George E. Brown, Jr.
Network for Earthquake Engineering Simulation

Workflows in the Cloud Using OpenSees and NEEShub

Frank McKenna

NEES

Pacific Earthquake Engineering Research Center

work·flow

/ˈwərkˌflō/

noun

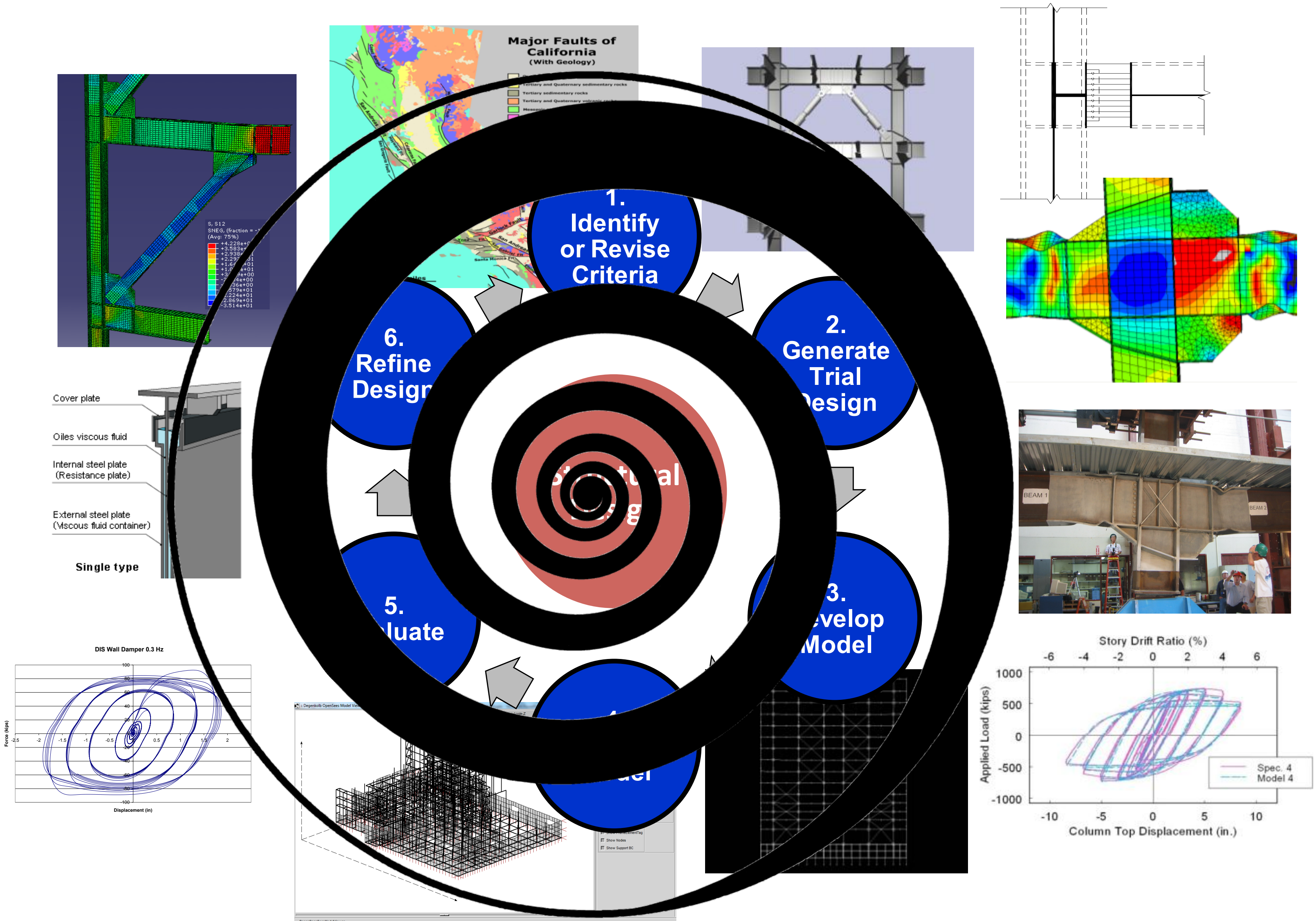
noun: **workflow**; plural noun: **workflows**

1. the sequence of industrial, administrative, or other processes through which a piece of work passes from initiation to completion.

Translate workflow to Choose language

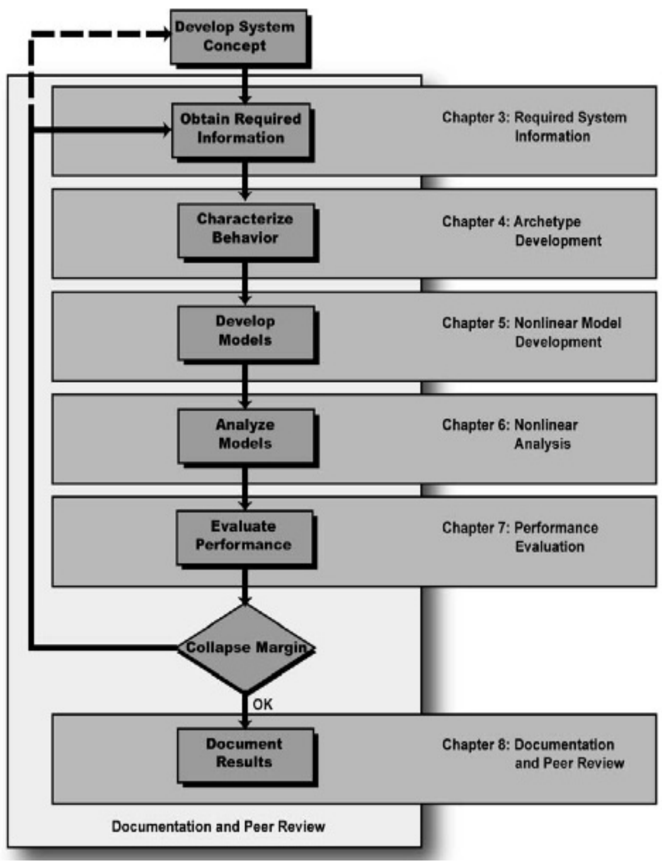
Use over time for: workflow

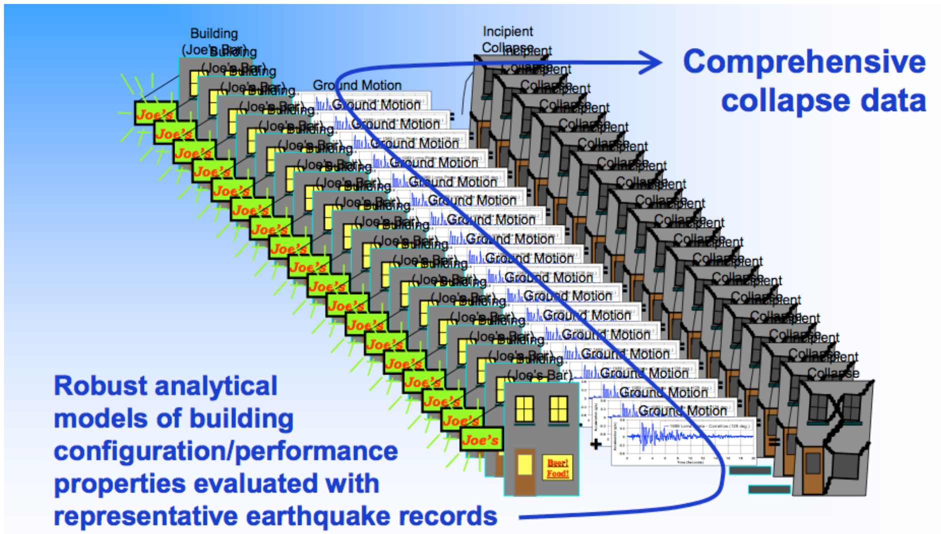
Example from Practice: The Design Process!

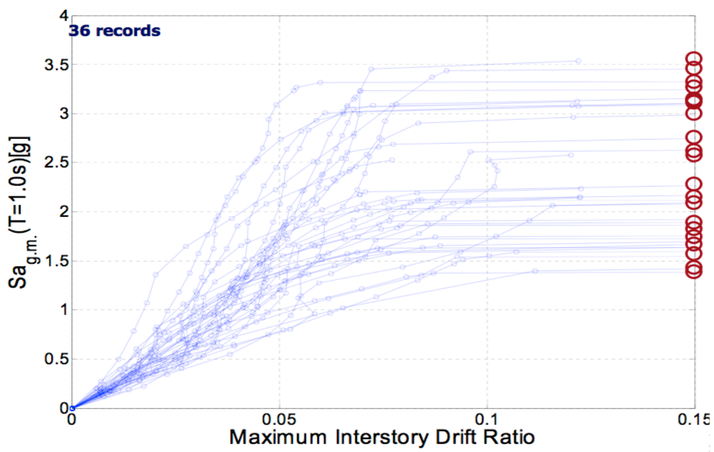


Acknowledgements: Silvia Mazzoni

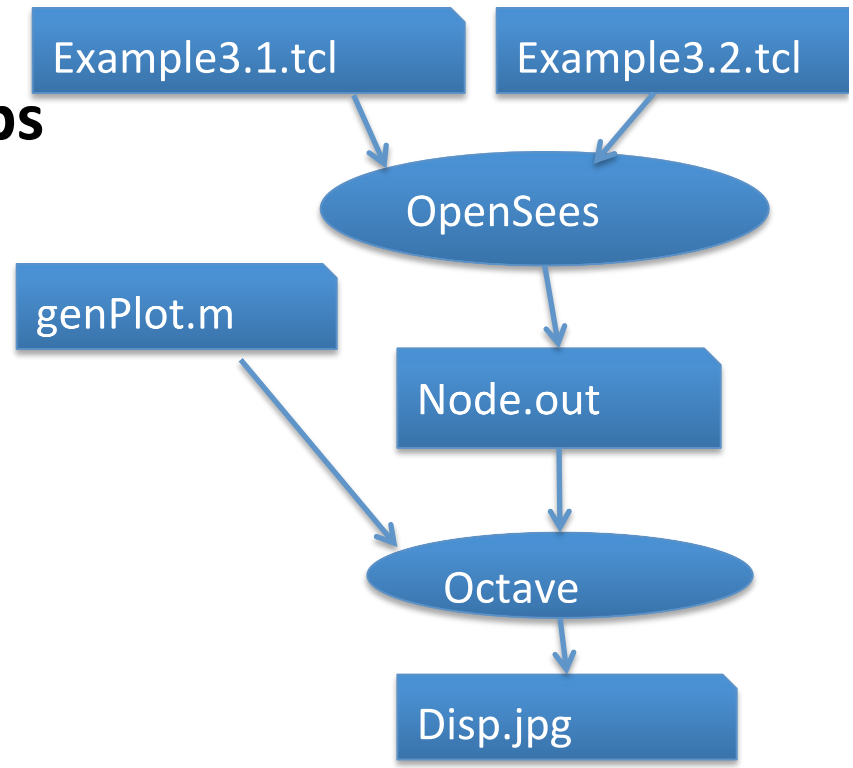
Example from Research: Using FEMA P-695 for new systems







- As engineers we often need to:
  - **Repeat Processing Steps on New Data**
  - **Automate Data Processing Steps**
  - **Reproduce Previous Results**
  - Share our analysis steps with other researchers
  - Reliably execute analyses on unreliable infrastructure
  - **Execute Analyses in Parallel on Distributed resources**

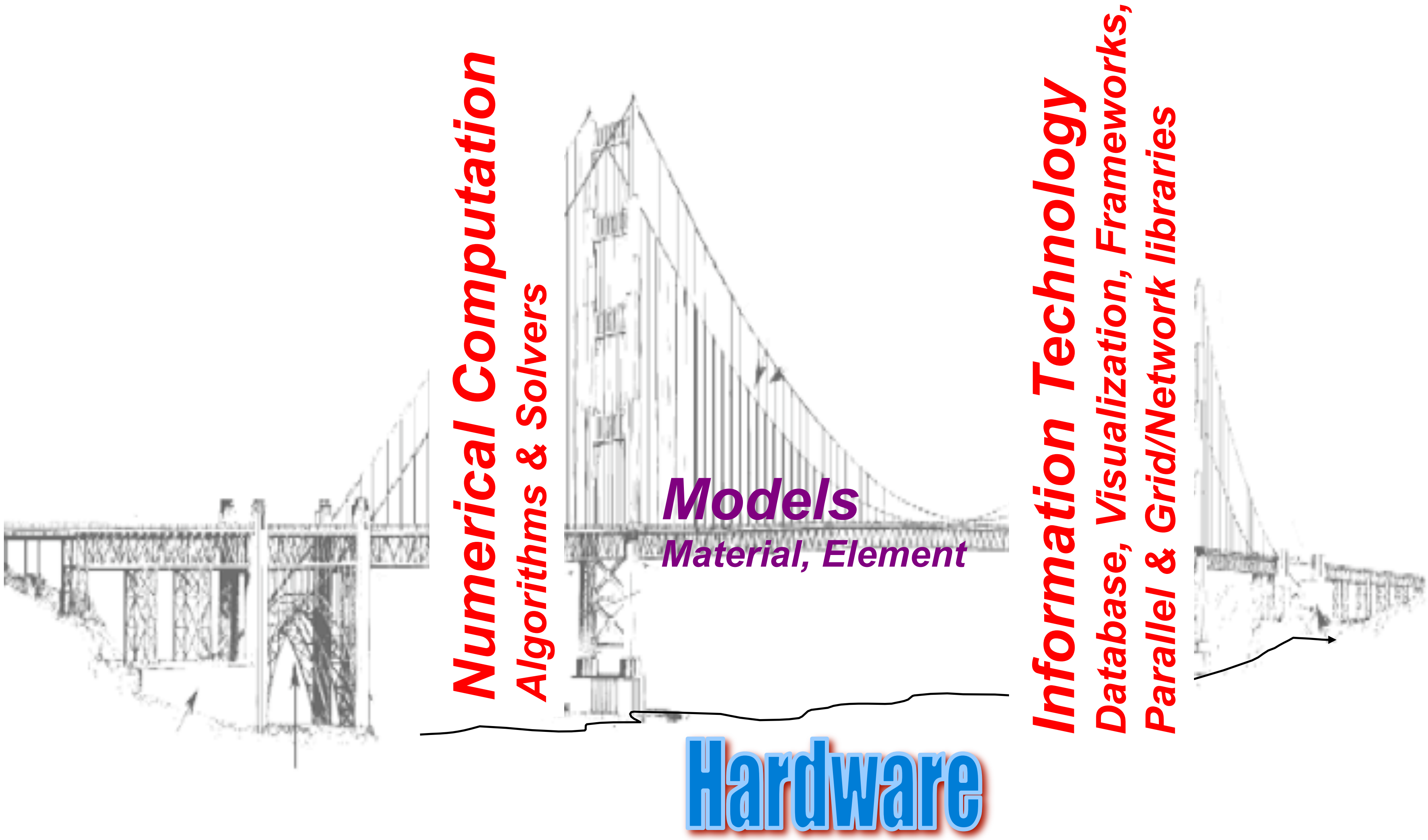


This you may not know you need!

So What!

Technology is Changing

Building Blocks for any FE Application

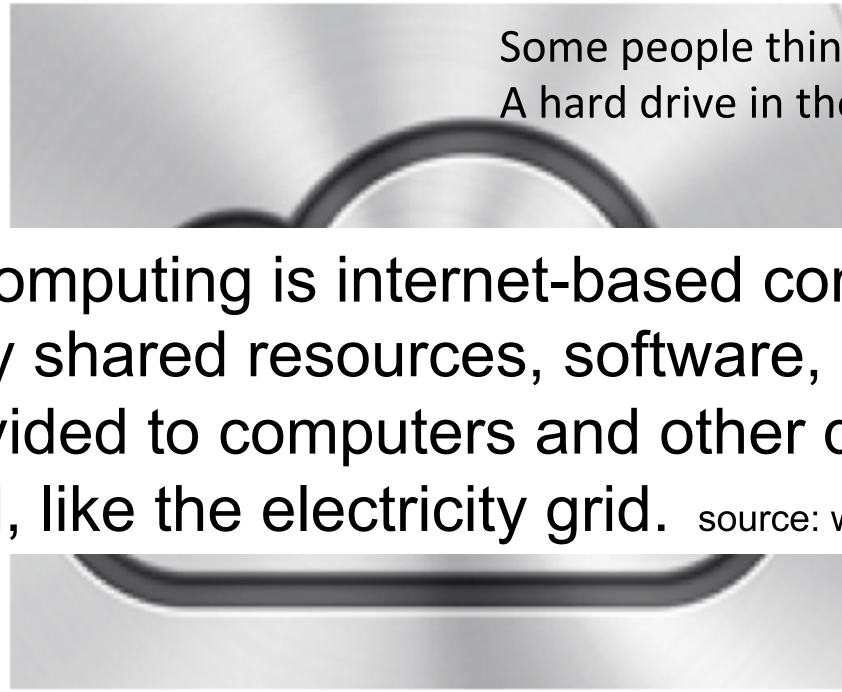


Bell's Law

Bell's Law of Computer Class formation was discovered about 1972. It states that technology advances in semiconductors, storage, user interface and networking advance every decade enable a new, usually lower priced computing platform to form. Once formed, each class is maintained as a quite independent industry structure. This explains mainframes, minicomputers, workstations and Personal computers, the web, emerging web services, palm and mobile devices, and ubiquitous interconnected networks.

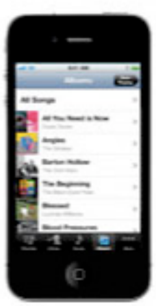
Gordon Bell, http://research.microsoft.com/~GBell/Pubs.htm

STEVE JOBS KEYNOTE WWDC 2011



Some people think the cloud is just
A hard drive in the sky!

Cloud computing is internet-based computing , whereby shared resources, software, and information are provided to computers and other devices on demand, like the electricity grid. source: wikipedia



“..PC and Mac Demoted to a Device”

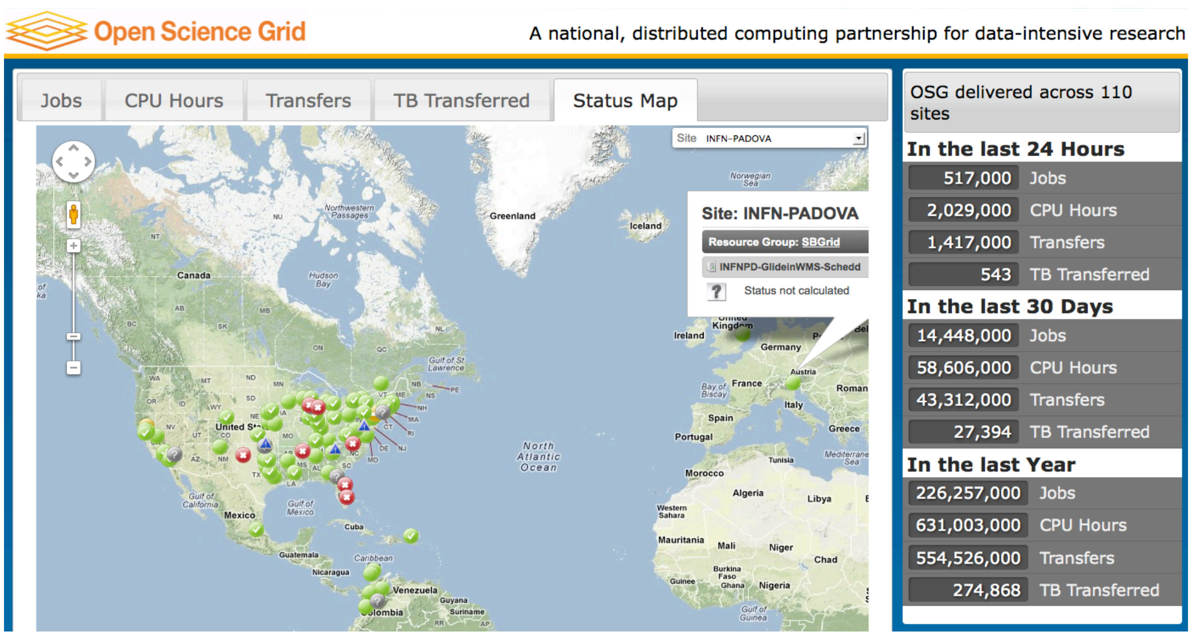
Cloud Computing **Compute Resources**

- Commercial
- Research
  - All of the above **PLUS**
  - Dedicated High Performance Computers
  - Distributed Computers (Grid)

OpenScience Grid

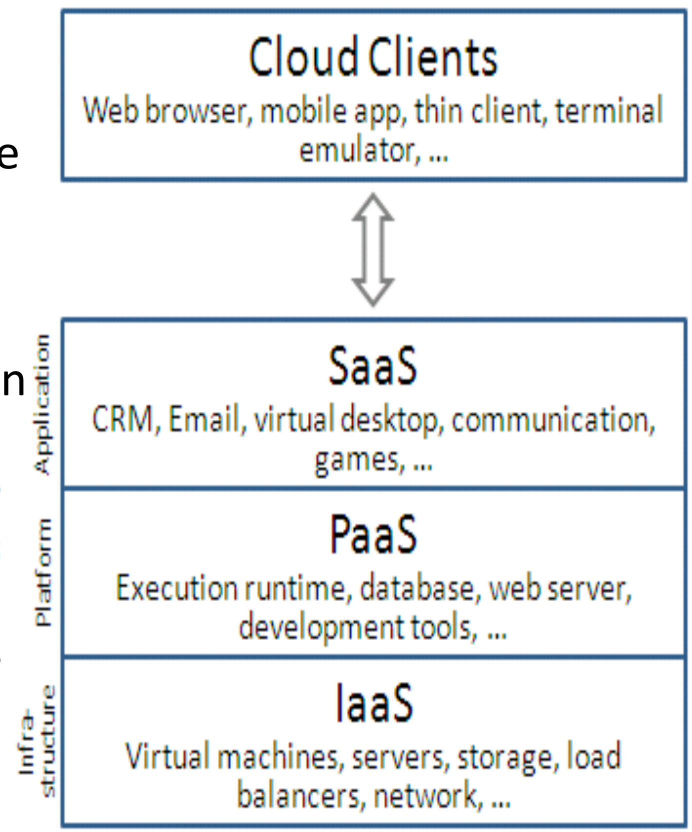
Open Science Grid

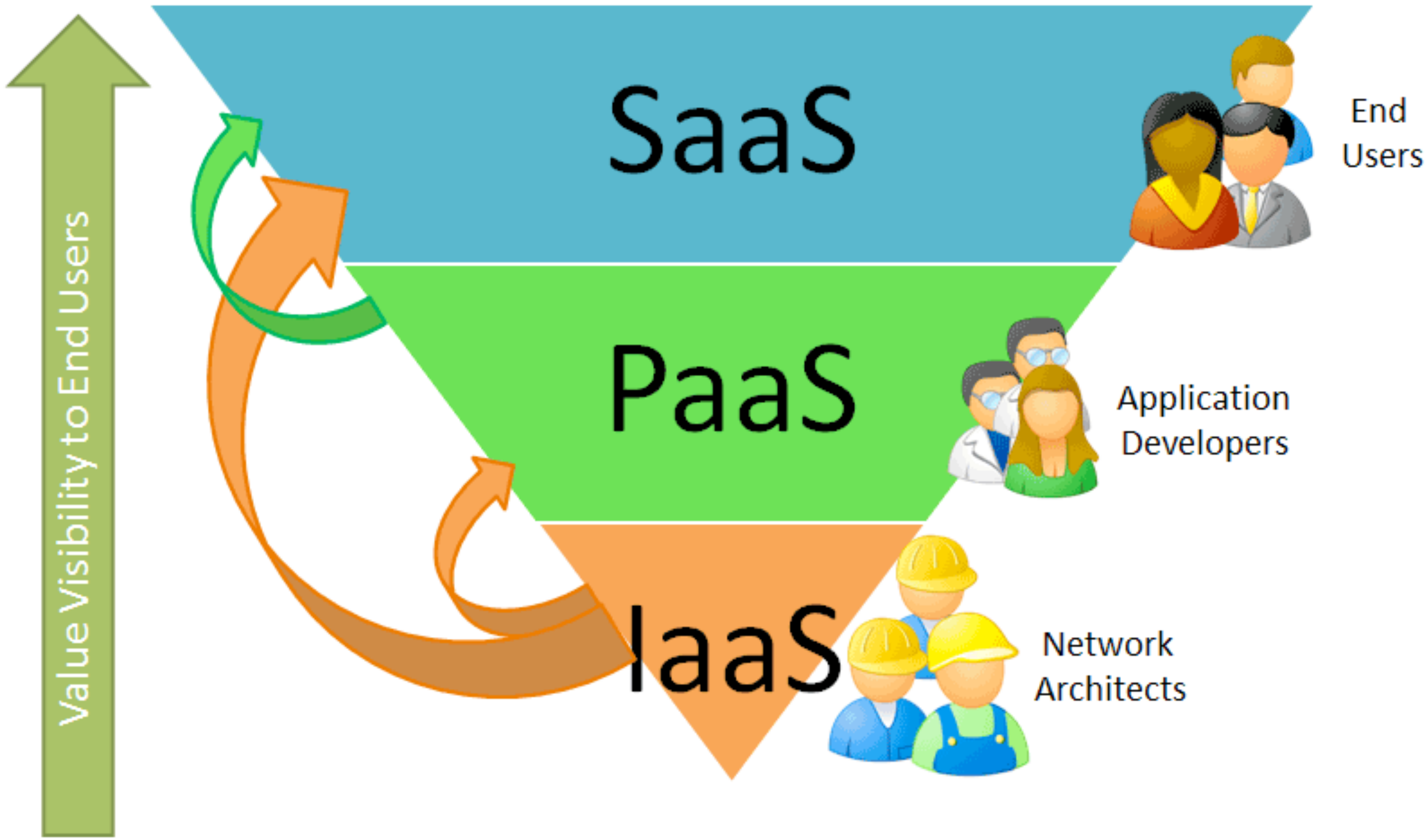
- OSG
  - 30 Virtual Organizations
  - 80 sites
  - +1 million CPU hours



How Do We Use These Resources Cloud Service Models

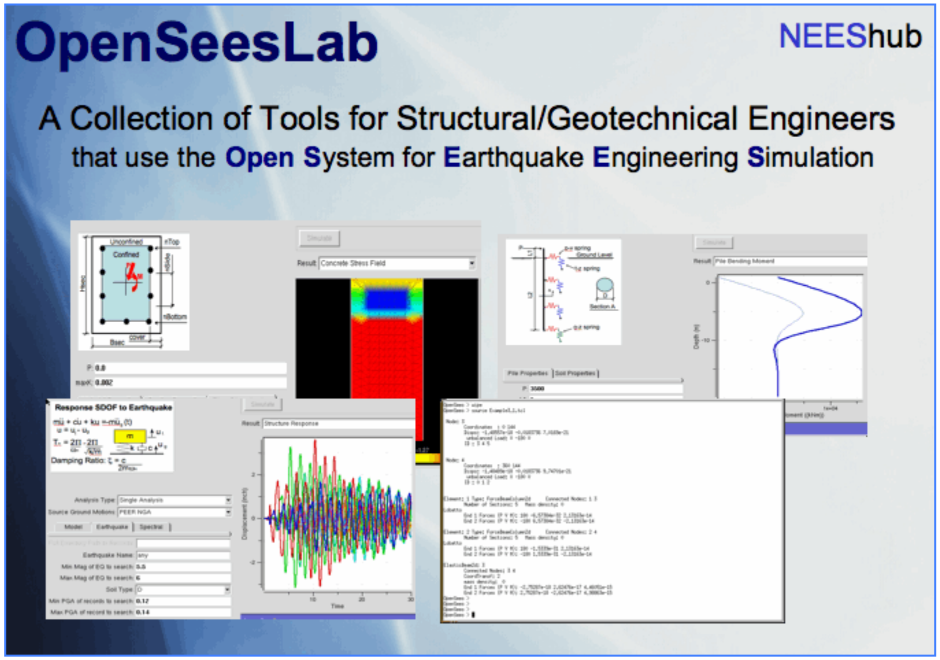
- **IaaS: Infrastructure as a Service**
  - In the most basic cloud-service model, provides companies with computing resources including servers, networking, storage, and data center space on a pay-per-use basis. (You Rent the Hardware)
- **PaaS: Platform as a Service**
  - Provides a computing platform, typically including operating system, programming language execution environment, database, and web server (You Rent the hardware, OS, basic software: databases, compilers, web server)
- **SaaS: Software as a Service**
  - Cloud-based applications—or software as a service (SaaS)—run on distant computers "in the cloud" that are owned and operated by others and that connect to users' computers via the Internet and, usually, a web browser. (




Value Visibility to End Users
SaaS
PaaS
IaaS
End Users
Application Developers
Network Architects

Example SaaS: The OpenSeesLab tool:

http://nees.org/resources/tools/openseeslab



Is a suite of Simulation Tools powered by OpnSees for:

1. Submitting OpenSees scripts (input files) to HUB resources
2. Educating students and practicing engineers
3. Providing Examples to Developers

OpenSees Interpreter Tool

hub — Java Applet Window

Application: OpenSees Interpreter

```
        OpenSees -- Open System For Earthquake Engineering Simulation
       Pacific Earthquake Engineering Research Center -- 2.2.1

           (c) Copyright 1999,2000 The Regents of the University of California
                                 All Rights Reserved
    (Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees/copyright.html)


OpenSees > tar xBf A_Example.tar
OpenSees > cd A_Example
OpenSees > source Ex8.tcl
couldn't read file "Ex8.tcl": no such file or directory
OpenSees > ls
A8.tcl
ExampleSP1.tcl
Node.out
analysis.tcl
model.tcl
peerRecords.txt
OpenSees > source A8.tcl
WARNING analysis Transient dt tFinal - no LinearSOE specified,
 ProfileSPDLinSOE default will be used

 Node: 525
        Coordinates  : 1 1 10
        Disps: 0.00977277 0.00977277 -0.00409793
        Velocities   : 0.0141832 0.0141832 -0.00878414
        commitAccels: 0.128284 0.128284 0.228457
         unbalanced Load: 0 0 0
        ID : 0 1 2

Simulation Time 192
OpenSees > cd ..
OpenSees > tar cBf A_Example.tar A_Example
OpenSees >
```

Cute
BUT
So What!

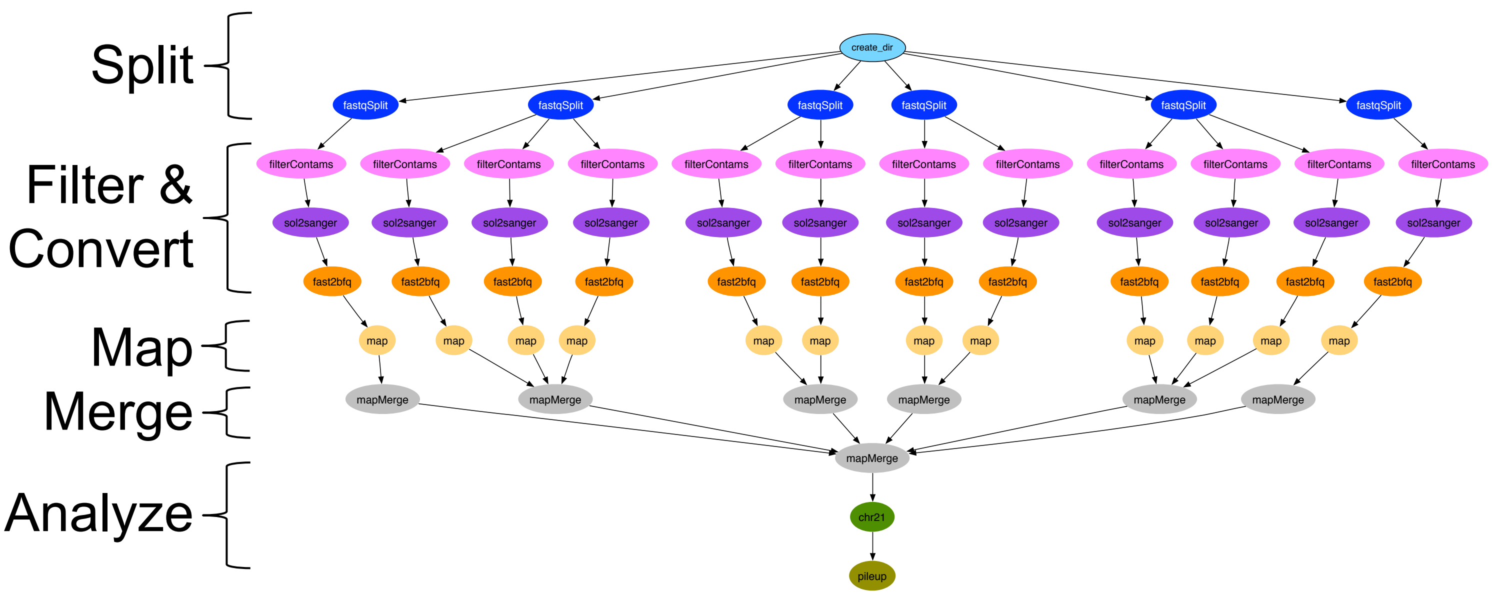
- Actually if Software Providers (Dassault Systemes, ANSYS, CSI, ..) provided their applications in a SaaS Model, engineering firms could adjust better to deadlines, market up's and down's than they can now!

Technology is Changing

Scientific Workflows will make this future possible

Software exists for creating such workflows that can take advantage of computational resources in the cloud! **Scientific workflow** allows engineers to compose and execute a series of computational or data manipulation steps in a scientific application.

- Orchestrate complex, multi-stage scientific computations
- Expressed in high-level workflow languages
  - DAGs, scripting languages, data flow, actors, etc.
- Can be optimized, and automatically parallelized for execution on distributed resources



Acknowledgements:Ewa Deelman, Mats Rynge, Karan Vahi , USC Information Sciences Institute

Pegasus

Workflow Management System

http://pegasus.isi.edu

- **Pegasus is a workflow planner ("compiler") which can handle everything from single-task workflows to workflows with millions of tasks**
    - Workflows enables parallel computation
    - Pegasus workflows are described in a higher level portable and reusable format
    - Enables execution on standard compute infrastructures (clouds, grids, campus clusters, ...)
- **Pegasus automatically restructures the workflow to improve performance and data management**
    - Task clustering - combining short tasks into longer jobs
    - Workflow reduction / data reuse - workflows are minimized based on existing data
    - Data cleanup - Pegasus maintains a minimal storage footprint during execution
- **Pegasus automatically plans and optimizes data placement**
- **Pegasus is not just for large-scale workflows. Other reasons for using Pegasus:**
    - Well defined failure recovery - automatic retries in case of failure to increase the overall reliability
    - Monitoring - provenance data
    - Debugging - tools to pinpoint failures

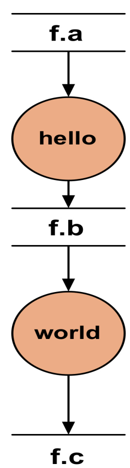
Acknowledgements:Ewa Deelman, Mats Rynge, Karan Vahi , USC Information Sciences Institute

Generating executable workflows with Pegasus

- Create a DAX
- A DAX contains a description of the abstract workflow in XML format.
- It is the primary input to Pegasus

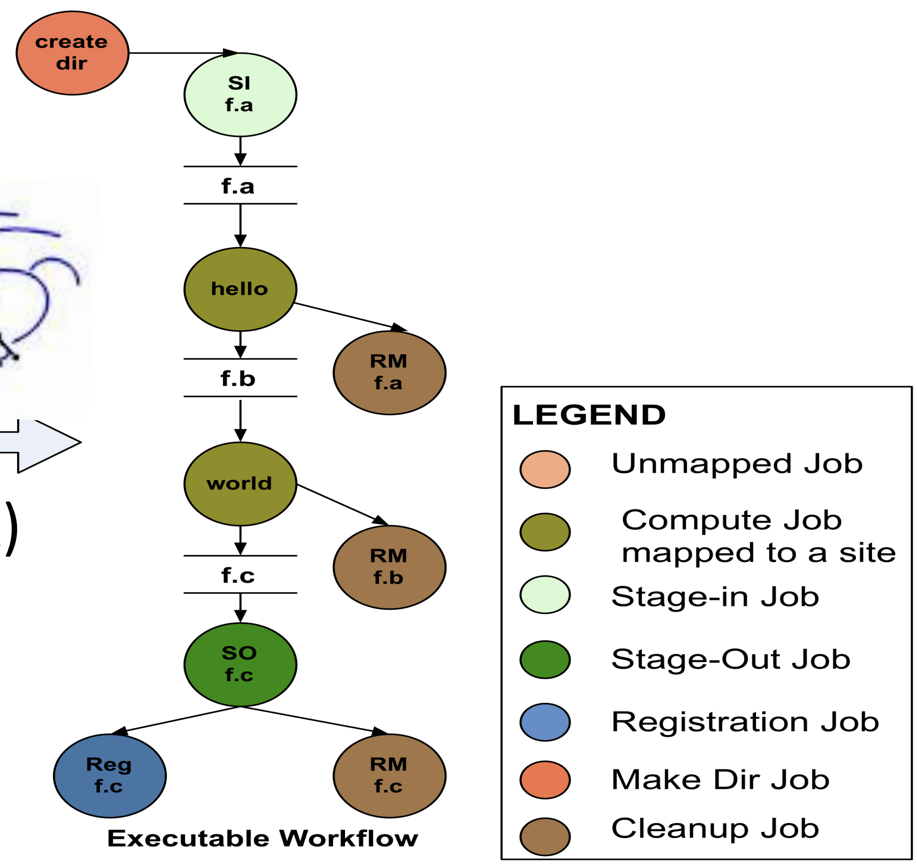
APIs for workflow specification (DAX--- DAG in XML)

Java, Perl, Python



Abstract Workflow

(DAX)



Executable Workflow

Example DAX File

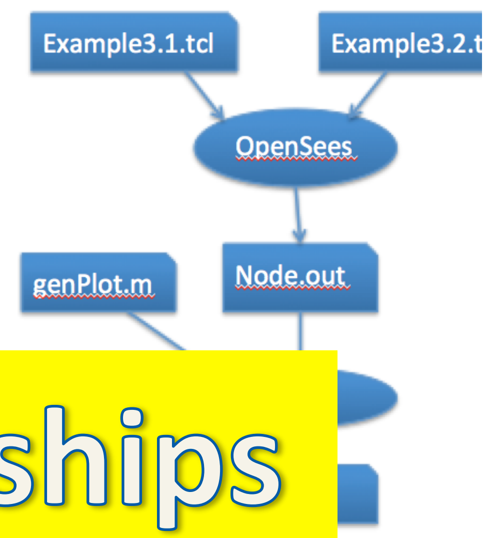
Example DAX File

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- generated: 2013-07-31 20:18:11.772766 -->
<!-- generated by: fmk -->
<!-- generator: python -->
<adag xmlns="http://pegasus.isi.edu/schema/DAX" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocatior
chema/DAX http://pegasus.isi.edu/schema/dax-3.4.xsd" version="3.4" name="dax">
    <file name="Example3.2.tcl">
        <pfn url="file:///home/neeshub/fmk/EXAMPLES/Pegasus1/inputs/Example3.2.tcl" site="local"/>
    </file>
    <file name="genPlot.m">
        <pfn url="file:///home/neeshub/fmk/EXAMPLES/Pegasus1/inputs/
    </file>
    <file name="Example3.1.tcl">
        <pfn url="file:///home/neeshub/fmk/EXAMPLES/Pegasus1/inputs/Example3.1.tcl" site="local"/>
    </file>
    <executable name="opensees" namespace="opensees" version="1.0" arch="x86_64" os="linux" installed="true">
        <pfn url="file:///grid/app/nees/opensees/bin/OpenSees" site="HCC_US_Fermigridosg1"/>
    </executable>
    <executable name="octave" namespace="system" version="1.0" arch="x86_64" os="linux" installed="true">
        <pfn url="file:///grid/app/nees/bin/octave-3.2.4.sh" site="HCC_US_Fermigridosg1"/>
    </executable>
    <job id="ID0000001" namespace="opensees" name="opensees" version="1.0">
        <argument>Example3.2.tcl</argument>
        <uses name="Example3.1.tcl" link="input"/>
        <uses name="Node.out" link="output"/>
        <uses name="Example3.2.tcl" link="input"/>
    </job>
    <job id="ID0000002" namespace="system" name="octave" version="1.0">
        <argument>--silent genPlot.m</argument>
        <uses name="Disp.jpg" link="output"/>
        <uses name="Node.out" link="input"/>
        <uses name="genPlot.m" link="input"/>
    </job>
    <child ref="ID0000002">
        <parent ref="ID0000001"/>
    </child>
</adag>
```

Resources

Jobs

Relationships



Scientific Workflows

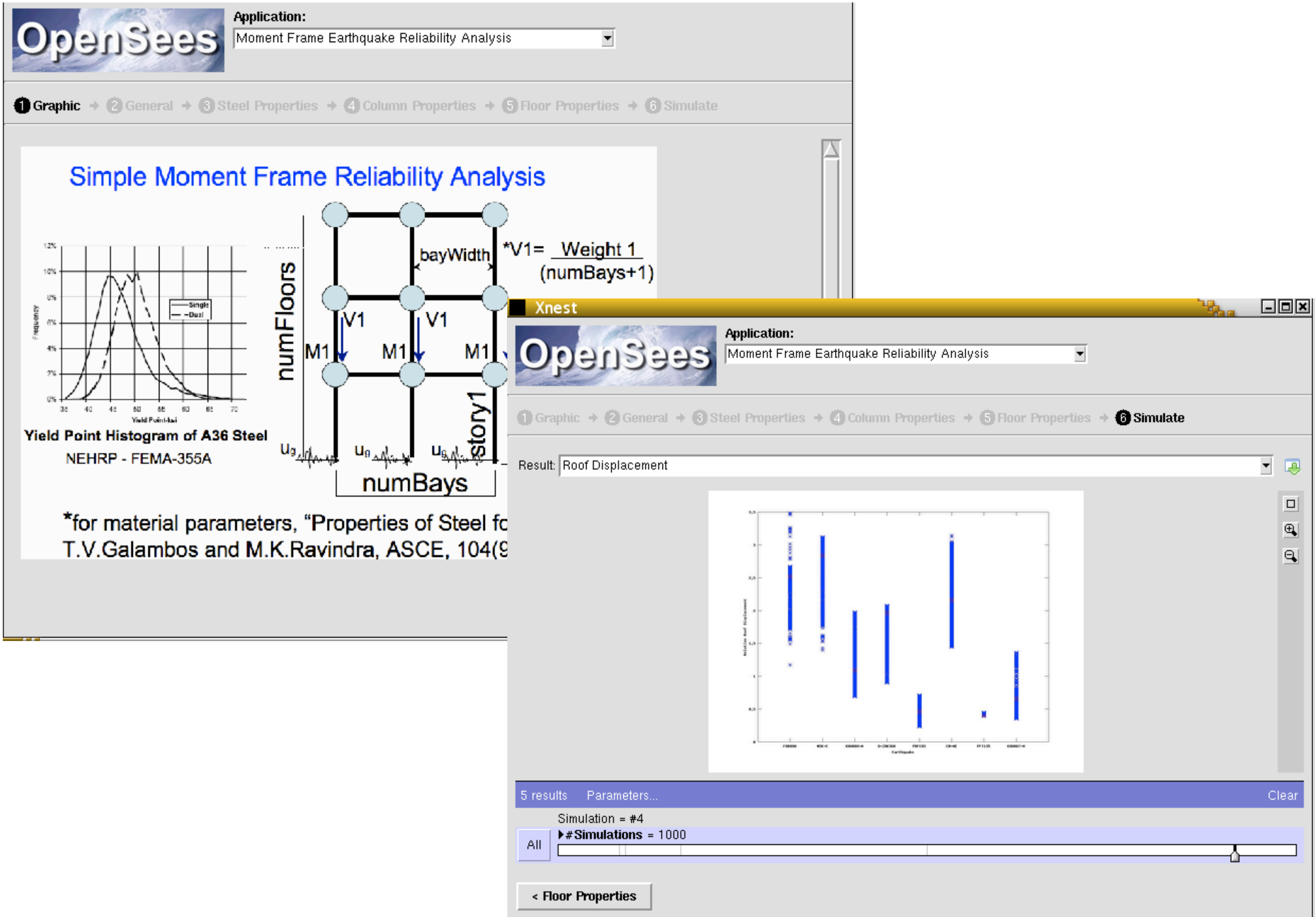
+

Cloud Computing

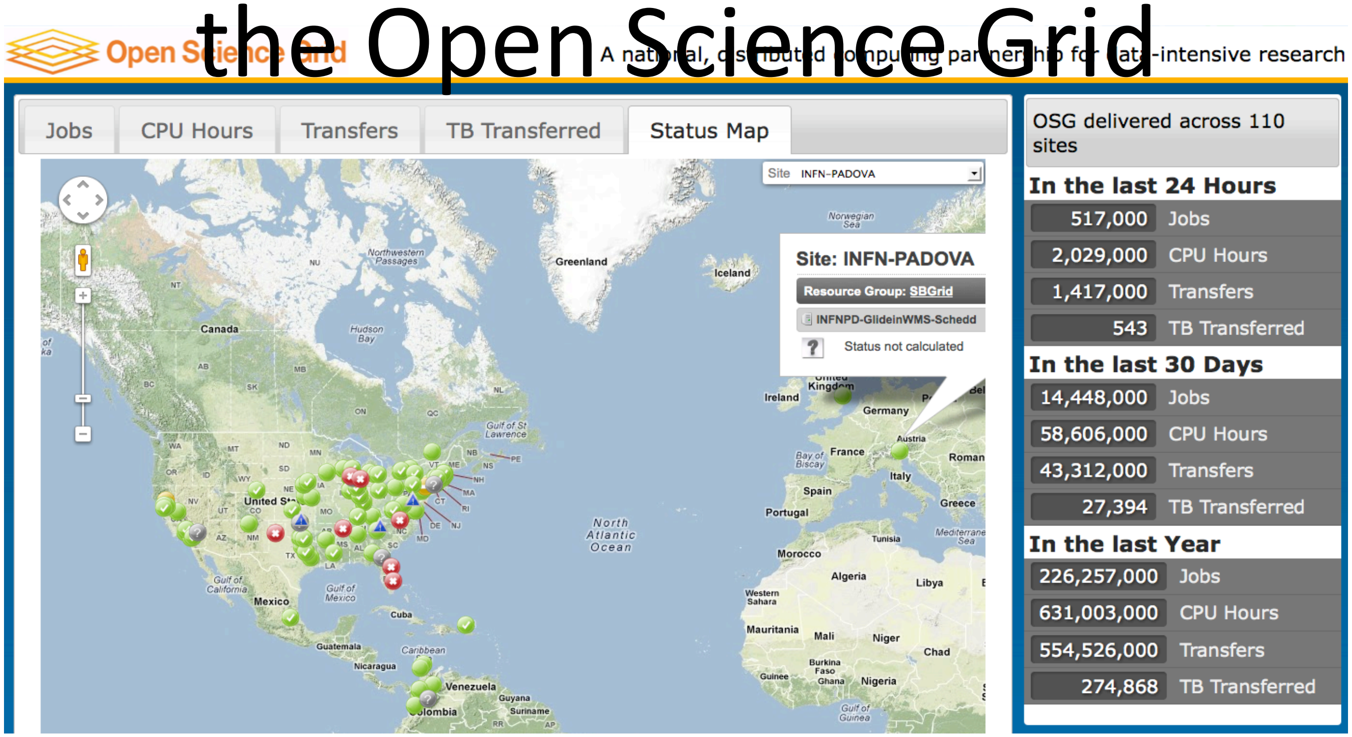
=

Future of Engineering Simulation

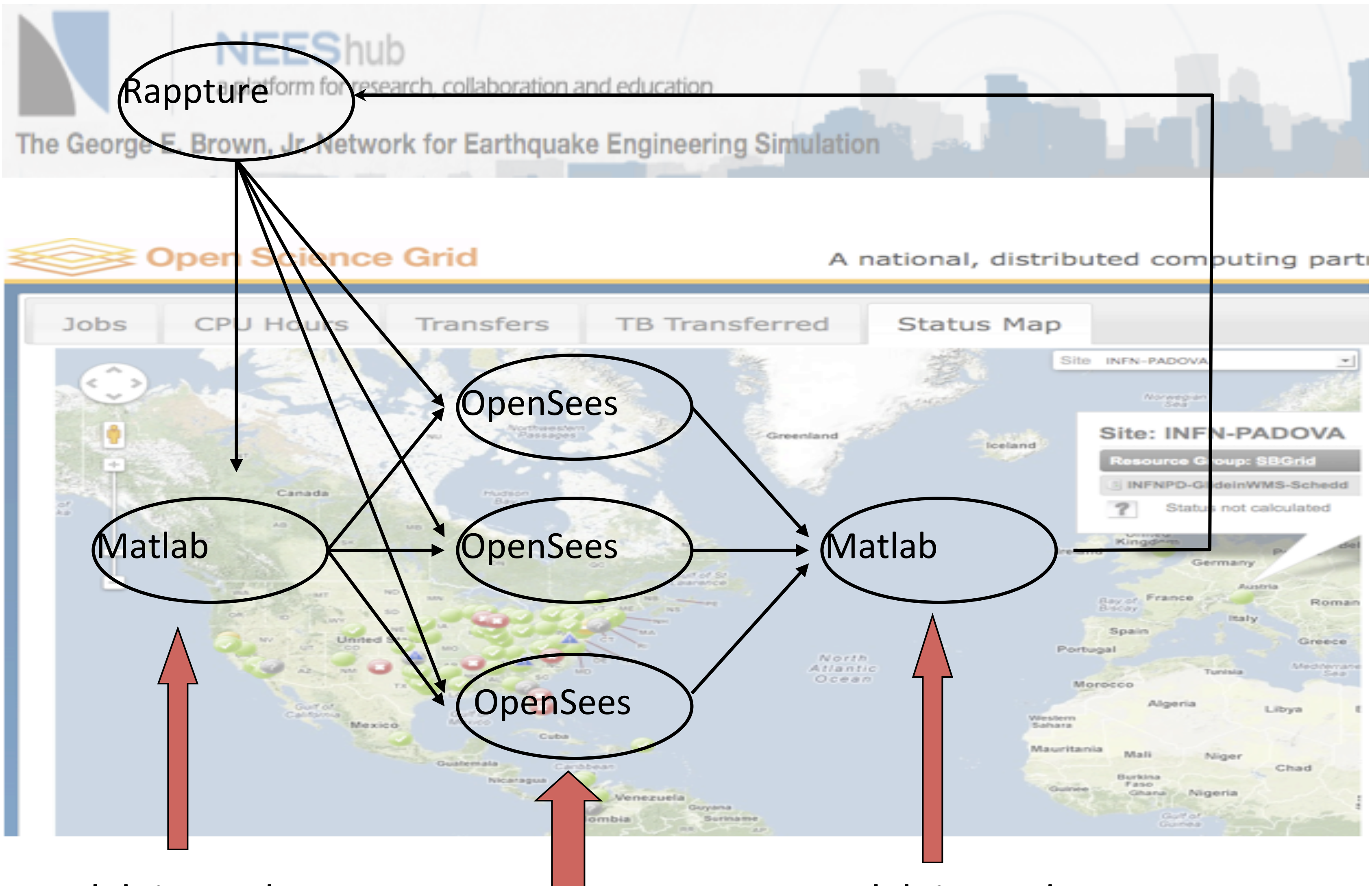
Example: Moment Frame Reliability



Moment Frame Reliability Tool in OpenSeesLab on NEEShub Makes use of Pegasus and the Resources of the Open Science Grid



And OpenSees and Matlab (Octave) running on these resources



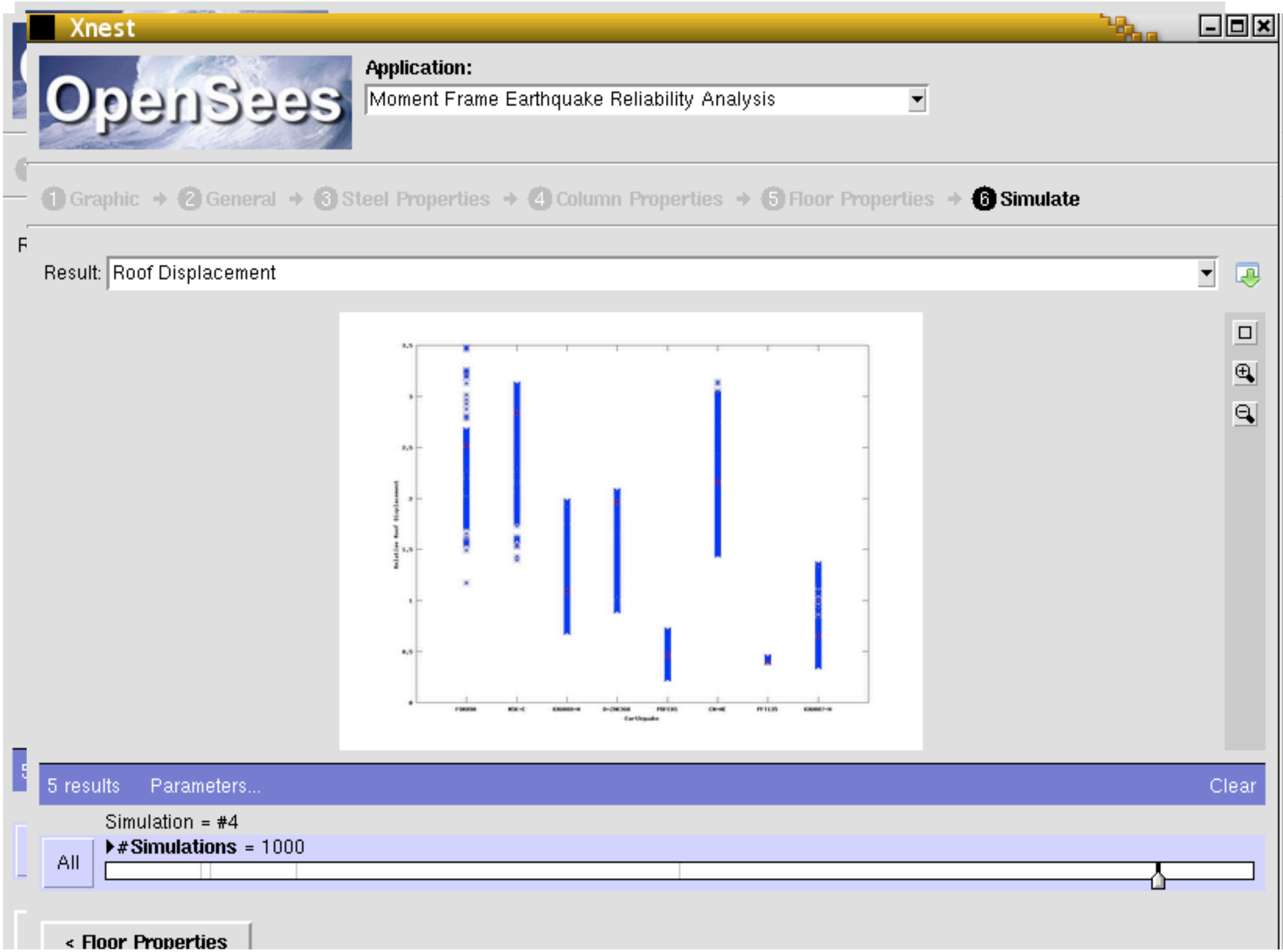
Matlab is used to generate random material properties

10's to 1000's of OpenSees Simulations

Matlab is used to process the results and generate figures

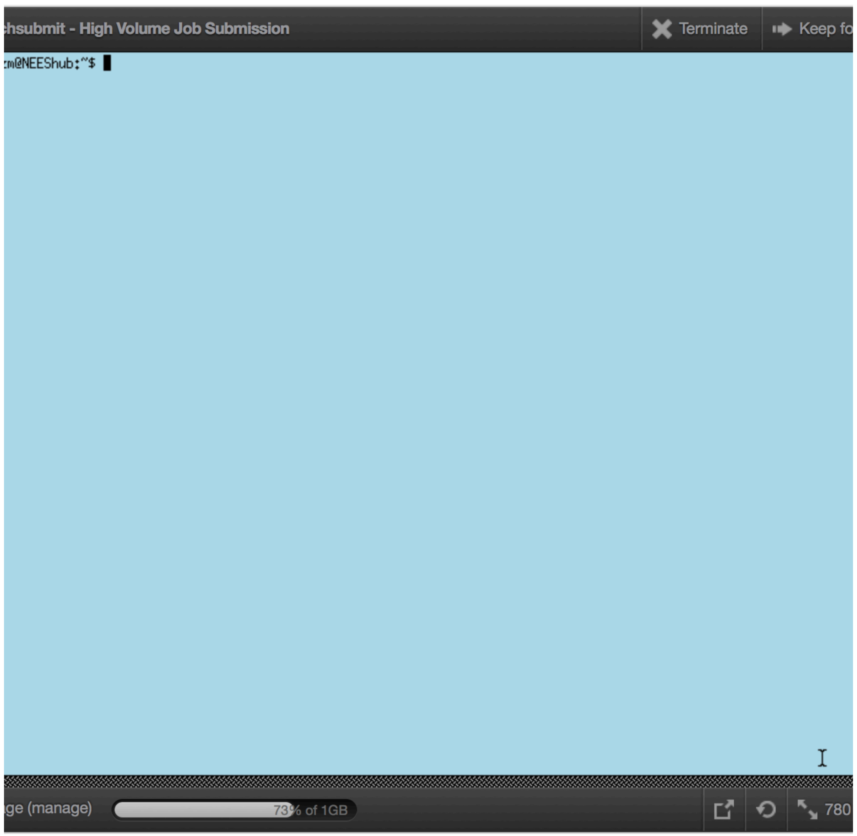
Pegasus is Responsible for moving the data from the NEEShub to the OSG, orchestrating the workflow and returning the results to NEEShub.

Moment Frame Reliability Analysis

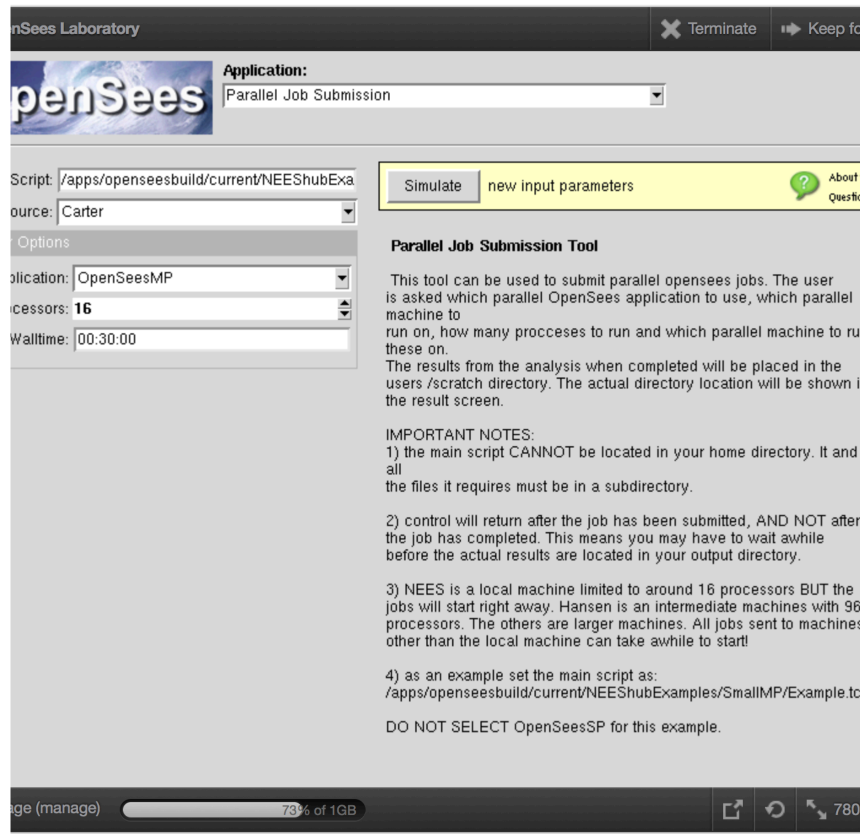


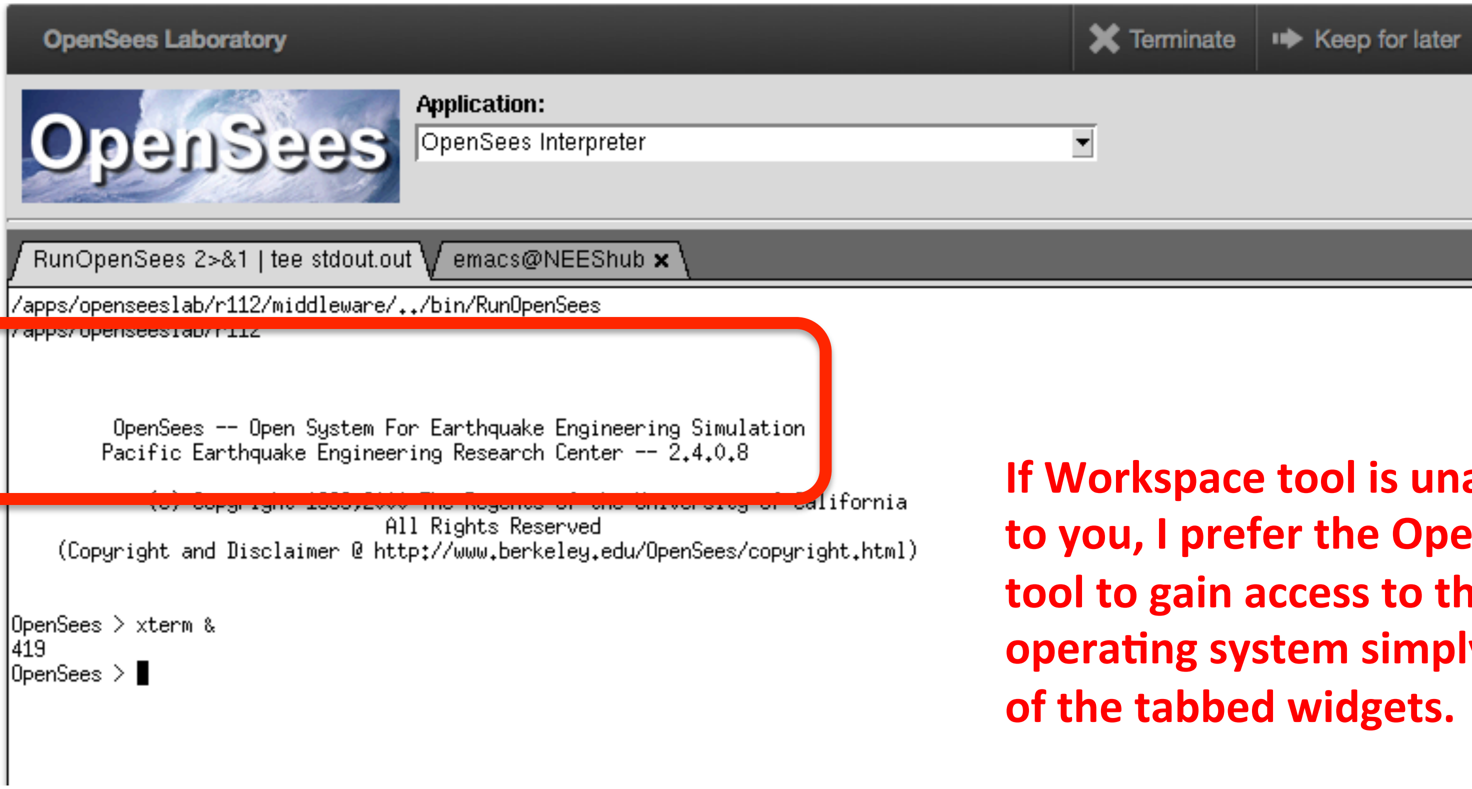
Using OpenSees & Cloud Resources on NEEShub?

Batchsubmit/Submit



OpenSees Lab

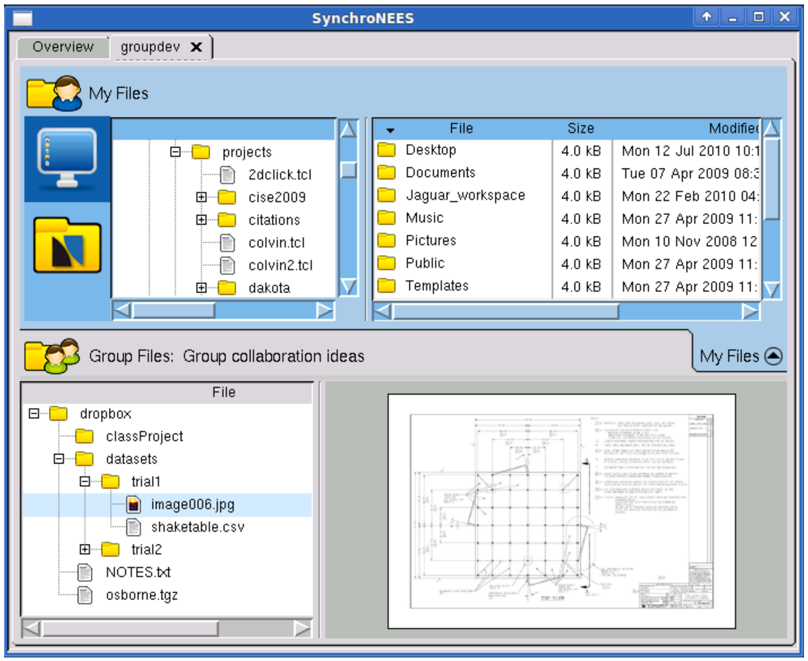




If Workspace tool is unavailablkle to you, I prefer the OpenSeesLab tool to gain access to the linux operating system simply because of the tabbed widgets.

Actually the Workspace Tool Is Best Tool for Job BUT Access To Tool is Limited to Developers

Manage Files using SynchroNEES



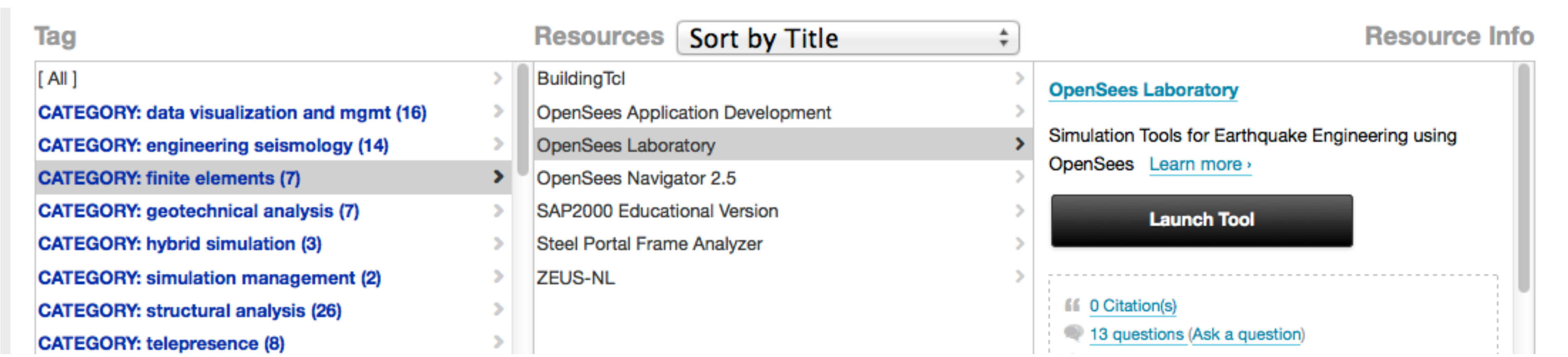
- SynchroNEES
  - Home space
  - Group space
  - Scratch space
  - Local machine

HANDS ON - Getting Started

- Register on NEEShub (nees.org)
- Might need (Submit a support ticket and ask for HPC access)
- https://nees.org/resources/openseeslab and click "Launch Tool"

OR

Find in https://nees.org/resources/tools



Now Some Demos

- Start the OpenSees Interpreter in OpenSeesLab
- Type xterm & at the prompt (see 3 slides back)
- Go to the xterm tab and type the following in the terminal:

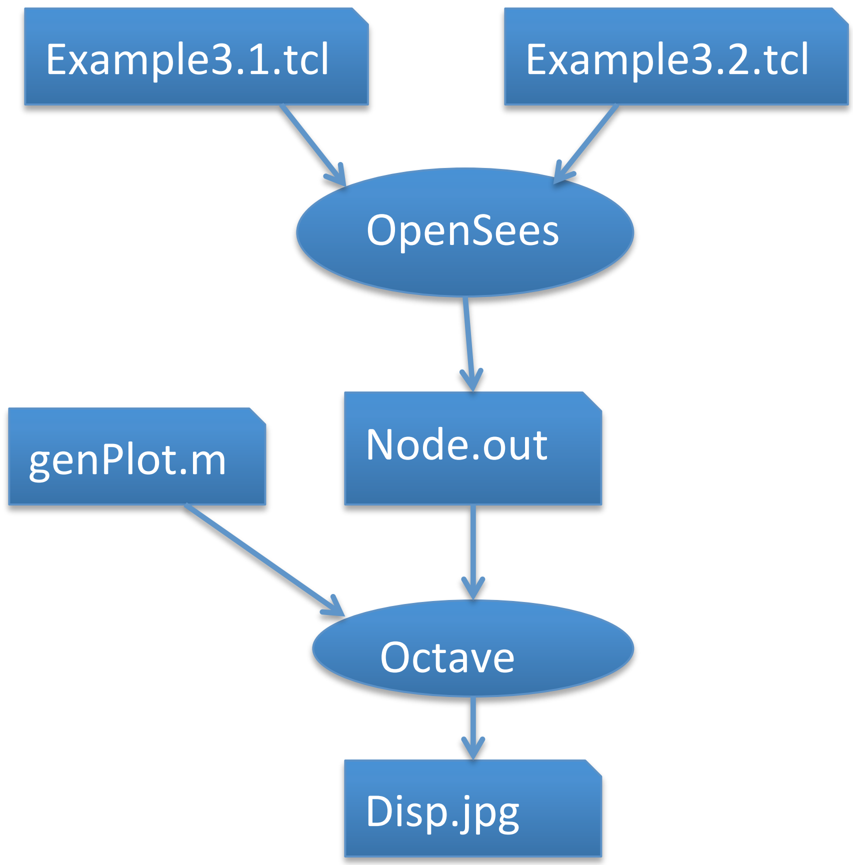
svn co svn://opensees.berkeley.edu/usr/local/svn/OpenSees/trunk/Workshops/SimWorkshop SimWorkshop

- You should now have SimWorkshop directory & 5 subdirectories!

/SimWorkshop/SmallSP
/SmallMP
/Pegasus1
/Pegasus2
/Pegasus3

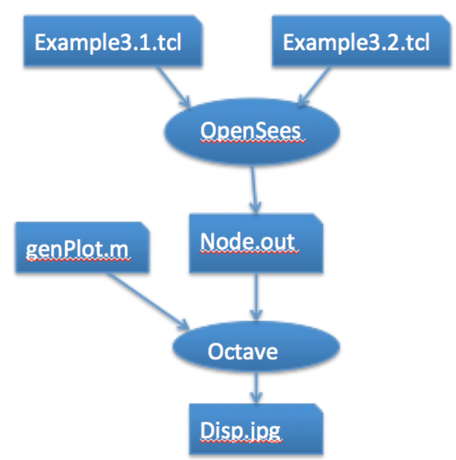
NOTE: xterm and batchsubmit offer a regular linux terminal. You you know linux you can do lots of things here (ls, cd, vi, emacs, ssh, scp!). It's just used so NEEShub can log how much time people spend using batchsubmit!

/Pegasus1 simple workflow



/Pegasus1 – dax.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!-- generated: 2013-07-31 20:18:11.772766 -->
<!-- generated by: fmk -->
<!-- generator: python -->
<adag xmlns="http://pegasus.isi.edu/schema/DAX" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLoca
chema/DAX http://pegasus.isi.edu/schema/dax-3.4.xsd" version="3.4" name="dax">
        <file name="Example3.2.tcl">
                <pfn url="file:///home/neeshub/fmk/EXAMPLES/Pegasus1/inputs/Example3.2.tcl" site="local"/>
        </file>
        <file name="genPlot.m">
                <pfn url="file:///home/neeshub/fmk/EXAMPLES/Pegasus1/inputs/genPlot.m" site="local"/>
        </file>
        <file name="Example3.1.tcl">
                <pfn url="file:///home/neeshub/fmk/EXAMPLES/Pegasus1/inputs/Example3.1.tcl" site="local"/>
        </file>
        <executable name="opensees" namespace="opensees" version="1.0" arch="x86_64" os="linux" installed="true">
                <pfn url="file:///grid/app/nees/opensees/bin/OpenSees" site="HCC_US_Fermigridosg1"/>
        </executable>
        <executable name="octave" namespace="system" version="1.0" arch="x86_64" os="linux" installed="true">
                <pfn url="file:///grid/app/nees/bin/octave-3.2.4.sh" site="HCC_US_Fermigridosg1"/>
        </executable>
        <job id="ID0000001" namespace="opensees" name="opensees" version="1.0">
                <argument>Example3.2.tcl</argument>
                <uses name="Example3.1.tcl" link="input"/>
                <uses name="Node.out" link="output"/>
                <uses name="Example3.2.tcl" link="input"/>
        </job>
        <job id="ID0000002" namespace="system" name="octave" version="1.0">
                <argument>--silent genPlot.m</argument>
                <uses name="Disp.jpg" link="output"/>
                <uses name="Node.out" link="input"/>
                <uses name="genPlot.m" link="input"/>
        </job>
        <child ref="ID0000002">
                <parent ref="ID0000001"/>
        </child>
</adag>
```



To submit the workflow type:

cd Pegaus1

emacs dax.xml (change fmk & dir location to reflect your setup

submit –v WF-OSGFactory_FERMI pegasus-plan --dax.xml &

/Pegasus2 same simple workflow

- Building & executing the .xml file is a pain!
- Can use python, java, …

```
#!/bin/sh

trap cleanup HUP INT QUIT ABRT TERM

cleanup()
{
#   echo "Abnormal termination by signal"
    kill -TERM `jobs -p`
}

. /etc/environ.sh
use -e -r pegasus-4.2.0

./opensees-dax-generator.py > dax.xml

unuse -e pegasus-4.2.0

submit -v WF-OSGFactory_FERMI pegasus-plan --dax dax.xml &
```

- To both create the dax and submit in Pegasus2 just type:
  ./submit

The complex stuff is in the opensees-dax-generator.py

opensees_dax_generator.py eof (end-of-file)

```
f# Create a abstract workflow
dax = ADAG("dax")

# Add executables to the DAX-level replica catalog
opensees = Executable(namespace="opensees", name="opensees", version="1.0", os=\
"linux", arch="x86_64", installed=True)

#opensees.addPFN(PFN("file:///apps/opensees/bin/OpenSees", "local"))
opensees.addPFN(PFN("file:///grid/app/nees/opensees/bin/OpenSees", "HCC_US_Ferm\
igridosg1"))
dax.addExecutable(opensees)

octave = Executable(namespace="system", name="octave", version="1.0", os="linux\
", arch="x86_64", installed=True)
#octave.addPFN(PFN("file:///usr/bin/octave", "local"))
octave.addPFN(PFN("file:///grid/app/nees/bin/octave-3.2.4.sh", "HCC_US_Fermigri\
dosg1"))
dax.addExecutable(octave)

add_opensees_job(dax)
add_plot_job(dax)

# Write the DAX to stdout
dax.writeXML(sys.stdout)
```

opensees_dax_generator.py near top of file

```
def add_opensees_job(dax):

    # input files to the DAX-level replica catalog
    input_files = list()
    names = ["Example3.1.tcl", "Example3.2.tcl"]

    for name in names:
        ifile = File(name)
        ifile.addPFN(PFN("file://" + os.getcwd() + "/inputs/" + name, "local"))
        dax.addFile(ifile)
        input_files.append(ifile)

    # job - materialProperties.tcl.XX model.tcl motion.tcl.YY dynamic WW
    job = Job(namespace="opensees", name="opensees", version="1.0")
    job.addArguments("Example3.2.tcl")

    for f in input_files:
        job.uses(f, link=Link.INPUT)

    out_name = "Node.out"
    f1 = File(out_name)
    job.uses(f1, link=Link.OUTPUT)
    bag_files[out_name] = f1

    dax.addJob(job)

    # add the job to the bag so we can look it up later
    bag_jobs["opensees"] =
```

/Pegasus3

more complex workflow
(the one used in the Reliability Analysis Tool)

- Looks complicated for simple problems .. What about bigger ones!

```
#!/bin/sh

trap cleanup HUP INT QUIT ABRT TERM

cleanup()
{
#   echo "Abnormal termination by signal"
    kill -TERM `jobs -p`
}

. /etc/environ.sh
use -e -r pegasus-4.2.0

./opensees-dax-generator.py --num-mat-props=$1 --num-motions=$2 > dax.xml

unuse -e pegasus-4.2.0

submit -v WF-OSGFactory_FERMI pegasus-plan --dax dax.xml &
```

To run in Pegasus2 type:
./submit 3 3

Look at the opensees-dax-generator.py

Creating Your Own Workflow Later

- The complex stuff is in the opensees-dax-generator.py file for both Pegasus1 and Pegasus2.
- If you look at the code for both you will see they are somewhat easy to understand and very easy to customize (once you understand them) to your own workflows
- **If you can write OpenSees scripts you can edit and modify these files!**

If you are stuck, contact Pegasus team directly (email: pegasus-support@isi.edu) **They can help with the scripts.**