



**OpenSees**

Open System for Earthquake Engineering Simulation  
Pacific Earthquake Engineering Research Center

# Getting Started With OpenSees and OpenSees on NEEShub

Frank McKenna

October, 2012

*OpenSees is Sponsored by:*

George E. Brown Network for Earthquake Engineering  
(NEES) through NEEScomm

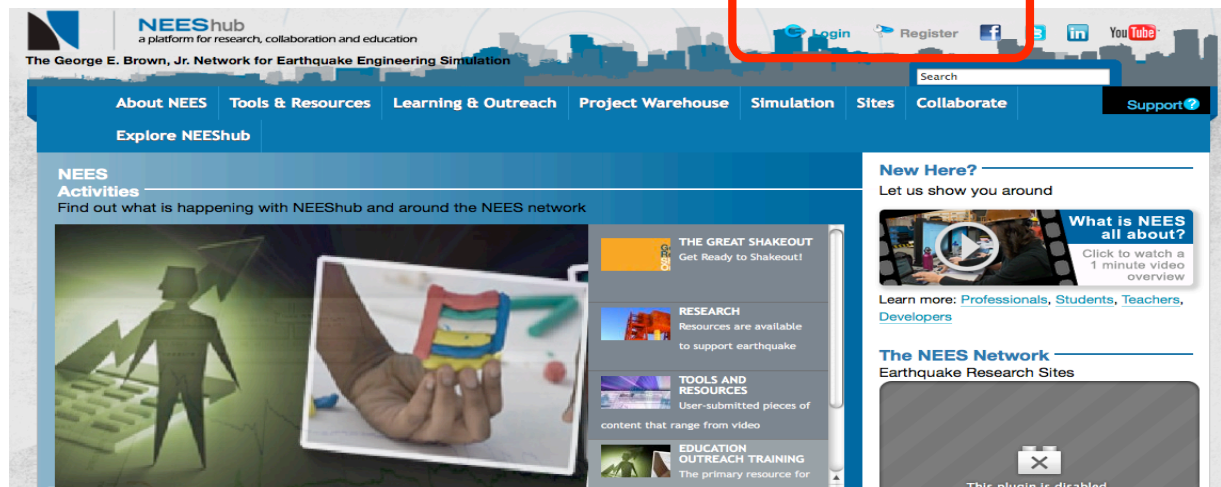
Pacific Earthquake Engineering Research Center (PEER)



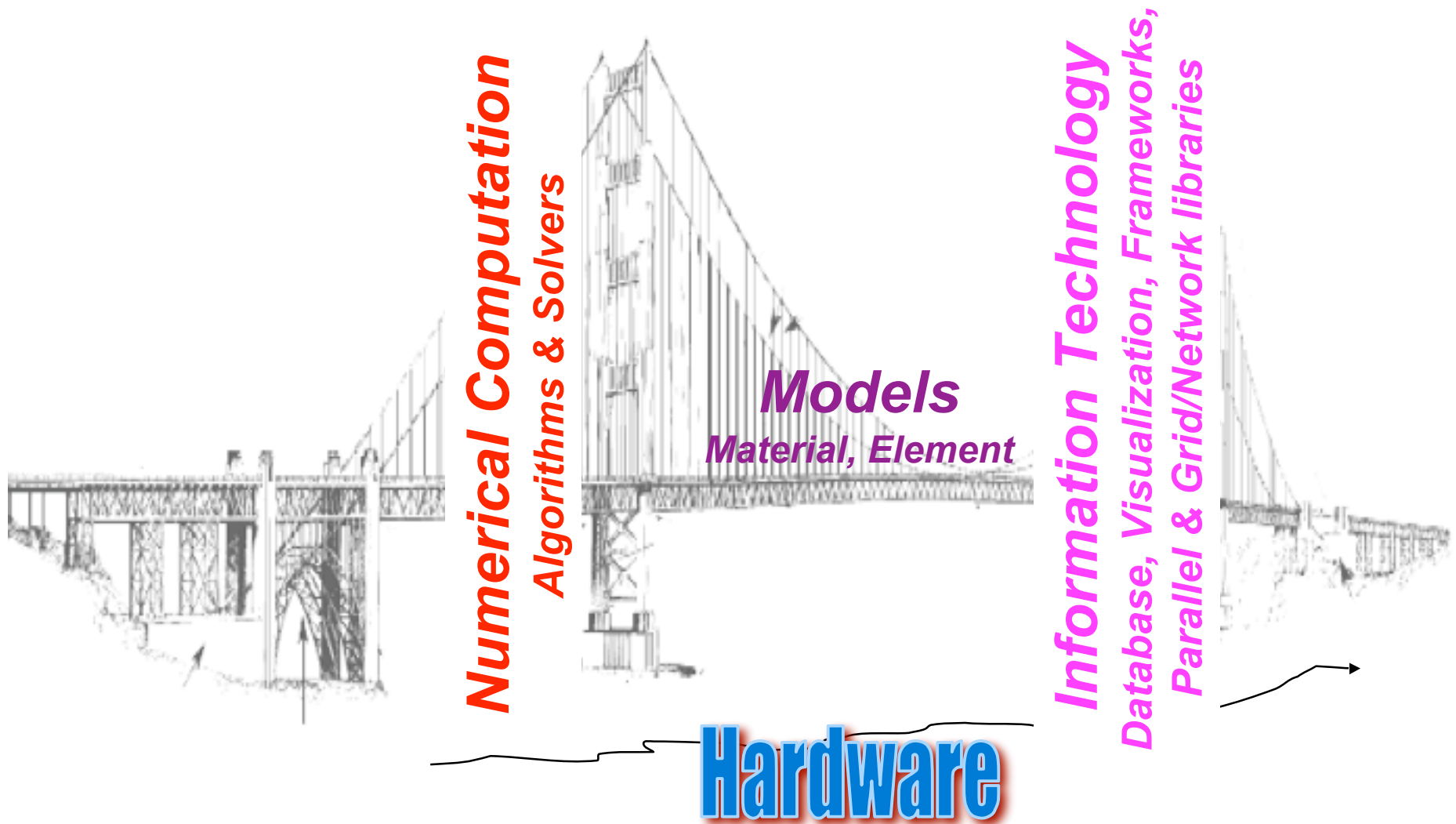
# Quick Exercise

1. *Register for Account on NEEShub'*
  - a. *Open Browser and go to NEES website (<http://nees.org>)*
  - b. *click register and enter the details*
  - c. *login*
  - d. *Select Tools from Tools & Resources Menu*
  - e. *find OpenSeesLab tool under simulation tools and launch it.*

**Login/Register**



# Building Blocks for Modern Simulation Code



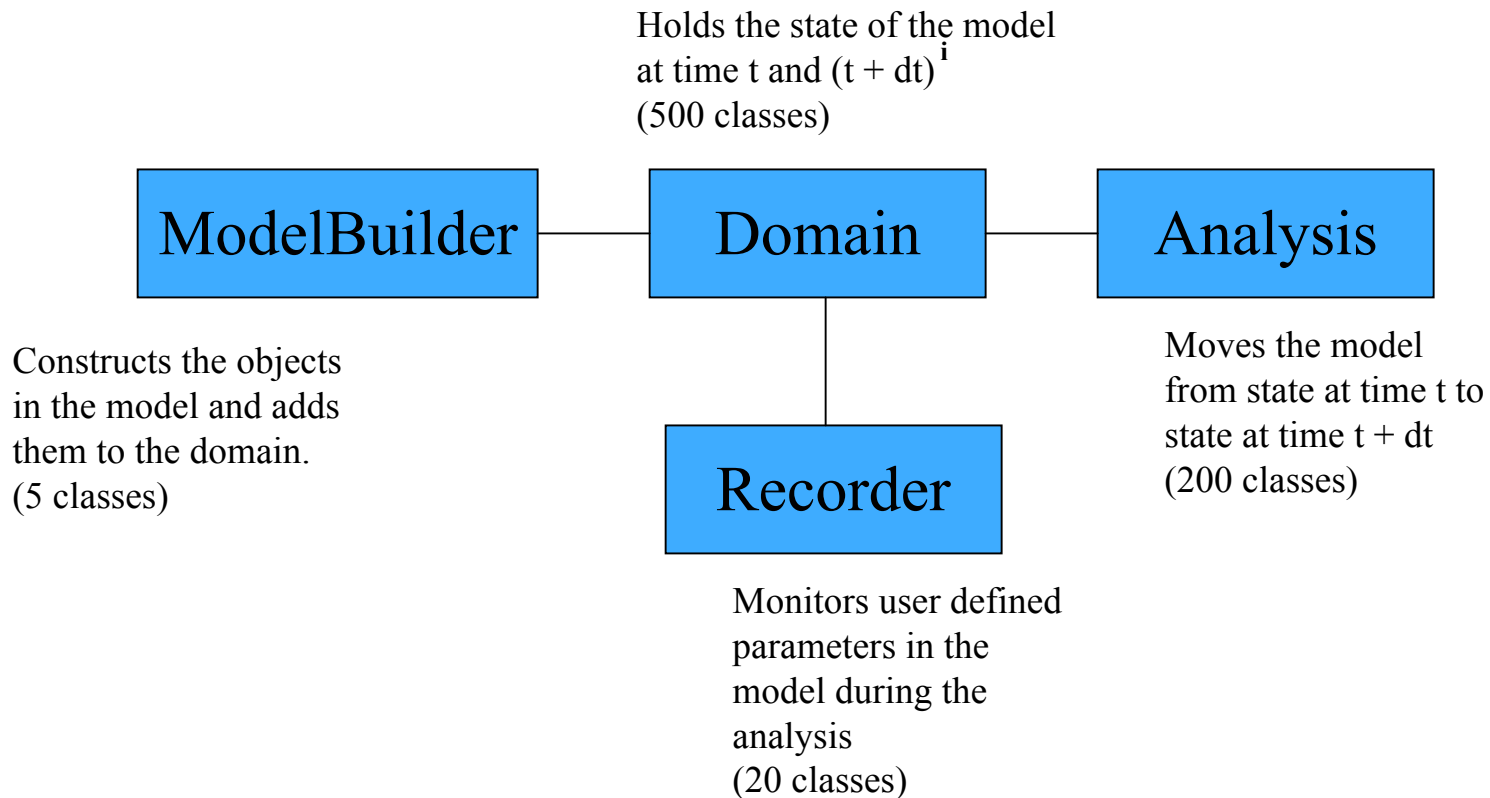
Open-Source - Leave it out there for community

# What is OpenSees?

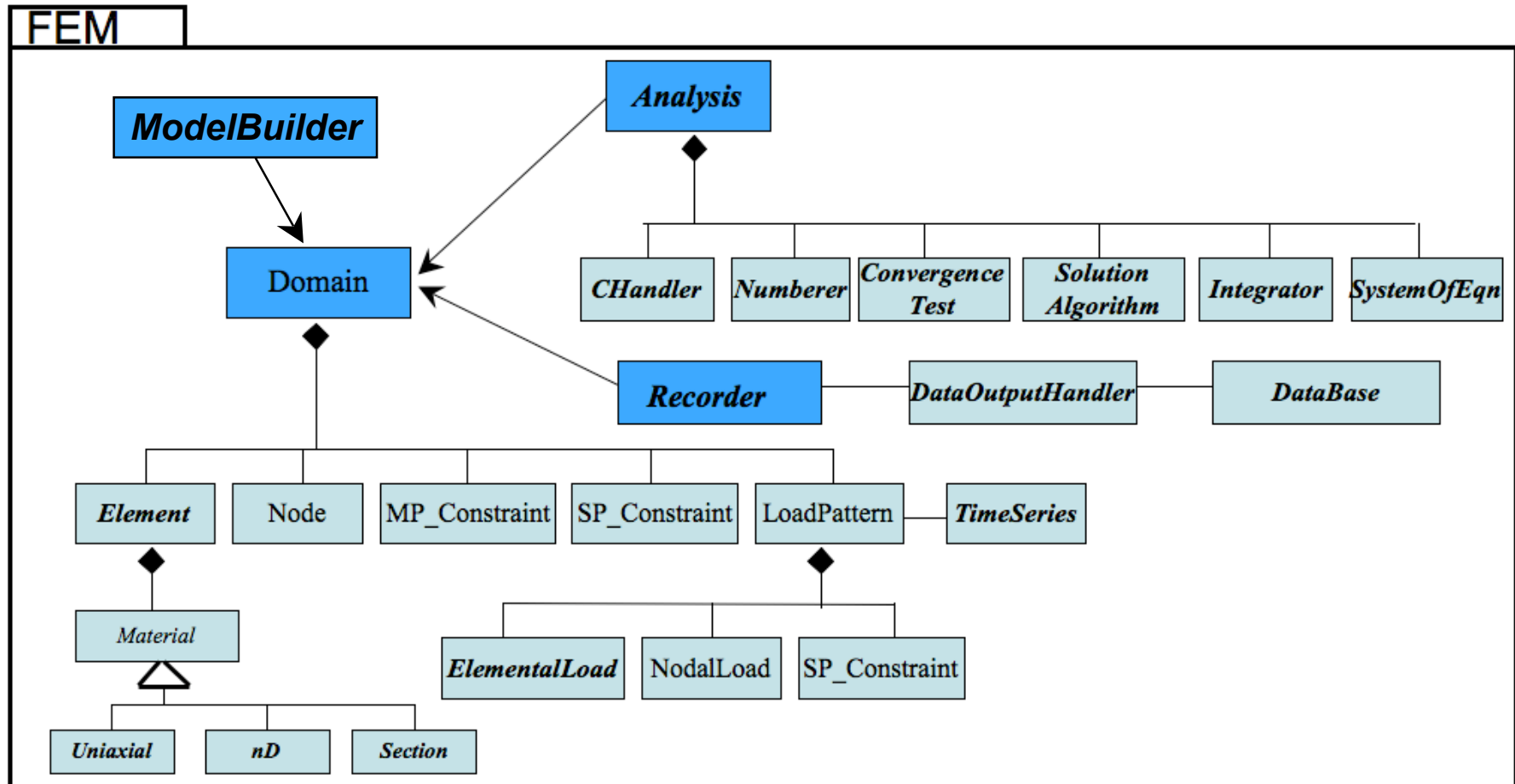
- A software *framework* for simulation applications in earthquake engineering using finite element methods. OpenSees is not an application.
- A communication mechanism for exchanging and building upon research accomplishments.
- As open-source software, it has the potential for a community code for earthquake engineering.

# Main Abstractions in OpenSees Framework

---

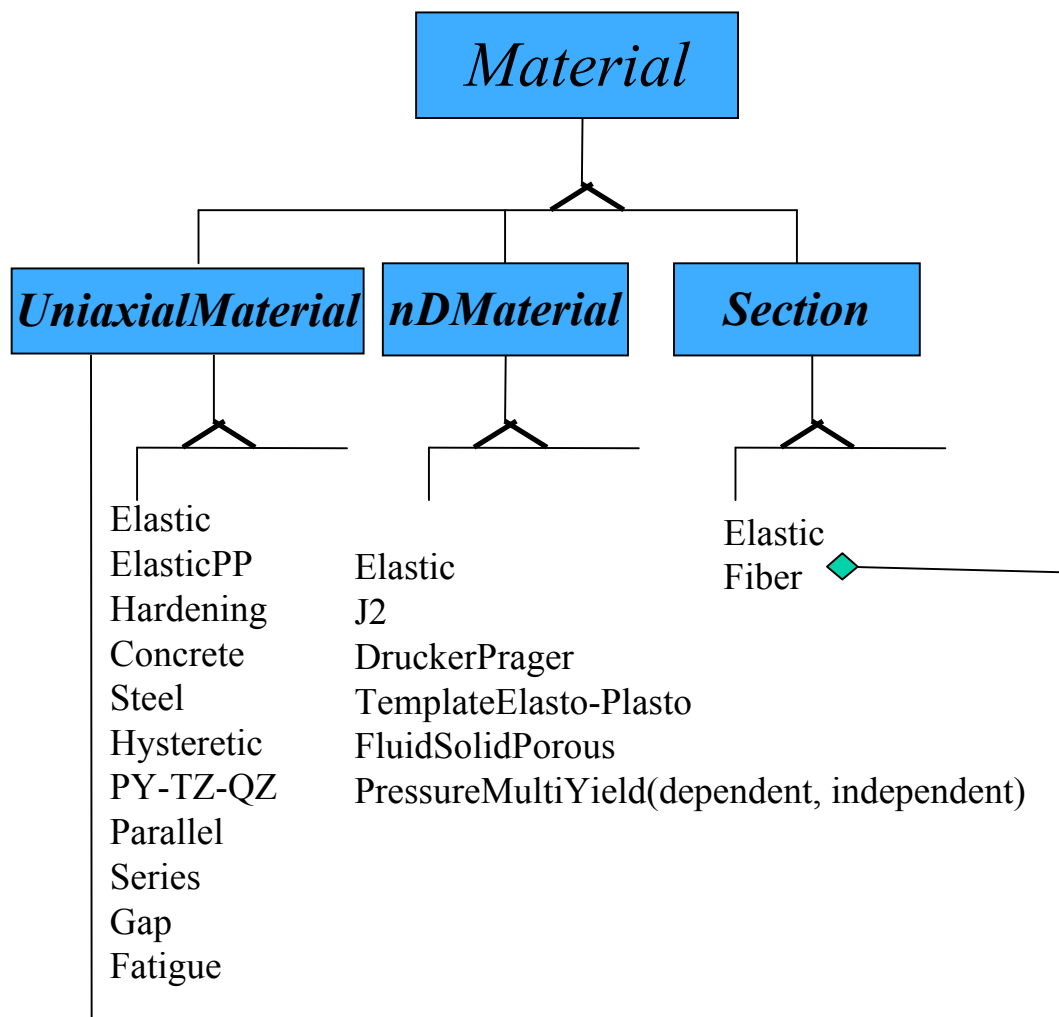


# Main Abstract Classes

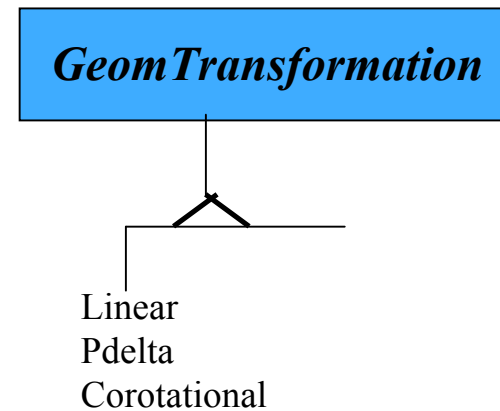


*There are other parts to the framework for Parallel Processing, Reliability & Optimization that we won't talk about today.*

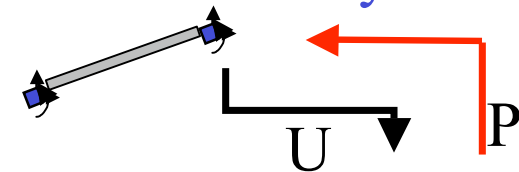
# Each Abstract Class has a Number of Concrete Classes



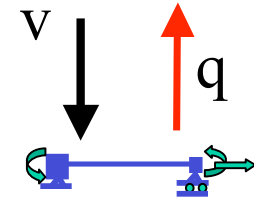
(over 250 material classes)



Element in Global System



Geometric Transformation



Element in Basic System

# How Do People Use the OpenSees Framework?

- Provide their own main() function in C++ and link to framework.
- Use OpenSees interpreter<sup>S</sup>. These are extensions of the Tcl interpreters, tclsh and wish, for performing finite element analysis.

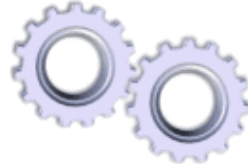
1. OpenSees.exe
  2. OpenSeesTk.exe
  3. OpseesSP.exe
  4. OpenSeesMP.exe
- } sequential processing
- } parallel processing



# Tcl Interpreters

- **wish** and **tclsh** **are tcl interpreters**.
  - Interpreters (Perl, Matlab, Ruby) are programs that execute programs written in a programming language immediately.
  - There is no separate compilation & linking.
  - An interpreted program runs slower than a compiled one.

puts “sum of 2 and 3 is [expr 2 + 3]”



sum of 2 and 3 is 5

```
Terminal — tclsh8.4 — 85x9
fmk:~$ tclsh
% puts "sum of 2 and 3 is [expr 2 + 3]"
sum of 2 and 3 is 5
% █
```

# What is Tcl

- **Tcl is a dynamic programming language.**
  - It is a string based command language.
  - Variables and variable substitution
  - Expression evaluation
  - Basic control structures (if , while, for, foreach)
  - Procedures
  - File manipulation
  - Sourcing other files.
- Command syntax:  
**command arg1 arg2 ...**
- Help
  1. <http://www.tcl.tk/man/tcl8.5/tutorial/tcltutorial.html>

# Example Tcl

## •variables & variable substitution

```
>set a 1
1
>set b a
a
>set b $a
1
```

## •file manipulation

```
>set fileId [open tmp w]
??
>puts $fileId "hello"
>close $fileID
>type tmp
hello
```

## •sourcing other files

```
>source Example1.tcl
```

## •expression evaluation

```
>expr 2 + 3
5
>set b [expr 2 + $b]
3
```

## •lists

```
>set a {1 2 three}
1 2 three
>set la [llength $a]
3
>set start [lindex $a 0]
1
>lappend a four
1 2 three four
```

## •procedures & control structures

```
> for {set i 1} {$i < 10} {incr i 1} {
    puts "i equals $i"
}
...
> set sum 0
foreach value {1 2 3 4} {
    set sum [expr $sum + $value]
}
>puts $sum
10
>proc guess {value} {
    global sum
    if {$value < $sum} {
        puts "too low"
    } else {
        if {$value > $sum} {
            puts "too high"
        } else { puts "you got it!"}
    }
}
> guess 9
too low
```

# Tcl Math Gotcha

## (most programming languages)

- If you add, subtract, multiply and divide two integer numbers the result is an integer.

```
> set a [expr 1/2]
0
```

- If you add, subtract, multiply and divide an integer number and a floating-point number, then the result is a floating-point number.

```
>set b [expr 1./2]
0.5
```

- Computers work store numbers in binary format and have a finite number of bytes for each number. Therefore computers cannot store numbers or do math exactly

```
>set c [expr 1.2/0.1]
11.999999999999998
```

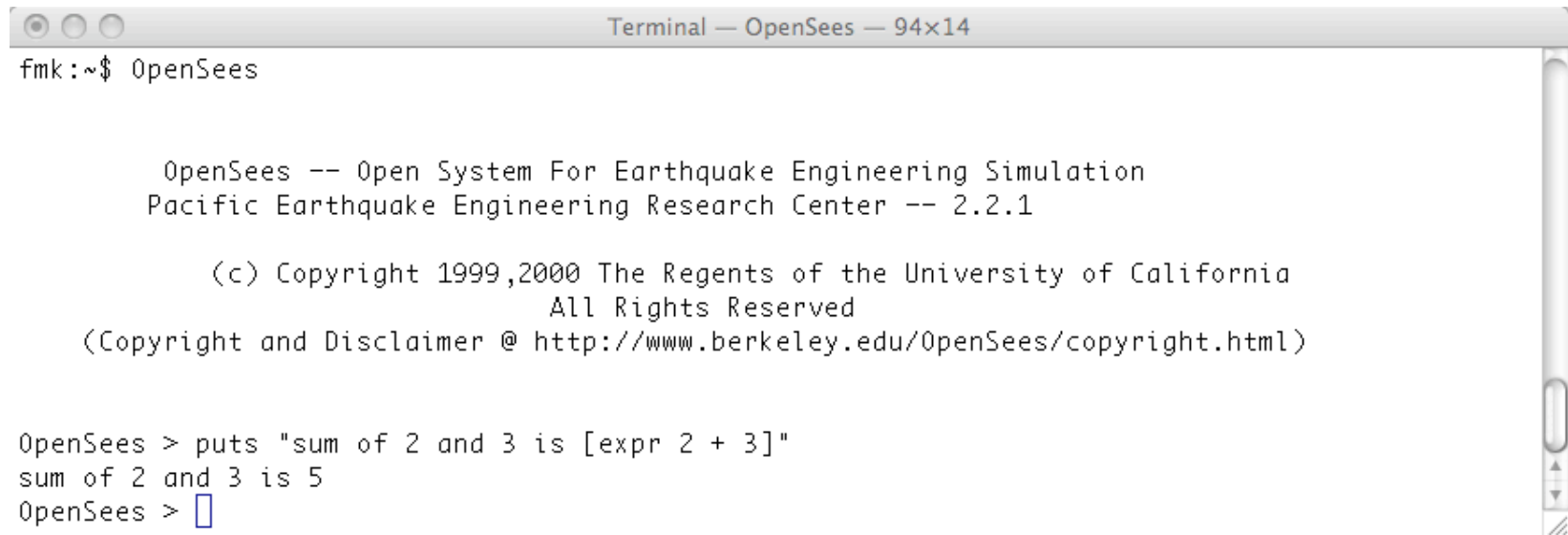
# OpenSees Interpreters

---

- The OpenSees interpreters are tcl interpreters which have been **extended** to include commands for finite element analysis:
  1. Modeling – create nodes, elements, loads and constraints
  2. Analysis – specify the analysis procedure.
  3. Output specification – specify what it is you want to monitor during the analysis.
  
- Being interpreters, this means that the files you create and submit to the OpenSees interpreters **are not input files**. You are creating and submitting **PROGRAMS**.

# OpenSees.exe

- An interpreter that extends tclsh for FE analysis.



```
Terminal — OpenSees — 94x14
fmk:~$ OpenSees

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

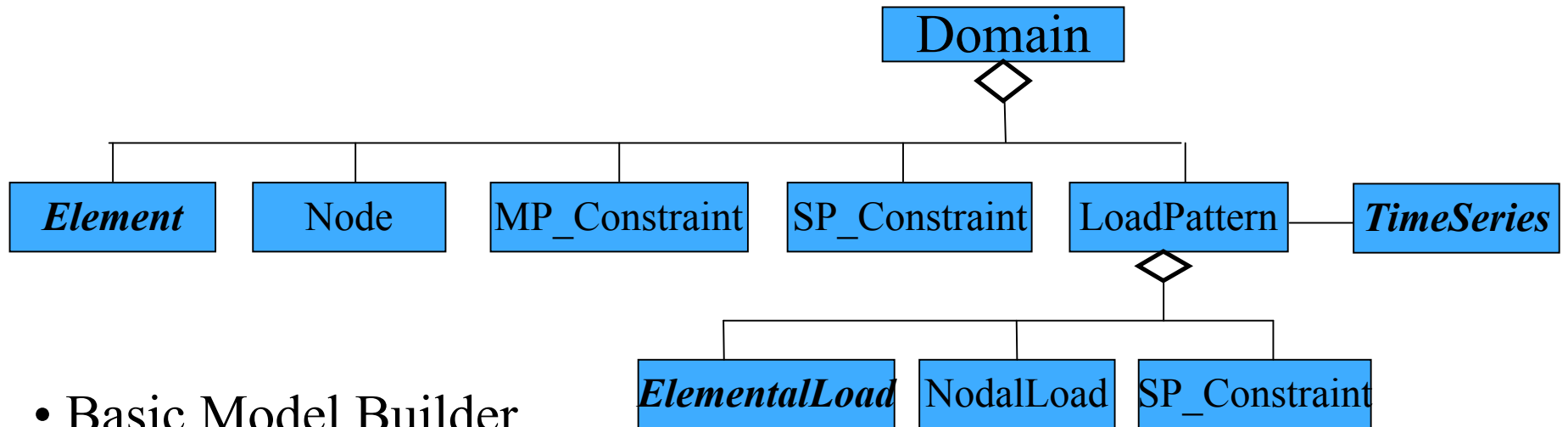
(c) Copyright 1999,2000 The Regents of the University of California
All Rights Reserved
(Copyright and Disclaimer @ http://www.berkeley.edu/OpenSees/copyright.html)

OpenSees > puts "sum of 2 and 3 is [expr 2 + 3]"
sum of 2 and 3 is 5
OpenSees > 
```

**WARNING: There is no GUI!**

# model Command

\*Adds the modeling commands to the interpreter.



- Basic Model Builder

```
model Basic -ndm ndm? <-ndf ndf?>
```

This command now adds the following commands to the interpreter:

**node mass element equalDOF fix fixX fixY fixZ**  
**pattern timeSeries load eleLoad sp**  
**uniaxialMaterial nDMaterial section geomTransf**  
**fiber layer patch block2D block3D**

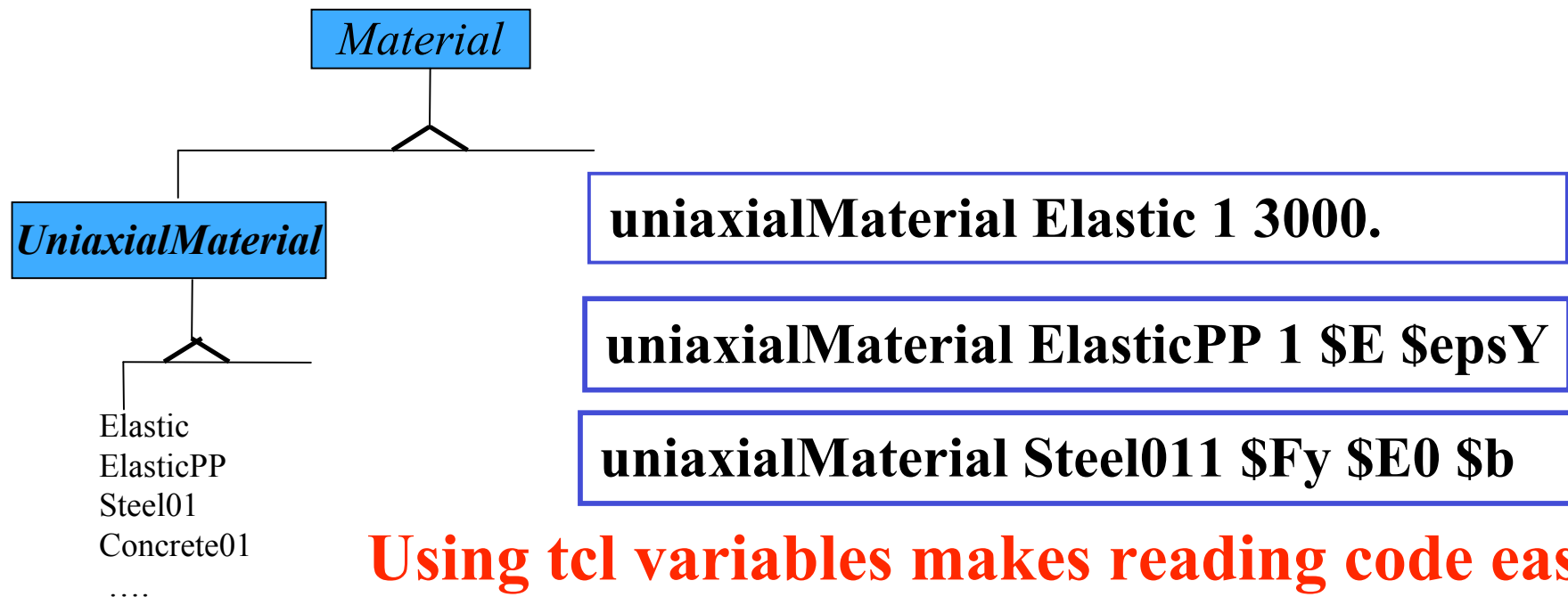
# Why understanding the class structure is useful

**command arg1 arg2 ...**

**command** typically resembles abstract class name

**arg1** typically resembles concrete class name

**arg2 ...** typically follows constructor args



**Using tcl variables makes reading code easier**

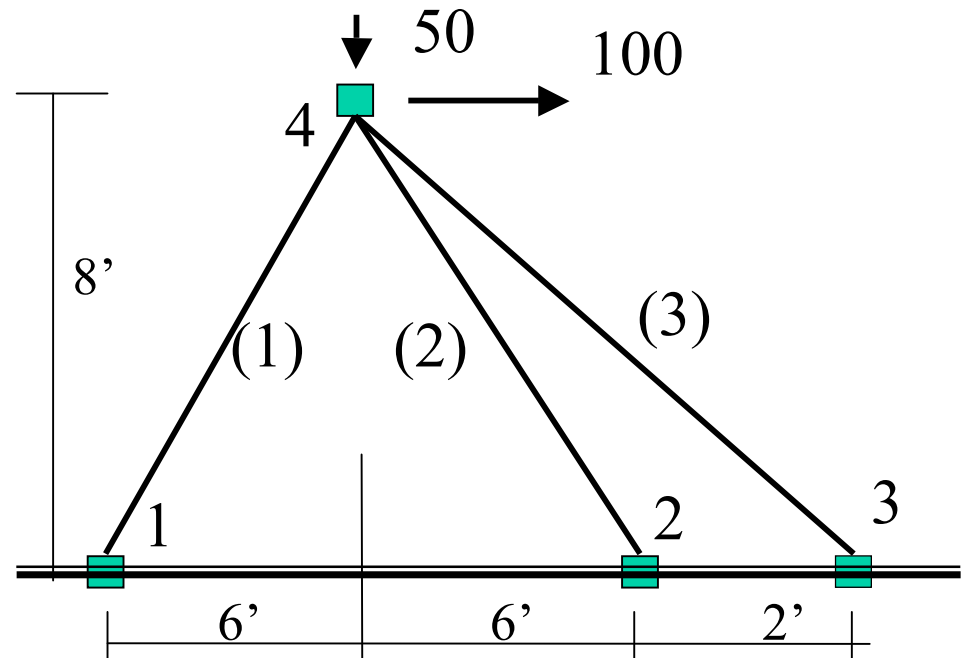


# Truss example:

```

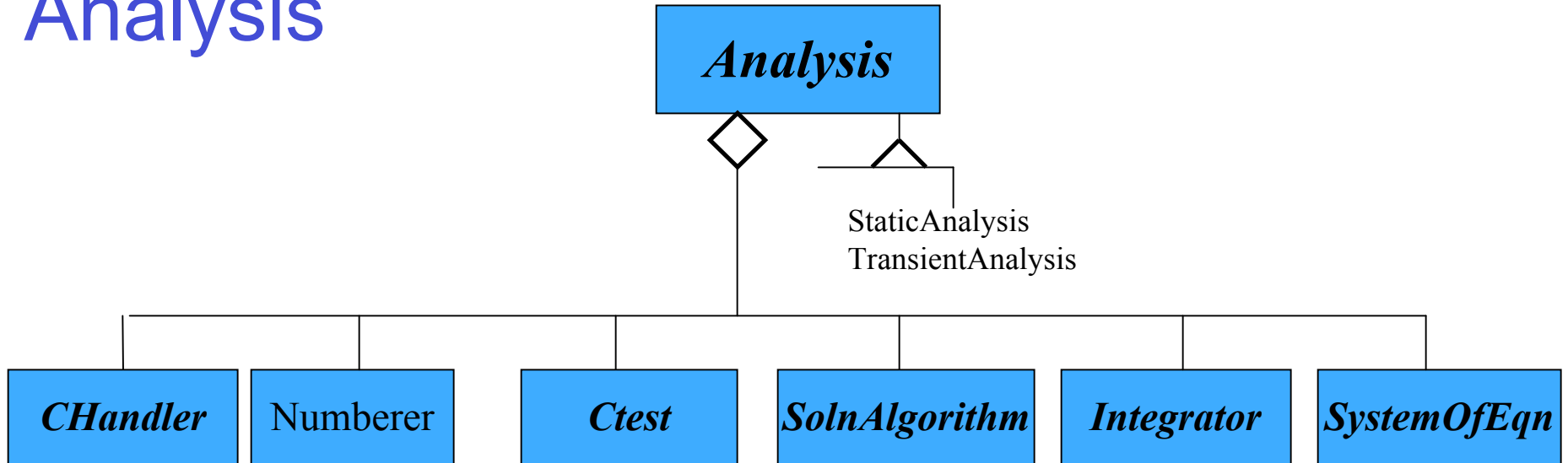
model Basic -ndm 2 -ndf 2
node 1 0.0 0.0
node 2 144.0 0.0
node 3 168.0 0.0
node 4 72.0 96.0
fix 1 1 1
fix 2 1 1
fix 3 1 1
uniaxialMaterial Elastic 1 3000.0
element truss 1 1 4 10.0 1
element truss 2 2 4 5.0 1
element truss 3 3 4 5.0 1
timeSeries Linear 1
pattern Plain 1 1 {
  load 4 100.0 -50.0
}

```



|   | E    | A  |
|---|------|----|
| 1 | 3000 | 10 |
| 2 | 3000 | 5  |
| 3 | 3000 | 5  |

# Analysis



handler type? args...

numberer type? args...

test type? args...

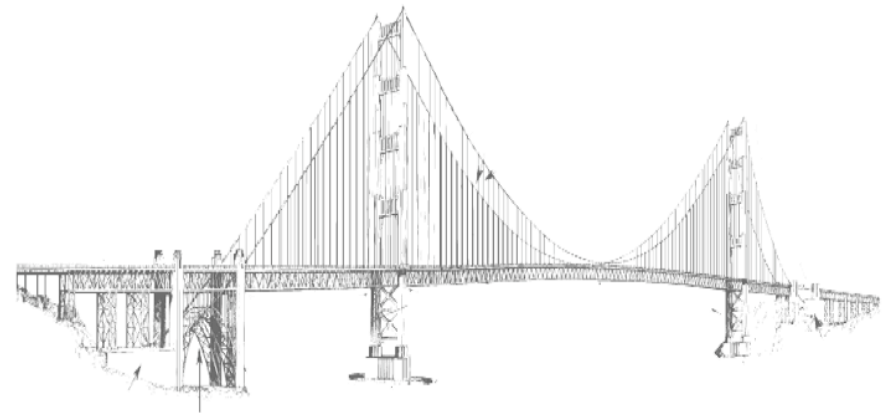
algorithm type? args...

integrator type? args...

system type? args...

analysis type? args..

analyze args ...



# Example Analysis:

- Static Nonlinear Analysis with LoadControl

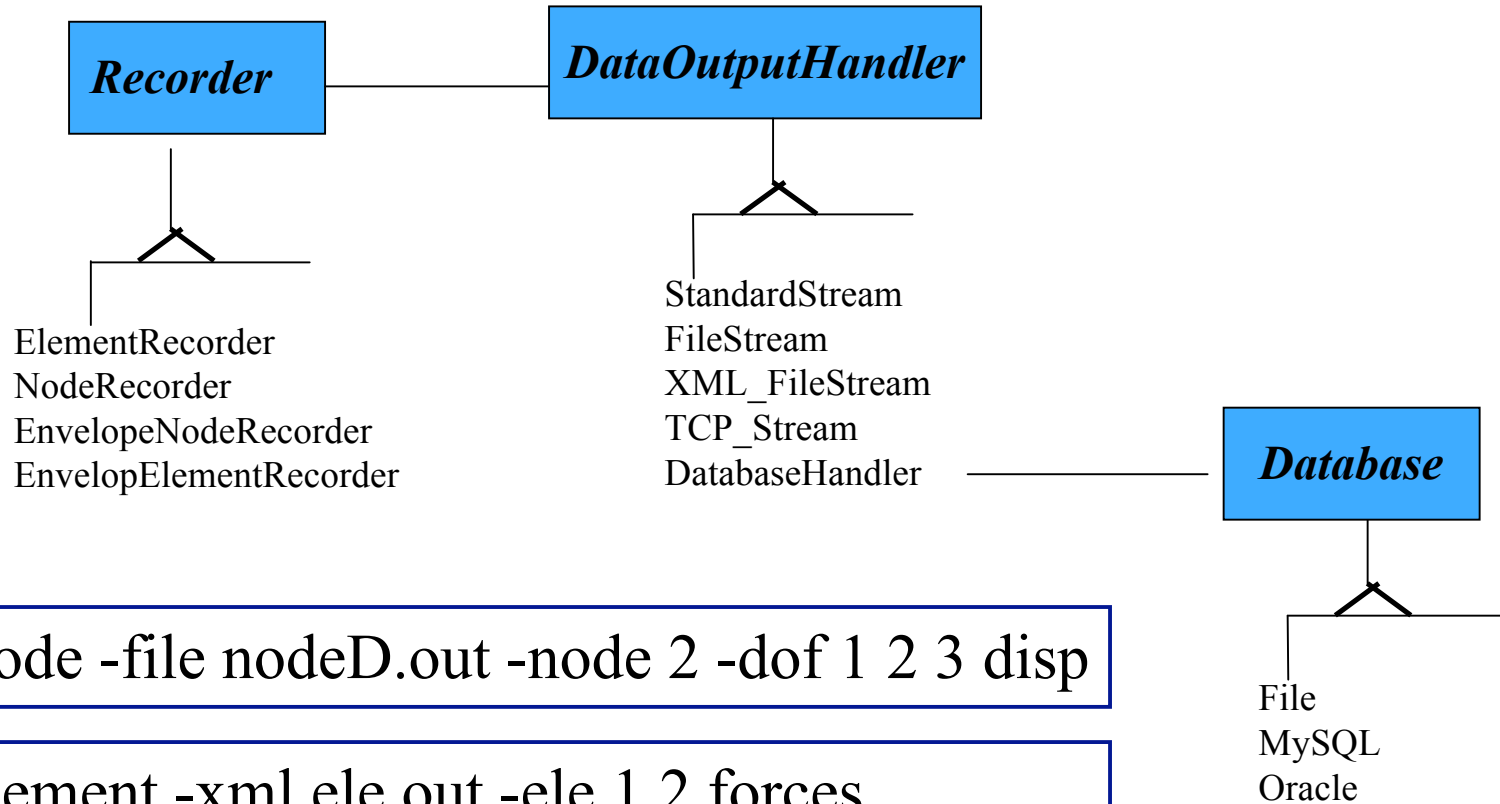
```
constraints Transformation  
numberer RCM  
system BandGeneral  
test NormDispIncr 1.0e-6 6 2  
algorithm Newton  
integrator LoadControl 0.1  
analysis Static  
analyze 10
```

- Transient Nonlinear Analysis with Newmark

```
constraints Transformation  
numberer RCM  
system BandGeneral  
test NormDispIncr 1.0e-6 6 2  
algorithm Newton  
integrator Newmark 0.5 0.25  
analysis Transient  
analyze 2000 0.01
```

# Recorders & Output

(no output unless you request it)



```
recorder Node -file nodeD.out -node 2 -dof 1 2 3 disp
```

```
recorder Element -xml ele.out -ele 1 2 forces
```

```
recorder Element -file ele1sect1fiber1.out -ele 1 2 section 1 fiber 1stress
```

**Recorder commands issued before analyze command**

# Commands that Return Values

---

- analyze command

The analyze command returns 0 if successful.  
It returns a negative number if not

```
set ok [analyze numIter <Δt>]
```

- getTime command

The getTime command returns pseudo time in Domain.

```
set currentTime [getTime]
```

- nodeDisp command

The nodeDisp command returns a nodal displacement.

```
set disp [nodeDisp node dof]
```

# Power of Scripting Language

## – Nonlinear Transient Analysis Example

```
set tFinal 15.0;
constraints Transformation
numberer RCM
system BandGeneral
test NormDispIncr 1.0e-6 6 2
algorithm Newton
integrator Newmark 0.5 0.25
analysis Transient
set ok 0; set currentTime 0.0
set checkFile [open checkFile.txt w]
while {$ok == 0 && $currentTime < $tFinal} {
    set ok [analyze 1 0.01]
    if {$ok != 0} {
        puts $checkFile "Problem $ok at [getTime]"
        test NormDispIncr 1.0e-6 1000 1
        algorithm ModifiedNewton -initial
        set ok [analyze 1 0.01]
        test NormDispIncr 1.0e-6 6 2
        algorithm Newton
    }
    set currentTime [getTime]
}
close $checkFile
```

# OpenSees & Matlab

- Calling matlab from an OpenSees script (mScript.m)

*# invoke matlab*

```
if {[catch{exec matlab -nosplash -nodesktop -r "mScript; quit"}}]{  
  puts "Ignore this $msg"  
}
```

- Calling OpenSees from a matlab script

*# invoke matlab*

```
!OpenSees opsScript.tcl
```

# OpenSees Resources

- Getting Started Manual:  
[http://opensees.berkeley.edu/wiki/index.php/Getting\\_Started](http://opensees.berkeley.edu/wiki/index.php/Getting_Started)
- Command Manual:  
[http://opensees.berkeley.edu/wiki/index.php/Command\\_Manual](http://opensees.berkeley.edu/wiki/index.php/Command_Manual)
- Examples Manuals:  
<http://opensees.berkeley.edu/wiki/index.php/Examples>
- Message Board:  
<http://opensees.berkeley.edu/community/viewforum.php?f=2>
- Discovering OpenSees web-based learning series:  
[http://opensees.berkeley.edu/wiki/index.php/Discovering\\_Open\\_Sees](http://opensees.berkeley.edu/wiki/index.php/Discovering_Open_Sees)



# Downloading OpenSees.exe and Installing Tcl/Tk

- Download OpenSees.exe and tcl/tk from here:

<http://opensees.berkeley.edu/OpenSees/user/download.php>

- Tutorial on installing tcl/tk:

[http://opensees.berkeley.edu/wiki/index.php/Getting\\_Started\\_with\\_OpenSees\\_-\\_Download\\_OpenSees](http://opensees.berkeley.edu/wiki/index.php/Getting_Started_with_OpenSees_-_Download_OpenSees)

# NEES

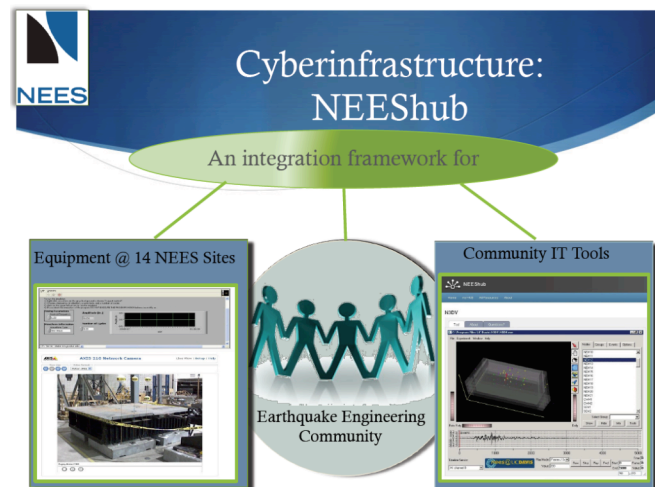
The George E. Brown Network for Earthquake Engineering Simulation (NEES) is a shared national network of 14 experimental facilities, collaborative tools, a centralized data repository, and earthquake simulation software.



# NEEShub



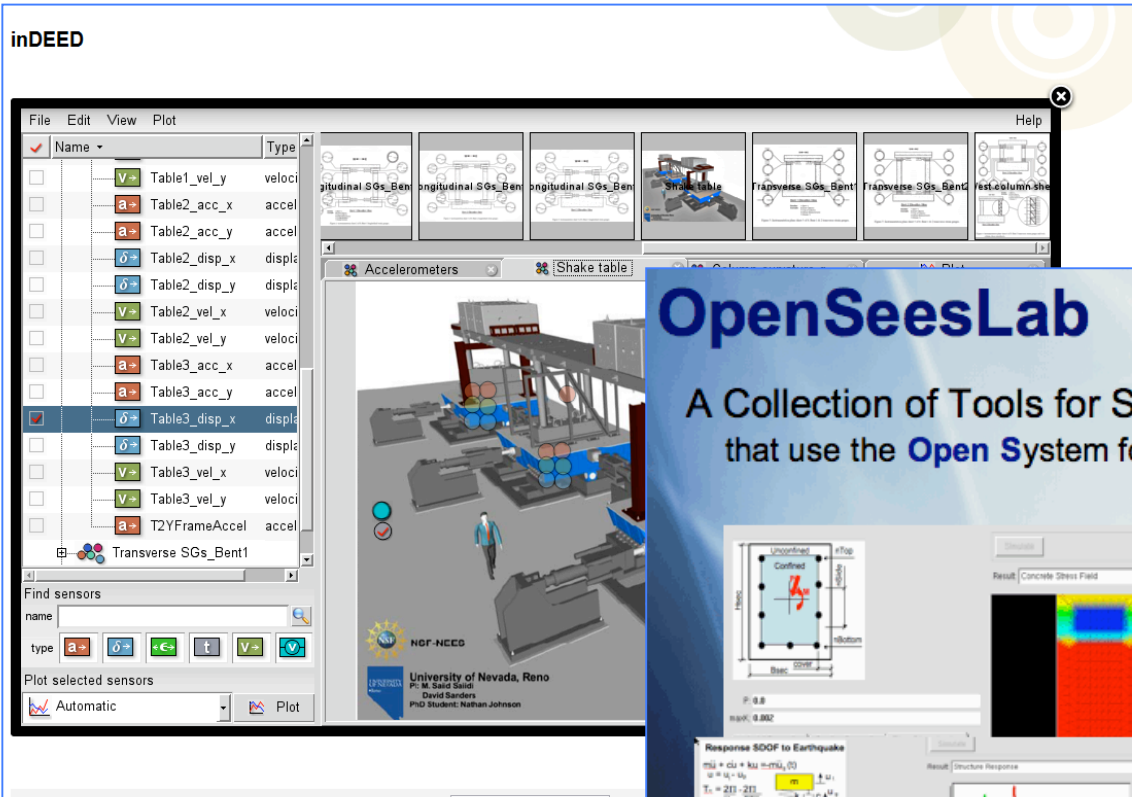
- The power behind NEES at <http://nees.org>
- Maintained and developed at Purdue by NEEScomm
- A science gateway for education and research in earthquake engineering



Through a browser engineers can:

- Upload and view experimental data
- Browse online seminars and courses
- Launch sophisticated tools using remote computational resources (OpenSeesLab)

# NEEShub Tools and Resources



Simulation

## OpenSeesLab

NEEShub

A Collection of Tools for Structural/Geotechnical Engineers that use the Open System for Earthquake Engineering Simulation

The image displays several screenshots from the OpenSeesLab software. One screenshot shows a 'Concrete Stress Field' with a color-coded stress distribution. Another shows a 'Structure Response' plot with multiple colored lines representing different components over time. A third screenshot shows a 'Response SOF to Earthquake' dialog box with various parameters like 'Analysis Type', 'Source Ground Motion', and 'Earthquake Name'. Other screenshots show structural diagrams and data tables.

Data Management

Application:

OpenSeesLaboratory



## OpenSeesLab

NEEShub

### Menu To Select Tool

A Collection of Tools for Structural/Geotechnical Engineers that use the **Open System for Earthquake Engineering Simulation**

**Concrete Stress Field**

Result: Concrete Stress Field

**Pile Bending Moment**

Result: Pile Bending Moment

**Response SDOF to Earthquake**

$m\ddot{u} + c\dot{u} + ku = m\ddot{u}_g(t)$   
 $u = u_x - u_y$   
 $T_x = 2EI - 2EI$   
Damping Ratio:  $\zeta = \frac{c}{2\sqrt{km}}$

Analysis Type: Single Analysis  
Source Ground Motions: PEER NGA  
Model: Earthquake | Spectral  
Earthquake Name: any  
Min Mag of EQ to search: 0.5  
Max Mag of EQ to search: 6  
Soil Type: 0  
Min PGA of records to search: 0.12  
Max PGA of records to search: 0.14

Structure Response

Displacement (m)

Time

OpenSees 2.10.0  
Nodes 2 size  
Nodes 3 source Elem(1,1,1)

Node 1  
Coordinates: 1 0 0 0  
Elev: -1.4800e+01 -6.0000e 7.668e-05  
Sublevel Level: 1 100 m  
E 1 1 1 1

Node 4  
Coordinates: 1 30 0 0  
Elev: -1.4800e+01 -6.0000e 9.701e-05  
Sublevel Level: 1 100 m  
E 1 1 1 1

Element 1 Type ForceBeamElems2 Connected Nodes: 1 2  
Number of Sections: 5 Res: default 0  
Labels:  
End 1 Force (P V K1) 100 -4.5730e-02 2.1703e-14  
End 2 Force (P V K1) 100 6.5730e-02 -1.1203e-14

Element 2 Type ForceBeamElems2 Connected Nodes: 2 4  
Number of Sections: 5 Res: default 0  
Labels:  
End 1 Force (P V K1) 100 -1.5730e-02 2.1703e-14  
End 2 Force (P V K1) 100 1.5730e-02 -1.1203e-14

ElementElems2 1  
Connected Nodes: 1 4  
CoordTransf: 0  
non default: 0  
End 1 Force (P V K1) 2.7020e+02 2.4247e-17 4.4800e-05  
End 2 Force (P V K1) 2.7020e+02 -2.4247e-17 4.4800e-05

# OpenSees Interpreter Tool

```
OpenSees Laboratory [X] Terminate [▶] Keep for later
OpenSees
Application: OpenSees Interpreter
OpenSees -- Open System For Earthquake Engineering Simulation
Pacific Earthquake Engineering Research Center -- 2.4.0

(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 > cp /home/neeshub/fmk/Examples.tar ./
OpenSees > tar xBF Examples.tar
OpenSees > cd Examples
OpenSees > source Example1.tcl

Node: 4
Coordinates : 72 96
Disps: 0.530093 -0.177894
Velocities : 0 0
unbalanced Load: 100 -50
ID : 0 1

Element: 1 type: Truss iNode: 1 jNode: 4 Area: 10 Mass/Length: 0
strain: 0.00146451 axial load: 43.9352
unbalanced load: -26.3611 -35.1482 26.3611 35.1482
Material: Elastic tag: 1
E: 3000 eta: 0

Element: 2 type: Truss iNode: 2 jNode: 4 Area: 5 Mass/Length: 0
strain: -0.00383642 axial load: -57.5463
unbalanced load: -34.5278 46.0371 34.5278 -46.0371
Material: Elastic tag: 1
E: 3000 eta: 0

Element: 3 type: Truss iNode: 3 jNode: 4 Area: 5 Mass/Length: 0
```

*Example We Will Do*

# Advantages of Running on NEEShub

- Can check status from any network device
- Can have multiple simulations running

The screenshot displays the NEEShub user interface with several panels:

- My Sessions:** Lists three sessions for 'OpenSees Laboratory' at 5:36 pm. A storage bar shows 22% of 1GB used.
- My Projects:** Shows 'No Projects Found' with an 'All My Projects' button.
- My Tools:** Includes tabs for 'Recent', 'Favorites', and 'All Tools'. It lists 'OpenSees Laboratory' as a recently used tool and provides links for 'Citations', 'Learning Objects', 'Publications', 'Tools', and 'Multimedia'.
- My Contributions:** States 'No contributions found' with a 'Start a new contribution' button.
- My Groups:** States 'You are not a member of any groups at this time.' with buttons for 'All My Groups', 'All Groups', and 'New Group'.

- Can Share Session with someone

The screenshot shows an OpenSees simulation window titled 'OpenSees Laboratory'. The main area displays the OpenSees application window with a terminal showing simulation output. A sharing overlay is visible at the bottom, highlighted with a red box. The overlay includes:

- A 'Share session with:' field with a placeholder '(supports usernames, user IDs, and e-mails)'. Below it is a checkbox for 'Read-Only? (only you control the session)'.
- A 'This session is shared with:' field showing '(none)'.
- A 'Share' button.

***FOR THOSE OF YOU WHO ARE NEW***

OpenSees does require time &  
effort to learn

**BUT**

**Eventually**

**Students Learn to Love  
OpenSees**



## ***And Why OpenSees?***

Access to modern FE theory

Access to modern numerical & IT advances

Reflects modern programming methods

Access to the source code

Faster

Scripting is a more powerful interface!

demo USING NEEShub

QUESTIONS?