

Getting Started: An Introduction to OpenSees and Tcl

Frank McKenna

OpenSees Users Workshop
Berkeley, CA
August 14, 2006

Sponsored by the National Science Foundation
through the Pacific Earthquake Engineering Research Center
and through NEESit

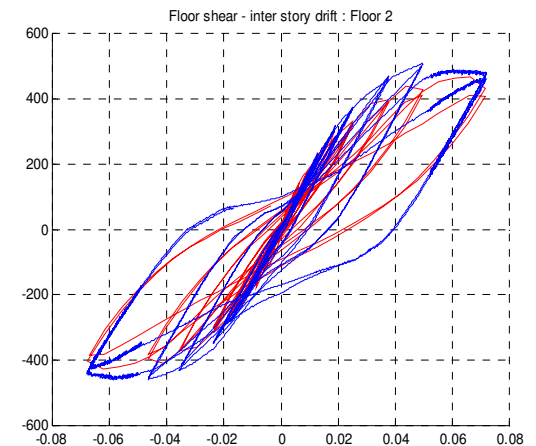
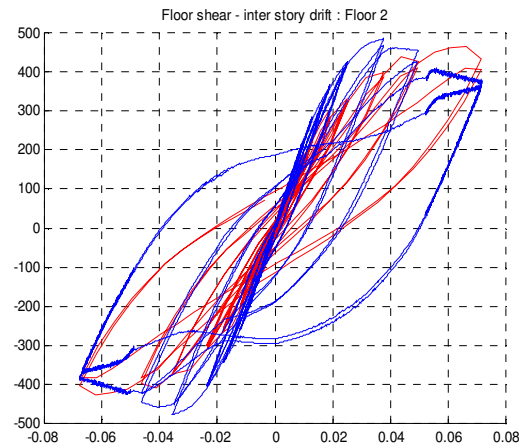
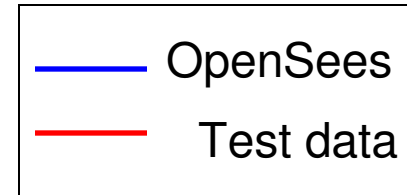


What is OpenSees

- OpenSees is an Open-Source Software Framework for developing nonlinear Finite Element Applications for both sequential and parallel environments.
- OpenSees.exe is an extension of the Tcl interpreter for finite element analysis which uses this framework.

And Why do Finite Element Analysis

NCEER frame tested at the Taiwan facility

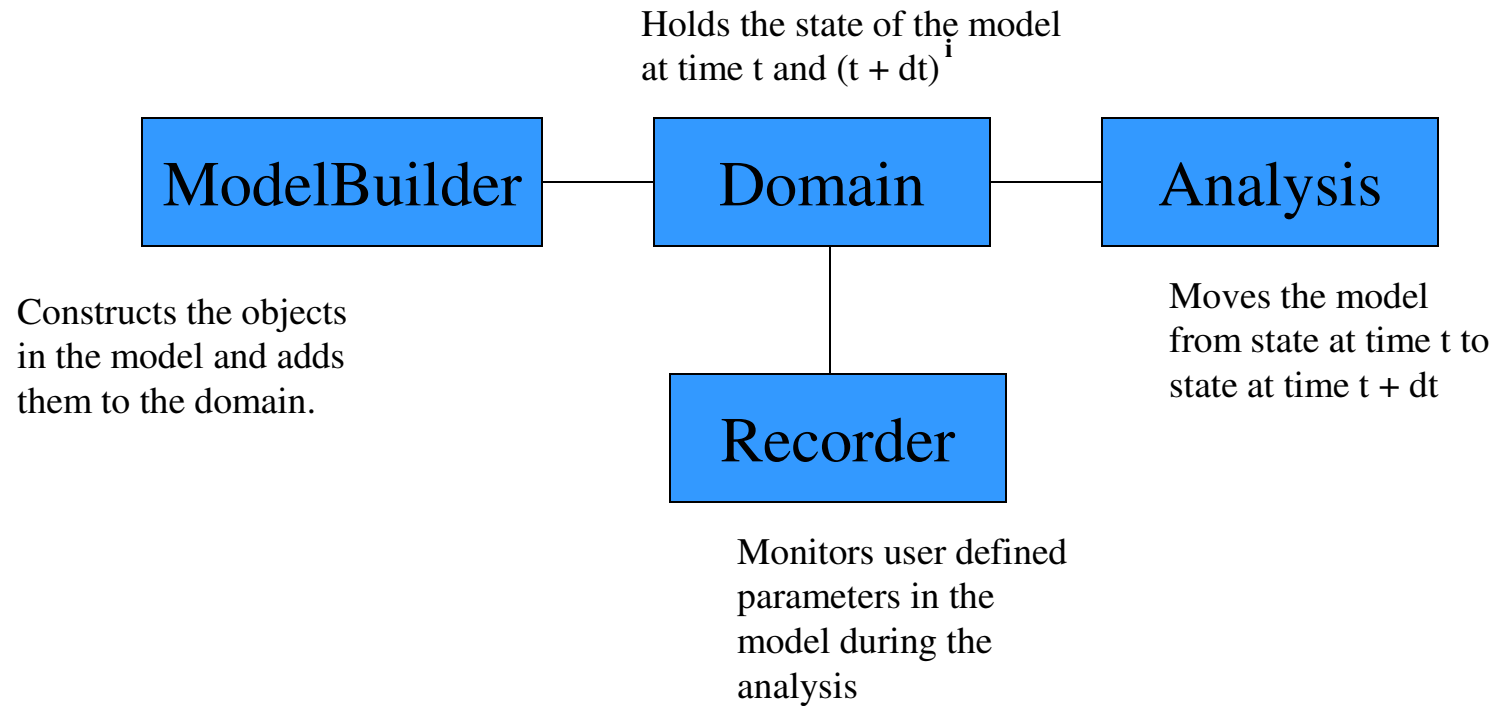


Centerline model and model with joint comparison

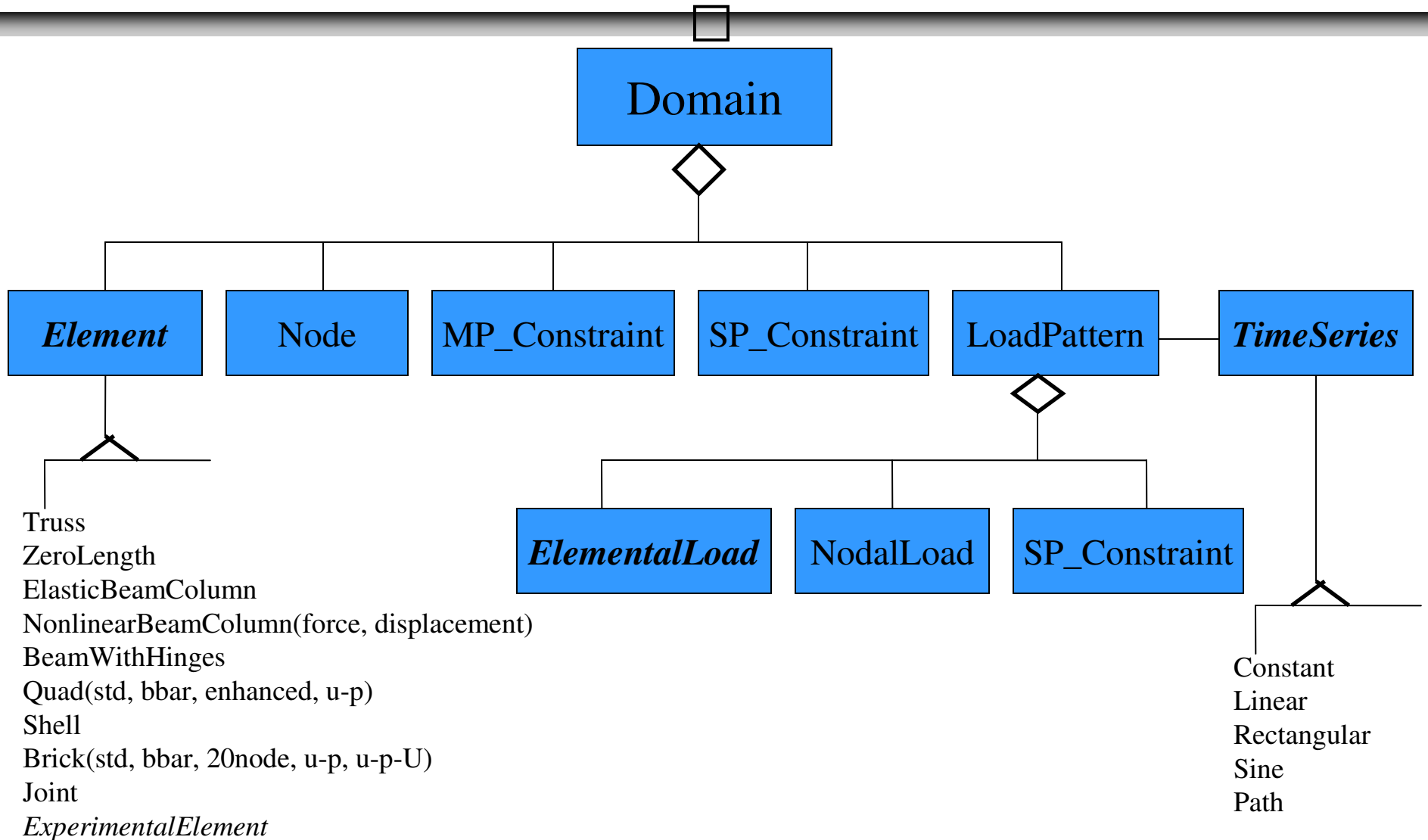
Getting OpenSees

- Web site: <http://opensees.berkeley.edu/>
- User Pages
 - Download Center
 - Documentation
 - Browse the Source Code
 - Message Board
 - Bug Reporting!

Main Abstractions in OpenSees

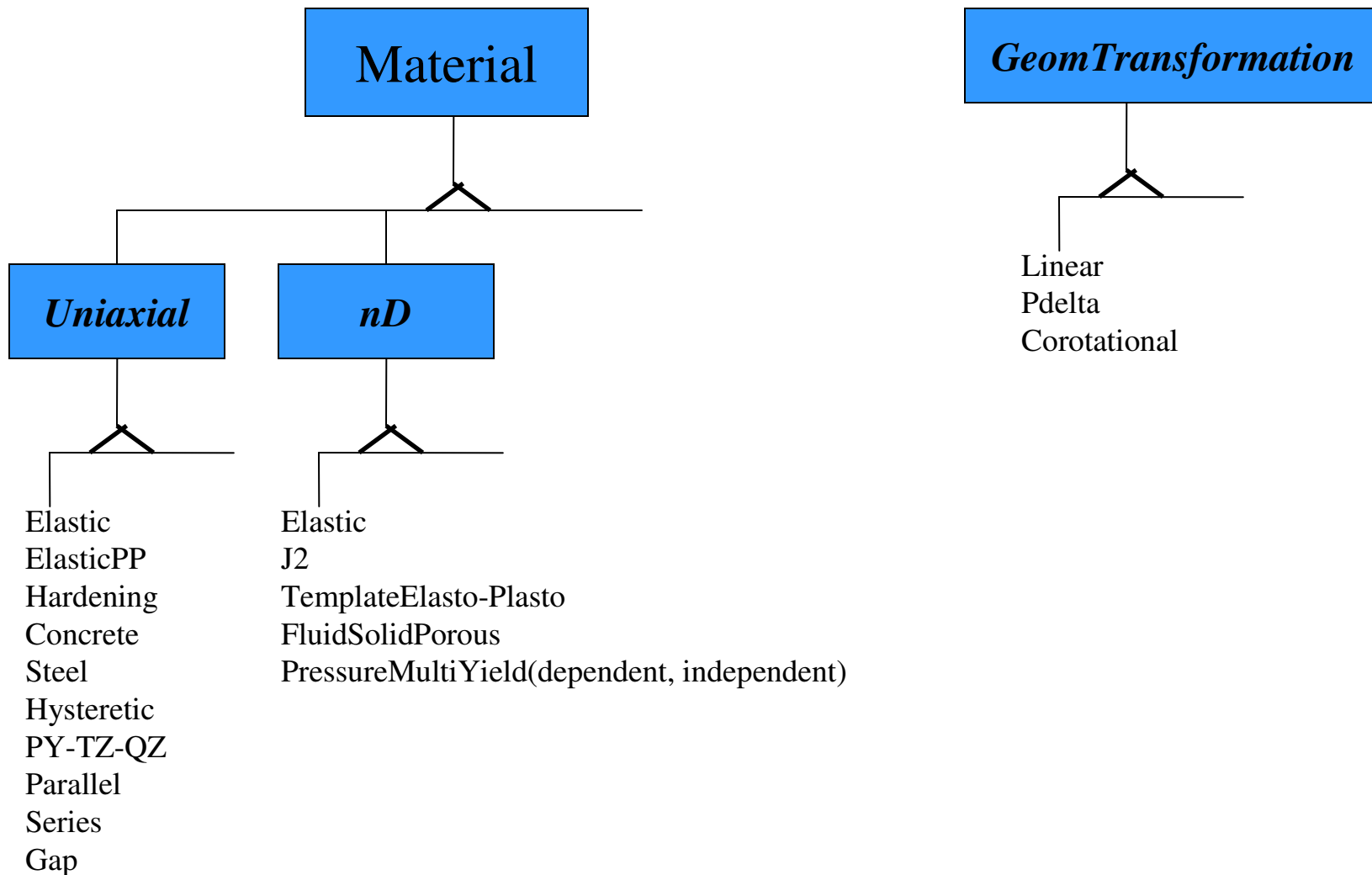


What is in a Domain

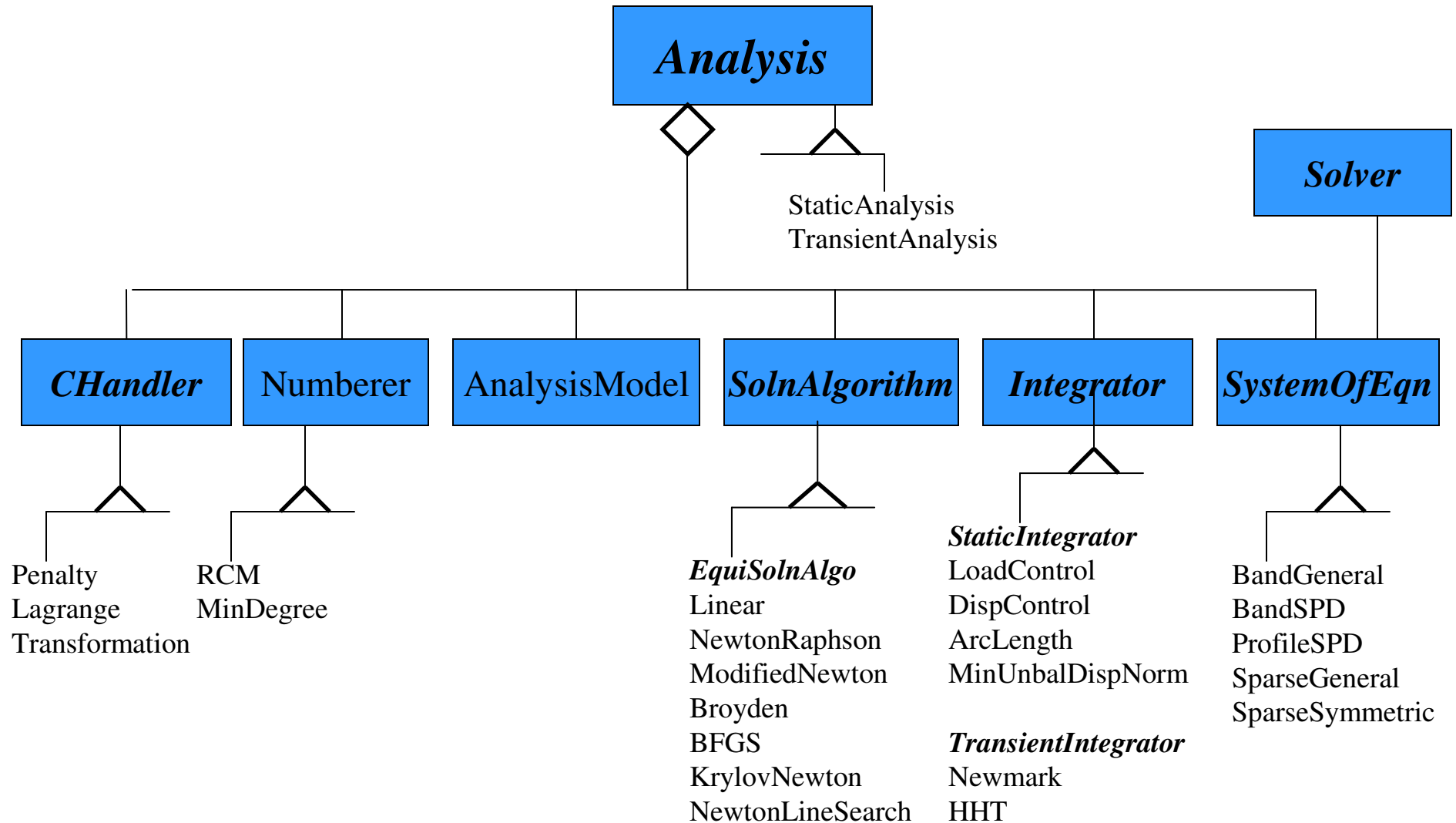


* It is the model builder which constructs these objects.

Some Others Classes associated with Elements:



What is an Analysis




What is Tcl

- Tcl is a string based scripting language.
- Variables and variable substitution
- Expression evaluation
- Basic control structures (if , while, for, foreach)
- Procedures
- File manipulation
- Sourcing other files.

Tcl

- Tcl scripts are made up of commands separated by newlines or ;
- Command syntax:

command arg1 arg2 ...

- Help
 1. <http://dev.scriptics.com/scripting/primer.html>
 2. Practical Programming in Tcl and Tk, Brent B. Welch, Prentice Hall.
- Let's demonstrate using 

Example Tcl:

```
>set a 1
>1
>set b a
>a
>set b $a
>1
>expr 2 + 3
>5
>expr 2 + $a
>3
>set b [expr 2 + $a]
>3
```

```
>proc sum{a b} {
    return [expr $a + $b]
}
>sum 2 3
>5
>set c [sum 2 3]
>5
```

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

```
>source Example1.tcl
```

```
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]
}
set $sum
>10
>proc guess {value} {
    global sum
    if {$value < $sum} {
        puts "too low"
    } else {
        if {$value > $sum} {
            puts "too high"
        } else { puts "you got it!"}
    }
}
> guess 9
too low
>
```

Commands to Tcl for OpenSees

- For OpenSees we have added commands to Tcl 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.

model generation:

*Adds the modelling commands to the interpreter.

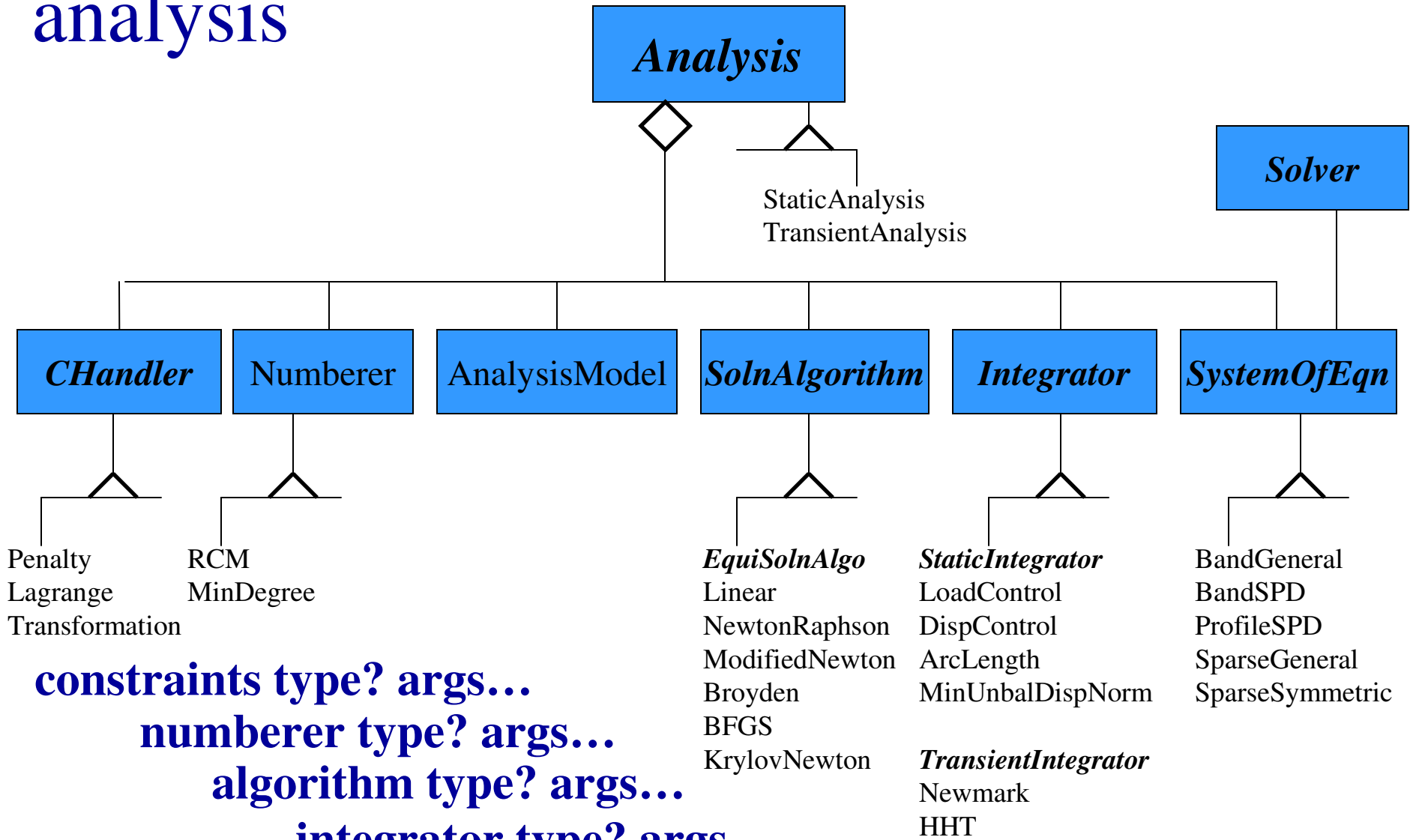
- BasicBuilder

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

This command now adds the following commands to the interpreter:

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

analysis



constraints type? args...

numberer type? args...

algorithm type? args...

integrator type? args...

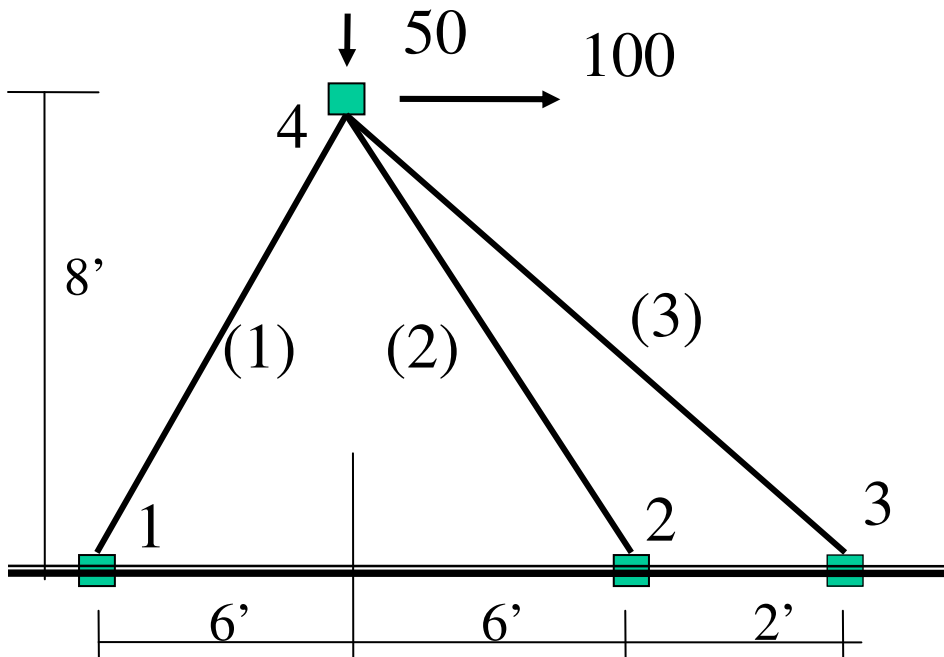
system type? args...

analysis type? args...

analyze args ...



Example Model:



	E	A
1	3000	10
2	3000	5
3	3000	5

```

model Basic -ndm -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
Pattern Plain 1 "Linear" {
  load 4 100.0 -50.0
}
    
```

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
```

