CAPÍTULO V

USO DE OPENSEES

RESUMEN

En este capítulo se presenta una breve introducción al uso de la plataforma informática OPENSEES (Open System for Earthquake Engineering Simulation), se muestra una descripción de los comandos y funciones básicas que tiene esta herramienta y se resuelven algunos ejemplos de aplicación.

Este trabajo de ninguna manera pretende sustituir el manual de usuario de OPENSEES, solo busca constituir una pequeña guía para quienes desean incursionar en el uso de esta poderosa herramienta para simulación en ingeniería sísmica.

El autor quiere expresar un sincero agradecimiento a la Unidad de Ingeniería Civil y Geología de la Universidad Técnica Particular de Loja, en especial a su director el Dr. Ing. Vinicio Suárez Chacón, por los valiosos conocimientos transmitidos sobre OpenSees y que han sido fundamentales para la elaboración de este capítulo.

5.1.- INTRODUCCIÓN

Antes de empezar, el autor considera importante hacer una breve reseña histórica sobre OPENSEES, como tributo a los profesionales que han dedicado muchos años de su vida al desarrollo de esta herramienta que hoy está disponible y al alcance de todos los investigadores y profesionales relacionados con la Ingeniería Sísmica.

El OPENSEES (Open System for Earthquake Engineering Simulation), ha sido promovido por el Pacific Earthquake Engineering Research Center a través de la Fundación Nacional de Ciencia de los Estados Unidos. Son muchos los profesionales involucrados en el desarrollo de esta herramienta, por mencionar algunos tenemos a: Dr. Frank McKenna, Gregory Fenves, de la Universidad de California, Berkeley. Es así mismo valioso el aporte que han realizado otros profesionales como la Dra. Silvia Mazoni, Dr. Filip Filippou, el Prof. Michael Scout, Prof. Boris Jeremic, Prof. Ahmed Elgamal, entre otros.

OPENSEES es una plataforma informática 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, para ello dispone de un amplio rango de modelos de materiales, elementos y algoritmos de solución. OpenSees utiliza métodos basados en elementos finitos, por lo tanto el primer paso para la modelación es subdividir el sistema en elementos y nudos, para de esta manera definir la acción de cargas, y las restricciones nodales. La característica principal de OpenSees es que dicha

modelación y simulación se la realiza a través de una fuente abierta. Esto quiere decir que OpenSees está bajo constante desarrollo, de tal suerte que los diseñadores y los usuarios pueden actualizar sus bases permanentemente.

El lenguaje de interpretación llamado lenguaje TCL, originado del acrónimo en inglés "Tool Command Language" o lenguaje de herramientas de comando, ha sido utilizado para soporte de los comandos de OpenSees, los cuales son usados para la definición de la geometría del problema, estados de carga, formulación y solución. Tcl es un lenguaje script (interpretación) creado por John Ousterhout en la Universidad de Berkeley, provee útiles herramientas de programación, permite manipulación de variables, contiene estructuras básicas de control (if, while, for, foreach), evaluación de expresiones matemáticas y manipulación de archivos. Por su fácil aprendizaje, Tcl rápidamente ganó amplia aceptación, además que resulta ser muy potente en las manos adecuadas. Es usado principalmente en el desarrollo rápido de prototipos, aplicaciones “script”, interfaces gráficas y pruebas. Entre las principales características del lenguaje Tcl podemos mencionar siguientes:

- Es un lenguaje interpretado, y su código puede ser creado y modificado dinámicamente.
- Todos los elementos de un programa son comandos, incluyendo las estructuras del lenguaje. Dichos comandos se escriben en notación polaca.
- Todos los comandos pueden ser redefinidos o sobrescritos de manera dinámica.

- Todos los datos son manejados como cadenas de caracteres Unicode, incluyendo el código fuente. En efecto, soporta totalmente Unicode desde su lanzamiento del año 1999.
- Sus reglas sintácticas son extremadamente simples.
- Posee reglas de alcance dinámico.
- Permite la programación orientada a eventos sobre "sockets" y ficheros. También son posibles los eventos basados en tiempo y los definidos por el usuario.
- Permite escribir código fácil de mantener. Los "scripts" Tcl son a menudo más compactos y legibles que los programas funcionalmente equivalentes en otros lenguajes de programación.
- Es fácilmente "extensible" vía C, C++ y Java.
- Está fuertemente integrado con los entornos gráficos, a través de su interfaz Tk.
- Es un lenguaje multiplataforma, con intérpretes que se ejecutan sobre Windows, Linux, UNIX, MacOS y OSX e incluso microprocesadores PIC.

Es importante resaltar que una de las características más importantes de Tcl es su extensibilidad. Y justamente gracias a ello es que se ha podido incluir comandos adicionales para el uso con OpenSees.

5.2.- INSTALACIÓN

Como se mencionó anteriormente el OpenSees está en constante desarrollo, gracias a la dedicación de muchos profesionales que aportan con nuevos materiales, nuevos modelos y mejoradas herramientas para el análisis, por lo tanto éste puede y debe ser actualizado permanentemente, es por eso que se

recomienda descargar al archivo correspondiente desde el Internet para mantener actualizadas las bases.

La instalación puede ser realizada desde un dispositivo extraíble, un CD, etc., o directamente a través del Internet, desde la página www.opensees.berkeley.edu. A continuación se describe el proceso de instalación para cada caso.

5.2.1 Instalación desde un dispositivo de almacenamiento extraíble

Los instaladores pueden estar almacenados en un CD, o en cualquier medio de almacenamiento de datos extraíble. Los archivos que necesitamos para la instalación deben estar guardados en una carpeta, la misma que deberá contener los siguientes archivos:

- OpenSees2.0.0.exe
- ActiveTcl

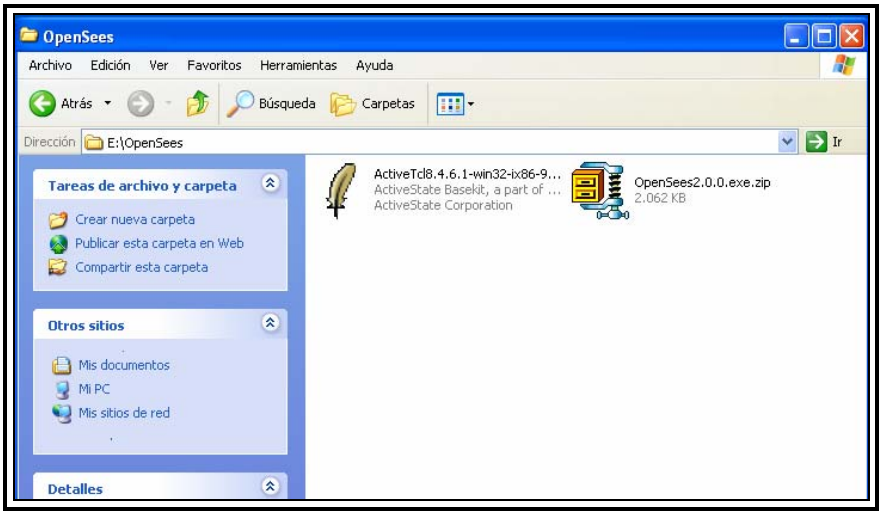

Figura 5.1: Carpeta con archivos para instalación

El primero es un archivo zip (comprimido) que contiene el ejecutable de OpenSees y el segundo es un archivo que se instala automáticamente y contiene el ejecutable del tcl/tk.

El primer paso es instalar el Tcl/Tk Version 8.4.6, que es la versión que al momento de la redacción del presente capítulo es la vigente. Este archivo es un ejecutable, y lo único que tenemos que hacer es doble clic en su icono y seguir las instrucciones.

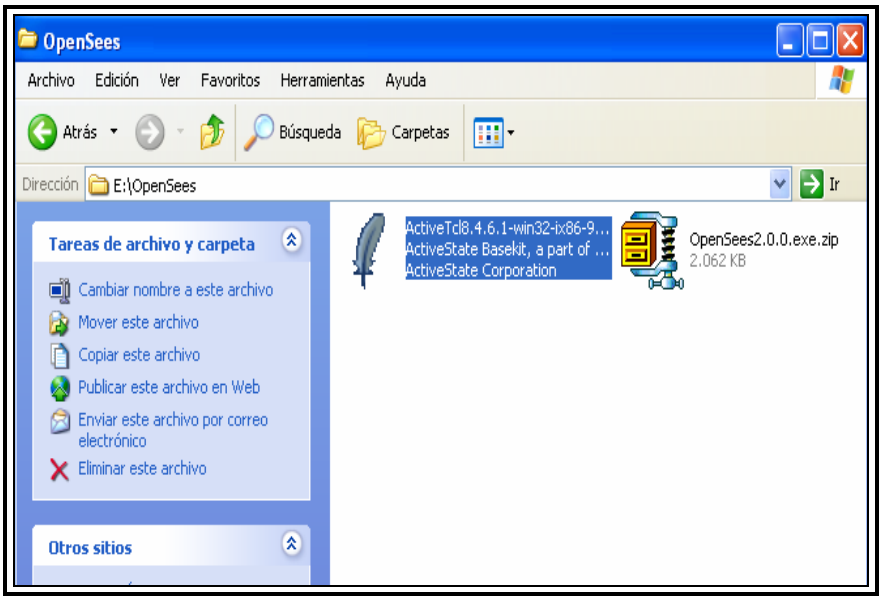

Figura 5.2: Archivo ActiveTcl 8.4.6.1

Se deben aceptar los términos de la licencia y seguir con el proceso de instalación.

Durante la instalación se le pregunta al usuario la ruta en la cual desea almacenar el archivo Activetcl, por default se instalará en C:\tcl, no obstante y como sugerencia tomada directamente desde la página oficial de OpenSees, se recomienda cambiar esta dirección a C:\Program Files\Tcl. Luego de

cambiar la dirección solamente resta continuar con el proceso de instalación, siguiendo las instrucciones que van siendo desplegadas en las ventanas de diálogo que van apareciendo.

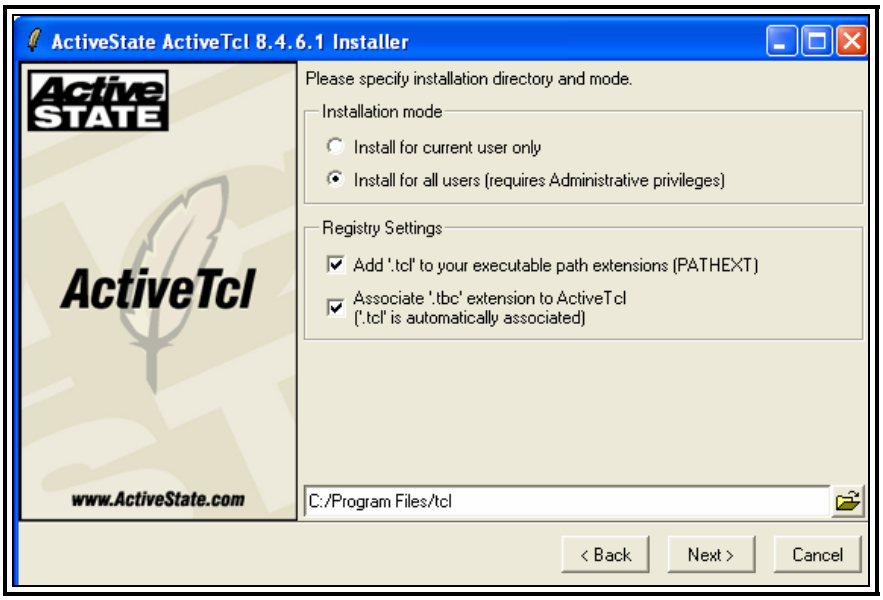

Figura 5.3: Cambio de directorio a C:\Program Files \Tcl.

Con esto queda instalado el interpretador de lenguaje TCL. Finalmente se debe descomprimir el archivo opensees.exe, y copiarlo en la ubicación que el usuario desee, preferiblemente en una carpeta en la que se vayan a guardar los archivos .tcl que vayan a generarse.

Una vez instalados estos dos componentes, OpenSees está listo para ser usado. Al hacer doble clic sobre el icono ejecutable de openSees aparecerá una ventana como la mostrada en la figura 5.4, la que corresponde a la interfaz mediante la cual el usuario ingresa los datos y comandos en lenguaje TCL para que sea procesada por OpenSees.

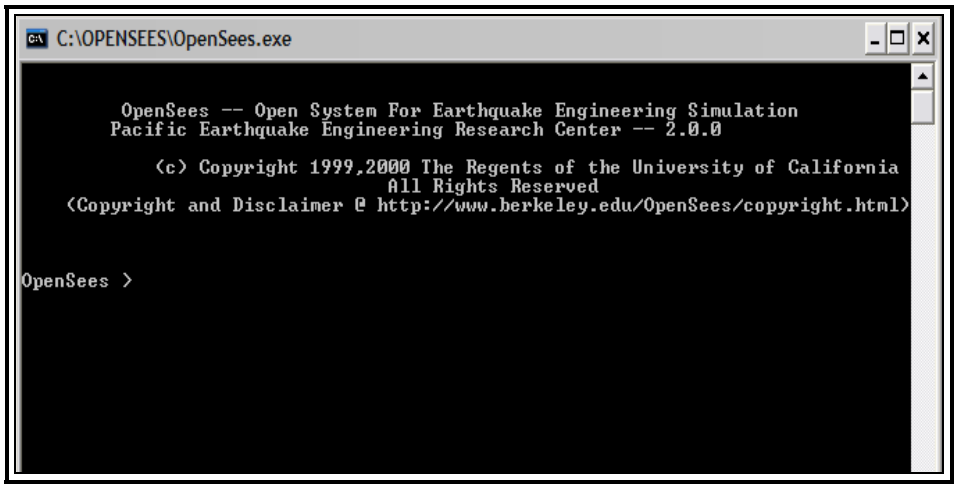

Figura 5.4: Ventana de OpenSees

Indudablemente que la ventana mostrada no resulta ser muy amigable. Más adelante se presentan las formas en las cuales pueden ser ingresados los comandos en lenguaje tcl para que sean analizados en OpenSEES, pues en la actualidad y gracias a la colaboración de muchos profesionales alrededor del mundo, se dispone de varias herramientas de software que nos permiten ingresar los códigos de una manera más amigable y que a la vez nos aseguran mayores niveles de eficiencia.

5.2.2 Instalación desde Internet (recomendada)

La instalación puede ser realizada a través del Internet, se recomienda este procedimiento porque así nos aseguramos de instalar la versión actualizada. La página desde donde se realiza la descarga es www.opensees.berkeley.edu.

Para descargar los archivos el usuario debe estar registrado, por lo tanto este es el primer paso que debe realizarse. El proceso de registro se lo hace mediante el link que se encuentra en la parte superior derecha, en donde se debe dar alguna información personal.

Figura 5.5: Ventana de inicio de la página de OpenSees

Una vez registrado se desplegará una pantalla similar a la mostrada en la figura 5.6. Se muestra la versión actual disponible y se solicita ingresar la dirección de correo electrónico del usuario registrado. Una vez realizado esto, debemos hacer clic en Submit.

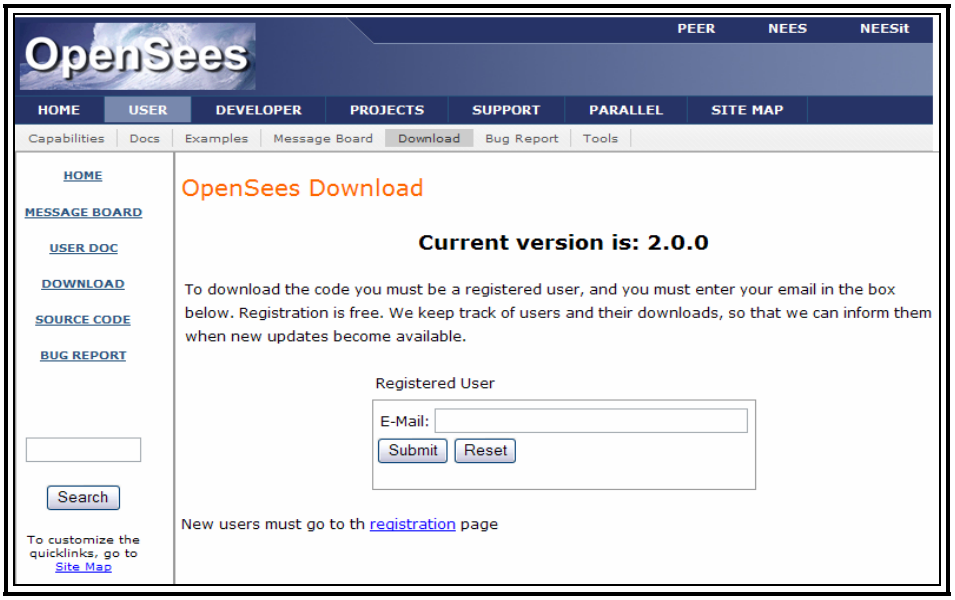

Figura 5.6: Ventana para registro

La siguiente ventana que aparece da una breve descripción respecto de los archivos que se encuentran disponibles para la descarga. En la parte inferior de la figura 5.7, vemos la opción para descargar los dos archivos. En este caso las versiones son: **OpenSees2.0.0.exe** y **tcl/tk8.4.6**.

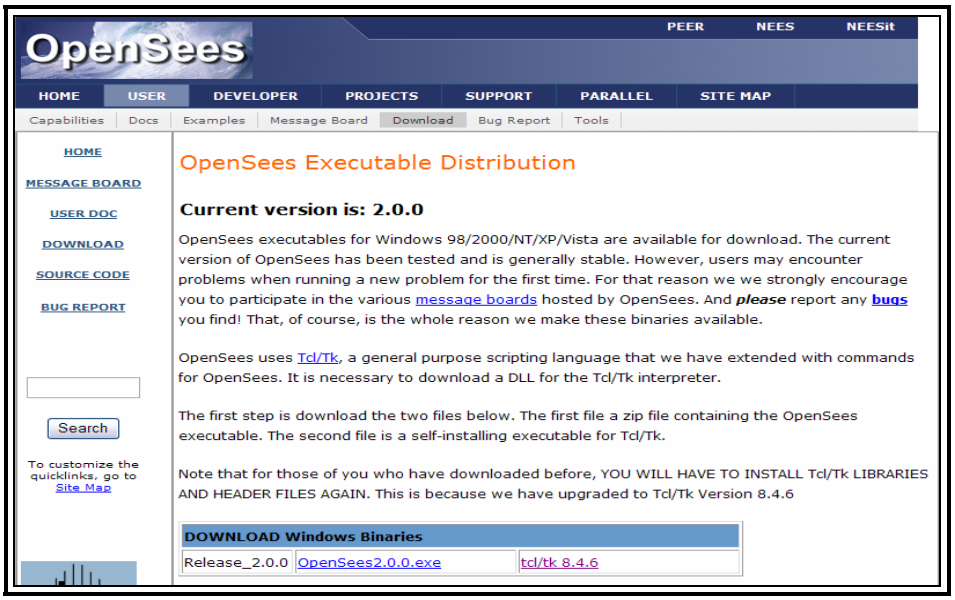

Figura 5.7: Ventana para descarga de archivos

Hacemos clic en el vínculo del primer archivo para descargarlo y nos aparecerá una ventana de diálogo que nos pregunta si deseamos abrir o guardar el archivo, podemos seleccionar cualquiera de las dos opciones, sin embargo se recomienda guardar los archivos en una carpeta en el disco duro del computador.

Una vez descargado el archivo **OpenSees2.0.0.exe** , debemos hacer lo mismo con el archivo **tcl/tk8.4.6,** y procedemos a guardarlo en la misma carpeta en la que almacenamos el primer archivo.

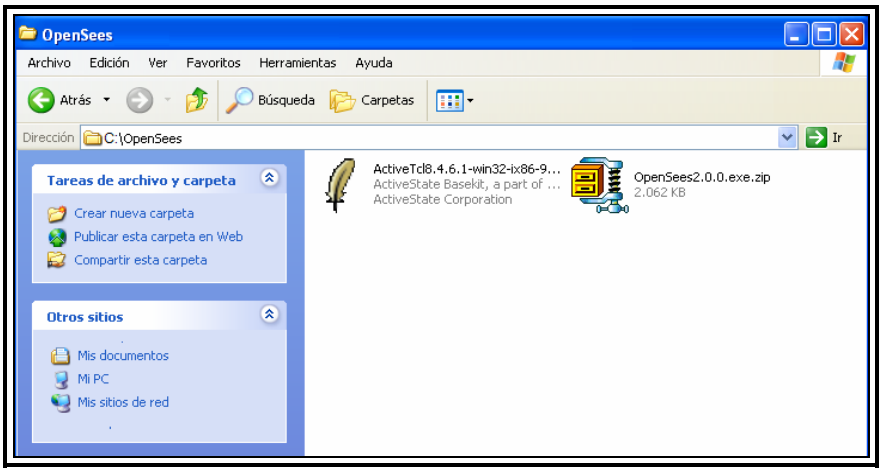

Figura 5.8: Archivos guardados en C:\OpenSees

A partir de este momento el proceso de instalación es idéntico al descrito para la instalación desde un dispositivo de almacenamiento extraíble.

.

5.3.- INGRESO DE COMANDOS OPENSEES - TCL

Es importante mostrar a los usuarios las distintas opciones que se tiene para el ingreso del código TCL. Por el momento no se hará hincapié en la codificación, puesto que dicho aspecto es abordado en detalle más adelante. El ingreso de comando OpenSees – Tcl puede hacerse de las siguientes maneras:

- Interactivo
- Ejecutar el archivo de entrada desde el prompt de OpenSees
- Ejecutar el archivo de entrada desde TCL Editor
- Otros.

5.3.1 Interactivo

La primera opción para ejecutar los comandos OpenSees – TCL, es a través de la pantalla mostrada en la figura 5.9, es decir directamente en el prompt del OpenSees.

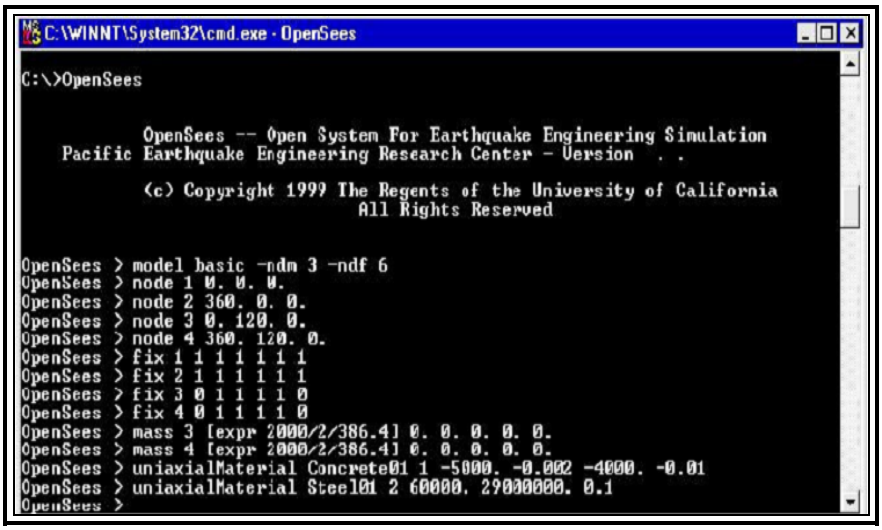

Figura 5.9: Ingreso Interactivo de comandos en OpenSees

Cada instrucción como se puede ver debe ser ingresada en una línea. Este formato de ingreso de código es poco utilizado, sobre todo por la poca eficiencia que se obtiene con su utilización, pues cada instrucción es almacenada en memoria y una vez ingresada no puede ser modificada, a menos que se vuelva a introducir la instrucción corregida, es semejante al trabajo en DOS.

5.3.2 Ejecutar el archivo de entrada desde el prompt de OpenSees

Este es uno de los métodos más usados. Consiste en generar un archivo externo que contiene los comandos de entrada. Este archivo debe tener la extensión correspondiente (.tcl), y puede ser generado mediante los siguientes programas:

- Bloc de notas
- Editor de Matlab
- Word pad.
- Editor tcl, etc.

En la figura 5.10 se ha generado un archivo de entrada en Bloc de notas, corresponde al mismo ejemplo que se mostró en la figura 5.9.

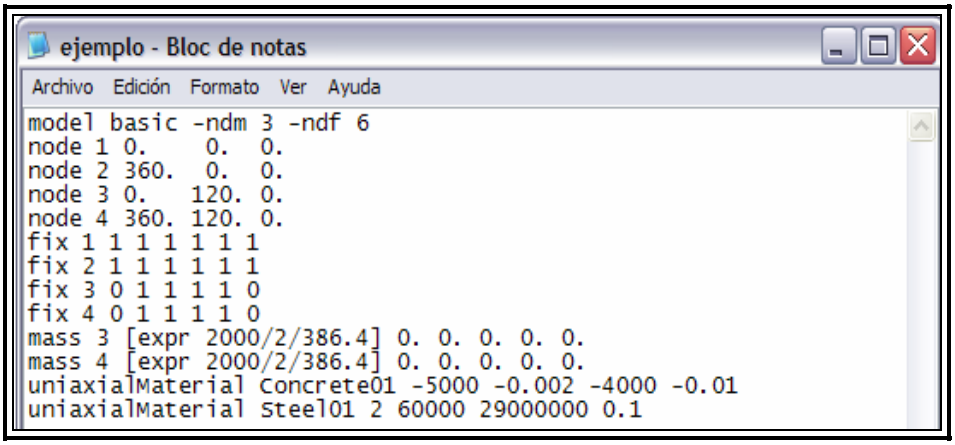

Figura 5.10: Archivo Tcl generado en Bolc de notas

Este archivo será ejecutado en OpenSees a través del prompt, mediante la instrucción **source,** la cual buscará en el disco el archivo solicitado. En la figura 5.11 se muestra la forma de ingreso del archivo externo, al cual se lo ha llamado ejemplo.

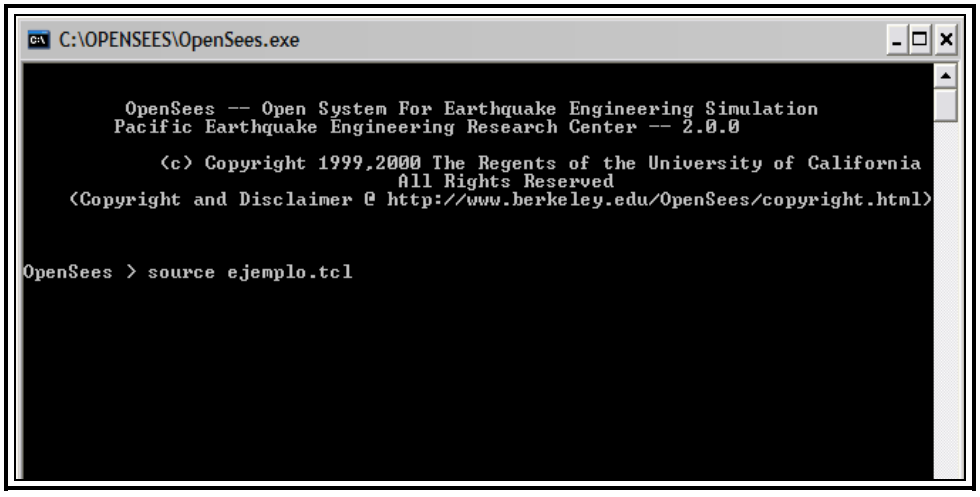

Figura 5.11: Ventana de inicio de la página de OpenSees

Es evidente la conveniencia de usar este método versus el anterior, pues al trabajar externamente se pueden realizar las modificaciones que sean necesarias y cuando el archivo esté listo podemos ejecutarlo en OpenSees, tantas veces como queramos y con las modificaciones que fueran del caso.

5.3.3 Ejecutar el archivo de entrada desde TCL Editor.

Como se manifestó anteriormente, el archivo de datos puede ser generado en distintos programas, lo que debemos asegurarnos es de que dicho archivo tenga la extensión correspondiente (.tcl). Los programas como Bloc de Notas o Word generan archivos .txt y .doc respectivamente, razón por la cual la extensión del archivo de entrada deberá ser cambiada al momento de grabarlo. Sin embargo, una alternativa válida es el uso del editor de Tcl (tcleditor), desarrollado por Rohit Kaul y por el Dr. Gregory Deierlein en la Universidad de Stanford.

Este editor genera archivos con la extensión (.tcl), además que tiene una serie de herramientas que resultan muy útiles al momento de programar en lenguaje TCL, tal como una lista de comandos que de despliega en la parte izquierda de la ventana y una completa explicación de los mismos que se muestra en la parte inferior de la misma, además de aquello los comandos de OpenSEES son mostrados en distintos colores, lo que sin duda alguna constituye una importante referencia para el programador, igualmente permite correr el archivo en OpenSEES directamente desde el editor Tcl, etc. Este editor puede ser descargo desde Internet para su instalación. En la figura se muestra la ventana de TclEditor.

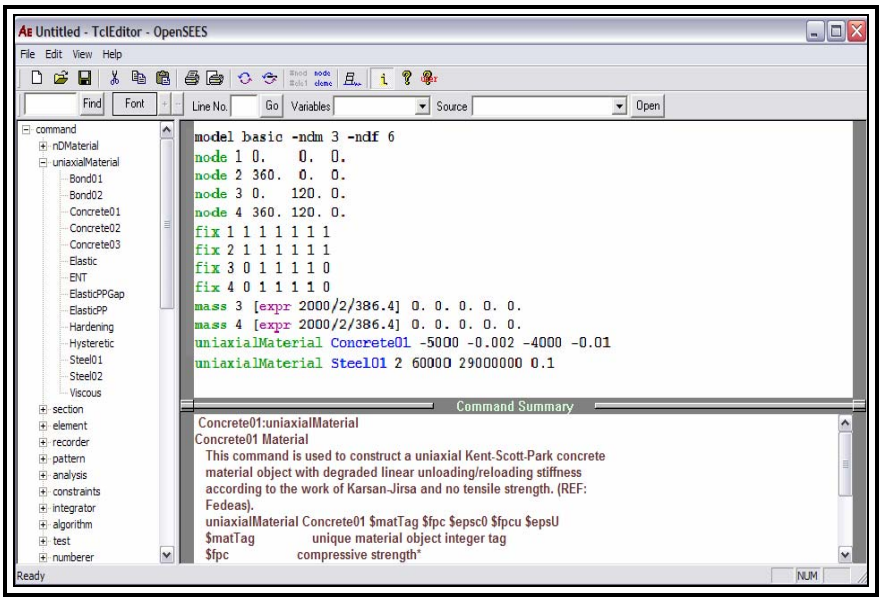

Figura 5.12: Ventana de TclEditor

Finalmente se menciona que existen formas adicionales de generar archivos de entrada para ser analizados en OpenSees, siempre en pos de obtener niveles de eficiencia mayores, pues como es de suponerse, el ingreso dato a dato además de ser tedioso, se presta para involuntariamente cometer errores. Es así que mediante lenguajes de programación como C++ o Visual Basic, se puede automatizar los procesos y generar archivos TCL, e incluso hacer correr OpenSees a través de instrucciones adecuadas. Esta metodología es superior a las que anteriormente se han citado, razón por la cual se le hace referencia.

5.4.- MANUAL DE USUARIO DE OPENSEES

En la página oficial de OpenSEES en Internet, está disponible un completo manual de comandos. Mencionado manual tiene alrededor de 500 páginas, por lo que indudablemente ante cualquier duda lo mejor y lo recomendable es recurrir a esta fuente. Permanentemente se hará mención a este manual, pues

ha sido la principal referencia para la elaboración del presente capítulo. Con fines prácticos y en vista del alcance del presenta trabajo, solamente se da un descripción de las instrucciones más usadas y que son aplicables a los ejemplos que se analizarán en este capítulo.

La creación de modelos y el posterior análisis en OpenSEES pueden ser divididos en varios módulos, cada uno de los cuales serán descritos a continuación:

- Construir el modelo
    - Definir variables y parámetros
    - Construir modelo y definir nudos
    - Definición de materiales
    - Definición de elementos
- Definición archivos de salida
    - Definir la generación de archivos de salida
    - Definir la impresión de datos durante el análisis
- Cargas Gravitatorias
    - Definir cargas gravitatorias
    - Realizar Análisis de cargas gravitatorias
- Análisis Estático
    - Definir análisis estático pushover
    - Realizar análisis estático pushover
- Análisis Dinámico
    - Definir Análisis considerando acción sísmica

- Realizar Análisis considerando acción sísmica

5.4.1 Creación del Modelo

En el presente apartado se describen algunas instrucciones orientadas a la solución y análisis de un pórtico plano. En primer lugar se presentan los comandos básicos y posteriormente se presentará la codificación como tal.

La forma en que se presentan los comandos corresponde a la misma utilizada por el manual de usuario de OpenSees, pues es importante que el usuario se familiarice con su notación en inglés.

Comando [wipe].- Este comando es usado para limpiar la memoria antes de empezar a generar un nuevo modelo. Es una práctica correcta colocarlo al inicio de cada aplicación.

Comando [set].- Este comando es usado para asignar un valor a una variable.

```
set variable $value
```

Por ejemplo para asignar a la variable **f** el valor de 4, se escribe:

set f 4

Comando [exp].- Este comando es usado para evaluar una expresión aritmética. Se muestran algunos ejemplos:

expr 2 + 3 # Suma 2 + 3
expr 2 + $a # Suma 2 + $a, una vez que se ha asignado un valor a la variable **a**.

Comando [Basic Model Builder].- Este comando es usado para construir un modelo, el cual puede ser definido en 1, 2, o 3 dimensiones.

```
model BasicBuilder -ndm $ndm <-ndf $ndf>
```

$ndm Dimensiones del problema (1, 2,3)

$ndf Número de grados de libertad por nudo (opcional)

Por ejemplo:

model BasicBuilder -ndm 2 -ndf 3 # Modelo en dos dimensiones y 3 GDL por nudo.

Se deja constancia y se recalca la importancia de escribir la instrucción tal y como se muestra en el manual, tomando en cuenta los espacios, las mayúsculas, los guiones etc. Conviene resaltar que las expresiones que se encuentren entre **< >** son datos opcionales y los comentarios deberán ser precedidos por el signo de numeral (#) y cuando dichos comentarios se coloquen a continuación de una línea de comandos deberá antecederle el signo (;). Igualmente y aunque es evidente, se recuerda que para los modelos en 3D (Espacial) se requerirá el ingreso de datos adicionales correspondientes a las dimensiones que para este tipo de análisis se deben considerar (x, y, z).

Comando [node].- Este comando es usado para construir un nudo. Este asigna las coordenadas y las masas al nudo.

```
node $nodeTag (ndm $coords) <-mass (ndf $MassValues)>
```

$nodeTag etiqueta que identifica al nudo

$coords coordenadas nodales

$MassValues masas nodales correspondientes a cada GDL

Por ejemplo:

node 1 0 0 # nudo 1, coordenadas x, y (0,0), corresponde a modelo en 2D.

node 2 0 180 -mass $m $m $mR #nudo 2, coordenadas x, y (0,180), masas en x e y igual a $m y en z igual a $mR, variables que se deben haber definido previamente.

Las masas en cada nudo pueden ser también definidas mediante el siguiente comando [mass].

Comando [mass].- Este comando es usado para asignar la masa a un nudo.

```
mass $nodeTag (ndf $MassValues)
```

$nodeTag etiqueta que identifica al nudo

$MassValues masas nodales correspondientes a cada GDL

Por ejemplo:

mass 2 6.8 0.0 0.0 # node 2, Mx (6.8), My(0.0) Mz(0.0), se desprecia la inercia rotacional en los nudos.

En caso de que la masa sea cero, se sugiere colocar un valor muy pequeño en lugar de cero, esto para asegurar la convergencia de la solución. En caso de desarrollar un modelo en 3D, se requiere el ingreso de masas para los 6 grados de libertad que corresponden.

Comando [fix].- Este comando es usado para definir las restricciones en los nudos correspondientes a los apoyos.

```
fix $nodeTag (ndf $ConstrValues)
```

$nodeTag etiqueta que identifica al nudo

$ConstrValues tipo de restricción. Se establece restringido (1) y no restringido (0)

Por ejemplo:

```
fix 1 1 1 0;          #  nudo 1, DX DY RZ (Para modelo 2D)
fix 3 1 1 1 1 1 1     # nudo 3, totalmente restringido (Para un modelo en 3D)
```

En cuanto a los materiales el OpenSees dispone de una amplia gama materiales que pueden ser usados para el análisis, es más, esta lista está en permanente actualización y no es raro que existan materiales que aún no constan en el manual de usuario de OpenSees. A continuación se describen brevemente algunos de ellos:

Los modelos de materiales disponibles en OpenSees se utilizan para construir un objeto uniaxial el cual representa relaciones uniaxiales fuerza-deformación.

- **Concrete01:** Este material asume que el concreto no posee resistencia a la tensión. Es un modelo propuesto por Kent-Scott-Park (1971) con degradación lineal de rigidez carga-descarga de acuerdo con el trabajo de Karsan-Jirsa (Karsan and Jirsa 1969).

- **Concrete02**: Este material uniaxial asume que el concreto posee resistencia a la tensión, considera un suavizado lineal en la zona de tensión.

- **Concrete03:** Es un modelo de material que considera al concreto con resistencia a tensión y suavizado no lineal en la zona de tensión del diagrama esfuerzo-deformación.

- **Concrete04:** Este modelo es usado para construir un material uniaxial con degradación lineal de rigidez carga-descarga de acuerdo con el trabajo de Karsan-Jirsa. Este material considera además un esfuerzo a tensión con un descenso exponencial.

- **Steel01:** Este tipo de material representa acero con propiedades esfuerzo-deformación que siguen un diagrama bilineal con endurecimiento cinemático y endurecimiento isotrópico opcional descrito por una ecuación no lineal.

- **Steel02**: Con este modelo se obtiene un material de acero uniaxial con endurecimiento de deformación isotrópico, además provee control sobre la transición de la región elástica a la plástica. Los efectos de endurecimiento de deformación son opcionales y pueden ser especificados en compresión o tensión.

Se presenta a continuación las instrucciones que permiten definir los materiales más usados:

Material Elástico [Elastic Material].- Este comando es usado para construir un material elástico uniaxial.

```
uniaxialMaterial Elastic $matTag $E <$eta>
```

$matTag etiqueta que identifica al material

$E Tangente (Módulo Elasticidad)

$eta Tangente amortiguamiento (opcional, por defecto 0).

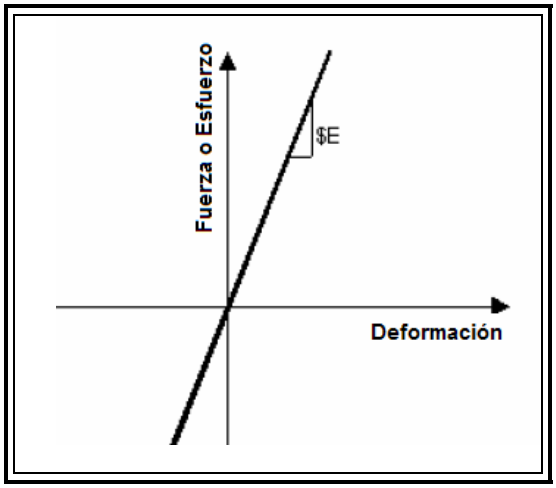

Figura 5.13: Diagrama Esfuerzo – Deformación Elastic Material

Material Elástico Perfectamente Plástico [Elastic-Perfectly Plastic Material].- Este comando es usado para construir un material uniaxial elástico perfectamente plástico.

```
uniaxialMaterial ElasticPP $matTag $E $epsyP <$epsyN $eps0>
```

$matTag etiqueta que identifica al material

$E Tangente (Módulo Elasticidad)

$epsyP Deformación en la cual el material alcanza el estado plástico en tensión.

$espyN Deformación en la cual el material alcanza el estado plástico en compresión.

$eps0 Deformación inicial

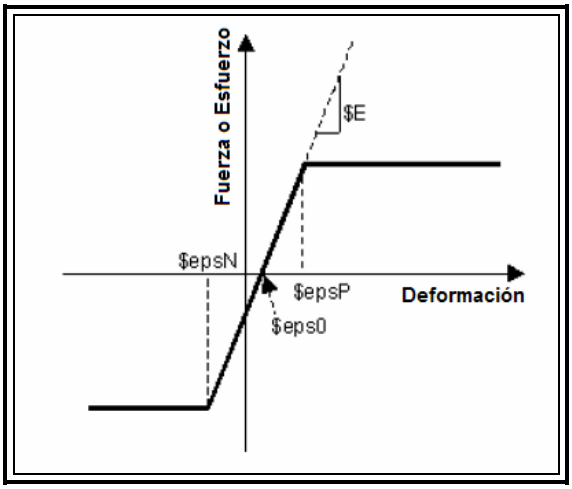

Figura 5.14: Diagrama Esfuerzo – Deformación
Elastic Perfectly Plastic Material

Comando [Concrete01].- Este comando es usado para construir un material uniaxial Concreto con degradación lineal de rigidez carga – descarga acorde con el trabajo de Karsan y Jirsa, no soporta esfuerzos de tensión.

```
uniaxialMaterial Concrete01 $matTag $fpc $epsc0 $fpcu $epsU
```

$matTag etiqueta que identifica al material

$fpc Resistencia a la compresión a los 28 días (compresión es negativa).

$epsc0 Deformación del concreto en el esfuerzo máximo

$fpcu Esfuerzo de fractura del concreto

$epsU Deformación del concreto al esfuerzo de fractura

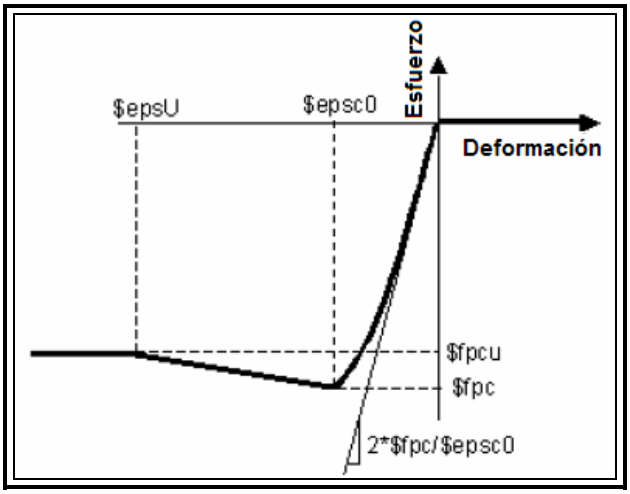

Figura 5.15: Diagrama Esfuerzo – Deformación Concrete01

Comando [Concrete02].- Este comando es usado para construir un material uniaxial de Concreto con esfuerzo de tensión, considera un suavizado lineal en la zona de tensión.

```
uniaxialMaterial Concrete02 $matTag $fpc $epsc0 $fpcu $epscu
$lambda $ft $Ets
```

$matTag etiqueta que identifica al material

$fpc Resistencia a la compresión a los 28 días.

$epsc0 Deformación del concreto en la resistencia a la compresión.

$fpcu Esfuerzo de fractura del concreto

$epscu Deformación del concreto al esfuerzo de fractura

$lambda coeficiente entre la pendiente descarga al $epscu y la pendiente inicial del esfuerzo de tensión.

$ft Esfuerzo de tensión

$Ets	Resistencia a tensión (pendiente de la rama lineal de tensión).

Es importante resaltar que los parámetros de resistencia y deformación a compresión deben ser definidos como negativos; mientras los parámetros de resistencia y deformación a tensión del concreto son considerados positivos.

Comando [Concrete03].- Este comando es usado para construir un material uniaxial de Concreto resistencia a la tensión, considera un suavizado no lineal en la zona de tensión del diagrama esfuerzo deformación.

```
uniaxialMaterial Concrete03 $matTag $fpc $epsc0 $fpcu $epscu
$lambda $ft $epst0 $ft0 $beta $epstu
```

$matTag	etiqueta que identifica al material
$fpc	Resistencia a la compresión a los 28 días.
$epsc0	Deformación del concreto en la resistencia a la compresión.
$fpcu	Esfuerzo de fractura del concreto
$epscu	Deformación del concreto al esfuerzo de fractura
$lambda	coeficiente entre la pendiente descarga al $epscu y la pendiente inicial del esfuerzo de tensión.
$ft	Esfuerzo de tensión
$epst0	Deformación de tensión en la transición del suavizado no lineal a lineal.
$ft0	Esfuerzo de tensión en la transición del suavizado no lineal a lineal.
$beta	Exponente de la curva de suavizado de tensión.

$epstu Deformación última de tensión.

Comando [Steel01].- Este comando es usado para construir un material uniaxial bilineal de Acero con endurecimiento cinemático y endurecimiento isotrópico opcional descrito por una ecuación no lineal.

```
uniaxialMaterial Steel01 $matTag $Fy $E0 $b <$a1 $a2 $a3 $a4>
```

$matTag etiqueta que identifica al material

$Fy Esfuerzo de fluencia

$E0 Rigidez elástica inicial

$b coeficiente endurecimiento - deformación (entre la rigidez post fluencia y la rigidez elástica inicial)

$a1, $a2, $a3 Parámetros de endurecimiento isotrópico.

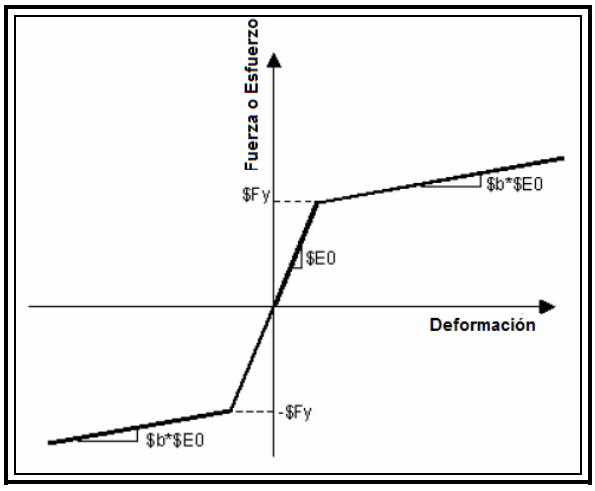

Figura 5.16: Diagrama Esfuerzo – Deformación Steel01

A continuación se describen los comandos más utilizados para la creación de secciones, se definen los tipos que pueden crearse en OpenSEES.

- **Elástica**: Este tipo de sección queda definida por el material y las constantes geométricas.

- **Resultante**: Descripción general no lineal de la respuesta fuerza - deformación. Por ejemplo momento – curvatura.

- **Fibra:** Es una sección que se discretiza en regiones más pequeñas llamadas “parches”, las cuales tienen formas geométricas simples y regulares (circulares, rectangulares, cuadriláteros), para estas subregiones la respuesta esfuerzo-deformación de los materiales es integrada a fin de obtener un comportamiento resultante para la sección. Por ejemplo las secciones de hormigón armado.

Sección Elástica [section Elastic].- Este comando es usado para construir una sección elástica.

```
section Elastic $secTag $E $A $Iz <$Iy $G $J>
```

$secTag	etiqueta que identifica a la sección.
$E	Módulo de Young
$A	Área de la sección transversal
$Iz	Momento de inercia del área con respecto al eje Z
$Iy	Momento de inercia del área con respecto al eje Y (opcional, usado en análisis 3D).
$G	Módulo de Corte (opcional, usado en análisis 3D).

$J Momento de inercia torsional (opcional, usado en análisis 3D).

Por ejemplo:

section Elastic 1 29000 100 100000 80000 20000 100000 # crea una sección elástica con identificación 1.

Sección Fibra [Fiber Section].- Este comando es usado para construir una sección tipo fibra.

```
section Fiber $secTag{
        fiber <fiber arguments>
        patch <patch arguments>
        layer <layer arguments>
}
```

$secTag etiqueta que identifica a la sección.

Esta herramienta es muy útil, con ella se puede definir secciones como las de hormigón armado, su uso se describe a continuación mediante un pequeño ejemplo.

Se plantea la siguiente sección de hormigón armado compuesta de tres capas (parches): hormigón no confinado, hormigón confinado y refuerzo de acero. La altura de la sección es HCol y la base Bcol.

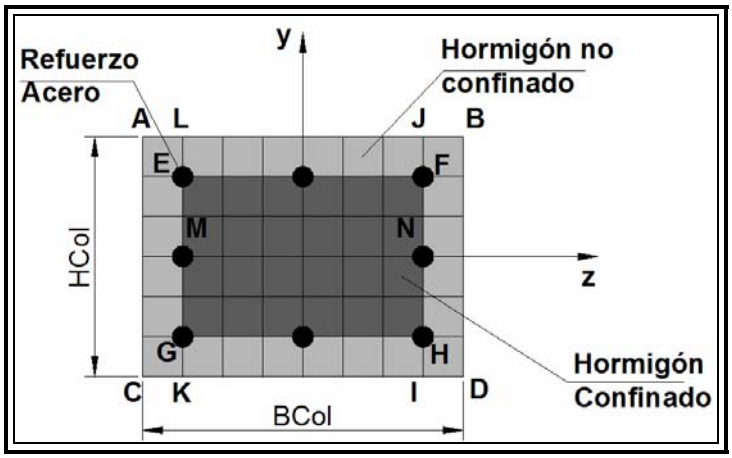

Figura 5.17: Sección tipo Fibra

Existen varias opciones para definir las secciones tipo fibra, a continuación se describe brevemente algunas de ellas, para mayores detalles será necesario recurrir al manual de usuario de Open Sees:

- **patch quad:** Permite construir una sección (parche) en forma de cuadrilátero, por lo tanto será necesario definir los cuatro vértices de la sección.
- **patch rect:** Permite construir una sección (parche) de forma rectangular, para esta instrucción se necesita definir las coordenadas de dos vértices opuestos.
- **patch circ**: Permite construir una sección (parche) de forma circular.
- **layer straight:** Esta instrucción es utilizada para construir las capas de refuerzo de acero.

En el presente ejemplo se trabaja con la instrucción **patch rect,** pues los "parches" con los que se trabaja son rectangulares. En tal virtud a continuación se presenta el detalle de la instrucción:

```
patch rect $matTag $numSubdivIJ $numSubdivJK $yI $zI $yJ $zJ
```

$matTag	etiqueta que identifica al material definido previamente.
$numSubdivIJ	Número de subdivisiones (fibras) en dirección z.
$numSubdivJK	Número de subdivisiones (fibras) en dirección y
$yI $zI	Coordenadas de un vértice.
$yJ $zJ	Coordenadas del vértice opuesto.

Otra instrucción que se utilizará para definir la sección de refuerzo de acero del ejemplo es el comando **layer straight,** el cual se describe a continuación:

```
layer straight $matTag $numBars $areaBar $yStart $zStart $yEnd $zEnd
```

$matTag	etiqueta que identifica al material definido previamente.
$numBars	Número de barras de refuerzo en la capa definida.
$areaBar	Área individual de las barras de refuerzo.
$yStart $zStart	Coordenadas (y, z) del punto inicial de la capa de refuerzo.
$yEnd $zEnd	Coordenadas (y, z) del punto final de la capa de refuerzo.

Se ha denominado "cover" al recubrimiento. Se muestra a continuación la codificación para crear la sección fibra mencionada.

```
#PARÁMETROS AUXILIARES
set y1 [expr $HCol/2.0] # Asigno a y1 el valor igual a HCol/2.
set z1 [expr $BCol/2.0] # Asigno a z1 el valor igual a BCol/2.
```

```
# CREO SECCIÓN FIBRA

section Fiber 1 {

# CREO LAS FIBRAS DE CONCRETO CONFINADO (Rectángulo EFHG)
patch rect 1 4 6  [expr $cover-$z1]  [expr $cover-$y1]  [expr $z1-
$cover]  [expr $y1-$cover] #COORDENADAS DE PUNTOS G Y F, DIVIDO LA
SECCIÓN EN UNA MATRIZ DE 4X6 Y LE ASIGNO MATERIAL 1.

# CREO LAS 4 FIBRAS DE CONCRETO NO CONFINADO(DER, IZQU, ABAJO, ARRIBA)
patch rect 2 6 1  [expr $z1-$cover] [expr -$y1] $z1  $y1 # Rectángulo
JIDB, DIVIDO LA SECCIÓN EN UNA MATRIZ DE 6X1 Y LE ASIGNO MATERIAL 2.
DOY COORDENADAS DE I Y B.

patch rect 2 6 1  [expr -$z1] [expr -$y1] [expr $cover-$z1]  $y1 #
Rectángulo ALKC, DIVIDO LA SECCIÓN EN UNA MATRIZ DE 6X1 Y LE ASIGNO
MATERIAL 2, DOY COORDENADAS DE C Y L.

patch rect 2  1 6  [expr $cover-$z1] [expr -$y1]  [expr $z1-$cover]
[expr $cover-$y1] # Rectángulo GHIK, DIVIDO LA SECCIÓN EN UNA MATRIZ
DE 1X6 Y LE ASIGNO MATERIAL 2, DOY COORDENADAS DE K Y H.


patch rect 2  1 6  [expr $cover-$z1] [expr $y1-$cover] [expr $z1-
$cover] $y1 # Rectángulo LEFJ, DIVIDO LA SECCIÓN EN UNA MATRIZ DE 1X6
Y LE ASIGNO MATERIAL 2, DOY COORDENADAS DE E Y J.


# CREO LAS FIBRAS DE REFUERZO (ARRIBA, MEDIO, ABAJO)
layer straight 3 3 $Asc [expr $cover-$z1] [expr $y1-$cover] [expr $z1-
$cover] [expr $y1-$cover] # COORDENADAS DE E Y F, TRES BARRAS DE
MATERIAL 3 Y DE ÁREA INDIVIDUAL "Asc".

layer straight 3 2 $Asc [expr $cover-$z1] 0.0  [expr $z1-$cover] 0.0 #
COORDENADAS DE M Y N, DOS BARRAS DE MATERIAL 3 Y DE ÁREA INDIVIDUAL
"Asc".

layer straight 3 3 $Asc [expr $cover-$z1] [expr $cover-$y1] [expr $z1-
$cover] [expr $cover-$y1] # COORDENADAS DE G Y H, TRES BARRAS DE
MATERIAL 3 Y DE ÁREA INDIVIDUAL "Asc".
#SECCIÓN FIBRA CREADA
}
```

Comando [Section Aggregator].- Este comando es usado para construir un objeto con grupos de objetos de material uniaxial previamente definidos, para formar una sección con un modelo fuerza – deformación único.

```
section Aggregator $secTag $matTag1 $string1 $matTag2 $string2...
<-section $sectionTag>
```

$secTag etiqueta que identifica a la sección

$matTag1 Materiales uniaxial previamente definidos

$matTag2….

$string1, Cantidades fuerza – deformación correspondientes a cada

$string2….. sección. Se utilizará P, Mz, Vy, My, Vz, T.

A través de esta instrucción se pueden crear secciones que describan un comportamiento resultante para una nueva sección. Por ejemplo se puede combinar los efectos de corte y momento a través de dos secciones previamente definidas que representen el efecto de corte y el de momento por separado.

Por ejemplo:

section Aggregator 2 2 Vy -section 4; # *crea una nueva sección (IDtag 2), tomando un material existente (tag 2) para representar el corte y lo añade a la sección existente (tag4), la cual puede ser una sección fibra en la cual la interacción entre la fuerza axial y la flexión ya se considera. De tal suerte que ahora se tiene una sección que considera el efecto de corte, la carga axial y el momento.*

En cuanto a los elementos, OpenSEES tiene una serie de alternativas para generarlos.

- Elementos Armadura (Truss Element)
- Elementos Armadura Corotacionales (Corotational Truss Element)
- Elementos Viga – Columna Elásticos (Elastic Beam Column Element)
- Elementos Viga Columna No Lineales (NonLinear Beam Column Elements)
- Elementos de longitud Cero (Zero – Lenght Elements)
- Elementos Muro (Brick Elements)
- Elementos membrana (Shell Elements)

- Elementos con plasticidad concentrada (Beam with hinges)
- Etc.

La lista de elementos que pueden ser creados en OpenSEES es bastante extensa y sumado a este hecho, debemos recordar que siendo una plataforma de fuente abierta, la lista de elementos sigue incrementándose permanentemente. En tal virtud es importante mantener las bases siempre actualizadas.

Comando [elasticBeamColumn].- Este comando es usado para construir un elemento Viga Columna elástico, los argumento necesarios para construir el elemento depende de la dimensión del problema.

Por ejemplo, para el caso de dos dimensiones (2D), se tiene lo siguiente:

```
element elasticBeamColumn $eleTag $iNode $jNode $A $E $Iz $transfTag
```

Para un problema de tres dimensiones (3D), se tiene lo siguiente:

```
element elasticBeamColumn $eleTag $iNode $jNode $A $E $G $J
$Iy $Iz $transfTag
```

$eleTag	etiqueta que identifica al elemento
$iNode $jNode	Nudos extremos
$A	Área transversal del elemento
$E	Módulo de Young.
$G	Módulo para Cortante
$J	Momento de Inercia Torsional de la sección
$Iz	Inercia de la sección respecto al eje local z

$Iy	Inercia de la sección respecto al eje local y
$transfTag	Identificador para la transformación de coordenadas

Comando [nonlinearBeamColumn].- Este comando es usado para construir un elemento Viga Columna no lineal, el cual está basado en una formulación iterativa o no iterativa de fuerza, además que considera plasticidad distribuida a lo largo del elemento.

```
element nonlinearBeamColumn $eleTag $iNode $jNode $numIntgrPts
$secTag $transfTag <-mass $massDens> <-iter $maxIters $tol>
```

$eleTag	etiqueta que identifica al elemento
$iNode $jNode	Nudos extremos
$numIntgrPts	Número de puntos de integración a lo largo del elemento
$secTag	Identificador de una sección previamente definida
$transfTag	Identificador para la transformación de coordenadas
$massDens	Densidad de masa del elemento (por unidad de longitud)
$maxIters	Máximo número de iteraciones para alcanzar a satisfacer la compatibilidad del elemento.
$tol	Tolerancia para satisfacción de compatibilidad del elemento

Comando [zeroLength].- Este comando es usado para construir un elemento de longitud cero, que será definido por dos nudos en la misma ubicación. Los nudos son conectados por objetos de material uniaxial para representar la relación fuerza - deformación para el elemento.

```
element zeroLength $eleTag $iNode $jNode -mat $matTag1 $matTag2..
....-dir $dir1 $dir2...<-orient $x1 $x2 $x3 $yp1 $yp2 $yp3>
```

$eleTag	etiqueta que identifica al elemento
$iNode $jNode	Nudos extremos
$matTag1	Materiales uniaxial previamente definidos
$matTag2….	
$dir1 $dir2	Direcciones para el material:
	1, 2, 3 traslación respecto a ejes locales x, y, z.
	4, 5, 6 rotaciones respecto a ejes locales x, y, z.

5.4.2 Definición de archivos de salida (Recorder)

Los recorders permiten monitorear parámetros definidos por el usuario en el modelo durante el análisis. Por citar algunos tenemos:

- La historia de desplazamientos en un nudo durante un análisis no estático.
- Fuerzas en los elementos, en coordenadas globales o locales.
- Derivas de piso.
- Reacciones en nudos de apoyo.
- Estado total del modelo en cada paso del proceso de solución, etc.

Como se indicó anteriormente, el usuario define mediante los “recorders” la información que desea obtener del análisis del modelo. Los recorders generarán archivos de salida en formato out, txt etc., el cual puede ser editado en bloc de notas, o puede ser importado desde Excel o Matlab para su manipulación y generación de gráficas, como se verá en los ejemplos posteriores.

A continuación se presenta un ejemplo de definición de los recorders, no obstante su uso se aclarará con la ayuda de algunos ejemplos que se muestran en este capítulo.

```
recorder Node -file Nudo3.out -time -node 3 -dof 1 2 disp
```

A través de este recorder se generará un archivo llamado Nudo3, el cual contendrá la historia de desplazamiento tanto horizontal como vertical (GDL 1 y 2) correspondiente en al nudo 3.

5.4.3 Definición de cargas

En OpenSEES la aplicación de carga es un proceso compuesto de tres pasos:

- Primero se definen las cargas en un patrón de carga.
- Se define el análisis y sus características.
- Las cargas son aplicadas cuando se ejecuta el análisis.

Los patrones de carga son definidos usando el comando **pattern**. Tres tipos de patrón de carga están disponibles actualmente:

a) **plain Pattern**: Este comando es usado para definir:
   - cargas nodales: cargas gravitatorias, cargas laterales.
   - Restricciones single point, control de desplazamiento en el nudo.
   - Cargas en lo elementos: cargas gravitacionales distribuidas.

b) **UniformExcitation Pattern**: Este patrón aplica las aceleraciones de un registro definido por el usuario a todos los nudos fijos, en una dirección específica.

c) **MultipleSupport Pattern**: Este patrón aplica los desplazamientos de un registro definido por el usuario en los nudos especificados, en una dirección especificada, o también un acelerograma.

5.4.4 Definición del Análisis

El la parte correspondiente al análisis, OpenSEES permite hacer uso de diferentes herramientas disponibles para análisis lineal y no lineal. Para cada análisis se deben definir los siguientes ítems, y de acuerdo a la recomendación del manual de usuario, preferentemente deberán ser definidos en el orden mostrado:

Comando [constraints]: Este comando es usado para construir un objeto gestor de las restricciones. El gestor determina como las ecuaciones de las restricciones son aplicadas en el análisis. Las ecuaciones de las restricciones aplican un valor específico para un GDL, o una relación entre los grados de libertad.

- Plain
- Lagrange
- Transformation
- Penalty

Comando [numberer]: Esta instrucción construye un objeto para numeración de los grados de libertad. Éste determina la asignación entre el número de ecuaciones y los grados de libertad, como los grados de libertad son numerados. Para ello se tienen dos opciones:

- Plain: Usa la numeración dada por el usuario.
- RCM: Vuelve a numerar los grados de libertad para minimizar el ancho de banda de la matriz usando el algoritmo reverso de Cuthill-McKee.

Comando [system]: Este comando es usado para construir un Sistema de Ecuaciones lineal y un objeto de solución lineal.

Comando [test]: Este comando es usado para construir un objeto para prueba de convergencia. Algunos algoritmos de solución requieren pruebas de convergencia para determinar si la convergencia ha sido alcanzada al final de un paso de iteración.

Comando [algorithm]: Este comando es usado para construir un objeto para determinar el algoritmo de solución, el cual determina la secuencia de pasos tomados para resolver la ecuación no lineal.

- Linear: Usa la solución de la primera iteración y continúa.
- Newton: Usa la tangente en la iteración actual para iterar la convergencia.
- Modified Newton: Usa la tangente a la primera iteración para iterar la convergencia.

Comando [integrator]: Este comando se usa para construir un objeto Integrador. Éste determina el significado de los términos en el sistema de ecuaciones.

- Newmark: Método desarrollado por Newmark, es necesario definir los parámetros α y β.
- HHT: Método de Hilbert-Hughes-Taylor.
- Central Difference: Aproxima la velocidad y la aceleración centrando diferencias finitas de desplazamiento.

Comando [analysis]: Este comando es usado para construir un objeto de Análisis, el cual define el tipo de análisis que será realizado, los cuales pueden ser:

- Análisis Estático: Resuelve el problema Q=K*q, sin matrices de masa o amortiguamiento.
- Análisis Transient: Resuelve un análisis dependiente del tiempo. El paso de tiempo en este análisis en constante.
- Análisis Transient Variable: Resuelve el análisis dependiente del tiempo. El paso de tiempo en este análisis es variable.

En esta sección simplemente se ha presentado una breve referencia de lo que constituye el Análisis en OpenSEES, su formato de uso será mostrado en los ejemplos resueltos en este capítulo, no obstante el detalle de cada uno de los parámetros definidos hasta el momento se encuentra en el Manual de Usuario de OpenSees.

5.5.- ANÁLISIS SÍSMICO DE ESTRUCTURAS CON AISLADORES DE BASE USANDO OPENSEES

Para el caso de los aisladores de tipo FPS la utilización del elemento **zerolength** es una buena opción**.** Estos son elementos tridimensionales de longitud cero, con seis grados de libertad por nudo, a los cuales se les pueden asignar propiedades de materiales en cada grado de libertad independientemente. Sin embargo, usar elementos zerolength para modelar aisladores elastoméricos también es posible, pues la altura del dispositivo es considerablemente menor a la de la estructura soportada. Para modelar un aislador elastomérico usando elementos zerolength se puede asignar un material que represente la rigidez vertical (GDL 2) y otro material que representa la flexibilidad del dispositivo para el grado de libertad horizontal (GDL 1), es claro que los materiales deberán ser definidos previamente.

Para contruir la respuesta de los aisladores se puede trabajar con modelos lineales y bilineales, solo será cuestión de seleccionar el material adecuado, definir correctamente sus parámetros y asignar las propiedades del material a los grados de libertad correspondientes. En los ejemplo 4 y 5 que se muestran más adelante, se han incluido aisladores con comportamieto lineal, para constuir su respuesta lateral se ha utilizado un material **uniaxialMaterial Elastic**, para el cual solo es necesario definir la etiqueta del material, la rigidez (pendiente de la recta) y opcionalmente se define el amortiguamiento.

En la figura 5.17 se muestra a la izquierda constitutiva bilineal de un aislador, y a la derecha el comportamiento del material **uniaxialMaterial Elastic.** Pese a

que el aislador tiene comportamiento bilineal, mediante un proceso de linealización se puede trabajar con la rigidez efectiva Kef y utilizar un material como el antes mencionado. Para restringir los grados de libertad y representar la rigidez vertical del elemento se puede trabajar igualmente con un material elástico con una rigidez alta.

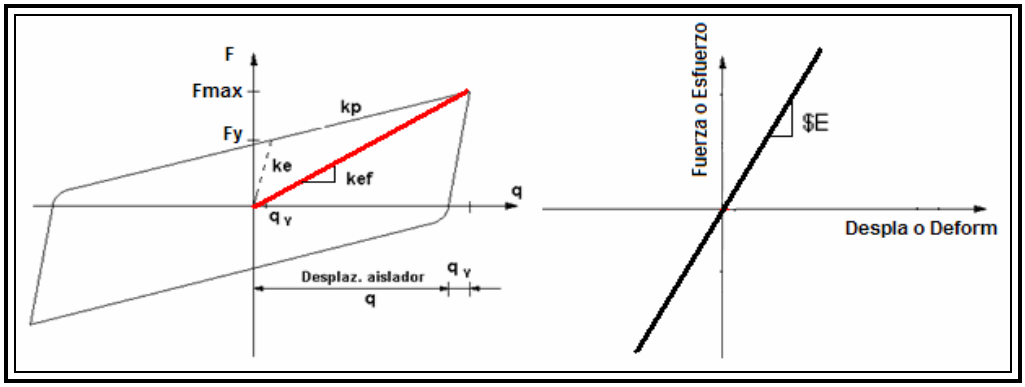

Figura 5.18: Rigidez Efectiva y Diagrama Fuerza – Deformación para material Elastic

Para modelar un aislador con comportamiento bilineal, se puede trabajar con un material uniaxialMaterial Steel01 (figura 5.15), para lo cual será necesario definir la fuerza de fluencia, la rigidez inicial y el coefinciente entre la rigidez inicial y la rigidez post – fluencia.

5.6.- EJEMPLOS

Se presentan a continuación varios ejemplos que ayudarán al lector a comprender de mejor manera los conceptos y las herramientas de OpenSees descritas hasta el momento. Es importante mencionar que los ejemplos son presentados en orden creciente de complejidad, desde pórticos planos con

base empotrada, hasta un ejemplo de estructura espacial que incluye aisladores de base.

No cabe duda que muchos de los procesos incluidos en los ejemplos, pudieron omitirse o desarrollarse de manera más eficiente, sin embargo no se lo ha hecho así por cuestiones didácticas. Una vez más se recuerda al lector la necesidad de recurrir permanentemente al manual de usuario de OpenSees, para solventar cualquier duda.

5.6.1 Ejemplo 1

Para comenzar se presenta un modelo sencillo en dos dimensiones (ndm 2) y de tres grados de libertad por nudo (ndf 3), el cual consiste en un pórtico de un vano y un piso, las columnas y vigas han sido modelados como elementos elásticos. La estructura está perfectamente empotrada en su base.

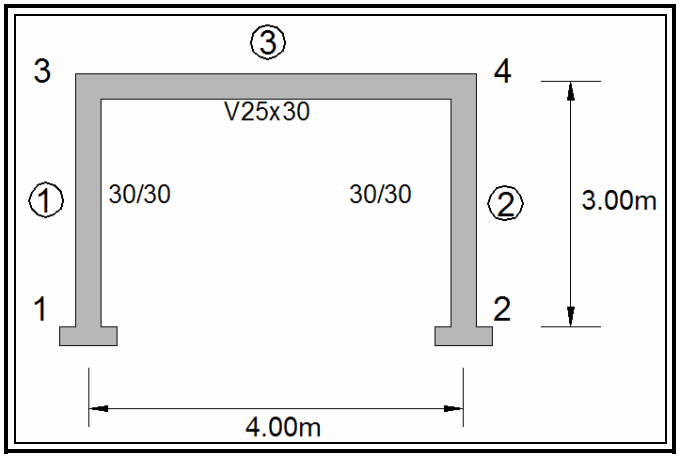

Figura 5.19: Pórtico Ejemplo 1

El modelo es sometido a cargas gravitatorias y a la acción dinámica uniforme inducida por el sismo del Centro, el archivo que contiene el registro de aceleraciones debe estar guardado en la misma carpeta que contiene al archivo tcl que se está generando. Adicionalmente se han colocado masas en los nudos 3 y 4, la masa tiene un valor de 0.6 Ts^2/m. Las propiedades de los materiales se muestran en el archivo de codificación. Los recorders definidos en el presente ejemplo serán almacenados en una carpeta llamada "Datos1". Con fines didácticos se ha asignado a la estructura un amortiguamiento tipo Rayleigh, con un factor de amortiguamiento $\xi = 0.05$.

Durante el análisis gravitatorio, la carga será aplicada linealmente con incrementos de 10% hasta alcanzar el 100% de la carga. Por otra parte, en el análisis dinámico, el sismo se aplicará en 4000 pasos con intervalos de 0.01, con lo que resultan 40 seg de aplicación de acción dinámica.

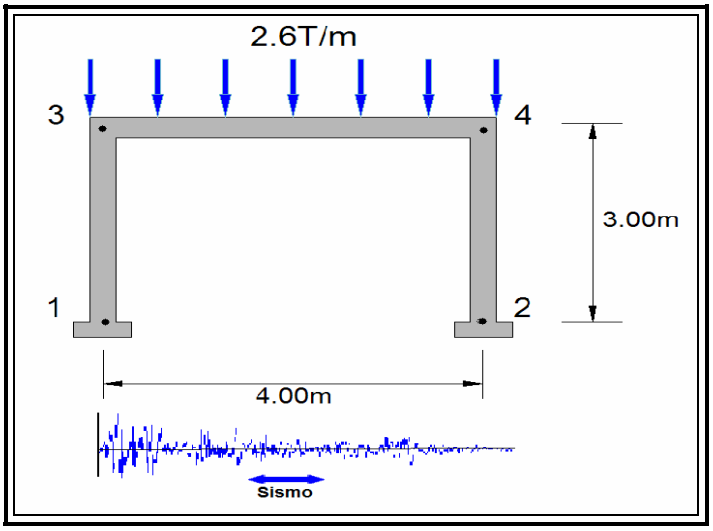

Figura 5.20: Acciones consideradas para ejemplo 1.

```
# EJEMPLO 1
# UNIDADES T, m, seg
# INICIO
#=========================================================
wipe;  # borra de la memoria modelos anteriores
model BasicBuilder -ndm 2 -ndf 3;  # defino el modelo 2d y 3gdl por nudo
set dataDir Datos1;  # asigno el nombre Datos1 al directorio datadir
file mkdir $dataDir;   # defino mi archivo de salida Datos1
puts "MODELO DEFINIDO!"

# COORDENADAS, RESTRICCIONES Y MASAS NODALES:
#=========================================================
node 1 0 0;                # nudo x, y
node 2 4 0
node 3 0 3
node 4 4 3
# RESTRICCIONES
fix 1 1 1 1;               # nudo DX DY RZ
fix 2 1 1 1
# MASAS NODALES:
mass 3 0.6  0.  0.;        # nudo#, Mx My Mz, Masa=Peso/g
mass 4 0.6  0.  0.;
puts "NUDOS CREADOS, RESTRINGIDOS Y MASAS ASIGNADAS!"

# DEFINO CARACTERÍSTICAS DE ELEMENTOS
#=========================================================
set LCol 3;                # longitud columna
set Lviga 4;               # longitud viga
set Wv 2.6;                # carga distribuido sobre la viga
set Peso 2.6*$Lviga;       # peso estructura
set xdamp 0.05;            # factor de amortiguamiento
set HCol 0.3;         # altura columna
set BCol 0.3;         # ancho columna
set Hviga 0.3;        # altura viga
set Bviga 0.25;       # base viga
set g 9.81;           # aceleración gravedad
set E 1800000;        # módulo de elasticidad del concreto


# CÁLCULO DE PARÁMETROS
set ACol [expr $BCol*$HCol];      # sección transversal para columna
set Aviga [expr $Bviga*$Hviga];   # sección transversal para viga
set IzCol [expr 1./12.*$BCol*pow($HCol,3)];     # Mom inercia de columna
set Izviga [expr 1./12.*$Bviga*pow($Hviga,3)]; # momento de inercia de viga


#TRANSFORMACIÓN GEOMÉTRICA
geomTransf Linear 1;       # asigno etiqueta de transformación geométrica
```

```
# CREO LOS ELEMENTOS
#=========================================================
#                                $eleTag $iNode $jNode $A $E $Iz $transfTag
element elasticBeamColumn 1 1 3 $ACol $E $IzCol 1        # elemento 1
element elasticBeamColumn 2 2 4 $ACol $E $IzCol 1        # elemento 2
element elasticBeamColumn 3 3 4 $Aviga $E $Izviga 1      # elemento 3
puts "ELEMENTOS CREADOS!"
# DEFINO RECORDERS
#=========================================================
recorder Node -file $dataDir/Dlibres.out -time -node  3 -dof 1 2 3 disp; #define
en donde se almacenarán desplazamientos de nudo 3.
recorder Node -file $dataDir/RBase.out -time -node 1 2 -dof 1 2 3 reaction;
#define reacciones en soportes
recorder Drift -file $dataDir/Deriv.out -time -iNode 1 2 -jNode 3 4 -dof 1  -
perpDirn 2 ;  # deriva lateral
recorder Element -file $dataDir/FCol.out -time -ele 1 2 globalForce; # fuerzas en
columnas
recorder Element -file $dataDir/FViga.out -time -ele 3 globalForce; # fuerzas en
vigas
puts "RECORDERS CREADOS!"

# DEFINO ANÁLISIS GRAVITATORIO
#=========================================================
pattern Plain 1 Linear {
   eleLoad -ele 3 -type -beamUniform -$Wv ; # carga distribuida sobre la viga
}
constraints Plain;     # forma en que se manejan las restricciones de nudo
numberer Plain;      # reenumera los dof para minimizar el ancho de banda
system BandGeneral; # como se resuelven las ecuaciones en el análisis
test NormDispIncr 1.0e-8 6 ; # determina si la convergencia ha sido alcanzada
algorithm Newton;            # usa el algoritmo de newton para la solución
integrator LoadControl 0.1; # aplico 10% de carga gravitatorio en cada paso
analysis Static;             # defino el tipo de análisis (estático)
analyze 10;                  # Indico a OpenSEES  realice 10 pasos de análisis
loadConst -time 0.0;  # Congela aplicación de cargas gravit y reinicia el tiempo
puts "ANÁLISIS GRAVITATORIO REALIZADO!"

# ANÁLISIS DINÁMICO (SISMO)
#=========================================================
# creamos patrón de carga
set accelSeries "Series -dt 0.02 -filePath centro.txt -factor 1"; # defino
acelerograma
pattern UniformExcitation 2 1 -accel $accelSeries;     # defino como y cuando
aplico aceleración
rayleigh 0. 0. 0. [expr 2*$xdamp/pow([eigen 1],0.5)];  # asigno amortiguamiento
basado en el primer modo

# CREAMOS EL ANÁLISIS
#=========================================================
wipeAnalysis; # borra los parámetros de análisis antes definidos
```

```
constraints Plain;     # como son consideradas las condiciones de borde
numberer Plain;        # reenumera los dor para minimizar ancho de banda
system BandGeneral; # como se resuelven las ecuaciones en el análisis
test NormDispIncr 1.0e-8 10; # determina si la convergencia ha sido alcanzada
algorithm Newton;            # usa el algoritmo de newton para la solución
integrator Newmark 0.5 0.25 ;# método de Newmark
analysis Transient;          # defino el tipo de análisis:dependiente del tiempo
analyze 4000 0.01;  # aplico 4000 0.01-sec intervalos de tiempo en el análisis
puts "ANÁLISIS DINÁMICO REALIZADO TIEMPO: [getTime]!"
```

El usuario podrá notar que en archivo de datos correspondiente a los desplazamientos del nudo 3 (Dlibres.out), aparecen varias columnas de datos, la primera corresponde al tiempo y las siguientes son los desplazamientos para los grados de libertada 1, 2 y 3 respectivamente. Las diez filas iniciales corresponden a las diez aplicaciones de carga gravitatoria que se especificarón en el análisis, luego de esto aparecen los resultados que corresponden a la aplicación de la acción dinámica.

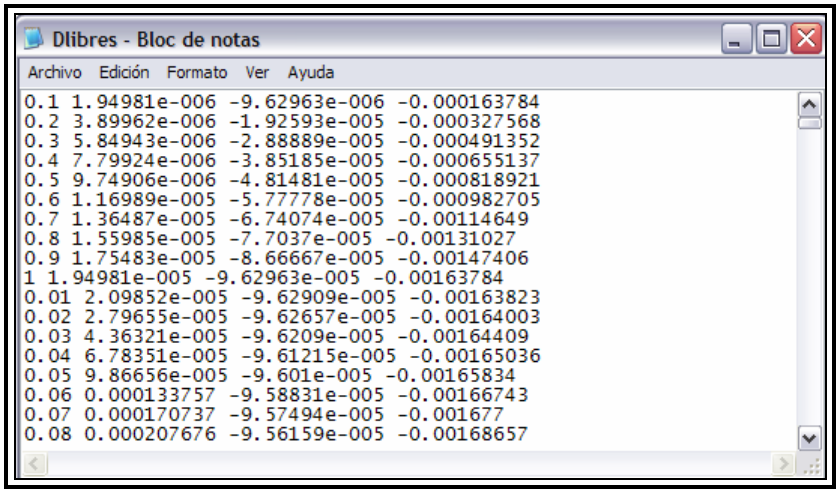

Figura 5.21: Historia de desplazamientos nudo 3 (GDL 1) formato txt.

Los resultados obtenidos pueden ser editados en Excel, Matlab etc. A continuación se presenta la historia de desplazamientos para el nudo 3, para el grado de libertad 1 (en sentido x).

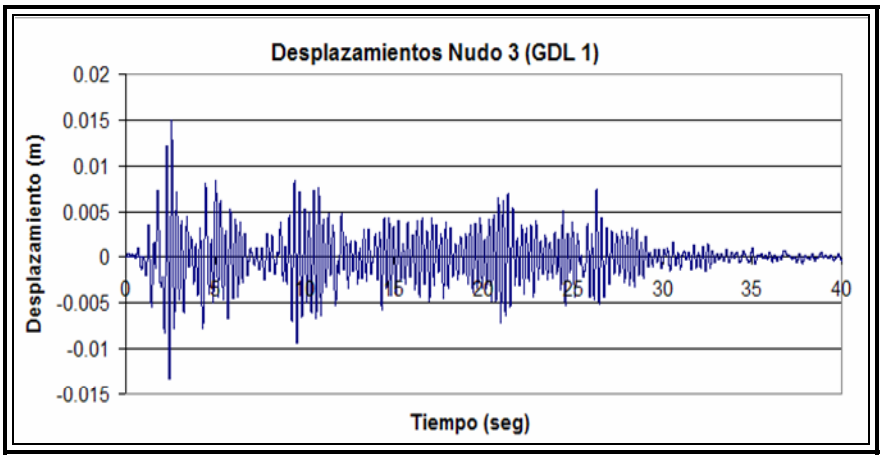

Figura 5.22: Historia de desplazamientos nudo 3 (GDL 1)

5.6.2 Ejemplo 2

Se presenta a continuación un pórtico de dos pisos y un vano en dos dimensiones (ndm 2) y tres grados libertad por nudo (ndf 3), columnas y vigas definidas como elementos fibra. La estructura esta empotrada en su base.

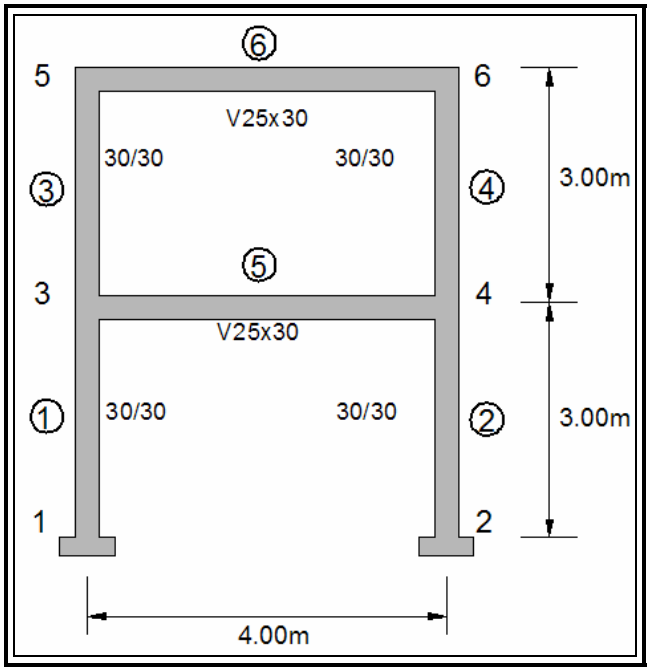

Figura 5.23: Pórtico Ejemplo 2

La estructura está sometida a cargas gravitatorias y a la acción dinámica inducida por el sismo del Centro. Se han colocado masas en los nudos 3, 4, 5 y 6. Los recorders serán almacenados en una carpeta llamada “Datos2”. En el presente ejemplo se ha tratado de trabajar de forma paramétrica, es decir que no se trabaja en función de valores, sino en función de variables. Con esto se busca dar generalidad al archivo tcl generado, en donde serán variables las longitudes y secciones de los elementos. Es una buena práctica definir todos estos parámetros al inicio de la codificación.

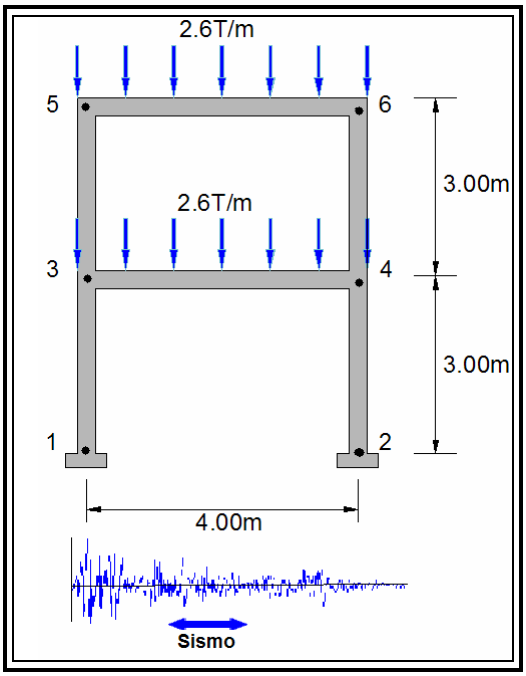

Figura 5.24: Acciones consideradas Ejemplo 2

Cabe destacar que en el código que se presenta a continuación se ha incluido un algoritmo desarrollado por la Dra. Silvia Mazoni de la Universidad de California, Berkeley, el cual es usado cuando el algoritmo definido por el

usuario para el análisis no alcanza convergencia y falla, dentro de un lazo se cambian algunos parámetros de análisis para alcanzar la convergencia.

```
# EJEMPLO 2
# ELEMENTOS VIGA, COLUMNA CON SECCIONES TIPO FIBRA
# UNIDADES T, m, seg

# INICIO
#=========================================================
wipe;  # borra de la memoria modelos anteriores
model BasicBuilder -ndm 2 -ndf 3;  # defino el modelo 2d y 3gdl por nudo
set dataDir Datos2;  # asigno el nombre Datos2 al directorio datadir
file mkdir $dataDir;   # defino mi archivo de salida Datos1
puts "MODELO DEFINIDO!"

# DEFINO GEOMETRÍA Y PARÁMETROS
#=========================================================
set LCol 3;              # longitud columna
set LViga 4;             # longitud viga
set HCol 0.3;                  # altura columna
set BCol 0.3;                  # ancho columna
set HViga 0.3;                 # altura viga
set BViga 0.25;                # base viga
set cover 0.03;                # recubrimiento
set fic 14;     # diámetro (mm) de varillas de acero columnas
set fiv 16;             # diámetro (mm) de varillas de acero vigas
set Asc [expr (3.1416*pow(0.001*$fic,2))/4];   # área varilla columna
set Asv [expr (3.1416*pow(0.001*$fiv,2))/4];   # área varilla viga
set H [expr 2*$LCol]; # altura de la estructura
set Wb 2.6;             # carga distribuida sobre la viga
set Peso 2*$Wb*$LViga;   # peso estructura (2Pisos*LongViga*Carga) (aprox)
set g 9.81;             # aceleración de la gravedad (m/seg2)
set PCol [expr $Peso/4]; # peso nodal en columnas
set Mass [expr $PCol/$g]; # masa nodal
set fy 42000;          # esfuerzo de fluencia del acero de refuerzo
set E 21000000.0;  # módulo de Young acero
set fc 2100;           # resistencia característica del hormigón
set xdamp 0.05;      # factor de amortiguamiento

# COORDENADAS NODALES:
#=========================================================
node 1 0 0;                  # NUDO X, Y
node 2 $LViga 0
node 3 0 $LCol
node 4 $LViga $LCol
node 5 0 $H
node 6 $LViga $H
```

```
# RESTRICCIONES EN APOYOS
#=========================================================
fix 1 1 1 1;                    # NUDO DX DY RZ
fix 2 1 1 1
puts "NUDOS CREADOS, RESTRINGIDOS!"

# MASAS NODALES:
#=========================================================
mass 3 $Mass  1.0e-8      1.0e-8;        # NUDO#, Mx My Mz, Mass=PESO/g
mass 4 $Mass  1.0e-8      1.0e-8
mass 5 $Mass  1.0e-8      1.0e-8
mass 6 $Mass  1.0e-8      1.0e-8
puts "MASAS NODALES  ASIGNADAS"

# DEFINO MATERIALES
#=========================================================
# CONCRETO DEL NÚCLEO (CONF)
# CONCRETO                  tag   f'c    ec0        f'cu        ecu
uniaxialMaterial Concrete01  1  -3300  -0.004   -2600     -0.018

# CONCRETO EXTERIOR (NO CONFINADO)
uniaxialMaterial Concrete01  2  -$fc   -0.002   0.0     -0.004

# ACERO DE REFUERZO
#                              tag  fy  E0   b
uniaxialMaterial Steel01  3  $fy $E 0.005

#PARÁMETROS AUXILIARES PARA DEFINIR SECCIÓN FIBRA
set y1 [expr $HCol/2.0]
set z1 [expr $BCol/2.0]

# CREO SECCIÓN FIBRA PARA COLUMNA

section Fiber 1 {

  # CREO LAS FIBRAS DE CONCRETO CONFINADO
   patch rect 1 10 1  [expr $cover-$z1]  [expr $cover-$y1]  [expr $z1-$cover]
[expr $y1-$cover]

   # CREO LAS FIBRAS DE CONCRETO NO CONFINADO (DER, IZQU,
ABAJO, ARRIBA)
   patch rect 2 10 1  [expr $z1-$cover] [expr -$y1] $z1  $y1
   patch rect 2 10 1  [expr -$z1] [expr -$y1] [expr $cover-$z1]  $y1
   patch rect 2  2 1  [expr $cover-$z1] [expr -$y1]  [expr $z1-$cover] [expr
$cover-$y1]
   patch rect 2  2 1  [expr $cover-$z1] [expr $y1-$cover] [expr $z1-$cover] $y1

   # CREO LAS FIBRAS DE REFUERZO (ARRIBA, MEDIO, ABAJO)
  layer straight 3 3 $Asc [expr $cover-$z1] [expr $y1-$cover] [expr $z1-$cover]
[expr $y1-$cover]
```

```
   layer straight 3 2 $Asc [expr $cover-$z1] 0.0  [expr $z1-$cover] 0.0
   layer straight 3 3 $Asc [expr $cover-$z1] [expr $cover-$y1] [expr $z1-$cover]
[expr $cover-$y1]

}
#PARÁMETROS ADICIONALES PARA DEFINIR SECCIÓN FIBRA VIGA
set y2 [expr $HViga/2.0]
set z2 [expr $BViga/2.0]

# CREO SECCIÓN FIBRA PARA VIGA

section Fiber 2 {

   # CREO LAS FIBRAS DE CONCRETO CONFINADO
    patch rect 1 10 1  [expr $cover-$z2] [expr $cover-$y2] [expr $z2-$cover] [expr
$y2-$cover]

   # CREO LAS FIBRAS DE CONCRETO NO CONFINADO (DER, IZQU,
ABAJO, ARRIBA)
    patch rect 2 10 1  [expr $z2-$cover] [expr -$y2] $z2 $y2
    patch rect 2 10 1  [expr -$z2] [expr -$y2] [expr $cover-$z2] $y2
    patch rect 2  2 1  [expr $cover-$z2] [expr -$y2]  [expr $z2-$cover] [expr
$cover-$y2]
    patch rect 2  2 1  [expr $cover-$z2] [expr $y2-$cover] [expr $z2-$cover] $y2

   # CREO LAS FIBRAS DE REFUERZO(ARRIBA, ABAJO)
  layer straight 3 2 $Asv [expr $cover-$z2] [expr $y2-$cover] [expr $z2-$cover]
[expr $y2-$cover]
  layer straight 3 2 $Asv [expr $cover-$z2] [expr $cover-$y2] [expr $z2-$cover]
[expr $cover-$y2]

}

puts "SECCIONES FIBRA CREADAS"

# DEFINO ELEMENTOS
#=========================================================
#                       tag
geomTransf Linear 1
set np 5;    # número de puntos de integración a lo largo del elemento

# CREO COLUMNAS Y VIGAS USANDO ELEMENTOS Beam-column
#                                    tag ndI ndJ nsecs secID transfTag
element nonlinearBeamColumn  1   1  3   $np    1       1
element nonlinearBeamColumn  2   2  4   $np    1       1
element nonlinearBeamColumn  3   3  5   $np    1       1
element nonlinearBeamColumn  4   4  6   $np    1       1

# CREO VIGAS USANDO ELEMENTOS Beam-column
geomTransf Linear 2
```

```
element nonlinearBeamColumn  5   3  4   $np   2      2
element nonlinearBeamColumn  6   5  6   $np   2      2
puts "ELEMENTOS CREADOS"

# DEFINO RECORDERS (ARCHIVOS DE SALIDA)
#=========================================================
recorder Node -file $dataDir/Dlibres.out -time -node 4 6 -dof 1 2 3 disp; #
DESPLAZAMIENTOS NUDOS LIBRES
recorder Node -file $dataDir/DBase.out -time -node 1 2 -dof 1 2 3 disp;      #
DESPLAZAMIENTOS NUDOS SOPORTE
recorder Node -file $dataDir/RBase.out -time -node 1 2 -dof 1 2 3 reaction;
        # REACCIONES EN SOPORTES
recorder Drift -file $dataDir/Deriv.out -time -iNode 2 4 -jNode 4 6 -dof 1  -
perpDirn 2 ;  # DERIVA LATERAL
recorder Element -file $dataDir/FCol.out -time -ele 1 2 globalForce;
        # FUERZAS EN COLUMNAS
recorder Element -file $dataDir/FViga.out -time -ele 3 globalForce;          #
FUERZAS EN VIGAS


# DEFINO GRAVEDAD
#=========================================================
pattern Plain 1 Linear {
  eleLoad -ele 5 6 -type -beamUniform -$Wb ; # carga distribuida sobre vigas
}
constraints Plain;     # forma en que se manejan las restricciones de nudo
numberer Plain;      # reenumera los dof para minimizar el ancho de banda
system BandGeneral; # como se resuelven las ecuaciones en el análisis
set Tol 1.0e-8;        # tolerancia para test de convergencia
test NormDispIncr $Tol 6 ; # determina si la convergencia ha sido alcanzada
algorithm Newton;   # usa el algoritmo de newton para la solución
set NpGrav 10;       # aplico la gravedad en 10 pasos
set DGrav [expr 1./$NpGrav]; # incrementos para la aplicación de carga
integrator LoadControl $DGrav;  # aplico 10% de carga gravitatoria por paso
analysis Static;       # defino el tipo de análisis (estático)
analyze $NpGrav;   # Indico OpenSEES los pasos de análisis que debe realizar
loadConst -time 0.0;  # Congela aplicación de cargas gravit y reinicia el tiempo
puts "ANÁLISIS GRAVITATORIO REALIZADO!"

# ANÁLISIS DINÁMICO (SISMO)
#=========================================================
# creamos patrón de carga
set accelSeries "Series -dt 0.02 -filePath centro.txt -factor 1"; # defino
acelerograma
pattern UniformExcitation 2 1 -accel $accelSeries; # defino como y cuando
aplico aceleración
rayleigh 0. 0. 0. [expr 2*$xdamp/pow([eigen 1],0.5)];  # asigno amortiguamiento
basado en el primer modo
```

```
# CREAMOS EL ANÁLISIS
#=========================================================
wipeAnalysis; # borra los parámetros de análisis antes definidos
constraints Plain;     # como son consideradas las condiciones de borde
numberer Plain;      # reenumera los dor para minimizar ancho de banda
system BandGeneral; # como se resuelven las ecuaciones en el análisis
set maxNumIter 30;          # máximo # de iteraciones que se realizarán
set printFlag 0;            # prueba de convergencia
set TestType NormDispIncr;       # tipo de prueba de convergencia
test $TestType $Tol $maxNumIter $printFlag;
set algorithmType Newton;
algorithm $algorithmType;          # usa el algoritmo de newton para la solución
set NewmarkGamma 0.5;          # integrador gamma algoritmo newton
set NewmarkBeta 0.25;          # integrador beta algoritmo newton
integrator Newmark $NewmarkGamma $NewmarkBeta; #Integrador
analysis Transient;          # defino el tipo de análisis dependiente de tiempo

# COLOCO PARÁMETROS PARA ANÁLISIS DE ACCIÓN SÍSMICA
set DtAnalysis  0.01; #paso de tiempo para análisis dinámico (recomend 0.5*dt)
set TmaxAnalysis    30;     # duración de acción dinámica (seg)
set Nsteps [expr int($TmaxAnalysis/$DtAnalysis)]; #número de pasos
set ok [analyze $Nsteps $DtAnalysis];    # se realiza el análisis; retorna ok=0 si
el análsis fue exitoso

# ALGORTIMO USADO EN CASO DE ANÁLISIS FALLIDO (Dra. Mazzoni)
if {$ok != 0} {      ;          # si el análisis no es exitoso.
        # Se cambia algunos parámetros de análisis para alcanzar la
convergencia
        # Proceso es más lento dentro de este lazo
        # Análisis controlado por tiempo
        set ok 0;
        set controlTime [getTime];
        while {$controlTime < $TmaxAnalysis && $ok == 0} {
                set ok [analyze 1 $DtAnalysis]
                set controlTime [getTime]
                set ok [analyze 1 $DtAnalysis]
                if {$ok != 0} {
                        puts "Trying Newton with Initial Tangent .."
                        test NormDispIncr   $Tol 1000  0
                        algorithm Newton -initial
                        set ok [analyze 1 $DtAnalysis]
                        test $TestType $Tol $maxNumIter  0
                        algorithm $algorithmType
                }
                if {$ok != 0} {
                        puts "Trying Broyden .."
                        algorithm Broyden 8
                        set ok [analyze 1 $DtAnalysis]
                        algorithm $algorithmType
                }
```

```
                if {$ok != 0} {
                        puts "Trying NewtonWithLineSearch .."
                        algorithm NewtonLineSearch .8
                        set ok [analyze 1 $DtAnalysis]
                        algorithm $algorithmType
                }
        }
};      # Finaliza si ok !0

puts "ANÁLISIS DINÁMICO REALIZADO: [getTime]"
```

5.6.3 Ejemplo 3

Se presenta a continuación una estructura espacial de un piso y un vano, con 6 GDL por nudo. La estructura está empotrada en su base, las columnas y vigas han sido definidas como elementos elásticos.

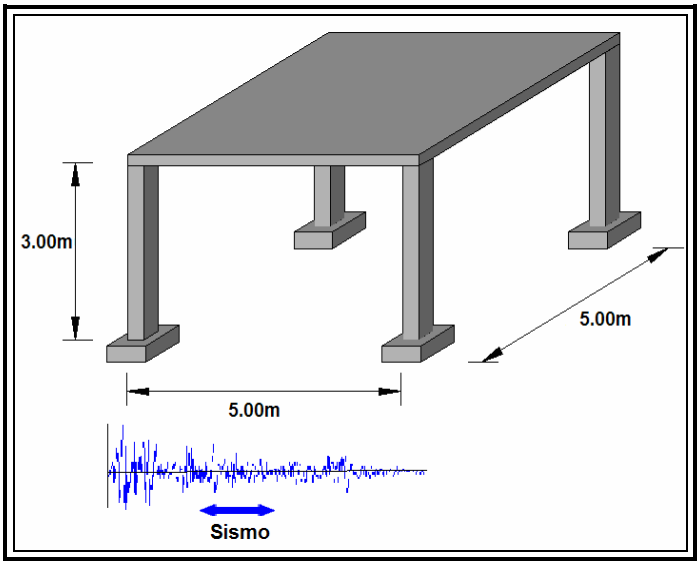

Figura 5.25: Estructura Ejemplo 3

La estructura es sometida a cargas gravitatorias y a la acción dinámica inducida por el sismo del Centro. Este modelo espacial requiere de algunas consideraciones que serán aclaradas con la presentación del archivo TCL. Las

columnas son de 30x30cm, las vigas de 30x20cm, se asume una losa de 20cm de espesor. Los resultado obtenidos con OpenSEES son comparados con los resultados obtenidos con el programa baserígidanew[2].

Como se verá en el archivo TCL, se han definido 6 GDL por nudo. Es necesario definir un diafragma rígido, se genera para el efecto el nudo 1121 (master) y se lo une con los nudos 121, 122, 221, 222 o nudos esclavos. Se concentra la masa en los nudos 121, 221, 122 y 222, y sobre los elementos columnas, vigas en sentido X y vigas en sentido Z (Girders) se aplican cargas uniformemente distribuidas.

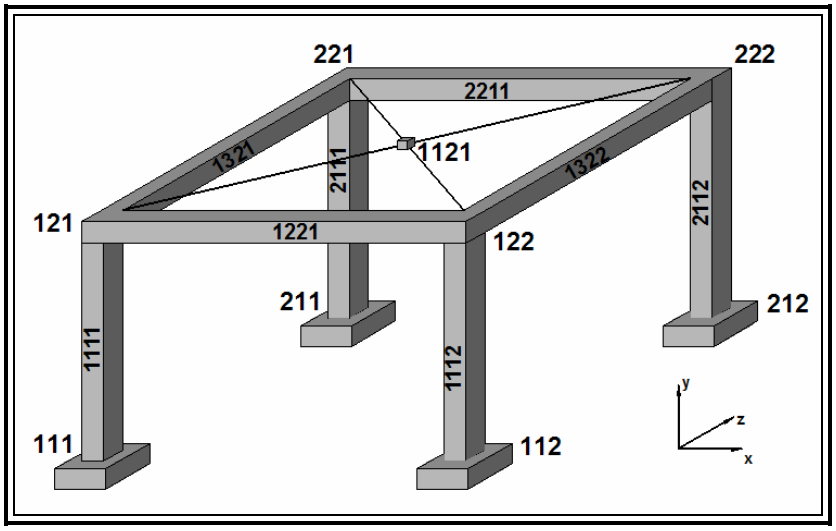

Figura 5.26: Nudos y elementos Ejemplo 3

En este ejemplo resulta conveniente hacer uso de algunas funciones que permiten lograr procedimientos más eficientes de análisis. La función que se utiliza es LibAnalysisDynamicParameters.tcl, la cual ha sido desarrollada por Silvia Mazzoni & Frank McKenna (2006), esta define los parámetros para el

[2] Aguiar Roberto, "Análisis Sísmico de Edificios", 2008.

análisis dinámico y puede ser invocada en un proceso mediante la instrucción **source.** Se hace hincapié en que esta función o cualquier otra que sea invocada en un archivo tcl debe estar guardada en la misma carpeta en la que está almacenado el archivo que invoca a la función.

```
# EJEMPLO 3
# ELEMENTOS VIGA, COLUMNA CON SECCIONES ELÁSTICAS
# UNIDADES T, m, seg

# INICIO
#=========================================================
wipe;  # borra de la memoria modelos anteriores
model BasicBuilder -ndm 3 -ndf 6;  # defino el modelo 3d y 6gdl por nudo
set dataDir Datos3;  # asigno el nombre Datos3 al directorio datadir
file mkdir $dataDir;   # defino mi archivo de salida Datos1
puts "MODELO DEFINIDO!"

# DEFINO LA GEOMETRÍA, SECCIONES Y PROP DE MATERIALES
#=========================================================
set LCol 3;    # longitud de columna (paralela eje y)
set LViga 5;  # longitud viga (paralela eje x)
set LGird 5;  # longitud viga (paralela eje z)
set HCol 0.30;        # altura columna
set BCol 0.30;        # base columna
set HViga 0.30;       # altura viga eje x
set BViga 0.20;       # base viga eje x
set HGird 0.30;       # altura viga eje z
set BGird 0.20;       # base viga eje z
set Ec 1800000;       # módulo Young del concreto
set nu 0.2;           # modulo de Poisson
set Gc [expr $Ec/(2./(1+$nu))];  # modulo rigidez por corte
set J 10000000;       # asigno gran rigidez torsional
set GammaConcreto 2.4;  # peso especifico hormigón armado
set g 9.8;       # Aceleración de la gravedad

# COORDENADAS NODALES
#=========================================================
# DETERMINO LA LOCALIZACIÓN DE LA INTERSECCIÓN DE EJES DE
VIGAS Y COLUMNAS
set X1 0.;
set X2 [expr $X1 + $LViga];
set Y1 0.;
set Y2 [expr $Y1 + $LCol];
set Z1 0.0;
set Z2 [expr $Z1 + $LGird];
```

```
node 111 $X1 $Y1 $Z1;    # PÓRTICO 1
node 112 $X2 $Y1 $Z1;
node 121 $X1 $Y2 $Z1;
node 122 $X2 $Y2 $Z1;
node 211 $X1 $Y1 $Z2;    # PÓRTICO 2
node 212 $X2 $Y1 $Z2;
node 221 $X1 $Y2 $Z2;
node 222 $X2 $Y2 $Z2;

#DIAFRAGMA RÍGIDO DE PISO
set RigidDiaphragm ON ;  # opciones: on, off. Especificar
set Xa [expr ($X2)/2];       # coordenadas para el diafragma rígido
set Za [expr ($Z2)/2];

# NUDOS PARA DIAFRAGMA RÍGIDO, EN EL CENTRO DE CADA
DIAFRAGMA
#============================================================
set RigidDiaphragm ON ;  # esto comunica al análisis los parámetros que
estaré usando para diafragma rígido
node 1121 $Xa $Y2 $Za;           # nudo master -- piso 2, vano 1, pórtico 1-2


# RESTRICCIONES PARA NUDOS MASTER DE DIAFRAGMAS RÍGIDOS
fix 1121 0  1  0  1  0  1
puts "NUDOS CREADOS Y RESTRINGIDOS"

# DEFINO DIAFRAGMA RÍGIDO, GDL 2 ES NORMAL AL PISO
#============================================================
set perpDirn 2;
rigidDiaphragm $perpDirn 1121 121 122 221 222;     # NIVEL 2

# RESTRICCIONES EN LOS APOYOS
fix 111 1  1  1  1  1  1        # nudos de apoyo empotrados
fix 112 1  1  1  1  1  1
fix 211 1  1  1  1  1  1
fix 212 1  1  1  1  1  1
puts "DIAFRAGMA CREADO"


# DEFINO ETIQUETAS DE SECCIONES:
set ColSecTag 1
set BeamSecTag 2
set GirdSecTag 3
set SecTagTorsion 70

# PROPIEDADES DE LA SECCIÓN COLUMNA:
set AgCol [expr $HCol*$BCol];    # área columna
set IzCol [expr 1./12*$BCol*pow($HCol,3)]; # inercia con respecto al eje local z
set IyCol [expr 1./12*$HCol*pow($BCol,3)]; # inercia con respecto al eje local z
```

```
# SECCIONES VIGA:
set AgViga [expr $HViga*$BViga];  # área vigas
set IzViga [expr 1./12*$BViga*pow($HViga,3)]; # inercia respecto a eje local z
set IyViga [expr 1./12*$HViga*pow($BViga,3)]; # inercia respecto a eje local y


# SECCIONES VIGA Z:
set AgGird [expr $HGird*$BGird];          # área vigas
set IzGird [expr 1./12*$BGird*pow($HGird,3)]; # inercia respecto a eje local z
set IyGird [expr 1./12*$HGird*pow($BGird,3)]; # inercia respecto a eje local y

section Elastic $ColSecTag $Ec $AgCol $IzCol $IyCol $Gc $J
section Elastic $BeamSecTag $Ec $AgViga $IzViga $IyViga $Gc $J
section Elastic $GirdSecTag $Ec $AgGird $IzGird $IyGird $Gc $J
puts "SECCIONES CREADAS"

# DEFINO LOS ELEMENTOS
#============================================================
# DEFINO TRANSFORMACIÓN GEOMÉTRICA DE ELEMENTO
set IDColTransf 1; # todas las columnas
set IDBeamTransf 2; # vigas en x
set IDGirdTransf 3; # vigas en z
set ColTransfType Linear ;
geomTransf $ColTransfType $IDColTransf  0 0 1
geomTransf Linear $IDBeamTransf 0 0 1
geomTransf Linear $IDGirdTransf 1 0 0

# DEFINO ELEMENTOS COLUMNAS Y VIGAS
# PÓRTICO 1
# Columnas
element elasticBeamColumn 1111 111 121 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;
element elasticBeamColumn 1112 112 122 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;

# Vigas paralelas a Y
element elasticBeamColumn 1221 121 122 $AgViga $Ec $Gc  $J  $IyViga
$IzViga $IDBeamTransf;;

# PÓRTICO 2
# Columnas
element elasticBeamColumn 2111 211 221 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;
element elasticBeamColumn 2112 212 222 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;

#Vigas
element elasticBeamColumn 2221 221 222 $AgViga $Ec $Gc  $J  $IyViga
$IzViga $IDBeamTransf;
```

```
#Vigas (eje Z) Conectan Pórticos
# Póticos 1-2
element elasticBeamColumn  1321 121 221 $AgGird $Ec $Gc  $J  $IyGird
$IzGird $IDGirdTransf;
element elasticBeamColumn  1322 122 222 $AgGird $Ec $Gc  $J  $IyGird
$IzGird $IDGirdTransf;
puts "ELEMENTOS CREADOS"

#DEFINO CARGAS GRAVITARIAS, PESO Y MASAS
#=========================================================
#DEFINICIÓN DE LAS CARGAS
set QdlCol [expr $GammaConcreto*$HCol*$BCol]; # peso por longitud columna
set QViga [expr $GammaConcreto*$HViga*$BViga];  # peso por longitud viga
set QGird [expr $GammaConcreto*$HGird*$BGird]; # peso por longitud girders
set TLosa 0.20;  # espesor supuesto para losa
set LLosa [expr $LGird/2]; # losa se extiende una distancia de $LGird/2
set Qlosa [expr $GammaConcreto*$TLosa*$LLosa];
set QdlViga [expr $Qlosa + $QViga];      # carga muerta distribuida en la viga
set QdlGird $QGird;                  # carga muerta distribuida en viga (eje z)
set PesoCol [expr $QdlCol*$LCol];                # peso total columna
set PesoViga [expr $QdlViga*$LViga];             # peso total viga
set PesoGird [expr $QdlGird*$LGird];             # peso total viga (eje z)

# ASIGNO MASAS EN LOS NUDOS
# CADA NUDO TOMA  1/2 DE LA MASA DE CADA ELEMENTO DEL
PÓRTICO
set Masa  [expr ($PesoCol/2 + $PesoViga/2+$PesoGird/2)/$g];
set Mcero 1.e-6;
# PÓRTICO 1
mass 121 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;
mass 122 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;

# PÓRTICO 2
mass 221 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;
mass 222 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;

set PesoPiso1 [expr 4*$PesoCol + 2*$PesoGird + 2*$PesoViga];
set PesoTotal [expr 1* $PesoPiso1];      # peso total de la estructura
set MasaTotal [expr $PesoTotal/$g];      # masa total del edificio
puts "CARGAS DEFINIDAS Y MASAS ASIGNADAS"

# DEFINO RECORDERS
#=========================================================
#recorder Node -file $dataDir/DEMP.out -time -node 111 112 211 212 -dof 1 2 3
disp;            #desplazamiento nudos de apoyo (debe ser cero)
recorder Node -file $dataDir/DLibres.out -time -node 121 122 221 222   -dof 1
disp;            #desplazamientos nudos libres para GDL 1
recorder Node -file $dataDir/RBase.out -time -node 111 112 211 212   -dof 1 2
3 reaction;              #reacciones en los apoyos
```

```
recorder Drift -file $dataDir/Deriv.out -time -iNode 112 212 -jNode 122 222 -dof
1 -perpDirn 2;          # deriva de piso
recorder Element -file $dataDir/Fel1.out -time -ele 1111 localForce; # fuerzas en
elementos coordenadas locales

# ANÁLISIS GRAVITATORIO
#=========================================================
# Defino cargas gravitatorias aplicadas en los elementos
pattern Plain 101 Linear {
# Pórtico 1
# Columnas
        eleLoad -ele 1111 -type -beamUniform 0. 0. -$QdlCol;
        eleLoad -ele 1112 -type -beamUniform 0. 0. -$QdlCol

# Vigas
        eleLoad -ele 1221 -type -beamUniform -$QdlViga 0.;

# Pórtico 2
# Columnas
        eleLoad -ele 2111 -type -beamUniform 0. 0. -$QdlCol;                1-2
        eleLoad -ele 2112 -type -beamUniform 0. 0. -$QdlCol

# Vigas
        eleLoad -ele 2221 -type -beamUniform -$QdlViga 0.;

# Vigas (eje z)
# Frame 1-2
        eleLoad -ele 1321 -type -beamUniform $QdlGird 0.;
        eleLoad -ele 1322 -type -beamUniform $QdlGird 0.;

}

set Tol 1.0e-8;       # Tolerancia para la prueba de convergencia
variable constraintsTypeGravity Plain;          # default;
if {  [info exists RigidDiaphragm] == 1} {
        if {$RigidDiaphragm=="ON"} {
        variable constraintsTypeGravity Lagrange; #  large model: try
Transformation
        };      # Si rigid diaphragm está “on”
};      # Si existe diafragma rígido

constraints $constraintsTypeGravity;  # forma en que se manejan las
restricciones de nudo
numberer RCM;     # reenumera los GDL para minimizar el ancho de banda
system BandGeneral ;      # como se resuelven las ecuaciones en el análisis
test EnergyIncr $Tol 6 ;     # determina si la convergencia ha sido alcanzada
algorithm Newton;          # usa algoritmo de newton para la solución
set NpGrav 10;                  # aplico carga gravitatoria en 10 intervalos
set DGrav [expr 1./$NpGrav];      # incrementos para la aplicación de la carga;
integrator LoadControl $DGrav;   # determino el paso de tiempo para el análisis
```

```
analysis Static;                # defino tipo de análisis (estático)
analyze $NpGrav;                # aplico cargas gravitatorias
loadConst -time 0.0; # mantiene constante las cargas gravit y reinicia el tiempo
set Tol 1.0e-6;  # reduce la tolerancia después de aplicar  las cargas gravitat
puts "ANÁLISIS GRAVITATORIO REALIZADO"

# ACCIÓN SÍSMICA UNIFORME
#=========================================================
# COLOCAR PARÁMETROS PARA ANÁLISIS SÍSMICO
wipeAnalysis; # Borra los objetos que definen el tipo de análisis
set DtAnalysis        [expr 0.02];   # paso para análisis sísmico
set TmaxAnalysis    [expr 40];      # duración del sismo (max 50seg)

# ASIGNO AMORTIGUAMIENTO RAYLEIGH
set xDamp 0.05;     # factor de amortiguamiento
set MpropSwitch 1.0;
set KcurrSwitch 0.0;
set KcommSwitch 0.0;
set KinitSwitch 0.0;
set nEigenI 1;                  # modo1
set nEigenJ 2;
set lambdaN [eigen [expr $nEigenJ]];
set lambdaI [lindex $lambdaN [expr $nEigenI-1]];       # eigenvalor modo i
set lambdaJ [lindex $lambdaN [expr $nEigenJ-1]];      # eigenvalor modo j
set omegaI [expr pow($lambdaI,0.5)];
set omegaJ [expr pow($lambdaJ,0.5)];
set alphaM [expr
$MpropSwitch*$xDamp*(2*$omegaI*$omegaJ)/($omegaI+$omegaJ)];
set betaKcurr [expr $KcurrSwitch*2.*$xDamp/($omegaI+$omegaJset
betaKcomm [expr $KcommSwitch*2.*$xDamp/($omegaI+$omegaJ)];
set betaKinit [expr $KinitSwitch*2.*$xDamp/($omegaI+$omegaJ)];
rayleigh $alphaM $betaKcurr $betaKinit $betaKcomm;

set IDloadTag 400;  # etiqueta para patrón de carga
set dt 0.02;            # paso con el cual se registro sismo
set GMdirection 1;   # dirección del sismo (GDL correspondiente)
set AccelSeries "Series -dt $dt -filePath centro.txt -factor  1"; # defino sismo y
factor de escala (En este caso sismo Centro)
pattern UniformExcitation  $IDloadTag  $GMdirection -accel  $AccelSeries; #
creo acción uniforme

# Parámetros para el análisis
source LibAnalysisDynamicParameters.tcl;  #Invoco a la función externa  que
define los parámetros de análisis
set Nsteps [expr int($TmaxAnalysis/$DtAnalysis)]; #número pasos de análisis
set ok [analyze $Nsteps $DtAnalysis];    # realizo análisis; regreso ok=0 si el
análisis es exitoso

#procedimiento en caso de análisis no exitoso
```

```
if {$ok != 0} {      ;                                    # análisis fallido

        # Se cambia algunos parámetros para alcanzar convergencia
        # El proceso es más lento dentro de este lazo
        # Análisis controlado por tiempo
        set ok 0;
        set controlTime [getTime];
        while {$controlTime < $TmaxAnalysis && $ok == 0} {
                set controlTime [getTime]
                set ok [analyze 1 $DtAnalysis]
                if {$ok != 0} {
                puts "Trying Newton with Initial Tangent .."
                test NormDispIncr   $Tol 1000  0
                algorithm Newton -initial
                set ok [analyze 1 $DtAnalysis]
                test $testTypeDynamic $TolDynamic $maxNumIterDynamic  0
                        algorithm $algorithmTypeDynamic
                }
                if {$ok != 0} {
                        puts "Trying Broyden .."
                        algorithm Broyden 8
                        set ok [analyze 1 $DtAnalysis]
                        algorithm $algorithmTypeDynamic
                }
                if {$ok != 0} {
                        puts "Trying NewtonWithLineSearch .."
                        algorithm NewtonLineSearch .8
                        set ok [analyze 1 $DtAnalysis]
                        algorithm $algorithmTypeDynamic
                }
        }
};      # end if ok !0

puts "ANÁLISIS DINÁMICO REALIZADO. End Time: [getTime]"
```

Es importante que el usuario se familiarice ampliamente con los conceptos que se utilizan en OpenSees, caso contrario tratar de comprender o interpretar las líneas de códigos puede resultar una tarea bastante dificultosa.

A continuación se presenta una comparación entre los resultados obtenidos para la estructura analizada, usando OpenSees y el programa Baserigidanew. Para que se visualice de mejor manera los resultados, solamente se presenta

la historia de desplazamientos en el Centro de Masa (CM) para un intervalo de tiempo de 5 segundos. Se puede ver bastante similitud en los resultados obtenidos con el uso de OpenSees y con el programa Baserigidanew.

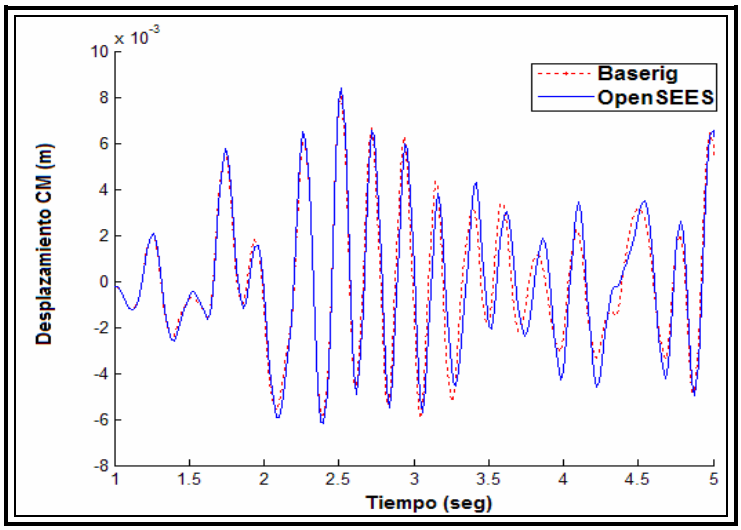

Figura 5.27: Comparación de resultados Baserigidanew y OpenSees

No obstante hay que reconocer que OpenSees provee al usuario mayor cantidad de herramientas, pueden desarrollarse modelos más complejos, pueden ser monitoreados más parámetros, y una vez almacenada esa información, puede ser editada con mucha facilidad. Por otra parte la gran cantidad de materiales, de elementos y algoritmos de cálculo, hacen de OpenSees una herramienta muy potente para el análisis de estructuras.

5.6.4 Ejemplo 4

Como introducción a los modelos que incluyen aislamiento base, se presenta a continuación una columna soportada sobre un aislador. La columna tiene dimensiones de 0.5 x 0.5 m y se modelará como elemento elástico. El aislador

se modelará a través de elemento zerolength (longitud cero), al cual se le asignará un material con comportamiento lineal.

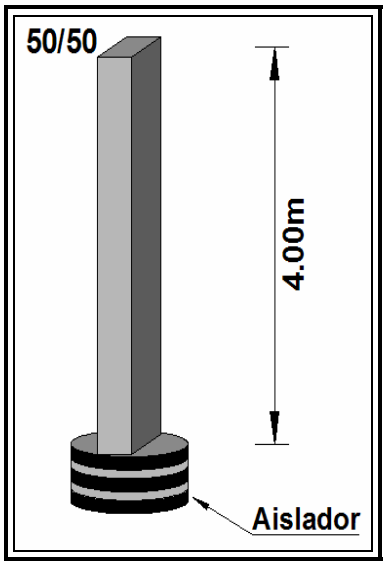

Figura 5.28: Columna de Ejemplo 4

La columna estará sometida a la acción de cargas gravitacionales y una carga puntual aplicada en el nudo superior. La acción dinámica será inducida por el sismo del Centro mayorado por un factor igual a 2.

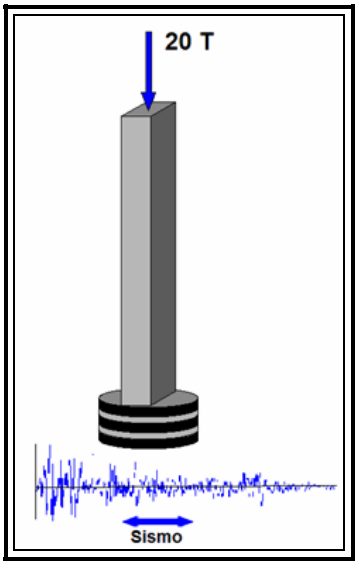

Figura 5.29: Acciones consideradas Ejemplo 4

```
# MODELO 3D, COLUMNA CON AISLADOR, ELEMENTO ELÁSTICO
# UNIDADES T, m, seg
# INICIO
#=========================================================
wipe;          # borra de la memoria modelos anteriores
model BasicBuilder -ndm 3 -ndf 6; # defino el modelo 3d y 6gdl por nudo
set dataDir DatosColAisl;   # asigno el nombre DatosAislad al directorio dataDir
file mkdir $dataDir;                # Defino mi archivo de salida
puts "MODELO DEFINIDO"

# DEFINO LA GEOMETRÍA, SECCIONES Y PROP DE MATERIALES
#=========================================================
# DEFINO PARÁMETROS GEOMÉTRICOS DE LA ESTRUCTURA
set LCol 4;            # altura de columna (paralela eje y)
set Peso 20;           # Peso superestructura
set g 9.81;            # Aceleración de la gravedad
set HCol 0.30;         # altura columna
set BCol 0.30;         # base columna
set Ec 1800000;        # módulo young del concreto
set nu 0.2;            # modulo de Poisson
set Gc [expr $Ec/(2./(1+$nu))];  # modulo rigidez por corte
set J 10000000;              # asigno gran rigidez torsional
set GammaConcreto 2.4;  # peso especifico horm armado

# PROPIEDADES DE LA SECCIÓN COLUMNA:
set AgCol [expr $HCol*$BCol];                    # área columna
set IzCol [expr 1./12*$BCol*pow($HCol,3)];     # inercia respecto al eje local z
set IyCol [expr 1./12*$HCol*pow($BCol,3)];     # inercia respecto al eje loal y
puts "SECCIONES INGRESADAS Y PROPIEDADES CALCULADAS"

# DEFINO COORDENADAS NODALES
#=========================================================
# determino la localización de la intersección de ejes de vigas y columnas
set X1 0.;
set Y1 0.;
set Y2 [expr $Y1 + $LCol];
set Z1 0.0;
node 111 $X1 $Y1 $Z1;
node 121 $X1 $Y2 $Z1;

# DEFINO NUDOS PARA AISLADORES
node 101 $X1 $Y1 $Z1;

# BOUNDARY CONDITIONS
fix 101 1  1  1  1  1  1;       # para aisladores
fix 111 0  1  1  1  1  1;       # pie de columnas

puts "NUDOS CREADOS Y RESTRINGIDOS"
# DEFINO ETIQUETAS DE SECCIONES:
set ColSecTag 1
```

```
set SecTagTorsion 70

# DEFINO TIPO DE SECCIONES ELÁSTICAS
section Elastic $ColSecTag $Ec $AgCol $IzCol $IyCol $Gc $J

#PARA LOS AISLADORES
uniaxialMaterial Elastic 4 20 0.1
uniaxialMaterial Elastic 5 21000000 #Material gran rigidez vertical
puts "SECCIONES ELASTICAS CREADAS"

# DEFINO LOS ELEMENTOS
#=============================================================
# defino transformación geométrica de elemento
set IDColTransf 1;
set ColTransfType Linear ;
geomTransf $ColTransfType $IDColTransf  0 0 1 ;

# CREO ELEMENTOS COLUMNAS
# columna
element elasticBeamColumn 1111 111 121 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;
# DEFINO LOS AISLADORES
#                  tag ndI ndJ nsecs secID transfTag
element zeroLength  1101     101  111  -mat 4  5  -dir 1  2
puts "ELEMENTOS CREADOS"

#DEFINO CARGAS GRAVITARIAS, PESO Y MASAS
#=============================================================
#DEFINICIÓN DE LAS CARGAS
set QdlCol [expr $GammaConcreto*$HCol*$BCol]; # peso por longitud columna
set PesoCol [expr $QdlCol*$LCol];        # peso total columna


# ASIGNO MASAS EN LOS NUDOS
set Masa  [expr ($PesoCol/$g)];
set Mcero 1.e-6;
mass 121 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;
puts "CARGAS DEFINIDAS Y MASAS ASIGNADAS"

# DEFINO RECORDERS
#=============================================================
recorder Node -file $dataDir/DAisl.out -time -node 111 -dof 1 2 3 disp;
        #DESPLA CABEZA AISL
recorder Node -file $dataDir/DLibres.out -time -node 121 -dof 1  disp;
        #DESPLA NUDO LIBRE
recorder Node -file $dataDir/RBase.out -time -node 111 101  -dof 1 2 3 reaction;
                #REACCIONES EN LOS APOYOS
recorder Drift -file $dataDir/Deriv.out -time -iNode 111 -jNode 121 -dof 1 -
perpDirn 2;    # DERIVA DE PISO
```

```
recorder Element -file $dataDir/Fel1.out -time -ele 1111 localForce;        # FUERZAS EN ELEMENTOS COORDENAS LOCALES
puts "RECORDERS CREADOS"

# ANÁLISIS GRAVITATORIO
#=========================================================
# cargas gravitatorias aplicadas en elemento
pattern Plain 101 Linear {
# columna
        eleLoad -ele 1111 -type -beamUniform 0. 0. -$QdlCol; # Peso Distribuido
        load 121 0 -$Peso 0 0 0 0 ;         # Carga axial aplicada en el nudo 121

}
set Tol 1.0e-8;         # Tolerancia para la prueba de convergencia
set constraintsTypeGravity Plain;          # default;
constraints $constraintsTypeGravity; # condiciones de borde
numberer RCM;      # reenumera los dof para minimizar el ancho de banda
system BandGeneral ;# como se  resuelven el sistema de ecuaciones
test EnergyIncr $Tol 6 ;      # determina si la convergencia ha sido alcanzada
algorithm Newton;             # usa algortimo de newton para la solución
set NpGrav 10;                          # aplico carg gravit en 10 intervalos
set DGrav [expr 1./$NpGrav];       # incrementos para la aplicación de la carga;
integrator LoadControl $DGrav;   # determino el paso de tiempo para el análisis
analysis Static;                # defino tipo de análisis (estático)
analyze $NpGrav;                # aplico cargas gravitatorias
loadConst -time 0.0;  # mantiene constante las cargas gravit y reinicia el tiempo
set Tol 1.0e-6;                 # reduce la tolerancia después de las cargas gravit
puts "ANALISIS GRAVITATORIO REALIZADO"

# ACCIÓN SÍSMICA UNIFORME
#=========================================================
# ASIGNO AMORTIGUAMIENTO RAYLEIGH
wipeAnalysis; # Borra los objetos que definen el tipo de análisis
set xDamp 0.05;      # factor de amortiguamiento
set lambda [eigen 1]; # eigenvalor modo 1
set omega [expr pow($lambda,0.5)];
set betaKcomm [expr 2.*$xDamp/($omega)];   #betaKcomm*KlastCommitt
rayleigh 0.0 0.0 0.0 $betaKcomm;         # Amortiguamiento RAYLEIGH

# COLOCAR PARÁMETROS PARA ANÁLISIS SÍSMICO
set DtAnalysis        [expr 0.02]; # paso para análisis sísmico
set TmaxAnalysis    [expr 40];      # duración del sismo (max 50seg)
set IDloadTag 400;  # etiqueta para patrón de carga
set dt 0.02;              # paso del sismo
set GMdirection 1;   # dirección del sismo
set AccelSeries "Series -dt $dt -filePath centro.txt -factor  2";  # defino sismo y factor de escala
pattern UniformExcitation  $IDloadTag  $GMdirection -accel  $AccelSeries  ; # creo acción uniforme
# Defino parámetros del análisis
```

```
constraints Transformation; # Forma en que se manejan  restricciones de nudo
numberer Plain ; # Forma de numerar nudos para minimizar ancho de banda
system SparseGeneral -piv;  # Algoritmo de solución de sistema de ecuaciones
set maxNumIter 6 ;  # Maximo numero de iteraciones para alcanzar tolerancia
set printFlag 0; ;      # Para que OpenSees notifique sobre falta de convergencia
set TestType EnergyIncr ; # Tipo de test para prueba de convergencia
set Tol 1.e-6 ;         # Tolerancia en la solución
test $TestType $Tol $maxNumIter $printFlag
set algorithmType ModifiedNewton ;      # Algoritmo de solución paso a paso
algorithm $algorithmType
set NewmarkGamma 0.5; # Parámetro gama para el método de Newmark
set NewmarkBeta 0.25; # Parámetro beta para el método de Newmark
integrator Newmark $NewmarkGamma $NewmarkBeta
analysis Transient ;# Tipo de análisis
set Nsteps [expr int($TmaxAnalysis/$DtAnalysis)];
set ok [analyze $Nsteps $DtAnalysis];    # Realizo análisis; regreso ok=0 si el
análisis es exitoso

#PROCEDIMIENTO EN CASO DE ANÁLISIS NO EXITOSO

if {$ok != 0} {      ;                                  # análisis fallido

        # Se cambia algunos parámetros para alcanzar convergencia
        # El proceso es más lento dentro del lazo
        #  Análisis controlado por tiempo
        set ok 0;
        set controlTime [getTime];
        while {$controlTime < $TmaxAnalysis && $ok == 0} {
                set controlTime [getTime]
                set ok [analyze 1 $DtAnalysis]
                if {$ok != 0} {
                        puts "Trying Newton with Initial Tangent .."
                        test NormDispIncr   $Tol 1000  0
                algorithm Newton -initial
                set ok [analyze 1 $DtAnalysis]
                test $testTypeDynamic $TolDynamic $maxNumIterDynamic  0
                        algorithm $algorithmTypeDynamic
                }
                if {$ok != 0} {
                        puts "Trying Broyden .."
                        algorithm Broyden 8
                        set ok [analyze 1 $DtAnalysis]
                        algorithm $algorithmTypeDynamic
                }
                if {$ok != 0} {
                        puts "Trying NewtonWithLineSearch .."
                        algorithm NewtonLineSearch .8
                        set ok [analyze 1 $DtAnalysis]
                        algorithm $algorithmTypeDynamic
                }
```

```
        }
};      # end if ok !0

puts "ANÁLISIS DINÁMICO REALIZADO: [getTime]"
```

Como se mencionó anteriormente, los recorders pueden ser manipulados fácilmente para la creación de figuras y gráficas en las cuales se visualice de mejor manera la respuesta de la estructura analizada. Por ejemplo, en la figura 5.30, se presenta el desplazamiento máximo en la cabeza del aislador (Punto A) que es igual al desplazamiento en el pie de la columna, y se muestra el desplazamiento máximo en la cabeza de la columna (Punto B), los cuales tienen lugar en el mismo instante de tiempo. En el eje de las ordenadas se presenta la altura del elemento, con fines didácticos se ha asumido arbitrariamente una altura de aislador. Esta figura muestra claramente el gran desplazamiento que tiene lugar en el aislador versus el bajo desplazamiento relativo en la cabeza de columna.

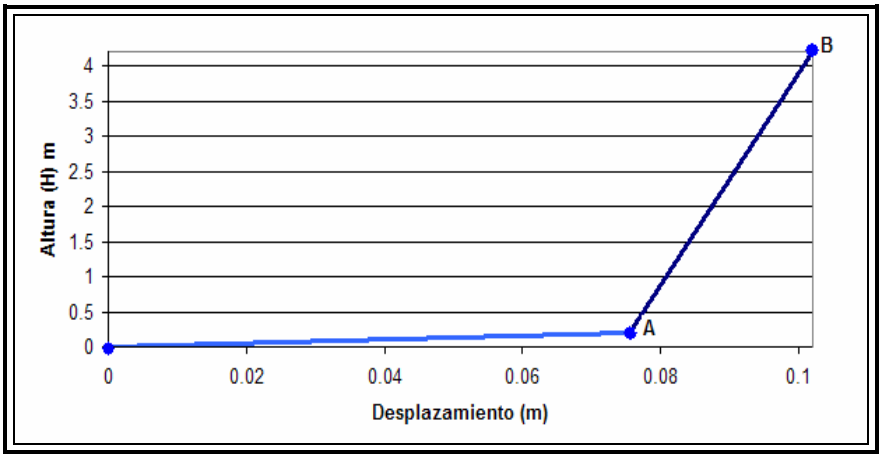

Figura 5.30: Desplazamientos del sistema de aislamiento y la superestructura

5.6.5 Ejemplo 5

Se presenta a continuación una estructura espacial de un piso y un vano, con 6 GDL por nudo. La estructura posee asilamiento sísmico en su base, las columnas y vigas han sido definidas como elementos elásticos.

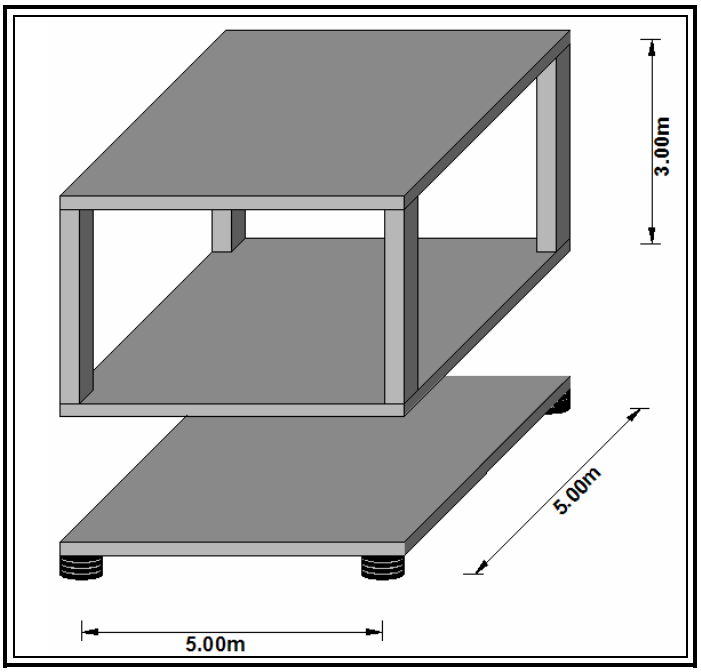

Figura 5.31: Estructura con aislamiento sísmico de base Ejemplo 5

La estructura será sometida a cargas gravitatorias y a la acción dinámica del sismo del Centro. Las columnas son de 30x30cm, las vigas de 30x20cm, se asume una losa de 20cm de espesor para el cálculo de la carga muerta. Es importante mencionar que los aisladores se modelan como elementos con comportamiento lineal, según lo manifestado en el apartado 5.5.

```
# EJEMPLO 5
# UN PISO UN VANO, ELEMENTOS ELÁSTICO AISLADORES
# UNIDADES T, m, seg
# INICIO
#=========================================================
wipe;                   # borra de la memoria modelos anteriores
model BasicBuilder -ndm 3 -ndf 6;   # defino el modelo 3d y 6gdl por nudo
set dataDir DatosAislad;  # asigno el nombre DatosAislad al directorio dataDir
file mkdir $dataDir;                # Defino mi archivo de salida

puts "MODELO DEFINIDO"

# DEFINO LA GEOMETRÍA, SECCIONES Y PROP MATERIALES
#=========================================================
set LCol 3;    # altura de columna (paralela eje y)
set LViga 5;  # longitud viga (paralela eje x)
set LGird 5;  # longitud viga (paralela eje z)
set HCol 0.30;        # altura columna
set BCol 0.30;        # base columna
set HViga 0.30;       # altura viga eje x
set BViga 0.20;       # base viga eje x
set HGird 0.30;       # altura viga eje z
set BGird 0.20;       # base viga eje z
set Ec 1800000;       # módulo young del concreto
set nu 0.2;           # modulo de Poisson
set Gc [expr $Ec/(2./(1+$nu))];  # modulo rigidez por corte
set J 10000000;       # asigno gran rigidez torsional
set GammaConcreto 2.4;  # peso especifico horm armado
set g 9.8;      #Aceleración de la gravedad

# DEFINO COORDENADAS NODALES
#=========================================================
set X1 0.;
set X2 [expr $X1 + $LViga];
set Y1 0.;
set Y2 [expr $Y1 + $LCol];
set Z1 0.0;
set Z2 [expr $Z1 + $LGird];

node 111 $X1 $Y1 $Z1;    # PÓRTICO 1
node 112 $X2 $Y1 $Z1;
node 121 $X1 $Y2 $Z1;
node 122 $X2 $Y2 $Z1;
node 211 $X1 $Y1 $Z2;    # PÓRTICO 2
node 212 $X2 $Y1 $Z2;
node 221 $X1 $Y2 $Z2;
node 222 $X2 $Y2 $Z2;
```

```
#Defino nudos para los aisladores
node 101 $X1 $Y1 $Z1;    # PÓRTICO 1
node 102 $X2 $Y1 $Z1;
node 201 $X1 $Y1 $Z2;    # PÓRTICO 2
node 202 $X2 $Y1 $Z2;


#DIAFRAGMA RÍGIDO DE PISO
set RigidDiaphragm ON ;  # opciones: on, off. Especificar esto antes
set Xa [expr ($X2)/2];       # coordenas para el diafragma rígido
set Za [expr ($Z2)/2];

# CREO NUDO MASTER EN EL CENTRO DEL DIAFRAGAMA
#=========================================================
set RigidDiaphragm ON ;  # comunica al análisis que estaré usando diafragma
node 1121 $Xa $Y2 $Za;          # nudo master -- piso 2, vano 1, PÓRTICO 1-2

# RESTRICCIONES PARA NUDOS MASTER DE DIAFRAGMAS RÍGIDOS
fix 1121 0  1  0  1  0  1

# BOUNDARY CONDITIONS
fix 101 1  1  1  1  1  1       # para base aisladores
fix 102 1  1  1  1  1  1
fix 201 1  1  1  1  1  1
fix 202 1  1  1  1  1  1
fix 111 0  1  1  1  1  1       # pie de columna (cabeza aislador)
fix 112 0  1  1  1  1  1
fix 211 0  1  1  1  1  1
fix 212 0  1  1  1  1  1
puts "NUDOS CREADOS Y RESTRINGIDOS"

# DEFINO DIAFRAGMA RÍGIDO, GDL 2 ES NORMAL AL PISO
#=========================================================
set perpDirn 2;
rigidDiaphragm $perpDirn 1121 121 122 221 222;     # NIVEL 2
puts "DIAFRAGMA CREADO"


# DEFINO ETIQUETAS DE SECCIONES:
set ColSecTag 1
set BeamSecTag 2
set GirdSecTag 3
set SecTagTorsion 70

# PROPIEDADES DE LA SECCIÓN COLUMNA:
set AgCol [expr $HCol*$BCol];    # área columna
set IzCol [expr 1./12*$BCol*pow($HCol,3)]; # inercia con respecto al eje local z
set IyCol [expr 1./12*$HCol*pow($BCol,3)]; # inercia con respeco al eje loal z
```

```
# SECCIONES VIGA:
set AgViga [expr $HViga*$BViga];  # área vigas
set IzViga [expr 1./12*$BViga*pow($HViga,3)];# inercia respecto a eje local z
set IyViga [expr 1./12*$HViga*pow($BViga,3)];# inercia respecto a eje local y

# SECCIONES VIGA Z:
set AgGird [expr $HGird*$BGird];          # área vigas
set IzGird [expr 1./12*$BGird*pow($HGird,3)];  # inercia respecto a eje local z
set IyGird [expr 1./12*$HGird*pow($BGird,3)];  # inercia respecto a eje local y

section Elastic $ColSecTag $Ec $AgCol $IzCol $IyCol $Gc $J
section Elastic $BeamSecTag $Ec $AgViga $IzViga $IyViga $Gc $J
section Elastic $GirdSecTag $Ec $AgGird $IzGird $IyGird $Gc $J

#PARA LOS AISLADORES
uniaxialMaterial Elastic 4 20 0.1
uniaxialMaterial Elastic 5 21000000
puts "SECCIONES CREADAS"

# DEFINO LOS ELEMENTOS
#=========================================================
# Defino transformación geométrica de elemento
set IDColTransf 1;  # todas las columnas
set IDBeamTransf 2;  # vigas en x
set IDGirdTransf 3;  # vigas en z
set ColTransfType Linear ;
geomTransf $ColTransfType $IDColTransf  0 0 1 ;
geomTransf Linear $IDBeamTransf 0 0 1
geomTransf Linear $IDGirdTransf 1 0 0

# DEFINO ELEMENTOS COLUMNAS Y VIGAS
# PÓRTICO 1
# columnas
element elasticBeamColumn 1111 111 121 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;
element elasticBeamColumn 1112 112 122 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;

# vigas paralelas a Y
element elasticBeamColumn 1221 121 122 $AgViga $Ec $Gc  $J  $IyViga
$IzViga $IDBeamTransf;;

# PÓRTICO 2
# columnas
element elasticBeamColumn 2111 211 221 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;
element elasticBeamColumn 2112 212 222 $AgCol $Ec $Gc  $J  $IyCol  $IzCol
$IDColTransf;
```

```
#Vigas
element elasticBeamColumn 2221 221 222 $AgViga $Ec $Gc  $J  $IyViga
$IzViga $IDBeamTransf;;

#Vigas (eje Z) Conectan Pórticos
# Póticos 1-2
element elasticBeamColumn  1321 121 221 $AgGird $Ec $Gc  $J  $IyGird
$IzGird $IDGirdTransf;
element elasticBeamColumn  1322 122 222 $AgGird $Ec $Gc  $J  $IyGird
$IzGird $IDGirdTransf;

# DEFINO LOS AISLADORES
#                  tag ndI ndJ nsecs secID transfTag
element zeroLength  1101     101  111  -mat 4  5  -dir 1  2
element zeroLength  1102     102  112  -mat 4  5  -dir 1  2
element zeroLength  2101     201  211  -mat 4  5  -dir 1  2
element zeroLength  2102     202  202  -mat 4  5  -dir 1  2
puts "ELEMENTOS CREADOS"

#elementos para simular la presencia de la  losa en el aislamiento
# SECCIONES VIGA:
set AgViga [expr $HViga*$BViga];         # área vigas
set IzVigaw [expr 10000./12*$BViga*pow($HViga,3)];# Gran Inercia
set IyVigaw [expr 10000./12*$HViga*pow($BViga,3)];# Gran inercia

# SECCIONES VIGA Z:
set AgGird [expr $HGird*$BGird];          # área vigas
set IzGirdw [expr 10000./12*$BGird*pow($HGird,3)]; # Gran inercia
 set IyGirdw [expr 10000./12*$HGird*pow($BGird,3)];# Gran inercia

element elasticBeamColumn 1571 111 112 $AgViga $Ec $Gc  $J  $IyVigaw
$IzVigaw $IDBeamTransf;
element elasticBeamColumn 1572 211 212 $AgViga $Ec $Gc  $J  $IyVigaw
$IzVigaw $IDBeamTransf;
element elasticBeamColumn 1573 111 211 $AgGird $Ec $Gc  $J  $IyGirdw
$IzGirdw $IDGirdTransf;
element elasticBeamColumn 1574 112 212 $AgGird $Ec $Gc  $J  $IyGirdw
$IzGirdw $IDGirdTransf;


#DEFINO CARGAS GRAVITARIAS, PESO Y MASAS
#=========================================================
set QdlCol [expr $GammaConcreto*$HCol*$BCol]; # peso por longitud columna
set QViga [expr $GammaConcreto*$HViga*$BViga];  # peso por longitud viga
set QGird [expr $GammaConcreto*$HGird*$BGird];   # peso por longitud vigas
set TLosa 0.20;       # espesor asumido para losa
set LLosa [expr $LGird/2]; # losa se extiente una distancia de $LGird/2
set Qlosa [expr $GammaConcreto*$TLosa*$LLosa];
set QdlViga [expr $Qlosa + $QViga];      # carga muerta distribuida en la viga
set QdlGird $QGird; # carga muerta distribuida en viga (eje z)
```

```
set PesoCol [expr $QdlCol*$LCol];       # peso total columna
set PesoViga [expr $QdlViga*$LViga];    # peso total viga
set PesoGird [expr $QdlGird*$LGird];    # peso total viga (eje z)

# ASIGNO MASAS EN LOS NUDOS
# cada nudo toma la 1/2 de la masa de cada elemento del pórtico
set Masa  [expr ($PesoCol/2 + $PesoViga/2+$PesoGird/2)/$g];
set Mcero 1.e-6;
# PÓRTICO 1
mass 121 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;            # level 2
mass 122 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;

# PÓRTICO 2
mass 221 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;            # level 2
mass 222 $Masa $Mcero $Masa $Mcero $Mcero $Mcero;

set PesoPiso1 [expr 4*$PesoCol + 2*$PesoGird + 2*$PesoViga];
set PesoTotal [expr 1* $PesoPiso1];     # peso total de la estructura
set MasaTotal [expr $PesoTotal/$g];     # masa total del edificio
puts "CARGAS DEFINIDAS Y MASAS ASIGNADAS"

# DEFINO RECORDERS
#=========================================================
recorder Node -file $dataDir/DAisl.out -time -node 111 112 211 212   -dof 1 2 3
disp;           #DESPLA CABEZA AISL
recorder Node -file $dataDir/DLibres.out -time -node 121 122 221 222   -dof 1
disp;           #DESPLA NUDOS LIBRES
recorder Node -file $dataDir/RBase.out -time -node 111 112 211 212   -dof 1 2
3 reaction;             #REACCIONES EN LOS APOYOS
recorder Drift -file $dataDir/Deriv.out -time -iNode 112 212 -jNode 122 222 -dof
1 -perpDirn 2;          # DERIVA DE PISO
recorder Element -file $dataDir/Fel1.out -time -ele 1111 localForce;         #
FUERZAS EN ELEMENTOS COORDENAS LOCALES

# ANÁLISIS GRAVITATORIO
#=========================================================
# Cargas gravitatorias aplicadas en los elementos
pattern Plain 101 Linear {
# Pórtico 1
# columnas
        eleLoad -ele 1111 -type -beamUniform 0. 0. -$QdlCol;
        eleLoad -ele 1112 -type -beamUniform 0. 0. -$QdlCol

# Vigas
        eleLoad -ele 1221 -type -beamUniform -$QdlViga 0.;

# Pórtico 2
# columnas
        eleLoad -ele 2111 -type -beamUniform 0. 0. -$QdlCol;
        eleLoad -ele 2112 -type -beamUniform 0. 0. -$QdlCol
```

```
# Vigas
        eleLoad -ele 2221 -type -beamUniform -$QdlViga 0.;

# Vigas (eje z)
# Frame 1-2
        eleLoad -ele 1321 -type -beamUniform $QdlGird 0.;
        eleLoad -ele 1322 -type -beamUniform $QdlGird 0.;

}

set Tol 1.0e-8;        # Tolerancia para la prueba de convergencia
variable constraintsTypeGravity Plain;            # default;
if {  [info exists RigidDiaphragm] == 1} {
        if {$RigidDiaphragm=="ON"} {
                variable constraintsTypeGravity Lagrange;
        };      # if rigid diaphragm is on
};      # if rigid diaphragm exists

constraints $constraintsTypeGravity; # condiciones de borde
numberer RCM;  # reenumera los dof para minimizar el ancho de banda
system BandGeneral ;  # como se resuelven el sistema de ecuaciones
test EnergyIncr $Tol 6 ;     # determina si la convergencia ha sido alcanzada
algorithm Newton;                    # usa algortimo de newton para la solución
set NpGrav 10;                       # aplico carg gravit en 10 intervalos
set DGrav [expr 1./$NpGrav];         # incrementos para la aplicación de la carga;
integrator LoadControl $DGrav;       # determino el paso de tiempo para el análisis
analysis Static;                     # defino tipo de análisis (estático)
analyze $NpGrav;                     # aplico cargas gravitatorias
loadConst -time 0.0; # mantiene constante las cargas gravit y reinicia el tiempo
set Tol 1.0e-6;        # reduce la tolerancia después de las cargas gravitatorias
puts "ANÁLISIS GRAVITATORIO REALIZADO"

# ACCIÓN SÍSMICA UNIFORME
#=========================================================
# COLOCAR PARÁMETROS PARA ANÁLISIS SÍSMICO
wipeAnalysis; # Borra los objetos que definen el tipo de análisis
set DtAnalysis        [expr 0.02]; # paso para análisis sísmico
set TmaxAnalysis    [expr 40];      # duración del sismo (max 50seg)

# ASIGNO AMORTIGUAMIENTO RAYLEIGH
set xDamp 0.05;     # factor de amortiguamiento
set MpropSwitch 1.0;
set KcurrSwitch 0.0;
set KcommSwitch 0.0;
set KinitSwitch 0.0;
set nEigenI 1;                  # MODO1
set nEigenJ 2;
set lambdaN [eigen [expr $nEigenJ]];
 set lambdaI [lindex $lambdaN [expr $nEigenI-1]];      # eigenvalor modo i
set lambdaJ [lindex $lambdaN [expr $nEigenJ-1]];      # eigenvalor modo j
```

```
set omegaI [expr pow($lambdaI,0.5)];
set omegaJ [expr pow($lambdaJ,0.5)];
set alphaM [expr$MpropSwitch*$xDamp*(2*$omegaI*$omegaJ)/($omegaI
+$omegaJ)]; # Parámetro alpha para amortiguamiento Rayleigh
set betaKcurr [expr $KcurrSwitch*2.*$xDamp/($omegaI+$omegaJ)];
set betaKcomm [expr $KcommSwitch*2.*$xDamp/($omegaI+$omegaJ)];
set betaKinit [expr $KinitSwitch*2.*$xDamp/($omegaI+$omegaJ)];
rayleigh $alphaM $betaKcurr $betaKinit $betaKcomm;

set lDloadTag 400; # ETIQUETA PARA PATRÓN DE CARGA
set dt 0.02;            # PASO DEL SISMO
set GMdirection 1;  # DIRECCIÓN DEL SISMO

set AccelSeries "Series -dt $dt -filePath centro.txt -factor  1"; # defino sismo y
factor de escala
pattern UniformExcitation  $lDloadTag  $GMdirection -accel  $AccelSeries; #
creo acción uniforme
source LibAnalysisDynamicParameters.tcl; # invoco función que define los
parámetros para el análisis
set Nsteps [expr int($TmaxAnalysis/$DtAnalysis)];
set ok [analyze $Nsteps $DtAnalysis]; # realizo análisis; regreso ok=0 si el
análisis es exitoso

#PROCEDIMIENTO EN CASO DE ANÁLISIS NO EXITOSO

if {$ok != 0} {     ;                                   # ANÁLISIS FALLIDO

        # Se cambia algunos parámetros para alcanzar convergencia
        # Proceso es más lento dentro de este lazo
        #   Análisis controlado por tiempo
        set ok 0;
        set controlTime [getTime];
        while {$controlTime < $TmaxAnalysis && $ok == 0} {
                set controlTime [getTime]
                set ok [analyze 1 $DtAnalysis]
                if {$ok != 0} {
                puts "Trying Newton with Initial Tangent .."
                test NormDispIncr   $Tol 1000  0
                algorithm Newton -initial
                set ok [analyze 1 $DtAnalysis]
                test $testTypeDynamic $TolDynamic $maxNumIterDynamic  0
                        algorithm $algorithmTypeDynamic
                }
                if {$ok != 0} {
                        puts "Trying Broyden .."
                        algorithm Broyden 8
                        set ok [analyze 1 $DtAnalysis]
                        algorithm $algorithmTypeDynamic
                }
                if {$ok != 0} {
```

```
                    puts "Trying NewtonWithLineSearch .."
                    algorithm NewtonLineSearch .8
                    set ok [analyze 1 $DtAnalysis]
                    algorithm $algorithmTypeDynamic
                }
            }
};      # end if ok !0

puts "ANÁLISIS SÍSMICO REALIZADO. End Time: [getTime]"
```

En la figura 5.32 se muestra la comparación entre los resultados de desplazamientos en el centro de masas en el sistema de aislamiento, obtenidos con los programas Masacorregida y OpenSEES. No obstante la variación de resultados, es importante destacar que para ambos casos, el intervalo de tiempo en el cual se presentan las respuestas máximas es el mismo. Para que se visualice de mejor manera, en esta figura solamente se presenta el intervalo de tiempo en el cual las respuestas son máximas.

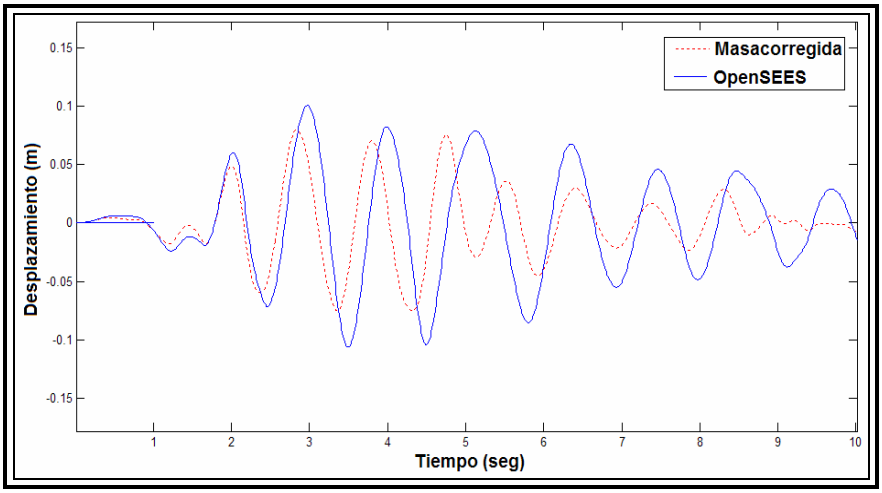

Figura 5.32: Comparación resultados Masacorregida - OpenSEES

Al inicio aparece una línea recta en la serie correspondiente a OpenSEES, esta representa la aplicación lineal de carga gravitatoria que sucede en un segundo,

según lo definido en el análisis. Luego de esto, el tiempo se reinicia y comienza el análisis dinámico.

La variación encontrada puede derivarse de varios aspectos, como son: los métodos de análisis, los modelos considerados, ya que en el programa masacorregida se trabaja con un modelo de tres grados de libertad por planta mientras que en OpenSEES se trabaja con modelo de seis grados de libertad por nudo, por otra parte OpenSEES trabaja con el procedimiento de integración de Newmark el cual se sabe es bastante sensible. En los resultados se ha encontrado discrepancias en el valor del periodo de vibración, esto se atribuye fundamentalmente al tratamiento que uno y otro programa dan al aislamiento, ya que el periodo de vibración es gobernado principalmente por las propiedades del sistema de aislamiento.

REFERENCIAS

1. Mazzoni S., McKennna F., Scout M., Fenves G., (2006), “OpenSees Command Language Manual”. 465p.

2. Aguiar R., Almazán D., Dechent P., Suárez V., (2008), “Aisladores de base elastoméricos y FPS”. Centro de Investigaciones Científicas. Escuela Politécnica del Ejército. 290 p. Sangolquí, Ecuador.