Grid Portals: Putting the User Interface on the Grid and Virtual Organizations

Charles Severance

University of Michigan

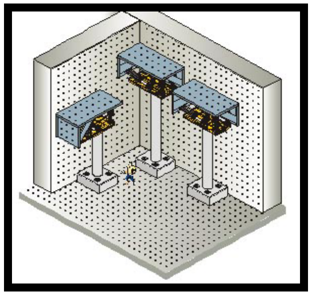
NEESGrid project, OGCE Project, Sakai Project

www.dr-chuck.com

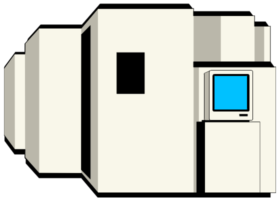
csev@umich.edu

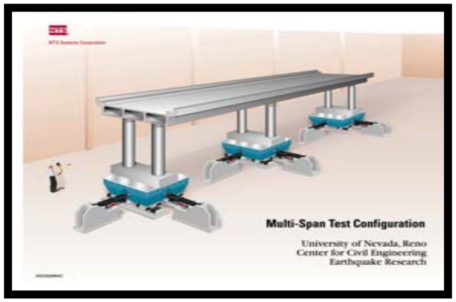
Grid: A collection of “things” (people, equipment, etc) which need to work together to accomplish some task

Multi-Axial Subassemblage Testing (MAST)
MAST Laboratory
Department of Civil Engineering
Institute of Technology
MTS Systems Corporation



NEESGrid: $200M, 13 years, 30+ organizations, 300+ people, a NEES non-profit consortium, 16 geographically distributed experimental sites all combine using the global public Internet to produce one national experimental laboratory which will enable fundamental advances in earthquake earthquake engineering.

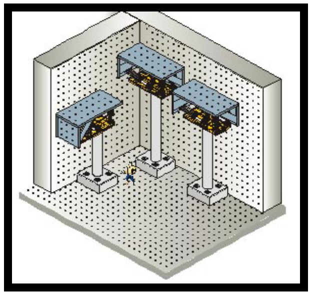


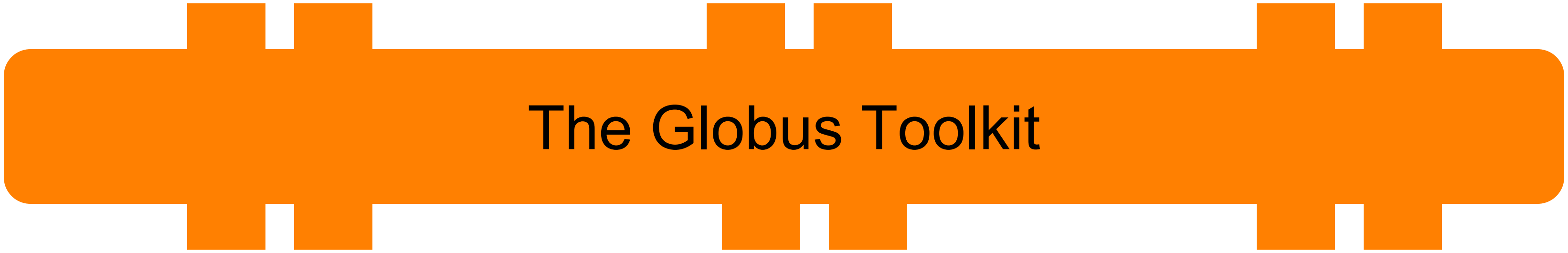

MTS Systems Corporation
Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

Note: In this story (like all good stories) everything sounds very good *but* there is a flaw - look for it… See if you can figure out what the flaw is before it is revealed at the end of the story…

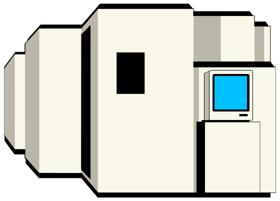
Making a Grid: Step 1 - Invent a Grid Toolkit with protocols

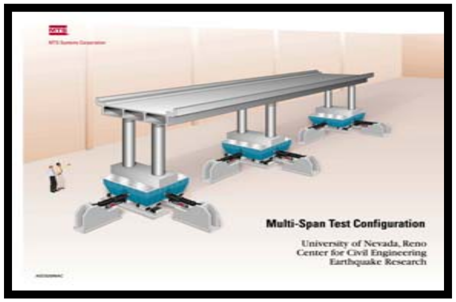
Multi-Axial Subassemblage Testing (MAST)
MAST Laboratory
Department of Civil Engineering
Institute of Technology
MTS Systems Corporation



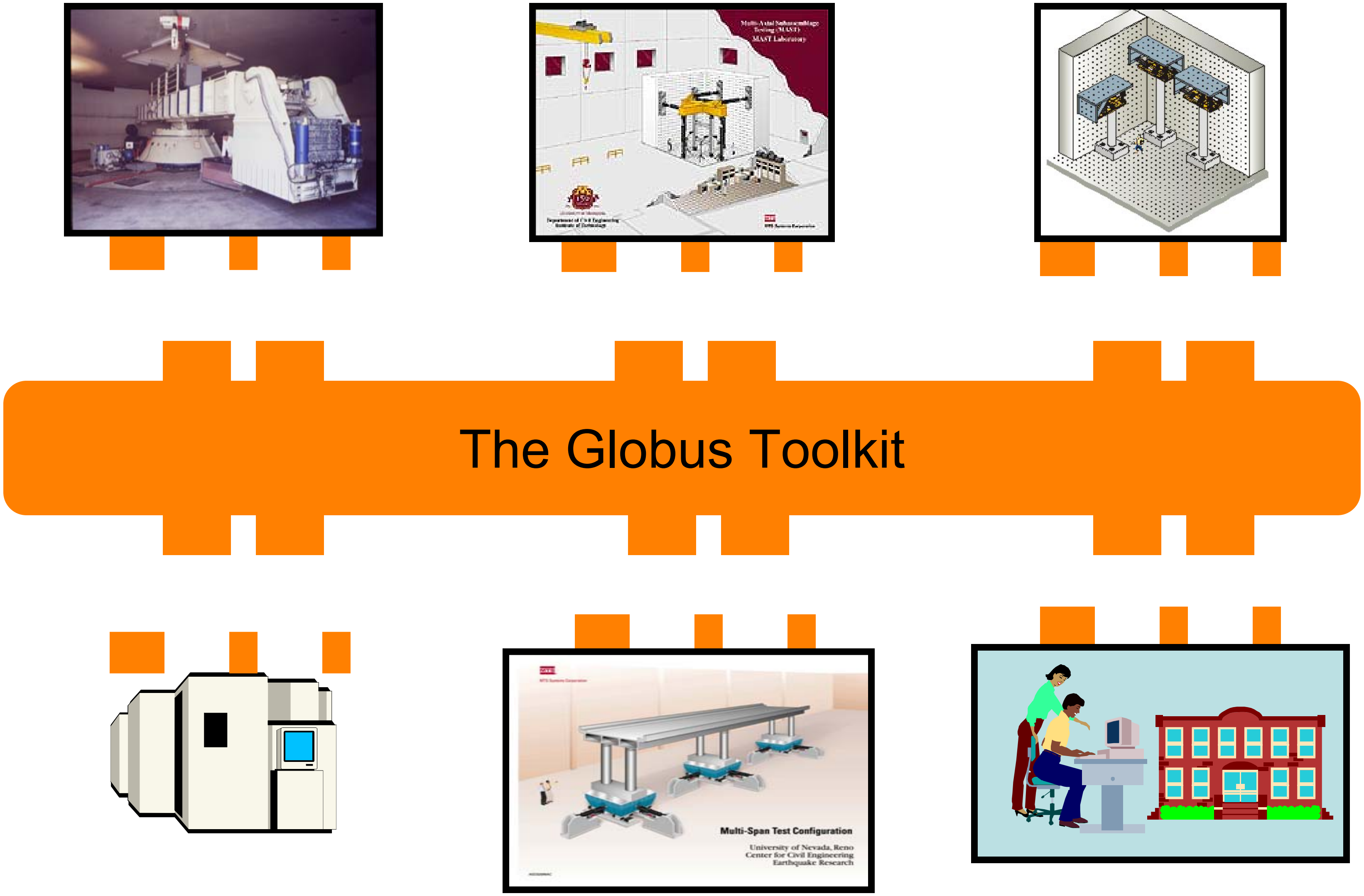


The Globus Toolkit




Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

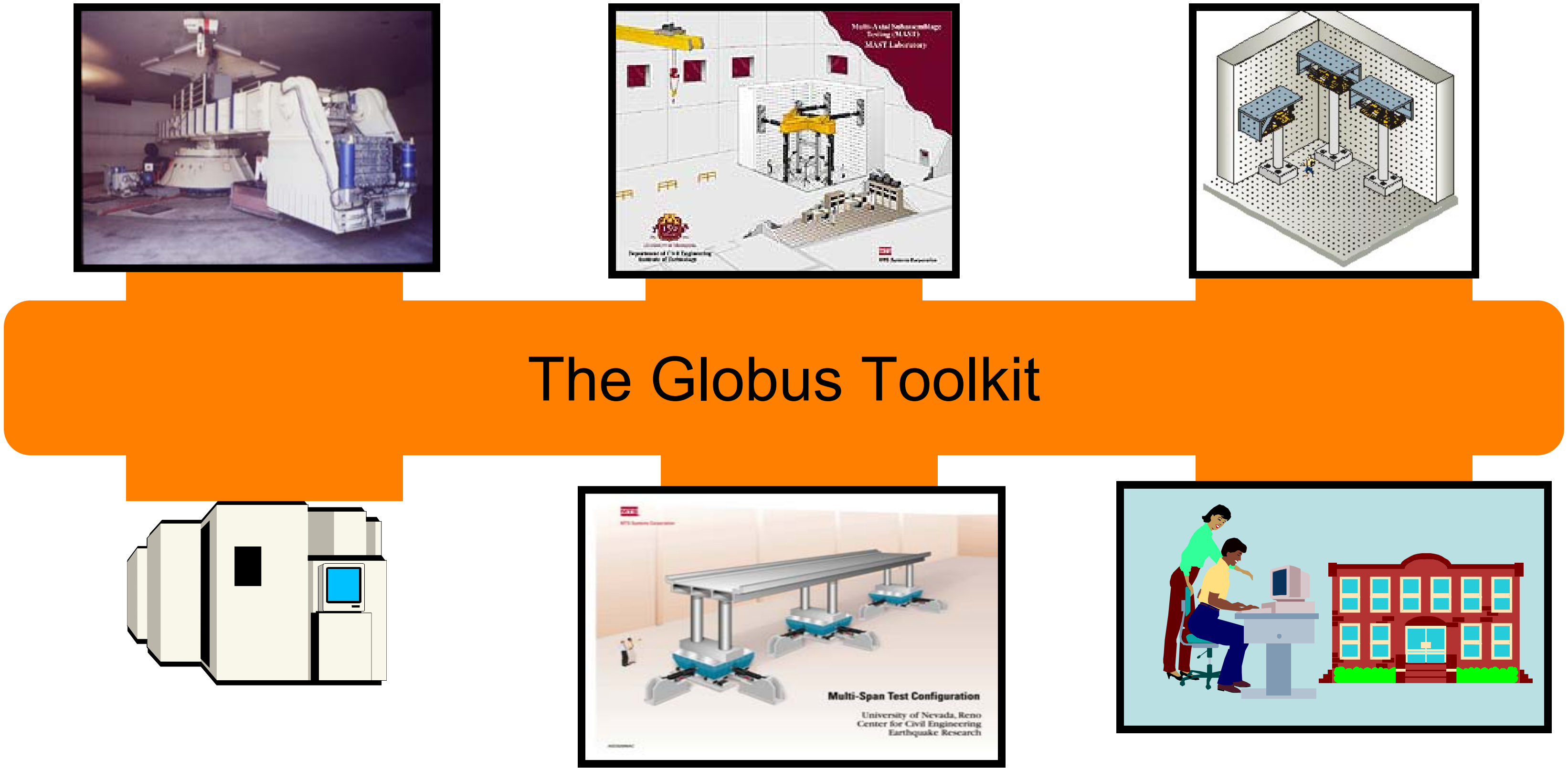
Making a Grid: Step 2 - Add those protocols to the “things” that need to work together…



The Globus Toolkit

Multi-Axial Subassemblage Testing (MAST)
MAST Laboratory
Department of Civil Engineering
Institute of Technology
MTS Systems Corporation
Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

Making a Grid: Step 3 - Plug it all in…



The Globus Toolkit

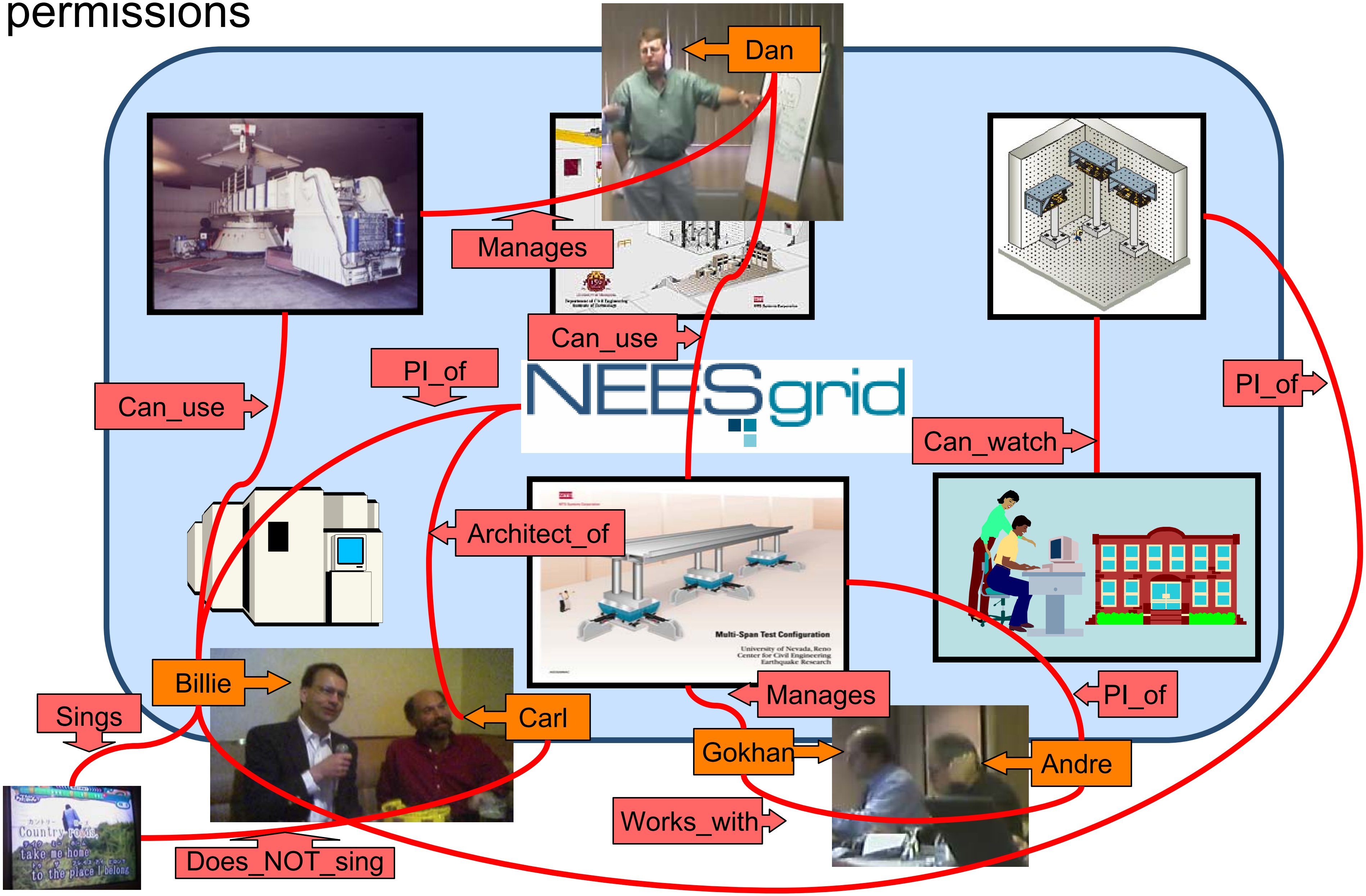
Multi-Axial Subassemblage Testing (MAST)
MAST Laboratory
Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

Note: By now you have seen the flaw several times - you should be forming a hypothesis about the flaw right about now….

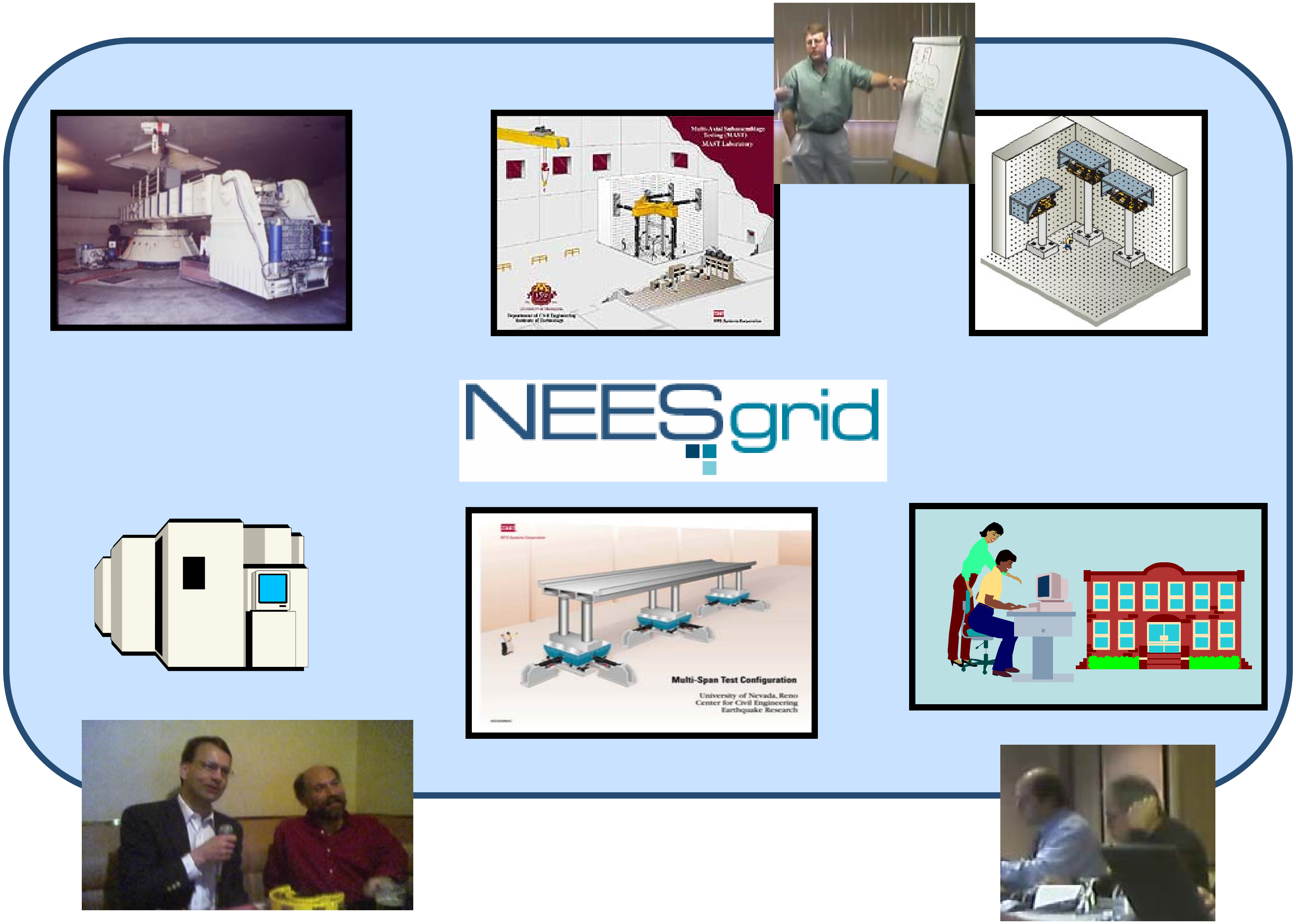
Making a Grid: Step 3 - Viola! NEESGrid is connected together.

Multi-Axial Subassemblage Testing (MAST) MAST Laboratory
NEESgrid
Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

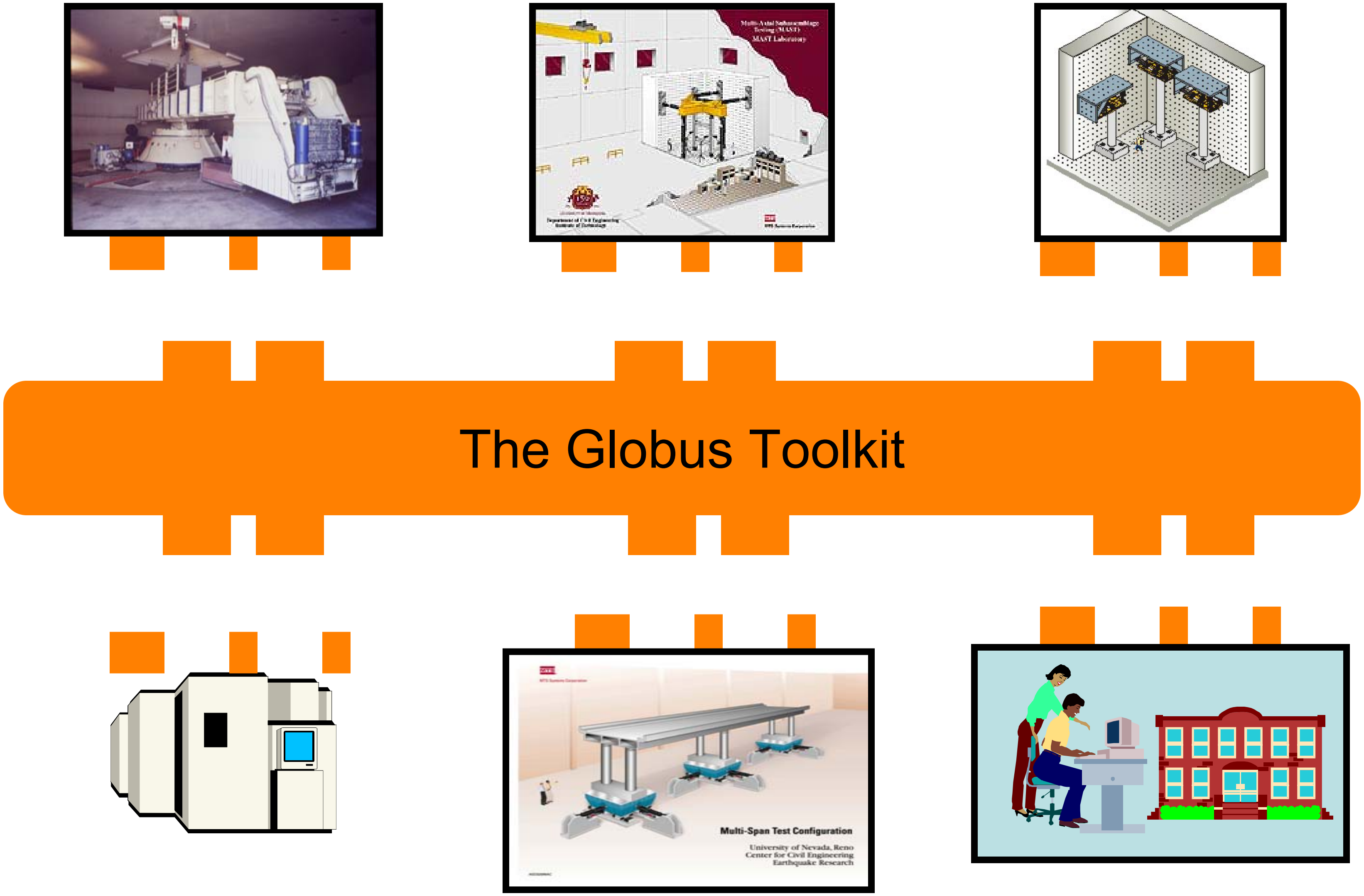
Making a Grid: Just connecting gadgets to a fabric is not sufficient for a Grid - there are “virtual organizations” with rights and permissions



Grid Technology + Virtual Organization = a Real-Live Grid

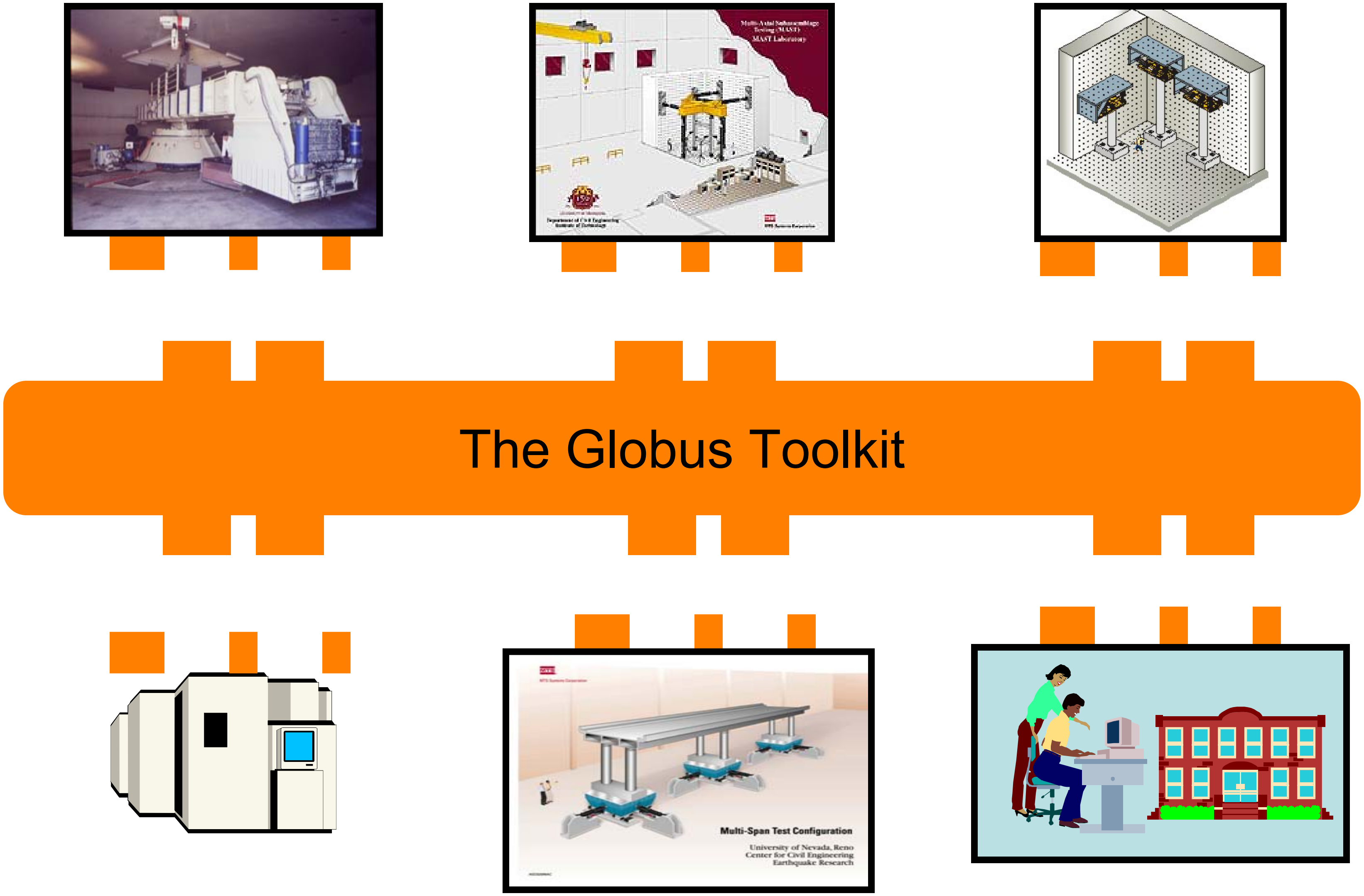

Multi-Axial Subassemblage Testing (MAST)
MAST Laboratory
NEESgrid
Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

Making a Grid: Back to the flaw (aka challenge) - it is in this picture somewhere….


Multi-Axial Subassemblage Testing (MAST)
MAST Laboratory
Department of Civil Engineering
Institute of Technology
MTS Systems Corporation
Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

The Globus Toolkit

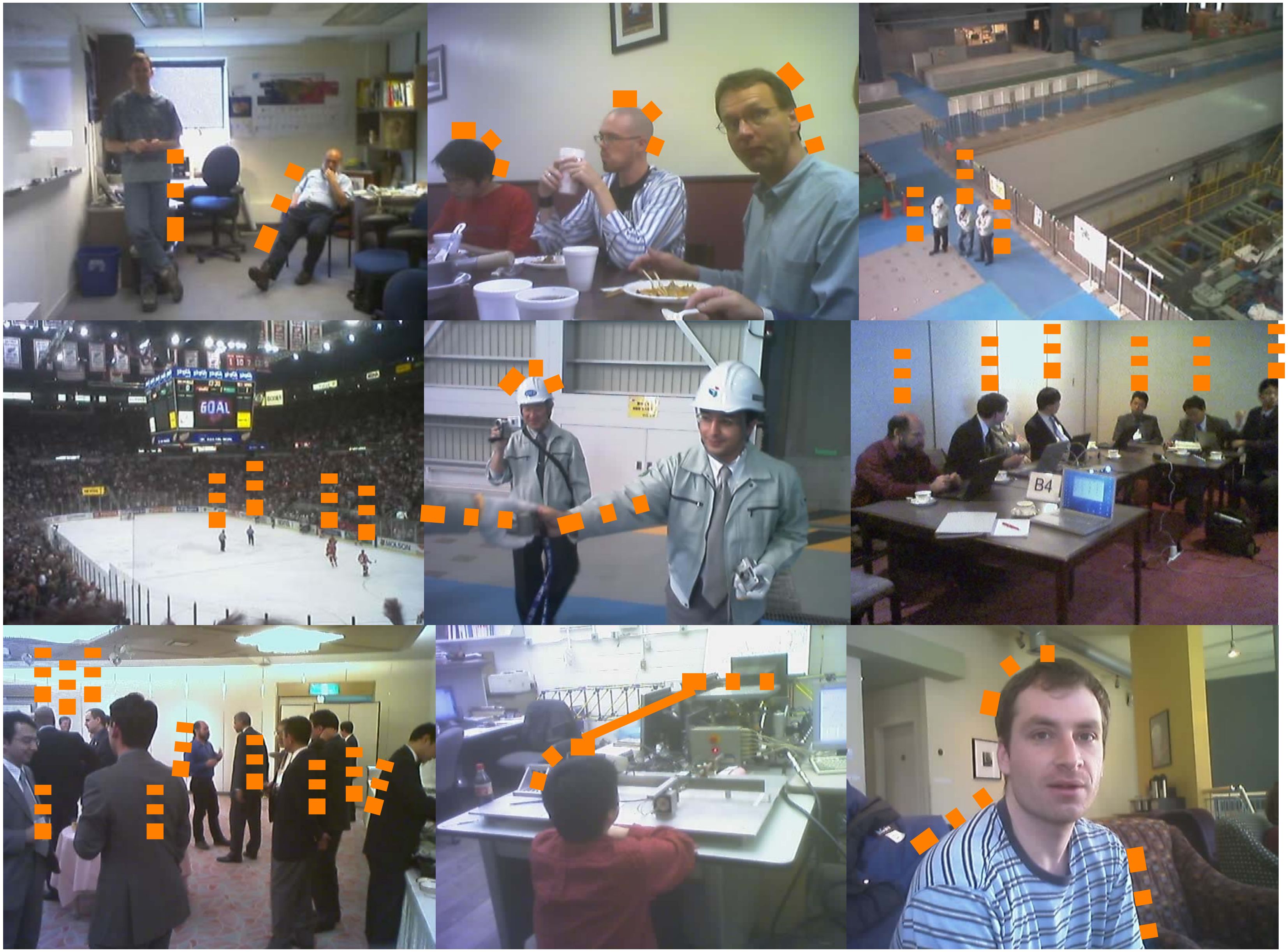
Making a Grid: Hint - Which of these six elements is the most difficult to connect to the Grid….



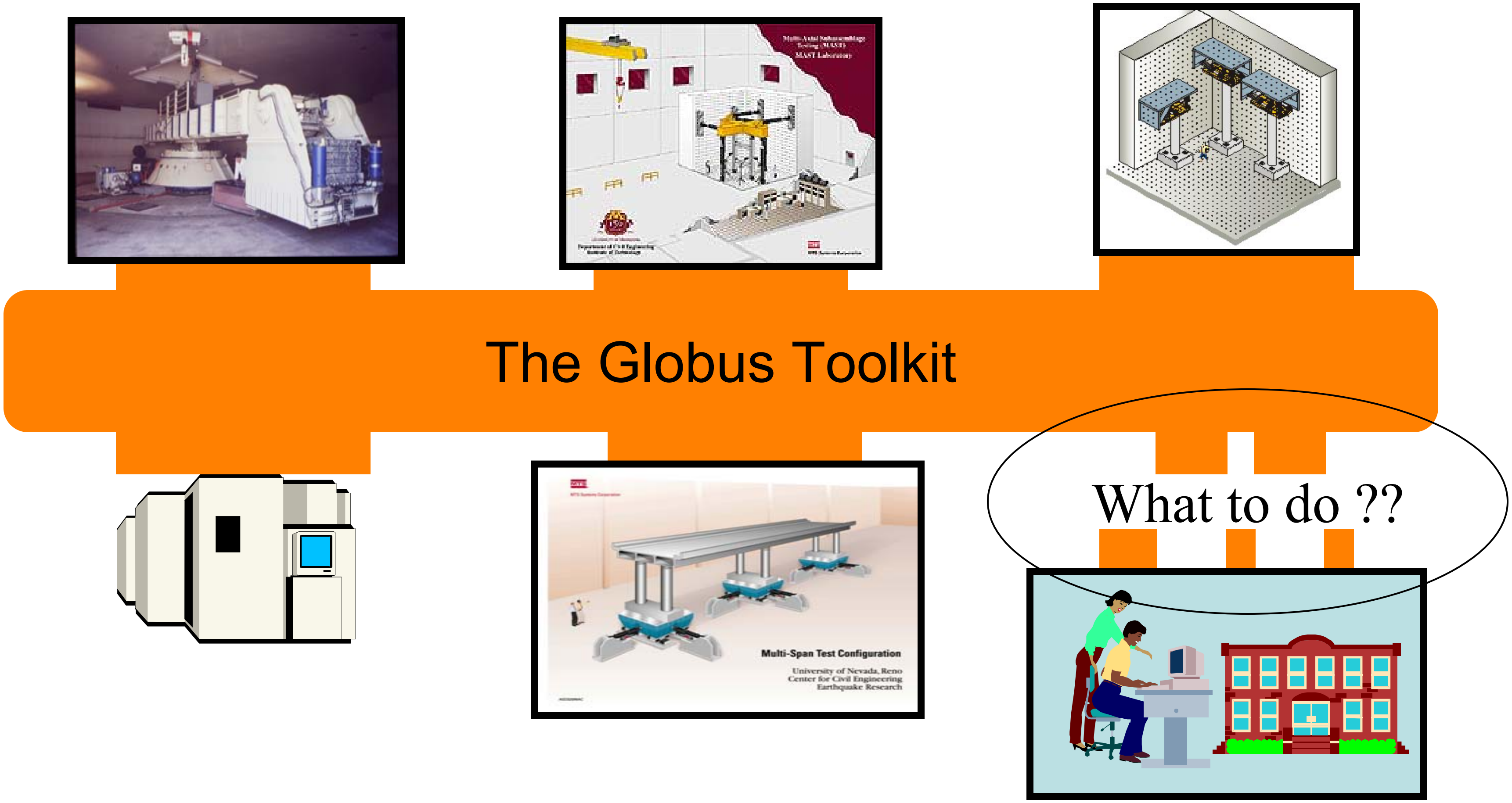
The Globus Toolkit

Multi-Axial Subassemblage Testing (MAST)
MAST Laboratory
Department of Civil Engineering
MTS Systems Corporation
MTS
MTS Systems Corporation
Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

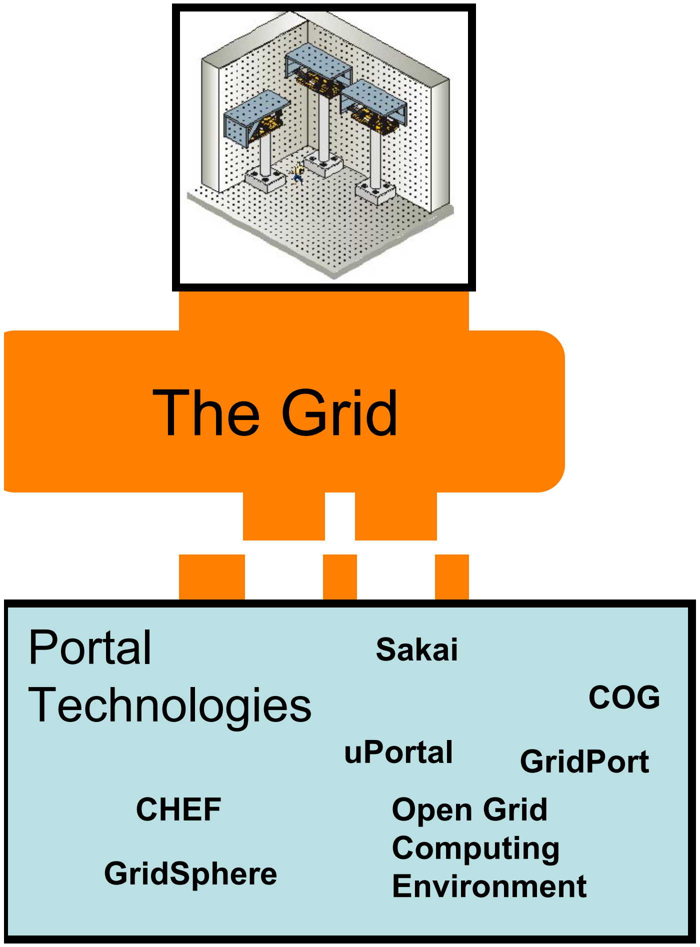
Making a Grid: The hard part is that people won’t stand still long enough to install the Globus toolkit on them…

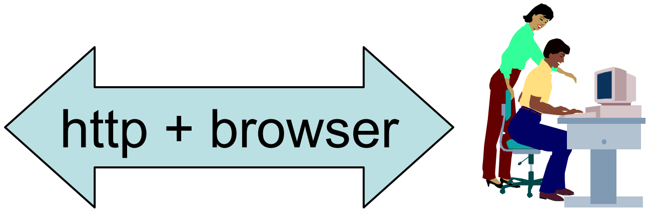


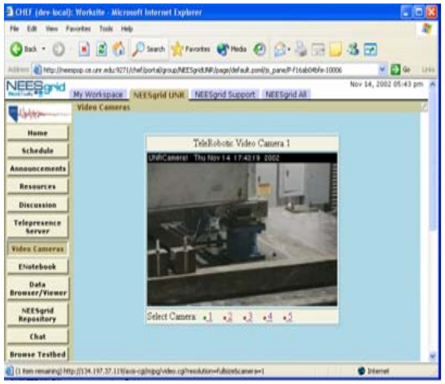
Making a Grid: People are more complex than computers.

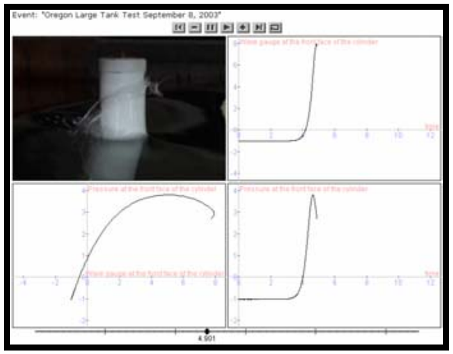

Multi-Axial Subassemblage Testing (MAST)
MAST Laboratory
The Globus Toolkit
What to do ??
Multi-Span Test Configuration
University of Nevada, Reno
Center for Civil Engineering
Earthquake Research

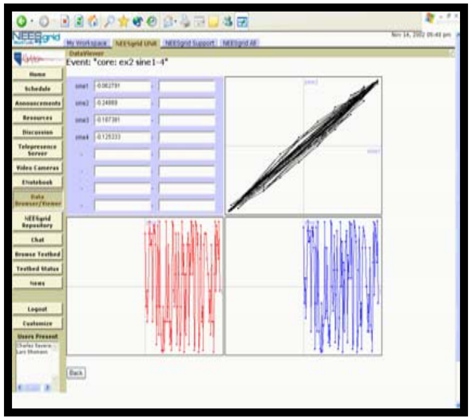
Making a Grid: Grid Portals bridge the gap creating a familiar browser-based user interface to their Grid

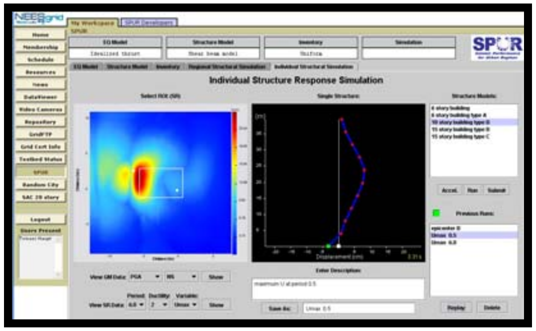




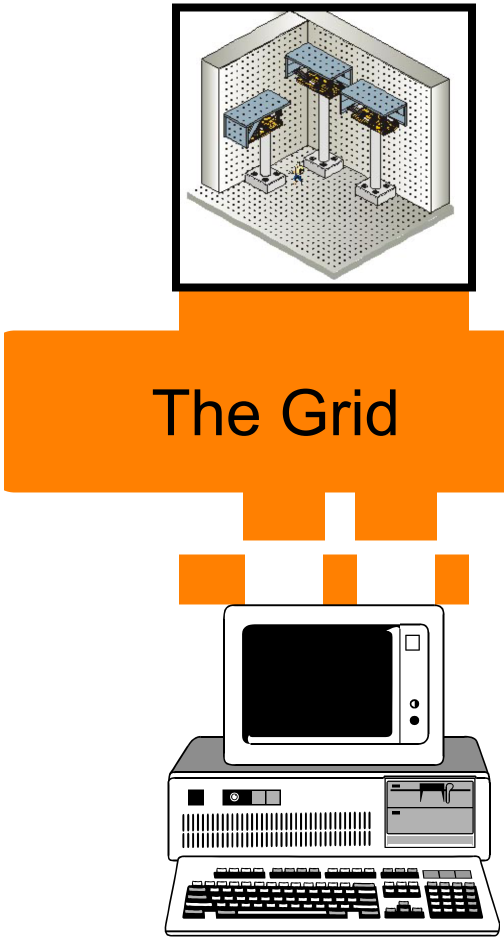


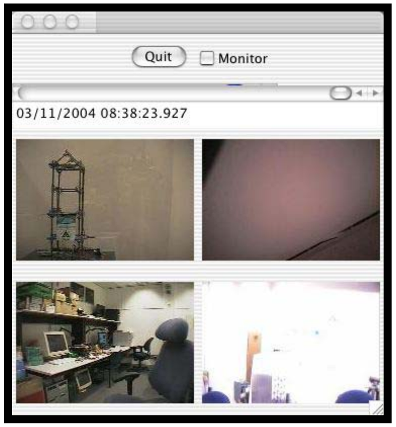

Event: "Oregon Large Tank Test September 8, 2003"


Event: "core: ex2 sine1-4"



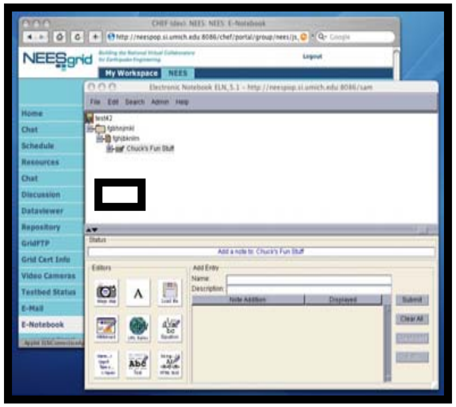
Making a Grid: Grid moving towards the desktop

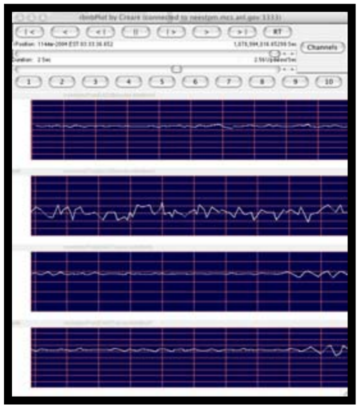

The Grid



While Grid Portals allow everyone to use Grid resources, and provide and excellent ways to collect and trivially distribute functionality, with the emergence of a well supported desktop Java and Java-based GT3 and GT4, it is more and more practical to make a standard Mac, PC, or UNIX desktop a true peer within the Grid with full credentials and desktop applications with higher-functionality interfaces and higher performance.

Grid portals still have a role in this work for those who can't or won't alter their personal system or move from computer to computer. We can also develop common APIs between portals and desktop tools so as to maximize reuse of non-display aspects of software.

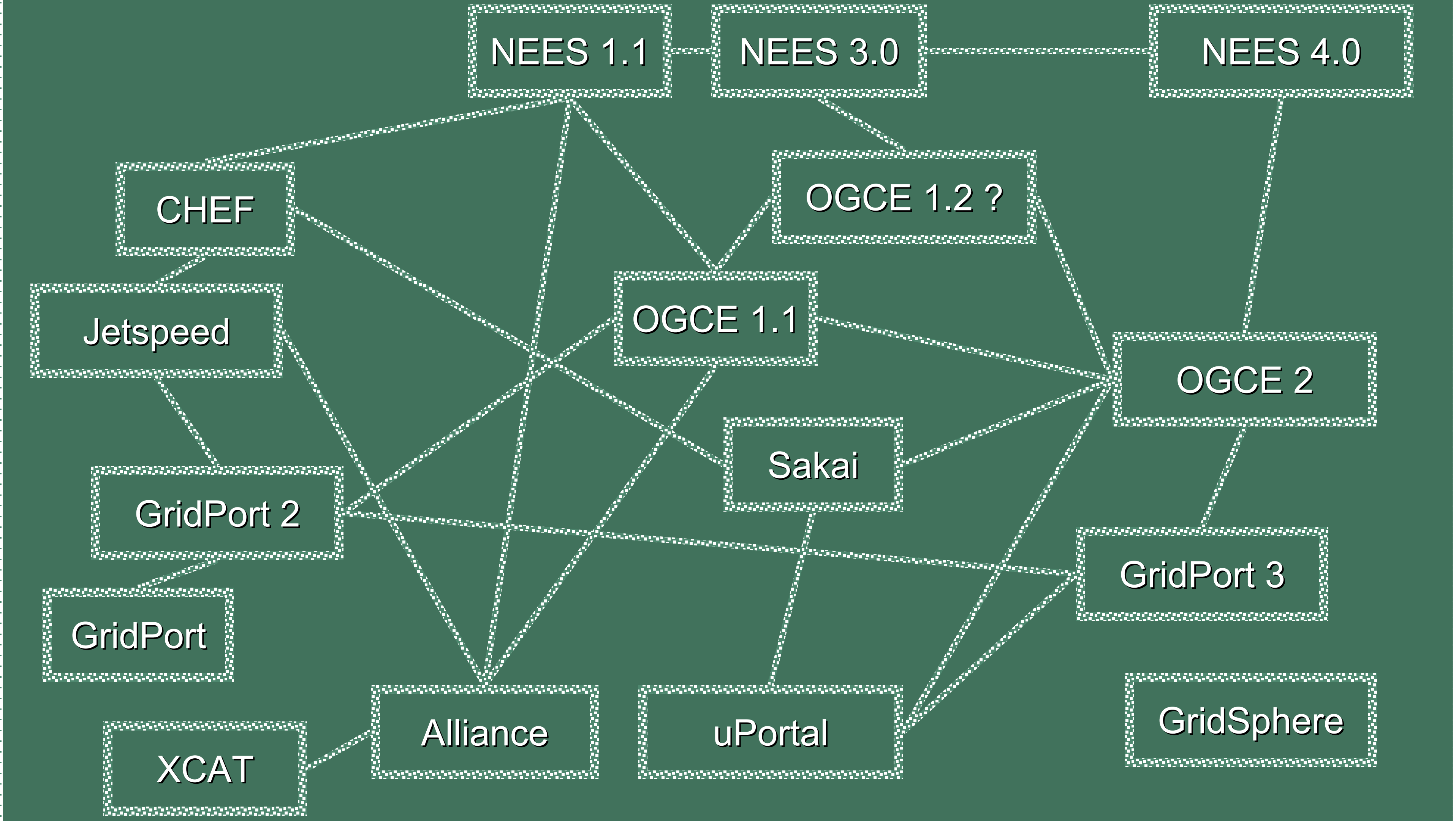




Grid Portals

- Provide single installation of Globus for many people - many of whom cannot install software on servers or their desktop
- Provide access to and help manage users credentials
- Provide a single-point of contact to allow many elements of an application to be assembled

Chalk Talk:School of Portals



NEES 1.1

NEES 3.0

NEES 4.0

CHEF

OGCE 1.2 ?

Jetspeed

OGCE 1.1

OGCE 2

Sakai

GridPort 2

GridPort 3

GridPort

GridSphere

XCAT

Alliance

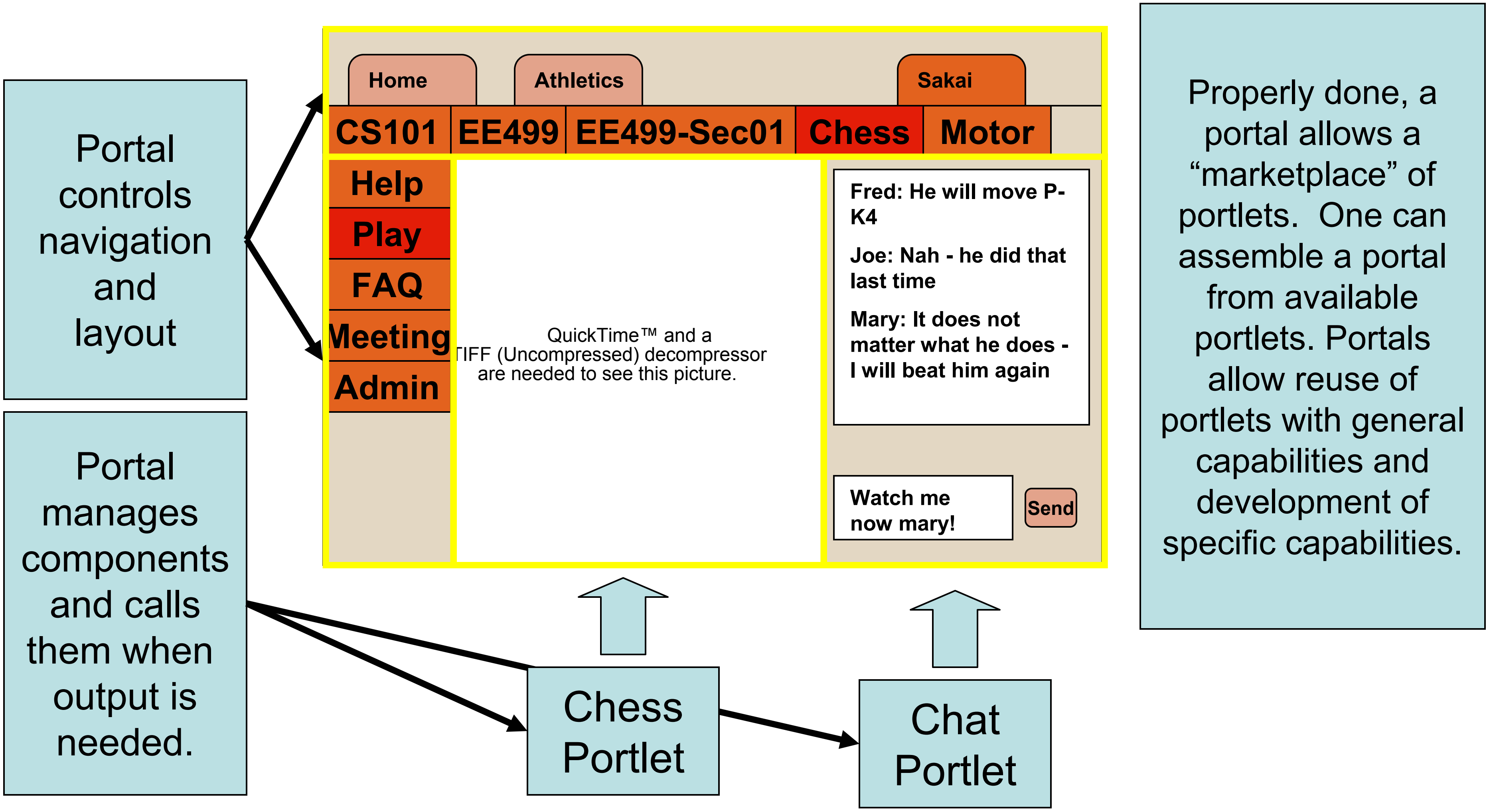
uPortal

Competition

Collaboration

Convergence

What is a Portal?



Portal controls navigation and layout

Portal manages components and calls them when output is needed.

Fred: He will move P-K4

Joe: Nah - he did that last time

Mary: It does not matter what he does - I will beat him again

Watch me now mary!

Properly done, a portal allows a “marketplace” of portlets. One can assemble a portal from available portlets. Portals allow reuse of portlets with general capabilities and development of specific capabilities.

How Do You Write a Portlet?

- Old way:
  - Write/Adapt some Perl - Gridport
  - Pick a Portlet API from: Jetspeed, CHEF, WebSphere, Plumtree, uPortal, Oracle, Microsoft (several versions), GridSphere
  - Have your code run on < 10% of the portals on the planet with little or no portability
- New Way:
  - Write JSR-168 (Portlet API) or WSRP (Web Services for Remote Portals)

Aside: Why JSR-168/WSRP ?

A slightly paraphrased conversation November 2003 at a Grid Portal meeting at Indiana University.

Charles Severance: (using his best expert from out-of-town voice) The architecture team from the CHEF project has done an evaluation of JSR-168 as best we can figure it out and have concluded that it is a bit too “HTML-oriented” - we want tools that are relevant beyond the web environment.

Geoffrey Fox (Indiana University): JSR-168 and WSRP will be standards - people will use them regardless of your opinion or your software.

Aside: Why JSR-168/WSRP ?

A slightly paraphrased conversation November 2003 at a Grid Portal meeting at Indiana University.

Charles Severance: (using his best expert from out-of-town voice) The architecture team from the CHEF project has done an evaluation of JSR-168 as best we can figure it out and have concluded that it is a bit too "HTML-oriented" - we want tools that are relevant beyond the web environment.

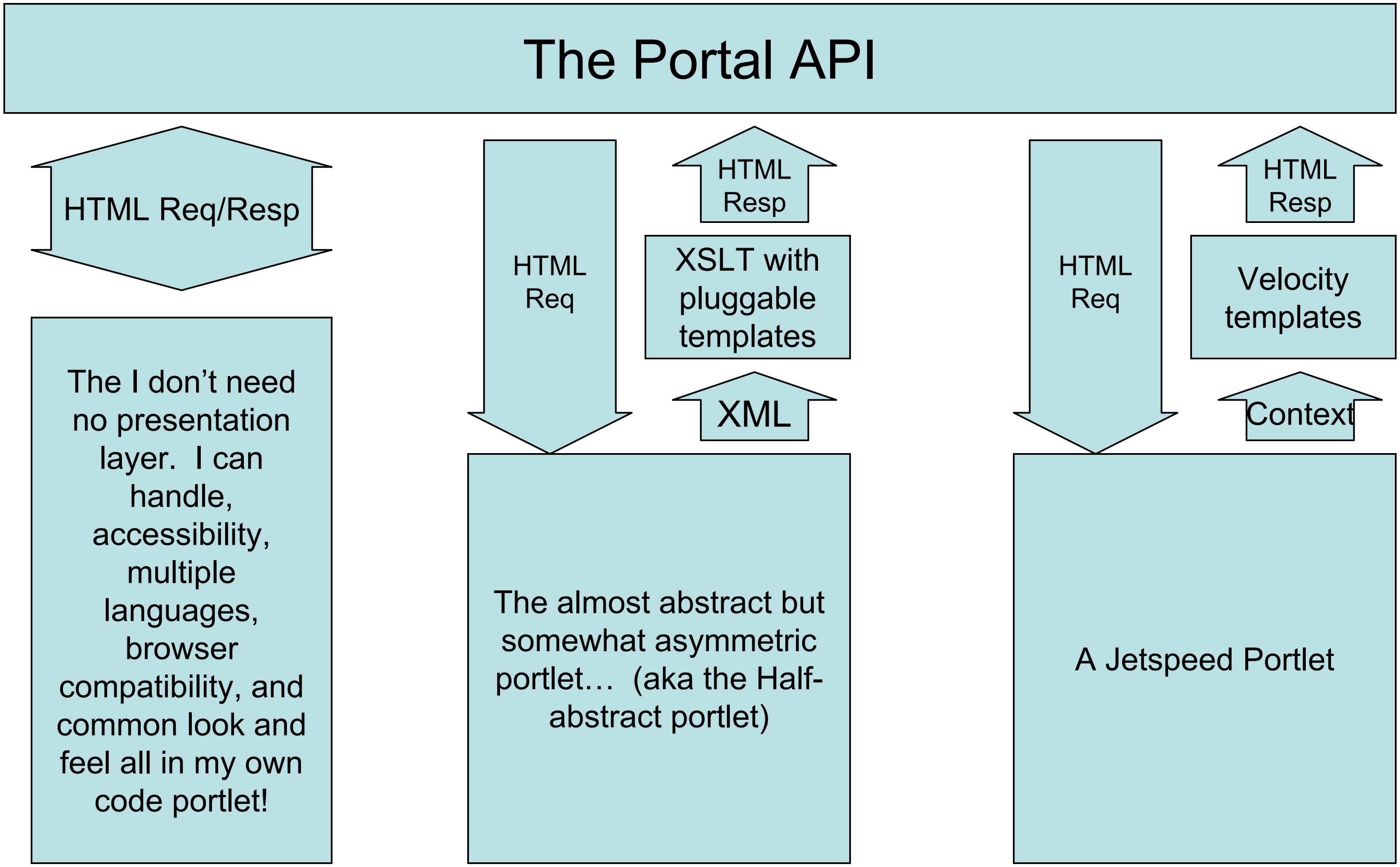
Geoffrey Fox (Indiana University): JSR-168 and WSRP will be standards - people will use them regardless of your opinion or your software.

➡ **Charles Severance: (pause) Good point. (sits down)**

How do you write a portlet?

- What presentation approach - raw HTML, XSLT, JavaBean-style presentation?
- Thick or thin?
- Do tools talk to a database?
- Web/Grid Services?
- Use an Applet?
- Use a Servlet?
- Is each portlet a stovepipe or do they cooperate?
- Do you rely on Portlet Services outside the standard or make your own cover API?
- What packages do you import?

Portlet Design Patterns - Presentation Layer - the old ways



The Portal API

HTML Req/Resp

The I don't need no presentation layer. I can handle, accessibility, multiple languages, browser compatibility, and common look and feel all in my own code portlet!

HTML Req

HTML Resp

XSLT with pluggable templates

XML

The almost abstract but somewhat asymmetric portlet… (aka the Half-abstract portlet)

HTML Req

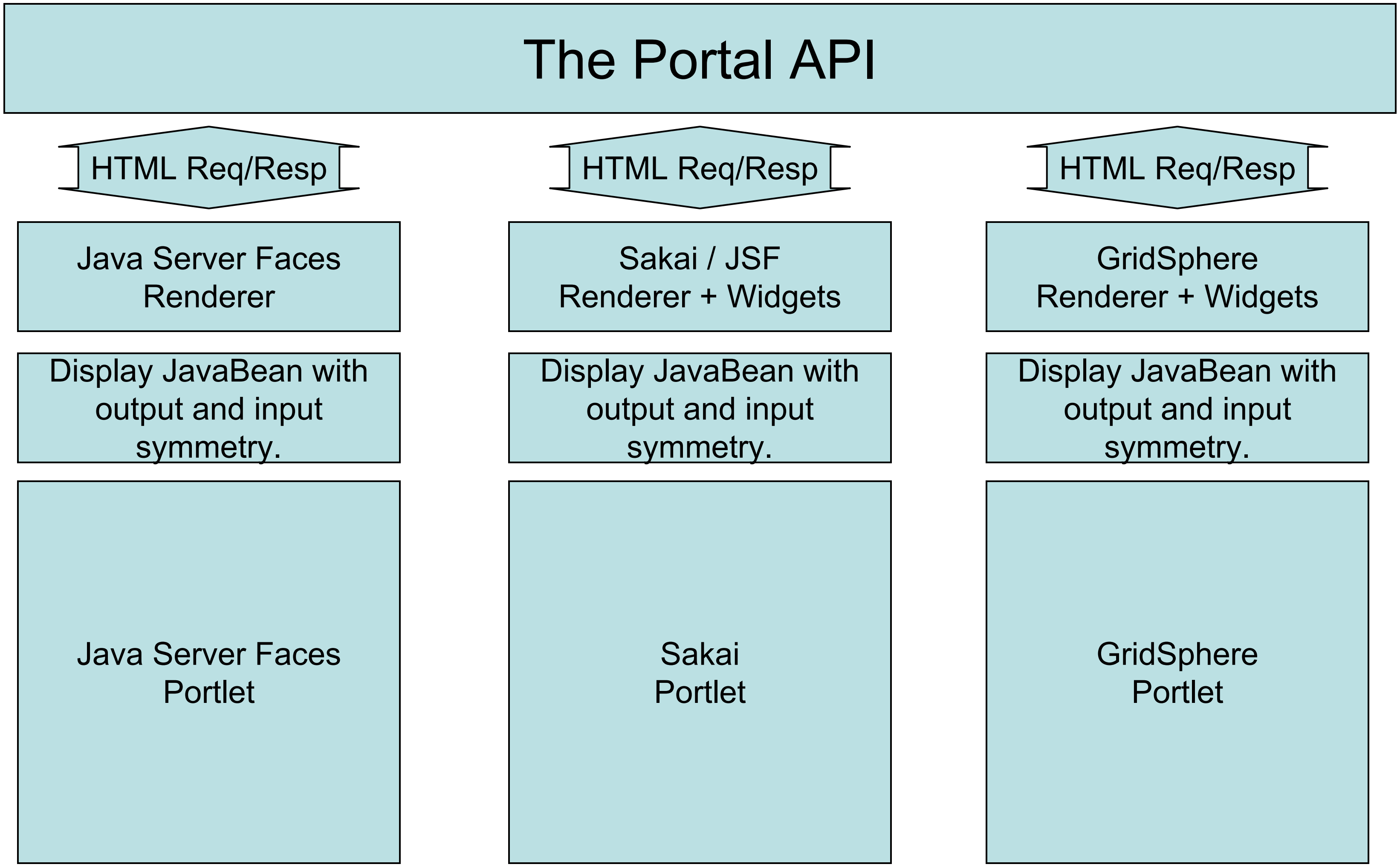
HTML Resp

Velocity templates

Context

A Jetspeed Portlet

Portlet Design Patterns - Symmetric Presentation Layers - the wave of the future



The Portal API

HTML Req/Resp

Java Server Faces
Renderer

Display JavaBean with output and input symmetry.

Java Server Faces
Portlet

HTML Req/Resp

Sakai / JSF
Renderer + Widgets

Display JavaBean with output and input symmetry.

Sakai
Portlet

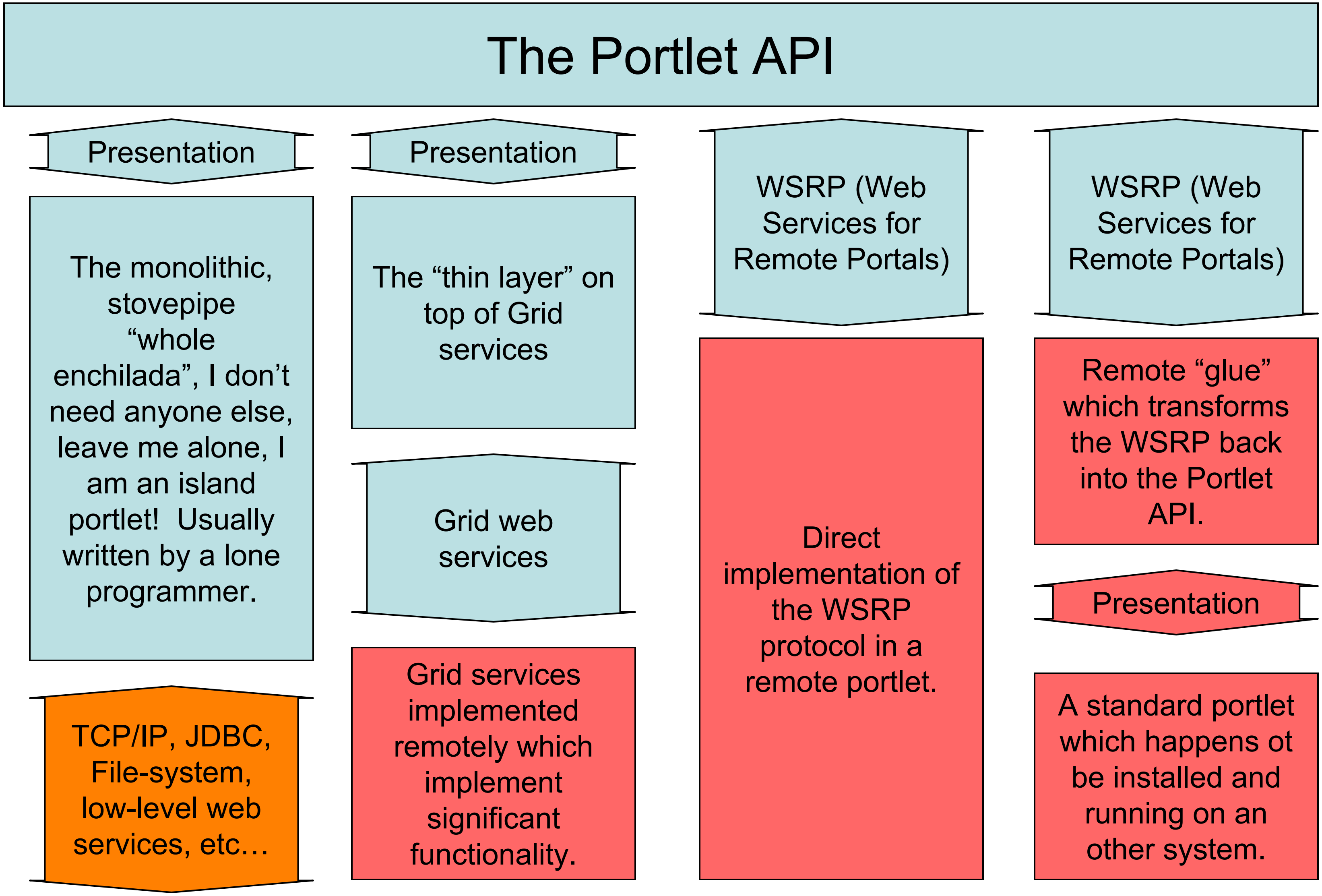
HTML Req/Resp

GridSphere
Renderer + Widgets

Display JavaBean with output and input symmetry.

GridSphere
Portlet

Portlet Design Patterns - Where does the information come from / go to?



The Portlet API

Presentation

The monolithic, stovepipe "whole enchilada", I don't need anyone else, leave me alone, I am an island portlet! Usually written by a lone programmer.

TCP/IP, JDBC, File-system, low-level web services, etc…

Presentation

The "thin layer" on top of Grid services

Grid web services

Grid services implemented remotely which implement significant functionality.

WSRP (Web Services for Remote Portals)

Direct implementation of the WSRP protocol in a remote portlet.

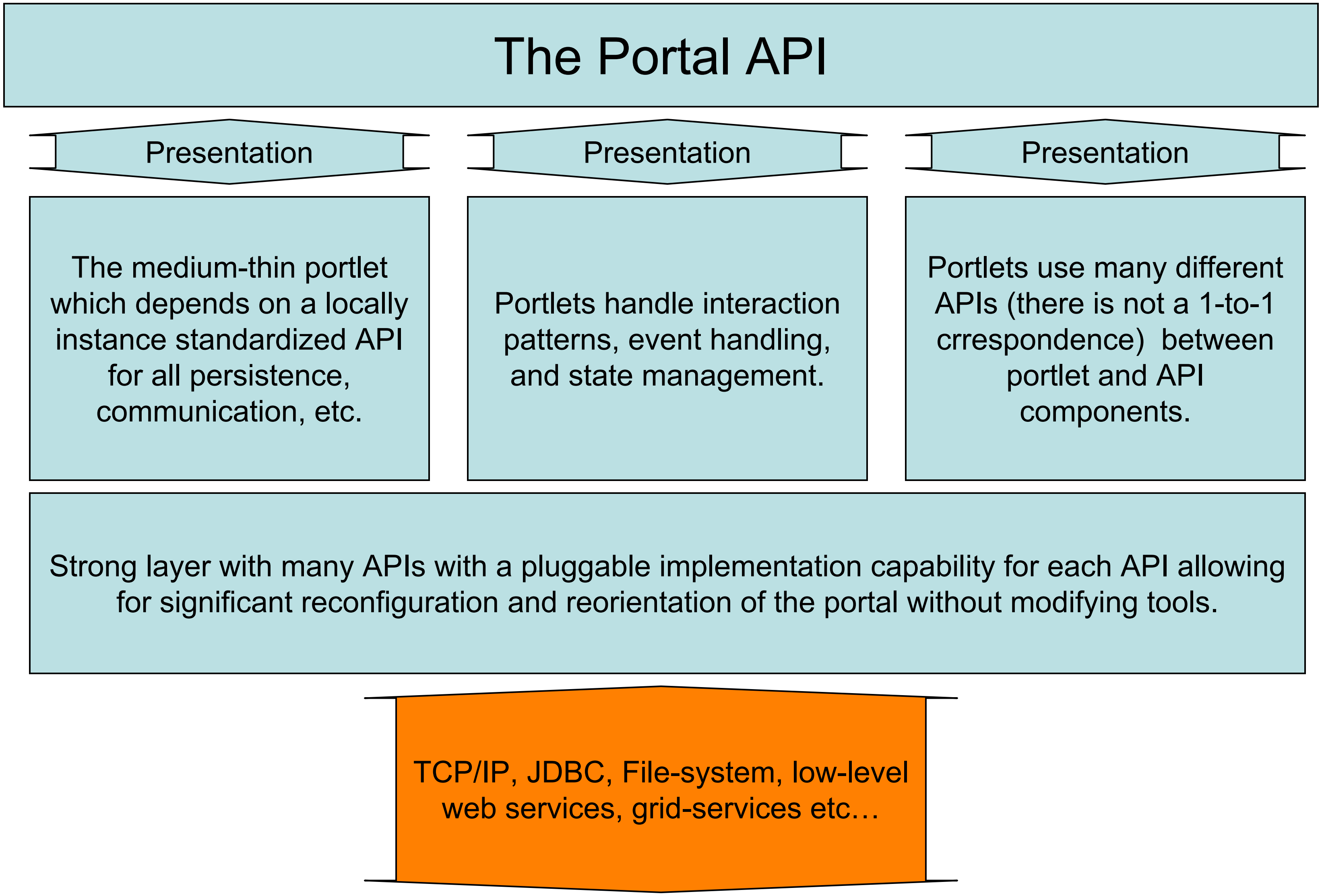
WSRP (Web Services for Remote Portals)

Remote "glue" which transforms the WSRP back into the Portlet API.

Presentation

A standard portlet which happens ot be installed and running on an other system.

Portlet Design Patterns - Where does the information come from / go to?



The Portal API

Presentation

Presentation

Presentation

The medium-thin portlet which depends on a locally instance standardized API for all persistence, communication, etc.

Portlets handle interaction patterns, event handling, and state management.

Portlets use many different APIs (there is not a 1-to-1 crrespondence) between portlet and API components.

Strong layer with many APIs with a pluggable implementation capability for each API allowing for significant reconfiguration and reorientation of the portal without modifying tools.

TCP/IP, JDBC, File-system, low-level web services, grid-services etc…

Portlet Design Choices

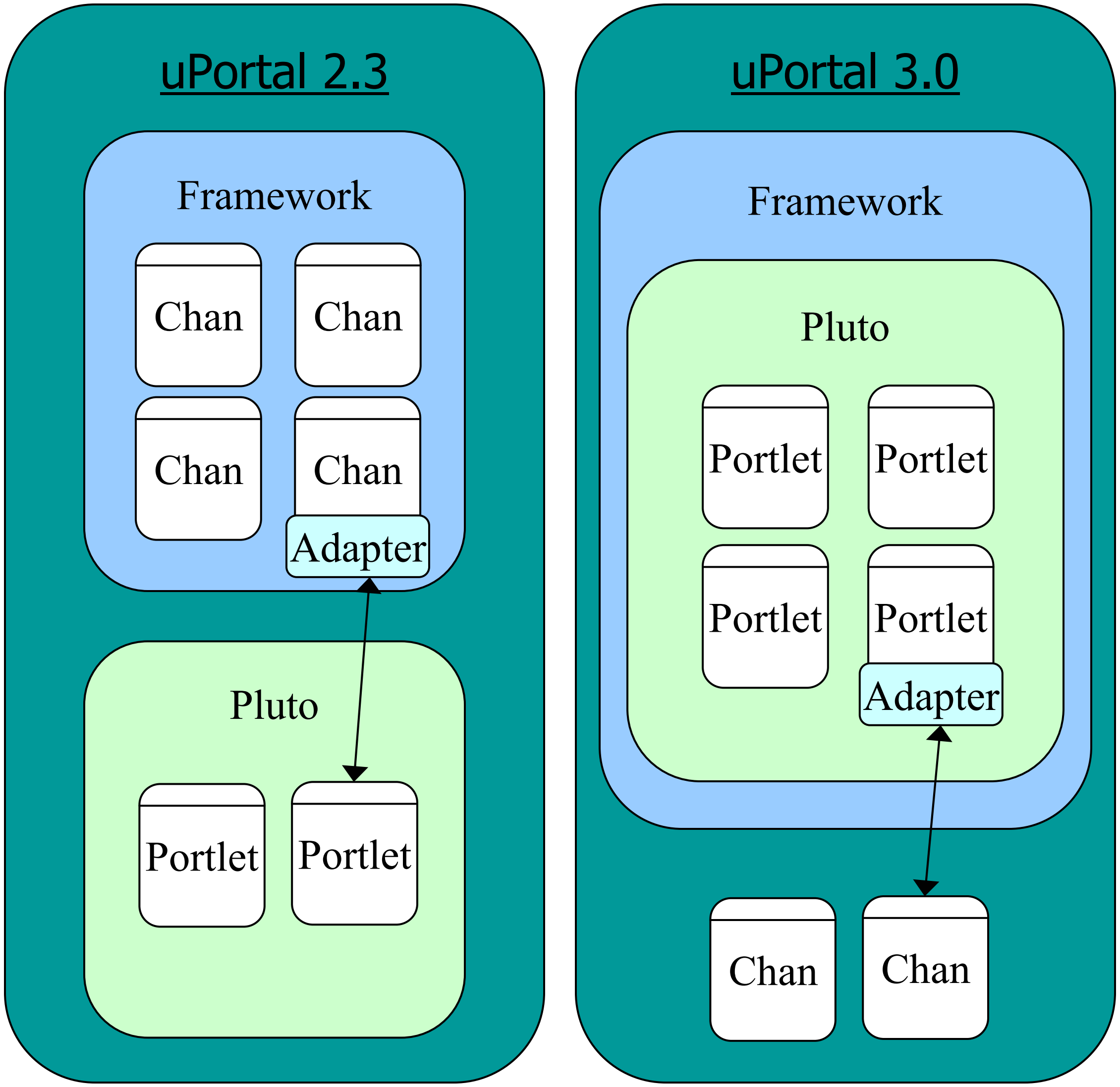
- All of these decompositions are useful given the constraints of the portlet implementation environment.
- There is a general trade-off between interdependence and reuse of code.
- Interesting questions
  - Do portlets hide behind an opaque API (Sakai-style)?
  - Where is the tradeoff between a portlet orchestrating a lot of web services and simply using WSRP?
  - Is it ever a good idea to write code directly to WSRP or should we just use JSR-168?
- In reality, given that a portal will be (should be) built from components from many sources - all variants are likely.

Exciting possibilities

- If the WSRP to JSR-168 API works out well (there is much left to understand), each portal should both be a WSRP producer and consumer.
- This leads to significant opportunities for creating federated portals with seamless ability to place functionality and integrate it in different ways.
- There is much work to do in this area to fully relalize this potential - it may lead to the need to a much richer WSRP based more on Grid concepts than web services concepts.

uPortal Portlet Roadmap

- uPortal 2.3
  - Support Portlets (JSR-168) via adapter
- uPortal 3.0
  - Implement Portlet Specification (JSR-168)
  - Support IChannel via adapter



What is next…

- Sakai Discussion - Building a portal with highly collaborative portlets with very strong presentation and API layering requirements.
- NEESGrid Disucssion - Building a real-life Grid portal with software from many collaborators and solving the problems of a community of scientists…