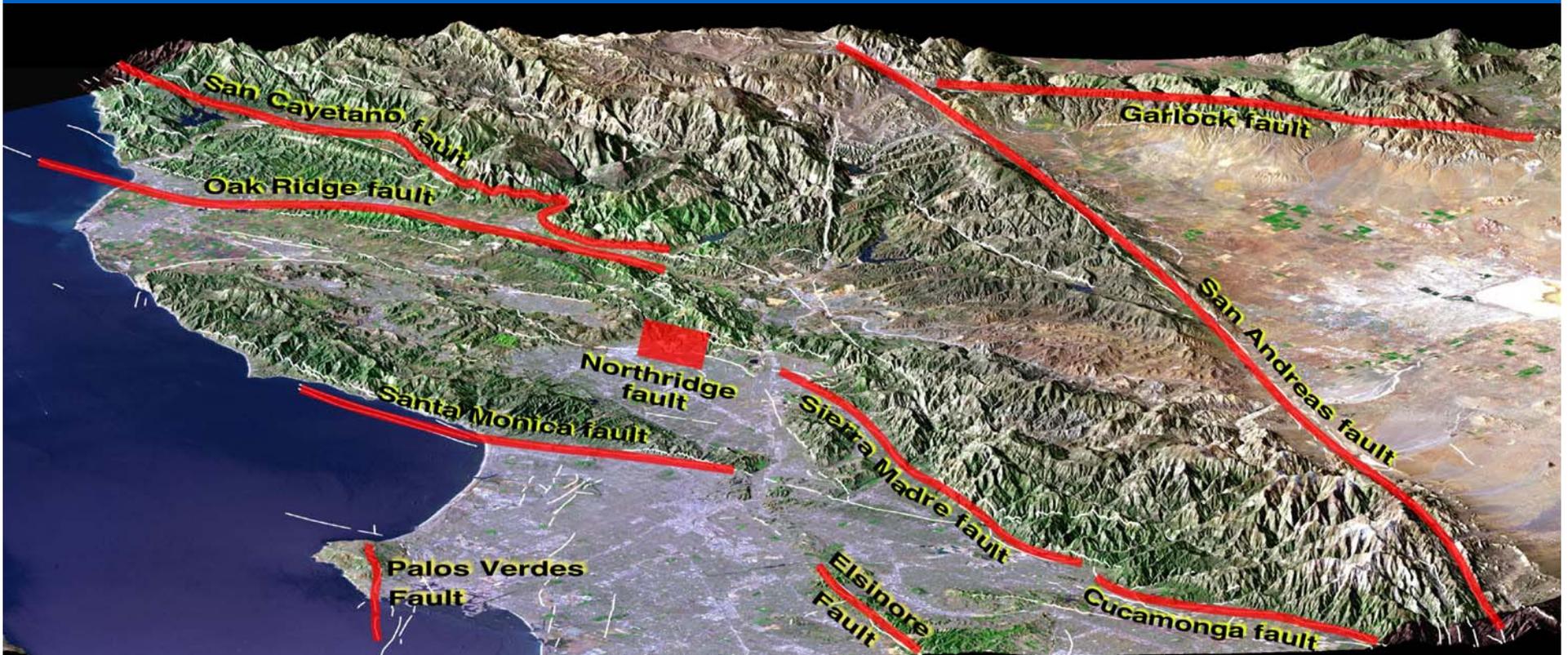


Developing SERVOnGrid: e- Science for Earthquake Simulation



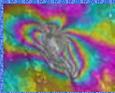
Marlon Pierce
Community Grids Lab
Indiana University

Solid Earth Modeling and Grids

What are the problems that
we are trying to solve?

Solid Earth Science Questions

1. **What is the nature of deformation at plate boundaries and what are the implications for earthquake hazards?**



2. **How do tectonics and climate interact to shape the Earth's surface and create natural hazards?**



3. **What are the interactions among ice masses, oceans, and the solid Earth and their implications for sea level change?**



4. **How do magmatic systems evolve and under what conditions do volcanoes erupt?**



5. **What are the dynamics of the mantle and crust and how does the Earth's surface respond?**



6. **What are the dynamics of the Earth's magnetic field and its interactions with the Earth system?**



From NASA's Solid Earth Science Working Group Report, *Living on a Restless Planet*, Nov. 2002

The Solid Earth is:

Complex, Nonlinear, and Self-Organizing

Relevant questions that Computational technologies can help answer:

1. How can the study of strongly correlated solid earth systems be enabled by space-based data sets?
2. What can numerical simulations reveal about the physical processes that characterize these systems?
3. How do interactions in these systems lead to space-time correlations and patterns?
4. What are the important feedback loops that mode-lock the system behavior?
5. How do processes on a multiplicity of different scales interact to produce the emergent structures that are observed?
6. Do the strong correlations allow the capability to forecast the system behavior in any sense?

Characteristics of Computing for Solid Earth Science

- **Widely distributed datasets** in various formats
 - GPS, Fault data, Seismic data sets, InSAR satellite data
 - Many available in state of art tar files that can be FTP'd
 - Provenance problems: faults have controversial parameters like slip rates which have to be estimated.
- **Distributed models and expertise**
 - Lots of codes with different regions of validity, ranging from cellular automata to finite element to data mining applications (HMM)
 - Simplest challenges are just making these codes useable for other researchers.
 - And hooking this codes to data sources
 - Some codes also have export or IP restrictions
 - Other codes are highly specialized to their deployment environments.
- **Decomposable problems** requiring interoperability for linking full models
 - The fidelity of your fault modeling can vary considerably
 - Link codes (through data) to support multiple scales

SERVOGrid Requirements

- **Seamless Access** to data repositories and computing resources
- **Integration of multiple data sources** including databases, file systems, sensors, ..., with **simulation codes**.
- **Core web services** for common tasks like command execution and file management.
- **Meta-data** generation, archiving, and access with extending openGIS (Geography as a Web service) standards.
- **Portals** with component model (portlet) for user interfaces and web control of all capabilities
- Basic Grid tools: **complex job management** and **notification**
- **Collaboration** to support world-wide work
 - "Collaboration" can range from data sharing to Narada-style AV.

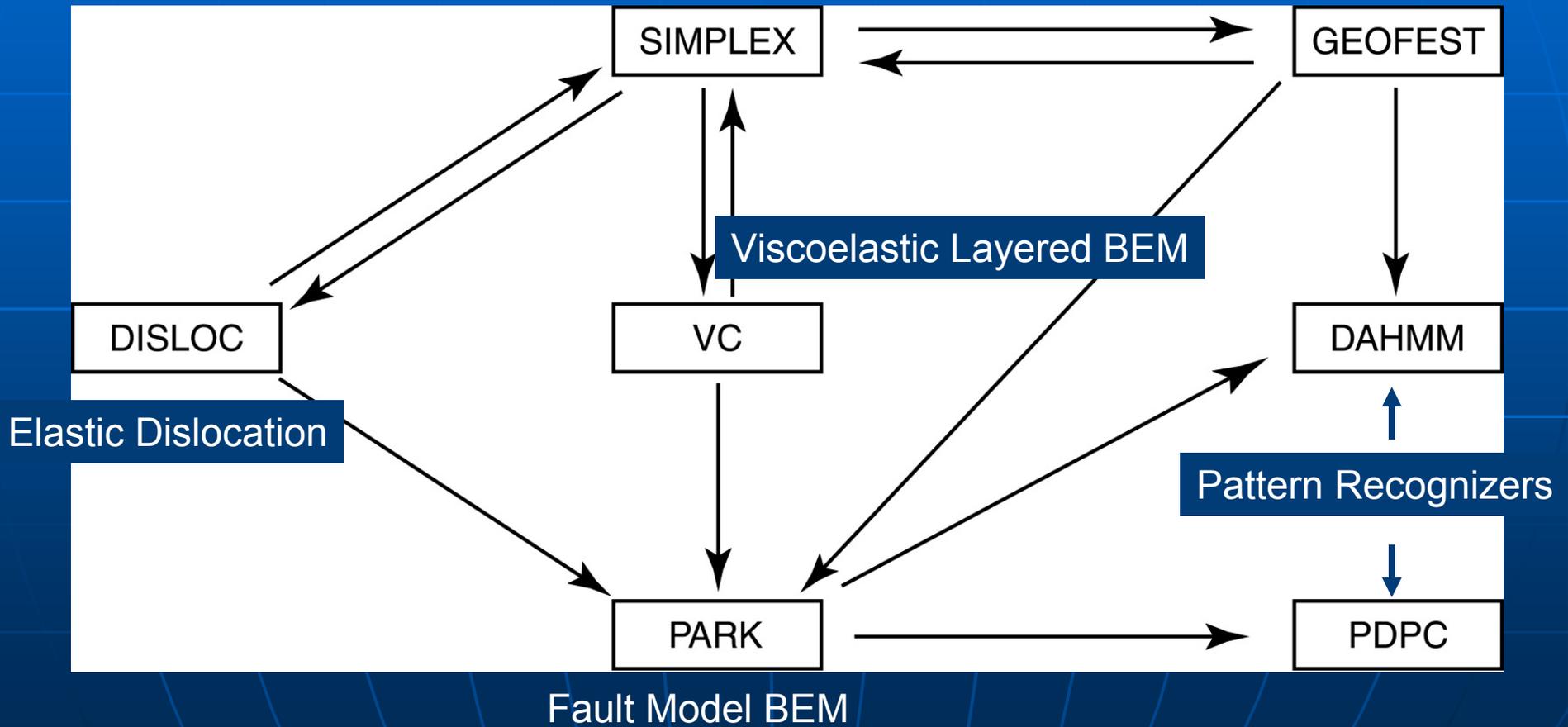
SERVOGrid Applications

- Codes range from simple “rough estimate” codes to parallel, high performance applications.
 - **Disloc**: handles multiple arbitrarily dipping dislocations (faults) in an elastic half-space.
 - **Simplex**: inverts surface geodetic displacements for fault parameters using simulated annealing downhill residual minimization.
 - **GeoFEST**: Three-dimensional viscoelastic finite element model for calculating nodal displacements and tractions. Allows for realistic fault geometry and characteristics, material properties, and body forces.
 - **Virtual California**: Program to simulate interactions between vertical strike-slip faults using an elastic layer over a viscoelastic half-space
 - **RDAHMM**: Time series analysis program based on Hidden Markov Modeling. Produces feature vectors and probabilities for transitioning from one class to another.
- Preprocessors, mesh generators: **AKIRA suite**
- Visualization tools: **RIVA, GMT, IDL**

SERVOGrid Codes, Relationships

Elastic Dislocation Inversion

Viscoelastic FEM



SERVO Data Sources

- **Fault Data**
 - Developed as part of the project
 - QuakeTables: <http://infogroup.usc.edu:8080>
- **Seismic data formats**
 - Available from www.scec.org
 - SCSN, SCEDC, Dinger-Shearer, Haukkson
- **GPS data formats**
 - Available from www.scign.org
 - JPL, SOPAC, USGS

SERVO: Solid Earth Research Virtual Observatory

- Framework arose from May 2002 NASA Workshop on Earth Science Computational Technologies
- SERVO team members
 - NASA JPL (lead), UC-Davis, UC-Irvine, USC, Brown, and Indiana University
- Team areas of expertise
 - Geology (Irvine)
 - Computational earthquake modeling (JPL, Davis, Brown)
 - Federated database design and semantic modeling (USC)
 - High performance computing (JPL, Davis)
 - Grids, Web services, and portals (Indiana)

Building Earthquake Modeling Services

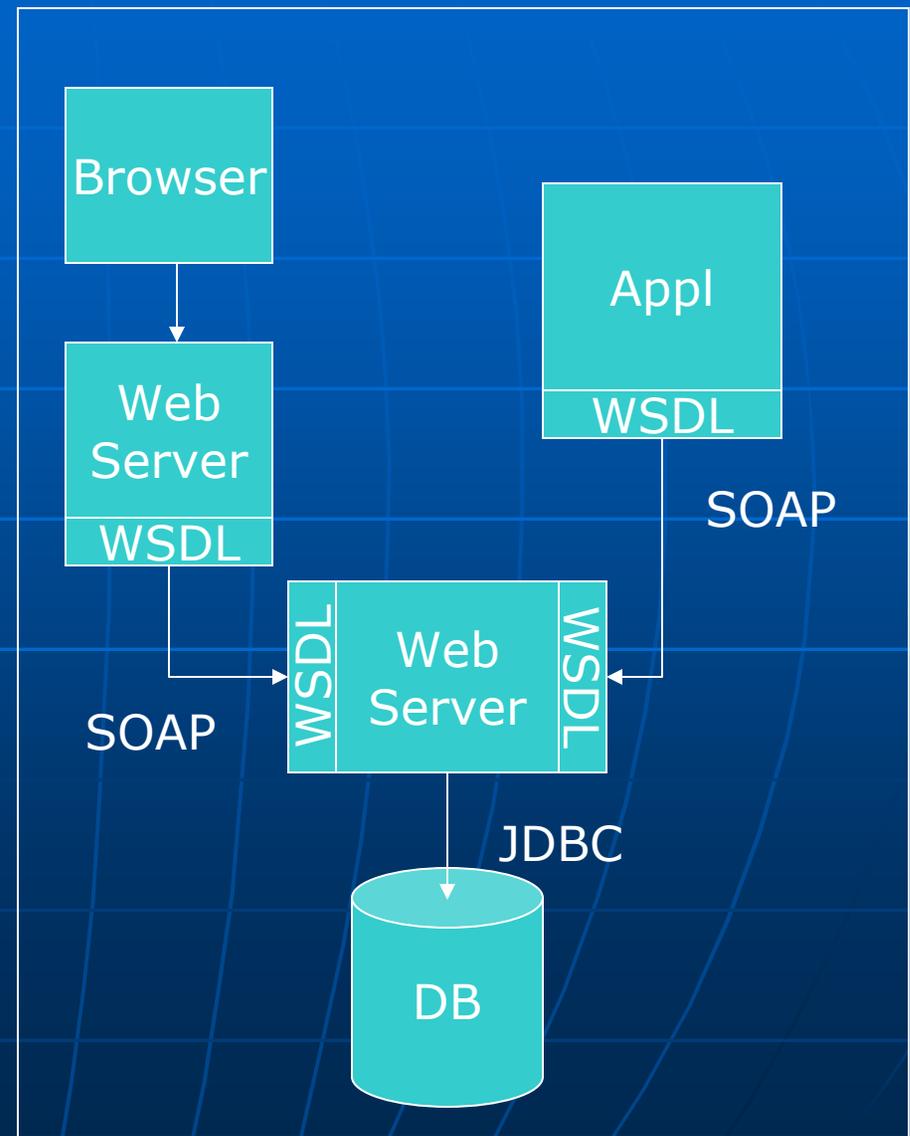
What did we do, and what did
we learn?

(i)SERVO Web (Grid) Services

- **Programs:** All applications wrapped as Services using proxy strategy
- **Job Submission:** support remote batch and shell invocations
 - Used to execute simulation codes (VC suite, GeoFEST, etc.), mesh generation (Akira/Apollo) and visualization packages (RIVA, GMT).
- **File management:**
 - Uploading, downloading, backend crossloading (i.e. move files between remote machines)
 - Remote copies, renames, etc.
- **Job monitoring**
- **Workflow:** Apache Ant-based remote service orchestration (NCSA)
 - For coupling related sequences of remote actions, such as RIVA movie generation.
- **Data services:** support remote data bases and query construction
 - XML data model being adopted for common formats with translation services to "legacy" formats.
 - Migrating to Geography Markup Language (GML) descriptions.
- **Metadata Services:** for archiving user session information.

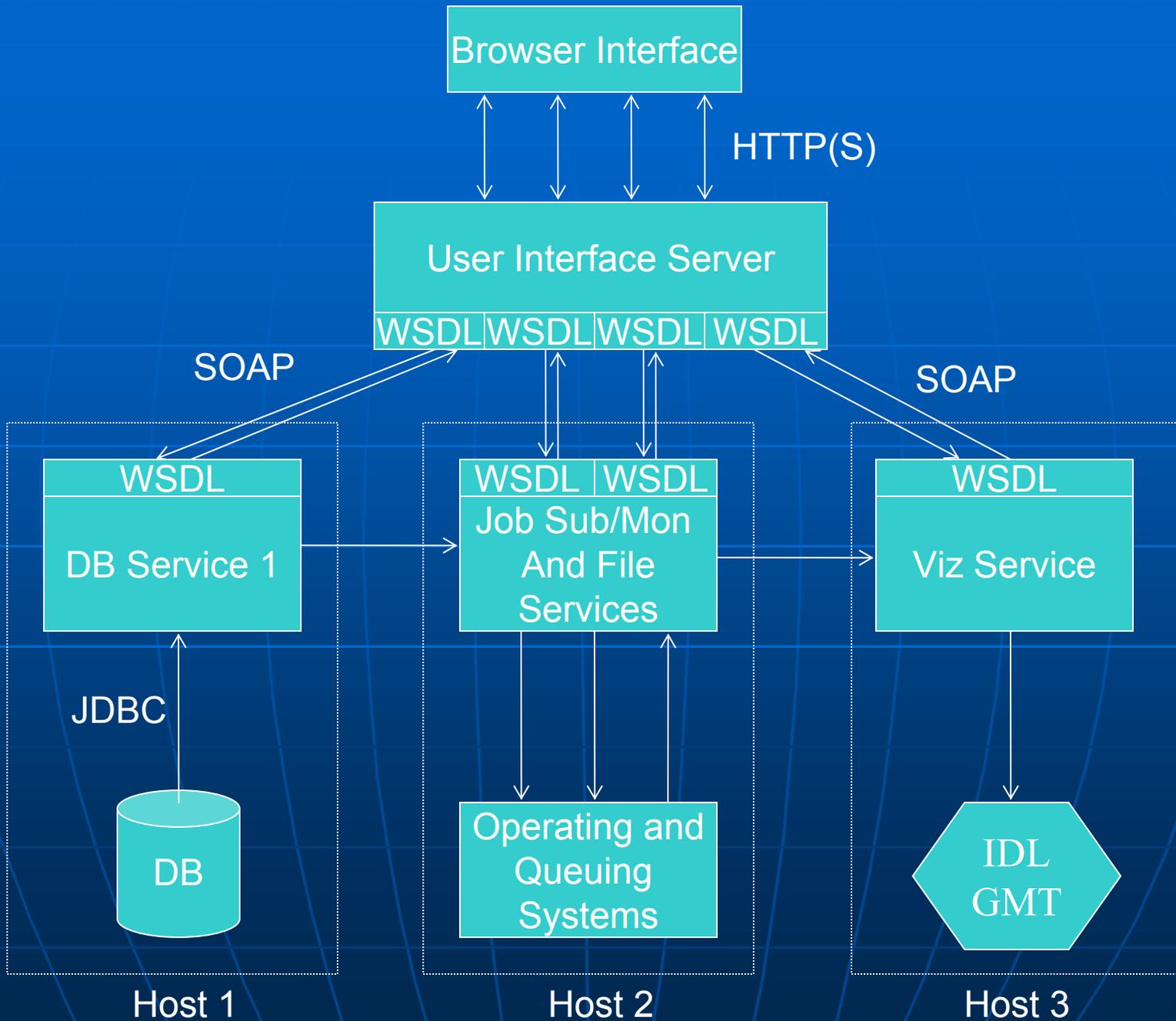
What Are Web Services? Once Again...

- Web Services are not web pages, CGI, or Servlets
- Web Services framework is a way for doing distributed computing with XML.
 - **WSDL**: Defines interfaces to functions of remote components.
 - **SOAP**: Defines the message format that you exchange between components.
- XML provides cross-language support
- Suitable for both human and application clients



Web Service Architectures

- SERVOnGrid is built around the **Service Oriented Architecture** Model.
- Constituent pieces
 - Remotely accessible services
 - Capabilities are defined through interface definition languages (WSDL).
 - Accessible through messages and protocols (SOAP).
 - Implementations may change but interfaces must remain the same.
 - Client applications access remote services.
 - **Client hosting environments**
 - Web Portals are an example.
- Going beyond services
 - Semantic descriptions for service and information modeling.
 - Programming/orchestration tools for connecting distributed services.



SERVOGrid Required Services

- Computing Grid services
 - Remote command execution/job submission, file transfer, job monitoring.
 - These services
 - We may develop these using any number of toolkits
 - Globus, Apache Axis, GSoap.
- Data Grid services
 - Access data bases and other data sources (faults, GPS, Seismic records).
- Information Grid services
 - Metadata management

Execution Grid Service Examples

- You almost always need to perform several remote steps.
 - “Job management” services
 - Don’t call it workflow
- More interesting combining several services into a single meta-service.
 - Run Disloc, when done move the output from darya to danube, generate a PDF image of the output using GMT, then pull the output back to the client browser for display.
- Simple solution: Apache Ant build tool.
 - Not a full fledged programming language, but it can do most of the workflow problems I encounter, and is easy to extend.
 - Tasks are expressible in XML, so you can build authoring tools to hide antisms and validate scripts.
 - Open source and because it is generally applicable, likely to outlive most workflow tools.

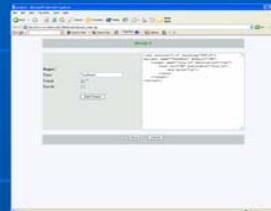
Hot Deploying Applications

- One of the challenges we have is that new codes need to be added, applications upgraded, etc.
 - It would be nice to give more control to the application developer rather than relying on the portal/service/grid folks.
 - A path fraught with peril, but we forge ahead.
- The Ant web service approach enables a few other nice things:
 - You can assemble remote build.xml templates from libraries of task templates.
 - And you can map the XML to HTML to generate the new interfaces.

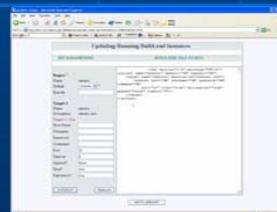
Templating Applications and Generating Interfaces

- Users fill in templates through web forms
- Execution services then invoke scripts.
 - Ant is a good way to wrap applications.
- Template authoring tools simplify deployment of new wrapped services.
- Templates used to automate user interface generation.

Author and Publish
Template



Execute Services
Through Generated
Interfaces

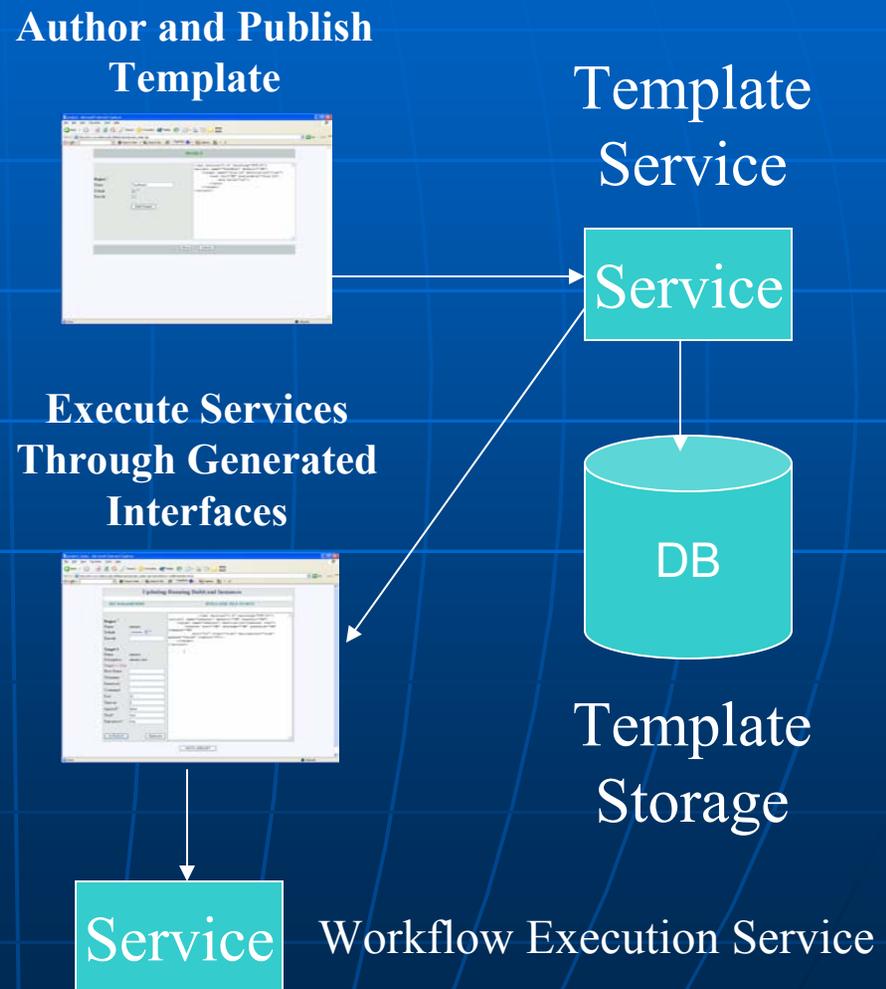


Workflow Execution Service

Template
Service



Template
Storage



Some Screen Shots of Prototype

project - Microsoft Internet Explorer

Address http://toro.ucs.indiana.edu:8086/project/project_index.jsp

PROJECT

Project +

Name TestNew2

Default *

Basedir

Edit Project

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="TestNew2" default="ON">
  <target name="/bin/lis" description="list">
    <exec dir="ON" executable="/bin/lis">
      <arg value="on"/>
    </exec>
  </target>
</project>
```

Save Cancel

project_index - Microsoft Internet Explorer

Address http://toro.ucs.indiana.edu:8080/project/project_index.jsp?xml=sshexec.xml&fromindex=true

Updating-Running Build.xml Instances

SET PARAMETERS BUILD.XML FILE TO RUN

Project +

Name sshexec

Default sshexec *

Basedir

Target-1

Name sshexec

Description sshexec test

Target-1 / Scp

Host Name

Username

Password

Command

Port 22

Timeout 0

Append? false

Trust? true

Failonerror? true

<< RUN >> Build.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="sshexec" default="ON" basedir="ON">
  <target name="sshexec" description="sshexec test">
    <sshexec host="ON" username="ON" password="ON"
      port="22" trust="true" failonerror="true"
      append="false" timeout="0"/>
  </target>
</project>
```

GOTO JOB-LIST

Some Ant Web Service Strengths and Weaknesses

■ Good

- Several built in features that can be used to interact with files, directories and executables.
- Easy to extend
 - Ant tasks may be web services
 - They may be Java COG calls to grids
 - Or ssh/scp
- Can be easily templated with properties

■ Bad to Ugly

- Need an external event model since tasks can take minutes to hours to days to complete.
 - Callback service
 - Reliable messaging
- Need a way to handle remote failures.
- Not high performance.
- Not a full-fledged programming language or workflow engine.
- Not good for streaming data.
 - www.hpsearch.org

Other Lessons Learned

- Web service **performance** is not an issue when used to invoke services that take hours to complete.
- **Reliability** is a larger problem.
 - Need monitoring/heartbeat services.
- **Information systems** still have a long way to go.
 - **UDDI** is part of WS-I but has/had some well known limitations.
 - **WS-Discovery** has some interesting concepts but is too specialized to ad-hoc networks.
 - **Peer-to-peer** systems provide many useful concepts like discovery and caching.
 - **Semantic Web** provides powerful resource descriptions that could be exploited.

GML Data Models and Web Services for GPS and Earthquake Catalogs

Using Geographic Information
System community
standards.

SERVO Applications

- Several SERVO codes work **directly with observational** data.
- Examples discussed at ACES include
 - GeoFEST, VirtualCalifornia, Simplex, and Disloc all depend upon fault models.
 - RDAHMM and Pattern Informatics codes use seismic catalogs.
 - RDAHMM primarily used with GPS data
- **Problem:** We need to provide a way to integrate these codes with the online data repositories.
 - QuakeTables Fault Database was developed
 - What about GPS and Earthquake Catalogs?
 - Many formats, data available in tars or files, not searchable, not easy to integrate with applicaitons
- **Solution:** use databases to store catalog data; use XML (**GML**) as exchange data format; use **Web Services** for data exchanges, invoking queries, and filtering data.

Geographical Information Service (GIS) Data Formats and Services

- **OpenGIS** Consortium is an international group for defining GIS data formats and services.
- Main data format language is the XML-based **GML**.
 - Subdivided into schemas for drawing maps, representing features, observations, ...
- **First Step**: design GML schemas and build specialized Web Services for GPS and Earthquake data.
- OGC also defines services.
 - Services include Web Features Services, Web Map Services, and similar.
 - These are currently pre-Web Service, based on HTTP Post, but they are being revised to comply with WS standards.
- **Next Step**: Implement OGC compatible Web Services for this problem.
 - Also build services to interact with QuakeTables Fault DB.

GML and Existing Data Formats

- GPS or seismic data used in this project are retrieved from different URLs and have different text formats.
- Seismic data formats
 - SCSN, SCEDC, Dinger-Shearer, Haukkson
- GPS data formats
 - JPL, SOPAC, USGS
- We defined 2 GML Schemas to unify these
 - <http://grids.ucs.indiana.edu/~gaydin/servo>
- A summary of all supported formats and data sources can also be found there.

So We Built It

- First version of the system available
 - Tried XML databases but performance was awful
 - Currently database uses MySQL
- Download results are in GML, but we can convert to appropriate text formats.

Download Catalogs

[Earthquake Catalogs](#)

[GPS Catalogs](#)

Convert Catalogs to GML

[Earthquake Catalogs To XML](#)

[GPS Catalogs To XML](#)

Insert Catalogs into MySQL Database

[Insert Earthquake Catalogs](#)

[Insert GPS Catalogs](#)

Search Databases

[Search Earthquake Catalogs](#)

[Search GPS Catalogs](#)

View Downloaded Catalogs

[View Earthquake Catalogs](#)

[View GPS Catalogs](#)

Manage Databases

[View-Delete Catalogs in the DB](#)

Search DB For Earthquake Catalogs

Search Earthquake Catalogs

Search Parameters

Start Date Year:

End Date Year:

Latitude Min: Max:

Longitude Min: Max:

Magnitude Min: Max:

Depth Min: Max:

NPH Min: Max:

RMS Min: Max:

Event ID

Quality

Submit

Reset

Constructed Query

```
/CatalogContainer/Catalog[TimeOfDate/Date{@Year=1932} and TimeOfDate/Date{@Year=1936} and Latitude/degrees[text()]>=33 and Latitude/degrees[text()]<=56 and Longitude/degrees[text()]>=114 and Longitude/degrees[text()]<=118 and Magnitude[text()]>=0 and Magnitude[text()]<=5 and Quality[text()='A']]
```

Date	Time	Latitude	Longitude	Quality	Magnitude	Depth	NPH	RMS	Event ID
1932-07-22	00:15:38.53	33 47.94	117 20.76	A	2.8	6.0	10	0.13	3359131
1933-11-20	13:27:51.16	33 45.84	118 14.76	A	2.4	6.0	6	0.38	3361683
1934-08-07	09:32:08.79	33 45.24	117 55.86	A	1.5	6.0	6	0.6	3362329
1934-08-28	10:49:31.67	34 16.56	118 12.90	A	1.8	1.77	6	0.02	3362383

Start Over

Search XML DB For GPS Catalogs

Search GPS Time Series Catalogs

Select a format to search

JPL-Filtered
 JPL-Point

Select the fields to be included in the search results

Decimal Year (eg. "2001.6427")
 String Year (eg. "01AUG23")
 Estimate
 Sigma

Submit Reset



Search filtered Time Series Catalogs

SEARCH PARAMETERS

Site to search: 7ODM

Component: ALL
 LAT LON RAD

Start Date: Year: 1932

End Date: Year: 1936

Decimal Start Date:

Decimal End Date:

Minimum Estimate: Maximum Estimate:

Minimum Sigma: Maximum Sigma:

Submit Reset



Constructed Query

```
/GPSDataContainer/GPSData[Site[text()='7ODM'] and Component[text()='LAT'] and Date  
[@DecimalYear>=2002] and Date[@DecimalYear<2004]}
```

Date	Decimal Year	Estimate	Sigma	Site Name	Component
02JAN01	2002.0013	-1.07118203063443	0.130505009739579	7ODM	LAT
02JAN02	2002.0042	-1.31999660592792	0.113830311236203	7ODM	LAT
02JAN03	2002.0068	-1.26575007898108	0.123528157135876	7ODM	LAT
02JAN04	2002.0095	-1.77219195902773	0.11664443761875	7ODM	LAT

Integration of Other Applications

- The screen shot fragments show part of the user interface.
- The important thing to note, though, is that the “downloaded results” go to the application, not the user’s desktop.
- We do this through a filtering process to convert to the expected file format for that code.
 - And push data out to the necessary execution host.
 - A provisional approach.
- In moving to a fully GIS-based system, this approach will also allow us to integrate in third party tools.

Fault Quest: QuakeTables+OGC Web Map Service Demo

The screenshot displays the deegree WMS Client interface. On the left, a vertical sidebar contains a list of projects: California, Indiana, US, Cite, Europe, and Osnabrück. The main map area shows a geographical view of the western United States, including Oregon, Idaho, Nevada, Utah, California, and Arizona. Red lines represent fault segments, and red circles with numerical values (e.g., 2.2, 2.5, 2.7) represent earthquake epicenters. Labels for cities like Sacramento, San Francisco, San Jose, Fresno, Los Angeles, Long Beach, Riverside, La Ana, and Drexel are visible. The interface includes a control bar at the top with buttons for ZOOMIN, ZOOMOUT, RECENTER, INFO, REFRESH, and RESET. A legend on the right lists layers such as US:fault_label, US:FaultSegments, US:earthquake_label, US:earthquake, US:cities_label, US:cities, and US:states. Below the legend, there are sections for 'Available Layers' and 'Selected Layers', each with an 'add' button and a 'remove' button. The bottom of the interface shows the copyright information: (c) deegree WMS 2003 and (c) deegree WMS 1.1.1 Sun Jul 11 14:49:14 EST 2004.

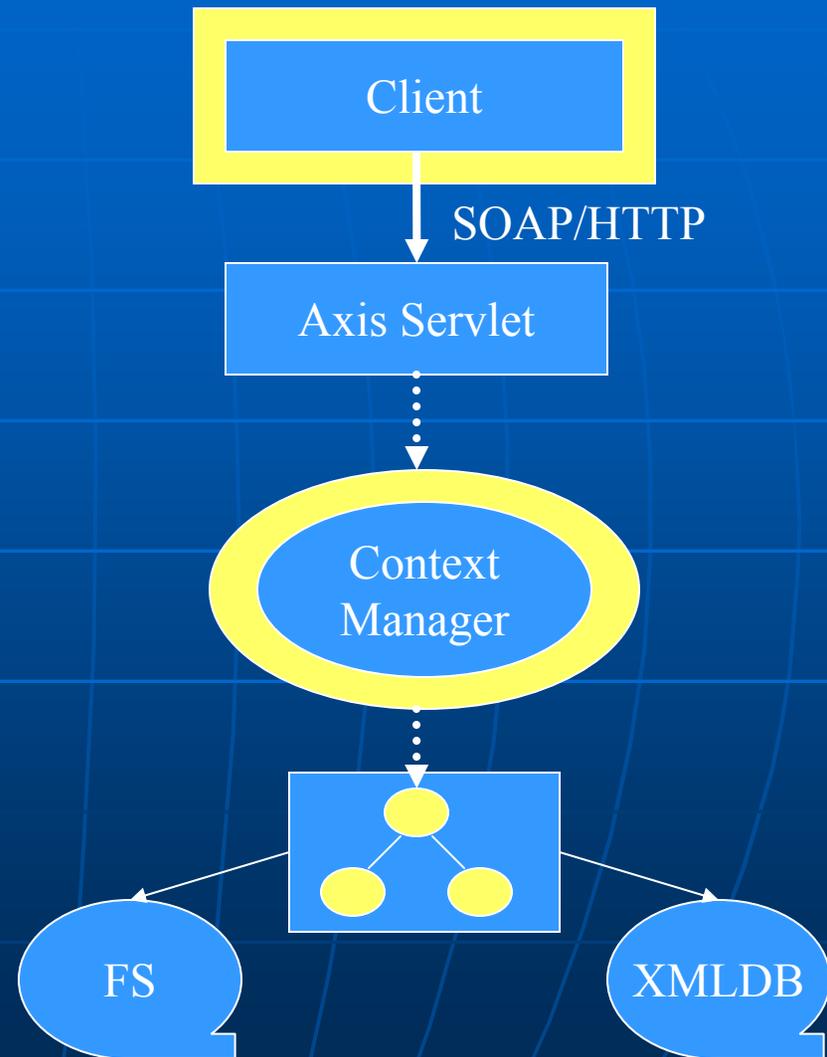
<http://rio.ucs.indiana.edu:8080/wmsClient/>

Metadata Management

- Common problems in computational science:
 - Where are the input and output files?
 - When was this created?
 - What parameters did I use to create this output?
 - What version of the code?
 - Is there a validation scenario for this code?
- These are all metadata problems.

Context Management Service

- Metadata may be organized into tree-like structures (see figure).
 - Context nodes hold one or more leaves and nodes.
 - Leaves are name/value pairs.
- We usually need to create arbitrary trees.
 - Represent with recursive XML schema.
 - Search with XPath.
- Context data storage and access is retrievable through a web service interface.
- Context data storage is implementation dependent but service interface is independent.



Lessons Learned

- Don't overlook some simple problems
 - The scientific computing community doesn't have extensive experience with databases.
- XML databases still have a long way to go.
 - We tried Berkley Sleepycat and Xindice
 - If you are ambitious, this might be a good research area.
 - Otherwise, stick with RDBs.

Computing Web Portals

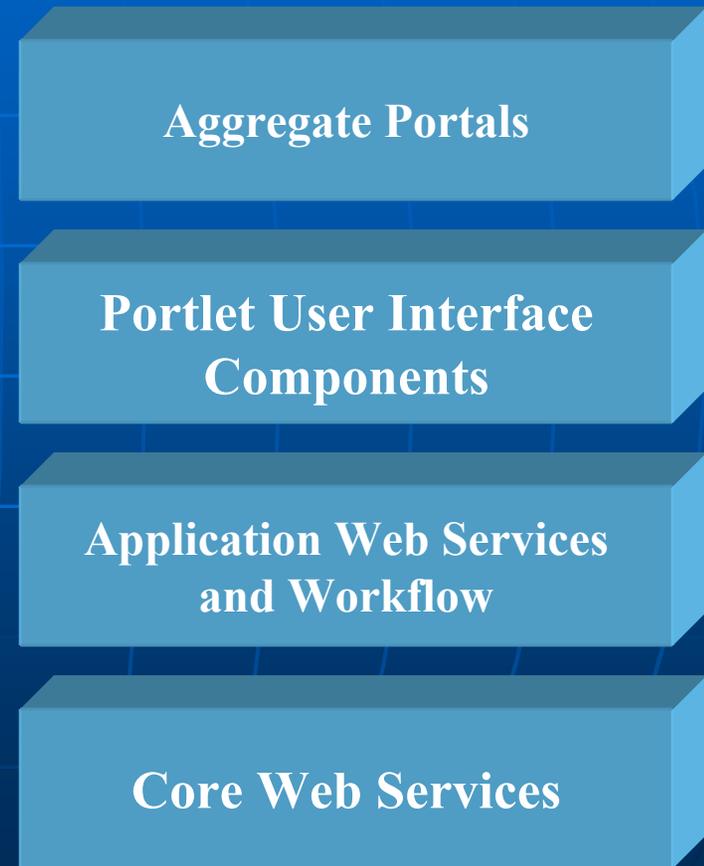
Building user interface
environments for e-Science

QuakeSim Portal for SERVOGrid

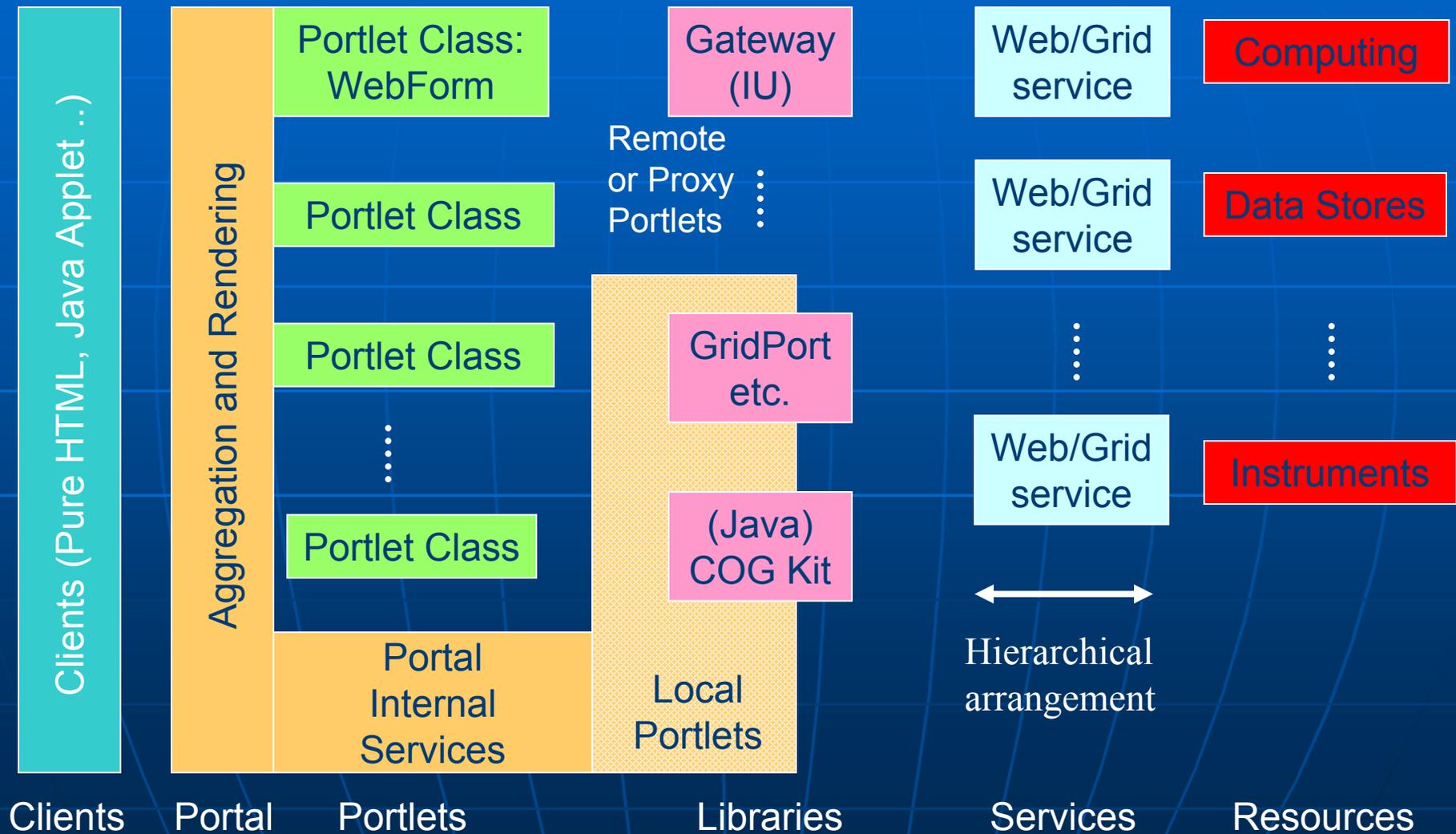
- The services we have previously described are headless.
 - WSDL descriptions are all you need to create client stubs (if not client applications).
- The QuakeSim portal effort aggregates these service interfaces into a *portal*.
 - Customizable displays, access controls to services, etc.
- QuakeSim is just one of many, many such projects.
- Challenge is to develop reusable portal components

Computational Web Portal Stack

- Web service dream is that core services, service aggregation, and user interface development decoupled.
- How do I manage all those user interfaces?
- Use portlets.



Portal Architecture



Why Are Portlets a Good Idea?

- You don't have to **reinvent** everything
 - Makes it easy (but not effortless) to **share portal components** between projects.
 - So you can pull in portlets from all the other earthquake grid projects.
- You can easily **combine** a wide range of **capabilities**
 - Add document managers, collaboration tools, RSS news lists, etc for your portal users.

Lessons Learned: Portals

- Developing good user interfaces is a lot of work.
 - Effort doesn't scale: how do you simplify this for computational scientists to do it themselves without lots of background in XML, Java, portlets, etc?
- Portal interfaces have advantages and disadvantages.
 - Everyone has a browser.
 - But it has a limited widget set, a limited event model, limited interactivity.
 - You can of course overcome a lot of this with applets.
- Following the service model, you can in principal use any number of GUIs
 - Browsers are not the only possible clients.
 - Web service interoperability means that Java Swing apps, Python, Perl GUIs are all possible, but this has not been fully exploited.

Learning and Using GeoFEST

Finite Element Software for
Analysis of Tectonic Strain
and Stress: An Example to
illustrate services.

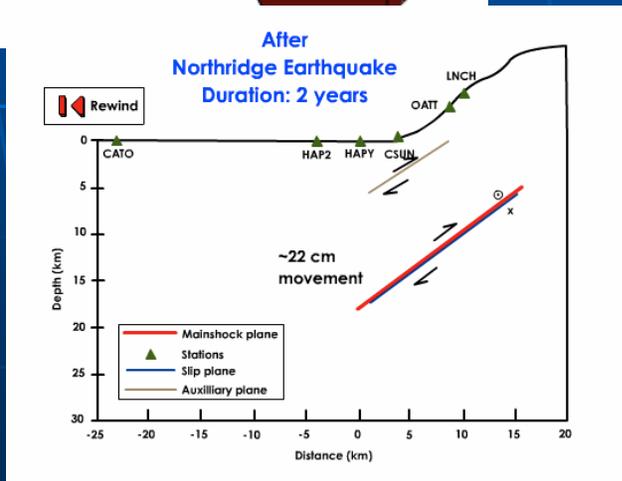
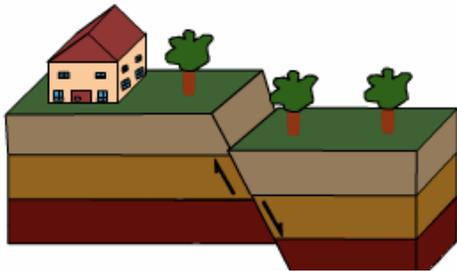
GeoFEST tutorial

- What is GeoFEST?
 - **Geo**physical **Finite Element Simulation Tool**
 - GeoFEST solves solid mechanics forward models with these characteristics:
 - 2-D or 3-D irregular domains
 - 1-D, 2-D or 3-D displacement fields
 - Static elastic or time-evolving viscoelastic problems
 - Driven by faults, boundary conditions or distributed loads
 - GeoFEST runs in a variety of computing environments:
 - UNIX workstations (including LINUX, Mac OS X, etc.)
 - Web portal environment
 - Parallel cluster/supercomputer environment

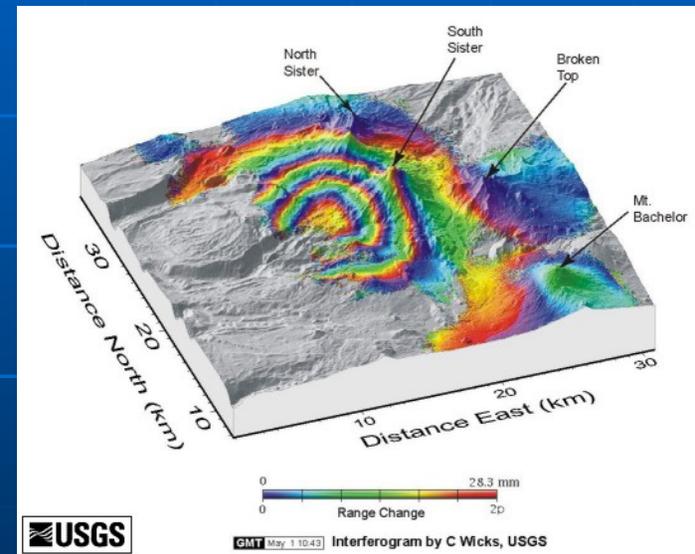
GeoFEST tutorial

- What are the applications of GeoFEST? (continued)

- Models of earthquake cycle



- Models of glacial and volcanic



GeoFEST tutorial

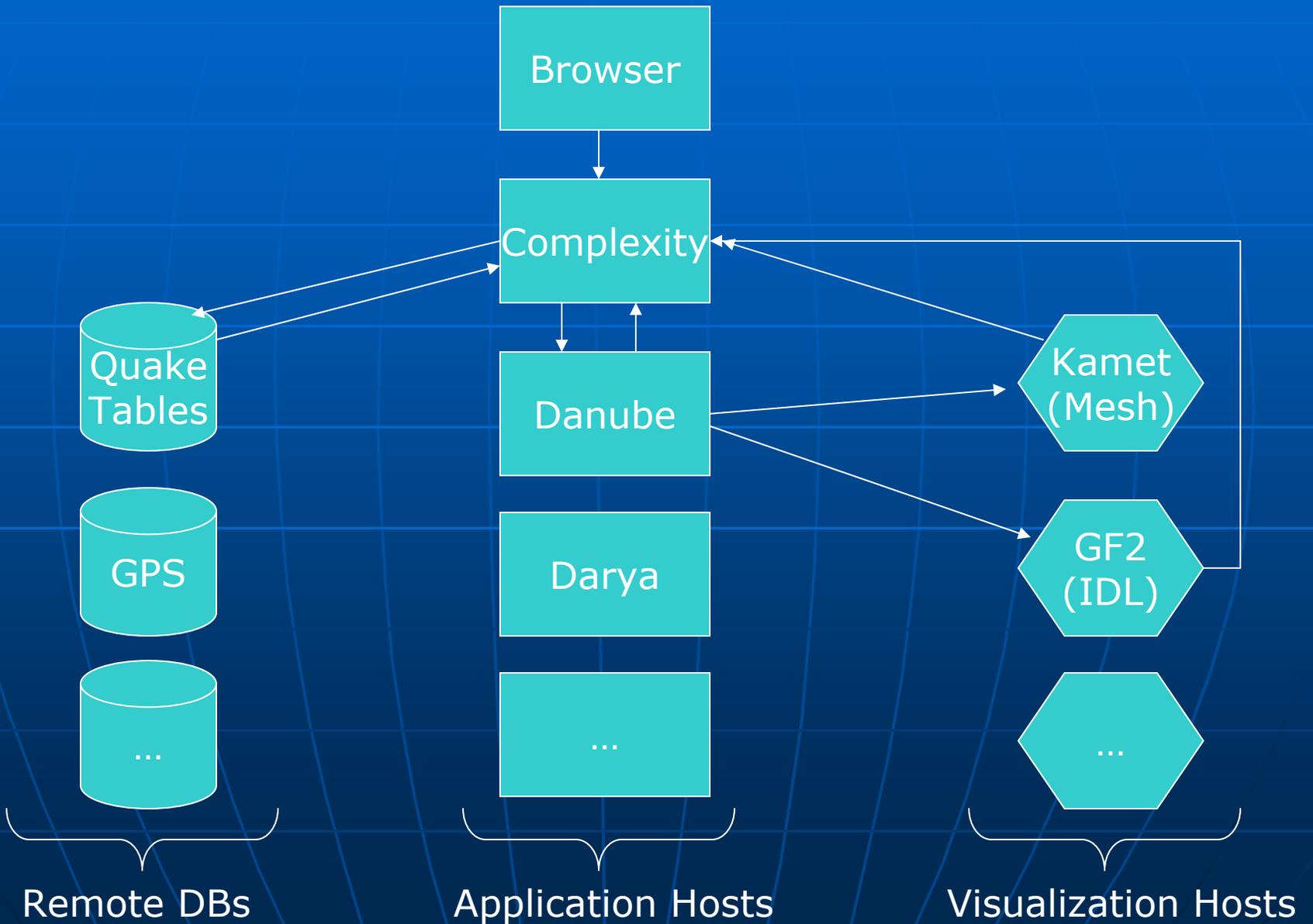
- What are the applications of GeoFEST?
(continued)
 - Calculation of irregular/heterogeneous Green's functions for use by other simulation or inversion software
 - Studies of frictional fault behavior



GeoFEST tutorial

- Using the web portal environment to create and run a typical 3-dimensional model
 - Use web portal to draft domain layers and boundaries
 - Using portal, add fault(s) to domain
 - Generate grid points and elements with desired refinement
 - Provide supplemental information on boundary conditions, material properties, time stepping, etc.
 - Submit run to GeoFEST for execution
 - Examine and visualize results

Under the Hood



GeoFEST tutorial

- Select the GeoFEST code in portal

SERVO Code Selection Disloc and GMT Danube Grids Fault Database CCE DB

SERVO Job Submit

Code Selection Menu

Please select a code and host machine from the following list of applications. When you have made your choice, click the "Make Selection" button at the bottom of the page.

- Disloc
- Simplex
- GeoFEST Plus Viz
- VirtualCalifornia
- MeshGenerator
- Geofit
- RDAMMM
- Slider
- PatternInformatics
- GeoFEST2**
- GeneticAlgorithm
- Karhunen Loeve
- GeoFEST ParVox
- GeoFEST Adaptive

Description: Three-dimensional viscoelastic finite element model for calculating nodal displacements and tractions. Allows for realistic fault geometry and characteristics, material properties, and body forces.

danube.ucs.indiana.edu

Make Selection

Cancel

Main Home

Select code

Select host

GeoFEST tutorial

- Create the desired geometry

Project Input
Create your geometry out of layers and faults.

Project Name: Model1sm

Create New Layer: Click to specify geometry for a layer.
 Create New Fault: Click to specify geometry for a fault segment.
 Add Layer from DB: Click to select a layer from the database.
 Add Fault from DB: Click to select a fault segment from the database.

Input Solid Layer Geometry

Layer Name:
Origin X:
Origin Y:
Origin Z:
Length:
Width:
Depth:
Lame Lambda:
Lame Mu:
Viscosity:
Exponent:

Current Project Components

Faults		
Name	View	Remove
SAFtop	<input type="radio"/>	<input type="radio"/>

Layers		
Name	View	Remove
elastic	<input type="radio"/>	<input type="radio"/>

Click the button below to plot Layers and Faults

Create Initial Mesh
Click the button below to generate a mesh for the geometry.

Mesh Size:
Mesh Refine Limit:

Create layer(s)

Create faults(s)

Enter dimensions and properties

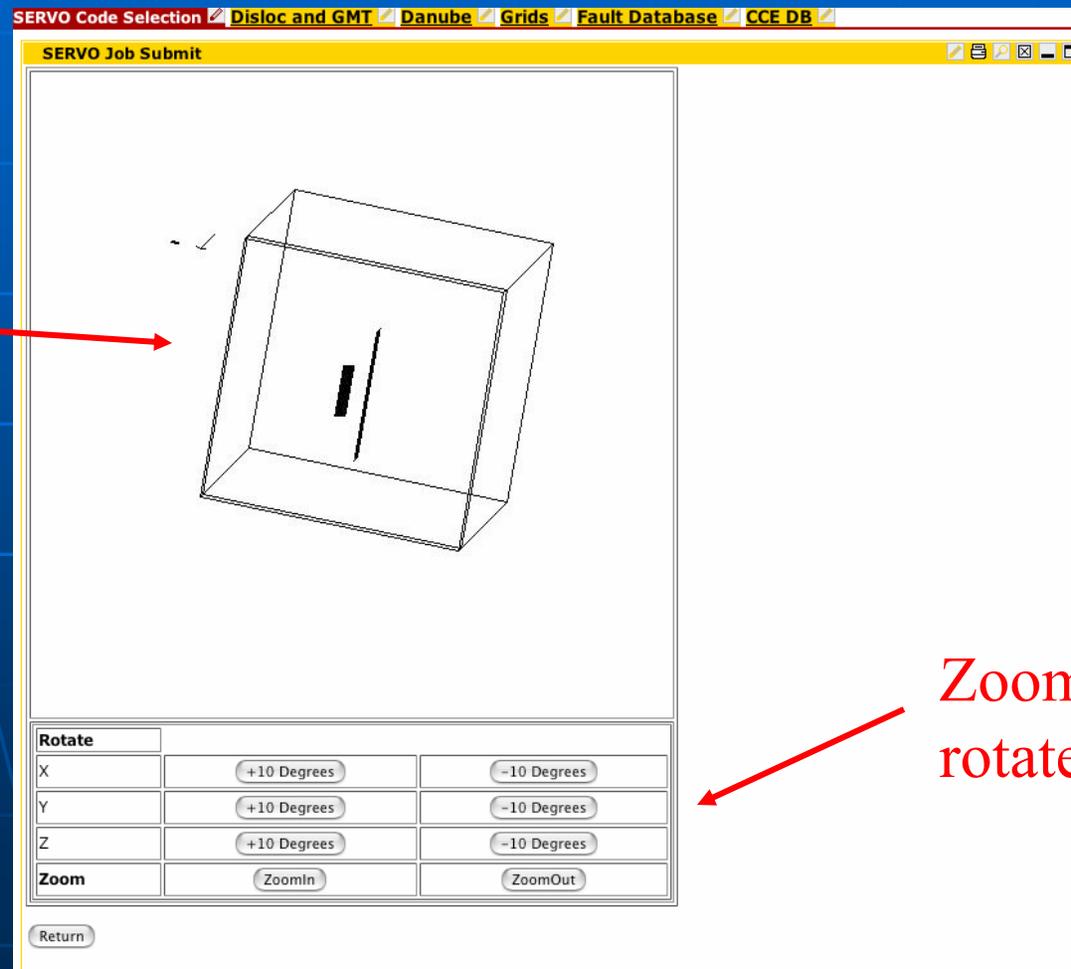
Create initial mesh

Plot results

GeoFEST tutorial

- Check the generated geometry

Pre-mesh
view of
layers and
faults



Zoom and
rotate view

GeoFEST tutorial

- After performing initial meshing of domain

SERVO Job Submit

Refine Mesh

Your initial mesh has been generated. You may now iteratively refine it by pressing the "Refine" button.

Mesh Refine Limit:

```
QDIST[8] = 263
QDIST[9] = 149
TOTAL POINTS 350 TETRAHEDRA 1407
Refine Wavelength Points 0
Refine Wavelength Points 0
The Worst Q 3.502845e-02 AVGQ 6.865582e-01
The Worst Q 3.468963e-02 AVGQ 6.863951e-01
The Worst Q 3.468963e-02 AVGQ 6.863951e-01
QDIST[0] = 2
QDIST[1] = 5
QDIST[2] = 38
QDIST[3] = 55
QDIST[4] = 132
QDIST[5] = 142
QDIST[6] = 283
QDIST[7] = 340
QDIST[8] = 260
QDIST[9] = 149
TOTAL POINTS 350 TETRAHEDRA 1406
```

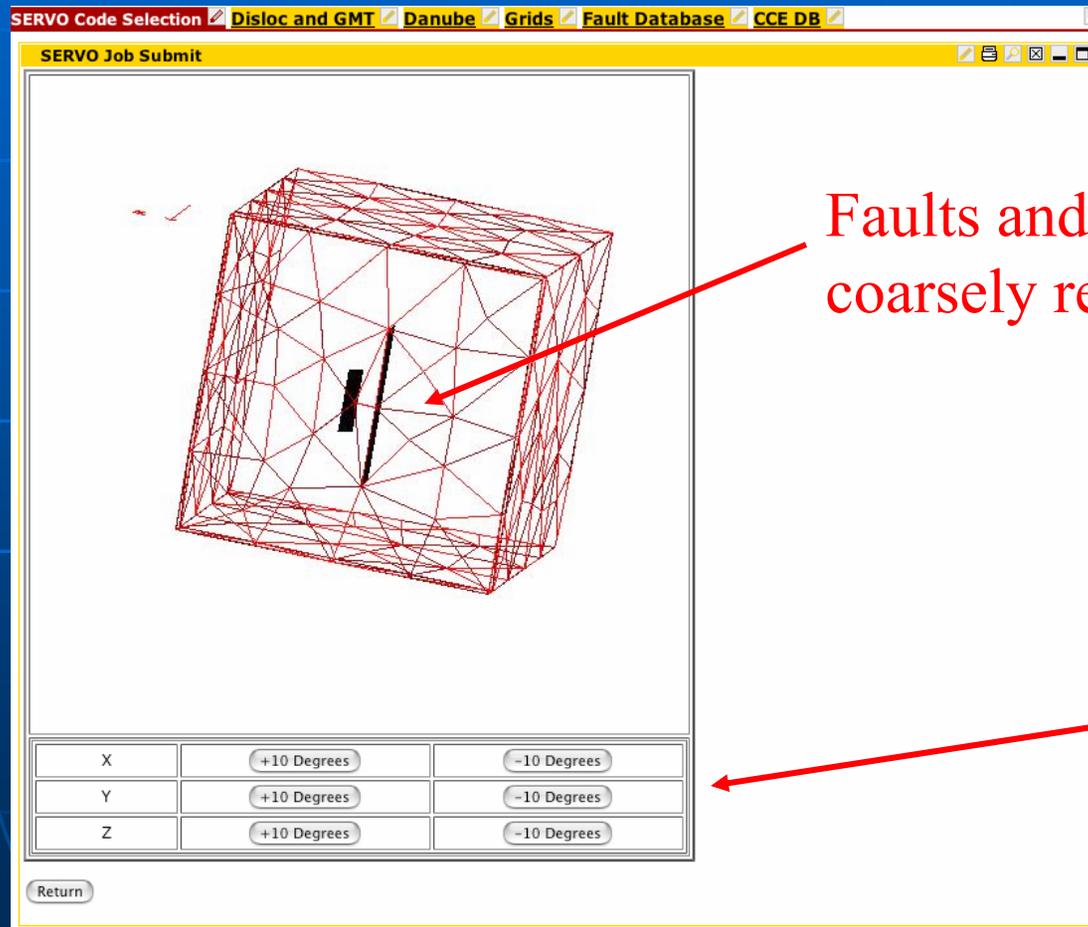
Click "Refine Mesh" to launch the Mesh Refiner. The Mesh Refiner may take several minutes to complete.
Click "View Messages" to view the Mesh Refiner's output messages.

Status of meshing

Look at resulting mesh

GeoFEST tutorial

- Viewing initial meshing of domain



Faults and volumes
coarsely resolved

Rotate
views

GeoFEST tutorial

- Requesting refined meshing of domain

SERVO Code Selection **Disloc and GMT** **Danube** **Grids** **Fault Database** **CCE DB**

SERVO Job Submit

Refine Mesh

Your initial mesh has been generated. You may now iteratively refine it by pressing the "Refine" button.

Mesh Refine Limit:

```
tagfault.pl Model1sm.node Model1sm.tetra SAFbottom.flt 1 Model1sm.index
number dip(o) strike(o) slip(m) rake(o) length(km)
width(km) depth(km)
1.0 52 0 5 90

opening Model1sm.node
Reading 621 nodes.
opening Model1sm.tetra
Reading 2817 tets.
Number of elts with substantial priority: 2118
APOLLO Model1sm 1.0
Refine Points 207
APOLLO Model1sm 1.0
Mesh Refine 0 / 207 Tetra 2817 Tri 5851
Mesh Refine 100 / 207 Tetra 3331 Tri 6879
Mesh Refine 200 / 207 Tetra 3853 Tri 7924
The Worst Q 1.911663e-02 AVGQ 5.886470e-01
The Worst Q 1.635721e-01 AVGQ 6.201361e-01
The Worst Q 1.576477e-01 AVGQ 6.208042e-01
The Worst Q 1.576477e-01 AVGQ 6.208042e-01
```

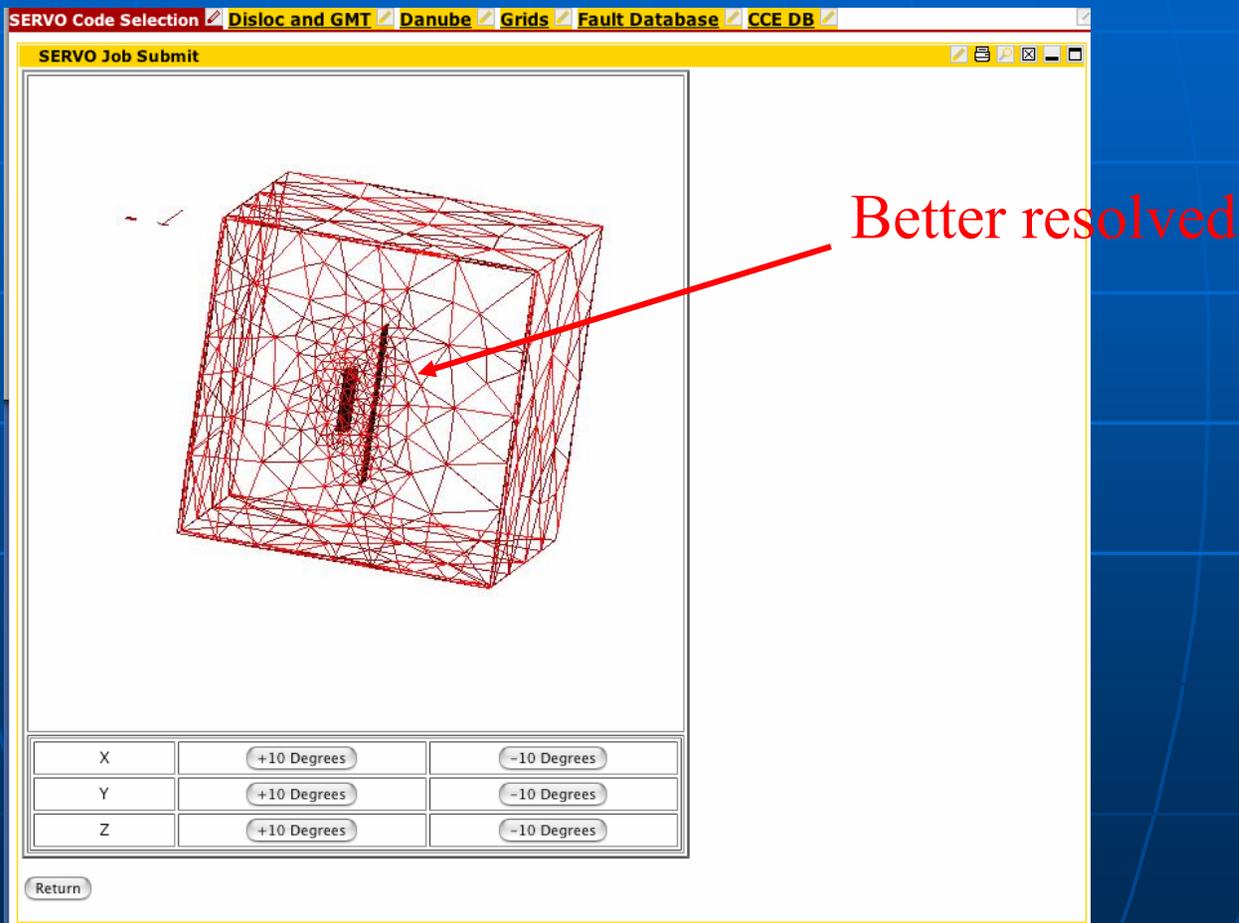
Click "Refine Mesh" to launch the Mesh Refiner. The Mesh Refiner may take several minutes to complete.
Click "View Messages" to view the Mesh Refiner's output messages.

Status of refinement
progress

Interface
controls

GeoFEST tutorial

- Viewing refined meshing of domain



GeoFEST tutorial

- Running prepared GeoFEST model

SERVO Code Selection | **Disloc and GMT** | **Danube** | **Grids** | **Fault Database** | **CCE DB**

SERVO Job Submit

Input and Output File Names

Input File Name:
Output File Name:
Log File Name:
Email Address:

Input Parameters

number_space_dimensions	<input type="text" value="3"/>
number_degrees_freedom	<input type="text" value="3"/>
nrates	<input type="text" value="0"/>
shape_flag	<input type="text" value="1"/>
solver_flag	<input type="text" value="2"/>
number_time_groups	<input type="text" value="1"/>
reform_steps	<input type="text" value="1"/>
backup_steps	<input type="text" value="5000"/>
fault_interval	<input type="text" value="3000.0"/>
end_time	<input type="text" value="1.0"/>
alpha	<input type="text" value="1.0"/>
time_step	<input type="text" value="0.5"/>

Boundary Conditions

top_bc	<input type="text" value="Free Node"/> BC Values: <input type="text" value="0 0. 0. 0. 1."/>
east_bc	<input type="text" value="Locked Node"/> BC Values: <input type="text" value="0 0. 0. 0. 1."/>
west_bc	<input type="text" value="Locked Node"/> BC Values: <input type="text" value="0 0. 0. 0. 1."/>
north_bc	<input type="text" value="Locked Node"/> BC Values: <input type="text" value="0 0. 0. 0. 1."/>
south_bc	<input type="text" value="Locked Node"/> BC Values: <input type="text" value="0 0. 0. 0. 1."/>
bottom_bc	<input type="text" value="Locked Node"/> BC Values: <input type="text" value="0 0. 0. 0. 1."/>

Output Parameters and Formatting

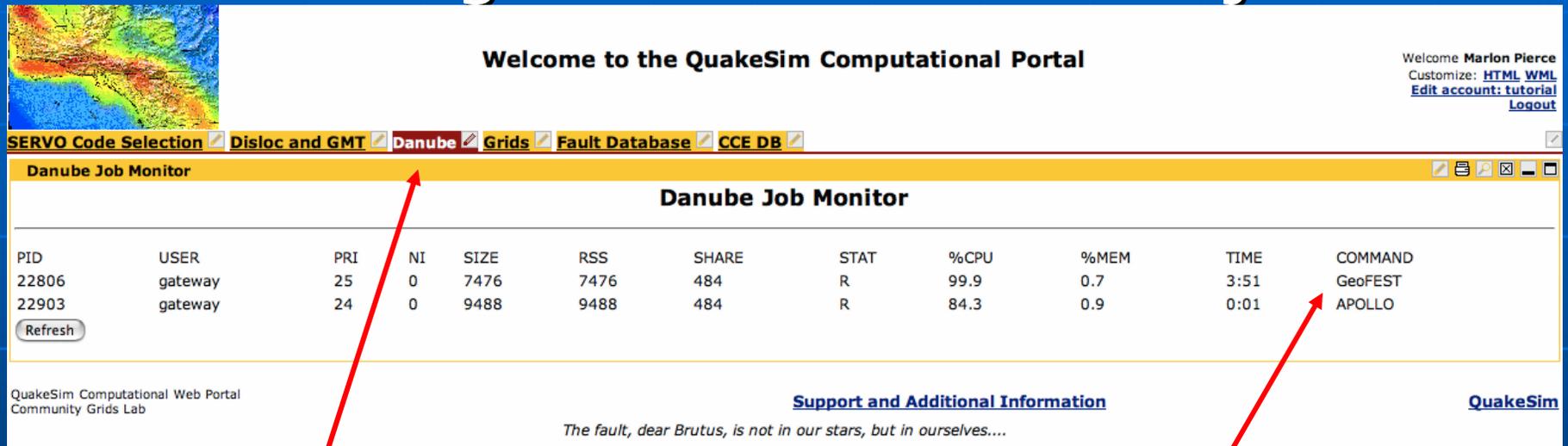
Reporting Nodes:
Reporting Elements:
Print Times Type:
Number of Print Times:
Print Times Interval:
Restart File:
Checkpoint File:

Enter additional run parameters and boundary conditions

Run GeoFEST

GeoFEST tutorial

■ Monitoring status of GeoFEST job



Welcome to the QuakeSim Computational Portal

Welcome Marlon Pierce
Customize: [HTML](#) [WML](#)
[Edit account: tutorial](#)
[Logout](#)

[SERVO Code Selection](#) [Disloc and GMT](#) [Danube](#) [Grids](#) [Fault Database](#) [CCE DB](#)

Danube Job Monitor

PID	USER	PRI	NI	SIZE	RSS	SHARE	STAT	%CPU	%MEM	TIME	COMMAND
22806	gateway	25	0	7476	7476	484	R	99.9	0.7	3:51	GeoFEST
22903	gateway	24	0	9488	9488	484	R	84.3	0.9	0:01	APOLLO

[Refresh](#)

QuakeSim Computational Web Portal
Community Grids Lab

[Support and Additional Information](#)

QuakeSim

The fault, dear Brutus, is not in our stars, but in ourselves....

Job Monitor portal tab

Process running
GeoFEST

GeoFEST tutorial

- Accessing completed GeoFEST

SERVO Code Selection Disloc and GMT Danube Grids Fault Database CCE DB

SERVO Job Submit

Archived Data

You have the following archived data files. Click the link to download the file.

Project Name	Storage Host	Creation Date	Data File
Model1sm	danube.ucs.indiana.edu	Wed Jun 23 17:46:20 EST 2004	Model1sm.inp Model1sm.out Model1sm.log

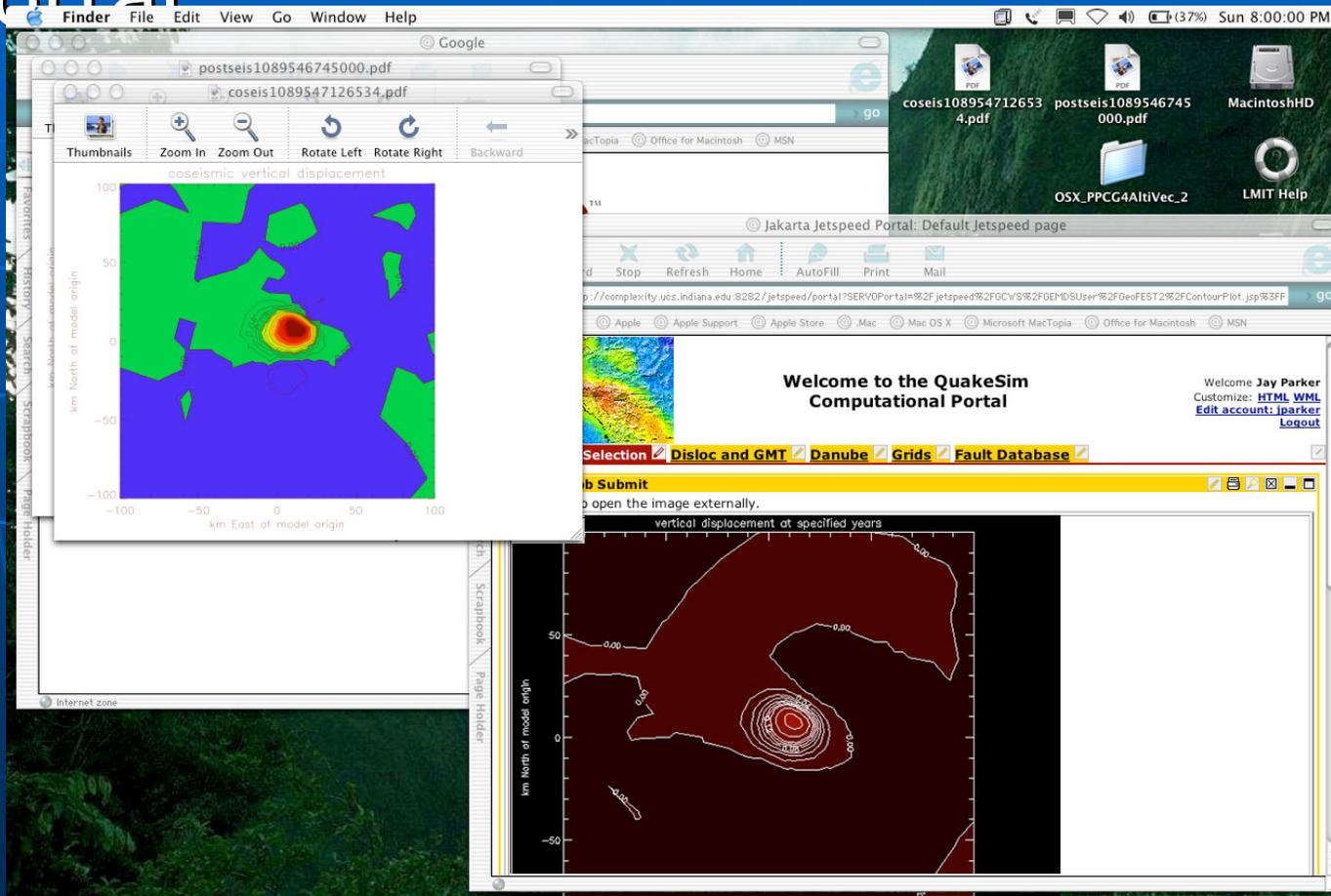
Main Home

ASCII input file

ASCII output file

GeoFEST tutorial

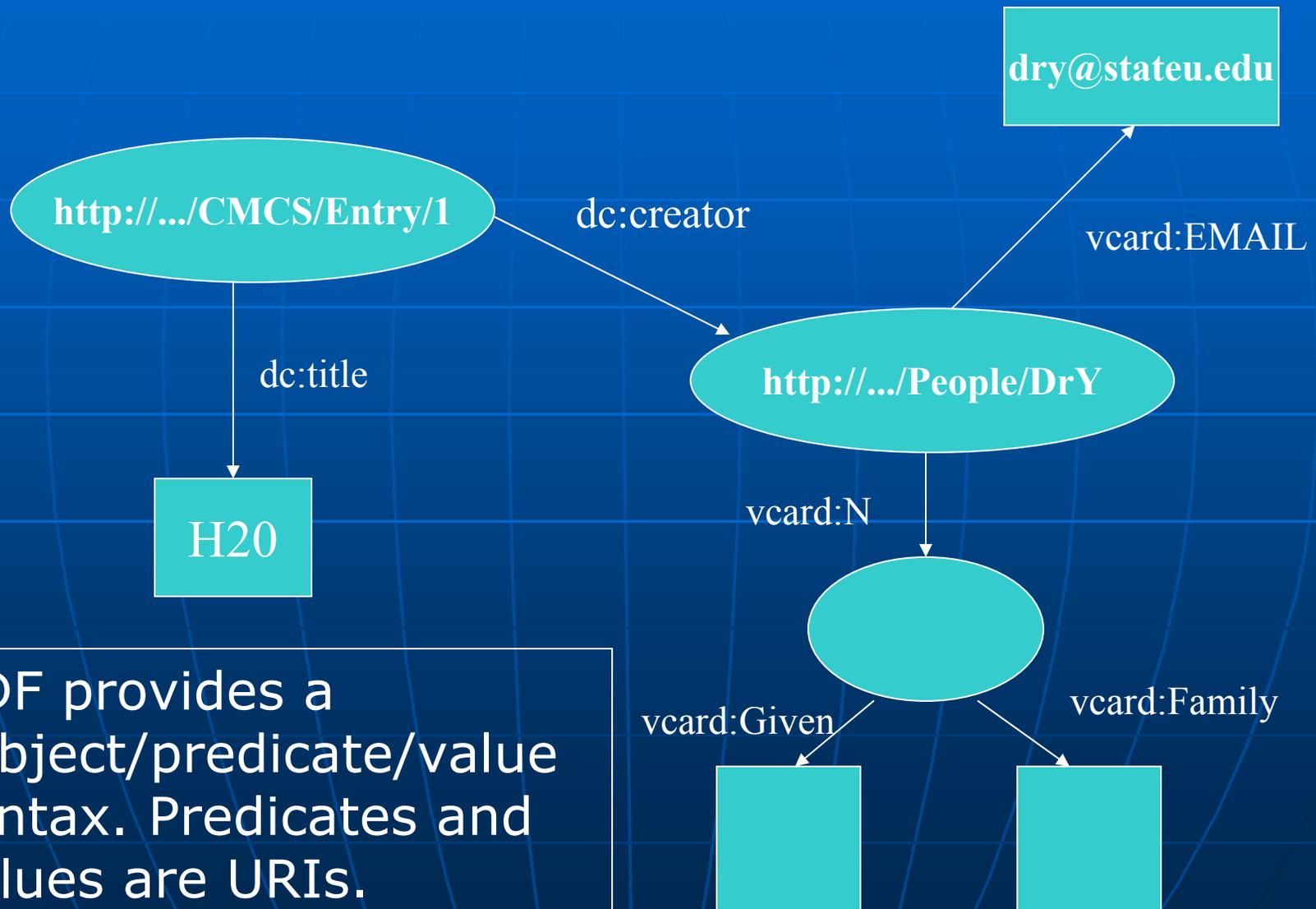
- Plotting, visualization of results via portal



Making SERVO Semantic

Application of Semantic Web
tools and concepts to
SERVOGrid

Semantic Web in One Slide



RDF provides a subject/predicate/value syntax. Predicates and values are URIs.

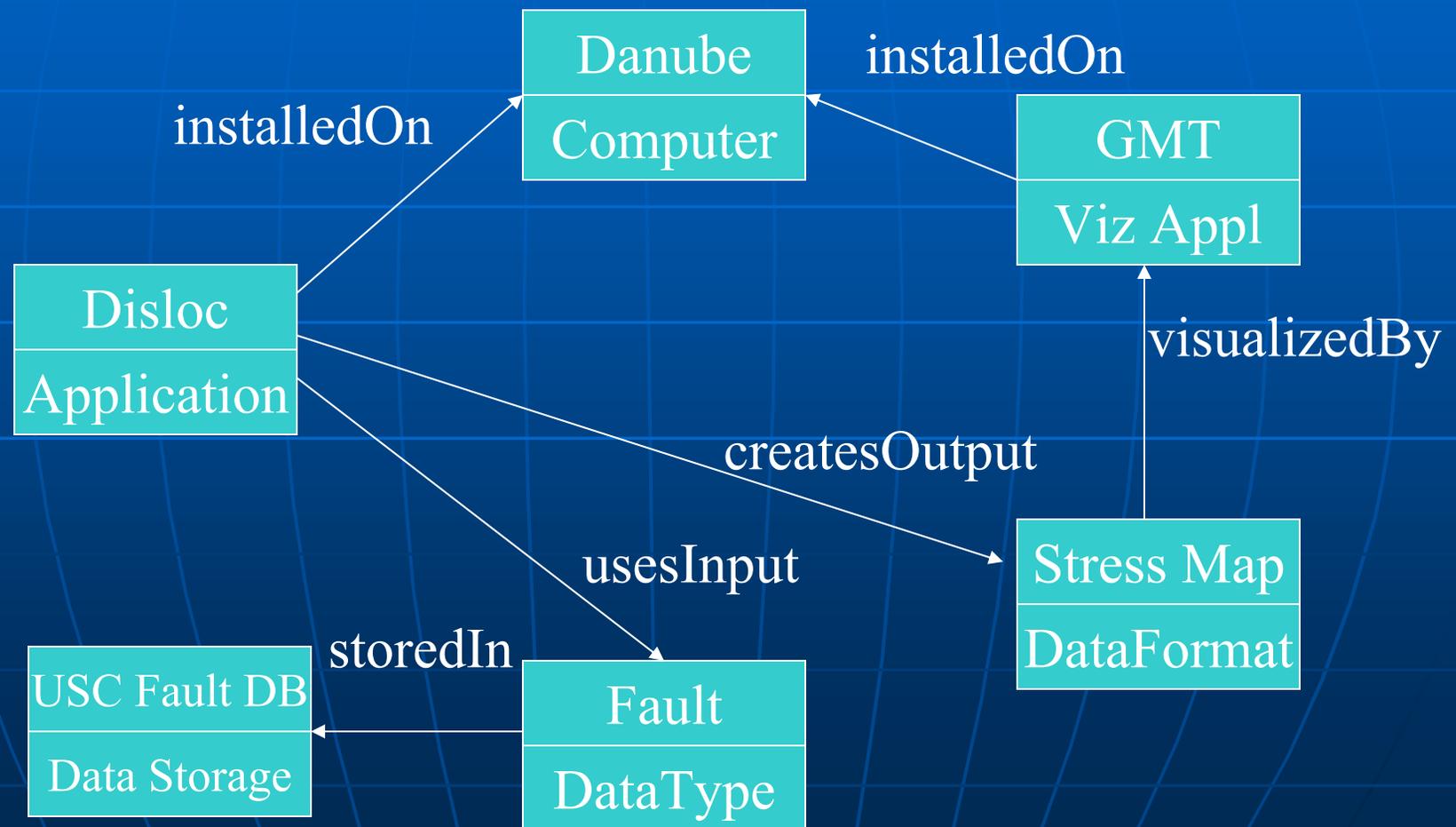
Semantic Needs for SERVOGrid

- SERVOGrid has many types of metadata
- Computing resources
 - Applications
 - Data
 - Services
- I have designed XML schemas and built services for this sort of metadata before, but they were too monolithic.
 - RDF has an interesting way of expressing linkages between different RDF fragments.
 - If we can exploit this, it will make for much more flexible metadata services.

Assembling a SERVOnGrid Ontology

- We are designing RDFS descriptions for the following components:
 - Simulation codes, mesh generators, etc.
 - Visualization tools
 - Data types
 - Computing resources
 - ...
- These are easily expressed as RDFS (actually DAML) "nuggets" of information.
 - Create instances of these
 - Use properties to link instances.

Some Sample Relationships



Making It Work

- One of the problems we encountered with processing RDF metadata is that tools assume all data is local.
- What we really have though are metadata fragments scattered throughout SERVOnGrid.
- Need ways of processing RDF triplets when predicate values are not local.

More Information

- SERVOnGrid/QuakeSim:
 - <http://quakesim.jpl.nasa.gov/>
- Full Portal Demo:
 - <http://complexity.ucs.indiana.edu:8080>
 - Request an account
 - Downloads available in November
- USC Fault database
 - <http://infogroup.usc.edu:8080>
- GPS and Seismic Database Demo:
 - <http://gf3.ucs.indiana.edu:6060/cce/sql/>
- Setting up your own GPS or Seismic database
 - <http://complexity.ucs.indiana.edu/~gaydin/cce/install/install.html>
- Publications:
 - <http://grids.ucs.indiana.edu/ptliupages/publications/>