# Modeling OpenSees using WebGME

Ali I Ozdagli
Supervisor: Xenofon Koutsoukos

December 13, 2018

## Overview

OpenSees, the **O**pen **S**ystem for **E**arthquake **E**ngineering **S**imulation, is an open-source finite element tool that has advanced capabilities for modeling and analyzing linear and nonlinear response of structural and geotechnical systems using a wide range of material models, elements, and solution algorithms. OpenSees is a text-based program and users develop their models by writing scripts in Tcl and employing commands special to OpenSees. The commands for scripts are specific to structural and geotechnical domains. Users can define variables such as geometry, boundary conditions, section size of structural elements, material and behavior of the elements, damping ratios, solvers, input excitations, and many more.

## Goals

This project seeks to add a basic support to WebGME to run OpenSees models and to visualize the structural responses. Using the model-integrated approach, OpenSees models can be developed and visualized in WebGME. Due to Lego-like nature of the meta-model, the simulation parameters can be re-used for different type of analysis.

The tool is especially useful for visualizing a structural model instead of using script which may be challenging for undergraduate students who want to learn how structures are working but don't know how to start. Additionally, the tool is capable of generating an executable script file that can be run by OpenSees. Therefore, by playing with the model in WebGME and generating scripts, one can understand how the script is working and what is the visual representation of the generated script.

## Domain Specifics

A typical OpenSees structure can be abstracted as three subdomains, shown in Fig. 1. Each domain can be isolated and paired with another two domains and a proper simulation can be performed.

The key domains are formalized by the OpenSees authors as:

- *Model*: This domain contains the information regarding the structure to be modeled, see Fig. 2. A structure can be idealized as a collection of nodes and elements. The node represents the joints of the structure and the element describes the structural member that connects the nodes. If necessary nodes may have constraints to emulate the boundary conditions. Each element has their own geometirc properties (section dimensions) and materials (steel, concrete, etc.). Finally, the loading patterns (what is the magnitude and direction of the load and where does it apply) can be defined in this domain. While in meta-modeling the patterns are assumed to be part of modeling domain, it is always associated with analysis to streamline the transformation from model to script.

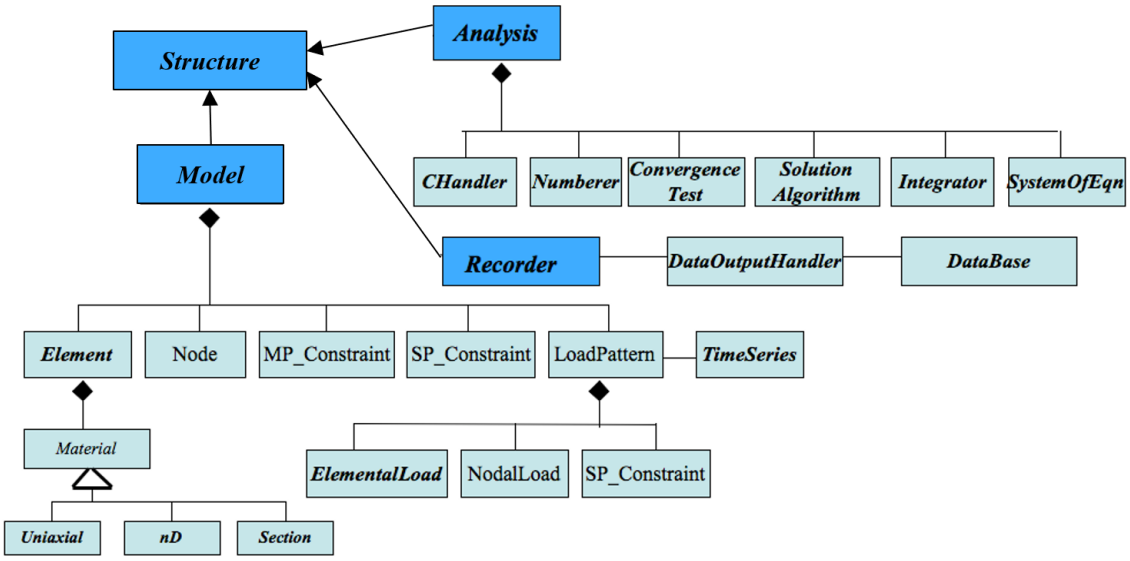


Figure 1: OpenSees Abstraction

- *Analysis*: The analysis domain contains the parameters related to solver mechanisms, convergence algorithms, etc. User can decide whether to run static or transient analysis.
- *Recorder*: This domain handles saving the simulation results to harddrive or to a adatabase. The results can be saved as a xml file or plain text. Node or element responses can be captured using recorders. Plain text does not provide metadata in the header. Therefore, this project relies on xml version.

There are also utility functions provided by OpenSees to ease the workflow during writing scripts.

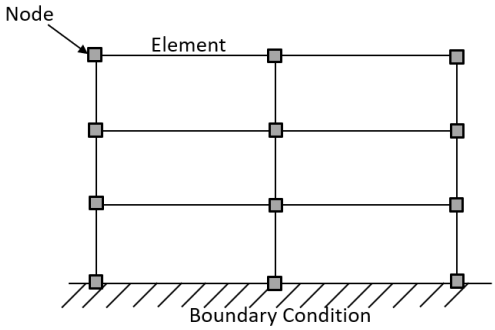


Figure 2: Idealization of the Structure

## Brief Summary on Implementation

One can run a simulation by coding the structure using the three domains explained above. Meta-modeling allows users to describe the structure using building blocks. Each building block in the meta-model is associated with script pieces. For example **node** block in the meta-model corresponds to **node** script. This tool encapsulated a major portion of the scripts documented in the OpenSees website.

## Meta-Model

The meta-model has three domains similar to OpenSees abstraction (*Modeling, Analysis, Recorder*). Additionally, a *Misc.* domain is defined that contains essential OpenSees utility function. Last but not least, the Meta-Model has its own utility domain that contains blocks such as Documentation and MetaData. Additionally, *constants* is defined here which is not part of OpenSees model but part of Tcl scripting scheme, see `https://wiki.tcl-lang.org/page/constants`.

## Composition

There are over 80 components in this meta-model including the abstract nodes. The compositions are structured in such a way that there each domain has a number of abstract components and the derivatives from those components. For example, **algorithm** is an abstract component in the domain analysis which is responsible for defining the solution algorithm. There are five algorithm types user can pick from such as **algorithm Linear**, **algorithm Newton** and so on. This pattern is pursued for all the major compositions. However, there are other components which does not need to be identified as abstract.

## Transformation

Using WebGME, a transformation plugin, **OpenSeesTransformation** is written. This plugin performs the following operations:

- It transforms the structure (modeled with meta-model blocks) into OpenSees Tcl script input file.
- It runs the simulation by executing OpenSees.exe using the generated script file.
- The results are written automatically by the OpenSees.exe. This file is used as the base to create json file for the visualizer. Upto this point, Python is used for the transformation.
- Using plotly.js the json file is visualized in webGME.

## Examples

There are two examples are provided within this tool.

- *2D Elastic Structure Static*: This example considers push-over analysis of a three story three bay structure. This model has 16 nodes, and 21 elements. The structure is fixed to the ground. For horizontal elements (beams), a linear geometric deformation is assumed. As for the vertical elements (columns), P-Delta effect (a common assumption for analysis of structures subjected to lateral loads) is assumed for realistic simulation. This structure has two combined and subsequent simulations to emulate a realistic responses. First, a gravity load is applied on the beams to emulate structural weight. The second simulation is a push-over analysis, where a lateral load is applied on the nodes. Both analysis are static, i.e. inertial forces are ignored. A recorder is defined to capture the lateral response of the structure to the applied lateral load for the nodes 5-15. Nodes 1 to 4 are fixed hence they don't move and there is no need to record their responses.
- *2D Elastic Structure Dynamic*: This example considers transient analysis of the same structure under the influence of ground motion. This example has also two combined and subsequent simulations. First, a gravity load is applied on the beams to emulate structural weight. This simulation is static. The second simulation is a transient analysis, where ground motion recorded by a nearby station during 1995 Kobe earthquake is applied from the ground. A recorder is defined to capture the lateral response of the structure during the simulation.

## Using Repo

Below, the steps are explained to run open OpenSees Meta-Model tool.

- Follow the standard installation and setup of WebGME (as instructed here `https://github.com/webgme/webgme`).
- The project assumes your mongodb folder is **webgmeData** under C drive and your application folder is **mywebgmeapp** under C drive.
- There is a plugin called OpenSeesTransformation that converts an OpenSees WebGME model into Tcl script. It is located under `src/plugins/OpenSeesTransformation`. Standard entry point for debug is **run_debug.py** under the same folder. The transformation code is given in **init.py** under `src/plugins/OpenSeesTransformation/OpenSeesTransformation`
- Python 3.6 is used. Packages like webgme_binding, sys, os, time, csv, subprocess, signal, atexit, ElementTree, StringIO and logging should be already installed since the transformation code.
- plotly.js is used for visualization. It is already provided in the repo so no need for new installation.
- Dumping the content to a freshly created **mywebgmeapp** should make the project work.

## Running Transformation

- There are two examples. One for 2D (pushover) static and the other one is dynamic (earthquake) simulation.
- Click on the example that you want to play with. Select validPlugins as OpenSeesTransformation if not selected.
- Open the example double-clicking on it. Do not click on any block and run OpenSees transformation.
- Under a minute, the plugin will convert the model into an executable Tcl script, run the simulation, and generate outputs.
- After successful run of the plugin, go to **SimulationResultPlotter** under Visualizer Selector on the left side. This will show simulation results. The visualization may take some a minute due to the amount of data.