OpenSees Days Bristol, June 25-26 2014

Conducting Hybrid Simulations with OpenSees/OpenFresco

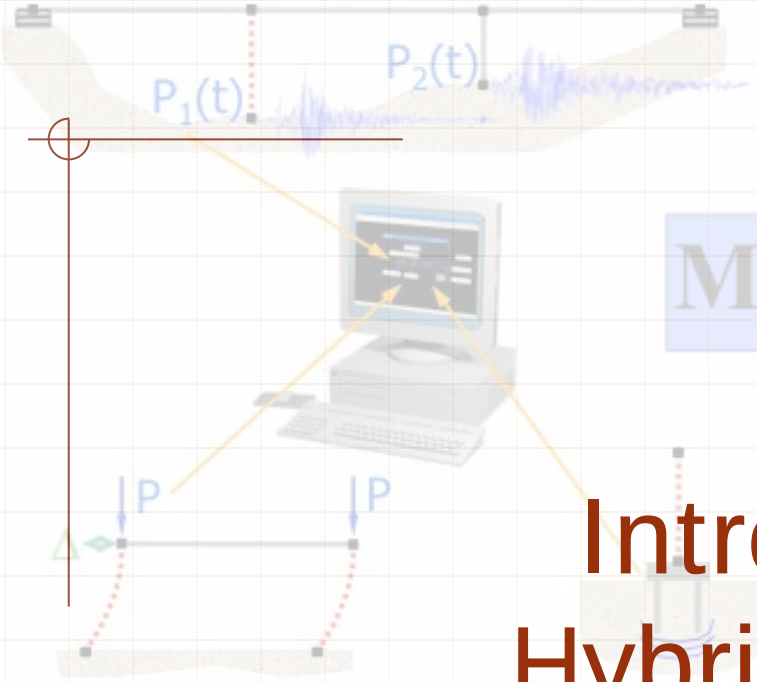
Andreas Schellenberg

Pacific Earthquake Engineering Research Center, University of California, Berkeley

OpenFresco

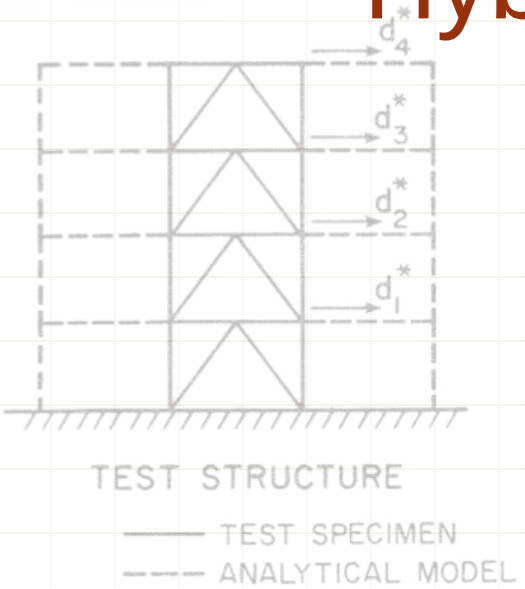
Outline of Presentation

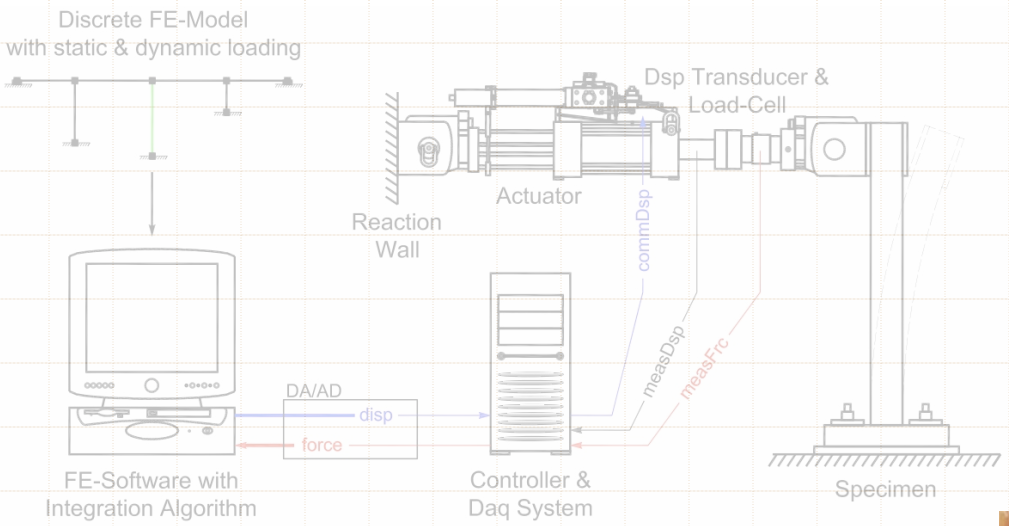
1. Introduction to Hybrid Simulation
2. OpenFresco Architecture and TCL commands
3. Downloading and Installing OpenFresco
4. Building a Hybrid Model in OpenSees/OpenFresco
5. Simulated vs. Real Controllers
6. Using other Computational Drivers
7. Summary & Conclusions



$$\mathbf{M}\cdot\ddot{\mathbf{u}}+\mathbf{C}\cdot\dot{\mathbf{u}}+\mathbf{P}_{\mathbf{r}}(\mathbf{u})=\mathbf{P}(t)$$

Introduction to Hybrid Simulation





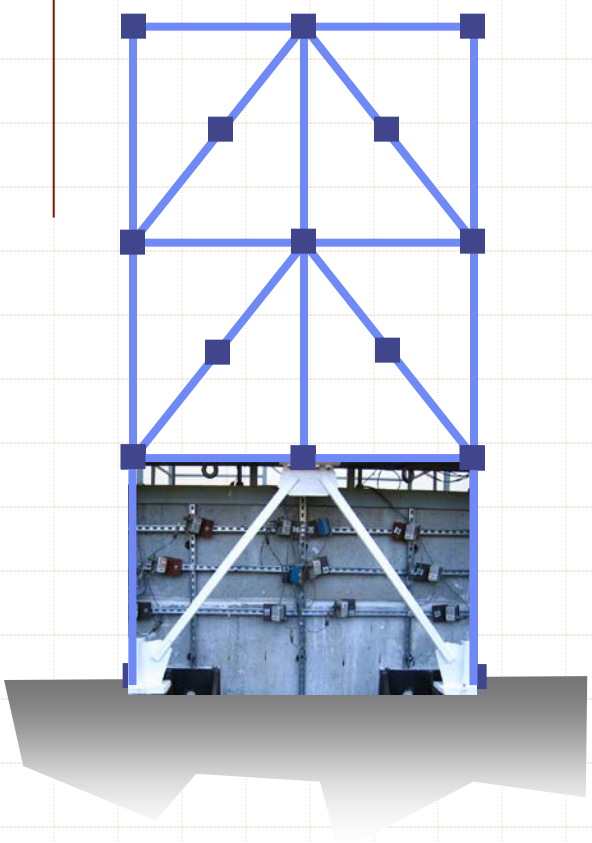
Comparison of Exp. Test Methods

	Quasi-Static	Shaking Table	Hybrid Simulation
Dynamics	NO	YES	YES
Strain Rate Effects	NO	YES	YES (if real-time test)
Large- or Full-Scale	YES	NO (limited by table)	YES

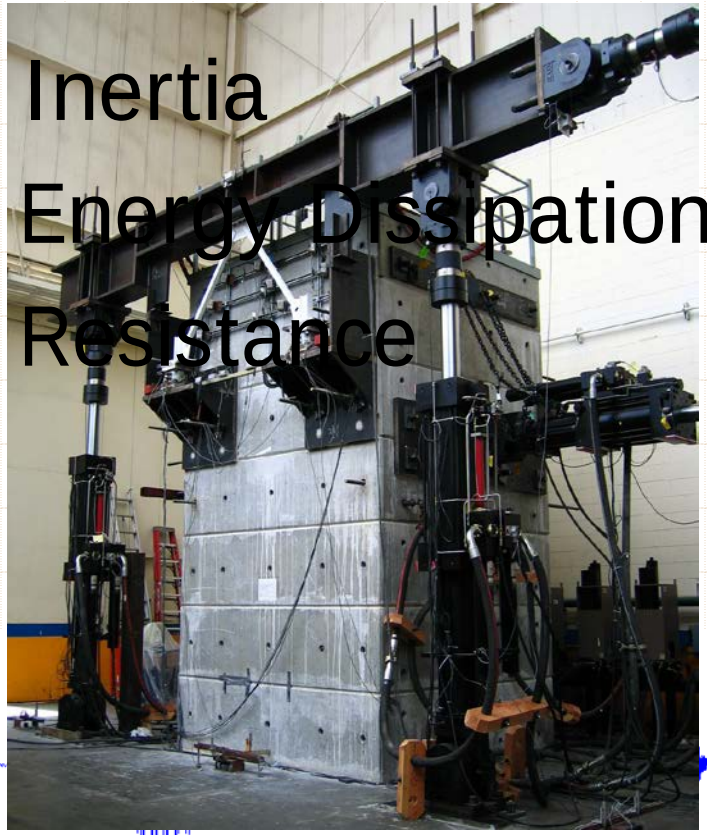
- First Hybrid Simulation (Online Test) in 1975 by Takanashi et al.

Hybrid Simulation

$$\mathbf{M}\cdot\ddot{\mathbf{u}}+\mathbf{C}\cdot\dot{\mathbf{u}}+\mathbf{P}_{\mathbf{r}}(\mathbf{u})=\mathbf{P}(t)$$

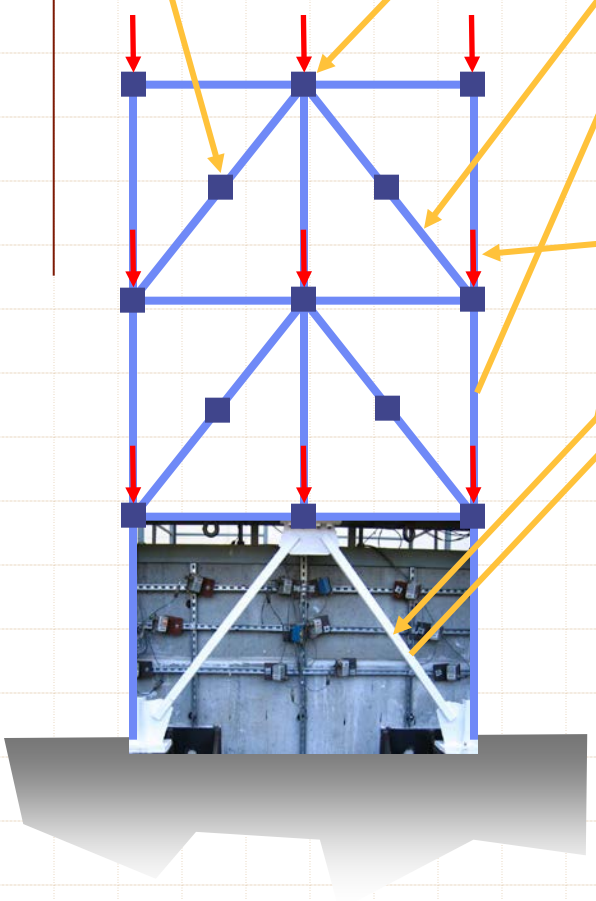


- Inertia
- Energy Dissipation
- Resistance



Hybrid Simulation

$$\mathbf{M}\cdot\ddot{\mathbf{u}}+\mathbf{C}\cdot\dot{\mathbf{u}}+\mathbf{P_r}(\mathbf{u})=\mathbf{P}(t)$$



Dynamic Loading:

- Seismic
- Wind
- Blast/Impact
- Wave
- Traffic

Static Loading:

- Gravity
- Prestress

analytically add nonlinear geometric effects to measured resisting forces

analytical model of structural energy dissipation and inertia

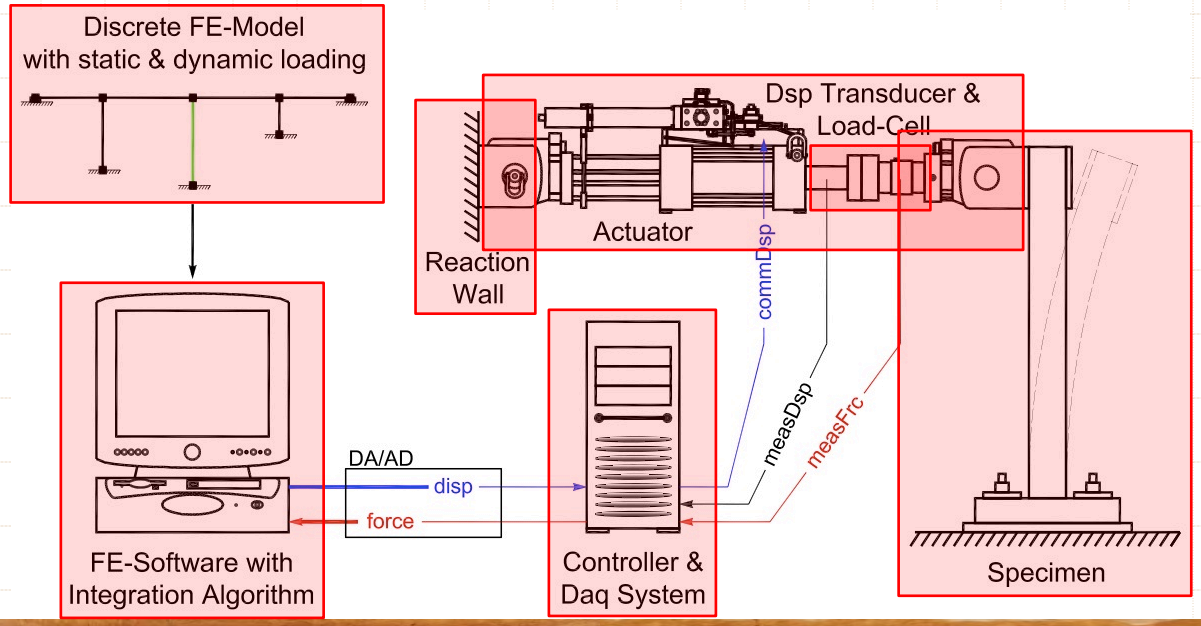
physical model of structural resistance

Hybrid Simulation

- Model the well understood parts of a structure in a finite element program on one or more computers
- Leave the construction and testing of the highly nonlinear and/or numerically hard to model parts of the structure in one or more laboratories
- Can be considered as a conventional finite element analysis where physical models of some portions of the structure are embedded in the numerical model

Required Components

1. Discrete model of the structure to be analyzed, including the static and dynamic loading
2. Servo-hydraulic control system with static or dynamic actuators
3. Physical test specimen, including a reaction-frame
4. Data acquisition system with instrumentation



Testing Methods

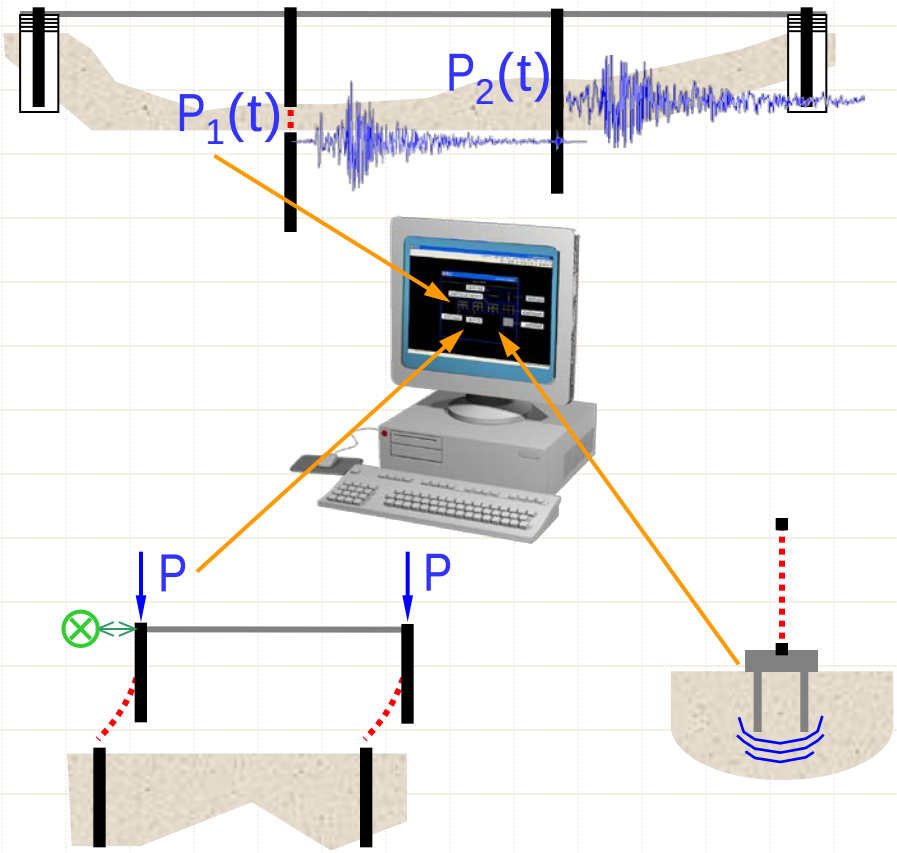
- Conventional hybrid simulation test where specimen is loaded using a ramp-and-hold loading procedure
- Continuous test where specimen is loaded at a continuous slow to moderately slow rate to avoid load relaxations
- Real-time test where specimen is loaded at correct velocities to account for rate-dependent material behaviors
- Geographically distributed network test

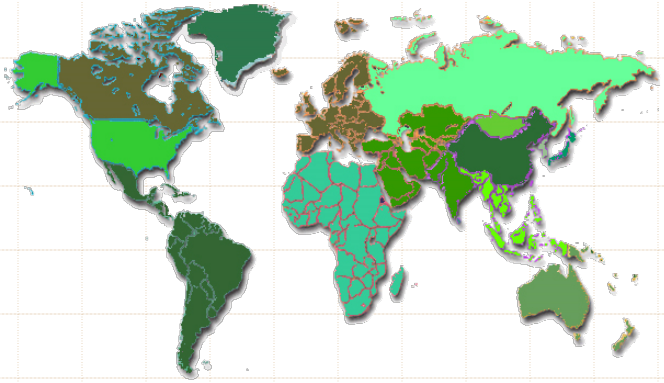
Advantages

- Enables dynamic testing of full-scale specimens
- Quasi-static testing equipment sufficient
- Fewer restrictions on size, weight and strength of a specimen

Advantages

- Geometric nonlinearities, three-dimensional effects, multi-support excitations and soil-structure interactions can be incorporated into the analytical model

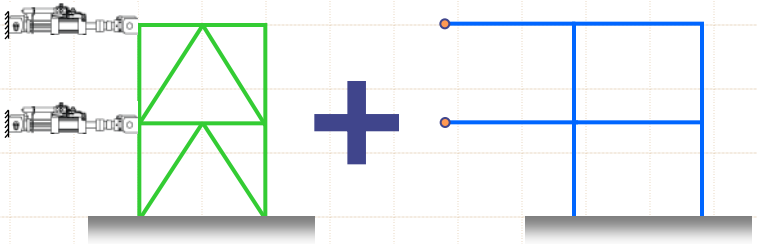




- Internationally, geographically distributed testing is made possible

Advantages

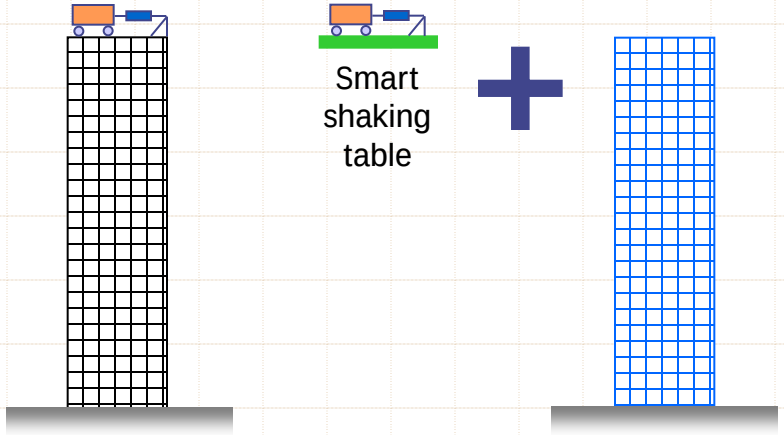
- Quasi-static applications



Drifts may concentrate in one story

Numerical model of portions of system that may alleviate weak story response

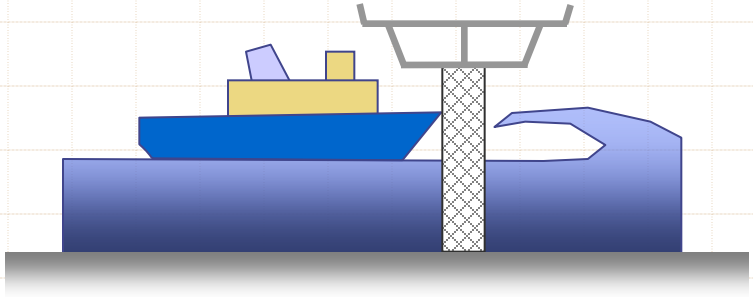
- Dynamic Applications



Shaking Table

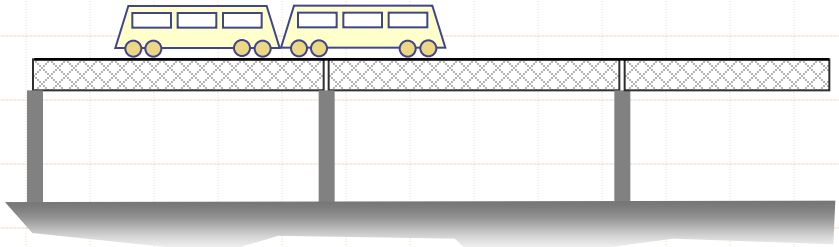
Numerical Model

- Hydrodynamic effects

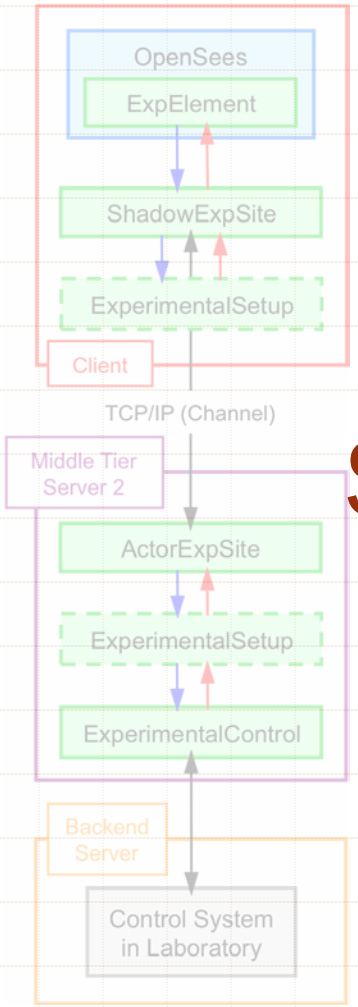


Including impact

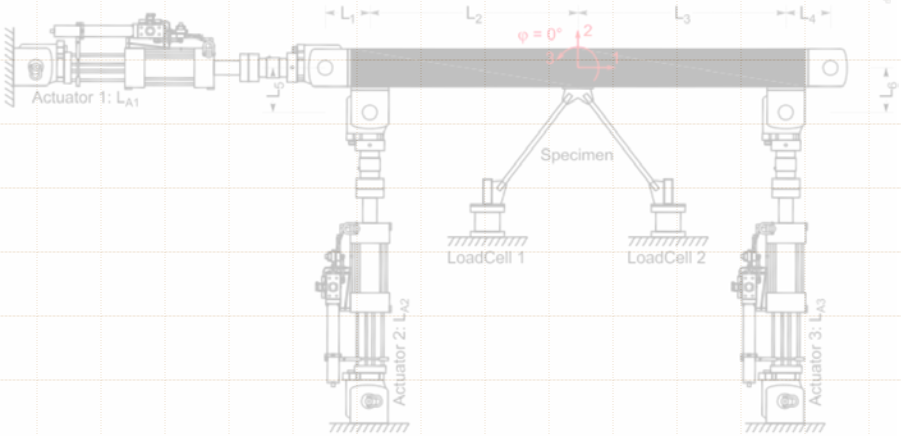
- Moving Loads



Interaction with moving loads, including impact



OpenFresco Software Framework



What is OpenFresco?

- Open source Framework for Experimental Setup and Control
- Secure, object oriented, network enabled "middleware" -- Pairs computer analysis software with laboratory control systems and other software to enable hybrid and collaborative computing:

- Computational Drivers
  - OpenSees
  - OpenFresco*Express*
  - Abaqus
  - LS-DYNA
  - Matlab
  - Simulink
  - Ansys
  - UI-SimCor

- Control Systems
  - dSpace
  - MTS
    - STS family
    - Flextest/CSI
    - Flextest/SCRAMNet
  - National Instruments
  - Pacific Instruments
  - ADwin

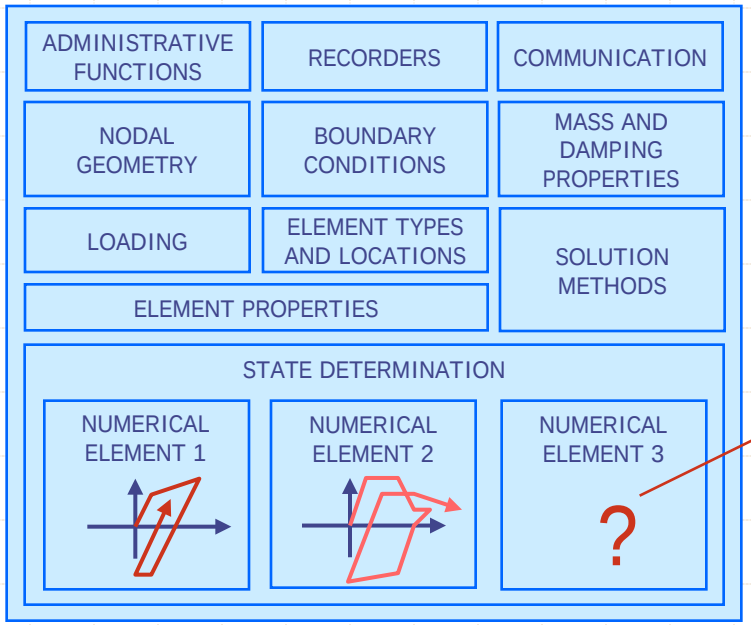
Why a Software Framework?

- Lack of a common framework for development and deployment of HS
- Problem specific implementations which are site and control system dependant
- Such highly customized software implementations are difficult to adapt to different structural problems

→ Need a robust, transparent, adaptable, and easily extensible software framework for research and deployment

Rethinking implementation strategies

- Embed test specimen(s) in an existing computational framework of user's choice

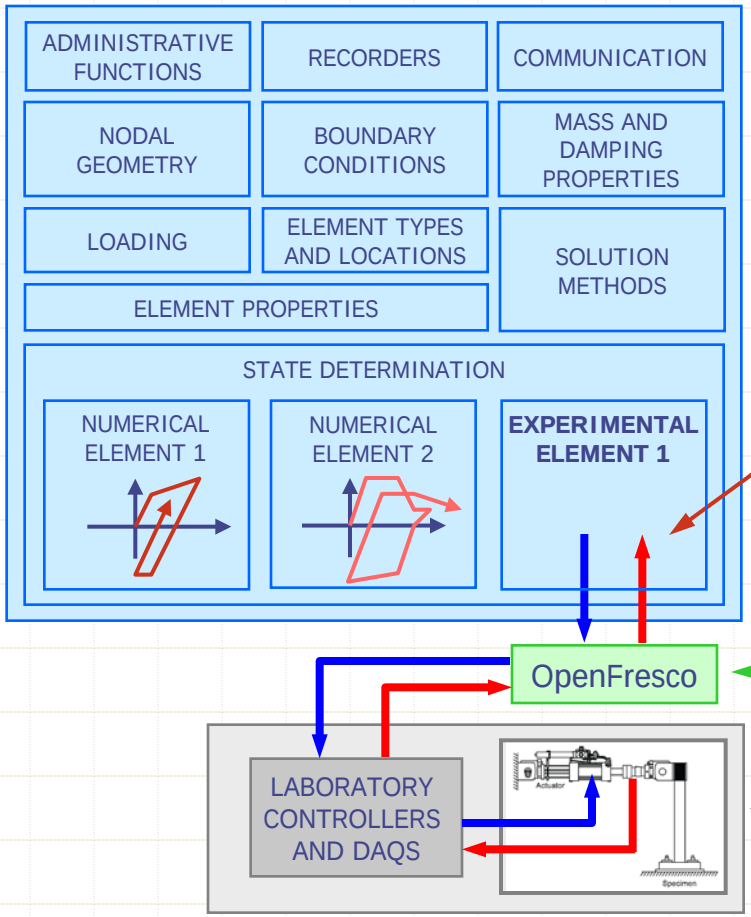


Typical features of an analysis framework

Proper numerical model uncertain

Rethinking implementation strategies

- Embed test specimen(s) in an existing computational framework of user's choice



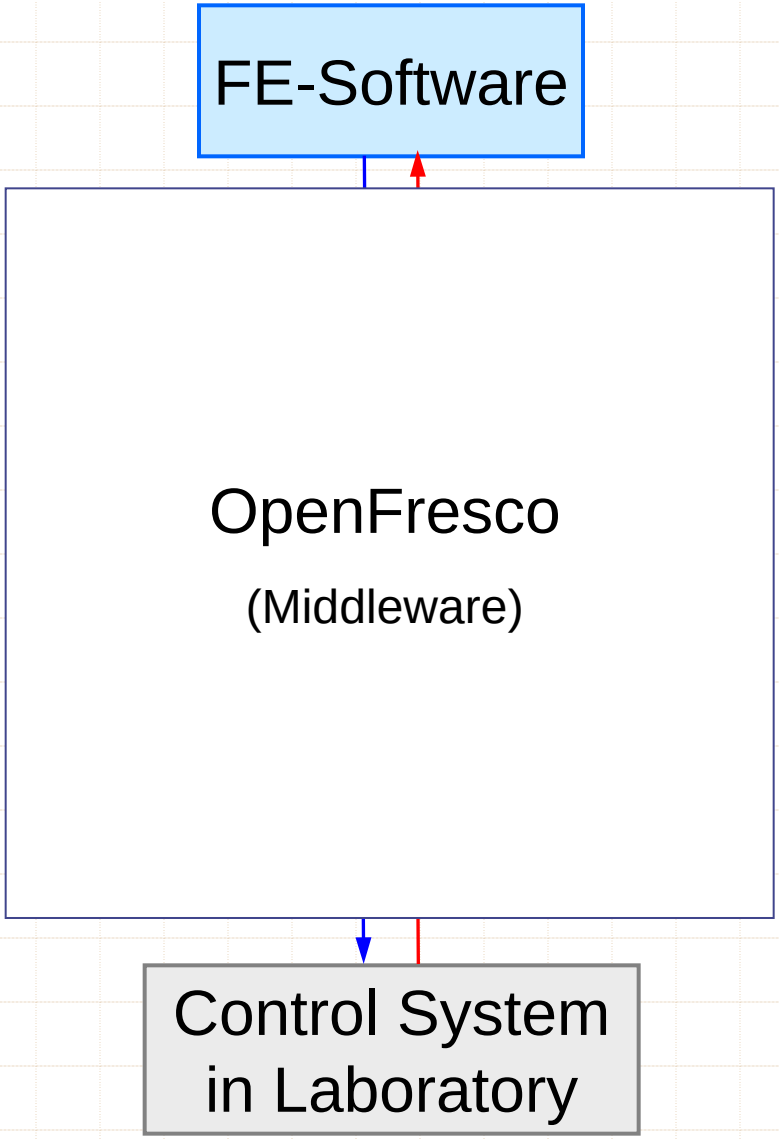
Typical features of an analysis framework

Define element as an "Experimental Element"

OpenFresco

Laboratory

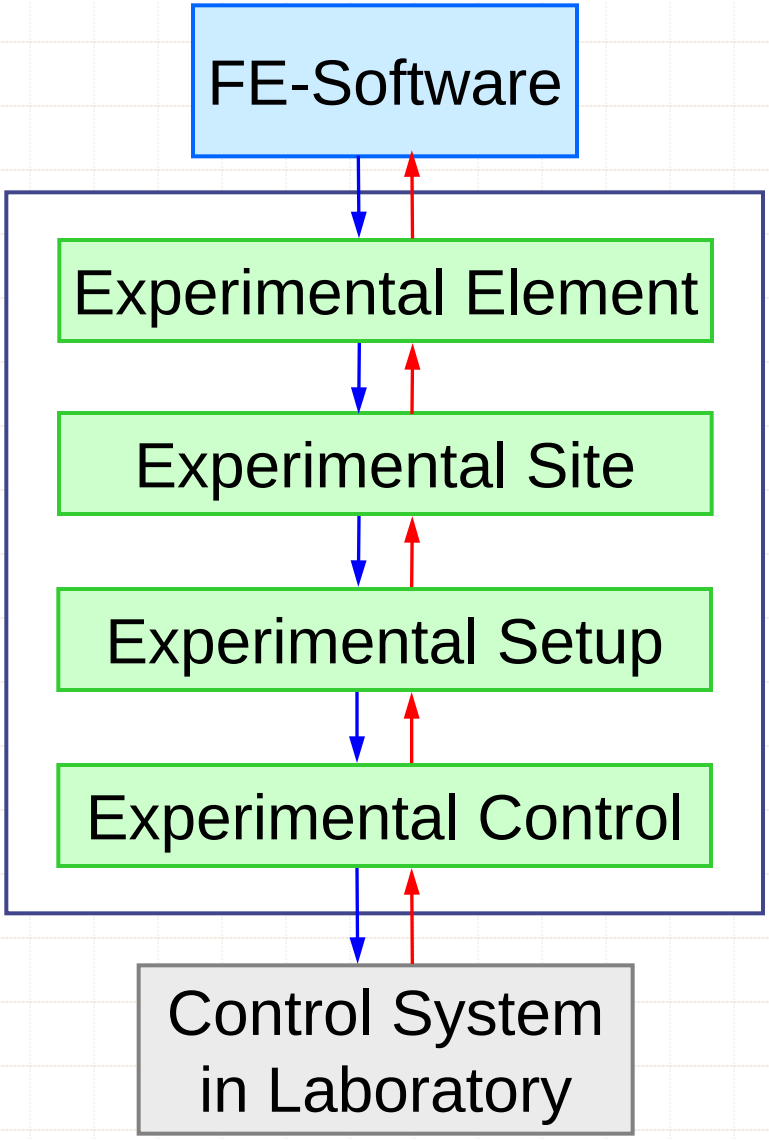
OpenFresco Components



provides all features of unmodified computational framework, including parallel and network computing

provides control of physical actuators as well as data acquisition using physical instrumentation devices

OpenFresco Components



provides all features of unmodified computational framework, including parallel and network computing

represents the part of the structure that is physically tested and provides the interface between the FE-software and the experimental software framework

stores data and provides communication methods for distributed testing

transforms between the experimental element degrees of freedom and the actuator degrees of freedom (linear or non-linear transformations)

interfaces to the different control and data acquisition systems in the laboratories

provides control of physical actuators as well as data acquisition using physical instrumentation devices

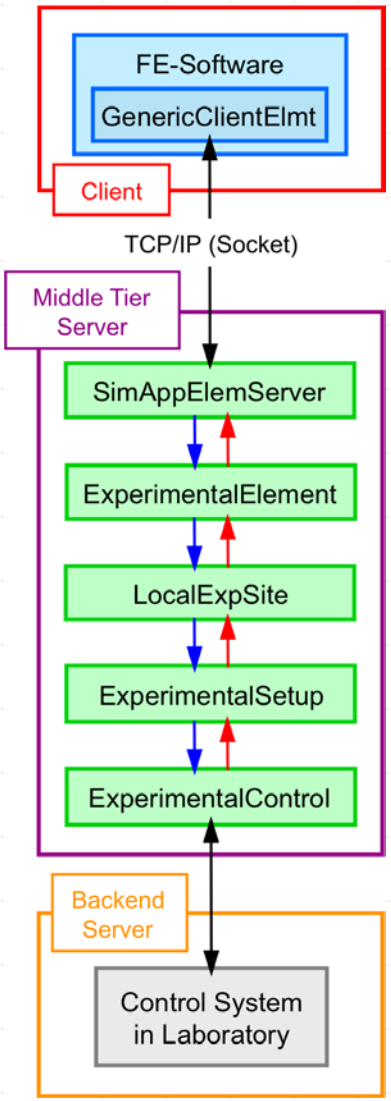
Requirements for Architecture

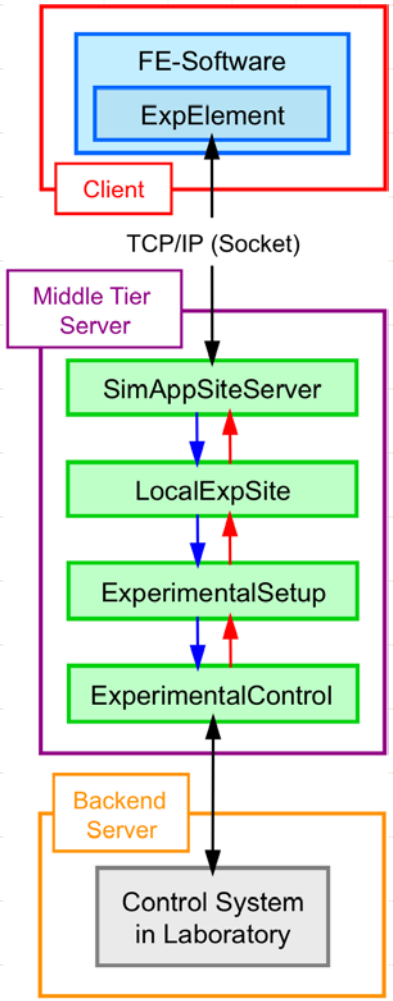
- Provide connectivity to a wide variety of FE-software (clients), independent of the language, such analysis software is programmed in
- Enable distributed testing and support different communication protocols
- Interface with rapidly evolving control and data acquisition systems deployed at testing facilities all over the world

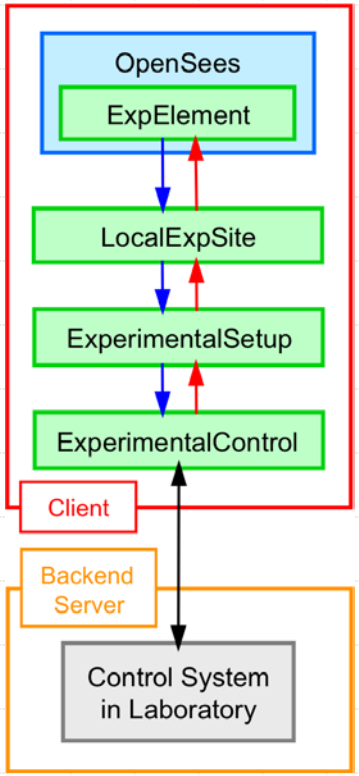
→ Multi/Three-Tier Software Architecture

OpenFresco Components

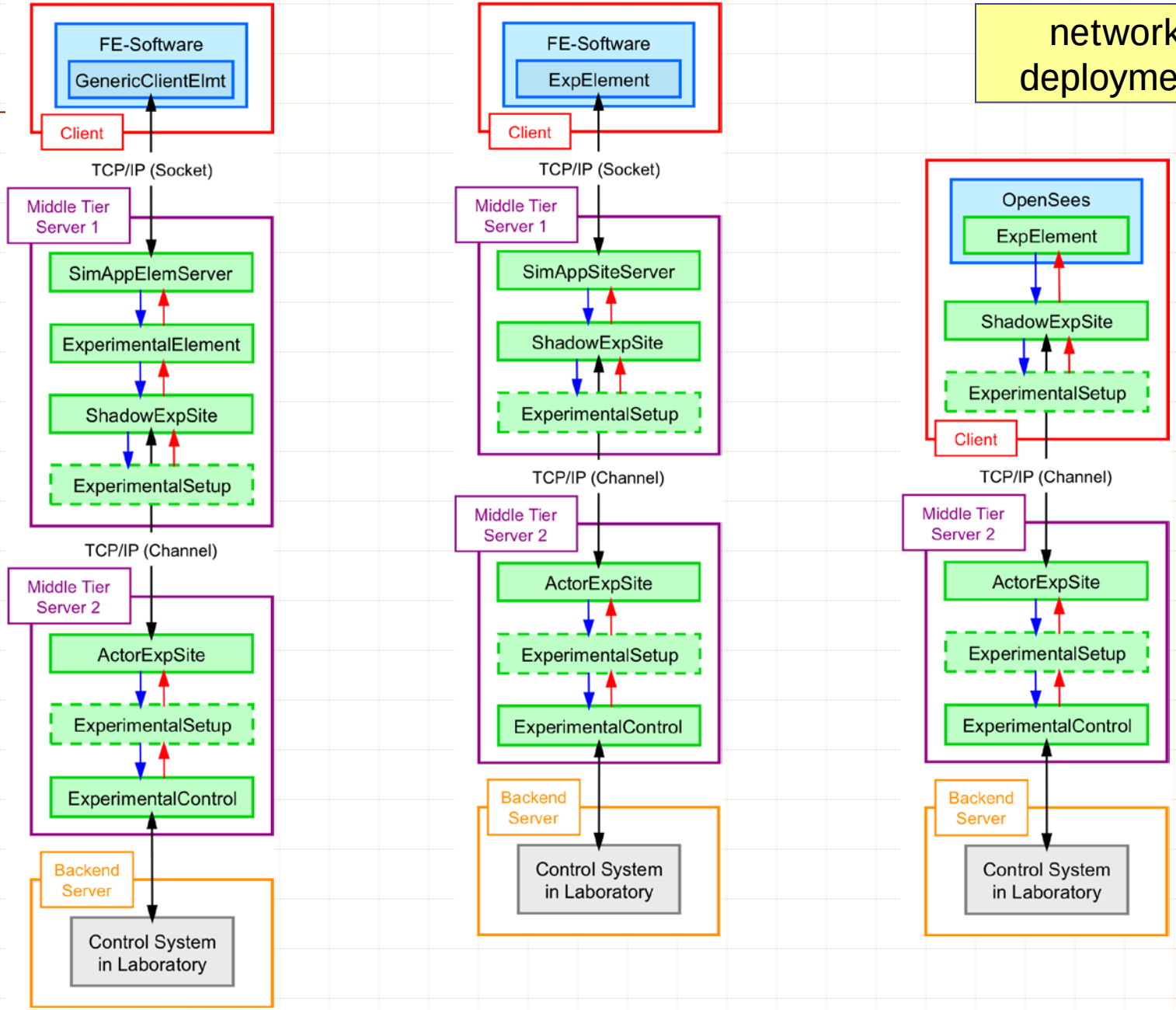
local deployment



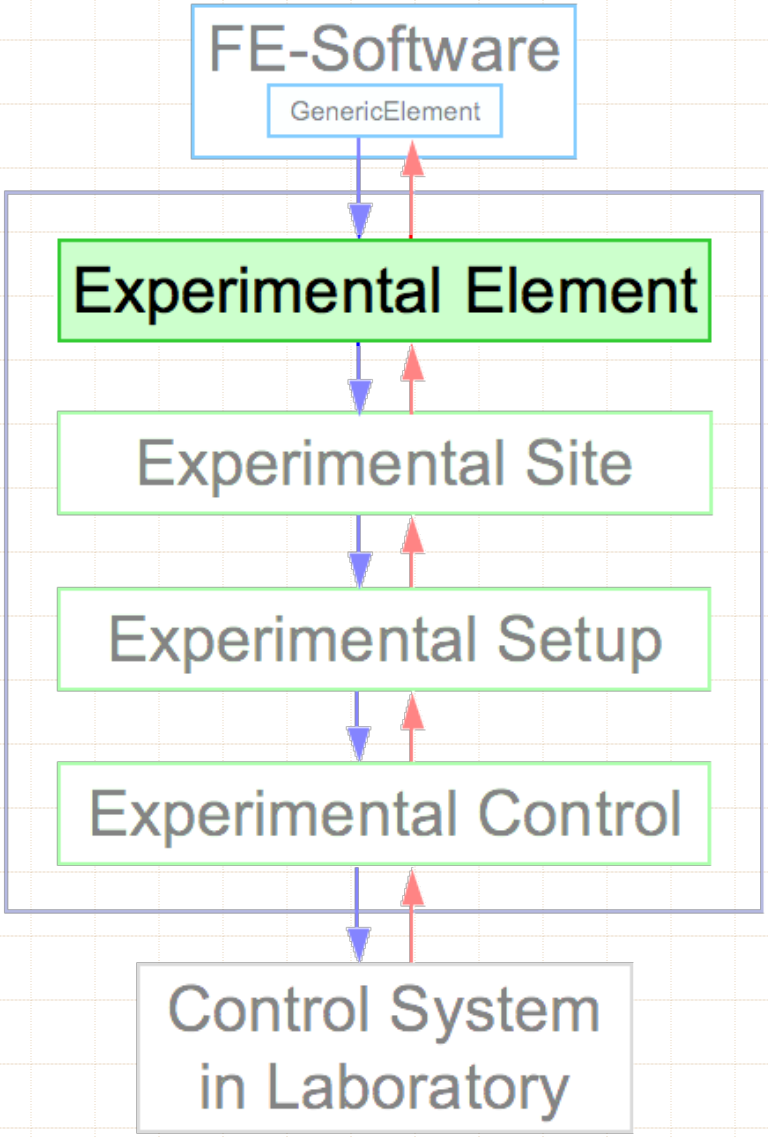




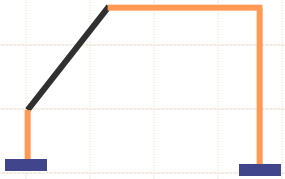
network deployment



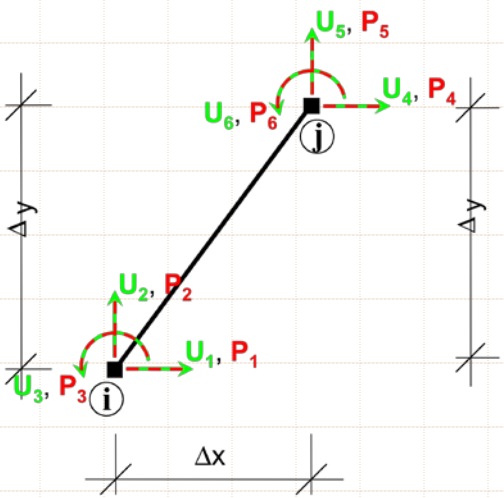
OpenFresco Components



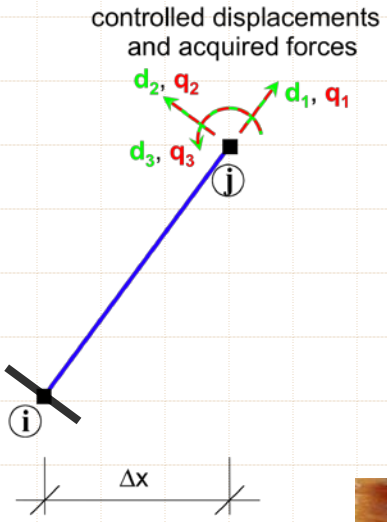
Experimental Element
Transforms between the global element degrees of freedom in the FE-Software and the basic element degrees of freedom in the experimental element



Consider element in structure
Two coordinate systems used in FE analysis

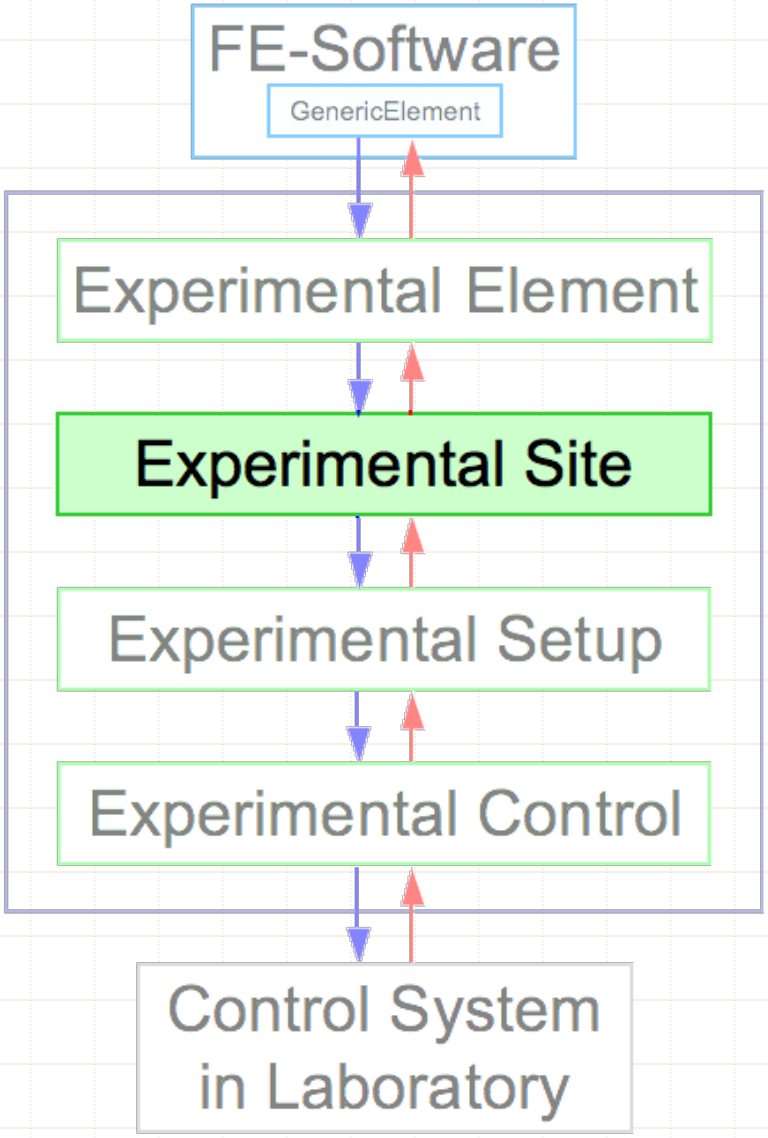


Global System



Cantilever Basic System

OpenFresco Components

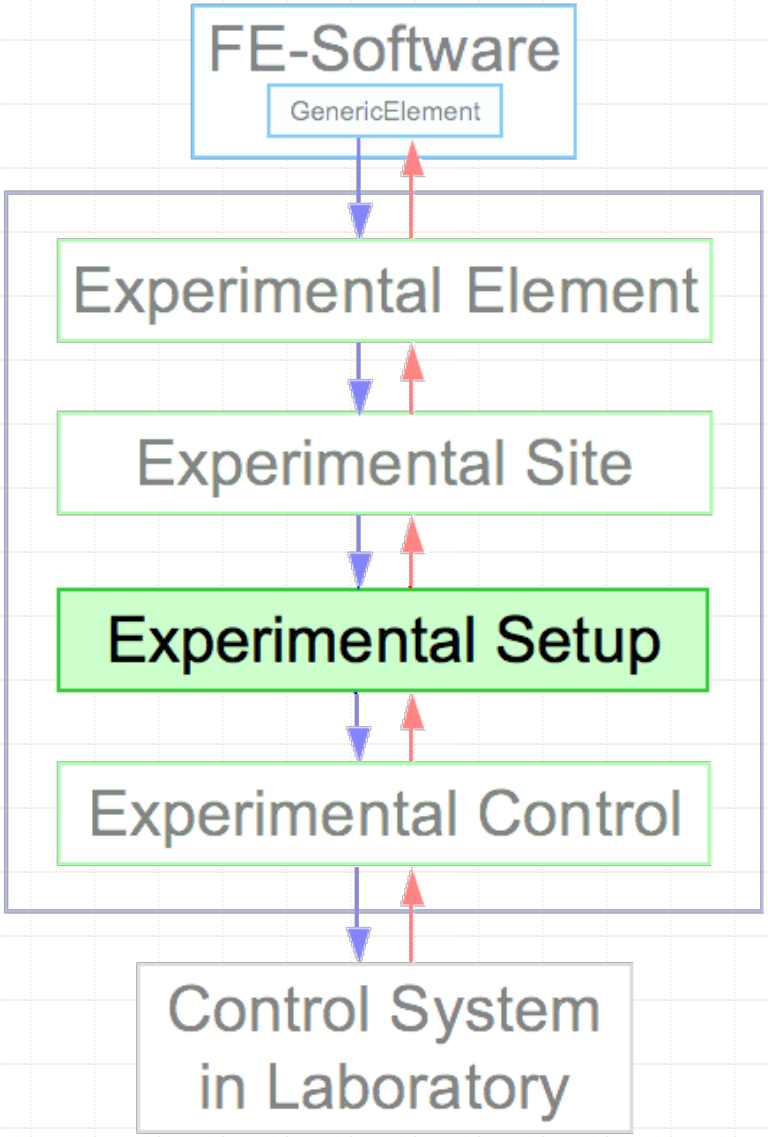


Experimental Site
Stores data and provides communication methods for distributed testing

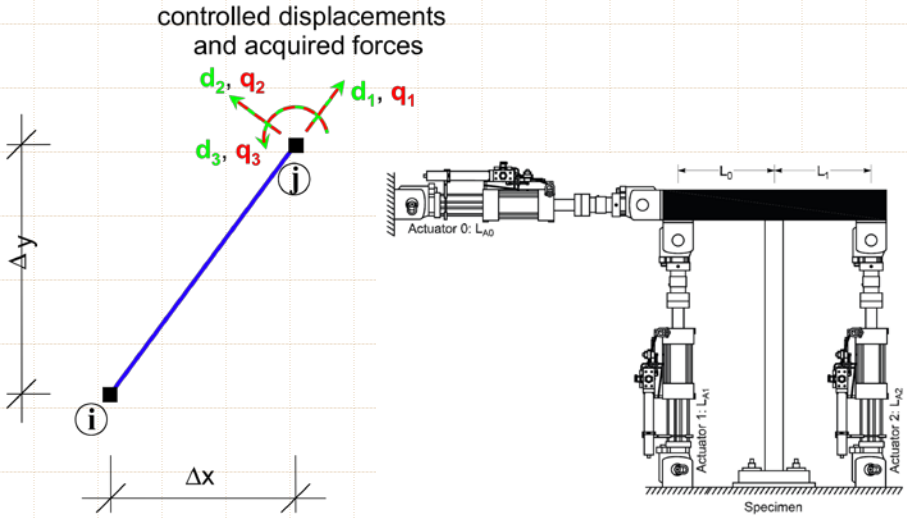
LocalExpSite available for local testing and RemoteExpSite/ActorExpSite pair available for geographically distributed testing

Utilizes communication channels with TCP, TCP+SSL or UDP communication protocols

OpenFresco Components



Experimental Setup
Transforms between the basic experimental element degrees of freedom in OpenFresco and the actuator degrees of freedom in the laboratory (linear vs. non-linear transformations are available)

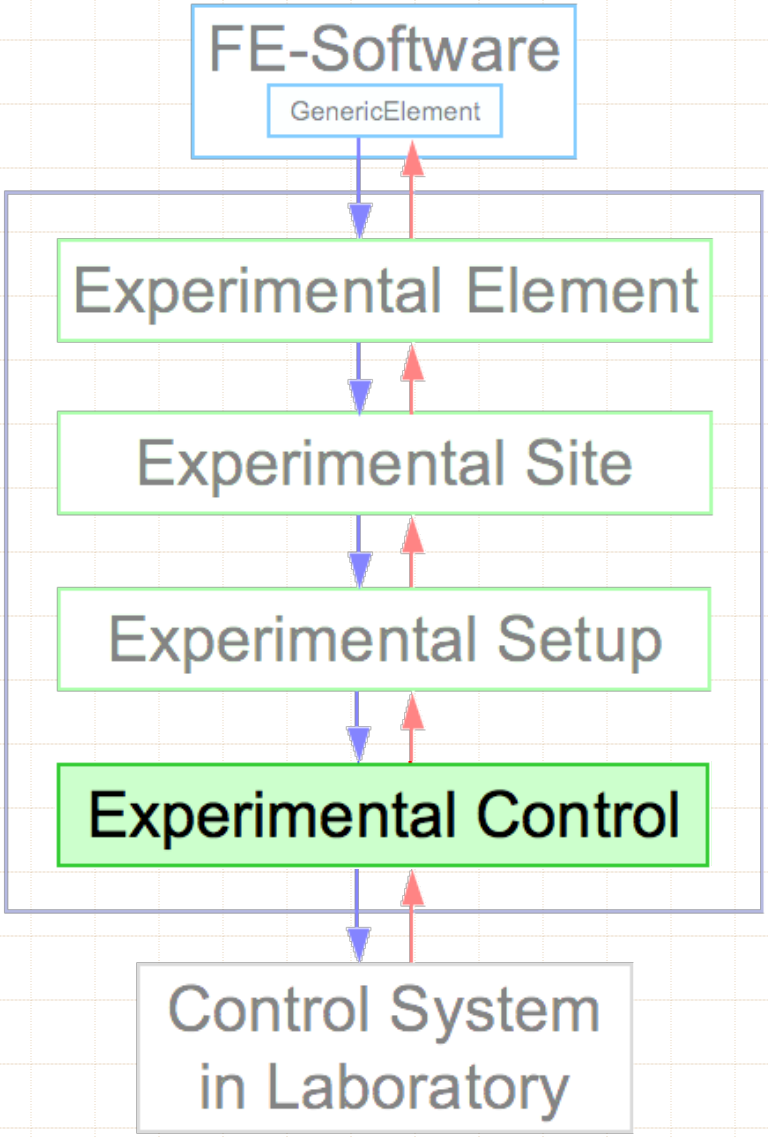


Cantilever Basic System

Actuator Setup

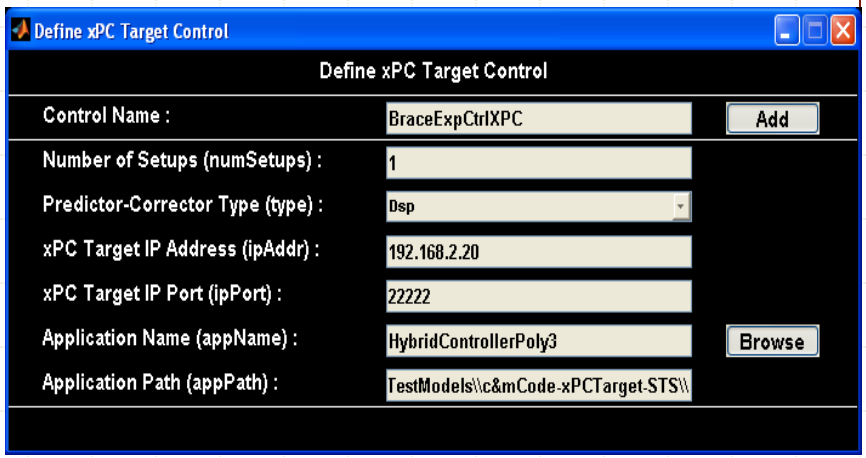
$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -L_0 \\ 0 & 1 & L_1 \end{bmatrix}$$

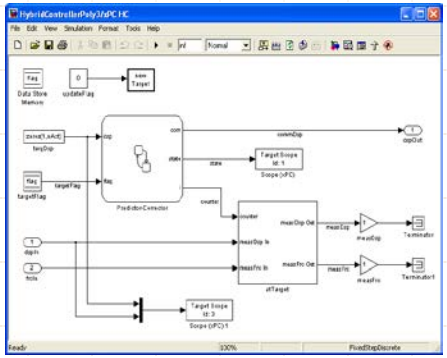
OpenFresco Components



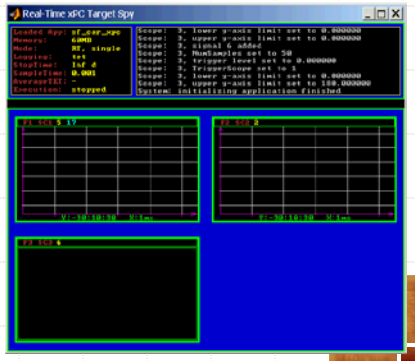
Experimental Control

Interfaces to the different control and data acquisition systems in the laboratories (IP addresses and port numbers)



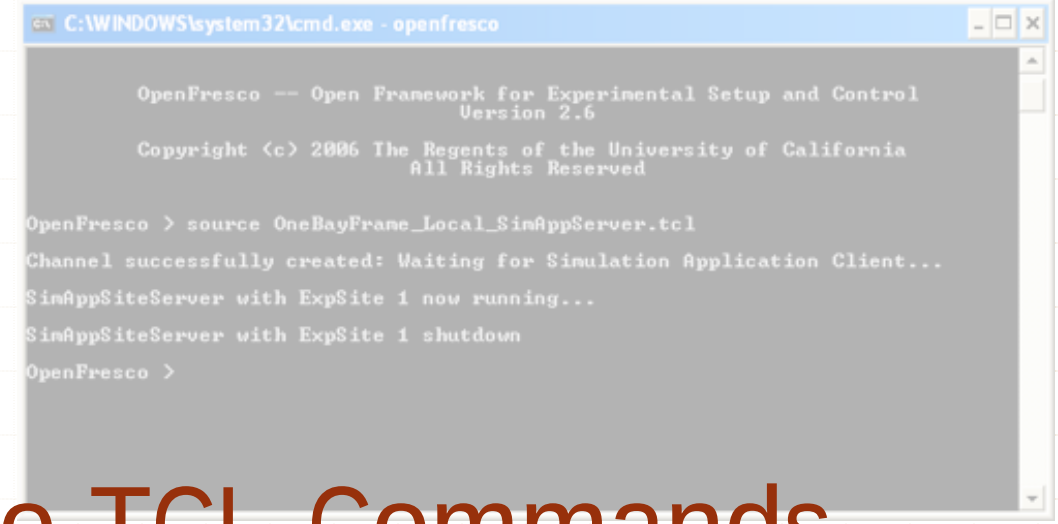


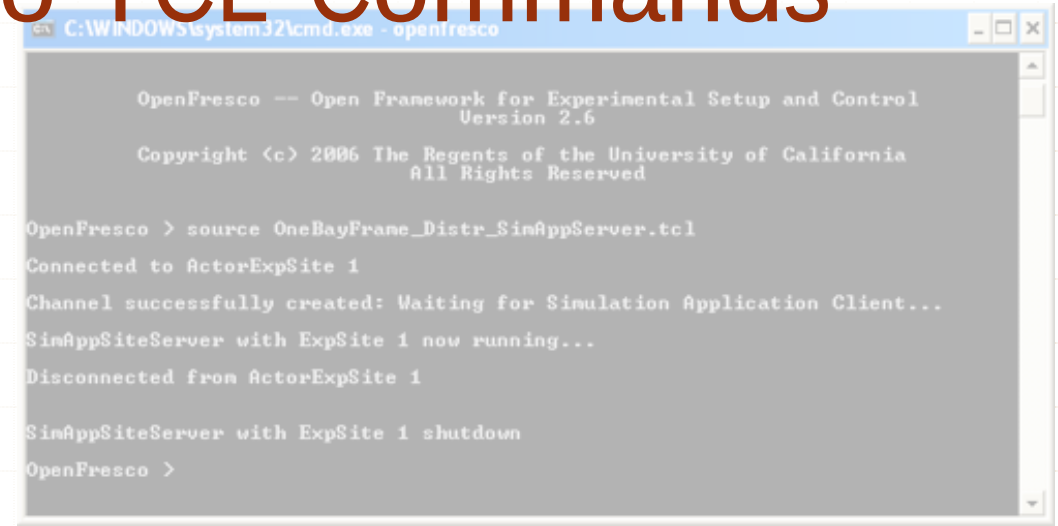
Predictor-Corrector



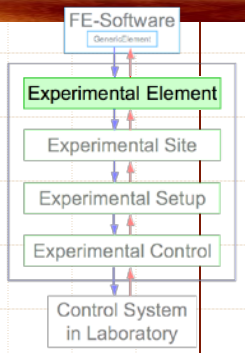
xPC Target

OpenFresco TCL-Commands





Experimental Elements

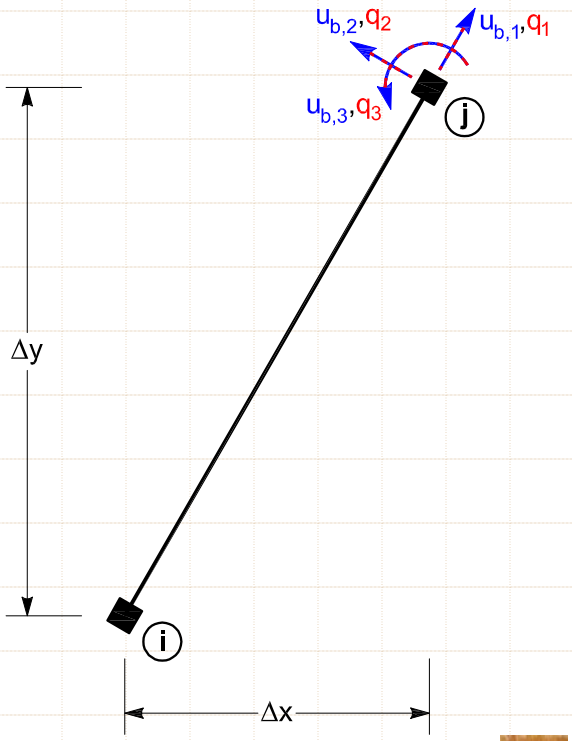


EEBeamColumn (2D,3D)

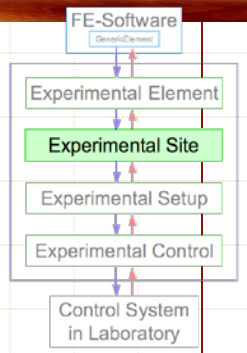
```
expElement beamColumn $eleTag $iNode $jNode
    $tranTag -site $siteTag -initStif $Kij
    <-iMod> <-rho $rho>
```

$eleTag	unique element tag
$iNode,$jNode	end nodes
$tranTag	tag of previously defined crd-transf object
$siteTag	tag of previously defined site object
$Kij	initial stiffness matrix elements (ndf x ndf)
-iMod	flag for I-Modification (optional, default=false)
$rho	mass per unit length (optional, default=0.0)

Cantilever Basic DOF



Experimental Sites

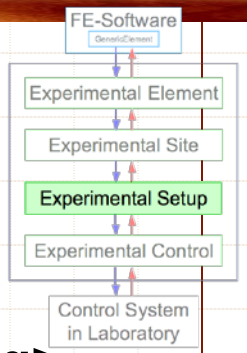


LocalExpSite

```
expSite LocalSite $tag $setupTag
```

$tag	unique site tag
$setupTag	tag of previously defined setup object

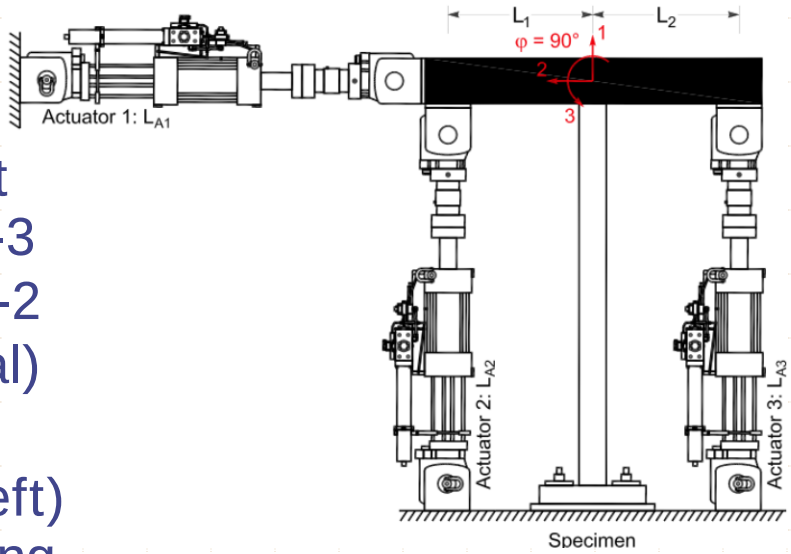
Experimental Setups



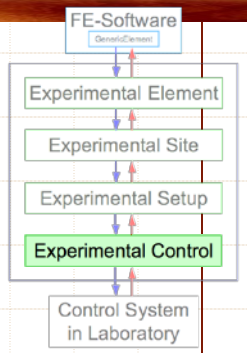
ESThreeActuators

```
expSetup ThreeActuators $tag <-control $ctrlTag>
   $La1 $La2 $La3 $L1 $L2 <-nlGeom> <-posAct1 $pos>
   <-phiLocX $phi> <-trialDispFact $f> …
```

`$tag`	unique setup tag
`$ctrlTag`	tag of previously defined control object
`$La1-3`	length of actuators 1-3
`$L1-2`	length of rigid links 1-2
`-nlGeom`	NL-geometry (optional)
`$pos`	position of actuator 1 (optional, default = left)
`$phi`	angle from rigid loading beam to local x-axis (optional, default = 0.0)



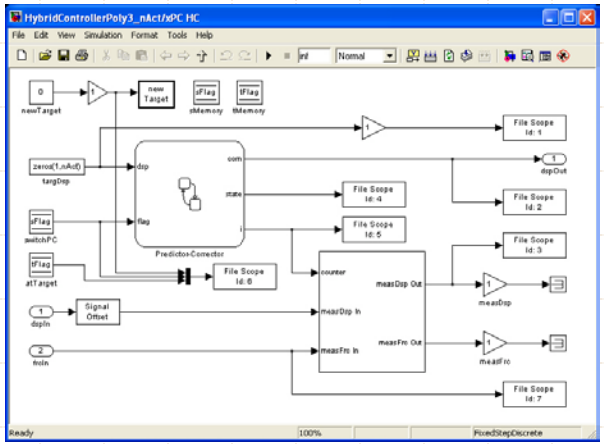
Experimental Controls

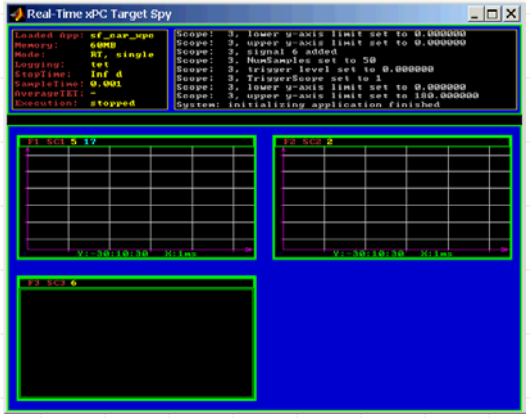


ECxPCtarget

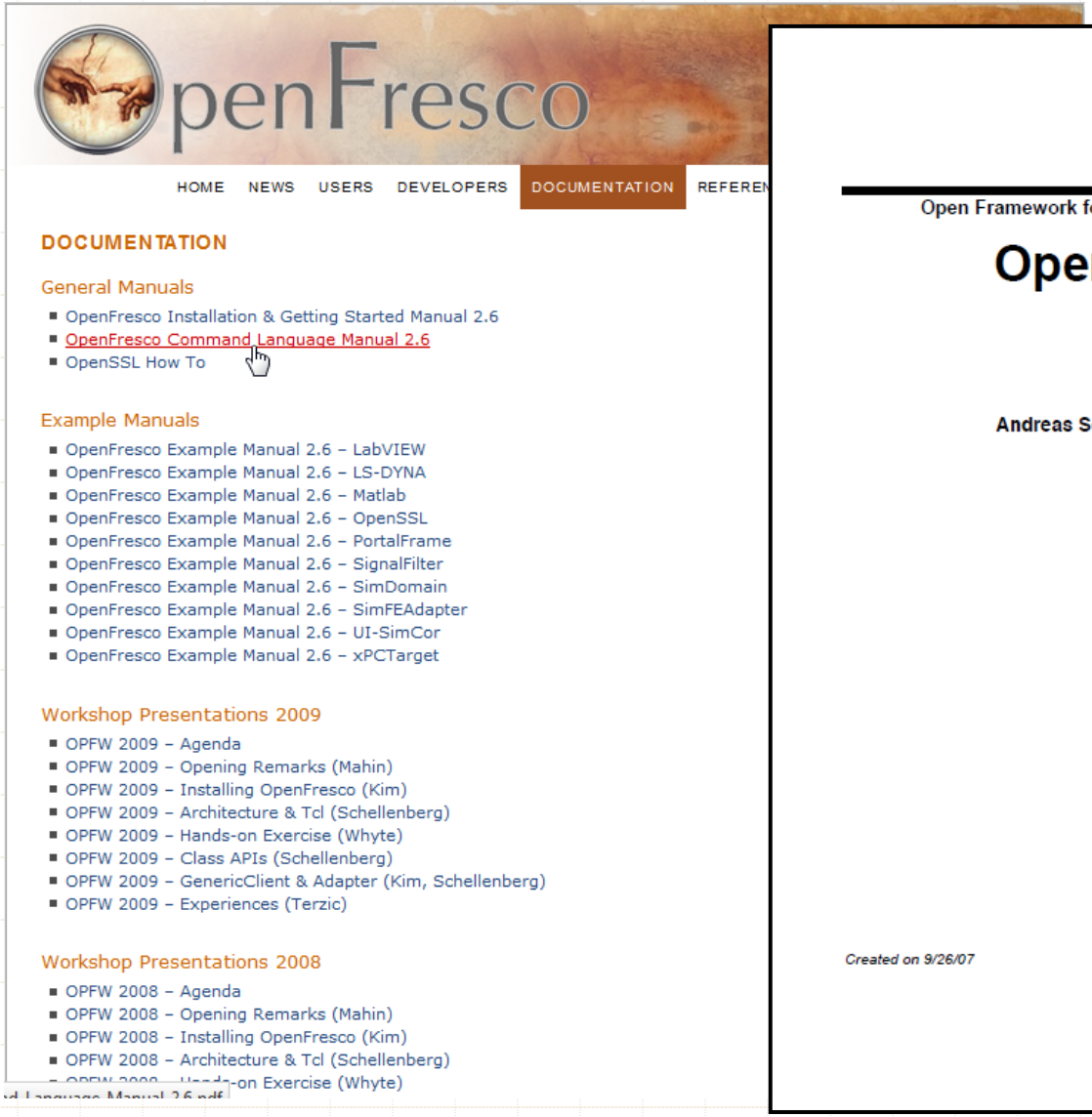
```
expControl xPCtarget $tag $type ipAddr $ipPort
    appName appPath
```

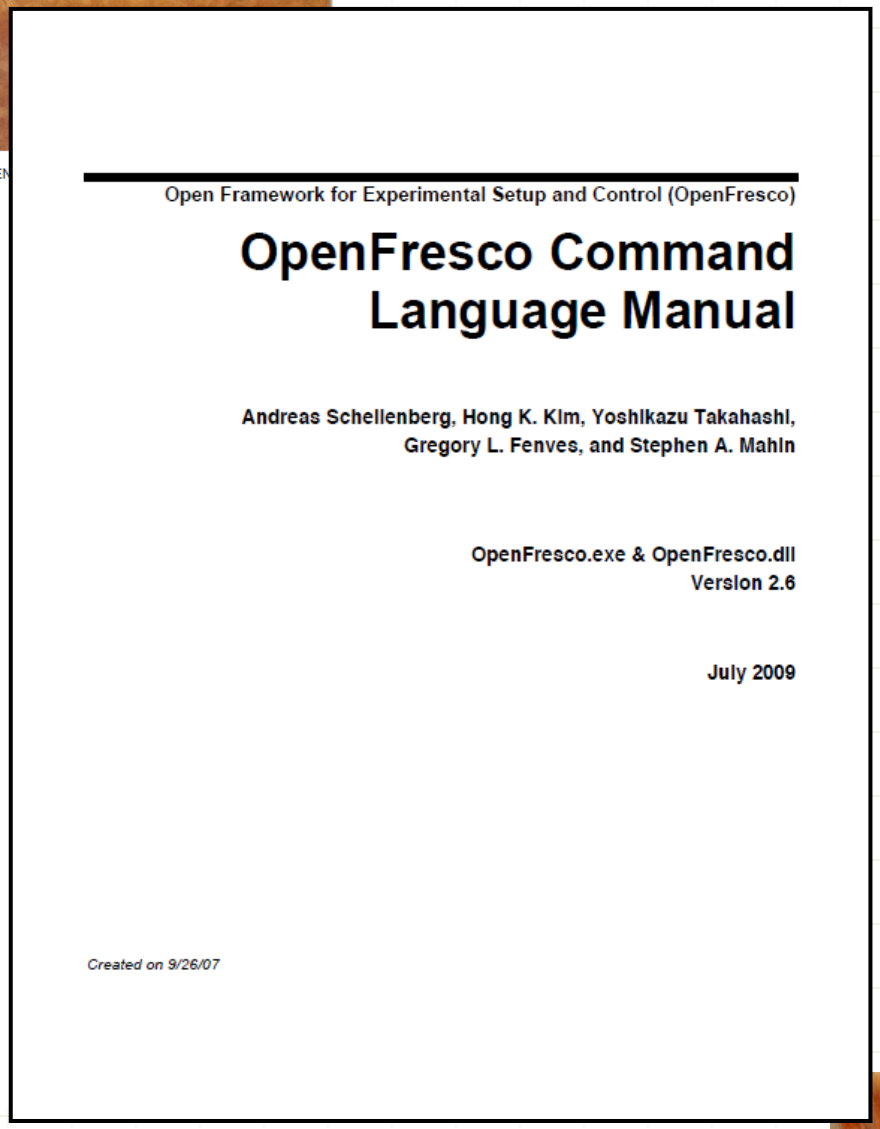
$tag	unique control tag
$type	predictor-corrector type
ipAddr	IP address of xPC Target
$ipPort	IP port of xPC Target
appName	name of Simulink application to be loaded
appPath	path to Simulink application





Command Language Manual





Open Framework for Experimental Setup and Control (OpenFresco)

OpenFresco Command Language Manual

Andreas Schellenberg, Hong K. Kim, Yoshikazu Takahashi, Gregory L. Fenves, and Stephen A. Mahin

OpenFresco.exe & OpenFresco.dll
Version 2.6

July 2009

Created on 9/26/07

Downloading and Installing OpenFresco

http://openfresco.berkeley.edu

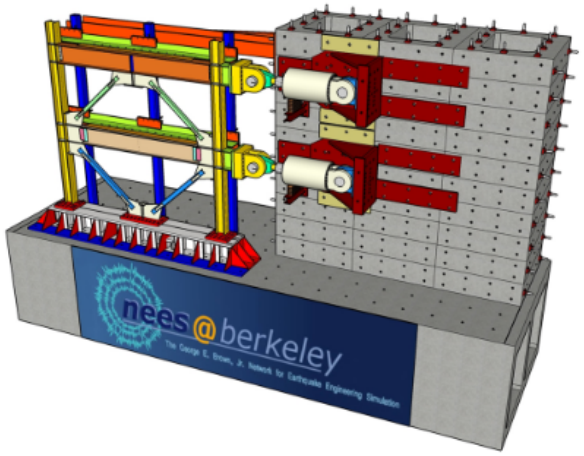
OpenFresco

HOME USERS DEVELOPERS DOCUMENTATION REFERENCES COPYRIGHT

HOME

OpenFresco (the Open-source Framework for Experimental Setup and Control) is an environment-independent software framework, that connects finite element models with control and data acquisition systems in laboratories to facilitate hybrid simulation of structural and geotechnical systems.

Hybrid simulation is an experimental testing technique where a test is executed based on a step-by-step numerical solution of the governing equations of motion for a hybrid model, formulated considering both the numerical and physical portions of a structural system. In order for the earthquake engineering community to take full advantage of this technique, OpenFresco standardizes the deployment of hybrid simulation and extends its capabilities to applications where advanced numerical techniques are utilized, boundary conditions are imposed in real-time, and dynamic loading conditions caused by wind, blast, impact, waves, fire, traffic, and, in particular, seismic events are considered. Accordingly, the architecture of the OpenFresco software package provides a great deal of flexibility, extensibility, and re-usability to the researcher or developer interested in hybrid simulation.



Search

Search

Menu

- Log in
- Register

Downloads

OpenFresco
OpenFrescoExpress
OpenSees Navigator

PEER

http://openfresco.berkeley.edu/users/openfresco

OpenFresco

HOME NEWS USERS DEVELOPERS DOCUMENTATION REFERENCES COPYRIGHT

OpenFrescoExpress
OpenFresco
Examples
Feedback

HOME

OpenFresco (the Open-source Fr al Setup and Control) is an environment-independent softwa ects finite element models with control and data acquisition syst ilitate hybrid simulation of structural and geotechnical syste

Hybrid simulation is an experimental testing technique where a test is executed based on a step-by-step numerical solution of the governing equations of motion for a hybrid model, formulated considering both the numerical and physical portions of a structural system. In order for the earthquake engineering community to take full advantage of this technique, OpenFresco standardizes the deployment of hybrid simulation and extends its capabilities to applications where advanced numerical techniques are utilized, boundary conditions are imposed in real-time, and dynamic loading conditions caused by wind, blast, impact, waves, fire, traffic, and, in particular, seismic events are considered. Accordingly, the architecture of the OpenFresco software package provides a great deal of flexibility, extensibility, and re-usability to the researcher or developer interested in hybrid simulation.

Search

Search

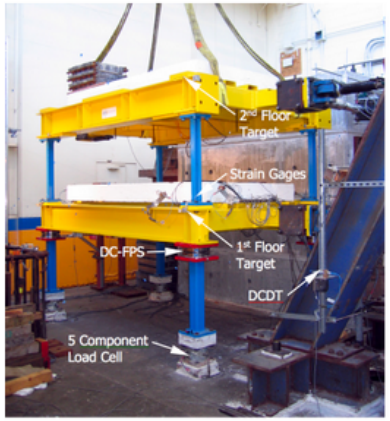
Menu

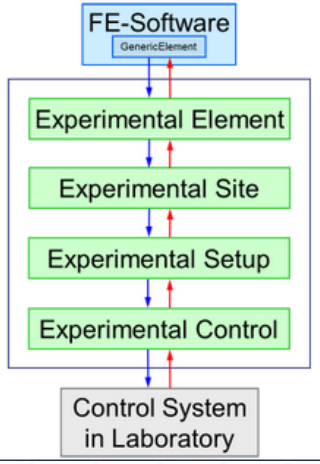
- Log in
- Register

Downloads

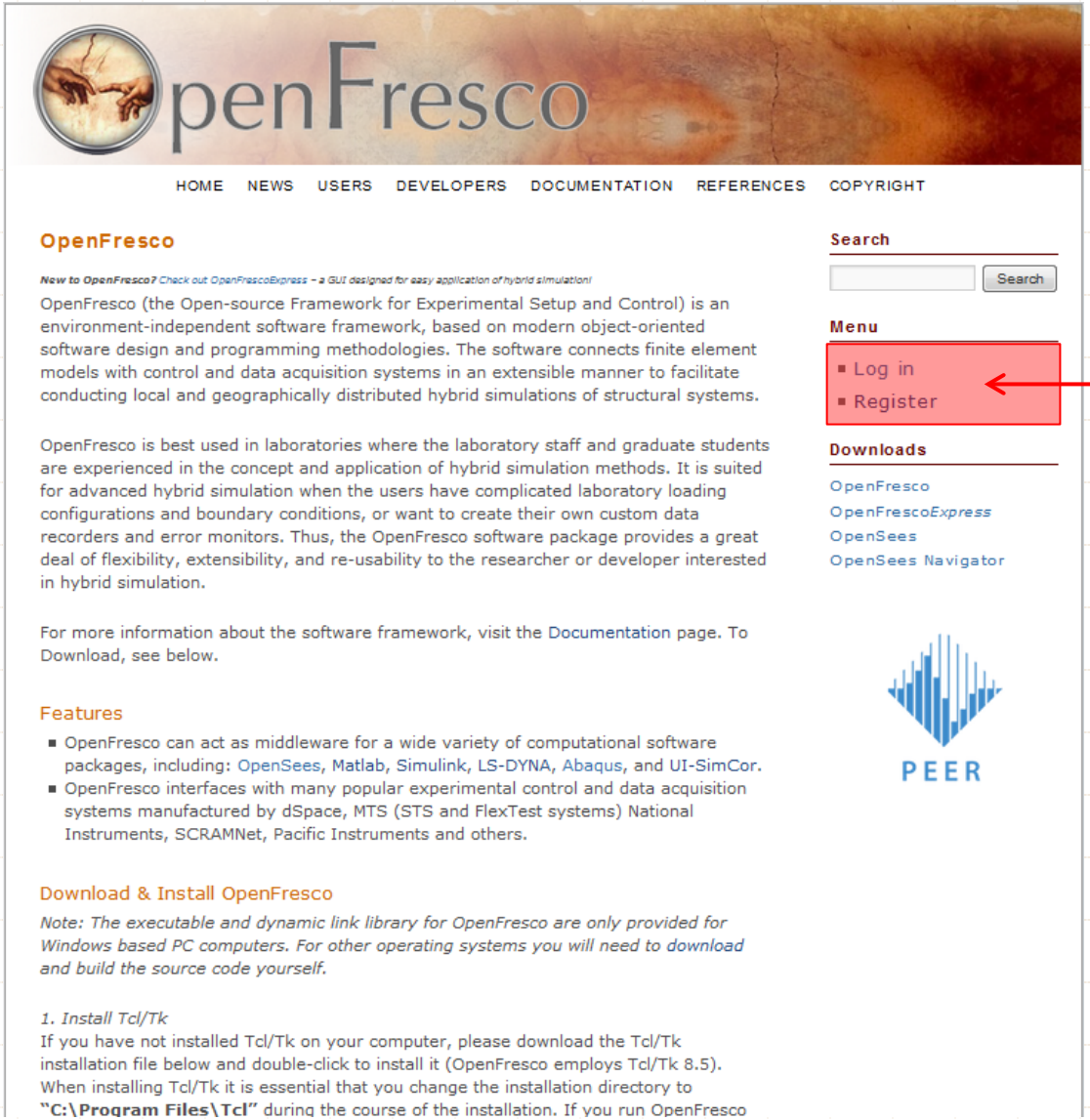
OpenFresco
OpenFrescoExpress
OpenSees
OpenSees Navigator

PEER





Login or Register



First, log in if you already have an account or register for a new account

Download Files

Download & Install OpenFresco

Note: The executable and dynamic link library for OpenFresco are only provided for Windows based PC computers. For other operating systems you will need to download and build the source code yourself.

1. Install Tcl/Tk

If you have not installed Tcl/Tk on your computer, please download the Tcl/Tk installation file below and double-click to install it (OpenFresco employs Tcl/Tk 8.5). When installing Tcl/Tk it is essential that you change the installation directory to **"C:\Program Files\Tcl"** during the course of the installation. If you run OpenFresco and you see an error message to the effect, "Cannot find tcl85.dll", you have skipped this step and must reinstall Tcl/Tk. Note that you will probably have to uninstall the version you just installed first. Download Tcl/Tk 8.5

2. Install OpenSSL (optional)

If you would like to use OpenFresco's capability to encrypt signals during geographically distributed hybrid simulations through Secure Socket Layer, please download the OpenSLL installation file from the website below and install it (OpenFresco employs the full (not light) version of OpenSSL 1.0). When installing OpenSSL it is essential that you change the installation directory to **"C:\Program Files\OpenSSL"** during the course of the installation. Download OpenSSL

3. Install Computational Driver

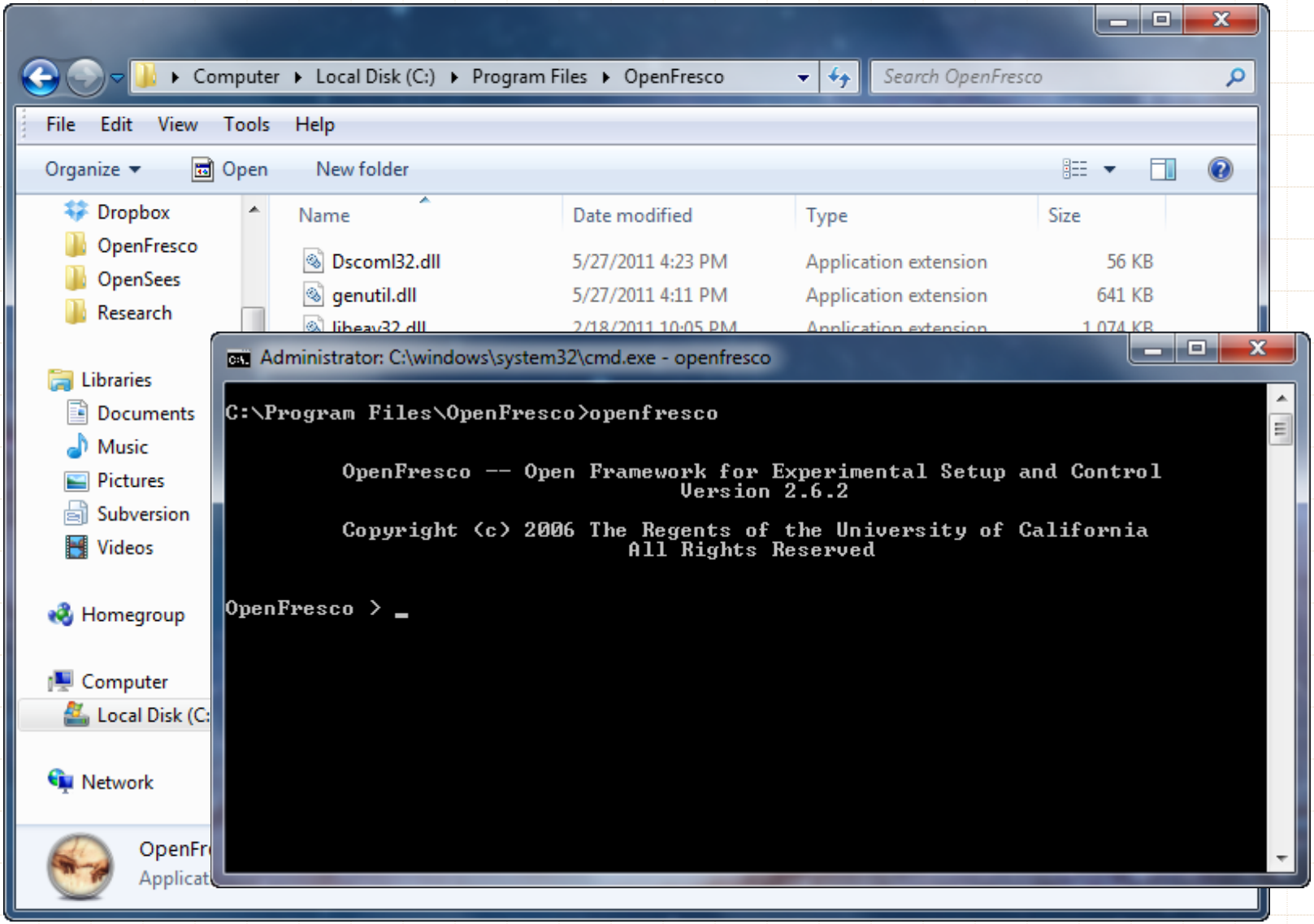
Install the structural analysis software package of your choice. OpenFresco can interface with many software packages including OpenSees, Abaqus, LS-DYNA, Matlab, Simulink, UI-SimCor, ANSYS (coming soon!)

4. Download & Extract OpenFresco

Download OpenFresco 2.6.2 *(.zip file, 5 MB)*

1. Download and install Tcl/Tk 8.5
   Important: install in C:\Program Files\Tcl
2. Download and install OpenSSL (optional)
3. Install OpenSees
4. Download and extract OpenFresco in a convenient directory

Run OpenFresco



openfresco.berkeley.edu/developers/svn

OpenFresco

HOME USERS DEVELOPERS DOCUMENTATION REFERENCES COPYRIGHT

SVN

SVN

The OpenFresco source code is stored using the Apache Subversion (SVN) software. SVN provides the means to store not only the current version of a piece of source code, but a record of all changes that have occurred to that source code over time and a record of who made those changes. The use of SVN is particularly common for software projects with multiple developers, because SVN guarantees that changes made by one developer are not accidentally removed when another developer commits changes to the source code. For the OpenFresco software project anyone can check out the code via anonymous SVN access, but only trusted developers have the ability to commit changes and additions to the code repository.

Getting the Code

To download the OpenFresco source code from the repository you
on your local machine first. You can download SVN for all major p
Linux, Windows, and MacOSX. If you are working on Windows, th
is particularly nice and easy to use. It lets you control SVN functio
menus as you navigate the file system in Windows Explorer.

Once you have SVN installed, you can download the OpenFresco

```
svn co svn://openfresco.berkeley.edu/usr/local/svn/
```

The checkout command makes a local copy of the entire OpenFr
your current working directory. By requesting .../OpenFresco/t
development trunk, which should have the latest stable source co

Browsing the Code

You can browse the source code online using WebSVN. The applic
the-minute view onto the OpenFresco repository that has been de
Subversion methodology. You can view the log of any file or direc
the files changed, added or deleted in any given revision. You car
differences between two versions of a file so as to see exactly wh
particular revision.

Search

Search

Menu

- Log in
- Register

Downloads

SUBVERSION REPOSITORIES OPENFRESCO

OpenFresco | calm | English - English

(root)/trunk/SRC - Rev 332

Rev HEAD Go

Rev 329 | Last modification | Compare with Previous | View Log | RSS feed

LAST MODIFICATION

Rev 332 2012-06-15 10:50:44

Author: aschell

Log message:

updated copyright information

Path		Last modification		Log	RSS
branches/	288	589d 07h	hongkim	Log	RSS
tags/	206	1042d 09h	aschellenberg	Log	RSS
trunk/	332	12d 11h	aschell	Log	RSS
EXAMPLES/	321	312d 23h	aschellenberg	Log	RSS
GUI/	332	12d 11h	aschell	Log	RSS
MAKES/	314	402d 08h	aschellenberg	Log	RSS
SRC/	332	12d 11h	aschell	Log	RSS
WIN32/	329	159d 00h	aschell	Log	RSS
COPYRIGHT	332	12d 11h	aschell	Log	RSS
Makefile	314	402d 08h	aschellenberg	Log	RSS

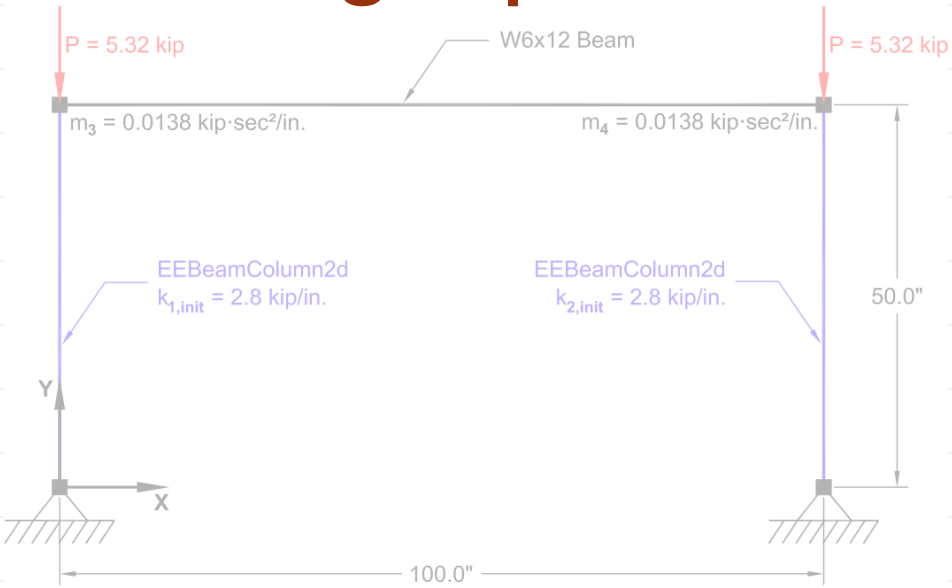
Compare Paths

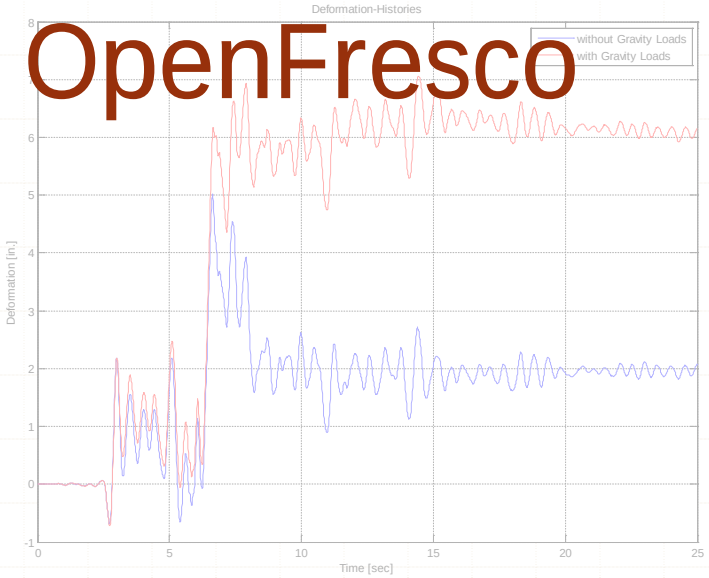
Powered by WebSVN [trunk] and Subversion 1.6.11 ✓ XHTML & CSS

PEER Center - 325 Davis Hall, University of California, Berkeley, CA 94720-1792 - (510) 642-3437 - peer_center@berkeley.edu

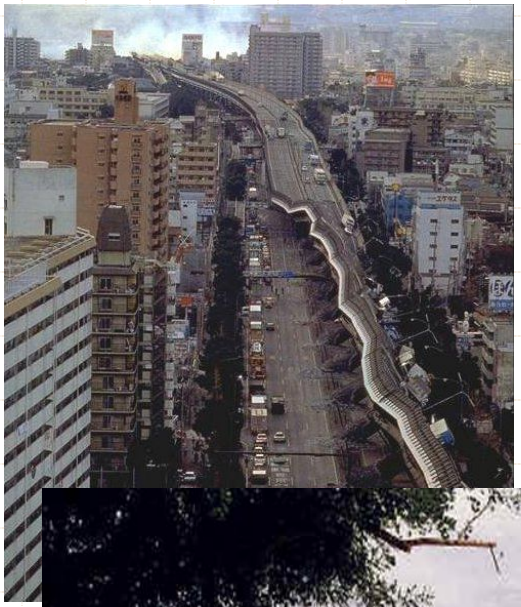
Example: Building a Hybrid Model using OpenSees & OpenFresco

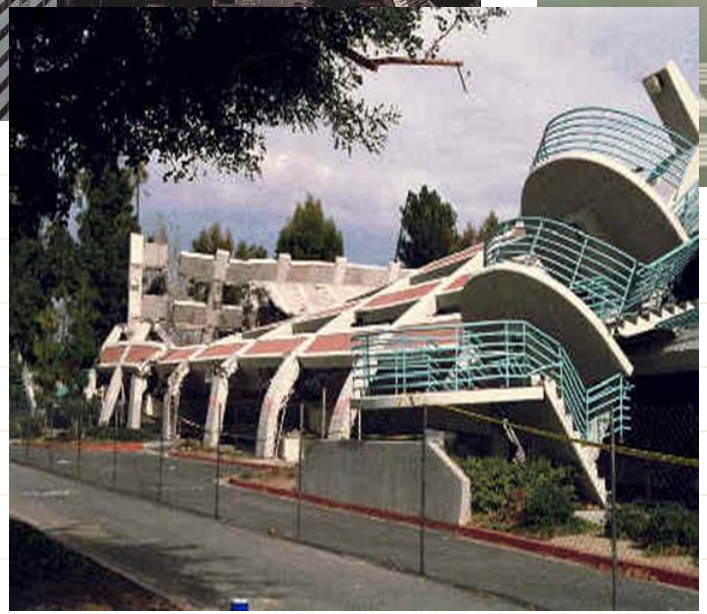
```
# Define geometric transformation
# -------------------------------
#geomTransf PDelta 1
geomTransf Corotational 1

# Define experimental elements
# ----------------------------
# left and right columns
# expElement beamColumn $eleTag $iNode $jNode $transTag -site $siteTag -initStif $Kij <-iMod
expElement beamColumn 1 3 1 1 -site 1 -initStif 1310.8 0 0 0 11.2 -280.0 0 -280.0 9333.3333
expElement beamColumn 2 4 2 1 -site 2 -initStif 1310.8 0 0 0 11.2 -280.0 0 -280.0 9333.3333

# Define numerical elements
# -------------------------
# element elasticBeamColumn $eleTag $iNode $jNode $A $E $Iz $transTag
element elasticBeamColumn 3 4 5 [illegible] 22[illegible] 1
```





HS of Structural Collapse





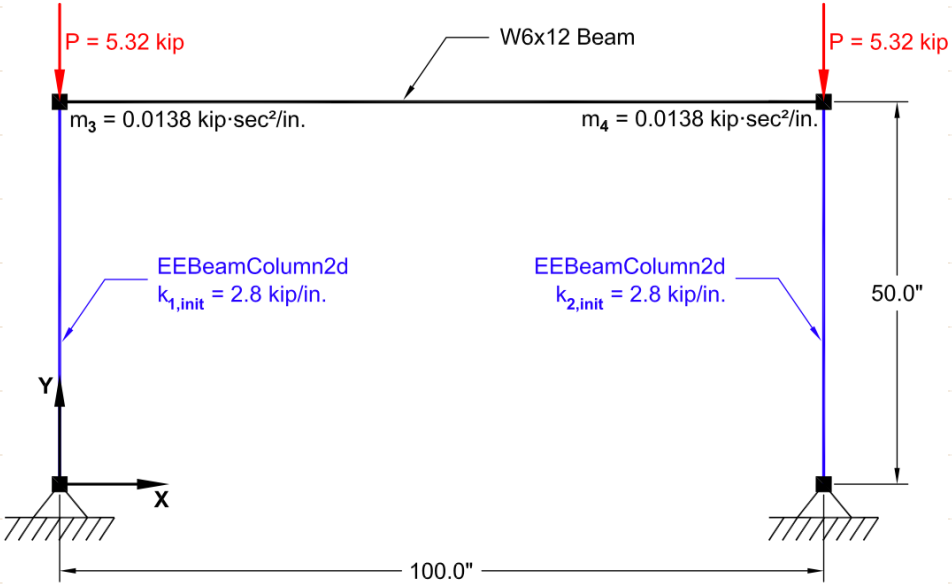
Structural Collapse

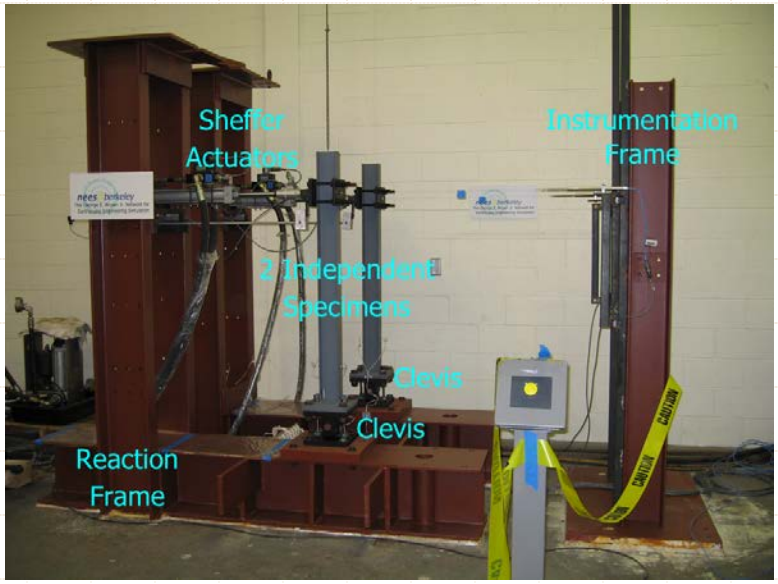
- On shaking tables, simulation of collapse is dangerous and expensive
- In hybrid simulations
  - Gravity loads and resulting geometric nonlinearities can be modeled analytically
    - Therefore, no complex active or passive gravity load setups are necessary
  - Actuator movements will limit displacements during collapse (safety)
    - Thus, there is no need to protect expensive test equipment from specimen impact
  - Only critical, collapse-sensitive elements of a structure need to be physically modeled

Implementation in a Hybrid Model

- Physical portion of the model:
  - Test material and cross-section level response
- Analytical portion of the model:
  - Apply the gravity and/or prestress loads
  - Provide the geometric transformations such that the second-order effects due to axial loads are accounted for
  - Model the rest of the structure

Structural Collapse of Portal Frame





Properties of Model:

- NDOF = 8 (4 with mass)
- Period: T_1 = 0.49 sec
- Damping: ζ_1 = 0.05
- P = 50% of φPn
- Crd-Trans: P-Delta, Corotational
- ExpElements: EEBeamColumn2d
- ExpSetups: ESOneActuator
- ExpControl: ECxPCtarget
- SACNF01: pga = 0.906g

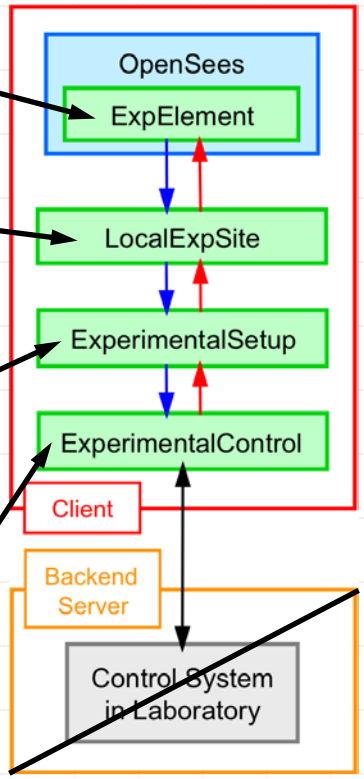
OpenFresco Local Architecture

Exp. beamColumn element defined in OpenSees

Communication methods for distributed testing. In this case, we are using a local site.

Transforms between experimental element DOFs in OPF and the actuator DOFs in the laboratory. Linear and non-linear transformations are available.

Interface with control and data acquisition systems. In this example, SimUniaxialMaterials will simulate the response of the experimental element using a material defined in OpenSees, Steel02



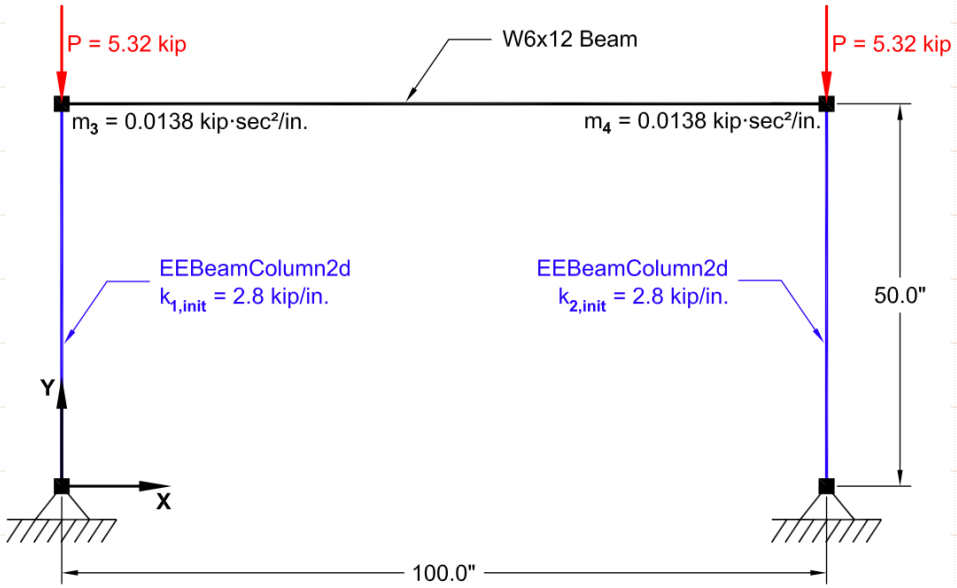
Fully simulated

Tcl File Components

Geometry
Materials
Experimental Control
Experimental Setup
Experimental Site
Geometric Transformation
Experimental Elements
Numerical Elements
Gravity Loads
Gravity Analysis
Dynamic Loads
Dynamic Analysis

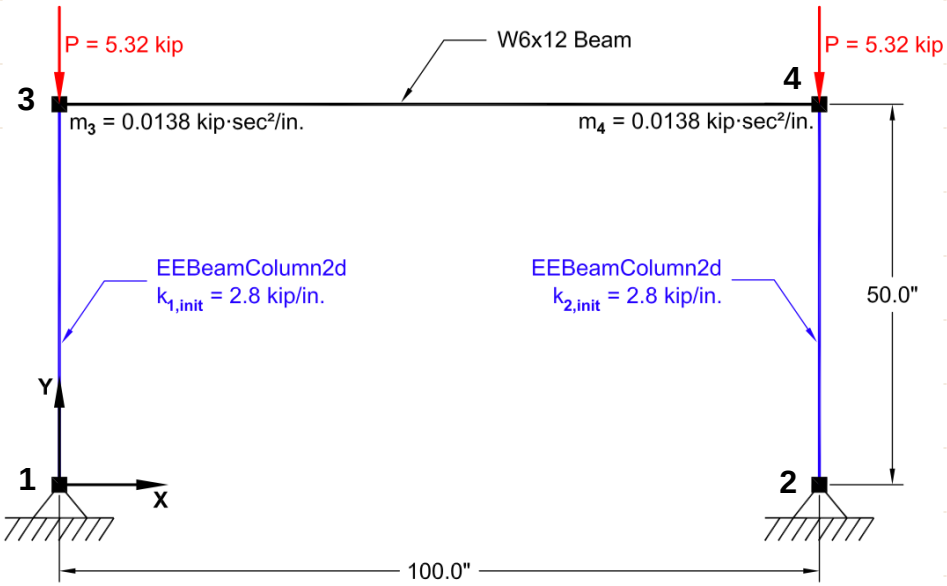
Portal Frame Model

```
# ------------------------------
# Start of model generation
# ------------------------------
# create ModelBuilder (with two-dimensions and 3 DOF/node)
model BasicBuilder -ndm 2 -ndf 3

# Load OpenFresco package
# -----------------------
# (make sure all dlls are in the same folder as openSees.exe)
loadPackage OpenFresco
```



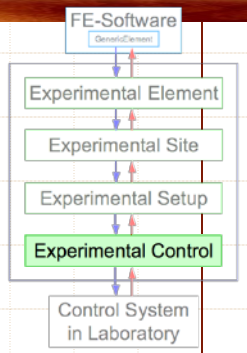
Geometry

```
# Define geometry for model
# -------------------------
set withGravity 1;
set Pc 10.638;
set P [expr 0.5*$Pc];
set mass3 [expr $P/386.1];
set mass4 [expr $P/386.1];
# node $tag $xCrd $yCrd $mass
node  1     0.0   0.0
node  2   100.0   0.0
node  3     0.0  50.0  -mass $mass3 $mass3 0.0
node  4   100.0  50.0  -mass $mass4 $mass4 0.0

# set the boundary conditions
# fix $tag $DX $DY $RZ
fix 1   1  1  0
fix 2   1  1  0
```



- withGravity 0: turns off gravity loads (no P-delta)
- withGravity 1: turns on gravity loads (P-delta)
- Assigned no rotational mass – must use implicit integration method

Tcl File Components



Geometry

Materials

Experimental Control

Experimental Setup

Experimental Site

Geometric Transformation

Experimental Elements

Numerical Elements

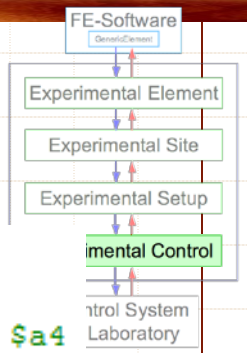
Gravity Loads

Gravity Analysis

Dynamic Loads

Dynamic Analysis

Materials/Experimental Control



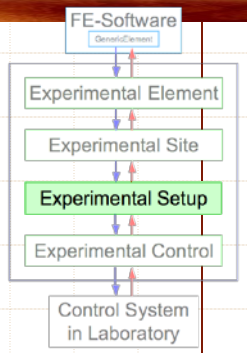

```
# Define materials
# ----------------
# uniaxialMaterial Steel02 $matTag $Fy $E $b $R0 $cR1 $cR2 $a1 $a2 $a3 $a4
uniaxialMaterial Steel02 1 1.5 2.8 0.01 18.5 0.925 0.15 0.0 1.0 0.0 1.0
#uniaxialMaterial Elastic 1 2.8

# Define experimental control
# ---------------------------
# expControl SimUniaxialMaterials $tag $matTags
expControl SimUniaxialMaterials 1 1
expControl SimUniaxialMaterials 2 1
```

Column 1 (left)

Column 2 (right)

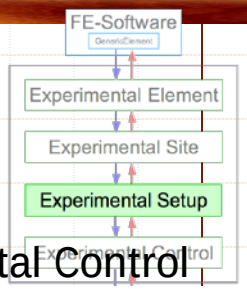
- Want to control two columns
- SimUniaxialMaterials used to simulate a specimen
- Need to create a separate experimental control for each element so create experimental control with tags “1” and “2”
- Assign a material tag to each

Tcl File Components



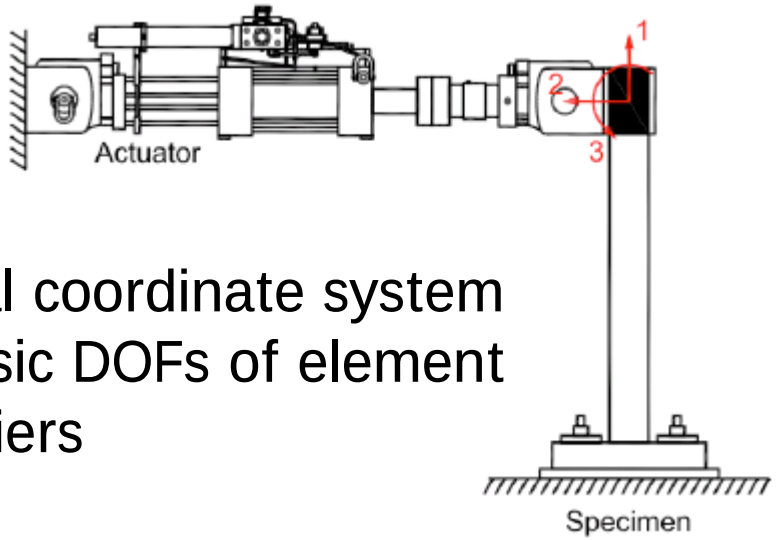
Geometry
Materials
Experimental Control
Experimental Setup
Experimental Site
Geometric Transformation
Experimental Elements
Numerical Elements
Gravity Loads
Gravity Analysis
Dynamic Loads
Dynamic Analysis

Experimental Setup and Site



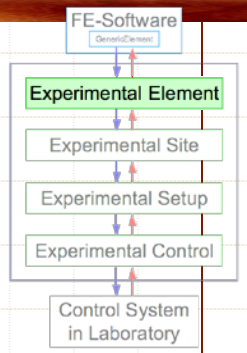
Refers back to SimUniaxialMaterials Experimental Control
Left and right columns (tags 1 and 2)

```
# Define experimental setup
# -------------------------
# expSetup OneActuator $tag <-control $ctrlTag> $dir -sizeTrialOut $t $o <-trialDispFact $f>
expSetup OneActuator 1 -control 1 2 -sizeTrialOut 3 3
expSetup OneActuator 2 -control 2 2 -sizeTrialOut 3 3

# Define experimental site
# ------------------------
# expSite LocalSite $tag $setupTag
expSite LocalSite 1 1
expSite LocalSite 2 2
```



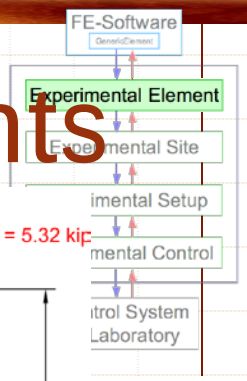
- OneActuator direction in element's local coordinate system
- sizeTrialOut are equal to number of basic DOFs of element
- Optional Factors: all factors are multipliers
- e.g.
- -crtlDispFact 2 = target displacement x 2
- -daqDispFact 0.5 = measured displacement x 0.5

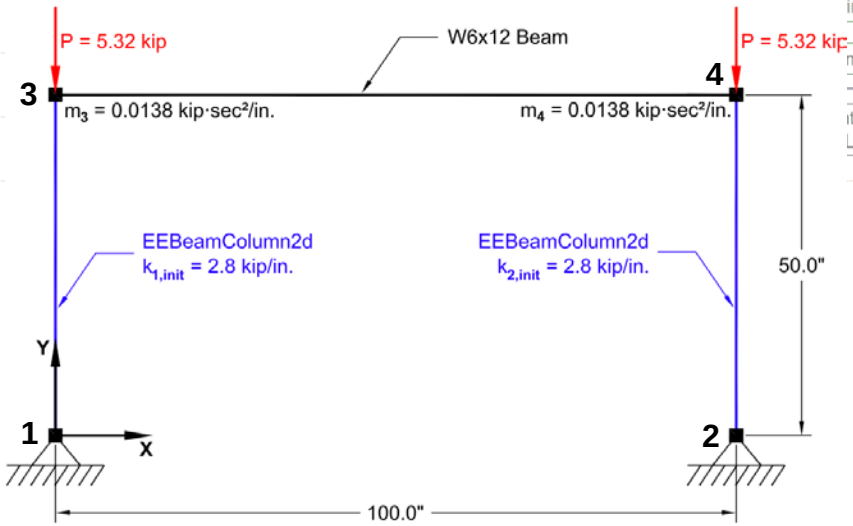
Tcl File Components



Geometry
Materials
Experimental Control
Experimental Setup
Experimental Site
Geometric Transformation
Experimental Elements
Numerical Elements
Gravity Loads
Gravity Analysis
Dynamic Loads
Dynamic Analysis

Geometric Transf & Exp. Elements

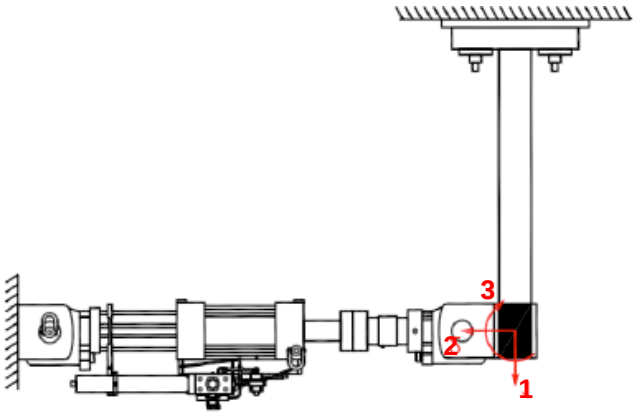





```
# Define geometric transformation
# -------------------------------
#geomTransf PDelta 1
geomTransf Corotational 1

# Define experimental elements
# ----------------------------
# left and right columns
# expElement beamColumn $eleTag $iNode $jNode $transTag -site $siteTag -initStif $Kij <-iMod
expElement beamColumn 1 3 1 1 -site 1 -initStif 1310.8 0 0 0 11.2 -280.0 0 -280.0 9333.3333
expElement beamColumn 2 4 2 1 -site 2 -initStif 1310.8 0 0 0 11.2 -280.0 0 -280.0 9333.3333

# Define numerical elements
# -------------------------
# element elasticBeamColumn $eleTag $iNode $jNode $A $E
element elasticBeamColumn 3 3 4 3.55 29000 22.1 1
```

$$K_{init} = \begin{bmatrix} \frac{EA}{L} & 0 & 0 \\ 0 & \frac{12EI}{L^3} & -\frac{6EI}{L^2} \\ 0 & -\frac{6EI}{L^2} & \frac{4EI}{L} \end{bmatrix} = \begin{bmatrix} 1310.8 & 0 & 0 \\ 0 & 11.2 & -280.0 \\ 0 & -280.0 & 9333.3 \end{bmatrix}$$

- L=50”
- A=2.26 in^2
- I=4.02 in^4
- E=29000 ksi



Tcl File Components

Geometry
Materials
Experimental Control
Experimental Setup
Experimental Site
Geometric Transformation
Experimental Elements
Numerical Elements
Gravity Loads
Gravity Analysis
Dynamic Loads
Dynamic Analysis

Gravity Loads

```
if {$withGravity} {
    # Define gravity loads
    # --------------------
    # Create a Plain load pattern with a Linear TimeSeries
    pattern Plain 1 "Linear" {
        # Create nodal loads at nodes 2
        #    nd    FX          FY  MZ
        load  3   0.0  [expr -$P] 0.0
        load  4   0.0  [expr -$P] 0.0
    }
    # ------------------------------
    # End of model generation
    # ------------------------------
```

Loads in the negative Y-direction at nodes 3 and 4

Gravity Analysis Options

```
# ------------------------------
# Start of analysis generation
# ------------------------------
# Create the system of equation
system BandGeneral
# Create the DOF numberer
numberer Plain
# Create the constraint handler
constraints Plain
# Create the convergence test
test EnergyIncr 1.0e-6 10
# Create the integration scheme
integrator LoadControl 0.1
# Create the solution algorithm
algorithm Newton
# Create the analysis object
analysis Static
# ------------------------------
# End of analysis generation
# ------------------------------
```

Banded General SOE

DOFs assigned arbitrarily (ok for small models)

Only using homogeneous single point constraints

Test EnergyIncr $tol $maxNumIter

Load Control with 10 steps

Newton-Raphson algorithm

Perform a static analysis

Gravity Recorders and Analysis

```
# ------------------------------
# Start of recorder generation
# ------------------------------
# create a Recorder object for the nodal displacements at node 2
recorder Node -file Gravity_Dsp.out -time -node 3 4 -dof 1 2 3 disp
recorder Element -file Gravity_Frc.out -time -ele 1 2 3 force
# --------------------------------
# End of recorder generation
# --------------------------------


# ------------------------------
# Perform the gravity analysis
# ------------------------------
# perform the gravity load analysis, requires 10 steps to reach the load level
if {[analyze 10] == 0} {
    puts "\nGravity load analysis completed"
} else {
    puts "\nGravity load analysis failed"
    exit -1
}
```

Tcl File Components

Geometry
Materials
Experimental Control
Experimental Setup
Experimental Site
Geometric Transformation
Experimental Elements
Numerical Elements
Gravity Loads
Gravity Analysis
Dynamic Loads
Dynamic Analysis

Dynamic Loads

```
# Define dynamic loads
# --------------------
# set time series to be passed to uniform excitation
set dt 0.01
set scale 1.2
timeSeries Path 1 -filePath SACNF01.txt -dt $dt -factor [expr 386.1*$scale]

# create UniformExcitation load pattern
# pattern UniformExcitation $tag $dir -accel $tsTag <-vel0 $vel0>
pattern UniformExcitation 2 1 -accel 1

# calculate the rayleigh damping factors for nodes & elements
set alphaM     1.2797;     # D = alphaM*M
set betaK      0.0;        # D = betaK*Kcurrent
set betaKinit  0.0;        # D = beatKinit*Kinit
set betaKcomm  0.0;        # D = betaKcomm*KlastCommit

# set the rayleigh damping
rayleigh $alphaM $betaK $betaKinit $betaKcomm
# ------------------------------
# End of model generation
# ------------------------------
```

Place ground motion file in the same folder as the PortalFrame_Local.tcl file

Dynamic Analysis Options

```
# ------------------------------
# Start of analysis generation
# ------------------------------
# create the system of equations
system BandGeneral

# create the DOF numberer
numberer Plain

# create the constraint handler
constraints Plain

# create the convergence test
test FixedNumIter 5

# create the integration scheme
integrator NewmarkHSFixedNumIter 0.5 0.25

# create the solution algorithm
algorithm Newton

# create the analysis object
analysis Transient
# ------------------------------
# End of analysis generation
# ------------------------------
```

Same as gravity analysis

5 iterations/time-step

NewmarkHSFixedNumIter: implicit Newmark method with 5 iterations/time-step

γ=0.5: second order accuracy, no numerical damping

β=0.25: average acceleration, unconditional stability

Dynamic Recorders

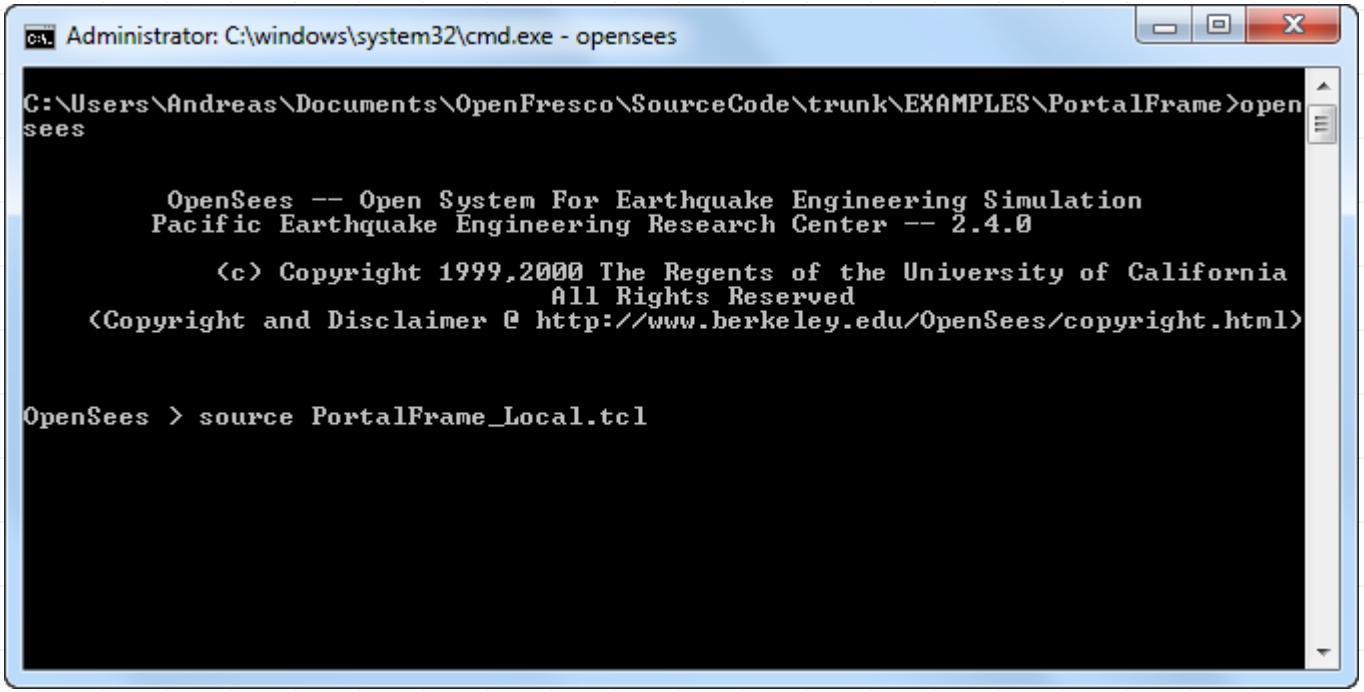
```
# ------------------------------
# Start of recorder generation
# ------------------------------
# create the recorder objects
recorder Node -file Node_Dsp.out -time -node     3 4 -dof 1 2 3 disp
recorder Node -file Node_Vel.out -time -node     3 4 -dof 1 2 3 vel
recorder Node -file Node_Acc.out -time -node     3 4 -dof 1 2 3 accel
recorder Node -file Node_Rxn.out -time -node 1 2 3 4 -dof 1 2 3 reactionIncludingInertia

recorder Element -file Elmt_glbFrc.out  -time -ele 1 2 3 forces
expRecorder Control -file Control_ctrlDsp.out -time -control 1 2 ctrlDisp
expRecorder Control -file Control_daqDsp.out  -time -control 1 2 daqDisp
expRecorder Control -file Control_daqFrc.out  -time -control 1 2 daqForce
# --------------------------------
# End of recorder generation
# --------------------------------
```

Dynamic Analysis

```
# ------------------------------
# Finally perform the analysis
# ------------------------------
# perform an eigenvalue analysis
set pi [expr acos(-1.0)]
set lambda [eigen -fullGenLapack 4]
puts "\nEigenvalues at start of transient:"
puts "|   lambda    |  omega    |  period | frequency |"
foreach lambda $lambda {
    set omega [expr pow($lambda,0.5)]
    set period [expr 2.0*$pi/$omega]
    set frequ [expr 1.0/$period]
    puts [format "| %5.3e | %8.4f | %7.4f | %9.4f |" $lambda $omega $period $frequ]
}

# open output file for writing
set outFileID [open elapsedTime.txt w]
# perform the transient analysis
set tTot [time {
    for {set i 1} {$i < 2500} {incr i} {
        set t [time {analyze  1  [expr $dt/1.0]}]
        puts $outFileID $t
        #puts "step $i"
    }
}]
```

Running the Hybrid Simulation

- Start the OpenSees executable file from the directory where you saved PortalFrame_Local.tcl
- At the prompt, type source PortalFrame_Local.tcl and press enter



Run Simulation

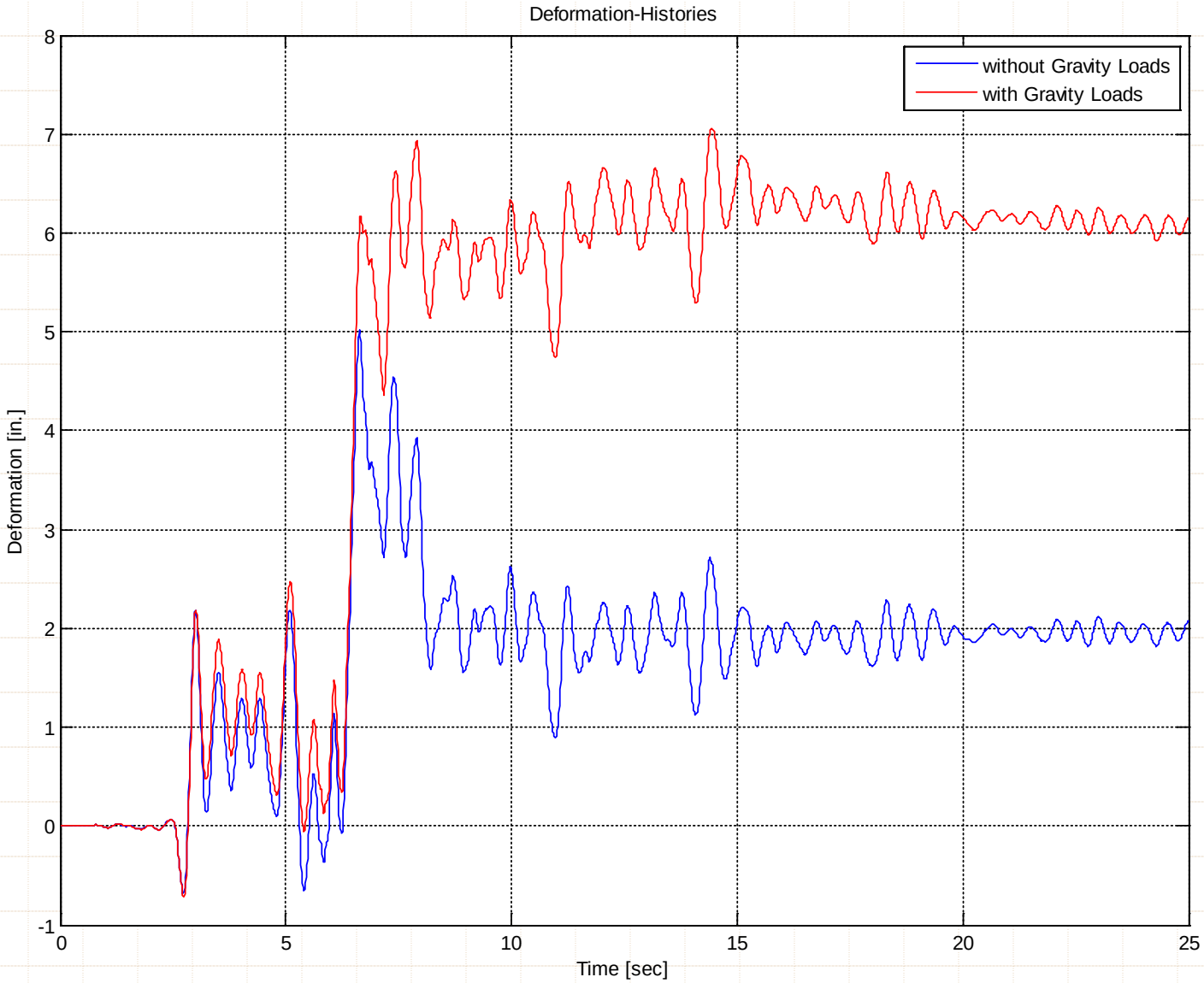
Administrator: C:\windows\system32\cmd.exe

```
OpenSees > source PortalFrame_Local.tcl


         OpenFresco -- Open Framework for Experimental Setup and Control
                                Version 2.6.2

         Copyright (c) 2006 The Regents of the University of California
                            All Rights Reserved



WARNING EEBeamColumn2d::getTangentStiff() - Element: 1
TangentStiff cannot be calculated.
Return InitialStiff including GeometricStiff instead.
Subsequent getTangentStiff warnings will be suppressed.

WARNING EEBeamColumn2d::getTangentStiff() - Element: 2
TangentStiff cannot be calculated.
Return InitialStiff including GeometricStiff instead.
Subsequent getTangentStiff warnings will be suppressed.

Gravity load analysis completed

WARNING: NewmarkHSFixedNumIter::domainChanged() - assuming Ut-1 = Ut

Eigenvalues at start of transient:
|    lambda    |   omega    |  period | frequency |
| 1.639e+002 |  12.8040 |  0.4907 |    2.0378 |
| 9.515e+004 | 308.4630 |  0.0204 |   49.0934 |
| 9.532e+004 | 308.7421 |  0.0204 |   49.1378 |
| 1.496e+005 | 386.7612 |  0.0162 |   61.5550 |

Elapsed Time = 815445 microseconds per iteration


C:\Users\Andreas\Documents\OpenFresco\SourceCode\trunk\EXAMPLES\PortalFrame>_
```

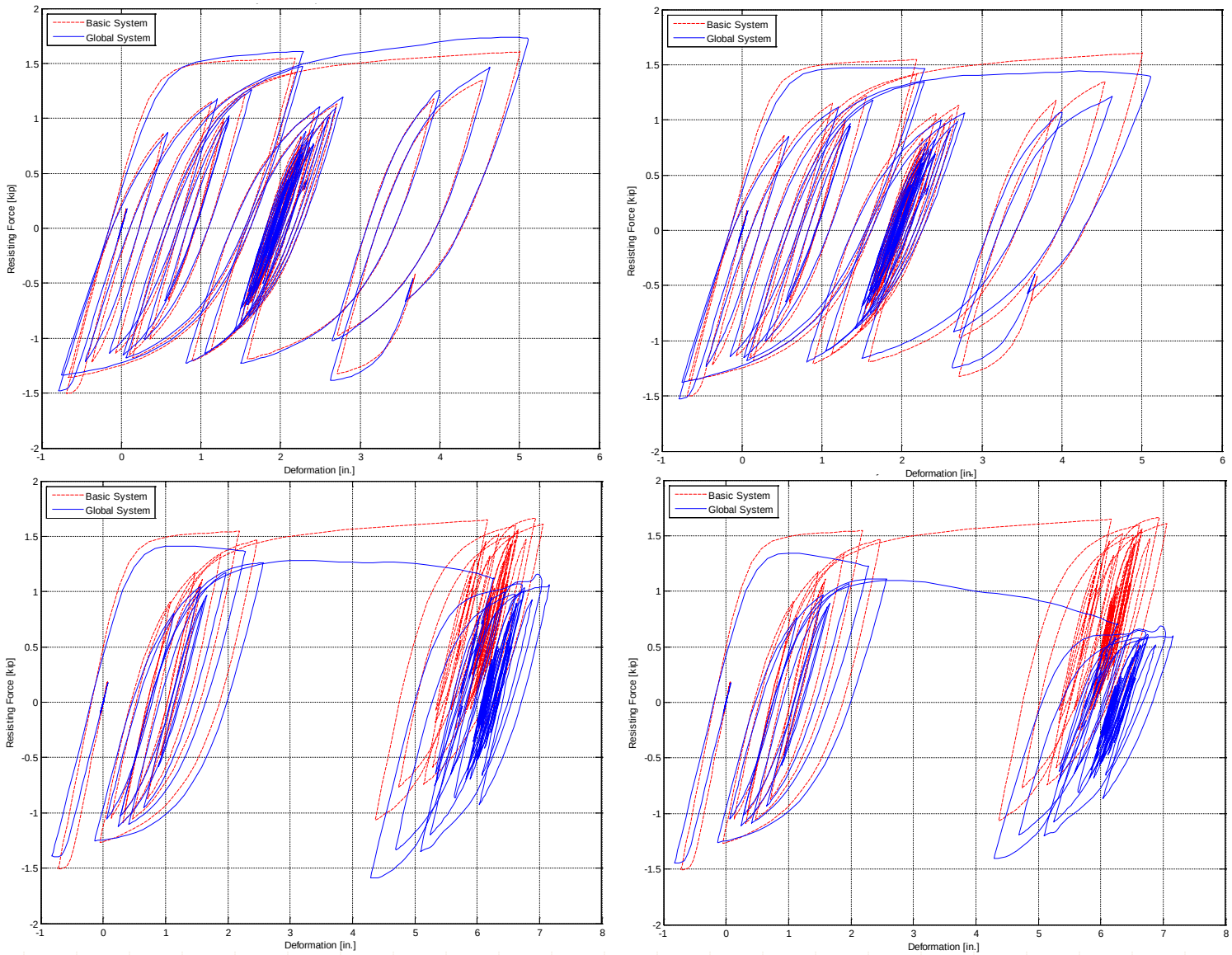
Recorders Save Output Files

Name	Date modified	Type	Size
PortalFrame_Local.tcl	3/6/2013 7:20 PM	TCL File	9 KB
Node_Vel.out	3/6/2013 8:50 PM	OUT File	164 KB
Node_Rxn.out	3/6/2013 8:50 PM	OUT File	286 KB
Node_Dsp.out	3/6/2013 8:50 PM	OUT File	168 KB
Node_Acc.out	3/6/2013 8:50 PM	OUT File	144 KB
Gravity_Frc.out	3/6/2013 8:50 PM	OUT File	2 KB
Gravity_Dsp.out	3/6/2013 8:50 PM	OUT File	1 KB
Elmt_glbFrc.out	3/6/2013 8:50 PM	OUT File	381 KB
Control_daqFrc.out	3/6/2013 8:50 PM	OUT File	63 KB
Control_daqDsp.out	3/6/2013 8:50 PM	OUT File	61 KB
Control_ctrlDsp.out	3/6/2013 8:50 PM	OUT File	61 KB
SACNF01.txt	10/13/2010 10:47 ...	Notepad++ Document	78 KB
elapsedTime.txt	3/6/2013 8:50 PM	Notepad++ Document	79 KB
PlotOutput.m	3/6/2013 8:59 PM	MATLAB Code	9 KB

Compare Deformations

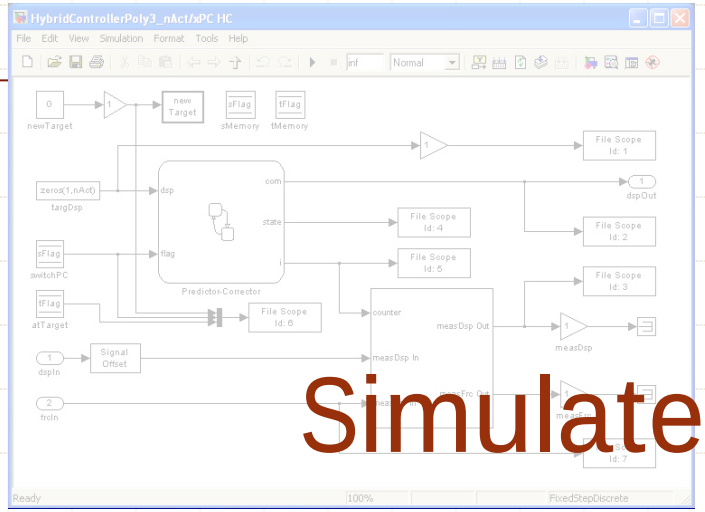


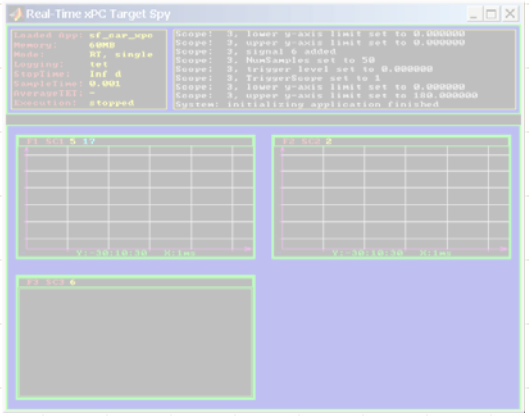
Compare Hysteresis Loops



w/o Gravity

with Gravity

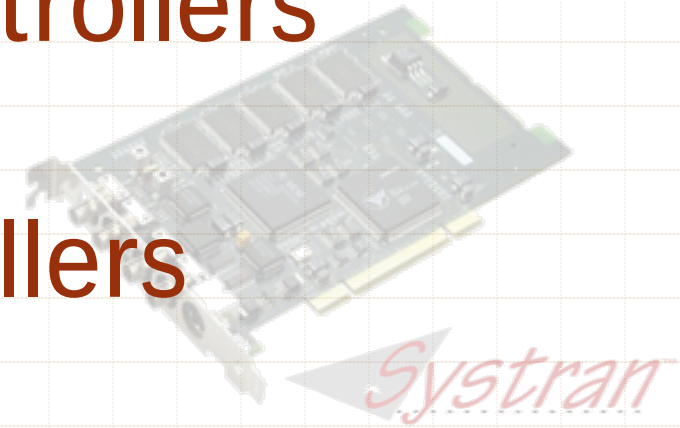


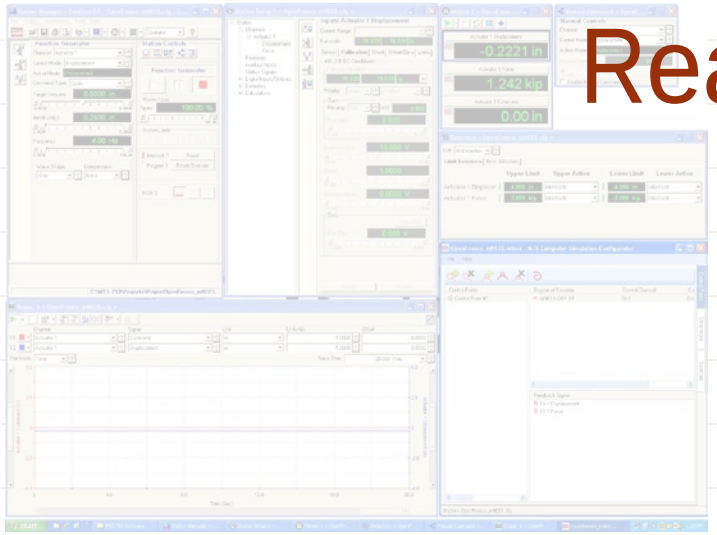


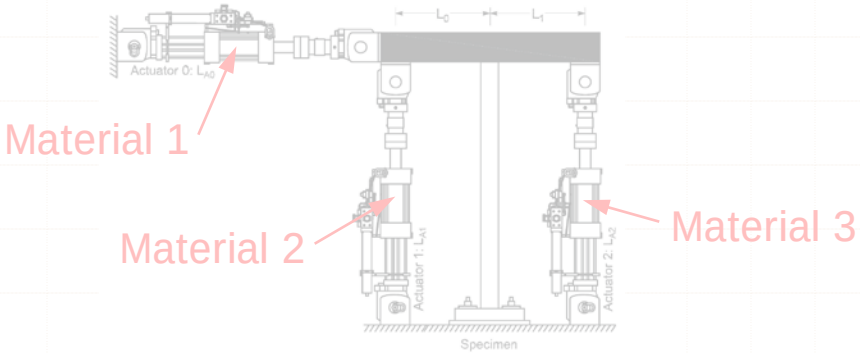
Simulated Controllers

vs.

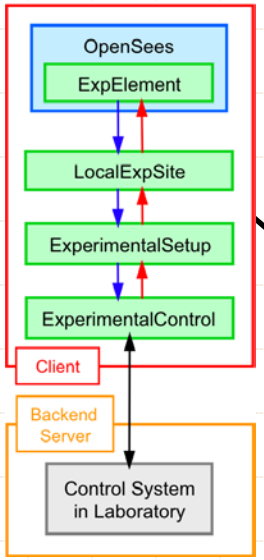
Real Controllers







Connecting to MTS 493 controller



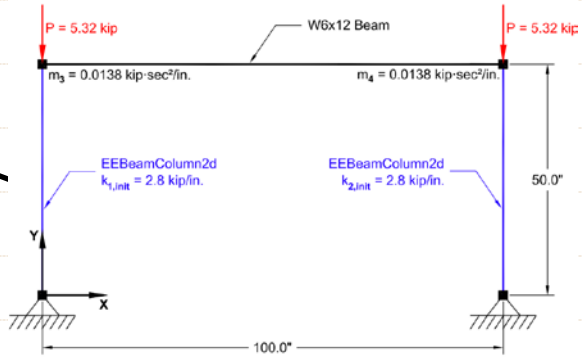
OpenSees Finite Element Model

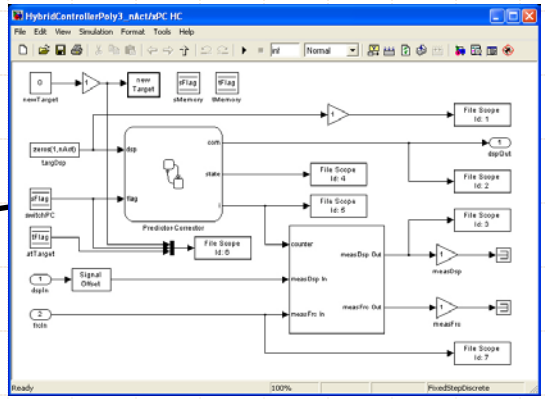
OpenFresco Middleware

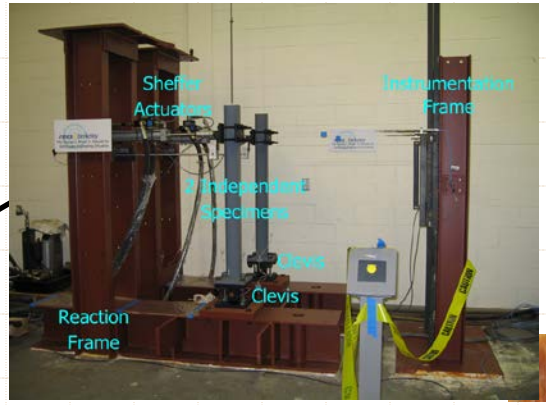
xPC-Target real-time Predictor-Corrector

MTS 493 real-time Controller

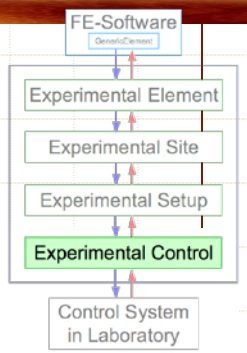
Physical Specimen in µNEES Lab





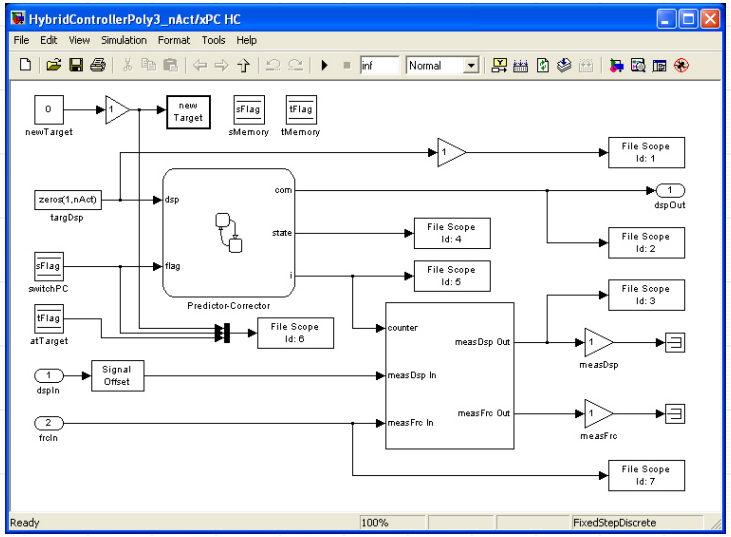


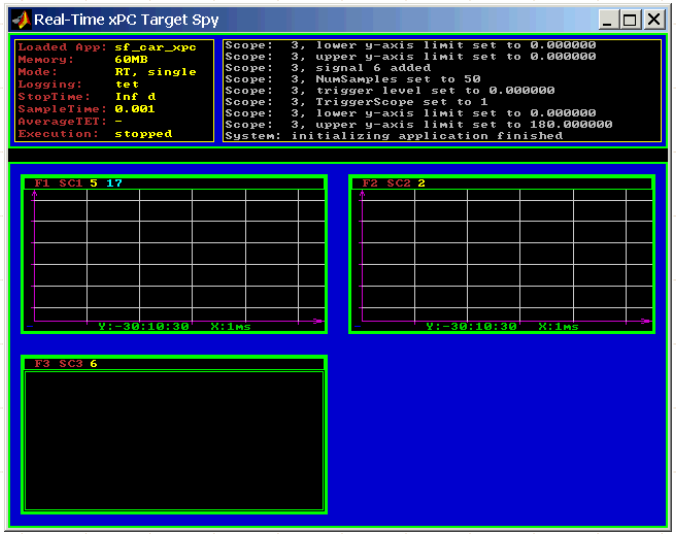
Modify Experimental Control




```
# Define experimental control
# ---------------------------
# expControl SimUniaxialMaterials $tag $matTags
#expControl SimUniaxialMaterials 1 1
expControl xPCtarget 1 1 "192.168.2.20" 22222 HybridControllerD3D3_1Act "D:/PredictorCorrector/
expControl SimUniaxialMaterials 2 1
```

- Want to control two columns
- xPCtarget used for left column
- SimUniaxialMaterials used to simulate right column





MATLAB SIMULINK

OpenSees

Computational Drivers

ABAQUS

LSTC Livermore Software Technology Corp.

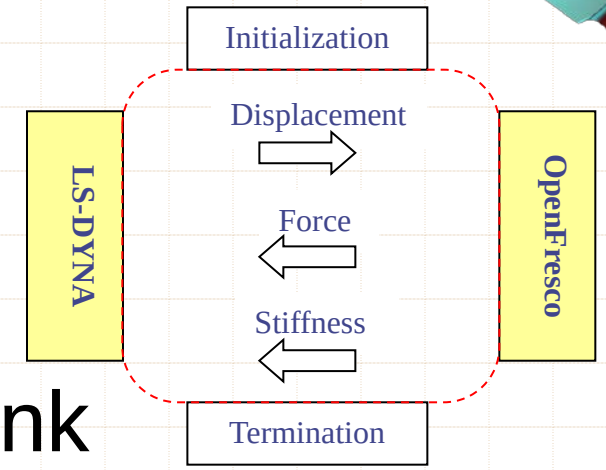
ANSYS®

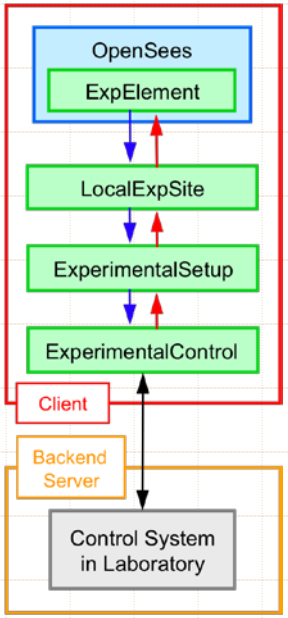
How to Interface

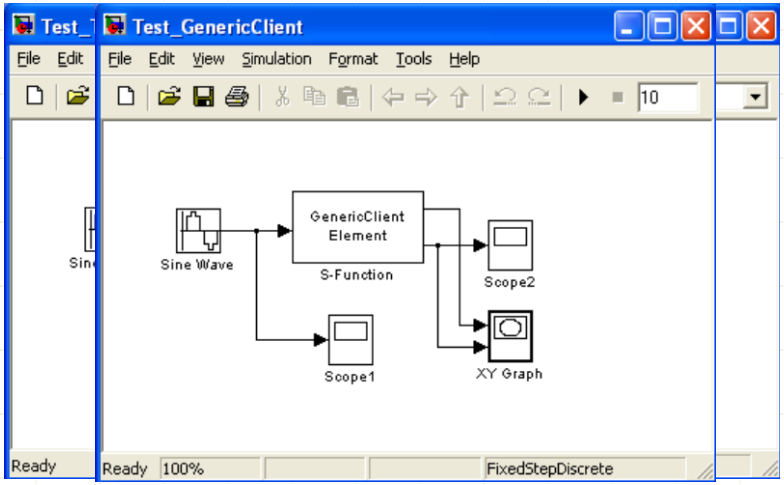
- Two Ways to Interface with FE-Software
  - Generic Client Element
  - Experimental Element Directly in FE-Software
- Generic Client Element to be Programmed by the Developers
- Several generic client elements available: /trunk/SRC/simApplicationClient

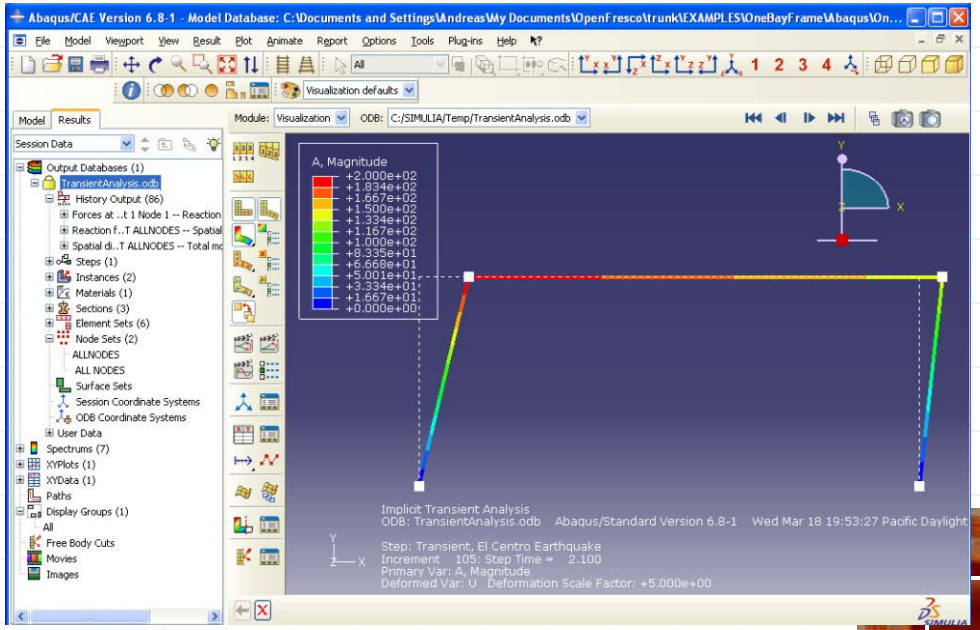
Computational Drivers

- OpenSees
- LS-DYNA
- Abaqus
- Matlab/Simulink
- UI-SimCor

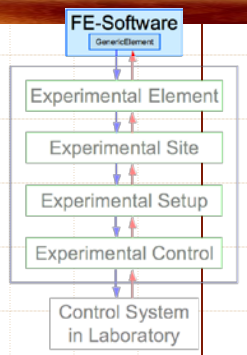








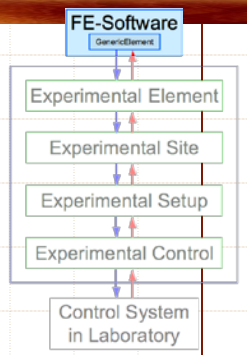
Integration Methods



$$\mathbf{M} \cdot \ddot{\mathbf{u}}_{\mathbf{n}} + \mathbf{C} \cdot \dot{\mathbf{u}}_{\mathbf{n}} + \mathbf{P}_{\mathbf{r}}\left(\mathbf{u}_{\mathbf{n}}\right) = \mathbf{P}\left(t_n\right)$$

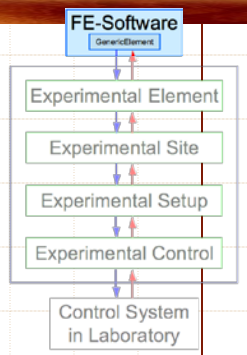
- Mass matrix M is often singular
  -> second order differential equation
  infinitely stiff -> fully implicit numerical methods
- Make as few function calls as possible
- Use constant Jacobian in the numerical methods since tangent stiffness is not available

Direct Integration Methods



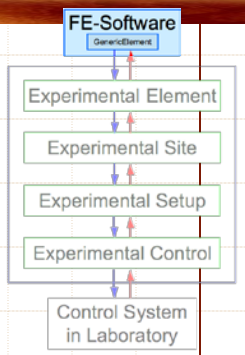
- Explicit Integrators
  - explicit Newmark Method
  - Central-Difference Method
  - explicit Alpha Method
  - explicit Generalized-Alpha Method
- Implicit Integrators (do not use for HS)
  - Newmark Method
  - Alpha Method
  - Generalized-Alpha Method
  - Collocation Method

Direct Integration Methods

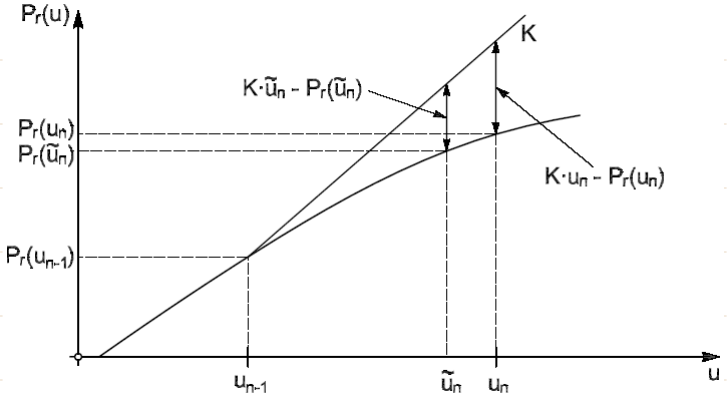


- Implicit Integrators with increment reduction factors
  - Newmark HS IncrReduct Method
  - Generalized-Alpha HS IncrReduct Method
  - Collocation HS IncrReduct Method
- Implicit Integrators with increment limits
  - Newmark HS IncrLimit Method
  - Generalized-Alpha HS IncrLimit Method
  - Collocation HS IncrLimit Method

Direct Integration Methods

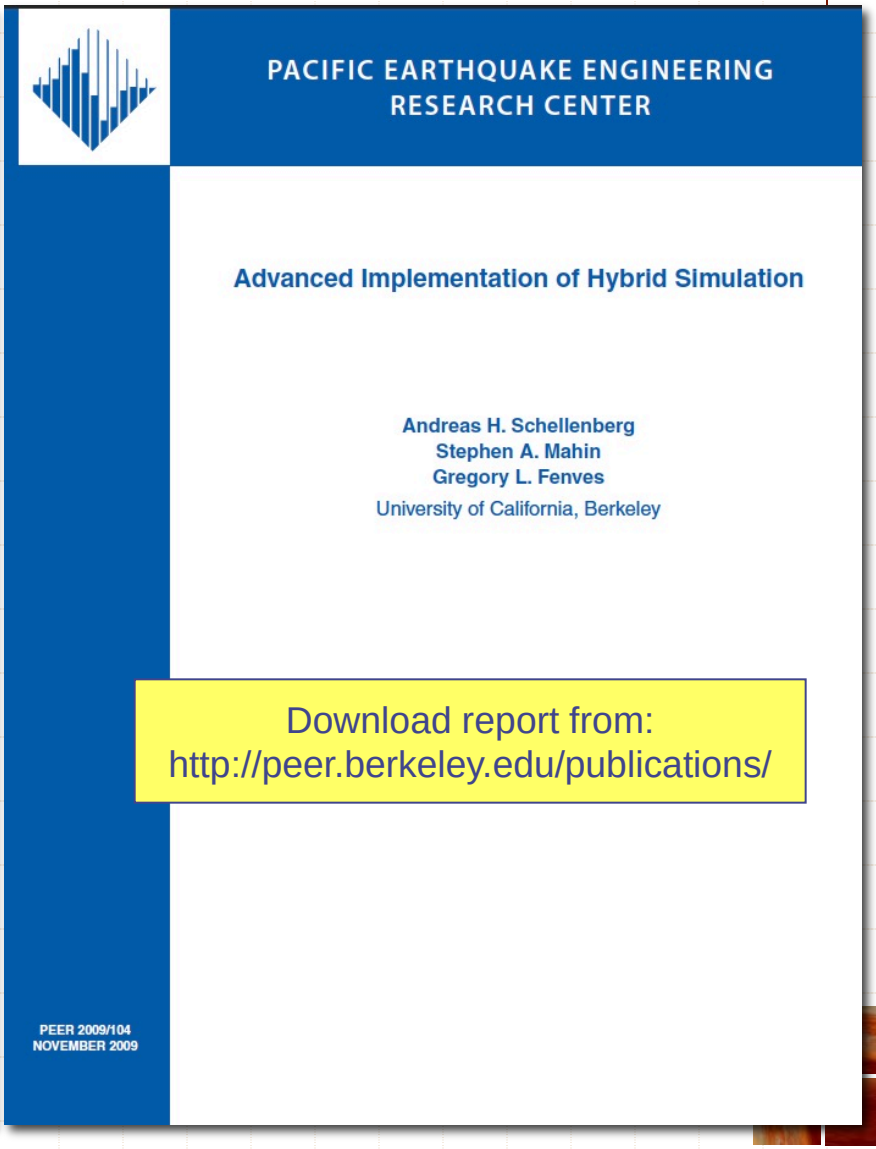


- Implicit Integrators with sub-stepping (constant number)
  - Newmark HS FixedNumIter Method
  - Generalized-Alpha HS FixedNumIter Method
  - Collocation HS FixedNumIter Method
- Predictor-Corrector Integrators
  - Alpha-OS Method
  - Generalized-Alpha-OS Method



More Information

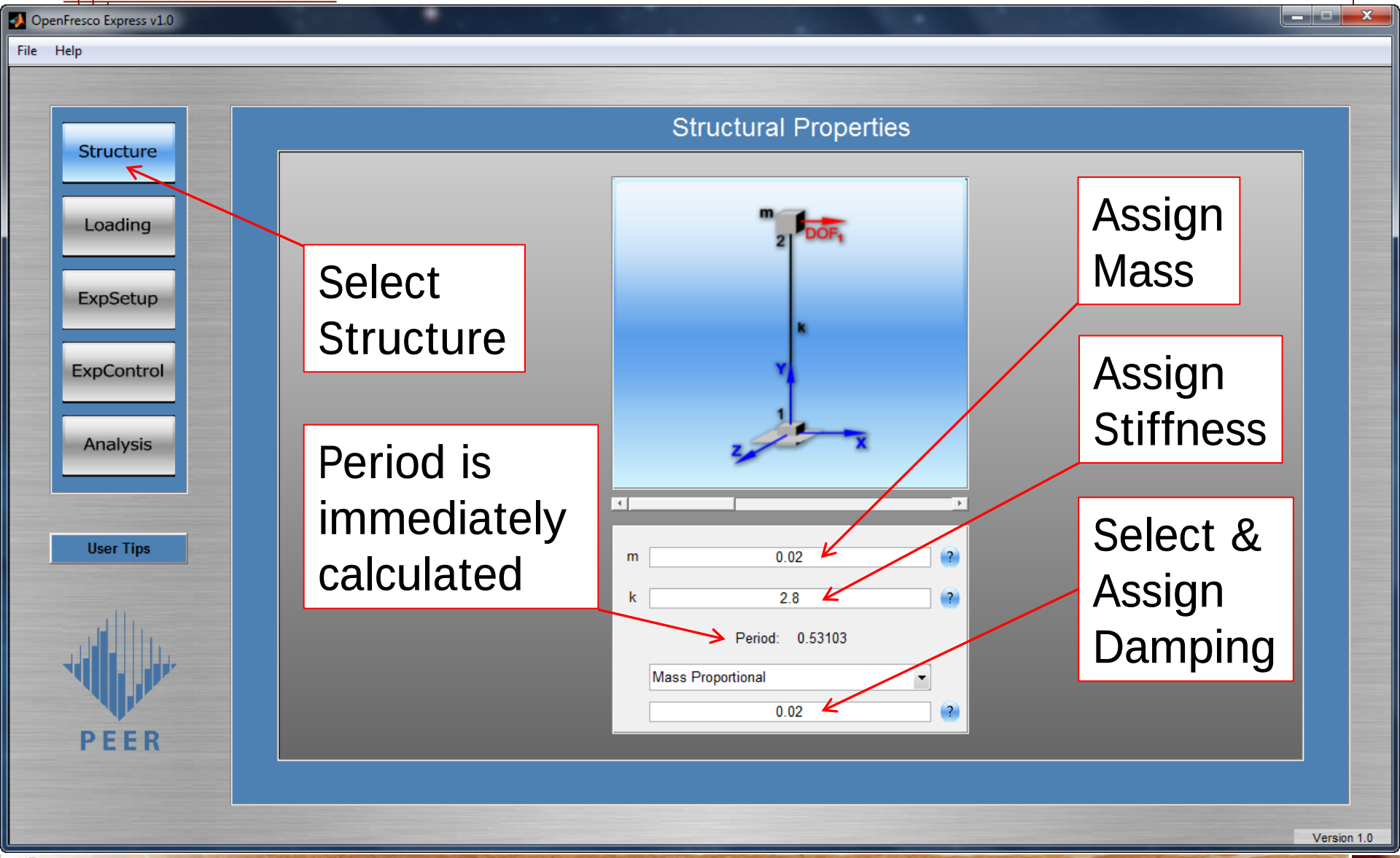
Download documentation and software:
http://openfresco.berkeley.edu



Download report from:
http://peer.berkeley.edu/publications/

GUI: OpenFrescoExpress

GUI: OpenFresco*Express*



Summary & Conclusions

- Hybrid simulation inherently requires close collaboration amongst experts from many different fields.
  - Structural behavior
  - Laboratory testing and control
  - Computational simulation
  - Information technology
- Hence, hybrid simulation fosters collaboration and communication among distant researchers in different labs.

Summary & Conclusions

- OpenFresco, the environment-independent software framework for the development and deployment provides an excellent platform for this collaboration (on-site and geographically distributed)
- The modularity and transparency of the framework permits existing components to be modified and new components to be added without much dependence on other objects.

Summary & Conclusions

- Large libraries of hybrid simulation direct integration methods, experimental elements, experimental setups, controller models, and event-driven solution strategies are available to the researchers to choose or adapt from.
- Needs:
  - User feedback on refinements and new features
  - Developer contributions to extend libraries

Questions?
Thank you!

http://openfresco.berkeley.edu/

The development of OpenFresco has been sponsored in parts by the National Science Foundation through grants from the NEES Consortium, Inc.