

**Estudio del Empleo del Software de Código Abierto OpenSees en el Análisis Estructural
de Edificaciones**

Federico Arias García

Monografía para optar al título de Especialista en Estructuras

Director:

Oscar Javier Begambre Carrillo

Ph.D en Ingeniería Civil

Universidad Industrial de Santander

Facultad de Ingenierías Físico-mecánicas

Escuela de Ingeniería Civil

Especialización en Estructuras

Bucaramanga

2018

Agradecimientos

A Dios que desde lo invisible e incomprensible nos anima con su amor a emprender nuevos caminos y retos para el fortalecimiento de nuestras almas. A mis queridos padres María Higinia y José Ángel, quienes en el tiempo en que adelantaba estos estudios de especialización, partieron juntos a la Eternidad, a ellos les seguiré debiendo, por siempre, mi formación personal y profesional. A mi amada esposa Sonia y mis hijos Braian, Sol y Luna, que han sacrificado su tiempo y me han dado aliento para poder concluir con éxito ésta tarea. A mis compañeros de estudio en la especialización, especialmente, con quienes tuve el gusto de trabajar. A todo el excelente grupo de profesores que nos transmitieron sus conocimientos y experiencias, especialmente al ingeniero Oscar Begambre, no solo por ser el director de este trabajo, sino por su sin igual manera de enseñar y por su amistad.

Contenido

	Pág.
Introducción	13
1. Justificación	15
2. Objetivos	16
2.1 Objetivo general.....	16
2.2 Objetivos específicos:	16
3. Alcance	17
4. Marco Teórico.....	17
4.1 Software de código abierto	17
4.2 Licencia Pública General	18
4.3 Programación Orientada a Objetos	19
4.4 El Método de los Elementos Finitos	20
4.5 Técnicas de Análisis Estructural (Mbie,Eqc,Nzsee,Sesoc & Nzgsi , 2017).	21
4.5.1 Técnicas de análisis estático	21
4.5.1.1 Análisis de Fuerza Horizontal Equivalente	21
4.5.1.2 Análisis de espectro de respuesta modal o análisis dinámico elástico	22
4.5.1.3 Análisis pseudo pushover estático no lineal.	22
4.5.2 Técnicas de análisis No lineal.....	22
4.5.2.1 Análisis Estático No Lineal de Plastificación Progresiva (Pushover)	23

4.5.2.2 Análisis no lineal del historial de tiempo o dinámico cronológico.....	23
5. OpenSees: Una Plataforma Informática para la Simulación de Ingeniería Sísmica	25
5.1 Plataforma Informática OpenSees	25
5.2 Copyright de OpenSees.....	26
5.3 Arquitectura de OpenSees (Mazzoni S., McKenna F, Scott M., & Fenves G., et al 2007)....	27
5.3.1 Pre-procesamiento o Módulo ModelBuilder	28
5.3.2 Procesamiento o Módulo Analysis	30
5.3.3 Módulo Recorder	31
5.3.4 Módulo Domain	32
5.4 Intérpretes de OpenSees.....	33
5.4.1 Aplicaciones Secuenciales de OpenSees.	33
5.4.1.1 Opensees.exe.....	34
5.4.1.2 OpenseesPy.....	34
5.4.2 Aplicaciones en Paralelo de OpenSees.	38
5.4.2.1 OpenSeesSP: (Intérprete de OpenSees en Paralelo Simple).....	38
5.4.2.2 OpenSeesMp: (Intérprete de OpenSees en Paralelo Múltiple).	39
5.5 Interfaces gráficas de usuario de OpenSees.....	41
5.5.1 BuildingTcl/BuildingTclViewer.	42
5.5.2 OpenSees Navigator.....	43
5.5.3 GiD+OpenSees.	44
5.5.4 Buil-X.	46
5.6 OpenSees frente al Software Comercial de Análisis Estructural.....	47
5.6.1 OpenSees y M.S.C. Marc.....	49

5.6.2 OpenSees, Sap2000 y Perform-3D.	52
5.6.3 OpenSees y Ansys.....	57
5.6.4 OpenSees y Cype	58
5.7 Estadísticas de OpenSees	59
6. Ejemplos de aplicación	63
6.1 Ejemplo de análisis estático lineal	63
6.2 Ejemplo de análisis estático no lineal	76
7. Conclusiones Finales	84
Referencias Bibliográficas	86

Lista de Tablas

	Pág.
Tabla 1. Ventajas y desventajas del código abierto	17
Tabla 2. Comparativo entre OpenSees y otros software comerciales de análisis estructural	48
Tabla 3. Comparación de los períodos previstos por OpenSees and MSC.Marc.	50
Tabla 4. Comparación de los resultados del cortante en la base por OpenSees and MSC.Marc..	51
Tabla 5. Comparación de los resultados del radio de deriva de entrepisos OpenSees and MSC.Marc.....	51
Tabla 6. Comparación de los períodos de los diferentes programas.....	56

Lista de Figuras

	Pág.
Figura 1. Esquema general de la implementación computacional de OpenSees	28
Figura 2. Modelo de cercha	35
Figura 3. Comparación de códigos de OpenSees.exe y OpenSeesPy	36
.Figura 4 Comparación de los resultados en OpenSees.exe y OpenSeesPy.	37
Figura 5. Aplicaciones OpenSeesMP y OpenSeesSP en la web.....	40
Figura 6. BuildingTclViewer GUI.....	43
Figura 7. Interface Gráfica de Usuario de OpenSees Navigator.....	44
Figura 8. Estructura de Interface Gráfica GiD+OpenSees	45
Figura 9. Ejemplo de modelado de la estructura en Build-X: A la izquierda el cuadro de las columnas con su sección detallada y la derecha su ubicación en la planta del edificio.	47
Figura 10. Edificio en estudio : (a) elevación y (b) planta.....	50
Figura 11. Modelo del pórtico y cargas aplicadas.	53
Figura 12. Curvas Pushover	54
Figura 13. Elevación del pórtico tipo de momento de acero (Dirección N-S) y planta del edificio.	56
Figura 14. Mapa de visitas alrededor del mundo 2012.....	60
Figura 15. Mapa de visitas alrededor del mundo 2016.	61
Figura 16. Planta de la estructura – planos en X.....	63
Figura 17. Fuerzas sísmicas para el edificio de 3 niveles y diagrama de corte.	64

Figura 18. Desplazamientos en los pórticos en dirección X.....	64
Figura 19. Modelo de pórtico.....	65
Figura 20. Ejecución del archivo script .tcl en OpenSees.exe, versión 2.5.0.	74
Figura 21. Desplazamientos en los nodos 4, 6 y 8.....	75
Figura 22. Archivo piso3.out con resultados de fuerzas globales de los elementos del último piso.....	76
Figura 23. Archivo disp8nl.out correspondiente contiene los desplazamiento del nodo 8.....	82
Figura 24. Archivo piso3nl.out contiene las fuerzas globales de los elementos de último piso..	83
Figura 25. Resultados de las fuerzas globales para un desplazamiento de 9 cm.	83

Resumen

TITULO: ESTUDIO DEL EMPLEO DEL SOFTWARE DE CÓDIGO ABIERTO OPENSEES EN EL ANÁLISIS ESTRUCTURAL DE EDIFICACIONES*

AUTOR: FEDERICO ARIAS GARCIA**

PALABRAS CLAVE: SOFTWARE, OPENSEES, ANALISIS, ESTRUCTURAL, CODIGO ABIERTO

DESCRIPCIÓN: Mediante el presente trabajo se pretende evaluar la confiabilidad y la validez del uso del software de código abierto OpenSees (*Open System for Earthquake Engineering Simulation*) en el análisis estructural de edificaciones, a partir de la evidencia científica reportada a nivel internacional.

El trabajo incluye lo siguiente: en los capítulos 1 y 2 se presenta una introducción y justificación del uso de OpenSees en ingeniería estructural así como los alcances del presente trabajo, que son definidos en el capítulo 3. El capítulo 4 presenta un marco teórico donde se exponen algunos conceptos preliminares y básicos referentes a temas de programación y análisis estructural estrechamente relacionados con OpenSees. Seguidamente, en el capítulo 5, se realiza una breve presentación del software OpenSees junto con una exposición de los aspectos fundamentales de su arquitectura de programación; una relación de las aplicaciones básicas de OpenSees; una exposición de algunos estudios donde se ha usado OpenSees y en los cuales ha sido comparado o integrado con el uso de reconocidos programas comerciales de análisis estructural y un reporte de datos estadísticos de OpenSees. A continuación, en el capítulo 6 se realizan dos ejemplos sencillos de aplicación. Finalmente, en el capítulo 7, se emiten una serie de conclusiones y recomendaciones que pretenden ayudar en la toma de una decisión en cuanto a la adopción de OpenSees como una herramienta para análisis estructural de edificaciones en nuestro medio profesional.

* Monografía

** Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería Civil. Especialización en Estructuras Director: Oscar Javier Begambre Carrillo, Ph.D

Abstract

TITLE: STUDY OF EMPLOYMENT OF OPEN SOURCE SOFTWARE OPENSEES IN THE STRUCTURAL ANALYSIS OF BUILDINGS*

AUTHOR: FEDERICO ARIAS GARCIA**

KEYWORDS: SOFTWARE, OPENSEES, ANALYSIS, STRUCTURAL, OPEN SOURCE

DESCRIPTION:

The present work intends to evaluate the reliability and the effectiveness of the open source software Opensees (Open System for Earthquake Engineering Simulation), in building structural analysis, from the scientific evidence reported at the international level.

The work includes the following: chapters 1 and 2 present an introduction and justification of the use of OpenSees in structural engineering as well as the scope of this work, which are defined in chapter 3. Chapter 4 presents a theoretical framework where some preliminary and basic concepts related to programming and structural analysis topics closely related to OpenSees are exposed. Then, in chapter 5, a brief presentation of the OpenSees software is made along with an exposition of the fundamental aspects of its programming architecture; a list of the basic applications of OpenSees; an exhibition of some studies where OpenSees has been used and in which it has been compared or integrated with the use of recognized commercial structural analysis programs and a statistical data report of OpenSees. Then, in chapter 6 two simple examples of application are made. Finally, in chapter 7, a series of conclusions and recommendations that aim to help in making a decision regarding the adoption of OpenSees as a tool for structural analysis of buildings in our professional environment.

* Monograph

** Faculty of Physical-Mechanical Engineering. School of Civil Engineering. Specialization in Structures Director: Oscar Javier Begambre Carrillo, Ph.D

Introducción

Al igual que en todos los campos científicos y tecnológicos modernos, en el diseño y la evaluación de edificios, puentes, cimentaciones y demás estructuras de las que se ocupa la ingeniería civil, la simulación de su comportamiento por medio de software juega un rol crítico y un papel muy importante ya que permite resolver problemas, sin importar su complejidad, de manera confiable y con mayor rapidez. Alrededor del mundo existen varios paquetes de software de análisis estructural comercial que ofrecen notable desempeño, particularmente en términos de estabilidad computacional, precisión y tiempo de ejecución. Además proporcionan interfaces gráficas de usuario que los hacen más amigables en su manejo. Sin embargo el software comercial presenta varios inconvenientes ya que, además de que su código fuente no está disponible, sus costos de licenciamiento son altos lo que hace que una empresa de diseño o un instituto de investigación difícilmente pueda estar dispuesto a comprar varios paquetes de software comercial con funcionalidades que validen la veracidad y la precisión de las simulaciones, convirtiéndose ésta en una de las principales causas de la falta de trabajos de investigación. Para superar esta limitación, un programa de software libre de código abierto es una alternativa para proporcionar un recurso útil para validar y mejorar la fiabilidad de las simulaciones numéricas que requieren realizar los diseñadores e investigadores. Con éste ánimo a finales de la década de 1990 se creó el software de código abierto para el análisis de elementos finitos, OpenSees (*Open System for Earthquake Engineering Simulation*) que es una plataforma que combina el estado del arte de modelos de

elementos finitos y algoritmos de solución no lineal (Minjie, McKenna & Scott 2017 Mittal & Singh 2013 Lu X., Xie L., Guan H., Huang L. & Lu X. 2015)

Mediante este trabajo se pretende hacer un estudio del uso de OpenSees. Como primera medida se desarrolla un marco teórico donde se exponen algunos conceptos preliminares y básicos referentes a temas de programación y análisis estructural muy relacionados con OpenSees (ver capítulo 4).

En el capítulo 5, luego de hacer una presentación del software OpenSees y teniendo en cuenta que al ser de código abierto, éste da la posibilidad de estudiar y desarrollar su código fuente, se exponen aspectos básicos de su arquitectura de programación.

Con el fin de conocer cómo se ejecuta el programa, se relacionan las aplicaciones básicas de OpenSees, incluyendo una breve descripción del funcionamiento de cada una de ellas y su ubicación en internet para su descarga e instalación o para su ejecución directa en la web.

Se exponen algunos estudios donde se ha usado OpenSees y en los cuales ha sido comparado o integrado con el uso de reconocidos programas comerciales de análisis estructural, lo que sirve para tener una idea del grado de confiabilidad de OpenSees.

Se presenta un reporte en cifras de las herramientas con que cuenta OpenSees, que da una noción del alcance de su funcionalidad. Además se muestran algunos datos de las descargas y visitas que se hacen a los diferentes servicios web de la plataforma de OpenSees, lo que indica el interés que ésta despierta a nivel mundial.

Por último, en el capítulo 6, se incluyen ejemplos de aplicación de OpenSees que consisten en el análisis de dos casos sencillos con el propósito de ilustrar acerca del escenario básico de cómo se ensambla un modelo.

1. Justificación

- Debido a que un gran porcentaje de la población colombiana vive en zonas de amenaza sísmica intermedia y alta (AIS 2010), es vital que el análisis estructural de las edificaciones se realice con herramientas informáticas que sean válidas, confiables y asequibles.
- El efecto de un sismo de diseño sobre una estructura es tal que en muchos sitios se esperan deformaciones importantes, lo que induce inelasticidad en los materiales componentes. La evaluación de la respuesta de una estructura que se deforma hasta alcanzar el comportamiento inelástico del material durante un movimiento sísmico fuerte es un problema de importancia en ingeniería sísmica.
- El uso de programas de código abierto, en los que el proceso de cálculo es visible y modificable, puede complementar el aprendizaje adquirido en los cursos de análisis dinámico elástico e inelástico de estructuras y profundizar en el estudio de problemas más avanzados relacionados con el tema (Noy F. & Linero D. 2014).
- OpenSees es uno de estos programas de código abierto que puede llegar a ser una herramienta útil en el desarrollo de la ingeniería sísmica, por tanto, es necesario estudiarlo para poder conocer su confiabilidad y validez.

2. Objetivos

2.1 Objetivo general

Estudiar la confiabilidad y la validez del uso del software de código abierto OpenSees en el análisis estructural de edificaciones, a partir de la evidencia científica reportada a nivel internacional.

2.2 Objetivos específicos:

- Conocer la arquitectura básica de programación que OpenSees utiliza para el análisis estructural de edificaciones.
- Revisar las diferentes formas de ejecutar OpenSees, relacionando las aplicaciones básicas disponibles, haciendo una breve descripción y mostrando el modo de acceder a ellas.
- Estudiar algunos casos de comparación e integración de OpenSees frente a reconocidos programas comerciales de análisis estructural.
- Revisar los reportes estadísticos respecto al uso de la plataforma informática de OpenSees.
- Mostrar ejemplos sencillos de aplicación con un paso a paso que ilustre sobre el modo de ensamblar los modelos y de revisar los resultados de su ejecución.

3. Alcance

- La información para la investigación propuesta será obtenida únicamente de documentos de estudios y publicaciones anteriores que se encuentran en las diferentes bases de datos científicas.

4. Marco Teórico

4.1 Software de código abierto

El software de código abierto (SCA) es aquel del cual un individuo o grupo de individuos dispone fácilmente para usar, modificar o redistribuir bajo una licencia que impone muy pocas restricciones. Una licencia SCA da a los usuarios más derechos que una licencia comercial general ya que permite el acceso al código fuente y permite modificarlo, además sin tener que pagar regalías ni tener que negociar una licencia (Hahn, 2014).

En la Tabla 1 se presentan las ventajas y desventajas que pueden llegar a tener un software de código abierto.

Tabla 1.

Ventajas y desventajas del código abierto

Ventajas	Desventajas
Se distribuye libremente (sin costo).	No posee garantía proveniente del autor.
El código fuente del software también se proporciona bajo código abierto.	Se necesita dedicar recursos (tiempo y dinero) a la reparación de errores.
Su código fuente se pueden descargar fácilmente de Internet.	En ocasiones el usuario debe tener conocimientos de programación.
Sus desarrolladores interactúan alrededor de todo el mundo por medio de Internet.	Como la mayoría se crean sin fines de lucro, pueden perder su financiación y el interés por parte de sus desarrolladores dejando de actualizarse y mejorarse.
Cualquier usuario tiene derecho a crear nuevo software derivado a través de modificación.	
La licencia no es específica para un producto, sino que se aplica a todas las sub-partes dentro del producto licenciado.	
No hay discriminación contra persona o grupos que pueda contribuir con el software y/o usar el software.	

Las licencias SCA proporcionan a los desarrolladores el acceso a código fuente que de otra manera no podrían pagar, ya que la mayoría de programas distribuidos bajo este tipo de licencia son gratuitos. Esto es particularmente importante en el caso de los países con recursos limitados, donde se necesita el acceso a software similar pero no se cuenta con los medios para pagar y mantener una licencia de software propietario (Hahn, 2014).

4.2 Licencia Pública General

Según Galeano Mantilla, Duque & Mejía (2007) se tiene que “...La licencia pública general (GPL por sus siglas en inglés) fue creada a mediados de los años 80 como un mecanismo para impulsar

el uso de software libre. La licencia GPL obliga al licenciatarlo a hacer públicos los códigos fuente del software desarrollado, con el fin de permitir a los usuarios del mismo poder hacer modificaciones, adaptaciones o mejoras al programa. Por otro lado el usuario tiene como obligación licenciar bajo el esquema GPL cualquier producto resultante de la modificación, adaptación o mejora a un software con licencia GPL. Esta filosofía de licenciamiento ha permitido el crecimiento acelerado del número de usuarios de estos programas y en forma paralela, casi como una consecuencia natural, el permanente desarrollo de estos paquetes GPL...”

4.3 Programación Orientada a Objetos

La programación orientada objetos (POO) es una metodología para hacer software cuya esencia es la creación de objetos los cuales tienen ciertas propiedades o atributos que lo caracterizan y métodos o funciones que modifican su estado. Así por ejemplo, un objeto puede ser una carga o fuerza actuante, cuyas propiedades son entre otras su magnitud, su dirección y su ubicación, y sus métodos corresponderían a acciones con las que se puede crear, aplicar, incrementar, borrar, en fin, todas aquellas que apliquen a la naturaleza del objeto. Para ayudar al desarrollo de un programa, los objetos con atributos y funcionamientos similares son agrupados en clases, las cuales son implementadas en un lenguaje de programación. La función de un programa orientado a objetos es la interacción entre los diferentes objetos o clases que lo conforman. Entonces, por ejemplo, un objeto Carga debe interactuar con los objetos que representan los Nodos o los Elementos de la estructura para poder determinar su ubicación, además, este objeto Carga debe tener disponible sus atributos para que el objeto que representa el Análisis de la estructura los utilice.

Un sistema POO bien diseñado permite a los programadores desarrollar software de manera independiente, validar nuevo código, mantener y revisar código nuevo e introducir nuevo código en programas existentes (McKenna 1997, Dutta 2016).

4.4 El Método de los Elementos Finitos

El método de elementos finitos (MEF) es un método numérico que consiste en el reemplazo de un conjunto de ecuaciones diferenciales, por un conjunto equivalente, pero aproximado, de ecuaciones algebraicas. El análisis de elementos finitos se considera una herramienta computacional intensiva que realiza operaciones con matrices muy grandes y que se creó por la necesidad de reemplazar los métodos de análisis iterativos como Cross y Kani entre otros, que se realizaban de manera manual y con los cuales el análisis estructural de un edificio de varios pisos podía tardar varias semanas, lo que no permitía invertir mucho tiempo en la optimización de la estructura. En la actualidad MEF se puede usar en computadoras de escritorio y estaciones de trabajo de ingeniería para obtener soluciones a problemas grandes en análisis estructural estático y dinámico, transferencia de calor, flujo de fluidos, electromagnetismo y respuesta sísmica. (Hutton 2004)

El MEF discretiza ecuaciones diferenciales parciales de equilibrio, cinemáticas y constitutivas para un problema estructural de un sistema de ecuaciones diferenciales no lineales ordinarias que deben ser resueltas con un procedimiento de paso de tiempo, que resulta ser la fase de cálculo computacional más intensiva. A grandes rasgos, los principales pasos en una simulación son el modelamiento que incluye la discretización de los elementos finito (pre-procesamiento), la

resolución de las ecuaciones gobernantes (procesamiento) y la interpretación de la respuesta (post-procesamiento). La operación más importante la fase de solución es calcular el nuevo estado de un dominio causado por la aplicación de una carga o por la adición o remoción de elementos. (McKenna, Scott, Fenves 2010, da Fonseca 2011).

En el campo de la ingeniería estructural, MEF es muy utilizado para encontrar repuestas numéricas aproximadas de las predicciones de daño que un sismo ocasiona sobre una estructura. Sin embargo, si se requiere predecir las respuesta de la estructura al sismo con mayor precisión como los requiere el diseño basado en desempeño se pueden usar Métodos de Elementos Finitos de Precisión (FERM – *Finite Element Reliability Methods*) que permiten definir los parámetros como variables aleatorias para representar las incertidumbres y emplean métodos probabilísticos aproximados para obtener estimaciones de la varianza y covarianzas de las repuestas, sensibilidades y probabilidades de exceder umbrales de rendimiento. (McKenna, 2011).

4.5 Técnicas de Análisis Estructural (Mbie,Eqc,Nzsee,Sesoc & Nzgsi , 2017).

4.5.1 Técnicas de análisis estático

4.5.1.1 Análisis de Fuerza Horizontal Equivalente . En éste análisis las fuerzas laterales de sismo calculadas se distribuyen a lo largo de la altura del edificio y las correspondientes fuerzas internas y desplazamientos se determinan usando análisis estático lineal elástico. El análisis por Fuerza Horizontal Equivalente (FHE) es una estimación del límite inferior probable de la capacidad lateral primaria de la estructura. Su uso no es muy recomendado pero puede ser apropiado en circunstancias donde se requiera alcanzar una capacidad de fuerza significativa.

4.5.1.2 Análisis de espectro de respuesta modal o análisis dinámico elástico . El análisis de espectro de respuesta modal (AERM) es un análisis dinámico lineal y se usa comúnmente en el diseño de edificaciones nuevas y es adecuada en estructuras que se espera que respondan elásticamente a la acción de un sismo. Además, AERM también es adecuado para estructuras con mecanismos plásticos bien definidos y distribuidos, como pórticos dúctiles, o para evaluar estructuras diseñadas con métodos de diseño de mayor capacidad u otros requisitos modernos de diseño sísmico.

El AERM se lleva a cabo utilizando espectros de respuesta linealmente elásticos, con las fuerzas resultantes generalmente escaladas para que coincidan con la fuerza lateral utilizada en el método de FHE y los componentes evaluados en el rango elástico de resistencia y de servicio. La capacidad de deformación post elástica de la estructura se aborda de la misma manera que para el FHE.

4.5.1.3 Análisis pseudo pushover estático no lineal. El análisis pseudo pushover estático no lineal (APPENL) es una técnica en la cual el Análisis Estático No Lineal *Pushover* se completa usando una serie de análisis elásticos progresivos.

EL APPENL es una aproximación manual para realizar análisis *pushover* no lineal particularmente útil para los profesionales que no están familiarizados con paquetes de software que realicen análisis *pushover*.

4.5.2 Técnicas de análisis No lineal

4.5.2.1 Análisis Estático No Lineal de Plastificación Progresiva (*Pushover*) . Este análisis es una técnica refinada de análisis no lineal simple que proporciona una estimación de la capacidad lateral probable del edificio como la suma de las capacidades probables de los mecanismos o sistemas identificados. El análisis *pushover* se basa en un modelamiento explícito de parámetros no lineales y de distribución de carga por medio de un software.

En este sentido, se lleva a cabo un análisis incremental inelástico lateral de la estructura bajo un vector lateral de las fuerzas del piso, cuya magnitud se incrementa gradualmente. El inicio de la acción inelástica de cada miembro se puede identificar y la deformación inelástica de los miembros críticos son rastreados directamente. Esto identifica la capacidad probable de la estructura de manera más confiable que usando técnicas elásticas lineales.

Es así que para implementar *pushover* en OpenSees es necesario definir el tipo de análisis como estático (*analysis Static*) y la serie de tiempo como lineal de tal manera que las cargas se vayan incrementando linealmente (*timeSeries Linear*) tal como se muestra en el ejemplo de aplicación del numeral 6.2 de este documento.

El análisis *pushover* da como resultado una respuesta de fuerza-desplazamiento que se puede usar como un procedimiento de evaluación no lineal para determinar la capacidad de la estructura.

4.5.2.2 Análisis no lineal del historial de tiempo o dinámico cronológico. El análisis no lineal de historial de tiempo (ANLHT) es una forma de análisis dinámico que, en principio, ofrece la predicción más realista de la respuesta sísmica de una estructura.

Lo más importante para destacar de ANLHT es que se usa como una herramienta de investigación para mejorar la comprensión de la tendencia general del mecanismo no lineal y las respuestas medias y ofrece la capacidad de rastrear el inicio de la respuesta inelástica que se obtiene

de los métodos de estímulo estático no lineal, mientras que al mismo tiempo incluye efectos de modo más altos de una manera realista, así como la manera en que podrían variar a medida que la estructura se vuelve no lineal.

Todas las técnicas de análisis vistas anteriormente, pueden abordarse con el uso de OpenSees ya que éste permite definir el tipo de análisis que se quiera desarrollar, ya sea análisis estático (*Static Analysis*), análisis dinámico (*Transient Analysis*) que resuelve análisis que dependen del paso constante del tiempo y/o análisis dinámico variable (*Variable Transient Analysis*) donde se considera el paso del tiempo variable (Mazzoni S., McKenna F, Scott M., & Fenves G., et al 2007).

5. OpenSees: Una Plataforma Informática para la Simulación de Ingeniería Sísmica

5.1 Plataforma Informática OpenSees

OpenSees fue creado a finales la década de 1990 (Minjie, McKenna & Scott 2017) y su desarrollo y mantenimiento ha sido respaldado por dos centros de investigación multi-institucionales patrocinadas por la *US National Science Foundation* (NSF) que son: el *Pacific Earthquake Engineering Research Center* (PEER) cuya misión es desarrollar, validar y diseminar tecnologías para el diseño de estructuras basado en desempeño y la *George E. Brown Jr. Network for Earthquake Engineering Simulation* (NEES) cuya visión era cambiar el énfasis de la investigación de la ingeniería sísmica frente a su dependencia en pruebas físicas para la experimentación integrada, computación, teoría, bases de datos y simulación basada en modelos. (McKenna 2011).

En el año 2015 la NSF reemplazó a NEES por el programa titulado *Natural Hazards Engineering Research Infrastructure* (NHERI) que permite a la comunidad de ingeniería encargada de estudiar los desastres naturales, el acceso a instalaciones experimentales de ingeniería sísmica y eólicas, infraestructura de cómputo, herramientas de modelado y simulación computacional y datos de investigación de la mano con actividades educativas y de alcance comunitario (NSF 2015)

OpenSees (*Open System for Earthquake Engineering Simulation*) es una plataforma informática de Código Abierto que se creó para el desarrollo de aplicaciones de simulación del comportamiento de sistemas estructurales y geotécnicos sometidos a eventos sísmicos. Posee capacidades avanzadas para la modelación y análisis de la respuesta no lineal de sistemas estructurales usando el Método de Elementos Finitos estándar o un análisis con Métodos de Elementos Finitos de

Precisión (McKenna 2011). Además de estas capacidades, OpenSees se ha ampliado para la actualización de parámetros y análisis de sensibilidad, simulación de las estructuras ante los efectos del fuego e interacción fluidos-estructura (Minjie, McKenna & Scott 2017).

OpenSees está basado en POO y está diseñado para ser flexible, extendible y portable. Está escrito básicamente en lenguaje C++ y usa varias librerías numéricas en los lenguajes Fortran y C (McKenna 2011).

5.2 Copyright de OpenSees

OpenSees está lanzado bajo una Licencia Pública General GPL modificada, en la cual los Administradores del software que pertenecen a la Universidad de California permiten, sin cargo económico y sin un acuerdo de licencia por escrito, en primer lugar el uso, la reproducción, la modificación de OpenSees y su documentación por parte de instituciones educativas, de investigación y sin ánimo de lucro solo para fines no comerciales, y en segundo lugar el uso, la reproducción y modificación de OpenSees por parte de otras entidades solo para fines internos lo que permite a los usuarios utilizar el código fuente para el propio uso y no les permite vender un producto que incluya código OpenSees. Sin embargo, las entidades o empresas que desarrollen software comercial pueden obtener permiso de incorporar OpenSees en sus productos con condiciones especiales, contactándose con sus Administradores en la Universidad de California.

Por otra parte, y como se mencionaba en el capítulo anterior respecto a las desventajas de usar un software de código abierto, los administradores de OpenSees suministran el programa y su documentación sin ningún tipo adicional de acompañamiento por parte de ellos y no garantizan

que su operación sea ininterrumpida o libre de errores y, además, no le recomiendan a los usuarios confiar exclusivamente en el programa por ningún motivo. Por esta razón es que los usuarios y los desarrolladores deben ir validando y confirmando la confiabilidad del Software, de acuerdo con los resultados de sus investigaciones y sus experiencias con su uso.

El texto completo de la Licencia o Copyright de OpenSees se encuentra en la Web en el siguiente enlace: <http://opensees.berkeley.edu/OpenSees/copyright.php>

5.3 Arquitectura de OpenSees (Mazzoni S., McKenna F, Scott M., & Fenves G., et al 2007)

Como se mencionó anteriormente OpenSees usa el Método de Elementos Finitos y como tal, su implementación computacional se realiza a través de tres unidades básicas que son en primer lugar el pre-procesamiento que se hace a través de un constructor del modelo o *ModelBuilder*, en segundo lugar el procesamiento que se hace implementando el módulo de Análisis (*Analysis*) y por último el post-procesamiento donde se especifica la salida de resultados en la clase *Recorder*. El funcionamiento de estos tres módulos gira alrededor de un operador central que es el Dominio o *Domain*. En la Figura 1 se muestra un esquema de la implementación de OpenSees en la resolución de un problema estructural.

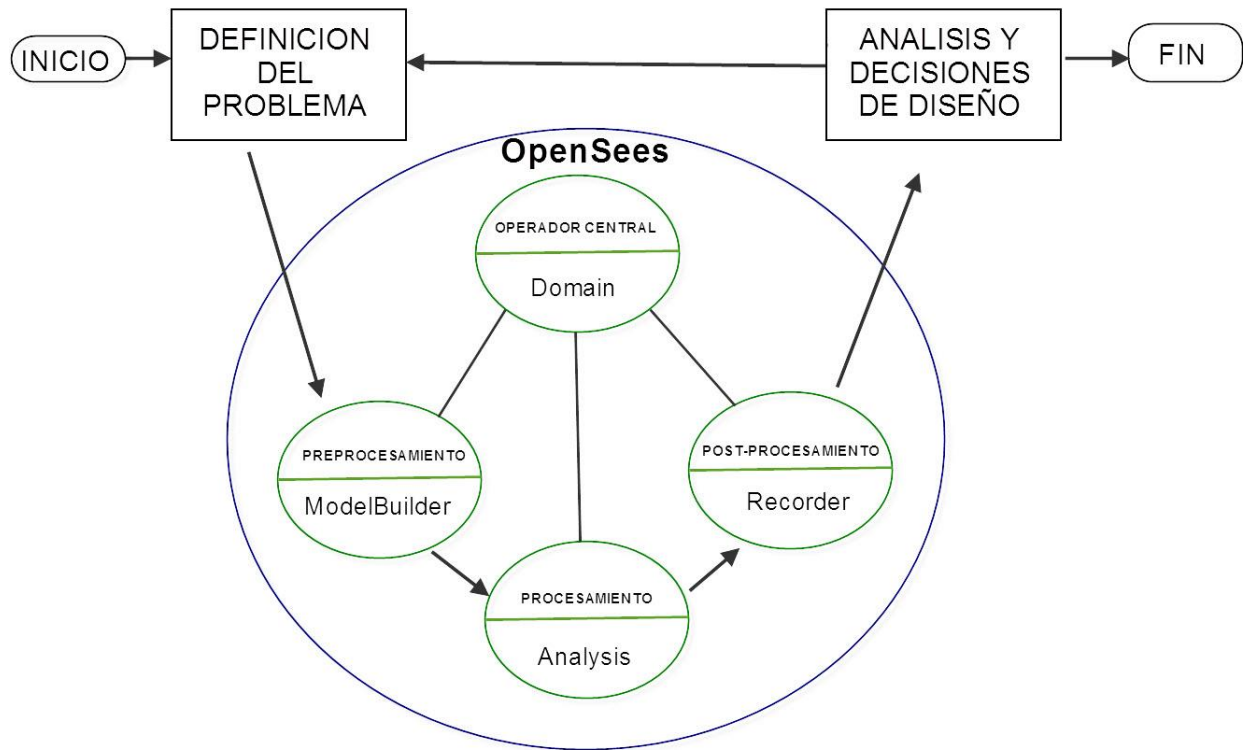


Figura 1. Esquema general de la implementación computacional de OpenSees

Nota: Basado en da Fonseca 2011

A continuación se hace descripción de los componentes y funciones de cada uno de los módulos que forman la arquitectura de OpenSees.

5.3.1 Pre-procesamiento o Módulo *ModelBuilder* En éste módulo se crean los modelos estructurales y geotécnicos mediante la definición de nodos, elementos, materiales, secciones, cargas, y el empleo de diversos tipos de elementos finitos. Se pueden modelar estructuras en 2D y 3D con diferentes grados de libertad.

Como primer paso para definir la geometría del modelo se crean los nodos a los cuales se les asigna sus coordenadas y de manera opcional las masas que le corresponden. Sobre estos nodos se establecen las restricciones que definen el tipo de apoyo externo (articulado, empotrado y demás) y las restricciones multi-punto entre objetos internos del modelo. Para crear los elementos del modelo se le asignan sus nodos extremos. Los elementos se crean dependiendo del tipo de estructura y de análisis que se requieran realizar, para lo cual OpenSees suministra varios tipos como: elementos de cercha, elementos elásticos, elementos no lineales, elementos de longitud cero, elementos planos, entre otros. Incluso también se puede chequear la relación esfuerzo-deformación de vigas, columnas y elementos planos con la definición detallada de sus secciones transversales como por ejemplo, en un elemento de concreto reforzado se pueden representar las capas de los diferentes materiales que la conforma como son el concreto confinado, el acero de refuerzo y el concreto no confinado. Por otra parte, OpenSees contiene una amplia gama de materiales con diferentes características, desde los que simulan un simple comportamiento lineal o bilineal hasta los que incorporan comportamientos más complejos como el isotrópico elástico o el histerético. En la etapa de pre-procesamiento es necesario definir el tipo de transformación de los parámetros estructurales de rigidez y fuerzas resultantes de elementos y secciones de un sistema de coordenadas locales a un único sistema de coordenadas globales. En OpenSees hay tres tipos de transformación que son: la transformación geométrica lineal, la transformación P-Delta y la transformación co-rotacional. Por último, se definen las fuerzas que actúan sobre la estructuras dentro de patrones de carga. En OpenSees es posible considerar las cargas en los nodos tales como cargas gravitacionales, cargas laterales o cargas controladas por desplazamientos en los nodos; también, las restricciones en un punto individual, las cargas sobre los elementos como cargas gravitacionales distribuidas sobre el elemento, registros de aceleraciones definido por el usuario,

registros de desplazamientos sobre nodos específicos, el registro de un sismo, entre otras. (Mazzoni et al., 2007)

En el Manual de Comandos de OpenSees desarrollado por Mazzoni et. al en el año 2007, se encuentra la descripción detallada de los comandos básicos para la creación de los modelos estructurales en OpenSees.

5.3.2 Procesamiento o Módulo *Analysis*. En este módulo se especifica el procedimiento del análisis estructural que se quiere ejecutar, que pueden ser tanto estáticos como dinámicos y lineales como no lineales. Además, en ésta sección se ejecutan los cálculos y operaciones por medio de múltiples sub-componentes relacionados. Según Mazzoni et al. 2007 entre dichos subcomponentes se encuentran los siguientes: *Constrain Handler* que determina cómo se aplican las ecuaciones de las restricciones en el análisis y maneja las condiciones de frontera y los desplazamientos impuestos; *DOF_Numberer* el cual determina la interrelación entre el número de ecuaciones y los grados de libertad; *AnalysisModel* que almacena y proporciona acceso a los grupos de grados de libertad y a los elementos que han sido creados para un análisis y es responsable de construir y devolver una gráfica de la conectividad de todos los grados de libertad que han sido adicionados al modelo; *SolutionAlgorithm* que determina la secuencia de los pasos que se toman para resolver las ecuaciones no lineales en tiempo real. De acuerdo con McKenna, Scott y Fenves 2010 esta clase implementa la solución para determinar la raíz de la ecuación de equilibrio residual, la cual se muestra a continuación.

$$\mathbf{R}[\mathbf{U}(t)] = \mathbf{P}_f(t) - \mathbf{P}_r(t)[\mathbf{U}(t)] = \mathbf{0} \quad (1)$$

donde \mathbf{P}_f es el vector de cargas nodales dependientes del tiempo y \mathbf{P}_r es el vector de fuerzas resistentes el cual está en función de los desplazamientos nodales \mathbf{U} , y se ensambla a partir de contribuciones de elementos. Se toma una iteración aproximada para encontrar la raíz de la Ecuación 1 en el paso de tiempo k

$$\mathbf{U}_k^{j+1} = \mathbf{U}_k^j + \Delta \mathbf{U}_k^{j+1} \quad (2)$$

donde j es en contador de iteraciones del paso de tiempo.

Otro componente del análisis es *Integrator* que determina la ocurrencia anticipada de un paso de tiempo $t+dt$. La clase *SystemofEqn/Solver* especifica como almacenar y resolver los sistemas de ecuaciones en el análisis y por último el subcomponente *Convergence Test* que determina cuando se ha alcanzado la convergencia en el análisis.

5.3.3 Módulo *Recorder* En este módulo se especifica la salida de resultados, supervisa los parámetros preestablecidos en el modelo durante el análisis y pone esta información en un conjunto de archivos definidos por el usuario. Este módulo es importante para documentar el comportamiento del sistema a medida que se realiza el análisis. Hay varios comandos para grabar las respuestas estructurales, en las que incluye, respuesta de los nodos, respuestas envolventes en los nodos, máximo desplazamiento en los nodos, derivas entre nodos, respuestas en los elementos y envolventes de los resultados de los elementos. Este esquema de registro de datos es beneficioso en el sentido de que solo se obtendrán los resultados especificados sin la inclusión de ningún dato extraño o innecesario (Mazzoni S., McKenna F, Scott M., & Fenves G., et al 2007).

5.3.4 Módulo *Domain* En este módulo se realiza el seguimiento de datos seleccionados para ser monitoreados durante el análisis, actúa como un operador central mientras el programa está siendo usado y es el responsable de almacenar los objetos creados en el objeto *ModelBuilder* y de proporcionar acceso a los objetos *Analysis* y *Recorder* en los diferentes pasos de tiempo y procedimientos del análisis (Mazzoni S., McKenna F, Scott M., & Fenves G., et al 2007).

5.4 Intérpretes de OpenSees

Durante los últimos quince años Tcl (*tool command language*) ha sido el principal lenguaje de comandos o tipo *script* al cual han sido vinculados los módulos de construcción del modelo y de análisis en OpenSees, pero a pesar de que éste es un lenguaje muy flexible y tiene muchas fortalezas en el procesamiento de cadenas de caracteres, no todos los usuarios de OpenSees están familiarizados con su funcionamiento, por lo tanto, y con la finalidad de permitir a los usuarios y desarrolladores que conocen más de otros lenguajes como MATLAB, Python, Ruby, entre otros y dentro del constante desarrollo de OpenSees, recientemente se creó la clase *DL_Interpreter* que permite el uso de múltiples intérpretes. Con esta herramienta se ha podido desarrollar la aplicación *OpenSeesPy* que usa lenguaje *Python* y queda abierta la posibilidad de desarrollar aplicaciones que vinculen otros lenguajes (Minjie, McKenna & Scott 2017).

Por otra parte, cuando los modelos que se van a analizar en OpenSees son pequeños y con un solo procesador basta para ejecutarlo en un tiempo prudencial, se dice que es una aplicación secuencial, por otra parte, si la estructura es grande y más compleja es necesario usar varios procesadores que trabajen en conjunto se dice que es una aplicación en paralelo. A continuación se presenta de manera detallada cada una de éstas aplicaciones.

5.4.1 Aplicaciones Secuenciales de OpenSees. Un ambiente secuencial es cuando un solo procesador maneja todo, desde correr el intérprete hasta resolver los sistemas de ecuaciones de los modelos (QuakeCore 2016).

5.4.1.1 *Opensees.exe*. OpenSees.exe es una implementación del intérprete Tcl para el análisis de elementos finitos. El código con el cual se construye el modelo y se define el tipo de análisis y los datos de salida requeridos, se escribe en un archivo de texto el cual tiene la extensión “.tcl.” Cuando OpenSees.exe se ejecuta se abre la terminal de DOS (*Disk Operating System*) desde donde es posible invocar el mencionado archivo con extensión “.tcl”, con lo cual se realiza el análisis y al final de éste, se crean los archivos de texto que contiene los resultados requeridos. Como guía para el aprendizaje de la aplicación, en la página web de usuario de OpenSees (http://opensees.berkeley.edu/wiki/index.php/OpenSees_User) se pueden consultar un manual para el uso del programa y varios ejemplos ilustrativos.

5.4.1.2 *OpenseesPy*. OpenSeesPy es una implementación del intérprete Python para el uso con OpenSees. Al igual que en OpenSees.exe se escribe un código para definir el modelo, análisis y salida de datos pero esta vez ya no en el lenguaje Tcl, sino usando comando propios del lenguaje Python. Dicho código es ejecutado en una aplicación propia del lenguaje Python (e.g. Python 3.6 o Anaconda 5.0) y al final arroja los resultado ya sea en pantalla o en archivos de salida, según lo requiera el usuario. Para mayor información las instrucciones de instalación en Windows y en Linux, tutoriales y ejemplos sencillos de su uso se encuentran en la página web *The OpenSeesPy Library* (<http://openseespydoc.readthedocs.io/>).

Con el fin de comparar la estructura de los *scripts* generados tanto en Opensees.exe y OpenSeesPy, tomaremos como ejemplo una estructura sencilla que corresponde a una cercha con tres barras elásticas sujeta a la acción de cargas estáticas, tal como se muestra en la Figura 2.

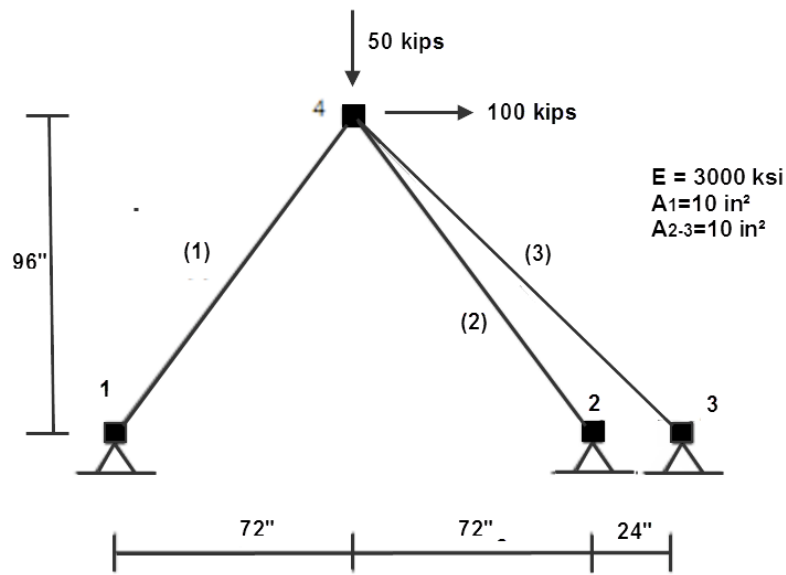


Figura 2. Modelo de cercha

Nota: Modificada de http://opensees.berkeley.edu/wiki/index.php/Basic_Truss_Example

El código en el lenguaje Tcl para el desarrollo en OpenSees.exe fue tomado de la página web http://opensees.berkeley.edu/wiki/index.php/Basic_Truss_Example y el código en el lenguaje Python para OpenSeesPy fue tomado de la página web <http://openseespydoc.readthedocs.io/en/latest/truss.html>.

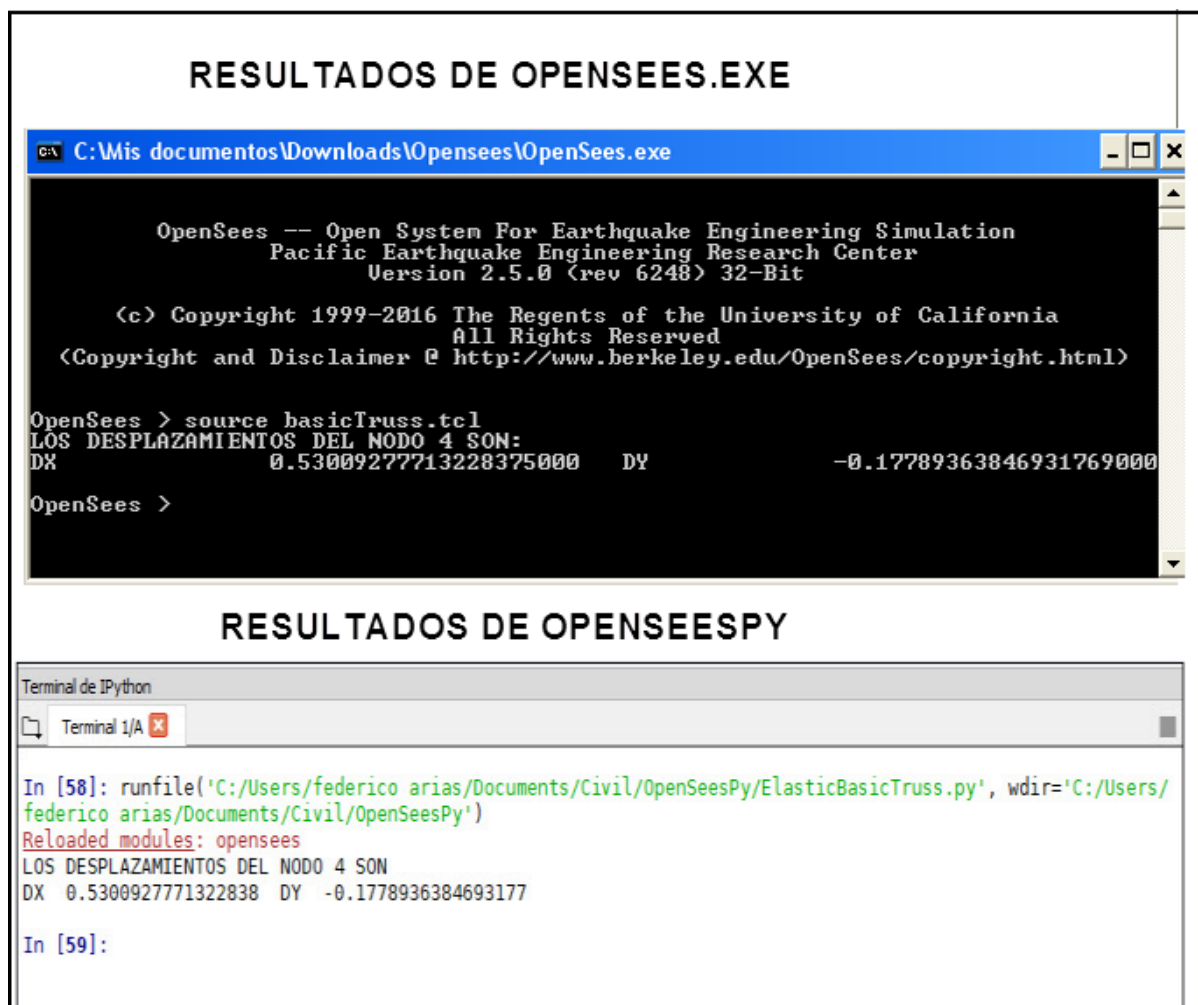
Comparando la estructura de los *scripts* tanto de OpenSees.exe y OpenSeesPy se observa que su estructura es similar en los dos lenguajes y que las diferencias existentes son propias de la sintaxis de cada uno de ellos tal como se muestra en la Figura 3.

OpenSees.exe (Tcl)	OpenSeesPy (Phyton)	
wipe	wipe()	Creación del modelo 2D y 2 grados de libertad por nodo.
model BasicBuilder -ndm 2 -ndf 2	model('basic', '-ndm', 2, '-ndf', 2)	
node 1 0.0 0.0 node 2 144.0 0.0 node 3 168.0 0.0 node 4 72.0 96.0	node(1, 0.0, 0.0) node(2, 144.0, 0.0) node(3, 168.0, 0.0) node(4, 72.0, 96.0)	Creación de nodos asignando un ID y las coordenadas.
fix 1 1 1 fix 2 1 1 fix 3 1 1	fix(1, 1, 1) fix(2, 1, 1) fix(3, 1, 1)	Condiciones de borde.
unialMaterial Elastic 1 3000	unialMaterial("Elastic", 1, 3000.0)	Creación de un material elástico asignando un ID y el módulo de elasticidad.
element Truss 1 1 4 10.0 1 element Truss 2 2 4 5.0 1 element Truss 3 3 4 5.0 1	element("Truss",1,1,4,10.0,1) element("Truss",2,2,4,5.0,1) element("Truss",3,3,4,5.0,1)	Definición de los elementos tipo cercha asignando un ID, los nodos extremos, el área y el ID del material.
timeSeries Linear 1 pattern Plain 1 1 { load 4 100 -50 }	timeSeries("Linear", 1) pattern("Plain", 1, 1) load(4, 100.0, -50.0)	Definición de los patrones de carga y la creación de carga aplicada asignando el nodo donde esta aplicada y su magnitud, la respectiva dirección.
system BandSPD numberer RCM constraints Plain integrator LoadControl 1.0 algorithm Linear analysis Static	system("BandSPD") numberer("RCM") constraints("Plain") integrator("LoadControl", 1.0) algorithm("Linear") analysis("Static")	Generación del análisis, creando el sistema de ecuaciones, el algoritmo para manejar los grados de libertad, el manejo de las restricciones, el sistema de integración, el algoritmo de solución y el tipod de análisis.
analyze 1	analyze(1)	Ejecución del análisis
puts "LOS DESPLAZAMIENTOS DEL NODO 4 SON:" puts "DX [nodeDisp 4 1] DY [nodeDisp 4 2]"	print('LOS DESPLAZAMIENTOS DE NODO 4 SON:') print('DX ', nodeDisp(4,1), ' DY ', nodeDisp(4,2))	Información que se quiere imprimir en la pantalla.

Figura 3. Comparación de códigos de OpenSees.exe y OpenSeesPy.

Nota: Elaborada por el autor.

De igual manera, ejecutando cada uno de los códigos en sus respectivas aplicaciones, los resultados obtenidos son iguales para las dos aplicaciones según se observa en la Figura 4. Por tanto, la decisión de usar una u otra aplicación simplemente depende de con cual lenguaje este más familiarizado el usuario, ya sea con Tcl o con Python, ya que las herramientas de análisis son las mismas que ofrece la Plataforma de OpenSees tanto para OpenSees.exe como para OpenSeesPy.



.Figura 4 Comparación de los resultados en OpenSees.exe y OpenSeesPy.

Nota: Elaborada por el autor.

5.4.2 Aplicaciones en Paralelo de OpenSees. Un análisis estructural dinámico y no lineal de alto nivel típicamente requiere considerables recursos de computación, sus inmanejables tiempos de ejecución son un obstáculo significativo en el desarrollo de métodos de diseño avanzado en ingeniería sísmica. Una prueba hecha de un análisis dinámico no lineal sobre un modelo definido como un edificio de cinco pisos sometido a cuarenta segundos de sismo, puede tomar hasta cinco meses de tiempo de cálculo en un computador moderno. Los incrementos en los tiempos de cálculo se deben primero, a la alta no linealidad, segundo a que las mallas para los modelos numéricos pueden requerir un refinamiento mayor desde un elementos viga-columna ordinarios hasta elementos más finos como *shell* o sólidos, y por último a que los incrementos de tiempo (Δt) son pequeños con el fin de lograr mayor precisión en los cálculos. Se han propuesto varias estrategias de paralelización con el fin de reducir los tiempos de cálculo para que los ingenieros puedan completar las simulaciones en un marco de tiempo razonable, adaptando los programas para que no solo funcionen secuencialmente sino también en paralelo (Yang, Yang & Hsieh 2014).

Un computador en paralelo es un máquina de cálculo que está compuesta por más de un procesador los cuales permiten procesos para comunicarse uno con otros, ya sea en forma de memoria compartida o paso de mensajes (McKenna & Fenves 2008).

La plataforma de OpenSees proporciona clases para interactuar con la interfaz de paso de mensajes (*MPI* por sus siglas en inglés) que es un estándar que define la sintaxis y la semántica de las funciones contenidas en una biblioteca. Para máquinas en paralelo que tengan esta librería, se pueden construir dos intérpretes desde OpenSees: OpenSeesSP y OpenSeesMP.

5.4.2.1 OpenSeesSP: (Intérprete de OpenSees en Paralelo Simple). OpenSeesSP es un aplicación cuyos usuarios tienen poco conocimiento en computación en paralelo y que sirve para

el análisis de modelos muy grandes, con archivos de entrada muy extensos y que toman mucho tiempo en correr en un máquina secuencial (un solo procesador). Este intérprete procesa el mismo *script* que el intérprete OpenSees procesaría sobre una máquina secuencial. No hay comandos especiales para los procesos en paralelo, aunque hay opciones adicionales cuando se trata de elegir *solvers* (McKenna & Fenves 2008).

5.4.2.2 OpenSeesMP: (Intérprete de OpenSees en Paralelo Múltiple). OpenSeesMP es una aplicación de OpenSees que se usa para correr muchos modelos de tamaño pequeño a moderado como por ejemplo estudios tipo parámetro, o para correr más complicados *scripts* en los cuales el usuario especifique los subdominios, numeradores paralelos y solucionadores de ecuaciones paralelos. Cuando se ejecuta la aplicación sobre un computador en paralelo, cada procesador está corriendo el mismo intérprete en el mismo *script* principal de entrada. El usuario controla a nivel de línea de comando y a nivel de *scripts* para especificar los diferentes trabajos que cada procesador realiza lo que requiere mayores conocimientos en programación (McKenna & Fenves 2008).

Una guía para compilación de las aplicaciones en paralelo de OpenSees se encuentran en la página web *Compilation Guideline of OpenSeeMP on Linux Machines* (http://opensees.berkeley.edu/wiki/index.php/Compilation_Guideline_of_OpenSeeMP_on_Linux_Machines).

Los códigos binarios de ambos intérpretes en paralelo para computadoras multi-core o de procesadores agrupados que corren en los sistemas operativos Microsoft Windows o Apple se pueden encontrar en la página web *OpenSees Parallel* (<http://OpenSees.berkeley.edu/OpenSees/parallel/parallel.php>). Estas aplicaciones requieren de la

instalación de otro software adicional como MPICH2 para Microsoft y OpenMPI para Apple. Los ejecutables de los dos intérpretes están disponibles en la página web de descarga de OpenSees (<http://opensees.berkeley.edu/OpenSees/parallel/download.php>) (McKenna & Fenves 2008).

Sin embargo, en la página web *Discovery WorkSpace* de la *National Natural Hazards Engineering Research Infrastructure* (NHERI) (<https://www.designsafe-ci.org/rw/workspace/#/>) se le permite a los usuarios realizar simulaciones y analizar datos utilizando los códigos de estas dos aplicaciones en paralelo de OpenSees tal como se muestra en la Figura 5.

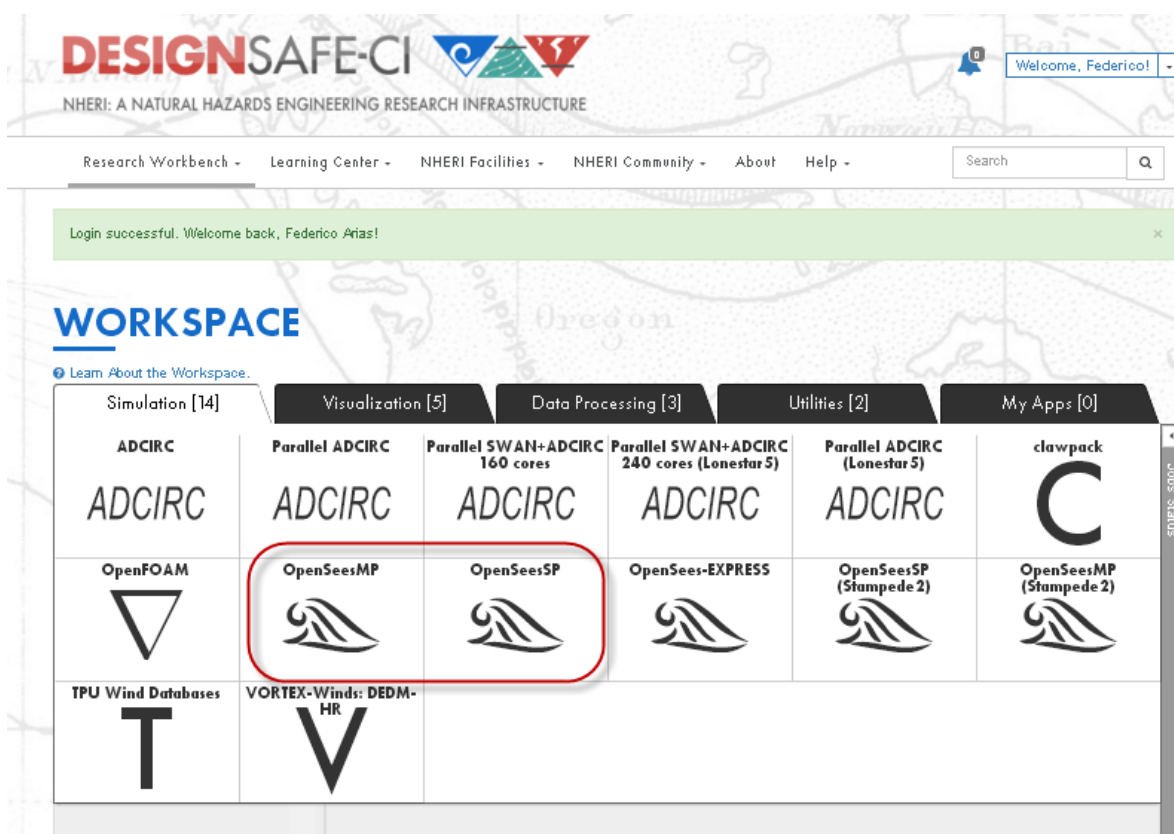


Figura 5. Aplicaciones OpenSeesMP y OpenSeesSP en la web.

En el estudio realizado por Zhen, Xinzheng & Kincho 2016 se demostró que una plataforma computacional distribuida y paralela es rentable, eficiente y flexible para la simulación de modelos de diferentes grados de confiabilidad o precisión. Dichos investigadores tomaron un caso de

estudio, empleando OpenSees, donde idealizaron una ciudad virtual constituida por 50 modelos que requerían alta precisión en el análisis, en los cuales se usó un balanceo de cargas dinámicas y además por otros 100.000 modelos en los que se requería una precisión moderada usando un balanceo de cargas estáticas. El tiempo de simulación del daño sísmico lo ejecutaron en 48 horas con una plataforma distribuida y paralela, logrando 52 veces de mejora cuando se compara con el tiempo de computación requerido para la misma simulación en una sola computadora (Zhen, Xinzheng & Kincho 2016). Por tanto, para el estudio y análisis de proyectos de gran envergadura resulta eficiente el uso de las aplicaciones en paralelo que ofrece la Plataforma de OpenSees en la web, que además es una solución rentable porque también son de acceso libre.

5.5 Interfaces gráficas de usuario de OpenSees

El conjunto de paquetes comerciales de la firma *Computers and Structures, Inc.* (ETABS, SAP2000 y Perform-3D) que están, quizás, entre los más utilizados en la industria estructural y de ingeniería sísmica, están equipados con interfaces gráficas de usuario (*GUIs* por sus siglas en inglés) que son amigables con el usuario y que le permiten un análisis de las estructuras eficiente y preciso (AlHamayden, Najib & Alawnah 2016).

La principal limitación de OpenSees es que requiere del uso de comandos de lenguajes de *scripts* basados en cadenas de caracteres (Tcl y Python) para construir modelos y realizar análisis. Por tanto, el usuario necesita tener habilidades de programación fluidas para realizar los análisis y los chequeos requeridos en los diseños e investigaciones (AlHamayden, Najib & Alawnah 2016).

Más importante aún, la mayor parte de los resultados numéricos, especialmente cuando son de gran tamaño, requieren el desarrollo de herramientas de post procesamiento específicas (por

ejemplo, MATLAB u otros idiomas) para una interpretación más fácil a través de trazado o visualización. Estas deficiencias son a menudo un factor disuasivo para que los usuarios nuevos e inexpertos adopten OpenSees en su investigación, por lo general recurriendo a otros paquetes de software con capacidades gráficas (Papanikolaou, Kartalis-Kauonis, Protopapadakis & Papadopoulos 2017).

Para superar este obstáculo y/o desventaja, algunas *GUIs* pre/post-procesamiento se han desarrollado, de las cuales algunas de éstas se describen a continuación.

5.5.1 BuildingTcl/BuildingTclViewer. *BuildingTcl* está desarrollado para proporcionar una biblioteca de comandos Tcl que permiten el uso de OpenSees a usuarios con poca habilidades en programación. En la Figura 6 se muestra *BuildingTclViewer* que es una *GUI* que facilita la creación de bases de datos *BuildingTcl*, ejecutando simulaciones en OpenSees, así como la visualización de resultados de forma interactiva. Permite el acceso a una biblioteca predefinida de diferentes materiales estructurales, secciones y elementos que pueden usarse para generar construcciones modelos y análisis de datos (AlHamayden, Najib & Alawnah 2016).

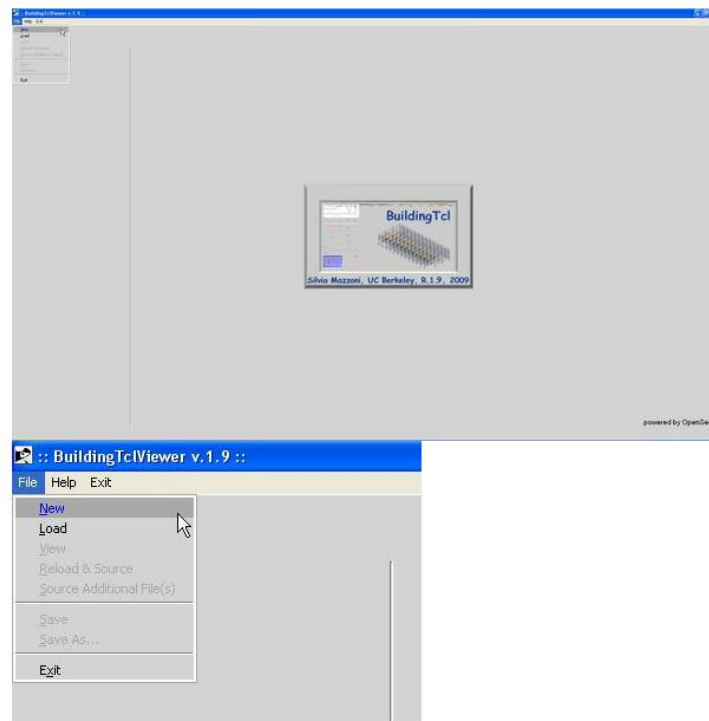


Figura 6. BuildingTclViewer GUI.

Nota: Tomado de <http://opensees.berkeley.edu/>

5.5.2 OpenSees Navigator. Otra GUI pre/post procesamiento para OpenSees es *OpenSees Navigator*. Básicamente, es una herramienta basada en MATLAB que se ejecuta en múltiples plataformas que ofrece una funcionalidad similar a *BuildingTcl/BuildingTclViewer*. Específicamente, proporciona plantillas de geometría paramétrica que pueden usarse para modelar rápidamente algunos de los sistemas estructurales comunes tal como se muestra en la Figura 7. También viene con secciones transversales de perfiles de acero estructural integradas junto con sus propiedades predefinidas; la base de datos de secciones transversales incluye AISC (Instituto Americano de Construcción de Acero) y CISC (Instituto Canadiense para la Construcción de Acero). Eso también permite el pre/post procesamiento y la ejecución de simulaciones de OpenSees sin programación de secuencias de comandos Tcl. Como tal, libera a los usuarios para

crear modelos de dos o tres dimensiones en OpenSees con relativa facilidad. Además, el post-procesamiento se procesa y administra de manera transparente con cajas de herramientas incorporadas. *OpenSees Navigator* también sirve como una gran herramienta para visualizar comportamiento de los sistemas estructurales, como las fuerzas de los elementos estructurales, desplazamientos, la forma de los modos, etc. (AlHamayden, Najib & Alawnah 2016).

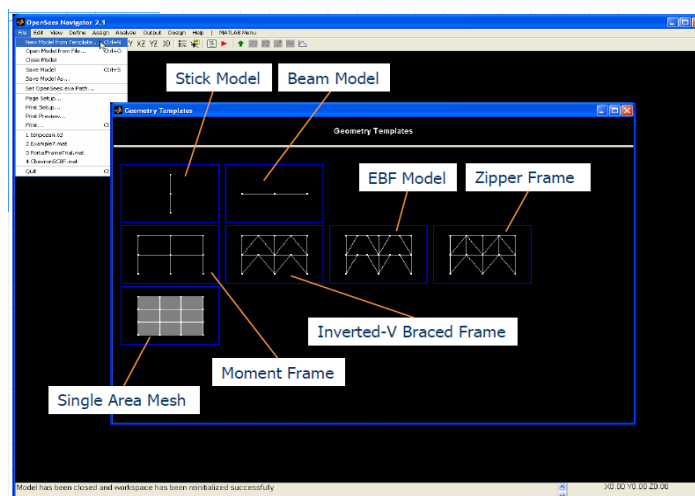


Figura 7. Interface Gráfica de Usuario de OpenSees Navigator.

Nota: Tomado de <http://peer.berkeley.edu/OpenSeesNavigator>

BuildingTcl y *OpenSees Navigator*, típicamente restringen al usuario a un tipo particular de estructuras y/o análisis (Minjie, McKenna & Scott 2017).

5.5.3 GiD+OpenSees. Recientemente se ha desarrollado una nueva *GUI* que sin problema conecta el OpenSees *solver* con el pre/post procesamiento general GiD, desarrollada por el Centro International para Métodos Numéricos en Ingeniería (*CIMNE* por sus siglas en Catalán), el cual no solo proporciona un ambiente gráfico completo sino también un lenguaje de *script* basado en

Tcl para generar archivos de entrada personalizados compatibles con cualquier procesador de texto (Papanikolaou, Kartalis-Kauonis, Protopapadakis & Papadopoulos 2017).

La estructura de la interfaz *GiD+OpenSees* se representa en la Figura 8. El modelo se crea en el procesador gráfico *GiD* usando el procedimiento estándar *point and click* (apuntar y hacer clic) para la geometría (puntos, líneas, superficies, volúmenes) con la ayuda de varios diálogos de usuario que contienen los comandos nativos OpenSees para definir y la asignar las propiedades del modelo y del análisis, junto con un mallado robusto proporcionado por el preprocesador mismo(Papanikolaou, Kartalis-Kauonis, Protopapadakis & Papadopoulos 2017).

Después de completar el modelo, un simple *click* crea un archivo de datos del modelo compatible con OpenSees e invoca el *solver* para correr el análisis. Los archivos de los resultados de salida (.out) son traducidos automáticamente al formato requerido e importados en el post-procesador. Entonces el modelo se abre en el ambiente de post procesamiento, donde todas las herramientas gráficas y de visualización están disponibles. (Papanikolaou, Kartalis-Kauonis, Protopapadakis & Papadopoulos 2017).

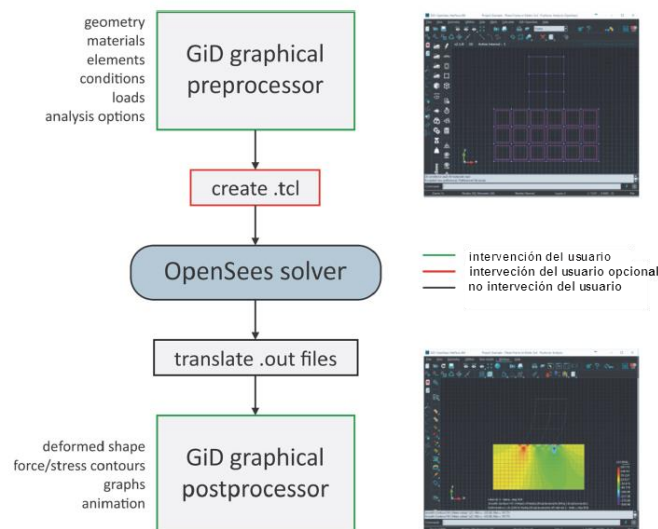


Figura 8. Estructura de Interface Gráfica GiD+OpenSees

Nota: Adaptado de Papanikolaou, Kartalis-Kauonis, Protopapadakis & Papadopoulos 2017.

En la página web <http://gidopensees.rclab.civil.auth.gr/> se encuentra las descarga, tutoriales y ejemplos y características de la aplicación.

5.5.4 Buil-X. *Build-X* es un programa basado en Windows desarrollado recientemente, que está vinculado con OpenSees para realizar los análisis de elementos finitos y predecir la respuesta sísmica de edificaciones en 3D. Esta aplicación posee una *GUI* sencilla que proporciona un guía práctica al usuario en el modelamiento eficiente y confiable de la estructura y además permite que las tareas propias de post-procesamiento sean fáciles de manejar. De acuerdo con Psyrras y Sextos 2018 se ha hecho estudios de comprobación de la eficiencia del programa y la confiabilidad de los resultados señalando que *Buil-X* puede servir para el análisis sísmico de edificaciones nuevas y la evaluación de edificaciones existente con un bajo costo computacional. (Psyrras & Sextos 2018). En la Figura 9 se muestra un ejemplo de pre-procesamiento en *Build-X* donde se observa cómo se define de manera detallada la sección de los elementos y su ubicación dentro de la planta del edificio.

La información de las características, instalación, tutoriales y ejemplos del uso de *Buil-X* se encuentra página web <https://www.buildx4opensees.eu>.

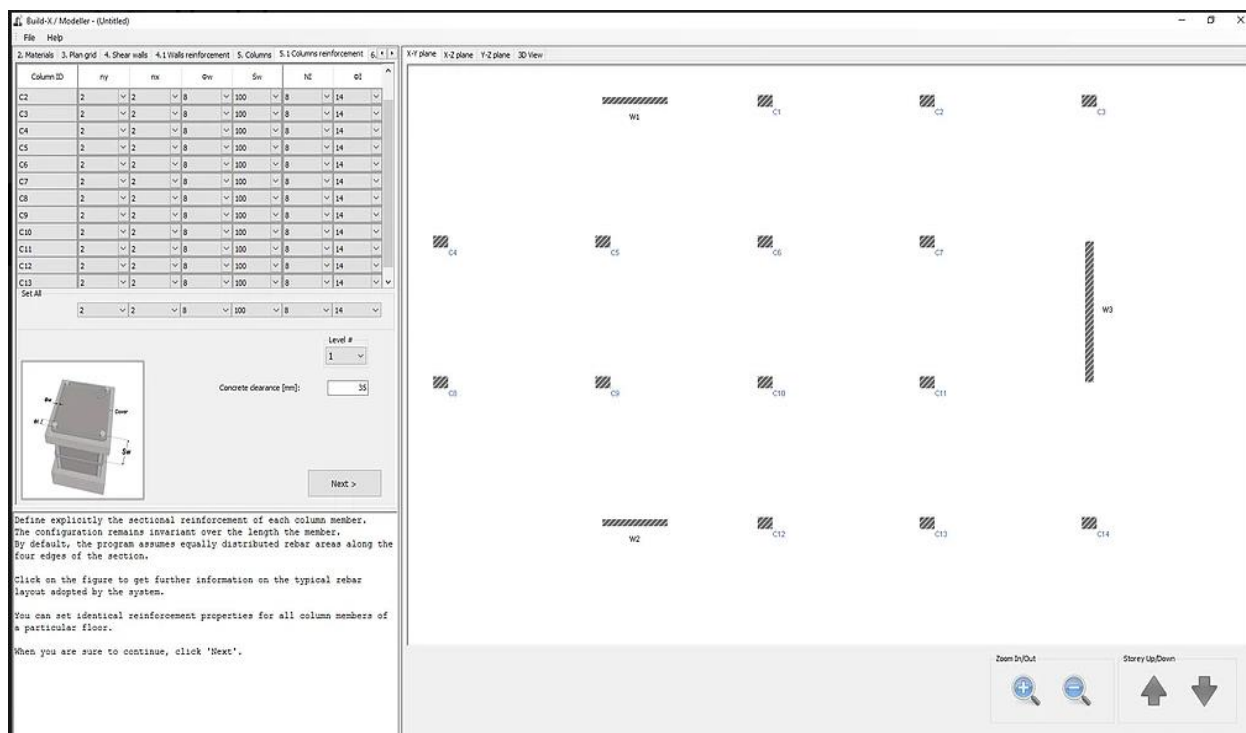


Figura 9. Ejemplo de modelado de la estructura en Build-X: A la izquierda el cuadro de las columnas con su sección detallada y la derecha su ubicación en la planta del edificio.

Nota: Adaptada de <https://www.buildx4opensees.eu>.

5.6 OpenSees frente al Software Comercial de Análisis Estructural

Teniendo en cuenta lo mencionado por Lu X., Xie L., Guan H., Huang L. & Lu X. 2015 referente a las ventajas que ofrece OpenSees frente a software comercial de análisis estructural, en la Tabla 2 se hace un comparativo entre el uso de dichos programas.

Tabla 2.

Comparativo entre OpenSees y otros software comerciales de análisis estructural

Propiedad	OpenSees	Softwares Comerciales
Ofrece avances en términos de estabilidad computacional.	Si	Si
Comodidad del uso de interfaces gráficas de usuario (GUI).	En desarrollo	Avanzado
Costos	Sin costo	Altos costos
Disponibilidad de código fuente.	Si	No
Permite la investigación y discusión de sus mecanismos internos y su funcionalidad.	Si	No
Se pueden agregar fácilmente nuevos módulos en el código	Si	No
Puede ser compartido por la comunidad de investigación	Si	No

Según lo expuesto por McKenna 2011, algunas *GUIs* comerciales que usan OpenSees como su motor de análisis, están disponibles para varias compañías de software pequeñas como *nvStructural* de *Novel CAE Solutions* que se puede consultar en la página web <http://www.noveltech.us/?page=nvStructural> y *CDS Win* de *Software Tecnico Scientifico* cuya información se encuentra en la página web <http://www.efddesignweb.it/CDSwin.htm>.

De igual manera, *Cype* (Desarrollador de software para ingeniería proveniente de España) ha integrado como su motor de cálculo para sus módulos *StruBIM Analysis*, *StruBIM Desing* y *StruBIM Foundations*, programas que se encargan del análisis, diseño y chequeo de estructuras incluyendo sus cimentaciones (Barceló J. & Fernández M. 2017).

Por otra parte y con el fin de establecer la confiabilidad de OpenSees frente a los programas de análisis estructural comerciales se han desarrollado varios estudios comparativos de su

funcionamiento. A continuación se exponen los aspectos más relevantes de las comparaciones hechas en algunos de esos estudios.

5.6.1 OpenSees y M.S.C. Marc. El sistema Marc de la firma *M.S.C. Software*, contiene una serie de programas integrados que facilitan el análisis de problemas de ingeniería en los campos de mecánica estructural, transferencia de calor y electromagnetismo (MSC System 2001).

Los códigos comerciales de elementos finitos de *M.S.C. Marc* han sido usados ampliamente en análisis no lineal de edificios altos y han sido validados en términos de veracidad y precisión (Lu X., Xie L., Guan H., Huang L. & Lu X. 2015).

Con el fin de validar el uso de OpenSees , Lu X. et al., en el año 2015, expusieron un estudio donde se utilizaban tanto OpenSees como en M.S.C. Marc, en el análisis sísmico de un edificio de gran altura (148.1 m.), que tiene como modelo estructural muros de cortante en concreto reforzado y la configuración que se muestra en la Figura 10. En primer lugar, dichos investigadores, realizaron análisis por gravedad y modal y compararon los resultados obtenidos encontrando que los períodos de la estructura calculados en cada uno de los programas tenían diferencias relativas no mayores al 1.3%, tal como se muestra en la Tabla 3, lo que valida la veracidad del modelo de OpenSees en la etapa elástica.

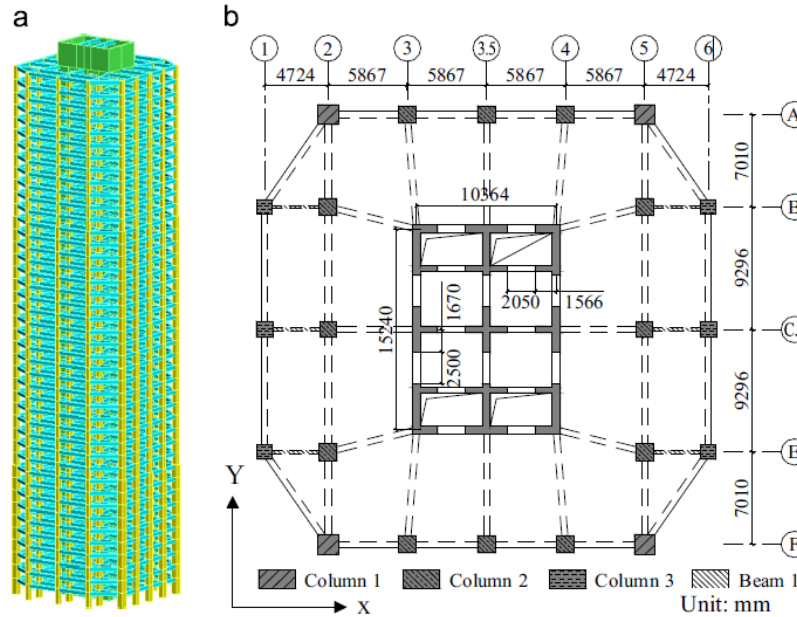


Figura 10. Edificio en estudio : (a) elevación y (b) planta.

Nota: Tomado de Lu X., Xie L., Guan H., Huang L. & Lu X. 2015

Tabla 3.

Comparación de los períodos previstos por OpenSees and MSC.Marc.

	Marc (s)	OpenSees (s)	Error relativo
T ₁ (1er-orden translación en dirección X)	1.791	1.815	1.3%
T ₁ (1er-orden translación en dirección Y)	1.580	1.579	0.06%
T ₁ (1er-orden torsión)	0.900	0.890	1.1%

Nota: Adaptado de Lu X. et al. 2015

En seguida realizaron sobre la estructura un análisis *pushover* considerando los efecto P-Delta y la ayuda de unas gráficas comparativas de los resultados de desplazamiento en el techo versus cortante en la base, concluyeron que hay una concordancia aceptable en los resultados de los dos programas lo que se muestra de manera aproximada en la Tabla 4, donde las diferencias no superan el 6% .

Tabla 4.

Comparación de los resultados del cortante en la base por OpenSees and MSC.Marc.

Desplazamiento en el techo (m)	Cortante en la Base (10^3 kN)		
	Marc	OpenSees	Error relativo
0.5	70	74	5.4%
1.0	90	94	4.4%
1.5	98	103	5.1%
2.0	102	107	4.9%

Nota: Elaborada por el autor con base en los resultados de Lu X., Xie L., Guan H., Huang L. & Lu X. 2015

De igual manera en la Tabla 5 se hace una comparación de la distribución de los porcentaje de deriva entre los pisos donde se muestran la diferencia entre los resultados de OpenSees y *Marc* cuyo valor aumenta en los piso superiores hasta un porcentaje de 8.1% que continúa siendo aceptable.

Tabla 5.

Comparación de los resultados del radio de deriva de entrepisos OpenSees and MSC.Marc.

Pisos	Radio de deriva entrepisos		
	Marc (%)	OpenSees (%)	Error relativo
10	1.36	1.28	5.9%
20	1.60	1.61	0.6%
30	1.65	1.74	5.5%
40	1.60	1.73	8.1%

Nota: Elaborada por el autor con base en los resultados de Lu X., Xie L., Guan H., Huang L. & Lu X. 2015

Por último realizaron análisis dinámicos no lineales del edificio imponiendo sobre la estructura los registros de aceleraciones de tres sismos diferentes (ChiChi, Centro y Northridge) escalados a diferentes valores de aceleración pico del suelo y consiguieron hacer diferentes comparaciones de los desplazamientos del techo de la estructura a lo largo del tiempo y donde se observa como en todos los casos los resultados de desplazamientos en los dos programas son muy parecidos hasta transcurridos 20 segundos del evento sísmico, sin embargo, después de este tiempo, en la mayoría de las comparaciones, se empiezan a observar diferencias donde por lo general OpenSees arroja desplazamientos mayores, diferencias que pueden presentarse por posibles variaciones en los enfoques con que cada programa aborda el problema de la no linealidad de la estructura. De igual manera se realizó una comparación de las envolventes de los porcentajes de distribución de derivas de los entrepisos, donde se observa como los valores son similares hasta el piso 30, de este piso en adelante empieza a incrementarse las diferencias progresivamente hasta el último piso alcanzando una diferencia máxima de error de 9.3%, que para los autores del estudio es considerada como aceptable.

Es de anotar que en la mayoría de las comparaciones, donde se presentan diferencias, los desplazamientos arrojados por OpenSees son mayores que en M.S.C. Marc, razón por la cual se podría decir que OpenSees es un poco más conservador y que de acuerdo con Lu X. et al. 2015, es una herramienta confiable y factible para el análisis no lineal de este tipo de edificios.

5.6.2 OpenSees, Sap2000 y Perform-3D.

Los programas Sap2000 y Perform-3d de propiedad de la firma *Computers and Structures, Inc.*, están entre los más usados en el análisis en el diseño estructural y en el análisis sísmico. Con

el fin de comparar el funcionamiento de éstos dos programas en relación a OpenSees se toman dos estudios realizados previamente.

El primer caso es un estudio hecho por Ladjinovic et. al. (2012) quienes analizaron un edificio de cinco pisos de concreto reforzado, con la configuración y las cargas impuesta que se muestra en la Figura 11. El método de análisis utilizado fue *pushover* teniendo en cuenta los efectos P-Delta, teniendo en cuenta tres modelos diferentes, dos utilizando diferentes tipos de concreto de la librería de OpenSees: Concrete02 cuyas características se pueden consultar en la página web [http://opensees.berkeley.edu/wiki/index.php/Concrete02_Material -- Linear Tension Softening](http://opensees.berkeley.edu/wiki/index.php/Concrete02_Material_-_Linear_Tension_Softening) y Concrete07 que se puede consultar en <http://opensees.berkeley.edu/OpenSees/manuals/usermanual/4055.htm> y un tercer modelo en Sap2000 utilizando el modelo de Mander .

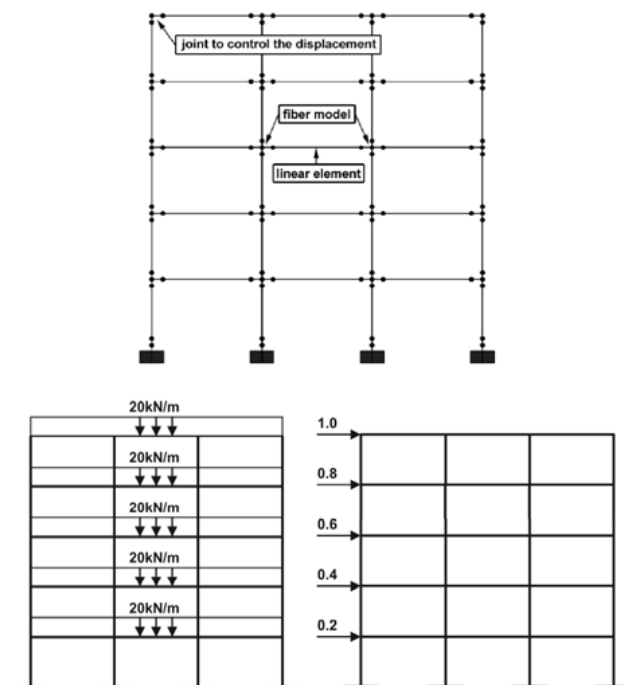


Figura 11. Modelo del pórtico y cargas aplicadas.

Nota: Tomado de Ladjinovic Dj., Raseta A., Radujkovic A., Folic R. & Prokic A 2012

Después de realizado lo análisis los autores del estudio construyeron las curvas de carga-desplazamiento de la estructura donde se observa que para los dos modelos el de OpenSees y para el de Sap2000 que usa sección bruta del modelo de Mander (curvas B, C y D de la Figura 12) en el rango elástico no se observan diferencias significativas, lo que no ocurre con el modelo de Sap2000 que usa una sección reducida del modelo de Mander que presenta mayores deformaciones con menor carga.

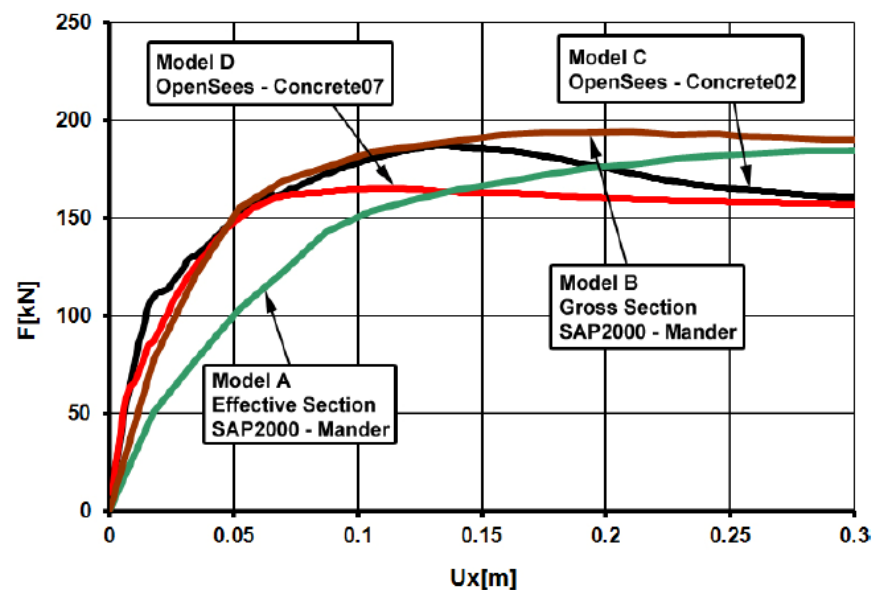


Figura 12. Curvas Pushover

Nota: Tomado de Ladjinovic Dj., Raseta A., Radujkovic A., Folic R. & Prokic A 2012

El en comportamiento no lineal más pronunciado los autores reportan diferencias significativas en la relación fuerza-desplazamiento. Además, que los modelos B (Sap2000) y D (OpenSees) se caracterizan por una leve disminución de la fuerza con el incremento del desplazamiento y que para alcanzar el mismo valor de desplazamiento se necesita más carga lateral en el modelo B (Sap2000). Por otra parte, concluyen que en el modelo A (Sap2000) se presenta un incremento permanente de la fuerza, mientras en el modelo C (OpenSees) se observa una fuerza decreciendo

para aumentar el desplazamiento. Y por último que en los modelos A y B se presentan diferencias significativas en los valores de las fuerzas laterales para el mismo desplazamiento lateral del nodo de control debido a los diferentes valores de la rigidez efectiva de la sección transversal (Ladjinovic Dj., Raseta A., Radujkovic A., Folic R. & Prokic A 2012).

De acuerdo a lo anterior los autores concluyen que los datos obtenidos del análisis *pushover* usando Sap2000 y OpenSees tiene resultados satisfactorios desde el punto de vista de la relación esfuerzo-desplazamiento, excepto en caso de los cálculos con secciones efectivas de modelo A, ya que sus valores de rigidez son reducidos respecto a las demás. Por lo anterior, es de anotar que es muy importante un modelamiento detallado de las secciones de los elementos de las estructuras, el cual debe ser lo más próximo a la realidad para que las simulaciones proporcionen datos más confiables para la toma de decisiones en el análisis y diseño de las edificaciones.

El segundo caso de comparación está tomado de Swensen y Kunnath (2012) que con el fin de evaluar el comportamiento de los diferentes modelos y métodos usados por los programas Sap2000, Perform-3D y OpenSees, en términos de sus capacidades predictivas y sensibilidad de respuesta se comparan con los datos grabados de respuesta de un edificio de nueve pisos instrumentado con 15 acelerómetros y que tiene pórticos resistentes a momento (PRM) de acero, cuyas dimensiones se muestran en la Figura 13.

En el trabajo de Swensen y Kunnath (2012) fueron desarrollados dos modelos en dos dimensiones usando los tres programas mencionados anteriormente. En el modelo se incluye un pórtico de gravedad equivalente para considerar la contribución de la rigidez del sistema de pórticos de gravedad y para poder dar cuenta de cualquier fuerza lateral inducida por los efectos P-delta dentro del sistema de pórticos de gravedad. También se incluyen, en el modelo, columnas

falsas que calibran el modelo con el fin de tener en cuenta la rigidez de los elementos que no están explícitos tales como los componentes no estructurales, muros divisorios, entre otros. Los modelos fueron calibrados a los movimientos registrados en el terremoto Chino Hills.

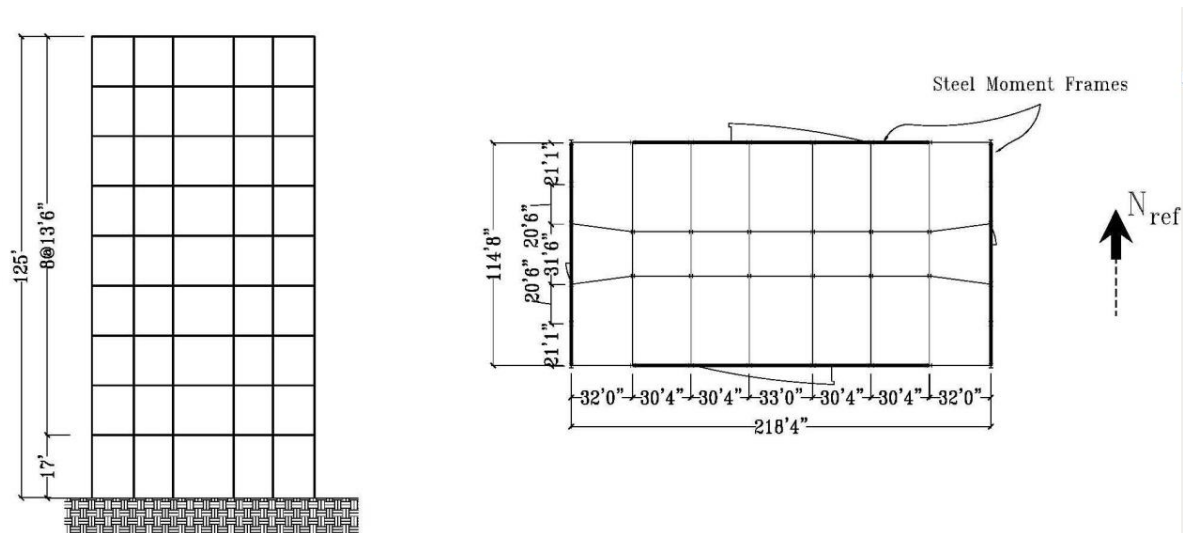


Figura 13. Elevación del pórtico tipo de momento de acero (Dirección N-S) y planta del edificio.

Nota: Tomado de Swensen y Kunnath 2012.

Entonces, Swensen y Kunnath (2012) tabularon los valores del primer y segundo períodos modales de los modelos en Sap2000, Perform-3D y OpenSees que se muestran en la Tabla 6, valores que son prácticamente iguales para los tres programas. Además, estimaron las formas modales para el primero y segundo modo de vibración lateral y llegaron a observar que dichas formas eran iguales para los tres programas de computadora en estudio.

Tabla 6.

Comparación de los períodos de los diferentes programas.

	PRM Solamente		PRM+Pórtico de Gravedad		PRM+Pórtico de Gravedad + Columnas falsa	
	T ₁	T ₂	T ₁	T ₂	T ₁	T ₂
Sap2000	2.08	0.72	2.00	0.69	1.58	0.50
Perform-3D	2.08	0.72	2.00	0.69	1.58	0.50
OpenSees	2.08	0.72	1.97	0.69	1.56	0.50

Todos los valores de período están en segundos

Nota: Adaptado de Swensen y Kunnath 2012

Por último, elaboraron la comparación directa de las aceleraciones calculadas y de los desplazamientos relativos del histórico de tiempos del techo y segundo piso para cada modelo para el terremoto de Chino Hills. Llegando a la conclusión de que los desplazamientos relativos de respuesta de los tres programas de análisis, coinciden casi exactamente, lo que aumenta la confiabilidad de OpenSees frente a éstos programas comerciales.

5.6.3 OpenSees y Ansys. Ansys desarrolla y comercializa software de análisis de elementos finitos usado para simular problemas de ingeniería (Ansys 1999).

En un estudio realizado por Santosh (2015), una cercha sencilla se modela y se le realiza un análisis estático en ambos programas, OpenSees y Ansys, De manera similar, los resultados del análisis de una estructura con vigas con varios elementos cuadrados fueron comparados. Además, un pórtico sencillo de 2D y otro 3D fueron evaluados con análisis estático *pushover* y con análisis dinámico (Santosh 2015).

Los resultados del análisis de las cerchas son iguales tanto para OpenSees y Ansys. Los resultados de análisis de la estructura de vigas se presentaron algunas variaciones. Y en los análisis

estáticos *pushover* y en análisis dinámicos los resultados fueron como se esperaba de acuerdo a Santosh (2015).

5.6.4 OpenSees y Cype Según Barceló J. & Fernández M. (2017) Cype ha creado un programa para generar, editar y mejora análisis estructurales denominado *StruBIM Analysis* que forma parte de la *suite* de *StruBIM*, que es un grupo de programas estructurales, cada uno relacionado con una etapa del diseño estructural (Barceló J. & Fernández M. 2017).

Los autores mencionados, presenta una versión secuencial de OpenSees como motor de cálculo del análisis *StruBIM* que ofrece la misma funcionalidad del motor que tenía anteriormente. Para este propósito varias extensiones se han implementado en OpenSees: dos nuevos elementos finitos (un elemento unidimensional y un shell), un nuevo controlador de restricciones, MUMPS (*Multifrontal Massively Parallel Sparse Direct Solver*), entre otras. Además, se han usado múltiples instancias de OpenSees secuencial para incrementar el paralelismo usando MPI. OpenSees le provee a *StruBIM* la posibilidad de mejorar el análisis dinámico y no lineal de estructuras complejas las cuales pueden ser muy difíciles de modelar sin las facilidades que *StruBIM* ofrece (Barceló J. & Fernández M. 2017).

5.7 Estadísticas de OpenSees

OpenSees es el software de código abierto y de elementos finitos más usado en ingeniería sísmica. Además de que el software es gratuito, se puede encontrar asesoramiento y entrenamiento sobre su uso. Los modelos en OpenSees pueden representar suelos, edificios, puentes de concreto reforzado o acero, entre otros. En OpenSees hay más de 150 tipos de materiales, 100 tipos de elementos, 30 esquemas de integración, 15 algoritmos de solución y 20 *solvers* numéricos. De los más de 30 esquemas de integración, 13 son para simulación híbrida donde se integra con análisis experimentales en laboratorio. OpenSees permite a los desarrolladores compartir directamente sus nuevos materiales, elementos, algoritmos de solución, integradores, *solvers* y desarrollos de salida de resultados usando archivos dll's (*dynamic-link library*) con otros usuarios sin tener que proporcionar el código al sitio web OpenSees (Mahin y McKenna 2013). En la actualidad hay más de 1000 clases en la plataforma con más de 100 desarrolladores en la comunidad de ingeniería (McKenna 2011).

OpenSees se ha utilizado como la *principal* herramienta para simulación en tesis de postgrado en los Estados Unidos y en las principales universidades alrededor del mundo y sus aplicaciones de OpenSees son multiplataforma (Window, Linux, OS X) (Minjie, McKenna & Scott 2017).

El sitio web de OpenSees (<http://opensees.berkeley.edu>) permite a los usuarios enviar preguntas y respuestas sobre el uso de OpenSees. Permite descargar el código fuente, verlo y descargar aplicaciones. El *OpenSeesWiki*, (<http://opensees.berkeley.edu/wiki/index.php>) sirve como un colaborador en línea para publicar manuales, ejemplos, artículos, discusiones y teoría. Además, está organizado para permitir a cualquier usuario editar cualquier página o crear nuevas

páginas dentro del sitio wiki y busca involucrar a la comunidad en un proceso continuo de creación y colaboración para cambiar constantemente el sitio web (Mahin y McKenna 2013).

Desde que OpenSees fue publicado en el año 2002, más de 20000 usuarios se han registrado en su página web oficial y más de 15000 artículos se han enviado a su foro de usuarios. OpenSees es probablemente la plataforma de análisis estructural y de código abierto más usada en el campo de la ingeniería sísmica. (Yang, Yang & Hsieh 2014).

Para el año 2012 la página web de OpenSees tuvo más de 75000 visitas de más de 175 países alrededor del mundo y las páginas fueron vistas más de 1.300.000 veces (Mahin y McKenna 2013). En la figura 14 se muestra las visitas de la mayoría de países del mundo en 2012.

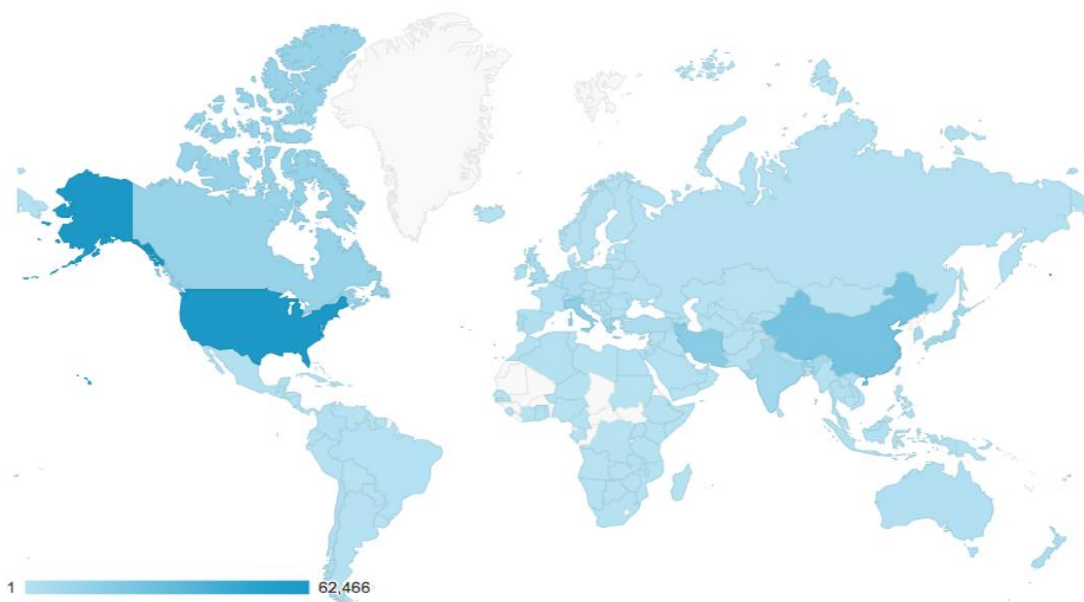


Figura 14. Mapa de visitas alrededor del mundo 2012.

Nota: Tomado de Mahin y McKenna 2013

A continuación se presentan algunos estadísticas de OpenSees, para el año 2016 tomada de Mosalan, Kasalanati & Kang G 2017.

- A. Descargas completas 11,736
- B. 112,104 visitantes en Google
- C. 1,809,743 vistas en la página web
- D. Visitas de la mayoría de países del mundo y de los estados de US en la Figura 15.

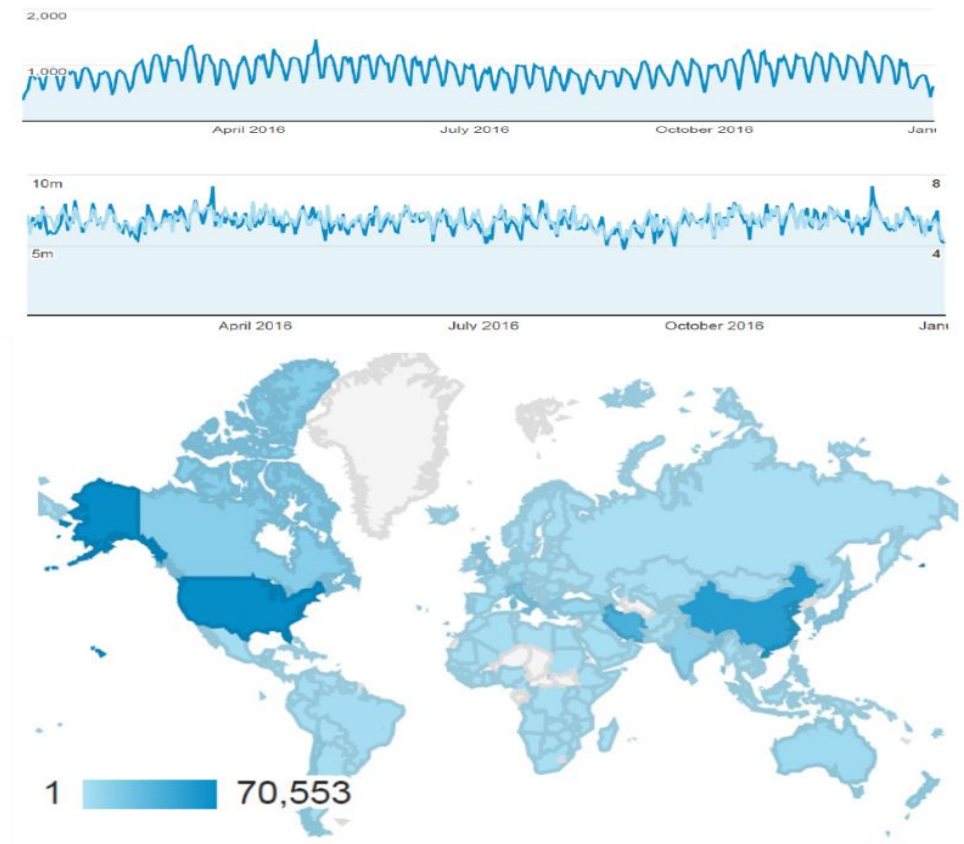


Figura 15. Mapa de visitas alrededor del mundo 2016.

Nota: Tomado de Mosalan, Kasalanati & Kang 2017.

Si observamos las dos Figuras anteriores vemos como OpenSees se ha ido fortaleciendo a través de los años gracias a su constante desarrollo y la aceptación de miles de usuarios alrededor del

mundo., sin embargo, Colombia está entre los países que menos consulta o está interesado en OpenSees.

Según lo consultado en la página web <https://www.scopus.com/> en los últimos 15 años se publicaron 703 artículos científicos con la palabra OpenSees en el título/resumen/palabra clave en comparación con 548 artículos que se publicaron con la palabra clave SAP2000, lo que demuestra que OpenSees, en su calidad de software de código abierto, es propicio para el desarrollo de nuevos estudios e investigaciones.

6. Ejemplos de aplicación

Con la intención de ilustrar cómo funcionan algunas tareas comunes de OpenSees en los problemas de análisis estructural, en este capítulo se presenta dos ejemplos de análisis de un modelo estructural, el primero se refiere a un análisis estático lineal y el segundo a un análisis estático no lineal.

6.1 Ejemplo de análisis estático lineal

Este caso de análisis es tomado de FAUDI (2014) que se trata de una estructura de tres pisos conformada por pórticos de hormigón H20 ($f'c = 20$ Mpa) dispuesto en direcciones X y Y y losas de hormigón armado como se muestra en la Figura 16.

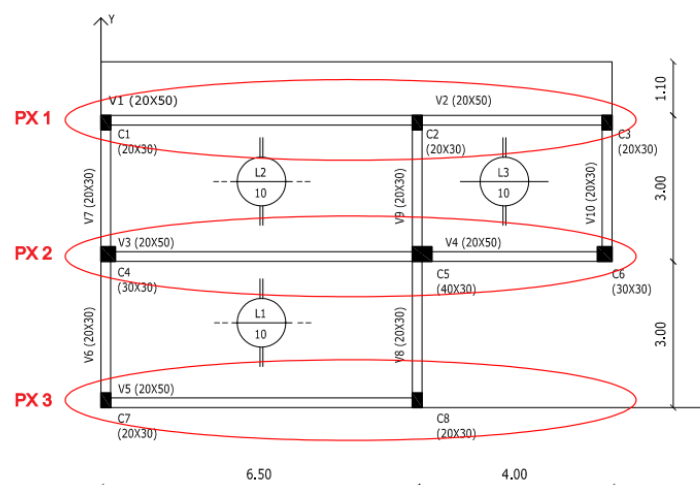


Figura 16. Planta de la estructura – planos en X.

Nota: Tomado de FAUDI (2014)

De acuerdo con FAUDI (2014) las fuerzas sísmicas para edificio de tres niveles es la que se muestra en la Figura 17.

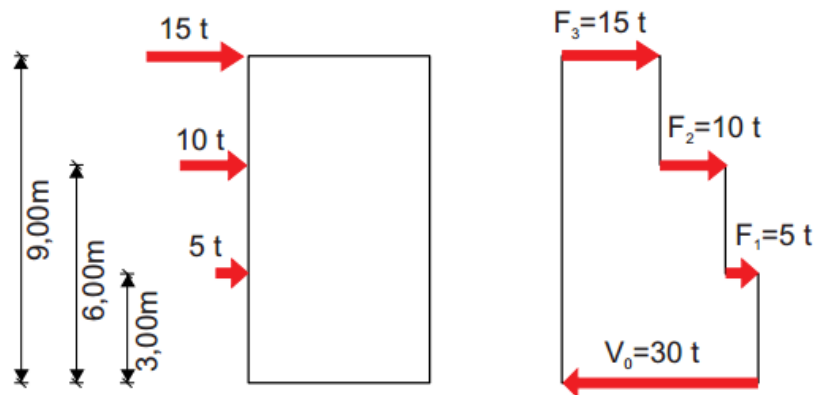


Figura 17. Fuerzas sísmicas para el edificio de 3 niveles y diagrama de corte.

Nota: Tomado de FAUDI (2014)

En la Figura 18 se muestra un esquema de la distribución de las fuerzas laterales sobre los pórticos en dirección X.

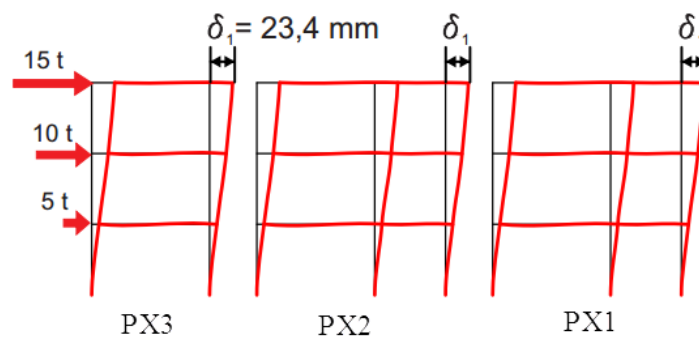


Figura 18. Desplazamientos en los pórticos en dirección X.

Nota: Tomado de FAUDI (2014)

Para el ejemplo tomaremos el pórtico PX3 el cual, y de acuerdo a lo calculado en FAUDI (2014), toma el 11.4% de las fuerzas laterales, entonces el modelo del pórtico queda tal como se muestra en la Figura 19.

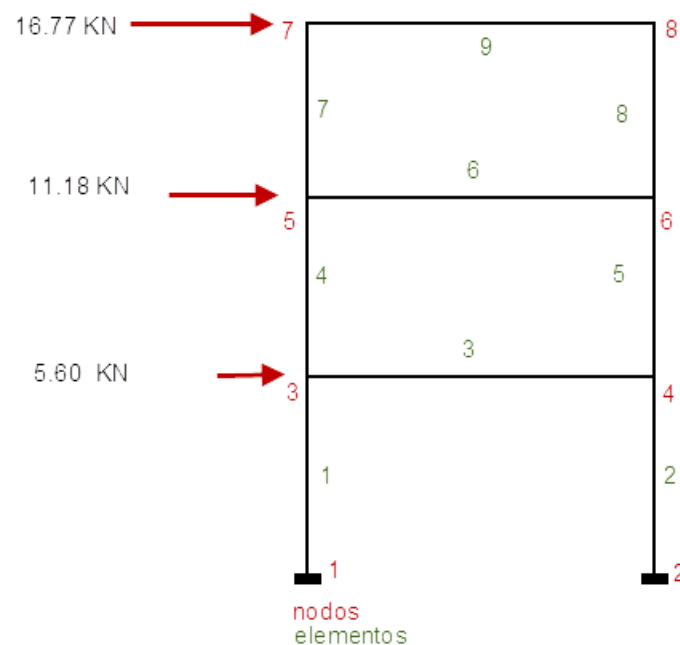


Figura 19. Modelo de pórtico.

Nota: Elaborado por el autor.

A continuación y de acuerdo con Mazonni et al. (2007) se muestra una descripción de los módulos que conforman el archivo que contiene los *scripts* en un archivo de extensión “tcl” donde se construye el modelo con su geometría, cargas, tipos de análisis, configuración de datos de salida, etc. En este caso el archivo se denomina `frameFaudiLineal.tcl`.

Se inicia con el comando *wipe* que se encarga de destruir todos los objetos construidos anteriormente. El comando *model* define la dimensión espacial de modelo (*-ndm*) que en este

caso en 2 porque es un modelo en 2 dimensiones y el número de grados de libertad (*-ndf*) que en este caso es 3.

```
wipe
```

```
# crear el constructor del modelo
```

```
model Basic -ndm 2 -ndf 3
```

Para crear los nodos se usa el comando *node* y en seguida el número del nodo y su correspondientes coordenadas x y y.

```
# se crean los nodos
```

```
node 1 0.0 0.0
```

```
node 2 6.5 0.0
```

```
node 3 0 3.0
```

```
node 4 6.5 3.0
```

```
node 5 0.0 6.0
```

```
node 6 6.5 6.0
```

```
node 7 0.0 9.0
```

```
node 8 6.5 9.0
```

Para definir las condiciones de borde se utiliza en comando *fix* y en seguida el número del nodo y se definen los grados de libertad del nodo, usando 1 si es fijo y 0 si está fijo. En este caso los nodos de base están totalmente fijos.

```
# condiciones de borde
```



```
fix 1 1 1 1
```

```
fix 2 1 1 1
```

Cuando se usan vigas es necesario crear objetos con el comando *geoTransf* para cada orientación de los elementos, seguido del tipo de transformación geométrica que este caso es Linear, creando un objeto designado con el número 1 para los elementos verticales y otro para los elementos horizontales designado con el número 2.

```
# crear los objetos de transformación geométrica
```

```
geomTransf Linear 1
```

```
geomTransf Linear 2
```

Con el comando *set* se establece el valor de las diferentes variables, en este caso el módulo de elasticidad E del concreto, el área de las columnas A_c , el área de las viga A_v , y el momento de inercia de las columna I_c y momento de inercia de la vigas I_v , calculados de la siguiente manera:

$$E = 5500\sqrt{20} = 24.6 \text{ GPa.}$$

$$A_c = 0.20 \times 0.30 = 0.06 \text{ m}^2.$$

$$A_v = 0.20 \times 0.50 = 0.10 \text{ m}^2$$

$$I_c = (0.30 \times 0.20^3)/12 = 0.0002 \text{ m}^4.$$

$$I_v = (0.20 \times 0.50^3)/12 = 0.0021 \text{ m}^4.$$

Para crear los elementos se utiliza el comando *element* seguido del tipo de elemento que este caso es *elasticBeamColumn*, el número del elemento, el nodo inicial, el nodo final, el área de la

sección transversal, el módulo de elasticidad, el momento de inercia y el objeto de transformación geométrica correspondiente.

```
# crear elementos
```

```
set E 24.6e6
```

```
set Ac 0.06
```

```
set Ic 0.0002
```

```
set Av 0.1
```

```
set Iv 0.0021
```

```
element elasticBeamColumn 1 1 3 $Ac $E $Ic 1
```

```
element elasticBeamColumn 2 2 4 $Ac $E $Ic 1
```

```
element elasticBeamColumn 3 3 4 $Av $E $Iv 2
```

```
element elasticBeamColumn 4 3 5 $Ac $E $Ic 1
```

```
element elasticBeamColumn 5 4 6 $Ac $E $Ic 1
```

```
element elasticBeamColumn 6 5 6 $Av $E $Iv 2
```

```
element elasticBeamColumn 7 5 7 $Ac $E $Ic 1
```

```
element elasticBeamColumn 8 6 8 $Ac $E $Ic 1
```

```
element elasticBeamColumn 9 7 8 $Av $E $Iv 2
```

Se establece el valor de la carga P1, P2 y P3. Se define el objeto *timeSeries* que representa la relación del tiempo del dominio con el factor de carga aplicado, que en este caso es constante y se le asigna el número 1. El comando *pattern Plain* crea un objeto de patrón de carga seguido con

su número correspondiente, en este caso 1 porque es el primer patrón creado y seguido del número del objeto *timeSeries* correspondientes que también es el número 1. Dentro de { } se define cada carga con el comando *load* seguido del número del nodos donde esta aplicada la carga, en este caso los nodos 3, 5 y 7, y su valores correspondiente en direcciones x, y y z.

```
# timeSeries y patrones de carga para las fuerzas horizontales
```

```
set P1 5.60
```

```
set P2 11.18
```

```
set P3 16.77
```

```
timeSeries Constant 1
```

```
pattern Plain 1 1 {
```

```
    load 3 $P1 0.0 0.0
```

```
    load 5 $P2 0.0 0.0
```

```
    load 7 $P3 0.0 0.0
```

```
}
```

Se configuran los objetos *analysis* en este caso un análisis estático, usando el algoritmo lineal con el integrador *LoadControl* y un factor de carga de 1.0.

```
# crear análisis
```

```
constraints Plain
```

```
numberer RCM
```

algorithm Linear

system BandSPD

integrator LoadControl 1.0

analysis Static

Usando el comando *recorder Node* para que grabe en el archivo dishor.out el desplazamiento horizontal (*-dof 1*) de los nodos 4,6 y 8 y el comando *recorder Element* para que grabe en el archivo piso3.out las fuerzas globales (cortante, axial, momento flector en coordenadas globales) de los elementos del último piso, es decir, en el rango de los elementos del 7 al 9. Es de anotar que también existen el comando *localForces* que muestra las fuerzas mencionadas en las coordenadas locales a cada elemento. Los datos de salida son los que el usuario requiera.

```
# graba los desplazamiento de los nodos y fuerzas globales
```

```
recorder Node -file dishor.out -time -node 4 6 8 -dof 1 disp
```

```
recorder Element -file piso3.out -time -eleRange 7 9 globalForces
```

Con el comando *analyze* y se aplica en análisis de un solo paso y se usa *wipe* nuevamente para destruir todos los objetos.

```
# realiza un solo paso para aplicar las fuerzas
```

```
analyze 1
```

```
wipe
```

A continuación se presenta el *script* completo del modelo creado, el cual se graba en el archivo `frameFaudiLineal.tcl` para ser ejecutado en `OpenSees.exe` con el comando *source*, como se muestra en la Figura 20.

Wipe

```
# create the model builder
```

```
model Basic -ndm 2 -ndf 3
```

```
# create nodes
```

```
node 1 0.0 0.0
```

```
node 2 6.5 0.0
```

```
node 3 0 3.0
```

```
node 4 6.5 3.0
```

```
node 5 0.0 6.0
```

```
node 6 6.5 6.0
```

```
node 7 0.0 9.0
```

```
node 8 6.5 9.0
```

```
# boundary conditions
```

```
fix 1 1 1 1
```

```
fix 2 1 1 1
```

```
# create geometric transformation objects
```

```
geomTransf Linear 1
```

```
geomTransf Linear 2
```

```
# create elements
```

```
set E 24.6e6
```

```
set Ac 0.06
```

```
set Ic 0.0002
```

```
set Av 0.1
```

```
set Iv 0.0021
```

```
element elasticBeamColumn 1 1 3 $Ac $E $Ic 1
```

```
element elasticBeamColumn 2 2 4 $Ac $E $Ic 1
```

```
element elasticBeamColumn 3 3 4 $Av $E $Iv 2
```

```
element elasticBeamColumn 4 3 5 $Ac $E $Ic 1
```

```
element elasticBeamColumn 5 4 6 $Ac $E $Ic 1
```

```
element elasticBeamColumn 6 5 6 $Av $E $Iv 2
```

```
element elasticBeamColumn 7 5 7 $Ac $E $Ic 1
```

```
element elasticBeamColumn 8 6 8 $Ac $E $Ic 1
```

```
element elasticBeamColumn 9 7 8 $Av $E $Iv 2
```

```
# timeSeries and loadPattern for horizontal forces
```

```
set P1 5.60
```

```
set P2 11.18
```

```
set P3 16.77
```

```
timeSeries Constant 1
```

```
pattern Plain 1 1 {
```

```
    load 3 $P1 0.0 0.0
```

```
    load 5 $P2 0.0 0.0
```

```
    load 7 $P3 0.0 0.0
```

```
}
```

```
# create analysis
```

```
constraints Plain
```

```
numberer RCM
```

```
algorithm Linear
```

```
system BandSPD
```

```
integrator LoadControl 1.0
```

```
analysis Static
```

```
# record the horizontal disp at node 8
```

```
recorder Node -file dishor.out -time -node 4 6 8 -dof 1 disp
```

```
recorder Element -file piso3.out -time -eleRange 7 9 globalForces
```

```
# perform one step to apply vertical loads
```

```
analyze 1
```

```
wipe
```

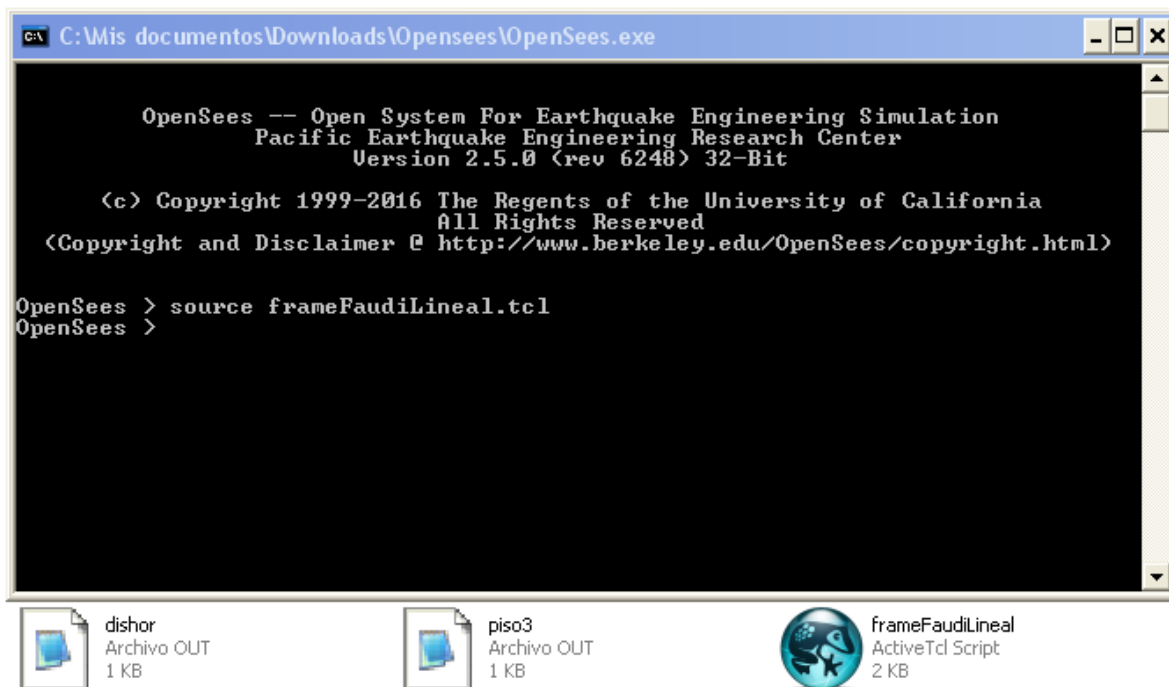


Figura 20. Ejecución del archivo script .tcl en OpenSees.exe, versión 2.5.0.

Los desplazamientos de salida son enviados al archivo dishor.out tal como se muestra en la Figura 21, donde aparece el resultado del desplazamiento horizontal del nodo 8 con un valor de 23.35 mm, valor que es similar a lo mostrado en la Figura 18 tomada de FAUDI (2014) donde previamente se había calculado este valor como 23.4 mm., lo que le da un buen grado de confiabilidad al modelo realizado en OpenSees.

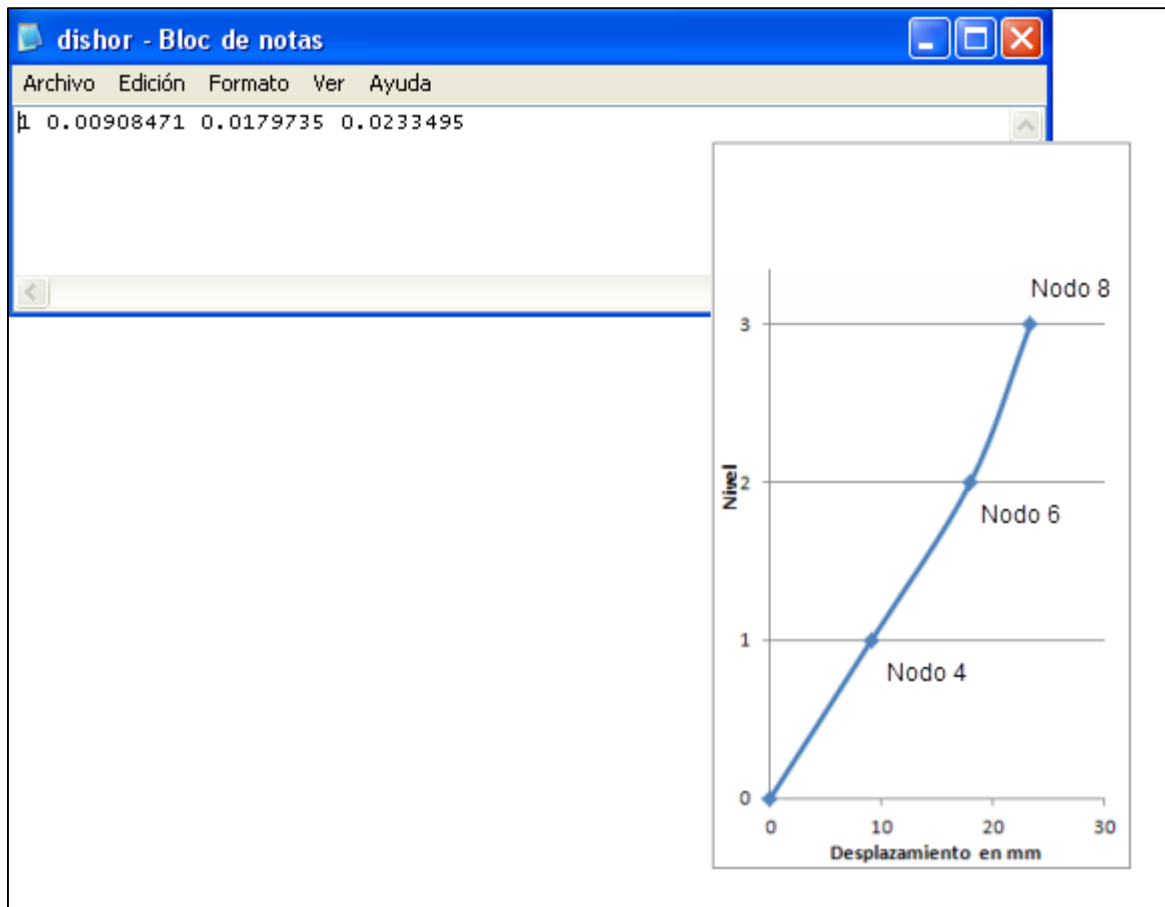


Figura 21. Desplazamientos en los nodos 4, 6 y 8.

Nota: Elaborado por el autor.

Los resultados de las fuerzas globales se envían al archivo piso3.out tal como se muestra en la Figura 22 . La disposición de los valores parecen complicados, pero tienen una distribución lógica. La primera columnas es el número de pasos que en esta caso es uno. Entonces para cada elemento hay 6 datos de información (N, V, M de cada nodo), así que hay 19 columnas en el archivo.

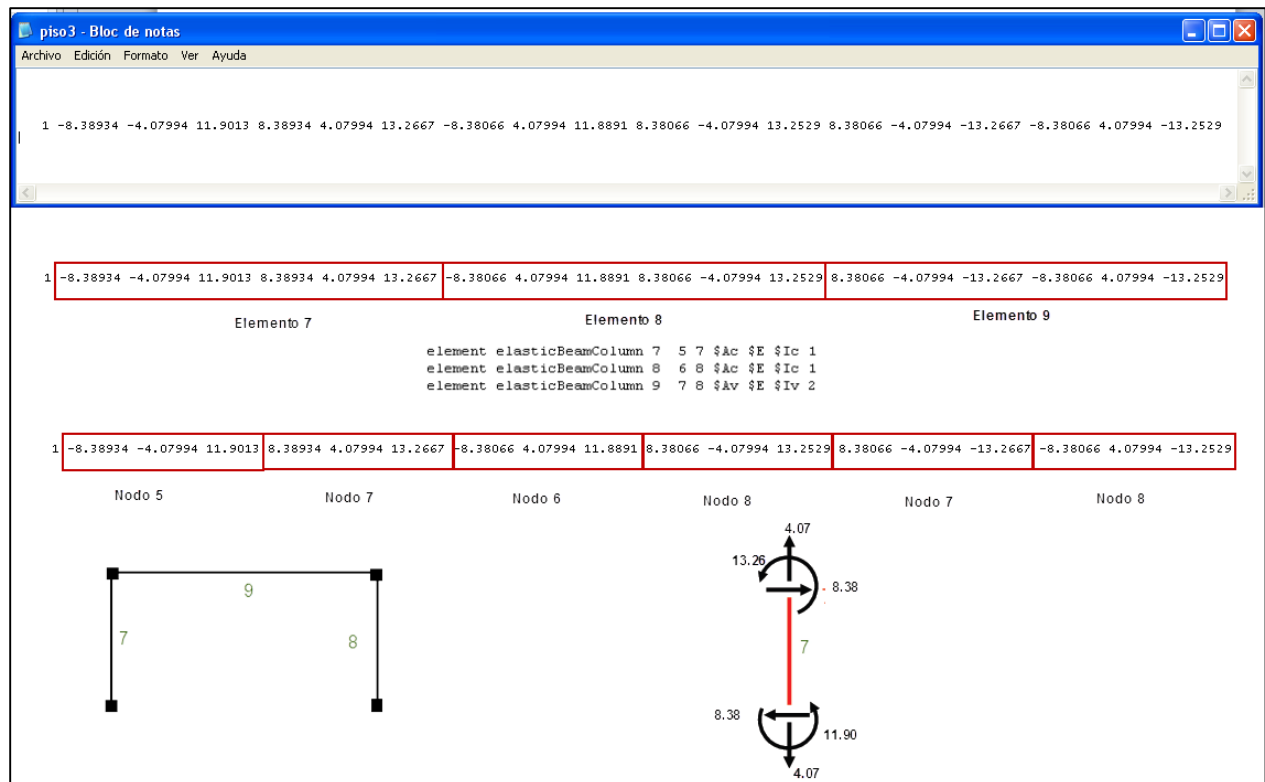


Figura 22. Archivo piso3.out con resultados de fuerzas globales de los elementos del último piso.
 Nota: Elaborado por el autor.

6.2 Ejemplo de análisis estático no lineal

Se configuran los objetos *analysis* que se quieran usar, este caso es un análisis estático, con el algoritmo Newton para análisis no lineal y entonces se aplica el análisis de un solo paso con el integrador *LoadControl* y un factor de carga de 1.0.

crear análisis

constraints Plain

numberer RCM

test NormDispIncr 1.0e-10 6 0

```
algorithm Newton  
system ProfileSPD  
integrator LoadControl 1.0  
analysis Static
```

Las cargas laterales se aplican en los nodos 3, 4, 5, 6, 7 y 8 usando el objeto *timeSeries Linear* lo que indica que el factor de carga de ese patrón varía linealmente con el tiempo, y en esta caso el patrón de carga también es *Plain*. Se crean *timeSeries* y el *pattern Plain* para las cargas laterales y se les asigna a cada uno de ellos el número 1.

```
# timeSeries y patrones de carga para las fuerzas laterales  
  
timeSeries Linear 1  
  
pattern Plain 1 1 {  
  load 3 1.0 0.0 0.0  
  load 4 1.0 0.0 0.0  
  load 5 1.0 0.0 0.0  
  load 6 1.0 0.0 0.0  
  load 7 1.0 0.0 0.0  
  load 8 1.0 0.0 0.0  
}
```

Usando el comando *recorder Node* para que grabe en el archivo *disp8nl.out* el desplazamiento horizontal (*-dof 1*) del nodo 8 y el comando *recorder Element* para que grabe en el archivo

piso3nl.out las fuerzas globales (cortante, axial, momento flector en coordenadas globales) de los elementos del 7 al 9. Los datos de salida son los que el usuario requiera.

```
# graba los resultados de los desplazamiento de las fuerzas globales  
recorder Node -file dis8nl.out -time -node 8 -dof 1 disp  
recorder Element -file piso3nl.out -time -eleRange 7 9 globalForces
```

El análisis de cargas laterales se realiza utilizando el integrador *DisplacementControl* de tal manera que en cada uno de los 90 pasos considerados, el incremento del desplazamiento horizontal (-dof 1) sea un 1 mm (0.001m). En el paso 90 el desplazamiento final del nodo 8 será de 9 cm. (0.09 m).

```
integrator DisplacementControl 3 1 0.001  
analyze 90
```

```
wipe
```

Se usa *wipe* nuevamente para destruir todos los objetos.

A continuación se presenta el script completo del modelo creado, el cual se graba en el archivo frameElastic.tcl para ser ejecutado en OpenSees.exe.

```
wipe  
# create the model builder
```

```
model Basic -ndm 2 -ndf 3
```

```
# create nodes
```

```
node 1 0.0 0.0
```

```
node 2 6.5 0.0
```

```
node 3 0 3.0
```

```
node 4 6.5 3.0
```

```
node 5 0.0 6.0
```

```
node 6 6.5 6.0
```

```
node 7 0.0 9.0
```

```
node 8 6.5 9.0
```

```
# boundary conditions
```

```
fix 1 1 1 1
```

```
fix 2 1 1 1
```

```
# create geometric transformation objects
```

```
geomTransf Linear 1
```

```
geomTransf Linear 2
```

```
# create elements
```

```
set E 24.6e6
```

```
set Ac 0.06
```

```
set Ic 0.0002
```

```
set Av 0.1
```

```
set Iv 0.0021
```

```
element elasticBeamColumn 1 1 3 $Ac $E $Ic 1
```

```
element elasticBeamColumn 2 2 4 $Ac $E $Ic 1
```

```
element elasticBeamColumn 3 3 4 $Av $E $Iv 2
```

```
element elasticBeamColumn 4 3 5 $Ac $E $Ic 1
```

```
element elasticBeamColumn 5 4 6 $Ac $E $Ic 1
```

```
element elasticBeamColumn 6 5 6 $Av $E $Iv 2
```

```
element elasticBeamColumn 7 5 7 $Ac $E $Ic 1
```

```
element elasticBeamColumn 8 6 8 $Ac $E $Ic 1
```

```
element elasticBeamColumn 9 7 8 $Av $E $Iv 2
```

```
# create analysis
```

```
constraints Plain
```

```
numberer RCM
```

```
test NormDispIncr 1.0e-10 6 0
```

```
algorithm Newton
```

```
system ProfileSPD
```

```
integrator LoadControl 1.0
```

```
analysis Static
```

```
# timeSeries and loadPattern for lateral loads

timeSeries Linear 1

pattern Plain 1 1 {

  load 3 1.0 0.0 0.0

  load 4 1.0 0.0 0.0

  load 5 1.0 0.0 0.0

  load 6 1.0 0.0 0.0

  load 7 1.0 0.0 0.0

  load 8 1.0 0.0 0.0

}

# record the horizontal disp at node 8

recorder Node -file dis8nl.out -time -node 8 -dof 1 disp

recorder Element -file piso3nl.out -time -eleRange 7 9 globalForces

integrator DisplacementControl 8 1 0.001

analyze 90

wipe
```

Los desplazamientos de salida son enviados al archivo `dis8nl.out` tal como se muestra en la Figura 23. Cuando se usa el integrador *DisplacementControl* en un análisis *Static*, el “time” (columna izquierda en la Figura 23) es la magnitud de la fuerza en el DOF específico.

Los resultados de las fuerzas globales se envían al archivo piso3nl.out tal como se muestra en la Figura 24. La disposición de los valores parece complicada, pero tiene una distribución lógica. La primera columna es el pseudotiempo. Entonces para cada elementos hay 6 columnas de información (N, V, M de cada nodo), así que hay 19 columnas en el archivo.

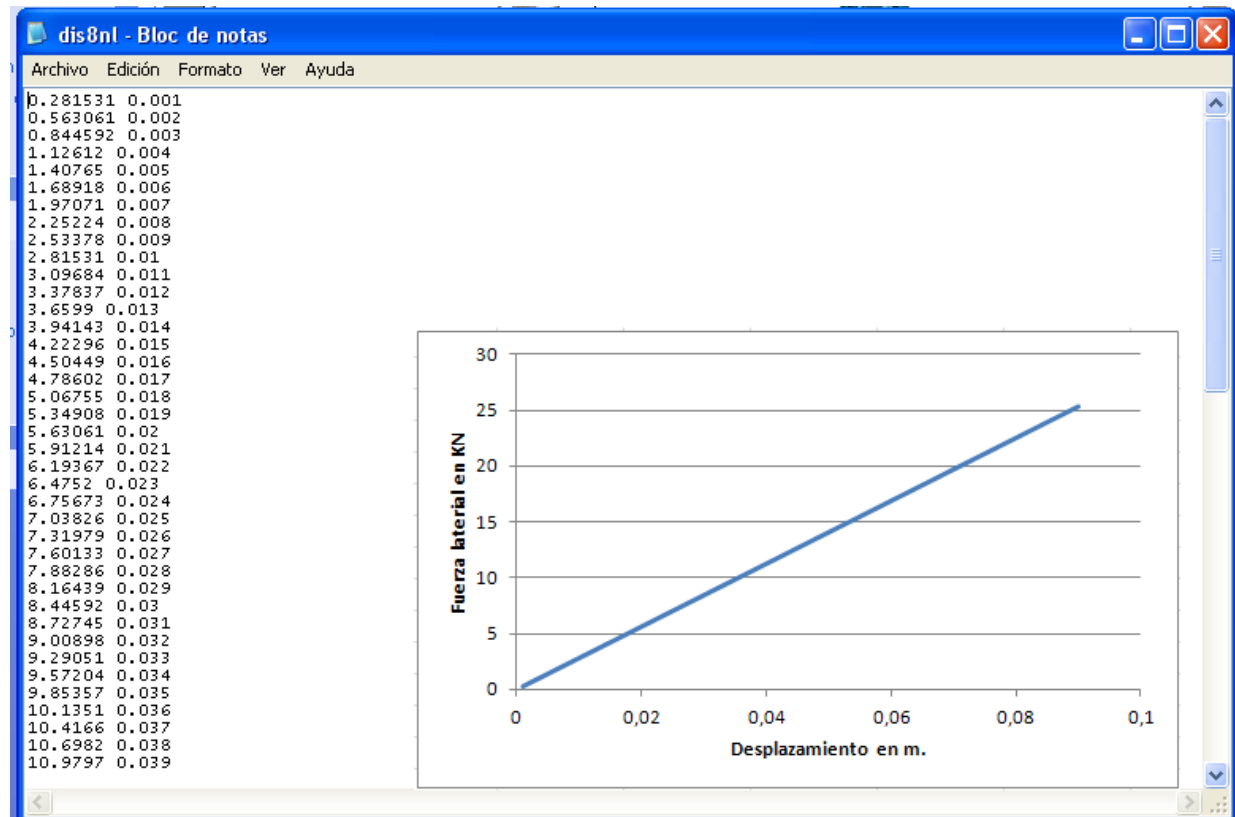


Figura 23. Archivo disp8nl.out correspondiente contiene los desplazamiento del nodo 8.

Nota: Elaborado por el autor.

Al examinar la última fila del archivo piso8nl.out, los valores corresponden a las fuerzas globales de cuando el desplazamiento horizontal del nodo 8 es de 9 cm mostrado en la Figura 25.

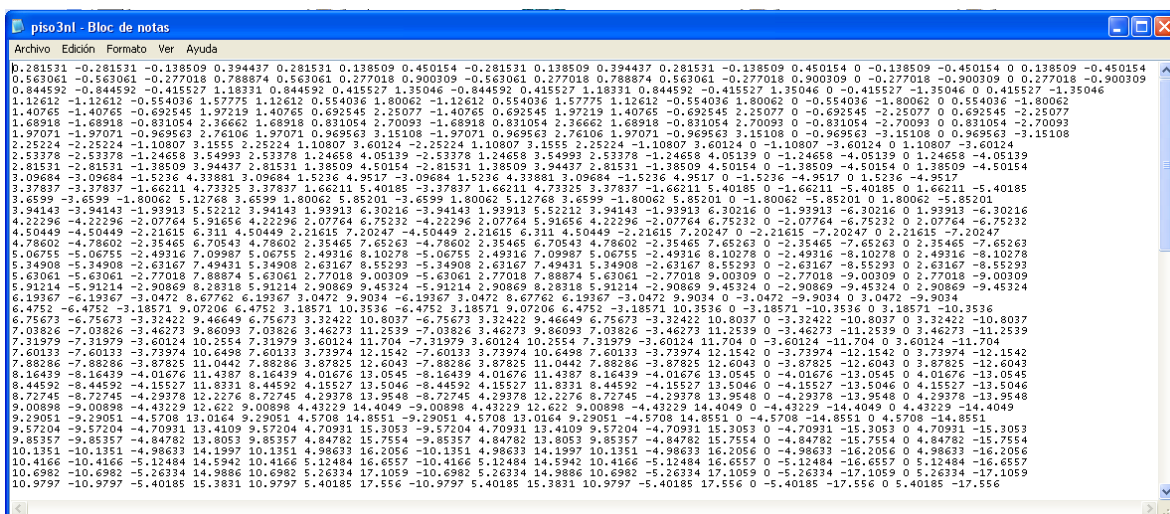


Figura 24. Archivo piso3nl.out contiene las fuerzas globales de los elementos de último piso.

Nota: Elaborado por el autor

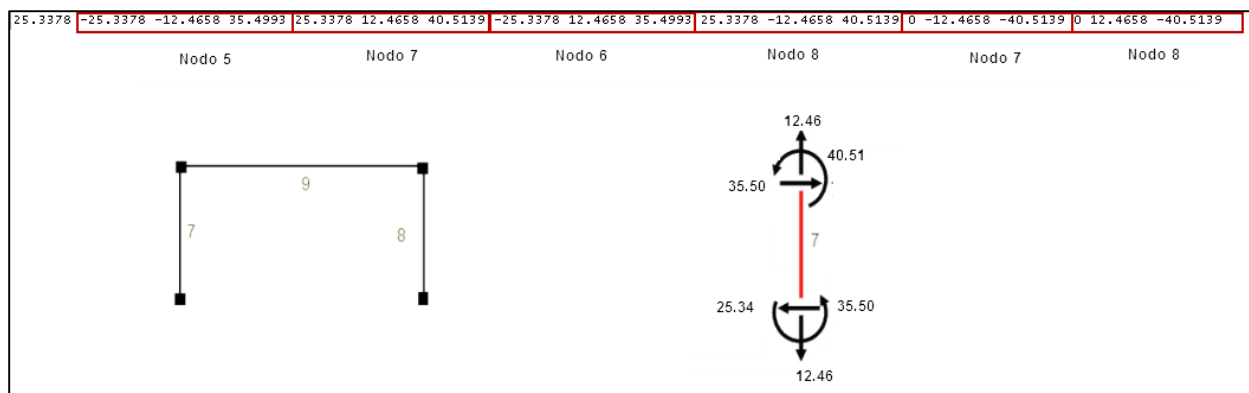


Figura 25. Resultados de las fuerzas globales para un desplazamiento de 9 cm.

Nota: Elaborado por el autor.

7. Conclusiones Finales

- OpenSees es una plataforma computacional que la comunidad de investigación de ingeniería sísmica emplea para el desarrollo aplicaciones de elementos finitos.
- En el capítulo 5.3 se estudia la arquitectura básica de donde se concluye que OpenSees implementa un escenario orientado a objetos con el uso del Método de Elementos Finitos para modelar, analizar y mostrar resultados de un problema estructural definido.
- Desde su creación OpenSees extiende el intérprete Tcl con comandos para construir y analizar modelos, sin embargo, con el fin de proporcionarle al usuario nuevas opciones de lenguajes de *scripts* la interfaz de interprete OpenSees fue reacondicionada para proveerle capacidades de multi-interprete, pudiéndose usar otro intérpretes como Phyton, el cual ya fue probado satisfactoriamente.
- Básicamente, la plataforma de OpenSees posee dos aplicaciones secuenciales que son OpenSees.exe que usa comandos del lenguaje Tcl y OpenSeesPy que fue creada recientemente y que usa comando del lenguaje Phyton.
- OpenSees ha construido OpenSeesMS y OpenSeesPS que son dos intérpretes para computadores en paralelo los cuales analizan modelos muy grandes o ejecutan *scripts* más complicados y de manera más rápida y eficiente. Estos intérpretes pueden descargarse de la plataforma de OpenSees o pueden ejecutarse directamente en la página web *Discovery WorkSpace* de la *National Natural Hazards Engineering Research Infrastructure* (NHERI) (<https://www.designsafe-ci.org/rw/workspace/#/>)

- La comunidad que desarrolla OpenSees ha ido creando aplicaciones con interfaz gráficas de usuario (GUIs) y entre éstas están *BuildingTcl/BuildingTclViewer*, *OpenSees Navitator* y otras dos creadas recientemente que son *GiD+OpenSees* y *Build-X*.
- Las comparaciones realizadas entre el uso de OpenSees y el uso de otros software comerciales reconocidos como *MSC Marc*, *Sap2000*, *Perform-3D* y *Ansys* muestran que son, en términos generales, coincidentes, lo que permite afirmar que este software de código abierto es confiable..
- Los algoritmos de cálculo de OpenSees son usados como el motor de análisis en otros programas de análisis estructural tales como *nvStructural* de *Novel CAE Solutions*, *CDS Win de Software Tecnico Scientifico* y *StruBIM* de Cype.
- OpenSees posee un alto grado de popularidad, sus usuarios han ido aumentando en número desde su creación hasta la fecha y es probablemente uno de los plataformas análisis estructural de código abierto más usadas en el campo de la ingeniería sísmica.
- OpenSees permite a las agencias de diseño, a los institutos de investigación, a las universidades públicas y privadas y en general a la comunidad de ingeniería sísmica, además de un enfoque eficiente hacia soluciones más económicas, la integración de las últimas investigaciones que entran en sus códigos fuente que se desarrollan alrededor del mundo.
- Por su enfoque en el análisis no lineal OpenSees podría ser una herramienta útil dentro de las investigaciones para la aplicación de los métodos de análisis de sistemas de resistencia sísmica reconocidos por Reglamento Colombiano de Construcción Sismo Resistente NSR-10, particularmente, el Método del análisis dinámico inelástico y el Método de análisis no lineal estático de plastificación progresiva (*push-over*) (ASI 2010).

Referencias Bibliográficas

- AIS. (2010). *Reglamento Colombiano de Construcción Sismo Resistente NSR-10*. Bogotá D.C (Colombia): Ministerio de Ambiente, Vivienda y Desarrollo Territorial.
- AlHamayden M., Najib M. & Alawnah S. (2016). INSPECT: A grafical user interface software package for IDARC-2D *SoftwareX* 5, 243–251.
- Ansys (1999). Pennsylvania Technology Directory. p. 25.
- Barceló J. & Fernández M. (2017) Two New Elements to OpenSees. *OpenSees Days Europe 2017 - 1st European Conference on OpenSees*, 6-9,
- Barceló J. & Fernández M. OpenSees Integrated in a Bim WorkFlow as Calculation Engine. *OpenSees Days Europe 2017 - 1st European Conference on OpenSees*, 69-72
- Da Fonseca Z.A. (2011). *El Método de los Elementos Finitos: una introducción*, Maracaibo, Venezuela: Grupo Editorial Biblioteca Universidad Rafael Urdaneta.
- Dutta M. (2016). Basic Concept of Object Oriented and Procedure Oriented Programming. *Internacional Journal of Information and Technology* 2, 1-4.
- FAUDI (2014) *Estructuras III. Estructuras Sismorresistentes*. Córdoba: Universidad Nacional de Córdoba.
- Galeano C., Mantilla, J., Duque, C. & Mejía, M. (2007) Herramientas de Software con Licencia Pública General para el Modelado por Elementos Finitos. *Dyna* 153, 313-324.
- Hahn E. (2014) An Overview of Open-Source Software Licenses and the Value of Open-Source Software to Public Health Initiatives *Johns Hopkins Apl Technical Digest* 2(4), 690-698.

- Hutton D.V. (2004). *Fundamentals of Finite Element Analysis*, New York, Estados Unidos: McGraw Hill.
- Ladjinovic D., Raseta A., Radujkovic A., Folic R. & Prokic A. (2012) Comparison of Structural Models for Seismic Analysis of Multi-Storey Frame Buildings, *15th World Conference on Earthquake Engineering*, Lisboa.
- Lu X., Xie L., Guan H., Huang L. & Lu X. (2015) A shear wall element for nonlinear seismic analysis of super-tall buildings using OpenSees *Finite Elements in Analysis and Design* 98, 14-25
- Mahin S. & McKenna F. (2013) *Report on OpenSees Highlights July 1, 2012 – June 30, 2013*. University of California, Berkeley.
- Mazzoni S., McKenna F, Scott M., & Fenves G., et al (2007). *OpenSees Command Language Manual*, Recuperado el 20 de agosto de 2017 a las 19:57 de <http://opensees.berkeley.edu/OpenSees/manuals/usermanual/OpenSeesCommandLanguageManualJune2006.pdf>
- McKenna F. (1997) *Object-Oriented Finite Element Programming: Frameworks for Analysis, Algorithms and Parallel Computing*. (Tesis doctoral). Universidad de California. Berkeley.
- McKenna F. (2011) OpenSees: A Framework for Earthquake Engineering Simulation. *Computing in Science & Engineering*, 58-66
- McKenna F. & Fenves G.(2008) Using the OpenSees Interpreter on Parallel Computers. *NEESit*, 1-16.
- McKenna F., Scott M., & Fenves G.(2010) Nonlinear Finite-Element Analysis Software Architecture Using Object Composition. *Journal and Computing in Civil Engineering* © ASCE, 95-107
- Ministry of Business, Innovation and Employment (MBIE), Earthquake Commission (EQC), New Zealand Society for Earthquake Engineering (NZSEE), Structural Engineering Society (SESOC) & New Zealand Geotechnical Society Inc. (NZGSI) (2017). *The Seismic Assessment of Existing Buildings*. Wellington, New Zealand: Ann Cunninghame and Sandy Cole.

- Minjie Z. (2017) *The OpenSeesPy Library*. Recuperado el 13 de febrero de 2018 a las 20:15 de <http://openseespydoc.readthedocs.io/> <http://web.engr.oregonstate.edu/~zhum/python.html>
- Minjie Z. McKenna F. & Scott M. (2017) *OpenSeesPy: Python library for the OpenSees finite element framework*. Recuperado el 28 de noviembre de 2017 a las 17:24 de Journal homepage: <https://www.sciencedirect.com/science/article/pii/S2352711017300584>
- Mittal, P.& Singh J.(2013). Use of Open Source Software in Engineering.*International Journal of Advanced Research in Computer Engineering & Technology* 2, 1114-1117.
- Mosalan K, Kasalanati A & Kang G (2017) PEER 2016 Annual Report *Pacific Earthquake Engineering Research Center University of California, Berkeley*
- MSC Software (2001). *MSC.Marc Volume A: Theory and User Information*. Recuperado el 11 de diciembre de 2017 a las 21:12 de <http://www.mate.tue.nl>
- National Science Foundation (2015). *Natural Hazards Engineering Research Infrastructure (NHERI)*. Arlington, USA : NSF.
- Noy F & Linero D (2014) Programa Didáctico de Código Abierto de Análisis Dinámico de Estructuras Undin 1,0 . *Educación en Ingeniería* 9, 118-131
- Papanikolaou V., Kartalis-Kauonis T., Protopapadakis E., & Papadopoulos T.. (2017). A new graphical user interface for OpenSees. *OpenSees Days Europe 2017 - 1st European Conference on OpenSees*, 73-76.
- Papanikolaou V., Kartalis-Kauonis T., Protopapadakis E., & Papadopoulos T.. (2017). A new graphical user interface for OpenSees. *OpenSees Days Europe 2017 - 1st European Conference on OpenSees*, 73-76.
- Psyrra N. & Sextos A. (2018) Build-X: Expert system for seismic analysis and assessment of 3D buildings using OpenSees. *Advances in Engineering Software* 116, 23-35.
- QuakeCore (2016) *OpenSees in Parallel and on HPC Resources* QuakeCore OpenSees Training Workshop 2016

- QuakeCore (2016). *Estructural Analysis Ejamples*. QuakeCore OpenSees Training Workshop 2016
- Santosh Y. (2015) . *Study on applicability of OpenSees for Structural Analysis*, Shinshu University, Japan.
- Swensen D. y Kunnath S. (2012). Calibrating Computer Models For Seismic Analysis: Case Studies Using Instrumented Building Records *SMIP2 Seminar on Utilization of Strong-Motion Data*, 57 - 68.
- US National Science Foundation NSF (2006) *OpenSees Developer*. Recuperado 20 de noviembre de 2017 a las 17:17 de <http://opensees.berkeley.edu//opensees.berkeley.edu>
- Yang Y., Yang Ch. & Hsieh T. (2014) GPU parallelization of an object-oriented nonlinear dynamic structural analysis platform. *Simulation Modelling Practice and Theory* 40, 112-121.
- Zhen X., Xinzheng L.& Kincho H.(2016) A computational framework for regional seismic simulation of buildings with multiple fidelity models *Advances in Engineering Software* 99, 100–110.