Desarrollo de una Interfaz Gráfica para una Herramienta de Cálculo de Estructuras

Rafael Olivera - Federico García

29 de noviembre de 2015

Índice

1. Introducción

1.1. Definición del problema y motivación

Desde las primeras casas construidas por el hombre, hasta el edificio más moderno y extravagante que exista en la actualidad, puede decirse que se buscó en el fondo el mismo objetivo: lograr una estructura segura, resistente y funcional. Hoy por hoy, la evolución del conocimiento humano y de la tecnología circundante ha permitido desarrollar a niveles altísimos el comprendimiento del problema y sus posibles soluciones.

El cálculo de estructuras, en ese sentido, es una rama fundamental dentro de la ingeniería civil. Se trata de una serie de complejos cálculos realizados con la finalidad de lograr estructuras óptimas con las condiciones descriptas anteriormente. A grandes rasgos, se busca que la estructura pueda soportar tanto su propio peso, como cualquier fuerza externa que pueda ser aplicada a la misma, lo que puede derivar en que la misma no siempre sea igual en la realidad a como se diseñó. Esto quiere decir que la estructura puede sufrir ciertas deformaciones antes de alcanzar su punto de equilibrio.

La Ingeniería en Computación no ha dejado este problema de lado, ya que existen diversos sistemas informáticos encargados de facilitar el diseño y cálculo de estructuras. Estos sistemas permiten, a grandes rasgos, dibujar una estructura mediante la definición de diferentes materiales, secciones, fuerzas externas, etc. Finalmente, realizan los cálculos correspondientes, mostrando la estructura en un estado de equilibrio y las deformaciones ocurridas en el proceso.

Así como existen estos sistemas reconocidos mundialmente, la Facultad de Ingeniería cuenta también son su propio sistema de cálculo de estructuras. Su nombre es IETFEM, y fué desarrollado por los Ing. Pablo Castrillo y Jorge Pérez pertenecientes al Instituto de Estructuras y Transporte(IET). Se trata de un motor de cálculo desarrollado en Octave que recibe una estructura descripta en formato texto y genera gráficas e imágenes con la deformación de la misma.

En este proyecto, se realizará una interfaz gráfica acorde para ser utilizada en conjunto con el motor de cálculo antes mencionado, y lograr así un sistema completo de diseño y cálculo de estructuras. Se busca, en particular, agregar funciones de dibujado y visualización de resultados que pueden observarse en otros sistemas de la misma índole, acercando al IETFEM a los sistemas comerciales y logrando una mayor usabilidad y eficiencia para los estudiantes que lo utilizan.

1.2. Desarrollo previo

Como mencionamos anteriormente, la FING cuenta con un motor de cálculo de estructuras desarrollado por los Ingenieros Pablo Castrillo y Jorge Pérez. El mismo resuelve problemas de cálculo de estructuras utilizando el Método de Elementos Finitos(MEF).

El MEF es, desde mediados del siglo XX, una de las principales herramientas utilizadas por los ingenieros para el análisis de sistemas estructurales, mecánicos, eléctricos, etc. El avance de la computación y la disponibilidad creciente de computadores potentes a bajo costo ha provocado que los programas comerciales de MEF para el cálculo estructural sean utilizados masivamente. De hecho, en los últimos cuarenta años el MEF ha transformado los procedimientos de trabajo de todas las áreas de ingeniería y constituye hoy una de las herramientas indispensables con las que un ingeniero debe contar en el ejercicio de su profesión. Por otra parte, el uso del MEF por parte de profesionales no debidamente capacitados podría eventualmente producir errores en el diseño de estructuras, y por tanto, riesgos para los usuarios.

En este contexto, la enseñanza del MEF en las carreras de Ingeniería se transforma en un desafío docente, donde además de formar a los estudiantes en el uso de diferentes programas de cálculo estructural es necesario transmitirles los conocimientos y herramientas que les permitan realizar un análisis crítico de los resultados. Es importante destacar además, que la mayoría de los programas comerciales (ej: SAP2000 y AxisVM) de MEF son de código cerrado, por lo que presentan como desventaja a nivel educativo, que no permiten a los estudiantes ver su funcionamiento interno, limitando la comprensión de los errores durante el aprendizaje.

De esta manera surge entre docentes del Grupo de Mecánica de Sólidos Computacional (MSC) del Instituto de Estructuras y Transporte (IET) la motivación de brindar una solución al problema a través del desarrollo de un software educativo adecuado: IETFEM.

IETFEM comenzó a desarrollarse en 2012 . El primer módulo desarrollado permitió resolver problemas de estructuras de barras articuladas o aporticadas en el plano con cargas aplicadas en los nodos. Esta primera versión fue utilizada por estudiantes del curso de Elasticidad 2013; luego se incluyó la posibilidad de generar un informe de salida en formato LaTeX. Posteriormente, la herramienta contó con el aporte del docente del IET, Agustín Spalvier, desarrollando la capacidad de ingresar cargas distribuidas uniformes en elementos de pórtico y el análisis modal de vibraciones de pórticos. Finalmente, a principios de 2014, Castrillo desarrolló un módulo para la resolución de problemas con variaciones de temperatura y fuerza de volumen en barras articuladas.

Se buscó una herramienta que sin ser compleja para su aplicación en cursos de grado, permita al estudiante visualizar el funcionamiento interno del método de cálculo. Por ello se optó por la sintaxis de programación de GNUOctave

(herramienta libre de alta compatibilidad con Matlab), ya conocida por los estudiantes. Se considera que contar con un software abierto donde los estudiantes pueden entender e incluso programar nuevos cálculos de acuerdo a sus necesidades, enriquece el trabajo desde el punto de vista didáctico.

La forma de ingreso de datos se eligió de acuerdo a otros programas de cálculo de estructuras como SAP2000 donde se deben definir: materiales, secciones, estados de carga, geometrías, conectividades, etc. En el IETFEM se optó por una entrada de archivo de texto plano donde el estudiante debe ingresar esta información. La salida también es en texto plano (.txt y .tex) y gráfica, al igual que en los programas comerciales.

Sin embargo, la generación del archivo de entrada y la comunicación con el IETFEM pueden llegar a ser tediosas y complicadas para el estudiante. Debe tenerse en cuenta que debe especificarse la estructura nodo por nodo, barra por barra, describiendo los materiales, secciones y fuerzas aplicadas, entre otras cosas, respetando además un formato fijo de documento que puede derivar en diversos errores de sintaxis.

Por lo tanto, se desarrollará en este proyecto una interfaz gráfica de código abierto donde el estudiante pueda dibujar la estructura de una manera sencilla e intuitiva, y que genere la entrada al IETFEM de manera automática. De esta manera, se pretende mejorar tanto la facilidad de uso como la eficiencia del mismo.

1.3. Objetivos y resultados esperados

Como mencionamos antes, a pesar de la increíble potencia en la resolución del problema del cálculo de estructuras, IETFEM presenta ciertos puntos a mejorar para ser comparado con otros sistemas del mismo rubro.

A lo largo de este proyecto perseguimos 2 grandes objetivos que consideramos esenciales para el enriquecimiento del sistema: Mejorar la eficiencia y mejorar la usabilidad

Para mejorar la usabilidad, se desarrollará una interfaz que permita al usuario dibujar la estructura de manera fluida y amigable. Se trata de un espacio 3D donde el usuario puede moverse libremente utilizando el mouse para mover y rotar la cámara. Se podrá dibujar la estructura de una manera continua e intuitiva. Además, facilitará la comunicación con el motor de cálculo previamente desarrollado y la visualización de los resultados obtenidos.

Para mejorar la eficiencia, reduciremos el tiempo de ejecución del motor de cálculo, eliminando el proceso de graficación y generación de imágenes, ya que ahora los resultados podrán verse en la nueva interfaz. Como regla básica, buscamos que el usuario pierda el menor tiempo posible en problemas tecnológicos o informáticos y que dirija sus esfuerzos al comprendimiento del problema en sí y su método de resolución.

A modo de resumen, se busca realizar un sistema ágil, de código abierto, que mejore ambos aspectos lo suficientemente como para poder ser utilizado sin problemas en el curso de Elasticidad dictado por el IET. Con el fin de verificar el cumplimiento de los objetivos por parte del sistema, una vez finalizado, será evaluado resolviendo ejercicios del curso antes mencionado, realizando comparaciones y análisis del tiempo de ejecución.

1.4. Desarrollo del proyecto

El proyecto comenzó con una fase fuerte de investigación. Inicialmente se realizaron reuniones ocasionales con los clientes, donde se reunió información valiosa sobre el problema de cálculo de estructuras y el método de elementos finitos. Además se definió qué tipo de herramienta se quería, qué funcionalidades eran deseadas y qué objetivos se buscaban. Durante esta etapa se utilizó el motor de cálculo directamente para comprender su funcionamiento y compararlo con otras herramientas comerciales.

Una vez comprendido el problema, se procedió a buscar herramientas con las cuáles desarrollar la interfaz. Se investigaron librerías y lenguajes de progamación 3D, optando al final por utilizar tecnologías web por su simplicidad de uso, agilidad y portabilidad

Posteriormente se comenzó a diseñar e implementar la herramienta, separando en diferentes módulos que serán descriptos en detalle en el capítulo 4. Se ejecutaron reuniones quincenales con los clientes para definir detalles, corregir errores, evaluar resultados y tomar decisiones en conjunto. Esta fase ocupó la mayor parte del tiempo del proyecto, debido a la dificultad técnica del mismo.

Finalmente, una vez alcanzado un producto inicial que cumplía las espectativas planteadas, se procedió a realizar pruebas sobre el mismo, detectando ciertos errores de performance que fueron solucionados hasta un nivel considerablemente bueno(se hablará de estas medidas en el capítulo 5).

1.5. Organización del documento

El resto del documento se organiza de la siguiente manera:

En el siguiente capítulo comenzamos analizando el estado del arte, tanto del problema de cálculo de estructuras como de herramientas de programación 3D, y su posible uso en sistemas de este tipo. Se realiza un estudio de diferentes herramientas investigadas, el estado de las mismas y su posibilidad de ser utilizadas en este proyecto. También se investigan otros sistemas de cálculo de estructuras y otro proyectos académicos similares en América Latina.

Posteriormente, en el capítulo 3, hablaremos de la organización del trabajo a lo largo del proyecto. Hablaremos del alcance del mismo, definiendo las fun-

cionalidades y características específicas que se buscan en el producto final. Se describirá la metodología de trabajo utilizada y se realizarán estimaciones para cada tarea comprendida, comparando finalmente con el esfuerzo efectivo.

A continuación, en el capítulo 4, se procederá a plantear la solución propuesta, detallando cada aspecto de la misma. Se describirá con exactitud su proceso de diseño e implementación, la arquitectura definida, el funcionamiento de cada componente, las herramientas utilizadas y su uso en general.

En el capítulo 5, se especifican los resultados obtenidos, analizando diferentes casos de prueba y comparando con resultados obtenidos desde IETFEM antes de la realización de este proyecto. Se analizan además los problemas obtenidos durante esta fase y cómo fueron resueltos.

Finalmente, el 6 capítulo, enumera las conclusiones obtenidas durante el proyecto, analizando el cumplimiento de objetivos y proponiendo posible trabajo a futuro a desarrollar sobre IETFEM.

2. Estado del Arte

2.1. Cálculo de estructuras

2.1.1. Cálculo implicados

aca hay que hablar el problea, les mandamos un mail paraver donde podemos leer mas

2.1.2. IETFEM

hablar de como resuelve el ietfem los problemas de arriba

beneficios del ietfem, resuelve probelmas complejos como los comerciales

carencias del ietfem, mencionar lso problemas un pocomas en detalle, mas que nada haciendo referencia a que a pesar de que soluciona probleas complejos eficientemente, asi como esta es muy dificil de usar

2.2. Herramientas comerciales

Existen en el mercado diversos productos de software enfocados al análisis de estructuras, con gran cantidad de funcionalidades y utilizados por ingenieros de todo el mundo en problemas reales. En el marco de este proyecto se exploraron con mayor rigurosidad dos herramientas: SAP2000 y AxisVM, las cuales en etapas mas avanzadas del desarrollo fueron tomadas como estándar para la

implementación de ciertas funcionalidades, basado principalmente en la experiencia de los clientes con las mismas.

2.2.1. SAP2000

Es un software comercial desarrollado por la empresa Computers & Structures,Inc. fundada en 1975 en California, siendo uno de los pioneros en herramientas de análisis de estructuras.

Actualmente en su versión 18, SAP2000 es una aplicación para computadoras que se ejecuta en ambientes Windows. Cuenta con un entorno gráfico 3D para el modelado y una interfaz de usuario muy completa que puede resultar demasiado compleja en los primeros pasos.

Entre las características mas importantes se encuentran:

- Un motor de análisis que puede resolver varios tipos de problemas tales como: (LES PREGUNTAMOS A LOS PROFES XQ ES UN HUEVO DE ENTEDNER),
- Diversas características para el modelado como templates, sistema de grillas, distintas vistas y herramientas de meshing.
- Diversos componentes estructurales como articulaciones, barras, cables sólidos, resortes, etc.
- Posibilidad de aplicar distintos tipos de cargas.
- Varias posibilidades para ver la salida de los cálculos con diagramas,tablas y reportes.
- Importación y exportación de modelos en distintos formatos estándar.

Por todo esto SAP2000 es uno de los productos lideres en el mercado siendo utilizado en mas de 160 países en todo el mundo.

2.2.2. AxisVM

Es un software comercial desarrollado por la empresa InterCAD Kft. en 1991 y con sede en Hungría.Fue una de las primeras herramientas 3D basada en el método de los elementos finitos.

Actualmente en su versión 13, AxisVM requiere computadoras con el sistema operativo Windows. El conjunto de características es muy similar al descripto en la anterior herramienta.

2.2.3. Herramientas Web

El sector del software de análisis estructural en la web (o nube) es un nicho poco explorado por los desarrolladores existiendo un conjunto muy limitado de ofertas en este sentido.

De acuerdo a la investigación realizada es importante destacar las siguientes ofertas:

- Idea StatiCa es un emprendimiento Checo que cuenta con calculadoras para 6 problemas principalmente en espacios 2D. Esta desarrollado en Silverlight y utilizando la nube de Microsoft Azure como plataforma de despliegue.
- CloudCalc es un software en crecimiento enfocado al análisis de estructuras de acero en la nube proveniente de Houston EEUU. Esta desarrollado utilizando WebGl para las características gráficas 3D.
- SkyCiv es un emprendimiento reciente de origen Australiano y es la suite mas desarrollada y con mayor calidad aparente de las vistas en este sector. Cuenta con calculadoras para distintos problemas en 2D y una versión pro que permite estructuras en 3D. Utiliza también WebGl para los gráficos.

Si bien existen algunas pocas ofertas, no logran niveles de calidad similares por ejemplo a herramientas de escritorio como SAP2000 o AxisVM, encontrándose así una ventana de oportunidad para el desarrollo de este tipo de herramientas con interfaces Web.

2.3. Desarrollo 3D

Dado el fuerte componente gráfico del proyecto fue necesario repasar un gran abanico de posibilidades a nivel tecnológico que permitan cumplir con los requerimientos 3D de la aplicación requerida. A continuación se muestran las principales opciones en este sentido que van, desde especificaciones estándar de muy bajo nivel de abstracción, pasando por wrappers de las mismas, hasta completos y potentes motores gráficos.

Se priorizaron fuertemente herramientas de código abierto dado que fue un requerimiento por parte de los clientes. Ademas se hizo especial foco en tecnologías conocidas por los desarrolladores tales como Java o tecnologías web.

2.3.1. OpenGL

Es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráfico 2D y 3D.

El funcionamiento básico consiste en aceptar primitivas como puntos, lineas y polígonos y convertirlas en píxeles. Es una API basada en procedimientos de bajo

nivel que requiere que el programador dicte los pasos exactos para renderizar la escena, eso es en contraste a APIs mas descriptivas, donde el programador solo debe describir la escena.

OpenGL tiene dos propósitos esenciales:

- Ocultar la complejidad de la interfaz con las diferentes tarjetas gráficas, presentando al programador una API única y uniforme.
- Ocultar las diferentes capacidades de las diversas plataformas hardware, requiriendo que todas las implementaciones soporten la funcionalidad completa de OpenGL

En la actualidad en su versión 4.5 se utiliza ampliamente en CAD, realidad virtual, representación científica, visualización de información, simulación y desarrollo de videojuegos donde compite con Direct3d (en plataformas Microsoft).

En virtud de lo detallado en cuanto al bajo nivel de abstracción y por consecuente prolongada curva de aprendizaje y baja productividad esta opción fue descartada rápidamente al menos en su uso directo.

2.3.2. JWJGL y JOGL

JWJGL (Lightweight Java Game Library 3) y JOGL (Java OpenGL) son wrappers de OpenGL que proveen acceso de bajo nivel a sus funcionalidades que a menudo no se implementan de manera correcta. No son librerías con gran cantidad de funcionalidades ni proveen utilidades de alto nivel.

En la actualidad existen muchas herramientas y motores gráficos para desarrollar aplicaciones 3D con mayor cantidad de funcionalidades, menor curva de aprendizaje y mayor productividad que utilizan estas librerías como base.

2.3.3. JMonkeyEngine 3

Es un motor de código abierto con fuerte inclinación para el desarrollo de videojuegos, hecho especialmente para desarrolladores Java para la creación de aplicaciones 3D utilizando las mas modernas tecnologías de una manera rápida y con una baja curva de aprendizaje.

Esta desarrollado en base a JWJGL y es la suite mas popular en el mundo java para desarrollo de videojuegos en alto nivel, con una gran comunidad de desarrolladores y extensivamente documentado. Si bien el enfoque principal son los videojuegos es importante destacar que tiene todas las capacidades para poder construir otro tipo de aplicaciones.

2.4. Desarrollo 3D en la Web

La utilización de tecnologías web para el desarrollo de la aplicación probaba a priori ser una opción con mucho potencial aportando gran flexibilidad, una opción multiplataforma ,multidispositivo ademas de ser innovadora para herramientas de este tipo.

Es por estas razones y la experiencia del equipo de desarrollo en estas tecnologías (HTML5,CSS, Bootstrap, javascript, angularjs , etc.) que se investigo la factibilidad de una solución gráfica 3D en este contexto.

2.4.1. HTML5 - Canvas

El contexto 2D para el elemento de HTML Çanvas"permite la creación de gráficos en paginas web. Es una tecnología que se usa principalmente para dibujar gráficos 2D en la web aunque con algunos trucos es posible realizar trabajos en 3D.

Si es posible, la relativa dificultad para realizar trabajos 3D y la gran diferencia de performance contra opciones como WebGL (Canvas corre en CPU) descartaron esta opción rápidamente.

2.4.2. WebGL

WebGL es una especificación estándar que está siendo desarrollada actualmente para mostrar gráficos en 3D en navegadores web. Permite mostrar gráficos en 3D acelerados por hardware (GPU) en páginas web, sin la necesidad de plug-ins en cualquier plataforma que soporte OpenGL 2.0 u OpenGL ES 2.0. Técnicamente es un API para javascript que permite usar la implementación nativa de OpenGL ES 2.0.

Existe una gran explosión en la creación de aplicaciones con esta tecnología, desde videojuegos 3D hasta aplicaciones científicas como visualizadores de estructuras moleculares, simulaciones del sistema solar o una aplicación de la NASA llamada .[Ex]perience Curiosity"por el aniversario del aterrizaje del robot Çuriosity rover.[en] Marte.

Es importante destacar que en la actualidad esta soportado por todos los principales navegadores web tanto en versiones de escritorio como de dispositivos móviles.

Con esta gran cantidad de demostraciones de calidad en WebGl y su amplio soporte se perfilo como una opción innovadora y altamente factible para la realización del proyecto.

2.4.3. Librerías para desarrollo 3D

Como WebGL es una tecnología diseñada para trabajar directamente con el GPU (unidad de procesamiento gráfico) es difícil de codificar en comparación con otros estándares web más accesibles, es por eso que muchas bibliotecas de JavaScript han surgido para resolver este problema.

Entre las mismas se privilegiaron aquellas con mayor cantidad de características implementadas, documentación y comunidad. La investigación entonces se simplifico a dos: ThreeJs y BabylonJs (Microsoft Open Source).

Ambas son librerías en javascript de alto nivel de abstracción sobre WebGl, tienen esencialmente el mismo conjunto de características tales como:

- Renderizacion con WebGl.
- Distintos efectos.
- Escenas, para añadir y eliminar objetos en tiempo de ejecución.
- Cámaras, perspectiva u ortográfica.
- Animaciones.
- Luces.
- Materiales.
- Shaders.
- Objetos y Geometrías
- Importación y exportación para texturas y otros assets.
- Gran comunidad de desarrolladores (bastante mayor en Threejs).

Ademas en sus paginas oficiales cuentan con cientos de códigos y aplicaciones de ejemplo que dejan ver el potencial de las librerías. Entre estos ejemplos se pueden observar editores estilo CAD que implementan varias funcionalidades similares a los requerimientos del proyecto lo cual asegura la factibilidad de la herramienta en este contexto.

2.5. Información complementaria

2.5.1. Investigación sobre proyectos similares en América Latina

buscar proyectos similares en internet, en america latina y el mundo, y compararlos

comparar lo encontrado con ietfem, y rematar señalando que es el primer proyecto de este tipo en sudamerica

3. Organización del trabajo

3.1. Alcance

Como mencionamos anteriormente, los objetivos planteadas en este proyecto se basan en mejorar tanto la eficiencia como la usabilidad del IETFEM. En ese sentido, existen dentro de la rama del cálculo de estructuras una infinidad de funcionalidades y mejoras posibles que pueden resultar útiles en nuestro sistema. Por lo tanto, se definió un conjunto acotado de funcionalidades y características deseables en el producto final, apuntando a alcanzar satisfactoriamente los objetivos planteados y lograr una herramienta de alto nivel.

Se consideró como prioridad apuntar a una herramienta académica, es decir, una herramienta libre, intuitiva para los estudiantes y aplicable en cursos dictados por el IET. En particular, se tomó como referencia el curso de Elasticidad, curso donde ya fué utilizado satisfactoriamente el IETFEM y donde será utilizado luego de la realización de este proyecto.

La principal y más grande funcionalidad que se desarrollará será le presencia de un espacio 3D. El mismo será el elemento central de la aplicación, mediante la cual el usuario efectuará la mayor parte de las interacciones posibles. Se pretenden integrar dentro de este espacio 3D las siguientes funcionalidades:

- Rotación de la cámara de visualización.
- Movimiento de la misma por todo el espacio 3D.
- Zoom In y Zoom Out.
- Dibujado de nodos y barras.
- Dibujado de grillas auxiliares.
- Selección de nodos y barras: Para setear propiedades en las mismas.
- Eliminación de nodos y barras.
- Visualización de propiedades: Fuerzas, puntos de apoyo y resortes.
- Visualización de estructura resultante: Observar la deformada y comparar con estructura original.
- Escalamiento la estructura deformada: «Exagerar» la deformación, para apreciar pequeñas deformaciones.
- Visualización de las propiedades de la estructura deformada utilizando escalas de colores: Deformación, Fuerzas, Tensiones, etc.

Más allá de que se pretende que el usuario tenga una experiencia interactiva mediante el dibujado de la estructura, es necesario definir ciertas funcionalidades fuera del espacio 3D. Ya sea tanto por comodidad como por intuición, estas opciones se encuentran en diferentes menús que rodean el espacio, similar a los

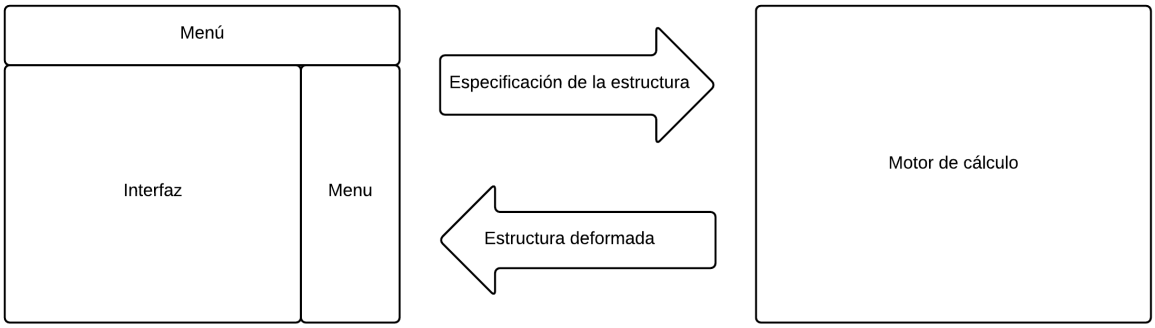

Figura 1: Ciclo de vida de IETFEM

demás programas comerciales dentro del rubro que se investigaron en el capítulo anterior.

- Abrir y Guardar Estructuras.
- Definición de Materiales.
- Definición de Secciones.
- Asignar propiedades a barras: Material y sección.
- Asignar propiedades a nodos: Fuerzas, Apoyos y Resortes.
- Selección de nodos y barras: Para setear propiedades en las mismas.
- Eliminación de nodos y barras.
- Prendido y apagado de elementos auxiliares
- Seteo de Factor de escalamiento para la estructura deformada

Si bien estos elementos nos permiten estimar una interfaz gráfica completa e intuitiva, resta definir aún la funcionalidad más importante del proyecto: la comunicación con el motor de cálculo. La salida de la interfaz debe ser un archivo reconocible por el IETFEM, del cuál pueda obtener todos los datos de la estructura. Así mismo, el motor debe ofrecer como salida otro archivo, el cuál será recibido por la interfaz con el fin de mostrar los resultados obtenidos. Dicha comunicación puede observarse en la Figura 1, aunque se hará hincapié en cómo se resolvió esta comunicación en el siguiente capítulo.

3.2. Metodología de trabajo

En las primeras etapas del proyecto se focalizó el trabajo en comprender el problema que se quiere resolver. Se tuvieron reuniones quincenales con los clientes

Descripción	Estado	Prioridad
Agregar la posibilidad de definir un material por defecto al dibujar barras	Resuelta	1
Agregar sprites y flechas a los nodos cuando se definen fuerzas o condiciones de desplazamiento	Resuelta	1
Agregar funcionalidad para "prender" y "apagar" grillas y fuerzas	Resuelta	2
Agregar funcionalidad Abrir Modelo / Guardar Modelo	En proceso	2
Agregar funcionalidad Nuevo Modelo, que limpia la escena y permite definir las nuevas unidades de medida	Para hacer	3

Figura 2: Planilla Excel utilizando metodología Kanban

dónde de habló del problema del cálculo de estructuras y cómo lo resuelve IETFEM. Dichas reuniones se apoyaron además en una permanente comunicación por e-Mail y una vasta investigación del problema por nuestra parte. Para esto no sólo se investigó sobre el problema, sino que además se utilizaron productos similares e incluso el propio IETFEM con ejemplos simples.

Una vez comprendido el problema, se pasó a buscar una solución al mismo. Dentro de esta etapa se pueden incluir la busqueda de herramientas, el análisis y el diseño de la aplicación. Se mantuvieron las reuniones con los clientes, evaluando herramientas y enseñando prototipos realizados a modo de prueba. Se investigaron lenguajes y librerías de programación 3D, tanto de escritorio como web, deciciendo en ultima instancia utilizar WebGL (se hablará mas en detalle en el siguiente capítulo).

Conforma pasaba el tiempo las reuniones se fueron enfoncando cada vez más en el producto final, comenzando a definir las funcionalidades y características del mismo. Mientras se mantenía contacto con los clientes, se realizó por nuestra parte la definición de casos de uso, con sus respectivos diagramas de flujo, al tiempo que se definió la arquitectura del sistema en base a los requerimientos obtenidos y las prestaciones de las herramientas definidas.

Finalmente, para las etapas de implementación y testing, se creó un repositorio en Github con el esqueleto de la aplicación y todo código reusable proveniente de la etapa de prototipación. Como metodología de trabajo se utilizó la metodología ágil Kanban. Kanban es un método para gestionar el trabajo intelectual, con énfasis en la entrega justo a tiempo, mientras no se sobrecargan a los miembros del equipo. En este enfoque, el proceso, desde la definición de una tarea hasta su entrega al cliente, se muestra para que los participantes lo vean y los miembros del equipo tomen el trabajo de una cola.

Existen diversas herramientas on-line de planificación y gestión de proyectos, tales como Jira o TFS. Sin embargo, debido a la poca cantidad de personas involucradas en el proyecto (2 desarrolladores y 2 clientes) y a que las tareas a realizar estaban bien definidas, se optó por utilizar una herramienta simple y natural: una planilla Excel online. La misma se encontró en todo momento de libre acceso y modificación para los 4 participantes, y cada tarea tenía asignada una descripción, un estado, y una prioridad.

En un principio, se agregaron todas las tareas a realizar, y ambos desarrolla-

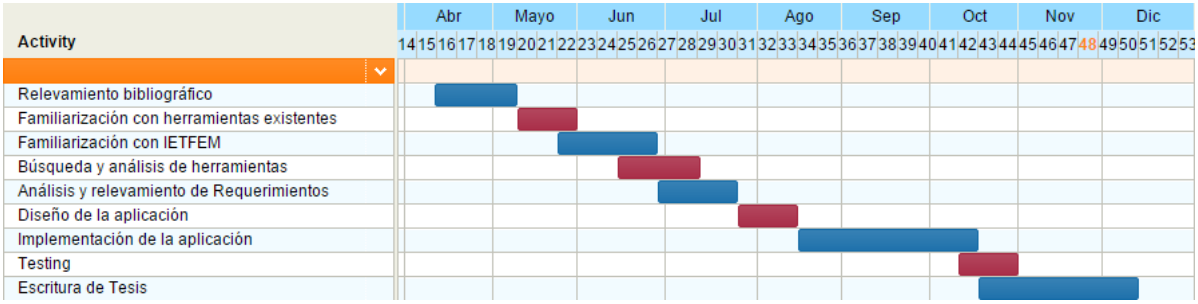

Figura 3: Diagrama de Gantt con la planificación del proyecto

dores tomaban cada una de ellas marcando su estado como «En proceso». Una vez finalizada, se marcaba la tarea como terminada y se subían los cambios al repositorio, marcando cada subida con la funcionalidad correspondiente.

A su vez, los clientes, los cuáles también tenían acceso a la última versión del IETFEM, iban relevando en la misma planilla problemas o cosas a mejorar que se encontraban en el produto, los cuáles pasaban a ser parte de nuestra «pizarra de kanban» y seguían el mismo flujo que las demás tareas.

3.3. Estimación y esfuerzo efectivo

La planificación del tiempo se realizó tomando en cuenta el desconocimiento inicial del problema de cálculo de estructuras y la dificultad de la programación gráfica en 3D. En la Figura 3 se pueden ver las estimaciones realizadas calculando 15 horas de trabajo semanal por desarrollador. Se puede apreciar que el período de trabajo se calculó entre Abril y Diciembre, logrando un total de 34 semanas de trabajo, que se traducen en un total de 1020 horas de trabajo.

Podemos ver también que ciertas etapas se planificaron en simultáneo por ciertos períodos de tiempo, especialmente en las etapas tempranas del proyecto donde se comenzó utilizando y comprendiendo tanto el IETFEM como otras herramientas, mientras se iba definiendo al mismo tiempo cómo realizar la interfaz. Se planificó de esta manera debido a que se consideró que sería bueno evaluar varias herramientas en simultáneo, a modo de comparar y definir qué funcionalidades y características nos gustaría que estén presentes en nuestro sistema. También evaluar cómo llevarlas a cabo utilizando las herramientas que existen en el mercado y el contexto académico en el cuál se quiere insertar la aplicacación.

También se observa la concurrencia de tareas en los instantes finales del desarrollo, donde se planificó al mismo tiempo el testing y la escritura de la Tesis. Debido a la metodología ágil elegida y al tiempo estipulado, resulta conveniente que el testeo de la aplicación comience cuanto antes, ya que corregir un error

pasará a ser parte de nuestra cola de tareas, y dependiendo de la prioridad de la misma podría ser resuelta antes que otras tareas definidas anteriormente pero con una baja prioridad. La escritura de la tesis se planificó en simultáneo simplemente para intentar reducir el tiempo total del proyecto.

El cronograma estimado se realizó de manera exitosa, siguiendo cada etapa en el orden estipulado sin demoras excesivas. Como agregado, durante la implementación se descubrieron nuevas funcionalidades que serían útiles en el sistema, las cuales fueron evaluadas con los clientes y algunas de ellas se llevaron a cabo sin problemas, debido a que la metodología de trabajo lo permitía.

También es necesario destacar el tiempo invertido en la Ingeniería de Muestra a fines del mes de Octubre, el cuál contempló el diseño de carteles, presentación del proyecto y la propia presencia en el evento. Esto redujo unos días el tiempo estipulado para la escritura de la tesis, el cuál se intenta recuperar durante el mes de noviembre aumentando la cantidad de horas a un promedio de 20 semanales por desrrollador dedicadas a dicha tarea.

4. Presentación de la solución

En esta sección se describe la solución propuesta para el problema planteado, describiendo cada aspecto de la misma y cómo fue realizada cada una de sus funcionalidades. Se detallan además las decisiones que fueron tomadas durante el proceso de análisis y diseño de la aplicación.

4.1. Análisis y relevamiento de requerimientos

Desde un principio se supo que IETFEM era una herramienta robusta, ofreciendo una solución para diferentes problemas posibles. En este sentido, el relevamiento de requerimientos se convirtió en una tarea delicada en dónde debía definirse un número acotado de funcionalidades, para un número acotado de la totalidad de problemas que IETFEM podía resolver.

Luego de concretar varias reuniones con los clientes, se decidió que la interfaz pueda resolver problemas de estructuras reticuladas, es decir, estructuras formadas por una serie de vigas entrecruzadas y conectadas entre sí por medio de nudos rígidos. Esto implica que para dibujar una estructura desde la interfaz, el usuario sólo tenga que colocar nodos y barras.

El estudiante coloca los nodos en el espacio 3D, y luego define barras entre 2 nodos ya dibujados, asignando para cada barra un material que la conforma y el área de su corte transversal, al que llamaremos sección, ambos previamente definidos. También pueden definirse ciertas propiedades para cada nodo, en particular, pueden definirse fuerzas aplicadas al mismo, condiciones de desplazamiento y resortes.

Una vez finalizado el proceso de dibujado, se extrae la estructura en un formato reconocible por el motor. Luego se ejecuta el mismo, y se analizan los resultados obtenidos.

Destacamos además como funcionalidades secundarias la posibilidad de definir grillas auxiliares con motivo de facilitar el ingreso de datos y la posibilidad de ocultar elementos adicionales, como por ejemplo, los vectores indicadores de fuerzas aplicadas.

Basándonos en esta realidad, se definieron los siguientes casos de uso:

- **Alta, Baja y Modificación de Materiales:** Los materiales se definen en base a 5 propiedades: Nombre, Modulo de Young, Gamma, Alpha y Nu.
- **Alta, Baja y Modificación de Secciones:** La sección es el corte transversal de una barra, y para este caso solo interesa conocer su área.
- **Alta, Baja y Modificación de Nodos:** Cada nodo tiene asignado un conjunto de coordenadas espaciales (x,y,z). Además es posible asignar al mismo una fuerza aplicada, así tambien como condiciones de desplazamiento y resortes en cada coordenada.
- **Alta, Baja y Modificación de Barras:** Cada barra tiene asignado un nodo inicial, un nodo final, un material y una sección.
- **Crear grilla:** Son «cuadrículas» auxiliares que facilitan el proceso de dibujado. Para cada coordenada se define la cantidad de líneas auxiliares y la separación entre ellas.
- **Modificar Visualización de Propiedad:** Los nodos con propiedades definidas, como por ejemplo fuerzas aplicadas o resortes, son marcados en la pantalla con vectores o geometrías básicas para ser diferenciados del resto. Esta funcionalidad permite ocultar, mostrar y escalar dichos elementos a gusto del usuario.
- **Nueva Estructura:** Permite limpiar la pantalla para comenzar una nueva estructura.
- **Abrir y Guardar Estructura:** Se busca la posibilidad de obtener un archivo con la estructura dibujada, de manera de poder seguir con el trabajo realizado en otro momento. También es deseable la carga de dicho archivo en la interfaz, obteniéndo la misma estructura en la que se estaba trabajando al momento de guardar.
- **Extraer Estructura:** A partir del dibujo realizado, se extrae un archivo reconocible por el motor de cálculo con la especificación de la estructura
- **Procesar Resultado:** Se trata de procesar el archivo resultante del motor, y actualizar la pantalla con la estructura deformada.

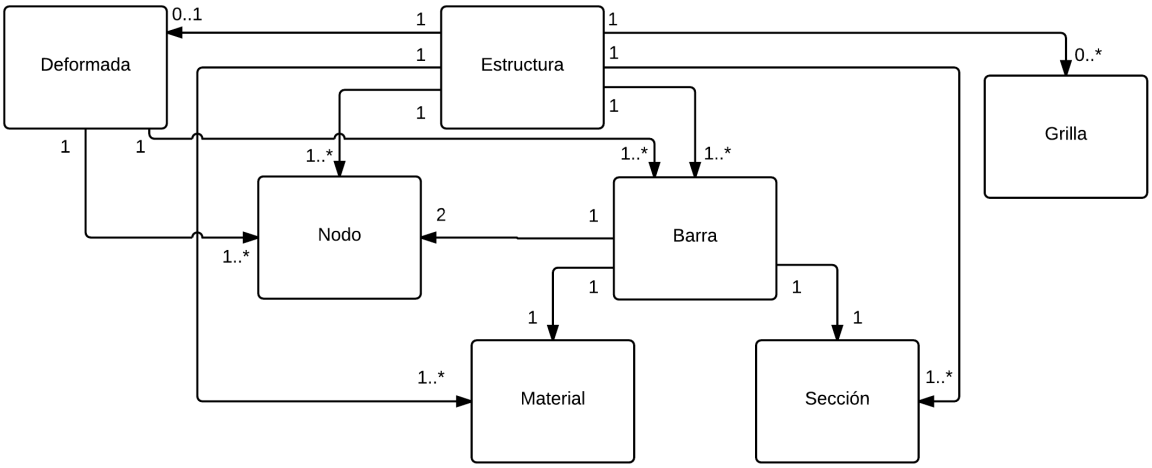

Figura 4: Modelo de dominio de IETFEM

- **Escalar Deformada:** Debido a que en algunos casos las deformaciones pueden ser tan pequeñas que pueden parecer imperceptibles a la vista, se incluye este caso de uso con el fin de «exagerar» la deformación y poder apreciar mejor los resultados obtenidos.
- **Colorear Estructura:** Al igual que la funcionalidad anterior, este caso de uso aplica a los resultados obtenidos del motor. Se trata de colorear la estructura en base a los datos obtenidos(por ejemplo, pintar de un color las barras que se comprimen y de otro las que se estiran). También se busca transparentar la estructura original o la deformada, para poder apreciar mejor los cambios entre una y otra.

En el anexo **??** se incluye la especificación de cada caso de uso descripto.

Puede apreciarse el Modelo de Dominio definido en la Figura 4. Como observaciones, se destaca la presencia de la entidad «Deformada», la cuál puede existir o no de acuerdo a si ya se procesaron los resultados obtenidos del motor o si se encuentra en el proceso de dibujado. De esta acotación se desprende el «por qué» de la relación 0..1 - 1 entre las entidades «Deformada» y «Estructura»: mientras se dibuja la estructura todavía no se tiene una deformada definida.

El resto del modelo se encuentra considerablemente intuitivo y adecuado a la realidad planteada.

4.2. Diseño de la solución

Finalizado el relevamiento de requerimientos y correspondiente análisis, se prosiguió con la etapa de diseño, donde se tomaron decisiones importantes tanto a nivel de diseño tecnológico como en la estructura propia de la aplicación.

Desde las primeras reuniones que se tuvieron con los clientes, el objetivo principal fué lograr una aplicación académica. De esta manera, se tuvo como prioridad mantener la simplicidad y la eficiencia de la herramienta por sobre acomplejizar la misma con funcionalidades potentes que serían útiles en un programa profesional. Por ejemplo, se toma en cuenta que en un ámbito académico, el estudiante no ingresará en el sistema estructuras gigantescas (veáse la seción 2.3 del capítulo 5), y sólo utilizará el mismo para los temas comprendidos dentro del curso que desarrolla. Cabe destacar que además, se busca en un segundo plano, lograr el mayor porcentaje de reusabilidad de código posible, ya que en un futuro, IETFEM puede crecer gradualmente para convertirse en un sistema profesional.

Teniendo en cuenta estos aspectos, sumado a las prestaciones destacadas en las herramientas de desarrollo 3D en la web, y la poca cantidad de sistemas de cálculo de estructuras en la nube, se decidió en conjunto con los clientes, realizar la interfaz en un ambiente web.

Sin embargo, realizar la interfaz en la nube implica ciertas situaciones preocupantes por parte de los clientes, por ejemplo, mantener un servidor donde se aloje la misma una vez finalizado el proyecto. Es necesario entonces destacar ciertas consideraciones sobre la solución elegida:

- Los clientes se sienten a gusto con considerar una versión final en la nube, ya que la mayoría de este tipo de sistemas son de escritorio y no para todos los sistemas operativos.
- Existe preocupación con respecto al servidor donde se aloje la aplicación. En particular, preocupa justamente encontrar un servidor gratuito donde alojarse y cómo mantener la aplicación una vez finalizado el proyecto
- Una de las prestaciones actuales del motor de cálculo existente es que al estar desarrollado en Octave, permite al estudiante ver el funcionamiento interno del código(o incluso programar nuevos cálculos de acuerdo a sus necesidades), enriqueciendo el proceso de aprendizaje de los estudiantes. Dichas características se quieren mantener en la nueva solución.
- Se busca reusabilidad en el código de la aplicación, ya que en un futuro se pretende evolucionar la herramienta a un nivel profesional, donde se pretende que la misma posea diferentes características (por ejemplo, no sería deseable en un sistema profesional que se pueda acceder al código del motor de cálculo directamente).

Tomando en cuenta las mencionadas premisas, la solución propuesta es la si-

guiente: Se desarrollará la interfaz como una herramienta web. Sin embargo, no se desplegará la misma en un servidor, sino que se encapsulará la misma en un framework que permita ejecutar la misma como una aplicación de escritorio. A ojos del estudiante, la aplicación parecerá ser de escritorio.

Una vez finalizado el dibujado de la estructura, el estudiante podrá generar un archivo con al especificación de la estructura, el cual podrá ingresar en el motor de cálculo de manera manual. Luego, puede desde la interfaz procesar la salida del motor para observar sus resultados.

De esta manera logramos las siguientes características:

- Para la versión inicial, es decir, la versión académica, se ahorra la utilización del servidor, ya que cada sistema ejecutará en la máquina de cada estudiante. Esto implica que el mantenimiento a posteriori sea nulo por parte de los clientes una vez finalizado el proyecto.
- Se desacoplan el motor y la interfaz, o sea, el estudiante puede visualizar los cálculos realizados en el motor, o incluso programar nuevos, sin necesidad de tocar el código de la interfaz. Es más, el archivo generado por la interfaz será en un formato legible, lo que hace que le estudiante pueda editar el archivo en caso de agregar cálculos nuevos.
- A su vez, esta solución no solo permite agregar cálculos nuevos a los estudiantes, sino que permite que el desarrollo del motor siga avanzando sin entorpecer la interfaz.
- Se obtiene un código totalmente reusable, ya que si en el futuro se quiere evolucionar la herramienta como un producto profesional en la nube, sólo basta con desplegar el código de la aplicación en un servidor.

De esta manera, la herramienta cumple con todas las especificaciones deseadas por los clientes, manteniendo las características positivas de la misma, y a su vez, potenciando la misma en vista de conseguir los objetivos planteados sobre mejorar la eficiencia y la usabilidad.

Finalmente, de acuerdo a las pautas establecidas en la sub-sección anterior, el flujo de la aplicación queda establecido como se muestra en la Figura 5. Los pasos 2 y 3 se anotan como opcionales debido a que el usuario puede cargar una estructura guardada como dibujar una nueva. Incluso podría realizar ambas, editando una estructura guardada antes de ejecutar el motor

De acuerdo a los casos de uso relevados, se distribuyeron las funcionalidades requeridas en diferentes módulos. Cada uno de estos módulos o subsistemas encapsula operaciones que se relacionan de alguna manera, logrando un nivel bajo de acoplamiento entre cada uno de ellos. Se hablará en detalle de cada subsistema en la siguiente sección.

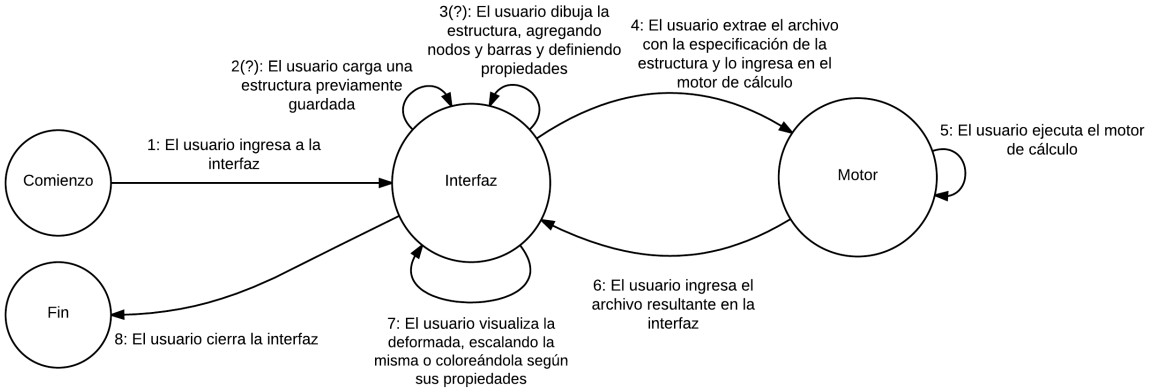

Figura 5: Flujo principal de la aplicación

4.3. Arquitectura

La arquitectura de la aplicación sigue el clásico patrón MVC(Modelo-Vista-Controlador), donde el usuario se encuentra permanentemente interactuando con el sistema, modificando el Modelo(en este caso, la estructura) y visualizando el mismo en el espacio 3D, al que llamaremos Escena.

Cómo se puede ver en la Figura 6, se agruparon los casos de uso relacionados con el fin de crear diferentes subsistemas encargados de realizar cierto tipo de funcionalidades. Cada una de estas componentes, ofrece al usuario diferentes operaciones que afectan tanto el modelo de la estructura que se mantiene almacenado en la aplicación como lo que se está viendo en el espacio 3D. Por tal motivo, se crearon las componentes «Modelo» y «Escena», las cuáles uniformizan todas las operaciones básicas que se hacen en el modelo de la estructura, y en el dibujo en la escena, respectivamente. Además se destaca el subsistema «Cámara», el cual se relaciona directamente con la escena, encargado de los movimientos del usuario dentro del espacio 3D (Rotación y Desplazamiento) y el subsistema «Deformada», el cual maneja las operaciones básicas que se hacen en la estructura deformada.

Finalmente, las operaciones que puede realizar el usuario se dividen en 8 subsistemas que se describen a continuación:

Main: Subsistema encargado de realizar la inicialización correcta del sistema. Contiene las operaciones relativas a todo el contexto de la aplicación: Cargar o guardar una nueva estructura, extraer la especificación de la estructura para el motor, y procesar el archivo con los resultados obtenidos. Por último, se encarga de la creación de las grillas auxiliares, ya que se considera una operación relativamente pequeña y poco relevante en el modelo de la estructura como para

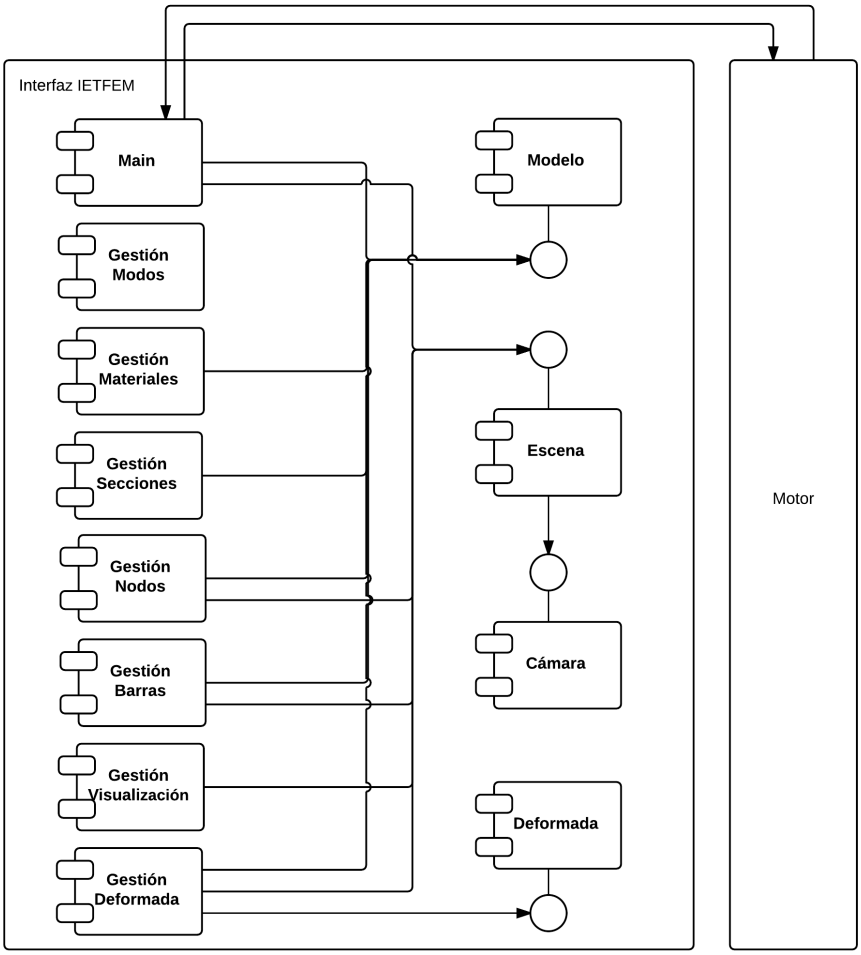

Figura 6: Diagrama de componentes de IETFEM UI

separarse en un módulo propio.

Gestión Modos: Debido a la necesidad de incluir diferentes características en interacción directa con la escena, se decidió mantener la aplicación en diferentes estados o modos. De esta manera, por ejemplo, un click en la escena realizará diferentes acciones dependiendo de en qué modo se encuentre el usuario. Este pequeño módulo se encarga de gestionar adecuadamente el estado actual y el pasaje entre diferentes estados.

Gestión Materiales: Este subsistema mantiene la creación, modificación y eliminación de Materiales. Debido a la naturaleza de la característica, se accede al mismo mediante un formulario en un menú superior, donde se definen las propiedades de cada material.

Gestión Secciones: Este subsistema mantiene la creación, modificación y eliminación de Secciones. Al igual que en con los materiales, la interacción con dicho módulo se lleva a cabo mediante un formulario.

Gestión Nodos: Este subsistema mantiene la creación, modificación y eliminación de nodos. Los nodos pueden agregarse haciendo click en la escena o ingresando sus coordenadas manualmente. También se ofrece un formulario en donde se pueden agregar propiedades a los nodos: Fuerzas, Condiciones de desplazamiento y Resortes. Cada una de estas propiedades, además agregan a la escena diferentes elementos que indican el valor de cada una de ellas:

- Si se define en el nodo una fuerza aplicada, se dibuja su correspondiente vector a puntando a ese nodo.
- Si se define en el nodo una condicion de desplazamiento en alguna de sus coordenadas, se dibuja una pequeña pirámide de color rojo donde su eje principal tiene la dirección de la propia coordenda en que se define.
- Si se define en el nodo una resorte en alguna de sus coordenadas, se dibuja una pequeña pirámide de color gris donde su eje principal tiene la dirección de la propia coordenda en que se define.

Gestión Barras: Este subsistema mantiene la creación, modificación y eliminación de barras. Las barras se agregan directamente en la escena, seleccionando un nodo inicial y un nodo final. También debe tener asignado un material y una sección, los cuales deben estar previamente definidos y pueden ser seteados una vez dibujada la barra. Se ofrece además la opción de definir propiedades «por defecto», es decir, se elije un material y una sección, y todas las proximas barras que se dibujen tendrán seteadas dichas propiedades.

Gestión Visualización: Módulo encargado de gestionar la visualización de elementos indicativos en la escena, es decir, muestra u oculta los vectores, resortes y condiciones de desplazamiento definidos en la estructura. También ofrece la posibilidad de escalar los vectores presentes en la escena, de manera de no entorpecer la imagen cuando las fuerzas aplicadas son muy grandes.

Gestión Deformada: Subsistema encargado de gestionar la deformada obtenida del procesamiento de resultados. Ofrece operaciones para visualizar la estructura deformada, escalar deformaciones y colorear la estructura en base a los resultados.

Cada uno de estos subsistemas interactúa con los módulos «Escena», «Modelo» y «Deformada» de acuerdo a sus necesidades:

Modelo: Expone operaciones básicas para modificar el modelado de la estructura que se está ingresando. Cada vez que otro módulo necesite ingresar, modificar o eliminar un nodo, barra, material o sección, llamará a funciones contenidas en este módulo.

Deformada: Expone operaciones básicas para interactuar con la estructura deformada obtenida. Cada vez el módulo de «Gestión Deformada» deba mover, colorear o transparentar un nodo o barra, se utilizarán funciones expuestas en este módulo

Escena: Expone operaciones básicas para interactuar con el espacio 3D. Cada vez que otro módulo necesite agregar o eliminar cualquier tipo de elemento del espacio 3D, se invocarán funciones expuestas en este módulo

Cámara: Mantiene el manejo de la cámara en la escena. Ejecuta funciones de desplazamiento, rotación y zoom.

De esta manera se define el sistema «IETFEM UI», el cuál interactúa con el sistema «IETFEM Core», que contiene el motor de cálculo, para totalizar lo que sería el IETFEM.

En la Figura 7 se puede preciar la distribución física de ambos componentes. En la imagen superior, se aprecia la versión estudiantil, donde todo se ejecuta en la máquina del usuario. Se observa la interfaz corriendo sobre un framework que simula ejecutar una aplicación de usuario, y el motor de cálculo en GNU-Octave, comunicándose manualmente mediante la acción del usuario.

En la imagen inferior, se aprecia una posible distribución física para una posterior versión, en donde se propone separar físicamente la interfaz del motor. De esta manera se logra una mayor escalabilidad al poder replicar N servidores Web utilizando el mismo motor de cálculo Se propone desplegar la interfaz en un servidor Web, mediante la cuál el usuario accede utilizando el protocolo estándar HTTP. El usuario dibuja la estructura de la misma manera que se realiza en la versión académica, pero al momento de ejecutar el motor, se consume un servicio REST expuesto por un servidor de aplicación en donde se encuentra corriendo el motor de cálculo, el cual provee a la interfaz con la estructura deformada. Esto implica que el usuario sólo tenga que presionar un botón para deformar la estructura, evitando el proceso de comunicación manual.

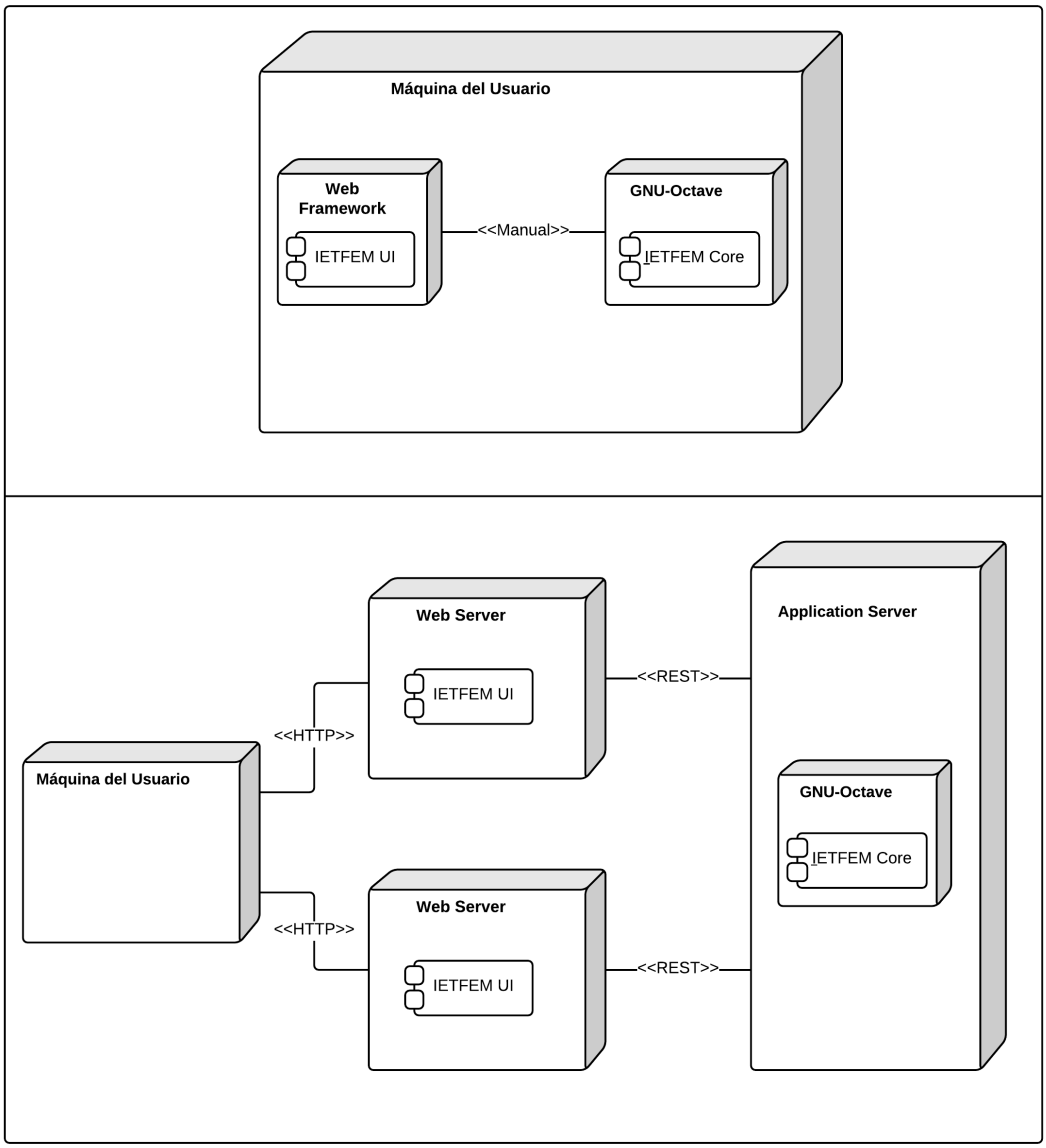

Figura 7: Diagrama de distribución física de IETFEM

4.4. Tecnologías y herramientas utilizadas

4.4.1. HTML5 - Javascript - CSS3

Dada la elección de utilizar tecnologías web para encarar el desarrollo del proyecto, existen ciertas tecnologías estándar e ineludibles en cualquier aplicación web estas son HTML5, Javascript y CSS3. A continuación se describen brevemente las mismas.

HTML es un lenguaje de marcado para la elaboración de paginas web. Es un estándar que sirve de referencia para la elaboración de paginas web y define una estructura básica y un código para la definición de contenido como texto, imágenes, videos, entre otros. En la actualidad se encuentra en su versión 5 la cual establece una serie de funcionalidades, elementos y atributos que reflejan el uso típico de los sitios web modernos.

JavaScript (abreviado comúnmente "JS") es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.Se utiliza principalmente del lado del cliente implementado como parte de los navegadores web, permitiendo mejoras en la interfaz de usuario y aplicaciones web dinámicas.Es el estándar de facto para scripting en la web y es interpretado por todos los navegadores web.

CSS actualmente en la versión 3 es un lenguaje de estilos que define la presentación de los documentos HTML. Esto abarca cuestiones relativas a fuentes, colores, margenes, altura, ancho, posicionamiento, etc. Una hoja de estilos fue utilizada en el proyecto con el fin de definir algunos estilos de menúes y formularios, esto es en aquellos no definidos o para personalizar los definidos en bootstrap.

4.4.2. Bootstrap

Bootstrap es un "front-end framework.ºpen source, es la mas popular de las librerías HTML,CSS y JS para el desarrollo de paginas web responsivas y mobile first, diseñado para ayudar a construir componentes de la interfaz de usuario.

Las principales características del framework las cuales fueron ampliamente utilizadas en el proyecto son:

- Sistema de grillas responsivo para posicionar todos los elementos de la pagina de una manera sencilla.
- Estilos para controles de HTML.
- Componentes personalizados.
- Componentes javascript (Ej. Modals)

4.4.3. AngularJS

Es un framework open source mantenido por Google que tiene como objetivo solucionar los principales problemas encontrados en el desarrollo de aplicaciones web dinámicas.

AngularJs propone la utilización de programación declarativa para las interfaces de usuario y programación imperativa para lógica de negocio. Implementa el patrón MVC para separar la presentación, datos y lógica. Todo esto da como resultado una aplicación prolija y testeable a nivel de código.

Principales características utilizadas:

- Two way data-binding: esto permite mantener el modelo y la vista (DOM) sincronizados sin necesidad de escribir código especial para mantener dicha sincronización.
- MVC
- Directivas, sirven para agregar funcionalidad a HTML mediante tags HTML, tanto built-in como personalizados.

4.4.4. ThreeJS

Dada la elección de realizar una solución Web,la clara superioridad de WebGl para las características gráficas y la baja productividad y dificultad de desarrollo directamente sobre el, resultó necesario elegir un framework que lo abstraiga.

En este sentido se decidió por ThreeJs por ser un proyecto activo, con la mayor cantidad de funcionalidades, buena documentación, con la mayor comunidad y por consiguiente mayor soporte entre todas las opciones.

En la sección 2.4.3 se describieron las características principales de este tipo de librerías.

4.4.5. Electron

Es un framework que permite escribir aplicaciones de escritorio multiplataforma usando HTML, javascript y CSS.

Esta herramienta nos permitirá distribuir la aplicación de una manera mas elegante que en una carpeta con código y un archivo index.html para ejecutarlo en un navegador local. A los ojos de los usuarios el producto final es una aplicación nativa corriendo transparentemente una implementación mínima del navegador Chromium, solucionando así también posibles problemas de compatibilidad con algunos navegadores.

4.5. Manejo del espacio 3D

4.5.1. Eventos de usuario

El espacio 3D ocupa la mayor parte de la pantalla, y es donde se espera que el usuario realice la mayor parte de interacciones posibles. Se busca que el usuario pueda manejar la escena mediante el uso del mouse, por lo tanto, al inicializar la aplicación se setean EventListenners a la ventana para cada tipo de acción con el mouse. Esto significa que la aplicación estará pendiente de todos los movimientos del mouse dentro de la escena. En particular: se tiene en cuenta el movimiento de la cámara y en qué modo se encuentra la aplicación

Los evenos definidos son los siguientes:

- Para el click izquierdo:
  - Si el mismo se presiona y suelta en el mismo lugar:
    - Si se encuentra en modo agregar nodos, y se hace el click encima de un punto de una grilla definida, se agrega el nodo
    - Si se encuentra en modo agregar barras, y se hace el click encima de un nodo, se selecciona el mismo para agregar una barra desde o hacia él
    - Si se encuentra en modo seleccionar, y se hace el click sobre un nodo o barra, se selecciona el elemento para modificar sus propiedades
  - Si el mismo se presiona y se arrastra, se rota la cámara, siempre apuntando al centro de la escena.
- Para el click derecho, se desplaza la cámara en dirección a donde se arrastre
- Para el scroll del mouse, se hace zoom in o zoom out dependiendo de la dirección del movimiento
- Para el movimiento del mouse, cuando no se presiona nada, se resalta en color celeste los nodos o barras a los cuáles se les hace Hover.

4.5.2. Adición, sustracción y transformación de objetos

El espacio 3D desarrollado se implementó como un objeto de ThreeJs denominado Scene, el cual provee operaciones add() y remove() para agregar y eliminar objetos del mismo. Internamente, una Scene contiene una gran cantidad de atributos, entre los que se encuentran la posición, rotación, escalamiento por coordenada, y una lista de objetos en donde se almacenan los elementos que conforman la escena.

Una vez definida la escena se agregan los primeros elementos: se utiliza el objeto GridHelper provisto por ThreeJS para definir una grilla auxiliar y se agregan 3 vectores definiendo los ejes x, y, z.

El resto de los objetos que se agregan a la escena se definen como objetos Mesh de ThreeJS, definidos por una geometría y un material:

- La geometría de un objeto define su figura geométrica, define si se trata de un cilindro, una esfera, etc...
- El material de un objeto define cómo se ve el mismo en pantalla, es decir, su color, transparencia, escalamiento, etc...

Para representar los nodos se decidió utilizar esferas y para representar las barras se decidió utilizar cilindros. Esta decisión se basa en que son figuras geométricas fácilmente escalables, es decir, si se quiere cambiar el tamaño de un nodo o barra, solo basta con cambiar el radio de su geometría. Por ejemplo, cuando se selecciona un nodo para modificar sus propiedades, el mismo aumenta su tamaño y cambia su color. Esto se logra siguiendo el siguiente proceso:

1. Se obtiene el objeto que se quiere modificar de la lista de objetos de la Escena.
2. Se genera una nueva geometría, en este caso, una esfera más grande.
3. Se genera un nuevo material, en este caso, el mimso que existía pero con un color diferente.
4. Se asignan el material y la geometria nuevos al objeto obtenido en el paso 1.
5. Se renderiza la imagen.

El renderizado se ejecuta cada vez que se realiza una acción en la escena. Esto significa que cada vez que un usuario modifica la estructura, los cambios quedan inmediatamente reflejados en el espacio 3D. De esta manera logramos una experiencia fluida y totalmente interactiva de dibujado donde la manipulación de la estructura se vuelve una tarea sencilla e intuitiva.

4.5.3. Manejo de la cámara

Una de las prestaciones más grandes que fue percibida en ThreeJS al momento de la investigación fue el sencillo manejo de la cámara. Three provee de un objeto denominado Camera, el cuál se define asignando el tamaño del viewport, hacia dónde apunta, los planos «near» y «far» que determinan qué objetos se ven dependiendo de su distancia a la cámara, etc. Incluso permite definir el tipo de perspectiva en que se visualizará el resto del espacio. Una vez definida, se setea la misma a la escena, logrando de esta sencilla manera obtener la visualización del espacio 3D.

Sin embargo, hasta el momento sólo se colocó la cámara en el espacio, sin resolver aún el problema del movimiento de la misma. Aquí es donde entran en juego los llamados «Controls». ThreeJS ofrece en su página Web y de manera libre diferentes tipos de controles para la cámara, es decir, movimientos que pueden ser asignados a la misma. En particular, OrbitControls cumple con todas las características que se buscan para la interfaz

diferentes tipo de «Controles» que pueden ser asignados a la escena para definir el movimiento de su cámara. En particular, se utili hablar de elcambio de ejes de coordenadas para que el queda arriba

hablar como se mueve la camara tratar de ver el js asqueroso y deducir matematicamente como se mueve con respecto al espacio 3d para dibujarlo o mostrar un diagramita

4.5.4. Trazado de rayos e intersecciones con objetos

hablar de como resolvimos la interseciion del 'click'con el rayo de la camara para seleccioar objetos

hacer formulacion matematica

hablar de las cosas que three automatia y de como qedo resuelto

que compontes se enargan de esto

4.5.5. Performance

hablar de los probelmas que encontramos al probar la torre eiffel

describir que se investigo y se hicieron camnios

hablar de los geometry de los objetos

de sacar a seleccion al hacer hover

de otras performances que se hicieron

hablar finalmente de otras mejoras que sepodrian hacer pero resultarian innecesarios porque serian muy complejos par este proyecto en el que los estudiantes nunca vana hacer una estructura tan grande

4.6. Manejo de datos

hablar de como es la estructura que guardamos

4.6.1. Entrada de información (dibujado e importación)

que informacino guardamos de cada elemento y porque

en que momento agregamos cosas al mdoelo

hablar del dibujado

hablar de la importacion(como reemplaamos el modelo)

hablar de abrir y guardar proyectos(como reemplaamos el modelo)

4.6.2. Mantenimiento de la estructura durante el proceso de dibujado

aca hablamos de como manipulamos los objetos, modificar y eliminae

hablar de mantener la consistencia, al abrir un modelo nuevo, guardar, abrir, etc...

4.6.3. Almacenamiento de la estructura

hablar de que el modelo se va guardando en una variable javascript

hacer un estudio sobre que tan eficiene=te es y si la memoria del navegador 'da'para almacenar algo asi

mostrar un mini ejemplo y exactamente qué se guarda

4.6.4. Salida de Datos

hablar de lo que se genera desde la ui

como se genera, proceso que ace el usuairo para generarlo

validaciones que se toman en cuenta

como manipulamos el modelo ara generar el txt

4.7. Análisis de resultados del Core

4.7.1. Generación de resultados

hablar sobre qué genera el core

mencionar las cosas que agrega el texto

4.7.2. Introducción de datos en la UI

como se ingresan

como se modelan y almacenanesos datos - deformedmodel

4.7.3. Visualización

qué se ve

hablar de las opciones que se tienen

como hicimos el sitcheo entre deformada en indeformada

como hicimos el escalamiento

como hicimos el colorado

5. Resultados obtenidos

5.1. Comparación IETFEM con y sin UI

5.1.1. Análisis del impacto en la usabilidad

hablar de opinion de estudiantes, posiblemente en la idm

mostrar unto por punto en que aspectos se mejoraron

5.1.2. Análisis del impacto en el tiempo de ejecución

usar ietfem viejo y nuevo y calcular tiempos

ver si es muy dificil hacer un mini servidor para hacer una comparaciond de tiempos mejor

5.2. Casos de prueba

5.2.1. Estudio de casos de pequeño porte (Torre pequeña)

hablar de que comenamos con ese

se utilio para realiar la primera integracion con el core

sedescurbrieron probelmasde ejes y se resolvieron facilmente

5.2.2. Estudio de casos de mediano porte (Grúa)

se intento realiar el caso inicialmente como una prueba de stress

no se encontraron problemas

cuando se constato que funcionaba bien, se decidio utiliar un caso mas grande

utiliado en la idm para mostrar le funcionamiento

hablar de que ya se considera exitoso que funcione bien para la grua ya que lso estudiantes nunca van a hacer algo tan grande

5.2.3. Estudio de casos de gran porte y pruebas de stress (Torre Eiffel)

comentar que se decidio hacer latorre eiffel para ver como respondia el programa

hablar del trabajo de la perfomrmance y memoria

cuanto se mejoro luego de los arreglos

importancia de que ande 'perfecto'ya que es un caso inalcanable

6. Conclusiones y trabajo futuro

6.1. Conclusiones

hablar si las estimaciones y el esfuerxo fueron acertados

si se cumplieron los objetivos

evaluar la herramienta

6.2. Trabajo a futuro

6.2.1. Trabajo en el motor

que se puede agregar en el motor

porticos

osibilidad de migrar a otro lenguaje e integrar en un solo proyecto con la ui

6.2.2. Trabajo en la interfaz

agregar cosas que ya se pueden hacer enel core

mejoras de performance

otros 'chiches'que tienen programar comerciales

delegar responsabilidades a aotra aplicacion

6.2.3. Despliegue de la aplicación

hablar del servidor

como funcionaria con servidor y porque no se hio asi

donde se podria alojar

mejoras que implicari en el sistema

7. Anexos

mini Manual de uso

ejemplos de estructuras

modelo de dominio

casos de uso

diagramas de flujo

diagramas de arquitectura

mas info threejs

masinfo otros proyectos similares

otras cosas XD