Develop your own OpenSees

Architecture & Components

SiF Workshop: OpenSees for fire, 3 Dec 2020

Presented by **Dr Liming Jiang**

Assistant professor

Research Center for Fire Safety Engineering

Department of Building Services Engineering

The Hong Kong Polytechnic University

Special thanks to:
Frank McKenna (University of California, Berkeley)

THE UNIVERSITY OF QUEENSLAND AUSTRALIA

THE HONG KONG POLYTECHNIC UNIVERSITY 香港理工大學

OpenSees

OUR ROADMAP

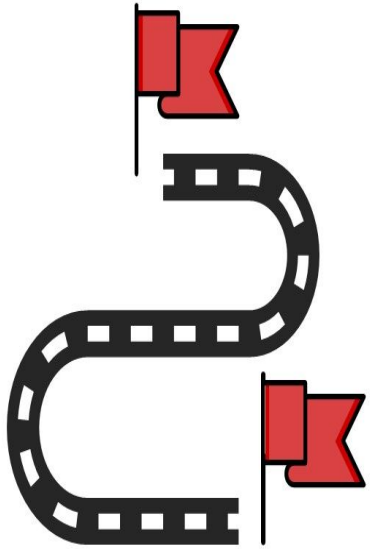
OPENSEES FOR FIRE ROADMAP

	By 2020	2020-2021	After 2021
Fire Model & Heat Transfer	• OpenFIRE (OpenSees-FDS Middleware) • Heat Transfer (HT) module -Tcl/Python Script - Idealised uniform fire models (standard, parametric) -Idealised non-uniform fire models (localised, travelling fires)	• OpenFIRE (OpenSees-FireFoam Middleware) • Travelling fire models -considering travelling mechanism • HT sections for composite column • HT material for timber sections	• AI enhanced fire model -predicting fire behaviour
Thermo-mechanical Analysis	• Frame members in fire -TM BeamColumn elements (Disp&Force based) -Fibre based TM sections -Uniaxial materials (concrete& steel) • Slabs in fire -TM Shell elements (ShellMITC4Thermal & ShellNLDKGQThermal) -Layered shell section -TM multiaxial material PlateRebarThermal ConcreteDamagePlasticity • Beam-Column Joint in fire	• 3D thermo-mechanical solid elements -Continuum elements • Integrated model for composite floor in fire -Rib section -Efficient model	• Large 'structure in fire' model • Two-way interaction (Fire-Structure) --Non-structural components in fire -- Structural deflection
Struc in Fire Application	• SIFBuilder (Integrated Structure in fire simulation tool) • GiD-OpenSees interface for SiF analyses • Hybrid Simulation Testing for Structures in Fire	• Tall Building Collapse in fire case studies (Plasco, WTC7) • Python based GUI pre-processor	• Post-processors for 'structure in fire' simulation
Algorithm & Solution	• Static analysis - Time step, fire duration	• Auto removal solution for failed elements • Arc-length solution	• Static-dynamic solution

openseesforfire.github.io @Hong Kong PolyU

OPENSEES WORKSHOP

1. OpenSees for fire website
2. OpenSees Framework
3. Build OpenSees
4. How OpenSees works
5. Interface of material classes
6. Manage a project
7. A summary of OpenSees for Fire



OPENSEES WORKSHOP

Part 1: OpenSees for Fire Website

Github Website

View the Project on GitHub
OpenSees@GitHub

View People
OpenSees for Fire Group

Go To Berkeley | Download OpenSees | View On GitHub

ResearchGate

OpenSees

SFPE Hong Kong Student Chapter

About

The OpenSees development for modelling 'structures in fire' was first started at University of Edinburgh in 2009. A number of students and researchers worked on this long-term project with their own contributions which enable OpenSees to perform analyses for 'structure in fire' including heat transfer ,thermo-mechanical analyses, and integrated analyses. [See it on Researchgate Project] [OpSees for fire Roadmap]

Users (command|examples)

For Users

A number of web pages are constructed to offer the users a detailed guidance to the recently added capabilities within OpenSees

Developers

For Developers

A detailed description of all the new or modified classes developed for enabling thermomehcanical analyses in OpenSees.

Documents

Relevent publications to OpenSees for fire development can be found from the above section

Download

The executable file for OpenSees for fire can be downloaded on this page. The version is updated frequently

Q&A

If you encounter any problems when using OpenSees for fire,please leave your messages on the "issues" board.

News Update

Here you can find the recent news in our research team and some key events.

-Related links-

HK PolyU

This project is maintained by [illegible]
Hosted on GitHub Pages, 2017

http://openseesforfire.github.io

Github Website

Back Home | Download OpenSees | View On GitHub

Download

- About
- People
- Users
  - Getting Started
  - Heat Transfer
  - SIFBuilder
  - Command manual
  - Examples
- Developers
- Documents
- Q&A

This project is maintained by JIANG Liming
Hosted on GitHub Pages, 2017

For Users

User Documentation at Main OpenSees Site

It is important for all users who intend to use the thermal version of OpenSees to first read the user manuals on OpenSees Berkley Site. A pdf version of the step-by-step instructions on downloading,installing and using OpenSees can be downloaded here.

Getting Started Manual Command Manual

User Documentation for OpenSees Fire

The following documents are specially developed for OpenSees for fire.

Getting Started with OpenSeesforFire

This documentation provides a basic understanding on how to evaluate the structural response in fire using OpenSeesThermal. Relevent Tcl Commands are illustrated by several examples.

Command Manual for OpenSeesThermal

Command Manual

This manual provides a list of all new commands along with its attributes.

OpenSees for fire Examples

Examples

Examples are presented to demonstrate the thermomechanical analyses using the thermal version of OpenSees Framework. These are in addition to the simple example discussed in Getting Started with OpenSees Thermal.

Projects of OpenSees for Fire

Heat Transfer in OpenSees

OpenSees now has been extended to heat transfer analysis, which is based on a simple mesh tool with particular emphasis on heat transfer to structural members from fire.

SIFBuilder Project

SIFBuilder is developed as an computational tool to perfomr integrated analysis of structures in fire

KSHOP

ub.io

OPENSEES FOR FIRE

Download Exectuable for Windows

The latest version of OpenSees for fire executable file can be downloaded here (Base version 3.0). To use it, please refer to the latest user manual.

Download OpenSeesforFire.exe (Updated on 7 July 2020)

Download Tcl/TK

OpenSees uses the Tcl interpreter which has been extended with model builder commands. Before using OpenSees, the Tcl library (8.6 version) should be installed and the installation file can be found from this site:

Activate Tcl Website

It should be noticed that the Tcl/tk is recommanded to be installed into C:/Program files/Tcl (NOTE: The current version only supports the 64bit version of Tcl. 32bit version has to be compiled and set up by the user himself)

Download Python

OpenSees has been extended for Python intepreter and can be run as a python application. Due to the inclusion of Python, the original package (Python 3 version) should be installed and the installation file can be found from this site:

Official Python Website

It should be noticed that the Python is recommanded to be installed into C:/Program files/Python (NOTE: The current version only supports the 64bit version)

Browse Source Code

Source code on GitHub

To browse our version of source code, you could simply go to our Github page or click the "view on github" link here(Remeber to change the branch to OpenSees_SRC)

Download source code package

You can download the source code package through the link shown as below, which is based on the latest version(main release) 3.0.0, and the build environment is suggested to be Visual Studio 2019 64bit version, Tcl 8.6 64bit, and Python 3.7.

Download source code package:OpenSees3_for_fire.zip

Development Environment

Source Codes

OpenSees-SRC | 3 branches | 0 tags | Go to file | Add file | Code

This branch is 27 commits ahead, 462 commits behind master. | Pull request | Compare

LimingXLiming update for timber — 00fa852 on Aug 17 — 29 commits

OTHER	update	11 months ago
SRC	update for timber	4 months ago
Win32	change directories	2 years ago
Win64	updateHT	5 months ago
.gitattributes	update	11 months ago
.gitignore	Update .gitignore	5 months ago
README.md	Update README.md	11 months ago

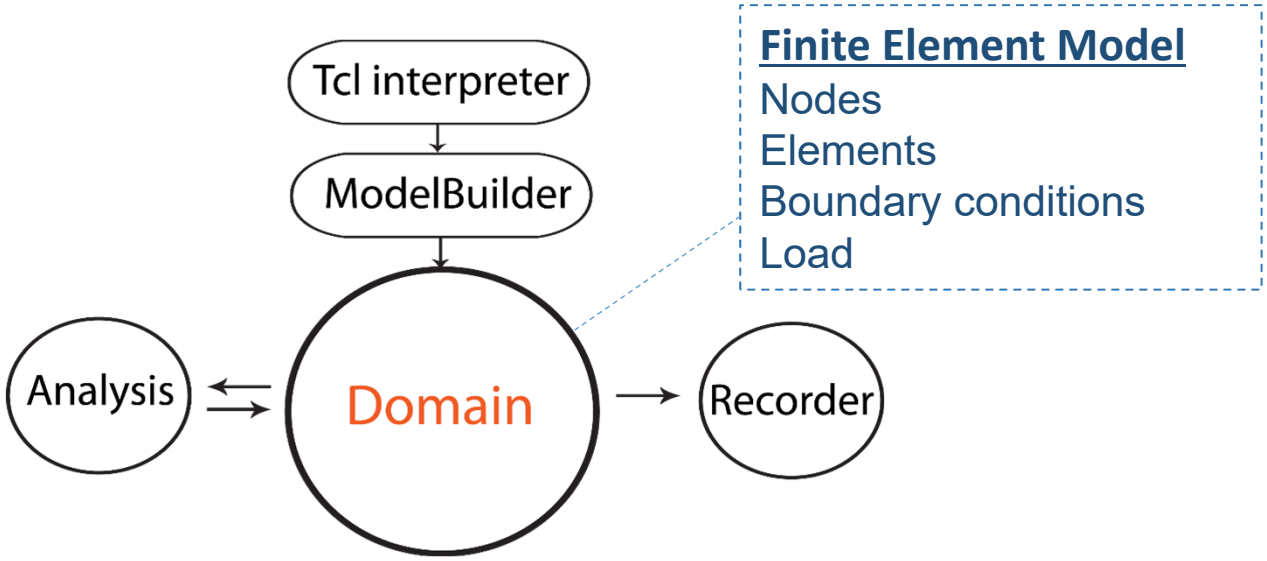
README.md

OPENSEES WORKSHOP

Part 2: OpenSees Framework

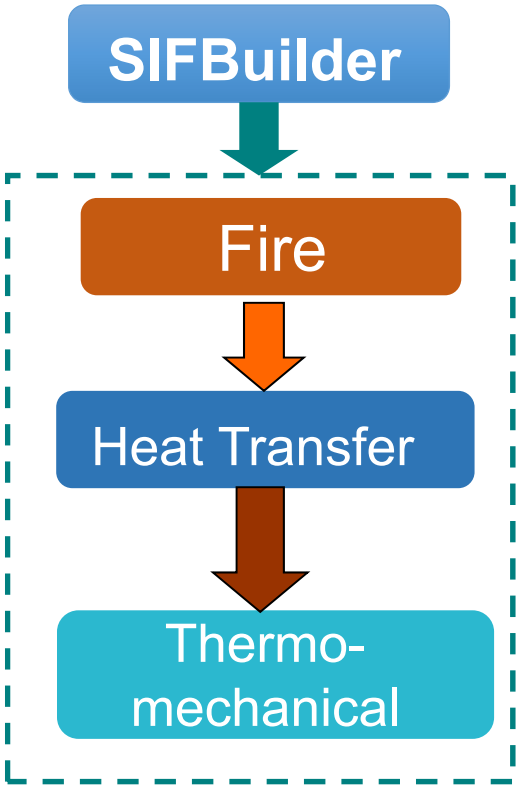
OpenSees Framework

- A framework is **NOT an executable**;
- It is a set of cooperating software components for building applications in a specific domain**;**
- It is a collection of **abstract and derived** classes;
- **Loose-coupling** of components within the framework is essential for extensibility and re-usability of the applications



OpenSees for Fire

- Started at Edinburgh University since 2009;
- Based on a group of PhD students' work**;**
- Developed for modelling '**Structures in Fire**';



User-friendly interface for creating (regular) structural models and enable consideration of realistic fire action

Models of fire action (only ***idealised*** fires), i.e., Standard fire, Parametric fire, EC1 Localised fire, Travelling fire

Heat transfer to the structural members due to fire action

Structural response to the elevated temperatures

OpenSees Framework

31 Projects in OpenSees

actor | cblas | convergence | cssparse | damage | database

graph | handler | matrix | modelbuilder | OpenSees | OpenSeesTk

OpenSeesTk | Optimization | quickMain | reliability | renderer

string | superLU | system | tagged | utility

---

analysis | domain | element | material | tcl | recorder

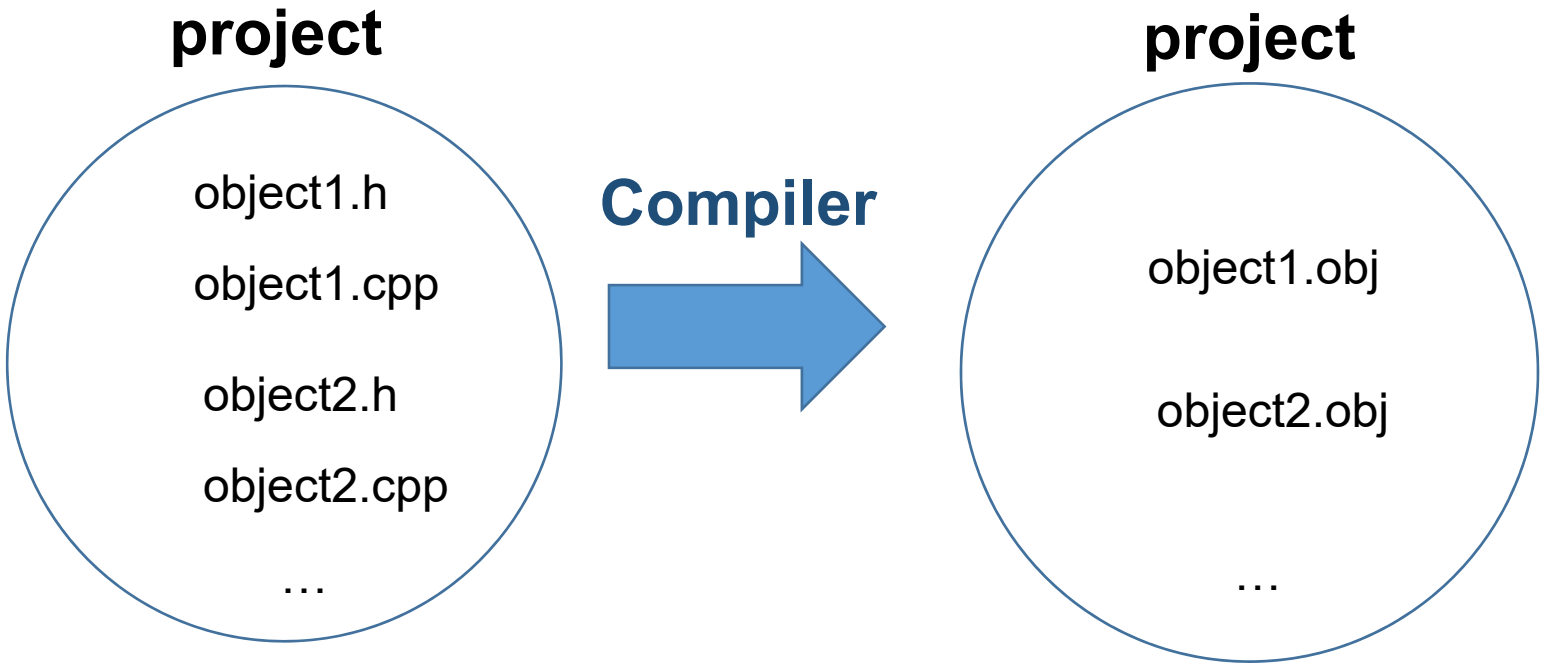
HeatTransfer | SIFBuilder | HTMain | fire

OPENSEES WORKSHOP

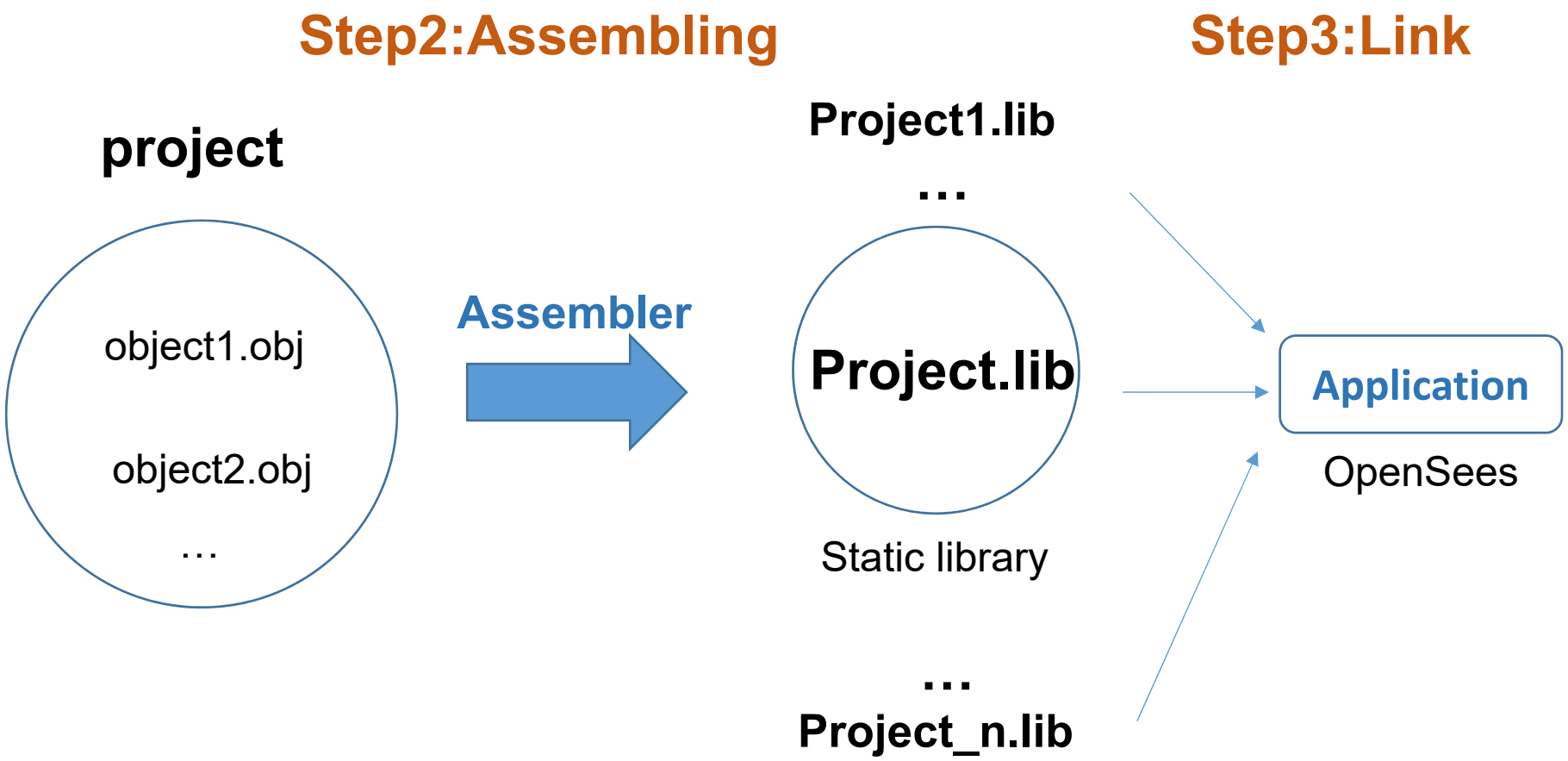
Part 3: Build OpenSees

Build OpenSees

Step1:Compilation

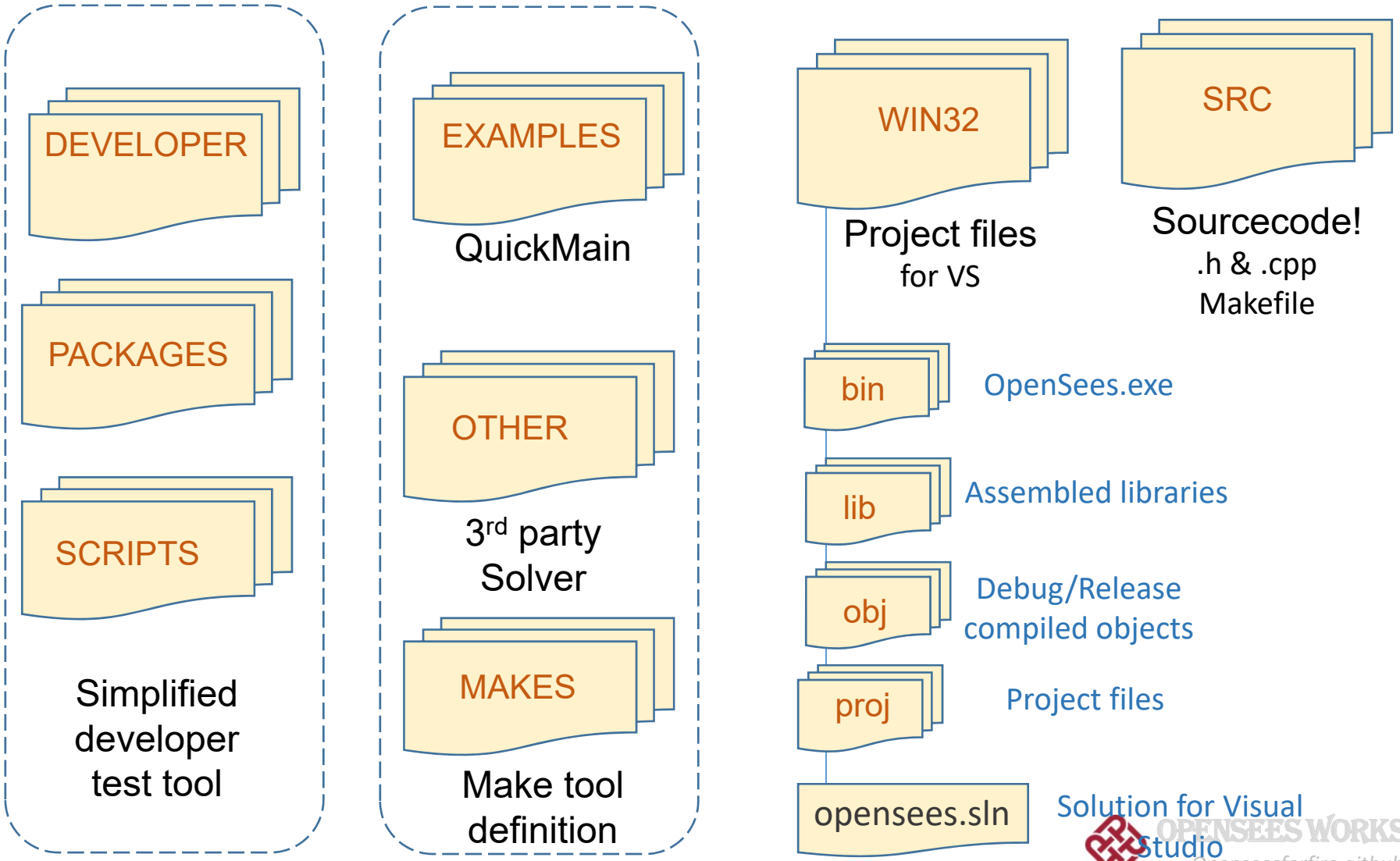


Build OpenSees



Build OpenSees

OpenSees Source Code Package

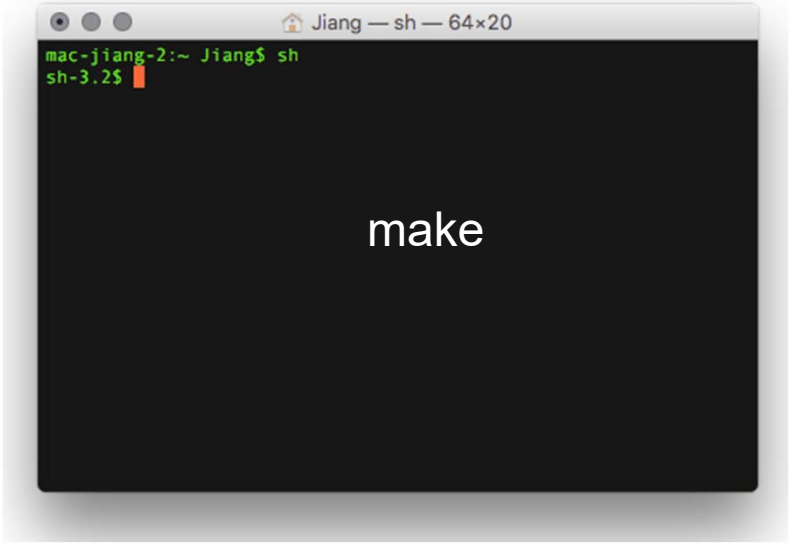


Build OpenSees

If you want to build it in Linux or MacOS?

GCC
&
GNU Make

Makefile.def + SRC + OTHER



Build OpenSees

GNU Make

GNU Make is a tool which controls the generation of executables and other non-source files of a program from the program's source files.

Make gets its knowledge of how to build your program from a file called the *makefile*, which lists each of the non-source files and how to compute it from other files. When you write a program, you should write a makefile for it, so that it is possible to use Make to build and install the program.

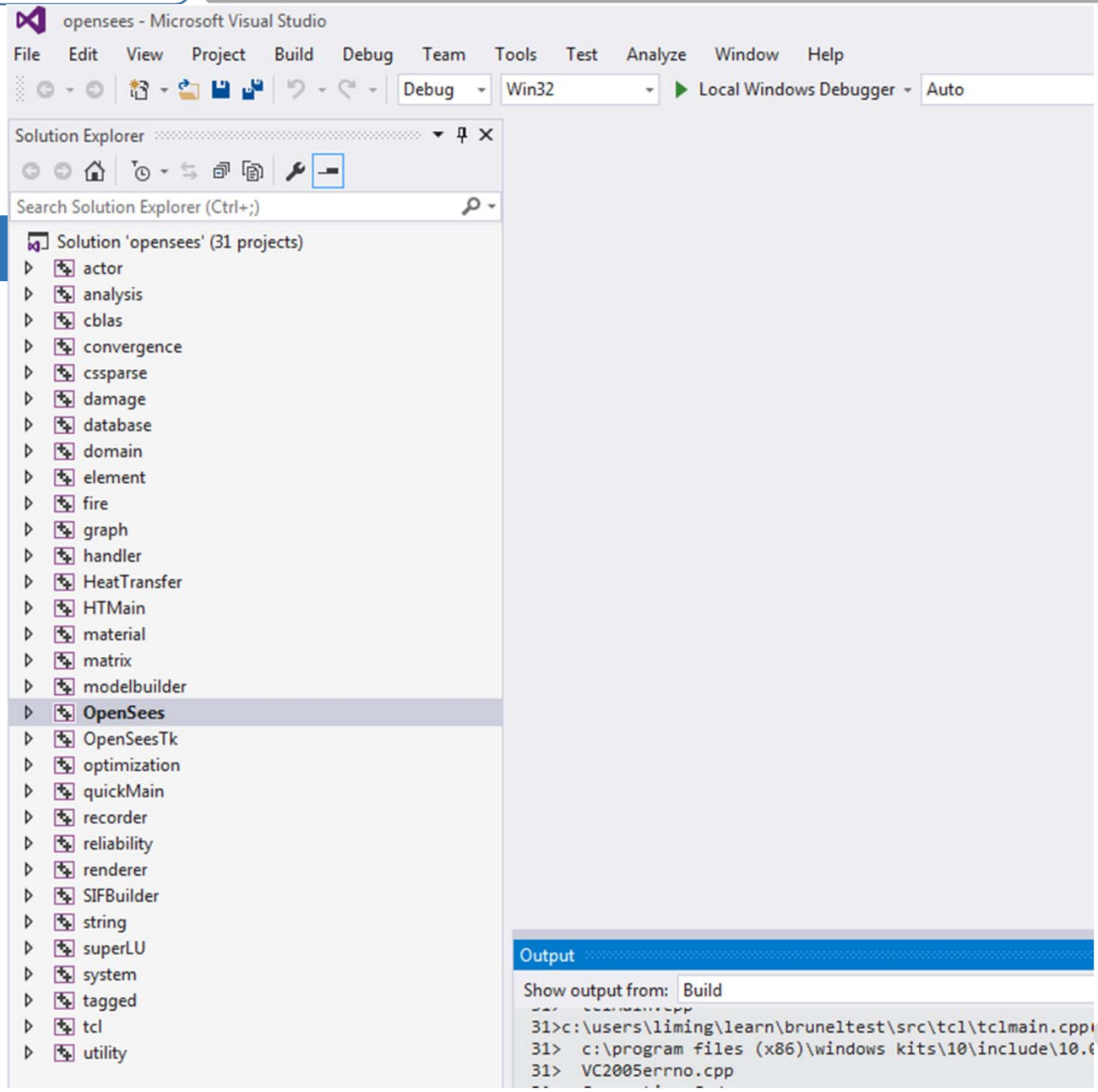
Makefile.def

- Program directory
- Paths (definition of SRC and OTHER directories)
- Libraries (definition of library location)
- Compilers (Compiler location & compiler and linker tags)
- Compilation behaviour
- Other supporting libraries
- Include files

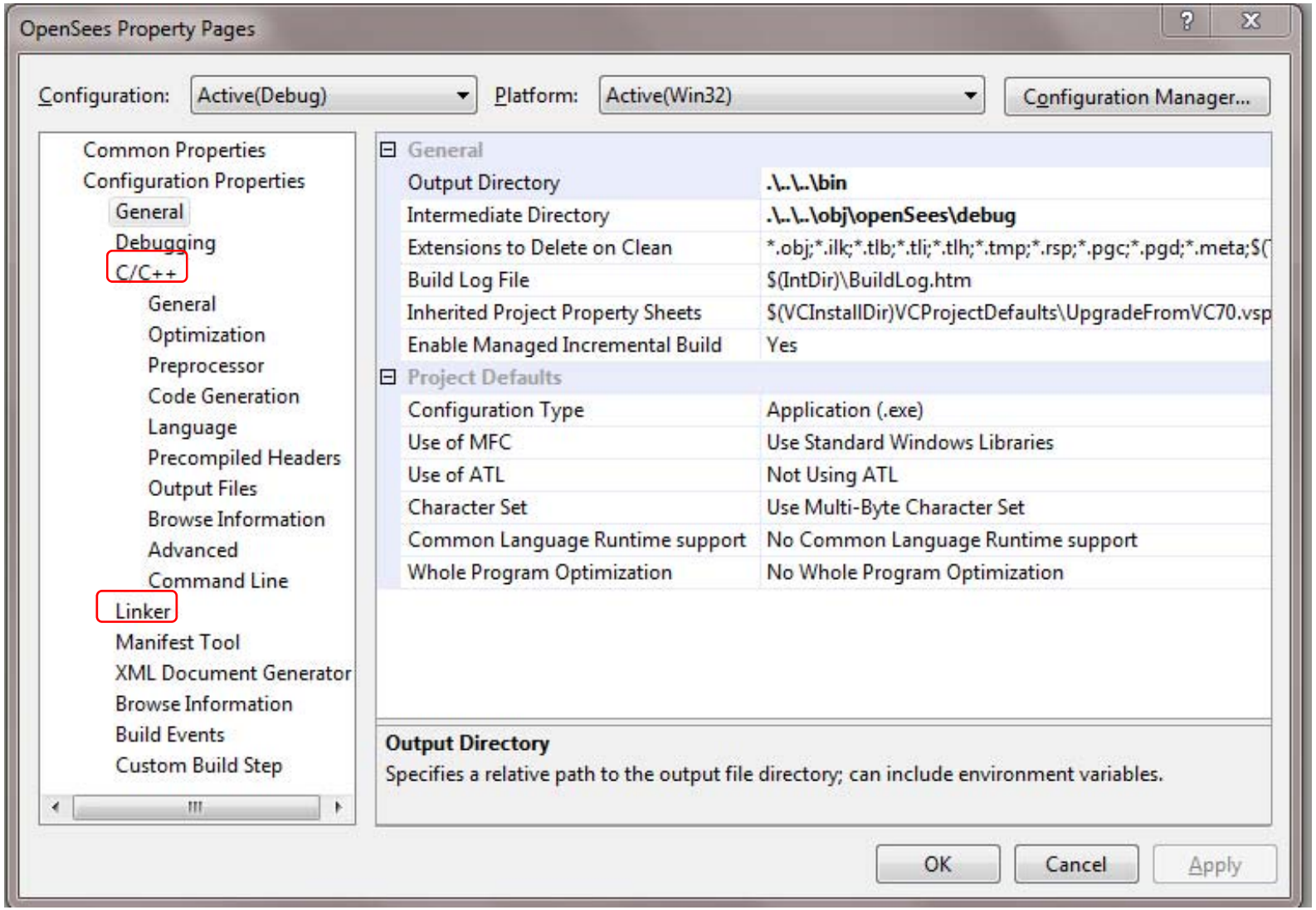
Build OpenSees

Using Windows PC

This is what **Visual Studio** looks like!



Build OpenSees



Give it a try to build your own OpenSees…

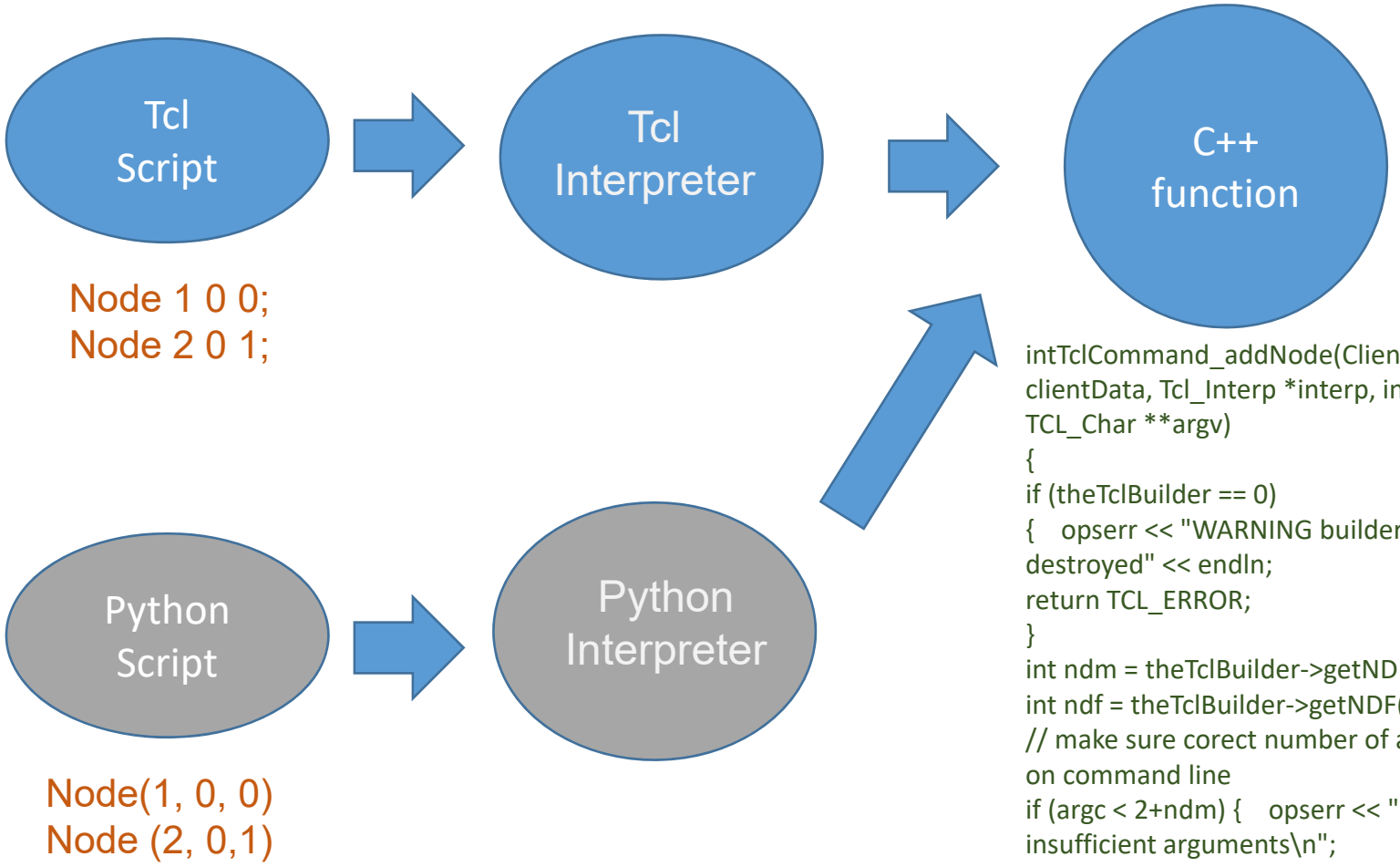
OPENSEES WORKSHOP

Part 4: How OpenSees Works

OpenSees Input Script

```
# ------------------------------------------------------------
# Example 1. portal frame in 2D
# static pushover analysis of Portal Frame, with gravity.
# all units are in kip, inch, second
# elasticBeamColumn ELEMENT
#       Silvia Mazzoni & Frank McKenna, 2006
# ------------------------------------------------------------

# SET UP ---------------------------------------------------------------------
wipe;                       # clear opensees model
model basic -ndm 2 -ndf 3;              # 2 dimensions, 3 dof per node
file mkdir data;                      # create data directory

# define GEOMETRY -----------------------------------------------------------
# nodal coordinates:
node 1 0 0;                 # node#, X Y
node 2 504 0
node 3 0 432
node 4 504 432

# Single point constraints -- Boundary Conditions
fix 1 1 1 1;            # node DX DY RZ
fix 2 1 1 1;            # node DX DY RZ
fix 3 0 0 0
fix 4 0 0 0

# nodal masses:
mass 3 5.18 0. 0.;                  # node#, Mx My Mz, Mass=Weight/g.
mass 4 5.18 0. 0.

# Define ELEMENTS ------------------------------------------------------------
# define geometric transformation: performs a linear geometric transformation of beam stiffness and resisting
force from the basic system to the global-coordinate system
geomTransf Linear 1;        # associate a tag to transformation

# connectivity: (make A very large, 10e6 times its actual value)
element elasticBeamColumn 1 1 3 3600000000 4227 1080000 1;  # element elasticBeamColumn $eleTag $iNode $jNode $A
$E $Iz $transfTag
element elasticBeamColumn 2 2 4 3600000000 4227 1080000 1
element elasticBeamColumn 3 3 4 5760000000 4227 4423680 1

# Define RECORDERS -----------------------------------------------------------
recorder Node -file Data/DFree.out -time -node 3 4 -dof 1 2 3 disp;         # displacements of free nodes
recorder Node -file Data/DBase.out -time -node 1 2 -dof 1 2 3 disp;         # displacements of support nodes
recorder Node -file Data/RBase.out -time -node 1 2 -dof 1 2 3 reaction;     # support reaction
recorder Drift -file Data/Drift.out -time -iNode 1 2 -jNode 3 4 -dof 1  -perpDirn 2 ;    # lateral drift
recorder Element -file Data/FCol.out -time -ele 1 2 globalForce;            # element forces -- column
recorder Element -file Data/FBeam.out -time -ele 3 globalForce;         # element forces -- beam

```

Inside OpenSees




```
intTclCommand_addNode(ClientData
clientData, Tcl_Interp *interp, int argc,
TCL_Char **argv)
{
if (theTclBuilder == 0)
{    opserr << "WARNING builder has been
destroyed" << endln;
return TCL_ERROR;
}
int ndm = theTclBuilder->getNDM();
int ndf = theTclBuilder->getNDF();
// make sure corect number of arguments
on command line
if (argc < 2+ndm) {    opserr << "WARNING
insufficient arguments\n";
printCommand(argc, argv);    opserr <<
"Want: node nodeTag? [ndm coordinates?]
<-mass [ndf values?]>\n";    return
TCL_ERROR;  }
```

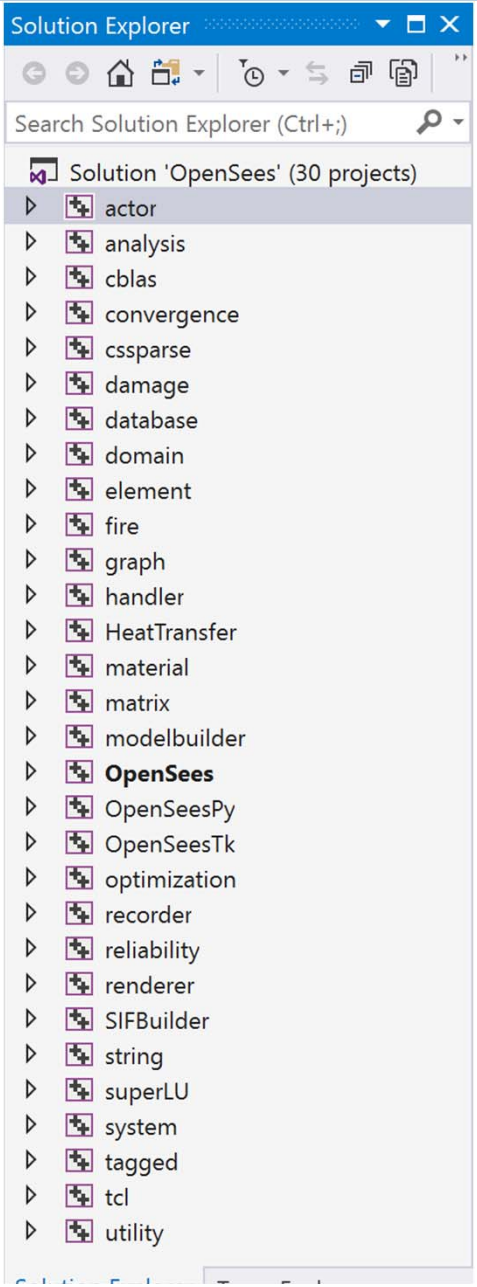
Inside OpenSees

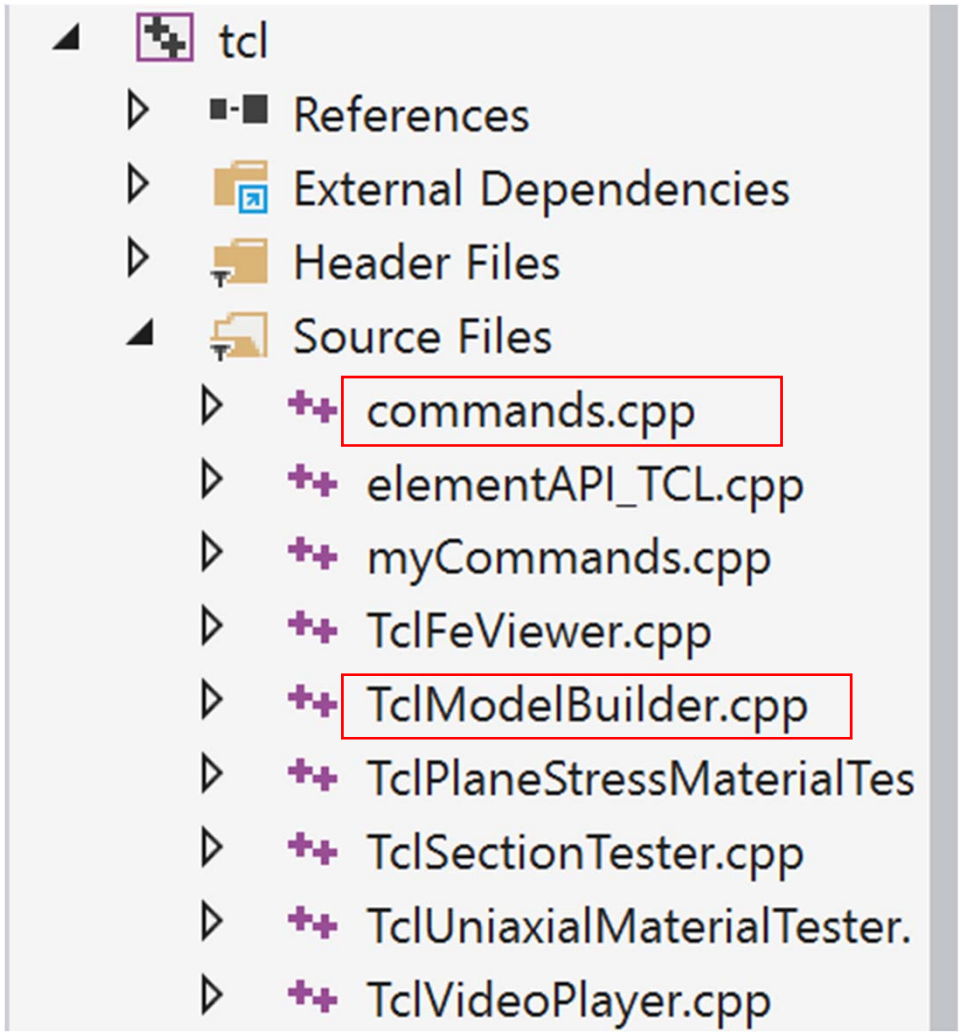
```
1  # ------------------------------------------------------------
2  # Example 1. portal frame in 2D
3  # static pushover analysis of Portal Frame, with gravity.
4  # all units are in kip, inch, second
5  # elasticBeamColumn ELEMENT
6  #       Silvia Mazzoni & Frank McKenna, 2006
7  # ------------------------------------------------------------
8
9  # SET UP ---------------------------------------------------------------------
10 wipe;                                 # clear opensees model
11 model basic -ndm 2 -ndf 3;                    # 2 dimensions, 3 dof per node
12 file mkdir data;                            # create data directory
13
14 # define GEOMETRY -----------------------------------------------------------
15 # nodal coordinates:
16 node 1 0 0;                           # node#, X Y
17 node 2 504 0
18 node 3 0 432
19 node 4 504 432
20
21 # Single point constraints -- Boundary Conditions
22 fix 1 1 1 1;                  # node DX DY RZ
23 fix 2 1 1 1;                  # node DX DY RZ
24 fix 3 0 0 0
25 fix 4 0 0 0
26
```

Global commands

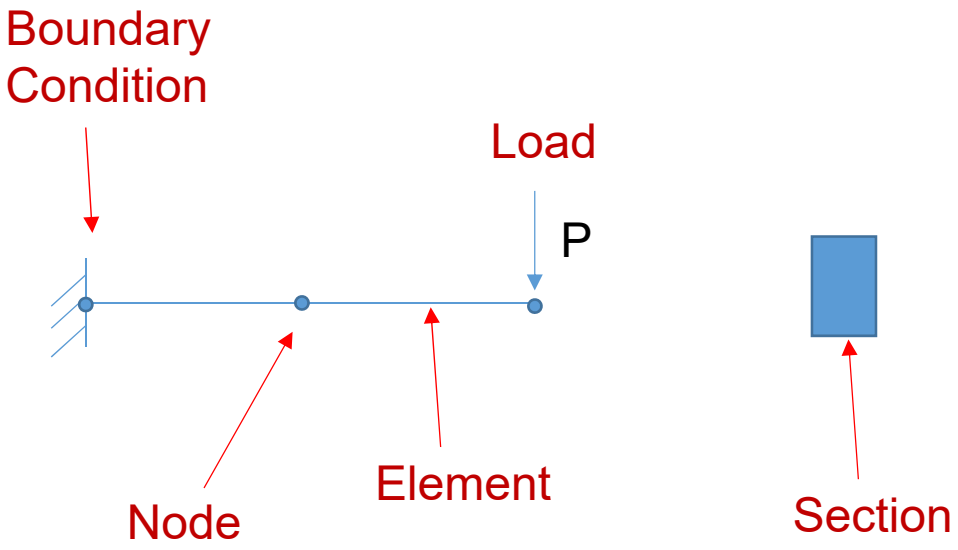
ModelBuilder commands

Inside OpenSees



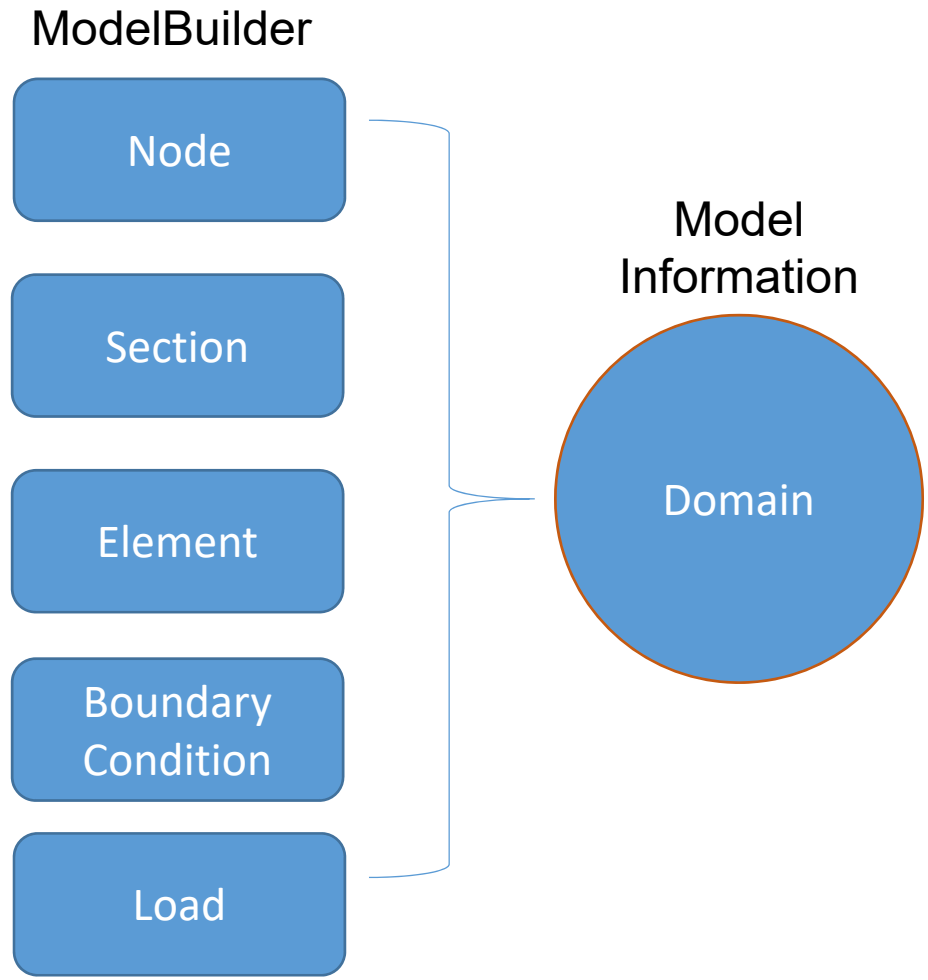


Model Builder



A simple model

Model Builder



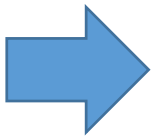
TclModelBuilder.cpp

```
 theNode = new
Node(nodeId,ndf,xLoc,yLoc);
…

if (theTclDomain->addNode(theNode)
== false) {
 opserr << "WARNING failed to add
node to the domain\n";

 opserr << "node: " << nodeId << endln;

delete theNode; // otherwise memory
leak    return TCL_ERROR;
 }
```

Run Analysis

Analysis Commands

```
constraints Plain;
numberer Plain;
system BandGeneral;
test NormUnbalance 1.0e-3 100 4;
algorithm Newton;
integrator LoadControl 0.005;
analysis Static;
analyze 200;
```



- Algorithm
- Solver
- Analysis type
- Convergence
- Integrator

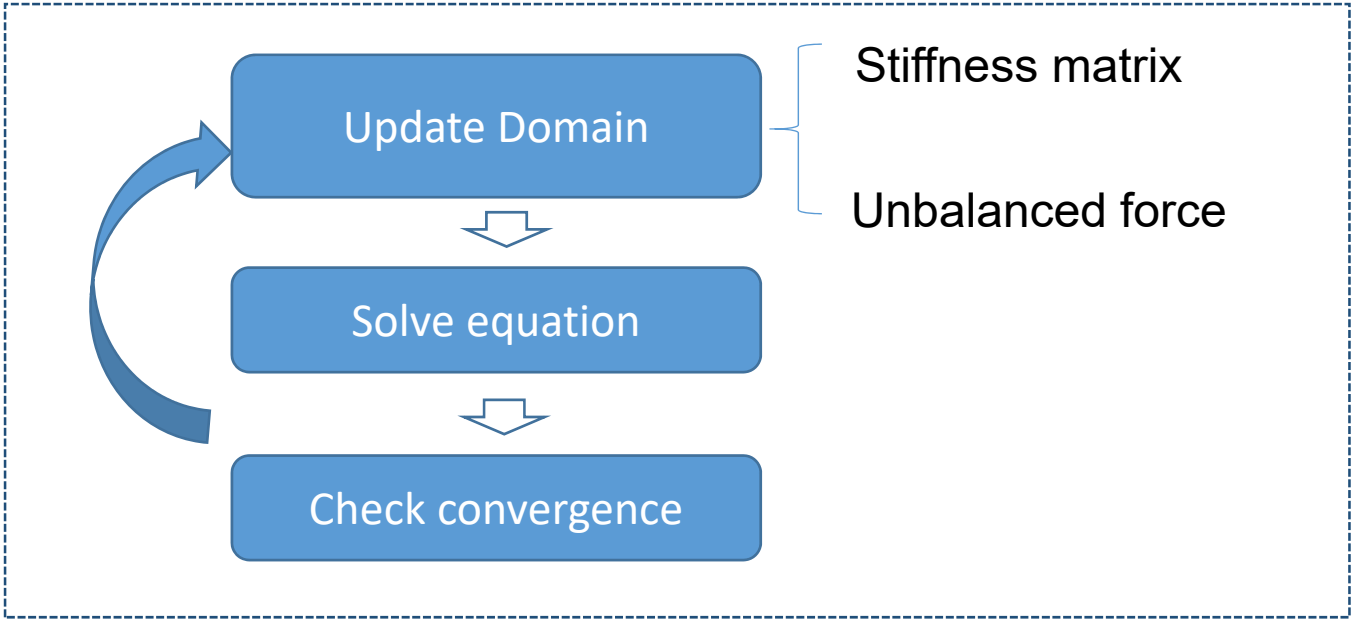
Run Analysis

Analyze command

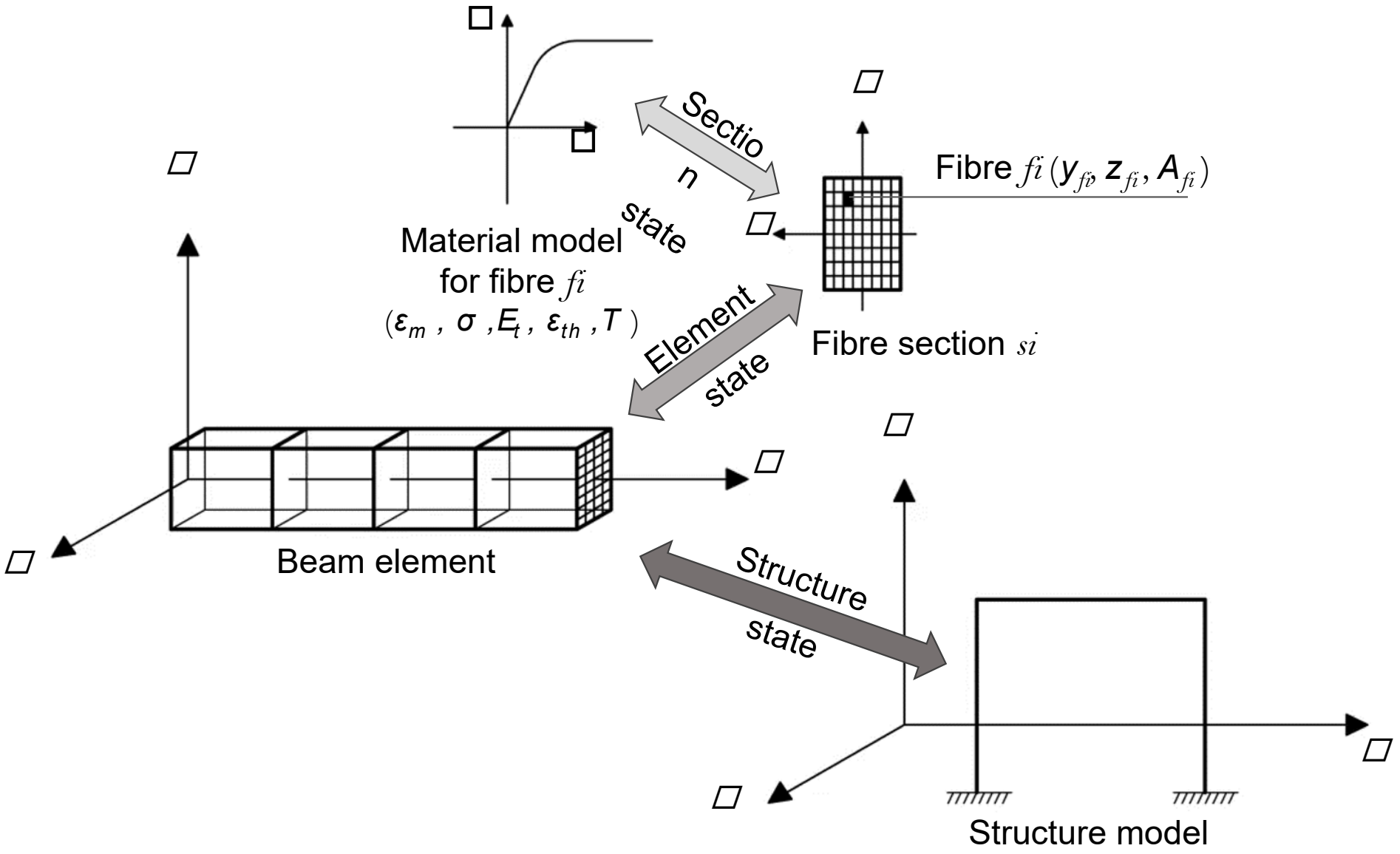
TclModelBuilder.cpp

```
Tcl_CreateCommand(interp, "analyze", &analyzeModel,
(ClientData)NULL, (Tcl_CmdDeleteProc *)NULL);
```

```
result = theStaticAnalysis->analyze(numIncr);
```



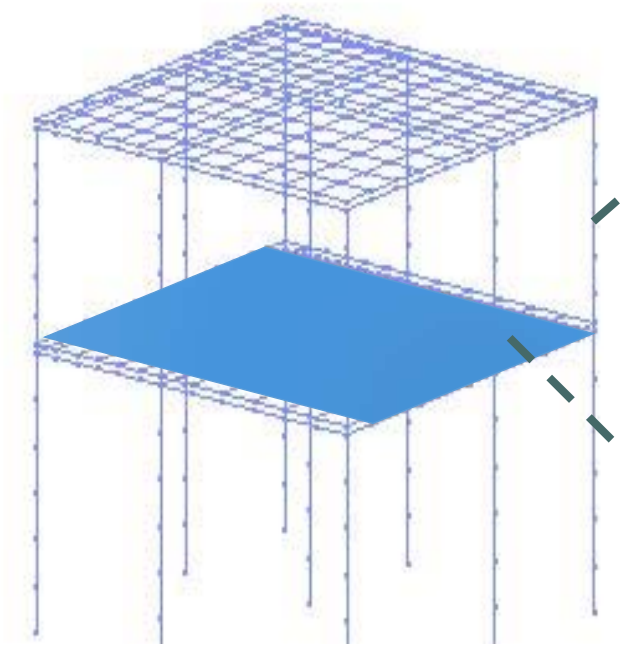
State Determination



OPENSEES WORKSHOP

Part 5: Interface of material classes

Material classes



- OpenSees FE model

Beams Columns

Uniaxial materials

- Fibre based sections
- Displacement based / force based

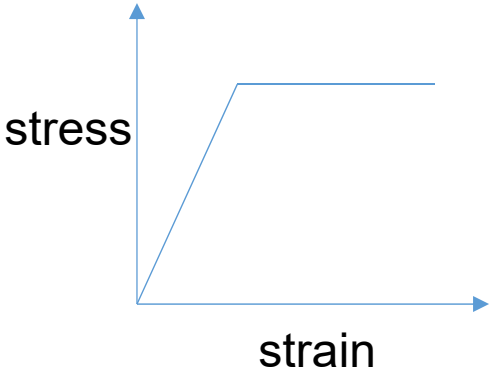
Slabs

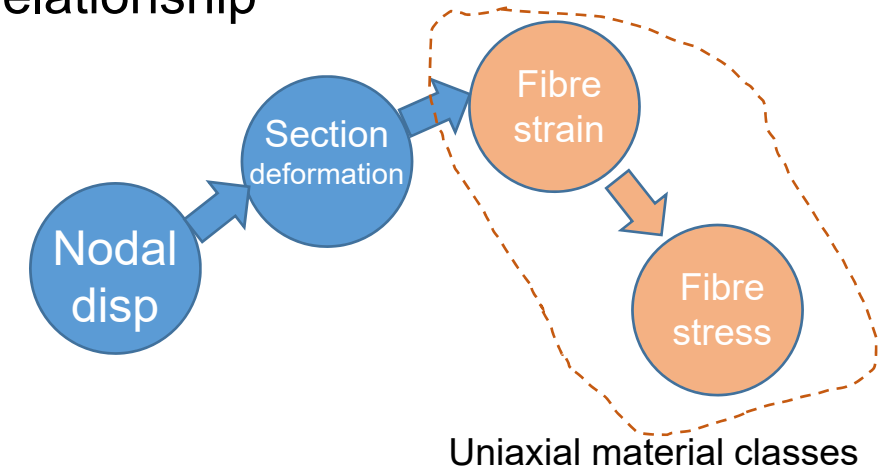
nD materials

- Shell elements
- Multi-axial materials
- Multi-layered plate sections

Uniaxial material class

- A material model: Stress-strain relationship





- Where to find the uniaixal material classes:

OpenSees/SRC/material/uniaxial

Uniaxial Material

uniaxialMaterial ElasticMaterial User script

TclCommand_addUniaxialMaterial TclModelBuilder.cpp

TclModelBuilderUniaxialMaterialCommand TclModelBuilderUniaxialMaterialCommand.cpp

ElasticMaterial.cpp and ElasticMaterial.h.

Standard interface of uniaxial materials?

- strain
- stress
- Tangent stiffness

Header file contains declaration of all class functions

Uniaxial Material

```
class UniaxialMaterial : public Material
{
  public:
    UniaxialMaterial (int tag, int classTag);
    virtual ~UniaxialMaterial();

    virtual int setTrialStrain (double strain, double strainRate =0) =0;
    virtual int setTrialStrain (double strain, double temperature, double strainRate);
    virtual int setTrial (double strain, double &stress, double &tangent, double strainRate = 0.0);
    virtual int setTrial (double strain, double temperature, double &stress, double &tangent, double &thermalElongation, double s

    virtual double getStrain (void) = 0;
    virtual double getStrainRate (void);
    virtual double getStress (void) = 0;
    virtual double getTangent (void) = 0;
    virtual double getInitialTangent (void) = 0;
    virtual double getDampTangent (void);
    virtual double getRho(void);

    virtual int commitState (void) = 0;
    virtual int revertToLastCommit (void) = 0;
    virtual int revertToStart (void) = 0;

    virtual UniaxialMaterial *getCopy (void) = 0;
    virtual UniaxialMaterial *getCopy(SectionForceDeformation *s);

    virtual Response *setResponse (const char **argv, int argc,
                   OPS_Stream &theOutputStream);
    virtual int getResponse (int responseID, Information &matInformation);

    // AddingSensitivity:BEGIN ////////////////////////////////////////
    virtual double getStressSensitivity     (int gradIndex, bool conditional);
```

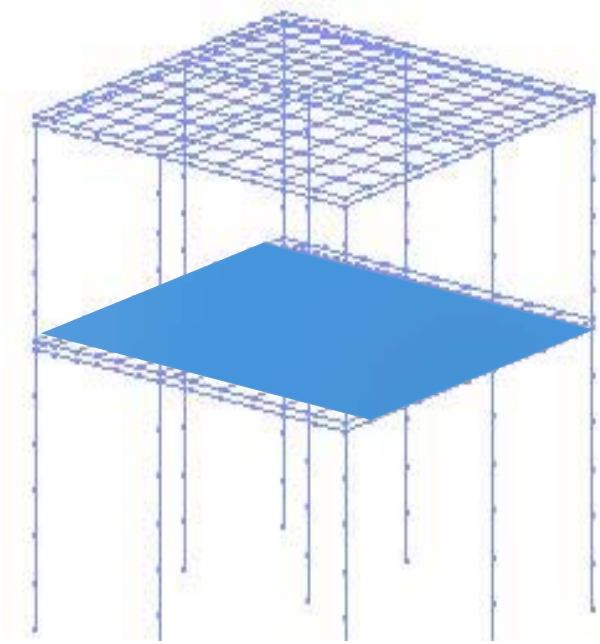
Update material status

Inquiry of material variables

OPENSEES WORKSHOP

Part 6: Interface of element classes

Element classess



Structural model

ZeroLength elements

Truss elements
Joint elements
Beam-Column elements

Shell elements

Other elements: Brick, contact, soil

OpenSees/SRC/element

BeamColumn Element

element dispBeamColumnThermal User script

TclCommand_addElement — TclModelBuilder.cpp

TclModelBuilderElementCommand — TclElementCommands.cpp

DispBeamColumn2dThermal.h & DispBeamColumn2dThermal.cpp

Standard interface of beam-column elements?

- Apply elemental load (UDL, Thermal load)
- section deformation
- Tangent stiffness matrix
- Resisting force

DispBeamColumn2dThermal.h

Element Interface

addLoad

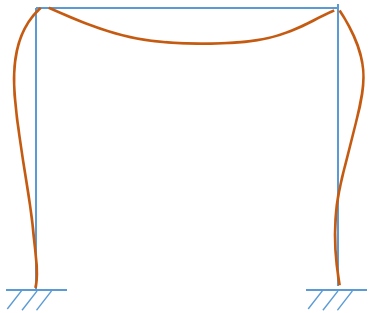
setDomain

Update

CommitState

```
class DispBeamColumn2dThermal : public Element
{
  public:
    DispBeamColumn2dThermal(int tag, int nd1, int nd2,
            int numSections, SectionForceDeformation **s,
            BeamIntegration &bi, CrdTransf &coordTransf,
            double rho = 0.0);
    DispBeamColumn2dThermal();
    ~DispBeamColumn2dThermal();

    const char *getClassType(void) const {return "DispBeamColumn2dThermal";};

    int getNumExternalNodes(void) const;
    const ID &getExternalNodes(void);
    Node **getNodePtrs(void);

    int getNumDOF(void);
    void setDomain(Domain *theDomain);

    // public methods to set the state of the element
    int commitState(void);
    int revertToLastCommit(void);
    int revertToStart(void);

    // public methods to obtain stiffness, mass, damping and residual information
    int update(void);
    const Matrix &getTangentStiff(void);
    const Matrix &getInitialStiff(void);
    const Matrix &getMass(void);

    void zeroLoad();
    int addLoad(ElementalLoad *theLoad, double loadFactor);
    int addLoad(ElementalLoad *theLoad, const Vector &loadFactors);

    int addInertiaLoadToUnbalance(const Vector &accel);

    const Vector &getResistingForce(void);
    const Vector &getResistingForceIncInertia(void);
```

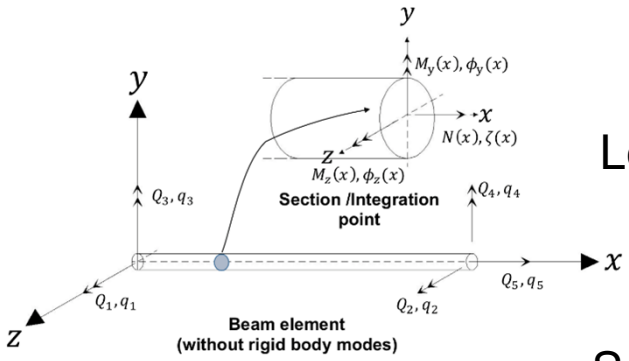
BeamColumn Element



Global displacement（Global coordinate system)

Geometric Transformation

- Linear
- PDelta
- Corotational



Local displacement（Basic system)

Section response (integration points)

Element stiffness & Resisting force(Basic system)

Geometric Transformation

Global stiffness& unbalanced forces(Global system)

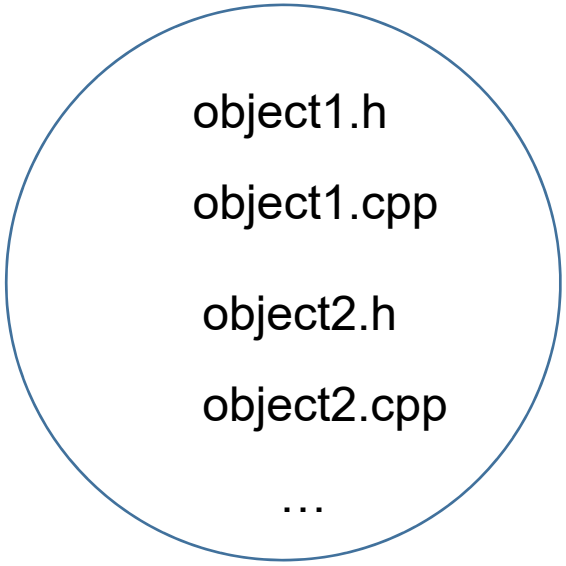
OPENSEES WORKSHOP

Part 7: How to add a project

How to Add a Project

What in a new Project?

project



Framework Hierarchy

In a header file (.h)

Inclusion of other header files
Declaration of variables
Declaration of functions

In a source file (.cpp)

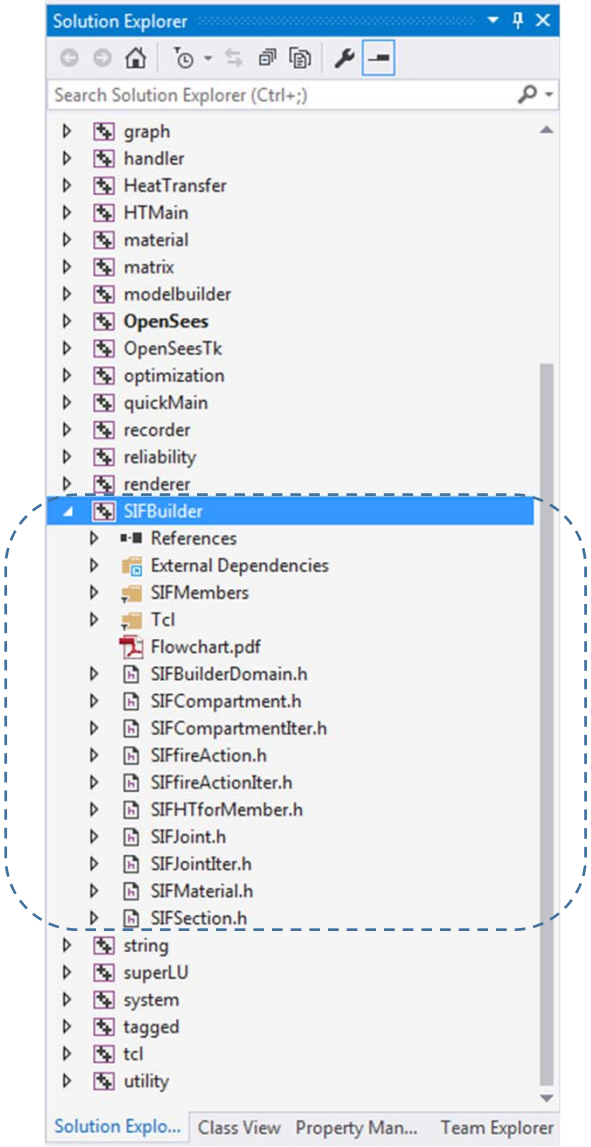
Inclusion of header files
Constructors of class
Destructor of class
Definition of functions

Prepare the files, and save them in the right folder

OpenSees/SRC/<your project>

How to Add a Project

- Add a new Project to OpenSees



❖ Create a project folder in

OpenSees/win32/proj/<your project>

❖ Add this new project

--if it is completely new, headers and sources have to be added;

--if it is not, files are imported automatically as the structure has been defined in the proj file

How to Add a Project

❖ **Project property** (right click at the project->configuration properties)

-Project properties are defined for **debug** and **release** separately

-Add the dependencies(additional included directories)

subfolders in SRC/<project name>

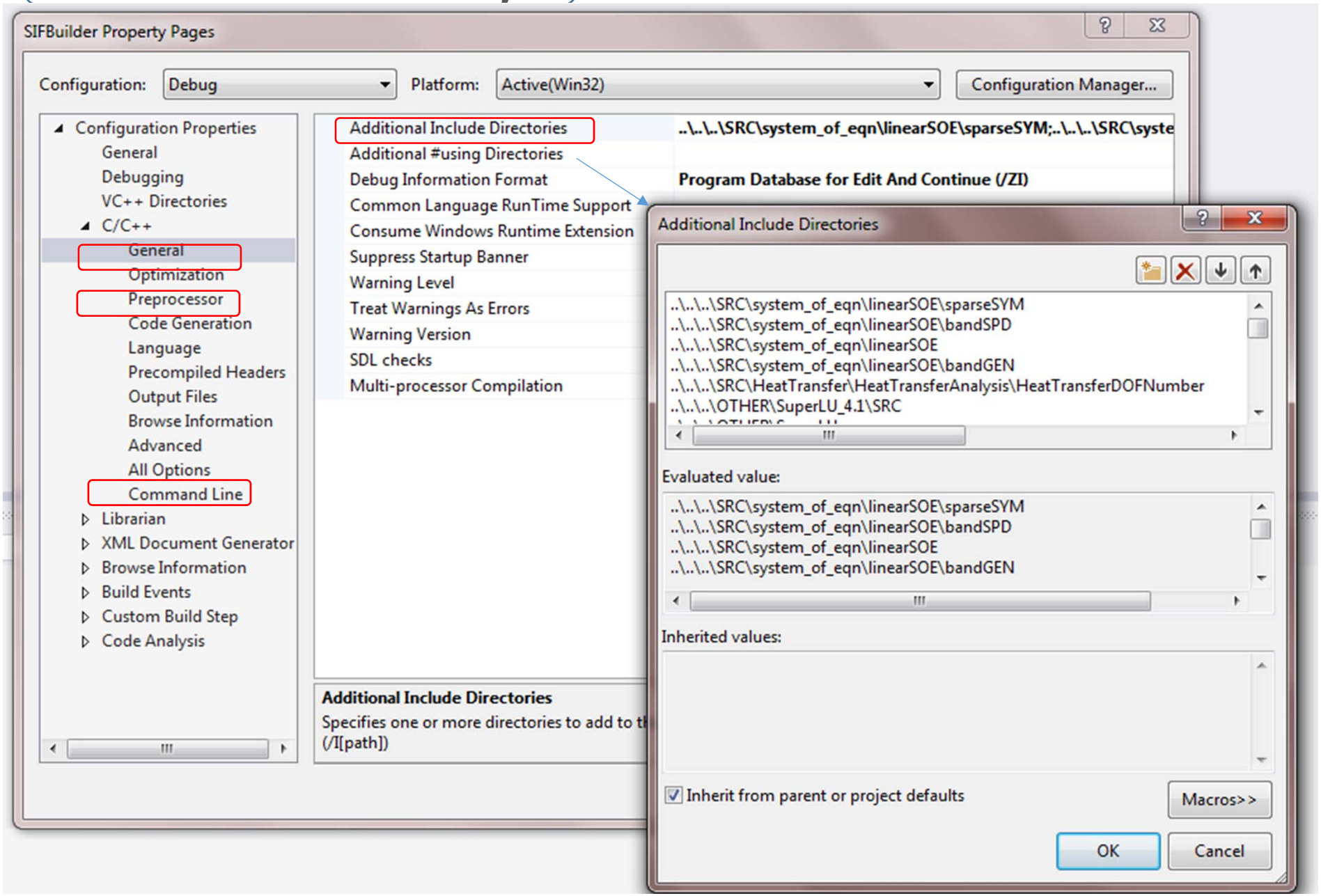
-Preprocessor tag(_SIFBUILDER, _HEATTRANSFER)

#ifdef could selectively activate code block

-output as multi-threaded debug for debugging build

-multi-thread for release

How to Add a Project



How to Add a Project

Possible Errors

Compiler

- Not including right headers
- Deleted variables (destructor)
- Mismatched returned value from a function
- Mismatched constructor and usage of a class
- Incorrect project properties

Linker

- Not including right libraries
- Referenced function can not be found because it's not correctly defined
- Library is not produced
- Linker property of OpenSees project

OPENSEES WORKSHOP

Part 7: A summary of OpenSees for Fire

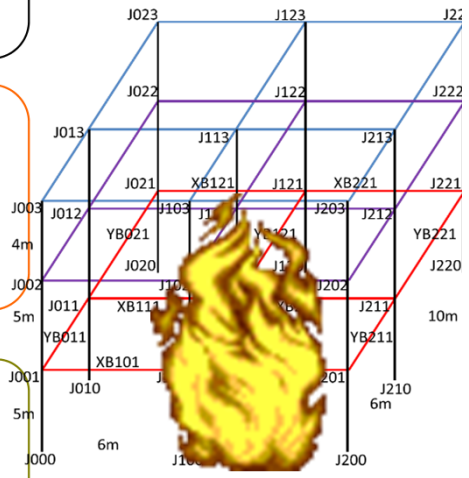
OpenSees for Fire

SIFBuilder

User-friendly interface for creating (regular) structural models and enable consideration of realistic fire action

Fire

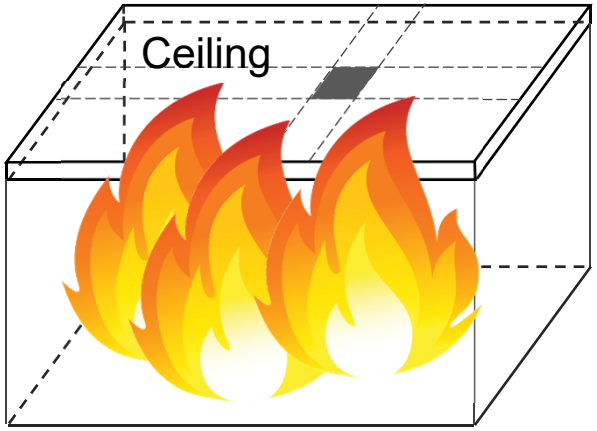
Models of fire action (only ***idealised*** fires), i.e., Standard fire, Parametric fire, EC1 Localised fire, Travelling fire



Thermo-mechanical

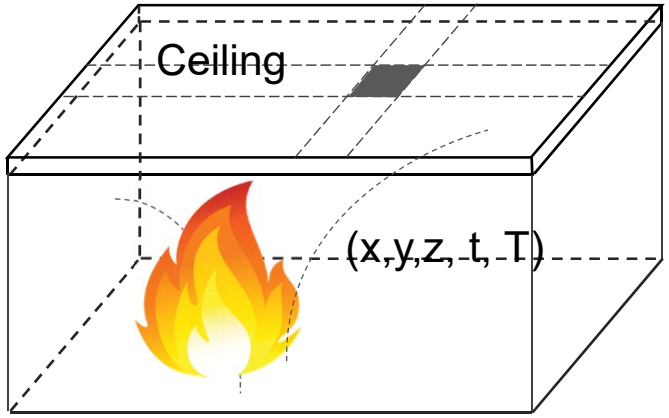
Structural response to the elevated temperatures

Fire modelling



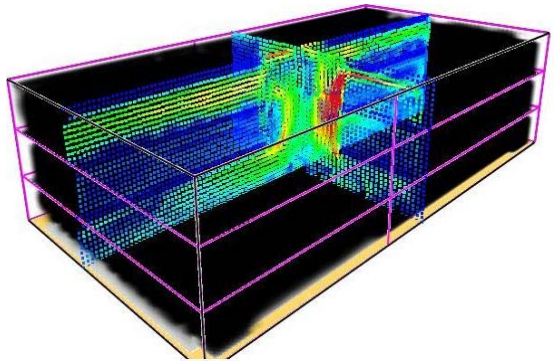
Idealised Uniform fire action

Standard fire
Parametric fire
…



Idealised non-uniform fire action

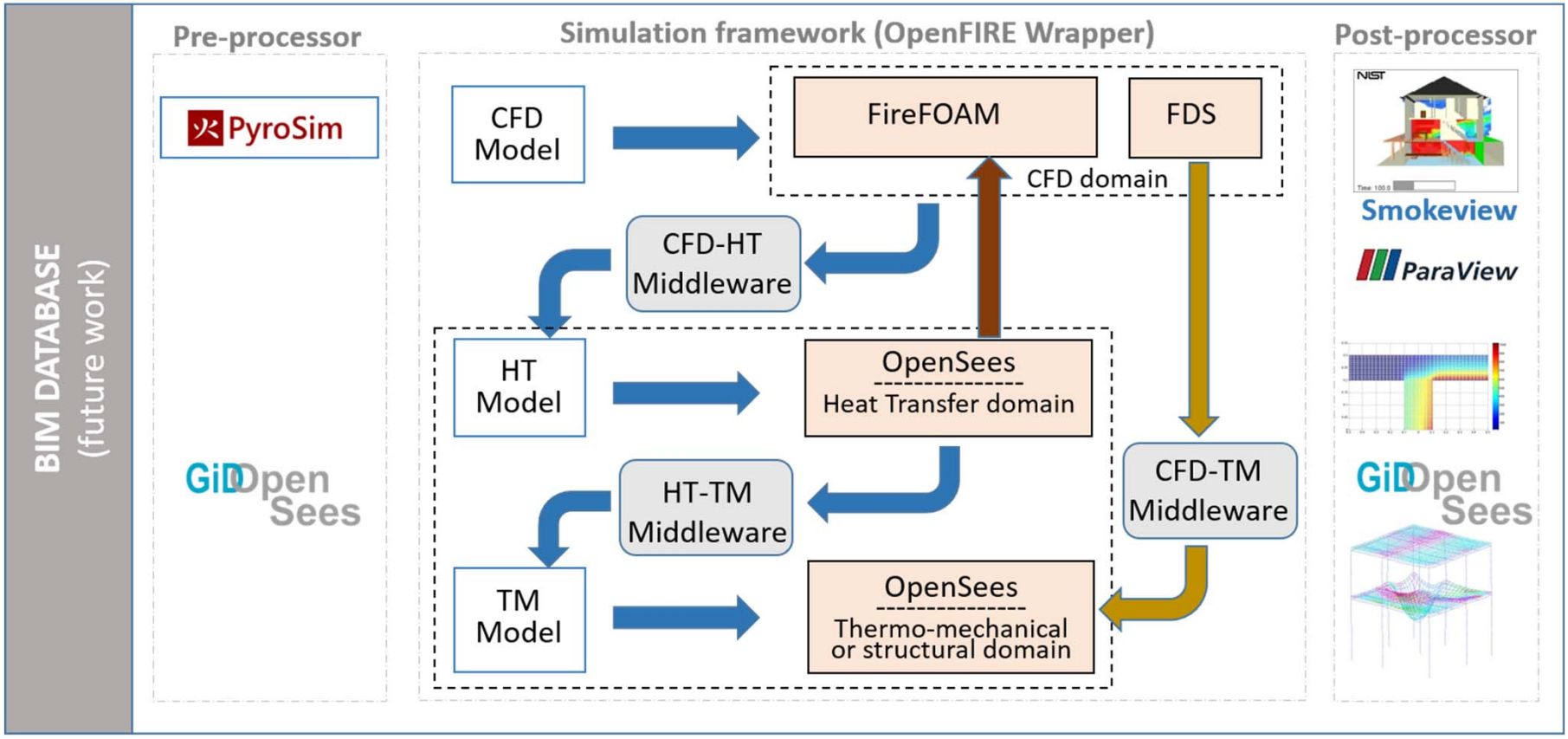
Localised fires
Travelling fires
…



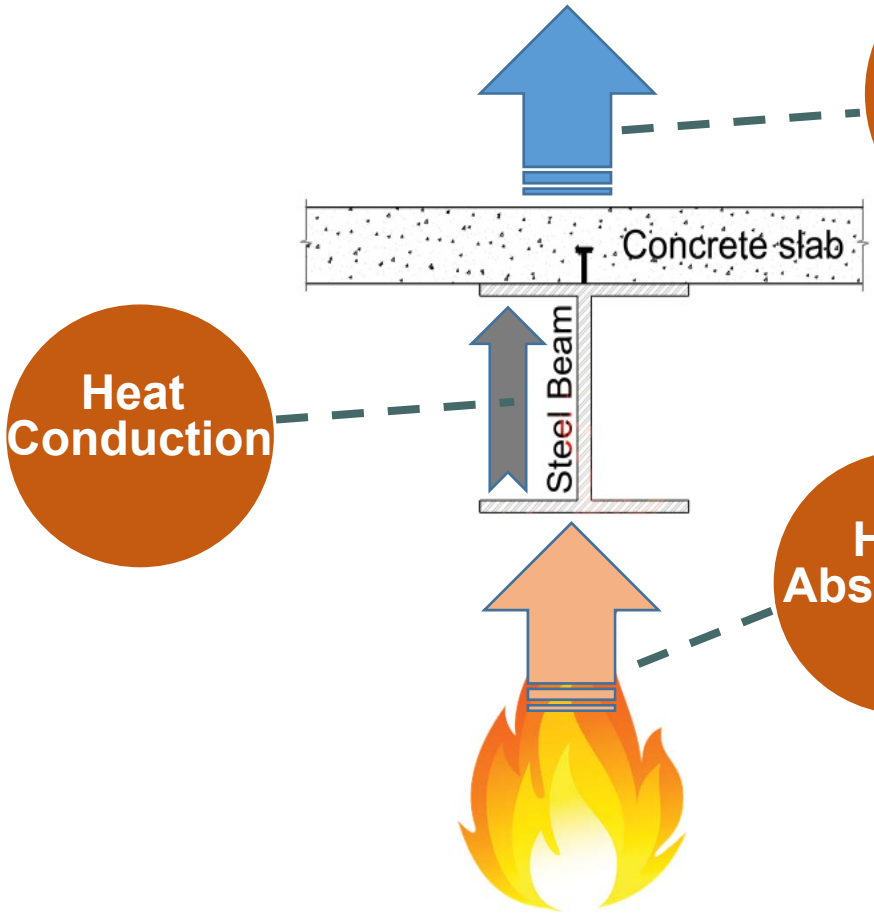
CFD fire simulation

Real fires
…

OpenFire


BIM DATABASE
(future work)
Pre-processor
PyroSim
GiD
OpenSees
Simulation framework (OpenFIRE Wrapper)
CFD Model
FireFOAM
FDS
CFD domain
CFD-HT Middleware
HT Model
OpenSees
Heat Transfer domain
HT-TM Middleware
TM Model
OpenSees
Thermo-mechanical or structural domain
CFD-TM Middleware
Post-processor
Smokeview
ParaView
GiD
OpenSees

Heat Transfer



Heat Loss

- Convection $\dot{q} = h(T_s - T_a)$
- Radiation $\dot{q} = \sigma(T_s^4 - T_a^4)$

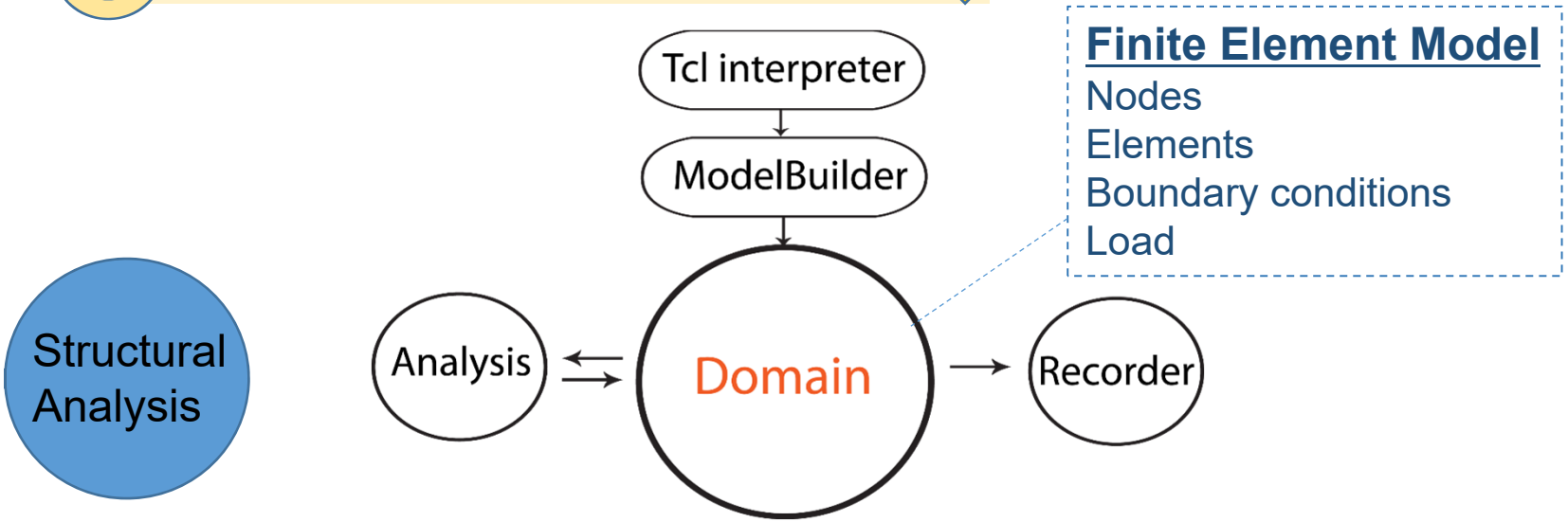
Heat Conduction

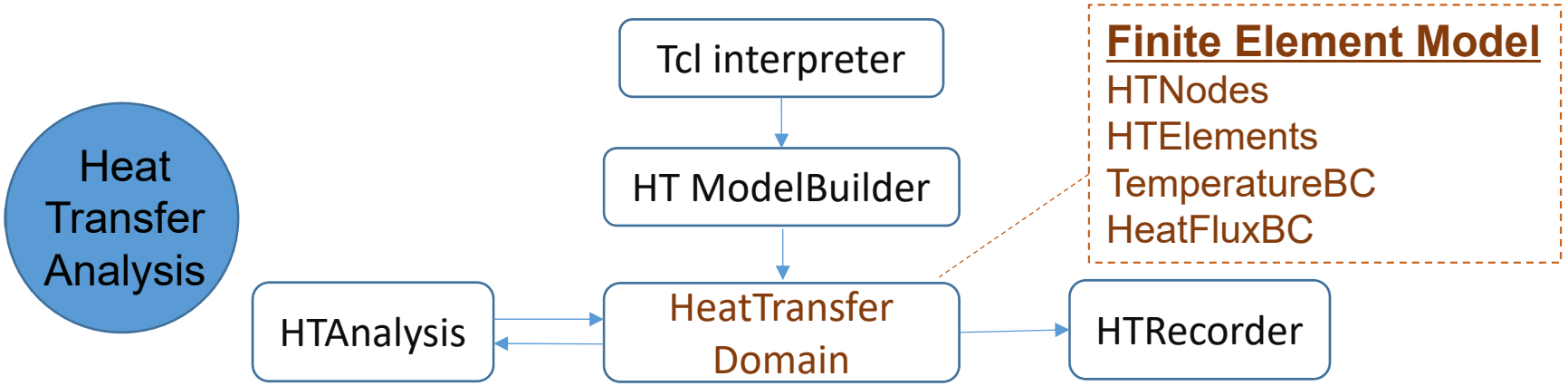
Heat Absorption

- Convection $\dot{q} = h(T_f - T_s)$
- Radiation $\dot{q} = \sigma(T_f^4 - T_s^4)$

Heat Transfer

Let's recall the OpenSees framework





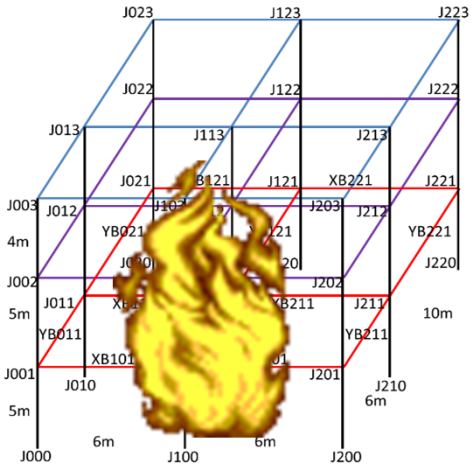
SIFBuilder Project

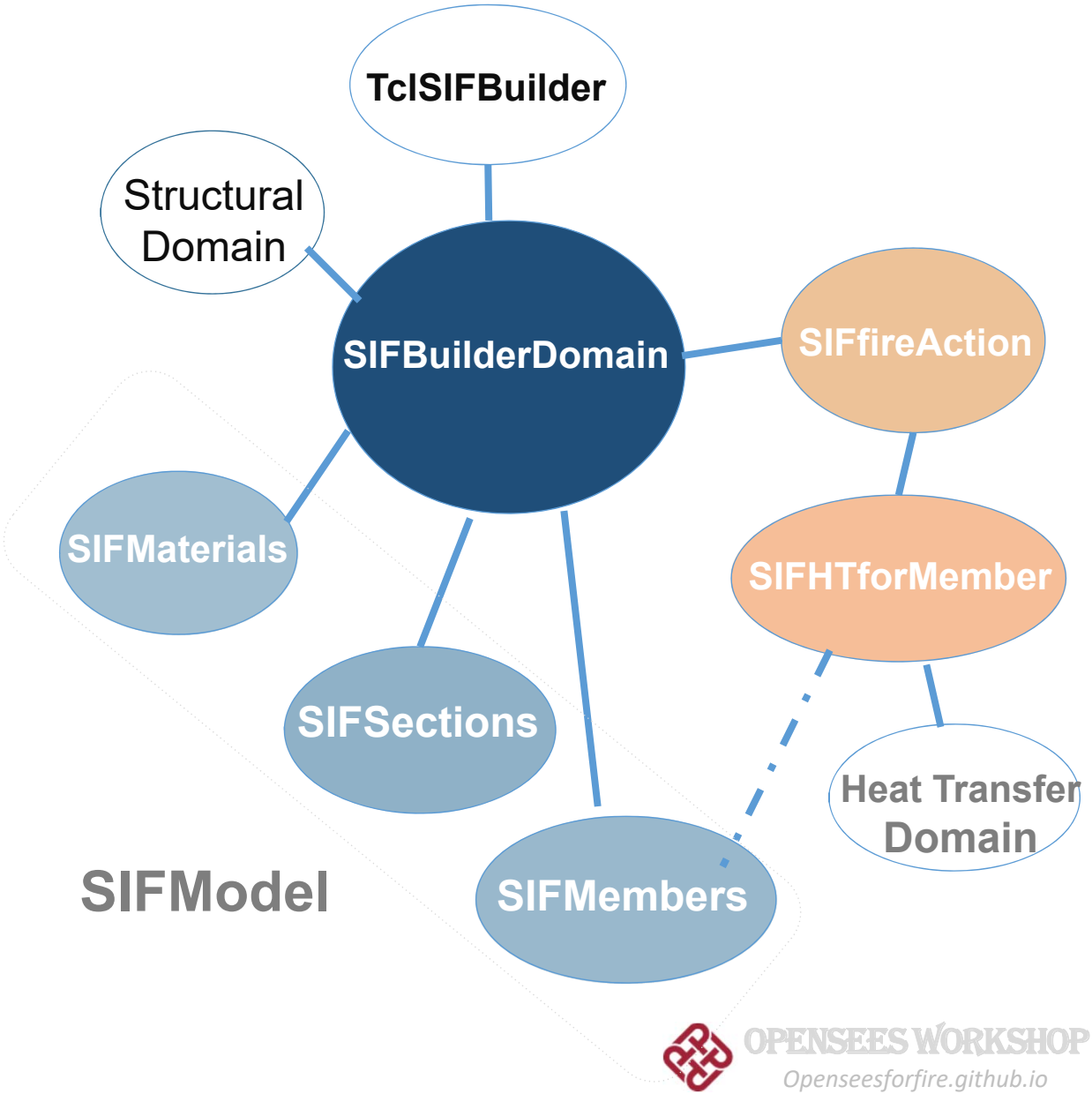
- **Tcl** supported

  (Tool command language)
- SIFBuilderDomain as main storage
- SIFModel created for building info

  (material, section ,members)

- Various types of Imposed loads
- Various types of fire action
- Automated heat transfer analyses
- Automated implementation of thermal action





Github Website

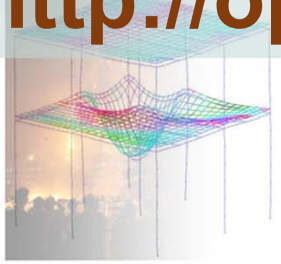
OPENSEES FOR FIRE

View the Project on GitHub
OpenSees@GitHub

View People
OpenSees for Fire Group

Go To	Download	View On
Berkeley	**OpenSees**	**GitHub**

OpenSees



http://openseesforfire.github.io

About

The OpenSees development for modelling 'structures in fire' was first started at University of Edinburgh in 2009. A couples of students and researchers worked on this long-term project with their own contributions which enable OpenSees to perform heat transfer and thermo-mechanical analyses.

Users

A number of web pages are constructed to offer the users a detailed guidance to the recently added capabilities within OpenSees

Developers

A detailed description of all the new or modified classes developed for enabling thermo-mechanical analyses in OpenSees.

Documents

Relevent publications to OpenSees for fire development can be found from the above section

Download

The executable file for OpenSees for fire can be downloaded on this page. The version is updated frequently

Q&A

If you encounter any problems when using OpenSees for fire,please leave your messages on the "issues" board.

This project is maintained by Liming Jiang

Hosted on GitHub Pages — Theme by orderedlist

Thanks! Any questions?

Develop your own OpenSees-
Architecture & Components

SiF Workshop: OpenSees for fire, 3 Dec 2020

Presented by **Dr Liming Jiang**
Assistant professor
Research Center for Fire Safety Engineering
Department of Building Services Engineering
The Hong Kong Polytechnic University

THE UNIVERSITY OF QUEENSLAND
AUSTRALIA

THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學