UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA

INSTITUTO DE INGENIERÍA

SOFTWARE PARA LA ENSEÑANZA DE LA DINÁMICA ESTRUCTURAL

T E S I S

QUE PARA OPTAR POR EL GRADO DE:

MAESTRO EN INGENIERÍA

INGENIERÍA CIVIL – ESTRUCTURAS

P R E S E N T A :

OCTAVIO HINOJOZA GABRIEL

TUTOR:
DR. MARIO GUSTAVO ORDAZ SCHROEDER

INSTITUTO DE INGENIERÍA UNAM

2009

JURADO ASIGNADO:

Presidente:	DR. AYALA MILIAN AMADO GUSTAVO
Secretario:	DR. REINOSO ANGULO EDUARDO
Vocal:	DR. ORDAZ SCHROEDER MARIO GUSTAVO
1er. Suplente:	DR. PÉREZ GAVILÁN ESCALANTE JUAN JOSÉ
2do. Suplente:	M.I. GARCÍA DOMÍNGUEZ OCTAVIO

Lugar donde se realizó la tesis:

INSTITUTO DE INGENIERÍA, UNAM.

TUTOR DE TESIS:

_____

MARIO GUSTAVO ORDAZ SCHROEDER

Dedicada a mi esposa y mi hijo

Agradecimientos:

A mis padres que sin su apoyo no hubiera podido llegar a ser un profesionista

A mi esposa e hijo que me han apoyado incondicionalmente durante los pocos años que llevamos juntos

A todos mis amigos por sus consejos y palabras de apoyo que me hicieron salir adelante

Al *CONACYT*, por la beca proporcionada durante mis estudios de maestría

Al *Instituto de Ingeniería, UNAM*, por el apoyo técnico proporcionado en sus instalaciones durante la elaboración de esta tesis.

Al *Dr. Mario Ordaz* por haber dirigido esta tesis

A los sinodales por sus valiosos comentarios y sugerencias hechas a este trabajo

Y muy especialmente a esa persona que me ayudó en esos momentos tan difíciles de superar y que cuando pensé que el camino llegaba a su fin me demostró que siempre hay una mejor manera de vivir.

CONTENIDO

RESUMEN……... iv

ABSTRACT……. iv

CONTENIDO…….. v

INTRODUCCIÓN…….. 1

PLANTEAMIENTO DEL PROBLEMA…….……..... 1
RAZÓN PARA CREAR UN SOFTWARE DE ENSEÑANZA…… 1
OBJETIVOS Y ALCANCES …… 2

CAPÍTULO 1. SOFTWARE EDUCATIVO……... 3

1.1. LA ENSEÑANZA……..…… 3
1.2. SOFTWARE EDUCATIVO…….……. 5
1.3. SOFTWARE PARA DINÁMICA ESTRUCTURAL ……..……. 8

CAPÍTULO 2. TEMAS DE DINÁMICA INCLUIDOS……. 9

2.1. CONCEPTOS DE DINÁMICA ESTRUCTURAL…… 9
2.2. OSCILADORES DE 1GL……..…… 10
2.2.1. Ecuación de equilibrio…… 11
2.3. OSCILADORES DE VGL……. 11
2.3.1. Ecuación de equilibrio…… 12
2.4. TEMAS INVOLUCRADOS…….. 13

CAPÍTULO 3. CONCEPTOS DE PROGRAMACIÓN…….. 15

3.1. PROGRAMAS DE COMPUTADORA……. 15
3.1.1. Programas de aplicación……. 15
3.2. INTERFAZ GRÁFICA DE USUARIO……. 16
3.2.1. Controles……. 16
3.3. CÓDIGO FUENTE…… 21
3.3.1. Algoritmo…… 22
3.3.2. Formas de programar…….. 22
3.3.2.1. Programación estructurada……... 23
3.3.2.2. Programación modular…….…. 23
3.3.2.3. Programación orientada a objetos…… 24
3.3.2.4. Programación dirigida por eventos……...…… 26
3.3.3. Errores…… 26

CAPÍTULO 4. DESCRIPCIÓN DEL SOFTWARE……. 27

4.1. CARACTERÍSTICAS DEL PROGRAMA……... 27
4.1.1. Objetivo del programa…… 27
4.1.2. Usuarios…….……. 27
4.1.3. Tipo de software.…… 27

4.1.4. Lenguaje de programación…….. 27
4.1.5. Requisitos del sistema……. 28
4.2. INTERFAZ GRÁFICA DE USUARIO……. 28
4.2.1. Ventana de presentación……. 30
4.2.2. Ventana general…….. 30
4.2.2.1. Barra de menús……..……. 31
4.2.3. Ventana de osciladores de 1GL…….. 32
4.2.3.1. Control de osciladores……. 36
4.2.3.2. Control de gráficas…….. 37
4.2.3.3. Control Animación de espectros…….. 38
4.2.3.4. Control Animación de osciladores……... 39
4.2.3.5. Gráfica de carga…….. 40
4.2.3.6. Gráficas de respuesta…….. 41
4.2.3.7. Menú secundario de las Gráficas de respuesta……... 42
4.2.3.8. Ventana Respuesta numérica……... 43
4.2.3.9. Ventana Definir k……. 44
4.2.3.10. Osciladores…….. 46
4.2.3.11. Espectros de respuesta……. 47
4.2.3.12. Menú secundario de espectros……. 48
4.2.3.13. Ventana Valores de espectros…….. 48
4.2.4. Ventana de osciladores de VGL……. 50
4.2.4.1. Barra de herramientas……. 54
4.2.4.2. Control de propiedades…… 54
4.2.4.3. Control de gráficas…….. 55
4.2.4.4. Control Animación del oscilador…… 56
4.2.4.5. Control Modos de vibrar…….. 57
4.2.4.6. Gráfica de carga…….. 58
4.2.4.7. Gráficas de respuesta……... 58
4.2.4.8. Menú secundario de las Gráficas de respuesta……... 59
4.2.4.9. Ventana Datos de respuesta……. 59
4.2.4.10. Oscilador…….. 64
4.2.4.11. Espectros de piso……. 66
4.2.4.12. Ventana tipo de amortiguamiento…… 67
4.2.4.13. Ventana Propiedades……... 67
4.2.5. Ventanas comunes…….. 68
4.2.5.1. Ventana Carga senoidal……... 68
4.2.5.2. Ventana lectura de archivo de datos…… 70
4.2.5.3. Ventana base de datos…….. 72
4.2.5.4. Ventana Ayuda……. 74
4.2.5.5. Ventana Acerca de……... 76
4.2.5.6. Ventana apariencia…….. 77
4.2.5.7. Ventana Espectros de respuesta…….. 77

CAPÍTULO 5. ALGORITMOS…… 81

5.1. RESPUESTA DINÁMICA…… 81
5.1.1. Respuesta de osciladores de 1GL……... 81
5.1.2. Espectros de respuesta…… 88
5.1.3. Fuerza cortante…… 90
5.1.4. Respuesta de osciladores de VGL…….. 91
5.1.5. Espectros de Piso…… 98
5.1.6. Fuerza cortante…… 100
5.2. GRÁFICAS…… 101
5.2.1. Señales…… 101
5.2.2. Osciladores……. 108
5.2.2.1. Osciladores de 1GL……... 108
5.2.2.2. Oscilador VGL…….. 113
5.2.3. Espectros…… 117
5.2.3.1. Espectros de respuesta……... 117
5.2.3.2. Espectros de piso……... 119
5.3. ANIMACIÓN……..... 121
5.3.1. Señales…… 122
5.3.2. Osciladores……. 128
5.3.2.1. Osciladores de 1GL……... 128
5.3.2.2. Oscilador VGL……... 128
5.3.3. Espectros……. 129
5.4. MANEJO DE ERRORES……... 131
5.4.1. Errores de ejecución…… 131
5.4.2. Errores lógicos…… 131
5.4.3. Mensajes de advertencia……. 131
5.4.4. Validación de campos de texto……... 134
5.4.5. Controles Try Cath…….. 139

CAPÍTULO 6. APLICACIÓN DEL PROGRAMA…… 141

6.1. INSTALACIÓN……. 141
6.1.1. Errores de instalación…….. 143
6.1.2. Desinstalación……. 143
6.2. MODO DE USO……. 144
6.2.1. Ventana osciladores de 1GL……... 144
6.2.2. Ventana osciladores de VGL…….. 154
6.3. COMPARACIÓN DE LA RESPUESTA DINÁMICA……. 159
6.4. EVALUACIÓN DEL SOFTWARE……... 162

CAPÍTULO 7. CONCLUSIONES…… 167

REFERENCIAS Y BIBLIOGRAFÍA…….. 169

RESUMEN

Aun con la aparición de la computadora en los centros escolares, el proceso enseñanza-aprendizaje se ha mantenido constante en la *forma tradicional* de dar clase, sin incorporar el uso de la computadora de manera sustancial en el aula, debido principalmente a la falta de un software estrictamente educativo. Ante tal situación, el profesor de cada materia trata de adecuar el uso de programas comerciales a su clase, pero en general éstos resultan ser complejos e insuficientes para los objetivos educativos de cada materia.

La dinámica estructural es una materia en la que generalmente los resultados de sus ejemplos dependen del tiempo, y por lo tanto no es sencillo de enseñar en un medio estático como el papel y el pizarrón.

El objetivo de este trabajo es crear un software educativo para el apoyo a la enseñanza de la dinámica estructural; el cual muestre las animaciones de la respuesta de los osciladores de 1GL y VGL ante diferentes tipos de excitación. Para lograrlo, se desarrolla un programa con interfaz gráfica (aplicación para Windows) en Visual Basic.Net 2008 que utiliza una aproximación numérica de la integral de Duhamel, conocido también como el método de las ocho constantes, para obtener la respuesta dinámica de los osciladores.

ABSTRACT

Even with the advent of computers in schools, the teaching-learning process has remained constant in the traditional way of teaching, without incorporating the use of computers in any substantial way in the classroom, mainly due to lack of strictly educational software. Given this situation, the teacher of each subject tries to fit the use of commercial programs to its class, but in general these are insufficient and complex for the educational objectives of each subject.

Structural dynamics is an area in which generally the results of its problems depend on time, and therefore it is not easy to teach in a static way as using paper and board.

The objective of this work is to create educational software to support the teaching of structural dynamics; this software shows animations of the response single degree of freedom system (SDOF) and multiple degree of freedom system (MDOF) to different excitation types. To achieve this, a program (Windows application) in Visual Basic.Net 2008 is developed; this program uses a numerical approximation of Duhamel's integral, also known as *eight constants* method, to obtain the dynamic response of both systems (SDOF and MDOF).

INTRODUCCIÓN

PLANTEAMIENTO DEL PROBLEMA

El proceso enseñanza-aprendizaje no ha cambiado mucho aun con la aparición de la computadora en los centros escolares. Dicho proceso se ha mantenido constante en la *forma tradicional* de dar clase, sin incorporar el uso de la computadora de manera sustancial en salón debido en gran medida a la falta de un software educativo (Cuevas, 2001) creado especialmente para cada clase.

El software que hay en el mercado no cumple con las necesidades para ser un software educativo (Cuevas, 2001) y por lo tanto, cuando su uso se llega a presentar en un salón de clase, resulta ser limitado y complejo de manejar.

La dinámica estructural es un excelente tema para crear un software educativo, ya que sus resultados son dependientes del tiempo y mostrarlos en clase en forma estática como en el papel y pizarrón es difícil cuando su respuesta varía en el tiempo.

Con ayuda de la computadora se puede incorporar un programa que muestre gráficas, sonidos y animaciones, para ayudar al profesor a mostrar la respuesta dinámica y al alumno a comprender mejor el comportamiento dinámico de las estructuras.

Si en clase se quisiera mostrar cada gráfica de respuesta tras cambiar los parámetros de cada ejercicio, el profesor tendría que dibujar el movimiento en cada instante de tiempo de los osciladores en estudio, o dibujar todas las gráficas de respuesta para cada ejemplo dependiendo de las variaciones de los parámetros. Situación que resulta muy compleja sin la ayuda de una computadora.

El principal problema entonces, que se trata en esta tesis, es la falta de software educativo para el apoyo a la enseñanza de la dinámica estructural, que podría ayudar a crear una mejor manera de enseñar y aprender al complementar la manera tradicional.

RAZÓN PARA CREAR UN SOFTWARE EDUCATIVO

Una de las razones para crear un software educativo sería evaluar si es necesario cambiar la forma de dar clase tradicional; si bien es cierto que durante muchos años la forma tradicional de enseñar ha sido la formadora de profesionistas y trabajadores (Benítez, 2000), complementarla con la ayuda de los recursos de la tecnología como la computadora podría traer un mayor beneficio en el proceso enseñanza-aprendizaje.

Sin duda, la tecnología está cambiando al mundo constantemente y cada día nos vemos inmersos en nuevos adelantos tecnológicos que facilitan más nuestras tareas diarias. Es inevitable que la tecnología llegue al salón de clase y que incluso ya ha comenzado con algunos programas educativos en temas como matemáticas y física (Franco, 2005); por esta razón hay que abrirle las puertas del salón de clase a la tecnología creando y utilizando software educativo, así como las demás herramientas de las que la tecnología nos puede proporcionar.

OBJETIVOS Y ALCANCES

El objetivo de esta tesis es crear un software educativo con las siguientes características:

- Que sea una aplicación para Windows y que su interfaz gráfica sea sencilla y amigable al usuario, para que éste no tenga complicaciones al usar el programa y en vez de entender los conceptos de los temas de contenido tenga que invertir demasiado tiempo en conocer el funcionamiento del programa.

- Ser una herramienta que sirva para apoyar la enseñanza de la Dinámica Estructural

- Ser un programa con el cual se puedan comparar los resultados obtenidos de ejemplos hechos en clase y mostrados en la bibliografía del tema.

- Mostrar animaciones del movimiento de los osciladores en el tiempo, así como de las señales de respuesta, para que el usuario observe y comprenda el comportamiento de los osciladores ante diferentes tipos de carga.

- Mostrar animaciones de los espectros de respuesta, para que el usuario vea cómo cambia la señal de respuesta de los osciladores conforme el valor del periodo de la estructura lo hace y construya el espectro de respuesta correspondiente.

- Mostrar la respuesta de cada grado de libertad de un oscilador de VGL, y que de igual manera se puedan observar sus animaciones.

También, demostrar que con la ayuda de un software educativo la manera de dar clase tradicional puede cambiar para bien, sin necesidad de sustituirla completamente sino complementarla con una herramienta como lo es un software educativo.

Fomentar el interés en los alumnos por estudiar el tema de dinámica estructural, al mostrar en clase las animaciones del comportamiento de los osciladores y las gráficas de respuesta.

Los alcances del presente trabajo son principalmente:

- El programa sólo puede mostrar estructuras modeladas como osciladores de 1GL y VGL, con propiedades elástico-lineales.

- El cálculo de la respuesta dinámica para ambos tipos de osciladores (1GL y VGL) se realizará con una aproximación numérica de la integral de Duhamel; conocido también como el método de las ocho constantes.

- El diseño de la interfaz gráfica se hará con el lenguaje de programación Visual Studio. Net

- El software se concibe como un *simulador* de la respuesta dinámica de los osciladores, y no como un programa que enseñe, por si solo, la dinámica estructural.

CAPÍTULO 1

SOFTWARE EDUCATIVO

1.1. LA ENSEÑANZA

La educación tiene un papel trascendental en la formación de profesionistas y trabajadores (Benítez, 2000), por ello siempre será necesario mejorarla.

Actualmente, la educación se encuentra en la situación de adaptar la tecnología a los procesos de enseñanza/aprendizaje propios de la sociedad del siglo XXI. Con esta adaptación viene el cambio a nuevas costumbres y requerimientos primordiales de un mundo tecnológico que ha pautado la forma de conocer y apropiarse de la realidad mediante los recursos tecnológicos actuales (Gértrudix, Álvarez, Galisteo y Gálvez *et. al.* 2007).

La computadora, uno de los avances tecnológicos de los que puede disponer la educación, ha producido el impacto cultural más importante del siglo pasado después del automóvil (Cuevas, 2001) y todos los que se dedican de una u otra forma a la investigación y a la docencia no pueden evitar.

La aparición de la computadora personal en la década de los 80's y su difusión en los centros escolares e incluso en los hogares, hizo pensar que este instrumento iba a tener un papel preponderante en el sistema educativo, que iba a cambiar el modo en el que se enseña y la forma en que los estudiantes aprenden (Bork, 1985).

Aunque la computadora se ha convertido en una herramienta imprescindible, su impacto dentro del aula de clase aún no se ha dado como en los campos laboral, docente y de investigación (Vilchis, 2005).

Con las computadoras es posible mejorar el proceso de enseñanza-aprendizaje (Franco, 2005), a fin de que:

- ✓ El aprendizaje sea más interesante
- ✓ El aprendizaje sea activo, y no pasivo como ocurre frecuentemente en las aulas
- ✓ Los estudiantes estén más motivados; esta última no equivale a estar más "entretenidos"
- ✓ El aprendizaje sea al ritmo individual del estudiante

El trabajo de un docente dentro de un salón de clase se dificulta menos si se incorpora la ayuda de la ciencia y la tecnología que ha transformado a nuestra sociedad.

Anteriormente la falta de acceso a una computadora, aunado con la poca preparación para usar las máquinas en algunos profesores, eran dos de los aspectos que limitaban el uso de esta herramienta dentro de los salones de clase. Actualmente, estos aspectos han sido superados pues cada día más estudiantes tienen acceso a una computadora portátil y los profesores se han unido al cambiante mundo de la computación.

No obstante, las clases en el aula siguen siendo de manera tradicional, en el mejor de los casos el profesor muestra en clase una serie de resultados en su computadora con la ayuda de un proyector para comprobar los cálculos realizados. Esta situación se debe a la falta de software estrictamente educativo adecuado para utilizar en clase (Mendoza, 2001). Por lo general, el docente se dedica a comprobar los

resultados de los cálculos con algún programa comercial, o en el mejor de los casos trata de adecuar un pequeño programa de computadora al tema de clase.

Nuestro mundo avanza tan vertiginosamente y todo apunta hacia el uso de la computadora en el aula, de tal manera que los involucrados (docentes y estudiantes) tienen que capacitarse y adecuarse a estos avances (Mendoza, 2001), de modo que el proceso de enseñanza/aprendizaje tiene que ser más dinámico e interesante.

La inquietud tanto de docentes como de investigadores, que han visto en la tecnología un motor de cambio del sistema educativo, han impulsado el uso de la tecnología dentro del salón de clase (Gértrudix, Álvarez, Galisteo y Gálvez *et. al.* 2007). El papel del profesorado es esencial para que tenga éxito cualquier cambio en el proceso educativo.

La tecnología permite conjuntar diversas herramientas en un software educativo, por ejemplo el empleo del sonido y la imagen fija o en movimiento (animaciones). Con la ayuda de los procesos multimedia es posible facilitar al estudiante la adquisición de nuevos conceptos al presentarle tanto imágenes como modelos animados de diversos procesos y permitirle llevar de forma interactiva con la computadora su propio aprendizaje.

Los temas de contenido son uno de los puntos más importantes en la creación de un software educativo; para ello los profesores juegan un papel primordial, ya que ellos deben personalizar los temas creando o seleccionando los contenidos que considera más apropiados dependiendo de sus criterios educativos y del tipo de alumnos a los que va destinado.

Algunos de los aspectos que deben tener en cuenta son:

- ✓ Conocer cómo percibe el usuario la información en una pantalla
- ✓ Estructurar jerárquicamente la información y establecer relaciones entre ventanas
- ✓ Proporcionar toda la información y actividades para enseñar un determinado tema

Para un profesor no programador es difícil crear un software interactivo; por lo tanto, es necesario contar con la ayuda de un desarrollador de software para integrar los temas correspondientes de una materia a una aplicación de computadora.

El diseñador debe crear el software educativo a partir de las especificaciones que los profesores proporcionen. Los programas pueden ser de diversos tipos, algunos pueden tener un cierto grado de interactividad, otros pueden ser fotografías, dibujos, animaciones simples, pequeñas secuencias de vídeo, o incluso componentes complejos como páginas Web.

Para desarrollar un software educativo se debe tener en cuenta los siguientes aspectos (Franco, 2005):

- ✓ A quiénes va dirigido
- ✓ Entender la materia que se va a enseñar
- ✓ Seleccionar los recursos
- ✓ Aprender a usar las herramientas para hacer estos recursos documentos digitales
- ✓ Aprender a usar estos recursos en el ámbito educativo.

Ahora bien, cualquier composición (software y temas de contenido) puede no ser válida desde el punto de vista educativo y por consiguiente llevar al fracaso, por lo que es necesario tener muy claros los objetivos educativos que se pretenden y los medios para alcanzarlos (seleccionar los recursos, símbolos y lenguajes de la tecnología que resulten apropiados para su uso educativo).

Uno de los objetivos de adaptar un software educativo al aula de clase es lograr que los estudiantes adquieran una visión creativa y participativa del aprendizaje, que asuman su papel como actores de sus propios procesos y descubran la manera de cómo aprenden, diferente a otros procesos y determinado por sus propias experiencias (Ávila, 2007).

Las bases pedagógicas y didácticas en la creación del software educativo servirán para que el docente reconozca que el uso de la tecnología y de otros auxiliares de la enseñanza tiene una influencia positiva en el salón de clase.

1.2. SOFTWARE EDUCATIVO

Es un producto tecnológico diseñado para apoyar procesos educativos que utilizan tanto el profesor como el alumno para alcanzar determinados propósitos. Además, es un medio de presentación y desarrollo de contenidos educativos, como puede ser un libro o un video, con su propio formato expresivo y secuencia narrativa (Morales, 1998).

La selección del software más adecuado para una determinada clase está en función de las necesidades de enseñanza y aprendizaje de los alumnos; además se debe aprovechar los avances tecnológicos tanto en hardware como en software para su elaboración.

Un software educativo involucra a tres ciencias (Cuevas, 2001):

- ✓ *La psicología*: para crear un modelo didáctico explícito (mediante un conocimiento no elemental de las ciencias cognitivas), en donde se tenga muy clara la participación de cada uno de los elementos involucrados: la computadora, el maestro y el estudiante y en donde se tenga una forma de evaluar la comprensión de los conceptos del tema a enseñar.

- ✓ *Temas de contenido (en el área correspondiente):* para poder enseñar el tema es necesario conocerlo y que se tengan muy claros por parte del profesor y/o diseñador los conceptos implícitos y explícitos del mismo.

- ✓ *La computación*: necesaria para la conjunción de las dos ciencias anteriores. Es indispensable tener amplios conocimientos de programación para proporcionar lo mejor a la interfaz gráfica de usuario.

Para la producción de un software educativo los desarrolladores y profesores deben tener presentes estos tres elementos y comprender que la carencia de alguno de ellos debilita la intención de ser un instrumento de ayuda en el aprendizaje y enseñanza del tema que se trate.

Al diseñar sistemas educativos basados en la computadora, nuestra preocupación primaria no ha de estar con la aplicación una nueva tecnología, ni debemos extraviarnos románticamente por metas poco realistas como remplazar a maestros, libros o incluso las actividades físicas y sociales de estudiantes a través de la interacción de la estudiante-máquina (Franco, 2005).

La creación del software debe estar centrada en los requerimientos del usuario, tales como: necesidades, criterios de búsqueda y formas de utilización.

La creación de programas interactivos no es una tarea sencilla. Es necesario tener amplios conocimientos de uno o varios lenguajes programación, y requiere de la formulación de objetivos educativos que se pretende enseñar de forma interactiva.

Otro aspecto que se debe tener en cuenta en la elaboración del software es el hecho de que en ocasiones no hay una solución analítica al problema en cuestión por lo que es necesario emplear procedimientos numéricos para poder encontrar la solución, esto lleva a la creación de algoritmos eficientes.

En la mayor parte de los casos, es necesario definir escalas para evitar que la computadora trabaje con números excesivamente grandes o pequeños.

Algunas ventajas de usar un software interactivo son:

- Las figuras y sonidos atraen rápidamente la atención del usuario, ayudándolo en la interpretación de gráficas mediante la asociación entre el fenómeno físico simulado y la representación grafica de dicho fenómeno, facilitando el proceso de análisis.

- La densidad de la información percibida en un área de la pantalla es superior a la equivalente de un libro.

- Cuando algún tema a enseñar es dependiente del tiempo es posible hacer uso de las técnicas de animación para mostrar su evolución en el tiempo. Las animaciones en un programa de computadora proporcionan un modo que no es posible en los medios estáticos como el papel.

Pero también se tienen algunas desventajas al hacer uso de un software educativo:

- Si bien las pantallas actuales tienen una resolución gráfica que facilita la lectura de documentos, la percepción de la información es diferente en una pantalla que en una hoja de papel impresa de un libro o artículo.

- La lectura del texto en la pantalla de la computadora produce mayor cansancio al lector que la equivalente en un papel impreso.

1.2.1. Composición

Una característica importante de un software educativo es su interfaz gráfica, ya que ésta permite al usuario interactuar con el programa. La interfaz debe tener ciertas características que le permitan al usuario entender fácilmente el manejo de sus controles así como la comprensión del tema que se trata de enseñar.

Algunas de las características que se deben definir dentro de la interfaz gráfica (Franco, 2005) son:

- ✓ Establecer el color de fondo del programa
- ✓ Dar formato al texto con diferentes tamaños, estilos y alineaciones del texto
- ✓ Insertar las figuras en la posición adecuada y otros elementos como un panel de controles
- ✓ Definir los enlaces a otras ventanas

La complejidad de uso de la interfaz gráfica se debe reducir al mínimo posible. Así mismo, se debe decidir qué parámetros se mantienen constantes y cuáles se permiten variar al usuario dentro de ciertos límites. Además, se tendrá que decidir la manera de introducir los datos mediante campos de texto, barras de desplazamiento, el uso del ratón, etc.

Un aspecto a considerar de especial relevancia es el desarrollo de la documentación técnica a lo largo del ciclo de vida del software, ya sea interna o externa. La primera considera a todos aquellos comentarios del programa que sean útiles a la hora de realizar modificaciones posteriores.

La documentación externa es la que se refiere a todo el material confeccionado a partir de la etapa inicial de análisis, conteniendo diagramas de entidades y relaciones, estructuras de datos, diagrama de flujos de procesos, diseño modular descendente etc. (Cataldi, 2006). Y toda aquella documentación que se considere pertinente para interpretar el desarrollo del programa.

Por último hay que señalar la necesidad de crear un manual de usuario claro y didáctico (Cataldi, 2006), para que el docente pueda recurrir a él, como elemento de ayuda. En este manual, se consideran todos los aspectos técnicos requeridos para el funcionamiento del programa y se podrá incluir una guía de soluciones para aquellos problemas más frecuentes.

Deberá proveer además ejemplos de uso, objetivos, contenidos, direccionamiento, actividades propuestas, teoría del aprendizaje considerado y así como el tratamiento de los posibles errores de los estudiantes durante el proceso de aprendizaje (Cataldi, Lage, Pessacq y García *et. al.* 2003).

1.2.2. Evaluación

Es necesario observar cómo los estudiantes perciben los distintos elementos que se disponen en el reducido espacio de una pantalla (Franco, 2005). La percepción de los usuarios no siempre coincide con la del diseñador del programa y se ha de tener en cuenta que aunque la interfaz gráfica ocupa toda la pantalla en algunos casos ésta puede ser de dimensiones pequeñas.

También se deberán incluir los resultados de las evaluaciones efectuadas, ya sean positivas o negativas y detallar cada una de las funcionalidades, considerando los aspectos técnicos, detallando los resultados estadísticos y teniendo en cuenta el tipo de instrumentos elaborados para la toma de dichos resultados, incluyendo además los criterios que se utilizaron para la evaluación desde los puntos de vista técnico y pedagógico (Cataldi, Lage, Pessacq y García *et. al.* 2003).

Se ha de verificar el comportamiento del programa, comprobando que los resultados que proporciona para todos los casos son correctos y que no se interrumpe su ejecución debido a errores internos.

Finalmente, se deberá de rediseñar los controles en función de los resultados del proceso de depuración de errores y de la observación del comportamiento de los usuarios cuando trabajan con el programa.

1.2.3. Sugerencias para la disposición del software

Se deben tener en cuenta los siguientes puntos (Cuevas, 2001):

Precisar en que forma y tiempo intervendrán la computadora y/o software en el curso; el profesor con las explicaciones pertinentes y el alumno. Es necesario aclarar el rol de cada uno, antes de incorporar a la computadora en el aula.

Se debe de tener claridad en que conceptos del tema se van a enseñar y para cada uno se deberá de plantear una serie de actividades cuyo propósito es guiar al estudiante para que a través de sus acciones adquiera las habilidades deseadas por el profesor, así como la comprensión del concepto.

Es responsabilidad del profesor identificar tales operaciones y conectarlas bajo la guía de un planteamiento didáctico, trasparente al estudiante, pero explicito para el docente.

Plantear problemas que sean de interés para los estudiantes de acuerdo a su nivel escolar, cuya solución conlleve a la construcción del concepto a enseñar.

Diseñar diversas actividades aprovechando la posibilidad de la computadora para que el estudiante visualice y manipule diferentes registros de representación de los conceptos bajo estudio.

Considerar a la computadora como una herramienta cognitiva más que como una herramienta auxiliar para realizar cálculos numéricos o simbólicos.

1.3. SOFTWARE PARA DINÁMICA ESTRUCTURAL

La dinámica estructural es el tema de contenido en el presente trabajo abarcando principalmente los sub temas: osciladores de uno y varios grados de libertad.

El estudio del movimiento ondulatorio de los osciladores no es sencillo de enseñar en un medio estático como el papel, pizarrón, etc. (Franco, 2005), ya que su comportamiento es dependiente del tiempo. Para explicar este tema, es importante no sólo obtener el valor máximo de la respuesta dinámica (un instante de tiempo), sino también como evoluciona la representación gráfica temporalmente.

La ventaja que puede proporcionar un software en este sentido es que el estudiante vea el movimiento del oscilador antes de comenzar a resolver el problema y poder analizarlo a partir de la observación de las distintas etapas del movimiento del mismo.

A falta de un software educativo en dinámica estructural, por lo menos documentado en nuestro país, se opta por programas comerciales de cálculo estructural como Staad, Sap, Eco etc., para mostrar los ejemplos hechos en clase, la mayoría de ellos tienen opciones para calcular la respuesta dinámica y mostrar la animación de la estructura que se haya modelado.

Sin embargo, estos programas están diseñados para otro tipo situaciones, con necesidades y características educativas casi nulas y además sin existir de por medio una adecuada adaptación al proceso enseñanza/aprendizaje (Mendoza, 2001), esto se entiende ya que no son diseñados con la finalidad de enseñar sino para proporcionar resultados de análisis de diversas estructuras ante diferentes tipos de cargas.

El uso de estos programas en el proceso de enseñanza/aprendizaje se limita a mostrar los resultados de la respuesta dinámica de un oscilador o una estructura. También, es difícil manipularlos para mostrar un ejemplo de manera rápida pues dichos programas por lo general necesitan muchos más parámetros (dimensiones, secciones, propiedades de material, etc.) adicionales a las propiedades dinámicas de los osciladores.

Existen otros programas, desarrollados por investigadores como el Degtra (Ordaz y Montoya, 2002) o el USSE (university Illinois, 2001) entre otros, que son creados especialmente para obtener resultados correspondientes al tema de dinámica estructural, pero que de igual manera no están diseñados con fines de ser software educativo, más bien son un medio para obtener cálculos específicos requeridos en una investigación.

El presente trabajo en un intento por cambiar la forma de enseñar el tema de dinámica estructural, tal vez en un futuro haya libros que su contendido estático esté inmerso en un programa de computación interactivo con los usuarios.

CAPÍTULO 2

TEMAS DE DINÁMICA INCLUIDOS

2.1. CONCEPTOS DE DINÁMICA ESTRUCTURAL

Desde el punto de vista de la ingeniería sísmica, el tema central de la dinámica es estudiar y entender la vibración de una estructura cuando está sujeta a una fuerza lateral u horizontal, o a un movimiento sísmico en su base (Chopra, 2001). Esto se hace a través del estudio de un modelo, que es la representación matemática de la estructura real y que adopta las propiedades la misma.

Particularmente para el estudio de las vibraciones el modelo recibe el nombre de *Oscilador* pues la estructura al ser sometida a una fuerza lateral efectúa un movimiento oscilante a manera de un péndulo.

Dentro de la dinámica estructural hay dos tipos de osciladores: de un grado de libertad (1GL) y de varios grados de libertad (VGL).

Desde el punto de vista dinámico, los grados de libertad que interesan son aquéllos en los que se consideran fuerzas generalizadas de inercia; es decir, fuerzas iguales a masa por aceleración. Por ejemplo en la figura 2.1 se muestra un marco plano con los 12 grados de libertad (lineales y angulares); sin embargo, para reducir el número de grados de libertad se considera que las fuerzas de inercia importantes son las que generan las masas m1 y m2 al moverse lateralmente; entonces en dinámica se habla de un sistema reducido de 2 grados de libertad, que son los desplazamientos laterales 1 y 2. (Bazan y Meli, *et. al* 1983)

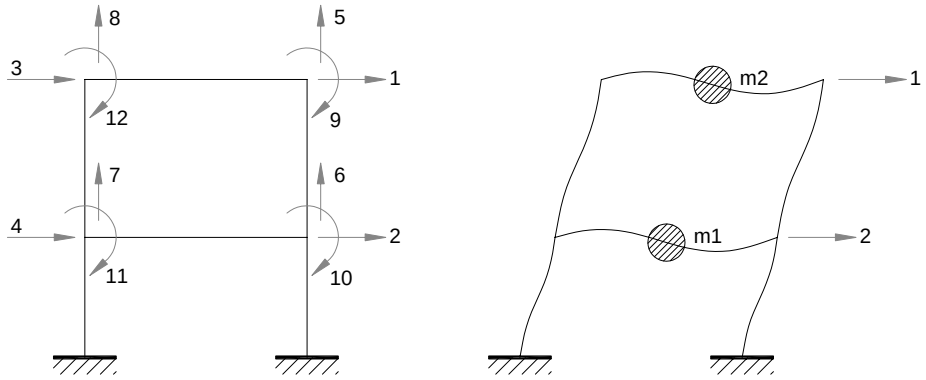


Figura 2.1 Grados de libertad

La respuesta dinámica de una estructura consiste en determinar el movimiento, velocidad y aceleraciones de su masa cuando está sometida a una fuerza lateral o a un movimiento sísmico en su base.

La respuesta dinámica depende de la magnitud y duración de la excitación, de las propiedades dinámicas de la estructura (masa, rigidez, frecuencia de vibrar y amortiguamiento) y de las características de los depósitos del suelo donde está cimentada. (Bazan y Meli, *et. al* 1983)

Las cargas gravitatorias que actúan sobre las estructuras son fuerzas estáticas, las cuales son independientes del tiempo; en cambio las fuerzas sísmicas, por efecto de la vibración del suelo, causan una respuesta dependiente del tiempo (Bazan y Meli, *et. al* 1983).

Antes de la excitación, el oscilador permanece en estado de reposo, pero una vez que comienza a moverse se generan en él fuerzas internas de inercia (f_I), de rigidez (f_S) y de amortiguamiento (f_D) que tratan de restaurar dicho estado de reposo contrarrestando a la fuerza ejercida.

$$f_S = k \cdot x \quad (2.1)$$

$$f_D = c \cdot \dot{x} \quad (2.2)$$

$$f_I = m \cdot \ddot{x} \quad (2.3)$$

Al hacer el equilibrio de estas fuerzas se genera la ecuación de equilibrio dinámico, que es la representación matemática del sistema. Para cada tipo de oscilador (1GL o VGL) la ecuación de equilibrio dinámico es diferente, aunque el planteamiento es el mismo.

2.2. OSCILADORES DE 1GL

El oscilador de un grado de libertad (ver figura 2.2a) es el modelo que representan a estructuras simples, es decir, que todo su peso o masa se considera concentrada en un sólo punto y está sostenida por un resorte (rigidez lateral "k" de la estructura).

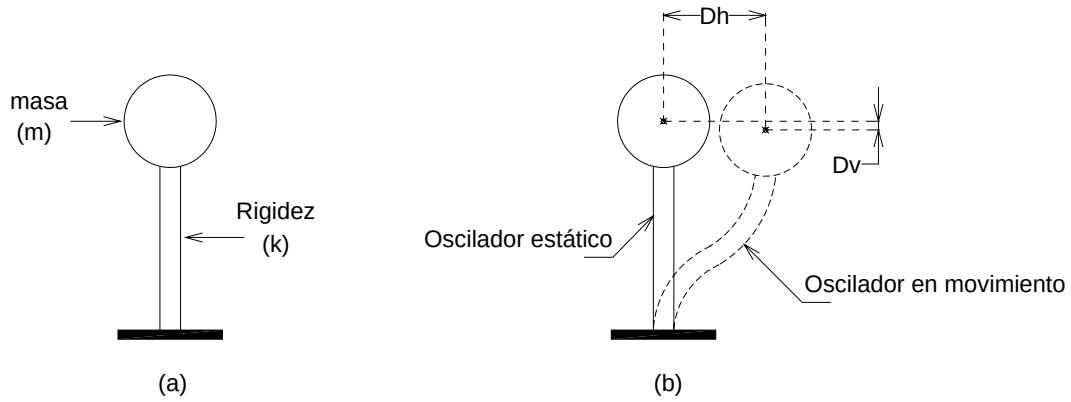


Figura 2.2 (a) Oscilador de un grado de libertad, (b) Desplazamientos del oscilador

Los grados de libertad de un oscilador son las direcciones en que la masa se puede desplazar. En la figura 2.2b se muestra el movimiento de un oscilador hacia el lado derecho; observe que la masa se desplaza en dos direcciones: horizontal (Dh) y vertical (Dv) medidas con respecto al centro de la masa.

Bajo la definición anterior (fuerzas de inercia generalizadas), este oscilador puede aproximarse como de un grado de libertad, ya que los desplazamiento verticales (Dv) son muy pequeños comparados con los horizontales (Dh); por tal motivo se desprecian quedando el oscilador de un sólo grado de libertad (1GL).

2.2.1. Ecuación de equilibrio

El equilibrio de fuerzas (ecuación 2.1 a 2.3) cuando no existe excitación alguna en los osciladores queda escrita de la siguiente manera:

$$f_I + f_D + f_S = 0 \quad \text{.....(2.4)}$$

Un caso particular de esta ecuación es cuando el sistema está sujeto a una aceleración en su base $\ddot{x}_s(t)$. Al introducir dicha aceleración a la ecuación 2.3 y a su vez sustituyendo las ecuaciones 2.1 a 2.3 en la ecuación 2.4 se obtiene la ecuación 2.5, la cual gobierna el movimiento de un oscilador de 1GL (figura 2.2a).

$$m \cdot (\ddot{x}(t) + \ddot{x}_s(t)) + c \cdot \dot{x}(t) + k \cdot x(t) = 0 \quad \text{.....(2.5)}$$

Dividiendo la ecuación 2.5 entre la masa y pasando el término de la aceleración del suelo $\ddot{x}_s(t)$ al lado derecho, se obtiene la ecuación 2.6

$$\ddot{x}(t) + \frac{c}{m} \cdot \dot{x}(t) + \frac{k}{m} \cdot x(t) = -\ddot{x}_s(t) \quad \text{.....(2.6)}$$

Definiendo las siguientes propiedades

$$\Omega^2 = \frac{k}{m} \quad \text{.....(2.7)}$$

$$\xi = \frac{c}{c_{cr}} \quad \text{.....(2.8)}$$

$$c_{cr} = 2\sqrt{k \cdot m} \quad \text{.....(2.9)}$$

Y sustituyendo las ecuaciones 2.7 a 2.9 en 2.6, la ecuación de equilibrio dinámico se muestra como:

$$\ddot{x}(t) + 2 \cdot \xi \cdot \Omega \cdot \dot{x}(t) + \Omega^2 \cdot x(t) = -\ddot{x}_s(t) \quad \text{.....(2.10)}$$

Esta última es la ecuación del equilibrio dinámico para un oscilador de 1GL cuando está sometido a una aceleración en su base. Es una ecuación diferencial de 2do grado, pues en ella intervienen la primera y segunda derivada de x, (velocidad y aceleración) con respecto al tiempo.

2.3. OSCILADORES DE VGL

Las estructuras no siempre pueden modelarse dinámicamente empleando un oscilador de un grado de libertad, y en general, es necesario modelar las estructuras como sistemas de varios grados de libertad. (Paz, 1992)

Los osciladores de varios grados de libertad (figura 2.3) representan a estructuras que su peso o masa no se puede concentrar en un sólo punto, sino que se necesario considerar un modelo de varias masas concentradas y resortes.

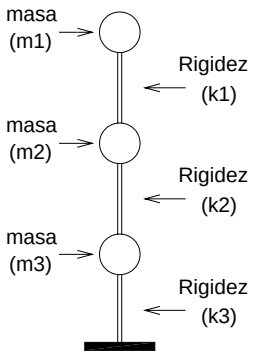


Figura 2.3 Oscilador de varios grados de libertad

En edificios es usualmente aceptable suponer que las masas están concentradas en los niveles de los pisos y que las fuerzas de inercia importantes son sólo las laterales.

El oscilador, cuando está sujeto a excitaciones que producen desplazamientos horizontales, tiene características similares a las de una viga en voladizo deformada solamente por el esfuerzo cortante. (Paz, 1992)

En este trabajo, un oscilador es de varios grados de libertad cuando el número de masas concentradas que tiene es de 2 o más. Al igual que los osciladores de 1 GL sólo se consideran los desplazamientos horizontales (Dh); por ello es posible identificar el número de grados de libertad del oscilador a través del número de masas con las que cuente.

2.3.1. Ecuación de equilibrio

De igual manera que los osciladores de 1GL, la ecuación de equilibrio dinámico para un oscilador de VGL parte del equilibrio de las fuerzas internas de la estructura (ecuaciones 2.1 a 2.3). Sólo que en esta parte las propiedades del oscilador están dadas de forma matricial.

Las fuerzas en los elementos elásticos se pueden expresar como el producto de la matriz de rigidez lateral [k] por los desplazamientos laterales, es decir:

$$[F_S] = [k] \cdot \{x\} \quad \text{.....(2.11)}$$

De manera análoga las fuerzas de amortiguamiento viscoso se pueden expresar como el producto de una matriz de amortiguamiento por las velocidades, o sea como:

$$[F_D] = [c] \cdot \{\dot{x}\} \quad \text{.....(2.12)}$$

Así mismo las fuerzas de inercia del sistema quedan como:

$$[F_I] = [m] \cdot \{\ddot{x}\} \quad \text{.....(2.12)}$$

De manera análoga a los osciladores de 1GL, la ecuación del equilibrio dinámico para osciladores de VGL se puede escribir en forma matricial como:

$$[m]\cdot\{\ddot{x}(t)\}+[c]\cdot\{\dot{x}(t)\}+[k]\cdot x(t)=-[m]\cdot\{\ddot{x}_S(t)\}\quad\text{.....(2.13)}$$

2.4. TEMAS INVOLUCRADOS

Los temas incluidos en el programa son las respuestas dinámicas de los osciladores de 1GL y VGL (con propiedades definidas) ante diferentes tipos de fuerzas o cargas horizontales, por ejemplo:

- ✓ Vibración libre
- ✓ Carga impulsiva
- ✓ Carga armónica
- ✓ Carga triangular
- ✓ Carga general

Dependiendo de las propiedades que se le proporcione a los osciladores, éstos pueden ser amortiguados o no amortiguados. Además, están considerados como sistemas elástico-lineales, es decir que las propiedades de los osciladores no se modifican en ningún momento.

El programa, a partir de la solución de las ecuaciones del equilibrio dinámico permite estudiar temas como:

- ✓ Resonancia
- ✓ Deformación máxima
- ✓ Espectros de respuesta elásticos
- ✓ Fuerza cortante
- ✓ Frecuencias y modos de vibrar
- ✓ Respuestas modales
- ✓ Espectros de piso

Para conocer la respuesta dinámica de cada oscilador (1GL y VGL) es necesario resolver las ecuaciones 2.10 y 2.13; para ello se usó el método conocido como el de las ocho constantes, el cual se describe más adelante en el capítulo correspondiente a los algoritmos.

CAPÍTULO 3

CONCEPTOS DE PROGRAMACIÓN

3.1. PROGRAMAS DE COMPUTADORA

Un programa es un conjunto de instrucciones secuenciales que definen la realización de una acción en la computadora. Existen diversos tipos, pero podemos dividirlos en forma general de la siguiente manera:

- Software de sistemas.- Son indispensables para que el sistema operativo funcione. Son los programas que hacen posible el uso de la computadora, al darle las instrucciones de partida y para que reconozca todos sus accesorios y puertos.

- Software de aplicación.- Son creados para llevar a cabo tareas de un tema en específico como por ejemplo: procesadores de textos, hojas de cálculos, juegos, programas de animación, etc.

Para crear un programa, se necesitan de los lenguajes de programación. Un lenguaje de programación es el conjunto de reglas sintácticas y semánticas, que nos indica cómo se debe escribir las instrucciones de un programa, para que la computadora las pueda ejecutar.

La computadora es capaz de ejecutar un programa escrito en un lenguaje de programación fijo, llamado lenguaje máquina, pero como éste es muy difícil de escribir para una persona, se utilizan los lenguajes ensambladores de alto nivel, como C, C++, Java, Visual Basic, etc. y el conjunto de instrucciones escritas de manera secuencial en un lenguaje de programación establecido se denomina código fuente; éste es convertido a un lenguaje máquina por un compilador.

3.1.1. Programas de aplicación

A través de un software de aplicación el usuario se comunica con la computadora para llevar a cabo tareas en particular. El programa tiene secuencias e instrucciones internas (algoritmos) escritas en un lenguaje de programación (código fuente) que sirven para poder recolectar los datos del usuario y posteriormente indicar a la computadora que ejecute las acciones y muestre el resultado obtenido.

Las aplicaciones son creadas para facilitar cálculos y procesos que el usuario necesite llevar a cabo con cierta rapidez. En la ingeniería como en muchas otras áreas los programas ayudan a realizar tareas repetitivas de manera más rápida de lo que lo haría una persona con su calculadora.

Los programas de aplicación han ido evolucionando al paso del tiempo debido principalmente a que los usuarios necesitan herramientas más complejas y procesos más potentes para realizar sus tareas. En un inicio los programas eran sólo aplicaciones para los sistemas operativos sin interfaz gráfica (que estrictamente era software basado en texto y operado por comandos, ver figura 3.1); los programas se ejecutaban al teclear su nombre en la línea de comandos y el usuario introducía los datos (escritos en la línea de comandos) que el programa necesitaba para funcionar.

Los programas tuvieron un cambio radical cuando aparecieron los sistemas operativos gráficos, (Windows, Linux, Macintosh Os etc.), pues éstos incluyeron por primera vez objetos gráficos tales como ventanas, botones, textos, imágenes de gran resolución, y permitieron el uso de otro tipo de hardware como el Mouse.

Figura 3.1. Editor de líneas de comando

De la misma manera en que los programas fueron cambiando, los desarrolladores fueron creando programas cada vez más amigables y vistosos para el usuario, basados en gráficos. En la actualidad los programas de aplicación se muestran como ventanas dentro del sistema operativo. Aún existen programas que se ejecutan desde la línea de comandos, pero no es el caso del presente trabajo.

3.2. INTERFAZ GRÁFICA DE USUARIO

Las aplicaciones en la actualidad se caracterizan principalmente por tener una interfaz gráfica de usuario ("GUI" por sus siglas en ingles Graphical User Interface) que les permite interactuar con el usuario. La interfaz, que es el enlace gráfico entre el usuario y el programa, está conformada por diversos controles como formularios, menús, barras de herramientas, botones, etiquetas, cuadros de texto, listas de datos, gráficas, etc.

Por medio de la interfaz gráfica el usuario introduce datos al programa, para que éste ordene una serie de procesos a la computadora, la cual recibe las órdenes y las ejecuta. Al termino del proceso la computadora indica al programa los resultados obtenidos, y el programa a través nuevamente de la interfaz muestra al usuario cuál fue el resultado de las operaciones realizadas.

Cada programa tiene una interfaz de usuario diferente, diseñadas acorde a la tarea que van a realizar pero básicamente todos tienen la forma de una ventana.

3.2.1. Controles

Son todas aquellas formas gráficas que permiten al usuario interactuar con el programa. Existe una gran variedad de controles para una aplicación. Cada lenguaje de programación incluye sus propios controles, que en algunos casos son los mismos aunque con diferentes nombres. El lenguaje de programación usado para este trabajo fue Visual Studio .Net, de aquí que los controles que a continuación se describen pertenecen a este lenguaje.

- Formulario (*Form*)

Es el control más importante de la GUI porque es el único lugar donde se pueden colocar los demás controles; si una aplicación no tiene al menos un formulario simplemente no podrá haber interfaz gráfica de usuario. Por lo general el formulario es una ventana con el mismo aspecto que las contenidas en el sistema operativo Windows (ver figura 3.2)

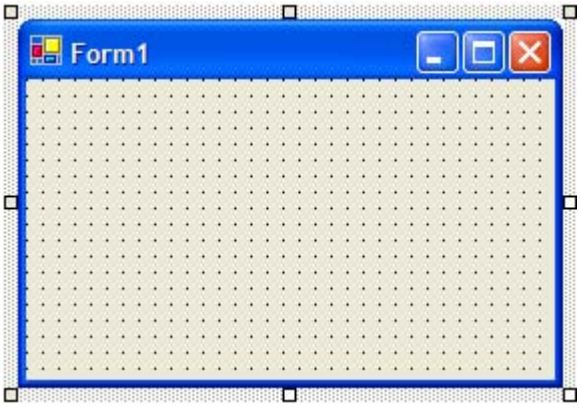


Figura 3.2 Formulario visto en modo diseño en el lenguaje de programación

- Botón (*Button*)

Es tal vez el control más conocido después del formulario, pues los botones fueron uno de los primeros controles que surgieron en la interfaz gráfica. Generalmente son de forma rectangular (figura 3.3a), pero pueden tener cualquier forma geométrica y tamaño, por ejemplo los programas que reproducen música o video tienen en general botones de forma circular (figura 3.3b). Tiene la característica que cuando se hace clic sobre ellos, parecería que se oprime de verdad.

(a) (b)

Figura 3.3 Botones

- Etiqueta (*Label*)

Las etiquetas son controles que permiten escribir texto con formato definido, y colocarlo en cualquier lugar de la interfaz gráfica. La etiqueta es un control que no se percibe visualmente, sólo su contenido que en este caso es el texto.

- Cuadro de imagen (*PictureBox*)

Así como la etiqueta permite ingresar textos a la interfaz gráfica, el *PictureBox* permite colocar imágenes dentro de la interfaz de usuario y de la misma manera este control no se percibe visualmente sólo su contenido.

- Barras de herramientas (*Tool Strip*)

Las barras son controles de forma rectangular alargadas que se encuentran generalmente en las orillas de las ventanas. Está compuesto por botones de forma cuadrada representados con imágenes.

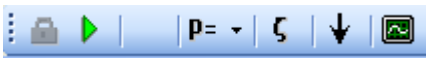


Figura 3.4 Barra de iconos o herramientas

- Barras de menús (Menú *Strip*)

Son similares a las barras de iconos, sólo que en éstas los botones son rectangulares y están representados por textos que al hacer clic en ellos se despliegan menús que son cuadros con varias opciones.

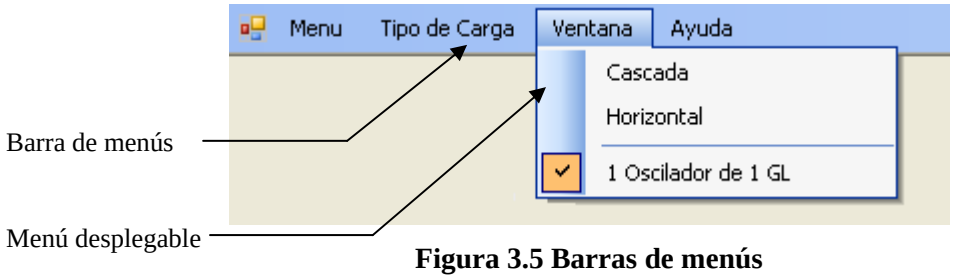


Figura 3.5 Barras de menús

- Menú contextual (*Context Menu Strip*)

Son menús en forma de ayuda emergente que aparecen al hacer clic derecho sobre un control.

- Cuadros de texto (*TextBox*)

Son controles rectangulares en los cuales se pueden escribir datos a través del teclado. Se usan principalmente para ingresar valores al programa.

Figura 3.6 Cuadros de texto y etiquetas

- Cuadros numéricos (*Numeric Up down*)

Son controles con la misma forma que los cuadros de texto, pero presentan una diferencia: que sólo se puede ingresan valores del tipo numérico y además muestran dos flechas en la parte derecha del control para incrementar o disminuir el valor.

Figura 3.7 Cuadro de texto numérico

- Cuadros de texto desplegable (*ComboBox*)

Este control combina las propiedades de un cuadro de texto y menús desplegables, muestra un texto que es una de las opciones del menú que se despliega cuando se hace clic en la flecha que tiene en su lado derecho. En ocasiones las opciones del menú pueden incrementarse con tan sólo escribir nuevos textos en el control, pero en otras, las opciones que se muestran son fijas y sólo se puede seleccionar de entre las predefinidas en el control.

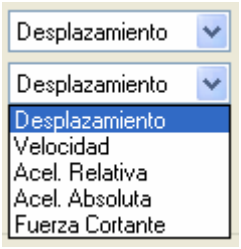


Figura 3.8 Cuadros de texto desplegable

- Cuadros de lista (*ListBox*)

Control de forma rectangular, de dimensiones fijas que muestra una lista de datos que se pueden seleccionar al hacer clic sobre ellos. Cuando el número de datos rebasan el tamaño del control, aparecen barras de desplazamiento en los extremos izquierdo e inferior del control.

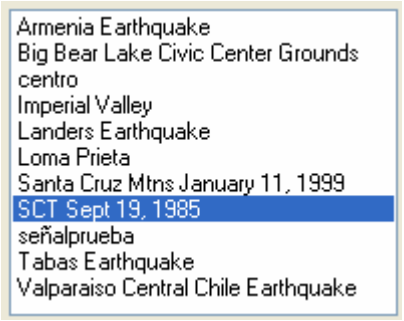


Figura 3.9 Cuadros lista

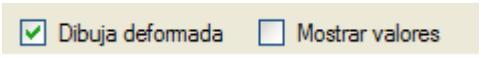
- Cajas de texto (*RichTextBox*)

Permite ingresar texto a la interfaz gráfica de usuario. A diferencia del control etiqueta (*label*) está permitido manipular su contenido; se usa principalmente para abrir archivos de texto, al crear una copia del archivo y mostrar su contenido en el control. También se puede ingresar texto escribiendo directamente sobre él. Su tamaño es ajustable y su apariencia es como el área de trabajo de un editor de texto.

Figura 3.10 *RichTextBox* mostrando el contenido de un archivo de registro sísmico

- Casillas de verificación (*CheckBox*)

Las casillas de verificación son pequeños recuadros con un texto delante de ellos que es el nombre de la opción a la que hace referencia; es posible seleccionar varios controles de este tipo a la vez. Al hacer clic sobre ellos se activa o desactiva mediante la aparición de una "palomita" sobre el recuadro, el control ya está predeterminado de esta manera.



(a) (b)

Figura 3.11 Casillas de verificación (a) Activada y (b) Desactivada

- Botones de opción (*OptionBox*)

Son muy parecidos a las casillas de verificación sólo que en este control el recuadro aparece de forma circular y cuando está activado se rellena de algún color. Sirven para activar o desactivar opciones pero a diferencia de los *Check Box* en un grupo de estos controles sólo se puede seleccionar una opción (ver figura 3.12).

- Grupo de opciones (*GroupBox*)

Su finalidad es crear y organizar a un conjunto de controles dentro de la interfaz gráfica, al que se le denomina sub grupo. Dentro de la interfaz gráfica todos los controles forman parte del grupo principal de controles, pero en algunos casos se requiere establecer sub grupos, por ejemplo para los controles de tipo *OptionBox*, esto se logra al crear un *GroupBox* y agregar los controles dentro de él.

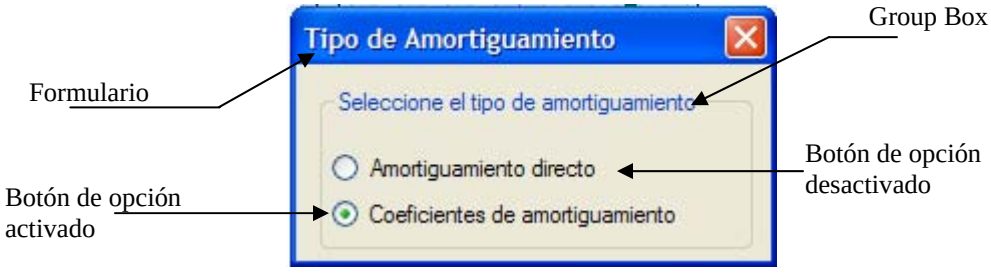


Figura 3.12 Grupo de opciones con controles del tipo *OptionBox*

- Tabla de datos (*DataGridView*)

Es una tabla con una cuadricula en su interior (figura 3.13) muy parecida a las hojas de Excel, y al igual que éstas, permite ingresar valores de cualquier tipo, seleccionar las columnas y filas, copiar su contenido al portapapeles de Windows, etc.

Osc	T(s)	Xhi	Xo	Vo
1	2	0.05	0	0
2	4	0.05	0	0
3	6	0.05	0	0

Figura 3.13 Control del tipo Tabla

- Barras de desplazamiento (*ScrollBar*)

Éstas aparecen comúnmente en los controles que su contenido rebasa las dimensiones del control. La barra asociada a un control permite desplazarse a lo largo y ancho de éste. No obstante, puede colocarse en cualquier parte de la interfaz gráfica y no estar asociadas a ningún control; son útiles en diferentes casos, por ejemplo, si se requiere establecer una relación con algún tipo de escala numérica.

3.3. CODIGO FUENTE

Es el conjunto de sentencias e instrucciones que forman al programa; se escribe por medio de líneas de texto que luego serán convertidas en lenguaje máquina por un compilador. El código fuente es la parte esencial del programa, pues todos sus componentes son creados por medio de líneas de código.

En Visual Studio, como en muchos programas ensambladores, existe un editor (figura 3.14) en el que se crea el código, línea por línea, siguiendo las reglas de sintaxis de cada programa ensamblador.

En el editor de código está escrito, en un lenguaje de programación, cada parte que conforma al programa, desde la interfaz gráfica hasta el procedimiento para obtener la solución al problema (algoritmo).

Como ya se mencionó, la interfaz de usuario es la parte gráfica del programa, pero aun esta parte es necesaria crearla a través de líneas de código; todas las propiedades (tamaño, ubicación, color, texto, valores, etc.) de cada control son definidas de esta manera pues incluso todos los controles tienen que convertirse al lenguaje máquina.

En los inicios de la programación con interfaz gráfica definir cada control directamente en el editor de código era demasiado complejo para los desarrolladores de programas, pero hoy en día los lenguajes ensambladores crean el código de manera automática para cada control una vez que se va formando la interfaz gráfica; esto ahorra mucho tiempo y trabajo a los desarrolladores.

Para formar la interfaz de usuario en Visual Studio, basta con arrastrar el control deseado desde la barra de controles hasta el formulario principal, y ubicarlos en la ventana hasta dar forma a la interfaz gráfica; al mismo tiempo el programa ensamblador va creando el código fuente en el editor de código.

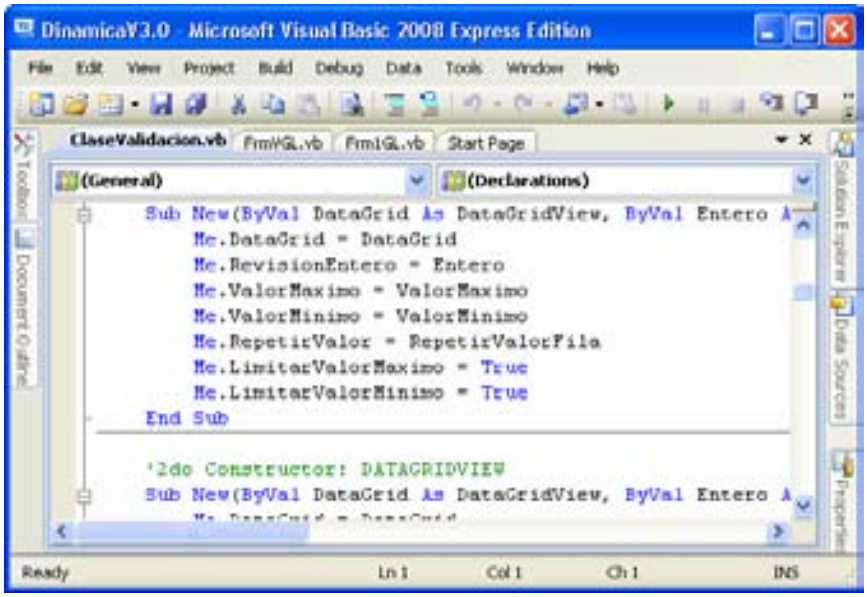


Figura 3.14 Editor de código del programa Visual Basic .Net 2008

Si bien es cierto que los programas ensambladores generan el código de la interfaz gráfica, el desarrollador debe escribir (en un lenguaje de programación) cuál es la relación entre los controles de la interfaz y el comportamiento que cada uno tendrá dentro del programa. Además, debe escribir el código fuente para el algoritmo que da la solución al problema.

Cuando el usuario interactúa con el programa, lo hace a través de la interfaz gráfica de usuario, y por lo general el código fuente de los programas de aplicación no está disponible para el usuario, que muchas veces no le interesa como fue escrito el programa sino la utilidad que tenga éste para resolver su problema.

3.3.1. Algoritmo

Es una secuencia de instrucciones cuyo objetivo es obtener la solución del problema para el que es creado el programa. No hay una única forma de resolver el problema, por lo que la idea de partida y el desarrollo del método para llegar a la solución son fundamentales para lograrlo.

Saber cuál es el algoritmo óptimo para la solución de nuestro problema puede definirse por medio de dos aspectos: un algoritmo es mejor cuanto menos tarde en resolver un problema, o bien, es mejor mientras menos memoria necesite.

El espacio en memoria, en general, tiene menos interés. El tiempo es un recurso mucho más valioso que el espacio en memoria. Así que desde este punto de vista el mejor algoritmo será aquél que resuelva el problema más rápidamente.

3.3.2. Formas de programar

Para programar es fundamental saber abstraer y descomponer los problemas en otros más pequeños, anticiparse y prever todos los posibles casos que ocurrirán. Así como los programas de aplicación han evolucionando, la forma de programar también lo ha hecho, que dicho sea de paso los programas han mejorado gracias a la forma de programar.

A continuación se hace una breve descripción de la evolución de la forma de programar, partiendo de la programación estructurada hasta la programación dirigida por eventos.

3.3.2.1. Programación Estructurada

Como ya se dijo el código fuente está escrito en líneas de texto siguiendo las reglas y estilo de un lenguaje de programación establecido. La programación estructurada surgió a finales de los años 60 y consiste en colocar las líneas de código de manera secuencial, es decir, la escritura de una instrucción debajo de otra, y el modo de procesar el programa será ejecutando las instrucciones línea por línea de arriba hacia abajo.

Para poder hacer una programación estructurada es necesario el uso de tres estructuras lógicas de control:

- Secuencia: Sucesión simple de las instrucciones, una debajo de la otra. Las instrucciones de un programa son ejecutadas en el mismo orden en que ellas aparecen en el programa.

- Selección: División condicional de las operaciones. Es la elección entre dos instrucciones tomando la decisión en base al resultado de evaluar una condición (falsa o verdadera).

- Interacción: Repetición de una operación mientras se cumple una condición

Estos tres tipos de estructuras lógicas de control pueden ser combinados para producir programas que manejen cualquier tarea de procesamiento de información. La estructura del programa mediante esta forma es clara puesto que las instrucciones están ligadas y relacionadas entre sí.

Una característica importante en un programa estructurado es que puede ser leído en secuencia, desde el comienzo hasta el final sin perder la continuidad de la tarea que cumple, contrario de lo que ocurre con otras formas de programación.

El principal inconveniente de este método de programación, es que se obtiene un único bloque de programa, que cuando el código fuente se hace demasiado grande puede resultar problemático su manejo. Para solucionar este problema se creó la programación modular.

3.3.2.2. Programación modular

Consiste en dividir al programa en módulos, para que sea más fácil de manejar y revisar. Cada modulo desempeña una acción especifica para el correcto funcionamiento del programa global. De las varias tareas que debe realizar un programa para cumplir con su objetivo, un módulo realizará una de dichas tareas.

Cuando el código fuente de un programa es demasiado grande, es conveniente crear módulos, para que el problema original sea divido en problemas más pequeños. Los módulos son creados dentro del mismo código fuente del programa y sólo pueden ser aprovechados por el código fuente de la misma aplicación.

Hoy en día los programas de aplicación son mucho más ambiciosos que las necesidades de programación existentes en los años 60, principalmente debido a las aplicaciones gráficas, por lo que las técnicas de programación estructurada y modular ya no son suficientes.

3.3.2.3. Programación Orientada a objetos (POO)

La programación orientada a objetos es una de las herramientas más poderosas con las que cuenta la programación. Básicamente consiste en crear partes de código que pueden ser reutilizables en cualquier parte del código fuente de un programa e incluso por diferentes aplicaciones; es parecida a la programación modular pero mucho más poderosa.

Durante años, los programadores se habían dedicado a construir aplicaciones muy parecidas que resolvían una y otra vez los mismos problemas, de manera que, cuando se creaba un programa el desarrollador tenía que volver a escribir líneas de código para problemas que probablemente ya se habían resuelto pero que no podía reutilizar. Para resolver esto y hacer que los avances de los programadores pudieran ser utilizados por otros desarrolladores se creó la POO.

De esta manera la programación avanzó a pasos agigantados, pues cada vez que un desarrollador creaba un programa podía implementar en su aplicación partes de código que alguien más ya se había tomado la molestia de desarrollar.

Un claro ejemplo de esto es la interfaz gráfica de usuario; los desarrolladores crearon (por medio de la POO), el código fuente para los controles, definiendo para éstos sus características y propiedades. Así, cada vez que un desarrollador crea una aplicación con interfaz gráfica, éste ya no se detiene para programar un control, sólo lo incorpora a su aplicación.

Su dominación fue consolidada gracias al auge de las Interfaces gráficas de usuario, para las cuales la programación orientada a objetos está particularmente bien adaptada. Su uso se popularizó a principios de la década de 1990. Actualmente son muchos los lenguajes de programación que soportan la orientación a objetos.

La programación orientada a objetos (POO) es un paradigma de programación que usa clases, objetos y sus interacciones para diseñar aplicaciones de computadora. Está basado en varias técnicas, incluyendo abstracción y herencia, entre otras.

- Clase

Es un modulo de código que contiene procedimientos y operaciones asociadas a un objetivo. Una clase tiene en un código fuente oculto que puede ser estructurado y modulado (maneras de programar que ya se han mencionado), y que puede ser pequeño o de gran tamaño, complejo o simple.

- Objeto

Es la manera de incorporar la clase a una aplicación. El objeto adopta las propiedades, atributos, y métodos de la clase cuando el desarrollador define al objeto, es decir se crea una instancia de esa clase.

- Abstracción

Es un concepto muy importante introducido por la POO y se puede definir como la capacidad de agregar una clase sin preocuparse de los detalles internos. En un programa es suficiente conocer que un procedimiento dado realiza una tarea específica. El cómo se realiza la tarea no es importante; mientras el procedimiento sea fiable se puede utilizar sin tener que conocer cómo funciona su interior.

- Herencia

Las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Una clase puede pasar a otras sus propiedades y métodos, y éstas a otras y así sucesivamente. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. Esto hace que la POO sea una forma de programar muy poderosa.

Para entender un poco más estos conceptos, se muestra el siguiente ejemplo:

La clase *Respuesta1GL.dll*, es un conjunto de sentencias y procedimientos que calculan la respuesta dinámica de un oscilador de 1GL ante una excitación en su base.

Para incorporar la clase *Respuesta1GL.dll* a un programa desarrollado en Visual Basic.Net, se declara un objeto, en este caso *ObjCalculo* (código fuente 3.1); a través de este objeto se puede utilizar la clase mencionada dentro del programa, introduciendo los parámetros requeridos por la clase (periodo T, amortiguamiento ξ, condiciones iniciales Xo y Vo, señal de excitación, número de puntos de la señal y el intervalo de tiempo).

```
Imports CalculoRes1GL.Respuesta1GL

Dim ObjCalculo As New CalculoRes1GL.Respuesta1GL(T,ξ,Xo,Vo,SeñalExitación,Numpuntos,dt)

ObjCalculo.Resulatdos(RX1, RV1, RAr1, RAa1)
```

Código fuente 3.1 Uso de la clase en el programa

La abstracción de la clase queda de manifiesto, pues note que para usar la clase no es necesario conocer su contenido ni cómo calcula la respuesta dinámica; sólo es necesario saber que los resultados que arroja son correctos.

Ésta clase puede ser usada, si así se requiriera, para crear otras clases más sofisticada, por ejemplo una que calcule la respuesta de un oscilador de varios grados de libertad; a esta propiedad de relacionar las clases se le llama herencia.

Las clases son tan diversas, pueden ser desde procedimientos que arrojan resultados de un cálculo hasta el dibujo y comportamiento de un control gráfico.

3.3.2.4. Programación dirigida por eventos

Un evento es un suceso en el sistema (tal como una interacción del usuario con la máquina, o un mensaje enviado por un objeto). Un evento se puede definir como la reacción que desencadena un objeto, es decir la acción que genera. Por ejemplo, en el caso del control botón (que vimos que es un objeto de la clase *button*), la acción de hacer clic sobre él se denomina *evento click*.

Mencionamos anteriormente que en los programas que sólo contienen una programación estructurada y modular, son de forma secuenciales, es decir que se ejecutan línea por línea desde el principio hasta el final (de arriba abajo); en el proceso, el programa sólo se detiene cuando encuentra una línea de código que indica que el usuario debe introducir un valor, después el programa sigue hasta terminar.

La programación dirigida por eventos cambia la secuencia de ejecución del programa, es decir, el programa no realiza acción alguna hasta que se genera un evento (cualquiera que éste sea). Por ejemplo hacer clic en un botón, pasar el Mouse sobre una gráfica, escribir texto en un TextBox, etc. Hasta que se genere un evento el programa ejecuta la parte de código que tiene programada, no importa en que parte del código fuente se encuentre.

Aunque el proceso del programa sea dirigido por eventos, las partes de código contenido en los eventos siguen siendo secuenciales.

La programación por eventos no es posible sin la programación orientada a objetos.

3.3.3. Errores

Los errores son acciones inesperadas en un programa las cuales no se habían contemplado. El desarrollador debe programar todos los posibles casos para que el programa se ejecute adecuadamente. Se puede clasificar los errores en dos tipos: errores en ejecución y errores lógicos.

- Errores de ejecución

Éstos ocurren cuando la aplicación detecta acciones para las cuales no está programada. Por ejemplo, cuando en un control Textbox relacionado para introducir sólo datos numéricos recibe un texto, si el programa no contempla este caso, el programa generara un error pues normalmente el valor numérico que se esperaba es parte de una solución matemática, y en cambio si recibe una palabra el programa no puede ejecutar la ecuación pues la palabra no tiene un valor numérico.

No programar la solución a estos problemas deriva en la finalización inmediata del programa; en el peor de los casos, la aplicación se cierra perdiendo la información que se procese en ese momento.

- Errores lógicos

Son errores de programación que hacen que el código fuente del programa produzca resultados erróneos en la solución del problema. Estos errores son de mucha mayor importancia que los anteriores, pues éstos no son detectados a menos que se calibren los resultados del programa con una fuente confiable.

CAPÍTULO 4

DESCRIPCIÓN DEL SOFTWARE

4.1. CARACTERÍSTICAS DEL PROGRAMA

4.1.1. Objetivo del programa

Ser una herramienta para la enseñanza y el aprendizaje de la dinámica estructural, mostrando gráficamente la respuesta numérica del problema en cuestión y colocando un proceso de animación de la misma.

No tiene como finalidad suplir al profesor y a la bibliografía del tema; el programa surge como una ayuda tanto para el profesor como para los alumnos y es imprescindible que se tome como tal.

4.1.2. Usuarios

Es un programa para estudiantes y maestros de la carrera de ingeniería civil en el área de estructuras. Está dirigido principalmente a los profesores que imparten las materias de Dinámica Estructural e Ingeniería Sísmica y a los alumnos que las cursan. A los maestros les puede ayudar a explicar conceptos y mostrar ejemplos de manera rápida y clara. Para los estudiantes es una herramienta que les auxilia a asimilar de manera más rápida los conceptos del tema y facilita la reproducción de ejemplos mostrados en la bibliografía del tema.

No obstante, puede beneficiarse del programa cualquier persona con conocimientos de estructuras que necesite una herramienta para aprender dinámica estructural.

4.1.3. Tipo de software

Puede definirse como un software de enseñanza y práctica. Se concibe como un software de enseñanza ya que a través de su uso en clase el estudiante enriquece sus conceptos de dinámica. También, es un software de práctica ya que el usuario puede usar el programa fuera de clase y analizar problemas de la bibliografía que le permitan adquirir una mejor comprensión y avanzar a su propio ritmo.

El alcance de este software es meramente educativo y debe tomarse como tal; es posible que en algunas ocasiones se use para obtener las respuestas numéricas de algún problema especifico pero se considera que existen otros programas con un mejor desempeño y con mucho más alcance que el aquí presentado para esos casos.

4.1.4. Lenguaje de programación

Para elaborar el programa se utilizó el entorno de desarrollo de Microsoft Visual Studio .Net (Microsoft Corporation, 2008b), que es un excelente entorno de programación personalizable que contiene herramientas para construir, con rapidez y eficiencia, sólidos programas para Microsoft Windows. Visual Studio maneja varios lenguajes de programación entre ellos: Visual Basic, Visual C++, Visual C#.

A partir de la introducción en el mercado de la versión 2005 de Visual Studio, Microsoft publicó lo que se conoce como ediciones Express de distintos programas. Las versiones Express son versiones gratuitas pero limitadas.

El lenguaje de programación utilizado para crear el programa Dinámica V3.1 fue Visual Basic 2008 Express Edition que es una versión limitada de Visual Studio. Esta versión permite sólo programar en VB.NET, y además limita los tipos de proyectos que se pueden desarrollar.

Visual Basic .NET (VB.NET) es un lenguaje de programación orientado a objetos que se puede considerar una evolución de Visual Basic implementada sobre el Framework .NET. Como pasa con todos los lenguajes de programación basados en .NET, los programas escritos en VB.NET requieren el Framework .NET para ejecutarse.

.NET Framework es toda una nueva arquitectura tecnológica, desarrollada por Microsoft para la creación y distribución del software como un servicio. Es la que proporciona la infraestructura para crear aplicaciones y el entorno de ejecución para las mismas. El Net Framework es el conjunto de bibliotecas de clases que hacen posible ejecutar las aplicaciones.

4.1.5. Requisitos del sistema

Para que el programa Dinámica V3.1 pueda ser ejecutado el equipo debe tener los siguientes requisitos como mínimo:

Hardware

Procesador Pentium 4, 2.80 Ghz o superior
Memoria en RAM: 512 Mb
Espacio libre en disco duro: 3 Mb
Resolución minima en pantalla de 1024 x 768 pixeles

Software

Windows Xp Service Pack 2 o superior
Net Framework 3.5

4.2. INTERFAZ GRÁFICA DE USUARIO

La GUI del programa se conforma de varias ventanas y controles dentro de éstas. Tiene ventanas principales y secundarias; en las primeras se muestran las animaciones de las gráficas de respuesta y los osciladores. Las ventanas secundarias o auxiliares sirven para poder generar opciones alternas al proceso expuesto en las ventanas principales.

Se ha diseñado para que los usuarios interactúen con la aplicación y les resulte fácil encontrar y utilizar las funciones del programa. El aspecto general de la aplicación se agilizó de manera que presenta a los usuarios opciones muy sencillas para obtener las respuestas dinámicas y las animaciones de forma que sólo tienen que elegir un tipo de carga y hacer clic para comenzar la animación sin necesidad de complicados cuadros de diálogo.

El programa permite obtener los resultados de una forma rápida y sencilla. Además, el software aprovecha las excelentes características de la biblioteca .NET, que hace que la aplicación pueda obtener resultados más rápidamente.

La función principal del programa es mostrar las animaciones de los osciladores de uno y varios grados de libertad al ser sometidos a una excitación. Para ello, primero se deben definir las propiedades de los

osciladores, después calcular la respuesta dinámica al introducir una carga y posteriormente dibujar las gráficas correspondientes a la carga y respuestas del oscilador.

El programa calcula la respuesta y dibuja las gráficas correspondientes de manera automática cada vez que el usuario cambia el tipo de carga y/o modifica las propiedades del oscilador.

Una vez dibujada la respuesta del oscilador, el usuario puede ver la animación con tan sólo dar clic en el botón Play. Además, la animación se puede controlar mediante los botones Pausa o Stop.

Los controles de las ventanas principales se encuentran siempre visibles en la parte derecha de las ventanas. A través de éstos se manejan principalmente las animaciones y las propiedades de los osciladores.

En general el conjunto controles que se usaron tienen un comportamiento sencillo dentro del programa pues cada vez que el usuario use un control puede ver rápidamente cuál es la función de éste en la interfaz gráfica. Además, el usuario puede consultar el archivo de ayuda que incluye el programa para conocer el uso de algún control en específico.

En total el programa tiene 17 ventanas distribuidas de la siguiente manera:

- 3 Ventanas principales
- 8 ventanas de uso común
- 6 Ventanas dependientes de cada tema

Las ventanas principales están distribuidas de la siguiente manera:

1. Ventana general: Es la ventana más importante del programa, contiene a todas las demás ventanas de la aplicación.

2. Ventana para osciladores de 1GL: muestra las animaciones para los osciladores de 1 GL.

3. Ventana para oscilador de VGL: muestra las animaciones del oscilador de VGL.

Las ventanas secundarias se pueden dividir en dos grupos: ventanas de uso común y ventanas particulares para cada tema. En la siguiente tabla se muestra la relación de las ventanas que hay en el programa.

Tabla 4.1 Clasificación de las ventanas por tema

Tema	Ventana común	Ventana particulares
Osciladores de 1GL	Presentación Senoide Abrir Archivo Base de Datos Ayuda Acerca de Apariencia Parámetros de Espectros	Datos de respuesta Definir K Datos de repuesta de Espectros
Oscilador de VGL		Datos de Respuesta Tabla de propiedades Tipo de amortiguamiento

4.2.1. Ventana de presentación

La ventana de presentación (figura 4.1) aparece siempre que se inicia el programa; su aparición es por pocos segundos y su objetivo es mostrar el nombre del programa, información de su versión y derechos de autor.

Su fondo es de color rojo y en ella aparecen también la imagen de un oscilador de 1GL (logotipo del programa y un registro de un acelerograma). Su tamaño es pequeño: 180 x 320 pixeles.

Figura 4.1 Ventana de presentación

4.2.2. Ventana general

La ventana general (figura 4.2) es la más importante del programa pues es la contenedora de las demás ventanas de la aplicación. Sus pocos controles son los botones para manipular el estado de la ventana y una barra con un sólo menú desplegable, a través de éste se controlan las otras ventanas principales.

Esta ventana aparece después de la ventana de presentación del programa, cuando el usuario inicia la aplicación. Su apariencia es rectangular con fondo gris.

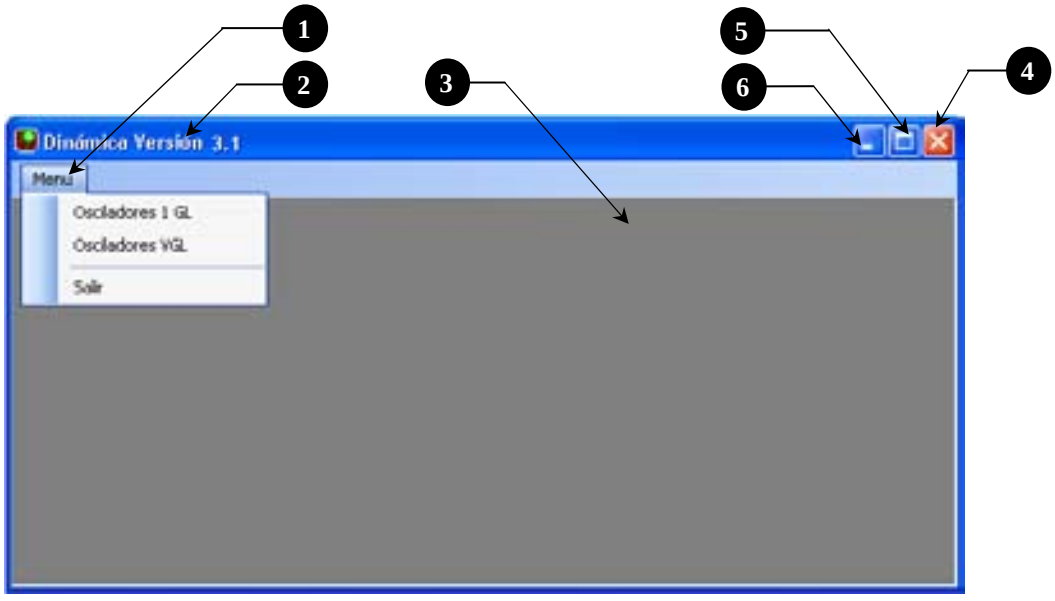


Figura 4.2 Ventana general

#	Control	Descripción
1	Barra de Menús	Controla el estado de las ventanas principales
	Osciladores 1GL	Abre la ventana de trabajo para los osciladores de 1GL
	Osciladores VGL	Abre la ventana de trabajo para el oscilador de VGL
	Salir	Cierra la aplicación del programa y todas sus ventanas abiertas
2	Título	Barra de título que muestra el nombre del programa
3	Área de trabajo	Espacio de color gris donde se abrirán las demás ventanas
4	Botón Cerrar	Cierra el programa y todas sus ventanas abiertas
5	Botón Maximizar	Maximiza la ventana al tamaño de la pantalla
6	Botón Minimizar	Minimiza la ventana

4.2.2.1. Barra de Menús

Cuando cualquiera de las otras ventanas principales está abierta, la barra de menús cambia agregándose tres opciones más (figura 4.3 b, c y d) al menú. La barra se encuentra siempre fija en la parte superior de la ventana general debajo del título de la misma.

Cuando en la aplicación se cierran las ventanas principales, la barra de menús regresa a su estado original, es decir con sólo una opción (figura 4.2, #1).

Al hacer clic sobre los nombres de la barra de menús, se despliegan opciones referentes al título de cada uno (figura 4.3a, b, c y d).

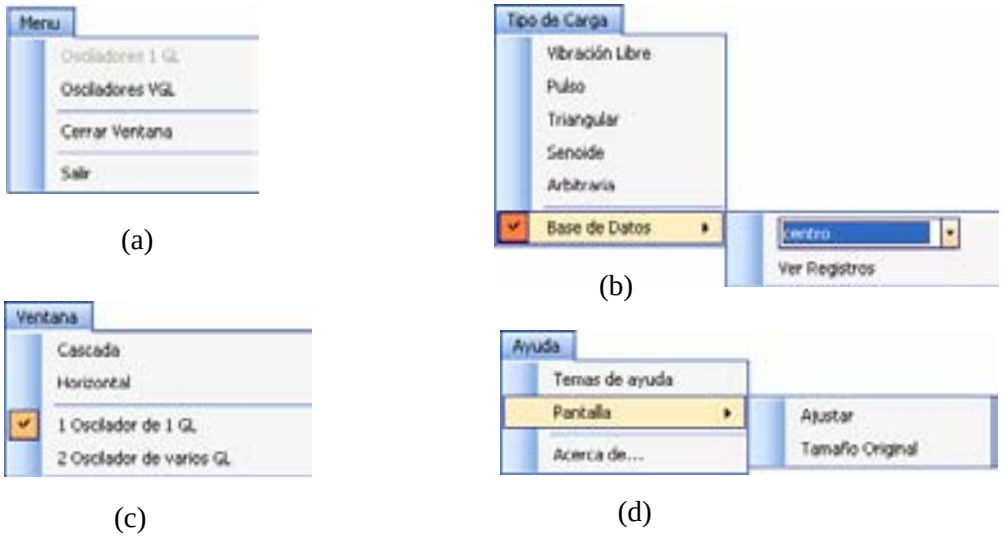
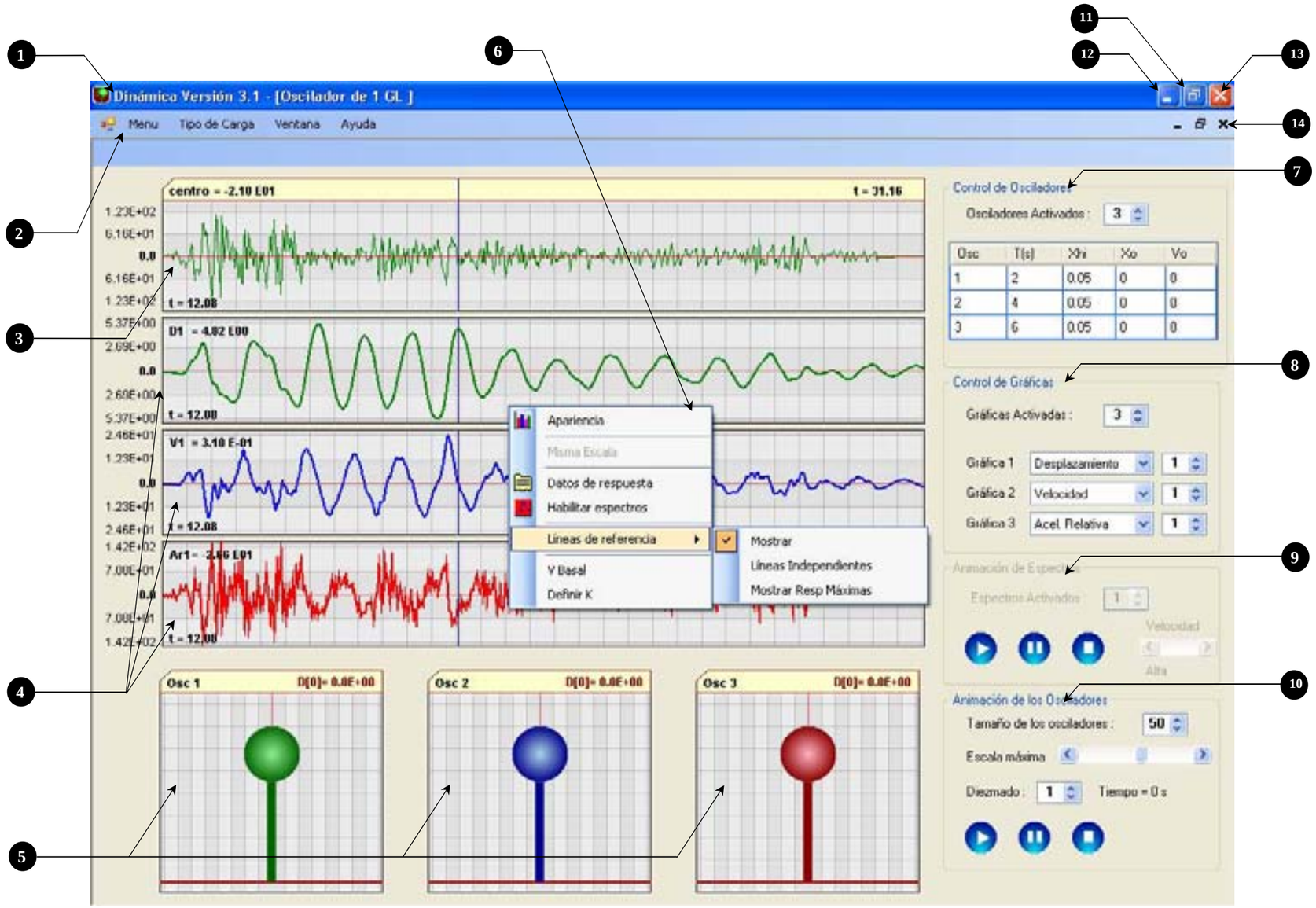


Figura 4.3 Menús desplegables en la barra de menús

#	Menú desplegable	Descripción
(a)	Menú	Controla el estado de las ventanas principales
	Osciladores 1GL	Abre la ventana de trabajo para los osciladores de 1GL (ver sección 4.2.3)
	Osciladores VGL	Abre la ventana de trabajo para el oscilador de VGL (ver sección 4.2.4)
	Cerrar Ventana	Cierra la ventana que esté activa en ese momento
	Salir	Cierra el programa y todas sus ventanas abiertas
(b)	Tipo de Carga	Aplica un tipo de carga a los osciladores
	Vibración Libre	Aplica una carga con amplitud constante de valor cero y modifica las condiciones iniciales de los osciladores para que sean diferentes de cero.
	Pulso	Carga un impulso con duración de 20 s, con amplitud constante igual a cero excepto en un sólo punto, cuando el tiempo vale 1 s la amplitud vale 1 unidad.
	Triangular	Aplica una carga que incrementa linealmente de 0 a 5 unidades de amplitud en un tiempo de 0 a 5 s y diminuye su amplitud a 0 unidades cuando la duración llega a los 10 s
	Senoide	Abre la ventana de carga senoidal (ver sección 4.2.5.1)
	Arbitraria	Abre la ventana de lectura de archivo de datos (ver sección 4.2.5.2)
	Base de datos	Opción para manipular las señales de la base de datos del programa
	Seleccionar registros	El usuario puede seleccionar una de las señales que se encuentran en la base de datos, al hacerlo el programa carga la señal automáticamente
	Ver registros	Abre la ventana de base de datos (ver sección 4.2.5.3)
(c)	Ventana	Manipula la forma de ver las ventanas principales dentro de la ventana general
	Cascada	Coloca a las ventanas en cascada
	Horizontal	Coloca a las ventanas en forma horizontal
	1 Oscilador de 1GL	Activa la ventana de osciladores de 1GL
	2 Oscilador de VGL	Activa la ventana de osciladores de VGL
(d)	Ayuda	Contiene opciones que proporcionan ayuda a el usuario (ver sección 4.2.5.4)
	Temas de ayuda	Abre la ventana de ayuda HTML
	Pantalla	Opciones para ajustar las gráficas al tamaño de la pantalla
	Ajustar	Aumenta de tamaño las gráficas para que ocupen más espacio dentro de la pantalla cuando la resolución de esta última sea mayor de las dimensiones mínimas requeridas (1024 x 768 píxeles)
	Tamaño Original	Regresa las dimensiones de las gráficas al tamaño inicial, ajustadas a las dimensiones mínimas de pantalla
	Acerca de…	Abre la ventana *Acerca de Dinámica V3.1* (ver sección 4.2.5.5)

4.2.3. Ventana de Osciladores de 1GL

El objetivo de esta ventana es mostrar de manera gráfica la respuesta dinámica de los osciladores de 1GL cuando están sujetos a la acción de una fuerza lateral o a un movimiento sísmico en su base, y a su vez mostrar la animación de su movimiento durante el tiempo que se ejerce la fuerza.



1	Título de la ventana principal y [nombre de la ventana secundaria]
2	Menú principal
3	Gráfica de carga
4	Gráficas de respuestas
5	Osciladores de 1 GL
6	Menú emergente de las gráficas de respuesta
7	Controles para indicar las propiedades de los osciladores
8	Controles para mostrar las gráficas de respuesta
9	Controles para la animación de los espectros de respuesta
10	Controles para la animación de los osciladores
11	Botón para maximizar la ventana principal
12	Botón para minimizar la ventana principal
13	Botón para cerrar la aplicación
14	Botones para controlar el tamaño de la ventana de Osciladores de 1GL dentro de la ventana principal

Figura 4.4 Ventana Osciladores de 1GL

La GUI está diseñada para que el usuario vea en un sólo espacio de trabajo las gráficas de las respuestas, los dibujos de osciladores de 1GL y los controles correspondientes a éstos.

Los controles de la ventana se pueden dividir en dos grupos: controles para mostrar las gráficas y controles para manipular las gráficas. Todos ellos se distribuyen de la siguiente manera:

En una franja del extremo derecho de la ventana su ubican los controles para manipular las gráficas (indicar propiedades de los osciladores, definir el tipo de gráfica de respuesta que se desee ver y mostrar las animaciones). La franja de controles a su vez está divida en 4 grupos (figura 4.4, #7, #8, #9 y #10):

- ✓ *Control de osciladores*: Se utilizan para modificar las propiedades y definir el número de los osciladores

- ✓ *Control de gráficas*: Controlan el número y tipo de gráficas de respuesta que el usuario desee ver en pantalla

- ✓ *Animación de espectros y osciladores*: Sirven para manipular la animación de espectros y osciladores, respectivamente

La mayor parte del área de trabajo de la ventana es ocupada por controles del tipo *PictureBox*, que son los que muestran las gráficas correspondientes a la fuerza ejercida y a la respuesta dinámica de los osciladores (figura 4.4, #3, #4). Dentro de cada gráfica de respuesta existe un menú emergente (figura 4.4, #6) que proporciona, entre otras, opciones para cambiar la apariencia de las gráficas.

El programa tienen la capacidad de mostrar de 2 a 4 gráficas, distribuidas de la siguiente manera: 1 gráfica para la fuerza ejercida y de 1 a 3 gráficas de respuesta. Esto dependerá de cómo el usuario manipule los controles contenidos en el subgrupo *Control de gráficas*.

En la parte baja de la pantalla se ubican otros controles del tipo *PictureBox*, que contienen los dibujos de los osciladores (figura 4.4, #5) y al igual que las gráficas de respuesta el programa puede mostrar de 1 a 3, dependiendo de los osciladores que se definan en el subgrupo *Control de Osciladores*.

Adicionalmente a todos estos controles, la ventana general sigue mostrando su barra de menús pero con tres opciones más: Tipo de carga, Ventana y Ayuda (figura 4.4, #2).

En general, el uso de los controles no es complicado; no obstante, se incorporó al programa un archivo de ayuda que contiene la descripción del funcionamiento de cada control; dicho archivo sirve como una guía para el usuario.

La GUI de los osciladores de 1 GL es una de las ventanas principales, pues en ésta se desarrolla uno de los dos temas principales del programa. Pero a pesar de que sea una ventana principal del programa, está ubicada dentro de la ventana general y por lo tanto su estado dependerá de esta última.

La ventana necesita dimensiones mínimas de 1024 x 768 píxeles para que los controles de la misma se aprecien con claridad. La ventana no tiene dimensiones máximas, es decir, se ajusta al tamaño de la ventana general. Cuando las dimensiones de la ventana sean mayores a las mínimas requeridas, el usuario puede indicar al programa que ajuste el tamaño de las gráficas dentro de la pantalla.

El usuario puede abrir y cerrar la ventana dentro de la aplicación cuantas veces sea necesario, pero si la aplicación se cierra sin duda todas sus ventanas se cerraran.

Cada vez que se abra la ventana, todos los controles antes mencionados aparecen bloqueados, a excepción de la barra de menús. Esto es porque el programa necesita *forzosamente* una carga (opción que se encuentra en la barra de menús) para calcular la respuesta dinámica de los osciladores de 1GL y así dibujar las gráficas de las respuestas.

Todos los controles se habilitan cuando el usuario selecciona un tipo de carga. Al hacerlo, automáticamente el programa calcula y dibuja las respuestas dinámicas de los osciladores que están definidos por default. Una vez seleccionado un tipo de carga el usuario puede modificar las opciones de los controles que antes aparecían bloqueados. Por ejemplo: definir el número de osciladores (de 1 a 3), cambiar sus propiedades, establecer el número y tipo de gráficas de respuestas que desea ver, y ejecutar la animación.

Sólo hay una ventana de osciladores de 1GL; desafortunadamente la aplicación no puede abrir más ventanas de este tipo al mismo tiempo debido a que requeriría mucha más memoria de la computadora. Lo que sí es posible hacer, es tener abiertas las tres ventanas principales en la aplicación al mismo tiempo.

Uno de los puntos importantes para los usuarios de un programa es la pérdida de información ante el cierre repentino de la aplicación. Dinámica V3.1 tiene una característica muy importante en este sentido: no es un programa que guarde automáticamente la información procesada sino que su poder radica en la velocidad de sus cálculos. Por ejemplo si por alguna razón la aplicación terminara inesperadamente el usuario perdería la información que estaba viendo en ese momento, pero puede recuperarla rápidamente con pocos clics sobre los controles, sin perder mucho tiempo en el proceso.

El programa tiene opciones que le permiten al usuario exportar los valores numéricos de las respuestas en archivos de texto; incluso puede copiar al portapapeles los valores de las columnas seleccionadas en las tablas de respuestas numéricas que se incluyen en la aplicación. El programa no tiene la capacidad de imprimir ningún tipo de archivo (texto, imagen, tablas) directamente a una impresora.

A continuación se describen cada una de las partes que conforman a la ventana.

4.2.3.1. Control de osciladores

Por medio de este grupo de controles (figura 4.5) el usuario define el número de osciladores de 1GL, mostrados en la ventana principal, proporcionando sus propiedades: periodo, amortiguamiento y condiciones iniciales (desplazamiento y velocidad) de cada uno. Es el primer grupo de controles en la franja del extremo derecho de la ventana de osciladores de 1GL y su tamaño no es modificable

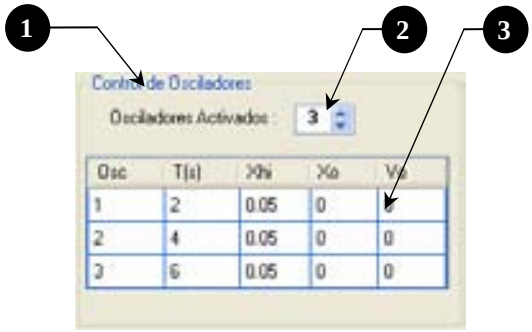


Figura 4.5 Control de osciladores

#	Control	Descripción
1	*GroupBox*	Conjunta a los controles dentro de un mismo espacio
2	*Numeric Up Down*	Define el número de osciladores de 1GL que habrá en la ventana. Automáticamente al modificar el número de osciladores en el control, el programa redibuja los osciladores en la parte baja de la pantalla
3	*DataGrid View*	Tabla que contiene las propiedades de los osciladores de 1GL. En el encabezado de cada columna se observa el nombre de la propiedad a la que se refiere. La primera columna identifica al número del oscilador. Las filas que muestre la tabla corresponden al número de osciladores definidos.

4.2.3.2. Control de gráficas

Este grupo de controles (figura 4.6) sirven para que el usuario seleccione y muestre en pantalla la gráfica de la respuesta dinámica de un determinado oscilador.

En la parte central de la pantalla el programa dibuja las gráficas de respuesta; en total puede dibujar 4 gráficas: 1 de carga y 3 de respuesta, pero estas últimas pueden variar de 1 a 3 según defina el usuario.

Los tipos de respuesta que hay disponibles para seleccionar son: desplazamiento, velocidad, aceleración relativa y absoluta, así como también la gráfica de la fuerza cortante.

Figura 4.6 Control de gráficas

#	Control	Descripción
1	*GroupBox*	Conjunta a los controles dentro de un mismo espacio
2	*Numeric Up Down*	Define el número gráficas de respuesta que se ve en pantalla
3	*ComboBox*	Permiten seleccionar la respuesta dinámica que el usuario desee ver en pantalla
4	*Numeric Up Down*	Define a que oscilador corresponde la respuesta dinámica seleccionada en el *ComboBox*

4.2.3.3. Control Animación de espectros

Con estos controles (figura 4.7) el usuario maneja la animación de los espectros de respuesta. Puede reproducir la animación, detenerla en cualquier instante o por completo y además retardar su velocidad de movimiento.

Para utilizar este grupo de controles es necesario tener calculados los espectros, lo cual se hace a través de una opción que se encuentra en el menú emergente de las gráficas de respuesta que se describe más adelante en este mismo capítulo.

El tema de las animaciones es explicado con profundidad en el capítulo correspondiente a los algoritmos.

El grupo de controles está conformado por tres botones: Play, Stop y Pausa, similares a los que se usan en las aplicaciones que reproducen audio y video; un control de tipo *Numeric Up Down* para mostrar un número determinado (de 1 a 3) de espectros y por último un control de tipo *ScrollBar* que permite ver la reproducción de la animación un poco más lenta. Esto con el fin de que el usuario aprecie mejor la animación. La velocidad con que se ejecute la animación dependerá del procesador y capacidad de memoria con que cuente la computadora.

Figura 4.7 Controles para animación de espectros

#	Control	Descripción
1	*GroupBox*	Conjunta a los controles dentro de un mismo espacio
2	*Numeric Up Down*	Define el número de espectros de respuesta que se muestran en la ventana
3	Botones	Botones para manipular la animación; son similares a los que se usan en las aplicaciones de audio y video. De izquierda a derecha son: Play, Pausa y Stop.
4	*ScrollBar*	Hace que la animación se retarde. Tiene tres niveles de reproducción dependiendo de la ubicación de la barra de desplazamiento (extremo izquierdo, derecho y centro). La animación se reproduce con la mayor velocidad posible cuando la barra se encuentra en el extremo izquierdo; entre más se mueva la barra hacia el lado derecho más lenta será la animación.

4.2.3.4. Control Animación de los osciladores

Con estos controles (figura 4.8) el usuario maneja la animación de los osciladores, puede reproducir la animación, detenerla en cualquier instante o por completo, con los botones: Play, Pause y Stop. También puede incrementar la velocidad de animación con la opción de diezmado.

La animación de los osciladores consiste en ver en cada intervalo de tiempo (Δt) el desplazamiento del oscilador con respecto a su estado original. El tema de las animaciones es explicado con profundidad en el capítulo correspondiente a los algoritmos.

Además de manejar la animación, este grupo de controles permite aumentar las dimensiones del dibujo de las masas de los osciladores; esto sólo cambia en el dibujo ya que no altera de ninguna manera las propiedades de la masa.

El programa también permite establecer el punto máximo de desplazamiento de los osciladores, a través del control de tipo *ScrollBar*.

Los grupos de controles *Animación de espectros* y *Animación de osciladores* no pueden estar activados al mismo tiempo, es decir, cuando el usuario pueda usar los controles de este grupo no podrá usar los de otro, y viceversa; esto porque el programa está diseñado para mostrar una animación a la vez (osciladores o espectros) y no las dos al mismo tiempo.

Figura 4.8 Controles para animación de osciladores

#	Control	Descripción
1	*GroupBox*	Conjunta a los controles dentro de un mismo espacio
2	*Numeric Up Down*	Tamaño de las masas (50 a 60 píxeles) en los dibujos de los osciladores
3	*ScrollBar*	Establece hasta dónde llegará el desplazamiento máximo de los osciladores
4	*Numeric Up Down*	Diezma la animación. El programa toma un punto de cada N (valor del control) para hacer la animación de los osciladores, esto hace que la velocidad de reproducción sea mayor. El rango de valores para el diezmado es de 1 a 4.
5	Etiqueta	Cronómetro en tiempo real, para conocer la duración de la animación.
6	Botones	Botones para manipular la animación, son similares a los que se usan en las aplicaciones de audio y video. De izquierda a derecha: Play, Pausa y Stop.

4.2.3.5. Gráfica de Carga

La gráfica de carga (figura 4.9) es un control del tipo *PictureBox* y muestra la fuerza lateral (carga), seleccionada por el usuario que se ejerce sobre los osciladores. El control se encuentra fijo en la ventana ubicado en la parte superior del área de trabajo, sus dimensiones mínimas son: 120 x 670 píxeles, y sus dimensiones máximas dependerán del tamaño de la pantalla pues el usuario puede ajustar el tamaño de las gráficas al de la pantalla de su computadora.

Cada vez que el usuario abra la ventana principal, la gráfica de carga aparecerá vacía, con una leyenda en la parte superior que dice: “Seleccione un tipo de carga”. Una vez seleccionado cualquier tipo de carga, el programa la dibuja automáticamente ocupando todo el ancho del *PictureBox* y colocando los valores de amplitud correspondientes a la carga, y cambia la leyenda por el nombre de la carga seleccionada.

A diferencia de las gráficas de respuesta, el usuario no puede cambiar la apariencia de esta gráfica, pues no cuenta con un menú secundario. La línea gráfica siempre se dibujará de color verde y su cuadrícula en dirección X dependerá del valor de las gráficas de respuesta.

Es la única gráfica que tiene una pestaña ubicada en la parte superior de la misma, con el fin de que el usuario diferencie la carga de las gráficas de respuesta.

El usuario puede cambiar, cuantas veces lo desee, el tipo de carga y el programa cambiara el dibujo de la gráfica automáticamente; no es necesario que cierre la ventana principal para cargar un tipo de carga diferente.

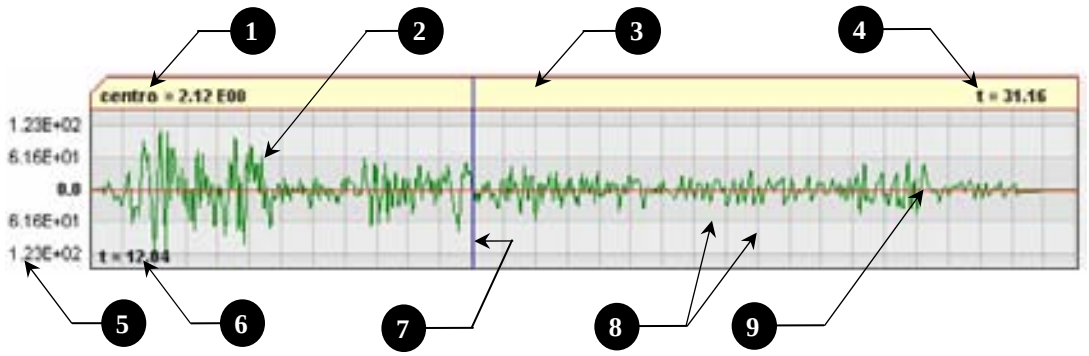


Figura 4.9 Gráfica de carga

#	Descripción
1	Nombre de la señal seguido del valor de la amplitud correspondiente a la línea de referencia vertical móvil.
2	Dibujo de la señal (color verde) escalada al tamaño del *PictureBox*
3	Pestaña de color amarillo que diferencia a la gráfica de carga de las gráficas de respuesta
4	Duración total de la señal en segundos
5	Escala de valores de amplitud

6	Valor del tiempo en segundos correspondiente a la línea de referencia vertical móvil
7	Línea de referencia vertical (color azul) que se mueve con el puntero del Mouse cuando pasa sobre la gráfica
8	Cuadrícula con fondo color gris. Las líneas verticales (dirección X) están separadas a equidistancias de segundos, por default el programa las dibuja a cada segundo pero el usuario puede cambiar este valor si así lo requiere. Las líneas horizontales (dirección Y) están a una distancia de amplitud, los valores correspondientes a éstas se muestran en la parte izquierda de la gráfica.
9	Línea horizontal de color rojo que marca la amplitud cero de la señal

4.2.3.6. Gráficas de respuestas

La gráfica de respuesta (figura 4.10) es un control del tipo *PictureBox* y muestra la respuesta dinámica (desplazamiento, velocidad o aceleraciones) de un oscilador, que es el resultado de ejercer una carga lateral al sistema de masas y resortes.

Se pueden ver en pantalla de 1 a 3 gráficas y se encuentran ubicadas debajo de la gráfica de carga; sus dimensiones mínimas son: 95 x 670 píxeles, y sus dimensiones máximas dependerán del tamaño de la pantalla pues el usuario puede ajustar el tamaño de las gráficas al de la pantalla.

Cada vez que el usuario cambie el número de gráficas que desee ver en pantalla (1, 2 o 3), el programa distribuye las gráficas de manera que éstas abarquen todo el espacio del área de trabajo. Esto se logra al aumentar o disminuir el tamaño vertical de las gráficas, según sea el caso.

Siempre que se abra la ventana principal, las gráficas de respuesta no aparecerán en el área de trabajo hasta que el usuario seleccione un tipo de carga. Una vez hecho esto, el programa calcula y dibuja automáticamente la respuesta en el *Picturebox*; además, coloca los valores de amplitud correspondientes a la respuesta.

A diferencia de la gráfica de carga, éstas no cuentan con una pestaña, y su nombre está ubicado en la parte superior izquierda dentro del fondo gris. Cuentan con un menú secundario que permiten al usuario cambiar la apariencia de las gráficas, entre otras opciones.

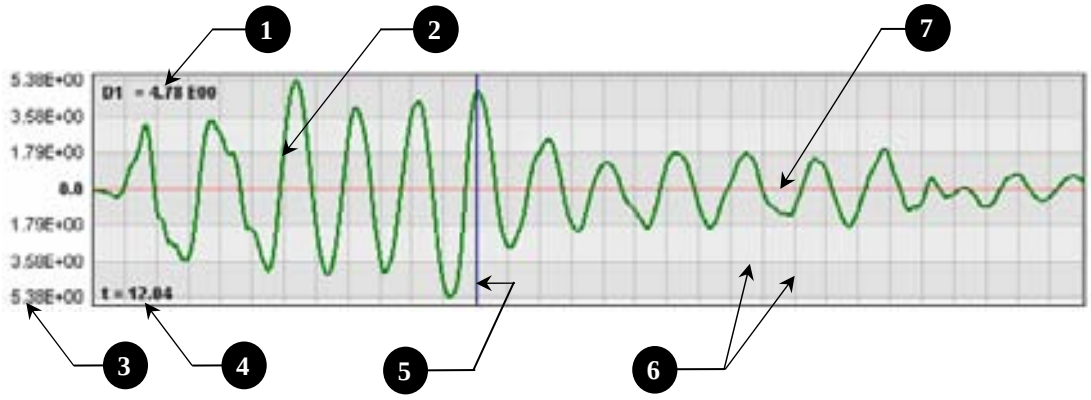


Figura 4.10 Gráfica de carga

#	Descripción
1	Nombre de la señal seguido del valor de la amplitud correspondiente a la línea de referencia vertical móvil. (D1: desplazamiento del oscilador 1)
2	Dibujo de la señal escalada al tamaño del *PictureBox*
3	Escala de valores de amplitud
4	Valor del tiempo en segundos correspondiente a la línea de referencia vertical móvil
5	Línea de referencia vertical (color azul) que se mueve con el puntero del Mouse cuando pasa sobre la gráfica. La línea se queda fija en una posición cuando se hace clic en las gráficas de respuesta, y se vuelve móvil de nuevo al hacer clic otra vez.
6	Cuadrícula con fondo color gris. Las líneas verticales (dirección X) están separadas a equidistancias de segundos, por default el programa las dibuja a cada segundo pero el usuario puede cambiar este valor si así lo requiere. Las líneas horizontales (dirección Y) están a una distancia de amplitud, los valores correspondientes a éstas se muestran en la parte izquierda de la gráfica.
7	Línea horizontal de color rojo que marca la amplitud cero de la respuesta

4.2.3.7. Menú secundario de las gráficas de respuesta

El menú secundario (figura 4.11), aparece cuando el usuario hace clic derecho en alguna de las gráficas. Contiene opciones que permiten modificar la apariencia de las gráficas, mostrar los valores de la respuesta dinámica, activar las líneas de referencia y opciones para mostrar el cortante basal.

El control se encuentra disponible en cualquier momento, excepto cuando las animaciones estén en ejecución, ya que el programa bloquea esta opción hasta que termine dicha animación.

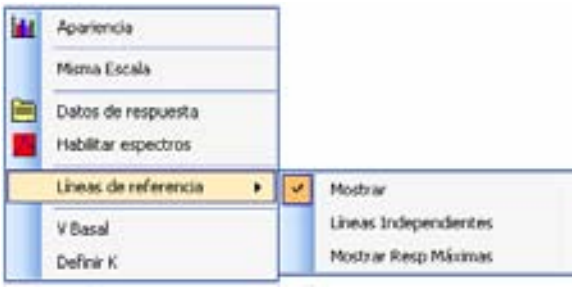


Figura 4.11 Menú secundario de las gráficas de respuesta

Opciones de Menú	Descripción
Apariencia	Abre la ventana "*Apariencia*" (ver sección 4.2.5.6)
Misma Escala	Dibuja las gráficas de respuesta a la misma escala. Esta opción se habilita cuando los tipos de las respuestas de los osciladores son iguales. El programa toma como base la gráfica que tenga la mayor respuesta y a partir de ésta dibuja las demás.

Datos de respuesta	Abre la ventana “Respuesta numérica” (ver sección 4.2.3.8)
Habilitar espectros	Abre la ventana “Espectros de respuesta” (ver sección 4.2.5.7). Cuando los espectros han sido calculados la opción cambia a *Deshabilitar espectros*. Al seleccionar esta opción desaparecen los espectros y se dibujan de nuevo los osciladores.
Líneas de referencia	Contiene opciones para manipular las líneas de referencia móviles sobre las gráficas de respuesta y carga.
Mostrar	Aparece/quita las líneas de referencia
Líneas independientes	Habilita/deshabilita que las líneas de referencia, de todas las gráficas, se muevan como una sola.
Mostrar Resp máximas	Ubica a las líneas de referencia donde se encuentra la mayor amplitud en cada gráfica.
V basal	Habilita/deshabilita la opción para que se muestre el cortante basal sobre los osciladores
Definir K	Abre la ventana “Definir k” (ver sección 4.2.3.9)

4.2.3.8. Ventana Respuesta numérica

Contiene los datos numéricos que conforman a la respuesta dinámica de cada oscilador, la fuerza cortante y los valores del método de las ocho constantes.

La ventana está diseñada para mostrar la información en dos formatos diferentes: por oscilador o por tipo de respuesta; para ello se tienen controles del tipo *OptionButton*.

El usuario puede copiar los datos (mostrados en la tabla de la ventana 4.12) al portapapeles de Windows con tan sólo seleccionarlos ya sea por celda, columnas enteras o incluso la tabla completa, y tecleando CTRL + C o haciendo clic en la opción “Copiar” del menú secundario. Además, el programa proporciona una opción para exportar todos los datos a un archivo de texto.

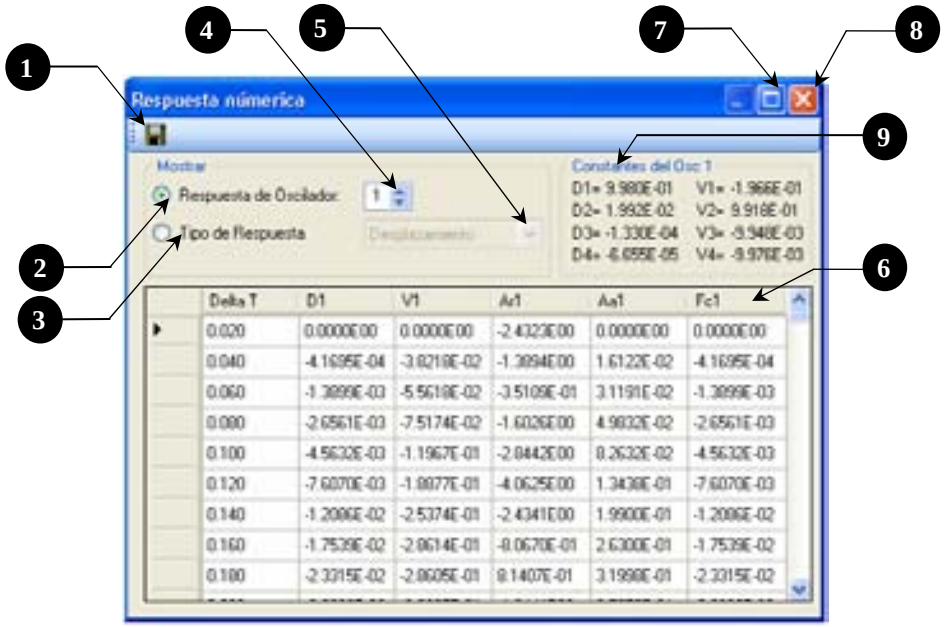


Figura 4.12 Ventana Respuesta numérica

#	Control	Descripción
1	Barra de herramientas	Contiene sólo una opción, que permite exportar todos los datos de la respuesta a un archivo de texto
2	*Option Button*	Muestra en la tabla la respuesta dinámica y la fuerza cortante del oscilador establecido en el control *Numeric Up Down*
3	*Option Button*	Muestra en la tabla la respuesta del mismo tipo (desplazamiento, velocidad o aceleraciones) de todos los osciladores al mismo tiempo
4	*Numeric Up Down*	Se habilita cuando está seleccionado el primer *Option Button*, y define a que oscilador corresponde la respuesta que se muestra en la tabla
5	*ComboBox*	Se habilita cuando está seleccionado el segundo *Option Button*, y define el tipo de respuesta que se muestra en la tabla
6	*DataGridView*	Tabla que muestra la respuesta dinámica dependiendo de la opción seleccionada.
7	Botón maximizar	Maximiza la ventana al tamaño de la pantalla
8	Botón Cerrar	Cierra la ventana
9	*GroupBox*	Muestra el valor de las ocho constantes para el oscilador definido en el control *Numeric Up Down*

4.2.3.9. Ventana Definir k

Adicional a la respuesta dinámica, el programa calcula y dibuja la gráfica de la fuerza cortante. Ésta se calcula mediante la ecuación 4.1

$$F(t) = D(t) * k \quad \text{.....(4.1)}$$

Donde:
F (t) = fuerza sobre el oscilador en un instante de tiempo t
D (t) = desplazamiento del oscilador en un instante de tiempo t
k = Rigidez del oscilador

La ecuación 4.1, es considerada como una ecuación estática, pues para conocer la fuerza sobre el oscilador en un determinado tiempo (t), sólo hay que hacer el producto de su desplazamiento en el tiempo (t) por el valor de la rigidez.

Como se mencionó anteriormente, los osciladores se consideraron con propiedades elástico- lineales, por lo tanto los valores de la rigidez y masa no cambian.

Para aplicar la ecuación 4.1, el programa determina el desplazamiento, pero no puede saber el valor de k, pues éste como el valor de la masa (m) queda implícito en el valor del periodo (T) del oscilador definido por la ecuación 4.2

$$T = 2\pi\sqrt{\frac{m}{k}} \quad \text{.....(4.2)}$$

Debido a que el usuario introduce al programa directamente el valor del periodo, no es posible determinar el valor de la rigidez k sin conocer antes el valor de la masa m, o viceversa.

Para conocer el valor de la rigidez k y aplicar la ecuación 4.1, se creó la ventana "*Definir k*" (figura 4.13), que permite al usuario ingresar el valor de la rigidez para los osciladores que tiene definidos. Por omisión, el programa coloca *k = 1* para todos los osciladores, y con éste calcula el valor de la masa de cada uno.

En la ventana hay una tabla con las propiedades del oscilador (periodo T, frecuencia w, rigidez K y masas M); la única columna de datos que se puede modificar es la rigidez y con este valor y la ecuación 4.1, es posible calcular la fuerza cortante.

La frecuencia y la masa se calculan con las ecuaciones 4.3 y 4.4 respectivamente:

$$\omega = \frac{2\pi}{T} \quad \text{.....(4.3)}$$

$$m = \left(\frac{T}{2\pi}\right)^2 k \quad \text{.....(4.4)}$$

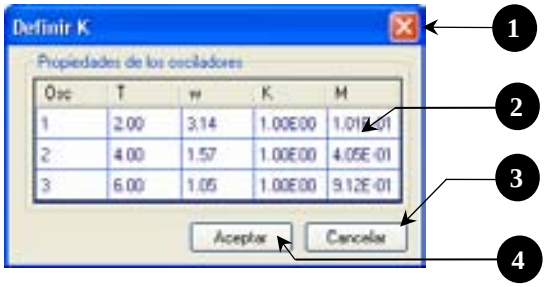


Figura 4.13 Ventana Definir k

#	Control	Descripción
1	Botón Cerrar	Cierra la ventana sin guardar los cálculos realizados
2	*Data Grid View*	Muestra las propiedades de los osciladores definidos por el usuario
	Osc	Número de oscilador
	T	Periodo en s
	w	Frecuencia en rad
	k	Rigidez del oscilador (única columna que recibe datos del usuario)
	M	Masa del oscilador
3	Botón Cancelar	Cierra la ventana sin guardar los cálculos realizados
4	Botón Aceptar	Desarrolla los cálculos e inmediatamente después cierra la ventana

4.2.3.10. Osciladores

Los osciladores de 1GL (figura 4.14) quedan representados por el sistema de masas y resortes. El control sobre el cual se dibujan es del tipo *PictureBox*.

El objetivo de este control es mostrar los desplazamientos que tiene el oscilador en cada instante de tiempo, debido a la fuerza lateral que se ejerce sobre él.

El programa puede mostrar de 1 a 3 osciladores; cada uno se diferencia del otro por el color de su masa y su resorte: verde, azul y rojo. El fondo donde están dibujados los osciladores es similar al de las gráficas de respuesta (cuadriculado con un fondo gris).

Todos los osciladores tienen las mismas características, aparece una pestaña en su parte superior que incluye el número del oscilador (ejemplo: Osc 1) y además muestra el valor del desplazamiento en un instante t, en el extremo derecho del control.

Mientras la animación no esté en ejecución, el oscilador permanecerá estático en su estado inicial, dibujado al centro del *PictureBox* (desplazamiento = 0).

Sus dimensiones son de 200 x 200 píxeles, y se ubican en la parte baja del área de trabajo.

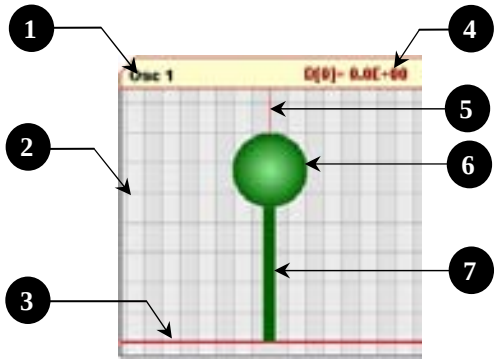


Figura 4.14 Dibujo de un oscilador

#	Descripción
1	Número del oscilador
2	Cuadrícula y fondo de color gris, en este caso las líneas verticales y horizontales no tienen valor de escala, como en las gráficas de respuesta.
3	Línea base en la que se desplanta el oscilador
4	Valor del desplazamiento y el instante de tiempo al que corresponde
5	Eje vertical (Línea color rojo) que indica el centro del oscilador y desplazamiento lateral = 0
6	Masa del oscilador
7	Columna que representa la rigidez del oscilador

4.2.3.11. Espectros de respuesta

El objetivo de la gráfica (figura 4.15) es mostrar el espectro de respuesta. A diferencia de la respuesta dinámica, los espectros no se generan automáticamente al seleccionar un tipo de carga, porque el programa necesita algunos parámetros para hacerlo.

Para indicar al programa que calcule los espectros de respuesta vea la sección 4.2.5.7.

Cuando los espectros ya han sido calculados son colocados en la misma posición que los osciladores; ciertamente se sustituyen unos por otros.

Las características de estas gráficas son similares a la de los osciladores. Tienen una pestaña en lo alto de la gráfica, y muestran una cuadrícula con fondo de color rojizo. Su tamaño es igual que los osciladores (200 x 200 píxeles). El espectro de respuesta es de color vino para que haya un contraste con el color de fondo.

La gráfica cuenta con dos líneas de referencia (horizontal y vertical) que se mueven cuando el cursor del mouse pasa sobre la gráfica. El cruce de las dos líneas siempre se desplaza a lo largo del espectro de respuesta.

Al igual que las gráficas de respuesta, este control presenta un menú secundario que se activa cuando se hace clic derecho sobre la gráfica.

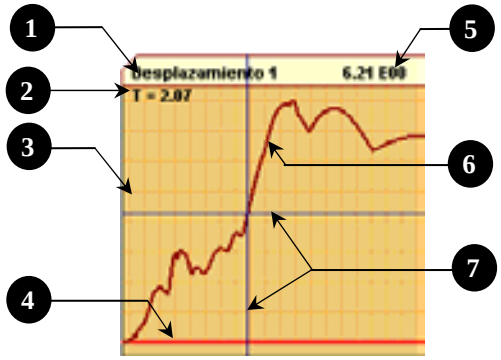


Figura 4.15 Gráfica espectros de respuesta

#	Descripción
1	Nombre del espectro
2	Valor del periodo que corresponde al línea de referencia vertical
3	Fondo del espectro, cuadriculado sin ningún valor de escala
4	Línea base que corresponde a la amplitud cero
5	Valor de la amplitud que corresponde a la línea de referencia horizontal
6	Espectro de respuesta
7	Líneas de referencia (vertical y horizontal) que se mueven con el puntero del Mouse cuando pasa sobre la gráfica. Éstos pueden permanecer estáticos si se hace clic sobre la gráfica.

4.2.3.12. Menú secundario de los espectros de respuesta

Menú secundario que aparece cuando se hace clic derecho sobre las gráficas de espectros de respuesta. Presenta sólo tres opciones, y siempre está disponible, incluso en la animación de los espectros de respuesta.

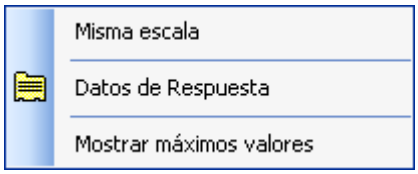


Figura 4.16 Menú secundario de los espectros de respuesta

Opciones de Menú	Descripción
Misma Escala	Dibuja los espectros de respuesta a la misma escala. Esta opción se habilita cuando los tipos de espectros son iguales. El programa toma como base el espectro que tenga la mayor respuesta y a partir de ésta dibuja los demás.
Datos de respuesta	Abre la ventana “Valores de espectros” (ver sección 4.2.3.13)
Mostrar máximos valores	Ubica a las líneas de referencia donde se encuentra la mayor amplitud en cada gráfica.

4.2.3.13. Ventana Valores de espectros

Contiene los datos numéricos que conforman a cada uno de los espectros de respuesta.

La ventana está diseñada para mostrar la información en dos formatos diferentes: por bloque de espectro y por tipo de espectro; para ello se tiene controles del tipo *Option Button*.

Un bloque de espectro es un grupo de 4 espectros de respuesta: desplazamiento, velocidad, aceleración relativa y absoluta. Para ver con más detalle esta información ver la sección 4.2.5.7.

El usuario puede copiar los datos (mostrados en la tabla de la ventana 4.17) al portapapeles de Windows con tan sólo seleccionarlos (por celda, columnas enteras o incluso la tabla completa) y oprimiendo CTRL+C; también puede copiar los datos seleccionados haciendo clic en el menú emergente “Copiar” que aparece cuando se hace clic derecho sobre la tabla.

Además, el programa proporciona una opción para exportar todos los datos a un archivo de texto.

El usuario no puede modificar los valores directamente en la tabla, sólo puede copiarlos para disponer de ellos fuera del programa.

El tamaño de la ventana no es modificable, y cuenta con un botón para controlar el estado de la ventana en la parte superior de la misma.

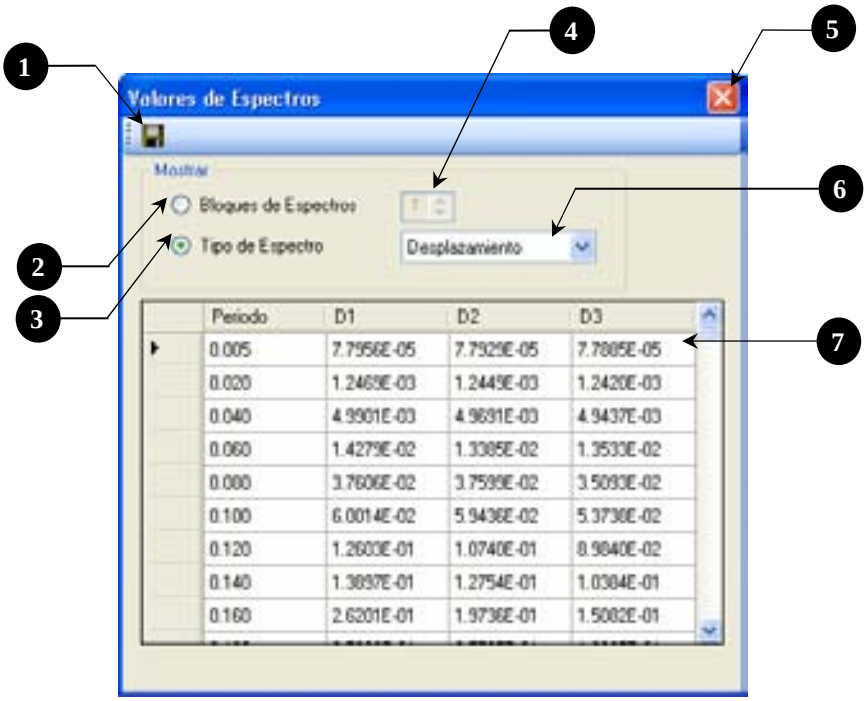


Figura 4.17 Ventana Valores de espectros

#	Control	Descripción
1	Barra de herramientas	Contiene sólo una opción, que permite exportar todos los datos de la respuesta a un archivo de texto
2	*Option Button*	Muestra en la tabla la respuesta dinámica y la fuerza cortante del oscilador establecido en el control *Numeric Up Down*
3	*Option Button*	Muestra en la tabla la respuesta del mismo tipo (desplazamiento, velocidad o aceleraciones) de todos los osciladores al mismo tiempo
4	*Numeric Up Down*	Se habilita cuando está seleccionado el primer *Option Button*, y define a que oscilador corresponde la respuesta que se muestra en la tabla
5	Botón Cerrar	Cierra la ventana
6	*ComboBox*	Se habilita cuando está seleccionado el segundo *Option Button*, y define el tipo de respuesta que se muestra en la tabla
7	*DataGridView*	Tabla que muestra la respuesta dinámica dependiendo de la opción seleccionada. La primer columna siempre corresponderá al valor del periodo

4.2.4. Ventana de Osciladores de VGL

El objetivo de esta ventana es mostrar de manera gráfica la respuesta dinámica de un oscilador de VGL cuando está sujeto a la acción de una fuerza lateral o a un movimiento sísmico en su base, y a su vez mostrar la animación de su movimiento durante el tiempo que se ejerce la fuerza.

El diseño de la GUI es similar a la ventana principal de los osciladores de 1GL, pues el usuario ve en un sólo espacio de trabajo las gráficas de las respuestas correspondientes a los grados de libertad, el dibujo del oscilador de VGL y los controles para manipular las gráficas y la animación.

Los controles de la ventana están divididos en tres grupos similares a la otra ventana principal. El primer grupo es una franja en el extremo derecho de la ventana, allí se ubican los controles para manipular las gráficas (indicar propiedades de los oscilador, seleccionar el tipo de gráfica de respuesta que se desee ver y mostrar la animación). La franja de controles a su vez está divida en 4 grupos (figura 4.18, #8, #9, #10 y #11):

- ✓ *Control de propiedades*: Se utilizan para modificar las propiedades del oscilador y definir el número de grados de libertad
- ✓ *Control de gráficas*: Controlan el número y tipo de gráficas de respuesta que el usuario desee ver en pantalla
- ✓ *Animación del oscilador*: Sirven para manipular la animación del oscilador
- ✓ *Modos de vibrar*. Muestra las deformadas modales

El segundo grupo está conformado por controles de tipo *PictureBox*, que son los que muestran las gráficas correspondientes a la fuerza ejercida y a la respuesta dinámica del oscilador en sus diferentes grados de libertad (figura 4.18, #4, #5). Éstos ocupan la mayor parte del área de trabajo de la ventana junto con el oscilador de VGL.

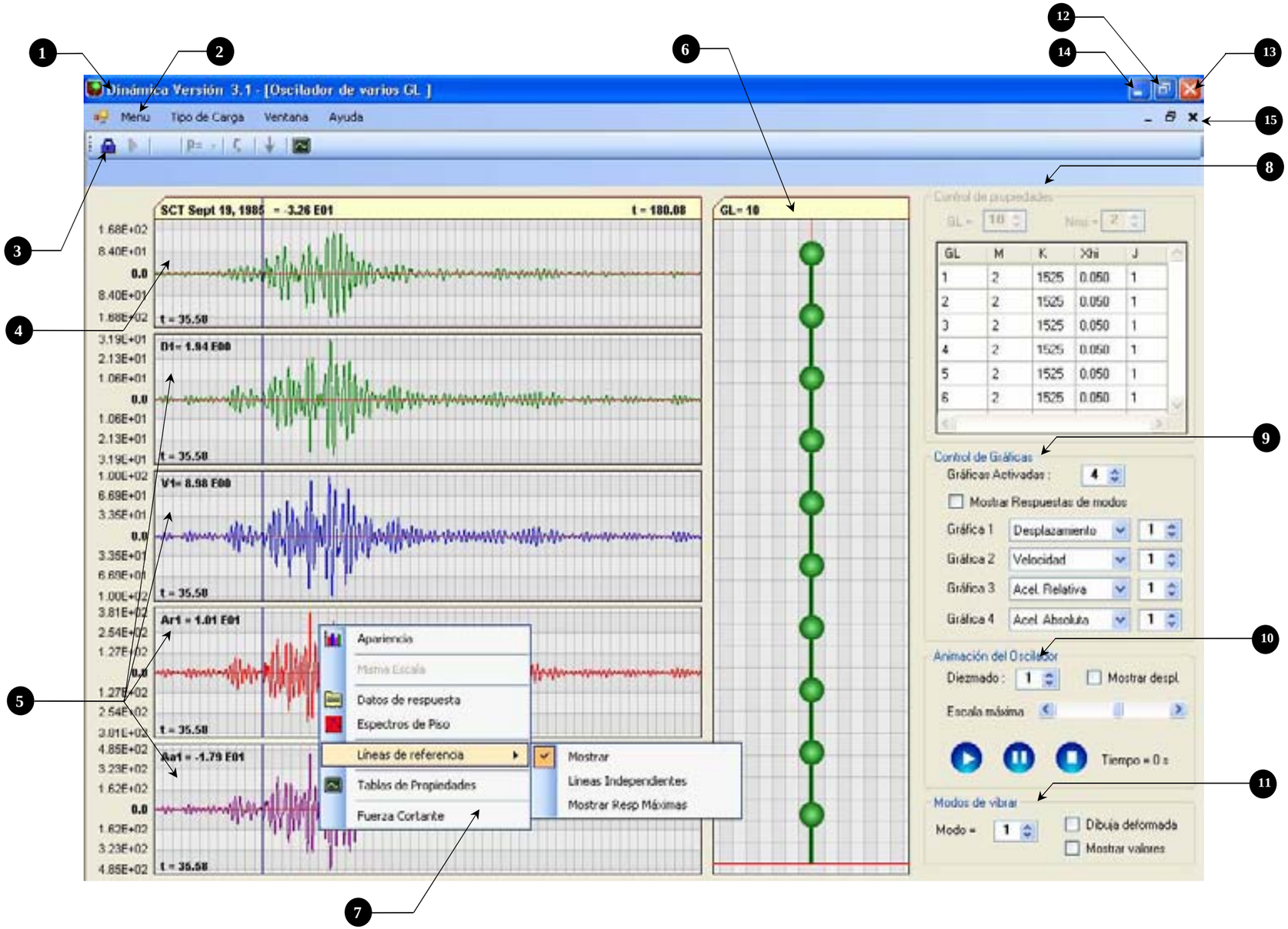
Cuando se abre la ventana aparecen 6 *PictureBox* en el área de trabajo (5 para las gráficas y 1 para el oscilador); los correspondientes a las gráficas están vacías, hasta que el usuario seleccione un tipo de carga e indique al programa que calcule la respuesta del oscilador.

El programa tienen la capacidad de mostrar de 3 a 5 gráficas, distribuidas de la siguiente manera: 1 gráfica para la fuerza ejercida y de 2 a 4 gráficas de respuesta. Esto dependerá de cómo el usuario manipule los controles contenidos en el subgrupo *Control de gráficas*.

Entre las gráficas y la franja de controles mencionada se ubica un control del tipo *PictureBox* destinado para mostrar al oscilador (figura 4.18, #6) y a diferencia de la ventana de osciladores de 1GL, esta ventana sólo tiene un oscilador. Los grados de libertad del oscilador serán establecidos por el usuario a través del *Control de propiedades*, y su rango es de 2 a 25 grados de libertad.

El tercer grupo de controles está integrado por una barra de herramientas y la barra de menús, aunque esta última pertenece a la ventana general pero con tres opciones más: Tipo de carga, Ventana y Ayuda (figura 4.18, #2).

En general, el uso de los controles no es complicado; no obstante, se incorporó al programa un archivo de ayuda que contiene la descripción del funcionamiento de cada control; dicho archivo sirve como una guía para el usuario.



1	Título de la ventana principal y [nombre de la ventana secundaria]
2	Menú principal
3	Barra de herramientas
4	Gráfica de carga
5	Gráficas de respuestas
6	Oscilador de varios grados de libertad
7	Menú emergente de las gráficas de respuesta
8	Controles para indicar las propiedades del oscilador de VGL
9	Controles para mostrar las gráficas de respuesta
10	Controles para la animación del oscilador
11	Controles para mostrar las deformadas modales
12	Botón para maximizar la ventana principal
13	Botón para cerrar la aplicación
14	Botón para minimizar la ventana principal
15	Botones para controlar el tamaño de la ventana de Osciladores de VGL dentro de la ventana principal

Figura 4.18 Ventana Osciladores de VGL

La GUI de los osciladores de VGL es una de las ventanas principales, pues en ésta se desarrolla el segundo de los dos temas principales del programa. Pero, al igual que la ventana de osciladores de 1GL está ubicada dentro de la ventana general y por lo tanto su estado dependerá de esta última.

La ventana necesita las mismas dimensiones mínimas de 1024 x 768 píxeles para que los controles de la misma se aprecien con claridad y tampoco tiene dimensiones máximas, es decir, se ajusta al tamaño de la ventana general. Cuando las dimensiones de la ventana sean mayores a las mínimas requeridas, el usuario puede indicar al programa que ajuste el tamaño de las gráficas dentro de la pantalla.

El usuario puede abrir y cerrar la ventana dentro de la aplicación cuantas veces sea necesario, pero si la aplicación se cierra sin duda todas sus ventanas se cerraran.

Cada vez que se abra la ventana algunos de los controles antes mencionados aparecen bloqueados, con excepción de la barra de menús, control de propiedades, modos de vibrar y algunas opciones de la barra de herramientas.

Para un oscilador de VGL, aun sin ejercer alguna carga, el programa calcula las siguientes propiedades:

Matriz de masas
Matriz de rigidez
Periodos y frecuencias
Modos de vibrar
Factores de participación

Debido a esto, los controles mencionados no aparecen bloqueados cada vez que se abre la ventana, pues las propiedades no dependen de un tipo de carga sino de los valores de masa y rigidez que se establezcan en el oscilador.

Cuando el usuario selecciona un tipo de carga el programa dibuja la gráfica correspondiente pero no calcula automáticamente la respuesta dinámica del oscilador como lo hace en la ventana de osciladores de 1GL. Esto se debe a que el proceso de cálculo para un oscilador de VGL es más tardado que para un oscilador de 1GL y por ello no se puede hacer automáticamente. El tiempo de cálculo depende del número de grados de libertad que tenga el oscilador.

Para conocer la respuesta dinámica del oscilador, se colocó una opción en la barra de herramientas "*Calcular respuesta*", la cual se activa cuando el usuario ha seleccionado un tipo de carga, y al hacer clic en esta opción el programa calcula la respuesta.

Una vez calculada la respuesta el usuario no puede modificar las propiedades del oscilador, hasta que desbloquee los cálculos realizados con ayuda de la opción "*Desbloquear el cálculo*" que también se encuentra en la barra de herramientas.

Cuando se calcula la respuesta, los controles que antes aparecían bloqueados se habilitan, y el usuario puede establecer el número y tipo de gráficas de respuestas que desea ver así como ejecutar la animación

Sólo hay una ventana de osciladores de VGL; desafortunadamente la aplicación no puede abrir más ventanas de este tipo al mismo tiempo debido a que requeriría mucha más memoria de la computadora.

A continuación se describen cada una de las partes que conforman a la ventana.

4.2.4.1. Barra de herramientas

La barra de herramientas (figura 4.19) es la única con la cuenta el programa; se coloca en esta ventana para proporcionar atajos al usuario además de colocar las opciones para generar la respuesta dinámica.

Es una barra fija, se muestra siempre en la parte superior de la ventana debajo de la barra de menús, sólo en esta ventana.

Cuando se abre la ventana, las dos primeras opciones (izquierda a derecha) están bloqueadas. Estas opciones son para calcular la respuesta y desbloquear el cálculo respectivamente; se activará una opción cuando el usuario seleccione cualquier tipo de carga. Las siguientes 4 opciones están habilitadas mientras no se genere el cálculo de la respuesta, a excepción de la última opción que siempre está disponible.

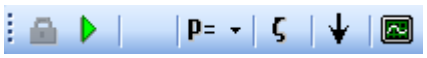


Figura 4.19 Barra de herramientas del oscilador de VGL

Opción	Nombre	Descripción
	Desbloquear	Desaparece la respuesta dinámica del oscilador de VGL
	Calcular respuesta	Calcula la respuesta del oscilador de VGL ante el tipo de carga seleccionado
	Igualar propiedades	Ayuda a que cada grado de libertad del oscilador tenga las mismas propiedades masa, rigidez, amortiguamiento, vector de forma y condiciones iniciales.
	Tipo de amortiguamiento	Abre la ventana "*Tipo de amortiguamiento*" ver sección 4.2.4.12
	Sentido de numeración	Cambia el sentido de numeración de los grados de libertad, en este caso el control indica de arriba hacia abajo. Es decir, que el grado de libertad 1 estará hasta arriba del oscilador. Esta opción fue creada con el fin de que el usuario pueda reproducir ejemplos mostrados en la bibliografía, ya que éstos a veces tienen diferente sentido de numeración.
	Tablas de propiedades	Abre la ventana "*Propiedades*" ver sección 4.2.4.13

4.2.4.2. Control de propiedades

Por medio de este grupo de controles (figura 4.20) el usuario define el número de grados de libertad del oscilador, proporcionando para cada uno sus propiedades de: masa, rigidez, amortiguamiento, vector de forma y condiciones iniciales (desplazamiento y velocidad). Es el primer grupo de controles en la franja del extremo derecho de la ventana de osciladores de VGL y su tamaño no es modificable

La tabla que contiene los valores de las propiedades tiene dimensiones que exceden al control, por ello se colocaron barras de desplazamiento vertical y horizontal para ver todos los valores contenidos en la tabla.

Figura 4.20 Control de propiedades

#	Control	Descripción
1	*GroupBox*	Conjunta a los controles dentro de un mismo espacio
2	*Numeric Up Down*	Define el número grados de libertad del oscilador, éste puede ser de 2 a 25 grados de libertad. Automáticamente al modificar el número de grados de libertad el programa redibuja al osciladores en la parte izquierda del control.
3	*DataGrid View*	Tabla que contiene las propiedades del oscilador de VGL. En el encabezado de cada columna se observa el nombre de la propiedad a la que se refiere. La primera columna identifica al grado de libertad. Las filas que muestre la tabla corresponden al número grados de libertad definidos.
4	*Numeric Up Down*	Establece el número de modos a superponer cuando se realice el cálculo de la respuesta. Este valor no puede ser mayor al número de grados de libertad definidos
5	*ScrollBars*	Permite ver las filas de la tabla que se encuentran ocultas debido a que las dimensiones de la tabla exceden a las del control de propiedades.

4.2.4.3. Control de gráficas

Este grupo de controles (figura 4.21) tiene la misma función que su similar de la ventana de osciladores de 1GL; sirven para que el usuario seleccione y muestre en pantalla la gráfica de la respuesta dinámica de un determinado grado de libertad.

En la parte central de la pantalla el programa dibuja las gráficas de respuesta; en total puede dibujar 5 gráficas: 1 de carga y 4 de respuesta, pero estas últimas pueden variar de 2 a 4 según defina el usuario.

Los tipos de respuesta que hay disponibles para seleccionar son: desplazamiento, velocidad, aceleración relativa y absoluta, así como también la gráfica de la fuerza cortante. Adicionalmente a estas opciones hay un control del tipo *Checkbox* que permite ver las respuestas de cada modo de vibrar.

Figura 4.21 Control de gráficas

#	Control	Descripción
1	*GroupBox*	Conjunta a los controles dentro de un mismo espacio
2	*CheckBox*	Muestra la respuesta dinámica por modos de vibrar. Al activar la casilla se dibujarán en las gráficas la respuesta por modo del tipo (desplazamiento, velocidad o aceleraciones) que esté definido en el primer *ComboBox*. Al mismo tiempo se bloquearán los *ComboBox* que pertenecen a las gráficas 2, 3 y 4, además sus respectivos controles *Numeric Up Down* corresponderán al modo de vibrar y ya no al grado de libertad. Cuando el usuario desactive la casilla, el estado de los controles volverá a su estado original.
3	*Numeric Up Down*	Define el número gráficas de respuesta que se ven en pantalla. Mientras la animación del oscilador esté en ejecución este control permanecerá bloqueado
4	*ComboBox*	Permiten seleccionar la respuesta dinámica que el usuario desee ver en pantalla
5	*Numeric Up Down*	Define a que oscilador corresponde la respuesta dinámica seleccionada en el *ComboBox*

4.2.4.4. Control Animación del oscilador

Con estos controles (figura 4.22) el usuario maneja la animación del oscilador; puede reproducirla, detenerla en cualquier instante o por completo con los botones: Play, Pause y Stop. También puede incrementar la velocidad de animación con la opción de diezmado.

La animación del oscilador consiste en ver en cada intervalo de tiempo (Δt) el desplazamiento del oscilador con respecto a su estado original. El tema de las animaciones es explicado con profundidad en el capítulo correspondiente a los algoritmos.

El programa también permite establecer el punto máximo de desplazamiento de los osciladores, a través del control de tipo *ScrollBar*. Existe también una opción para que el programa muestre el valor de los desplazamientos en cada grado de libertad.

Cuando la animación haya sido activada, los controles para los modos de vibrar se bloquean, hasta que la animación termine.

Figura 4.22 Controles para animación del oscilador

#	Control	Descripción
1	*GroupBox*	Conjunta a los controles dentro de un mismo espacio
2	*Numeric Up Down*	Diezma la animación. El programa toma un punto de cada N (valor del control) para hacer la animación de los osciladores; esto hace que la velocidad de reproducción sea mayor. El rango de valores para el diezmado es de 1 a 4
3	Botones	Botones para manipular la animación. Son similares a los que se usan en las aplicaciones de audio y video. De izquierda a derecha: Play, Pausa y Stop
4	*CheckBox*	Cuando se activa esta casilla el programa muestra el valor de los desplazamientos de cada grado de libertad durante la ejecución de la animación
5	*ScrollBar*	Establece hasta donde llegará el desplazamiento máximo del oscilador
6	Etiqueta	Cronómetro en tiempo real, para conocer la duración de la animación

4.2.4.5. Control Modos de vibrar

Con los controles (figura 4.23) el programa muestra las formas modales del oscilador, cada una con su valor correspondiente. El máximo número de modos será igual al número de grados de libertad que tenga el oscilador.

Cuando la casilla “Dibujar deformada” sea activada, el *Control de gráficas* y *Animación del oscilador* se bloquearán hasta que se desactive la casilla.

Para conocer más acerca de cómo el programa calcula y muestra los modos de vibrar vea el capítulo referente a los algoritmos.

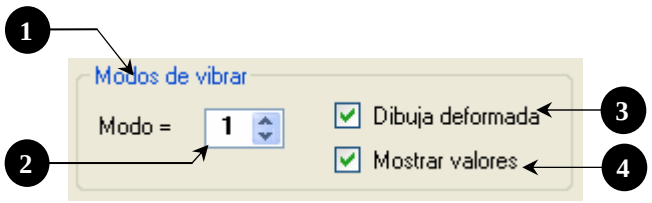


Figura 4.23 Controles para ver los modos de vibrar

#	Control	Descripción
1	*GroupBox*	Conjunta a los controles dentro de un mismo espacio
2	*Numeric Up Down*	Tamaño de las masas (50 a 60 píxeles) en los dibujos de los osciladores
3	*ScrollBar*	Establece hasta donde llegará el desplazamiento máximo de los osciladores
4	*Numeric Up Down*	Diezma la animación. El programa toma un punto de cada N (valor del control) para hacer la animación de los osciladores, esto hace que la velocidad de reproducción sea mayor. El rango de valores para el diezmado es de 1 a 4.

4.2.4.6. Gráfica de Carga

La gráfica de carga es exactamente la misma que la contenida en la ventana de osciladores de 1GL. Tiene las mismas propiedades con una pequeña excepción; las dimensiones mínimas de la gráfica para esta ventana son: 120 x 540 píxeles, un poco más corta que la gráfica contenida en la otra ventana principal.

Para ver las características de este control, vea la sección 4.2.3.5.

4.2.4.7. Gráficas de respuestas

Las gráficas de respuestas también tienen las mismas características que las ubicadas en la ventana de osciladores de 1GL. Muestran el mismo tipo de respuesta dinámica (desplazamiento, velocidad o aceleraciones) pero en este caso corresponden a los diferentes grados de libertad del oscilador

En esta ventana se pueden ver en pantalla de 2 a 4 gráficas de respuesta y se encuentran ubicadas debajo de la gráfica de carga, sus dimensiones mínimas son: 120 x 540 píxeles, y sus dimensiones máximas dependerán del tamaño de la pantalla pues el usuario puede ajustar el tamaño de las gráficas al de la pantalla.

Siempre que se abra la ventana principal, las gráficas de respuesta aparecerán vacías en el área de trabajo hasta que el usuario seleccione un tipo de carga e indique que se calcule la respuesta. Una vez hecho esto, el programa dibuja automáticamente la respuesta en los *PictureBox*, además coloca sus valores de amplitud correspondientes a cada respuesta.

La única diferencia con respecto a las gráficas de los osciladores de 1GL es el significado del nombre, por ejemplo "**D1**" en esta ventana significa *desplazamiento del grado de libertad 1*, mientras que en la otra ventana principal esto significa *desplazamiento del oscilador 1*.

Las gráficas de respuesta también tienen un menú secundario que aparece cada vez que se hace clic derecho sobre la gráfica, este menú se describe en la sección siguiente 4.2.4.8.

Para ver las características de estas gráficas, vea la sección 4.2.3.6.

4.2.4.8. Menú secundario de las gráficas de respuesta

El menú secundario (figura 4.24) aparece cuando el usuario hace clic derecho en alguna de las gráficas de respuesta. Contiene opciones que permiten modificar la apariencia de las gráficas, mostrar los valores de la respuesta dinámica, activar las líneas de referencia y mostrar la fuerza cortante.

El control se encuentra disponible en cualquier momento, excepto cuando la animación esté en ejecución, ya que el programa bloquea esta opción hasta que termine dicha animación.

Figura 4.24 Menú secundario de las gráficas de respuesta

Opciones de Menú	Descripción
Apariencia	Abre la ventana “Apariencia” (ver sección 4.2.5.6)
Misma Escala	Dibuja las gráficas de respuesta a la misma escala. Esta opción se habilita cuando los tipos de las respuestas de los osciladores son iguales. El programa toma como base la gráfica que tenga la mayor respuesta y a partir de ésta dibuja las demás.
Datos de respuesta	Abre la ventana “Datos de respuesta” (ver sección 4.2.4.9)
Espectros de piso	Abre la ventana “Espectros de respuesta” (ver sección 4.2.5.7)
Líneas de referencia	Contiene opciones para manipular las líneas de referencia móviles sobre las gráficas de respuesta y carga.
Mostrar	Aparece/quita las líneas de referencia
Líneas independientes	Habilita/deshabilita que las líneas de referencia, de todas las gráficas, se muevan como una sola.
Mostrar Resp máximas	Ubica a las líneas de referencia donde se encuentra la mayor amplitud en cada gráfica.
Tablas de propiedades	Abre la ventana “Propiedades” (ver sección 4.2.4.13)
Fuerza Cortante	Muestra los valores de la fuerza cortante en el oscilador para cada grado de libertad

4.2.4.9. Ventana Datos de respuesta

Contiene los datos que numéricos que conforman a la respuesta dinámica, la fuerza cortante y los valores de las ocho constantes para obtener la respuesta de cada grado de libertad.

La ventana está divida en cinco secciones por medio del control de tipo *TabPage*. Cada sección muestra la respuesta numérica en diferentes formatos.

Como sucede con todas las tablas que hay en el programa, el usuario puede copiar los datos (tabla de la ventana 4.25) al portapapeles de Windows con tan sólo seleccionarlos, ya sea por celda, columnas enteras o incluso la tabla completa y oprimiendo *CTRL+C*. Además, el programa proporciona en varias de las secciones de la ventana una opción para exportar todos los datos a un archivo de texto.

La primera sección (figura 4.25) muestra la respuesta dinámica por grado de libertad. Es decir, el usuario selecciona el grado de libertad y en la tabla se observará los valores numéricos del desplazamiento, velocidad, aceleración relativa y absoluta correspondiente al GL.

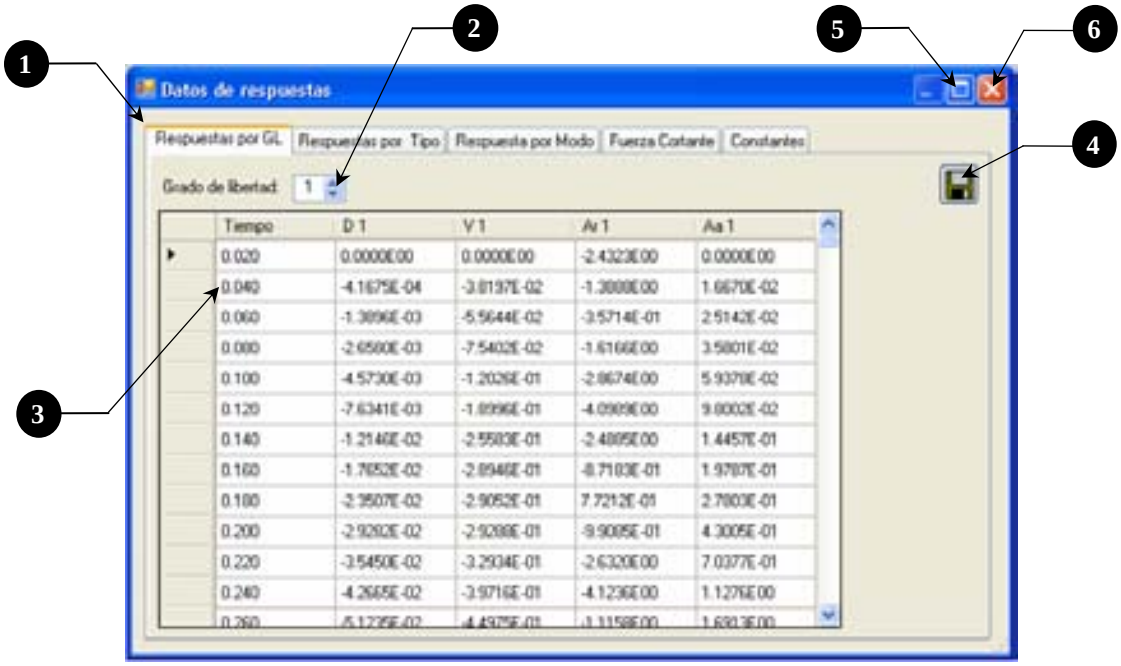


Figura 4.25 Ventana Respuesta numérica por GL

#	Control	Descripción
1	*Tab Page*	Primera sección de la ventana, muestra la respuesta dinámica por grado de libertad
2	*Numeric Up Down*	Establece el grado de libertad que se desea ver
3	*DataGrid View*	Tabla que contiene la respuesta dinámica dependiendo de la opción seleccionada. Tiene 5 columnas, cada una lleva su nombre en el encabezado, que corresponde a la primera letra del tipo de respuesta seguida por el número del grado de libertad al que pertenece la respuesta
4	*Button*	Exporta la respuesta de cada grado de libertad en archivos de texto. Se tendrán tantos archivos de texto como grados de libertad tenga el oscilador
5	*Button*	Maximiza la ventana
6	*Button*	Cierra la ventana

La segunda sección (figura 4.26) muestra la respuesta numérica de los grados de libertad por tipo, es decir, por desplazamiento, velocidad o aceleraciones. En esta sección la ventana tiene dos tablas, una es la que contiene los datos de la respuesta y la otra es para que el usuario especifique los grados de libertad que desea ver.

La tabla para los datos de la respuesta muestra sólo 5 columnas para los grados de libertad del oscilador, pues por cuestiones de capacidad de memoria no es factible mostrar un mayor número de columnas.

Suponga que el usuario define el máximo número de GL para el oscilador (25) y que selecciona un tipo de carga con el máximo número de puntos permitido por el programa (12,000) si la ventana mostrara todos los datos a la vez, mostraría 312,000 datos, lo que llevaría a una gran demanda de memoria por parte del programa a la computadora, por esta razón se muestran sólo 5 columnas de datos de las 25 posibles, reduciendo el número de datos a una quinta parte.

Sin embargo, si el usuario necesitara ver todos los datos a la vez puede usar la opción de exportar los datos a un archivo de texto; éste a diferencia de la sección anterior guarda un sólo archivo por tipo de respuesta que dependerá de la opción seleccionada en la sección.

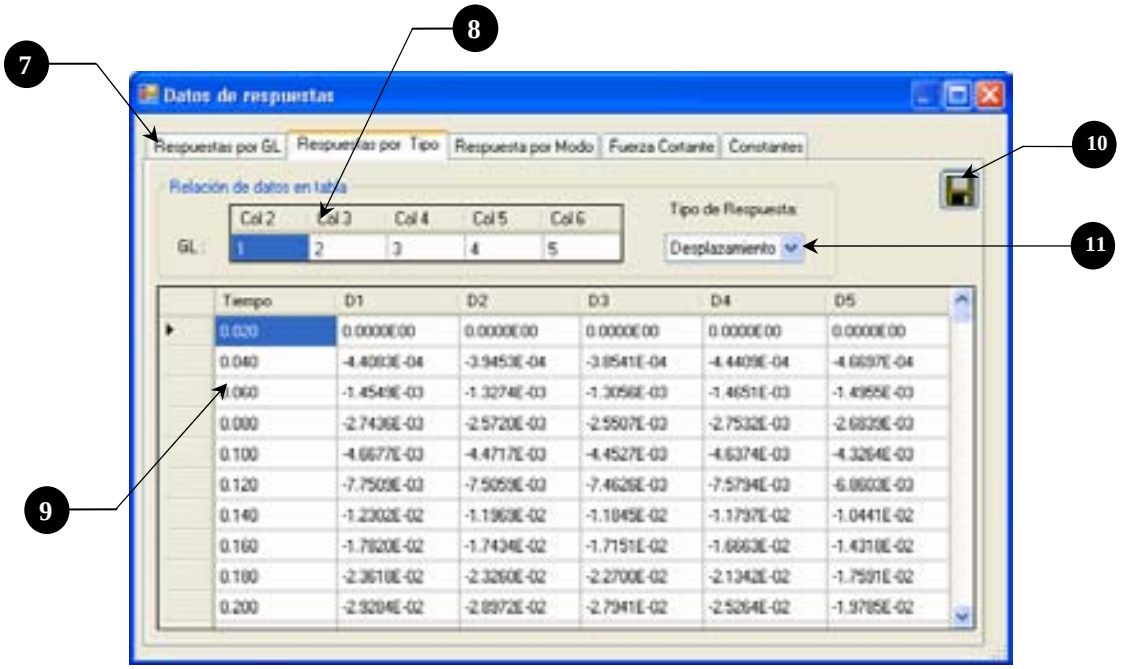


Figura 4.26 Ventana Respuesta numérica por Tipo

#	Control	Descripción
7	*Tab Page*	Segunda sección de la ventana, muestra la respuesta dinámica de los grado de libertad por tipo de respuesta
8	*Data Grid View*	Tabla secundaria con una sola fila y 5 columnas, los valores que contiene corresponden a la respuesta del grado de libertad que se desee ver en la tabla de abajo. El encabezado de cada columna tiene el nombre de la columna (ejemplo: “Col 2”, Columna 2) de la tabla principal.

		Por ejemplo, en la columna "Col2" se encuentra el valor "1", esto significa que en la tabla principal en la columna 2 se mostrará la respuesta (establecida en el *Combo Box*) del grado de libertad 1, en este caso el desplazamiento del grado de libertad 1: **D1**
9	*Data Grid View*	Muestra la respuesta dinámica dependiendo de la opción seleccionada en el *Combo Box* y los valores especificados en la primer tabla. Tiene 6 columnas, cada una lleva su nombre en el encabezado, que corresponde a la primera letra del tipo de respuesta seguida por el número del grado de libertad al que pertenece la respuesta
10	*Combo Box*	Contiene las opciones del tipo de respuesta: Desplazamientos, Velocidad, Aceleración relativa y absoluta
11	*Button*	Exporta la respuesta de un mismo tipo de todos los grados de libertad en un archivo de texto. El tipo de respuesta que se exporte será el seleccionada en el *ComboBox*

La tercera sección (figura 4.27) tiene como objetivo mostrar la respuesta dinámica por modos de vibrar, su diseño es similar a la mostrada en la figura 4.26; tiene también dos tablas con las mismas funciones que la anterior sección, además de un control *ComboBox* para seleccionar el tipo de respuesta y un control *Numeric Up Down* para establecer el grado de libertad para el cual se desea ver la respuesta.

Por cada grado de libertad hay la misma cantidad de modos de vibrar. La tabla secundaria contiene los datos para mostrar la respuesta por modos de vibrar.

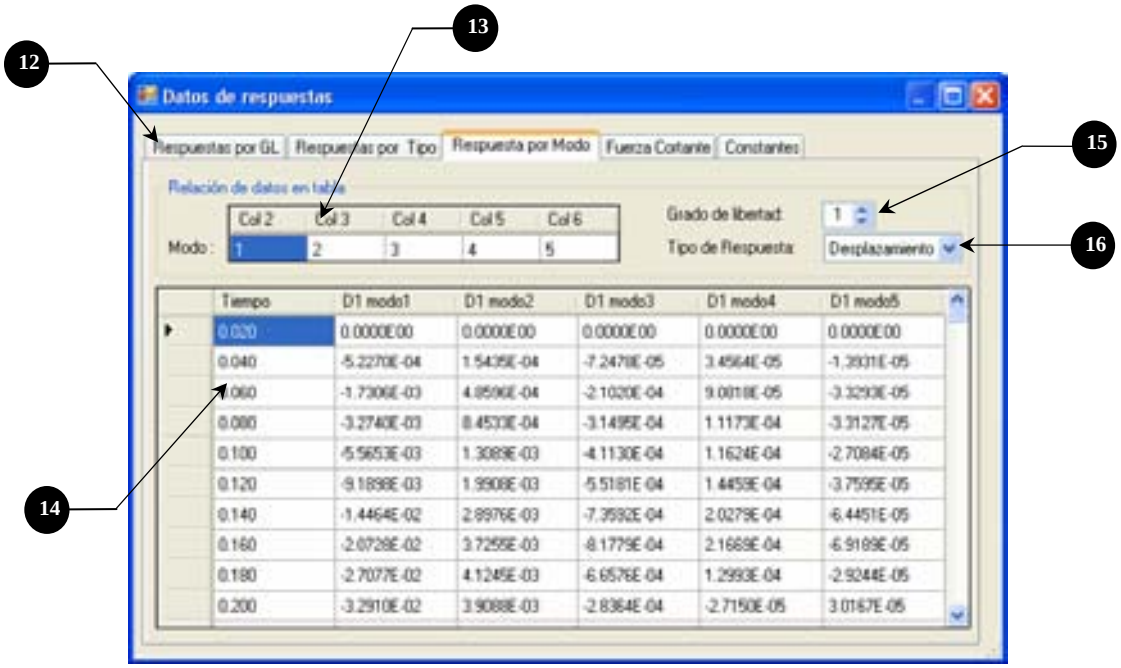


Figura 4.27 Ventana Respuesta numérica por Modo de Vibrar

#	Control	Descripción
12	*Tab Page*	Tercera sección de la ventana, muestra la respuesta dinámica de un mismo tipo por modos de vibración para cada grado de libertad

13	*Data Grid View*	Tabla secundaria con una sola fila y 5 columnas, los valores que contiene corresponden a la respuesta por modo de vibrar del grado de libertad que se desee ver. El encabezado de cada columna es el mismo que el de la sección anterior
14	*Data Grid View*	Muestra la respuesta dinámica dependiendo de la opción seleccionada en el *ComboBox*, el grado de libertad definido en el control *Numeric Up Down* y los valores especificados en la primer tabla. Tiene 6 columnas, cada una lleva su nombre en el encabezado, que corresponde a la primera letra del tipo de respuesta seguida por número del grado de libertad al que pertenece la respuesta y por el modo de vibrar que corresponde a la tabla secundaria
15	*Numeric Up Down*	Establece el grado de libertad al que pertenece la respuesta
16	*Combo Box*	Contiene las opciones del tipo de respuesta: Desplazamientos, Velocidad, Aceleración relativa y absoluta

La sección número 4 (figura 4.28) muestra la fuerza cortante de cada grado de libertad del oscilador, tiene las mismas tablas que las secciones anteriores y un control de tipo *Button* que permite exportar los datos a un archivo de texto.

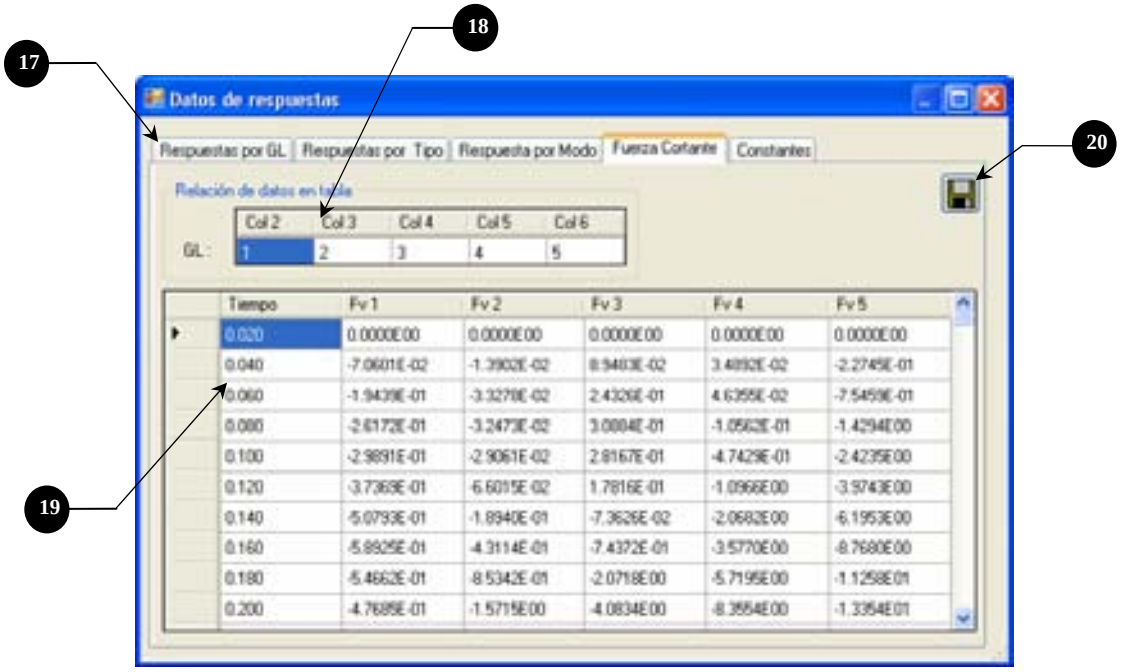


Figura 4.28 Ventana Respuesta numérica de la Fuerza Cortante

#	Control	Descripción
17	*Tab Page*	Cuarta sección de la ventana, muestra la fuerza cortante para cada grado de libertad
18	*Data Grid View*	Tabla secundaria con una sola fila y 5 columnas, los valores que contiene corresponden a los grado de libertad que se desee ver. El encabezado de cada columna es el mismo que el de la segunda sección

19	*Data Grid View*	Muestra la fuerza cortante de los grados de libertad especificados en la primer tabla. Tiene 6 columnas, cada una lleva su nombre en el encabezado, que corresponde a la Fv (fuerza cortante) seguido por número del grado de libertad al que pertenece la respuesta
20	*Button*	Exporta los datos de la fuerza cortante de todos los grados de libertad en un archivo de texto

La última sección (figura 4.29) muestra los valores de las ocho constantes (4 de desplazamiento y 4 de velocidad) que se obtuvieron para obtener la respuesta de cada grado de libertad del oscilador con sus diferentes periodos de vibrar.

A diferencia de las secciones anteriores, esta tabla puede mostrar los 25 grados de libertad al mismo tiempo.

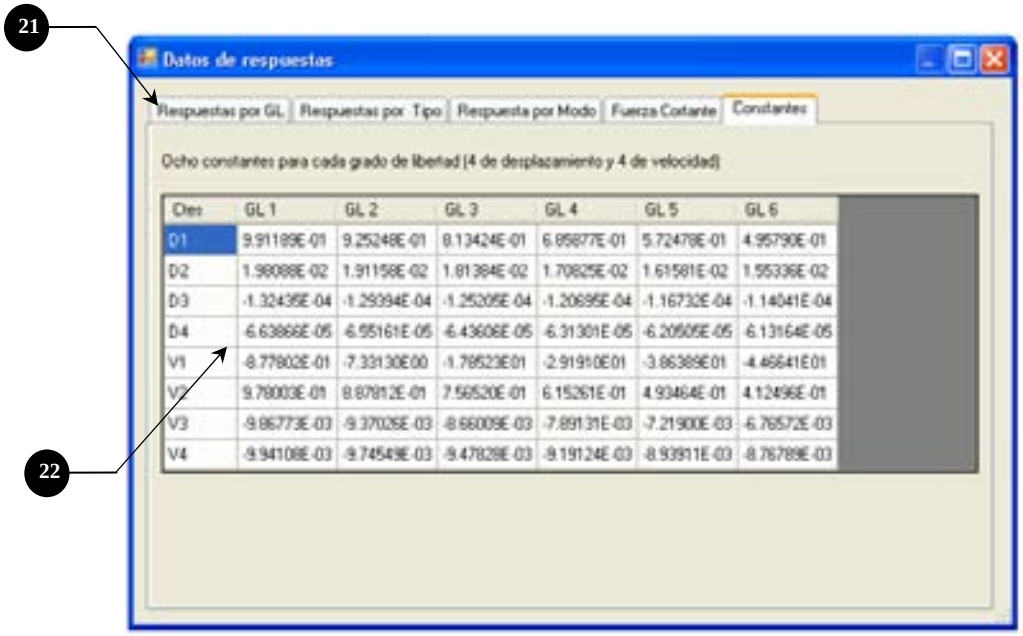


Figura 4.29 Ventana con los valores de las ocho constantes

#	Control	Descripción
21	*Tab Page*	Quinta sección de la ventana, muestra las ocho constantes para cada grado de libertad del oscilador.
22	*Data Grid View*	Muestra los valores de las ocho constantes correspondientes a todos los grados de libertad del oscilador. Tiene N columnas, cada una lleva su nombre del grado de libertad en el encabezado.

4.2.4.10. Oscilador

El oscilador de VGL (figura 4.30) queda representado por el sistema de masas y resortes. El control es del tipo *PictureBox* y su objetivo es mostrar los desplazamientos que tiene el oscilador en cada instante de tiempo, debido a la fuerza lateral que se ejerce sobre él.

En esta ventana sólo hay un oscilador (color verde). El fondo del oscilador es similar al de las gráficas de respuesta (cuadriculado con un fondo gris).

En su parte superior aparece una pestaña que incluye el número de grados de libertad del oscilador (ejemplo: GL= 4) y además cuando la animación está en ejecución se muestra el instante de tiempo en el extremo derecho del control.

Mientras la animación no esté en ejecución el oscilador permanecerá estático en su estado inicial, dibujado al centro del *PictureBox* (desplazamiento = 0).

El ancho del *PictureBox* es de 180 píxeles; su altura dependerá de los grados de libertad del oscilador y se ubica entre las gráficas y la franja de controles del extremo derecho de la ventana.

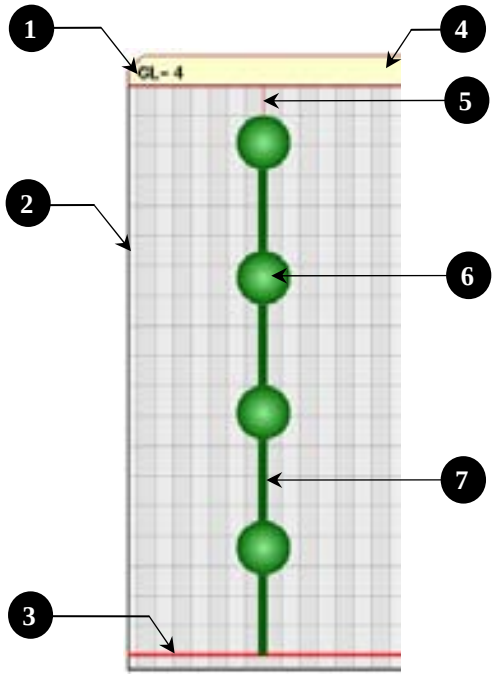


Figura 4.30 Oscilador de VGL

#	Descripción
1	Número de grados de libertad del oscilador
2	Cuadrícula y fondo de color gris, en este caso las líneas verticales y horizontales no tienen valor de escala, como en las gráficas de respuesta.
3	Línea base en la que se desplanta el oscilador
4	Ubicación del texto que corresponde al instante de tiempo t, cuando se ejecuta la animación
5	Eje vertical (línea color rojo) que indica el centro del oscilador y desplazamiento lateral = 0
6	Masas del oscilador
7	Columnas que representan la rigidez del oscilador

4.2.4.11. Espectros de piso

La gráfica (figura 4.31) muestra el espectro de piso (definido en la sección 5.2.3.2) que se genera para un oscilador de VGL. De la misma manera que en la ventana de osciladores de 1GL, el programa no calcula automáticamente estos espectros debido a que necesita algunos parámetros para generarlos. Para indicar al programa que calcule los espectros de piso vea la sección 4.2.5.7.

Una vez que ya han sido calculados los espectros de piso el usuario puede verlos haciendo clic en cualquiera de las masas del oscilador; el espectro aparecerá al lado de la masa en donde haya hecho clic. La gráfica estará visible mientras el usuario no haga clic en cualquier otro lado.

Las características de estas gráficas son similares a la de los espectros de respuesta de la ventana de los osciladores de 1GL. Tienen una pestaña en lo alto de la gráfica y muestran una cuadrícula con fondo de color rojizo. Su tamaño es de 180 x 180 píxeles. El espectro de piso es de color vino para que haya un contraste con el color de fondo.

El espectro de piso que se genera será siempre el de la aceleración absoluta y no hay controles en el programa que modifiquen este parámetro.

La gráfica cuenta con una línea de referencia (vertical) que se mueven cuando el cursor del Mouse pasa sobre la gráfica. Este control no presenta ningún menú secundario.

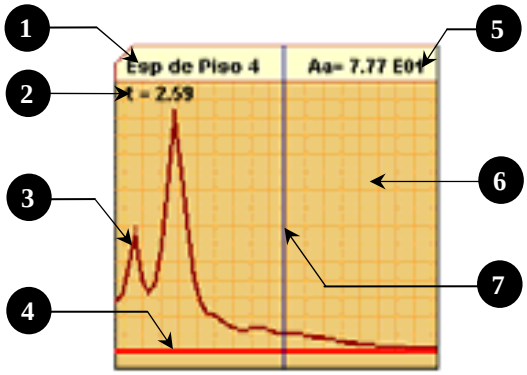


Figura 4.31 Gráfica de espectros de piso

#	Descripción
1	Nombre del espectro seguido del número de grado de libertad al que corresponde
2	Valor del periodo que corresponde al línea de referencia vertical
3	Espectro de piso
4	Línea base que corresponde a la amplitud cero
5	Valor de la amplitud (Aceleración absoluta) que corresponde a la intersección de la línea de referencia con el espectro de piso
6	Fondo del espectro, cuadriculado sin ningún valor de escala
7	Línea de referencia (vertical) que se mueven con el puntero del Mouse cuando pasa sobre la gráfica.

4.2.4.12. Ventana Tipo de amortiguamiento

El objetivo de la ventana es proporcionar al usuario dos formas de establecer los valores de amortiguamiento del oscilador: Amortiguamiento modal y de Rayleigh.

Es una ventana pequeña de 100 x 100 píxeles, de tamaño no modificable. Por medio de los controles permite que el usuario seleccione que tipo de amortiguamiento desea introducir.

Figura 4.32 Ventana para generar amortiguamientos de Raylegh

#	Control	Descripción
1	*Button*	Cierra la ventana sin hacer ningún cambio en el programa
2	*Option Button*	Opción que permite al usuario introducir directamente los valores del amortiguamiento modal en el *control de propiedades*. Cuando se selecciona esta opción, los *TextBox* de la ventana se bloquean.
3	*Option Button*	Opción que permite introducir los coeficientes Alfa y Beta para calcular los amortiguamientos de Rayleigh. Cuando esta opción es seleccionada, la columna de amortiguamientos en el control de propiedades es bloqueada, debido a que los valores son calculados y no pueden ser modificados por el usuario a menos que seleccione la primera opción.
4	*TextBox*	Casillas de texto que reciben los coeficientes para calcular los amortiguamientos
5	*Button*	Calcula los amortiguamientos a partir de los valores introducidos o establece que se pueden ingresar los valores directamente en el *control de propiedades*, dependiendo de la opción seleccionada. Inmediatamente después se cierra la ventana
6	*Button*	Cancela la opción seleccionada y cierra la ventana

4.2.4.13. Ventana Propiedades

Su objetivo es mostrar las siguientes propiedades del oscilador:

- ✓ Matriz de masas
- ✓ Matriz de rigidez

- ✓ Periodos y frecuencias de vibrar
- ✓ Amortiguamientos
- ✓ Modos de vibrar
- ✓ Factores de participación
- ✓ Vector de forma

Además, el programa proporciona una opción mediante la cual se pueden exportar todas las tablas de propiedades a un archivo de texto.

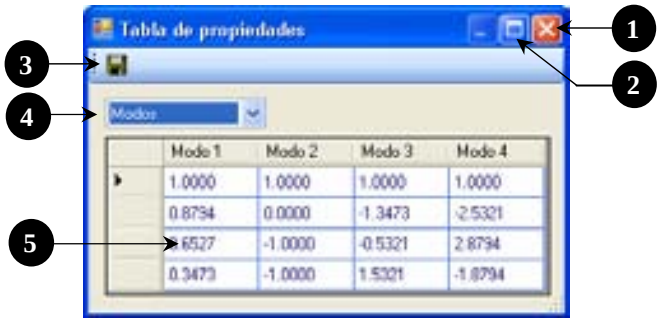


Figura 4.33 Ventana para mostrar las propiedades del oscilador de VGL

#	Control	Descripción
1	*Button*	Cierra la ventana
2	*Button*	Maximiza la ventana
3	*Button*	Exporta todos los datos a un archivo de texto
4	*ComboBox*	Contiene las opciones antes mencionadas
5	*DataGrid View*	Muestra los valores de las propiedades especificadas en el *ComboBox*

4.2.5. Ventanas comunes

Son ventanas secundarias que se usan por igual en ambas ventanas principales (Osciladores 1GL y Osciladores de VGL). En total son 7 ventanas y se dividen de la siguiente manera:

- 3 para generar un tipo de carga
- 2 para mostrar información (ayuda al usuario e proporciona información sobre la aplicación)
- 1 para modificar la apariencia de las gráficas de respuesta
- 1 para introducir los parámetros que generar espectros de respuesta.

4.2.5.1. Ventana de carga senoidal

El objetivo de la ventana es capturar los datos necesarios para crear una señal del tipo senoide. Los datos que se piden para generar la señal son: Amplitud, duración, periodo e intervalo de tiempo entre los puntos que conformarán a la señal.

La manera de generar la señal es con la ecuación 4.6

$$\Omega = \frac{2\pi}{T} \quad \text{.....(4.5)}$$

$$S(t) = A \cdot seno(\Omega \cdot t) \quad \text{.....(4.6)}$$

Donde:

Ω = *frecuencia de la señal en rad*
T = *periodo en s*
t = *tiempo en s, se incrementa desde cero hasta la duración tota, en intervalos de tiempo* Δt
A = *amplitud*
S (t) = *vector de datos de la señal*

El usuario ingresa los valores a través de los controles de tipo *Textbox* que se encuentran en la parte izquierda de la ventana (figura 4.34), además la ventana cuenta con una imagen en la que se indican cada parámetro de la señal.

La ventana tiene dos botones (Aceptar y Cancelar) para que el usuario indique al programa que genere o no la señal en base a los valores ingresados. Si el usuario hace clic en el botón aceptar, el programa carga automáticamente la señal, al mismo tiempo que cierra la ventana. En caso contrario, si el usuario hace clic en el botón cancelar o cerrar, el programa no carga la señal y además cierra la ventana.

La ventana no se puede maximizar ni minimizar, es de un sólo tamaño y tiene sólo un botón de control (cerrar) en la parte superior. Cada vez que se abra esta ventana el usuario no podrá acceder a la aplicación principal hasta que cierra dicha ventana.

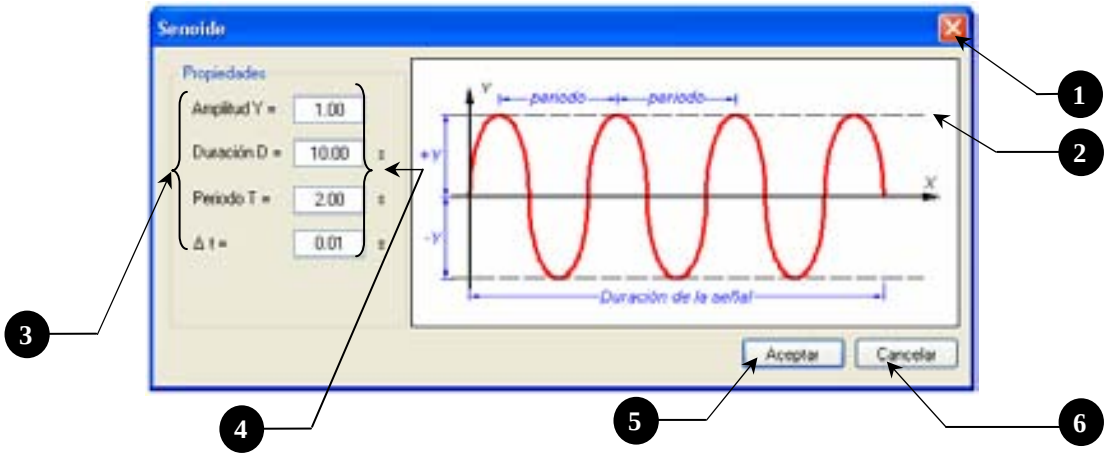


Figura 4.34 Ventana del tipo de carga Senoide

#	Control	Descripción
1	Botón Cerrar	Cierra la ventana sin generar la señal
2	*PictureBox*	Muestra una imagen que indica las características de una señal senoidal

3	Etiquetas	Indican el nombre de los parámetros de la señal
4	*TextBox*	Reciben los datos numéricos para crear la señal
5	Botón Aceptar	Cierra la ventana y crea la señal senoidal a partir de los datos introducidos
6	Botón Cancelar	Cierra la ventana y cancela los datos capturados

4.2.5.2. Ventana lectura de archivo de datos

El objetivo de la ventana es leer un archivo de texto que contenga información numérica para crear una señal de tipo arbitraria.

Cuando el usuario seleccione esta opción del menú de tipo de carga, lo primero que verá será un cuadro de dialogo (predeterminado por Windows) para que indique la ubicación del archivo, una vez hecho esto, el programa abre la ventana (figura 4.35) y muestra el contenido de dicho archivo.

El programa puede abrir cualquier archivo de texto pero no significa que siempre se pueda procesar la información contendida. Para ello es necesario que el archivo tenga el siguiente formato:

- ✓ Los datos que se procesen deben ser sólo datos numéricos
- ✓ Los datos pueden estar acomodados en columnas
- ✓ El archivo debe tener un mínimo de 50 datos y un máximo de 12000 por columna

Los datos numéricos del archivo deben corresponder a la amplitud de la señal. La separación de tiempo entre los datos se toma como un valor constante (Δt) y se especifica en un control *TextBox* dentro de la ventana.

Cuando los archivos tengan, además de los datos numéricos, otro tipo de texto como información acerca del registro, fecha, hora, lugar, etc. el usuario puede quitar dicha información directamente del control con tan sólo seleccionar y borrar los datos que no sean numéricos. Es importante aclarar que el programa al abrir el archivo de texto crea una copia del mismo, así cuando el usuario modifica el archivo en realidad el original no sufre cambio alguno.

La ventana está programada para leer los datos del archivo en un formato de varias columnas, como se muestra en la ventana de la figura 4.35; esto lo hace bastante práctico, pues es común que los archivos de registros sísmicos contengan más de una columna de datos.

De los tipos de carga que hay disponibles en el programa, éste se considera como el más general de todos, pues por medio de un archivo de texto el usuario puede ingresar cualquier tipo de señal que desee, basta con crear su señal y guardarla en un archivo. Además de esto, el usuario puede introducir registros de instrumentación sísmica ya que también son creados como archivos de texto.

Una de las opciones del programa es que puede guardar, si así lo especifica el usuario, los registros que se lean del archivo en la base de datos del programa.

La ventana tiene controles que le permiten ajustarse al tamaño de la pantalla, esto con el fin de que se vea en un mayor espacio la información del archivo. Esta última se muestra a través de un control de tipo *RichTextBox* que se encuentra ubicado en la parte izquierda de la ventana, los demás controles

sirven para especificar el número de columnas, el intervalo de tiempo (Δt) y determinar si se desea guardar en la base de datos.

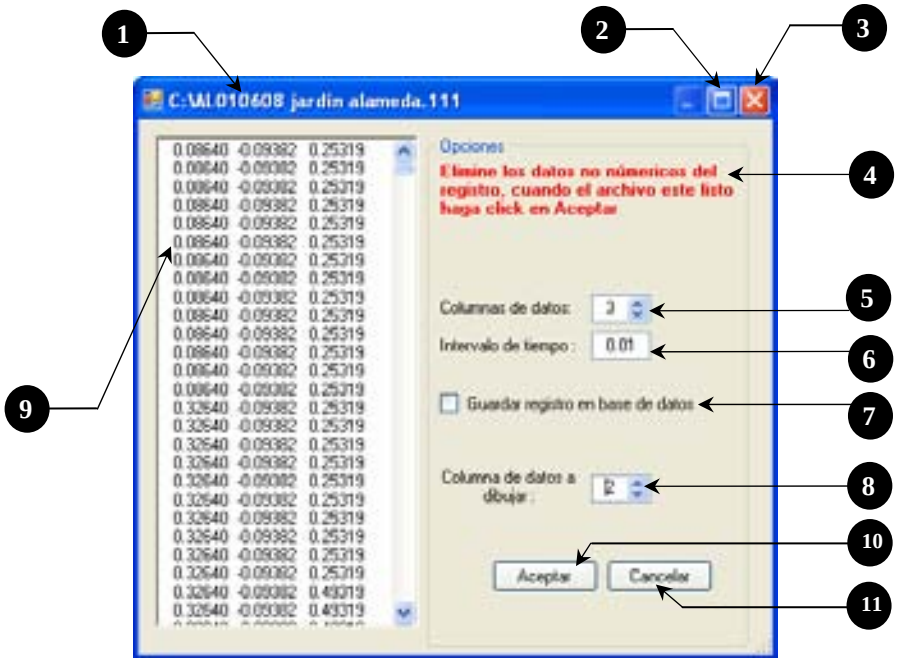


Figura 4.35 Ventana para abrir un archivo de texto

#	Control	Descripción
1	Título	Ubicación del archivo de texto
2	Botón Maximizar	Se maximiza la ventana
3	Botón Cerrar	Cierra la ventana sin cargar la señal
4	Etiqueta	Mensaje para el usuario elimine los datos innecesarios del archivo
5	Control numérico	Permite especificar el número de columnas de datos que tiene el archivo
6	*TextBox*	Recibe el valor de las muestras por segundo de la señal en el archivo
7	*CheckBox*	Al activar esta opción y oprimir el botón aceptar se abre una ventana donde permite guardar las señales del archivo como registros en la base de datos del programa.
8	Control numérico	A través de éste se especifica cuál columna de datos se cargará al programa.
9	*Rich Text Box*	Muestra el contenido del archivo de texto, se puede ingresar texto o eliminar directamente desde él. (El archivo original no sufre ningún cambio)
10	Botón Aceptar	Cierra la ventana y carga la señal a partir de los datos introducidos
11	Botón Cancelar	Cierra la ventana y cancela los datos capturados

4.2.5.3. Ventana Base de Datos

El objetivo de la ventana es guardar y mostrar los registros de la base de datos; esto con el fin de proporcionarle al usuario una manera rápida de aplicar carga de tipo arbitraria a los osciladores.

Los registros de la base de datos son archivos de texto guardados en una carpeta llamada *Registros* y se ubica en la carpeta donde se instaló el programa. Cuando se abre la ventana el programa hace un enlace a la carpeta y muestra los nombres de los archivos que encuentra.

No es necesario que el usuario tenga conocimiento de esta carpeta para usar la base de datos, ya que el programa vincula inmediatamente la carpeta con él mismo para que el usuario a través de la aplicación manipule la base de datos.

Cada vez que el usuario hace cambios (agregar, cambiar nombre, borrar, etc.) en la base de datos, el programa efectúa dichos cambios en los archivos de texto.

El tamaño de la ventana no es modificable y tiene un sólo botón de control (cerrar) en la parte superior de la misma. A través de un control de tipo TabPage (pestañas) la ventana muestra dos grupos de opciones diferentes: “Registros existentes” y “Guardar”.

La pestaña “Registros existentes” (figura 4.36) cuenta con un control de tipo *ListBox* ubicado en la parte izquierda de la ventana, que enlista los registros existentes en la base de datos. Al hacer clic sobre alguno de estos registros el usuario puede ver las propiedades de ellos (número de puntos, intervalo de tiempo, fecha de creación del registro y unidades de la amplitud) en la parte derecha de la ventana.

El usuario puede eliminar registros de la base de datos o cambiarles el nombre, esto con la ayuda del menú emergente en el control *ListBox*. Complementan al grupo de controles los botones Actualizar y Cargar señal ubicados en la parte inferior derecha de la ventana.

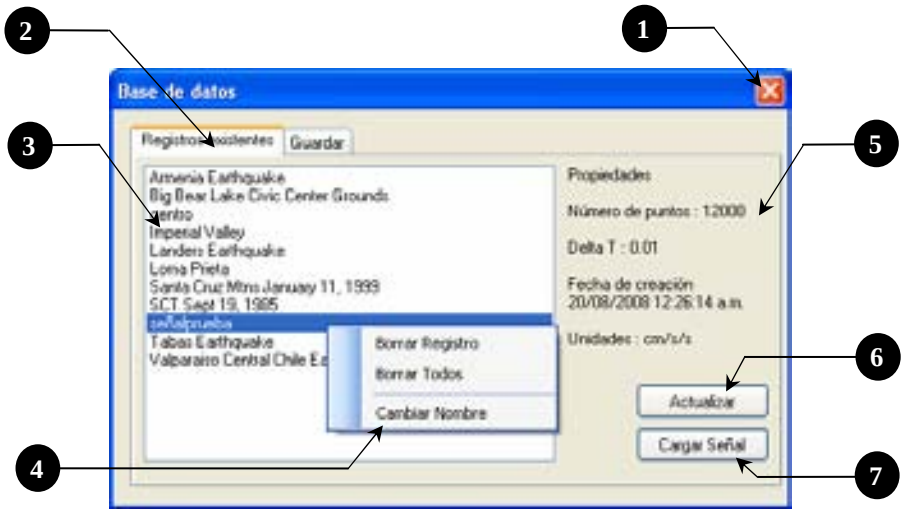


Figura 4.36 Ventana base de datos (Ver registros)

#	Control	Descripción
1	Botón cerrar	Cierra la ventana sin agregar ningún registro a la base de datos ni cargar señal alguna al programa.

2	*Tab page*	Divide a la ventana en dos grupos: *Registros existentes y Guardar*
3	*ListBox*	Muestra la lista de los registros que se encuentran en la base de datos
4	Menú emergente	Menú secundario que aparece al hacer clic derecho sobre el control *ListBox*
	Borra registro	Borra el registro (seleccionado en el *ListBox*) permanentemente de la base de datos
	Borrar todos	Borra todos los registros permanentemente de la base de datos
	Cambiar nombre	Permite que el usuario cambie el nombre del registro
5	*Label*	Muestra las propiedades (Número de puntos, intervalo de tiempo, fecha de creación y unidades) del registro que esté seleccionado
6	Botón	Actualiza la base de datos en caso de que el usuario haya modificado la carpeta contenedora de los registros
7	Botón	Carga el registro seleccionado al programa

La pestaña "Guardar" (figura 4.37) permite agregar nuevos registros a la base de datos. La ventana cuenta con dos controles de tipo *ListBox*, el de la parte izquierda enlista los registros que están disponibles para guardar, y el de la parte derecha muestra las registros que se seleccionaron para agregar a la base de datos.

Para que el control *ListBox* izquierdo muestre algún registro disponible, es forzoso que el usuario seleccione la opción "Guardar en base de datos" en la ventana "*abrir archivo de texto"*.

Los registros mostrados en el *ListBox* izquierdo tendrán el mismo nombre que el archivo de datos abierto en la ventana "*abrir archivo de texto"* seguido de la identificación de la columna (ejemplo: C-1), la letra C es por columna y el número corresponde al de la columna, numerada de izquierda a derecha en el archivo de texto. En caso de que se especifique que el archivo de datos contenga una sola columna de datos, el nombre de registro no llevará dicha identificación.

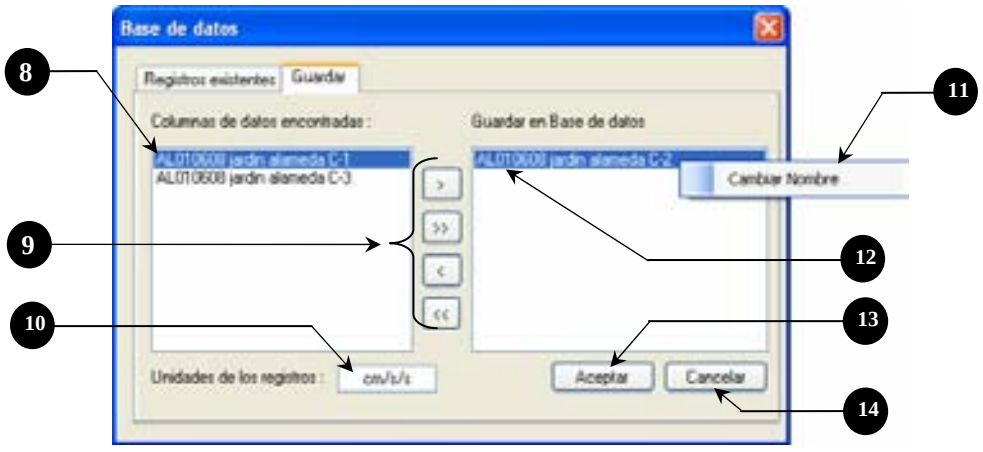


Figura 4.37 Ventana base de datos (Guardar registros)

Si el usuario desea que su registro se guarde con un nombre diferente del que el programa le asigno, basta con hacer clic derecho sobre él en el control *ListBox* derecho y seleccionar en el menú emergente *"Cambiar nombre"*.

#	Control	Descripción
8	*ListBox*	Muestra la lista de los registros disponibles (columnas de datos) en el archivo de texto abierto en la ventana "*abrir archivo de texto*"
9	Botones	Sirven para agregar o quitar los registros del *ListBox* derecho
10	*TextBox*	Recibe un texto que indica las unidades de los valores de los registros. No tiene ningún formato establecido y el usuario puede ingresar o no un texto que le ayude a recordar las unidades de su registro
11	Menú emergente	Menú secundario que aparece al hacer clic derecho sobre el control *ListBox* derecho, su única opción permite al usuario cambiar de nombre a los registros
12	*ListBox*	Muestra la lista de los registros que el usuario selecciono para agregar a la base de datos
13	Botón	El programa agrega los registros en el *ListBox* derecho a la base de datos acción seguida de cerrar la ventana
14	Botón	El programa cierra la ventana sin agregar ningún registro a la base de datos

4.2.5.4. Ventana Ayuda

Para proporcionar una guía con temas de ayuda (función cumple cada control dentro del programa) al usuario, se creó un archivo externo de tipo *chm*. Éste no fue desarrollado directamente en Visual Studio, como el resto del programa, sino que para su creación se utilizó la aplicación *Microsoft HTML WorkShop*[1]. Este programa recibe archivos de tipo HTML (paginas) con la información que se desea mostrar en la ayuda y se especifican algunas características para el archivo, tales como: botones a usar y nombre del mismo.

Cuando toda la información está contenida en el HTML WorkShop, ésta se compila para crear un archivo de extensión *chm*. El mismo HTML WorkShop se encarga de genera los controles y ventanas necesarias para que el archivo tenga una apariencia como la que se muestra en la figura 4.38.

Para enlazar el archivo *chm* con la aplicación hecha en Visual Studio, se hace a través de la siguiente instrucción de código.

```
Me.HelpProvider1.HelpNamespace = "Ayuda.chm"
System.Windows.Forms.Help.ShowHelp(Me, Me.HelpProvider1.HelpNamespace)
```

Código fuente 4.1 Enlace al archivo de ayuda.chm

En la primera línea del código fuente 4.1 se especifica al control *Helpprovider1* el nombre del archivo de ayuda requerido: *Ayuda.chm*. En la segunda línea, se indica al programa que muestre el archivo de ayuda contenido en el control *HelpProvider1* cuando sea llamado por el usuario.

El archivo de ayuda muestra un sólo formulario que está divido en dos partes principales: la hoja de menús y la hoja para mostrar la información.

[1] Programa de Microsoft, Copyright 1996-199 para crear archivos de ayuda. Descargado de la pagina oficial de Microsoft

Varias páginas de información tienen vínculos (links hacia otras hojas dentro del archivo) que le dan mayor versatilidad al archivo de ayuda, pues el usuario puede saltar desde una página a temas relacionados con lo que esté leyendo. Para navegar entre las hojas de información, el programa coloca una barra de herramientas en la parte superior de la ventana.

El archivo de ayuda sólo tiene información acerca del uso de los controles del programa y no proporciona ningún algoritmo de cálculo, si el usuario necesita conocer más a fondo el funcionamiento del programa, tendrá que consultar la información contenida en el capítulo siguiente.

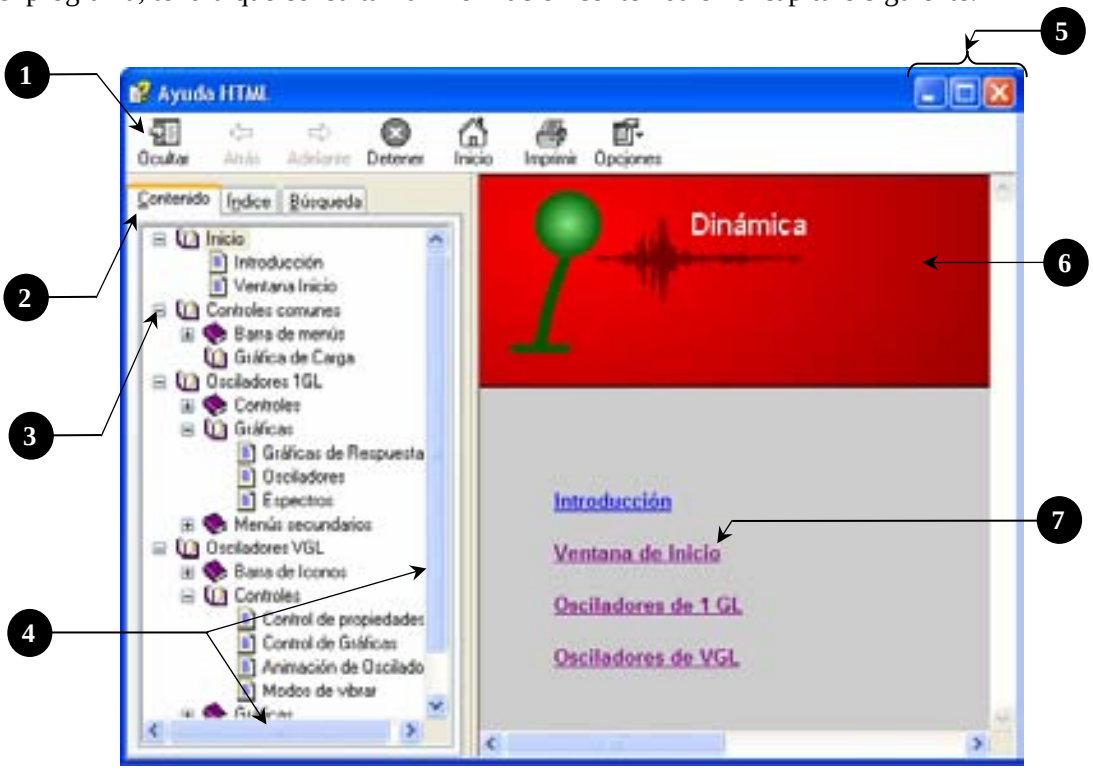


Figura 4.38 Ventana de ayuda

#	Control	Descripción
1	Barra de herramientas	Botones para navegar dentro del archivo de ayuda
	Ocultar	Oculta la hoja de menús (parte izquierda de la ventana)
	Atrás	Coloca la hoja de información anterior
	Siguiente	Muestra la siguiente hoja de información
	Detener	Suspende el proceso de cambiar de hoja
	Inicio	Regresa a la hoja de inicio del archivo de ayuda
	Imprimir	Abre una ventana con opciones para imprimir la información
	Opciones	Reúne las opciones anteriores en un menú desplegable
2	Tab Page	Divide a la hoja de menús en tres partes
	Contenido	Muestra los temas de ayuda de forma jerárquica, de arriba hacia abajo
	Índice	Coloca todos los temas ordenados alfabéticamente
	Búsqueda	Enlista los temas relacionados con una brusquedad que el usuario haya definido.

3	Hoja de menús	Contiene los temas que hay en el archivo de ayuda
4	Barras de desplazamiento	Permiten moverse por la ventana, se habilitan cuando la información contendida rebasa el tamaño de la hoja
5	Botones	Botones para controlar el estado de la ventana, (Minimizar, maximizar y cerrar)
6	Hojas de información	Es la parte más importante, aquí se muestra la información de ayuda contenida en las paginas HTML que se compilaron anteriormente
7	Links	vínculos en la paginas de ayuda, que permiten saltar a otras hojas, dentro del archivo de ayuda, relacionadas con el tema

4.2.5.5. Ventana Acerca de…

Su función es mostrar al usuario información acerca de la aplicación: Nombre, versión, derechos de autor y nombre del desarrollador, así como una breve descripción del objetivo del programa.

Es una ventana parecida a la ventana de presentación, pero ésta se abre en cualquier momento que el usuario lo requiera. Su tamaño es pequeño de 373 x 244 píxeles, presenta un sólo botón en la parte superior de la ventana. No tiene controles para ingresar o modificar valores, pues es una ventana para dar un mensaje al usuario.

Figura 4.39 Ventana acerca de…

#	Control	Descripción
1	Botón Cerrar	Cierra la ventana
2	*PictureBox*	Muestra una imagen de fondo, la cual está compuesta por un fondo de color rojo, un oscilador de 1GL y una señal de respuesta. Adicionales a éste, se colocaron textos para la información del programa.
3	*Rich Text Box*	Contiene una breve descripción del objetivo del programa

4.2.5.6. Ventana Apariencia

Su función es modificar las características (color de línea, grosor y escala de tiempo) de la gráfica de respuesta sobre la que se hizo clic derecho. La ventana es valida para cada gráfica de respuesta en ambas ventanas principales. Cada vez que se abra la ventana tomará los valores de las características que tenga la gráfica en ese momento, color, grosor y equidistancia en la escala de tiempo.

Por default las características iniciales son: color de línea verde, grosor igual a cero y separación de la cuadrícula de 1 segundo.

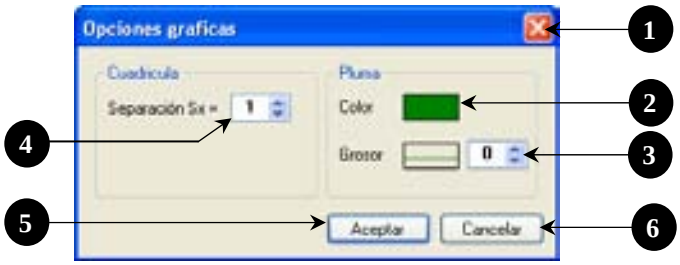


Figura 4.40 Ventana de Opciones Gráficas

#	Control	Descripción
1	Botón Cerrar	Cierra la ventana sin efectuar cambio alguno sobre la señal
2	*PictureBox*	Muestra el color de la línea de la señal, para cambiarlo haga clic sobre él y seleccione otro color
3	*NumericUpDown*	Grosor de la línea de la gráfica
4	*NumericUpDown*	Separación vertical de la malla
5	*Button*	Cierra la ventana y ejecuta los cambios especificados
6	*Button*	Cierra la ventana sin efectuar cambio alguno sobre la señal

4.2.5.7. Ventana Espectros de respuesta

Esta ventana recibe los parámetros para generar espectros de respuesta o de piso según sea el caso. El programa maneja bloques de espectros, éstos son grupos de 4 respuestas: desplazamiento, velocidad y aceleraciones.

Para cada grupo de espectros el usuario debe definir el amortiguamiento y las condiciones iniciales. Cada uno debe ser diferente del otro al menos en un valor, porque si el programa detecta que los parámetros son exactamente los mismos automáticamente suprimirá los grupos de que se repitan.

Independientemente del grupo de espectros que se defina, el usuario debe introducir los datos de periodo inicial, final e intervalo de tiempo entre estos dos.

Para los espectros de repuesta, el programa sólo permite que se definan un total de 500 puntos y para los espectros de piso sólo 100. Para ambos casos el mínimo número de puntos es de 10.

El número de puntos para el espectro se obtiene con la siguiente manera:

$$puntos = \frac{Tf - Ti}{\Delta t} \quad (4.7)$$

Donde:

Ti = Periodo inicial en s
Tf = Periodo inicial en s
Δt = Intervalo de tiempo en s

En el caso de los espectros de respuesta, se pueden definir de 1 a 3 bloques, pero para el caso de los espectros de piso sólo puede haber uno.

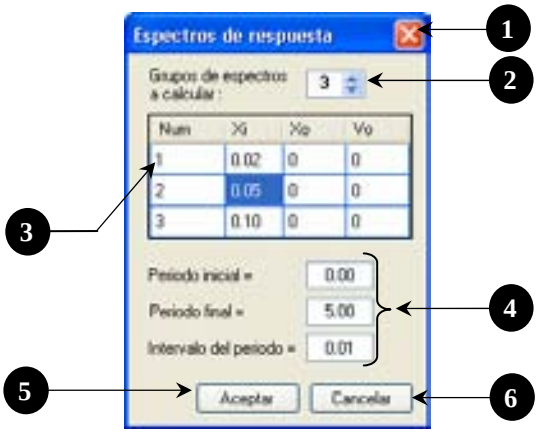


Figura 4.41 Ventana para generar Espectros de respuesta

#	Control	Descripción
1	Botón Cerrar	Cierra la ventana sin generar ningún espectro
2	*Numeric Up Down*	Establece el número de bloques de espectros que habrá en el control *Data Grid View*
3	*Data Grid View*	Recibe los parámetros para cada uno de los bloques de espectros, el encabezado de cada columna indica a que corresponde cada valor
4	*TextBox*	Recibe los datos de los periodos final, inicial e intervalo entre puntos
5	*Button*	Acepta los parámetros para generar los espectros, aparecerá un mensaje que avisa al usuario cuantos bloques de espectros va a calcular (ver figura 42a)
6	*Button*	Cierra la ventana sin generar ningún espectro

Al hacer clic en el botón aceptar (figura 4.41) aparece la ventana (figura 4.42a); si el usuario está de acuerdo con el número de bloques de espectros a calcular al hacer clic en aceptar se abrirá la ventana (figura 4.42b) con una barra progresiva que va indicando el avance del cálculo.

(a) **(b)**

Figura 4.42 Ventanas auxiliares

En caso contrario, al hacer clic en el botón cancelar (figura 4.42a), la ventana se cierra y el programa regresa a la ventana de los espectros de respuesta (figura 4.41) para que el usuario modifique los parámetros que sean necesarios. En el título de la ventana (figura 4.42b) indica el grado de libertad que se está calculando; para el caso de los espectros de respuesta siempre se mostrará la leyenda GL=1.

CAPÍTULO 5

ALGORITMOS

5.1. RESPUESTA DINÁMICA

5.1.1. Respuesta de osciladores de 1GL

Para obtener la respuesta dinámica de un oscilador de 1GL es necesario resolver la ecuación de equilibrio dinámico (2.10). No es posible obtener una solución analítica de la ecuación de movimiento si la carga $p(t)$ o la aceleración del suelo $\ddot{x}_S(t)$ varían arbitrariamente con el tiempo y dado que el objetivo del programa es calcular y graficar la respuesta de los osciladores ante cualquier tipo de excitación que el usuario defina, es necesario utilizar un método numérico para resolver la ecuación de movimiento (2.10).

Existen varios métodos que resuelven la ecuación de equilibrio:

- Métodos basados en la interpolación lineal de la función de la excitación.
- Métodos basados en expresiones de diferencia finita de velocidad y aceleración
- Métodos que asumen variación de aceleración

No obstante, todos deben cumplir con tres características importantes: ser precisos, estables y deben converger a la solución exacta del problema.

Para utilizar un método numérico es necesario que la aplicación de la señal de la excitación $p(t)$ sea dada por valores discretizados:

$p_i = p(t_i),\quad Desde\ i = 0\ \ hasta\ i = N$

Donde:

i = instante de tiempo *discretizados*

N = número total de puntos de la señal de exitación

Para determinar la respuesta dinámica se utilizó una aproximación numérica de la integral de Duhamel, conocido también como el método de las *ocho constantes*, que es un método numérico altamente eficiente para sistemas lineales. Consiste en determinar la respuesta de desplazamiento y velocidad para cada intervalo de tiempo por medio de la ayuda de constantes (4 para desplazamiento y 4 para velocidad) que se calculan una sola vez en el procedimiento (Ordaz, 2006).

El método es especialmente eficiente (Chopra, 2001) cuando el registro de aceleraciones tiene un intervalo de tiempo (Δt) constante entre cada dato del registro, y además, supone una relación lineal entre ellos. Si los intervalos (Δt) son pequeños la interpolación lineal se considera satisfactoria.

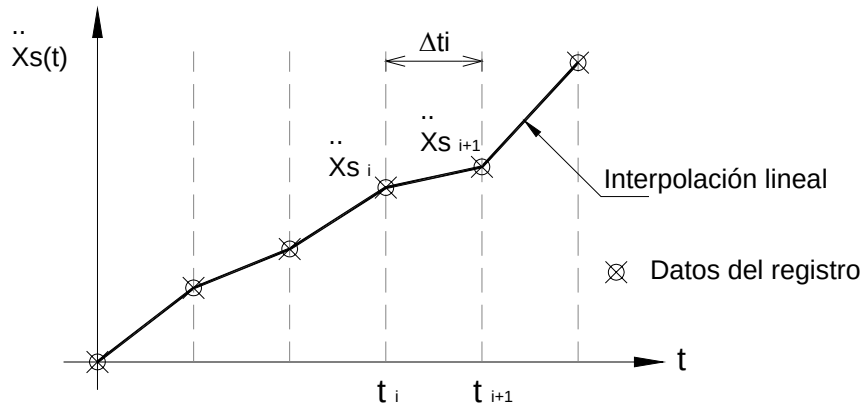


Figura 5.1 Función de carga arbitraria

La figura 5.1 muestra la representación de un registro de aceleraciones arbitrarias con intervalo Δti, donde *i* es un instante de tiempo que puede valer desde 1 hasta N. El intervalo de tiempo se calcula como:

$$\Delta t_i = t_{i+1} - t_i \quad \text{.....(5.1)}$$

El método de las *ocho constantes* resuelve la ecuación 2.10 y obtiene dos expresiones: una para calcular el desplazamiento (5.2) y otra para la velocidad (5.3), ambas en cada instante de tiempo i.

$$x_{i+1} = \alpha_1 x_i + \alpha_2 \dot{x}_i + \alpha_3 \ddot{x}s_i + \alpha_4 \ddot{x}s_{i+1} \quad \text{.....(5.2)}$$

$$\dot{x}_{i+1} = \beta_1 x_i + \beta_2 \dot{x}_i + \beta_3 \ddot{x}s_i + \beta_4 \ddot{x}s_{i+1} \quad \text{.....(5.3)}$$

Los coeficientes *α1, α2, α3 y α4* son las constantes para desplazamiento y se obtienen con las ecuaciones 5.10 a 5.13 respectivamente. Asimismo, los coeficientes *β1, β2, β3 y β4* son las constantes para calcular la velocidad y se obtienen con las ecuaciones 5.14 a 5.17.

Para calcular la aceleración relativa simplemente se despeja $\ddot{x}(t)$ de la ecuación 2.10 obteniendo la siguiente ecuación para cada instante i.

$$\ddot{x}_i = -\ddot{x}_{S\,i} - 2\xi\Omega\dot{x}_i - \Omega^2 x_i \quad \text{.....(5.4)}$$

La aceleración absoluta para cada intervalo de tiempo se calcula como:

$$\ddot{x}_{Ai} = \ddot{x}_i + \ddot{x}_{Si} \quad \text{.....(5.5)}$$

Para simplificar un poco las ecuaciones 5.10 a 5.17, se hicieron las siguientes sustituciones:

$$E = e^{-\xi\Omega\Delta t} \quad (5.6)$$

$$S = \text{sen}(\Omega_D \Delta t) \quad (5.7)$$

$$C = \cos(\Omega_D \Delta t) \quad (5.8)$$

$$\Omega_D = \Omega\sqrt{1-\xi^2} \quad (5.9)$$

Las siguientes ecuaciones son las conocidas como las ocho constantes:

$$\alpha_1 = E\left(\frac{\xi \cdot \Omega \cdot S}{\Omega_D} + C\right) \quad (5.10)$$

$$\alpha_2 = \frac{E \cdot S}{\Omega_D} \quad (5.11)$$

$$\alpha_3 = \frac{\left[\frac{\left(\xi\Omega + \frac{2\xi^2 - 1}{\Delta t}\right)S}{\Omega_D} + \left(1 + \frac{2\xi}{\Omega\Delta t}\right)C\right]E - \frac{2\xi}{\Omega\Delta t}}{\Omega^2} \quad (5.12)$$

$$\alpha_4 = \frac{\left[-\frac{\left(2\xi^2 - 1\right)S}{\Omega_D \Delta t} - \frac{2\xi C}{\Omega\Delta t}\right]E - 1 + \frac{2\xi}{\Omega\Delta t}}{\Omega^2} \quad (5.13)$$

$$\beta_1 = -\frac{E\Omega S}{\sqrt{1-\xi^2}} \quad (5.14)$$

$$\beta_2 = \left(-\frac{E\Omega S}{\Omega_D} + C\right)E \quad (5.15)$$

$$\beta_3 = \frac{1 + \left(-\frac{\left(\Omega\Delta t + \xi\right)S}{\sqrt{1-\xi^2}} - C\right)E}{\Omega^2 \Delta t} \quad (5.16)$$

$$\beta_4 = \frac{\left(\frac{\xi S}{\sqrt{1-\xi^2}} + C\right)E - 1}{\Omega^2 \Delta t} \quad \text{.....(5.17)}$$

Para cada valor discretizado de la señal de excitación se genera un valor de la respuesta dinámica; el número total de puntos (N) de las señales de excitación puede variar, pero cuando este valor comienza a ser muy grande, calcular la respuesta para cada punto puede ser muy tardado; no obstante, para la computadora es muy rápido calcular las respuestas.

El método resulta ser muy práctico, pues para calcular la respuesta dinámica de un oscilador sólo es necesario calcular una sola vez las ocho constantes y sustituirlas en las ecuaciones 5.2 y 5.3 para cada instante de tiempo i.

Para implementar la respuesta dinámica al programa, se creó la clase *Respuesta1GL.dll*, que genera el cálculo de la respuesta de un oscilador de 1GL sujeto a cualquier tipo de excitación en su base.

La clase a través de su único constructor *Sub New* (código 5.1) recibe los siguientes parámetros para calcular la respuesta:

Tabla 5.1 Parámetros que necesita el constructor de la clase *Respuesta1GL*

T	Periodo en segundos
Xi	Amortiguamiento
Xo	Desplazamiento inicial
Vo	Velocidad inicial
DatosSeñal	Registro completo de la señal de excitación
MaxP	Número total de puntos de la señal
dt	Intervalo de tiempo

```
    Sub New(ByVal T As Double, ByVal Xi As Double, ByVal x0 As Double, ByVal v0 As Double,
ByVal DatosSeñal() As Double, ByVal MaxP As Integer, ByVal dt As Double)

        Me.Omega = (2 / T) * PI       'Calcula la frecuencia apartir del periodo
        Me.amort = Xi                 'Pasa el valor del amortiguamiento
        Me.Xo = x0                    'Indica el desplazamiento inicial
        Me.Vo = v0                    'Indica la velocidad inicial
        Me.OmegaD = Me.Omega * Math.Sqrt(1 - Me.amort ^ 2)  'cálculo de la omega D

        Me.MaxP = MaxP                'Número total de puntos de la señal de exitación
        Me.deltaT = dt                'Intervalo de tiempo

        Call variablesESC()                         'Invoca a la función
        Call ochoconstantesA()                      'Invoca a la función
        Call metodoJaramilloA(DatosSeñal, Me.MaxP)  'Invoca a la función
    End Sub
```

Código fuente 5.1 Constructor de la clase Respuesta1GL.dll

En el cuerpo del constructor (código 5.1), se calculan las variables *Omega* y *OmegaD*; además, se pasan los valores de los parámetros mencionados a las variables globales de la clase.

Posteriormente se invocan las funciones por medio de la instrucción *Call*, dichas funciones se encuentran en los códigos 5.2, 5.3 y 5.4.

El código fuente 5.2 calcula las ecuaciones 5.6 a 5.8, que serán utilizadas para calcular las ocho constantes.

```
    Private Sub variablesESC()
        Me.E = Exp(-Me.amort * Me.Omega * Me.deltaT)
        Me.S = Sin(Me.OmegaD * Me.deltaT)
        Me.C = Cos(Me.OmegaD * Me.deltaT)
    End Sub
```

Código fuente 5.2 Función para calcular las variables E, S y C

El código fuente 5.3 muestra la función que calcula las ocho constantes; ésta no recibe ningún parámetro pues los valores que necesita para calcular las constantes se declararon como variables globales dentro de la clase y dichas variables se les asignó valor en el código fuente 5.1.

Para la función se declararon algunas variables auxiliares (ax1, ax2, ax4, bx1, bx2, bx3, bx4 y bx5) que simplifican un poco el cálculo de algunas constantes. Dentro del código, las ocho constantes son: para desplazamiento ctes.a1, ctes.a2, ctes.a3 y ctes.a4 que corresponden a las ecuaciones 5.10 a 5.13 respectivamente y para velocidad ctes.b1, ctes.b2, ctes.b3 y ctes.b4 que corresponden a su vez a las ecuaciones 5.14 a 5.17.

```
Private Sub ochoconstantesA()
    Dim ax1, ax2, ax4, bx1, bx2, bx3, bx4, bx5 As Double

    '<<<<<<<<<< Desplazamiento  >>>>>>>>>>>>>>>>>

    ctes.a1 = E * ((amort * Omega * S / OmegaD) + C)
    ctes.a2 = E * S / OmegaD

    ax1 = ((amort * Omega) + ((2 * amort ^ 2) - 1) / deltaT) * S / OmegaD
    ax2 = (2 * amort) / (deltaT * Omega)
    ctes.a3 = ((ax1 + (1 + ax2) * C) * E - ax2) / Omega ^ 2

    ax4 = ((2 * amort ^ 2) - 1) * S / (OmegaD * deltaT)
    ctes.a4 = ((-ax4 - ax2 * C) * E - 1 + ax2) / Omega ^ 2

    '<<<<<<<<<< Velocidad  >>>>>>>>>>>>>>>>>

    ctes.b1 = (-1 * E * S * Omega) / Sqrt(1 - amort ^ 2)
    ctes.b2 = (((-amort * Omega * S) / OmegaD) + C) * E

    bx1 = amort * (-1 + amort * deltaT * Omega + 2 * amort ^ 2) / (Omega * OmegaD * deltaT)
    bx2 = (deltaT * Omega + 2 * amort) * OmegaD / (deltaT * Omega ^ 3)
    bx3 = amort * (deltaT * Omega + 2 * amort) / (deltaT * Omega ^ 2)
    bx4 = (-1 + amort * deltaT * Omega + 2 * amort ^ 2) / (deltaT * Omega ^ 2)
    bx5 = 1 / (Omega ^ 2 * deltaT)

    ctes.b3 = ((-bx1 - bx2) * S + (-bx3 + bx4) * C) * E + bx5
    ctes.b4 = (-1 + E * (amort * S / (Sqrt(1 - amort ^ 2)) + C)) / (deltaT * Omega ^ 2)
End Sub
```

Código fuente 5.3 Función para calcular las ocho constantes

Las ocho constantes también fueron declaradas como globales; por lo tanto, se pueden usar en cualquier parte del código de la clase.

El código fuente 5.4 contiene las ecuaciones para calcular la respuesta dinámica para cada instante de tiempo *i*. Esta función del tipo *Private sub* con nombre "metodoJaramilloA", recibe un sólo parámetro para calcular la respuesta: el vector completo de la excitación "acelerograma ( )".

```
Private Sub metodoJaramilloA(ByVal acelerograma() As Double)

    ReDim Me.X(Me.MaxP)
    ReDim Me.V(Me.MaxP)
    ReDim Me.Ar(Me.MaxP)
    ReDim Me.Aa(Me.MaxP)

    Me.X(1) = Me.Xo : Me.V(1) = Me.Vo
    Me.Ar(1) = -acelerograma(1) - 2 * amort * Omega * V(1) - Omega ^ 2 * X(1)
    Me.Aa(1) = Ar(1) + acelerograma(1)

    For Me.i = 2 To Me.MaxP
        Me.X(i) = ctes.a1 * X(i - 1) + ctes.a2 * V(i - 1) + ctes.a3 * acelerograma(i - 1) +
ctes.a4 * acelerograma(i)
        Me.V(i) = ctes.b1 * X(i - 1) + ctes.b2 * V(i - 1) + ctes.b3 * acelerograma(i - 1) +
ctes.b4 * acelerograma(i)
        Me.Ar(i) = -acelerograma(i) - 2 * amort * Omega * V(i) - Omega ^ 2 * X(i)
        Me.Aa(i) = Ar(i) + acelerograma(i)
    Next i
End Sub
```

Código fuente 5.4 Función para calcular la respuesta dinámica

Los 4 renglones que comienzan con la leyenda *Redim* indican que el programa dimensiona a los vectores de desplazamiento X, velocidad V, aceleración relativa Ar y absoluta Aa, que contendrán a la respuesta dinámica, con el mismo número de puntos que tiene la señal (MaxP).

Posteriormente se indican los valores para la respuesta en el instante i=1, en el caso de X (1) y V (1) corresponden a las condiciones iniciales (desplazamiento y velocidad respectivamente). Para las aceleraciones (relativa y absoluta) se calculan los valores en el mismo instante de tiempo i=1, utilizando las ecuaciones 5.4 y 5.5 respectivamente.

Para el cálculo de la respuesta dinámica a partir del segundo punto (i=2) hasta el número total de puntos N, se hace a con la ayuda de un bucle *for* que alberga las ecuaciones 5.2 a 5.5 para cada instante *i*; los valores que se calculan en cada instante i se van almacenando en los vectores que se definieron al inicio de la función.

Como se puede observar, el método es bastante práctico pues permite calcular la respuesta dinámica con pocas ecuaciones y a su vez el código de programación es pequeño.

Para obtener los vectores con la respuesta dinámica, se creó una función (código fuente 5.5) que recibe como parámetros 4 vectores vacíos del tipo *double* donde se guardará la respuesta dinámica. Lo que hace la función es pasar por referencia los vectores completos de la respuesta dinámica que se calculó en el código fuente 5.4.

```
Sub Resulatdos(ByRef respX() As Double, ByRef respV() As Double, ByRef respAr() As Double,
ByRef respAa() As Double)
        respX = Me.X
        respV = Me.V
        respAr = Me.Ar
        respAa = Me.Aa
End Sub
```

Código fuente 5.5 Función que regresa los vectores de respuesta

La clase cuenta con la función “CalculaMaximos” (código fuente 5.6) para obtener el valor máximo absoluto de cada tipo de respuesta (X, V, Ar y Aa); para esto el programa barre los vectores de cada respuesta y va calculando el máximo para cada una a través de un ciclo *for* y una sentencia *if* para cada tipo respuesta.

Los valores máximos se van almacenando en el vector “Max” de tipo *double* con 4 dimensiones, que corresponden a los tipo de respuesta: X, V, Ar y Aa.

A través de la función “ValoresMaximos” del tipo *ReadOnly Property* (mostrada en la parte baja del código fuente 5.6), la clase exporta los valores máximos calculados en la función “CalculaMaximos”.

```
    Private Sub CalculaMaximos()
        Max(1) = 0 : Max(2) = 0 : Max(3) = 0 : Max(4) = 0

        For i As Integer = 1 To Me.MaxP
            If Abs(Max(1)) < Abs(Me.X(i)) Then
                Max(1) = Me.X(i)
            End If

            If Abs(Max(2)) < Abs(Me.V(i)) Then
                Max(2) = Me.V(i)
            End If

            If Abs(Max(3)) < Abs(Me.Ar(i)) Then
                Max(3) = Me.Ar(i)
            End If

            If Abs(Max(4)) < Abs(Me.Aa(i)) Then
                Max(4) = Me.Aa(i)
            End If
        Next
    End Sub

'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
    Public ReadOnly Property ValoresMaximos()
        Get
            Call CalculaMaximos()
            Return Max
        End Get
    End Property
```

Código fuente 5.6 Función para obtener las respuestas máximas

Ahora bien, para llamar a la clase *Respuesta1GL* desde el programa principal, se hace con en el código fuente 5.7. Para esto se declara un objeto llamado “ObjCalculo” seguido de los parámetros (tabla 5.1) que necesita la clase para calcular la respuesta.

La declaración del objeto “ObjCalculo” se hizo dentro de un bucle *for*; esto es porque el programa calcula las respuestas de 1, 2 o 3 osciladores, dependiendo de cuantos osciladores defina el usuario.

Posteriormente, la variable “ObjCalculo” hace la referencia a la función *Resultados* (código fuente 5.5) y pasa como parámetros cuatro vectores (RX, RV, RAr y RAa) para recibir la respuesta dinámica. Cada vector es seguido del número de oscilador al que corresponde; por ejemplo RX1: respuesta de desplazamiento del oscilador 1.

La referencia a la función *Resultados* se hace dentro de una sentencia *Select Case* porque para cada oscilador se almacena la respuesta en vectores diferentes.

```
Private Sub CalculoRespuesta(ByVal Oscil As Integer)

  For n As Integer = 1 To Oscil
     Dim ObjCalculo As New CalculoRes1GL.Respuesta1GL(Propiedades(n, 1), Propiedades(n, 2),
Propiedades(n, 3), Propiedades(n, 4), DatosGráficaSeñal, maxpuntos, deltaT)

            Select Case n
                Case 1
                    ObjCalculo.Resulatdos(RX1, RV1, RAr1, RAa1)
                Case 2
                    ObjCalculo.Resulatdos(RX2, RV2, RAr2, RAa2)
                Case 3
                    ObjCalculo.Resulatdos(RX3, RV3, RAr3, RAa3)
            End Select

            Dim ValMax() As Double = ObjCalculo.ValoresMaximos
            For Me.i = 1 To 4
                RespMax(n, i) = ValMax(i)
            Next

     Call CalcularVbasal(n)
  Next n
End Sub
```

Código fuente 5.7 Uso de la clase en el programa

En el código fuente 5.7 también se puede observar que se hace referencia a la función *ValoresMaximos* para obtener la máxima amplitud de cada respuesta. Para esto, los valores se almacenan primero en un vector de tipo *double* (ValMax).

Posteriormente con la ayuda de un ciclo *for* se pasan los valores a una matriz bidimensional llamada *RespMax*; su tamaño fue determinado previamente con 5 columnas y tantas filas como osciladores haya definido el usuario.

5.1.2. Espectros de respuesta

Es un gráfico (figura 5.2) que muestra la respuesta máxima (expresada en términos de desplazamiento, velocidad o aceleraciones) que produce una acción dinámica determinada en varios osciladores de 1GL.

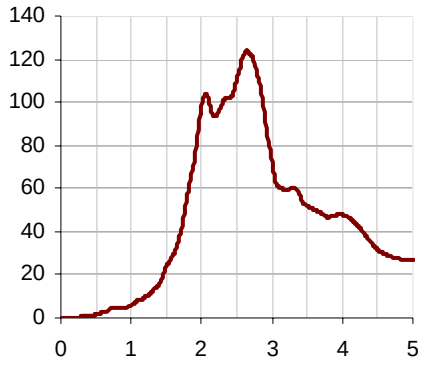


Figura 5.2 Espectro de respuesta

En el eje de las absisas (X) se encuentra el periodo propio de la estructura (o la frecuencia) y en las ordenadas (Y) la respuesta máxima calculada para cada periodo T del oscilador, con el mismo factor de amortiguamiento.

Los espectros se utilizan fundamentalmente para estudiar las características del terremoto y su efecto sobre las estructuras. Las curvas de los espectros presentan variaciones bruscas, con numerosos picos y valles, que resultan de la complejidad del registro de aceleraciones del terremoto. El concepto de espectro de respuesta es una importante herramienta de la dinámica estructural, de gran utilidad en el área de diseño sismorresistente. (Crisafulli y Villafañe *et. al* 2002)

El programa calcula sólo espectros de respuesta lineales, debido a que la respuesta que se obtiene es de osciladores elásticos lineales.

Para construir un espectro de respuesta es necesario calcular la respuesta dinámica de varios osciladores de 1GL con diferentes periodos de vibrar T y con igual factor de amortiguamiento. Para todos y cada uno de ellos se exhibirá una respuesta diferente.

Una vez calculada la respuesta de los osciladores, se determina el máximo (en valor absoluto, dado que el sigo no tiene importancia) de cada uno de ellos y se coloca en un gráfico en función del periodo de vibración, para obtener así un espectro de respuesta. Es decir, que la respuesta máxima de cada oscilador con periodo T representa un punto del espectro.

El programa calcula los espectros de respuesta con la ayuda de un control de tipo *Timer* llamado "TmEspectros"; el cual llama a la función "DatosPreliminaresRespuesta" para cada punto del espectro a intervalos de tiempo de 1 milisegundo.

El código fuente 5.8 muestra la función "DatosPreliminaresRespuesta", que entre otras instrucciones se encuentran el llamado a las funciones *CalculodeRespuesta* (código fuente 5.7) para determinar la respuesta dinámica, e inmediatamente después llama a la función *CrearMatrizEspectros* para construir el vector del espectro con los valores máximos de cada tipo respuesta.

```
Private Sub DatosPreliminaresRespuesta(ByVal gráficas As Integer, ByVal CalculaBloque As
Boolean)
……
…….
Call CalculoRespuesta(NumOscil)             'Calcula la respuesta del sistema
Call CrearMatrizEspectros(p, NumEspectros)  'Crea la matriz de espectros
……
…….
End Sub
```

Código fuente 5.8 Función DatosPrelimaresRespuesta

Obsérvese que el valor máximo de cada respuesta, para un valor de T, se calculó en el código fuente 5.7 y se guardó en la matriz "Respmax"; no obstante, esta matriz sólo puede almacenar las respuestas máximas de tres osciladores, por lo que no es suficiente para almacenar cada punto del espectro que puede tener hasta 500 datos por respuesta.

En este caso, se utiliza la función "CrearMatrizEspectros" (código fuente 5.9) que va guardando las respuestas máximas para cada cambio de valor del periodo T (punto del espectro) en la matriz "PuntoMax".

Se declaró a la matriz "PuntosMax" como una matriz de matrices de tres dimensiones. Este tipo de matrices tiene una ventaja importante con respecto a las matrices comunes, y es que sus elementos pueden ser también matrices o vectores, de ahí el nombre de matriz de matrices.

El programa puede calcular de 1 a 3 bloques de espectros; cada bloque contiene un grupo de 4 respuestas (D, V, Ar y Aa) y cada respuesta es un vector de datos que corresponden a los valores máximos de cada respuesta. Recuperar los datos almacenados en la matriz "PuntosMax" sería mucho más laborioso de no ser una matriz de matrices, ya en que de esta manera sólo se llama al vector completo que contiene a un espectro en específico.

```
    Private Sub CrearMatrizEspectros(ByVal n As Integer, ByVal BloqueEsp As Integer)
        For j = 1 To BloqueEsp
            PuntosMax(j)(1)(n) = Math.Abs(RespMax(j, 1)) 'Desplazamientos
            PuntosMax(j)(2)(n) = Math.Abs(RespMax(j, 2)) 'Velocidad
            PuntosMax(j)(3)(n) = Math.Abs(RespMax(j, 3)) 'Aceleración relativa
            PuntosMax(j)(4)(n) = Math.Abs(RespMax(j, 4)) 'Aceleración absoluta
        ……..
        ……..
        Next
    End Sub
```

Código fuente 5.9 Creación de la matriz para espectros de respuesta

5.1.3. Fuerza cortante

Es la fuerza interna del oscilador que se genera por ejercer un desplazamiento horizontal en él. Una vez calculada la respuesta de desplazamiento $x(t)$ por el análisis dinámico del oscilador, las fuerzas internas pueden determinarse mediante un análisis estático para cada instante de tiempo *i*, a través de la ecuación 2.1 basada en el concepto de la fuerza estática equivalente.

$$f_S(t) = k \cdot x(t)$$

Para un oscilador simple la fuerza cortante basal (V_B) se puede determinar a partir de la ecuación 5.18. La figura 5.3 muestra la representación gráfica de las fuerzas Fs y V_B.

$$V_B(t) = f_S(t) \quad \text{.....(5.18)}$$

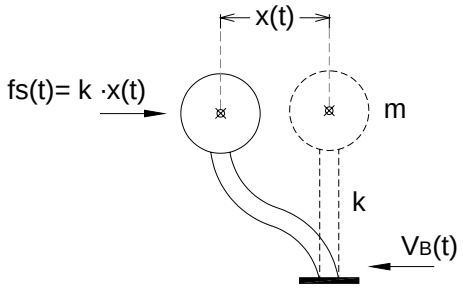


Figura 5.3 Fuerza cortante

En el código 5.7 hay una línea con la sentencia *Call CalcularVbasal(n)*, la cual invoca a la función para determinar la fuerza cortante; ésta se muestra en el código fuente 5.10.

```
Private Sub CalcularVbasal(ByVal Osc As Integer)
    Select Case Osc
        Case 1
            ReDim Fs1(maxpuntos)       'Fuerza cortante del 1er oscilador
            For j = 1 To maxpuntos
                Fs1(j) = K1GL(1) * RX1(j)
            Next j
        Case 2
            ReDim Fs2(maxpuntos)       'Fuerza cortante del 2do oscilador
            For j = 1 To maxpuntos
                Fs2(j) = K1GL(2) * RX2(j)
            Next j
        Case 3
            ReDim Fs3(maxpuntos)       'Fuerza cortante del 3er oscilador
            For j = 1 To maxpuntos
                Fs3(j) = K1GL(3) * RX3(j)
            Next j
    End Select
End Sub
```

Código fuente 5.10 Calculo del cortante basal

La función, dentro del código fuente 5.10, requiere de un parámetro (número del oscilador) para calcular la fuerza cortante; dependiendo de este valor del parámetro (1, 2 o 3) el programa calcula la fuerza fs, con la ecuación 2.10, para cada instante de tiempo i.

5.1.4. Respuesta de osciladores de VGL

Al igual que los osciladores de 1GL, para encontrar la respuesta de un oscilador de VGL es necesario resolver la ecuación de equilibrio dinámico; para este caso se trata de la ecuación 2.20. Las características que tiene la señal excitación son las mismas: variar arbitrariamente con el tiempo y su registro debe estar discretizado en intervalos de tiempo pequeños.

Para el oscilador de VGL, descrito en la sección 2.3 del capítulo 2, las matrices de masas y rigidez se formulan de la siguiente manera:

$$m = \begin{bmatrix} m_1 & & & \\ & m_2 & & \\ & & \ddots & \\ & & & m_N \end{bmatrix} \qquad k = \begin{bmatrix} k_1 + k_2 & -k_2 & & \\ -k_2 & k_N + k_{N+1} & -k_{N+1} & \\ & -k_{N+1} & \ddots & -k_{N+1} \\ & & -k_{N+1} & k_N \end{bmatrix}$$

Debe notarse que la matriz de masa es diagonal para este sistema, sin considerar acoplamiento entre las masas; no obstante, en la matriz de rigidez se presentan acoplamientos entre los valores, lo que complica la solución de la ecuación de equilibrio dinámico.

La obtención de la respuesta dinámica de este oscilador se hizo a través del método de la superposición modal. Que consiste en modificar el sistema de referencia de tal forma que se pueda determinar la respuesta para cada grado de libertad como un sistema de 1GL

En el método de la superposición modal, la determinación de los eigenvectores es la fase más importante y que consume, relativamente, más tiempo dentro de la solución. (Pérez, 1990)

Uno de los métodos para obtener los eigen valores es el de la sub estructuración dinámica, que reduce el sistema de n grados de libertad a otro de menor tamaño, pero de tal manera que los vectores y valores característicos de baja frecuencia se conservan. (Pérez, 1990)

El oscilador de VGL (considerado en este trabajo) se razona como un sistema reducido de grados de libertad. En general, las estructuras tienen un gran número de grados de libertad, pero estos se pueden reducir de manera significativa de tal manera que el sistema se simplifica.

Una de las maneras de reducir el sistema de n grados de libertad, es por medio del método de Guyan, que reduce simultáneamente a las matrices de rigidez y masas del problema, en base a la selección de grados de libertad maestros; estos últimos son aquellos en los que se supone que está concentrada la masa de la estructura. Después de la reducción, sólo los grados de libertad maestros quedan representados en el oscilador (Pérez, 1990).

Este sistema parece razonable en estructuras como lo son los edificios regulares, pues a menudo se suponen las siguientes condiciones (Paz, 1992):

- Toda la masa de la estructura está concentrada al nivel de los pisos. Esto transforma el sistema, con un número infinito de grados de libertad a un sistema que tiene solamente tantos grados de libertad como número de masas concentradas.

- Las vigas en los pisos son infinitamente rígidas, con relación a la rigidez de las columnas. Esto introduce el requisito de que las uniones entre las vigas y las columnas estén fijas sin rotación.

- La deformación de la estructura es independiente de las fuerzas axiales presentes en las columnas. Establece que las vigas rígidas en los pisos permanezcan horizontales durante el movimiento de la estructura.

En este caso no se requieren más que número muy reducido de coordenadas para determinar, con mucha precisión, la respuesta de la estructura.

Por otro lado, el método de reducción de grados de libertad se vuelve incapaz de representar adecuadamente la respuesta de la estructura cuando se trata de una excitación que contenga componentes de alta frecuencia porque en general la configuración de las estructuras ante tal tipo de excitación es compleja y requiere en gran número de coordenadas para ser representada adecuadamente. (Pérez, 1990)

En el presente trabajo no se trata el tema de cómo reducir los grados de libertad de una estructura, sino que, para usar el programa, el usuario previamente debe haber reducido los grados de libertad de manera que la estructura se asemeje al oscilador de VGL descrito en esta tesis.

Para ello es necesario calcular las frecuencias y modos de vibrar (formas modales) del oscilador; el sistema de VGL tendrá tantas frecuencias como grados libertad.

La siguiente ecuación se obtiene de analizar el oscilador de VGL cuando está sujeto a vibración libre; en este caso no existen fuerzas externas y su amortiguamiento es considerado cero.

$$\left(\left[k\right]-\Omega^2\cdot\left[m\right]\right)\cdot\phi=0 \quad \text{.....(5.19)}$$

La ecuación 5.19 es un problema de valores característicos que tiene una solución no trivial sólo si el determinante de los coeficientes es igual a cero, es decir, las frecuencias naturales Ω y los modos de vibrar deben satisfacer la ecuación 5.20.

$$Det\left(\left[k\right]-\Omega^2\cdot\left[m\right]\right)=0 \quad \text{.....(5.20)}$$

El desarrollo del determinante (ecuación 5.20) conduce a un polinomio de grado n, las raíces del cual son los valores de las frecuencias del oscilador al cuadrado (Ω^2). Sustituyendo estos valores en la ecuación 5.19 se obtienen los valores para cada modo de vibrar (ϕ); estos pueden ser acomodados en forma matricial de la siguiente manera:

$$\Omega^2=\begin{Bmatrix}\Omega_1\\ \Omega_2\\ \vdots\\ \Omega_N\end{Bmatrix} \qquad \phi=\begin{bmatrix}\phi_{11} & \phi_{21} & \cdots & \phi_{N1}\\ \phi_{12} & \phi_{22} & \cdots & \phi_{N2}\\ \vdots & \vdots & \ddots & \vdots\\ \phi_{1N} & \phi_{2N} & \cdots & \phi_{NN}\end{bmatrix}$$

Con la matriz ϕ es posible hacer el cambio de coordenadas mediante la ecuación de la superposición modal:

$$x=\phi\cdot y \quad \text{.....(5.21)}$$

Por lo tanto las matrices de masas, rigidez y amortiguamiento quedan de la siguiente manera:

$$m_n=\left[\phi^T\right]\cdot\left[m\right]\cdot\left[\phi\right] \quad \text{.....(5.22)}$$
$$k_n=\left[\phi^T\right]\cdot\left[k\right]\cdot\left[\phi\right] \quad \text{.....(5.23)}$$
$$C_n=\left[\phi^T\right]\cdot\left[c\right]\cdot\left[\phi\right] \quad \text{.....(5.24)}$$

Sustituyendo estas ecuaciones por sus correspondientes en la ecuación 2.20, se obtiene:

$$\left[\phi^T\right]\cdot\left[m\right]\cdot\left[\phi\right]\cdot\ddot{y}+\left[\phi^T\right]\cdot\left[c\right]\cdot\left[\phi\right]\cdot\dot{y}+\left[\phi^T\right]\cdot\left[k\right]\cdot\left[\phi\right]\cdot y=-x_S\cdot\left[\phi^T\right]\cdot\left[m\right]\cdot\left\{j\right\} \quad \text{.....(5.25)}$$

Dividiendo la ecuación entre m_n y sustituyendo los factores de participación (Fp) se obtiene:

$$Fp=\frac{\left[\phi^T\right]\cdot\left[m\right]\cdot\left\{j\right\}}{\left[\phi^T\right]\cdot\left[m\right]\cdot\left[\phi\right]} \quad \text{.....(5.26)}$$

$$\ddot{y}(t)+2\xi\cdot\Omega\cdot\dot{y}(t)+\Omega^2\cdot y=-Fp\cdot\ddot{x}_S(t) \quad \text{.....(5.27)}$$

Con la ecuación 5.27 es posible resolver cada modo como si fuera un oscilador de 1GL; para resolver la ecuación de movimiento para cada modo, se aplica el método de las ocho constantes explicado en la sección 5.13.

Finalmente, para obtener la respuesta en las coordenadas originales del sistema se aplica de nuevo la ecuación de la superposición modal para cada tipo de respuesta (desplazamiento, velocidad y aceleración relativa respectivamente):

$$\{x_i(t)\} = \sum_{i=1}^{N} \{y_i(t)\} \cdot \phi_i \quad (5.28)$$

$$\{\dot{x}_i(t)\} = \sum_{i=1}^{N} \{\dot{y}_i(t)\} \cdot \phi_i \quad (5.29)$$

$$\{\ddot{x}_i(t)\} = \sum_{i=1}^{N} \{\ddot{y}_i(t)\} \cdot \phi_i \quad (5.30)$$

Para las condiciones iniciales también es necesario hacer el cambio de coordenadas, para ello se utilizan las siguientes ecuaciones:

$$\{y_0\} = [\phi]^{-1} \cdot \{x_0\} \quad (5.31)$$

$$\{\dot{y}_0\} = [\phi]^{-1} \cdot \{\dot{x}_0\} \quad (5.32)$$

El algoritmo se implementó al programa a través de un *Módulo* llamado “Cálculos” (código fuente 5.11); el cual contiene dos funciones declaradas como públicas y dos del tipo *private* (propias del módulo) para llevar a cabo el procedimiento descrito anteriormente.

```
Module Cálculos
    Public Sub ObtenerPropiedades(ByVal Ordenar As String)

    Public Sub ObtenerRespuesta(ByVal DatosGráficaSeñal() As Double, ByVal DeltaT As Single,
ByVal MaxPuntos As Integer)

    Private Sub CalculaVbasal(ByVal maxpuntos As Integer)

    Private Sub EnsambladoK(ByVal Numeracion As String)

End Module
```

Código fuente 5.11 Módulo para obtener la respuesta del oscilador de VGL

La función *Public Sub ObtenerPropiedades* (código fuente 5.12) calcula las propiedades del oscilador de VGL que se muestran en la tabla 5.2.

Tabla 5.2 Propiedades que se calculan para el oscilador de VGL

Ω^2	Frecuencias del sistema al cuadrado (ordenadas de menor a mayor)
ϕ	Matriz de valores característicos (formas modales)
Mn	Matriz de masa modal
Ln	Matriz de carga modal
Kn	Matriz de rigidez modal
qXo	Desplazamientos iniciales en forma modal
qYo	Velocidades iniciales en forma modal
Fp	Factores de participación
Ω	Frecuencias del sistema
T	Periodos del sistema

En las primera líneas del código fuente 5.12 se encuentra la declaración de las matrices involucradas, todas ellas comienzan con la leyenda *Dim* y en algunas casos *ReDim* que significa que esas matrices ya habían sido declaradas como tipo global y que ahí sólo se dimensionaron.

Después de las declaraciones de matrices el programa invoca a la función "EnsambladorK" (código fuente 5.13) pasando un parámetro de tipo *string* (para indicar la forma de numeración), esta función es la que se encarga de crear la matriz de rigidez dependiendo de la forma de numerar los grados de libertad en el oscilador (Arriba hacia abajo o viceversa).

```
Public Sub ObtenerPropiedades(ByVal Ordenar As String)
        Dim Mn(NGL, NGL) As Double    'Matriz de masa modal
        Dim Kn(NGL, NGL) As Double    'Matriz de rigidez modal
        Dim Ln(NGL, 1) As Double      'Matriz de carga modal
        Dim FiT(NGL, NGL) As Double   'Matriz Transpuesta de Fi
        ReDim qXo(NGL, 1)             'Matriz de desplazamientos iniciales Modal
        ReDim qVo(NGL, 1)             'Matriz de velocidades iniciales Modal
        Dim MatA(NGL, NGL) As Double  'Matriz auxiliar
        Dim FiInv(NGL, NGL) As Double 'Matriz inversa de Fi

        ReDim Tvgl(NGL)
        ReDim OmegasVGL(NGL)

        Call EnsambladoK(Ordenar)           'Ensambla las matrices de Rigidez
        Omegas2VGL = EigenVal(KVGL, MasaVGL) 'obtiene las frecuencias al cuadrado, ordenadas

        'Obtiene los eigenvectores normalizados
        Fi = EigenValores(NGL, KVGL, MasaVGL, Omegas2VGL, Ordenar)

        'Ahora pasamos a las coordenadas modales
        FiT = Transpuesta(Fi)
        MatA = MultMatriz(MasaVGL, Fi)
        Mn = MultMatriz(FiT, MatA)

        MatA = MultMatriz(KVGL, Fi)
        Kn = MultMatriz(FiT, MatA)

        MatA = MultMatriz(MasaVGL, Vj)
        Ln = MultMatriz(FiT, MatA)

        FiInv = Inversa(Fi, NGL)
        qXo = MultMatriz(FiInv, XoVGL)
        qVo = MultMatriz(FiInv, VoVGL)

        ReDim MatA(NGL, NGL)
        For i = 1 To NGL
            MatA(i, i) = 1 / (Mn(i, i)) 'inversa de la matriz de masa "Solo Para Masa"
        Next
        FP = MultMatriz(MatA, Ln)       'Factores de particiapción

        For i = 1 To NGL
            OmegasVGL(i) = Sqrt(Omegas2VGL(i))   'Frecuencias del sistema
            Tvgl(i) = 2 * pi / OmegasVGL(i)      'Periodos del sistema
        Next i
    End Sub
```

Código fuente 5.12 Función para calcular las propiedades del oscilador

Con la matriz de rigidez ensamblada y la matriz de masas (creada al introducir los datos) se llama a la función "EigenVal" que calcula las frecuencias de vibrar al cuadrado y guarda el vector en la matriz "Omegas2VGL".

Teniendo las matrices de masas, rigidez y frecuencias al cuadrado se procede a calcular la matriz de valores característicos ϕ, para esto se llama a la función "Eigenvalores ( )" pasando como parámetros el

número de grados de libertad, la matriz de masas, rigidez, y el vector de frecuencias al cuadrado, así como el parámetro de tipo *string* para indicar la forma de numeración de los grados de libertad.

Lo que hace la función es sustituir las Ω^2 en la ecuación 5.19 y despejar cada término para encontrar los valores de ϕ.

```
Private Sub EnsambladoK(ByVal Numeracion As String)
  ReDim KVGL(NGL, NGL)              'Se dimensiona la matriz de rigidez

  If Numeracion = "Arriba" Then 'Crea la matriz k con la numeración de ARRIBA-ABAJO
            KVGL(1, 1) = Rigidez(1)
            For i = 2 To NGL
                KVGL(i, i) = Rigidez(i) + Rigidez(i - 1)
                KVGL(i - 1, i) = -1 * Rigidez(i - 1)
                KVGL(i, i - 1) = -1 * Rigidez(i - 1)
            Next i
  ElseIf Numeracion = "Abajo" Then'Crea la matriz k con la numeración de ABAJO-ARRIBA
            For i = 1 To NGL - 1
                KVGL(i, i) = Rigidez(i) + Rigidez(i + 1)
                KVGL(i + 1, i) = -1 * Rigidez(i + 1)
                KVGL(i, i + 1) = -1 * Rigidez(i + 1)
            Next i
            KVGL(NGL, NGL) = Rigidez(NGL)
        End If
End Sub
```

Código fuente 5.13 Función para ensamblar la matriz de rigidez

Después de calcular la matriz ϕ, el programa calcula las matrices Mn y Kn correspondientes a las ecuaciones 5.22 y 5.23, Ln que corresponde a $\left[\phi^T\right]\cdot[m]\cdot\{j\}$ que posteriormente contribuye a calcular los factores de participación y también calcula las condiciones iniciales (ecuaciones 5.31 y 5.32).

Finalmente la función calcula frecuencias y periodos de vibrar del oscilador con las siguientes expresiones:

$$\Omega = \sqrt{\Omega^2}$$

$$T = \frac{2\pi}{\Omega}$$

Para calcular la respuesta del oscilador se invoca a la función pública “ObtenerRespuesta” (código fuente 5.14) pasando como parámetros: el vector de la excitación “DatosGráficaSeñal ()”, el intervalo de tiempo “DeltaT” y el número total de puntos de la señal “NP”. Las demás variables que necesita la función (Ω^2, ξ, y condiciones iniciales) fueron declaradas como globales, así que se pueden usar en cualquier parte del módulo.

Lo primero que hace la función (código fuente 5.14) es dimensionar las matrices (*Redim*) que almacenaran la respuesta de cada grado de libertad. Estas matrices fueron declaradas como matriz de matrices de dos dimensiones, es decir, que cada dato de la matriz es un vector de datos que corresponde a la respuesta de un grado de libertad en específico.

```
Public Sub ObtenerRespuesta(ByVal DatosGráficaSeñal() As Double, ByVal DeltaT As Single,
ByVal NP As Integer)
        ReDim auxD(NGL)
        ReDim auxV(NGL)
        ReDim auxAr(NGL)
        ReDim auxAa(NGL)

        'se llama a la clase respuesta, se le pasan los arguemtos incluyendo el del numero
        'de grado de libertad para indicar que se trata del sistema de VGL
        For i = 1 To NGL
            Dim calculo As New Respuesta(i, OmegasVGL(i), XhiVGL(i), qXo(i, 1), qVo(i, 1),
DatosGráficaSeñal, NP, DeltaT)
        Next i

        'Ahora las respuestas de cada grado de libertad se deben sumar alterandose por
        'la matriz Fi (vectores caracteristicos) y los Factores de participación
        'El número de nodos a superponer afectara solo en cuantos grados de libertad se
        'suman para obtener la respuesta final

        For i = 0 To NGL
            For j = 1 To NP
                For k = 1 To NMS
                    Dvgl(i)(j) += auxD(k)(j) * Fi(i, k) * FP(k, 1)
                    Vvgl(i)(j) += auxV(k)(j) * Fi(i, k) * FP(k, 1)
                    Arvgl(i)(j) += auxAr(k)(j) * Fi(i, k) * FP(k, 1)
                    Aavgl(i)(j) += auxAa(k)(j) * Fi(i, k) * FP(k, 1)

                Next k
            Next j
        Next i

        Call CalculaVbasal(MaxPuntos) 'Calcula el cortante Basal en cada grado
    End Sub
```

Código fuente 5.14 Función para calcular las propiedades del oscilador VGL

Después, el programa calcula la respuesta dinámica para cada grado de libertad con la ecuación 5.27. En este caso, la señal de excitación no se multiplicó por los factores de participación (FP) como se muestra en la ecuación, ya que por motivos de programación se decidió introducirlos cuando se calcula la respuesta completa. Esto no afecta el resultado de los cálculos ya que los factores de participación siguen siendo introducidos en la ecuación.

Para calcular la respuesta se creó un módulo de clase llamado “Respuesta” (código fuente 5.15), el cual calcula la respuesta dinámica de la misma manera que la clase *Respuesta1GL.dll*, la diferencia es que la clase “Respuesta” recibe un parámetro más (número de grado de libertad) y va guardando la respuesta en matrices especialmente creadas para ello (AuxD (), AuxV (), AuxAr () y AuxAa ()).

Observe que el código fuente 5.15 es similar al código del constructor de la clase “Respuesta1GL” (código fuente 5.1). Las funciones que son invocadas en el constructor de la clase “Respuesta” (código 5.15) son las mismas que se utilizaron anteriormente y se muestran en los códigos fuente 5.2, 5.3 y 5.4 respectivamente.

Al tener calculada la respuesta para cada grado de libertad lo que prosigue en el programa es superponer la respuesta por modos para encontrar la respuesta real en cada grado de libertad. Esto se hace con las ecuaciones 5.28, 5.29 y 5.30, multiplicando además por los FP.

En el código fuente 5.14 se observa la superpoción modal en la parte que muestra tres ciclos *for* anidados, indicando que la respuesta de cada grado de libertad se obtiene multiplicando la matriz auxiliar (Aux ()) correspondiente por la matriz ϕ, y por los factores de participación (FP).

```
Module Respuesta
     Sub New(ByVal grado As Integer, ByVal Frec As Double, ByVal Xi As Double, ByVal x0 As
Double, ByVal v0 As Double, ByVal DatosSeñal1() As Double, ByVal MaxP As Integer, ByVal dt
As Double)

        Me.Omega = Frec
        Me.amort = Xi
        Me.Xo = x0
        Me.Vo = v0
        Me.deltaT = dt
        Me.OmegaD = Me.Omega * Math.Sqrt(1 - Me.amort ^ 2)  'calculo de la omega D

        Call variablesESC()
        Call ochoconstantesA()
        Call metodoJaramilloA(DatosSeñal1, MaxP)

        auxD(grado) = X
        auxV(grado) = V
        auxAr(grado) = Ar
        auxAa(grado) = Aa
    End Sub
End Module
```

Código fuente 5.15 Módulo "Respuesta" para osciladores de VGL

Al mismo tiempo que se van calculando las respuestas modales para cada grado de libertad, estas se van adicionando dependiendo del número de modos a superponer (NMS) que el usuario haya definido.

De esta manera el programa calcula la respuesta dinámica de cada grado.

5.1.5. Espectros de piso

Un espectro de piso es un espectro de respuesta que se obtiene de analizar un oscilador de 1GL ubicado en un nivel de otro oscilador de VGL.

Para visualizar mejor el concepto de espectro de piso, en la figura 5.4 se muestra un oscilador de 1GL ubicado en el segundo nivel de la estructura; el oscilador puede ser un equipo o componente con su propio periodo de vibración.

Para dicho oscilador es posible determinar su espectro de respuesta tomando como excitación ya no el registro de aceleración en la base de la estructura, sino la historia de aceleraciones absolutas en el segundo nivel; al espectro obtenido se le conoce como espectro de piso.

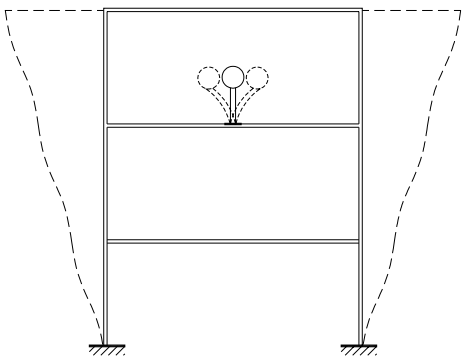


Figura 5.4 Esquema conceptual para espectros de piso

Obviamente, se debe de calcular previamente la respuesta dinámica de la estructura principal (como un oscilador de VGL) considerando como excitación el registro de aceleraciones en su base y posteriormente calcular el espectro de piso.

El análisis de espectro de piso es útil para equipos, componentes o alguna otra estructura pequeña que se ubique dentro del oscilador de VGL y que su masa sea menor en relación a las masas del oscilador de VGL ya que de lo contrario su masa podría interactuar con las demás masas de la estructura principal afectando la ecuación de equilibrio 2.13.

En la figura 5.5 se muestra una gráfica de espectros de piso superpuestos a la misma escala. Se puede observar que las aceleraciones en el equipo o componente a diseñar pueden ser significativamente mayores, dependiendo de su periodo de vibrar, especialmente cuando se acerca al periodo fundamental de la estructura.

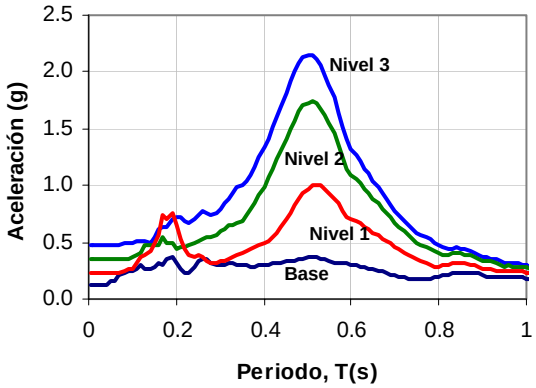


Figura 5.5 Espectros de piso para distintos niveles de la estructura

Los espectros de piso tienen las mismas características que los espectros de respuesta descritos en la sección 5.1.2.

La generación de los espectros de piso en el programa se hizo de manera similar a los espectros de respuesta; sólo que en este caso se requiere de más cálculos, pues el programa calcula cada espectro de piso para cada grado de libertad que tenga el oscilador de VGL.

El programa a través del control *Timer3* (código fuente 5.16), llama a la clase “Respuesta1GL” (código fuente 5.2) que calcula la respuesta para cada periodo T del espectro de respuesta de cada nivel del oscilador de VGL.

El código fuente 5.16 muestra al objeto “ObjCalculo” que hace referencia a la clase “Respuesta1GL.dll” y pasa los parámetros mostrados en la tabla 5.1; a diferencia de los espectros de respuesta aquí la señal de la excitación es el registro de la aceleración absoluta “Aavgl”

El programa, a través de los contadores “p” y “q1”, controla los puntos del espectro y los grados de libertad del oscilador respectivamente. El contador p, tiene un rango de valores de 10 a 200 puntos, que son los puntos del espectro, y el contador q1 tiene rango de valores de 2 a 25, que es el rango de los grados de libertad que puede tener el oscilador de VGL.

Cuando los contadores llegan a sus valores límite, el programa manda un mensaje (con la sintaxis *MsgBox*) que indica que el cálculo de los espectros ha terminado.

```
    Private Sub Timer3_Tick(ByVal sender As Object, ByVal e As System.EventArgs) Handles
Timer3.Tick
        If p < Me.ListaPuntosEsp.Length Then

            Dim ObjCalculo As New CalculoRes1GL.Respuesta1GL(Me.ListaPuntosEsp(p),
Me.P_Espectros(1, 1), Me.P_Espectros(1, 2), Me.P_Espectros(1, 3), Aavgl(q1), maxpuntos,
deltaT)
            ObjCalculo.ValoresMaximos(Me.PuntosEspPiso(q1)(1)(p),
Me.PuntosEspPiso(q1)(2)(p), Me.PuntosEspPiso(q1)(3)(p), Me.PuntosEspPiso(q1)(4)(p))
            p += 1
        Else
            If q1 < Me.NUDGL.Value Then
                q1 += 1
                p = 1
            Else
                Me.Timer3.Stop()
                q1 = 1
                p = 1
                MsgBox("Terminaron los cálculos de los Espectros de Piso",
MsgBoxStyle.Information, version)
            End If
        End If
    End Sub
```

Código fuente 5.16 Función para calcular los espectros de piso

5.1.6. Fuerza cortante

La fuerza cortante en el oscilador de VGL se calcula con la misma ecuación 2.1 que se usa en los osciladores de 1GL, sólo que para aplicarla se introduce el desplazamiento relativo entre las masas del oscilador.

La figura 5.6 muestra un esquema de cómo se toman los desplazamientos (X_1, X_2 y X_3) para calcular la fuerza cortante en cada grado de libertad. La fuerza Vbasal es el resultado de sumar todas las fuerzas cortantes (*fs*) que actúan al mismo tiempo en el oscilador de VGL.

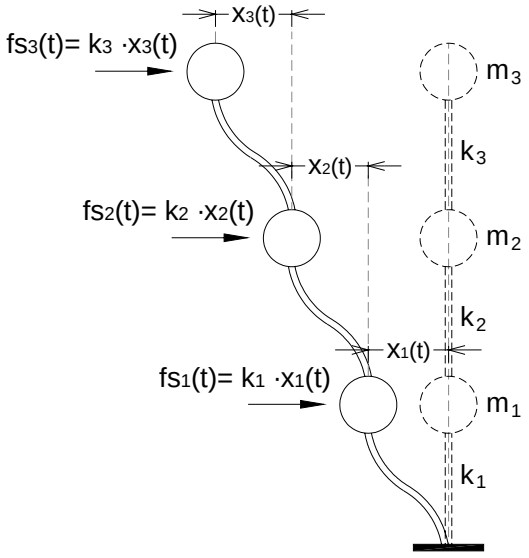


Figura 5.6 Fuerzas sísmicas en el oscilador de VGL

El programa calcula las fuerzas cortantes de cada instante de tiempo *i*, para cada grado de libertad, mediante la función "CalculaVbasal" (código fuente 5.17). Éste muestra dos ciclos *for* para calcular los desplazamientos relativos entre las masas (X_1, X_2 y X_3 mostrados en la figura 5.6) que se multiplican por el valor de la rigidez correspondiente de cada grado de libertad.

El resultado de las operaciones se almacena en la matriz "FuerzasV".

```
    Private Sub CalculaVbasal(ByVal maxpuntos As Integer)
        ReDim FuerzasV(NGL)

        For i = 1 To NGL - 1
            ReDim FuerzasV(i)(maxpuntos)
            For j = 1 To maxpuntos
                FuerzasV(i)(j) = (Dvgl(i)(j) - Dvgl(i + 1)(j)) * Rigidez(i)
            Next
        Next

        ReDim FuerzasV(NGL)(maxpuntos)
        For j = 1 To maxpuntos
            FuerzasV(NGL)(j) = Dvgl(NGL)(j) * Rigidez(NGL)
        Next
    End Sub
```

Código fuente 5.17 Función para calcular la fuerza cortante

5.2. GRÁFICAS

Son diagramas que representan datos numéricos por medio de una serie de puntos unidos mediante líneas para demostrar una conexión entre ellos.

El objetivo de las gráficas dentro de la interfaz gráfica de usuario es mostrar la respuesta dinámica, así como los dibujos de los osciladores para que el usuario vea el movimiento de éste en cada instante de tiempo debido a ejercer una señal excitación al sistema.

Para generar las gráficas en el programa fue necesario usar controles del tipo *PictureBox*, ubicarlos dentro del formulario y posteriormente dibujar sobre éste cada parte necesaria para formar las gráficas.

El programa muestra 3 tipos de gráficas: los dibujos de las señales (respuesta dinámica y excitación del sistema), el dibujo de los osciladores (1GL y VGL) y los dibujos de los espectros respuesta y de piso.

5.2.1. Señales

Para este tipo de gráficas se declaró una matriz de controles de tipo *PictureBox* de la siguiente manera:

```
Dim AreaGráfica(3) As New PictureBox
```

La matriz de controles permite optimizar el código, pues en vez de declarar una variable para cada gráfica podemos agrupar todas en una sola matriz. Se llama matriz de controles porque sus elementos son precisamente controles, en este caso controles del tipo *PictureBox*.

Para ambas ventana (osciladores de 1GL y VGL) se declaró una matriz de controles con el mismo nombre "AreaGráfica"; lo único que cambia entre ellas es el tamaño 3 y 4 elementos respectivamente.

Una vez declarados los *PictureBox*, se establecen las dimensiones mínimas (tabla 5.3) para cada control y posteriormente se ubican dentro del área de trabajo de cada ventana.

Tabla 5.3 Dimensiones mínimas (píxeles) de las gráficas en las ventanas principales

Ventana	Gráfica de carga		Gráfica de respuesta	
	Ancho	Alto	Ancho	Alto
Osciladores 1GL	670	120	670	95
Osciladores VGL	520	120	520	120

El área de trabajo tiene un sistema de coordenadas (x, y) ubicado en la parte superior izquierda de la ventana (figura 5.7); donde la dirección "y" positiva es hacia abajo y la dirección "x" positiva es hacia la derecha.

Para ubicar los controles dentro del formulario sólo se necesita especificar un solo punto (superior izquierdo) de cada gráfica. La ubicación del *PictureBox* para la gráfica de carga, en ambas ventanas, se encuentra en la coordenada: *x=10, y=10* (figura 5.7).

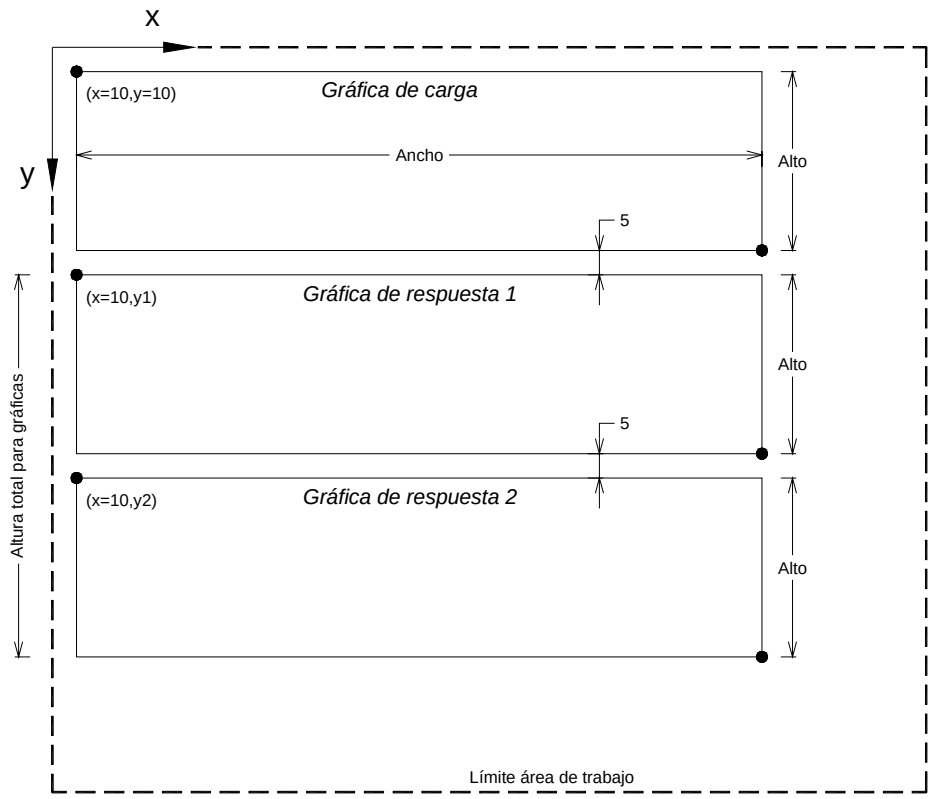


Figura 5.7 Ubicación de los *PictureBox* en el área de trabajo

Los *PictureBox* para la respuesta dinámica se ubican debajo de la gráfica de carga y tienen una separación entre ellos de 5 píxeles, su altura mínima se especifica en la tabla 5.7. Cuando el usuario cambia el número de gráficas que desea ver en pantalla, la altura de las gráficas se modifica ajustándose de manera que se distribuyen en todo lo alto de la "altura total para las gráficas" (figura 5.7) manteniendo la misma separación.

Las gráficas contenidas en el programa están formadas por los siguientes elementos:

- ✓ Escala numérica
- ✓ Fondo cuadriculado
- ✓ Pestaña
- ✓ Dibujo de la señal

Cada *PictureBox* se dividió, como se muestra en la figura 5.8, para albergar a cada elemento de la gráfica. La parte 1 se destinó a la escala numérica; la parte 2 corresponde (en su caso) a la pestaña de la gráfica de carga y por último la parte 3 contiene al dibujo de la señal correspondiente en un fondo cuadriculado.

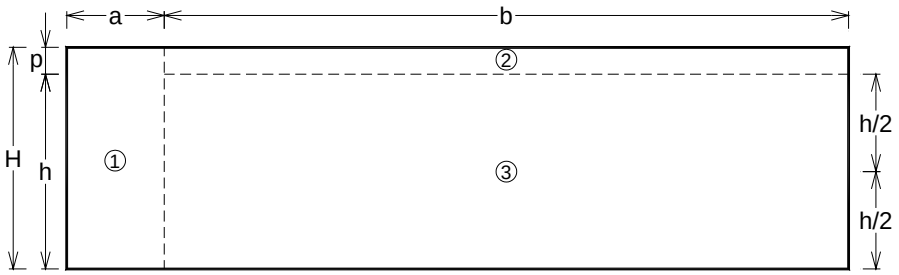


Figura 5.8 División de los *PictureBox* para las gráficas

Para dibujar cada parte de las gráficas dentro del *PictureBox*, se creó la clase *GráficarSeñal* (código fuente 5.18). Esta clase contiene 7 constructores definidos con la palabra clave *Sub New* y una función del tipo *Private*. Los constructores 1, 2, 3 y 7 dibujan la gráfica en forma estática y los constructores 4, 5 y 6 son para dibujar las señales cuando se encuentren en animación. Estos últimos se explican en la sección 5.3.

Los cuatro constructores mencionados (1, 2, 3 y 7) son similares entre si. Cada uno dibuja la gráfica con diferentes elementos; por ejemplo, el constructor 2 dibuja la gráfica con todos sus elementos, mientras que el constructor 1 omite el dibujo de la señal.

Cada constructor necesita de parámetros para generar la gráfica. La tabla 5.4 muestra el significado de cada parámetro que se ocupan en los constructores.

Tabla 5.4 Parámetros en los constructores

Variable	Descripción
pic	Nombre del *PictureBox* donde se va a dibujar la gráfica
texto	Título que se le va a colocar a la gráfica
Sx	Separación (píxeles) de la cuadrícula en dirección X
Sy	Separación (píxeles) de la cuadrícula en dirección Y
DatosSeñal() Puntos()	Vector de coordenadas (x,y) de la señal a dibujar
FE / FEV	Factor de escala vertical
Tiempo	Etiqueta para mostrar la duración de la señal

ColorLapiz	Color de la línea que representa a la señal
Grosor	Espesor (píxeles) de la línea que representa a la señal
Btmp	*Bitmap* que va unido a cada gráfica
X1, Y1	Coordenadas (x,y) del punto inicial para dibujar una línea
X2, Y2	Coordenadas (x,y) del punto final para dibujar una línea

```
Public Class GráficarSeñal

'Constructor 1 Dibuja la gráfica con el fondo, sin la señal y con la pestaña
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Single, ByVal Sy As
Single)

'Constructor 2 Dibuja la gráfica con el fondo, la señal y la pestaña
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Single, ByVal Sy As
Single, ByVal DatosSeñal() As PointF, ByVal FEV As Double, ByVal Tiempo As String)

'Constructor 3 Dibuja la gráfica con el fondo, la señal y sin la pestaña
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Single, ByVal Sy As
Single, ByVal FE As Double, ByVal puntos() As PointF, ByVal ColorLapiz As Color, ByVal
Grueso As Integer, ByRef Btmp As Bitmap)

'Constructor 4 Dibuja la gráfica unicamente con la señal, sin fondo, sin pestaña, para la
animación en la parte de los espectros
Sub New(ByVal pic As PictureBox, ByVal puntos() As PointF, ByVal ColorLapiz As Color, ByVal
Grueso As Integer)

'Constructor 5 Dibuja la gráfica con el fondo, sin la señal, sin la pestaña antes de cada
animación de la señal
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Single, ByVal Sy As
Single, ByVal FE As Double, ByRef Btmp As Bitmap)

'Constructor 6 Dibuja la gráfica con la señal por pasos, sin la pestaña y sin el fondo
Sub New(ByVal pic As PictureBox, ByRef Btmp As Bitmap, ByVal X1 As Single, ByVal Y1 As
Single, ByVal X2 As Single, ByVal Y2 As Single, ByVal ColorLapiz As Color, ByVal Grueso As
Integer)

'Constructor 7 Dibuja la gráfica con el fondo, sin la señal y sin la pestaña
 Sub New(ByVal pic As PictureBox, ByVal Sx As Single, ByVal Sy As Single)

'Función del tipo Privada para dibujar el fondo cuadriculado en las gráficas
Private Sub dibujafondo(ByVal Pic As PictureBox, ByVal mapa As Graphics, ByVal Sx As Single,
ByVal Sy As Single, ByVal CoordY As Integer)

End Class
```

Código fuente 5.18 Clase *GráficarSeñal*

Para que el dibujo de la señal se vea adecuadamente dentro de los *PictureBox*, se definieron factores de escala vertical y horizontal, que se calculan de la siguiente manera:

$$Fev = \frac{h/2}{A\max} = \frac{h}{2A\max}$$

$$Feh = \frac{b}{Duración}$$

Donde:

Fev = Factor de escala vertical
Feh = Factor de escala horizontal
*A*max = Amplitud máxima de la señal en valor absoluto

Los factores de escala multiplican a cada uno de los valores de la señal (x, y) para que se ajusten al tamaño que les corresponde en los *PictureBox*. Para calcularlos es necesario tener primero los datos de cada señal que se va a dibujar.

El código fuente 5.19 muestra el contenido del constructor 2, que dibuja la gráfica con todos sus elementos. El código se encuentra definido dentro de las palabras clave *Sub new* y *End Sub*.

Para dibujar sobre un *PictureBox*, es necesario primero crear un área de dibujo relacionada a él; esto se hace con la ayuda de un mapa de bits (*Bitmap*) y se conectan entre si como se muestra en las primeras 4 líneas dentro del constructor en el código 5.19.

Después, se calculan las variables *Posicionmedia, MitadAltura y Numlineas* que sirven para conocer la distancia vertical donde se va a dibujar la línea de referencia de color rojo que se muestra en las gráficas y a partir de ésta dibujar la cuadrícula de fondo y el dibujo de la señal.

La siguiente línea en el código invoca a la función *dibujafondo* (código 5.20) que es la encargada de dibujar el fondo cuadriculado de color gris en la gráfica; la cual al ser del tipo *Private* sólo puede ser usada dentro de la clase *GráficarSeñal*.

Posteriormente se tienen las instrucciones (*mapa.Drawlines*) para dibujar por completo la señal correspondiente a la gráfica y la línea horizontal de referencia color rojo.

```
'Constructor 2 Dibuja la gráfica con el fondo, la señal y la pestaña
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Single, ByVal Sy As
Single, ByVal DatosSeñal() As PointF, ByVal FEV As Double, ByVal Tiempo As String)

Dim btmp1 As New Bitmap(pic.Width, pic.Height) 'Crea un Bitmap del tamaño del pictureBox
Dim mapa As Graphics = pic.CreateGraphics()    'Crea un área de trabajo a partir del Pb
pic.Image = btmp1                              'La imagen del PictureBox será el Bitmap
mapa = Graphics.FromImage(btmp1)               'Permite un buen efecto de animación
                                               'en cada cambio de dibujo

Posicionmedia = pic.Height / 2 + 10   'Valor de la coordenada en Y de la línea de referencia
MitadAltura = (pic.Height - 20) / 2   'Valor de la mitad de la altura libre para dibujar
NumLineas = CInt(MitadAltura \ Sy)    'Número de líneas a dibujar en la cuadrícula

Call dibujafondo(pic, mapa, Sx, Sy, 20) 'Dibuja el fondo con los valores calculados

mapa.DrawLines(lapiz1, DatosSeñal)      'Dibuja la señal
mapa.DrawLine(lapiz2, CoordX, Posicionmedia, pic.Width, Posicionmedia) 'línea de y= 0

Dim objetoPESTAÑA As New Pestaña(mapa, pic, CoordX) 'Dibuja la pestaña sobre el PictureBox

Dim objetoTEXTO2 As New TextoHorizontal(NumLineas, CInt(Posicionmedia), Sy, FEV, mapa, "t =
" & Tiempo, pic.Width - 70)            'Dibuja la escala númerica

mapa.DrawString(texto, fuenteN, Brushes.Black, CoordX + 5, 5)'Dibuja el título de la gráfica
End Sub
```

Código fuente 5.19 Constructor 2 de la clase *GráficarSeñal*

Para colocar la pestaña y la escala numérica en la gráfica se crearon dos clases auxiliares: la clase *Pestaña* (código fuente 5.21) y la clase *Textohorizontal* (código fuente 5.22) respectivamente. En el código fuente 5.19 se muestra cómo se utilizaron los objetos (*objetoPestaña y objetoTEXTO2*) para usar las clases *Pestaña y Textohorizontal*, respectivamente.

Por último, en el constructor 2 de la clase *GráficarSeñal* se indica mediante la instrucción *mapa.DrawingString* que se coloque el título de la gráfica.

```
Private Sub dibujafondo(ByVal Pic As PictureBox, ByVal mapa As Graphics, ByVal Sx As Single,
ByVal Sy As Single, ByVal CoordY As Integer)
        Dim respuesta As Boolean

        With mapa
            .FillRectangle(Brushes.Transparent, 0, 0, Pic.Width, Pic.Height)
            .FillRectangle(Brushes.White, 53, 0, Pic.Width - 53, Pic.Height)

            .FillRectangle(relleno, CoordX, CoordY, Pic.Width - CoordX, Pic.Height - CoordY)
            .DrawRectangle(Pens.Black, CoordX, CoordY, Pic.Width - (CoordX + 1), Pic.Height
- (CoordY + 1))

            'malla vertical
            For Me.i = 1 To Pic.Width
                .DrawLine(lapiz, CoordX + i * Sx, CoordY, CoordX + i * Sx, Pic.Height)
                If (CoordX + (i + 1) * Sx) > Pic.Width Then
                    Exit For
                End If
            Next

            'malla horizontal parte baja
            For Me.i = NumLineas To 1 Step -1
                .DrawLine(lapiz, CoordX, Posicionmedia + i * Sy, Pic.Width, Posicionmedia +
i * Sy)
            Next

            'malla horizontal parte alta
            For Me.i = NumLineas To 1 Step -1
                .DrawLine(lapiz, CoordX, Posicionmedia - i * Sy, Pic.Width, Posicionmedia -
i * Sy)
            Next
  End Sub
```

Código fuente 5.20 Función *dibujaFondo*

La clase *Pestaña* (código fuente 5.21) tiene un solo constructor y no cuenta con más funciones de ningún tipo, por lo tanto todas sus instrucciones se encuentra dentro de él. Primero, se definen cuatro variables (*relleno, relleno2, lápiz y lapiz2*) para almacenar los colores de relleno y el color de la línea de contorno.

Después, se definen dos vectores (*puntos y puntos2*) para contener las coordenadas de la poligonal de la pestaña; ésta se dibujará a todo lo ancho del *PictureBox* con una altura definida de 10 píxeles. Por último, las instrucciones (*FillPolygon* y *DrawPolygon*) indican que se dibuje el polígono con un color de relleno y su contorno de otro color.

La clase *TextoHorizontal* (código fuente 5.22) también cuenta con un solo constructor. Primero, se especifica la distancia vertical de la línea que marca la amplitud cero de cada gráfica; en seguida establece el tipo de letra, formato y color que tendrá la escala numérica. Después con la ayuda de un bucle *for* calcula y coloca los textos de la escala numérica correspondientes a la cuadrícula de la gráfica creada anteriormente. Por último, coloca un texto que indica la duración de la señal dentro de la pestaña en la parte superior derecha.

```
Public Class Pestaña
 Sub New(ByVal mapa As Graphics, ByVal pic As PictureBox, ByVal X1 As Integer)

  Dim relleno As New SolidBrush(Color.FromArgb(50, Color.Yellow)) 'Color de relleno
  Dim lapiz As New Pen(Color.FromArgb(250, Color.DarkRed), 1)     'Color de línea
  Dim relleno2 As New SolidBrush((SystemColors.Control))          'Color de relleno
  Dim lapiz2 As New Pen(SystemColors.Control, 1)                  'Color de línea

  Dim puntos() As Point = {New Point(X1, 10), New Point(X1 + 10, 0), New Point(pic.Width -
1, 0), New Point(pic.Width - 1, 20), New Point(X1, 20)}  'Coordenadas del poligono

  'Coordenadas del triangulo en la esquina superior izqueirda
  Dim puntos2() As Point = {New Point(X1, 0), New Point(X1, 10), New Point(X1 + 10, 0)}

  mapa.FillPolygon(relleno2, puntos2)  'Dibujo del poligono relleno
  mapa.DrawPolygon(lapiz2, puntos)     'Dibujo del contorno del poligono
  mapa.FillPolygon(relleno, puntos)    'Dibujo del triangulo relleno
  mapa.DrawPolygon(lapiz, puntos)      'Dibujo del contorno del triangulo

  End Sub
End Class
```

Código fuente 5.21 Clase *Pestaña*

```
Public Class TextoHorizontal
    Dim Yo As Integer

    Sub New(ByVal Numlineas As Integer, ByVal Yo As Integer, ByVal Sy As Single, ByVal
FactEscalaV As Double, ByVal mapa As Graphics, ByVal Titulo As String, ByVal PosX As
Integer)

        Me.Yo = Yo – 7     'Altura inicial para comenzar a colocar los textos

        Dim texto As String                     'Variable que recibe la etiqueta
        Dim fuente As New Font("Arial", 8)                  'Tipo de letra
        Dim fuenteN As New Font("Arial", 8, FontStyle.Bold) 'Tipo de letra en Negrita
        Dim brocha As New SolidBrush(Color.Black)           'Color del texto
        Dim x As Single = CoordX – 3                        'Posición en X
        Dim formato As New StringFormat                     'Variable para formato

        'Se define formato de escritura de derecha a izquierda
        formato.FormatFlags = StringFormatFlags.DirectionRightToLeft

        'Se escribe el texto “0.0” en la mitad de la gráfica
        mapa.DrawString(texto, fuenteN, brocha, x, Me.Yo, formato)

        For i = 1 To Numlineas                   'Ciclo para colocar los demás textos
            If FactEscalaV = 0 Then              'En caso de que el Factor de escala sea 0
                texto = "" & Format(0, "0.00E+00") 'Coloca texto “0”
            Else                                 'En caso contrario
                texto = "" & Format(i * Sy / FactEscalaV, "0.00E+00")'Calcula las escalas
            End If

           'Coloca los textos en la escala númerica
            mapa.DrawString(texto, fuente, brocha, x, Me.Yo - i * Sy, formato) 'arriba
            mapa.DrawString(texto, fuente, brocha, x, Me.Yo + i * Sy, formato) 'abajo
        Next i

        mapa.DrawString(Titulo, fuenteN, Brushes.Black, PosX + 5, 5) 'Duración de la señal

    End Sub
```

Código fuente 5.22 Clase *TextoHorizontal*

5.2.2. Osciladores

El procedimiento para dibujar los osciladores es similar al descrito anteriormente para las gráficas de las señales. Cada oscilador fue dibujado en con la ayuda de dos *PictureBox* superpuestos uno encima del otro, es decir, un dibujo por capas. En un *PictureBox* se dibujó el fondo del oscilador mientras que en el otro se dibujó el sistema de masas y resortes. Esta manera de dibujar fue muy útil en el proceso de animación del oscilador, que se explica en la sección 5.3.2.

5.2.2.1. Osciladores de 1GL

Para la ventana de osciladores de 1GL se declaró un par de matrices de controles del tipo *PictureBox* con el nombre "AreaOscilador" y "CapaOscilador"; la primera para el dibujo del fondo y la segunda para el oscilador (masa y resorte). La dimensión de ambas matrices es de 3 elementos, ya que en esta ventana se muestran hasta 3 osciladores al mismo tiempo.

```
Dim AreaOscilador(3) As New PictureBox
Dim CapaOscilador(3) As New PictureBox
```

Las dimensiones de los osciladores son fijas: 200 x 200 píxeles; éstas no se modifican aun cuando el usuario seleccione la opción de ajustar las gráficas al tamaño de la pantalla.

La ubicación de los osciladores es diferente para cada ventana. En el caso de los osciladores de 1GL los *PictureBox* se encuentran en la parte baja de la ventana, 20 píxeles debajo de las gráficas de respuesta. La separación lateral máxima entre ellos es de 40 píxeles y la mínima de 8 píxeles. La figura 5.9 muestra su ubicación dentro de la ventana.

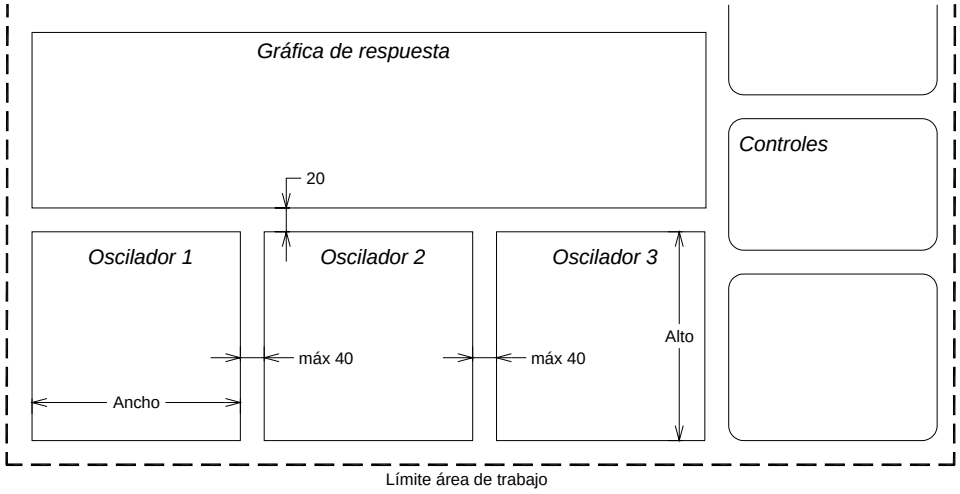


Figura 5.9 Ubicación de los osciladores de 1GL en la ventana

Cada gráfica de los osciladores consta de 3 elementos:

- ✓ Fondo cuadriculado
- ✓ Pestaña
- ✓ Dibujo del oscilador según corresponda

De igual forma que en las gráficas de las señales, los *PictureBox* (variables "AreaOscilador") se dividieron para dibujar por partes las gráficas de los osciladores (figura 5.10). La parte 1 corresponde,

para todos los osciladores, a la pestaña de la gráfica y la parte 2 contiene al dibujo del fondo cuadriculado, similar al que se encuentra en las gráficas de las señales.

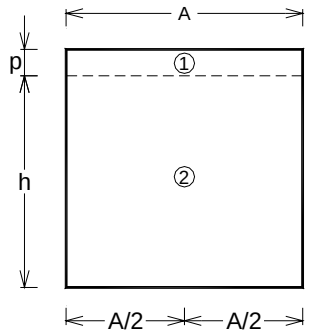


Figura 5.10 División de los *PictureBox* para las osciladores

Para dibujar los elementos de las gráficas fue necesario crear la clase “fondoOscilador” (código fuente 5.23).

Esta clase tiene 3 constructores y 6 funciones del tipo *Private*. El constructor 1 dibuja en el *PictureBox* dos líneas verticales que representan hasta dónde llegará el desplazamiento máximo de la masa. El constructor 2 dibuja sólo la masa y el resorte dependiendo del valor de la posición vertical que se indique y por último el constructor 3 dibuja el fondo cuadriculado de la gráfica; éste es muy parecido al mostrado en el código fuente 5.20.

```
Public Class FondoOscilador

'Constructor 1  Dibuja las líneas de escala horizontal del Oscilador
Sub New(ByVal pic As PictureBox, ByVal Sx As Integer, ByVal Sy As Integer, ByVal tamaño As
Integer, ByVal ColorCentro As Color, ByVal ColorExt As Color, ByVal ValorX As Integer)

'constructor 2 Dibuja SOLO los osciladores
Sub New(ByVal pic As PictureBox, ByVal tamaño As Integer, ByVal ColorCentro As Color, ByVal
ColorExt As Color, ByVal Posx As Double, ByVal N As Integer, ByVal despl As Double)

'constructor 3 Dibuja SOLO el fondo
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Integer, ByVal Sy As
Integer)


Private Function CreaAreaDibujo(ByVal Pic As PictureBox) As Graphics

Private Sub dibujafondo(ByVal mapa As Graphics)

Private Sub dibujaPestaña(ByVal mapa As Graphics, ByVal Texto As String)

Private Sub dibujamasa(ByVal mapa As Graphics, ByVal X As Double, ByVal Tamaño As Integer)

Private Sub dibujaColumna(ByVal mapa As Graphics, ByVal X As Double, ByVal Diam As Integer)

Private Sub dibujaLineasEscala(ByVal mapa As Graphics, ByVal tamaño As Integer)

End Class
```

Código fuente 5.23 Clase *fondoOscilador*

La tabla 5.5 muestra el significado de cada parámetro que se requieren introducir en cada constructor de la clase *fondoOscilador*.

Tabla 5.5 Parámetros en los constructores

Variable	Descripción
pic	Nombre del *PictureBox* donde se va a dibujar la gráfica
Sx	Separación (píxeles) de la cuadrícula en dirección X
Sy	Separación (píxeles) de la cuadrícula en dirección Y
Tamaño	diámetro de la masa en píxeles
ColorCentro	Color del centro de la masa
ColorExt	Color del relleno de la masa
ValorX	Distancia (píxeles) horizontal de las líneas verticales al centro del *PictureBox*
PosX	Posición (píxeles) de la masa dentro del *PictureBox*
N	Número de punto de la señal de desplazamiento
despl	Valor del desplazamiento de la masa que corresponde al punto N
Texto	Título de la gráfica

El código fuente 5.24 contiene al constructor 2, que dibuja solamente la masa y resorte de cada oscilador. El código se encuentra definido dentro de las palabras clave *Sub new* y *End Sub*.

En las primeras líneas del código, el constructor pasa los valores recibidos a las variables globales, después llama a la función “CrearAreaDibujo” (código fuente 5.25) para establecer un espacio sobre el cual se dibujan cada elemento de la gráfica y lo almacena en la variable “mapa”. Posteriormente se indica que se coloque un texto con el valor del deplazamiento de la masa en un instante N, que corresponde a un punto de la señal. Por último, el constructor llama a dos funciones “dibujaColumna” y “dibujamasa” (código fuente 5.26 y 5.27) para dibujar la masa y columna respectivamente.

```
'constructor 2 Dibuja SOLO los osciladores
Sub New(ByVal pic As PictureBox, ByVal tamaño As Integer, ByVal ColorCentro As Color, ByVal
ColorExt As Color, ByVal Posx As Double, ByVal N As Integer, ByVal despl As Double)
        Me.pic = pic                    'Pasa el picturebox a la variable global de la clase
        Me.ColorCentro = ColorCentro                'Color del centro de la masa
        Me.ColorExt = ColorExt                      'Color del exterior de la masa
        Me.Posx = (pic.Width / 2) + Posx            'Posición en X del centro de la masa

        mapa = CreaAreaDibujo(pic)      'crea el área de dibujo a partir del PictureBox

        'Escribe el instante y el valor del desplazamiento en ese instante
        mapa.DrawString("D[" & N & "]= " & Format(despl, "0.0E+00"), fuenteN,
Brushes.DarkRed, pic.Width - 5, 4, formato)

        Call dibujaColumna(mapa, Me.Posx, tamaño)     'Dibuja la columna
        Call dibujamasa(mapa, Me.Posx, tamaño)        'Dibuja la masa
End Sub
```

Código fuente 5.24 Constructor 2 de la clase *fondoOscilador*

El cuerpo de la función para establecer el área gráfica (código fuente 5.25) es el mismo que se encuentra en las primeras cuatro instrucciones del constructor del código fuente 5.19.

Se define primero un mapa de bits (*Bitmap*) y se conecta con el *PictureBox*, por medio de la propiedad *Image* del control.

```
Private Function CreaAreaDibujo(ByVal Pic As PictureBox) As Graphics
   Dim Btmp1 As New Bitmap(Pic.Width, Pic.Height) 'Crea un Bitmap del tamaño del pictureBox
   Dim mapa As Graphics = Pic.CreateGraphics()    'Crea un área de trabajo a partir del
                                                   PictureBox
   Pic.Image = Btmp1                              'La imagen del PictureBox será el Bitmap
   mapa = Graphics.FromImage(Btmp1)               'Permite un buen efecto de animación en
                                                   cada cambio de dibujo
   Return mapa                                    'Regresa la variable mapa
End Function
```

Código fuente 5.25 Función para asignar el área gráfica

El código fuente 5.26 se encarga de dibujar únicamente la masa de cada oscilador; para ello se requiere establecer el diámetro de la masa en píxeles y su posición dentro del *PictureBox*. Cuando los osciladores permanecen estáticos el valor "X" de la posición es la mitad del ancho del *PictureBox*.

La función define un círculo estableciendo su ubicación y degradación de colores (del centro hacia el exterior) y agregando un contorno de color negro del mismo tamaño que el diámetro de la masa.

```
Private Sub dibujamasa(ByVal mapa As Graphics, ByVal X As Double, ByVal Tamaño As Integer)

        Dim trayecto As GraphicsPath = New GraphicsPath  'variable para dibujar el degradado
        Dim rectangulo As New RectangleF(X - (Tamaño / 2), 50, Tamaño, Tamaño)
        trayecto.AddEllipse(rectangulo)                  'Estable el límite del circulo

        Dim brocha As PathGradientBrush = New PathGradientBrush(trayecto) 'Inica gradiente

        brocha.CenterPoint = New PointF(X, (50 + Tamaño / 2)) 'Ubicación del centro de la m
        brocha.CenterColor = ColorCentro                      'Establece el color del centro
        brocha.SurroundColors = New Color() {ColorExt}        'Establece el color exterior

        mapa.FillPath(brocha, trayecto)          'Dibuja el relleno con degradación de color
        mapa.DrawEllipse(lapiz1, rectangulo)     'Dibuja el contorno de la masa
End Sub
```

Código fuente 5.26 Función para dibujar la masa

La columna se dibujó como una línea curva con tres puntos de control: inicio, centro y final. En la figura 5.11a se muestra la ubicación de estos tres puntos (p1, p2 y p3 respectivamente). La altura máxima de la columna es de 95 píxeles y depende del diámetro de la masa que especifique el usuario.

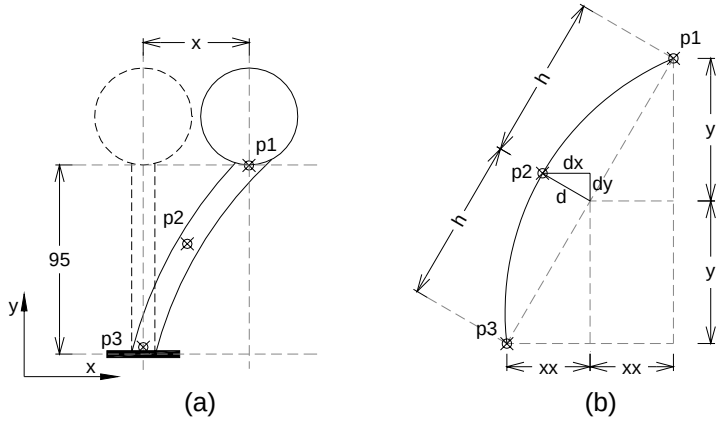


Figura 5.11 Puntos de control para dibujar la columna

La función que dibuja la columna (código fuente 5.27) establece primero la posición de los puntos 1 y 3 que se encuentran en la parte baja de la masa (p1) y la base de la columna (p3).

Después, para determinar la posición del punto intermedio de la curva (p2), se calculan las distancias "dx" y "dy" con la relación de triángulos semejantes a partir de las distancias "xx", "y" y "h" (mostradas en la figura 5.11b); la distancia "d" (perpendicular a la línea recta entre los puntos p1 y p3) se toma como el 20% de la distancia "xx".

Las coordenadas de los tres puntos cambian conforme el desplazamiento del oscilador va cambiando también; esto sucede cuando la animación del oscilador está en ejecución. Mientras el oscilador permanezca en posición estática (desplazamiento=0) la columna se dibujará de manera vertical, pues la coordenada "x" de los tres puntos es la misma.

La instrucción para dibujar la columna es "mapa.DrawCurve" la cual recibe como parámetros el formato de la línea (espesor de 8 píxeles y color igual al color exterior de la masa) y las coordenadas de los puntos mencionados.

```
Private Sub dibujaColumna(ByVal mapa As Graphics, ByVal X As Double, ByVal Diam As Integer)
        Dim LapizColumna As New System.Drawing.Pen(ColorExt, 8)
        Dim h, dx, dy, xx, y, d As Single

        Dim punto1 As New PointF(X, 45 + Diam)                   'Parte baja de la masa
        Dim punto3 As New PointF(pic.Width / 2, pic.Height - 10) 'Base de la columna

        xx = (punto1.X - punto3.X) / 2
        y = (punto3.Y - punto1.Y) / 2
        h = Sqrt(xx ^ 2 + y ^ 2)
        d = 0.2 * xx

        dx = d * y / h
        dy = d * xx / h

        Dim punto2 As New PointF(punto3.X + xx - dx, punto3.Y - y - dy) 'Punto de control 2
        Dim puntoscurva As PointF() = {punto1, punto2, punto3}          'Puntos de la curva
        mapa.DrawCurve(LapizColumna, puntoscurva)                        Dibuja la columna
End Sub
```

Código fuente 5.27 Función para dibujar la columna

El constructor 3 de la clase "fondoOscilador" (código fuente 5.28) dibuja sobre los *PictureBox* el fondo de las gráficas de los osciladores y la pestaña con el título de cada una.

```
'constructor 3 Dibuja SOLO el fondo
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Integer, ByVal Sy As
Integer)
        Me.Sx = Sx                      'Valor de la separación en X de la malla
        Me.Sy = Sy                      'Valor de la separación en Y de la malla
        Me.pic = pic                    'Pasa el picturebox a la Variable global de la clase

        mapa = CreaAreaDibujo(pic)      'crea el área de dibujo a partir del PictureBox

        MitadAncho = (pic.Width \ 2)          'Mitad de ancho del picturebox
        NumLineas = CInt(MitadAncho \ Sx)     'Numero de líneas verticales
        Call dibujafondo(mapa)                'Dibuja el fondo de los osciladores
        Call dibujaPestaña(mapa, texto)       'Dibuja la pestaña con el nombre del oscilador
End Sub
```

Código fuente 5.28 Constructor 3 de la clase *fondoOscilador*

En el constructor 3, al igual que en el código fuente 5.24, pasan los valores de los parámetros recibidos a las variables globales; después, se llama a la función “CrearAreaDibujo” (código fuente 5.25). Al final se invocan las funciones “dibujafondo” y “dibujaPestaña” (código fuente 5.29) para dibujar los elementos mencionados.

El código fuente de la función “dibujafondo” es similar al código fuente 5.20 de la gráfica de señales; éste se encarga de cuadricular el fondo de las gráficas tomando como parámetros el valor de las variables globales “Sx” y “Sy”. En este caso los valores son constantes de 12 y 20 píxeles respectivamente, y no hay manera de que el usuario manipule estos valores.

Para dibujar la pestaña en las gráficas de los osciladores, se creó la función “dibujaPestaña” (código fuente 5.29), en la cual se define el objeto “objetoPESTAÑA” que hace referencia a la clase “Pestaña” (código fuente 5.21); esta clase se encarga de dibujar sobre el *PictureBox* la pestaña en lo alto del control con un ancho de 10 píxeles y establece el título del oscilador.

```
    Private Sub dibujaPestaña(ByVal mapa As Graphics, ByVal Texto As String)
        Dim objetoPESTAÑA As New Pestaña(mapa, pic, 0)
        mapa.DrawString(Texto, fuenteN, Brushes.Black, 5, 5)
    End Sub
```

Código fuente 5.29 Función para dibujar la pestaña

5.2.2.2. Oscilador VGL

En la ventana de osciladores de VGL se declaró un par de variables (“AreaOscilador” y “CapaOscilador”), sin necesidad de ser matrices, pues en esta ventana sólo hay un oscilador.

```
Dim AreaOscilador As New PictureBox
Dim CapaOscilador As New PictureBox
```

Las dimensiones mínimas para la gráfica del oscilador varían dependiendo del número de grados de libertad; estas se muestran en la tabla 5.6.

Tabla 5.6 Dimensiones mínimas (píxeles) del oscilador de VGL

Grados de libertad	Alto	Ancho
2	230	180
3	320	
4	410	
5	500	
6 ≤	590	

El *PictureBox* se ubica entre las gráficas de respuesta y el panel de controles. El oscilador se halla siempre centrado en lo alto de la ventana como se muestra en la figura 5.12, y dependerá de su altura mostrada en la tabla 5.6.

El *PictureBox* fue dividido en partes de la misma forma que los osciladores de 1GL descritos anteriormente. De igual manera se elaboró una clase “fondoOsciladorVGL” para dibujar la gráfica del oscilador.

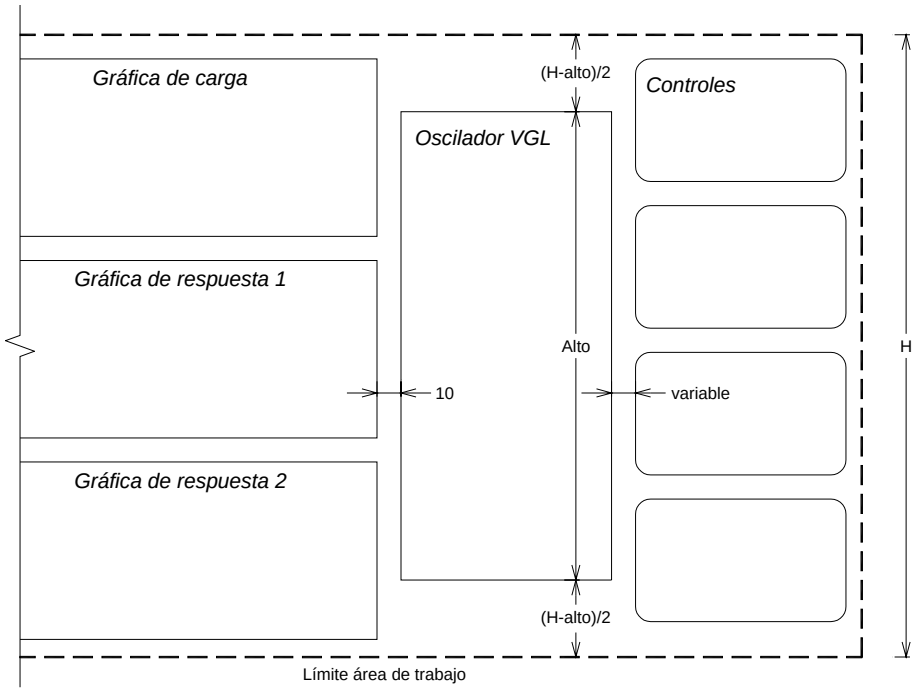


Figura 5.12 Ubicación del oscilador de VGL en la ventana

La clase "fondoOsciladorVGL" (código fuente 5.30) tiene 2 constructores y 4 funciones del tipo *Private*. Los constructores son similares a los de la clase "fondoOscilador": uno dibuja el fondo cuadriculado del oscilador (constructor 1) y el otro (constructor 2) dibuja sistema de masas y resortes para N grados de libertad que se hayan definido.

La tabla 5.7 muestra el significado de cada uno de los parámetros que requieren los constructores de la clase.

```
Imports System.Drawing.Drawing2D
Imports System.Math

Public Class FondoOsciladorVGL

'Constructor 1 Dibuja EL FONDO del oscilador
Sub New(ByVal pic As PictureBox, ByVal Sx As Integer, ByVal Sy As Integer, ByVal nGL As
Integer)

'constructor 2 Dibuja SOLO los osciladores (masas y resortes) en cada instante
Sub New(ByVal pic As PictureBox, ByVal Texto As String, ByVal tamaño As Single, ByVal
Puntos() As Double, ByVal Valores() As Double, ByVal nGL As Integer, ByVal MostrarVals As
Boolean, ByVal N As Integer)

Private Sub dibujafondo(ByVal mapa As Graphics)

Private Sub dibujaPestaña(ByVal mapa As Graphics, ByVal n As Integer)

Private Sub dibujamasa(ByVal mapa As Graphics, ByVal X As Double, ByVal Y As Double)

Private Sub dibujaColumna(ByVal mapa As Graphics, ByVal X1 As Double, ByVal Y1 As Double,
ByVal X2 As Double, ByVal Y2 As Double)

End Class
```

Código fuente 5.30 Clase *fondoOsciladorVGL*

En el constructor 2 (código fuente 5.31), se asignan los valores de los parámetros recibidos a las variables globales de la clase y declara una matriz “PosXx ( )” para almacenar los valores de las coordenadas de las masas en cada instante de tiempo *i*, así como las variables para almacenar las coordenadas (x, y) de los textos que muestran los valores de los desplazamientos de cada masa

Después se establece un área de dibujo ligado al *PictureBox* a través de un *Bitmap*, como se mencionó anteriormente, esto se encuentra en el código fuente 5.31 a partir de la sexta línea de código.

```
'constructor 2 Dibuja SOLO los osciladores en cada instante
Sub New(ByVal pic As PictureBox, ByVal Texto As String, ByVal tamaño As Single, ByVal
Puntos() As Double, ByVal Valores() As Double, ByVal nGL As Integer, ByVal MostrarVals As
Boolean, ByVal N As Integer)

Me.pic = pic                              'Pasa el picturebox a la Variable global de la clase
Me.Tamaño = tamaño                        'Tamaño de la masa
Me.nGL = nGL                              'Número de grados de libertad

Dim PosXx() As Double      'Coordenadas en pixeles de las masas dentro del PictureBox
Dim x, y As Single         'Coordenadas del texto que muestra el valor desplazamiento

Dim Btmp As New Bitmap(pic.Width, pic.Height)   'Se crea un Bitmap de las dimensiones del Pb
Dim mapa As Graphics = pic.CreateGraphics()     'Crea un área de trabajo a partir del Pb
pic.Image = Btmp                                'La imagen del PictureBox será el Bitmap
mapa = Graphics.FromImage(Btmp)                 'Permite un buen efecto de animación en cada
                                                 cambio de dibujo

ReDim PosXx(nGL)                   'Valores de los desplazamientos de cada grado de libertad

For Me.i = 0 To (Me.nGL - 1)                     'Recorre todos los espacios del vector
   PosXx(i) = (pic.Width / 2) + Puntos(i + 1)    'Establece la posición a partir de la mitad
Next i                                            de ancho del PictureBox

Dim Py1, Py2, Largo As Double                    'Variables auxiliares
Py1 = 40 + Me.Tamaño * 0.99
Largo = 1.5 * Me.Tamaño + Me.Tamaño * 0.02
Py2 = Py1 + Largo

For Me.i = 0 To (Me.nGL - 1)                     'Ciclo que recorre cada GL
   If i = Me.nGL - 1 Then
      Call dibujaColumna(mapa, PosXx(i), Py1, Me.pic.Width / 2, Me.pic.Height - 10)
   Else
      Call dibujaColumna(mapa, PosXx(i), Py1, PosXx(i + 1), Py2)
      Py1 = Py2 + Me.Tamaño * 0.98
      Py2 = Py1 + Largo
   End If

Call dibujamasa(mapa, PosXx(i), 40 + i * Me.Tamaño * 2.5) 'Dibuja la masa

   If MostrarVals = True Then                            'Muestra los valores de Despl
      y = (32 + Me.Tamaño / 2) + i * Me.Tamaño * 2.5     'Coordenada Y de cada texto
      If Valores(i + 1) <= 0 Then                        'Coordenada X
          x = PosXx(i) + Me.Tamaño / 2 + 5
          mapa.DrawString(Format(Valores(i + 1), "0.000"), fuenteN, Brushes.Black, x, y)
      Else
          x = PosXx(i) - Me.Tamaño / 2 - 5
          mapa.DrawString(Format(Valores(i + 1), "0.000"), fuenteN, Brushes.Black, x, y,
formato)
      End If
   End If
Next i
'Título de la gráfica del oscilador
mapa.DrawString(Texto, fuenteN, Brushes.DarkRed, pic.Width - 10, 4, formato)
End Sub
```

Código fuente 5.31 Constructor 2 de la clase *fondoOsciladorVGL*

El segundo bucle *for* dentro del código fuente 5.31 controla el llamado a las funciones que dibujan las masas y las columnas para cada grado de libertad del oscilador. Además, el segundo *if* evalúa si se colocan los letreros de texto que muestran los valores de desplazamientos en cada masa del oscilador.

Los parámetros que se necesitan para cada constructor de la clase “fondoOsciladorVGL” se encuentran en la tabla 5.7.

Tabla 5.7 Parámetros en los constructores

Variable	Descripción
pic	Nombre del *PictureBox* donde se va a dibujar la gráfica
Sx	Separación (píxeles) de la cuadrícula en dirección X
Sy	Separación (píxeles) de la cuadrícula en dirección Y
nGL	Número de grados de libertad
Texto	Título de la gráfica
Tamaño	diámetro de las masas en píxeles
Puntos()	Posición (píxeles) de las masas dentro del *PictureBox*
Valores()	Valor del desplazamiento horizontal de cada masa
MostrarVals	Condicional (falso o verdadero) para mostrar los valores de los desplazamientos
N	Número de punto de la señal de desplazamiento

Las funciones “dibujafondo”, “dibujaPestaña” y “dibujamasa”son las mismas que se utilizaron para el oscilador de 1GL. La función para dibujar la columna es diferente para este tipo de oscilador ya que estas columnas se deforman con una doble curvatura, como se muestra en la figura 5.13.

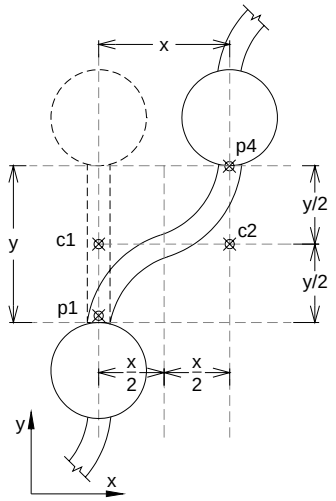


Figura 5.13 Puntos de control para dibujar la columna

El código fuente 5.32 muestra la función “dibujaColumna”; ésta declara primero una variable (grueso) para controlar el espesor de la columna dependiendo del número de GL seleccionado.

Después, se calculan cuatro puntos de control para la columna (p1, c1, c2 y p4 mostrados en la figura 5.13). Para dibujar la columna con doble curvatura se utilizó la función *DrawBezier* la cual recibe como parámetros el formato de la línea y los puntos de control mencionados.

La función *DrawBezier* dibuja una línea curva tomando el primer y último punto (p1 y p4) como el inicio y final de la curva respectivamente. Los dos puntos intermedios (c1 y c2) establecen el radio de giro para dibujar la curvatura de la columna.

Como el llamado de la función "dibujaColumna" en el código fuente 5.31, que se encuentra dentro del segundo bucle *for*, se ejecuta tantas veces como grados de libertad tenga el oscilador, y para cada uno van cambiando las coordenadas de los puntos de control de la curva.

```
Private Sub dibujaColumna(ByVal mapa As Graphics, ByVal X1 As Double, ByVal Y1 As Double,
ByVal X2 As Double, ByVal Y2 As Double)

Dim grueso As Integer            'Espesor de la columna por default es igual a cero

'Las siguientes condiciones evaluan a los GL, es importante el orden en que están acomodadas
las condiciones pues de esta manera la evaluación se hace por rango de valores

If Me.nGL < 15 Then : grueso = 2 : End If   'Si los GL < 15 el espesor de la columna es de 2
If Me.nGL < 12 Then : grueso = 4 : End If   'Si los GL < 12 el espesor de la columna es de 4
If Me.nGL < 9 Then : grueso = 6 : End If    'Si los GL < 9 el espesor de la columna es de 6

'Se crea el tipo de línea definiendo Color y Grosor
Dim LapizColumna As New System.Drawing.Pen(ColorExt, grueso)

Dim Y As Single = (Y1 - Y2) / 2 + Y2  'Punto medio de la columa
Dim punto1 As New PointF(X2, Y2)      'Base de la columna
Dim Control1 As New PointF(X2, Y)     'Punto de control 1
Dim Control2 As New PointF(X1, Y)     'Punto de control 2
Dim punto4 As New PointF(X1, Y1)      'Punto final de la columna

'Se dibuja la curva Bezier de 4 puntos
mapa.DrawBezier(LapizColumna, punto1, Control1, Control2, punto4)

End Sub
```

Código fuente 5.32 Clase *fondoOsciladorVGL*

5.2.3. Espectros

5.2.3.1. Espectros de respuesta

Los espectros se dibujan sobre los *PictureBox* "AreaOscilador", mismos que se utilizaron para dibujar los osciladores, tomando las mismas dimensiones y ubicación dentro de la ventana de los osciladores de 1GL. Así la declaración de matrices que se usó es la misma que se mostró en la sección 5.2.2.1.

Para dibujar los espectros de repuesta, los *PictureBox* se dividieron de la misma forma que se muestra en la figura 5.10. Además, se estableció la clase "fondoEspectro" (código fuente 5.33) para dibujar cada elemento de la gráfica; la clase contiene 3 constructores y 2 funciones del tipo *Private*.

El constructor 1 dibuja sólo el fondo y la pestaña de la gráfica; el constructor 2 dibuja la gráfica con todos sus elementos y por último el constructor 3 sirve para dibujar la gráfica cuando ésta se encuentra en animación. Las funciones del tipo *Private* se utilizan para dibujar sobre el *PictureBox* la pestaña correspondiente a cada gráfica y el fondo cuadriculado en la misma, este último es muy parecido al fondo de que se utilizó en las gráficas de los osciladores, sólo que ocupan un color rojizo en lugar de un color gris.

```
Imports System.Drawing.Drawing2D
Imports System.Math

Public Class FondoEspectro

'Constructor 1 Dibuja la gráfica vacia
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Integer, ByVal Sy As
Integer, ByRef Btmp As Bitmap)

'Constructor 2 Dibuja la gráfica CON el espectro en forma ESTATICA
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Integer, ByVal Sy As
Integer, ByRef puntos() As PointF)

'Constructor 3 Dibuja la gráfica del espectro POR PASOS
Sub New(ByVal pic As PictureBox, ByRef Btmp As Bitmap, ByVal X1 As Single, ByVal Y1 As
Single, ByVal X2 As Single, ByVal Y2 As Single)

Private Sub dibujafondo(ByVal mapa As Graphics)

Private Sub dibujaPestaña(ByVal mapa As Graphics)

End Class
```

Código fuente 5.33 Clase *fondoEspectro*

El código fuente 5.34 muestra el cuerpo del constructor 2 que dibuja la gráfica de los espectros con todos sus elementos. Primero pasa los valores de los parámetros que recibe a las variables globales, después establece un área de dibujo como ya se ha mencionado en repetidas ocasiones, posteriormente llama a las funciones “dibujafondo” y “dibujaPestaña”. La primera es similar a la que se encuentra en el código fuente 5.20, para dibujar el fondo de las gráficas de señales y de los osciladores. La segunda se muestra en el código fuente 5.29.

Por último, con la sentencia “mapa.DrawLines” dibuja la gráfica correspondiente al espectro de respuesta; ésta sólo requiere de dos parámetros, el formato de la línea (“lapiz3”) y el vector con las coordenadas de cada punto que conforman al espectro (“puntos”).

```
'Constructor 2 Dibuja la gráfica CON el espectro en forma ESTATICA
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Integer, ByVal Sy As
Integer, ByRef puntos() As PointF)

Me.Sx = Sx                          'Separación en dirección X de la malla
Me.Sy = Sy                          'Separación en dirección Y de la malla
Me.pic = pic                        'PictureBox sobre el que se dibuja
Me.Texto = texto                    'Nombre de la gráfica

Dim Btmp1 As New Bitmap(pic.Width, pic.Height) 'Crea un Bitmap del tamaño del pictureBox
Dim mapa As Graphics = pic.CreateGraphics()    'Crea un área de trabajo a partir del
                                                PictureBox
pic.Image = Btmp1                              'La imagen del PictureBox será el Bitmap
mapa = Graphics.FromImage(Btmp1)               'Permite un buen efecto de animación en cada
                                                cambio de dibujo

Call dibujafondo(mapa)                         'Dibuja fondo
Call dibujaPestaña(mapa)                       'Dibuja Pestaña

mapa.DrawLines(lapiz3, puntos)                 'Dibuja la señal

End Sub
```

Código fuente 5.34 Constructor 2 de la clase *fondoEspectro*

5.2.3.2. Espectros de piso

A diferencia de las demás gráficas mencionadas, (señales, osciladores y espectros de respuesta) los espectros de piso no tienen una ubicación fija dentro de la ventana, debido a que aparecen cuando el usuario hace clic sobre el área de una masa en específico y desaparecen cuando se hace clic en cualquier otra parte.

Debido a este efecto, para dibujar los espectros de piso fue necesario hacerlo sobre un formulario nuevo, el cual posee el tamaño justo del espectro de piso (180 x 180 píxeles) y el único control que contiene es del tipo *PictureBox* en el cual se dibuja la gráfica del espectro.

Para dibujar el espectro se utilizó la clase “fondoEspectro” (código fuente 5.33), que se explicó en la sección anterior.

El código fuente 5.35 muestra la clase que corresponde al formulario “frmEspecrosPiso”. La clase contiene un constructor y además tres funciones delimitadas por las palabras clave *Private Sub* y *End Sub*.

El constructor recibe los parámetros para poder ubicar el formulario y dibujar el espectro de piso. El único objetivo del constructor es pasar los valores de los parámetros a las variables globales de la clase. La tabla 5.8 muestra el significado de cada uno.

Tabla 5.8 Parámetros del constructor

Variable	Descripción
X, Y	Coordenadas para ubicar al formulario
Texto	Título de la gráfica
Puntos()	Vector de las coordenadas de cada punto que conforman al espectro
Ti	Valor del periodo inicial del espectro
Intervalo	Intervalo de tiempo entre los puntos del espectro
FEV	Factor de escala vertical
FEH	Factor de escala horizontal

La primera función en el código fuente 5.35 corresponde al evento *Load*; éste se ejecuta cada vez que el formulario aparece en pantalla. Esta función ubica a el formulario en las coordenadas (X, Y) y después invoca a la función “CambiarForma”.

La segunda función, corresponde al evento *LostFocus*, que se ejecuta cuando el formulario pierde el enfoque, es decir cada vez que el usuario haga clic en cualquier otra parte que no sea el formulario. La única sentencia dentro de esta función es *Close*, que cierra el formulario.

Por último, la función “CambiarForma” establece una nueva forma geométrica al formulario, para ello se definen las coordenadas de los puntos mediante las sentencias *ogPath.AddLine* y con la instrucción *Me.Region=New Region (ogPath)* se indica la forma del formulario. En la última línea de la función se declaró la variable “ObjetoEspectro” para hacer referencia a la clase “FondoEspectro” que dibuja la gráfica del espectro de piso sobre el *PictureBox*.

```
Imports System.Windows.Forms
Imports System.Drawing.Drawing2D
Imports System.Drawing

Public Class FrmEspectroPiso

Sub New(ByVal X As Integer, ByVal Y As Integer, ByVal Texto As String, ByVal Puntos() As
PointF, ByVal Ti As Single, ByVal Intervalo As Single, ByVal FEV As Single, ByVal FEH As
Single)

        ' This call is required by the Windows Form Designer.
        InitializeComponent()
        Me.X = X
        Me.Y = Y
        Me.PuntosGráfica = Puntos
        Me.Texto = Texto
        Me.Intervalo = Intervalo
        Me.Ti = Ti

        If FEV = 0 Then
            Me.FEV = 1
        Else
            Me.FEV = FEV
        End If

        Me.FEH = FEH
End Sub

'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Private Sub FrmEspectroPiso_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load                         'Evento cargar formulario

       Me.Location = New Point(Me.X, Me.Y) 'Ubicación del formulario
       Call CambiarForma()                  'Cambiar la forma del formulario
End Sub

'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Private Sub Dialog1_LostFocus(ByVal sender As Object, ByVal e As System.EventArgs) Handles
Me.LostFocus                                'Evento cambiar de enfoque
       Me.Close()                           'Cierra el formulario
End Sub

'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Private Sub CambiarForma()        'Función cambiar forma del formulario
Dim ogPath As New GraphicsPath    'variable para almaenar las coordenadas de la nueva
                                   forma del formulario

ogPath.AddLine(9, 0, 180, 0)      'Coordenadas de cada punto de la nueva forma
ogPath.AddLine(180, 0, 180, 180)  'que adoptara el formulario
ogPath.AddLine(180, 180, 0, 180)
ogPath.AddLine(0, 180, 0, 9)

Me.Region = New Region(ogPath)    'Se pasa el poligono como region para el form

Dim ObjetoEspectro As New FondoEspectro(Me.PictureBox1, Me.Texto, 10, 10, Me.PuntosGráfica)
End Sub

End Class
```

Código fuente 5.35 Formulario para espectros de Piso

5.3. ANIMACIÓN

En general, los procesos de animación consisten en colocar de forma consecutiva una serie de imágenes que al proyectarse en un mismo espacio dan una ilusión de movimiento.

Por ejemplo, para crear la animación de movimiento del oscilador (mostrado en la figura 5.14) de la posición A la posición B en un tiempo de 0.4 s, es necesario dibujar las imágenes intermedias entre estas dos posiciones y colocarlas consecutivamente una tras otra a cada intervalo de tiempo.

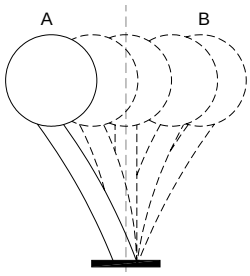


Figura 5.14 Movimiento de un oscilador de 1GL

Al ejecutar la animación, lo que el usuario verá serán las imágenes mostradas en la figura 5.15, una seguida de la otra en la misma posición pero no al mismo tiempo, lo que da la ilusión de que el oscilador se mueve desde el punto A hasta el punto B, manteniendo el eje y la base del oscilador en la misma ubicación. En este ejemplo se usaron 5 imágenes que se muestran en un intervalo de tiempo de 0.1 s, es decir, el usuario observará a cada 0.1 s una imagen distinta del oscilador proyectándose en la misma posición.

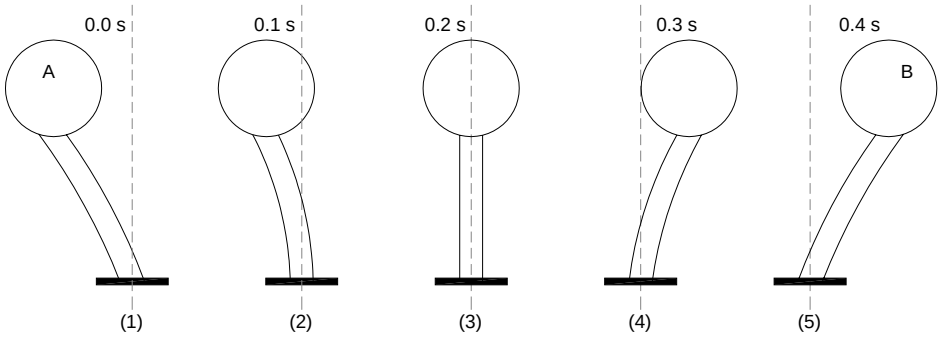


Figura 5.15 Movimiento por escenas de un oscilador de 1GL

Para tener un buen efecto de animación, las imágenes que se proyecten deben tener una estrecha relación con la consecutiva siguiente. Por ejemplo, en la figura 5.15 se observa cómo el oscilador se va moviendo paulatinamente del punto A al punto B, debido a que las imágenes varían sólo un poco una con respecto a su consecutiva siguiente. Por el otro el lado, si para mostrar la animación sólo se seleccionaran las imágenes 1, 3 y 5 el efecto de animación sería menos notorio y no se apreciaría bien.

Entre más imágenes haya para mostrar una animación mejor efecto se tendrá. Además, es necesario contar con un intervalo de tiempo pequeño entre las imágenes, ya que entre más corto sea mejor será el efecto. No obstante, existe un límite de imágenes, ya que el ojo no percibe más allá de 24 cuadros por segundo.

El programa a través del control *Timer* controla y ejecuta la proyección consecutiva de imágenes (señales de respuesta, osciladores y espectros de respuesta) a intervalos de tiempo definidos. Dichas imágenes son dibujadas dependiendo de los cálculos obtenidos para la respuesta dinámica.

5.3.1. Señales

Sólo las señales de la respuesta dinámica muestran un efecto de animación dentro del programa mientras que la señal de carga permanece estática en todo el proceso.

Las señales presentan dos tipos de animación: por adición y por sustitución.

- ✓ Por adición

Ocurre cuando las imágenes proyectadas cambian una con respecto a otra por que se le agrega una parte a la imagen anterior, manteniendo todo lo demás igual. Por ejemplo, en la figura 5.16 se muestran 6 imágenes que pertenecen a una animación de una señal; se puede observar que cada imagen es igual a la anterior pero con la diferencia de que se le agregó una línea más, dicha línea une al punto siguiente en la gráfica.

Si se colocan estas imágenes en una proyección, se vería a la respuesta dinámica recorriendo su historia en el tiempo, uniendo cada punto de la señal con una línea recta hasta completar su duración.

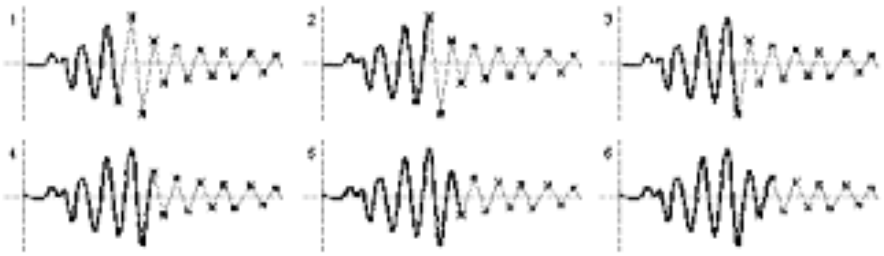


Figura 5.16 Secuencia de imágenes para la animación de una señal

Este tipo de animación se muestra en el programa cuando se presiona el botón *Play* en el panel de controles, al hacerlo se puede ver en pantalla la animación de las señales sincronizada con el movimiento del oscilador, este último se describe en la sección 5.3.2.

Para ejecutar la animación primero se calculan y luego se dibujan las señales que conforman a la respuesta dinámica; esto se describió en las secciones 5.1 y 5.2 respectivamente.

Cuando el usuario presiona el botón *Play* de la animación de los osciladores en el panel de controles, el programa llama al constructor 5 (código fuente 5.36) de la clase “GráficarSeñal” e inmediatamente después se ejecuta la función (evento *tick*) del control *Timer* mostrado en el código fuente 5.37.

El constructor 5 crea la primera imagen de la animación, la cual contiene cada parte de la gráfica (fondo, título, escala numérica, línea de referencia horizontal) con excepción de la señal. Dicha imagen funciona como base para las demás imágenes requeridas en el proceso de animación.

La explicación del contenido del constructor 5 es similar a la del constructor 2 mostrado en el código fuente 5.19 en la sección 5.2. La diferencia entre ellos es que el constructor 5 omite las líneas de código para dibujar la señal y la pestaña de la gráfica.

La instrucción que se encuentra en la penúltima línea del código fuente 5.36, copia el dibujo realizado sobre el *PictureBox* (imagen base) a la variable *Btmp*.

```
'Constructor 5 Dibuja la cuadrícula SIN la señal, SIN LA PESTAÑA antes de cada animación de
la señal
Sub New(ByVal pic As PictureBox, ByVal texto As String, ByVal Sx As Single, ByVal Sy As
Single, ByVal FE As Double, ByRef Btmp As Bitmap)

Dim Btmp1 As New Bitmap(pic.Width, pic.Height) 'Crea un Bitmap del tamaño del pictureBox
Dim mapa As Graphics = pic.CreateGraphics()    'Crea un área de trabajo a partir del Pb
pic.Image = Btmp1                              'La imagen del PictureBox será el Bitmap
mapa = Graphics.FromImage(Btmp1)               'Permite un buen efecto de animación en cada
                                                cambio de dibujo
Posicionmedia = pic.Height / 2       'Valor de la coordenada en Y de la línea de referencia
MitadAltura = (pic.Height) / 2       'Valor de la mitad de la altura libre para dibujar
NumLineas = CInt(MitadAltura \ Sy)             'Número de líneas a dibujar en la malla
Call dibujafondo(pic, mapa, Sx, Sy, 0)         'Dibuja el fondo con los valores calculados

mapa.DrawLine(lapiz2, CoordX, Posicionmedia, pic.Width, Posicionmedia) 'línea de referencia

Dim objetoTEXTO As New TextoHorizontal(NumLineas, CInt(pic.Height / 2), Sy, FE, mapa, texto,
CoordX)          'Coloca la escala númerica de la gráfica

Btmp = Btmp1     'Pasa el dibujo que hay en el Bitmap para guardarlo como fondo
End Sub
```

Código fuente 5.36 Constructor 5 de la clase *GráficaSeñal*

El código fuente 5.37 es el que hace posible la animación de las señales y los osciladores; el código muestra la función del control *Timer1*. Cuando la función es llamada por el programa, las instrucciones que se encuentran dentro de ella se ejecutan una y otra vez hasta que se indique que se detenga. Cada ciclo del *Timer* significa que dibuja una imagen de la animación.

El control tiene un comportamiento parecido a un bucle (*for o while*) pero con un intervalo de tiempo entre cada ciclo. El intervalo de tiempo del control *Timer1* es de 1 milisegundo, pero la velocidad de ejecución de las instrucciones contenidas en él, es relativa a las propiedades de la computadora, por ello la animación de la señal no puede tener un tiempo de reproducción igual al de la duración real de la respuesta dinámica.

Para saber en qué momento detener el control *Timer*, se colocó una sentencia *if* al principio de la función; ésta evalúa que el contador de puntos de la señal (n) no exceda al número total de puntos. El contador (n) se incrementa cada vez que el control realiza un ciclo.

Dentro del *if* se encuentran dos bucles *for*, el primero sirve para crear las animaciones de los osciladores, y el segundo para las animaciones de las señales. Los bucles controlan la cantidad de gráficas correspondientes a dibujar (osciladores y señales).

En el segundo bucle *for*, se encuentra la sentencia para llamar a la clase “GráficarSeñal” a través de su constructor 6 (código fuente 5.38). Cada vez que el control *Timer* ejecuta un ciclo, el bucle *for* ejecuta la llamada a la clase tantas veces como gráficas se hayan definido.

El constructor 6 (código fuente 5.38) es el encargado de dibujar sobre la imagen base (contenida en la variable *Btmp*) cada línea siguiente en la gráfica, para ello recibe entre sus parámetros un par de coordenadas (X1, Y1) y (X2, Y2) que indican el punto de inicio y final de la línea a dibujar, así como la imagen contenida en la variable *Btmp*.

```
Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Timer1.Tick

If n < maxpuntos Then         'Verifica que n sea menor al número de puntos de la señal
'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
  For j = 1 To Me.NUD.Value   'Ciclo para dibujar el movimiento de los osciladores
    Dim ObjetoOsc As New FondoOscilador(CapaOscilador(j), Me.NUDTamañoOscil.Value,
CCenOsc(j), CExtOsc(j), RD(j)(n) * FactorEscalaHorizOscil, n, RD(j)(n))
  Next j

'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
  For j = 1 To Me.NUDGráficasAct.Value 'Ciclo para dibujar el movimiento de las señales
    Dim objetoGráfica1 As New GráficarSeñal(AreaGráfica(j), BtmpGraf(j), Rr(j)(n).X,
Rr(j)(n).Y, Rr(j)(n + 1).X, Rr(j)(n + 1).Y, LapizGráfica(j), Grueso(j))
  Next j

  n += Me.NUDincremento.Value  'Contador de puntos de la señal
'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Else                           'En caso de llegar al numero de puntos de la señal
  Call DetenerAnimacion(True)  'Llama a la función para detener la animación
End If                         'Fin de la condición

End Sub                        'Fin de la función Timer
```

Código fuente 5.37 Evento *Tick* del control *Timer1*

Para que el constructor 6 dibuje sobre el *PictureBox* es necesario establecer un área de dibujo, (como se ha mencionado anteriormente). En este caso se observa que el *Bitmap* pasa como parámetro de referencia del constructor; esto quiere decir que la variable *Btmp* almacena una imagen previa, y además al pasar por valor de referencia la variable guarda cada cambio que sufra la misma.

Es decir, el constructor 6 dibuja una línea sobre una imagen definida (en este caso la gráfica de la señal hasta un determinado punto), después la misma variable *Btmp* guarda la imagen modificada y se repite el proceso en cada ciclo hasta que el dibujo de la señal termina.

Al ver este proceso ejecutarse de manera secuencial en intervalos de tiempo pequeños entre las imágenes, se obtiene el efecto de movimiento de la señal como si recorriera su historia en el tiempo.

```
'Constructor 6 Dibuja la señal por PASOS SIN LA PESTAÑA, SIN fondo
Sub New(ByVal pic As PictureBox, ByRef Btmp As Bitmap, ByVal X1 As Single, ByVal Y1 As
Single, ByVal X2 As Single, ByVal Y2 As Single, ByVal ColorLapiz As Color, ByVal Grueso As
Integer)

Dim PuntosAux(1) As System.Drawing.PointF   'Se crea un par de puntos(X,Y) para el inicio y
                                             final de la línea a dibujar
Dim mapa As Graphics = pic.CreateGraphics() 'Crea un área de trabajo a partir del PictureBox
pic.Image = Btmp                            'La imagen del PictureBox será el Bitmap que
                                             guardo el fondo
mapa = Graphics.FromImage(Btmp)             'Permite un buen efecto de animación en cada
                                             cambio de dibujo
PuntosAux(0) = New PointF(X1, Y1)           'Se definen las coordenadas del punto inicial
PuntosAux(1) = New PointF(X2, Y2)           'Se definen las coordenadas del punto final

Dim lapiz1 As New System.Drawing.Pen(ColorLapiz, Grueso) 'Formato de la línea de la gráfica
mapa.DrawLines(lapiz1, PuntosAux)                        'Se dibuja la línea
End Sub
```

Código fuente 5.38 Constructor 6 de la clase *GráficarSeñal*

- ✓ Por sustitución

Éste ocurre cuando en las imágenes a proyectar es necesario sustituir alguna parte del dibujo por otra nueva, manteniendo los demás elementos del dibujo igual.

La animación por sustitución se utiliza en la parte correspondiente a los espectros de respuesta; al momento de calcular los espectros y cuando se hace clic en el botón *Play* en el panel de controles, se observa como la respuesta dinámica de los osciladores va cambiando conforme su periodo cambia.

Para lograr esta animación es necesario dibujar las gráfica de respuesta del oscilador para cada periodo T. Las imágenes mantienen ciertos elementos de las gráfica constantes (fondo, título, escala numérica, línea de referencia mostrados en la figura 5.17B) y sólo cambia de una imagen a otra el dibujo de la señal (figura 5.17A).

En este caso se trata de una animación por sustitución pues en cada imagen sólo hay que sustituir la línea que representa a la respuesta dinámica por la que corresponda en cada valor del periodo.

A diferencia de la animación por adición, los cálculos correspondientes se van realizando al mismo tiempo que se ejecuta la animación, esto con el fin de evitar saturar la memoria almacenando los datos de las respuestas de los osciladores para cada espectro.

Las imágenes de la animación son dibujadas con la ayuda de dos *PictureBox* superpuestos uno encima del otro, es decir, un dibujo por capas. En la figura 5.17_A se muestra el dibujo de la señal sobre un *PictureBox* con fondo transparente; la figura 5.17_B muestra el dibujo de los elementos de las gráficas que permanecen constantes en cada imagen. Al superponer las dos imágenes se obtiene la figura 5.17_C.

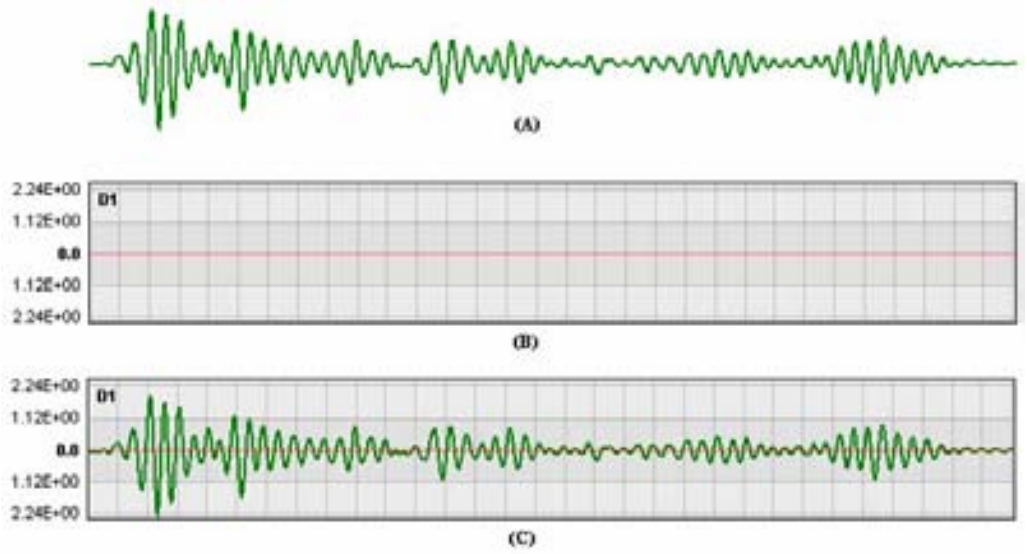


Figura 5.17 Dibujo de una señal por capas

En el proceso de las animaciones, el *PictureBox* declarado anteriormente en la sección 5.2.1 para dibujar las gráficas funcionará ahora para crear la imagen que permanece constante (figura 5.17_B).

Para dibujar las señales (5.17_A) se declaró una nueva matriz de controles *PictureBox* de nombre "CapaAnimaciónSeñal" en ambas ventanas (osciladores de 1GL y VGL), dichas matrices tienen el mismo tamaño que la matriz "AreaGráfica" correspondiente en cada ventana.

```
Dim CapaAnimacionSeñal(3) As New PictureBox
```

Cada vez que la animación se ejecuta, el programa llama a la función “PosiciónCapasAnimacionSeñal” (código fuente 5.39); ésta contiene un bucle *for* que establece las características para todas las gráficas activadas en la ventana.

Primero, se crea la imagen base para las gráficas (figura 5.17_B) llamando a la clase “GráficarSeñal” a través de su constructor 5 (código fuente 5.36); después se vinculan las dos matrices de controles definidas anteriormente mediante la sentencia *Controls.Add*, esto con la finalidad de establecer el fondo transparente en la matriz de controles “CapaAnimaciónSeñal” por medio de la propiedad *BackColor*. Por último se especifican las dimensiones (*Width y Height*) de los *PictureBox*.

```
Private Sub PosiciónCapasAnimacionSeñal(ByVal visibles As Boolean, ByVal gráficas As
Integer)

'Establece las propiedades de las capas para las animaciones de las señales
For Me.i = 1 To 3

Dim ObjetoGráfica As New GráficarSeñal(AreaGráfica(i), título(i), SeparacionX, SeparacionY,
Factor(i), BtmpGraf(i))

AreaGráfica(i).Controls.Add(CapaAnimacionSeñal(i))  'Agrega la capa como control del Pb
CapaAnimacionSeñal(i).BackColor = Color.Transparent 'Asigna el color transparente a la capa

CapaAnimacionSeñal(i).Visible = visibles
CapaAnimacionSeñal(i).Width = AreaGráfica(i).Width
CapaAnimacionSeñal(i).Height = AreaGráfica(i).Height
Next
End Sub
```

Código fuente 5.39 Llamado de la clase *GráficaSeñal*

Una vez dibujada la imagen base, se ejecuta la función del control *Timer* (código fuente 5.40) para generar la animación. La función tiene las mismas características que la descrita en el código 5.37. En este caso el control tiene el nombre de *TmEspectros* para diferenciarlo de la función que controla la animación de los osciladores.

```
Private Sub TmEspectros_Tick(ByVal sender As Object, ByVal e As System.EventArgs) Handles
TmEspectros.Tick

If p <= PuntosEsp Then         'Verifica que n sea menor al número de puntos del espectro
'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
For Me.i = 1 To NUDGráficasAct.Value 'Ciclo para dibujar el movimiento de los espectros
  Dim ObjetoFondo As New FondoEspectro(Me.AreaOscilador(i), BtmpEsp(i), Er(i)(m).X,
Er(i)(m).Y, Er(i)(m + 1).X, Er(i)(m + 1).Y)
Next

'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
For Me.i = 1 To gráficas         'Ciclo para dibujar el movimiento de las señales
  Dim ObjetoSeñal As New GráficarSeñal(CapaAnimacionSeñal(i), Raux(i), LapizGráfica(i),
Grueso(i))
Next
p += 1                          'Contador de puntos del espectro
'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Else                                    'En caso de llegar al numero de puntos de la señal
  Call DetenerAnimacionEspectros(True) 'Llama a la función para detener la animación
End If                                  'Fin de la condición
End Sub                                 'Fin de la función Timer
```

Código fuente 5.40 Evento *Tick* del control *TmEspectros*

El código fuente 5.40 es el encargado de mostrar la animación de las señales en sincronía con los espectros de respuesta.

Al igual que la función del control *Timer* (código fuente 5.37), se colocó una sentencia *if* al principio de la función; la cual evalúa que el contador de puntos del espectro (p) no exceda al número total de puntos del mismo. El contador (p) se incrementa cada vez que el control realiza un ciclo.

Dentro de la sentencia *if* se encuentran dos bucles *for*, el primero controla las animaciones de los espectros de respuesta (descritos en la sección 5.3.3) y el segundo hace lo propio para las animaciones de las gráficas de respuesta.

En el segundo bucle *for* se encuentra la sentencia para llamar a la clase "GráficarSeñal" a través de su constructor 4 (código fuente 5.41). Cada vez que el control *Timer* ejecuta un ciclo, el bucle *for* ejecuta la llamada a la clase tantas veces como gráficas se hayan definido.

Los parámetros que requiere el constructor 4 se enlistan en la siguiente tabla.

Tabla 5.9 Parámetros del constructor

Variable	Descripción
Pic	Nombre del *PictureBox* donde se va a dibujar la gráfica
Puntos()	Vector de las coordenadas de cada punto que conforman a la señal
ColorLapiz	Color de la línea de la señal
Grueso	Espesor (píxeles) de la línea de la señal

Obsérvese que en el código fuente 5.39 el parámetro que pasa como *PictureBox* es la matriz de controles "CapaAnimaciónSeñal" destinada para dibujar sólo la señal de la gráfica, tal como se mencionó anteriormente.

```
'Constructor 4 Dibuja la señal completa, SIN fondo, SIN pestaña, SIN nada, para la animación
en la parte de los espectros
Sub New(ByVal pic As PictureBox, ByVal puntos() As PointF, ByVal ColorLapiz As Color, ByVal
Grueso As Integer)

Dim lapiz1 As New System.Drawing.Pen(ColorLapiz, Grueso) 'Establece formato de la línea

Dim Btmp As New Bitmap(pic.Width, pic.Height)   'Crea un Bitmap del tamaño del PictureBox
Dim mapa As Graphics = pic.CreateGraphics()     'Crea un área de trabajo a partir del Pb
pic.Image = Btmp                                'La imagen del PictureBox será el Bitmap
mapa = Graphics.FromImage(Btmp)                 'Permite un buen efecto de animación en cada
                                                 cambio de dibujo
mapa.DrawLines(lapiz1, puntos)                  'Dibuja la señal
End Sub
```

Código fuente 5.41 Constructor 4 de la clase *GráficaSeñal*

El constructor 4 de la clase "Gráficarseñal" (código fuente 5.41), define primero un formato para la línea de la señal a partir de los valores de los parámetros que recibe (color y espesor de línea), después establece un área de dibujo (descrita anteriormente), y por último la única instrucción de dibujo *DrawLines* dentro del constructor indica que se dibuje la señal a partir del vector de puntos que también paso como parámetro.

5.3.2. Osciladores

La animación de ambos tipos de osciladores (1GL y VGL) es por sustitución. Se dibujan en un *PictureBox* las partes de la señal que permanecen estáticas en todo el proceso y en un segundo control se dibuja el oscilador (masas y resortes) que es el elemento de las gráficas que cambia en cada instante de tiempo.

5.3.2.1. Osciladores de 1GL

Debido a la forma de dibujar las gráficas de los osciladores (descrita en la sección 5.2.2.1), sobre el control *PictureBox* "AreaOscilador" se dibuja el fondo de los osciladores (figura 5.18b) y en el *PictureBox* "CapaOscilador" se dibuja la masa y el resorte junto con el texto que indica el desplazamiento en cada instante de tiempo (figura 5.18a). Al superponer las dos imágenes se obtiene la figura 5.18c.

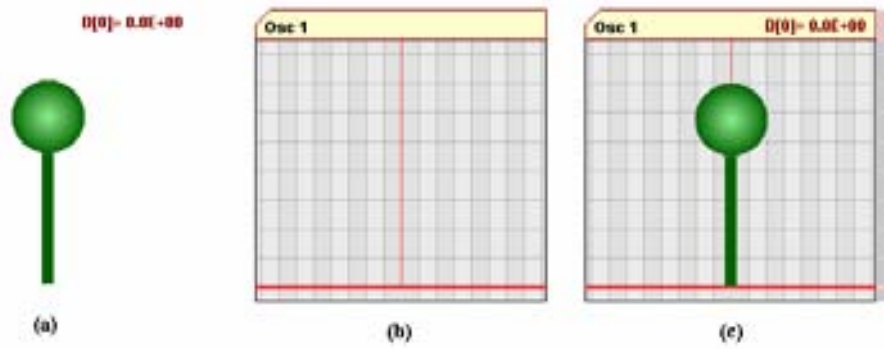


Figura 5.18 Dibujo del oscilador por capas

Debido a que el movimiento de los osciladores está sincronizado con la animación de las señales, el llamado a las clases para dibujar ambas gráficas (oscilador y señales) se encuentra dentro del mismo control *Timer*, (código fuente 5.37).

El llamado a la clase "FondoOscilador" se encuentra dentro del primer bucle *for* de la función del control *Timer*, y se hace por medio del constructor 2 de dicha clase (código fuente 5.24).Los parámetros que necesita el constructor se encuentran en la tabla 5.4.

Se puede observar que al llamar a la clase "FondoOscilador", en el código fuente 5.37, pasa como parámetro la matriz "CapaOscilador", que es la que contiene a los *PictureBox* sobre los cuales se dibujará cada imagen del oscilador (figura 5.18a) desplazado de su eje según corresponda la respuesta dinámica.

5.3.2.2. Oscilador de VGL

Para crear el efecto de animación de este oscilador se sigue el mismo procedimiento descrito para los osciladores de 1GL. La forma de dibujarlo es por medio de dos *PictureBox*, el proceso se describió en la sección 5.2.2.2.

Una vez que el usuario hace clic en el botón *Play* en el panel de controles, el programa llama a la función *Timer* (código fuente 5.42); esta función y la mostrada en el código fuete 5.37 se declaran en diferentes ventanas (osciladores 1GL y VGL) por lo tanto no tienen relación entre ellos.

El código fuente 5.42 es el encargado de crear la animación de las señales y el oscilador en la ventana de Osciladores de VGL. La función tiene las mismas características y comportamiento que la función mostrada en el código fuente 5.37.

La primera instrucción dentro de la sentencia *if*, es para llamar a la clase "FondoOsciladorVGL", por medio de su constructor 2 (código fuente 5.31); obsérvese que pasa como parámetro la matriz "CapaOscilador" que es la que contiene a los *PictureBox* sobre los cuales se dibujará cada imagen del oscilador (masas y resortes).

En el bucle *for*, se encuentra la sentencia para llamar a la clase "GráficarSeñal" a través de su constructor 6 (código fuente 5.38). Esta parte de la función es exactamente igual a la descrita en el código fuente 5.37 y su función es también crear la animación de las señales de respuesta.

```
Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Timer1.Tick

If n < maxpuntos Then    'Verifica que n sea menor al número de puntos de la señal

'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
   Dim ObjetoFondo1 As New FondoOsciladorVGL(CapaOscilador, "[" & paso & "] Punto", tamañoM,
PuntosAux, Valores, NGL, Me.ChBMostrarDesp.Checked, paso)

'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
   For Me.i = 1 To Me.NUDGráficasAct.Value
       Dim ObjetoGráfica As New GráficarSeñal(AreaGráfica(i), BtmpGraf(i), Rr(i)(n).X,
Rr(i)(n).Y, Rr(i)(n + 1).X, Rr(i)(n + 1).Y, LapizGráfica(i), Grueso(i))
   Next

   n += Me.NUDincremento.Value  'Contador de puntos de la señal
'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Else                           'En caso de llegar al numero de puntos de la señal
   Call DetenerAnimacion(True)  'Llama a la función para detener la animación
End If                         'Fin de la condición

End Sub                        'Fin de la función Timer
```

Código fuente 5.42 Evento *Tick* del control *Timer1*

5.3.3. Espectros

Sólo los espectros de respuesta tienen efecto de animación, mientras que los espectros de piso se muestran estáticos a lo largo del programa y aparecen cuando el usuario hace clic sobre alguna de las masas del oscilador de VGL.

Los espectros de respuesta presentan el tipo de animación por adición. Se dibuja en un *PictureBox* la imagen base (figura 5.19a) y para cada imagen consecutiva se va agregando la línea que une el punto siguiente en el espectro de respuesta a cada intervalo de tiempo (figura 5.19b, c y d).

Debido a que el movimiento de los espectros está sincronizado con la animación de las señales, el llamado a las clases para dibujar ambas gráficas (señales y espectros) se encuentra dentro del control *TmEspectros*, (código fuente 5.40).

El llamado a la clase "FondoEspectro" se encuentra dentro del primer bucle *for* de la función, y se hace por medio del constructor 3 de dicha clase (código fuente 5.43).Los parámetros que necesita el constructor se encuentran en la tabla 5.10.

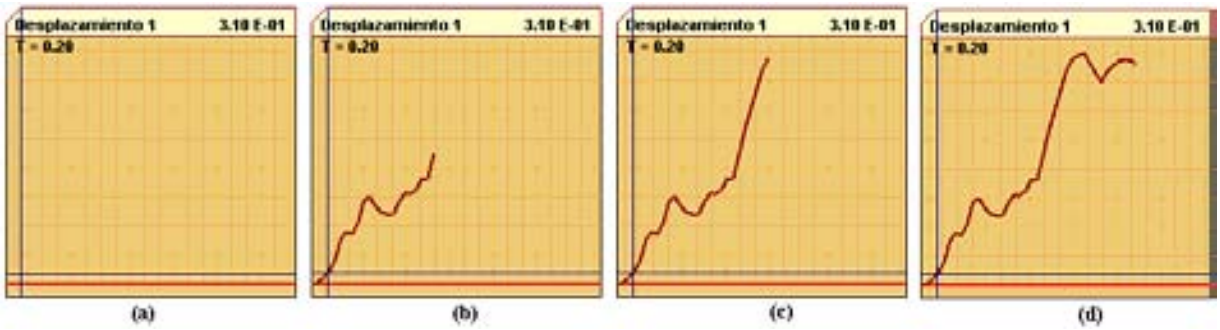


Figura 5.19 Dibujo por adición del espectro

Para que el constructor 3 dibuje sobre el *PictureBox* es necesario establecer un área de dibujo, (como se ha mencionado anteriormente). En este caso se observa que el *Bitmap* pasa como parámetro de referencia del constructor; esto quiere decir que la variable *Btmp* almacena una imagen previa, y además al pasar por valor de referencia la variable guarda cada cambio que sufra la misma.

Es decir, el constructor 3 dibuja una línea sobre una imagen definida (figura 5.19a), después la misma variable *Btmp* guarda la imagen modificada y se repite el proceso en cada ciclo hasta que el dibujo del espectro esté terminado.

Tabla 5.10 Parámetros del constructor 3 de la clase "FondoEspectro"

Variable	Descripción
Pic	Nombre del *PictureBox* donde se va a dibujar la gráfica
Btmp	Imagen contenida en un mapa de bits (*Bitmap*)
(X1, Y1)	Coordenada del punto inicial de la línea que se va dibujar
(X2, Y2)	Coordenada del punto final de la línea que se va dibujar

Al ver este proceso ejecutarse de manera secuencial en intervalos de tiempo pequeños entre las imágenes, se obtiene el efecto de movimiento de la señal del espectro y pareciera como si recorriera su trayectoria en el espectro.

```
'Constructor 3 Dibuja la gráfica del espectro POR PASOS
Sub New(ByVal pic As PictureBox, ByRef Btmp As Bitmap, ByVal X1 As Single, ByVal Y1 As
Single, ByVal X2 As Single, ByVal Y2 As Single)

Dim PuntosAux(1) As System.Drawing.PointF    'Se crea un par de puntos(X,Y) para el inicio y
                                              final de la línea a dibujar
Dim mapa As Graphics = pic.CreateGraphics() 'Crea un área de trabajo a partir del PictureBox
pic.Image = Btmp                             'La imagen del PictureBox será el Bitmap que
                                              guardo la imagen base
mapa = Graphics.FromImage(Btmp)              'Permite un buen efecto de animación en cada
                                              cambio de dibujo
PuntosAux(0) = New PointF(X1, Y1)            'Se definen las coordenadas del punto inicial
PuntosAux(1) = New PointF(X2, Y2)            'Se definen las coordenadas del punto final

mapa.DrawLines(lapiz3, PuntosAux)            'Se dibuja la línea
End Sub                                      'Fin del constructor
```

Código fuente 5.43 Constructor 3 de la clase *FondoEspectro*

5.4. MANEJO DE ERRORES

Se contemplaron dos tipos errores: lógico y de ejecución. Estos últimos son los más notorios para el usuario pues el programa muestra un mensaje de error cuando se producen.

5.4.1. Errores de ejecución

Los errores de ejecución se controlaron mediante la restricción de datos de entrada en los campos de texto. El programa está diseñado para detectar en qué momento se introducen valores erróneos en los controles *Textbox*, *DataGridView* y *RichTexbox*.

El software realiza el proceso de recibir los datos, evaluarlos y determinar si son aceptables o no; en el caso de ser datos incorrectos se envía un mensaje de advertencia y se suspende el proceso (que se esté llevando a cabo en ese momento) hasta que se corrija el dato erróneo. Para efectuar el proceso se usaron los mensajes de advertencia junto con la validación de los campos de texto; estos puntos se describen en las secciones 5.4.3 y 5.4.4 de este capítulo.

5.4.2. Errores lógicos

Estos fueron resueltos en la depuración del programa, a través del proceso de “prueba y error”, es decir, durante su elaboración el software se ponía a prueba para cerciorarse de que funcionara adecuadamente cada vez que se agregaba un control a la interfaz gráfica y/o modifica el código fuente.

El programa es confiable en cuanto al procesamiento de datos (respuesta dinámica) ya que ésta fue verificada con otros programas y cálculos en libros; esto se describe en el capítulo siguiente.

Con respecto al uso de los controles, la depuración continua ayudó a establecer perfectamente el uso adecuado de cada uno en el programa; además, el software fue proporcionado a los alumnos de la clase de Dinámica Estructural para su valoración, esto se describe también en el siguiente capítulo.

El programa puede sufrir errores debido a un mal manejo, en cuyo caso el software terminará su ejecución debido a que dichas acciones no están contempladas dentro de su uso. No obstante, la GUI está diseñada de manera tal que es poco probable que se presenten errores debido a estas situaciones.

5.4.3. Mensajes de advertencia

Los Message Box (*MsgBox*) son controles predeterminados de Visual Studio para mostrar un mensaje al usuario. Su uso es muy variado y depende propiamente de cómo los utilice el programador.

Su función (en el presente trabajo) es mostrar un mensaje de texto al usuario, que indica que ha habido un error en la introducción de datos y advierte al usuario de que es o no posible llevar a cabo la operación.

La figura 5.20 muestra un *MsgBox* que aparece en el programa cuando se intenta introducir datos erróneos en campos de texto numéricos. El formato de este control es estándar y las partes que lo componen se indican debajo de la figura.

Cada elemento del MsgBox señalado en la figura 5.20 lo puede editar el programador. Los mensajes que se muestran a lo largo del programa dependen del tipo de error en el que se incurra, el icono expuesto se utilizó siempre para los mensajes de advertencia o aviso, y los botones en la parte baja del formulario dependen de las opciones para resolver dicho problema.

Figura 5.20 Ventana de MsgBox

#	Control	Descripción
1	Título	Muestra el nombre del programa y su versión
2	Icono	Figura predeterminada de Visual Studio para indicar una advertencia
3	*Button*	Botón de opción del mensaje, en algunos casos mostrará más de uno con opciones como: Aceptar, Cancelar y/o Reintentar
4	*Button*	Botón para cerrar la ventana
5	*Label*	Mensaje que muestra cuál es el error en el programa

Para mostrar el *MsgBox* de la figura 5.20 en el programa, se requiere del código fuente 5.44; esta instrucción es general para cualquier Msgbox que se desee mostrar:

```
MsgBox("Valor no aceptable, revisa tus datos", MsgBoxStyle.Exclamation, "Dinámica V 3.1")
```

Código fuente 5.44 Llamado de un MsgBox

La sentencia *MsgBox* ( ) muestra al control; dentro del paréntesis se indican los parámetros, separados por comas, para cada parte del mismo, de izquierda a derecha son:

- ✓ Mensaje de texto a mostrar en el control
- ✓ Tipo de icono
- ✓ Título de la ventana

Se puede observar que en ninguna parte del código fuente 5.44 se especificaron los botones que se deseaba ver; Visual Studio coloca por default el botón aceptar (parte baja de la ventana) y el botón para cerrar la ventana (parte superior derecha).

Para definir los botones a mostrar en la ventana se hace en conjunto con la indicación del icono agregando el signo “+” y el grupo de botones a mostrar. El código fuente 5.45 expone cómo se agregan los botones “Ok” y “Cancel” a un *MsgBox*.

```
MsgBox("La señal tiene más de 12000 datos " & vbCrLf & "¿Desea cargar los datos?",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, version)
```

Código fuente 5.45 Llamado de un MsgBox con especificación de botones

Específicamente los *Msgbox* se programaron para aparecer cuando se presenta cualquiera de los siguientes casos:

- ✓ Por la mala escritura de un número, por ejemplo: en vez de escribir *12.34* escribe *12.3.4*.
- ✓ Cuando los valores que introduce están fuera de un rango, tal es el caso de los valores de amortiguamiento ($0 < \xi \leq 1$) y el valor de los periodos ($T \geq 0$) por mencionar algunos.
- ✓ Cuando los resultados de las operaciones con dichos datos es errónea. Por ejemplo, al calcular el número de puntos (Np) de la señal tipo Senoide esté fuera de rango ($50 \leq Np \leq 12000$).

Los mensajes de advertencia aparecen sólo en los controles que admiten valor, es decir, cuando contienen un campo de texto (*Textbox, DataGridView* o *RichTexbox*). Debido al diseño de la GUI son pocas las ventanas que tienen esta característica, a lo largo del programa el usuario encontrará diversos mensajes que le indicaran cuando se presente un error.

Como ejemplo de los mensajes de advertencia, la tabla 5.11 muestra los que la ventana "Senoide" contiene.

Tabla 5.11 Mensajes de error en la ventana "Senoide"

Ventana	Control	Causa de error	Mensaje
Senoide	*TextBox*	(Amplitud, Duración, Periodo y Δt) < 0	Ninguno de los datos puede ser menor o igual a CERO
		Número de puntos: Np ($50 < Np < 12000$)	La señal debe tener como mínimo 50 puntos y como máximo 12000 puntos
		Escribir mal algún valor	Valor no aceptable, revisa tus datos

Como se mencionó anteriormente, el programa puede contener ciertos errores debido a un mal uso. Cuando esto suceda, Visual Studio abrirá una ventana de mensaje de error predeterminada (figura 5.21), la cual muestra el motivo de dicho error (explicado en términos de lenguaje de programación) y da opciones de continuar o salir del programa en ese momento.

Figura 5.21 Ventana de error no controlado

5.4.4. Validación de los campos de texto

Validar significa (en términos de programación) aprobar los datos que se introducen en un control, a partir de ciertos parámetros establecidos.

Para validar los datos que se introducen en los campos de texto de los controles *TextBox* y *DataGridView* de cualquier ventana dentro del programa se creó la clase "Validación" (código fuente 5.46).

La clase "Validación" presenta 4 constructores, uno para ingresar los datos provenientes del *TextBox* y los otros tres para el control *DataGridView*. Además, contiene cuatro funciones (eventos de validación) para el control *TextBox* y tres funciones (un evento de validación, una función del tipo *Public* y otra *Private*) para el control *DataGridView*.

```
Public Class ClaseValidacion
    WithEvents TextBox As New TextBox       'Recibe todos los argumentos del Textbox
    WithEvents DataGrid As New DataGridView 'Recibe todos los argumentos del DatagGridView

    Dim TextoValido As String               'Caracteres validos para el textBox
    Dim RevisionEntero As Boolean           'Indica si revisa que el dato sea de tipo entero
    Dim ValorMinimo, ValorMaximo As Single  'Valores que delímitan al rango
    Dim RepetirValor As Boolean             'Indica si buscan valores repetidos en la fila
    Dim LimitarValorMaximo As Boolean
    Dim LimitarValorMinimo As Boolean

    '1er Constructor: TEXTBOX
    Sub New(ByVal TextBox As TextBox, ByVal TextoValido As String)

    '2do Constructor: DATAGRIDVIEW
    Sub New(ByVal DataGrid As DataGridView, ByVal Entero As Boolean, ByVal ValorMinimo As
Single, ByVal ValorMaximo As Single, ByVal RepetirValorFila As Boolean)

    '3ro Constructor: DATAGRIDVIEW
    Sub New(ByVal DataGrid As DataGridView, ByVal Entero As Boolean, ByVal ValorMinimo As
Single, ByVal RepetirValorFila As Boolean)

    '4to Constructor: DATAGRIDVIEW
    Sub New(ByVal DataGrid As DataGridView, ByVal Entero As Boolean, ByVal RepetirValorFila
As Boolean)

    '************************** VALIDACION DEL TEXTBOX ************************

    Private Sub TextBox_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TextBox.KeyPress

    Private Sub TextBox_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles TextBox.Validating

    Private Sub TextBox_GotFocus(ByVal sender As Object, ByVal e As System.EventArgs)
Handles TextBox.GotFocus

    Private Sub TextBox_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
TextBox.Click

    '************************** VALIDACION DEL DATAGRID ************************

   Private Sub DataGridView_CellValidating(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellValidatingEventArgs) Handles DataGrid.CellValidating

   Public Sub DetenerValidacionDataGrid()

   Private Sub MensajeError(ByVal Mensaje As String, ByVal fila As Integer)
```

Código fuente 5.46 Clase Validación

Al principio de la clase “Validación” se declararon dos variables del mismo tipo que los controles en cuestión (*TextBox y DataGrid*); esto con la finalidad de pasar todos los valores y reconocer los eventos de dichos controles en la clase.

Los cuatro constructores sólo tienen la finalidad de pasar los valores de los parámetros que reciben a las variables globales definidas en la parte superior de la clase con la palabra clave *Dim*.

El código fuente 5.47 muestra el contenido de los cuatro constructores. El primero es para la validación de datos del *TextBox;* los parámetros que recibe son: el nombre del control y el texto valido o permitido para ese control.

En los constructores para los DataGridView se requieren más parámetros, ya que se revisan las siguientes condiciones para los valores introducidos en las celdas:

- ✓ Si es número entero o decimal
- ✓ Si requiere cumplir con un valor mínimo
- ✓ Si requiere cumplir con un valor máximo
- ✓ Si se permite repetir el mismo valor en la fila

```
     '1er Constructor: TEXTBOX
    Sub New(ByVal TextBox As TextBox, ByVal TextoValido As String)
        Me.TextBox = TextBox
        Me.TextoValido = TextoValido
    End Sub

    '2do Constructor: DATAGRIDVIEW Revisar valores máximos y mínimos y repeteción de datos
    Sub New(ByVal DataGrid As DataGridView, ByVal Entero As Boolean, ByVal ValorMinimo As
Single, ByVal ValorMaximo As Single, ByVal RepetirValorFila As Boolean)
        Me.DataGrid = DataGrid
        Me.RevisionEntero = Entero
        Me.ValorMaximo = ValorMaximo
        Me.ValorMinimo = ValorMinimo
        Me.RepetirValor = RepetirValorFila
        Me.LimitarValorMaximo = True
        Me.LimitarValorMinimo = True
    End Sub

    '3er Constructor: DATAGRIDVIEW Revisar valores mínimos y repeteción de datos
    Sub New(ByVal DataGrid As DataGridView, ByVal Entero As Boolean, ByVal ValorMinimo As
Single, ByVal RepetirValorFila As Boolean)
        Me.DataGrid = DataGrid
        Me.RevisionEntero = Entero
        Me.ValorMinimo = ValorMinimo
        Me.RepetirValor = RepetirValorFila
        Me.LimitarValorMaximo = False
        Me.LimitarValorMinimo = True
    End Sub

    '4to Constructor: DATAGRIDVIEW Revisar repeteción de datos
    Sub New(ByVal DataGrid As DataGridView, ByVal Entero As Boolean, ByVal RepetirValorFila
As Boolean)
        Me.DataGrid = DataGrid
        Me.RevisionEntero = Entero

        Me.RepetirValor = RepetirValorFila
        Me.LimitarValorMinimo = False
        Me.LimitarValorMaximo = False
    End Sub
```

Código fuente 5.47 Constructores de la clase Validación

Los parámetros que requieren los constructores de la clase “Validación” (para el control *Data Grid View*) se muestran en la tabla 5.12. Obsérvese que cada constructor sólo pasa los valores de sus parámetros a las variables globales.

Los tres constructores para el control *DataGridView* requieren como parámetros, el nombre del control, la variable condicional para revisar si el valor es número entero y también la variable condicional para permitir la repetición del valor en la misma fila. La diferencia entre los constructores es que el primero requiere de un valor máximo y un mínimo, el segundo sólo requiere el valor mínimo y el tercero no requiere valores.

Tabla 5.12 Parámetros de los constructores (2, 3 y 4) de la clase Validación

Variable	Descripción
DataGrid	Nombre del DataGridView que recibe el valor
Entero	Condición (True o False) que indica si revisa que el número sea entero o no
ValorMínimo	Valor numérico mínimo que debe cumplir el valor introducido
ValorMaximo	Valor numérico máximo que debe cumplir el valor introducido
RepetirValorFila	Condición (True o False) que indica si permite la repetición del valor introducido en la misma fila.

La revisión de los datos se hace a través de las funciones (eventos de validación) de cada control. Para el *TextBox* se muestran en el código fuente 5.48 el cuerpo de dichas funciones y en la tabla 5.13 el momento cuando se activa cada evento.

```
'****************************** VALIDACION DEL TEXTBOX *********************************

    Private Sub TextBox_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles TextBox.KeyPress
        If InStr(Me.TextoValido & Chr(8), e.KeyChar) = 0 Then
            e.Handled = True
            Beep()
        End If
    End Sub

    Private Sub TextBox_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles TextBox.Validating
        If Not IsNumeric(Me.TextBox.Text) Then
            MsgBox("Valor no aceptable, rebice sus datos", MsgBoxStyle.Exclamation, version)
            e.Cancel = True
        End If
    End Sub

    Private Sub TextBox_GotFocus(ByVal sender As Object, ByVal e As System.EventArgs)
Handles TextBox.GotFocus
        Me.TextBox.SelectionStart = 0
        Me.TextBox.SelectionLength = Len(Me.TextBox.Text)
    End Sub

    Private Sub TextBox_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
TextBox.Click
        Me.TextBox.SelectionStart = 0
        Me.TextBox.SelectionLength = Len(Me.TextBox.Text)
    End Sub
```

Código fuente 5.48 Eventos de validación del control *TextBox*

El evento *KeyPress* (código fuente 5.48) del control *TextBox* permite sólo escribir en el control los caracteres permitidos, contenidos en el parámetro “TextoValido” más la tecla “Delete”, bloqueando todos los demás.

El evento *Validating* del control *TextBox* verifica que el texto introducido sea un valor numérico; en caso de no serlo el programa muestra un *MsgBox* con el mensaje "Valor no aceptable, revisa tus datos", y mediante la instrucción *e.cancel*, se especifica que el control no pierda el enfoque hasta corregir el valor erróneo.

Los eventos *Got Focus* y *Click* seleccionan el texto en el control *TextBox* cuando se hace clic en él.

Tabla 5.13 Eventos para el control *TextBox*

Evento	Se activa cuando …
KeyPress	se introduce un dato cualquiera
Validating	se hace clic fuera del control
Got Focus	el control obtiene el enfoque
Click	se hace clic en el control

El *DataGridView* sólo requiere de un evento de validación llamado *CellValidating*, (código fuente 5.49); éste se activa cuando el usuario introduce un valor en cualquiera de las celdas del control. En el cuerpo de la función se revisan cinco condiciones con la ayuda de la sentencia *if*.

La primera condición verifica que el valor introducido sea numérico; de no serlo el programa llama a la función “MensajeError” para mostrar un MsgBox con el mensaje " El dato no es un valor numérico"; después con la instrucción *Me.DataGrid.CancelEdit* ( ) se borra el valor erróneo en la celda y se recupera el valor correcto anterior, al ejecutar esto el programa sale de la función.

En caso de pasar la primera verificación (que el dato sea numérico), el programa revisa cuatro condiciones más, dependiendo de los valores capturados en las variables globales.

El segundo *if* sirve para comprobar si el valor numérico es del tipo entero (en caso de que el parámetro del constructor haya especificado “Entero” = *True*). El *if* muestra las mismas líneas de código que la primera condición, en caso de no ser un entero el programa llama a la función “MensajeError” especificando el mensaje como: "El valor debe ser un valor ENTERO”.

El tercer *if* evalúa (si “Me.LimitarValorMinimo” = *True*) que el valor introducido en la celda sea mayor que el mínimo especificado. Al igual que los dos *if* anteriores, de no cumplir con la condición el programa llama a la función “MensajeError” con el texto "El valor MÍNIMO es: " seguido del valor que se especificó como mínimo en los parámetros de constructor.

El cuarto *if* hace lo propio con el valor máximo (en el caso de que “Me.LimitarValorMaximo” = *True*) llamando a la función “MensajeError” con el texto "El valor MAXIMO es: " seguido del valor que se especificó en los parámetros de constructor.

El último *if* restringe la repetición del valor introducido en la misma fila (en caso de que “Me.RepetirValor” = *True*) y de igual forma que los anteriores llama a la función “MensajeError” con el texto "No se pueden repetir los valores en la tabla".

```
    Private Sub DataGridView_CellValidating(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellValidatingEventArgs) Handles DataGrid.CellValidating
        Dim Valor As String = e.FormattedValue.ToString() 'Dato que se introdujo
        Me.DataGrid.Rows(e.RowIndex).ErrorText = ""       'Mensaje de error para la fila

        '1) Primera revisión {DATO NUMERICO}
        If Not IsNumeric(Valor) Then
            e.Cancel = True
            MensajeError("El dato no es un valor númerico", e.RowIndex)
            Me.DataGrid.CancelEdit() 'Deshace el cambio de valor en la celda
        Else
            '2) Segunda revisión {DATO NUMERICO DEL TIPO ENTERO}
            If Me.RevisionEntero = True Then
                Dim ValorRes As Integer
                If Not Integer.TryParse(Valor, ValorRes) Then 'Solo numeros enteros
                    e.Cancel = True
                    MensajeError("El valor debe ser un valor ENTERO", e.RowIndex)
                    Me.DataGrid.CancelEdit()'Deshace el cambio de valor en la celda
                    Exit Sub
                End If
            End If

            '3) Tercera revisión {VALOR MINIMO}
            If Me.LimitarValorMinimo = True Then
                If Valor < ValorMinimo Then
                    e.Cancel = True
                    MensajeError("El valor MÍNIMO es: " & ValorMinimo, e.RowIndex)
                    Me.DataGrid.CancelEdit()'Deshace el cambio de valor en la celda
                    Exit Sub
                End If
            End If

            '4) Cuarta revisión {VALOR MAXIMO}
            If Me.LimitarValorMaximo = True Then
                If Valor > ValorMaximo Then
                    e.Cancel = True
                    MensajeError("El valor MÁXIMO es: " & ValorMaximo, e.RowIndex)
                    Me.DataGrid.CancelEdit()'Deshace el cambio de valor en la celda
                    Exit Sub
                End If
            End If

            '5) Quinta revisión {REPETICIÓN DE VALOR EN LA MISMA FILA}
            If Me.RepetirValor = True Then
                For i = 0 To Me.DataGrid.ColumnCount - 1
                    If CInt(Valor) = Me.DataGrid.Item(i, e.RowIndex).Value Then
                        If i <> e.ColumnIndex Then
                            e.Cancel = True
                            MensajeError("No se pueden repetir los valores en la tabla",
e.RowIndex)
                            Me.DataGrid.CancelEdit()'Deshace el cambio de valor en la celda
                            Exit Sub
                        End If
                    End If
                Next
            End If
        End If
    End Sub

    Public Sub DetenerValidacionDataGrid()
        Me.DataGrid.CancelEdit()'Deshace el cambio de valor en la celda
    End Sub

    Private Sub MensajeError(ByVal Mensaje As String, ByVal fila As Integer)
        Beep()
        Me.DataGrid.Rows(fila).ErrorText = Mensaje
        MsgBox(Mensaje, MsgBoxStyle.Exclamation, version)
    End Sub
```

Código fuente 5.49 Funciones del control DataGridView

El código fuente 5.50 muestra como se implementa la clase “Validación” sobre los controles *TextBox* y *DataGridView* de la ventana Espectros de respuesta. Se puede observar en la sección 4.2.5.7 del capítulo anterior que la ventana presenta los dos tipos de controles.

Primero se declara un objeto de la clase “Validación” al principio del código. Después en el evento *Load* del formulario (ventana), dentro de una sentencia *if*, se hace la referencia completa a la clase. La variable “ObjetoData” invoca a la clase validación mediante su constructor 4 (mostrado en el código fuente 5.47) indicando en sus parámetros el nombre del control *DataGridView* (Me.DGEspectros), que no haga la revisión de numero entero (Entero = *False*) y tampoco que haga la revisión para repetir el mismo valor en la fila (RepetirValor = *False*).

Después, se declaran tres objetos de la clase “Validación” (ObjetoText1, ObjetoText 2 y ObjetoText 3) para cada control *TextBox* que se encuentre en la ventana, pasando como parámetros: el nombre del control *Textbox* (según corresponda) y entre comillas los caracteres validos para ese control (0.123456789); en este caso se trata sólo de caracteres numéricos.

De esta manera quedan relacionados los controles con la clase “Validación”. Cada vez que se introduzca valores en los controles mencionados los eventos de dichos controles se activarán y ejecutarán como se describió anteriormente.

```
Dim ObjetoData As ClaseValidacion 'Objeto para la validación del DataGrid

Private Sub FrmEspectros_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
……
…….

If ObjetoData Is Nothing Then
    ObjetoData = New ClaseValidacion(Me.DGEspectros, False, False)

   Dim ObjetoText1 As New ClaseValidacion(Me.TbFinal, "0.123456789")
   Dim ObjetoText2 As New ClaseValidacion(Me.TbInicio, "0.123456789")
   Dim ObjetoText3 As New ClaseValidacion(Me.TbIntervalo, "0.123456789")
End If
……
……
…….
End Sub
```

Código fuente 5.50 Llamado de la Clase Validación desde la ventana Espectros

Los puntos (…….) dentro del código fuente 5.50 indican que hay más instrucciones propias de cada ventana pero no es necesario mostrarlas para efectos de este ejemplo.

5.4.5. Controles Try Cath

Para la validación de los datos en el control *RichTextBox* de la ventana “AbrirArchivo” se implementó otro tipo de revisión. Ésta se hizo con la ayuda del bloque de código *Try Cath*, conocido también como controlador estructurado de errores y se utiliza (en el programa) para detectar errores al momento de leer el archivo de datos.

El código fuente 5.51 muestra las instrucciones para leer un archivo de texto; dentro del bucle *for* se encuentra el bloque de código *Try Cath*. Primero, el programa lee cada valor del archivo de texto con la instrucción *Input* ( ), en el caso de que la lectura de datos llegue a su máximo valor (12000 datos por columna) el programa por medio de la instrucción *Catch When Err.Number = 9* muestra un *MsgBox* para advertir al usuario de que el archivo contiene más datos de los permitidos.

La instrucción *Err.Number = 9* corresponde a un error de desbordamiento de datos en el vector donde se almacenan los valores, pero antes de que se produzca el error la instrucción *Cath When* captura el error y ejecuta las instrucciones que se muestran debajo de esa línea de código.

En este caso el programa muestra al usuario un *MsgBox* con dos botones (*Ok y Cancel*). Si la elección del usuario es *Ok*, el programa acepta la lectura de datos capturada hasta ese momento y finaliza de inmediato la lectura del archivo; en el caso de seleccionar *Cance*l, el programa rechaza los datos capturados y detiene también la lectura del archivo.

```
Do Until EOF(1)                          'Lee hasta el final del archivo
For i = 1 To NUDColumnas.Value           'Lee valores por columnas
    Me.Longitudes(i) = j                 'Determina la longitud de los vectores

    Try                                  'Prueba si hay datos disponibles
        Input(1, LecturaValores(i)(j)) 'Los guarda en un vector
    Catch When Err.Number = 9            'Cuando el indice está fuera del intervalo de puntos
                                         'Envia mensaje de texto
        If MsgBox("La señal tiene más de 12000 datos " & vbCrLf & "Desea cargar los datos",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, version) = MsgBoxResult.Cancel Then
           Me.ArchivoCorrecto = False  ' El usuario selecciona el Boton Cancelar
        Else
           Me.ArchivoCorrecto = True   ' El usuario selecciona el Boton Aceptar
        End If

        Me.Longitudes(i) = j - 1         'Se obtiene el valor la longitud de cada vector
        FileClose(1)                     'Cierra el archivo de texto
        Exit Sub                         'Sale completamente de la función
……
…….
    End Try                              'Termina la función Try Catch
Next                                     'Siguiente columna de datos
j += 1                                   'Siguiente Línea del archivo de texto
Loop                                     'Finaliza la lectura del archivo
FileClose(1)                             'Cierra el archivo
```

Código fuente 5.51 Bloques de código Try Catch

El uso del controlador de errores es muy útil para descubrir y reconocer errores durante la ejecución del programa, justo en el momento en que ocurren; además suprime mensajes de error no deseado, como el mostrado en la figura 5.21 y ajusta las condiciones del programa de manera que la aplicación pueda retomar el control y seguir en ejecución.

CAPÍTULO 6

APLICACIÓN DEL PROGRAMA

6.1. INSTALACIÓN

Para instalar el programa, abra el CD que se encuentra adjunto a este trabajo; en él encontrara una carpeta de nombre “Paquetes de instalación” que a su vez contiene dos sub carpetas llamadas “Instalar” y “Pre Requisitos”.

Si su computadora no tiene instalada la biblioteca Net Framework 3.5, el archivo contenido en la carpeta de “Pre Requisitos” le ayudara a descárgalo gratuitamente del sitio oficial de Microsoft, o puede acceder directamente a dicho sitio para descargarlo. Es importante que al momento de ejecutar el archivo cuente con una conexión a Internet, de lo contrario no podrá descargar la biblioteca mencionada.

El tiempo de instalación del NetFramework 3.5 dependerá de la velocidad de su conexión a Internet.

Una vez instalado el NetFramework 3.5, abra la sub carpeta llamada “Instalar” que se encuentra en el CD; en ella encontrará varios archivos de los cuales debe ejecutar “SetUp.Exe” señalado en la figura 6.1.

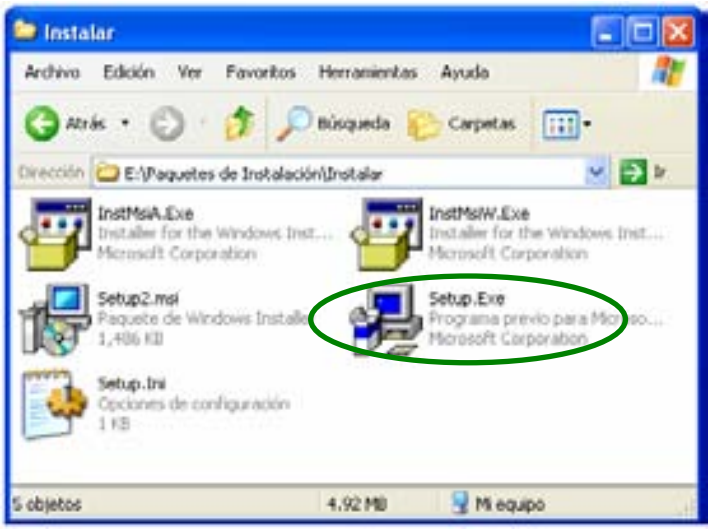
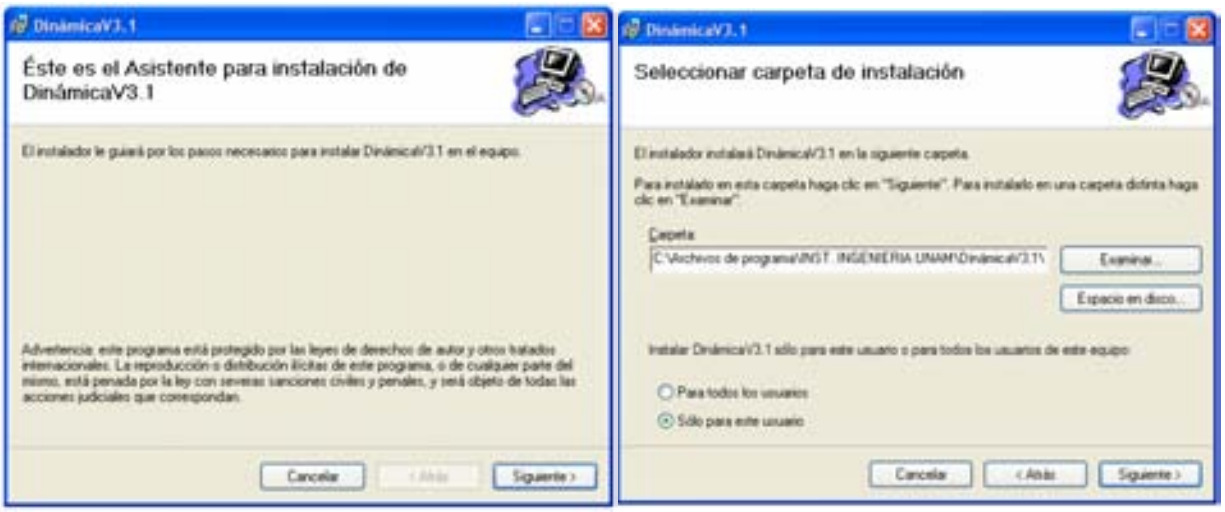


Figura 6.1 Archivos contenidos en la carpeta “Instalar”

Al hacer doble clic sobre el archivo “SetUp.Exe”, se abrirá el asistente para la instalación de “Dinámica V3.1” (figura 6.2a); en esta primer ventana el usuario decidirá si desea instalar o no el programa bajo los términos que se muestran en la ventana. Si decide dar clic en el botón “Cancelar” en esta ventana o en cualquier otra durante el proceso, la instalación terminará sin instalar el programa.

Al dar clic en el botón “Siguiente” (figura 6.2a) aparecerá la ventana que se muestra en la figura 6.2b; en la cual se debe especificar la carpeta dónde se almacenarán los archivos que necesita el programa para su ejecución y además se debe indicar para qué usuarios estará disponible el programa.

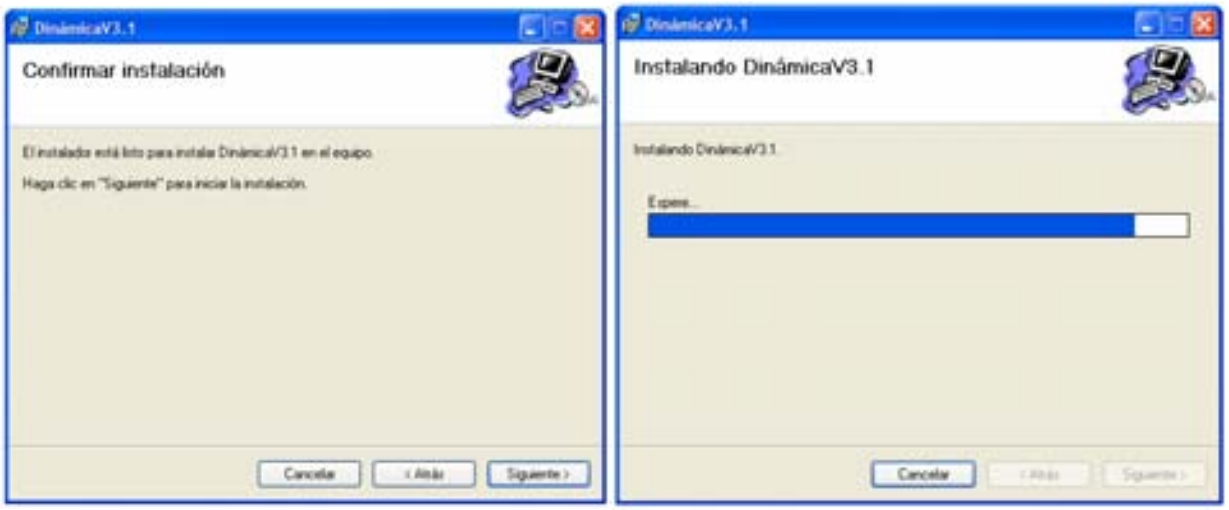


(a) **(b)**

Figura 6.2 Ventanas para instalar el programa

Al hace clic en el botón siguiente de la ventana (figura 6.2b), aparecerá la ventana (figura 6.3a) para confirmar la instalación del programa. Sí se decide cambiar alguno de los parámetros mencionados en la figura 6.2b sólo se debe hacer clic en el botón “Atrás” y cambiar lo que sea necesario; en caso de estar seguro oprima el botón “Siguiente”.

Inmediatamente el asistente comenzará a instalar el programa y se mostrará en pantalla una ventana (figura 6.3b) que indicará el avance de la instalación; al terminar, el asistente le mostrará la ventana de terminación de instalación (figura 6.4a).



(a) **(b)**

Figura 6.3 Ventanas para instalar el programa

(a) **(b)**

Figura 6.4 Ventanas a) terminación de instalación y b) archivos copiados

Para terminar el proceso de instalación, sólo haga clic en el botón “Cerrar” de la ventana mostrada en la figura 6.4a.

Con esto, el programa “Dinámica V3.1” quedará instalado en la computadora. Puede observar que en su escritorio se agregó un nuevo acceso directo correspondiente al programa así como también en el menú inicio. Para iniciar el programa sólo haga clic en cualquiera de ellos.

Si el usuario lo desea, puede ver la carpeta dónde quedó instalado el programa, (en la ubicación especificada en la figura 6.2b); esta carpeta contendrá una carpeta llamada “Dinámica V3.1” la cual mostrará varios archivos tal como muestra la figura 6.4b; dichos archivos son necesarios para el funcionamiento del programa y es indispensable (como cualquier otro programa) que no borre ninguno de ellos.

6.1.1. Errores en la instalación

Si por alguna razón el proceso de instalación no se puede llevar a cabo o es interrumpido en algún momento, puede deberse a alguna de las siguientes razones:

- ✓ El archivo de instalación está dañado, en tal caso debe conseguir otro CD en cualquiera de las demás copias del presente trabajo.

- ✓ Su computadora no cuenta con los requisitos mínimos que se especificaron en el capítulo 4, en este caso el programa no puede instalarse en su computadora.

6.1.2. Desinstalar

Para desinstalar el programa, debe entrar al “Panel de control” en el menú inicio, hacer clic en “Agregar o quitar programas” y en el renglón donde aparezca “Dinámica V3.1” oprimir el botón “Quitar”. De esta manera todos los archivos y componentes del programa que se hayan copiado a su computadora serán borrados.

6.2. MODO DE USO

Al abrir el programa (con cualquiera de los accesos directos mencionados en la sección anterior) aparecerá la pantalla de bienvenida por sólo algunos segundos, después se mostrará la ventana principal del programa.

Maximice la ventana principal al tamaño de la pantalla para tener un área de trabajo mayor. Después, seleccione una opción del menú desplegable según desee.

La descripción de las dos primeras opciones del menú se describen a continuación. La última opción "Salir", cierra por completo la ejecución del programa.

6.2.1. Ventana Osciladores de 1GL

Para este caso, se explicará el modo de uso del programa con la reproducción de un ejemplo tomado del libro "Dynamics of structures" (Chopra 2001), pág 205.

El ejemplo muestra tres sistemas de 1GL sometidos al movimiento sísmico (El centro) en su base. En la primera parte, se coloca a los osciladores con diferentes periodos de vibrar (T=0.5s, 1s y 2 s) e igual amortiguamiento (ξ=2%). En la segunda parte, se establece el mismo periodo (T=2 s) para cada oscilador y se varían los valores del amortiguamiento (ξ=0%, 2% y 5%). El objetivo es ver cómo varia la respuesta de desplazamiento de los osciladores en ambos casos.

Una vez determinada la respuesta dinámica, el ejemplo muestra como obtener las fuerzas internas por medio de la fuerza estática equivalente *f*s. Para ello obtiene las gráficas de la seudo aceleración para los osciladores con un mismo amortiguamiento y diferentes periodos de vibrar, especificados en la primera parte del problema.

Por último, en el ejemplo se obtiene el espectro de desplazamiento con un rango de periodos (T) de 0 a 3s, y mismo amortiguamiento ξ=2%.

A continuación se describe el procedimiento a seguir, adicionalmente a lo descrito en el ejemplo se indicará cómo mostrar las animaciones de los osciladores en cada caso así como la animación del espectro de respuesta.

Paso 1: Abrir ventana de osciladores de 1GL

Seleccione del menú desplegable la opción de "Osciladores de 1GL" para abrir la ventana de dicho tema; en ella aparecen el panel de controles del lado derecho de la pantalla y la gráfica de carga vacía (sin señal de excitación); todos estos controles están deshabilitados por el momento.

Obsérvese que en la barra de menús se agregaron tres opciones más: "Tipo de carga", "Ventana" y "Ayuda".

Paso 2: Ajustar el tamaño de las gráficas

Ajuste el tamaño de las gráficas con el menú "Ayuda\Pantalla\Ajustar" (figura 6.5); verá cómo la gráfica de excitación aumenta su ancho abarcando toda el área de trabajo. En caso de que esto no ocurra verifique que su pantalla cumple con las dimensiones mínimas especificadas (1024 x 768 píxeles).

Es importante que ajuste el tamaño de las gráficas antes de aplicar un tipo de carga, ya que después de esto no será posible ajustarlas, a menos que cierre la ventana y la vuelva a abrir. Cada vez que abra esta ventana deberá (si así lo desea) ajustar el tamaño de las gráficas.

Figura 6.5 Opciones del menú para ajustar el tamaño de las gráficas

Si desea retomar el tamaño original de las gráficas dentro de la pantalla lo puede hacer con el menú "Ayuda\Pantalla\Tamaño Original" (figura 6.5), antes de aplicar un tipo de carga.

Paso 3: Aplicar el tipo de carga

De manera predeterminada el programa cuenta con varios registros sísmicos en su base de datos incluyendo el "Centro". No obstante, si el usuario necesita cargar un registro en particular lo puede hacer mediante la opción del menú "Tipo de carga\Arbitraria".

Para cargar el registro del movimiento sísmico "El centro" selecciónelo haciendo clic en el menú "Tipo de Carga", como se muestra en la figura 6.6, después haga clic en cualquier gráfica.

Figura 6.6 Selección de un registro de la base de datos

Al aplicar el tipo de carga verá cómo el panel de controles se habilita, a excepción del grupo de controles "Animación de los Espectros" (más adelante se explica por qué) y se dibujan en el área de trabajo las gráficas de respuesta y debajo de ellas los osciladores de 1GL.

Los dibujos de las gráficas corresponden a las propiedades de los osciladores establecidos en el "Control de osciladores" y al tipo de respuesta definidos en el "Control de gráficas".

Paso 4: Establecer las propiedades de los osciladores

En la tabla que se encuentra en el "Control de osciladores" están contenidas las propiedades los osciladores; escriba los valores 0.5, 1 y 2 en la columna de los periodos T. Asimismo los valores del amortiguamiento 0.02, y mantenga las condiciones iniciales con valor igual a cero, tal como se muestra en la figura 6.7.

Figura 6.7 Propiedades de los osciladores

Al modificar los valores de las propiedades, automáticamente se calcula la respuesta dinámica y se dibujan las gráficas de respuesta según correspondan en el "Control de Gráficas".

Paso 5: Seleccionar el tipo de gráfica de respuesta

Como el objetivo del ejemplo es ver las diferentes respuestas de desplazamiento para cada uno de los osciladores, seleccione en el "Control de gráficas" el tipo "Desplazamiento" para todos y coloque el número de oscilador al que corresponde cada gráfica tal como se muestra en la figura 6.8.

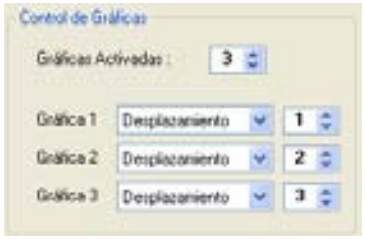


Figura 6.8 Tipo de gráficas de respuesta

Observe que al cambiar el tipo de gráfica y/o el oscilador al que pertenece, automáticamente se dibujan las gráficas de respuesta. Compare las gráficas de cada uno de los osciladores con las del ejemplo del libro (figura 6.9).

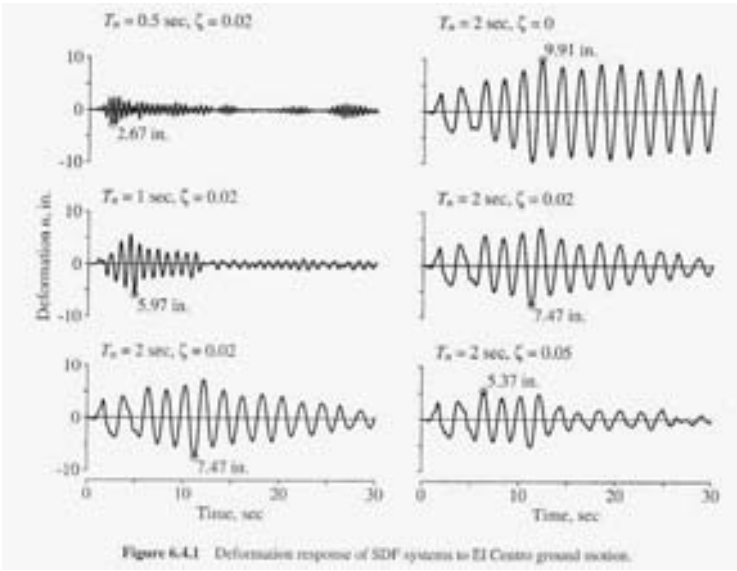


Figura 6.9 Gráficas de respuesta del ejemplo del libro (pág. 205)

Paso 6: Cambiar la apariencia de las gráficas

Para dar un mayor realce visual a cada gráfica de respuesta, haga clic derecho sobre cada una para mostrar el menú secundario y seleccione la opción “Apariencia”; al hacerlo aparecerá una ventana con título “Opciones Gráficas” (figura 6.10).

Haga clic en el pequeño cuadro de color y seleccione uno nuevo de la paleta colores, después haga clic en aceptar; posteriormente establezca el grosor de la línea en 2 unidades como se muestra en la figura 6.10. Repita los mismos pasos para cada gráfica de respuesta seleccionando un color diferente.

Figura 6.10 Ventana para cambiar la apariencia de las gráficas

El control numérico con la etiqueta de “Separación Sx” (figura 6.10) indica la equidistancia de las líneas verticales de la cuadrícula en el fondo de las gráficas; por default se establecen a cada segundo pero puede cambiar el valor si así lo requiere.

Paso 7: Activar las líneas de referencia

La escala numérica en la parte izquierda de cada gráfica muestra las amplitudes de cada una; sin embargo, para conocer cuál es la amplitud exacta en un determinado tiempo es necesario utilizar las líneas de referencia.

Para activarlas, haga clic derecho sobre cualquiera de las gráficas y seleccione la opción “Líneas de Referencia\Mostrar”; verá sobre cada gráfica una línea vertical (color azul) que se mueve con el puntero del Mouse cada vez que éste pasa encima de las gráficas.

También aparecerán en la parte izquierda de las gráficas (dentro de la cuadrícula) dos textos, uno indicando el valor de la amplitud y el otro el valor del tiempo en segundos, que corresponden a la posición de las líneas de referencia de cada gráfica.

Para mostrar en pantalla las amplitudes máximas de las respuestas, haga clic en el menú secundario de las gráficas en la opción “Líneas de Referencia\Mostrar Resp Máximas”; observe cómo las líneas de referencia se posicionan fijamente donde se encuentra la amplitud máxima (misma amplitud que en las gráficas del ejemplo).

Para mover de nuevo las líneas de referencia haga clic sobre alguna gráfica.

Paso 8: Gráficas a la misma escala

Para mostrar las gráficas de respuesta dibujadas a la misma escala, basta con seleccionar la opción “Misma Escala” del menú secundario de las gráficas. Con la ayuda de esta opción se ve más claramente que gráfica de respuesta tiene más amplitud que las demás.

Para que esta opción esté disponible es necesario que las gráficas que se muestren en pantalla sean del mismo tipo.

Paso 9: Animación de los osciladores y gráficas de respuesta

Para poner en marcha la animación de los osciladores en sincronía con las gráficas de respuesta sólo debe oprimir el botón "Play" en el bloque de controles "Animación de los osciladores"; inmediatamente comenzará la animación.

En cualquier momento puede pausar la animación o detenerla completamente con la ayuda de los botones "Pausa" y "Stop" respectivamente.

Al poner en marcha la animación, observe que las líneas de referencia desaparecen, esto por cuestiones de velocidad de reproducción. La velocidad dependerá de cada maquina y de los procesos que se ejecuten en ella al mismo tiempo que se coloque la animación. Si desea ver de nuevo las líneas de referencia ejecute el paso 7.

Ahora bien, existen otros controles dentro del bloque "Animación de los osciladores" para modificar la animación. Por ejemplo, el control *ScrollBar* ayuda a establecer hasta dónde llegará el máximo desplazamiento de los osciladores. Para ver su efecto en la animación coloque la barra en el extremo derecho como se muestra en la figura 6.11, y ejecute de nuevo la animación.

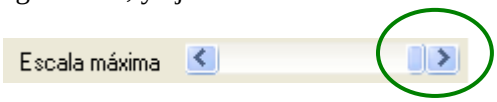


Figura 6.11 Barra para indicar el máximo desplazamiento

Notará que los desplazamientos de los osciladores son mayores; esto no quiere decir que el valor de su desplazamiento calculado se modifique, simplemente el dibujo de los osciladores es relativo al punto que se estableció como desplazamiento máximo.

Por otro lado, si coloca la barra en el extremo izquierdo y ejecuta la animación, verá que los osciladores se desplazan muy poco.

Una característica importante de la animación es que puede ser diezmada; por ejemplo, si establece el control numérico en 2 (figura 6.12) y ejecuta la animación de nuevo, verá que las gráficas de respuesta se dibujan con línea punteada, esto debido a que el programa hace la animación con 1 punto de cada N (en este caso N=2), lo que provoca que la animación sea más rápida que cuando N=1. El máximo valor de diezmado en el programa es N= 4.

Figura 6.12 Control para el diezmado de la animación

Repita el paso 4 para mostrar ahora las gráficas de respuesta de los osciladores cuando tienen el mismo periodo de vibrar (T=2 s) pero diferente amortiguamiento (ξ=0%, 2% y 5%). Asimismo realice los pasos 7, 8 y 9.

Paso 10: Fuerzas internas

En el ejemplo del libro, no se calculan los valores de las fuerzas internas sino que sólo se plantea la ecuación para obtenerlas quedando en función de la masa y de la seudo aceleración. Las gráficas que se muestran en el ejemplo (figura 6.13) corresponden a la seudo aceleración de los osciladores con valores de periodo diferente (T=0.5s, 1s y 2 s) e igual amortiguamiento (ξ=2%).

El programa no obtiene gráficas de seudo aceleración y seudo velocidad, en cambio obtiene las respuestas de aceleración y velocidad. Aunque las gráficas no son iguales, las primeras son muy aproximadas a las segundas.

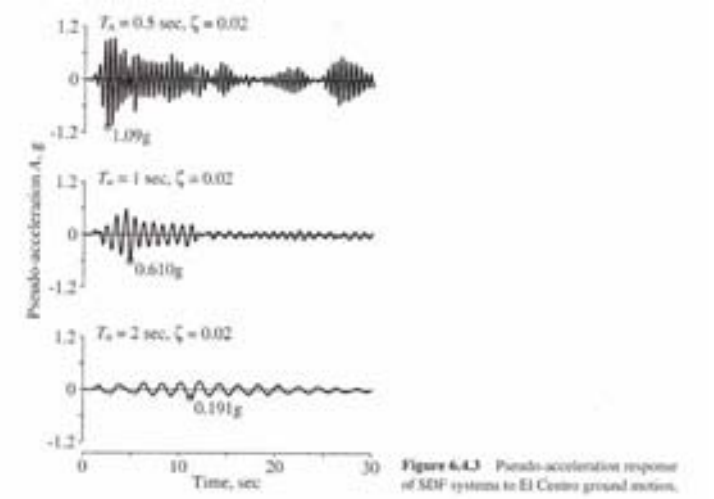


Figura 6.13 Gráficas de seudo aceleración del ejemplo del libro (pág. 207)

Para comparar las gráficas, cambie el tipo de carga a "Aceleración absoluta" en todos los osciladores y observe como las respuestas máximas son parecidas a las gráficas mostradas en el ejemplo (figura 6.13).

El programa tiene la característica de mostrar gráficamente los valores de las fuerzas internas en los osciladores, calculadas con la ecuación 4.1. Para ello ejecute las siguientes instrucciones:

- ✓ Active las líneas de referencia como se indicó en el paso 7
- ✓ Cambie el tipo de gráfica a "Fuerza cortante" en cada uno de los osciladores en el "Control de gráficas"
- ✓ Active la opción "V Basal" del menú secundario de las gráficas de respuesta
- ✓ Fije las líneas de referencia haciendo clic en cualquier punto de las gráficas de respuesta

Observe que en los dibujos de los osciladores aparecen escritos los valores de la masa, rigidez, fuerza interna y cortante basal, como se muestra en la figura 6.14.

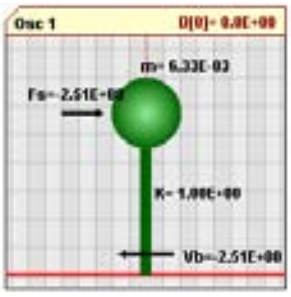


Figura 6.14 Fuerzas internas en el oscilador

Puede ver las fuerzas internas sobre los osciladores fijando las líneas de referencia en el punto de la gráfica de respuesta que desee. Para mover de nuevo las líneas de referencia haga clic sobre alguna gráfica.

El programa por default coloca el valor de la rigidez k=1, con este valor y el periodo del oscilador se calcula la masa (ecuación 4.4). Si desea cambiar el valor de la rigidez haga clic en la opción “Definir k” del menú secundario de las gráficas de respuesta y siga las indicaciones descritas en la sección 4.2.3.9.

El oscilador 1 (Osc 1) siempre mostrará la fuerza cortante de la gráfica 1, el Osc2 con la gráfica 2 y el Osc 3 con la gráfica 3.

Para borrar las fuerzas de los dibujos de los osciladores, desactive la opción “V Basal” del menú secundario de las gráficas o ponga en ejecución la animación.

Paso 11: Exportar archivo de datos

El programa no está diseñado para imprimir ningún tipo de archivo o gráfica directamente a la impresora; sin embargo, si tiene la capacidad de exportar los datos de la respuesta dinámica de cada oscilador a un archivo de texto.

Para ver los valores de la respuesta dinámica ejecute la opción “Datos de respuesta” del menú secundario de las gráficas de respuesta. Al hacerlo aparecerá una ventana que muestra los valores de respuesta de cada oscilador así como también la fuerza cortante. La descripción de esta ventana se encuentra en la sección 4.2.3.8.

Para exportar los valores a un archivo de texto, haga clic en el botón de la ventana “Respuesta numérica”; inmediatamente después se abrirá una ventana en la cual tiene que especificar el nombre del archivo y la dirección donde lo quiere guardar, al hacer clic en aceptar su archivo se guardará.

El archivo se guarda con extensión “.txt”, y se puede abrir con cualquier editor de texto. El archivo (figura 6.15) contendrá las respuestas de todos los osciladores definidos en el panel de controles, primero muestra un encabezado con información acerca del número de osciladores generados, fecha y hora de creación del archivo.

Después, la información correspondiente a cada oscilador: número de oscilador, T, k, m, ξ y condiciones iniciales; debajo de esto los valores de las ocho constantes correspondientes al método para obtener la respuesta dinámica y por último la respuesta numérica del oscilador por columnas.

Encabezado

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                    D I N Á M I C A                                            *
* Archivo: Respuestas de Osciladores de 1GL                                     *
* Número de Osciladores: 3                                                      *
* Fecha: 09/01/2009                                                             *
* Hora: 10:41:08 a.m.                                                           *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

Propiedades del oscilador

```
Oscilador:                1
Periodo Natural        T  = 0.5 s
K definida             K  = 1
Masa calculada         M  = 0.006332577
Amortiguamiento        Xi = 0.02
Desplazamiento Inicial Xo = 0
Velocidad Inicial      Vo = 0
```

Valores de las 8 constantes

```
*******************************************
Constantes de desplazamiento
D1= 0.9686880665
D2= 0.0196909689
D3=-0.0001319952
D4=-0.0000662899

Constantes de velocidad
V1=-3.1094731732
V2= 0.9587903062
V3=-0.0097767119
V4=-0.0099142570
```

Respuesta del oscilador

Tiempo	Desplazamiento	Velocidad	Acel. Relativa	Acel. Absoluta	Fza cortante
0.020	0.0000E+00	0.0000E+00	-2.4323E+00	0.0000E+00	0.0000E+00
0.040	-4.1422E-04	-3.7714E-02	-1.3211E+00	8.4369E-02	-4.1422E-04
0.060	-1.3547E-03	-5.2403E-02	-1.4201E-01	2.4027E-01	-1.3547E-03
0.080	-2.5042E-03	-6.6151E-02	-1.2237E+00	4.2870E-01	-2.5042E-03
0.100	-4.1405E-03	-1.0081E-01	-2.2223E+00	7.0451E-01	-4.1405E-03
0.120	-6.6604E-03	-1.5400E-01	-3.0677E+00	1.1292E+00	-6.6604E-03
0.140	-1.0213E-02	-1.9409E-01	-9.2279E-01	1.7103E+00	-1.0213E-02
0.160	-1.4133E-02	-1.9068E-01	1.2580E+00	2.3277E+00	-1.4133E-02
0.180	-1.7554E-02	-1.4443E-01	3.3387E+00	2.8446E+00	-1.7554E-02

Figura 6.15 Contenido del archivo de texto

Paso 12: Generar espectros de respuesta

Una herramienta muy importante del programa es el cálculo y la animación de los espectros de respuesta.

El ejemplo de libro sólo muestra el espectro de desplazamiento para un mismo amortiguamiento (ξ=2%); no obstante, el programa calcula 4 espectros de respuesta (desplazamiento, velocidad, aceleración relativa y absoluta) por cada bloque definido en la ventana “Espectros de respuesta” (figura 6.16).

Figura 6.16 Ventana para generar espectros de respuesta

Para generar los espectros seleccione la opción “Habilitar Espectros” del menú secundario de las gráficas, con esto aparecerá la ventana “Espectros de Respuesta”; cambie los valores de los parámetros para que se vean como en la ventana de la figura 6.16. Con estos valores el programa calculará tres bloques con 4 espectros de respuesta cada uno.

Cada bloque tendrá diferentes amortiguamientos (ξ=0%, 2% y 5%) y para cada uno su rango de periodos será de 0.01 a 3 segundos con intervalos de periodo de 0.01s entre cada punto del espectro.

Al hacer clic en aceptar (figura 6.16), aparecerá una ventana que le indicará cuántos bloques de espectros va a calcular el programa (figura 6.17a); haga clic en aceptar para comenzar a calcular los espectros al mismo tiempo observe que las gráficas de respuesta se colocan en animación. Durante el proceso de cálculo aparecerá la ventana que indica el avance de los cálculos (figura 6.17b).

(a) **(b)**

Figura 6.17 Ventanas auxiliares

Al termino de los cálculos se dibujarán en la parte baja de la pantalla los espectros de respuesta sustituyendo a los dibujos de los osciladores. Con ayuda del “Control de gráficas” en el panel de controles seleccione el tipo de espectro que desee ver en pantalla.

Para comparar los espectros de respuesta coloque el “Control de gráficas” como se muestra en la figura 6.8; al hacerlo verá en la parte baja de la pantalla las siguientes gráficas:

Figura 6.18 Espectros de respuesta

Las gráficas (figura 6.18) corresponden a los espectros de respuesta de desplazamiento con diferente amortiguamiento (ξ=0%, 2% y 5% respectivamente).

Los espectros tienen dos líneas de referencia (horizontal y vertical) de color azul que se mueven a lo largo de la gráfica del espectro cuando el puntero del Mouse pasa sobre ellas. Las características de estas gráficas se explicaron en la sección 4.2.3.11.

La gráfica de en medio es la que corresponde al ejemplo del libro (figura 6.19), puede comparar los valores posicionando el puntero del Mouse en el valor del periodo que desee.

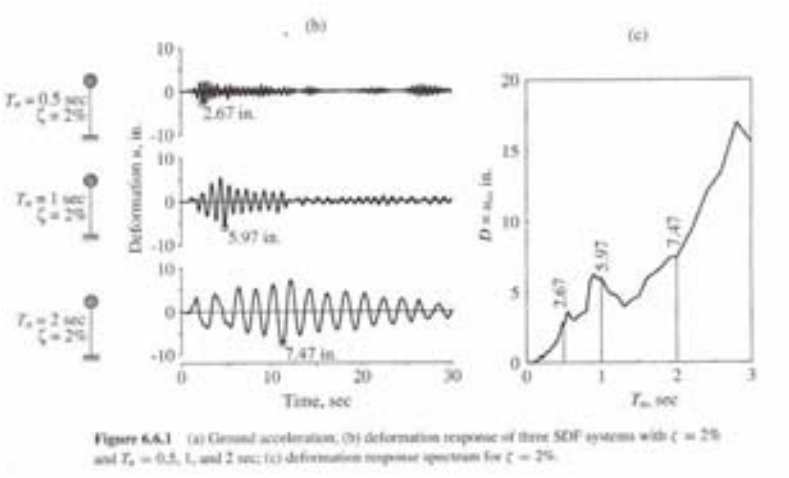


Figura 6.19 Espectro de respuesta del ejemplo del libro (pág 209)

Paso 13: Menú secundario de los espectros de respuesta

Al igual que las gráficas de respuesta los espectros tienen un menú secundario que se activa haciendo botón derecho sobre ellos. Una de las opciones de este menú es “Misma Escala” que permite dibujar los espectros con la misma escala (siempre y cuando los espectros sean del mismo tipo).

También, puede ver y exportar en un archivo de texto la respuesta numérica de cada espectro; para esto ejecute la opción “Datos de respuesta” y siga las instrucciones mencionadas en el paso 11.

Paso 14: Bloqueo de algunos controles

Al generar los espectros de respuesta se bloquea el control “Animación de osciladores”; esto debido a que el dibujo de los osciladores fue sustituido por los espectros y por lo tanto no puede haber animación de osciladores. También se bloquea el menú tipo de carga debido a que el cálculo de los espectros depende directamente de la carga seleccionada y no se puede cambiar mientras los espectros estén habilitados. Para seleccionar otro tipo de carga primero hay que deshabilitar los espectros (paso 16).

Observe que en el “Control de osciladores” sólo se permite cambiar los valores correspondientes a la columna del periodo “T” debido a que los espectros fueron calculados con un valor de amortiguamiento y condiciones iniciales fijos.

Paso 15: Animación de los espectros de respuesta

Para comenzar la animación de los espectros en sincronía con las gráficas de respuesta sólo debe oprimir el botón “Play” en el bloque de controles “Animación de los espectros”.

En cualquier momento puede pausar la animación o detenerla completamente con la ayuda de los botones “Pausa” y “Stop” respectivamente.

Al poner en marcha la animación las líneas de referencia en las gráficas de respuesta desaparecen (por razones mencionadas en el paso 9) mientras que las líneas de referencia que se encuentran en los espectros estarán siempre visibles.

Para controlar la velocidad de animación de los espectros se colocó un control *ScrollBar*, que se muestra en la figura 6.20. Cuando la barra esté ubicada en el extremo izquierdo la velocidad de la animación será la más alta y entre más a la derecha se coloque más lenta será.

Figura 6.20 Controles para animación de espectros

Paso 16: Deshabilitar espectros de respuesta

Para deshabilitar los espectros, ejecute del menú secundario de las gráficas de respuesta "Deshabilitar Espectros"; con esto se borrarán los espectros y se dibujarán de nuevo los osciladores.

6.2.2. Ventana Osciladores de VGL

Para este caso, se explicará el modo de uso del programa con la reproducción de un ejemplo tomado del libro "Dynamics of structures" (Chopra, 2001), pág 525.

El ejemplo muestra un oscilador de 5GL sometido a un movimiento sísmico (El centro) en su base. Las propiedades del oscilador para cada piso son: m=100 kips/g (0.25892 kips in/s^2), k= 31.54 kips, h=12 ft y amortiguamiento modal de ξ=5%. Los periodos naturales de vibrar (T) calculados son: 2.0s, 0.6852s, 0.4346s, 0.3383s y 0.2966s.

El objetivo del ejemplo (en el libro) es mostrar cómo contribuyen las respuestas modales a la respuesta completa de cada grado de libertad. De igual forma se obtienen las fuerzas cortantes para el primer y último nivel del oscilador.

A continuación se describe el procedimiento a seguir, adicionalmente a esto se mostrará cómo ejecutar la animación del oscilador y obtener los espectros de piso.

Paso 1: Abrir la ventana de Osciladores de VGL

Seleccione del menú desplegable de la ventana principal la opción de "Osciladores de VGL" para abrir la ventana de dicho tema; en la cual aparecen una barra de herramientas, el panel de controles del lado derecho de la pantalla, cinco gráficas vacías (una de carga y 4 de respuestas) y un oscilador de varios grados de libertad.

En el panel de controles se encuentran habilitados el "Control de propiedades" y "Modos de vibrar", y por el contrario aparecen bloqueados el "Control de Gráficas" y "Animación del Oscilador", debido a que en este momento aún no se ha calculado la respuesta del oscilador.

Paso 2: Ajustar el tamaño de las gráficas

Ejecute el paso 2 del ejemplo mostrado para la ventana de osciladores de 1GL (sección 6.2.1).

Paso 3: Establecer las propiedades del oscilador

En el "Control de propiedades" establezca los grados de libertad GL=5 al igual que el número de modos a superponer (Nms=5). Después, en la tabla que se muestra (figura 6.21) escriba los valores correspondientes a la masa, rigidez y amortiguamiento que se definieron al principio del ejemplo.

Coloque el vector de forma en J=1 y las condiciones iniciales igual a cero, tal como se muestra en la figura 6.21.

Figura 6.21 Propiedades del oscilador de VGL

Paso 4: Numeración de los grados de libertad

Defina el sentido de numeración de los grados de libertad de "Abajo a arriba" oprimiendo el botón  , que se encuentra en la barra de herramientas. Al hacerlo el sentido de la flecha cambiará hacia arriba.

El ejemplo del libro enumera los grados de libertad de abajo a arriba. En realidad esto no tiene mayor importancia pero es necesario hacerlo para que al momento de comparar las gráficas los grados de libertad coincidan.

Paso 5: Mostrar deformadas modales

Para ver las formas modales del oscilador, active la casilla "Dibuja deformada" en el control "Modos de vibrar" ubicado en la parte baja del panel de controles. Después, active la segunda casilla "Mostrar valores" para ver en pantalla los valores de los modos de vibrar. Para ver cada modo de vibrar oprima el control numérico que se encuentra al lado derecho de la etiqueta: "Modo =".

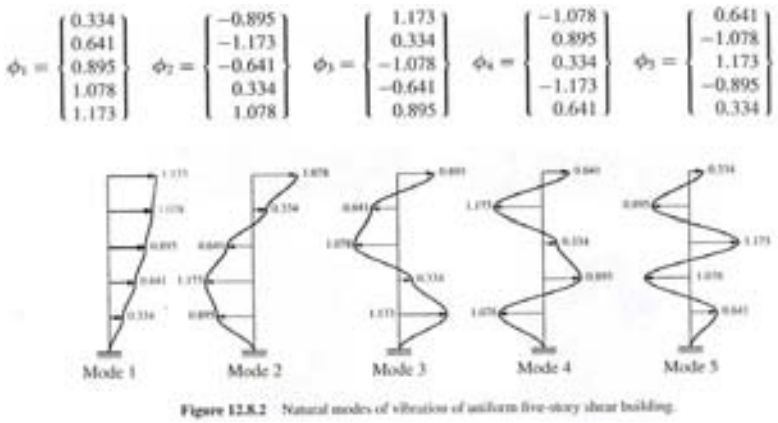


Figura 6.22 Formas modales del ejemplo del libro (pág. 484)

Compare las formas modales del programa con las del ejemplo (figura 6.22); observará que los valores no coinciden, esto se debe a que el programa normaliza los modos vibrar colocando igual a “1” el primer grado de libertad del oscilador en cada modo.

Para comprobar que los valores de los modos son iguales, la tabla 6.1 muestra como normalizar los valores del modo 1 del libro. Compare de nuevo los valores y observe que si coinciden.

Tabla 6.1 Modo de vibrar 1

No Normalizados (libro)	Operación	Normalizados (programa)
0.334	/ 0.334 =	1.000
0.641	/ 0.334 =	1.919
0.895	/ 0.334 =	2.680
1.078	/ 0.334 =	3.228
1.173	/ 0.334 =	3.512

Paso 6: Periodos y frecuencias

Para mostrar los periodos y las frecuencias oprima el botón  (ubicado en la barra de herramientas), al hacerlo se abrirá la ventana “Tabla de propiedades” (figura 6.23) descrita en la sección 4.2.4.13; ésta contiene una cuadrícula de datos que muestra las propiedades del oscilador que estén seleccionadas en el control *ComboBox*.

Cambie la opción del control a “Periodos” para ver los periodos y frecuencias del oscilador en la tabla. Observe que los periodos calculados en el programa son iguales a los del ejemplo.

Figura 6.23 Ventana de propiedades mostrando los periodos y frecuencias de vibrar

Las matrices y vectores que se muestran en la ventana “Tabla de propiedades” no se pueden modificar directamente de la tabla, sólo se puede copiar los datos (al portapapeles) seleccionándolos y oprimiendo la teclas “CTRL+C”.

Para exportar las tablas de propiedades a un archivo de texto, haga clic en el botón  de la ventana mencionada; inmediatamente después se abrirá una ventana en la cual tiene que especificar el nombre del archivo y la dirección donde lo quiere guardar, al hacer clic en aceptar su archivo se guardará con extensión “.txt”, y se puede abrir con cualquier editor de texto.

Paso 7: Aplicar el tipo de carga y calculo de la respuesta

Para aplicar la carga al oscilador ejecute el paso 3 del ejemplo mostrado para la ventana de osciladores de 1GL (sección 6.2.1). Observe que al cargar la señal la respuesta del oscilador no se calcula automáticamente, para ello haga clic en el botón  (de la barra de herramientas) e inmediatamente el programa calculará la respuesta dinámica.

Al calcular la respuesta el "Control de propiedades" se bloqueará en el panel de controles; para modificar algún valor de la tabla deberá quitar los cálculos realizados oprimiendo el botón  .

A diferencia de la ventana de osciladores de 1GL, aquí no se calcula la respuesta automáticamente al seleccionar un tipo de carga debido a que el tiempo de calculo puede ser mayor dependiendo del número de grados de libertad del oscilador y el número de puntos que conformen a la señal de carga.

Paso 8: Seleccionar el tipo de gráficas de respuestas

Para ver las diferentes respuestas de desplazamiento para cada grado de libertad del oscilador, seleccione en el "Control de gráficas" el tipo "Desplazamiento" para todos y coloque el número de grado de libertad como se muestra en la figura 6.24.

Figura 6.24 Tipo de gráficas de respuesta

Observe que al cambiar el tipo de gráfica y/o el grado de libertad al que pertenece, automáticamente se dibujan las gráficas de respuesta.

Para cambiar la apariencia de las gráficas, activar las líneas de referencia y dibujar las gráficas a la misma escala, ejecute los pasos 6, 7 y 8 del ejemplo mostrado en la ventana de osciladores de 1GL (sección 6.2.1).

Paso 9: Animación del oscilador y gráficas de respuesta

Para ver la animación del oscilador ejecute el paso 9 del ejemplo anterior (sección 6.2.1) ya que los controles para la animación del oscilador de VGL tienen el mismo comportamiento. Si activa la casilla "Mostrar despl" (figura 6.25), al ejecutar la animación verá el desplazamiento de cada grado de libertad del oscilador en cada instante de tiempo.

Figura 6.25 Controles para la animación del oscilador

Paso 10: Respuesta modal

El programa puede mostrar la respuesta de cada grado de libertad en forma modal incluyendo la fuerza cortante; para esto active la casilla "Mostar Respuestas de modos" en el "Control de gráficas", seleccione el tipo de gráfica en "Fuerza Cortante" del grado de libertad 1 y los modos 1, 2 y 3 tal como se muestra en la figura 6.26.

Al activar la casilla verá que los controles *ComboBox* de las gráficas 2, 3 y 4 se bloquean (figura 6.26); además, las gráficas (2 ,3 y 4) que se dibujen corresponderán a la respuesta por modos del tipo de gráfica que esté dibujada en la gráfica 1.

Figura 6.26 Tipo de gráficas de respuesta modal

Compare las gráficas por modos de los grados de libertad 1 y 5 que obtiene el programa con las del ejemplo del libro mostradas en la figura 6.27, las del lado izquierdo corresponden a la base (GL=1) y las del lado derecho al GL=5.

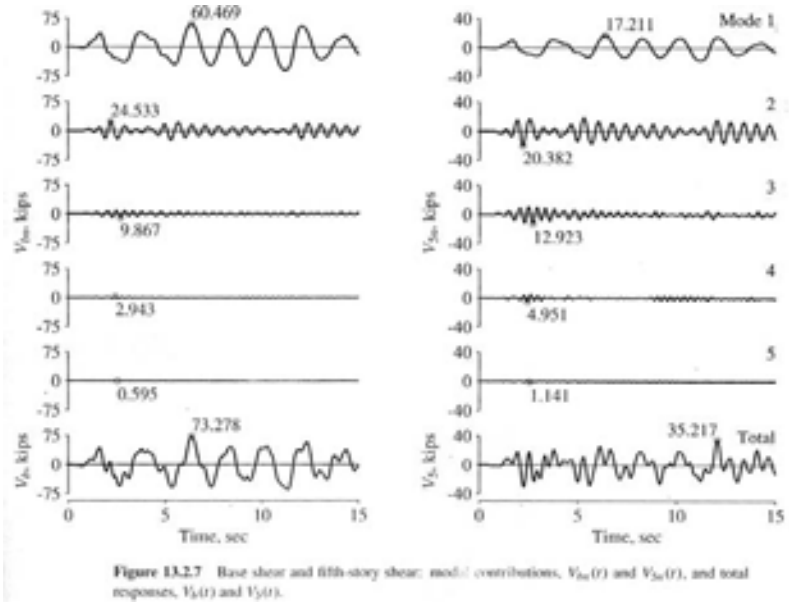


Figura 6.27 Formas modales del ejemplo del libro (pág. 529)

Paso 11: Fuerzas internas

Para ver los valores de las fuerzas internas en los grados de libertad del oscilador ejecute las siguientes instrucciones:

- ✓ Active las líneas de referencia como se indicó en el paso 8
- ✓ Active la opción "Fuerza Cortante" del menú secundario de las gráficas de respuesta
- ✓ Fije las líneas de referencia haciendo clic en cualquier punto de las gráficas de respuesta

Observe que en el dibujo del oscilador aparecen escritos los valores correspondientes a la fuerza cortante total en cada grado de libertad.

Para borrar las fuerzas de los dibujos de los osciladores, desactive la opción "Fuerza Cortante" del menú secundario de las gráficas o ponga en ejecución la animación.

Paso 12: Generar espectros de piso

Para generar los espectros seleccione la opción "Espectros de piso" del menú secundario de las gráficas; con esto aparecerá la ventana "Espectros de Respuesta". Los parámetros que se necesita para generar los espectros son los mismos que los descritos en el paso 12 del ejemplo de la sección 6.2.1.

En el caso de los osciladores de VGL, sólo se permite especificar un bloque de espectros. Coloque los parámetros ξ=5%, rango de periodos será de 0.01 a 1 segundo con intervalos de periodo de 0.01s y haga clic en aceptar.

En este caso, no hay animación de las señales de respuesta como en la ventana de osciladores de 1GL. Al termino del cálculo de los espectros, haga clic sobre la masa del oscilador para ver el espectro de piso correspondiente; éste aparecerá al lado del dibujo de la masa y desaparecerá al hacer clic en cualquier otro lado de la ventana que no sea el espectro.

6.3. COMPARACIÓN DE LA RESPUESTA DINÁMICA

La respuesta dinámica del programa, para osciladores de 1GL, se comparó con dos programas afines: DEGTRA (Ordaz y Montoya, 2002) y USEE (University Illinois, 2001); además, también se hizo una comparación con valores tomados de un ejemplo del "Dynamics and structures" (Chopra, 2001).

La respuesta dinámica se comparó mediante los espectros de respuesta de: desplazamiento, velocidad y Acel Absoluta, con un amortiguamiento ξ=5%, y un rango de periodos de 0.01s a 3.0s con intervalos de 0.01s, de un oscilador de 1GL sometido al movimiento sísmico de "El Centro".

La tabla 6.2 muestra los valores obtenidos para cada programa en tres periodos específicos (T= 0.5s, 1.0s y 2.0s); en el caso del ejemplo tomado del libro no se tiene la respuesta de velocidad y los valores de aceleración corresponden a la seudo aceleración (marcado con un * en la tabla 6.2). Asimismo, el programa USEE no proporciona la respuesta de velocidad.

Tabla 6.2 Respuestas de los diferentes programas

Parámetros:	T=0.5s, ξ=2%			T=1.0s, ξ=2%			T=2.0s, ξ=2%		
Programa	Despl. (in)	Vel. (in/s)	A. Abs. (in/s^2)	Despl. (in)	Vel. (in/s)	A. Abs. (in/s^2)	Despl. (in)	Vel. (in/s)	A. Abs. (in/s^2)
Chopra	2.670	----	420.98*	5.970	----	235.59*	7.470	----	73.768*
Degtra	2.681	31.964	422.17	5.932	41.604	234.53	7.469	31.957	73.775
Usee	2.686	----	424.45	5.954	----	235.13	7.466	----	73.729
Dinámica	2.676	32.165	421.65	5.970	41.734	235.89	7.469	31.976	73.785

Obsérvese que los valores de las respuestas en la tabla 6.2 son muy parecidos, el rango de diferencias entre una respuesta y otra es menor al 1%; esto puede deberse al redondeo de los decimales y el tipo de método que usa cada programa para obtener la respuesta dinámica.

Las gráficas completas de los espectros, de los diferentes programas, se muestran en la figura 6.28. Se puede observar que las respuestas son similares y que las diferencias entre una y otra son relativamente pequeñas.

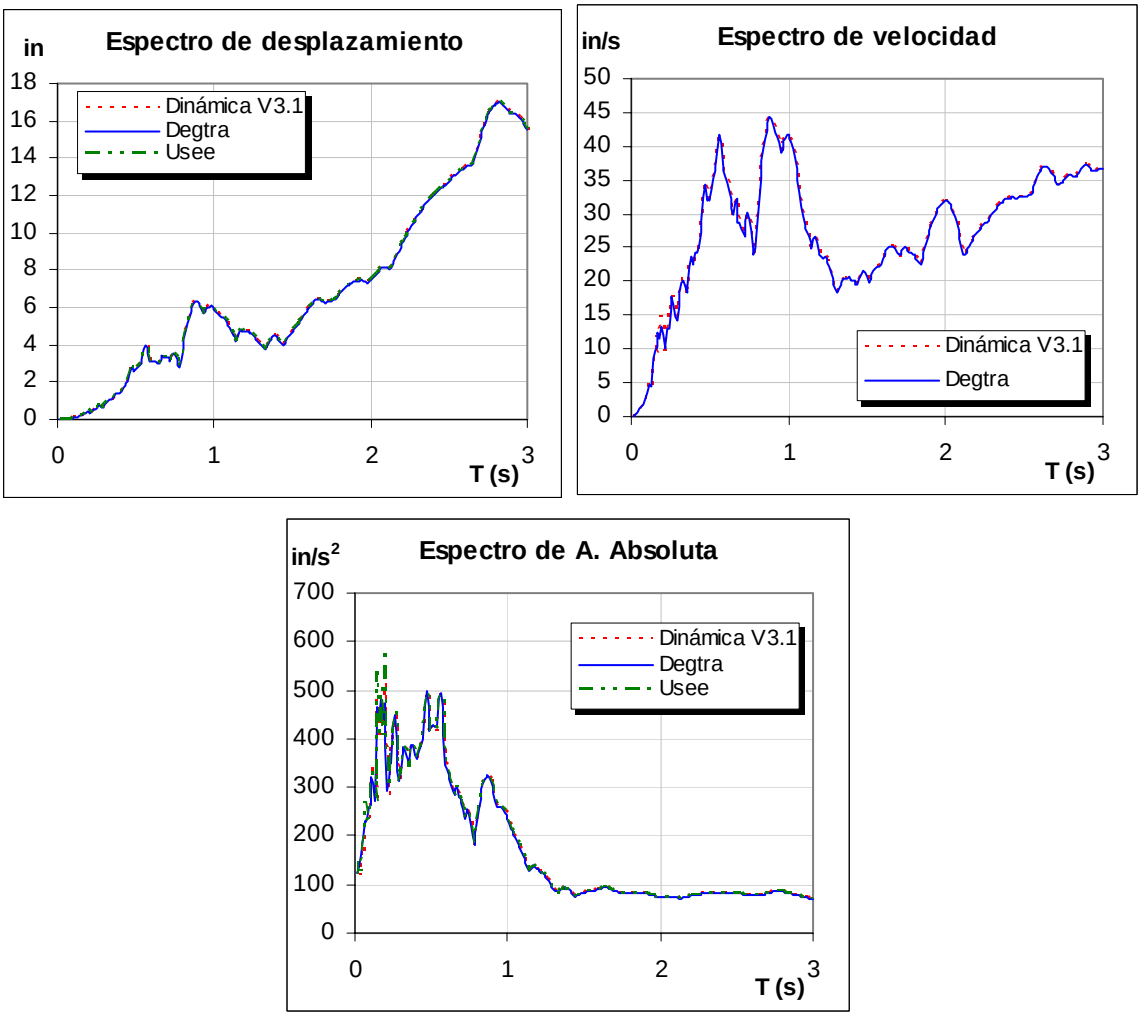


Figura 6.28 Gráficas de los diferentes espectros de respuesta

El oscilador de VGL que se tomó de ejemplo para comparar la respuesta es el mismo que se describe en la sección 6.2.2. En este caso sólo se compararon los modos de vibrar, factores de participación y periodos naturales de vibración, debido a que la respuesta del oscilador de VGL se obtiene superponiendo respuestas de osciladores de 1GL de diferentes periodos y alteradas por los modos de vibrar y los factores de participación correspondientes.

La comparación de los factores de participación (tabla 6.3) se hizo con los valores tomados del ejemplo del libro de Anil K. Chopra (sección 6.2.2) y con el programa USEE. Observe que los valores obtenidos por “Dinámica V3.1” son iguales a los de USEE y los del ejemplo del libro.

Tabla 6.3 Factores de participación

Factores de participación			
Grado	Chopra	Usee	Dinámica
GL=1	0.356	0.356	0.356
GL=2	0.301	0.301	0.301
GL=3	0.208	0.208	0.208
GL=4	0.106	0.106	0.106
GL=5	0.029	0.029	0.029

Tabla 6.4 Periodos de vibrar

Periodos T				
Grado	Chopra	Staad	Mehdi	Dinámica
GL=1	2.000	2.000	2.000	2.000
GL=2	0.685	0.685	0.685	0.685
GL=3	0.435	0.435	0.435	0.435
GL=4	0.338	0.338	0.338	0.338
GL=5	0.297	0.297	0.297	0.297

Los periodos de vibrar (tabla 6.4) se compararon con los programas Staad Pro 2007, el software del Dr. Medhi y los mostrados en el libro de Chopra. Observe que los valores para todos los programas y libro son los mismos.

Para la comparación de los modos de vibrar (tablas 6.5), se utilizaron los valores del libro de Chopra, y los programas Staad Pro 2007 y del Dr. Mehdi. En los tres casos se normalizaron los modos tomando como valor = 1 el primer grado de libertad para poder compararlos directamente con los obtenidos por el programa "Dinámica".

Los valores de los modos obtenidos del libro de Chopra, Staad Pro y Dinámica son prácticamente los mismos, con sólo diferencias de milésimas en algunos valores. No así, los resultados del Dr. Mehdi, que fuera del primer modo sus valores difieren en décimas con los demás programas.

Tablas 6.5 Modos de vibrar normalizados

Modos de vibrar (Chopra)					
Grado	Modo 1	Modo 2	Modo 3	Modo 4	Modo 5
GL=1	1.000	1.000	1.000	1.000	1.000
GL=2	1.919	1.311	0.285	-0.830	-1.682
GL=3	2.680	0.716	-0.919	-0.310	1.830
GL=4	3.228	-0.373	-0.546	1.088	-1.396
GL=5	3.512	-1.204	0.763	-0.595	0.521

Modos de vibrar (Staad pro)					
Grado	Modo 1	Modo 2	Modo 3	Modo 4	Modo 5
GL=1	1.000	1.000	1.000	1.000	1.000
GL=2	1.916	1.309	0.285	-0.831	-1.683
GL=3	2.681	0.715	-0.919	-0.310	1.832
GL=4	3.225	-0.373	-0.546	1.088	-1.399
GL=5	3.509	-1.203	0.764	-0.594	0.522

Modos de vibrar (Mehdi)					
Grado	Modo 1	Modo 2	Modo 3	Modo 4	Modo 5
GL=1	1.000	1.000	1.000	1.000	1.000
GL=2	1.919	1.320	0.305	-0.807	-1.656
GL=3	2.683	0.749	-0.887	-0.322	1.765
GL=4	3.229	-0.311	-0.542	1.054	-1.322
GL=5	3.513	-1.157	0.732	-0.553	0.460

Modos de vibrar (Dinámica)					
Grado	Modo 1	Modo 2	Modo 3	Modo 4	Modo 5
GL=1	1.000	1.000	1.000	1.000	1.000
GL=2	1.919	1.310	0.285	-0.831	-1.683
GL=3	2.683	0.715	-0.919	-0.310	1.831
GL=4	3.229	-0.373	-0.546	1.088	-1.398
GL=5	3.513	-1.204	0.764	-0.594	0.521

Con base en los resultados mostrados, se puede decir que la respuesta dinámica del programa es buena, y comprobable con otros programas afines.

6.4. EVALUACIÓN DEL SOFTWARE

El programa “Dinámica” en su versión 3.0 se puso a prueba en la clase de Dinámica Estructural del posgrado de ingeniería de la UNAM ciclo escolar 2009-1. Con el objetivo de depurar los errores y tomar en cuenta las consideraciones de los alumnos hacia el programa.

Con ayuda del software el profesor de la materia (Dr. Mario Ordaz) explicó en varias ocasiones conceptos de los tema involucrados. Además, el programa fue proporcionado a los alumnos de dicha clase para que pudieran manejarlo fuera de clase.

Al final del semestre en cuestión se aplicó un cuestionario a los alumnos de la clase para evaluar los aspectos de: aprendizaje, Dinámica e interfaz gráfica del programa. Los dos primeros se enfocaron a conocer cuál es la percepción del alumno hacia el programa en el salón de clase y el tercero sirvió para obtener la opinión de los alumnos hacia el programa en sí.

La escala de calificaciones es de 1 a 5 unidades, siendo este último el valor más alto. Los valores mostrados en las tablas 6.6 a 6.8 contienen las preguntas del cuestionario y los valores promediados obtenidos de evaluar a 14 alumnos del grupo.

Tabla 6.6 Cuestionario para el tema de Aprendizaje

Pregunta	Promedio
1. ¿Está de acuerdo con el uso del programa en clase?	4.21
2. ¿Comprendió mejor los conceptos del tema en clase con ayuda del programa?	4.07
3. ¿Las animaciones le ayudaron a visualizar la forma de vibración de los osciladores?	4.86
4. Tras ver en el programa las gráficas de la respuesta dinámica, ¿complemento sus conocimientos en el tema?	4.36
5. ¿La reproducción de ejemplos en clase le ayudo a comprender el resultado de las formulas obtenidas en clases previas?	4.50
6. ¿Comprendió lo que el programa mostraba cuando se explicaba algún concepto en clase?	4.57
7. ¿El programa despertó interés en usted por la dinámica?	4.36
8. ¿La aportación del programa es positiva en la manera de dar clase?	4.86
9. ¿Está de acuerdo con la introducción del software educativo en clase de dinámica?	4.36
10. ¿Recomendaría tomar la clase con el uso del programa?	4.36
11. ¿Considera que el programa es interactivo?	4.07
12. ¿Con que frecuencia le gustaría que el programa se usará en el salón de clase?	3.57
13. ¿Con que frecuencia utilizo el programa fuera de clase?	3.43
14. ¿El programa le sirvió de alguna manera a realizar sus tareas?	3.79
15. ¿Intento reproducir ejemplos mostrados en los libros?	2.71
16. ¿Antes de cursar esta clase había utilizado algún otro software educativo?	1.93

El cuestionario para evaluar el aprendizaje de los alumnos (tabla 6.6) contiene 16 preguntas para conocer cómo percibió el alumno el uso del programa en clase, si le sirvió y le ayudó a comprender mejor los conceptos de dinámica y para conocer con qué frecuencia se usó el programa fuera de clase.

Los resultados son muy favorables: en 11 preguntas se obtuvieron promedios mayores a 4.0 unidades. Las preguntas (#12, #13 y # 14 de la tabla 6.6) tuvieron calificaciones cercanas a 3.5 unidades, que indican la preferencia del alumno por usar el programa dentro del salón de clase como fuera de éste.

La calificación a la última pregunta del cuestionario indica que los alumnos han tomado pocas clases con software educativos, lo que indica el retraso de este tema en los salones de clase.

Para complementar las preguntas mostradas en la tabla 6.6, se aplicó el siguiente cuestionario (tabla 6.7), con el objetivo evaluar en forma general los temas de contenido del programa.

Tabla 6.7 Cuestionario para el tema de Dinámica

Pregunta	Promedio
1. ¿Le da buena información acerca de los temas?	3.71
2. ¿Considera adecuados los temas contenidos en el programa?	4.29
3. ¿Le facilita la comprensión del tema?	4.29
4. ¿Recomienda el uso del programa en niveles de licenciatura?	4.29
5. ¿Se entienden los temas que se manejan en el programa?	4.29

Con base en los resultados de la tabla 6.7, los alumnos consideran que la información de los temas involucrados es buena pero que aún hay mejoras por hacer. Las preguntas 2 a 5, muestran un resultado bastante aceptable teniendo calificaciones de 4.29 unidades, que indica que los alumnos consideran adecuados los temas que se manejan en el programa.

Para evaluar la interfaz gráfica del programa, se aplicó el cuestionario mostrado en la tabla 6.8. Los resultados también son bastante favorables, pues 14 de las 18 preguntas obtuvieron una evaluación de más de 4.0 unidades que indica que a los alumnos les agrada la interfaz gráfica del programa.

La pregunta # 13 tiene una evaluación de 2.07 la cual revela que es poco necesaria la inclusión de efectos de sonido en el programa.

Los errores encontrados en el programa fueron casi nulos, como lo demuestra la calificación de la pregunta #18; los pocos errores que se encontraron fueron depurados para dar paso a la siguiente versión del programa: “Dinámica v 3.1”.

Tabla 6.8 Cuestionario para el tema de Interfaz gráfica

Pregunta		Promedio
1. ¿La interfaz de usuario es amigable?		4.36
2. ¿Es adecuada la distribución de los controles en las ventanas?		4.29
¿Considera adecuado el uso general de….?	3. Ventanas	4.36
	4. Botones	4.43
	5. Colores	4.36

	6. Tipos de letra	4.43
	7. Gráficas	4.36
	8. Iconos	4.50
	9. Menús secundarios	3.86
10. ¿Está clara la función de cada control?		3.86
11. ¿El programa es de fácil manejo?		4.07
12. ¿Está de acuerdo con la distribución de las ventanas?		4.00
13. ¿Hacen falta efectos de sonido?		2.07
14. ¿Considera adecuadas las animaciones?		4.43
15. ¿El archivo de ayuda contiene la información necesaria para el uso de los controles?		4.14
16. ¿La aparición de mensajes de advertencia fueron útiles?		4.07
17. ¿Es posible reproducir ejemplos rápidamente en el programa?		4.43
18. ¿Encontró errores en el programa?		1.50

Además de los cuestionarios mostrados en las tablas 6.6 a 6.8, se aplicó el cuestionario de la tabla 6.9, para conocer los temas en específico que el alumno considera que comprendió más, y temas que piensan hacen falta en el programa.

El cuestionario fue de opción múltiple, y el alumno pudo seleccionar varias de las opciones para una misma pregunta; por esta razón la suma de los porcentajes de las respuestas en cada pregunta no es 100%. Los resultados que se muestran en cada pregunta son relativos a los 14 alumnos que se les aplicó el cuestionario.

Se puede observar que el “comportamiento de los osciladores” (71.4%) fue el tema que más impacto tuvo en los alumnos, seguido de “los espectros de respuesta” (57.1%) y “la respuesta dinámica en el tiempo” (42.9%).

Con la pregunta #3 de la tabla 6.9, se observa que a los alumnos les gustaría que el programa mostrará más temas de contenido de los que se manejan, señalando los temas de “torsión” y “sistemas elastoplásticos” como los más relevantes.

La última pregunta de la tabla 6.9, muestra interesantes resultados. Ninguno de los alumnos encuestados piensa que es mejor seguir tomando clases de la manera tradicional. Lo que habla de que el impacto del software educativo en la clase es favorable. El 78.6% de los alumnos considera que combinar las dos formas de dar clase (manera tradicional y con ayuda de Software educativo) es la mejor opción para recibir clase.

Tabla 6.9 Cuestionario para el tema de Dinámica

Pregunta	Porcentaje
1. ¿Qué tema comprendió más rápido o mejor que de la manera tradicional de aprender en clase?	
a. Comportamiento de los osciladores	71.4%
b. Respuesta dinámica en el tiempo	42.9%
c. Respuestas máximas	28.6%
d. Fuerza cortante	14.3%
e. Espectros de respuesta	57.1%

2. ¿Que parte del programa le ayudo a comprender mejor algún concepto de dinámica?	
a. Movimiento de los osciladores	57.1%
b. Gráficas de la respuesta dinámica	35.7%
c. Evolución de la respuesta en el tiempo	35.7%
d. Gráficas de la fuerza cortante	14.3%
e. Espectros de respuesta y de piso	57.1%
f. Respuesta por modo de vibrar	64.3%
3. ¿Que temas le gustaría que se le agregara al programa?	
a. Torsión	64.3%
b. Ductilidad	42.9%
c. Mostrar estructuras planas como marcos	35.7%
d. Sistemas elastoplásticos	64.3%
4. ¿Piensa que es mejor utilizar software educativo o seguir con la enseñanza tradicional (forma estática)?	
a. Seguir con la enseñanza tradicional	0.0%
b. Combinar las dos maneras de dar clase	78.6%
c. Uso del software con intervenciones del profesor	28.6%
d. Dar clase sólo utilizando el software educativo	7.1%

CAPÍTULO 7

CONCLUSIONES

Se creó un programa de computadora especialmente para el tema de dinámica estructural abarcando dos sub temas: osciladores de 1GL y VGL, que muestra en forma gráfica la respuesta dinámica y el comportamiento de los osciladores en el tiempo. Su composición entre la interfaz gráfica y los temas de contenido es adecuada para ser un software educativo.

La interfaz gráfica de usuario del programa es amigable y de fácil manejo, la distribución de las ventanas es adecuada y la función de cada uno sus controles está clara; diseñada de esta manera para evitar ser un programa complicado en su utilización que en lugar de ayudar al usuario a alcanzar sus objetivos se convirtiera en un impedimento para ello.

Los temas de contenido (osciladores de 1GL y VGL) incluidos en el programa son favorables para la enseñanza de la dinámica estructural, desde el punto de vista de la ingeniería sísmica. No obstante, con base en los resultados de las encuestas realizadas en el trabajo, el alumno está interesado en que el programa muestre más temas tales como: torsión y sistemas elastoplásticos entre otros.

Para evaluar al programa, se hicieron dos revisiones: la aproximación de la respuesta dinámica y la complejidad de la interfaz gráfica.

Con base en los resultados y comentarios obtenidos de las encuestas realizadas no fue necesario rediseñar el programa o hacer grandes cambios en su interfaz gráfica, sólo se corrigieron los pocos errores que se encontraron para dar paso a la versión 3.1.

Al no haber (por lo menos documentado en nuestro país) un software educativo para dinámica estructural, la comparación de la respuesta dinámica y las propiedades del oscilador de VGL se hicieron con los programas: *Staad Pro*, *USEE*, *Degtra* y un ejemplo tomado del libro de *Anil K. Chopra*. Con cada uno de ellos se revisaron diferentes aspectos de la respuesta dinámica y se concluyó que los resultados obtenidos del programa son correctos para ambos tipos de osciladores.

Con base en las comparaciones hechas de la respuesta dinámica, se concluye que la aproximación numérica de la integral de Duhamel, conocido también como el método de las ocho constantes, resulta altamente eficiente para calcular la respuesta dinámica de sistemas lineales.

El programa se define como un software educativo que sirve para el apoyo de la enseñanza de la Dinámica Estructural; los resultados que se obtienen de él no tienen aplicación práctica en el campo laboral.

El programa puede ser utilizado por cualquier persona, pero se enfatiza que fue creado como un software educativo para alumnos y maestros de ingeniera civil o posgrado, que cursan o enseñan respectivamente, la materia de dinámica estructural.

La reproducción de ejemplos con ayuda del programa resulta ser más sencilla, rápida y practica que de la forma de enseñanza tradicional; en esta última, el profesor no puede desarrollar tantos ejemplos para mostrar en el salón como lo haría con un software educativo creado especialmente para la materia.

Los programas que se encuentran en el mercado no proporcionan la respuesta dinámica completa que en el caso educativo es necesario para comparar resultados. En ocasiones el estudiante se encuentra en

situaciones de que al resolver su problema no sabe si en realidad el valor al que ha llegado es correcto. En este sentido el software educativo le puede ayudar a comprobar los valores obtenidos.

Una manera recomendable de usar el programa es que el profesor en clases previas explique los conceptos del tema, y después con ayuda del programa realice ejemplos que sirvan al alumno para visualizar dichos conceptos.

Fuera del salón de clase, el alumno puede utilizar el programa para reproducir ejemplos mostrados en clase, u otros que se encuentran en la bibliografía del tema. El objetivo es que el usuario indague, por si mismo, cuál es el comportamiento de los osciladores al variar los parámetros que se le permite cambiar como: periodo, masa, rigidez, amortiguamiento, señal de excitación, etc.

Los modos de uso del programa presentados en el capítulo 6 son optativos, pues el usuario puede usar el programa sin necesidad de seguir forzosamente la secuencia de pasos que se mostraron para cada una de las ventanas principales.

Las animaciones mostradas por el programa son de gran ayuda en el proceso de enseñanza-aprendizaje, ya que muestran al alumno los efectos de aplicar una carga dependiente del tiempo, situación que no es posible mostrar en la manera tradicional de dar clase.

Con base en los resultados de la encuesta aplicada, se concluye una mejor manera de dar clase es combinando y equilibrando la forma tradicional con el uso de un software educativo, sin llegar a caer en los extremos de cada caso.

Aunque la computadora se ha convertido en una herramienta imprescindible, su impacto dentro del aula de clase no se ha dado como en los campos laboral y de investigación debido a la falta de software educativo que sirva de herramienta al profesor para dar clase. Esta situación hace evidente que la magnitud del beneficio que la tecnología computacional puede ejerce sobre la educación está hoy muy por debajo de su potencial.

Los programas comerciales no satisfacen los requerimientos pedagógicos de un software educativo, ya que sus objetivos son completamente distintos. No obstante, el que no exista aún el software pedagógicamente ideal no debe ser motivo para dejar de lado la oportunidad de encontrar en la computadora a un poderoso aliado educativo.

El software educativo por si solo no puede enseñar al alumno; es necesario siempre contar con la explicación del profesor antes y durante cada tema. Por ello, el programa “Dinámica” se considera como una herramienta para la enseñanza-aprendizaje y no un sustituto del profesor y la bibliografía del tema.

El uso de la tecnología dentro del salón de clase resulta ser muy benéfica para el proceso enseñanza-aprendizaje, siempre y cuando ésta sea bien dirigida y que la conjunción entre la tecnología y los temas de contenido esté bien estructurada.

REFERENCIAS Y BIBLIOGRAFÍA

Aschheim M. y Abrams D. (2001), "Programa: Utility Software Earthquake Engineering USEE", University of Illinois, EUA.

Ávila P. (1999), "Aprendizaje con nuevas tecnologías paradigma emergente", Instituto latinoamericano de la comunicación educativa (ILCE), México.
http://investigacion.ilce.edu.mx/panel_control/doc/c37aprendizaje.pdf

Ávila P. (2007), "Aplicaciones didácticas de la tecnología", VII Congreso Internacional de material didáctico innovador, UAM, México.

Balena F. (2008), "Programación avanzada con Visual Basic 2005", Mc Graw Hill Interamericana

Bazan E. y Meli R. (1983), "Manual de diseño sísmico de edificios, de acuerdo con el reglamento de construcciones para el Distrito Federal", Series del Instituto de Ingeniería, UNAM, México.

Benítez R. (2000), "La educación virtual. Desafío para la construcción de culturas e identidades", ILCE México. Dirección electrónica:
http://investigacion.ilce.edu.mx/panel_control/doc/c37laeducacionvirtualq.pdf

Blanco M. (2002), "Programación en Visual Basic .Net" Editorial: Grupo EIDOS, España

Bork A. (1985), "Personal Computers for Education". Nueva York: Harper & Row Publishers

Cataldi Z, Lage F, Pessacq R y García R (2003) "Metodología extendida para la creación de software educativo desde una visión integradora" Revista latinoamericana de tecnología educativa, Volumen 2 No 1, Argentina.

Cataldi Z. (2006), "El ciclo de vida y la matriz de actividades como base para el diseño y desarrollo metodológico de software educativo", Revista de Ingeniería Informática Edición 13, Universidad de Concepción, Chile.

Charte F. (2002), "Programación con Visual Basic.NET", Editorial: Anaya multimedia, España.

Chopra A. K. (2001), "Dynamics of structures: theory and applications to earthquake engineering" Second edition, Prentice Hall, EUA.

Crisafulli F. y Villafañe E. (2002), "Espectros de respuesta y de diseño" Facultad de ingeniería UN-Cuyo, Argentina. Dirección electrónica:
http://www.ecv.ufsc.br/henriette/Analise_Dinamica/isr-espectros.pdf

Cuevas V. (2001), "Software Educativo", CINVESTAV, México, Dirección electrónica: http://www.matedu.cinvestav.mx/~ccuevas/SoftwareEducativo.htm

Franco G. (2005), "Capítulo 1: Creación de contenidos interactivos para la enseñanza de la física" Dirección electrónica:
http://webs.uvigo.es/educacion.editora/volumenes/Libro%203/C1_%20Angel%20Franco.pdf

Gálvez D (2001), “Modelado de las dudas de los alumnos y su integración en software educativo”, Tesis de maestría, Programa de posgrado en ciencias e ingeniería de la computación, UNAM, México.

Gértrudix M, Álvarez S, Galisteo A, Gálvez M y Gértrudix F (2007), “Acciones de diseño y desarrollo de objetos educativos digitales: programas institucionales”, Revista de Universidad y Sociedad del conocimiento Volumen 4 No 1, España.

Halvorson M, (2002), “Visual Basic.Net, aprenda ya”, Mc Graw Hill Interamericana, EUA.

Halvorson M, (2007), “Aprenda ya Visual Basic 2005”, Mc Graw Hill Interamericana, EUA.

Mendoza M. (2001), “Metodología para el desarrollo de software educativo multimedia”, Tesis de maestría, Programa de posgrado en ciencias e ingeniería de la computación, UNAM, México.

Microsoft Ayuda y soporte (2001), “Cómo crear archivos de la Ayuda HTML Context confidencial” Dirección electrónica: http://support.microsoft.com/kb/242433/es

Microsoft Corporation (2008) “Programa: Microsoft Visual Basic 2008, Edition Express”, EUA.

Microsoft Corporation (2008b) “Microsoft Visual Studio 2008 Version 9.0.21022.8 RTM”, EUA.

Microsoft Corporation (2008c), “MSDN Library” Dirección electronica: http://msdn.microsoft.com/en-us/library/default.aspx

Microsoft Press (2006), “Generar un programa con Visual Basic 2005 Express Edition”

Morales C. (1998), “Evaluación de software educativo” ILCE, México. Dirección electrónica: http://investigacion.ilce.edu.mx/panel_control/doc/c36,evaluacsoft.pdf

Ordaz M. (2006), “Apuntes del curso Dinámica Estructural”, Posgrado en Ingeniería UNAM, México

Ordaz M. y Montoya C. (2002), “Programa: DEGTRA A4, Versión 4.06”, Instituto de Ingeniería, Universidad Nacional Autónoma de México, México.

Paz M. (1992), “Dinámica estructural teoría y cálculo”, Editorial Reverté S.A., España.

Pérez G. J. (1990), “Análisis espectral modal por superposición directa de vectores de Ritz”, Tesis de maestría, Programa de posgrado de ingeniería UNAM, México.

Robinson, Ed. y Bond, M.J. (2003), “Seguridad para Microsoft Visual Basic.Net” Mc Graw Hill Interamericana, EUA.

Serrano J. (2005), “Manual de introducción a Visual Basic 2005 Express Edition” Microsoft

Som G. (2007), “Crear un proyecto de ayuda HTML” Dirección electrónica: http://www.elguille.info/hhw/dnm/hhw_01.aspx

Valdezate C. y Valdezate M. (2003), “Gráficos vectoriales con Visual Basic .NET” dirección electrónica: http://www.elguille.info/colabora/puntonet/tutGDI/indice.htm

Vilchis N. (2005), “El software educativo y su utilidad como recurso didáctico”, Tesis de maestría, Facultad de Filosofía y Letras, UNAM, México.