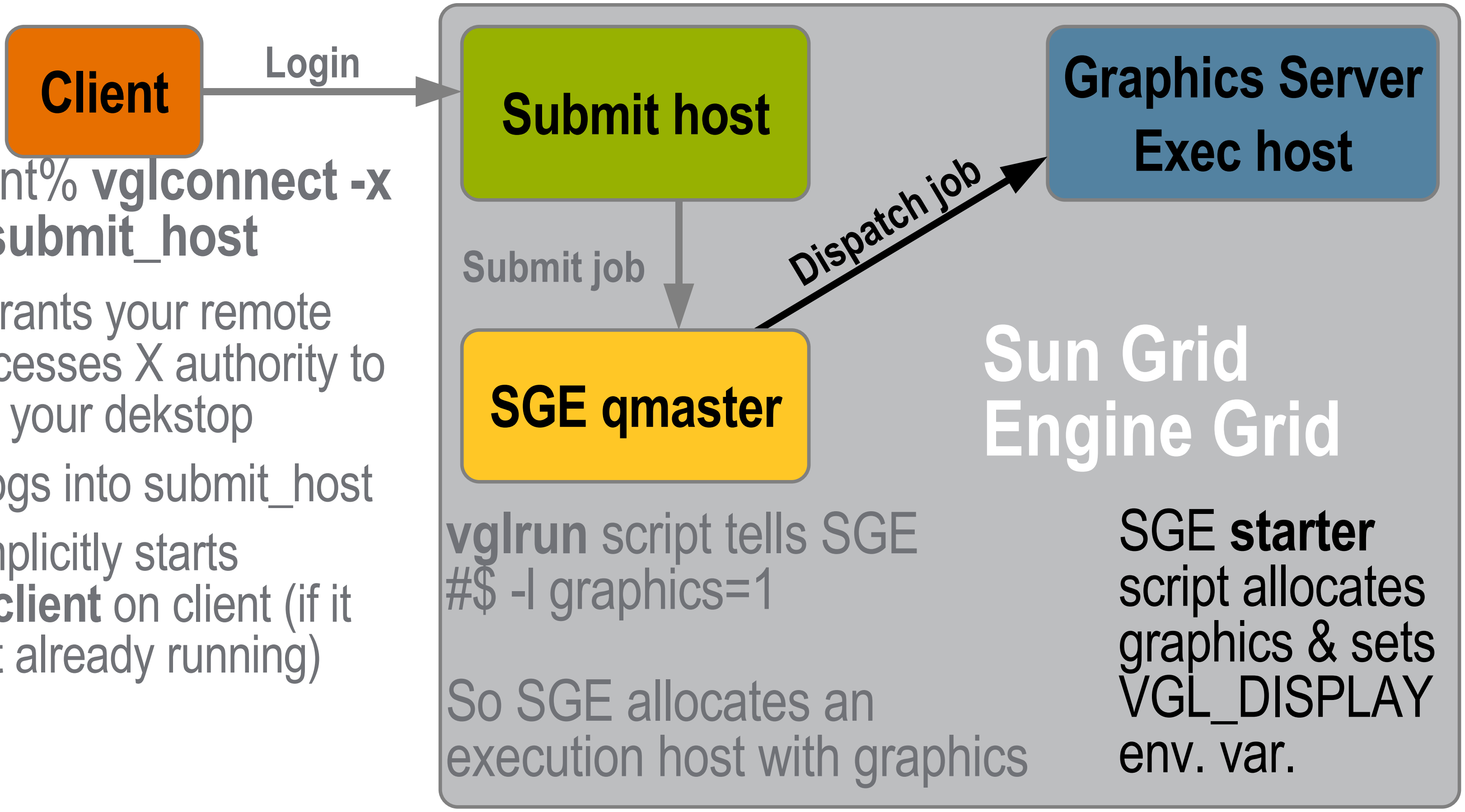


vglrun script tells SGE
#$ -l graphics=1

So SGE allocates an
execution host with graphics

Sun Grid Engine With Shared Visualization

```
submit_host% qrsh -b no /opt/VirtualGL/bin/vglrun my_app
```



```
client% vglconnect -x
    submit_host
```

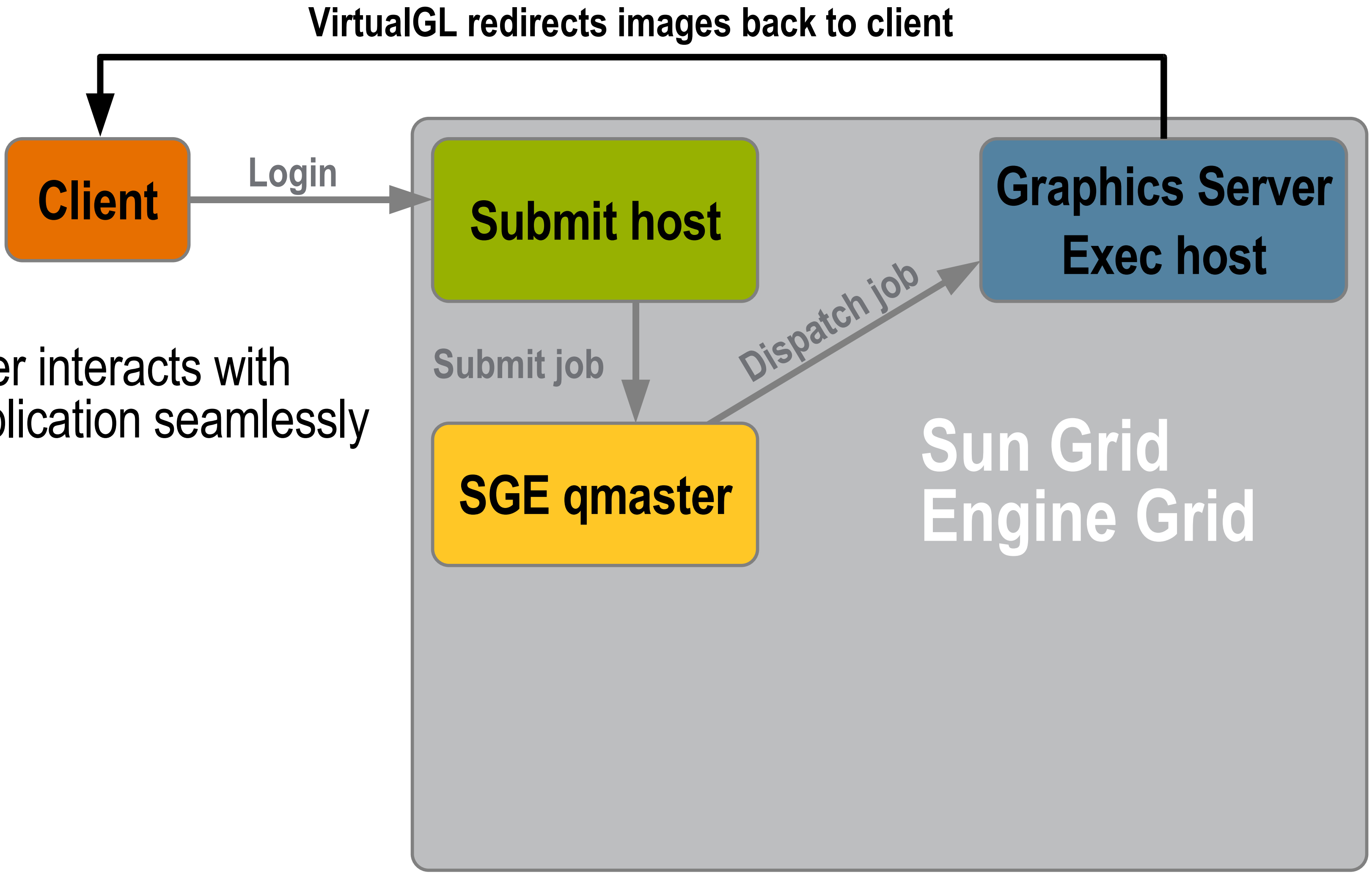
- Grants your remote processes X authority to use your dekstop
- Logs into submit_host
- Implicitly starts **vglclient** on client (if it isn't already running)

vglrun script tells SGE
#$ -l graphics=1

So SGE allocates an execution host with graphics

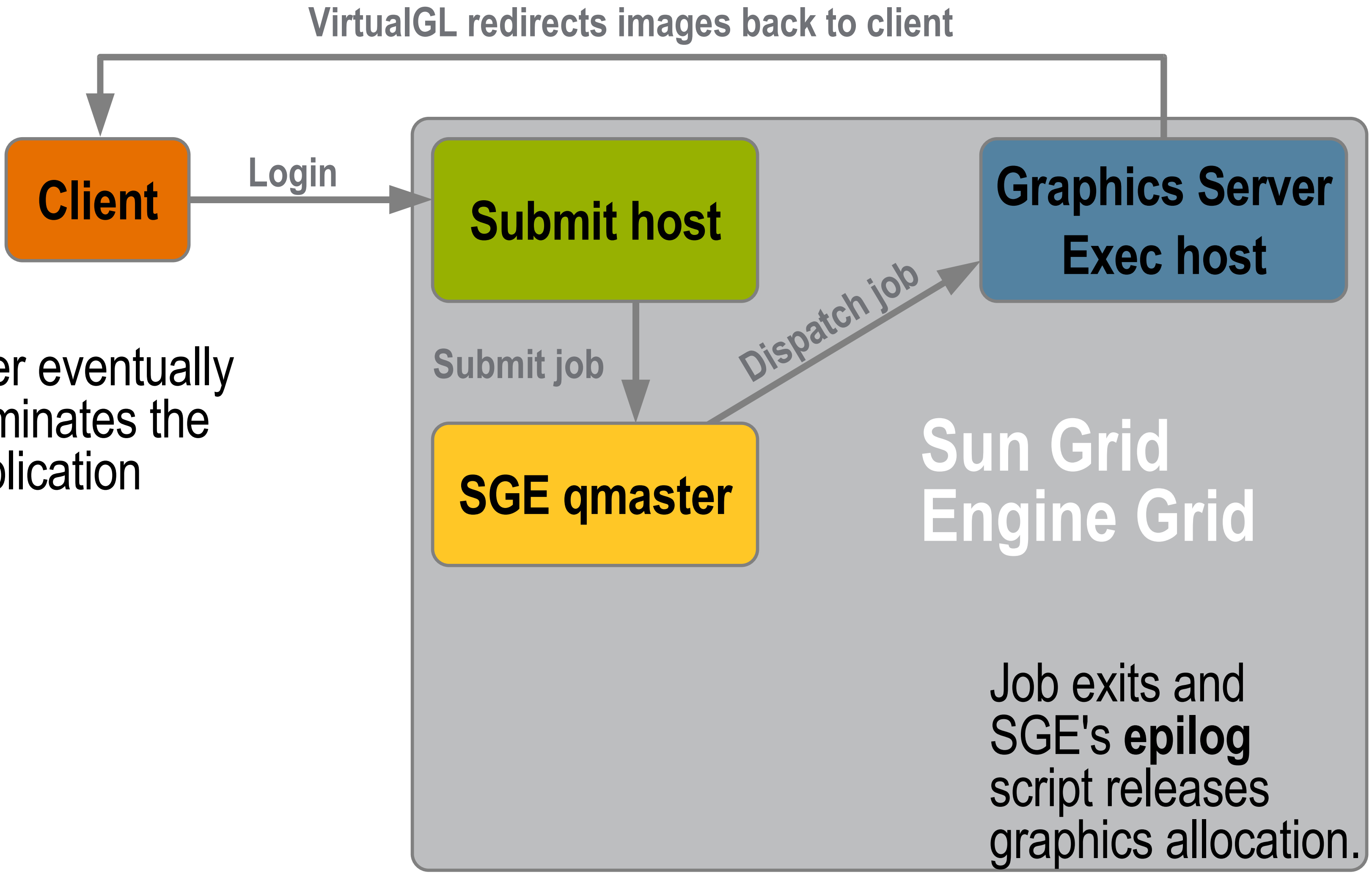
SGE **starter** script allocates graphics & sets VGL_DISPLAY env. var.

Sun Grid Engine With Shared Visualization



User interacts with application seamlessly

Sun Grid Engine With Shared Visualization



User eventually terminates the application

Job exits and SGE's **epilog** script releases graphics allocation.

Grid Engine Application Script

Script Uses #$ Comments to Describe its Needs to Grid Engine

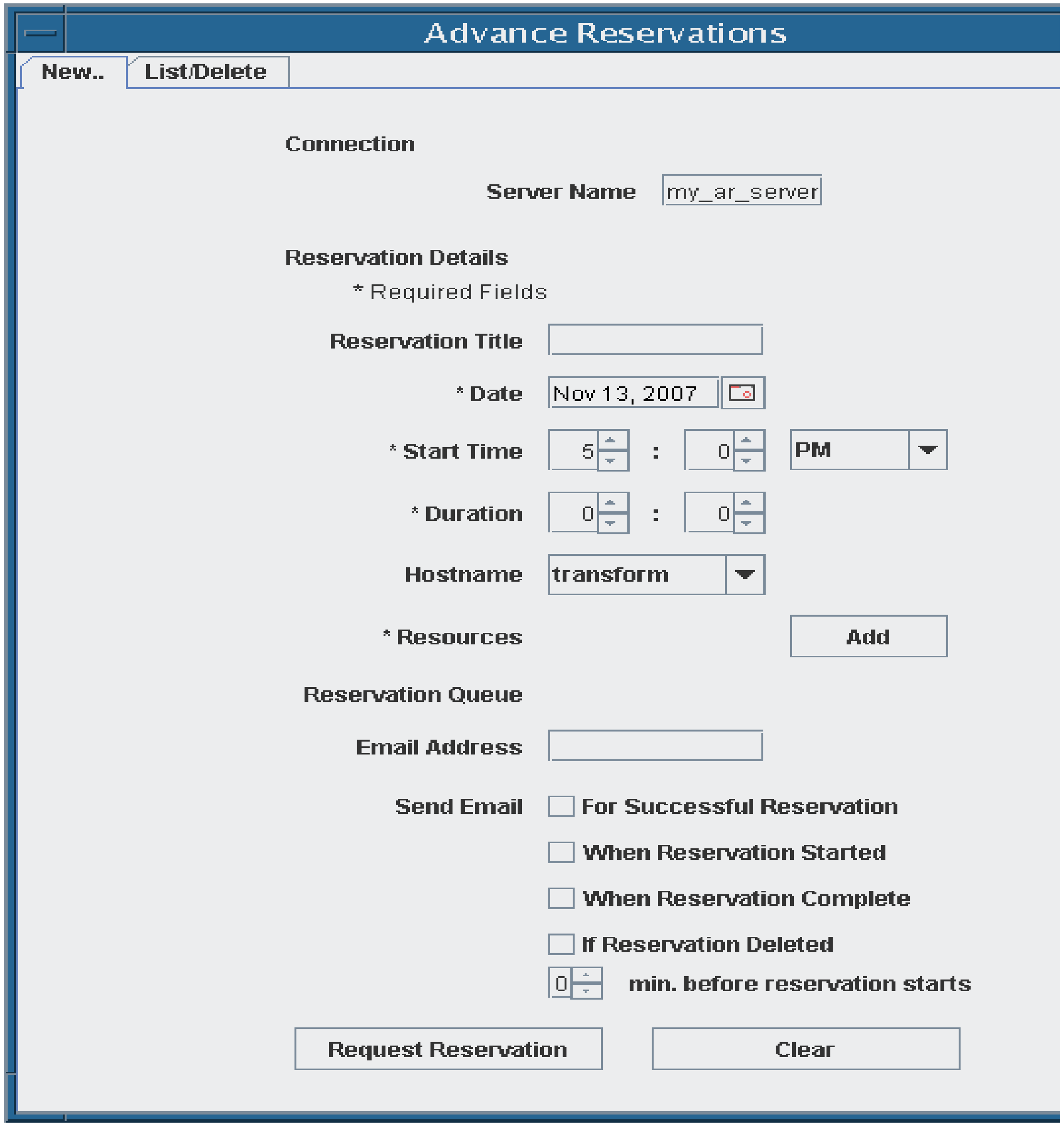
- Grid Engine will allocate acceptable host and graphics
  - > **qrsh -b no my_script**
  - > **qsub -now y my_script**

```
#!/bin/sh
#$ -N my_app              ← Name of Grid Engine job
#$ -l graphics=1          ← Job requires a graphics card
#$ -l arch=lx24-x86|lx24-amd64 ← Required Architecture(s)
#  Here, required architecture is "Linux (with 2.4 or 2.6 kernel) on x86 or x64"
#$ -v DISPLAY,SSH_CLIENT     ← Save environment vars. with job
#  These variables enable VirtualGL to find the client's X display (desktop).

#  Invoke application using VirtualGL to route 3D to allocated graphics card
vglrun my_application "$@"   ← Invoke app with any args
#  Script invokes VirtualGL implicitly, so submitter need not mention vglrun
```

Making an Advance Reservation

- GUI invoked with **`runar`** script (that invokes Java)
  - > $SGE_ROOT/ar/bin/runar ReserveGUI
- Command line
  - > also uses **`runar`**
  - > **`-help`** shows options.
  - >
    ```
    $SGE_ROOT/ar/bin/runar \
    Reserve -a 12250730 \
    -duration 1:30:0 \
    -l graphics=1
    ```



Connection

Server Name: my_ar_server

Reservation Details

* Required Fields

Reservation Title:

* Date: Nov 13, 2007

* Start Time: 5 : 0 PM

* Duration: 0 : 0

Hostname: transform

* Resources

Reservation Queue

Email Address:

Send Email:
□ For Successful Reservation
□ When Reservation Started
□ When Reservation Complete
□ If Reservation Deleted
0 min. before reservation starts

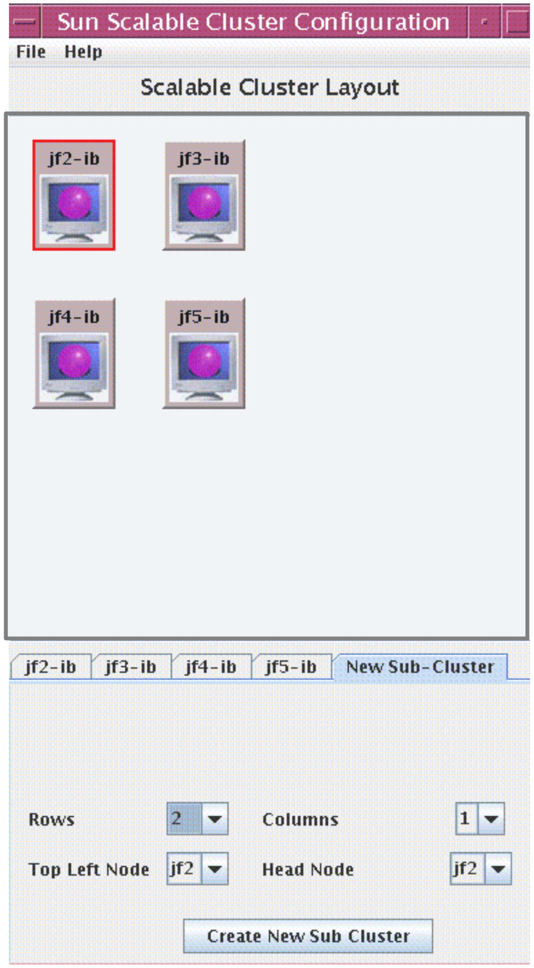
Using an Advance Reservation

- AR server confirms request and creates a queue.
  - AR queue name such as **deans123456** is shown, E-mailed
    - Specific to the reserved execution host and to the reserving user
    - User can submit jobs to the reservation before or during its period
    - **qrsh -b no -q deans123456 /opt/VirtualGL/bin/vglrun my_app**
  - Before the reservation's start time, resources are removed from the generic queue so others won't start using them.
  - During the reservation period, jobs can run on the reservation queue, using reserved resources.
  - After the reservation period, the resources are returned to the host's generic queue.
- Grid Engine 6.2 plans to provide its own AR facility.

Integration with Scalable Visualization 1.1

Each graphics cluster is exposed as an SGE queue

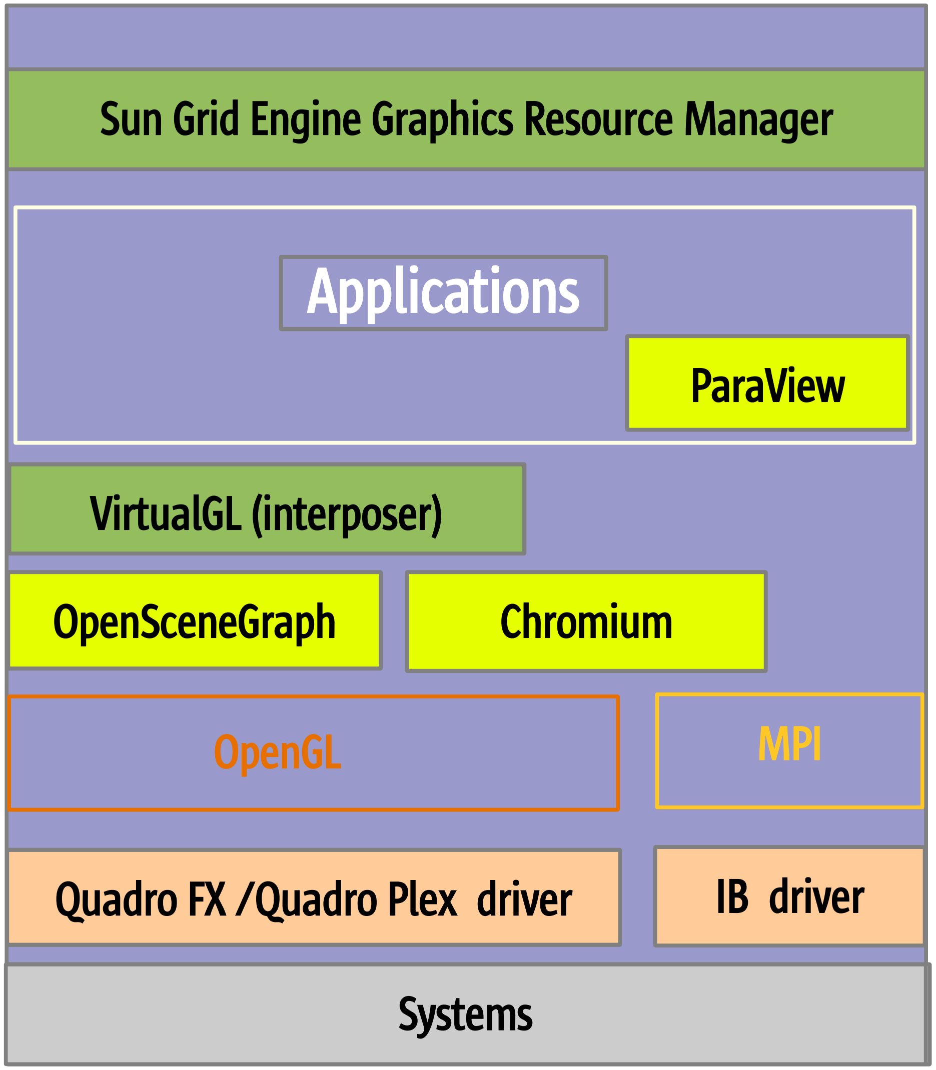
- New GUI for admin to define a subcluster
  - > Scalable Viz configuration files for the subcluster are created and stored for use with Shared Viz
- Shared Visualization adds to SGE a subcluster "parallel environment" **sc** and its master job script.
  - > Submission for a 2x1 host (2x2 display) power wall:
    - > **qrsh -b n -q head_2x1 $SGE_ROOT/graphics/sc/master cr_start.sh ...**
    - > The master script starts the Scalable Viz script (here, cr_start.sh) on the subcluster's head node.



Rows 2 Columns 1

Top Left Node jf2 Head Node jf2

Sun Visualization Stack



- **Shared Visualization software stack** - visualization services to a variety of remote clients
  - **SGE** - graphics resource management
  - **VirtualGL** - remote access via any client over standard IP networks
- **Scalable Visualization software**
  - **ParaView** - open-source parallel rendering application optimized for SMPs with multiple graphics.
  - **OpenSceneGraph** - open-source parallel rendering toolkit for building parallel applications.
  - **Chromium** - virtualized graphics devices for Solaris or Linux. Provides transparent parallelization for fill-rate limited applications; api for parallelizing applications by splitting up the data.
- **Quadro Plex** - connects graphics devices to Linux or Solaris servers over a PCI-E cable
- **Systems** - Sun Fire x86 & SPARC systems provide the most scalable platform

Dean Stanton,
Advanced Visualization,
Sun Microsystems

dean.stanton@sun.com

www.sun.com/visualization/

OpenSolaris.org/os/project/visualization-hpc/