Universidad de las Ciencias Informáticas

Facultad 15

Nodo Virtual de Procesos

Versión 2.0

Trabajo de Diploma para optar por el título de
Ingeniero Informático

Autores: Liosdany Muñoz Carballo.
Jorge E. Rodríguez Escalona.

Tutor: Ing. Yalice Gámez Batista.

Ciudad de La Habana, junio 2010.
"Año 52 de la Revolución."

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Facultad 15 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____ _____

Liosdany Muñoz Carballo — Autor

Jorge E. Rodríguez Escalona — Autor

_____

Yalice Gámez Batista

Tutor

AGRADECIMIENTOS

- *Al profesor Yoan Martínez por poner siempre a nuestra disposición sus conocimientos para perfeccionar nuestro trabajo.*

- *Al profesor Raúl Velázquez por ser paciente ante cada pregunta de calidad u otra materia, mostrando su interés de ayudar siempre.*

- *A las personas que de una forma u otra contribuyeron a la realización del Trabajo de Diploma.*

DEDICATORIA

Liosdany:

A mis padres, por convertirme en la persona que soy, por enseñarme desde el ejemplo, que confiar en sí mismo y en los demás es la garantía para vencer cualquier dificultad, por brindarme el constante amor que sólo ellos son capaces de dar, por acompañarme en cada preocupación, desvelo, en cada alegría por vencer una materia. Por ser ambos, la razón principal de lograr que mejore como ser humano y buen profesional cada día.

A mi hermano por confiar en mí y compartir con respeto mis decisiones.

A mi eterna amiga Marién, ese rayito de luz que alumbró mi cordura, para poder conseguir el equilibrio entre el amor y la distancia, por enseñarme que, con y por amor, todo es posible. Por su apoyo incondicional y su sincera fidelidad.

A mi familia toda, en especial a mis abuelos Dora, María y Vicente siempre pendientes de mi trayectoria. A mi tía Belkis, que desde pequeño me ayudó a realizar mis tareas y siempre ha estado atenta de mi desempeño como ser humano.

A los pocos, pero sinceros amigos que me ha regalado el tiempo durante estos cinco años de carrera, unos desde el inicio, como el inseparable Jorge Ernesto, mi compañero de tesis, que ha demostrado ser el mejor, un hermano; a Pedroso y Yadira y a las más recientes Isabel y Yanet. En general a todos mis compañeros que de una forma u otra me han ayudado a superarme.

A mi tutora Yalice, por su amabilidad y confianza plena de haberme seleccionado desde segundo año a ser partícipe del desarrollo de su tesis de maestría, por el ejemplo que grabó en mí desde su primer día de clases y por aconsejarme en los momentos de tensión.

Ernesto:

A mi madre, a quien le debo mi vida, mi ser, mis mayores alegrías.

A mi pareja, por enseñarme un nuevo significado de la palabra felicidad.

A mi tutora, por su confianza, su apoyo incondicional y su ejemplo.

A mis amigos, por su apoyo incondicional y ese tesoro inmenso que es la amistad.

A mi hermano Yozamy, a quien hubiera querido tener hoy a mi lado.

A Liosdany, por estar siempre a mi lado, por enseñarme lo que es tener un hermano.

RESUMEN

El presente trabajo se fundamenta en la programación de una aplicación para desarrollar la simulación de procesos industriales, mediante el uso de modelos matemáticos, que permitirá a los estudiantes de la especialidad Ingeniería Automática probar estrategias de control. Actualmente se cuenta con una primera versión del producto, pero la misma presenta no conformidades que impiden su integración a la enseñanza. Por estas razones se realizó un análisis del diseño propuesto y de los módulos y componentes reutilizables, y se desarrolló una segunda versión como aplicación web, implementando las funcionalidades necesarias para solucionar los problemas encontrados. Fue desarrollada en PHP y C++, este último con el framework Qt, bajo el paradigma orientado a objetos, utilizando Net Beans como entorno de desarrollo y Apache como servidor web. Del lado del cliente se utilizó el framework ExtJs. También se desarrolló un adaptador en C++ que permitiera la comunicación entre las diferentes partes del sistema, y se dotó a la aplicación de nuevas funcionalidades como conexión a la base de datos mediante ODBC, reportes y alertas. Para obtener como resultado una herramienta robusta, se llevaron a cabo una serie de pruebas tanto al código como a las interfaces, que permitieron detectar los fallos de la aplicación y darles solución en posteriores iteraciones.

PALABRAS CLAVE

Nodo Virtual de Procesos, aplicación web.

TABLA DE CONTENIDOS

INTRODUCCIÓN

En la actualidad, las tecnologías de la informática y las comunicaciones (TIC) desempeñan un rol esencial en todas las esferas de la sociedad, como la medicina, la economía, la industria, las investigaciones científicas y en la educación, por sólo citar algunos ejemplos.

En el Instituto Politécnico José Antonio Echeverría (CUJAE), Facultad de Ingeniería Eléctrica, se estudia la especialidad Ingeniería Automática. En esta se imparten una serie de contenidos que proveen a los estudiantes de toda la teoría necesaria para ejercer como profesionales de la rama. Sin embargo, no siempre estos conocimientos proveen a los estudiantes de las herramientas prácticas para enfrentarse a una situación real. Esto está muy asociado a la forma tradicional en la que se imparten estos tópicos y es entonces que se hace necesario el uso de las TIC. (Gámez Batista, et al., 2009) Para esta especialidad se recomienda el uso de simuladores, permitiendo que un alumno pueda trabajar sobre un problema de forma gráfica y observar cómo el cambio en un determinado elemento se ve reflejado de forma inmediata en el resto, como si estuviera ante el proceso real. (Vaquero, 2007)

En la actualidad existen diversos programas que permiten realizar simulaciones de procesos industriales. Por lo general, estos realizan las simulaciones virtuales en las aplicaciones clientes, lo que limita el trabajo grupal de los estudiantes, requisito esencial para poder incluir estos programas en la docencia. Como alternativa se crea un Nodo Virtual de Procesos (NVP) que permite la simulación simultánea de diferentes procesos concurrentes sin que interfieran unos con otros. (Gámez Batista, et al., 2009) Las simulaciones se ejecutan en un servidor, y los usuarios tienen acceso a ellas desde clientes situados en máquinas independientes.

En la Universidad de las Ciencias Informáticas (UCI), en conjunto con desarrolladores del Instituto Superior Politécnico José Antonio Echeverría (CUJAE), fue desarrollado un proyecto para la elaboración de un simulador de procesos industriales. Esta aplicación, también conocida como nodo virtual de procesos (NVP) debido al uso que hace de los mismos en su funcionamiento, está dirigida a los estudiantes de la especialidad Ingeniería Automática de la CUJAE.

La aplicación implementada está dividida en dos partes, siguiendo la filosofía del NVP: un servidor donde se desarrolla la simulación de los procesos y clientes donde trabajan los usuarios. Tanto el cliente como el servidor son aplicaciones de escritorio, que utilizan el framework Qt y C++ como lenguaje de

programación. Como gestor de bases de datos fue utilizado PostgreSQL. La comunicación cliente – servidor fue implementada haciendo uso de los sockets, disponibles en el lenguaje de implementación utilizado.

El equipo de desarrollo realizó un análisis de las funcionalidades que hasta el momento presenta el NVP, de los requisitos que este debe cumplir y de las recomendaciones señaladas en trabajos de diploma (Escobar Pompa, et al., 2008) (Gonce Fernández, 2008) (Cleger Despaigne, et al., 2009) (Hernández Garrigó, et al., 2009) (Rivelo Ayala, et al., 2009) que han abordado y sustentado el proceso de desarrollo del producto.

La versión actual del NVP genera varias no conformidades, que fueron señaladas luego de realizado el análisis antes mencionado.

La CUJAE cuenta con limitados recursos informáticos, entiéndase por esto: computadoras con varios años de explotación y hardware desactualizado, con escasa memoria RAM, limitada velocidad de procesamiento y poca capacidad de disco duro. La aplicación requiere la instalación de las librerías Qt, tanto en el servidor como en los clientes, ocupando espacio en los discos duros, disminuyendo las capacidades de las computadoras que pueden ser utilizadas por los usuarios.

El servidor sólo cuenta con los drivers para utilizar PostgreSQL como gestor de bases de datos, y únicamente en la plataforma GNU/Linux. Además, sólo cuenta con dos modelos de procesos, lo que está lejos de satisfacer las necesidades de los usuarios. Estos modelos se adicionan a la aplicación utilizando bibliotecas compartidas, las que sólo han sido compiladas en GNU/Linux, aumentando la dependencia del servidor hacia este sistema operativo. Por último, queda pendiente la implementación de las funcionalidades que permiten la generación de reportes y alertas.

Debido a la situación problémica planteada anteriormente, surge el siguiente **problema**: Las no conformidades detectadas por el equipo de desarrollo dificultan la integración del NVP al proceso docente de la especialidad Ingeniería Automática.

Objeto de estudio: Proceso de desarrollo de software.

Campo de acción: Implementación de un nodo virtual de procesos.

Objetivo general: Desarrollar una solución informática que elimine las no conformidades detectadas y posibilite la integración del NVP al proceso docente de la especialidad Ingeniería Automática.

Idea a defender: El desarrollo de una solución informática que elimine las no conformidades detectadas por el equipo de desarrollo permitirá que el nodo virtual de procesos pueda ser integrado al proceso docente de la especialidad Ingeniería Automática.

Del objetivo general se derivan los siguientes **objetivos específicos** y **tareas**:

1. Estudiar las no conformidades.
    - Tarea 1: Estudio de los trabajos de diploma mencionados anteriormente.
    - Tarea 2: Análisis y comparación de las funcionalidades del nodo virtual de procesos y de los requisitos del cliente.
2. Elaborar el marco teórico.
    - Tarea 3: Búsquedas bibliográficas relacionadas al objetivo.
3. Seleccionar las herramientas a utilizar.
    - Tarea 4: Estudio de los paradigmas de programación de acuerdo a los requisitos del problema.
    - Tarea 5: Estudio de los lenguajes de programación que se ajusten a los requisitos del problema.
    - Tarea 6: Estudio de los entornos de desarrollo que se ajusten a los requisitos del problema.
    - Tarea 7: Estudio de los estándares de comunicación con los gestores de bases de datos que se ajusten a los requisitos del problema.
4. Desarrollar la solución.
5. Validar la solución implementada.
    - Tarea 8: Desarrollo de los casos de prueba que permitan validar la solución.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se realiza un análisis general de las diferentes técnicas de programación, lenguajes de programación, tecnologías, frameworks y tendencias de uso para el desarrollo de una nueva versión del nodo virtual de procesos.

1.2 Nodo virtual de procesos

En la actualidad existen muchas herramientas informáticas que permiten realizar simulaciones de procesos industriales (en tiempo real o no), pero todas tienen limitaciones en cuanto al número de recursos que necesitan para su implementación o en cuanto al número de procesos simultáneos que se pueden simular. Incluso, en su mayoría, o simulan los procesos o permiten probar aplicaciones reales, nunca las dos prestaciones. (Gámez, 2008)

Para dar solución a esta problemática se optó por un Nodo Virtual. Un Nodo Virtual es una aplicación que permite ejecutar diferentes instancias del software en un único nodo (nodo físico) y cada instancia del software trabaja en un entorno de ejecución independiente (nodo virtual). (Maier, et al., 2005)

Atendiendo a esto se define como nodo virtual de procesos a aquel software que permitirá implementar los modelos de diferentes procesos para su simulación o para probar aplicaciones en tiempo real. Será necesario que varios procesos se encuentren activos de manera simultánea, lo que requiere de gran cantidad de nodos y computadoras que no están disponibles generalmente. (Gámez, 2008)

Para ello se establece como nodo a aquella estructura a la cual se interconectan varios elementos. No hay que pensar en un nodo como un elemento constituido solamente por una parte física, sino más bien considerarlo como una unidad funcional en donde tiene que haber tanto hardware como software. (Hernández Garrigó, et al., 2009)

Por otra parte, al ser el punto de conexión de dos o más elementos, el nodo por lo general tiene la capacidad de recibir información, procesarla y enrutarla a uno o varios nodos. De esta forma, un nodo puede ser el punto de conexión para transmitir los datos, el punto desde el cual se distribuyan los datos hacia otros nodos y el punto al que se transmitan los datos. (Escobar Pompa, et al., 2008)

Esta filosofía de trabajo permite la simulación simultánea de diferentes procesos concurrentes sin que interfieran uno con otros. La virtualización de nodos provee una vía de regular el acceso a recursos de hardware exclusivos de un determinado número de consumidores. En este caso los consumidores son los entornos de ejecución para cada proceso, los cuales están sujetos a las propiedades de la simulación.

De ahí se derivan los siguientes requerimientos para la virtualización del nodo: (Gámez, 2008)

- El parámetro más importante es minimizar los gastos de virtualización para preservar los recursos para el proceso en ejecución.
- Cada entorno de ejecución introducido por la virtualización del nodo debe ser tan transparente como sea posible para los restantes. Esto es importante para que la medición de la implementación no sufra modificaciones en comparación con la real.

En el mundo se han desarrollado varias aplicaciones que utilizan los nodos virtuales, con el objetivo de aprovechar al máximo los recursos de las computadoras.

A continuación se mencionan algunos programas que hacen uso de los nodos virtuales.

En la Universidad de Nottingham Trent, se creó un software para la simulación de sistemas de agua (Hosseinzaman, et al., 1995) surgido por la necesidad que tenían en la industria del agua de adquirir y almacenar datos de estaciones remotas para la inspección ingenieril.

En la Universidad de Stanford se creó un algoritmo de nodos virtuales para el trabajo con imágenes en tres dimensiones (Molino, et al., 2004). De esto resultan elementos que contienen material real y regiones vacías. El material faltante está contenido en otra u otras copias. El algoritmo de nodo virtual determina automáticamente el número de réplicas así como la asignación de material para cada una. Provee además los grados de libertad requeridos para simular el material parcial o completamente fragmentado en una imagen consistente con la geometría. Aprovecha las posibilidades de la simulación de una geometría compleja con una simple mezcla.

Para la simulación de redes, en la Universidad de Stuttgart en Alemania, instituto especializado en sistemas distribuidos y paralelos, se creó la aplicación “Network Emulation Testbed” (Maier, et al., 2005). Esta aplicación va dirigida a simular un entorno de redes configurable que permita reproducir un escenario

real de cientos de nodos en comunicación. De esta manera, posibilita medir de manera comparativa el comportamiento de una aplicación en diferentes entornos de redes o de varias aplicaciones en un mismo entorno de red. Esta aplicación es utilizada tanto en redes tradicionales como en un entorno de redes ad hoc inalámbricas. Permite a partir de una red de 64 computadoras traducirlo a un escenario de más de 1920 nodos.

En Japón se creó la herramienta “StarBED” para dar solución al vacío que existe entre internet y los entornos para experimentación, atendiendo a los aspectos de escala, complejidad y realidad (Miyachi, et al., 2006). Es una aplicación de pruebas basada en lotes de nodos que tiene como objetivos construir entornos de experimentaciones reales, complejos y de gran escala.

Se concluye que a pesar de que existen varias aplicaciones que hacen uso de los nodos virtuales, no se ha identificado una herramienta informática que funcione como un nodo virtual de procesos, y específicamente, que simule procesos industriales.

La ausencia de un software con estas características motivó el desarrollo del simulador de procesos industriales por la UCI en conjunto con la CUJAE, aunque como se refleja en la introducción, la versión implementada generó no conformidades que hicieron imposible su integración en el proceso de enseñanza – aprendizaje de la especialidad Ingeniería Automática.

1.3 Análisis de posibles soluciones para las no conformidades encontradas.

Como se abordó en la introducción, la versión existente del NVP presenta numerosas no conformidades que han hecho imposible su integración al proceso de enseñanza – aprendizaje de la especialidad Ingeniería Automática. A continuación se enumeran las principales no conformidades encontradas y se hace un análisis de las posibles soluciones que se les podrían dar, para seleccionar las que mejor resuelvan el problema planteado.

1.3.1 Consumo de recursos en las computadoras clientes.

La versión anterior del nodo virtual de procesos requería de su instalación en cada computadora que funcionara como cliente, necesitando que se instalaran también las librerías Qt para que pudiera funcionar correctamente. De esta forma, se utilizaba una considerable parte del disco duro, que podría ser usada en otras actividades necesarias para los estudiantes.

A continuación, aparecen las posibles soluciones que se les podía dar a la situación planteada.

En trabajos de diploma anteriores se realizó un extenso análisis para seleccionar las herramientas y el lenguaje que mejor se adaptaran a la problemática existente; y *"… se optó por escoger como lenguaje de programación el C++ para aprovechar su velocidad de ejecución, eficiencia y todas las potencialidades que ofrece de manera general*", (Hernández Garrigó, et al., 2009) con lo que se elimina el uso de cualquier otro lenguaje de programación. Debido a esto, cualquier mejora a realizar en la aplicación cliente existente conllevaría a cambios de las funcionalidades del sistema, pero se mantendría la no conformidad de que ocupa un excesivo espacio en el disco duro de las computadoras donde se instale.

Por otra parte, *"... hay otra forma de enviar software que va a evitar que los usuarios se conviertan en administradores de sistemas. Las aplicaciones basadas en web son programas que corren en servidores web y utilizan páginas web como la interfaz de usuario. Para el usuario promedio este nuevo tipo de software será más sencillo, económico, más móvil, más confiable, y a menudo más poderoso que el software de escritorio.*" (Graham)

Cada vez es mayor el número de empresas que optan por usar la arquitectura de Internet, sus protocolos y servicios, como medio para conectar los sistemas de su propia empresa, así como la propia Internet como infraestructura de comunicación para conseguir esa conexión. En un entorno como el de Internet no se puede asumir que todos los clientes cuenten con una computadora funcionando con Windows como sistema operativo para poder ejecutar programas específicos de este. Existen también otros dispositivos y otros sistemas operativos, a los cuales hay que tener en cuenta a la hora de desarrollar un software. (Charte Ojeda, 2002)

Existen varios criterios de comparación entre las aplicaciones web y las de escritorio. A continuación, aparecen los principales:

- Compatibilidad multiplataforma: Las aplicaciones web tienen un camino mucho más sencillo para la compatibilidad multiplataforma que las aplicaciones de escritorio. Varias tecnologías incluyendo PHP, Java, Flash, ASP y Ajax permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales. (Solutions, 2009)

- Actualización: Las aplicaciones basadas en web están siempre actualizadas sin requerir de la intervención del usuario, todo lo contrario de las aplicaciones de escritorio. (Solutions, 2009)
- Inmediatez de acceso: Las aplicaciones basadas en web no necesitan ser descargadas, instaladas y configuradas. (Solutions, 2009)
- Menos requerimientos de memoria: Las aplicaciones basadas en web tienen más razonables demandas de memoria RAM de parte del usuario final que los programas instalados localmente. Al residir y correr en los servidores del proveedor, esas aplicaciones basadas en web usan en muchos casos la memoria de las computadoras servidoras, dejando más espacio libre en la memoria RAM de los clientes. (Solutions, 2009)
- Múltiples usuarios concurrentes: Las aplicaciones basadas en web pueden ser utilizadas por múltiples usuarios al mismo tiempo. (Solutions, 2009)
- Versatilidad y potencia: las aplicaciones web en varias ocasiones están restringidas por las limitaciones propias del HTML, y por la necesidad de establecer una buena infraestructura de red que permita la velocidad necesaria en las comunicaciones. (Alicante)
- Cliente delgado: cuando las aplicaciones web se ejecutan completamente en el servidor, a dicha arquitectura se le denomina cliente delgado. Tiene como inconvenientes que delega mucha carga en el servidor, por lo que este requiere tener mayores requisitos de hardware para garantizar una buena escalabilidad, por lo que aumenta el coste del mismo. Por otra parte, el cliente es más ligero, existe un menor tráfico de la red y la administración de la misma es menos compleja. (Souto)

Se puede concluir que la implementación de los clientes del nodo virtual de procesos como aplicación web eliminará la necesidad de instalar en las computadoras clientes aplicaciones adicionales, como las librerías Qt; necesitando solamente tener instalado un navegador web, aplicación que no consume muchos recursos de la máquina. No obstante, esta solución requerirá de una computadora adicional que tenga buenas prestaciones, de manera que pueda funcionar como servidor, pero permitirá ahorrar gran cantidad de recursos en las computadoras restantes que sean utilizadas para trabajar con el simulador, principalmente espacio de disco duro.

1.3.2 Conexión a la base de datos

Para desarrollar la versión anterior del sistema, se seleccionó como gestor de bases de datos PostgreSQL, manteniéndose dicha selección para la versión actual debido al buen desempeño presentado por el mismo. No obstante, el servidor existente del nodo virtual de procesos sólo cuenta con los drivers de conexión a la base de datos disponibles para el sistema operativo GNU/Linux. El equipo de desarrollo tuvo problemas para encontrar los drivers que cumplieran dicha función en Windows, por lo que el servidor sólo fue desarrollado para la plataforma GNU/Linux.

Para seleccionar una solución que resolviera esta situación se realizó un análisis, obteniéndose los siguientes resultados:

Hay dos posibles accesos al sistema gestor de bases de datos (SGBD): directo e indirecto.

El que se denomina directo consiste en que normalmente, una compañía diseña una aplicación que accederá a un SGBD. Esta aplicación se desarrolla hacia un tipo de SGBD, y por tanto sólo se implementa el acceso para ese tipo en específico, es decir, utiliza un driver nativo de ese SGBD. Si se requiere acceso a otro SGBD, es necesario por tanto, utilizar un nuevo controlador o driver de acceso. Este sistema hace que el acceso sea directo a la base de datos, pero tiene el inconveniente de que hay que desarrollar un enlace para cada SGBD que se quiera soportar. (Aula-Tec) (Aragon)

Otro posible acceso es el indirecto, es decir, se adiciona una capa intermedia entre la aplicación y el SGBD, liberando a la primera de contener el driver específico para ese SGBD. Con esto se garantiza la conectividad entre el software y cualquier SGBD que soporte dicho sistema. (Aula-Tec) (Aragon)

Esta última opción permitiría seleccionar el gestor que más se adecue a las necesidades y posibilidades de los usuarios sin necesidad de cambiar el código, haciendo más flexible y robusto al NVP, y facilitando su adaptación a cambios que siempre están presentes en el mundo del software. Por tal motivo se decidió utilizar esta última opción.

1.4 Paradigmas de programación.

Los paradigmas son marcos de referencia que imponen reglas sobre cómo se deben hacer las cosas, indican qué es válido dentro de su universo y qué está fuera de sus límites. Un paradigma distinto implica nuevas reglas, elementos, límites y maneras de pensar, o sea, implica un cambio. Los paradigmas son

considerados patrones de pensamiento para la resolución de problemas. Desde luego siempre teniendo en cuenta los lenguajes de programación, según el interés de estudio. (Hernández Garrigó, et al., 2009)

Definición:

Un paradigma de programación es un modelo básico de diseño y desarrollo de programas, que permite producir programas con unas directrices específicas, tales como: estructura modular, fuerte cohesión, alta rentabilidad, etc. (Bonaparte, et al., 2008)

No es mejor uno que otro sino que cada uno tiene ventajas y desventajas. Es necesario valorar cada paradigma y la situación donde se debe aplicar para poder seleccionar al más apropiado.

1.4.1 Tipos de Paradigmas de Programación

Existe una amplia bibliografía donde se describen y analizan las diferentes clasificaciones de paradigmas existentes. En ellas se clasifican de modos similares, destacando siempre el imperativo o por procedimientos, el orientado a objetos, el funcional y el lógico. De forma adicional, algunos autores o profesores, mencionan paradigmas heurísticos, concurrentes, procedimentales, declarativos y demostrativos.

1.4.1.1 Paradigmas Imperativos o Procedimentales

Es una secuencia computacional ejecutada etapa a etapa para resolver el problema. Su mayor dificultad reside en determinar si el valor computado es una solución correcta del problema. (Rodríguez, 2008) Son un conjunto de instrucciones que le indican al computador cómo realizar una tarea, es decir, se describe paso a paso la solución del problema a resolver.

1.4.1.2 Paradigmas Declarativos

Se construye señalando hechos, reglas, restricciones, ecuaciones, transformaciones y otras propiedades derivadas del conjunto de valores que configuran la solución. (Rodríguez, 2008)

1.4.1.3 Paradigmas Demostrativos

Cuando se programa bajo un paradigma demostrativo (también llamada programación por ejemplos), el programador no especifica procedimentalmente cómo construir una solución sino que presenta soluciones

de problemas similares (Bonaparte, et al., 2008). El software será el encargado de seleccionar la solución que, según sus criterios de evaluación, sea la mejor para solucionar el problema.

1.4.1.4 Paradigmas Funcionales

Estos operan solamente a través de funciones. Cada función devuelve un solo valor, dada una lista de parámetros. No se permiten asignaciones globales, llamados efectos colaterales. La programación funcional proporciona la capacidad para que un programa se modifique a sí mismo, es decir, que pueda aprender. (Hernández Garrigó, et al., 2009)

1.4.1.5 Paradigmas Lógicos

Esta programación se basa en un subconjunto del cálculo de predicados, incluyendo instrucciones escritas en formas conocidas como cláusulas de Horn. Este paradigma puede deducir nuevos hechos a partir de otros hechos conocidos. Un sistema de cláusulas de Horn permite un método particularmente mecánico de demostración llamado resolución (Bonaparte, et al., 2008).

1.4.1.6 La Programación Orientada a Objetos (POO)

Es una metodología de diseño de software y un paradigma de programación que define los programas en términos de clases y objetos. Los objetos son entidades que combinan estado (datos) y comportamiento (procedimientos o métodos). La programación orientada a objetos expresa un programa como un conjunto de estos objetos, que se comunican entre ellos para realizar tareas. Esto difiere de los lenguajes procedimentales tradicionales, en los que los datos y los procedimientos están separados y sin relación. Este paradigma está pensado para hacer los programas y módulos más fáciles de escribir, mantener y reutilizar (Bonaparte, et al., 2008).

Por su parte, una clase es un molde o prototipo en donde se definen los atributos y las acciones comunes de una entidad.

Se denomina encapsulación al ocultamiento del estado, es decir, de los datos miembros de un objeto, de manera que sólo se puede cambiar mediante las operaciones definidas para ese objeto. De esta forma, el usuario de la clase puede obviar la implementación de los métodos y propiedades para concentrarse sólo en cómo usarlos. Por otro lado, se evita que el usuario pueda cambiar su estado de maneras imprevistas e incontroladas (Wesley, 2001).

La herencia es uno de los mecanismos de la programación orientada a objetos, por medio de la cual una clase se deriva de otra de manera que extiende su funcionalidad. Una de sus funciones más importantes es la de proveer polimorfismo. (Wesley, 2001)

El polimorfismo es la propiedad que poseen algunas operaciones de tener un comportamiento diferente dependiendo del objeto sobre el que se aplica. Gracias a él una entidad puede tomar diferentes formas.

De acuerdo a los requisitos del NVP se selecciona el paradigma orientado a objetos por tener características que facilitan la implementación de la aplicación y le dan facilidades como reutilización, eficiencia y robustez.

1.5 Selección del lenguaje de programación y del entorno de desarrollo

Para la implementación de la aplicación web, se investigaron los lenguajes de programación que más se han destacado. (Tiobe) De los utilizados en la web, quedaron seleccionados para ser analizados C#, Java y PHP.

1.5.1 C#

C# es un nuevo lenguaje, potente, sencillo y orientado a objetos, desarrollado para satisfacer las necesidades actuales y de un futuro cercano. Es una herramienta como todos los lenguajes de programación, pero adaptada al trabajo actual y sin compromisos de compatibilidad ascendente con versiones anteriores.

Es una tecnología propietaria de Microsoft, por lo que el uso de la misma implica al mismo tiempo el uso de los productos de esa empresa. A pesar de que se han hecho fuertes intentos de transportarlo al sistema operativo GNU/Linux, como por ejemplo el proyecto Mono, estos esfuerzos no han rendido aún suficientes logros como para utilizarlos en proyectos de envergadura.

El objetivo de Microsoft ha sido la creación del primer lenguaje orientado a componentes, al estilo de Visual Basic, pero con la flexibilidad y potencia de C++ y sin muchas de sus complejidades. C, C++ o Visual Basic están pensados para el desarrollo de aplicaciones en sistemas operativos que ofrecen sus servicios a través de cientos o miles de funciones independientes que, en conjunto, forman lo que se conoce coloquialmente como API (Application Programming Interface). En contraposición, C# ha sido

diseñado para una plataforma, la plataforma Microsoft .NET, en la que los servicios son ofrecidos en forma de componentes.

Una de las novedades que han acompañado a C# es el surgimiento de ASP.NET. Esta tecnología propietaria fue escrita desde cero, por lo que es completamente distinta a las tecnologías desarrolladas por Microsoft para ser usadas en la web. Como una novedad importante brinda la posibilidad de separar el diseño gráfico de la hoja de negocios. (Aburto, 2001)

Las páginas ASP se procesan en el servidor, habitualmente IIS (Internet Information Server). Este ejecuta el código y combina su resultado con el contenido HTML original del documento, generando una página dinámicamente que es lo que se envía al cliente. Las ventajas de ASP son muchas facilitando, por ejemplo, la generación dinámica de contenido a partir de información almacenada en una base de datos sin necesidad de escribir un CGI (Common Gateway Interface) clásico. Basta con algo de código embebido en el documento para conseguir un resultado rápido.

Como se mencionó anteriormente, para utilizar ASP.NET es imprescindible tener instalado el Internet Information Server (IIS) de Windows; aunque es posible su uso mediante otros servidores como Apache, adicionándole previamente bibliotecas especiales que soporten esta tecnología. (Charte Ojeda, 2002)

1.5.2 Java

Java es un lenguaje de programación moderno, de alto nivel y orientado a objetos, desarrollado a principio de los años 90s por Sun Microsystems, la cual también controla los estándares del mismo a través de un mecanismo de apertura parcial denominado Java Community Process (JCP). (Allende, 2005)

La sintaxis del lenguaje heredó características de C y C++, eliminando las que resultaban excesivamente complejas e inseguras a la hora de implementar aplicaciones extensas.

Con el auge de Internet, se decidió aprovechar este lenguaje para desarrollar aplicaciones distribuidas y portables. La primera implementación de Java data de 1995 y pronto los navegadores web incorporaron soporte Java para la ejecución de pequeñas aplicaciones interactivas, más conocidas como applets. Actualmente es utilizado para el desarrollo de aplicaciones empresariales del lado del servidor y en dispositivos móviles a través de los estándares J2EE y J2ME, respectivamente.

Se destacan en la programación del lado del servidor los servlets y las Java Server Pages (JSP). Los servlets son programas que se ejecutan en el servidor y que extienden sus funcionalidades. Para su uso, además de la máquina virtual de Java se debe instalar un programa denominado contenedor de servlets, como por ejemplo el Tomcat. Por otra parte, las JSP son una forma alternativa y sencilla de construir servlets. Generan contenido web dinámico en forma de documentos HTML, aunque también generan otros formatos como XML. Permiten la utilización del código Java mediante scripts y etiquetas que aparecen dentro del documento HTML antes de ser enviados al cliente. Estas tecnologías facilitan en gran medida la implementación de sitios web dinámicos, por lo que han tenido un auge en popularidad.

1.5.3 PHP

Este lenguaje fue creado en 1994 por Rasmus Lerdorf como un complemento para el lenguaje PERL. Lo incorporó por primera vez en su propia página web para monitorizar las visitas que recibía. La implementación principal de PHP es producida actualmente por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Es publicado bajo la PHP License, considerada como software libre por la Free Software Foundation. (Gallego Vázquez, 2003)

PHP (acrónimo de PHP Hypertext Preprocessor) es un lenguaje interpretado "del lado del servidor" especialmente creado para el desarrollo de páginas web dinámicas. Puede ser incluido con facilidad dentro del código HTML, y permite un gran número de funcionalidades. No obstante, se han desarrollado aplicaciones que permiten su utilización desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+. (Gallego Vázquez, 2003)

Combinado con la base de datos MySQL, es el lenguaje estándar a la hora de crear sitios de comercio electrónico o páginas web dinámicas. (Gallego Vázquez, 2003)

1.5.4 Criterios de comparación.

Para hacer la comparación entre diferentes lenguajes de programación con el fin de seleccionar el que mejor se ajuste a las necesidades de los desarrolladores y clientes, es necesario siempre hacer un análisis lo más objetivo posible. Generalmente los miembros de cualquier equipo de desarrollo de software optarán por aquel lenguaje que conozcan mejor, e incluso se pueden introducir involuntariamente en el proceso de selección factores altamente subjetivos, como la preferencia personal por algún lenguaje en

particular. Por estos motivos, es imprescindible seleccionar criterios de comparación que permitan valorar las opciones existentes en su justa medida.

Teniendo en cuenta los requerimientos que se le exigen al nodo virtual de procesos, así como las restricciones que representan las condiciones en las que este va a ser usado, se seleccionaron los siguientes criterios de comparación:

- Ejecución en múltiples plataformas.
- Capacidades de generación de gráficos.
- Gestión de memoria.
- Velocidad de ejecución.
- Rendimiento.
- Licencia.
- Modularidad.
- Mantenibilidad.
- Coste de desarrollo.
- Integración externa.
- Seguridad.
- Escalabilidad.

1.5.4.1 Ejecución en múltiples plataformas.

Tanto PHP como Java son lenguajes que presentan un alto nivel de portabilidad.

En el caso de Java, la máquina virtual sobre la que corren los programas implementados en este lenguaje lo ha independizado casi completamente de la plataforma sobre la que se ejecuta. Ya que existen numerosas versiones de la máquina virtual para diferentes sistemas operativos, las aplicaciones Java pueden ser ejecutadas en cada uno de ellos.

PHP también es multiplataforma. Existen implementaciones de su intérprete para varios sistemas operativos, garantizando que como en Java, los programas implementados en PHP, esencialmente sitios

o aplicaciones web, puedan ser ejecutados en esas plataformas. La característica de este lenguaje de ser interpretado ha sido un factor decisivo para garantizar su portabilidad.

C#, y específicamente ASP.NET, es una tecnología propietaria de Microsoft. Como se argumentó anteriormente, su uso conlleva la instalación de otras herramientas de dicha empresa, tales como el Internet Information Server y el framework .NET, por mencionar los más importantes. Aunque es posible integrarla a otros servidores web como Apache, su dependencia al framework .NET restringe su uso sólo al sistema operativo Windows, eliminando las posibilidades de ser multiplataforma.

1.5.4.2 Capacidades de generación de gráficos.

Uno de los requisitos esenciales que se le exigen al nodo virtual de procesos es la representación de los datos numéricos que se obtienen como resultado de la simulación, en formato de gráficos.

Java tiene un buen soporte para la generación de gráficos en dos dimensiones, aunque sus capacidades tridimensionales están limitadas a x86. No obstante, el NVP no requiere la utilización de imágenes en tres dimensiones.

PHP contiene funciones y bibliotecas que facilitan el trabajo con imágenes. Por otra parte, existen bibliotecas externas que se le pueden importar y que se especializan en la generación de diferentes tipos de gráficos para la representación de conjuntos de datos, presentando las funcionalidades que son requeridas por el NVP para la correcta visualización de los resultados obtenidos en la simulación de los procesos industriales.

C# tiene muy buen soporte para 2D, contenido en numerosas bibliotecas de clases que implementan estas funcionalidades. Ejemplos de estas son las que contiene el framework .NET.

1.5.4.3 Gestión de memoria.

En lo referente al uso de la memoria de la computadora, tanto PHP como Java y C# poseen recolectores automáticos de memoria que facilitan enormemente la programación. No es necesario liberarla, eliminándose de esta forma los errores que pudiera cometer el programador a la hora de implementar la aplicación, y que podrían conllevar a una falla o al colapso total de la misma.

1.5.4.4 Velocidad de ejecución.

Tanto PHP como Java son lenguajes interpretados, por lo que deben ser traducidos al lenguaje máquina cada vez que son ejecutados. Esto los hace muy ineficientes a la hora de seleccionarlos para ser utilizados por el NVP teniendo en cuenta su velocidad de ejecución. Aunque se debe destacar que C# genera un código intermedio que también es interpretado, por lo que se encuentra en la misma situación.

Las aplicaciones Java están compiladas en un bytecode, lenguaje parecido al código máquina pero que no es este en realidad. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución por una máquina virtual denominada Java Virtual Machine (JVM) o Java Runtime Environment (JRE). En esta misma situación se encuentra C#. Esto representa un retardo considerable en el tiempo de ejecución, pero aún así, les da cierta ventaja en este sentido respecto a PHP, el cual cada vez que se ejecuta tiene que ser traducido directamente de código fuente a código máquina.

Sin embargo, la mayor carga de operaciones y los requerimientos de velocidad se centran principalmente en el servidor del NVP, dejando a los clientes solamente la responsabilidad de atender las solicitudes de los usuarios, enviándolas al servidor, y mostrando los resultados devueltos. Esto da un cierto grado de flexibilidad a la hora de analizar la velocidad de ejecución de los lenguajes de programación comparados.

1.5.4.5 Rendimiento.

Como se ha abordado anteriormente, la utilización de C# requiere de la instalación del framework .NET, lo que unido a la instalación de un servidor web que soporte ASP.NET, consume parte de la capacidad del disco duro de la computadora. Por otra parte, ASP.NET es un código de Common Language Runtime compilado que se ejecuta en el servidor. A diferencia de sus predecesores, puede aprovechar las ventajas del enlace anticipado, la compilación just-in-time, la optimización nativa y los servicios de caché desde el primer momento.

En cuanto a Java, aunque se han realizado versiones ligeras de las máquinas virtuales de este lenguaje, especialmente para dispositivos portátiles, por lo general consumen muchos recursos de la computadora, principalmente espacio de disco duro y capacidad de memoria RAM cuando se ejecuta.

PHP por su parte, es un lenguaje bastante ligero, lo que unido al poco espacio que requiere su instalación, lo hace un lenguaje bastante eficiente en cuanto a su rendimiento.

1.5.4.6 Licencia.

ASP.NET es una tecnología propietaria de Microsoft.

Java tiene un largo camino andado en relación con el desarrollo de su arquitectura sobre diferentes plataformas. La tecnología Java es abierta y se basa en gran medida en estándares de organizaciones de normalización y "de facto". (Gámez, 2008) Existen diferentes implementaciones alternativas a la desarrollada por Sun Microsystem, algunas de ellas bajo licencias de software libre.

PHP es publicado bajo la PHP License, considerada como software libre por la Free Software Foundation.

1.5.4.7 Modularidad.

Este criterio está referido a la posibilidad de desarrollar componentes de manera independiente, los que eventualmente interactuarían.

Los tres lenguajes analizados permiten desarrollar funciones, clases, y paquetes de modo independiente, cada cual con sus convenciones particulares.

1.5.4.8 Mantenibilidad.

Aunque para algunos desarrolladores PHP no es un lenguaje fácil de entender, esta característica está influenciada por las convenciones y los estilos con que los programas son implementados. Haciendo un uso adecuado de dichos estilos y de estándares de codificación, el código en PHP es fácilmente entendible, lo que facilita el mantenimiento de los programas escritos en este lenguaje.

Java y C# son lenguajes muy similares, al tomar gran parte de su sintaxis de C++. De manera general ambos son fácilmente entendibles, garantizando también facilidad en el mantenimiento de las aplicaciones desarrolladas.

1.5.4.9 Coste de desarrollo

El coste estimado en un proyecto PHP siempre será menor que en un proyecto Java. Mientras que la programación de un sistema PHP es mucho más directa con resultados inmediatos, el uso de Java supone la instalación de diferentes programas, como por ejemplo la máquina virtual, lo que alarga el tiempo de desarrollo y con esto, su coste.

Otro punto a tener en cuenta en la estimación de costes es el hecho de que, para la programación de un sistema Java es necesaria mayor preparación y experiencia, lo que puede aumentar el coste total.

C# también requiere de la instalación de diferentes herramientas para su correcto funcionamiento, como por ejemplo el framework .NET. Al igual que Java, el equipo de desarrollo debe tener experiencia en el uso de este lenguaje. Además, al ser un lenguaje privado, se incrementa su coste al ser obligatorio el pago de licencias y de sus actualizaciones, sin sumar el coste original de estas herramientas.

1.5.4.10 Integración externa

Integración externa es el uso de herramientas, métodos y funcionalidades desarrollados por otros programadores y que son integrables fácilmente en el sistema a desarrollar.

Tanto para Java como para PHP existe una gran comunidad de desarrolladores que constantemente actualizan y extienden las funcionalidades y bibliotecas de estos lenguajes.

Al ser privados tanto el lenguaje C# como el framework .NET, no existe un gran desarrollo de herramientas externas que puedan integrárseles, lo que los deja en desventaja en este aspecto.

1.5.4.11 Seguridad

El aspecto de la seguridad siempre ha sido un punto a tener muy en cuenta en cualquier sistema informático y un portal web es especialmente vulnerable por estar expuesto a todo el público en internet.

Uno de los puntos más vulnerables que con relación a la seguridad presentan los portales y aplicaciones web es la validación de usuarios, si no se ha desarrollado el sistema correctamente. Java implementa en diferentes niveles un sistema seguro de validación que en PHP falta.

Como sistemas de seguridad usados en proyectos Java, cabe destacar los que se implementan a nivel de Servidor de aplicaciones (como “JAAS”) y los que están incluidos en Frameworks externos (como por ejemplo “Spring Security” o “ACEGI”), ambos eficaces y transparentes a usuarios y programadores. En el desarrollo de un portal web con PHP se debe controlar la seguridad de acceso a nuestro sistema de una forma mucho más manual, realizando comprobaciones minuciosas de los diferentes ataques que podemos recibir (como por ejemplo SQL Injection). (Gómez López, 2010)

ASP.NET otorga un mayor grado de control para implementar la seguridad en las aplicaciones. La seguridad de ASP.NET funciona en conjunción con la seguridad de Microsoft Internet Information Server (IIS) e incluye servicios de autenticación y autorización para implementar el modelo de seguridad de ASP.NET. También incluyen una característica de seguridad basada en funciones que puede implementar para cuentas de usuario de Microsoft Windows y que no son de Windows.

IIS mantiene las opciones de configuración relacionadas con la seguridad en su metabase. Sin embargo, ASP.NET mantiene las opciones de la configuración de seguridad en los archivos de configuración del lenguaje de marcado extensible (XML). Aunque esto generalmente simplifica la implementación de la aplicación desde el punto de vista de la seguridad, el modelo de seguridad que adopta necesita la configuración correcta de la metabase de IIS y la aplicación ASP.NET a través del archivo de configuración (Web.config).

1.5.4.12 Escalabilidad

ASP.NET se ha diseñado específicamente a la medida teniendo en cuenta la escalabilidad, con el fin de mejorar el rendimiento en entornos agrupados y de múltiples procesadores. Además, su motor de tiempo de ejecución controla y administra los procesos de cerca, por lo que si uno no se comporta adecuadamente (filtraciones, bloqueos), se puede crear un proceso nuevo en su lugar, lo que ayuda a mantener la aplicación disponible constantemente para controlar solicitudes.

Existen diferentes criterios respecto a la escalabilidad en PHP. No obstante, en el mundo existen múltiples ejemplos de sitios web desarrollados en este lenguaje que reciben miles de visitantes diariamente, sin afectar el rendimiento de los mismos. Por ejemplo: Digg.com recibe alrededor de 200 millones de visitas de páginas por mes, y funciona solamente con 3 servidores web y 8 pequeñas bases de datos. Su primer año fue soportado con un solo servidor. De manera general, los requerimientos de correr una aplicación escrita en PHP con altísimo tráfico no son más costosos que los de Java.

Java también ha sido utilizado en una gran cantidad de sitios web, extensamente visitados. No tiene problemas en relación a la escalabilidad.

No obstante, el servidor web a utilizar por el nodo virtual de procesos solo requerirá de pocas conexiones concurrentes, por lo que se puede ser flexible a la hora de valorar este criterio de evaluación.

1.5.5 Elección del lenguaje a utilizar.

Para seleccionar el lenguaje a utilizar, se realizó una tabla resumen (tabla 1) donde se le asignaron valores a los diferentes criterios de selección en función de su importancia para el desarrollo de la aplicación. En la tabla, el peso representa el valor máximo que se le puede asignar a cada criterio, y fue seleccionado atendiendo a la importancia que el mismo presenta respecto a los requisitos del sistema, tomando en cuenta además la opinión de expertos en cada criterio analizado. Se obtuvieron los siguientes resultados:

Criterio de selección	Peso	C# (ASP.NET)	Java	PHP
Ejecución en múltiples plataformas	10	5	10	10
Capacidades de generación de gráficos	9	9	9	9
Gestión de memoria	10	10	10	10
Velocidad de ejecución	10	8	8	6
Rendimiento	10	8	6	10
Licencia	9	0	9	9
Modularidad	5	5	5	5
Mantenibilidad	5	5	5	5
Coste de desarrollo	8	3	5	7
Integración externa	3	0	3	3
Seguridad	5	5	5	4
Escalabilidad	3	3	3	3
Total		**61**	**78**	**81**

Tabla 1: Comparación entre los lenguajes de programación.

Teniendo en cuenta este resultado se seleccionó PHP como lenguaje a utilizar en la aplicación web, debido a todas las facilidades que ofrece y que aparecen reflejadas en la tabla 1.

1.5.6 Elección del tipo de comunicación entre los clientes y el servidor y de la herramienta para graficar.

En las aplicaciones web tradicionales la comunicación entre los clientes y el servidor es síncrona: el cliente realiza una operación, la página envía la información al servidor, espera la respuesta de este y cuando la recibe se recarga en su totalidad. Esto genera una sobrecarga innecesaria de trabajo para el servidor y del tráfico de la red, al ser necesario que se envíen todos los componentes de la página, incluyendo archivos de tamaño relativamente grande como imágenes.

Con el objetivo de encontrar una alternativa a la situación anteriormente planteada, se realizó una investigación para conocer las tendencias que a nivel mundial existen en este sentido. Se obtuvo como resultado que se ha ido incrementando el uso de la tecnología AJAX (Asynchronous JavaScript And XML o JavaScript asíncrono y XML, en español) para mejorar la comunicación de la web.

Esta tecnología permite hacer solicitudes al servidor y recibir los resultados de las mismas de manera totalmente transparente al usuario. Una vez completada la operación, los datos obtenidos son utilizados para actualizar la página web mediante el uso del lenguaje JavaScript y de manipulaciones del DOM (Document Object Model o Modelo de Objetos del Documento, en español). De esta forma sólo se transmite la información mínima indispensable, constituyendo un ahorro considerable del ancho de banda de la red y del trabajo a realizar por el servidor.

Por otra parte, en el momento de analizar los diferentes criterios para seleccionar el lenguaje de programación a utilizar, se tuvo en cuenta las capacidades que cada uno de ellos tenía para generar gráficos. Esta cualidad es necesaria cuando la computadora cliente presenta tan pocos recursos que se hace imprescindible la generación de los gráficos en el servidor web. No obstante, es recomendable reducir al mínimo el trabajo a realizar por el servidor web, para mejorar su rendimiento y tiempo de respuesta. Por tal motivo, se investigó con el objetivo de encontrar una herramienta capaz de generar los gráficos correspondientes a los datos obtenidos por la simulación, en el cliente, sin que requiriera la instalación de programas de gran tamaño ni el consumo de muchos recursos de estas computadoras. A continuación se hace una breve descripción de las más relevantes encontradas:

- Dygraphs: biblioteca escrita en JavaScript, que tiene un tamaño inferior a los 100 KB. Genera gráficos de excelente calidad, indicándole únicamente los datos en formato csv (datos separados por comas). Permite la personalización de los gráficos generados. Compatible con un gran número de navegadores, incluyendo Mozilla Firefox e Internet Explorer. Tiene la desventaja de que la

actualización periódica de los gráficos incrementa el consumo de memoria RAM por parte del navegador, superando en ocasiones un gigabyte de consumo en el caso de Mozilla Firefox. (Vanderkam)

- jQuery UI: es una biblioteca de componentes para el framework jQuery que le añade un conjunto de componentes y efectos visuales para la creación de aplicaciones web. jQuery es un framework de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el DOM, manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web. Es compatible con los navegadores (y sus versiones posteriores) Internet Explorer 6.0, Mozilla Firefox 2, Safari 3.1, Ópera 9.0 y Google Chrome. (jQuery)

- Yahoo User Interface (YUI): una serie de bibliotecas escritas en JavaScript para la construcción de aplicaciones interactivas (RIA). Están sujetas a la licencia BSD. Son utilizadas para el desarrollo web, imitando la programación de aplicaciones de escritorio. Presentan componentes gráficos atractivos y personalizables y un amplio uso de AJAX. Entre los componentes que brinda se encuentran clases para la generación de diferentes tipos de gráficos. (YUI)

- ExtJs: es una biblioteca escrita en el lenguaje JavaScript para el desarrollo de aplicaciones web interactivas usando tecnologías como AJAX, DHTML y DOM. Aunque fue construida originalmente como una extensión de la biblioteca YUI, puede ejecutarse como una aplicación independiente desde la versión 1.1. También puede usarse como extensión para otras bibliotecas como jQuery. Dispone de un conjunto de componentes para incluir dentro de una aplicación web como cuadros y áreas de texto, campos para fechas, campos numéricos, combos, radiobuttons y checkboxs, entre otros. Varios de estos componentes pueden comunicarse con el servidor web utilizando AJAX mediante funciones predefinidas propias de los mismos. Tiene clases especiales para la elaboración de gráficos. Es compatible con los principales navegadores utilizados, como Mozilla Firefox e Internet Explorer. (ExtJs)

Atendiendo a las características antes mencionadas, se puede concluir que las tres últimas bibliotecas mencionadas son las más recomendables, ya que permiten la generación de gráficos y el uso de la tecnología AJAX. En el caso de Dygraphs, además de los problemas que presenta con el consumo de memoria, se centra únicamente en la generación de gráficos.

Se recomienda el uso de la biblioteca ExtJs, ya que, además de presentar todas las características antes mencionadas, es fácil obtenerla, es software libre y presenta la más completa colección de componentes para su uso en aplicaciones web. Por otra parte, sus componentes tienen métodos que realizan internamente la comunicación con el servidor usando AJAX y la actualización automática de la página, liberando al desarrollador de implementar esta funcionalidad.

1.5.7 Elección del entorno de programación.

Para la elección del entorno de desarrollo integrado (IDE) se tuvo en cuenta, además de las características propias que debe tener un IDE, como autocompletamiento, identificación de errores, facilidad de uso, entre otras; que fuera software libre y multiplataforma.

- Quanta Plus (Quanta+) es una herramienta libre para el desarrollo de páginas web, diseñada para el proyecto KDE que forma parte del paquete kdewebdev. Presenta resaltado de sintaxis de HTML, JavaScript, CSS y varios más. Contiene un analizador que informa acerca de la correcta creación de nuestras páginas. Soporta plugins como: Konsole, Cervisia (CVS), KFileReplace, KLinkStatus y KImageMapEditor. Permite preprocesar los documentos a través de un servidor web y previsualizarlos dentro de la aplicación usando el motor KHTML. Sólo compatible con GNU/Linux. (Quanta)

- Bluefish es un software libre utilizado para la edición de HTML. Corre en muchos de los sistemas operativos compatibles con POSIX (Portable Operating System Interface) tales Linux, FreeBSD, MacOS-X, OpenBSD, Solaris y Tru64. Es capaz de reconocer diversos lenguajes de programación y de marcas. (Bluefish)

- Gedit es un software libre utilizado para la edición de textos, disponible en sistemas tipo Unix. Se caracteriza por su facilidad de uso. Presenta un corrector ortográfico multi – idioma, numeración de líneas y resaltado de la sintaxis de varios lenguajes de programación como C, C++, Java, Python, PHP, entre otros. (Gedit)

- Zend Studio (o Zend Development Environment) es un IDE para PHP, soportando PHP 4 y PHP 5. Está escrito en Java, y disponible para diversas plataformas, como Windows, MacOS-X y GNU/Linux, aunque es un software privado. No requiere la instalación previa de PHP ni del entorno

de ejecución de Java. Presenta resaltado de sintaxis, autocompletamiento de código, detección de errores de sintaxis en tiempo real, ayuda de código (integra phpDoc) y lista de parámetros de funciones y métodos de clase. (Studio)

- Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma. Ofrece editor de texto con resaltado de sintaxis, compilación en tiempo real, pruebas unitarias, control de versiones, asistentes para creación de proyectos, clases, pruebas y refactorización. A través de plugins es posible añadir control de versiones con subversion e integración con Hibernate. Mediante el plugin PHP Development Tools (PDT), se convierte en un potente IDE para el lenguaje PHP. (Eclipse)

- NetBeans es un IDE libre y gratuito sin restricciones de uso, escrito en Java, que facilita el desarrollo de programas en múltiples lenguajes de programación gracias al importante número de módulos que extienden sus funcionalidades. Permite crear aplicaciones web con PHP gracias al PHP Pack, soportando también PHP 5, Symfony y AJAX, entre otros. Posee un potente debugger integrado. Ofrece un sistema de proyectos basado en Ant, control de versiones y refactorización, autocompletamiento, sangrado automático, resaltado e identificación de errores, entre otras características. Es un programa más liviano que otros IDE igualmente potentes, como Eclipse. (NetBeans)

Después de un análisis de las características anteriormente abordadas, se seleccionó NetBeans como IDE debido a las facilidades que soporta en cuanto a la edición y corrección del código, su facilidad de uso y previo conocimiento de sus funcionalidades por parte del equipo de desarrollo, su alto rendimiento respecto a velocidad de ejecución y poco consumo de recursos, y a ser software libre, multiplataforma y poseer un gran número de módulos externos que extienden y perfeccionan sus funcionalidades.

1.6 Selección del servidor web.

No se puede hablar de aplicaciones web sin haber seleccionado un servidor web potente y eficiente. A continuación se hace un breve análisis de las características de los servidores web más utilizados, para seleccionar al que mejor se adecue al problema planteado.

- Tomcat: es un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Implementa las especificaciones de los servlets y de Java Server Pages de Sun Microsystems. Está diseñado esencialmente para el lenguaje de programación Java. (Tomcat)

- Internet Information Server (IIS): es una serie de servicios para los ordenadores que funcionan con Windows. Ofrece los servicios: NNTP, SMTP, FTP y HTTP/HTTPS. Basa su funcionamiento en varios módulos que le permiten procesar distintos tipos de páginas. Incluye Active Server Pages (ASP) y ASP.NET. También pueden ser incluidos los de otros fabricantes diferentes de Microsoft, como PHP o PERL. Es un sistema propietario. (Corporation)

- Apache: es un servidor de código abierto para plataformas Unix, Windows, Macintosh y otras. Se desarrolla dentro del proyecto HTTP Server de la Apache Software Foundation. Es modular, extensible y su popularidad ofrece la ventaja de que es muy fácil conseguir ayuda y soporte técnico en internet, lo que facilita su uso y mantenimiento. Tiene una excelente integración con PHP, hasta tal punto que en numerosas ocasiones se distribuyen ambos en un solo paquete de instalación. (Apache)

Por las características anteriormente descritas se puede concluir que el servidor Apache es el que reúne las mejores condiciones para ser utilizado por el NVP, debido a ser libre, fácilmente integrable con PHP y multiplataforma, entre otras ventajas.

1.7 Selección del estándar de conexión a bases de datos.

La conexión y el intercambio de información entre las aplicaciones y los gestores de bases de datos (SGBD) han tenido su máximo exponente con el surgimiento del lenguaje de consulta estructurado (SQL). Gracias a este y al surgimiento de otras condiciones favorables, como su adopción por parte de un gran número de fabricantes de SGBD y a un mutuo consenso de estos en el diseño de los SGBD, ha sido posible la creación y desarrollo de varios estándares para establecer dicha conexión.

Como se ha abordado anteriormente, el equipo de desarrollo decidió utilizar un driver estándar para la conexión del NVP a la base de datos, debido a las facilidades que ofrece esta opción.

A continuación se hace un breve estudio de los principales estándares y bibliotecas de acceso a datos existentes actualmente compatibles con las herramientas seleccionadas hasta el momento, y sus características más relevantes.

- Object Linking and Embedding for Databases (Enlace e incrustación de objetos para bases de datos u OLEDB) es una tecnología desarrollada por Microsoft para tener acceso a diferentes bases de datos de manera uniforme. Permite separar los datos de la aplicación que los requiere. Está dividido en consumidores y proveedores; el consumidor es la aplicación que requiere acceso a los datos y el proveedor es el componente de software que expone una interfaz OLEDB a través del uso del Component Object Model (COM). (OpenLink Software, 2010)

- Open Database Connectivity (Conectividad abierta a base de datos u ODBC) es un estándar de acceso a bases de datos desarrollado por Microsoft Corporation, aunque se han desarrollado diferentes implementaciones del mismo, tanto libres como propietarias. Por otra parte, los fabricantes de un gran número de gestores de bases de datos, como Access, PostgreSQL, MySQL, Oracle y Microsoft SQL Server, han implementado drivers para los mismos compatibles con este estándar. Permite acceder a cualquier base de datos desde cualquier aplicación, sin importar qué sistema gestor de bases de datos (SGBD) se utilice. Inserta una capa intermedia entre la aplicación y el SGBD, cuyo propósito es traducir las consultas de datos de la aplicación en comandos que el SGBD entienda. Para que esto funcione tanto la aplicación como el SGBD deben ser compatibles con ODBC, esto es que la aplicación debe ser capaz de producir comandos ODBC y el SGBD debe ser capaz de responder a ellos. Desde la versión 2.0 el estándar soporta SAG y SQL. (Schultz, 2000) (Ripley, 2009) (Microsoft Corporation, 2010)

Atendiendo a las anteriores características señaladas de cada estándar, se decidió utilizar ODBC para establecer la conexión a la base de datos, por las ventajas antes expuestas, ofrecidas por el mismo. Por otra parte, es necesario destacar que el framework Qt, utilizado para desarrollar el NVP, así como el sistema gestor de bases de datos, PostgreSQL, son compatibles con dicho estándar.

1.8 Estándares de codificación.

"*Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo*

práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un proyecto de software, se debe establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Cuando el proyecto de software incorpore código fuente previo, o bien cuando realice el mantenimiento de un sistema de software creado anteriormente, el estándar de codificación debería establecer cómo operar con la base de código existente." (Hernández Garrigó, et al., 2009)

Según Bjarne StrousTrup, creador del C++ (Sutter, et al., 2004), no existe un estándar de codificación para todos los usos y todos los usuarios. Para una aplicación determinada, usar un mal estándar de codificación es peor que no usar estándar de codificación. Por tal razón se debe elegir las reglas de codificación cuidadosamente y con un conocimiento sólido del área donde se desarrolla la aplicación.

En el caso de la aplicación web, desarrollada en PHP, se decidió aplicar reglas con respecto a los identificadores, la identación, las líneas, los espacios en blanco y los comentarios, con la intención de mantener una armonía entre el código escrito en C++ y el de PHP, teniendo presente las características propias de este último lenguaje y los hábitos que en este sentido tiene el equipo de desarrollo. Atendiendo a esto se decidió emplear identificadores para clases y métodos donde cada palabra del mismo comenzará con letra inicial mayúscula. En el caso de los identificadores de variables y atributos se aplicará lo contrario: todas las palabras comenzarán con letra inicial minúscula. En los identificadores compuestos las palabras se separarán con el caracter _. Para la identación se utilizará la escritura de cuatro espacios en blanco cada vez que se presione la tecla Tab. Las llaves se abrirán una línea debajo de la declaración del método o clase, manteniéndose al mismo nivel de esta. Siempre que sea posible las líneas tendrán una longitud que permita su visualización completa en la pantalla, facilitando la legibilidad del código. Los espacios en blanco sólo serán permitidos para separar bloques de código que realicen una misma función, como por ejemplo: clases, métodos o bloques dentro de métodos que realicen funciones diferentes según la lógica del programa. Los comentarios seguirán la misma regla que las líneas, utilizando los caracteres //, excepto en los casos en que el bloque del comentario sea compacto y extenso.

1.9 Conclusiones parciales.

En el presente capítulo se realizó un análisis de las diferentes soluciones posibles para resolver las no conformidades presentadas por la versión anterior del NVP, escogiéndose las más favorables a la

situación planteada. Se analizaron las diferentes herramientas, paradigmas y lenguajes de programación necesarios para desarrollar dicha solución, así como los estándares a utilizar para asegurar la conexión a la base de datos sin perder la condición de portabilidad de la aplicación, y el estándar de codificación necesario para mantener la legibilidad y reutilización del código fuente en todo el software, facilitando su posterior mantenimiento y actualización.

Teniendo en cuenta los criterios de comparación y los resultados de las investigaciones realizadas, fueron seleccionados: el desarrollo de una aplicación web, el paradigma orientado a objetos y PHP como lenguaje de programación, ODBC como estándar de comunicación con la base de datos, NetBeans como entorno de desarrollo, Apache como servidor web y ExtJs como framework del lado del cliente.

CAPÍTULO 2: SOLUCIÓN PROPUESTA

2.1 Introducción

En este capítulo se realiza un análisis del diseño propuesto para la nueva versión del sistema Nodo Virtual de Procesos. Además, se hace una breve descripción de las principales clases y de los componentes reutilizados, con la finalidad de describir la solución propuesta para la implementación de esta versión.

2.2 Valoración de la arquitectura y el diseño propuestos

La arquitectura es el esqueleto o base de una aplicación. En ésta se analiza la aplicación desde varios puntos de vista. En la arquitectura aparecen los artefactos más importantes y diferentes, para establecer un esquema de cómo deben ser los próximos artefactos a construir. De obtenerse un artefacto demasiado diferente de los demás, éste formaría parte de la arquitectura. (Jacobson, et al., 2004)

Para estructurar el diseño de la versión anterior del NVP, se escogió la arquitectura en 3 capas. Las interfaces y los formularios que permiten la interacción con el usuario se encuentran en la capa de Presentación. Las clases encargadas de implementar las reglas del negocio y las restricciones que desea la parte interesada se encuentran en la capa Lógica o de Negocio, éstas son las clases controladoras y entidades que permiten la realización de los casos de uso. Por su parte, la capa de acceso a datos contiene todas las funcionalidades necesarias para garantizar la persistencia de la información. (Edith, et al., 2008)

Esta arquitectura presenta flexibilidad y robustez a la hora de realizar cambios, por lo que se decidió reutilizarla en la actual versión. Por otra parte, presenta la ventaja adicional de que se le pueden insertar capas intermedias adicionales que permitan la comunicación entre las capas adyacentes sin modificar estas, ventaja que adquiere especial importancia para la actual versión, al necesitarse la comunicación entre capas implementadas en diferentes lenguajes de programación.

Adicionalmente se decidió aplicar el estilo Modelo – Vista – Controlador (MVC) en la capa de Presentación de la nueva versión. Este es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. La vista se encarga simplemente de mostrar los datos al usuario, enviados por el modelo, y el controlador es el responsable de recibir los eventos de entrada desde la vista y enviárselos a su vez al modelo, en el cual se ejecutarán

las acciones correspondientes. En la aplicación se utilizó en las interfaces web, con el objetivo de independizar el diseño del código de las mismas.

Asimismo, se reutilizó el diseño de clases empleado por la versión anterior, debido a su robustez y su eficiencia desde el punto de vista de los requisitos que debe cumplir el sistema. Dicho diseño aparece en la siguiente figura:

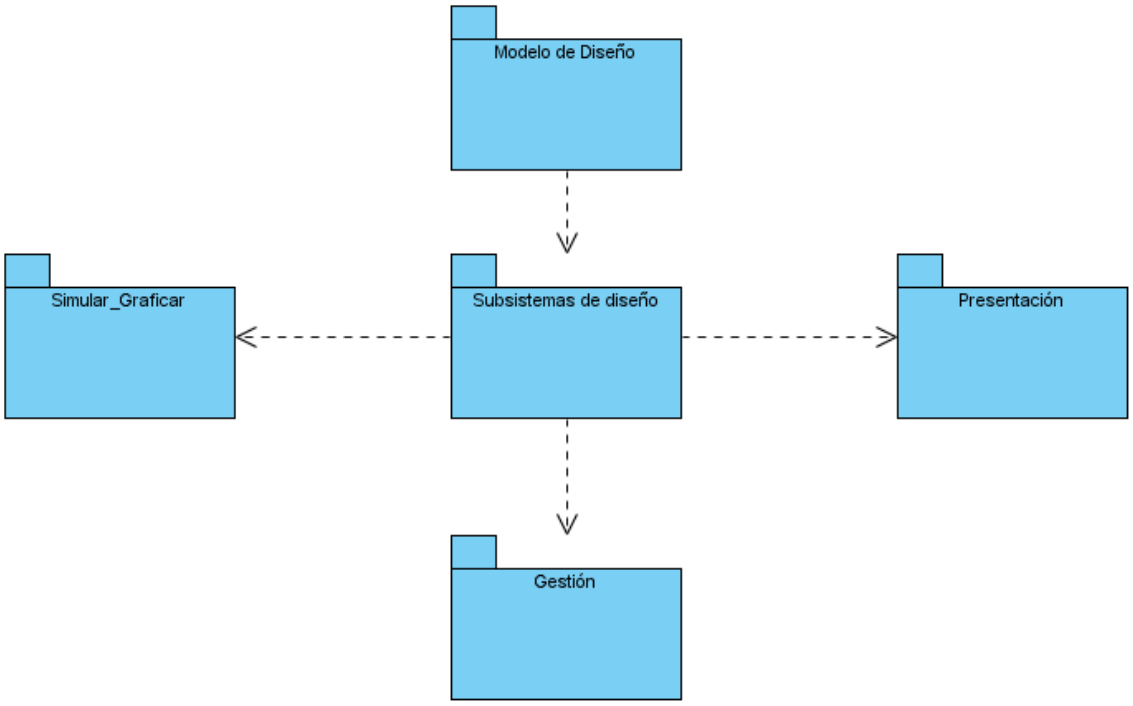


Fig. 1: Modelo de diseño del sistema (versión anterior).

Para adaptarlo a la web, se eliminó el subsistema Simular_Graficar, repartiendo sus funcionalidades entre los subsistemas Gestión y Presentación, quedando de la siguiente forma:

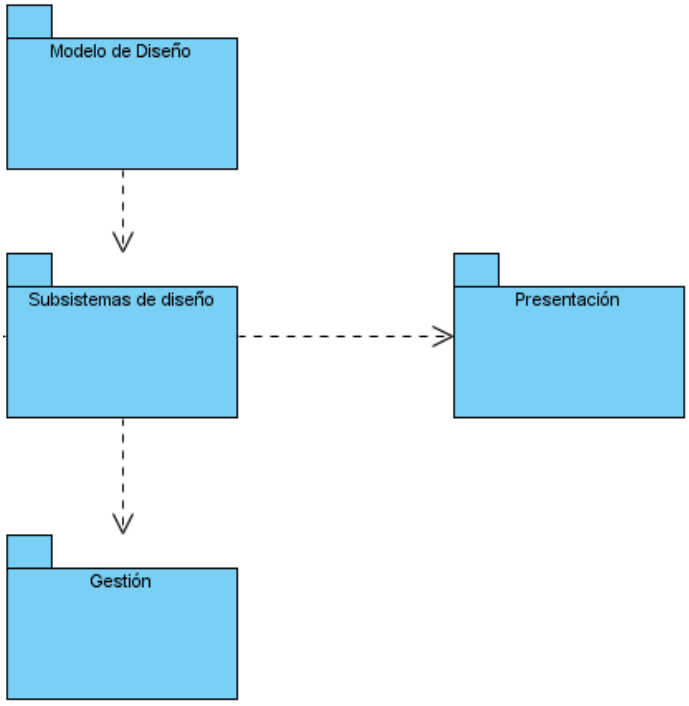


Fig. 2: Modelo de diseño del sistema (versión actual).

En este diseño se utilizaron los siguientes patrones:

Patrón Solitario (Singleton)

Con este patrón se garantiza una única instancia de aquellas clases de las que se desee tener una sola en toda la aplicación, proporcionando un punto de acceso global a las mismas. Tiene como ventajas que reduce el espacio de nombres y es una mejora sobre las variables globales. Es usado debido a la necesidad de trabajar con el mismo objeto en distintos momentos y distintos subsistemas (Edith, et al., 2008). Específicamente en el NVP, este patrón se utiliza en las clases gestoras y en las fachadas de los subsistemas, garantizando una única instancia de ellas. Esto suele hacerse cuando se aplica el Patrón Fachada, que se explicará a continuación.

Patrón Fachada (Facade)

Proporciona una interfaz sencilla y unificada para un conjunto de clases o subsistemas, siendo más fácil de usar. Permite reducir la complejidad y minimizar las dependencias, el acceso de los usuarios a los

subsistemas es por medio de la clase fachada. Ella es la encargada de reenviar las peticiones a los objetos de los subsistemas, para no acceder directamente a los mismos, ocultando la complejidad de ellos. Este patrón favorece un bajo acoplamiento entre los usuarios y los subsistemas, respondiendo a uno de los patrones GRASP. Va a permitir variar las clases internas, de manera transparente a los usuarios que las utilizan, favoreciendo de esta forma a la división en capas de la aplicación. Como se dijo anteriormente estas clases van a utilizar el patrón Solitario lo que va a permitir un acceso global a ellas. (Edith, et al., 2008)

Patrón Prototipo (Prototype)

Es un patrón de creación, cuyo objetivo es especificar los tipos de objetos a crear por medio de una instancia que hace de prototipo, creando nuevos objetos copiando dicha instancia; es decir, se basa en clonar un prototipo dado. Específicamente en el NVP, cuando un usuario inicia un nuevo proceso se hace una copia del modelo existente utilizando este patrón, manteniéndose la configuración que el administrador del sistema le estableció con anterioridad.

Con el objetivo de garantizar una mejor comprensión del sistema existente, se incluye a continuación una breve descripción de los subsistemas que lo componen.

Subsistema Gestión: Contiene las clases que intervienen en el proceso de autenticación y registro de los usuarios al sistema y las interfaces relacionadas con sus privilegios. Además, permite la conexión a las diferentes partes de la aplicación y a los procesos. En este subsistema se gestiona toda la información necesaria para el correcto funcionamiento del software. Permite obtener reportes de la información referente a los procesos y al sistema. También permite a los administradores llevar un mejor control del funcionamiento del sistema.

Subsistema Simular-Graficar: En este subsistema se realizan las operaciones de simulación de los procesos. Posibilita la interacción del usuario con el proceso simulado, permitiéndole monitorizar su realización.

Subsistema Presentación: Este subsistema contiene las Interfaces de Usuario que se generan durante la aplicación.

El subsistema Gestión fue reutilizado en su totalidad para la nueva versión. El subsistema Presentación fue escogido para ser implementado para la web. Por su parte, el subsistema Simular-Graficar fue repartido entre los dos anteriores, debido a que se decidió dejar la simulación como responsabilidad de las clases gestoras, y la graficación de los datos quedaría como tarea de la presentación. El subsistema Presentación se ubica en la aplicación cliente; comunicándose con el resto mediante el uso de sockets y del protocolo TCP/IP. Debido a esto, la implementación del cliente como aplicación web no afecta el sistema en general y permite la reutilización de casi todas las clases del sistema.

Para la adición de nuevos modelos de procesos no es necesario modificar el código de la aplicación. Solamente se debe compilar a dichos modelos en bibliotecas dinámicas e indicar al simulador dónde se encuentran, siendo cargadas por este en tiempo de ejecución. En el fichero fuente de cada modelo a adicionar sólo hay que incluir las ecuaciones que describen el comportamiento del proceso a simular, y un pequeño número de datos adicionales que ofrecen información sobre dichas ecuaciones, como el número de parámetros que reciben y la descripción de las mismas. Dichas ecuaciones representan matemáticamente al modelo en cuestión. La simulación consiste en sustituir en las ecuaciones las variables por los valores introducidos por el usuario, luego es resuelta por un método numérico en específico (en el NVP están disponibles el Runge-Kutta y el Runge-Kutta 4), y la solución encontrada es representada en un gráfico para que el usuario pueda visualizarla.

Estos modelos han sido compilados utilizando el GCC (GNU Compiler Collection o colección de compiladores GNU en español) del sistema operativo GNU/Linux, y su versión para Windows conocida como MinGW (Minimalist GNU for Windows o GNU mínimo para Windows). Para compilar los modelos en bibliotecas dinámicas se emplea el comando siguiente:

```
gcc -shared -o liblibreria.dll libreria.c
```

Donde liblibreria.dll es el nombre especificado de la biblioteca dinámica a obtener, con la extensión .dll para Windows y .so para GNU/Linux; y libreria.c es el archivo de código fuente del modelo de proceso, implementado en el lenguaje C.

Debido a lo anteriormente planteado, se reutilizó el diseño de clases existente, adaptando las clases del subsistema Presentación para la web. Solamente se excluyó de la aplicación web la funcionalidad de administrar modelos y tipos de procesos, debido a que es necesaria la presencia física en el servidor del

administrador, para poder adicionar las bibliotecas correspondientes a los modelos e indicarle al simulador la dirección de las mismas en la computadora donde radican.

Para el desarrollo de la aplicación web que funcionaría como cliente del NVP se decidió utilizar la tecnología AJAX (Asynchronous JavaScript And XML o JavaScript asíncrono y XML, en español), debido a la necesidad de optimizar el tiempo de respuesta del servidor web. La misma permite cargar pequeñas partes de las páginas web y mostrarlas. De esta forma, se evita volver a cargar todo el contenido de la página, liberando al servidor web de esta tarea, solicitándole solamente pequeñas cantidades de datos. La aplicación web queda compuesta por una página web, una clase escrita en PHP que se encarga de gestionar la comunicación con el servidor NVP, y scripts adicionales también escritos en PHP, que se encargan de ofrecer los datos a la página cuándo esta realiza las solicitudes de los mismos mediante la tecnología AJAX. La comunicación entre los fragmentos individuales, que cumplen la función de formularios, y los scripts PHP correspondientes, se establece mediante solicitudes (request) y respuestas (response), los que permiten enviar órdenes y parámetros al servidor, y obtener los resultados de las operaciones realizadas.

Para facilitar la comprensión por parte de los programadores de la actual versión, se incluye el diagrama de componentes y el de despliegue, con los cuales se representan los componentes utilizados o elaborados, así como la infraestructura de hardware necesaria para el buen funcionamiento del sistema en su totalidad.

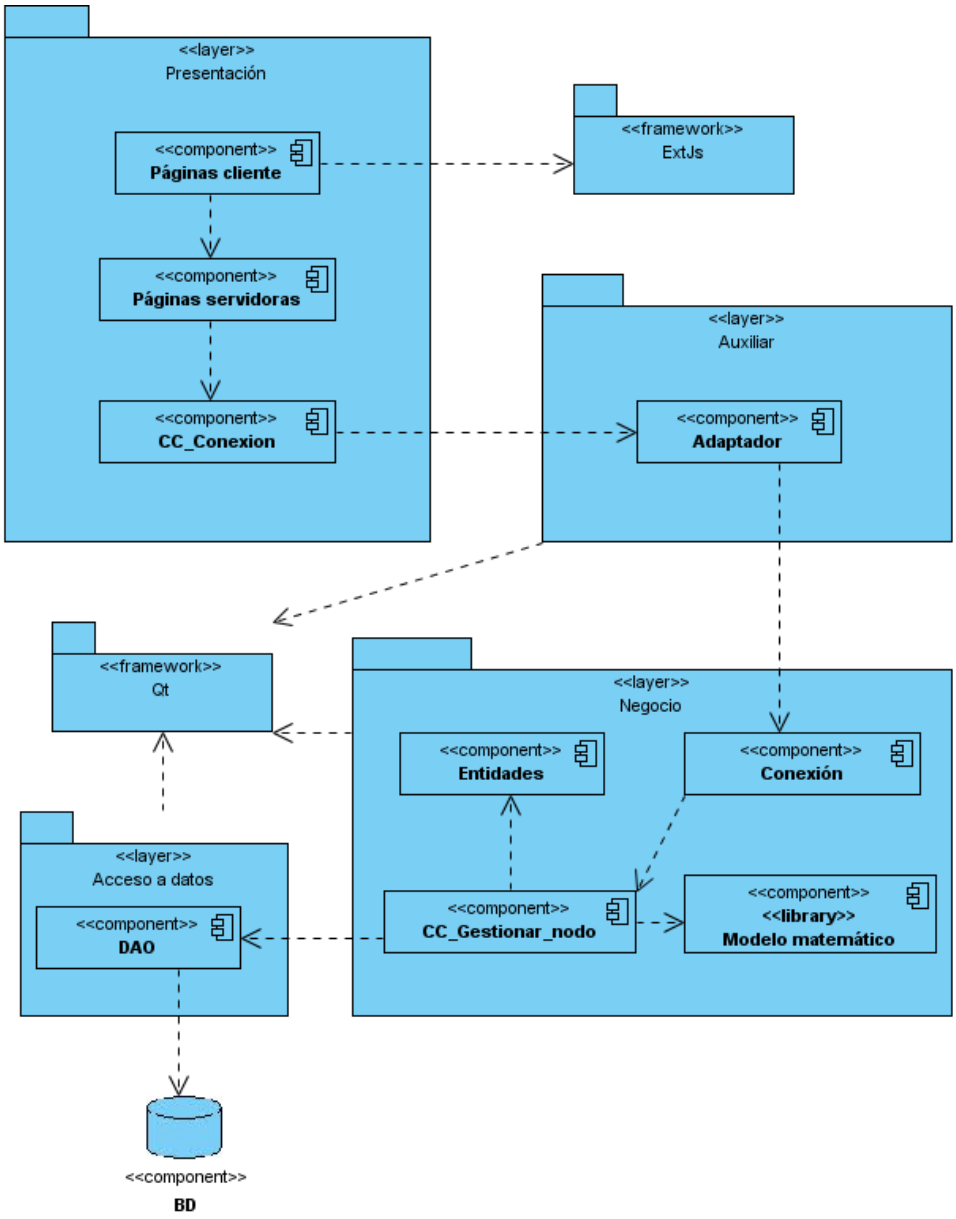


Fig. 3: Diagrama de componentes.

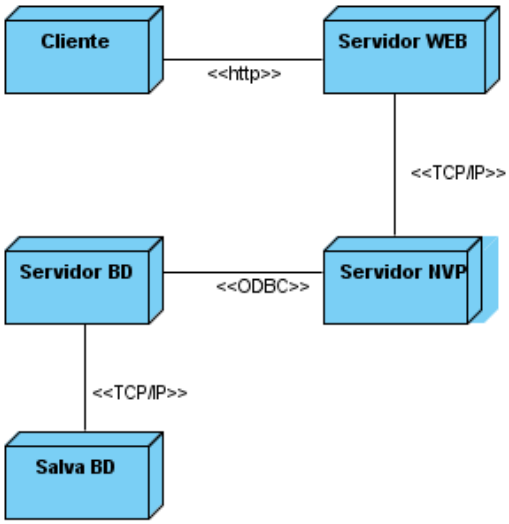


Fig. 4: Diagrama de despliegue.

2.3 Análisis de posibles implementaciones y componentes existentes

2.3.1 Clases y métodos usados para el trabajo multihilos

En la versión anterior, tanto el servidor como los clientes del simulador estaban implementados en C++, utilizando el framework Qt. Debido a esto se hizo uso de las facilidades que estas bibliotecas brindan para la comunicación mediante sockets, como el uso de la clase QDataStream, que permite enviar diversos tipos de datos a través del canal de comunicación, liberando al programador de la tarea de empaquetar dichos datos, siendo realizada esta operación por esta clase. Esto constituye un obstáculo a la hora de conectar el servidor NVP con la aplicación web cliente, escrita en el lenguaje PHP. Para darle solución a esta situación fue necesario implementar una aplicación intermedia que permitiera la comunicación entre el cliente y el servidor. Esa aplicación, denominada adaptador, utiliza múltiples hilos de ejecución concurrentes, cada uno de los cuales atiende a una instancia del cliente web. De esta forma, se garantiza que un mismo proceso pueda atender varias peticiones al mismo tiempo, sin que ninguna de ellas interfiera con las otras. Es decir, existirá un hilo independiente para cada usuario conectado.

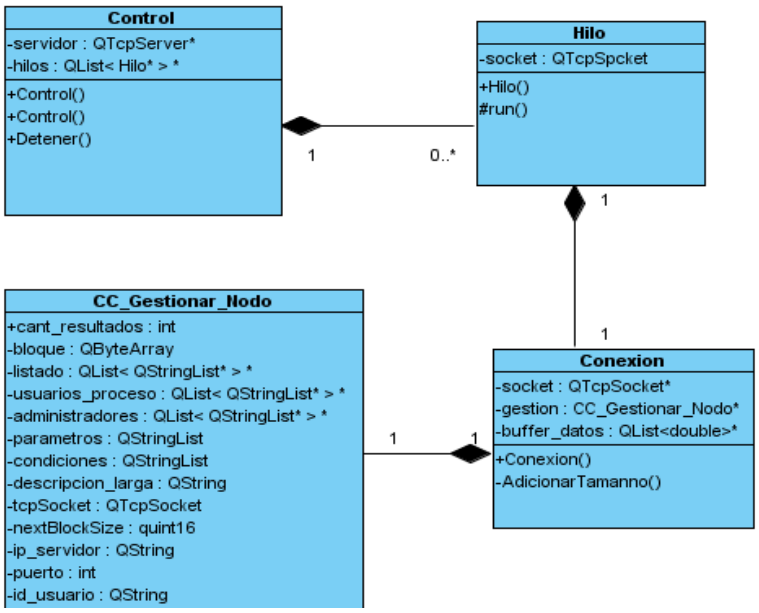


Fig. 5: Diagrama de clases del adaptador.

Para el trabajo con múltiples hilos de ejecución se utilizó la clase QThread brindada por Qt. Para hacer uso de esta funcionalidad se crea una clase que hereda de QThread. Esta permite obtener funcionalidades como start() para iniciar un hilo, terminate() para terminar la ejecución del hilo, isRunning() para saber si se encuentra ejecutándose, entre otras. En la clase hija se sobrescribe el método run(), dentro del cual se inserta el código que se va a ejecutar en un hilo independiente.

Para iniciar la ejecución del hilo concurrente se crea un objeto de la clase hija de QThread, y se llama al método start().

2.3.2 Comunicación entre las aplicaciones mediante el protocolo TCP/IP

Para la comunicación entre las diferentes partes del simulador se mantuvo el uso de los sockets y del protocolo TCP/IP. Estos mecanismos están disponibles en una gran cantidad de lenguajes de programación, incluyendo PHP y C++, lo que garantiza una mayor compatibilidad.

En el adaptador, que permite la comunicación entre las diferentes partes del simulador, se utilizaron las clases QTcpServer y QTcpSocket, disponibles en el framework Qt. El proceso de gestionar la comunicación se describe a continuación.

En la clase que gestionará la comunicación se crea como atributo un objeto de tipo QTcpServer. Este se encarga de atender las nuevas conexiones. Se le indica cuál es la dirección y el puerto donde se dirigirán las conexiones a atender y se le especifica el código a ejecutar cuando se establezca una nueva conexión. Esta última operación se realiza mediante el uso de señales.

Las señales son parte de un mecanismo que permite conectar una señal específica a un método. Un objeto lanza la señal, otro objeto contiene el método a ejecutarse y de manera independiente se realiza la conexión entre ambos. De esta forma, el objeto que lanza la señal no tiene que conocer al que la recibe, y este tampoco conoce al primero. Así se garantiza disminuir el acoplamiento entre las clases, fortaleciendo el diseño realizado.

Una vez que se recibe una solicitud de conexión, se ejecuta el método correspondiente a la señal newConnection() del objeto de tipo QTcpServer. Este contiene un método denominado nextPendingConnection(), que retorna un puntero a un objeto de tipo QTcpSocket, el cual será el encargado de manejar esa conexión en específico.

La clase QTcpSocket contiene métodos que facilitan el envío y la recepción de los datos a través de la red. Entre estos se encuentran write, writeData, writeLine, writeAll, read, readData, readLine y readAll.

Las señales readyRead, connected y disconnected permiten saber cuándo se han recibido nuevos datos, cuándo se ha establecido la conexión y cuándo se ha perdido la misma, respectivamente.

Por otra parte, con el método connectToHost se inicia una nueva conexión, especificándole la dirección y el puerto; y con el método disconnectFromHost se cierra la conexión.

Cada vez que se recibe una nueva solicitud de conexión, se crea un hilo de ejecución independiente para atenderla, garantizándose la existencia de múltiples usuarios conectados de manera concurrente.

A su vez, en el método run de la clase que hereda de QThread se crea una nueva instancia de la clase Conexion, que será la encargada de gestionar la comunicación con una conexión en específico. Dicha clase tiene un atributo de tipo CC_Gestionar_Nodo, que contiene los métodos necesarios para establecer la comunicación directa con el servidor del NVP. La clase CC_Gestionar_Nodo fue reutilizada de la versión anterior.

La clase Conexión tiene un método que atiende la señal readyRead del socket, ejecutándose cada vez que se reciben nuevos datos. Los demás métodos atienden las señales emitidas por el objeto de tipo CC_Gestionar_Nodo, encargándose de enviar las respuestas de este a través del socket a la aplicación web.

Los datos se transmiten entre la aplicación web y el adaptador en bloques, que tienen el siguiente formato:

[tamaño][código][datos]

- **Tamaño:** es una cadena de seis caracteres que contendrá el tamaño del bloque, sin incluir esos seis caracteres. Es la cantidad de bytes que el socket debe leer a continuación para recibir el bloque completo.
- **Código:** es un número de dos dígitos que indica la operación solicitada que se debe realizar.
- **Datos:** es el conjunto de los datos que la operación recibe como parámetros. Estos están separados por caracteres específicos. Los parámetros se separan mediante el caracter de tabulación ("\t"); en las listas y arreglos los elementos se separan mediante el caracter "&", y si son matrices las filas se separan mediante el caracter "!", utilizándose el caracter "&" como separador de los elementos individuales de cada fila, las celdas.

Para la implementación de los sockets en la aplicación web se utilizaron las funciones que para este fin están disponibles en PHP. Las principales se describen a continuación.

fsockopen: abre un socket especificándole la dirección y el puerto del servidor. Retorna una variable de tipo recurso que cumple la función de descriptor del socket. El socket abierto se cerrará una vez finalice la ejecución del script actual. Para abrir un socket persistente se utiliza la función pfsockopen, que funciona de la misma manera.

fclose: cierra el socket que se le especifique como parámetro.

fread: devuelve una cadena de caracteres con los datos recibidos por el socket. Se le especifica como parámetros el descriptor del socket y la cantidad de bytes a leer.

fwrite: envía una cadena de caracteres a través del socket. Recibe como parámetros el descriptor del socket y la cadena. Retorna la cantidad de bytes escritos.

stream_set_blocking: recibe como parámetros un descriptor de socket y un valor booleano que especifica si el socket será bloqueante o no. En caso de ser bloqueante, cada vez que se le solicite un servicio al socket, como leer o escribir, el socket se bloqueará hasta que finalice dicha operación. En caso contrario, retornará el resultado o un mensaje indicando el motivo por el cual no se pudo completar la operación, y continuará la ejecución del programa sin bloquearlo.

2.3.3 Clases y métodos usados para la comunicación asíncrona entre los clientes y el servidor web

En las páginas web tradicionales, cada vez que se realice una operación que implica un cambio en el contenido de la misma, se actualiza la página completa. Esto provoca que un mismo contenido sea enviado una y otra vez, lo que, en el caso de contenido multimedia como imágenes, constituye una carga innecesaria de trabajo para el servidor.

Una alternativa a esto es el uso de AJAX (Asynchronous JavaScript And XML o JavaScript asíncrono y XML, en español). Esta técnica permite actualizar el contenido de partes de la página sin tener que recargarla completa. Envía solicitudes asíncronas al servidor que son transparentes al usuario, el cual sólo ve la respuesta cuando ha sido actualizado el componente correspondiente de la página.

Para hacer uso de esta técnica se empleó la biblioteca ExtJs. Esta permite utilizar novedosos componentes visuales, como tablas, ventanas, cuadros de diálogo y de mensajes, entre otros; implementados en JavaScript. Además, brinda la opción de encapsular elementos del DOM en objetos especiales que aumentan el número de operaciones que se puedan realizar con los mismos. Entre estas se incluyen métodos para actualizar los componentes de la página mediante AJAX. Por otra parte, tiene clases que facilitan realizar solicitudes directamente al servidor usando esta tecnología, y recibir los resultados.

A continuación se realiza una breve descripción de los métodos y objetos utilizados por la solución propuesta para garantizar la comunicación con el servidor empleando AJAX, los cuales pertenecen a la biblioteca ExtJs.

Una de las principales clases que presenta ExtJs es Element. Esta envuelve los elementos del DOM, adicionándoles nuevas funcionalidades que facilitan las operaciones a realizar sobre los mismos.

Para convertir un elemento DOM a Element se llama a la función Ext.get, la cual recibe como parámetro el id del elemento DOM. Retorna este mismo elemento, pero ahora de tipo Element.

Los objetos de tipo Element poseen un método llamado load. A este método se le proporcionan como principales parámetros una dirección URL y una lista de parámetros. El mismo invoca al script que tiene la dirección URL indicada, pasándole los parámetros especificados. El texto devuelto por el script reemplaza al contenido situado entre las etiquetas del elemento DOM contenido en el objeto Element. De esta forma, el contenido de dicho elemento es actualizado mediante AJAX, sin necesidad de recargar la página web en su totalidad. A continuación se incluye un ejemplo del uso de estos métodos.

```
Ext.get('oKButton').on('click', function(){
        var msg = Ext.get("msg");
        msg.load({
           url: ‘tutorial-ajax.php’
           params: "name=" + Ext.get('name').dom.value
        });
        msg.show();
    });
```

En este ejemplo, se recupera el elemento DOM que tiene como id oKButton, envuelto en un objeto Element, y se le asocia a su evento click una función anónima. Dicha función recupera el elemento de id msg, se ordena que su contenido se actualice mediante AJAX a partir de la respuesta enviada por el script tutorial-ajax.php, pasándole a este último un parámetro name cuyo valor es el mismo que el del elemento DOM de id name. Finalmente, se le ordena al objeto msg que sea visible en la página web.
ExtJs presenta una clase independiente llamada específicamente Ajax, la cual se encarga de realizar varias operaciones usando esta técnica de forma directa. El método más importante que presenta es request.

Request envía una solicitud al servidor de forma asincrónica. Debido a esto no retorna directamente la respuesta, sino que hay que especificarle como parámetro la función que se deberá ejecutar una vez se

haya recibido la respuesta por parte del servidor. Esta función recibe como parámetros un objeto response y un objeto opts. El primero contiene la respuesta del servidor en el atributo responseText, y en caso de que ocurra un error el mismo podrá ser recuperado mediante el atributo status. El segundo parámetro contiene las opciones pasadas al servidor al ser invocado. El otro parámetro necesario para establecer la comunicación con el servidor es la dirección URL del script a ser ejecutado. A continuación, aparece un ejemplo que esclarece el uso de esta función:

```
Ext.Ajax.request({
   url: 'ajax_demo/sample.json',
   success: function(response, opts) {
      alert(response.responseText);
   },
   failure: function(response, opts) {
      alert(response.status);
   }
});
```

En el fragmento de código anteriormente expuesto se hace una solicitud AJAX al script ajax_demo/sample.json. En caso de que se ejecute la petición de forma satisfactoria se muestra una ventana de alerta con la respuesta del servidor. Si ocurre algún error, la ventana de alerta contendrá el código del error ocurrido.

2.3.4 Clases y métodos usados para la creación de gráficos

Como se señaló anteriormente, una de las necesidades principales a tener en cuenta al desarrollar la aplicación web fue liberar al servidor web de la mayor cantidad posible de tareas. Esto implica delegar dichas tareas a los clientes, específicamente a los navegadores. Esta distribución se realizó de manera que no se afectara el funcionamiento del servidor, pero que tampoco se excediera la capacidad disponible en las computadoras clientes, debido a las características de las mismas que ya se analizaron en capítulos previos.

Para la elaboración de los gráficos que mostrarán los resultados de la simulación, se utilizaron las clases que para tal fin presenta ExtJs, específicamente la clase LineChart. A la misma se le pasa como

parámetro un arreglo con los datos a graficar, y esta se encarga de elaborar el gráfico correspondiente. A continuación se incluye un ejemplo del uso de esta funcionalidad:

```
var store = new Ext.data.JsonStore({
                url: './obtener_datos_simulacion.php',
                root: 'campo',
                fields: ['indice', 'valor']
        });
store.load();
var win = new Ext.Window({
                items: {
                        id: 'chart',
                        xtype: 'linechart',
                        store: store,
                        xField: 'indice',
                        yField: 'valor'
                 }
});
win.show(this);
```

En este ejemplo, se crea un objeto store que se encargará de cargar los datos desde el servidor y almacenarlos. A continuación, se crea un objeto window, que contendrá al gráfico y se le agrega al mismo un literal objeto que representa al elemento de tipo linechart.

Para actualizar los datos del gráfico sólo será necesario llamar al método reload del objeto store, el cual utilizará la tecnología AJAX para recuperar los nuevos datos desde el servidor y actualizar el gráfico.

2.3.5 Clases y métodos usados para la comunicación con la base de datos

Las bibliotecas Qt presentan un amplio framework para el acceso a las bases de datos. Dicha comunicación puede realizarse directamente utilizando un driver nativo del sistema gestor de bases de datos (SGBD) utilizado, o de forma indirecta mediante algún protocolo estándar que garantice dicha comunicación. Estos drivers son integrados a las aplicaciones en forma de plugins. Algunos se deben

adicionar, mientras que otros se integran por defecto en la instalación de las bibliotecas Qt. Este es el caso de ODBC, estándar seleccionado que fue analizado en el Capítulo 1.

Una de las clases principales que contiene el módulo de acceso a datos es QSqlDatabase. Esta contiene métodos para indicar los parámetros necesarios para establecer la conexión con el SGBD, abrirla y cerrarla. Entre estos se encuentran setHostName, setPort, setDatabaseName, setUserName, setPassword, mediante los cuales se puede especificar la dirección del servidor, el puerto para la conexión, el nombre de la base de datos a consultar, el usuario y la contraseña, respectivamente. Los métodos open y close permiten abrir y cerrar la conexión.

Para crear un nuevo objeto de tipo QSqlDatabase se llama al método estático addDatabase, el cual adiciona la nueva conexión a la lista de conexiones existentes y además la almacena en el objeto creado. Este método tiene varios parámetros, pero el principal es el tipo de conexión a establecer. Es indicado mediante una cadena de caracteres con un código específico para cada tipo de conexión. A continuación se muestran los principales tipos de conexiones utilizados y el SGBD al que corresponde. Es necesario destacar que cada uno de ellos tiene un plugin correspondiente, por lo que para poder ser usado, dicho plugin deberá estar instalado correctamente en la computadora.

QDB2	IBM DB2
QIBASE	Driver Borland InterBase
QMYSQL	Driver MySQL
QOCI	Driver Oracle Call Interface
QODBC	Driver ODBC (incluido Microsoft SQL Server)
QPSQL	Driver PostgreSQL
QSQLITE	SQLite versión 3 o superior
QSQLITE2	SQLite versión 2
QTDS	Sybase Adaptive Server

En el caso del servidor NVP, el tipo de conexión utilizado por la versión anterior era QPSQL. En la versión actual se utilizará QODBC.

El estándar ODBC está integrado por defecto al sistema operativo Windows, facilitando su uso en esta plataforma. En el caso de GNU/Linux se usará unixODBC, herramienta my similar tanto en su aspecto como en sus funcionalidades a la que presenta Windows. Los drivers necesarios para la conexión entre el estándar ODBC y el gestor de bases de datos utilizado, en este caso PostgreSQL, se pueden encontrar en el sitio oficial del mismo, disponibles tanto para Windows como para GNU/Linux.

Sin olvidar la importancia de la seguridad ante ataques externos, se implementó en el servidor NVP un autómata que garantizara la ausencia de código sql en los datos introducidos por el usuario, evitando la ocurrencia de ataques de inyección sql. Además, se almacenaron las contraseñas en la base de datos utilizando el algoritmo MD5, de manera que las mismas no pudieran ser leídas por los posibles atacantes.

Para evitar que se comprometiera la integridad de los datos, se utilizaron los métodos transaction(), commit() y rollback(), implementados en la clase QSqlDatabase, que permiten realizar las operaciones en la base de datos en forma de transacciones, es decir, o se realiza la operación completa o no se ejecuta. A pesar de esto, se recomienda a los usuarios finales la realización de salvas diarias de la información en una computadora independiente y en soportes de almacenamiento externo, para evitar pérdidas en caso de que ocurra cualquier suceso imprevisto.

2.3.6 Clases y métodos utilizados para la generación de reportes y alertas

La clase CC_Conexion, implementada para la versión anterior y reutilizada en la actual, es la encargada de gestionar los mensajes que se intercambian entre los diferentes clientes y el servidor NVP. Por su parte, en la clase CC_Gestionar_Nodo, igualmente reutilizada de la anterior versión, contiene los datos necesarios para mantener el control del sistema, como el listado de los procesos, tanto activos como inactivos, los usuarios conectados y los diferentes límites dentro de los cuales el sistema debe funcionar.

Atendiendo a esto, y dando cumplimiento al patrón experto, se decidió implementar las funcionalidades de generación de reportes y alertas en la clase CC_Gestionar_Nodo, ya que es la que tiene el conocimiento y control de la información a utilizar. Se le asignó la tarea de gestionar esta información con los diferentes clientes a la clase CC_Conexion, respetando de esta forma el diseño adoptado.

2.4 Conclusiones parciales

En el presente capítulo se han abordado las principales razones para reutilizar el diseño de clases de la versión anterior. Se realizó un análisis de las principales clases y componentes que se podían reutilizar, así como de herramientas novedosas, tales como AJAX y ExtJs, que facilitarán el trabajo a realizar por el servidor web y proporcionarán una interfaz lo más amigable y atractiva posible para el usuario. Por otra parte, se describieron las funcionalidades del adaptador y cómo se establecerá la comunicación entre las diferentes partes de la aplicación y entre el servidor del NVP y el gestor de bases de datos, utilizando sockets y el estándar ODBC, respectivamente.

CAPÍTULO 3: PRUEBAS

Introducción:

Las pruebas de software son elementos esenciales para la garantía de la calidad del software, y representan a una revisión final de las especificaciones, del diseño y de la codificación.

Para ello se diseñan casos de prueba con una alta probabilidad de encontrar errores, aplicando técnicas de pruebas de software que le den fiabilidad a los mismos. Estas técnicas facilitan una guía sistemática para diseñar pruebas desde dos perspectivas:

1-) Comprobar la lógica interna de los componentes de software, que se realiza utilizando técnicas de diseño de casos de prueba de “caja blanca”.

2-) Verificar los dominios de entrada y salida del programa para descubrir errores en la funcionalidad, el comportamiento y rendimiento, utilizando técnicas de diseño de casos de prueba de “caja negra”.

En ambos casos, se intenta encontrar el mayor número de errores con la menor cantidad de esfuerzo y tiempo.

Al diseñar los casos de prueba se determinan los resultados esperados y se guardan los resultados realmente obtenidos. (Pressman, 2002)

Descripción de los niveles de pruebas realizadas:

Para garantizar la calidad de la aplicación se realizaron pruebas a diferentes niveles y utilizando diversas técnicas, los cuales se mencionan a continuación.

A nivel de desarrollador, la implementación se realizó utilizando la programación por pares, permitiendo que mientras un programador implementaba una funcionalidad, el otro la fuera revisando y detectando errores que serían solucionados inmediatamente.

Para realizar las pruebas a nivel de unidad se elaboraron casos de prueba atendiendo a los resultados obtenidos al aplicar las métricas de la técnica de caja blanca: prueba del camino básico, prueba de bucles, prueba de condición y prueba de flujo de datos. De acuerdo a la complejidad ciclomática obtenida y a los datos a incluir en el caso de prueba, se realizaron las pruebas unitarias, utilizando el módulo PHPUnit.

Este es miembro de la familia de frameworks de prueba XUnit. Entre las principales características que presenta este módulo están:

Fácilmente extensible: Tipos predefinidos pueden ser fácilmente adicionados a la información del test y a la salida del test.

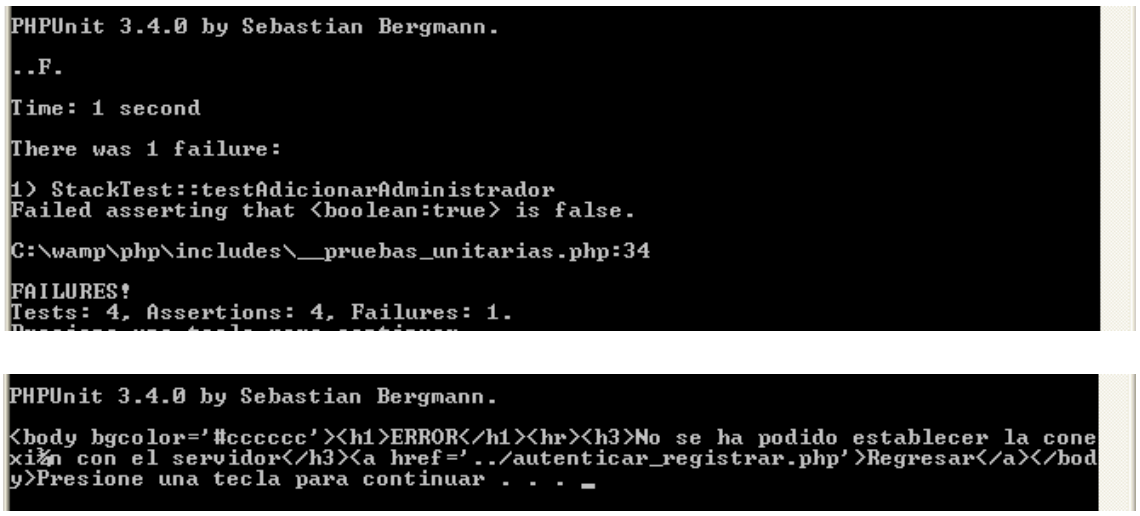
Ligero: Ocupa sólo 2,6 MB de espacio en el disco duro.

Prueba repetida de datos: Una prueba puede ser ejecutada múltiples veces con diferentes datos de prueba.

Probado rápido: No necesita corredores de prueba especiales, ni estar registrado para poder realizar las pruebas.

Auto contenido: Contiene todo el código que necesita para hacer las pruebas, requiriendo únicamente las funcionalidades básicas de PHP como lenguaje.

En las primeras pruebas realizadas se detectaron errores en algunas de las funcionalidades principales y de mayor uso en la aplicación y se obtuvo el resultado que se muestra en la figura 6.



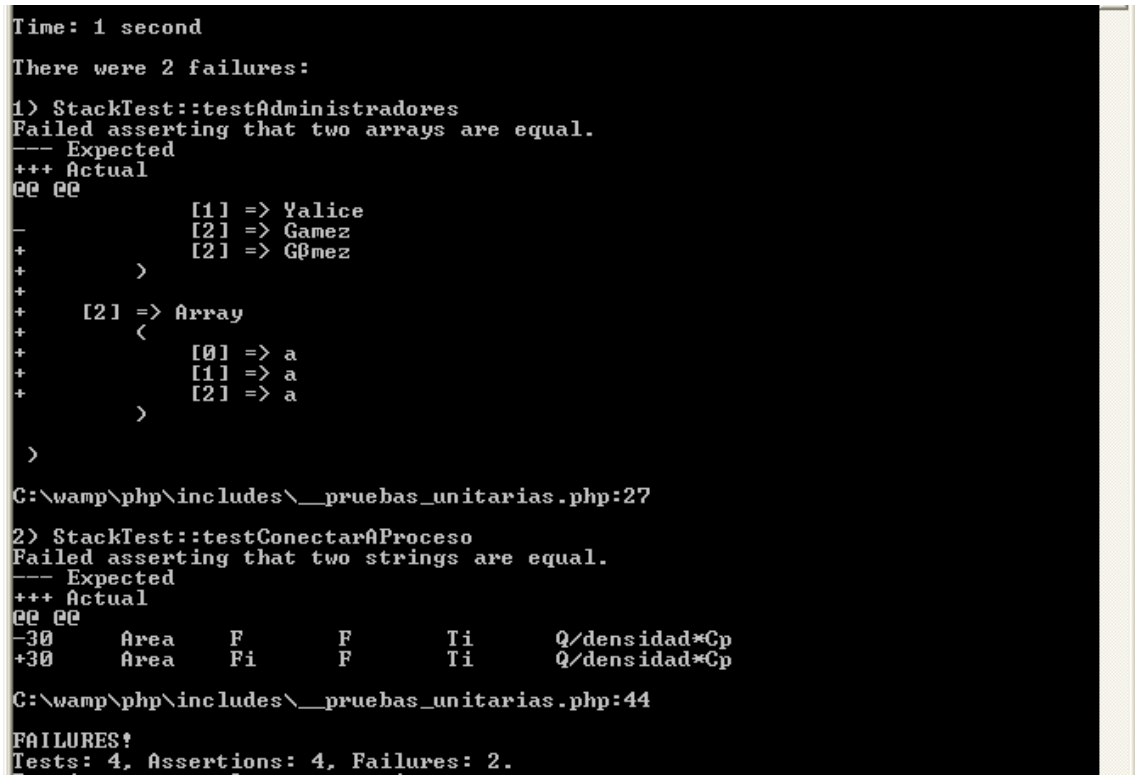


Fig. 6: Pruebas de la primera iteración.

Para dar solución a los errores detectados se realizó el análisis y corrección de las funcionalidades afectadas, hasta lograr un correcto funcionamiento de todos los métodos, como se muestra en la figura 7.

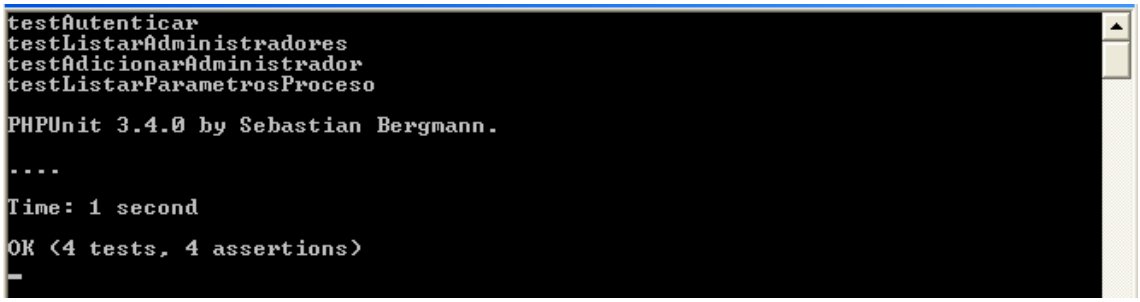


Fig. 7: Pruebas de la segunda iteración.

A continuación se muestra un gráfico con los resultados obtenidos en ambas iteraciones.

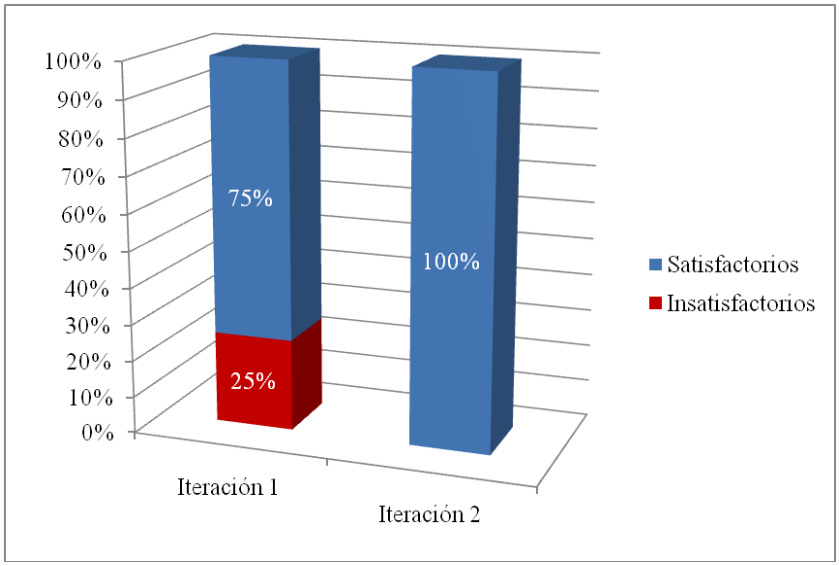


Fig. 8: Resultados de las iteraciones de las pruebas de unidad.

A nivel de integración, se utilizó la estrategia ascendente. Los módulos a integrar fueron: el servidor NVP en el nivel inferior, el adaptador a nivel intermedio y la aplicación web a nivel superior. Primero se probó la integración entre el servidor NVP y el adaptador, utilizando un script controlador que permitiera introducir los datos y mostrar la respuesta recibida. Esto fue necesario debido a que el adaptador tiene como únicas interfaces de comunicación dos sockets. Se detectaron errores que fueron solucionados inmediatamente por el equipo de desarrollo, sustituyéndose posteriormente el script por la aplicación web y repitiéndose nuevamente las pruebas de integración, arrojándose resultados satisfactorios.

Por último, en el nivel de sistema se utilizaron las técnicas de caja negra: partición de equivalencia y análisis de valores límites.

Las pruebas de caja negra son pruebas funcionales. Se parte de los requisitos funcionales, a muy alto nivel, para diseñar pruebas que se aplican sobre el sistema sin necesidad de conocer como está construido por dentro.

Presentan una limitación en cuanto a que es prácticamente imposible reproducir todo el espectro por la innumerable cantidad de combinaciones de entrada posibles, agravada por el desconocimiento de la lógica interna. (Pressman, 2002)

En las siguientes tablas se representan los casos de prueba, los diferentes escenarios y el registro de las dificultades encontradas.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
[SC1: Registrarse]	EC1.1 Usuario no registrado.	En este escenario se prueba que el usuario pueda registrarse en el sistema.	1. El usuario selecciona la opción registrarse. 2. El sistema activa un formulario para que el usuario se registre. 3. El usuario introduce los datos. 4. El sistema verifica los datos. 5. El sistema crea un usuario con los datos introducidos. 6. El sistema muestra un mensaje indicándole al usuario que ha sido añadido a la base de datos.
	EC 1.2: Usuario registrado.	En este escenario se prueba que el sistema muestre un mensaje indicándole al usuario que ya se encuentra en la base de datos	1. El usuario selecciona la opción registrarse 2. El sistema activa un formulario para que el usuario se registre. 3. El usuario introduce los datos. 4. El sistema verifica los datos. 5. El sistema muestra un mensaje indicando que el usuario ya está registrado.
	EC 1.3: Sistema sin conexión.	En este escenario se verifica la respuesta del sistema cuando no hay conexión.	1. El usuario selecciona la opción registrarse 2. El sistema activa un formulario para que el usuario se registre. 3. El usuario introduce los datos. 4. El sistema verifica los datos. 5. El sistema verifica que exista disponibilidad de conexión. 6. El sistema muestra un mensaje indicando que no hay capacidad de conexión.
[SC2:Autenticarse]	EC 2.1: Usuario	En este escenario	1. El usuario selecciona la opción

	registrado.	se prueba que un usuario que se encuentra registrado se puede conectar al sistema.	autenticarse (seleccionada por defecto). 2. El usuario introduce los datos. 3. El sistema verifica los datos. 4. El sistema da acceso al usuario mostrando la página principal.
	EC 2.2: Usuario no registrado	En este escenario se verifica que el sistema le impida el acceso a un usuario que no se encuentre registrado.	1. El usuario selecciona la opción autenticarse (seleccionada por defecto). 2. El usuario introduce los datos. 3. El sistema verifica los datos. 4. El sistema muestra un mensaje de error indicando que debe registrarse para poder acceder.
	EC 2.3: Usuario autenticado.	En este escenario se verifica que el sistema le impida al usuario el acceso si ya está autenticado.	1. El usuario selecciona la opción autenticarse (seleccionada por defecto). 2. El usuario introduce los datos. 3. El sistema verifica los datos. 4. El sistema muestra un mensaje de error.
	EC 2.4: No disponibilidad de autenticación.	En este escenario se verifica que el sistema le impida al usuario el acceso cuando se ha excedido el límite de usuarios autenticados.	1. El usuario selecciona la opción autenticarse (seleccionada por defecto). 2. El usuario introduce los datos. 3. El sistema verifica los datos. 4. El sistema muestra un mensaje de error indicando que no hay disponibilidad de autenticación.
	EC 2.5: Usuario registrado rol usuario.	En este escenario se verifica que un usuario que se encuentra registrado con rol de usuario tenga acceso al módulo de usuario luego de introducir los	1. El usuario selecciona la opción autenticarse (seleccionada por defecto). 2. El usuario introduce los datos. 3. El sistema verifica los datos. 4. El sistema da acceso a las funcionalidades del usuario común.

		datos correctos.	
	EC 2.6: Usuario registrado rol administrador.	En este escenario se verifica que un usuario que se encuentra registrado con rol de administrador tenga acceso al módulo de administrador.	1. El usuario selecciona la opción autenticarse (seleccionada por defecto). 2. El usuario introduce los datos. 3. El sistema verifica los datos. 4. El sistema da acceso a las funcionalidades del administrador.
	EC 2.7: Sistema sin conexión.	En este escenario se verifica la respuesta del sistema cuando no hay conexión.	1. El usuario selecciona la opción autenticarse (seleccionada por defecto). 2. El usuario introduce los datos. 3. El sistema verifica los datos. 4. El sistema verifica que exista disponibilidad de conexión. 5. El sistema muestra un mensaje indicando que no hay capacidad de conexión.
[SC3: Conexión a Proceso]	EC 3.1: Conexión a proceso.	En este escenario se comprueba la respuesta del sistema cuando el usuario se conecta a un proceso.	1. El usuario selecciona el proceso y presiona el botón Simular. 2. El sistema muestra una ventana para que el usuario introduzca los datos del proceso. 3. El sistema verifica los datos. 4. El sistema comienza la simulación.
	EC 3.2: Conexión a proceso no seleccionado.	En este escenario se comprueba la respuesta del sistema cuando el usuario no seleccionó el proceso.	1. El usuario presiona el botón Simular. 2. El sistema verifica que el proceso esté seleccionado. 3. El sistema muestra un mensaje indicando que el proceso no está seleccionado.
[SC4: Desconexión de Proceso]	EC 4.1: Desconexión de proceso.	En este escenario se comprueba la respuesta del sistema cuando se solicita ejecutar la	1. El usuario selecciona el proceso a desconectar. 2. El usuario presiona el botón Desconectar.

		desconexión de un proceso.	3. El sistema desconecta al usuario del proceso. 4. El sistema muestra el mensaje de desconexión satisfactoria.
	EC 4.2: Desconexión de proceso no seleccionado.	En este escenario se comprueba la respuesta del sistema cuando se solicita ejecutar la desconexión de un proceso que no ha sido seleccionado.	1. El usuario presiona el botón Desconectar. 2. El sistema muestra el mensaje de que no hay ningún proceso seleccionado.
	EC 4.3: Desconexión de proceso sin disponibilidad de conexión.	En este escenario se comprueba la respuesta del sistema cuando se solicita ejecutar la desconexión de un proceso cuando no hay conexión.	1. El usuario selecciona el proceso a desconectar. 2. El usuario presiona el botón Desconectar. 3. El proceso no desaparece de la lista de procesos conectados.
[SC5: Selección del tipo de reporte]	EC 5.1: Selección del tipo de reporte.	En este escenario se comprueba la respuesta del sistema cuando se solicita visualizar un reporte en específico.	1. El usuario selecciona el tipo de reporte deseado. 2. El usuario presiona el botón Aceptar. 3. El sistema visualiza el reporte solicitado.
	EC 5.2: Selección del tipo de reporte sin disponibilidad de conexión.	En este escenario se comprueba la respuesta del sistema cuando se solicita visualizar un reporte en específico y no hay conexión.	1. El usuario selecciona el tipo de reporte deseado. 2. El usuario presiona el botón Aceptar. 3. El sistema no visualiza nada.
[SC6: Administrar usuario]	EC 6.1: Adicionar usuario.	En este escenario se comprueba la respuesta del sistema cuando se solicita adicionar	1. El usuario solicita adicionar un usuario. 2. El usuario introduce los datos. 3. El sistema valida los datos. 4. El sistema crea un nuevo

		un usuario.	usuario con los datos especificados. 5. El sistema muestra un mensaje informando que la operación se realizó correctamente.
	EC 6.2: Adicionar usuario con los datos incorrectos.	En este escenario se comprueba la respuesta del sistema cuando se solicita adicionar un usuario con los datos incorrectos.	1. El usuario solicita adicionar un usuario. 2. El usuario introduce los datos. 3. El sistema valida los datos. 4. El sistema muestra un mensaje informando que los datos no están correctos.
	EC 6.3: Adicionar usuario existente.	En este escenario se comprueba la respuesta del sistema cuando se solicita adicionar un usuario existente.	1. El usuario solicita adicionar un usuario. 2. El usuario introduce los datos. 3. El sistema valida los datos. 4. El sistema muestra un mensaje informando que la operación se realizó correctamente pero no inserta el usuario en la base de datos.
	EC 6.4: Modificar usuario.	En este escenario se comprueba la respuesta del sistema cuando se solicita modificar un usuario.	1. El usuario solicita modificar un usuario. 2. El usuario introduce los datos. 3. El sistema valida los datos. 4. El sistema muestra un mensaje informando que la operación se realizó correctamente.
	EC 6.5: Modificar usuario con los datos incorrectos.	En este escenario se comprueba la respuesta del sistema cuando se solicita modificar un usuario introduciendo datos incorrectos.	1. El usuario solicita modificar un usuario. 2. El usuario introduce los datos. 3. El sistema valida los datos. 4. El sistema muestra un mensaje informando que los datos no están correctos.
	EC 6.6: Modificar usuario no seleccionado.	En este escenario se comprueba la respuesta del	1. El usuario solicita modificar un usuario sin seleccionarlo. 2. El sistema muestra un mensaje

		sistema cuando se solicita modificar un usuario sin haberlo seleccionado previamente.	informando que no ha sido seleccionado ningún usuario.
	EC 6.7: Eliminar usuario.	En este escenario se comprueba la respuesta del sistema cuando se solicita eliminar un usuario.	1. El usuario solicita eliminar un usuario. 2. El usuario presiona el botón Eliminar. 3. El sistema muestra un mensaje informando que la operación se realizó satisfactoriamente.
	EC 6.8: Eliminar usuario no seleccionado.	En este escenario se comprueba la respuesta del sistema cuando se solicita eliminar un usuario sin seleccionar ninguno.	1. El usuario presiona el botón Eliminar para eliminar un usuario sin seleccionarlo. 2. El sistema muestra un mensaje informando que no ha sido seleccionado ningún usuario.
[SC7: Administrar límites]	EC 7.1: Establecer límites.	En este escenario se comprueba la respuesta del sistema cuando se solicita establecer los límites del mismo.	1. El usuario introduce los datos de los límites. 2. El usuario presiona el botón Aceptar. 3. El sistema valida los datos. 4. El sistema muestra un mensaje informando que la operación se realizó correctamente.
	EC 7.2: Establecer límites con los datos incorrectos.	En este escenario se comprueba la respuesta del sistema cuando se solicita establecer los límites del mismo cuando se especifican datos	1. El usuario introduce los datos de los límites. 2. El usuario presiona el botón Aceptar. 3. El sistema valida los datos. 4. El sistema muestra un mensaje informando que los datos especificados no están correctos.

		incorrectos.	

SC1: Registrarse

Id del escenario	Escenario	Variable Nombre	Variable Apellido	Variable Usuario	Variable Contraseña	Variable Repetir Contraseña	Respuesta del sistema	Resultado de la prueba
EC1.1	Usuario no registrado.	F	V	V	V	V	El sistema crea un usuario con los datos introducidos.	Se obtuvo el resultado esperado
		V	F	V	V	V		
		V	V	F	V	V		
		V	V	V	F	V		
		V	V	V	V	F		
EC1.2	Usuario registrado.	F	V	V	V	V	El sistema muestra un mensaje.	El resultado fue satisfactorio.
		V	F	V	V	V		
		V	V	F	V	V		
		V	V	V	F	V		
		V	V	V	V	F		
EC1.3	Sistema sin conexión.						El sistema muestra un mensaje.	Se obtuvo el resultado esperado.

SC2:Autenticarse

Id del escenario	Escenario	Variable Nombre	Variable Contraseña	Respuesta del sistema	Resultado de la prueba
EC 2.1	Usuario registrado.	F	V	El sistema da acceso al usuario.	Se obtuvo el resultado esperado.
		V	F		
EC 2.2	Usuario no registrado	F	V	El sistema muestra un mensaje de error.	Resultado satisfactorio.
		V	F		
EC 2.3	Usuario autenticado.	F	V	El sistema muestra un mensaje de error.	Se detectó una no conformidad
		V	F		
EC 2.4	No disponibilidad de autenticación.	F	V	El sistema muestra un mensaje de error.	Resultado esperado.
		V	F		
EC 2.5	Usuario registrado rol usuario.	F	V	El sistema da acceso a las funcionalidades del usuario común.	Resultado esperado.
		V	F		

EC 2.6	Usuario registrado rol administrador.	F	V	El sistema da acceso a las funcionalidades del administrador.	Resultado satisfactorio.
		V	F		
EC 2.7	Sistema sin conexión.			El sistema muestra un mensaje de error.	Resultado esperado.

SC3: Conexión a Proceso

Id del escenario	Escenario	Variable 1	Variable 2	Variable N	Respuesta del sistema	Resultado de la prueba
EC 3.1	Conexión a proceso.	F	V	V	El sistema comienza la simulación.	Se obtuvo el resultado esperado.
		V	F	V		
		V	V	F		
EC 3.2	Conexión a proceso no seleccionado.	F	V	V	El sistema muestra un mensaje de error.	Resultado satisfactorio.
		V	F	V		
		V	V	F		

SC4: Desconexión de Proceso

Id del escenario	Escenario	Respuesta del sistema	Resultado de la prueba
EC 4.1	Desconexión de proceso.	El sistema muestra un mensaje indicando que el proceso fue desconectado.	Se obtuvo el resultado esperado.
EC 4.2	Desconexión de proceso no seleccionado.	El sistema muestra un mensaje de error.	Se obtuvo el resultado adecuado.
EC 4.3	Desconexión de proceso sin disponibilidad de conexión.	El proceso no es eliminado de la lista de procesos conectados.	Se obtuvo el resultado esperado.

SC5: Selección del tipo de reporte

Id del escenario	Escenario	Respuesta del sistema	Resultado de la prueba
EC 5.1	Selección del tipo de reporte.	El sistema muestra el reporte solicitado en la ventana de los reportes.	Resultado satisfactorio.
EC 5.2	Selección del tipo de reporte sin disponibilidad de conexión.	La ventana de reportes no se actualiza.	Resultado esperado.

SC6: Administrar usuario

Id del escenario	Escenario	Variable Usuario	Variable Nombre	Variable Apellidos	Variable Contraseña	Respuesta del sistema	Resultado de la prueba
EC 6.1	Adicionar usuario.	F	V	V	V	El sistema adiciona el usuario y muestra un mensaje.	Se obtuvo el resultado esperado.
		V	F	V	V		
		V	V	F	V		
		V	V	V	F		
EC 6.2	Adicionar usuario con los datos incorrectos.	F	V	V	V	El sistema muestra un mensaje.	Resultado satisfactorio.
		V	F	V	V		
		V	V	F	V		
		V	V	V	F		
EC 6.3	Adicionar usuario existente.	F	V	V	V	El sistema muestra un mensaje.	Se obtuvo una no conformidad.
		V	F	V	V		
		V	V	F	V		
		V	V	V	F		
EC 6.4	Modificar usuario.	F	V	V	V	El sistema modifica al usuario y muestra un mensaje.	Se obtuvo el resultado adecuado.
		V	F	V	V		
		V	V	F	V		
		V	V	V	F		
EC 6.5	Modificar usuario con los datos incorrectos.	F	V	V	V	El sistema muestra un mensaje.	Resultado satisfactorio.
		V	F	V	V		
		V	V	F	V		
		V	V	V	F		
EC 6.6	Modificar usuario no seleccionado.					El sistema muestra un mensaje.	Resultado esperado.
EC 6.7	Eliminar usuario.					El sistema elimina al usuario y muestra un mensaje.	Se obtuvo el resultado esperado.
EC 6.8	Eliminar usuario no seleccionado.					El sistema muestra un mensaje.	Resultado satisfactorio.

SC7: Administrar límites

Id del escenario	Escenario	Variable Usuarios conectados al sistema	Variable Procesos activos	Variable Proceso	Variable Usuarios conectados a un proceso	Respuesta del sistema	Resultado de la prueba
EC 7.1	Establecer límites.	F	V	V	V	El sistema registra los límites y muestra un mensaje.	Resultado satisfactorio.
		V	F	V	V		
		V	V	F	V		
		V	V	V	F		
EC 7.2	Establecer límites con los datos incorrectos.	F	V	V	V	El sistema muestra un mensaje.	Resultado esperado.
		V	F	V	V		
		V	V	F	V		
		V	V	V	F		

Registro de defectos y dificultades encontrados

Elemento	No.	No conformidad	Aspecto correspondiente	Etapa de detección	Significativa	No significativa	Recomendación	Estado NC	Respuesta del equipo de desarrollo
Autenticarse.	1	El mensaje no está acorde al error.	Autenticarse estando autenticado.	Pruebas de validación.		X	Escribir un texto más explícito y acorde al error.	[PD] 14/4/10	Se mostraron receptivos.
Adicionar usuario.	2	No se muestra ningún mensaje.	Adicionar usuario existente.	Pruebas de validación.	X		Adicionar la alerta correspondiente.	[PD] 14/4/10	Se mostraron receptivos.

Una vez realizadas las pruebas de caja negra sobre la aplicación, los desarrolladores corrigieron los errores encontrados y se aplicaron las pruebas de caja negra en una segunda iteración para garantizar que dichos errores fueran solucionados. La misma arrojó que se les dio solución a los defectos encontrados en la primera iteración.

A continuación, aparece un gráfico con los resultados obtenidos.

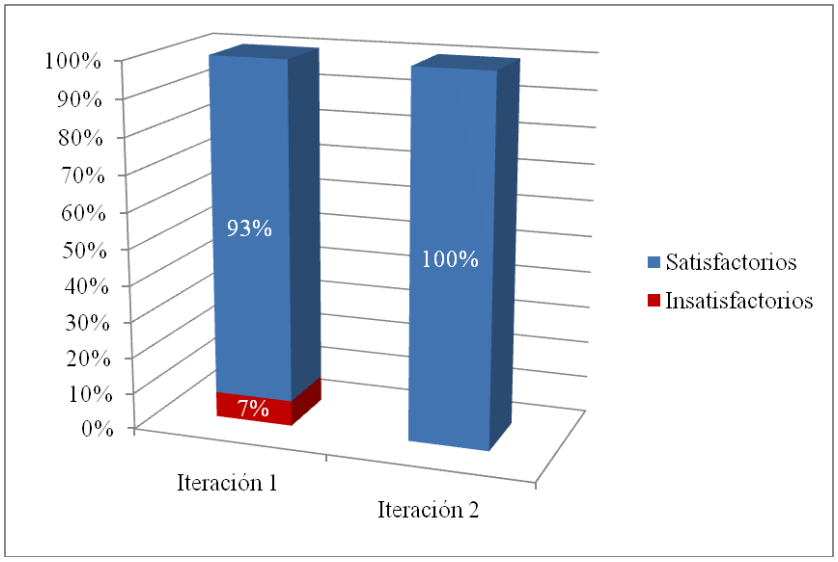


Fig. 9: Resultados de las iteraciones de las pruebas de caja negra.

Conclusiones parciales

En el presente capítulo se destacó la importancia que tienen las distintas pruebas de software aplicadas para validar el correcto funcionamiento de la aplicación, permitiendo la detección de errores, principalmente de la etapa de implementación. Para ello se usaron las técnicas de prueba caja negra y caja blanca, y se desarrollaron en los niveles de desarrollador, unidad, integración y sistemas. Para realizar las pruebas de unidad se utilizó el módulo de pruebas PHPUnit, el cual posibilitó agilizar el proceso de las pruebas; y en las de integración se escribió un script controlador para revisar los datos enviados por el adaptador. Por otra parte, las pruebas de caja negra permitieron detectar otras no conformidades teniendo en cuenta las descripciones de los casos de prueba, a las cuales se les dio solución por parte del equipo de desarrollo. Por último, es necesario destacar que la aplicación fue revisada por el Grupo de Calidad del Centro CEGEL de la facultad 15, siendo liberada con resultados satisfactorios, como aparece en el Anexo 1.

CONCLUSIONES GENERALES

En el presente trabajo se han estudiado las no conformidades encontradas en la versión anterior del Nodo Virtual de Procesos (NVP), siendo las principales: alto consumo de recursos en las computadoras clientes, problemas con la conexión a la base de datos, insuficientes modelos de procesos implementados y la ausencia de funcionalidades necesarias como la generación de reportes y alertas. Se realizaron investigaciones de la bibliografía relacionada con el tema para seleccionar las mejores soluciones que a nivel mundial se le dan a problemáticas similares.

Atendiendo a los requisitos del cliente, se desarrolló una aplicación web con PHP como lenguaje de programación del lado del servidor y ExtJs como framework del lado del cliente, NetBeans como entorno de desarrollo y Apache como servidor web; ya que de esta forma se podría reducir de manera significativa el consumo de recursos en las computadoras clientes. Se incluyó en el servidor NVP el estándar ODBC para la comunicación con la base de datos y se mantuvo el uso de sockets en la comunicación entre el cliente y el servidor NVP.

Por otra parte, se utilizó la tecnología Ajax para disminuir el tráfico de la red y evitar sobrecargas en el servidor y se construyó un adaptador para posibilitar la comunicación entre los sockets implementados en C++ y PHP.

Por último, se validó la implementación realizada a nivel de unidad con las técnicas de pruebas de caja negra y caja blanca, que arrojaron resultados satisfactorios.

De forma general, se obtuvo una herramienta de gran utilidad para los estudiantes de la especialidad Ingeniería Automática, con la cual se puede reforzar el proceso docente de los mismos.

RECOMENDACIONES

Durante el desarrollo de la nueva versión del NVP, el equipo de desarrollo encontró nuevas no conformidades cuya solución se encuentra fuera del alcance del presente trabajo, por lo que se enumeran a continuación como recomendaciones para las versiones futuras del software:

- La incorporación de nuevos protocolos industriales que permitan comunicar a la aplicación con equipos físicos utilizados por la especialidad Ingeniería Automática.
- La incorporación de controladores virtuales que simulen el funcionamiento de los controladores físicos utilizados en los procesos industriales.
- La solución de las incompatibilidades presentadas con el navegador Internet Explorer.

BIBLIOGRAFÍA

Aburto, W. 2001. *Programación de aplicaciones para internet con active server pages.* Lima. Perú : s.n., 2001.

Alicante, Repositorio Institucional de la Universidad de. Repositorio Institucional de la Universidad de Alicante . [Online] http://rua.ua.es/dspace/bitstream/10045/4412/5/03c-AplicacionesWeb.pdf.

Allende, J. 2005. *Java 2.* 2005.

Apache, Sitio Oficial de. Sitio Oficial de Apache. [Online] http://www.apache.org/.

Aragon, Universidad de. Universidad de Aragon. [Online] http://informatica.uv.es/iiguia/2000/BD2/2_0_BD2Tema2_06.pdf .

Aula-Tec. Aula-Tec. [Online] http://www.aula-tec.es/CURSOS/PHP/CLASE18.pdf.

Bluefish, Sitio Oficial de. Sitio Oficial de Bluefish. [Online] http://bluefish.openoffice.nl/index.html.

Bonaparte, U y otros. 2008. Universidad Tecnológica, Facultad Regional Tucumán. [Online] Universidad Tecnológica, Facultad Regional Tucumán, 12 2008. http://www.frt.utn.edu.ar/sistemas/paradigmas/poo.htm.

Charte Ojeda, Francisco. 2002. *Programación con Visual C#.NET.* Madrid : s.n., 2002.

Cleger Despaigne, Eliober y Tornés Montes de Oca, Annarella María. 2009. *Análisis y Diseño de una herramienta interactiva de simulación de procesos: Nodo Virtual de Procesos. Segunda Iteración.* Universidad de las Ciencias Informáticas. Ciudad de La Habana : s.n., 2009.

Corporation, Microsoft. Microsoft Corporation. [Online] http://www.microsoft.com/en/us/default.aspx.

Dormido, S. 2007. *Proyecto AutomatL@bs. Red de laboratorios remotos en Automática.* E.T.S. Ingeniería Informática, UNED. Madrid, España : s.n., 2007.

Eclipse, Sitio Oficial de. Sitio Oficial de Eclipse. [Online] http://www.eclipse.org/.

Edith, Maylén y Ortiz, Leidi A. 2008. *Análisis y Diseño de un Nodo Virtual de Procesos.* La Habana : s.n., 2008. p. 169.

Escobar Pompa, Mailén Edith y Ortiz Azahares, Leydis Andis. 2008. *Análisis y Diseño de un Nodo Virtual de Procesos.* Universidad de las Ciencias Informáticas. Ciudad de La Habana : s.n., 2008.

ExtJs. ExtJs. [Online] http://extjs.es/.

Gallego Vázquez, José Antonio. 2003. *Desarrollo web con PHP y MySQL.* Madrid : Anaya Multimedia, 2003.

Gámez Batista, Yalice, Moreno Vega, Valery y Martínez Márquez, Yoan. 2009. *Nodo Virtual de Procesos.* Universidad de las Ciencias Informáticas. Ciudad de La Habana : s.n., 2009.

Gámez, Yalice. 2008. *Herramienta interactiva para la enseñanza en la carrera de Ingeniería Automática: Nodo Virtual de Procesos.* Instituto Superior Politécnico "José Antonio Echeverría". Ciudad de La Habana : s.n., 2008.

García A., R. A. y L., E. 2009. *Programación y Desarrollo de un Nodo Virtual de Procesos.* Instituto Politécnico Jose Antonio Echeverría. Ciudad de La Habana : s.n., 2009.

Gedit, Sitio Oficial de. Sitio Oficial de Gedit. [Online] http://www.gedit.org.

Gómez López, Rubén. 2010. *PHP Vs. Java.* Universidad Autónoma de Madrid : s.n., 2010.

Gonce Fernández, Susana. 2008. *Arquitectura de un Nodo Virtual de Procesos.* Universidad de las Ciencias Informáticas. Ciudad de La Habana : s.n., 2008.

Graham, Paul. paulgraham.com. [Online] http://www.paulgraham.com/road.html.

Hernández Garrigó, Daniel y Moreno Martínez, Rolando. 2009. *Solución Informática Nodo Virtual de Procesos para la carrera Ingeniería Automática.* Universidad de las Ciencias Informáticas. Ciudad de La Habana : s.n., 2009.

Hosseinzaman, A. y Bargiela, A. 1995. *ADA's Virtual Node based Water System Simulator.* Department of Computing Nottingham Trent University Burton Street. Nottingham : s.n., 1995.

Jacobson, I, Booch, G y Rumbaugh, J. 2004. *El Proceso Unificado de Desarrollo de Software.* La Habana : Félix Varela, 2004. p. 435.

JavaSE. Java SE Technologies. [Online] http://java.sun.com.

jQuery. jQuery. [Online] http://jqueryui.com/.

Maier, S. y Herrscher, D. 2005. *On Node Virtualization for Scalable Network Emulation.* University of Stuttgart, Institute of Parallel and Distributed Systems (IPVS). Alemania : s.n., 2005.

Martínez D., H. G. y M., R. 2009. *Solución Informática Nodo Virtual de Procesos.* Universidad de las Ciencias Informáticas. Ciudad de La Habana : s.n., 2009.

Microsoft, Corporation. 2010. ODBC: Introducción a la conexión abierta a bases de datos. [Online] 2010. [Citado: enero 27, 2010.] http://support.microsoft.com/kb/110093.

Miyachi, T. y Chinen, K. 2006. *StarBED and SprinOS: Large scale general purpose network testbed and supporting software.* Japan Advanced Institute of Science an Technology. Ishikawa, Japón : s.n., 2006.

Molino, N y Bao, Z. 2004. *A Virtual Node Algorithm for Changing Mesh Topology During Simulation.* Stanford University. 2004.

NetBeans, Sitio Oficial de. Sitio Oficial de NetBeans. [Online] http://www.netbeans.org.

Nguyen, A.V, Gillet, D. y Sire, S. 2004. *Sustaining Collaboration within a Learning Community in Flexible Engineering Education.* School of Engineering, Swiss Federal Institute of Technology (EPFL). Switzerland : s.n., 2004.

Nguyen, A.V. 2004. *Sustaining the continuity of interaction in Web-based experimentation for Engineering Education.* Grenoble, Francia : s.n., 2004.

OpenLink, Software. 2010. OLEDB Data Providers. [Online] 2010. [Citado: enero 27, 2010.] http://uda.openlinksw.com/oledb.

Pressman, Roger S. 2002. *Ingeniería de Software, un enfoque práctico.* 2002.

Quanta, Sitio Oficial de. Sitio Oficial de Quanta. [Online] http://quanta.kdewebdev.org.

Ripley, B. 2009. ODBC connectivity. [Online] 2009. [Citado: enero 27, 2010.] http://cran.r-project.org/web/packages/RODBC/vignettes/RODBC.pdf.

Rivelo Ayala, Alberto y López García, Eddiel. 2009. *Programación y Desarrollo de un Nodo Virtual de Procesos.* Instituto Superior Politécnico José Antonio Hecheverría. Ciudad de La Habana : s.n., 2009.

Rodríguez, P. 2008. *Paradigmas de Programación.* Universidad Nacional Autónoma de México. Ciudad de México : s.n., 2008.

RTAI-XML, equipo de trabajo. 2004. RTAI-XML. [Online] Appalachian State University, 2004. http://rtaixml.dsi.unifi.it/.

Schultz, R. 2000. Using the Oracle ODBC drivers with third party products. [Online] 2000. [Citado: enero 27, 2010.] http://www.oracle.com/technology/software/tech/windows/odbc/htdocs/ODBCFAQ.pdf.

Solutions, Solcre Techonology. 2009. Solcre Techonology Solutions. [Online] 2009. http://www.solcre.com/files/ventajas_de_las_aplicaciones_web.pdf.

Souto, David Cabrero. Universidad de Coruña. [Online] http://quegrande.org/apuntes/EI/OPT/IU/teoria/08-09/introduccion_a_las_aplicaciones_web.pdf.

Studio, Sitio Oficial de Zend. Sitio Oficial de Zend Studio. [Online] http://www.zend.com/en/products/studio/.

Sutter y Alexandrescu. 2004. *C++ Coding Standards.* 2004.

Tiobe. Tiobe. [Online] www.tiobe.com.

Tomcat, Sitio Oficial de. Sitio Oficial de Tomcat. [Online] http://tomcat.apache.org/.

Vanderkam, Dan. dygraphs JavaScript Visualization Library. [Online] http://www.danvk.org/dygraphs/.

Vaquero, J. 2007. *Herramienta interactiva para la enseñanza y entrenamiento en la técnica de control predictivo.* Instituto Politécnico Jose Antonio Echeverría. La Habana, Cuba. : s.n., 2007.

Vargas, H. 2008. *A systematic two-layer approach to develop web based experimentation environments for control engineering education.* Dept. of Computer Science and Automatic Control UNED. Madrid, España : s.n., 2008.

Wesley, A. 2001. *The Annotated C++ Reference Manual.* 2001.

YUI. Yahoo User Interface. [Online] http://developer.yahoo.com/yui/.

ANEXOS

Anexo 1: Acta de liberación de calidad del Nodo Virtual de Procesos, emitida por el Grupo de Calidad del Centro CEGEL.

UCI
Universidad de las Ciencias Informáticas

Acta de Liberación de Artefactos, Grupo de Calidad Centro CEGEL de la Facultad 15 de la Universidad de las Ciencias Informáticas.

Viernes, 14 de mayo de 2010.

Luego de haber efectuado 2 iteraciones de revisiones al sistema informático Nodo Virtual de Procesos v 2.0 y haberse detectado un promedio de 13 No Conformidades, se puede afirmar que se han corregido los defectos encontrados, por lo que queda liberada la aplicación.

Firma del Asesor y Jefe del Grupo de Calidad Centro CEGEL

Ing. Raúl Velázquez Alvarez

ecanato Facul

GLOSARIO

Bytecode: Es el código independiente del hardware generado por el compilador Java y ejecutado por su intérprete.

DLL (dynamic-link library)**:** Biblioteca de vínculos dinámicos. Es un archivo ejecutable que actúa como una biblioteca compartida de funciones.

Hilo: Es un subprograma que se invoca a través de la ejecución de un programa principal y a partir de allí se considera independiente.

Procesos industriales: Es un conjunto de procesos físicos y químicos interrelacionados, implementados por medios físicos. Todo sistema de procesos tiene entradas y salidas. Entradas pueden ser materia prima, temperatura, concentración, presión, etc. Un sistema está sujeto usualmente a señales o perturbaciones que para compensarlas se hace uso de correcciones o acciones de control.

Signal: Aviso que un objeto puede emitir cuando le ocurre algo (un cambio de estado importante, también denominado evento).

Slot: Método de un objeto que puede ser llamado cuando se genera una señal particular.

Sockets: Es el punto final de un flujo de comunicación bidireccional a través de una red de computadoras basada en protocolo IP.