Universidad de Valladolid

E. T. S. DE INGENIERÍA INFORMÁTICA

Ingeniería Informática

Ion Implant Simulator GUI

Alumno: Álvaro Fernández Díez

Tutor: Dr. Jesús M. Hernández Mangas

1. Introducción

Para poder entender el planteamiento del proyecto comenzaremos describiendo en qué consiste el proceso de implantación iónica: "Uno de los procesos de la Tecnología Microelectrónica (fabricación de Circuitos Integrados) cuya finalidad principal es la impurificación precisa del material, es la implantación iónica, sin que esta sea su única utilidad.

La implantación iónica consiste en la proyección contra el material (blanco) de un haz de iones (proyectiles) acelerados con suficiente energía para penetrar más allá de las capas superficiales. El sistema que realiza esta operación es el implantador iónico. Es obvio, dadas las dimensiones actuales de los dispositivos y otros elementos electrónicos que forman los circuitos integrados, que el control de la distribución de los iones que han penetrado en el material (perfiles de implantación) es imprescindible. Cómo conocer estos perfiles es uno de los temas de máximo interés en nuestros días. Como en tantos otros casos, puede accederse al resultado a través del análisis experimental de los perfiles de implantación (caracterización) o mediante modelos matemáticos de naturaleza predictiva que, actualmente, pasan forzosamente por el ordenador (simulación).

Las metodologías usadas en el cálculo de perfiles por ordenador son varias: Principalmente Dinámica Molecular y Aproximación de Colisiones Binarias. La primera es de gran exactitud, pero imposible de aplicar a sistemas con un gran número de partículas y con las energías de implantación actuales por el tiempo de cálculo requerido. La Aproximación de Colisiones Binarias (BCA) no es tan exacta, pero la modelización de la trayectoria del ión en el material mediante sucesivas colisiones del mismo con un número muy reducido de átomos del blanco no es tan costosa en tiempo de cálculo y sus resultados son adecuadamente útiles a tecnólogos y diseñadores."[1]

La motivación del desarrollo y empleo de un simulador de implantación iónica en el campo de la investigación es obvio, pero para comprender la importancia del mismo en el campo de la industria haremos referencia a esta cita:

"El desarrollo de la tecnología microelectrónica a lo largo de los últimos 30 años ha sido posible gracias al esfuerzo realizado en la investigación básica de los procesos tecnológicos que forman parte de ella. Sin embargo, esta investigación básica tiene un coste muy elevado dado que requiere de sistemas de investigación muy especializados. En un intento, de abaratar costes y de entender qué ocurre dentro del semiconductor, se ha desarrollado a lo largo de estos años, la simulación de los procesos tecnológicos y de los efectos físicos que conllevan.

Cuando se desarrolla un simulador, se intenta solucionar un problema dado, para poder diseñar y fabricar dispositivos más rápidos, más pequeños y con menos problemas. Por ello, las soluciones que se aportan pueden carecer de un sentido físico estricto del problema planteado, empleándose un modelo fenomenológico que tenga un conjunto de parámetros ajustables, que permita obtener las soluciones para cada tipo de problema. Esta aproximación está justificada desde el punto de vista de la industria, dado que esta progresa gracias a la fabricación y venta de los dispositivos y circuitos diseñados."[2]

Este último tipo de simulación es el que implementa el simulador desarrollado por el departamento de electrónica y consecuentemente el que captará nuestro interés.

Una vez descrito el problema de la implantación de iones y su interés en el área de la industria y la investigación, podemos centrarnos en el objetivo central del proyecto: el desarrollo de un front-end o interfaz gráfica para un simulador de implantación de iones de línea de comandos para plataformas Windows y Unix.

El simulador implementado por el departamento de electrónica está programado en C++ y se ejecuta en la consola de comandos, bien sea en una box Unix en las diferentes versiones para este sistema operativo y derivados, como Linux o Solaris. También existen versiones para Windows, compartiendo con sus correspondientes versiones de Unix el modo de ejecución y entrada de parámetros. Sobre este último punto, la entrada de parámetros, el único método posible es a través de la especificación de un fichero de definición de una simulación pasado como argumento en la línea de comandos al ejecutar el simulador.

El fichero de especificación de la simulación es un documento de texto plano que consta de tokens (palabras clave) que permiten definir valores para cada tipo de parámetro de la simulación.

Una vez iniciado el simulador, alimentado con el fichero de simulación correspondiente, se inicia un proceso de lectura y generación de tablas necesarias para la simulación especificada en el fichero de entrada. Las tablas generadas son empleadas por el simulador en el proceso de implantación y son refinadas con cada simulación, lo cual indica que dichas tablas pueden ser reutilizadas en otras simulaciones que compartan características parecidas a la actual, como pueden ser los átomos de los implantes. Tras la generación de las tablas, se comienza el proceso de implantación de iones que como ya se ha explicado anteriormente consiste en el lanzamiento de iones contra un material o blanco. Mientras discurre este proceso de implantación iónica, se van generando una serie de ficheros con resultados que contienen varias columnas de coordenadas que describirán las gráficas de resultados.

[1] Simulación de la Implantación iónica en semiconductores. Jesús M. Hernández Mangas. Valladolid, 19 de abril de 2000
[2] Simulación de la Implantación iónica en semiconductores. Jesús M. Hernández Mangas. Valladolid, 19 de abril de 2000

Con todo lo explicado hasta ahora deducimos que al finalizar la simulación tenemos una serie de tablas nuevas y un conjunto de ficheros de resultados que nos servirán para representar diferentes aspectos de la simulación y del comportamiento del material a la implantación de iones.

La definición de parámetros de una simulación es un proceso tedioso ya que se realiza mediante un editor de texto y debemos tener en cuenta siempre el formato y conocer los tokens para poder especificar todos los aspectos de la simulación, con lo que la edición o modificación de dichos parámetros se presenta no menos tediosa debido a la labor de localizar primero en el fichero qué es lo que queremos modificar. La visualización de resultados no es menos ardua, ya que para poder ver un resultado habrá que extraer las columnas deseadas del fichero e insertarlas en otro para después abrir este fichero con gnuplot. Con todos estos inconvenientes nace la idea de desarrollar un front-end gráfico para el simulador que facilite la definición y visualización de resultados.

El front-end gráfico debería ser desarrollado en VC++ y permitiría trabajar con varias simulaciones de forma intuitiva y sencilla. Además de la visualización de resultados debería permitir realizar simulaciones en la misma máquina y en máquinas remotas para poder obtener ventaja de la ejecución en paralelo. Con éstos y otros requisitos y restricciones se ha desarrollado “Ion Implant Simulator GUI” (iisGUI de ahora en adelante).

iisGUI permite la especificación de parámetros de simulación de forma visual a la vez que muestra la estructura tridimensional del material que se está definiendo. La simulación remota se realiza mediante el protocolo de comunicación SSH, tanto para ejecutar el simulador en la máquina seleccionada como para la transferencia del simulador y ficheros de tablas y resultados. Como existen diferentes versiones del simulador para las distintas plataformas y sistemas operativos, iisGUI implementa un control de versiones con dicho cometido que permite controlar qué versión del simulador se empleará en cada simulación mediante la selección de la misma antes de cada proceso de implantación.

La visualización de resultados sólo se ha implementado en 2D por falta de tiempo, quedando la representación 3D para futuras versiones. Se permite la visualización de varias curvas en la misma gráfica con el cometido de comparar diferentes resultados, pudiendo seleccionar dos columnas de cada fichero de resultados para la representación gráfica.

Sobre el control de simulaciones, además de las operaciones típicas de abrir, cerrar, cambiar propiedades e iniciar/detener simulación, se permite el duplicado de simulaciones con el objetivo aprovechar las características de una de ellas para la definición de otras con similares parámetros, teniendo que cambiar sólo los parámetros que difieran y ahorrándonos la definición de las propiedades comunes.

iisGUI cuenta además con un sistema de ayuda contextual que facilita la compresión de los diferentes parámetros de simulación y el resto de características y operaciones del front-end.

En el Capítulo 2, se detallará el objetivo central del proyecto, explicando de forma extensiva las características del mismo y los diferentes objetivos particulares.

Se detalla en el Capítulo 3 el problema de la simulación de implantación iónica, haciendo especial énfasis en los datos de entrada y de salida del simulador, lo cual servirá como marco conceptual de cara a realizar el análisis y diseño de la herramienta.

En los Capítulos 4, 5 y 6 se desarrollan Análisis, Diseño y Pruebas de la solución aportada.

Finalmente las conclusiones y trabajo futuro ocupan el Capítulo 7, donde se describirán las posibles mejoras y cambios que se pueden introducir en la solución, que formarán parte del mantenimiento perfectivo.

2. Objetivos

Para la comprensión de los objetivos del proyecto, es necesario conocer antes el paradigma del simulador del que dispone el departamento de electrónica para el que se ha desarrollado la solución.

A modo de resumen se detalla a continuación la propuesta de proyecto que viene a introducir parte del paradigma del simulador y algunos de los requisitos propuestos para la solución:

"Actualmente el Departamento de Electrónica de la Universidad de Valladolid tiene en funcionamiento una potente herramienta de simulación de procesos tecnológicos para la fabricación de circuitos integrados y otros dispositivos electrónicos. En particular, el módulo de simulación de implantación iónica en semiconductores (IIS) está paralelizado para su ejecución en un cluster heterogéneo de computadores. La interfaz de usuario actual está implementada en base a ficheros de entrada con el problema a resolver y los resultados se obtienen en ficheros de salida con datos numéricos que hay que representar y analizar posteriormente.

Se pide el desarrollo de un interfaz gráfico de usuario para el S.O. Windows, para la entrada de datos, iniciación, recogida y visualización de los resultados. Además éste programa maestro deberá permitir la ejecución distribuida de esta aplicación de cálculo intensivo. Se pretende dar forma a una interfaz gráfica para Windows con entrada de datos con múltiples diálogos que sea capaz de lanzar la ejecución de una aplicación de cálculo intensivo en diferentes hosts remotos (heterogéneos) o en local, permitiendo la recolección y representación gráfica de las salidas, tanto en tiempo de ejecución como cuando finalice la aplicación."

Con este planteamiento queda definido más o menos el paradigma inicial del proyecto y a partir de esto vamos a tratar de desglosar las diferentes partes del planteamiento y de la solución propuesta.

2.1 El Simulador:

Comenzamos explicando detalladamente las características de la herramienta de cálculo intensivo del Departamento de Electrónica.

Se trata de una herramienta desarrollada en C++ para la consola de comandos, por lo tanto sin interfaz gráfica de usuario. La introducción de datos se realiza mediante la edición de ficheros con palabras clave (tokens) que especifican los diferentes parámetros de cada simulación. Para que el simulador tome dicho fichero de entrada, se deberá especificar como parámetro en la línea de comandos al ejecutar el simulador.

Como entrada también forman parte unas tablas que se emplearán en el proceso de simulación y que ya están definidas previamente, por lo que no se necesitará la posibilidad de definirlas en la interfaz gráfica desarrollada. A parte de estas tablas predefinidas, el simulador genera otras cuando se inicia y que se emplearán durante el proceso de simulación también como entrada.

Algunas versiones del simulador permiten la especificación de los directorios de tablas mediante la definición de variables de entorno.

La salida de datos se realiza también por medio de ficheros, bien sean numéricos, que será los que acaparen nuestra atención o de otros tipos, que no serán necesarios para nuestra solución, pero sin embargo se deberán poder visualizar al igual que los numéricos.

Los ficheros de datos numéricos están divididos por columnas y cada uno de estos números representa una coordenada cartesiana. Cada columna se puede representar en un eje de coordenadas, y cada fichero contiene dos o más columnas, por lo que se podrán obtener varias representaciones bidimensionales haciendo pares de columnas y cambiando su representación en los diferentes ejes X e Y, así como varias representaciones tridimensionales en caso de haber tres o más columnas.

Finalmente queda decir que existen diferentes versiones para cada plataforma y sistema operativo, de forma que la simulación paralela se puede realizar en todas ellas empleando para cada una la versión apropiada del simulador.

2.2 Objetivos de la solución:

2.2.1 Definición de parámetros de la simulación:

La definición de parámetros debe hacerse en base a una hoja de propiedades con diferentes fichas que dividan los mismos en distintas agrupaciones. La definición de parámetros deberá realizarse de forma totalmente visual, con listas desplegables en caso de disponer de múltiples opciones de selección.

Existe un apartado de definición de átomos en el que deberá mostrarse una tabla periódica de elementos para que el usuario elija de forma intuitiva el elemento que desee.

También deberá mostrarse una representación tridimensional con posibilidad de rotaciones, traslados y zoom de la estructura cristalina del material que se defina en el apartado de material del blanco.

Además deberá mostrarse la salida estándar y de error que el simulador imprima durante el proceso de simulación.

Con todos estos requisitos, el objetivo es facilitar la labor del usuario a la hora de definir, cambiar y visualizar los diferentes parámetros de la simulación, así como hacer totalmente transparente para el usuario todo el proceso.

2.2.2 Generación del fichero de entrada para alimentar al simulador

Una vez definidos los datos o parámetros que caracterizan a una simulación, se deberá crear un fichero de entrada para alimentar al simulador. Dicho fichero deberá tener el formato adecuado para que el simulador lo comprenda e interprete correctamente.

El objetivo de esto es servir como puente entre la definición visual de los parámetros, su representación y almacenamiento interno de la interfaz gráfica y la entrada de parámetros en el simulador o comunicación con el mismo. Este paso es totalmente transparente para el usuario y sirve, como ya se ha dicho, de nexo de unión entre la interfaz y la entrada del simulador. Dicho fichero de entrada es el que anteriormente se debía generar y editar de forma manual para especificar las simulaciones cuando no se contaba con la herramienta visual o interfaz gráfica.

2.2.3 Ejecución de la simulación

Como puede leerse en el planteamiento general del proyecto, se deberán poder realizar dos tipos de ejecución o simulación, local y remoto.

Con simulación local nos referimos a la ejecución del simulador en la misma máquina desde la que está operando en usuario. Para este tipo de simulación basta con crear un nuevo proceso del simulador con la ventana oculta por motivos estéticos y redirigir la salida estándar y de error del mismo para poder visualizarla desde la interfaz cómodamente y controlar todo el proceso de simulación.

Por simulación remota se entiende aquella en la que el simulador se lanza en una máquina diferente a la que está utilizando el usuario y consecuentemente de ejecución en paralelo. Como requisito se especifica que la comunicación entre cliente (máquina que opera el usuario) y servidor (máquina en la que se ejecuta el simulador) sea por el protocolo de comunicación SSH. Con estas características, es preciso el desarrollo de una librería de comunicaciones SSH que ofrezca tanto ejecución en shell remoto para poder lanzar el simulador como transferencia de archivos empleando el subsistema de transferencia de ficheros segura SFTP. Gracias a la implementación de esta librería tendremos absoluto control sobre el desarrollo tanto de la simulación como de la conexión con el host remoto y ganaremos en fiabilidad, eficiencia y reusabilidad en contraposición al empleo de herramientas externas de conexión SSH, que de cualquier forma y después de analizar el paradigma no sería posible la integración completa.

Cabe notar y es obvio que antes de poder realizar éste proceso de simulación será necesario acondicionar el directorio destino donde se realizará la simulación, bien sea local o remoto, con la previa transferencia del ejecutable del simulador (que dependerá de la versión seleccionada) y de las tablas de entrada así como del fichero de parámetros o de definición de la simulación. Con esto en mente, debe existir la posibilidad de definir una lista de las tablas de entrada que se desean transferir, además de la especificación de un script o batch que será opcional y tendrá como objetivo la definición de las variables de entorno relativas al directorio de tablas entre otras cosas.

Como objetivo buscamos la ejecución del simulador en una máquina remota o en local de forma transparente e independiente, de forma que el usuario no tenga que preocuparse nada más que de aportar los datos necesarios para la conexión en caso de efectuar una simulación remota, disponiendo en todo momento de la salida del simulador tanto estándar como de error.

2.2.4 Recogida de resultados

La recogida de resultados es un proceso que debe realizarse cada cierto intervalo de tiempo configurable y que tiene como objetivo la transferencia de los ficheros de resultado generados por el simulador durante el proceso de implantación bien sea desde un host remoto a la máquina local o bien desde un directorio a otro en local.

En la simulación remota se realizará utilizando el protocolo de transferencia segura de ficheros SFTP, y en local con una simple copia de archivos. Todo este proceso debe ser completamente transparente y el usuario únicamente se dará cuenta del proceso a medida que se actualicen de manera automática todos los resultados que esté visualizando en ese momento. Para que este proceso gane en eficiencia, solamente se transferirán aquellos ficheros que hayan sido modificados, y además existirán unas listas de exclusión donde se podrán definir los resultados que no se desean transferir durante la simulación pero sí al final, donde se transferirán todos con independencia de que estén o no en esta lista de exclusión.

Como apéndice, cabe notar que además de los resultados, ha de poderse especificar una lista de tablas que se desean transferir cuando finalice la simulación; Dichas tablas se copiarán en el directorio de tablas local y se podrán utilizar en futuras simulaciones.

2.2.5 Representación de resultados

Con la representación de resultados nos referimos tanto a la visualización del contenido de los ficheros de resultados, esto es en texto plano, como a la representación gráfica de los mismos. Será necesario poder seleccionar las columnas que definirán cada uno de los ejes cartesianos de la curva, así como la posibilidad de visualizar varias curvas en la misma gráfica, bien sean del mismo fichero o de otros.

Con esto se pretende conseguir la comprensión del proceso de simulación y el mejor análisis de los resultados obtenidos, pudiendo ser comparados con previos resultados de otras simulaciones a fin de juzgar diversos parámetros relativos al material y al resto de los aspectos de una simulación.

2.2.6 Cuestiones generales

Además de los objetivos definidos con anterioridad, que son los esenciales, se han definido una serie de requisitos que serán importantes para la mejora de la escalabilidad del proyecto y la personalización de la interfaz entre otros aspectos.

La interfaz debe contar con un entorno multilenguaje que a priori deberá poderse elegir entre idioma castellano e inglés, no debiendo existir problemas para la futura inclusión de otros lenguajes.

Existirá la posibilidad de definir nuevas versiones del simulador mediante un cuadro de diálogo que permitirá el control absoluto sobre dichas versiones, pudiendo realizar el típico mantenimiento de altas, bajas y modificaciones. La especificación de la versión vendrá descrita por su nombre, número de versión, arquitectura, descripción y fichero ejecutable que la representa y que se empleará en el proceso de implantación.

Debe guardarse una lista con las simulaciones abiertas en el momento del cierre de la aplicación para que en la siguiente sesión pueda recuperarse el estado anterior al cierre. El objetivo de esto es muy claro, no tener que volver a abrir las simulaciones con las que se estaba trabajando en la sesión anterior.

También se podrán configurar algunos aspectos como el intervalo de actualización de salida del simulador y aspectos relativos a la estética de la interfaz gráfica.

El objetivo general de todo esto es facilitar el manejo de la interfaz de modo que sea muy intuitivo y sencillo, poniendo además a disposición del usuario de una completa ayuda contextual donde se explicará de manera objetiva cada uno de los aspectos del programa.

3. Marco conceptual

En este capítulo se intenta presentar el problema de la simulación de implantación iónica, haciendo especial énfasis en los datos de entrada y de salida del simulador.

En anteriores capítulos ya se ha presentado a grandes rasgos el paradigma de la simulación de implantación de iones así como del simulador para el que se ha desarrollado esta solución, de modo que en este capítulo trataremos de precisar de manera más objetiva ambos problemas y denotaremos los parámetros de entrada de los que hará uso el simulador y consecuentemente que deberá aportar el front-end gráfico y de la salida del mismo que deberá representarse en la interfaz gráfica.

4. Análisis y Diseño

4.1 Planificación Inicial

En este documento voy a exponer los aspectos relacionados con la planificación y seguimiento de la realización del proyecto. Este documento está estructurado en siete apartados, estando relacionado cada uno de ellos con un aspecto diferente de la planificación del proyecto.

- En el primer apartado se detallan los objetivos del mismo.
- El segundo apartado está destinado a especificar la metodología del proceso de desarrollo que voy a seguir y el modo de utilizarla.
- El tercer apartado explica la composición, distribución de tareas y límites de la organización.
- En el cuarto apartado se recogen características propias de este proyecto como su coste, la documentación que se va a generar, los recursos utilizados y algunos aspectos técnicos.
- La cuestión de los riesgos del proyecto es tratada en el apartado cinco, estando incluidos en este punto la lista de riesgos, su incidencia sobre la planificación del proceso y las acciones destinadas a mitigar su efecto.
- El sexto apartado incluye los diagramas de planificación temporal del proceso y la lista de recursos que se van a utilizar.
- Finalmente, el séptimo apartado recoge el plan de calidad, sus objetivos y las métricas de proceso y producto.

4.1.1 Objetivos del Proyecto

Ya se han comentado a lo largo de los diversos capítulos anteriores cuáles eran los objetivos de este proyecto y de igual modo se han mencionado las especificaciones requeridas antes y durante la elaboración del mismo. Por este motivo no vamos a repetir aquí la información que se ha brindado con anterioridad.

4.1.2 Metodología de Desarrollo

En este proyecto se va a utilizar el ***Proceso Unificado de Rational (RUP)***. Este modelo de proceso es iterativo, orientado a la arquitectura y guiado por los casos de uso. En dicho proceso se establecen 4 etapas (inicio, elaboración, construcción y transformación) a través de las cuales va a ir pasando el proceso. Cada etapa se descompone en iteraciones, siendo responsabilidad del jefe de proyecto determinar el número de iteraciones que se precisan en cada etapa. Cada iteración va a suponer la puesta en práctica de un mini ciclo de vida en cascada, donde cada etapa clásica de desarrollo (análisis, diseño, implementación, pruebas) supondrá una etapa del mini ciclo, realizándose estas etapas secuencialmente.

He decido realizar cada etapa de RUP en una sola iteración, ya que cada una de ellas es relativamente pequeña y no compensa generar excesiva documentación para proyectos de interfaz de usuario en los que la arquitectura del programa no suele ser compleja. De este modo, la herramienta se ha de construir a través de una primera etapa de iteración (centrada en obtener los requisitos), una segunda de elaboración (definir una estructura básica e implementar un primer prototipo y una primera interfaz del producto), una tercera de construcción (la versión beta de la herramienta ha de ser su resultado final) y una última de transición (a cuyo final se tendrá la herramienta definitiva y los manuales de la misma).

Cada iteración se descompone en una serie de actividades que se han de realizar por determinados actores, donde un actor significa un puesto de trabajo con unas responsabilidades bien determinadas en el proyecto. Las ***actividades*** de cada iteración deben hacerse en un determinado orden, orden que ha sido recogido en los diagramas de planificación (ver el apartado 6 de este documento). RUP también establece qué ***artefactos*** se necesitan para realizar cada actividad y qué artefactos debe producir cada actividad, artefactos que vamos a generar en el transcurso de este proyecto. Como único integrante del grupo de desarrollo, tendré la responsabilidad de asumir la tarea de todos los actores, de modo que algunas partes de la planificación y métricas relacionadas con los recursos humanos son innecesarias.

4.1.3 Estructura de la Organización

A continuación se expone una lista de los actores que existen en el proceso de desarrollo RUP y de las actividades que deben realizar en cada fase:

Actor	Actividades
Analista de sistemas	Encontrar actores y casos de uso (Fases de Inicio, Elaboración y Transición).
Arquitecto	Priorizar los casos de uso (Fases de Inicio, Elaboración y Construcción). Realizar el análisis de la arquitectura (Fases de Inicio, Elaboración y Construcción). Realizar el diseño de la arquitectura (Fases Elaboración y Construcción). Realizar la implementación de la arquitectura (Fases de Elaboración y Construcción.
Especificador de casos de uso	Detallar un caso de uso (Fases de Inicio, Elaboración y Transición)
Ingeniero de casos de uso	Analizar casos de uso (Fases de Inicio, Elaboración y Transición) Diseñar casos de uso (Fases de Elaboración y Construcción).
Diseñador de la interfaz de usuario	El diseñador de la interfaz de usuario va a encargarse de realizar el diseño de pantallas de la aplicación que se le mostrarán al usuario, así como las pantallas de los sucesivos prototipos generados durante el proceso de desarrollo, *más concretamente de las páginas que constara el sistema.*
Ingeniero de componentes	Analizar las clases (Fases de Elaboración y Construcción). Analizar los paquetes (Fases de Elaboración y Construcción). Diseñar las clases (Fases de Elaboración y Construcción). Diseñar los subsistemas (Fases de Elaboración y Construcción). Implementar los subsistemas (Fases de Elaboración y Construcción). Implementar las clases (Fases de Elaboración y Construcción). Realizar las pruebas de unidad (Fases de Elaboración y Construcción). Implementar pruebas (Fases de Elaboración y Construcción)
Ingeniero de pruebas	Planificar pruebas (Fases de Elaboración y Construcción). Evaluar pruebas (Fases de Elaboración y Construcción).
Ingeniero de pruebas de integración	Realizar las pruebas de integración (Fases de Elaboración y Construcción).
Ingeniero de pruebas de sistema	Realizar pruebas de sistema (Fases de Elaboración y Construcción).
Integrador de sistemas	Integrar sistemas (Fases de Elaboración y Transición)

Dado que yo soy la única persona que desarrollará el proyecto, haré el papel de todos los actores, de tal modo que la organización y distribución de tareas y responsabilidades no tiene sentido en este caso.

4.1.4 Características de Gestión del Proyecto

En este apartado vamos a considerar cuestiones cómo el coste del proyecto, los recursos (humanos, hardware y software) que en él se van a emplear y las restricciones o aspectos técnicos que van a estar presentes en el mismo.

4.1.4.1 Coste económico del proyecto

Las herramientas utilizadas en el proyecto son de un coste muy elevado, en gran parte debido a los requisitos del cliente. Como se trata de un proyecto docente, no se han adquirido las licencias de los mismos, lo que también queda justificado al analizar el objetivo principal del proyecto, ya que la herramienta no está destinada al usuario final, sino a su continuidad en el proceso de desarrollo, y en caso de querer obtener beneficio de su venta se deberá adquirir las correspondientes licencias de las herramientas de desarrollo empleadas.

4.1.4.2 Recursos Software y Hardware

Recursos Hardware:
PC: Empleado para el desarrollo de todo el proyecto y para generar toda la documentación correspondiente al proceso.

Recursos Software:
Herramientas:
Adobe Acrobat 6: Utilizado para generar la documentación y memoria del proyecto en formato PDF.
MS Project 2000: Para la representación gráfica del seguimiento y planificación de todo el proceso.
Racional Rose 2002: Herramienta CASE para desarrollos en UML y la elaboración de los respectivos diagramas de análisis y diseño.
STI Understand For C++: Software para facilitar la ingeniería inversa de proyectos en C++ y elaborar métricas del producto.
Visual Studio 2003: Empleado para la implementación de la herramienta
BCGControlBar Pro 6.74: Librería MFC que proporciona mejoras visuales para las aplicaciones desarrolladas en VC++
Parasoft C++ Test: Herramienta empleada para realizar las pruebas del producto
F-Secure SSH Server: Servidor SSH para la realización de algunas pruebas de funcionalidad del producto.

Lenguajes:
VC++: Lenguaje con el que se implementará la herramienta.
UML: Lenguaje de modelado. Lo emplearemos para construir los diferentes diagramas de análisis, diseño e implementación que se usarán durante el proyecto.

4.1.5 Gestión de Riesgos del Proyecto

4.1.5.1 Enumeración de Riesgos

Riesgos relacionados con el cliente
Posibilidad de que el cliente cambie las necesidades en el transcurso del proyecto.
Todos los riesgos de esta categoría son mayores, ya que al estar el simulador en pleno desarrollo es muy posible que cambien los requisitos, además de que tampoco han sido todos especificados en el momento de realizar el análisis.

Riesgos del Proceso
Aspectos del proceso:

- Puedo tener problemas con el proceso de desarrollo, ya que el proceso es iterativo y tengo poca experiencia con él.
- Puedo tener problemas con el proceso de desarrollo RUP, ya que es un proceso pensado para grandes proyectos, y genera mucha documentación que es superflua en algunos casos.
- Puedo tener problemas con la descomposición exacta de las tareas, ya que tampoco he hecho nunca una planificación en condiciones.
- Puedo tener problemas con las aproximaciones temporales que he hecho de la duración de las tareas, ya que tampoco tengo experiencia en anteriores planificaciones de proyectos de similares dimensiones y hay riesgo de cambio de requisitos.
- Puedo tener problemas, ya que no tengo estándares de documentación claros para el proceso de desarrollo.
- Puedo tener problemas al hacer las revisiones del proyecto, dado que tampoco tengo experiencia en este tipo de revisiones.

Aspectos técnicos:

- Puedo tener problemas al hacer el análisis y el diseño, ya que mi conocimiento del lenguaje de modelado UML, no es excesivamente amplio.
- Puedo tener problemas al hacer la arquitectura del software para el proyecto, ya que no tengo mucha experiencia en modelar la arquitectura de un sistema.
- Problemas, por las pocas tomas de datos que hay en algunas de las métricas utilizadas y que harán que muchas no pueda utilizarlas.

Riesgos tecnológicos:

- Problemas dados por no tener disponible el entorno en el que debe estar nuestra aplicación en el momento que lo necesitemos.
- Problemas porque se demanda una interfaz especial, que puede no ajustarse a los deseos del cliente.

Riesgos del entorno de desarrollo:

Riesgos dados porque las herramientas pueden no tener la funcionalidad (potencia), necesaria para lo que queremos hacer (VC++, BCGControlBar).
Riesgos dados porque las herramientas que utilicemos, pueden no estar disponibles en el momento en el que sean necesarias.
Riesgos dados porque no tengo formación en alguna de las herramienta que se van a utilizar para el desarrollo, como por ejemplo la librería de clases BCGControlBar.

Riesgos relacionados con la implementación en entorno distribuido de la herramienta:

Problemas por generar un volumen de tráfico o consultas alto, tal que el servidor no pueda darnos el servicio esperado.
Problemas por sobrecargar el servidor de la aplicación con un fuerte componente de cálculo que no sea capaz de procesar.
Problemas por una caída del servidor que deje a los clientes sin el servicio esperado.

4.1.5.2 Tabla de Riesgos sobre el estándar R.U.P.

La tabla hace referencia a los principales riesgos del proyecto.

Descripción	Prioridad	Impacto	Monitor	Responsabilidad	Contingencia
Puedo tener problemas con el proceso de desarrollo, ya que el proceso es iterativo y tengo poca experiencia en él.	Media	Sobre la planificación y el propio proceso	Planificador	De todos	Estudio de la situación y resolución de la misma.
Puedo tener problemas con el proceso de desarrollo RUP, ya que es un proceso pensado para grandes proyectos, y genera mucha documentación que es superflua en algunos casos.	Media	Sobre la planificación y el propio proceso	Planificador		Reducción del proyecto (tareas y artefactos), para adecuarse a la planificación (tiempos) establecida.
Puedo tener problemas con la descomposición exacta de las tareas, ya que tampoco hemos hecho nunca una planificación en condiciones.	Media	Sobre la planificación y el proceso	Planificador		Adecuación del proyecto a las necesidades del momento.
Puedo tener problemas al hacer las revisiones del proyecto, dado que tampoco tenemos experiencia en este tipo de revisiones.	Media	Sobre las revisiones de los artefactos desarrollados y el plan de calidad del proyecto	El responsable del plan de Calidad	El responsable del plan de Calidad	Intentar seguir reglas de revisión sencillas que permitan hacer las revisiones sin problemas.
Puedo tener problemas al hacer el análisis y el diseño, ya que nuestro conocimiento del lenguaje de modelado UML, no es muy amplio.	Medio	Sobre los modelos de análisis, diseño, implementación, arquitectura		Todos los miembros del equipo	Estudio en profundidad de las técnicas de desarrollo de modelos.
Riesgos dados porque no tengo formación en alguna de las herramienta que se van a utilizar para el desarrollo, como por ejemplo la librería de clases BCGControlBar.	Media	En las fases en las que se necesiten esas herramientas	Todo el equipo	Todo el equipo	Obtención acelerada de los conocimientos mínimos para manejar esas herramientas.

4.1.6 Planificación:

En este apartado se presenta la planificación temporal de la secuencia de actividades que componen el proyecto. La secuencia de acciones a realizar será la siguiente:

Definición de las actividades: La secuencia de actividades de la planificación coincide con el orden de la secuencia de actividades que se especifican para cada iteración de RUP, aunque considerando la extensión de la metodología, el tamaño relativamente reducido del proyecto y las restricciones temporales del mismo, algunas de esas actividades (actividades de diseño en la fase de inicio) no van a ser realizadas.
Secuenciación de las actividades: Dado que se seguirá RUP no tendrá mayor problema la secuenciación de las actividades
Estimación de la duración de las actividades: Teniendo en cuenta que las actividades de este proyecto van a ser relativamente cortas, se ha decido estimar su duración en horas, lo que supone emplear una granularidad más fina de lo habitual.
Desarrollo de la planificación:

Archivo de planificación en **Microsoft Project**

Control de la planificación:

- *Iteraciones del proyecto:* Cada iteración del proyecto coincidirá con una fase de RUP, por lo que el proyecto se va a realizar en 4 iteraciones, correspondiendo la primera a la fase de Inicio, la segunda a la fase de Elaboración, la tercera a la de Construcción y la cuarta a la de transición.
- *Artefactos generados:* En RUP se especifican los artefactos necesarios para realizar una actividad ("entrada" de la actividad) y los artefactos generados por la actividad ("salida" de la actividad). En este proyecto vamos a intentar ser fieles a RUP en la medida de lo posible, por lo cual vamos a tratar de generar todos los artefactos de las actividades que realicemos.
- *Hitos del proyecto:* Los hitos principales del proyecto van a coincidir con la actividad RUP que marca el final de una fase (dado que las iteraciones del mismo coinciden con las fases).
- *Fecha de Entrega:* La fecha de comienzo de la planificación es el 26 de Junio de 2003, mientras que el comienzo real del proyecto, es decir, el comienzo de la primera actividad planificada será el 5 de Julio de 2003. Por otra parte, no se han fijado fechas límite para la entrega del producto final dada su complejidad aunque inicialmente se ha estimado un límite de un año.

Especificando un poco más las distintas actividades:

CONCEPTO

- **Definición de requisitos**
  - *Requisitos de los usuarios:* En este apartado se especificarán las necesidades que tiene el usuario y que tiene que cumplir nuestro proyecto.
  - *Requisitos del contenido:* En este apartado se decide qué información necesitan los usuarios y cómo prefieren obtener acceso a ella. Se determinará exactamente a qué parámetros de simulación tendrá acceso el usuario.
  - *Requisitos del sistema:* En este apartado se definirá aspectos relativos al sistema, como puede ser:
    - ¿Cómo se va a efectuar la comunicación con los ordenadores remotos?
    - Sistema operativo que se va utilizar...
- **Desarrollo del Plan Proyecto**. En esta tarea se realiza la planificación del proyecto, por lo tanto se va a separar el proyecto en tareas.
- **Definición de la Funcionalidad Específica**. En este punto se toman los requisitos de los usuarios de manera que se expresen como funciones para los desarrolladores. Al terminar esta tarea se obtiene una serie de funciones específicas que definen el comportamiento del sistema.

ANÁLISIS

- **Análisis del Sistema**. La metodología que se va utilizar para realizar el Análisis del Sistema será UML, esta metodología define una serie de pasos:
  - *Casos de uso*
  - *Diagrama de Clases*
  - *Diagrama de Transición*
  - *Diagrama de Estados*
  - *Diagrama de Actividades*

DISEÑO

- **Diseño de la Interfaz de Usuario**. En esta parte se imaginará conceptualmente cuál será el aspecto de la interfaz gráfica. De manera que quede definido el aspecto, la funcionalidad y los vínculos de datos que va a tener el proyecto.
- **Estudio de las herramientas de desarrollo**. En este punto se deberán de obtener los conocimientos suficientes para poder desarrollar la aplicación. Las herramientas que se van a utilizar son el entorno de desarrollo Visual Studio .NET 2003 y la librería de clases BCGControlBar Pro.

DESARROLLO

- **Desarrollo de la interfaz**
  - *Creación de la interfaz.* En esta tarea se va a definir el aspecto general que va a tener el front-end del proyecto. De esta manera se van a dar las directrices generales para que el aspecto de la aplicación sea uniforme.
  - *Desarrollo de la interfaz.* En esta parte se realizará fase de codificación y desarrollo del front-end.
- **Pruebas de la Aplicación**. Una vez que se ha codificado la Aplicación, se procederá a realizar pruebas para ver su correcto funcionamiento.

DOCUMENTACIÓN

- **Documentación del proyecto.** En este punto se recapitulará toda la información relativa a la realización del proyecto. La documentación se dividirá en los siguientes apartados:
  - *Documentación del Análisis*
  - *Documentación de la Fase de Diseño*
  - *Documentación de la Fase de Desarrollo y Pruebas*

4.1.7 PLAN DE CALIDAD: MÉTRICAS DE PROCESO Y DE PRODUCTO.

PROPOSITO DEL PLAN

Seguimiento del Proyecto, para asegurar una calidad alta tanto en el proceso de desarrollo, como en los productos que se obtengan de este. Disminuyendo de esta forma el tiempo que se debe dedicar a la resolución de fallos y defectos, al ser detectados estos en frases tempranas del proceso de desarrollo.
Definiremos también unos factores clave para el proyecto que serán los siguientes:

- Facilidad de uso.
- Portabilidad.
- Corrección.
- Fiabilidad.
- Flexibilidad.
- Integridad.

GESTIÓN

Se harán revisiones para comprobar la corrección y la calidad de varios de los artefactos obtenidos durante el ciclo de desarrollo de la aplicación. Estas revisiones serán de dos tipos:

- **Revisiones informales:** Estas revisiones no estarán programadas y se producirán cuando se necesiten, el propósito de estas revisiones es hacer más fluido el proceso de desarrollo, alargándolo lo menos posible, para estas revisiones se evaluará junto con el tutor del proyecto lo que se está haciendo en ese momento, comparando opiniones y buscando nuevos enfoques a los problemas.
- **Revisiones formales:** para estas revisiones, será preciso reunirse con el tutor y se evaluará el artefacto correspondiente, para ello se hará un estudio de este artefacto y como resultado generar una documentación de la revisión donde se explicarán las desviaciones detectadas y las medidas que se tomaron para corregirlas.

TAREAS

Las tareas serán las revisiones formales que se hagan (y más cosas si se desea). Y las mediciones de calidad de los productos que se hayan obtenido.

RESPONSABILIDAD

La responsabilidad de las mediciones recaerá sobre el responsable del plan de calidad (formado por el tutor del proyecto y por mí). La responsabilidad de las revisiones también recaerá sobre el responsable del plan de calidad que actuará de moderador en las mismas.

REVISIONES Y AUDITORIAS

Se hará una revisión de cada artefacto de interés que se haya desarrollado durante el proceso, cuando este acabado. Para cada artefacto, se especificará:

- **Propósito de la revisión**: el propósito será encontrar posibles desviaciones / errores dentro del artefacto.
- **Identificación de lo que fue revisado:** Identificaremos el artefacto o característica que este que queremos revisar.
- **Identificar lo que se descubrió y cuáles fueron las conclusiones:** Aquí expondremos todas las posibles desviaciones del producto sobre los requisitos y las conclusiones que se toman de ello.

El informe será breve de una página a lo sumo. Se limitará el debate, para que la revisión no sea larga. Se enunciarán todos los problemas que hayan sido encontrados, pero no se intentará resolver todos los problemas que se propongan. Las revisiones deberán ser preparadas por anticipado (estudio previo de lo que se va a revisar). Y se repasarán las revisiones posteriores (si las hubiese).

PRUEBAS

Se harán tres tipos de prueba en el proyecto:

- **Pruebas de unidad:** nos encargaremos de probar cada modulo del software. En este caso, consideraremos como módulos los escenarios de los casos de uso.
- **Pruebas de integración:** nos encargaremos de probar el ensamblaje de los módulos del sistema, es decir, nos aseguraremos de probar los escenarios conjuntos de varios casos de uso del sistema.
- **Pruebas de interfaz:** Se van a realizar pruebas de utilización de la interfaz lo más exhaustivas posible. Dada su extensión y su modo de realización, no se incluirá documentación sobre ellas en el diseño de las pruebas, reflejándose sólo los errores hallados.
- **Pruebas de sistema:** comprobaremos que el producto funcione correctamente de forma conjunta con el resto de elementos del sistema (usuarios, hardware, software de soporte, etc.), en el que se encuentra la herramienta. Estas pruebas consistirán el probar el producto final en la/s plataformas que habrán sido preparadas para el proyecto, y con usuarios externos al equipo de desarrollo.

GESTIÓN DE RIESGOS

Al principio se hará una lista de riesgos, donde se identificarán los riesgos detectados que puedan afectar al buen desarrollo del mismo. Estos riesgos estarán clasificados por su importancia y por su efecto en el proceso de desarrollo, además se especificarán las posibles medidas a tomar si se produjese el riesgo, y quien sería el responsable de tomar esas medidas. Además, se evaluará cada riesgo y se decidirá si el riesgo se ha producido o no y si se deben tomar las medidas pertinentes.

MEDICIONES DE LOS PRODUCTOS OBTENIDOS, FACTORES DE CALIDAD

Se harán mediciones dentro del proceso de desarrollo para asegurar la calidad del producto final. Sobre los factores de calidad necesarios para la herramienta, tendremos criterios.

- **Facilidad de uso.** Criterios:
  - Aprendizaje sencillo: Número de pantallas de ayuda, facilidad de trabajo con menús, manuales detallados, etc.
  - Habilidad inicial necesaria: Elementos extraños relativos a la comunicación remota y programas Windows.
  - Habilidad de manejo necesaria: Si es fácil la recuperación de la herramienta ante errores (extraños o múltiples), etc.
- **Portabilidad.** Criterios:
  - Independencia de la maquina en la que va a ser ejecutado dentro del campo actual de plataformas soportadas (PC).
  - Independencia de la maquina con la que se realizará la comunicación remota.
    - Los criterios anteriores se cumplirán ya que la comunicación se realizará por el estándar de comunicación SSH.
  - Independencia del sistema operativo dentro de un rango de sistemas soportados (Windows 98, XP, 2k3, NT, etc.).
  - Independencia del software que se ejecute en la maquina cliente.
- **Integridad.** Criterios:
  - Acceso de los usuarios a nuestra herramienta.
- **Corrección.** Criterios:
  - Completitud: El sistema cumpla todos los requisitos del usuario.
  - Consistencia: El sistema cumpla correctamente esos requisitos.
- **Fiabilidad.** Criterios:
  - Tolerancia a errores. La aplicación responda bien ante los errores producidos por el usuario o el sistema.
- **Flexibilidad.** Criterios:
  - Facilidad de ampliación: La aplicación es fácil de ampliar (funcionalidad, numero de clientes, etc.).

Métricas relacionadas con el acoplamiento y la cohesión de las clases:

- **Número de Asociaciones:** Es el número de asociaciones no jerárquicas en las que participa una clase.
- **Número de Clases Asociadas:** Es el número de clases distintas de un asset que están relacionadas con una clase a través de asociaciones no jerárquicas o a través de relaciones de herencia en las que dichas clases desempeñan el papel de hijas de la clase.
- **Número de Parámetros de Método:** Es el número total de parámetros que hay en los métodos de una clase.
- **Número de Métodos Públicos:** Es el número total de métodos públicos de una clase.
- **Número de Métodos:** Es el número total de métodos de una clase.
- **Porcentaje de Métodos Heredados:** Es el porcentaje de métodos heredados entre los métodos de una clase.
- **Número de Atributos Públicos:** Es el número de atributos públicos de una clase.
- **Número de Atributos:** Es el número total de atributos de una clase.
- **Número de Hijos:** Es el número total de hijos que tiene una clase.
- **Número de Padres:** Es el número total de padres que tiene una clase.
- **MUI:** Es una métrica que nos indica si la clase el número de asociaciones no jerárquicas en las que participa una clase.
- **HNL:** Es el nivel del árbol de herencia en el que se encuentra una clase.
- **NMR:** Es el número de métodos renombrados por una clase.

- **NMI:** Es el número métodos heredados por una clase.

Descripción detallada de las métricas:

- **Número de Asociaciones:** Es el número de asociaciones no jerárquicas en las que participa una clase.
  - *Modo de cálculo:* Se suma una unidad por cada relación no jerárquica en la que participe la clase.
  - *Criterios de contabilización:* También se consideran relaciones no jerárquicas a las relaciones de agregación y de composición. Se considera que una clase participa en estas relaciones independientemente de si desempeña el papel de agregadora, agregante, componente o compuesta.
  - *Característica de calidad medida:* Acoplamiento de la clase. Teóricamente, a mayor número de relaciones en las que participe la clase, mayor acoplamiento existirá entre esta clase y el resto de las clases.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Ninguna.

- **Número de Clases Asociadas:** Es el número de clases distintas de un asset que están relacionadas con una clase a través de asociaciones no jerárquicas o a través de relaciones de herencia en las que dichas clases desempeñan el papel de hijas de la clase.
  - *Modo de cálculo:* Se suma una unidad por cada clase diferente del asset que está relacionada con la clase actual a través de asociaciones no jerárquicas. Se suma una unidad por cada clase hija de la clase actual.
  - *Criterios de contabilización:* No se considera para el cálculo del número de clases asociadas cuando una clase se relaciona consigo misma. Si una clase se relaciona con otra más de una vez a través de más de una asociación no jerárquica o de una asociación no jerárquica y una relación de herencia, sólo será contabilizada una vez.
  - *Característica de calidad medida:* Acoplamiento de la clase.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Ninguna.

- **Número de Parámetros de Método:** Es el número total de parámetros que hay en los métodos de una clase.
  - *Modo de cálculo:* Se suma una unidad por cada parámetro que haya en cada método de la clase.
  - *Criterios de contabilización:* Si n parámetros de n métodos distintos de la clase tienen el mismo nombre y tipo, se contabilizarán como n parámetros distintos.
  - *Característica de calidad medida: Métricas de implementación relacionadas:* Tamaño de la clase.
  - *Métricas del perfil utilizadas:* Ninguna.

- **Número de Métodos Públicos:** Es el número total de métodos públicos de una clase.
  - *Modo de cálculo:* Se suma una unidad por cada método público de la clase.
  - *Criterios de contabilización:*
  - *Característica de calidad medida:* Tamaño de la clase.
  - *Métricas de implementación relacionadas:* El número de métodos públicos de una clase será igual al número de métodos públicos propios de la clase más el número de métodos públicos heredados de otras clases. Si en el diagrama se sobrescribe o renombra un método público heredado una clase dentro de esa clase, sólo se contabilizará el método público sobrescrito.
  - *Métricas del perfil utilizadas:* Ninguna.

- **Número de Métodos:** Es el número total de métodos de una clase.
  - *Modo de cálculo:* Se suma una unidad por cada método de la clase.
  - *Criterios de contabilización:* El número de métodos de una clase será igual al número de métodos propios de la clase más el número de métodos heredados de otras clases. Si en el diagrama se sobrescribe o renombra un método heredado una clase dentro de esa clase, sólo se contabilizará el método sobrescrito.
  - *Característica de calidad medida:* Tamaño de la clase.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Ninguna.

- **Porcentaje de Métodos Heredados:** Es el porcentaje de métodos heredados entre los métodos de una clase.

  - *Modo de cálculo:* Se calcula el número de métodos heredados por la clase y se divide entre el número de métodos de la clase. El resultado de la división se multiplica por cien.
  - *Criterios de contabilización:* Si un método heredado por la clase es modificado dentro de la propia clase, ya no se contabilizará dicho método como heredado.
  - *Característica de calidad medida:* Herencia de la clase.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Número de Métodos.

- **Número de Atributos Públicos:** Es el número de atributos públicos de una clase.
  - *Modo de cálculo:* Se suma una unidad por cada atributo público de la clase. *Criterios de contabilización:* El número de atributos públicos de una clase será igual al número de atributos públicos propios de la clase más el número de atributos públicos heredados de otras clases.
  - *Característica de calidad medida:* Tamaño de la clase.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Ninguna.

- **Número de Atributos Privados:** Es el número de atributos privados de una clase.
  - *Modo de cálculo:* Se suma una unidad por cada atributo privado de la clase.
  - *Criterios de contabilización:* El número de atributos privados de una clase será igual al número de atributos privados propios de la clase más el número de atributos privados heredados de otras clases.
  - *Característica de calidad medida:* Tamaño de la clase.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Ninguna.

- **Número de Atributos:** Es el número total de atributos de una clase.
  - *Modo de cálculo:* Se suma una unidad por cada atributo de la clase.
  - *Criterios de contabilización:* El número de atributos de una clase será igual al número de atributos propios de la clase más el número de atributos heredados de otras clases.
  - *Característica de calidad medida:* Tamaño de la clase.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Ninguna.

- **Número de Hijos:** Es el número total de hijos que tiene una clase.
  - *Modo de cálculo:* Se suma una unidad por cada clase hija que tiene la clase.
  - *Criterios de contabilización:*
  - *Característica de calidad medida:* Herencia de la clase.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Ninguna.

- **Número de Padres:** Es el número total de padres que tiene una clase.
  - *Modo de cálculo:* Se suma una unidad por cada clase padre que tiene la clase.
  - *Criterios de contabilización:*
  - *Característica de calidad medida:* Herencia de la clase.
  - *Métricas de implementación relacionadas:*
  - *Métricas del perfil utilizadas:* Ninguna.

- **MUI:** Es una métrica que nos indica si la clase el número de asociaciones no jerárquicas en las que participa una clase.
  - *Modo de cálculo:* Se suma una unidad por cada relación no jerárquica en la que participe la clase.
  - *Criterios de contabilización:* También se consideran relaciones no jerárquicas a las relaciones de agregación y de composición. Se considera que una clase participa en estas relaciones independientemente de si desempeña el papel de agregadora, agregante, componente o compuesta.
  - *Característica de calidad medida:* Herencia de la clase.
  - *Métricas de implementación relacionadas:* MUI.
  - *Métricas del perfil utilizadas:* Ninguna.

- **HNL:** Es el nivel del árbol de herencia en el que se encuentra una clase.

- *Modo de cálculo:* Se recorre el árbol de herencia en el que se encuentra la clase. Se suma una unidad por cada nivel del árbol de herencia en el que no hallemos a la clase actual.
- *Criterios de contabilización:* La raíz del árbol se considera el nivel 0. Si una clase está en más de un árbol de herencia (debido a alguna relación de herencia múltiple que tenga alguna clase antecesora del árbol o la misma clase) se considerará el valor de la métrica HNL para la clase al valor más grande de la métrica para esa clase.
- *Característica de calidad medida:* Herencia de la clase.
- *Métricas de implementación relacionadas:* HNL.
- *Métricas del perfil utilizadas:* Ninguna.

- **NMR:** Es el número de métodos renombrados por una clase.
  - *Modo de cálculo:* Se suma una unidad por cada método renombrado por la clase.
  - *Criterios de contabilización:* Se consideran métodos renombrados a aquellos métodos declarados de forma explícita en la clase que tienen un nombre igual que algún método de una clase antepasada de la clase actual.
  - *Característica de calidad medida:* Herencia del método.
  - *Métricas de implementación relacionadas:* NMR.
  - *Métricas del perfil utilizadas:* Ninguna.

- **NMI:** Es el número métodos heredados por una clase.
  - *Modo de cálculo:* Se suma una unidad por cada método que la clase ha heredado.
  - *Criterios de contabilización:* Se consideran métodos heredados a los métodos que están declarados explícitamente en las clases antepasadas de la clase actual. dichos métodos sólo se contabilizan una vez, no se cuentan una vez por cada nivel de herencia que haya entre la clase que hereda y la clase antepasada. Si una clase tiene un método renombrado se contabiliza como método heredado, dejándose de tener en cuenta el método de igual nombre que esté en una clase antepasada.
  - *Característica de calidad medida:* Herencia del método.
  - *Métricas de implementación relacionadas:* NMI.
  - *Métricas del perfil utilizadas:* Ninguna.

- **NMO:** Es el número de métodos añadidos por una clase.
  - *Modo de cálculo:* Se suma una unidad por cada método añadido por la clase.
  - *Criterios de contabilización:* Un método de una clase es añadido cuando no hay ningún método de igual nombre en ninguna clase antepasada de la clase actual. Si una clase no hereda de ninguna otra, todos sus métodos se considerarán añadidos.
  - *Característica de calidad medida:* Herencia del método.
  - *Métricas de implementación relacionadas:* NMO.
  - *Métricas del perfil utilizadas:* Ninguna.

4.2 Planificación

4.2.1 Planteamiento de las iteraciones y desarrollo del proyecto

Una vez realizado el estudio de los requisitos del cliente, y escogido el proceso de desarrollo unificado (RUP) para llevar a cabo el proyecto, se determinó abordar las tareas del siguiente modo:

- **Primera iteración**: Encontrar una lista de los casos de uso y de los actores que intervienen en el sistema. Además, se decidió abordar en primer lugar las partes de introducción de datos y generación de fichero intermedio de parámetros, escogiéndose como arquitectura básica del sistema un conjunto de los casos de uso correspondientes a estas dos partes.
- **Segunda iteración**: Obtener una vista de casos de uso completa. Realizar el análisis y diseño completo correspondientes a los casos de uso del simulación remota y local y obtención de resultados y creación de gráficas. Obtener al final de esta fase un prototipo con la funcionalidad de introducción de datos, generación del fichero intermedio y simulación remota y local.
- **Tercera iteración**: Desarrollar la funcionalidad de creación de gráficas y visualización de resultados, además de completar aspectos de las partes de entrada de datos y simulación. Corregir posibles defectos de las fases anteriores que no hubieran sido detectados en los procesos de prueba realizados en esas fases. (En esta fase se pondrán en marcha procedimientos de revisión).
- **Cuarta Iteración**: No tiene una lista de actividades excesivamente rígida (cosa que tampoco tiene la metodología). Como planteamiento general, se decidió realizar actividades de "ajuste" y refinamiento que permitan obtener una versión definitiva de la herramienta, además de la creación de un diálogo de preferencias y una ayuda contextual.

El modelo de desarrollo RUP establece la realización de un gran número de actividades dentro de cada iteración, así como la generación de una gran cantidad de documentación (vistas, modelos, etc.), que, si bien es cierto que suponen un soporte para la labor de desarrollo, también implican el empleo de una gran cantidad de tiempo para su realización. Esta inversión en tiempo puede ser rentable en el desarrollo de proyectos largos y complejos (se lleva un control más estricto y se tiene información más detallada sobre la marcha del proyecto), pero no tienen demasiado sentido en grupos de desarrollo de una sola persona. Utilizar una metodología supone seguir sus pasos, pero también es importante poder cumplir con las restricciones temporales del desarrollo, por lo que se ha tratado de adaptar RUP a las necesidades del proyecto, buscando así un compromiso adecuado entre fidelidad al modelo y un plazo de entrega razonable. En el comentario detallado del seguimiento de cada etapa, se ha añadido un apartado en el que se especifica la documentación que se ha generado en esa etapa. También se ha modificado en parte el modelo de pruebas RUP (consultar el documento del modelo de pruebas para obtener una información detallada).

Por último, reseñar que el seguimiento no hace referencia a uso (en tiempo) de recursos software o hardware, uso que tampoco se encontraba reflejado en la planificación. El motivo de ello es que el software usado en la realización de esta práctica o es gratuito o es propiedad del equipo de desarrollo, por lo que en ningún caso se ha producido una carencia del mismo que pudiera suponer un impedimento para la realización de la herramienta. Así pues, el seguimiento se ha limitado a un control del tiempo y esfuerzo necesitado para la finalización de las actividades de desarrollo.

4.2.2 Seguimiento de las iteraciones

4.2.2.1 Primera iteración

La primera tarea realizada ha sido la localización de actores y casos de uso. Los actores de nuestro sistema vienen detallados en el diagrama Rose. Posteriormente hemos asignado prioridad a esos casos de uso, siendo los de mayor prioridad los correspondientes a la toma de datos y generación del fichero intermedio de parámetros. Además, se han seleccionado algunos casos de uso con las funcionalidades básicas del sistema. Estos casos de uso serán los de prioridad máxima (y, por lo tanto, los primeros en ser desarrollados).

Los casos de uso con prioridad de segundo nivel (reflejan el resto de las interacciones de ejecución de simulaciones, control de resultados y representación de gráficas) se abordarán después, durante la segunda iteración. Finalmente, los casos de uso correspondientes a visualización de resultados y de gráficas (prioridad más baja) serán abordados durante la tercera etapa.

Esta prioridad más baja no significa que su relevancia sea menor, lo único que indica es que prefiero abordarles cuando el proceso de desarrollo esté más avanzado y pueda extraer de él información que facilite su realización (hay que considerar que el proceso de simulación coordina todo el proceso de toma de resultados y por consiguiente la visualización y representación de los mismos, por lo que puede ser interesante desarrollar por completo esos roles y posteriormente, desarrollar la funcionalidad de representación de gráficas de tal forma que permita el ensamblaje de todas las demás funcionalidades en torno a las suyas).

DOCUMENTACIÓN GENERADA:

- Modelo de casos de uso (primera versión, sólo se han detallado los casos de uso de la arquitectura). **MDL HTML**
- Modelo de análisis (primera versión, análisis de los casos de uso de la arquitectura). **MDL HTML**

4.2.2.2 Segunda iteración

En esta segunda iteración se aborda la construcción de un prototipo que permita realizar las tareas correspondientes a la toma de datos o modelado de las simulaciones, además del control de las mismas, la generación del fichero intermedio, la simulación local y remota y el control de resultados. Respecto a la planificación inicial, no se ha incluido la etapa de Implementación de pruebas, ya que las pruebas se han realizado manualmente. Tampoco se ha realizado la actividad de Integración del sistema como tal (no se ha planificado una secuencia de pasos para esa integración).

DOCUMENTACIÓN GENERADA:

- Modelo de casos de uso con todos los casos de uso detallados. **MDL HTML**
- Modelo de análisis de los casos de uso de simulación y toma de resultados. **MDL HTML**
- Modelo de diseño de los casos de uso de simulación y toma de resultados. **MDL HTML**
- Modelo de pruebas (para la parte de la herramienta desarrollada). **Aquí**
- Archivos con el seguimiento de cada actividad (realizado con Microsoft Project). En él se puede observar los tiempos empleados en cada actividad de la fase, así como la corrección de las estimaciones de la duración total del proyecto (en las actividades que es hito de final de la fase) y las incidencias surgidas durante el desarrollo de la actividad (obviamente, si han tenido lugar). **Aquí**

4.2.2.3 Tercera iteración

En esta tercera iteración se ha obtenido una versión beta de la herramienta (funcionalidad completa, posibilidad de introducir ligeras modificaciones en aspectos no cruciales de la herramienta). Se incluye por tanto toda la funcionalidad de visualización de resultados como de representación de gráficas (y cuestiones como el control de versiones, el control de gráficas, etc). En esta iteración no se han realizado tampoco la Implementación de los casos de prueba ni la Integración del sistema.

DOCUMENTACION GENERADA:

- Modelo de casos de uso con todos los casos de uso detallados (En él se reflejaran posibles modificaciones respecto a la versión de este modelo de la etapa anterior, las cuales pueden haber surgido por la aparición de algún nuevo enfoque de un caso de uso o la detección de algún error nuevo en este modelo). **MDL HTML**
- Modelo de análisis de los casos de uso (incluye los casos de uso de representación de gráficas). **MDL HTML**
- Modelo de diseño de los casos de uso (incluye los casos de uso de representación de gráficas). **MDL HTML**
- Modelo de pruebas. **Aquí**
- Archivos con el seguimiento de cada actividad (realizado con Microsoft Project). En él se puede observar los tiempos empleados en cada actividad de la fase, así como la corrección de las estimaciones de la duración total del proyecto (en las actividades que es hito de final de la fase) y las incidencias surgidas durante el desarrollo de la actividad (obviamente, si han tenido lugar). **Aquí**

4.2.2.4 Cuarta iteración

Esta cuarta iteración se ha dedicado a realizar últimos ajustes de la herramienta (interfaces, errores no detectados en el proceso de pruebas) así como a la realización de un manual y de ayuda conextual.

DOCUMENTACIÓN GENERADA:

- Manual de la herramienta
- Ayuda contextual de la herramienta

4.3 Seguimiento de los Riesgos del Proyecto

Se han tenido problemas con el proceso de desarrollo, debido a que muchas veces no sabía exactamente que artefactos debía desarrollar (en el RUP viene claro, pero son demasiados y he tenido que recortar un poco).

También ha habido modificaciones en la forma de hacer el análisis, ya que al principio se empezó haciendo un análisis normal, pero se cambio al poco tiempo al poder hacerlo según el proceso marcado por RUP (utilización de clases boundary, control y entity).

Hubo igualmente problemas en la etapa de diseño al utilizar el modelado orientado a aplicaciones desarrolladas en VC++, que requiere estereotipar las clases de MFC, ATL, etc. Es un sistema bastante lento y complejo y al final opté por el sistema de diseño normal sin incluir el framework de las MFC.

También ha habido problemas con la herramienta de modelado, Rational Rose, en especial con los diagramas de secuencia, ya que la herramienta descolocaba todos los diagramas cuando le daba la gana y hacía caso omiso a los intentos de recolocación de los elementos y mensajes.

He tenido problemas con la implementación, debido a la complejidad del proyecto, la interacción con objetos ActiveX no documentados, la complejidad del protocolo SSH y a que el lenguaje de desarrollo VC++ es muy complejo.

También ha habido problemas con las pruebas, debidos a que estas han resultado ocupar bastante más tiempo que el previsto (la aplicación es más compleja de lo esperado) y además se han producido errores que ha llevado más de una semana encontrarlos. VC++ es un lenguaje muy propenso a los errores por la sobre-utilización de punteros y manejo de memoria dinámica.

El hecho de ser una aplicación multi-hilo ha complicado también bastante las cosas, ya que la sincronización entre los diferentes hilos ha requerido un minucioso análisis y muchas pruebas.

5. Pruebas

Además de las pruebas realizadas a mano, se ha utilizado una herramienta para la generación de planes y casos de prueba: Parasoft C++ Test. La herramienta permite realizar todos los tipos de prueba especificados en el estándar RUP y está especialmente diseñada para proyectos realizados en VC++. Además se pueden especificar nuevos casos y modelos de prueba, así como cambiar las reglas de los que ya están preestablecidos.

La documentación generada con las pruebas se encuentra **aquí.**

6. Conclusiones y trabajo futuro

La elaboración de esta herramienta ha tenido una complejidad elevada, en gran medida por la ejecución remota de simulaciones, debido a la necesidad de implementar una librería del protocolo de comunicación SSH y al desconocimiento de dicho protocolo.
Al ser una aplicación multithread, se ha complicado bastante la sincronización entre los diferentes hilos del programa, bien sean internos de la librería de SSH o del propio programa a la hora de la ejecución de simulaciones.
El aspecto de la interfaz ha sido un punto muy cuidado, ya que al tratarse del desarrollo de un front-end, la calidad del mismo se valora en gran medida por la facilidad de uso y su vistosidad. Para ello se ha empleado una librería de clases, que extiende las clases MFC, proporcionando efectos visuales, mayor colorido, sombras en los menús, etc.
La aplicación ha sido ampliamente probada, lo que no quiere decir que no existan errores, pero sí que se trata de una herramienta robusta y fiable.
La eficiencia ha sido también uno de los puntos preferentes en el desarrollo, y esto viene reflejado en la gran velocidad y el poco consumo de recursos por parte de la misma.
El tratamiento gráfico de los resultados no ha sido todo lo óptimo posible por falta de tiempo, y se denota en una bajada del rendimiento al emplear un control ActiveX para la representación gráfica 2D. El uso de controles ActiveX, COM+ o .NET, ralentiza las aplicaciones pero por otro lado facilita el desarrollo de algunas tareas, simplificando el desarrollo de la aplicación al aportar partes funcionales que se integran fácilmente en las aplicaciones.
A pesar de ser una de las metas, por falta de tiempo no se ha desarrollado la parte de representación gráfica 3D, quedando la misma como uno de los trabajos futuros de mayor preferencia.
La simulación remota se realiza por el protocolo de comunicación SSH. SSH es un protocolo muy seguro pero a la vez lento, ya que cada paquete de datos a transferir ha de ser encriptado antes de ser transferido y desencriptado una vez recibido. El proceso de encriptación/desencriptación es un proceso lento, por lo que las comunicaciones se ralentizan, por lo que éste protocolo puede no es el más adecuado para algunas las situaciones en las que se precise mucha velocidad y sin embargo no se precise seguridad. Por éste motivo, puede ser útil la utilización de otros protocolos como telnet y FTP para la ejecución y transferencia de ficheros remota. La librería SSH implementada en el proyecto, no solo añade capacidad de comunicación SSH y SFTP a la aplicación, sino que además implementa los protocolos Telnet, Raw y Rlogin. De este modo sólo habría que dotar al programa del protocolo de transferencia FTP, del cual existen muchas implementaciones de librería de código abierto e incluso existe una clase MFC que lo implementa, por lo que la complejidad sería mínima.
Otro de los puntos que puede ser implementado en futuras versiones es la replicación de simulaciones, que viene a ser como la duplicación, pero generando varias simulaciones a partir de una, tomando como base algunas de sus propiedades y dándoles valores consecutivos hasta completar el número de réplicas.

Otra de las posibles mejoras podría ser el acoplamiento de una nueva barra de herramientas para las vistas de texto con opciones de cambio de tamaño y tipo de letra, lo que aportaría una personalización completa de la visualización del contenido de los ficheros.
Y por último sería interesante añadir iconos a los diferentes nodos del árbol de simulaciones y a los elementos de los menús, cosa que resulta muy sencilla cuando se dispone de los recursos gráficos (iconos) apropiados, que resultan casi imposibles de encontrar para éste tipo de aplicaciones a medida.

Como nota final cabe añadir que el desarrollo del proyecto ha sido enriquecedor, no tanto a nivel de análisis y diseño como de conocimientos de VC++ y SSH.
Aún así me gustaría mencionar que creo más interesante la realización de proyectos en grupo, debido a la interacción entre los diferentes miembros, la necesidad de una mejor planificación, reparto de tareas, y seguimiento del mismo, ya que la planificación de un proyecto en el que sólo hay una persona no tiene mucho sentido y bien menos el reparto de tareas o interacción entre los diferentes componentes del equipo de desarrollo.

7. Bibliografía

8. Apéndices

8.1 Manual de Usuario

8.2 Manual de Instalación

8.3 Librería de comunicación SSH (SSHLib)

8.4 Componente de representación gráfica NTGraph