Universidad de las Ciencias Informáticas

UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 1

Automatización del Funcionamiento de la Federación Estudiantil Universitaria

Trabajo de Diploma presentado para optar por el título de Ingeniero en Ciencias Informáticas.

Autores:

Yeimys William Seife Pérez

Antonio Marrero Palomino

Tutor:

Lic. Yoemny González Almaguer

"Año 50 de la Revolución"

Ciudad de la Habana, Cuba

Junio de 2008

Dedicatoria

A nuestros padres

Agradecimientos

A nuestro tutor Yoemny por ayudarnos y sabernos formar como profesionales.

A la nuestra cotutora, La yahimita, por pasar las mismas malas noches que nosotros…

A nuestros compañeros de proyecto, la cesaria y la pedrá, por las horas de desvelo.

A los socios de campaña.

Yeimys:

Agradezco a todos los que de una forma u otra han aportado a mi formación en estos ya casi 6 años de estudio. Quisiera poder mencionar a todos los que debo gratitud por ayudarme y soportarme.

A mi mamá, por ser mi madre y mi padre y quererme siempre.

A Leidys, mi novia, por quererme tanto y a mi suegra que ya me quiere casi igual…

A la botija, El Antony por suministrarme tanta comida…

Antonio:

A mis padres por educarme, por brindarme su apoyo en todo momento, por hacerme el hombre que soy.

A mi queridísima novia, Yahima que ha sido tan buena y comprensiva conmigo, por apoyarme y ayudarme incondicionalmente en todo momento.

A mis suegros por acogerme como un hijo.

A todos aquellos que de una forma u otra me ayudaron a lo largo de la carrera.

Y a la mielda de Yeimys por ser tan buen compañero y comprar tanto cigarro…

Declaración de Autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas para que haga el uso que estime pertinente del mismo.

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

_____ _____

Firma del Autor
Antonio Marrero Palomino

Firma del Autor
Yeimys William Seife Pérez

_____

Firma del Tutor
Lic. Yoemny González Almaguer

Opinión del Tutor del Trabajo de Diploma

Título: Automatización del funcionamiento de la Federación Estudiantil Universitaria.

Autores: Yeimys William Seife Pérez y Antonio Marrero Palomino.

El tutor del presente Trabajo de Diploma considera que durante su ejecución el estudiante mostró las cualidades que a continuación se detallan.

El estudiante mostró independencia en la realización del trabajo de diploma, logrando gran originalidad y creatividad en las soluciones brindadas, en la investigación fue consultada una gran variedad de fuentes bibliográficas bastante actualizada y con muy buena calidad. El desarrollo del trabajo se corresponde perfectamente con los objetivos planteados, los cuales fueron cumplidos satisfactoriamente. El trabajo realizado es de gran importancia para el desarrollo del portal de la FEU y el cumplimiento de todas las tareas propuestas garantiza la eficiencia y calidad del proceso productivo.

Por todo lo anteriormente expresado considero que el estudiante está apto para ejercer como Ingeniero Informático; y propongo que se le otorgue al Trabajo de Diploma la calificación de 5 puntos.

_____

Firma del Tutor

Lic. Yoemny González Almaguer

Resumen

Desde sus inicios la Federación Estudiantil Universitaria (FEU), ha estado en constante cambio a la par del proceso revolucionario cubano. Su labor es representar los intereses del estudiantado universitario, así como contribuir a la formación integral de los graduados.

Actualmente la FEU tiene dificultades para llevar el control de las informaciones y estadísticas de sus integrantes y de la propia organización; así como del óptimo manejo de datos referentes al funcionamiento y dirección de la misma. Los trabajos se realizan utilizando métodos obsoletos, como lápiz y papel o con herramientas que no son las más idóneas para la gestión de grandes volúmenes de información, como el Excel. El acta de las reuniones efectuadas en cada brigada y las caracterizaciones de sus miembros se archivan, acumulando una considerable cantidad de papeles. Esta situación provoca que las cifras que se gestionan sean inexactas, teniendo un margen de error elevado; además que se dificulte buscar información referente a una brigada o de un miembro de la organización. Por este motivo se le asigna la tarea al proyecto “KAINOS” de desarrollar una aplicación enfocada a dar una solución informática a los problemas de gestión y control de las estadísticas y el funcionamiento de la Federación en todo el país. En su conjunto la aplicación está compuesta por varios módulos, entre los que se encuentran los módulos de Funcionamiento, Estadística y Seguridad.

El objetivo general de este trabajo es desarrollar los módulos de Funcionamiento, Estadística y parte del módulo de Seguridad como parte del sistema informático para la automatización de la gestión de la información en la FEU cubana. Los módulos permiten gestionar las estructuras de la organización, así como las caracterizaciones de sus miembros e información referente a las reuniones.

En el presente documento se exponen las principales características de las herramientas, tecnologías y metodología empleadas, se describen las funcionalidades de los módulos implicados en la solución del sistema. Finalmente se realizan las pruebas unitarias de los módulos, y se dejan algunas recomendaciones para próximas versiones del mismo.

Índice

Índice de Figuras

Índice de Tablas

Introducción

Desde sus inicios la Federación Estudiantil Universitaria (FEU) ha estado en constante cambio a la par del proceso revolucionario cubano. La labor de esta organización es representar los intereses del estudiantado universitario, así como contribuir a la formación integral de los graduados. Desde el impactante Primer Congreso Nacional Revolucionario de Estudiantes, efectuado en octubre de 1923, ha transitado por un sinnúmero de hechos históricos, eventos relevantes, líderes y circunstancias sociopolíticas que han ido moldeando y transformando su papel como organización estudiantil. En los últimos tiempos, al calor de la batalla de ideas, han surgido nuevos y trascendentales retos que han marcado pautas en sus formas de trabajo. La universalización de la enseñanza, hace ya cinco años, creó un nuevo escenario en el que los estudiantes universitarios ya no se concentran solamente en los Centros de Educación Superior. Dadas las nuevas condiciones, la descentralización del accionar de la organización se convirtió en el mayor desafío para la FEU.

El contenido de trabajo que genera la organización está en correspondencia con la extensión y complejidad de su estructura. Por este motivo, la FEU tiene dificultades para la gestión y control de la información y las estadísticas generadas en sus niveles, así como el óptimo manejo de datos referentes al funcionamiento y dirección de la misma. Cualquier planificación, censo, o auditoría, se realiza utilizando herramientas poco óptimas en la gestión de grandes volúmenes de información, como documentos *Excel* y en muchos casos métodos obsoletos, utilizando lápiz y papel. El acta de las reuniones efectuadas en cada brigada y las caracterizaciones de sus miembros se archivan, acumulando una considerable cantidad de documentos. Esto provoca que las cifras que se gestionan, referentes a los reportes estadísticos de las diferentes estructuras de la organización, sean inexactas, propiciando que se produzcan errores. En estas condiciones, obtener datos específicos referentes a las estructuras más simples, como una brigada o un miembro de la organización, se convierte en una búsqueda larga y compleja. Como consecuencia del trabajo con valores aproximados y las dificultades para el manejo eficiente de la información generada, se presentan retrasos en las tareas, incumpliendo en muchas ocasiones con la entrega de las mismas, provocando además discrepancias en la toma de decisiones que son de vital importancia para la organización y sus miembros.

Con la necesidad creciente de mejorar su funcionamiento y el desarrollo de sus actividades, en el marco del VII Congreso de la FEU, la organización decide aprovechar los beneficios de las nuevas Tecnologías

de la Información y las Telecomunicaciones. A partir de esta decisión y en colaboración con la Universidad de la Ciencias Informáticas se le asigna la tarea al proyecto “KAINOS” de desarrollar una aplicación enfocada a dar una solución informática a los problemas de gestión y control de las estadísticas y el funcionamiento de la Federación en todo el país.

Partiendo de los requerimientos iniciales planteados al proyecto se lleva a cabo un estudio orientado a dos objetivos fundamentales: conocer la estructura y funcionamiento de la FEU y puntualizar las condiciones tecnológicas con que cuenta para soportar una solución que responda a sus necesidades. Analizando los resultados obtenidos se decide conformar tres módulos de trabajo (Funcionamiento, Estadística y Seguridad) para ser desarrollados en una solución *web*[1].

Los módulos de Funcionamiento y Estadística responden a la forma de trabajo de la organización y sus resultados, mientras que el módulo de Seguridad constituye la parte administrativa de este sistema.

Teniendo en cuenta las situaciones antes expuestas se plantea el problema científico: ¿Cómo optimizar en la FEU de Cuba la gestión de su funcionamiento y sus estadísticas? Tomando como objeto de investigación el proceso de gestión de la información de la FEU y como campo de acción los módulos de Funcionamiento, Estadística y Seguridad del sistema informático para la gestión de la información de la FEU cubana.

Para darle solución al problema planteado se definió como objetivo general desarrollar los módulos de Funcionamiento, Estadística y parte del módulo de Seguridad como parte del sistema informático para la automatización de la gestión de la información en la FEU cubana. En orden de cumplir con este objetivo se proponen las siguientes tareas de la investigación:

- Realizar un estudio de las herramientas de desarrollo y el lenguaje de programación que se utilizarán en el desarrollo de la solución.
- Describir las funcionalidades de los módulos Funcionamiento, Estadística y parte del módulo Seguridad.
- Analizar el posible uso de patrones de diseño.
- Implementar la gestión de caracterizaciones estadísticas, reuniones, estructuras de la FEU nacional.
- Realizar pruebas unitarias a la solución.

El presente documento consta de tres capítulos:

Capítulo 1 “Fundamentación Teórica”: expone las principales características de las herramientas, tecnologías y la metodología empleadas en el desarrollo del sistema.

Capítulo 2 “Descripción y análisis de la solución propuesta”: se describen las funcionalidades de los módulos implicados en la solución del sistema. Además se exponen los estándares de codificación usados en la implementación.

Capítulo 3 “Validación de la solución propuesta”: se realiza el diseño, ejecución y evaluación de las pruebas unitarias para verificar el correcto funcionamiento de la aplicación.

Fundamentación Teórica

1.1 Introducción

En el presente capítulo se estudian y definen algunos conceptos necesarios para comprender el dominio del problema. Se abordan aspectos de las diferentes categorías de software, especificando los sistemas de gestión, así como elementos que se deben tener en cuenta para su funcionamiento.

Se justifica la selección del lenguaje de modelado gráfico a utilizar y de la metodología de desarrollo escogida para guiar el desarrollo del software. Además se explican las características de las herramientas que han sido utilizadas en el marco del desarrollo del software y del lenguaje de programación para llevar a cabo la implementación del sistema.

1.2 Software

Uno de las definiciones más formales de software es la dada por *IEEE*[2] en el estándar 729-1983:

“Es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación.” (1)

El software es un conjunto de programas de cómputo y procedimientos necesarios para hacer posible la realización de una tarea específica, en contraposición a los componentes físicos del sistema, es decir, es el soporte lógico a todos los componentes intangibles de un ordenador o computadora. Está formado por una serie de instrucciones y datos, que permiten aprovechar todos los recursos que el ordenador tiene, de manera que pueda resolver gran cantidad de problemas. Una computadora en sí, es sólo un conglomerado de componentes electrónicos; el software le da vida al computador, haciendo que sus componentes funcionen de forma ordenada.

Entre las principales funciones de un software se encuentran:

- Administrar los recursos del ordenador.
- Proporcionar las herramientas para optimizar estos recursos.
- Actuar como intermediario entre el usuario y la información almacenada.

El software puede aplicarse a numerosas situaciones del mundo real, como por ejemplo, a todos aquellos problemas para los que se haya establecido un conjunto específico de acciones que lleven a su resolución.

1.2.1 Aplicación

Aplicación es el término que se utiliza para designar un programa que se ejecuta en la computadora. Para evaluar si una aplicación está realmente bien construida no solo basta con que realice su tarea correctamente, sino también que sea fácil de utilizar por el usuario. Es decir, que el usuario se pueda relacionar con ella de forma rápida y comprensible. Para esto la aplicación dispone de un diseño el cual se llama Interfaz de usuario o conexión de usuario, actualmente casi todo el esfuerzo de quienes diseñan aplicaciones está orientado a lograr una interfaz lo más amistosa e intuitiva posible.

Una aplicación web es un sistema informático que los usuarios utilizan accediendo a un servidor web a través de Internet o de una intranet. Las aplicaciones web son populares debido a la practicidad del navegador web como cliente ligero. Otra razón de su popularidad es la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software en miles de clientes potenciales. Es importante mencionar que una página web puede contener elementos que permiten una comunicación activa entre el usuario y la información, lo cual permite que el usuario acceda a ella de modo interactivo, gracias a que la página responderá a cada una de sus acciones.

1.3 Software de Gestión

La gestión, en el amplio mundo de la informática se entiende por el subsistema encargado de la gestión y control de los repositorios de información, de los grupos de usuarios, y de los procesos de soporte para otros subsistemas. El software de gestión se encarga de definir y controlar los flujos de trabajo que son utilizados por los otros subsistemas, y de la definición de parámetros para el funcionamiento del sistema.

El procesamiento de información de gestión constituye, casi desde los inicios de la informática la mayor de las áreas de aplicación de los ordenadores. Estos programas utilizan grandes cantidades de información almacenadas en bases de datos con objeto de facilitar las transacciones comerciales o la toma de decisiones. Además de las tareas convencionales de procesamiento de datos, en las que el tiempo de procesamiento no es crítico y los errores pueden ser corregidos a posteriori.

Las empresas, independientemente de su tamaño, enfrentan demandas respecto a rentabilidad, calidad, tecnología y desarrollo sostenible. Un sistema de gestión eficiente, diseñado a la medida de sus procesos comerciales, puede ayudar a enfrentar los desafíos del cambiante mercado global de hoy. (2)

La gestión de la información puede definirse como el conjunto de actividades realizadas con el fin de controlar, almacenar, y posteriormente, recuperar adecuadamente la información producida, recibida o retenida por cualquier organización en el desarrollo de sus actividades. Un sistema de gestión puede ayudar a centrar, organizar y sistematizar los procesos para la gestión y mejora. (3)

1.4 Proceso Unificado de Desarrollo de Software (RUP)

Una metodología es necesaria para el proceso de desarrollo de software; conlleva a construir software de calidad, en el tiempo esperado y con el coste esperado. Para guiar el desarrollo de este software se utilizará la metodología RUP, llamada así por sus siglas en inglés *Rational Unified Process*.

RUP "se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo concluye con una versión del producto para los clientes". (4)

Esta metodología utiliza el Lenguaje Unificado de Modelado (UML, *Unified Modeling Language*) para preparar todos los esquemas de un sistema de software. UML es una parte esencial del Proceso Unificado, fue desarrollado paralelamente con RUP por las mismas personas, haciendo que su integración sea un éxito.

RUP posee tres características fundamentales que lo distinguen del resto de las metodologías:

- Dirigido por casos de uso: A partir de la identificación de los casos de uso todos los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso.
- Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente.
- Iterativo-incremental: RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Es práctico dividir el trabajo en partes más pequeñas o mini proyectos. Cada mini

proyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto.

Consta de cuatro fases Inicio, Elaboración, Construcción y Transición y nueve Disciplinas. En cada fase se emplean todas las disciplinas de ingeniería, pero con diferentes énfasis. Además contempla disciplinas de soporte que involucran actividades de planificación de recursos humanos, tecnológicos y financieros. (Ver Anexo 1)

Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación del software

Específicamente para el desarrollo del proyecto se escogió la metodología RUP porque se necesita de extrema organización y abundante documentación para garantizar la continuidad del proyecto, mitigando el riesgo que acarrean las características del equipo de desarrollo. El mismo está constituido, en su mayoría, por estudiantes, una cantidad significativa de ellos son de quinto año y próximos a graduarse e irse del proyecto y la otra parte del equipo pueden salir del proyecto por variadas razones, como por ejemplo dar clases en las MiniUCI, IPI.

1.5 Arquitectura Multicapa

Una arquitectura multicapa divide el sistema en subsistemas o capas independientes y en constante comunicación, lo que permite una mayor flexibilidad y mantenimiento de las distintas capas y el sistema en general, además, proporciona una clara división de responsabilidades. Conforman estas capas: presentación o interfaz, lógica de negocio, persistencia de dato; esta estructura da la posibilidad de desarrollar cada subsistema en paralelo. Los sistemas con tres o más capas se han probado como más escalables y flexibles que un sistema cliente-servidor, en el que no existe la capa central de lógica de negocio. (5)

La arquitectura utilizada para el desarrollo del proyecto es multicapa donde la capa de presentación tiene como objetivo crear un punto de unión entre la capa de cliente y la capa de lógica de negocio. La capa de la lógica de negocio contiene los objetos y servicios de negocio de la aplicación. Se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y se apoya del acceso a datos, para hacer solicitudes de almacenamiento o recuperación de datos al gestor de base de datos. La capa de persistencia de datos contiene bases de datos relacionales, bases de datos orientadas a objetos, y sistemas antiguos.

1.5.1 Tecnologías elegidas para cada capa

En el Anexo 2 se representa la gráfica de la arquitectura del sistema, con su correspondiente distribución de las capas de la aplicación.

Capa de Presentación

Java Server Faces (JSF) es un *framework*[3] de componentes de interfaz de usuario del lado del servidor para aplicaciones *web* basadas en Java. Contiene dos librerías de etiquetas personalizadas para *JavaServer Pages*[4] (JSP) que permiten expresar una interfaz *JavaServer Faces* dentro de una página JSP. Incluye un *API*[5] para representar componentes de interfaz de usuario (UI) y manejar sus estados, manejar sus eventos, la validación del lado del servidor y la conversión de datos, definir la navegación entre páginas y proporcionar extensibilidad para todas estas características.

JSF (*Java Server Faces*) es un *framework* de desarrollo basado en el patrón MVC (Modelo Vista Controlador). Ofrece una clara separación entre el comportamiento y la presentación. Une los componentes *UI* con los conceptos de la capa-*web* sin limitarse a una tecnología de *script* o lenguaje de marcas particular.

Capa de la Lógica de Negocio

Spring es un *framework* de código abierto que define la forma de desarrollar aplicaciones *J2EE*, dando soporte y simplificando complejidad propia del software corporativo. Promueve el bajo acoplamiento a partir de la inyección de dependencias (*DI*) entre los objetos relaciones. Provee un contexto apropiado para el desarrollo de aplicaciones *web* e integración con otros *frameworks* (*Struts, JSF, Tapestry, Hibernate*, etc).

Spring actúa como un contenedor de *beans*[6]. Un *beans* puede verse como una clase java simple o *Plain Old Java Object*[7] *(POJO*) y revalora la simplicidad de las clases Java aportando manejo de transacciones de forma no intrusiva, como la configuración basada en archivos *XML*[8]. Además, Spring provee un modelo de programación consistente, ayuda a promover la reutilización de código, a facilitar el diseño orientado a objeto en aplicaciones *J2EE*. En el desarrollo del sistema se usa este *framework* para comunicar la abstracción del modelo relacional de *Hibernate* con la capa de negocio.

Capa de Acceso a Datos

Persistencia de Objetos utilizando *Hibernate*.

Hibernate es *framework* de código abierto para el mapeo objeto-relacional, permite desarrollar clases persistentes y lógica persistente sin importar cómo manejar los datos. *Hibernate* no sólo se ocupa de la

abstracción de los datos persistentes en el gestor de base de datos utilizando el mapeo de clases Java a tablas de la base de datos y de tipos de datos Java a tipos de datos *SQL*[9]; sino que también provee facilidades de consultas de datos y de recuperación. Además, exime al desarrollador de un gran por ciento de las tareas de programación relacionadas con la persistencia de datos.

Hibernate es el *framework* que se utiliza para el mapeo de las clases porque provee mecanismos de soporte de transacciones, propagación de persistencia, recuperación eficiente de objetos, fijación de niveles de aislamiento y bloqueo para datos sensibles. Además permite la integración con *Spring* aunque puede usarse de forma independiente.

1.6 Servidor Web Apache Tomcat

Apache Tomcat (también llamado **Jakarta Tomcat** o **Tomcat**) implementa las especificaciones de los *servlets*[10] y de *JavaServer Pages (JSP)*. Jakarta Tomcat es un subproyecto de Jakarta de la fundación de software Apache que provee un poderoso servidor *web* integrado con Apache (Apache Tomcat). Tomcat es desarrollado en Java, además multiplataforma y fácil de extender en funcionalidades. Apache por su parte aporta la ventaja de su arquitectura modular. Estas especificaciones son desarrolladas bajo el proceso de la comunidad Java. Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Licencia Apache (*Apache Software Licence*).

1.7 Plataforma Java

La plataforma Java permite crear y ejecutar aplicaciones de forma interactiva, dinámica y segura en un entorno de red. Dispone de dos componentes fundamentales:

- La Máquina Virtual *(JVM)* o Entorno en Tiempo de Ejecución para Java (*JRE*): es la encargada de ejecutar los programas escritos en este lenguaje de programación. Permite que las aplicaciones desarrolladas en dicho lenguaje puedan ser ejecutadas en diversos sistemas con arquitecturas diferentes. Es lo mínimo que se necesita en un sistema para ejecutar una aplicación.
- La Interfaz de Programación de Aplicaciones (*API*): es un conjunto de clases ya desarrolladas que ofrecen un gran abanico de posibilidades al programador; por lo que ofrece un conjunto de librerías estándar, que contiene muchas de las funciones reutilizables disponibles en los sistemas operativos actuales, como el manejo de interfaces gráficas, de archivos, acceso a Internet, acceso a bases de datos.

Una de las ventajas de la plataforma Java es que al ser basada en el lenguaje Java, es posible desarrollar arquitecturas utilizando cualquier sistema operativo donde se pueda ejecutar su máquina virtual.

Actualmente existen varias ediciones de la plataforma Java, cada una de ellas destinada a cubrir un conjunto diferente de necesidades de programación: *J2ME (Java2 Micro Edition)* usada en aplicaciones para móviles, *J2EE* (*Java2 Enterprise Edition*) usada en aplicaciones de empresas y *J2SE* o *JSE* (*Java Standard Edition*) para aplicaciones de escritorio.

La plataforma *Java 2 Platform, Standard Edition (J2SE)* 5.0 es también conocida como *J2SE 1.5* o *JDK 1.5*. Dentro de las ventajas de esta plataforma para el desarrollo *web*, se encuentran; gran rendimiento en 64-bit y escalabilidad, tomando como referencia que, tanto para el cliente como para el servidor, cuenta con una gran cantidad de librerías, además permite a las aplicaciones utilizar más de 4 GB de espacio de almacenamiento dinámico. También proporciona herramientas para generar código de lenguaje e información de configuración, incluyendo servicios *web*.

Se utiliza el *J2EE* debido a que recoge todas las funcionalidades del *JSE* y agrega otras que son útiles para programas que se ejecutan en servidores como *JavaMail*[11], *RMI*[12] (Java Remote Method Invocation), Servicios Web, entre otras.

1.7.1 Ventajas de la Plataforma J2EE.

Ventajas para el usuario

La plataforma Java ofrece la posibilidad de acceder a aplicaciones a través de la *web* con acceso instantáneo a los programas. Desde el principio, las aplicaciones están disponibles para su ejecución en cualquier sistema operativo liberando a los usuarios de tener que elegir un cierto sistema operativo. Con Java, los más pequeños sistemas pueden ser utilizados para ejecutar aplicaciones especializadas.

Ventajas para el desarrollador

El lenguaje Java es un conjunto de *APIs* reducido, bien conocido y documentado. Los desarrolladores pueden escribir sus aplicaciones sólo una vez para ejecutarlas en cualquier lugar, por lo que los entornos de desarrollo Java generan un único código binario en cualquier sistema operativo.

Ventajas para el administrador

La plataforma Java proporciona ventajas para los departamentos de administración de sistemas de una compañía. Los controles de versiones y actualizaciones se simplifican ya que las aplicaciones Java son

capaces de mantenerse en un lugar centralizado a través del cual se puede distribuir a cada usuario lo que necesite usar.

Esta plataforma ha sido creada con la participación de diversos desarrolladores, y es una plataforma conjunta basada en el lenguaje de programación Java.

1.7.2 Lenguaje de Programación Java

Java nace como un lenguaje de programación fácil de utilizar, lo que lo convierte en uno de los lenguajes más elaborados y utilizados para la creación de software.

El lenguaje de programación Java surge a partir de las características de diferentes lenguajes, principalmente de C++. (8) A continuación se describen algunas de sus características:

Simple

Ofrece toda la funcionalidad de un lenguaje potente. Java elimina muchas de las características de otros lenguajes como C++, para mantener reducidas las especificaciones del lenguaje y añadir características muy útiles como el *garbage collector* (reciclador de memoria dinámica). Java reduce en un 50% los errores más comunes de programación con lenguajes como C y C++ al eliminar muchas de las características de éstos.

Orientado a Objetos

Java implementa la tecnología básica de C++ con algunas mejoras y elimina algunas cosas para mantener el objetivo de la simplicidad del lenguaje. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo.

Distribuido

Java se ha construido con extensas capacidades de interconexión *TCP/IP*[13]. Existen librerías de rutinas para acceder e interactuar con protocolos como *http*[14] y *ftp*[15]. La característica de ser distribuido es debido a que proporciona las librerías y herramientas para que los programas puedan ser distribuidos, es decir, que puedan ejecutarse en varias máquinas.

Robusto

Java realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores, lo antes posible, en el ciclo de desarrollo.

Arquitectura Neutral

Para establecer Java como parte integral de la red, el compilador Java compila su código a un fichero objeto de formato independiente de la arquitectura de la máquina en que se ejecutará.

Interpretado

Se traduce el código fuente a un código intermedio denominado *ByteCodes*, que es interpretado por la Maquina Virtual de *Java*, lo cual permite que se pueda ejecutar en cualquier sistema operativo.

Seguro

No se permite el acceso ilegal a memoria ya que no se trabaja con punteros.
El código Java pasa muchas *pruebas* antes de ejecutarse en una máquina. El código se pasa a través de un verificador de *ByteCodes* que comprueba el formato de los fragmentos de código y aplica un probador de teoremas para detectar fragmentos de código ilegal, que falsean punteros, violan derechos de acceso sobre objetos o intentan cambiar el tipo o clase de un objeto.

Multithread. (Multihilos)

Permite la ejecución de varias tareas a la vez en un programa (*multi-threaded*).
El beneficio de ser *multi-threaded* consiste en un mejor rendimiento interactivo y mejor comportamiento en tiempo real. Aunque el comportamiento en tiempo real está limitado a las capacidades del sistema operativo subyacente (*Unix, Windows, etc.*), aún supera a los entornos de flujo único de programa (*single-threaded*) tanto en facilidad de desarrollo como en rendimiento.

Dinámico

No conecta todos los módulos que comprende una aplicación hasta el tiempo de ejecución, ya que las librerías nuevas o actualizadas no paralizarán las aplicaciones actuales siempre que mantengan el *API* anterior. (9)

1.8 Entorno Integrado de Desarrollo Eclipse

Eclipse es un Entorno de Desarrollo Integrado (IDE) sustentado por la comunidad de código abierto “Fundación Eclipse”. Permite crear entornos de desarrollo integrados para distintos lenguajes programación como Java, PHP, Python y C++; permitiendo incorporar módulos que brindan el soporte a un lenguaje de programación específico, como también módulos de herramientas que complementan el

entorno de trabajo. Un ejemplo de esto es la integración con sistemas de control de versiones como *SVN*, integración con *Visual Paradigm*[17], *Jacarta Tomcat*, entre otros.

Este entorno de desarrollo fue creado inicialmente por la *IBM*[16] y actualmente por la Fundación Eclipse. Es multiplataforma y permite trabajar con varios proyectos a la vez.

Dentro de sus principales características están:

- Editor de texto Resaltado de sintaxis.
- Compilación en tiempo real.
- Asistentes (wizards): para creación de proyectos, clases, tests, etc.
- Refactorización.

1.9 Servidor de Base de Datos PostgreSQL

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (*Object-Relational Database Management System* (ORDBMS)) libre, no tiene costo asociado, liberado bajo la *licencia BSD*[18], por lo que cualquiera puede disponer de su código fuente, modificarlo a voluntad y redistribuirlo libremente.

PostgreSQL presenta alta concurrencia, para esto utiliza la tecnología de Control de Concurrencia Multi-Versión (*Multiversion concurrency control* (MVCC)), con lo que se logra que ningún lector sea bloqueado por un escritor. Es altamente extensible, soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario. Tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/PGSQL. Este lenguaje es comparable al lenguaje procedural del sistema de gestión de base de datos relacional *Oracle*, PL/SQL.

En cuanto a sus funciones, poseen bloques de código que se ejecutan en el servidor los cuales pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos brinda. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que se intente conectar a la base de datos. Tiene una excelente documentación y está bien organizada, además de contar con una comunidad de usuarios y desarrolladores a los que acudir en caso de tener problemas.

1.10 Control de Versiones

Con el objetivo de garantizar un correcto control de versiones y gestión de cambio se decidió utilizar el Subversion con el cliente TortoiseSVN. Subversion, maneja y almacena ficheros y directorios. Los ficheros se almacenan en un repositorio central. El repositorio es prácticamente igual a un servidor de ficheros

ordinario, salvo que lleva el control de todos los cambios que se hayan hecho a sus ficheros y directorios. Esto permite que puedan recuperar versiones antiguas, examinar el historial de modificaciones, así como identificar quién realizó dicho cambio en cualquiera de estos ficheros según se desee.
TortoiseSVN se integra con el explorador de Windows, lo que permite un fácil acceso a los comandos de Subversión, al estar disponibles desde el menú contextual del explorador. Los menús contextuales de TortoiseSVN también funcionan en otros administradores de archivos.

1.11 Dreamweaver

Dreamweaver es una herramienta de diseño de páginas *web*. Se integra con otras herramientas como Adobe Flash (herramienta de animación) y soporta los estándares del *World Wide Web Consortium*[19]. Aunque su licencia no es libre las funciones de edición visual permiten agregar rápidamente diseño y funcionalidad a las páginas, sin la necesidad de programar manualmente el código *HTML*[20] y es fácil de usar. Contiene útiles herramientas para su utilización como: tablas, marcos, trabajo con capas, inserción de comportamientos *JavaScript*[21], etc. Además tiene funciones de auto-completamiento y resaltado de la sintaxis para instrucciones en *HTML*, confección de hojas de estilos y lenguajes de programación como *PHP*, *JSP* o *ASP*.

1.12 Microsoft Office Visio

Microsoft Visio es un conjunto de software de dibujo vectorial para Microsoft Windows. Las herramientas que lo componen permiten realizar diagramas de oficinas, diagramas de bases de datos, diagramas de flujo de programas, UML, entre otros. Contiene aditamentos para desarrollar diagramas de negocios de manera sencilla. Visio se utiliza en la solución para crear los grafos de flujos en los casos de prueba.

1.13 Conclusiones

En el presente capítulo se realizó un análisis de las tecnologías y herramientas utilizadas a lo largo del desarrollo del sistema propuesto. Para llevar a cabo el sistema se hace uso de la tecnología para la programación de páginas dinámicas en el lenguaje Java y como plataforma de desarrollo Java. Se seleccionó como metodología de desarrollo RUP, el cual está basado en la orientación a objetos, el moldeamiento visual usando UML y como soporte de base de datos en PostgreSQL

Descripción y análisis de la solución propuesta

2.1 Introducción

En este capítulo se describen las funcionalidades de los módulos que fueron implementadas. Se muestran los estándares de codificación utilizados en la implementación de la solución propuesta. Además se describen los patrones de diseño usados para garantizar una mayor robustez y reutilización del código.

2.2 Funcionalidades de los módulos

El sistema cuenta con dos módulos principales (módulos “Funcionamiento” y “Estadística”) los que recogen las principales funcionalidades del sistema; y uno de apoyo (módulo “Seguridad”) que se encarga de la gestión de usuarios, estructuras y lo referente a la autenticación para acceder al sistema. A continuación se explican las funcionalidades que brindan cada uno de estos, especificando en el módulo “Seguridad” lo referente a la gestión de las estructuras solamente.

Al sistema pueden acceder usuarios de diferentes niveles estructurales. Estos niveles son: Nacional, Provincial, Municipio o Centro de Enseñanza Superior y Sede Universitaria o Facultad.

2.2.1 Módulo Funcionamiento

El módulo “Funcionamiento” es el encargado de adicionar las reuniones que se llevan a cabo por parte de la organización a diferentes niveles (provincial, municipal, brigada, etcétera).

En las figuras 1 y 2 se muestran los datos necesarios para crear una nueva reunión. En dependencia de los datos escogidos en los campos de **Estructura** y **Tipo de reunión**, variarán los datos que tendrán que ser introducidos. Una vez introducidos los datos obligatorios del primer formulario (figura 1), se puede llegar al formulario de la figura 2 haciendo clic sobre el botón **Siguiente>>**, al llenar los datos se pulsa en el botón **Guardar** y se almacenan los datos de la reunión. En caso de querer modificar algún dato del primer formulario se presiona el botón **<<Anterior**.

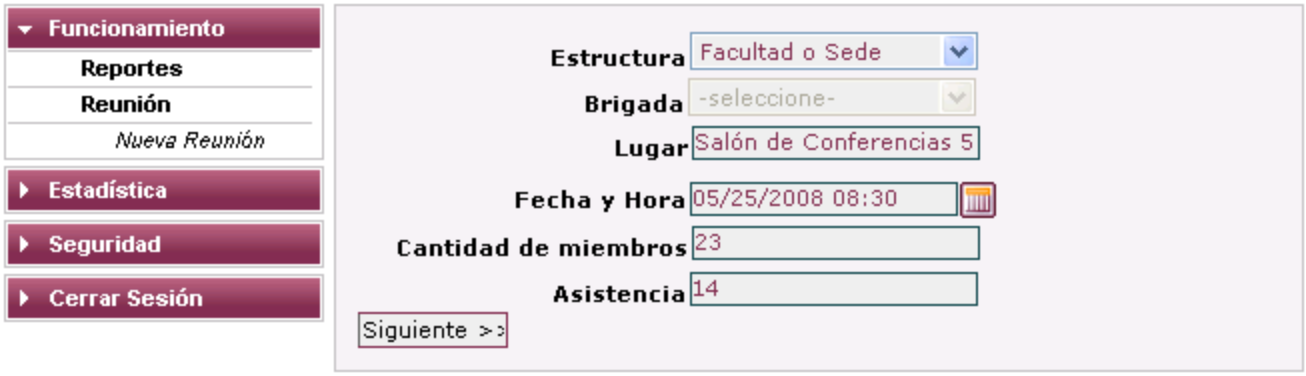


Figura 1. Crear Nueva Reunión 1.

Figura 2. Crear Nueva Reunión 2.

2.2.2 Módulo Estadística

El módulo “Estadística” permite buscar caracterizaciones estadísticas. Además permite a los usuarios con los permisos suficientes crear, modificar, trasladar o dar baja a una caracterización.

La figura 3 muestra los datos por los que se pueden realizar búsquedas de caracterizaciones estadísticas. La cantidad de parámetros de búsqueda estará en consecuencia del nivel al que pertenezca el usuario que está autenticado. Para iniciar una búsqueda desde cero después de haber especificado algunos parámetros de búsqueda, se pulsa en el botón **Resetear Búsqueda**. Una vez que se oprima el botón **Buscar**, el sistema muestra el nombre y apellidos de las caracterizaciones encontradas, donde el usuario podrá hacer clic para los detalles.

Figura 3. Buscar Caracterización Estadística.

Para darle baja a una caracterización estadística (figura 4) el usuario selecciona la brigada, la caracterización y el motivo de la baja; en dependencia de este último tendrá que introducir datos adicionales (figura 5). Por último pulsa en **Aceptar**.

Figura 4. Dar Baja a una Caracterización 1.

Figura 5. Dar Baja a una Caracterización 2.

En las figuras 6, 7 y 8 se muestran los campos necesarios para crear una nueva caracterización estadística. Una vez introducidos los datos obligatorios del primer formulario (figura 6) se puede llegar al formulario de la figura 7 haciendo clic sobre el botón **Siguiente>>**, al igual que para llegar al tercer formulario, donde al llenar los datos se pulsa en el botón **Guardar** y se almacena la caracterización. En caso de querer modificar algún dato de los formularios anteriores se presiona el botón **<<Anterior**.

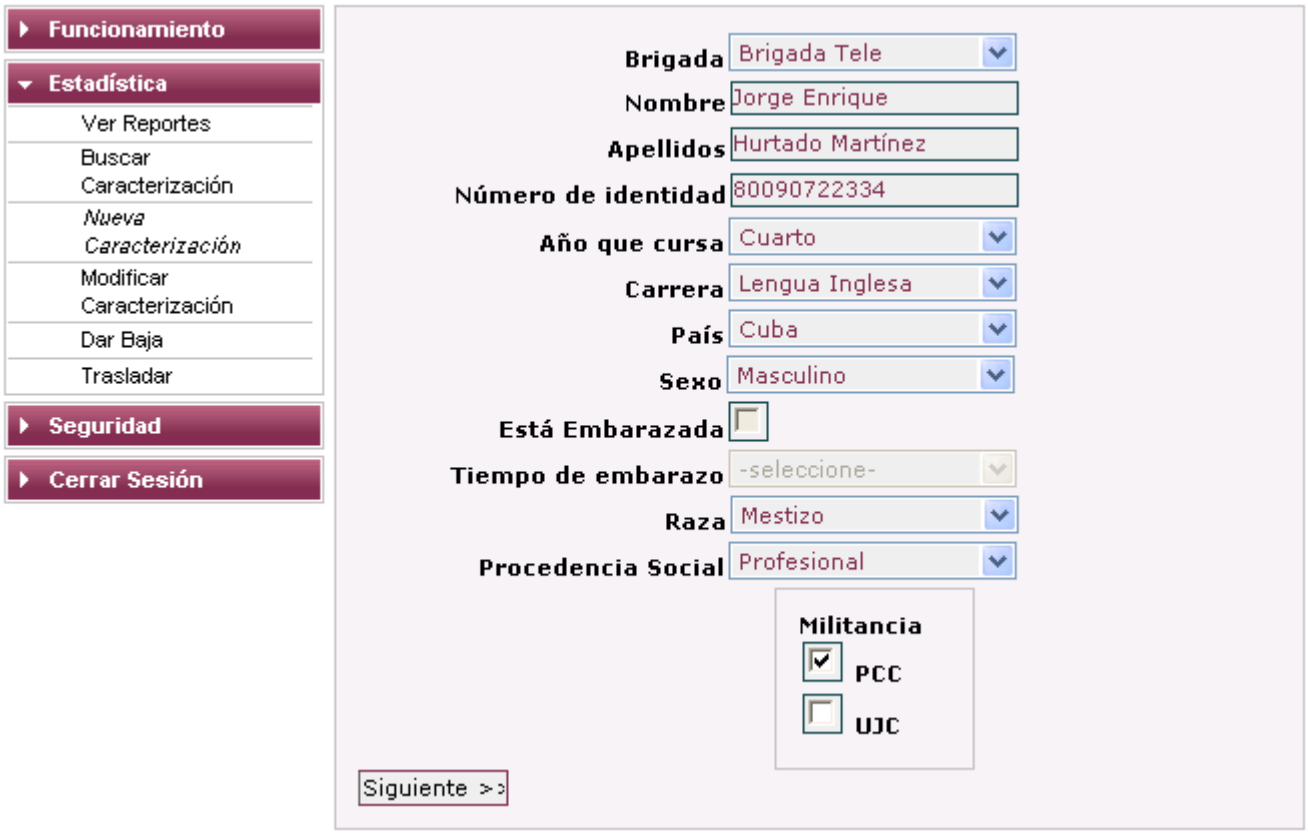


Figura 6. Crear Nueva Caracterización 1.

Figura 7. Crear Nueva Caracterización 2.

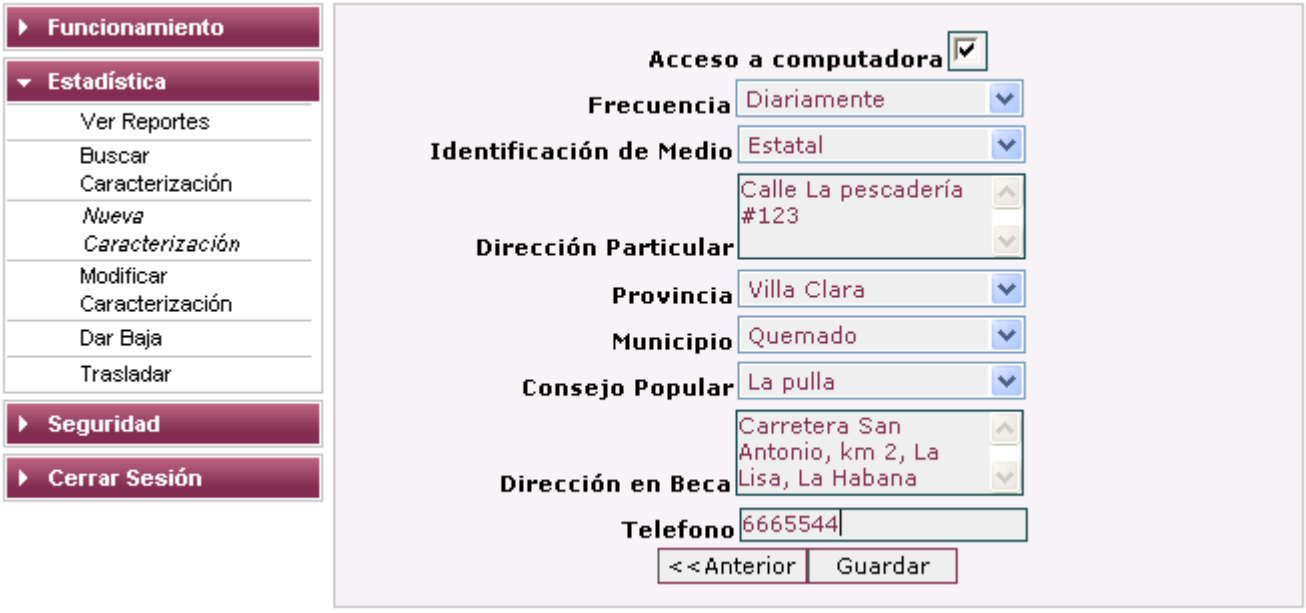


Figura 8. Crear Nueva Caracterización 3.

Otra funcionalidad que brinda el módulo “Funcionamiento” es la de trasladar caracterizaciones de una brigada hacia otra (figura 9). Una vez seleccionada la brigada de origen se listan las caracterizaciones existentes en dicha brigada, permitiendo escoger las que se deseen trasladar hacia **Nueva Brigada** haciendo clic en el botón **Trasladar**.

Figura 9. Trasladar Caracterización.

En las figuras 10, 11 y 12 se muestran los campos necesarios para modificar una caracterización estadística en caso de que algún dato sufra cambios o se haya cometido algún error a la hora de guardarla. Para llevar a cabo la acción primero se selecciona una brigada y luego la caracterización. Después se cargan los campos con los datos específicos de la caracterización seleccionada anteriormente. Mediante los botones **Siguiente>>** y **<<Anterior** se navega entre los distintos formularios. El usuario modifica los campos que se deseen. Para terminar se hace clic en el botón **Actualizar** y se guardan los cambios realizados.

Figura 10. Modificar Caracterización Estadística 1.

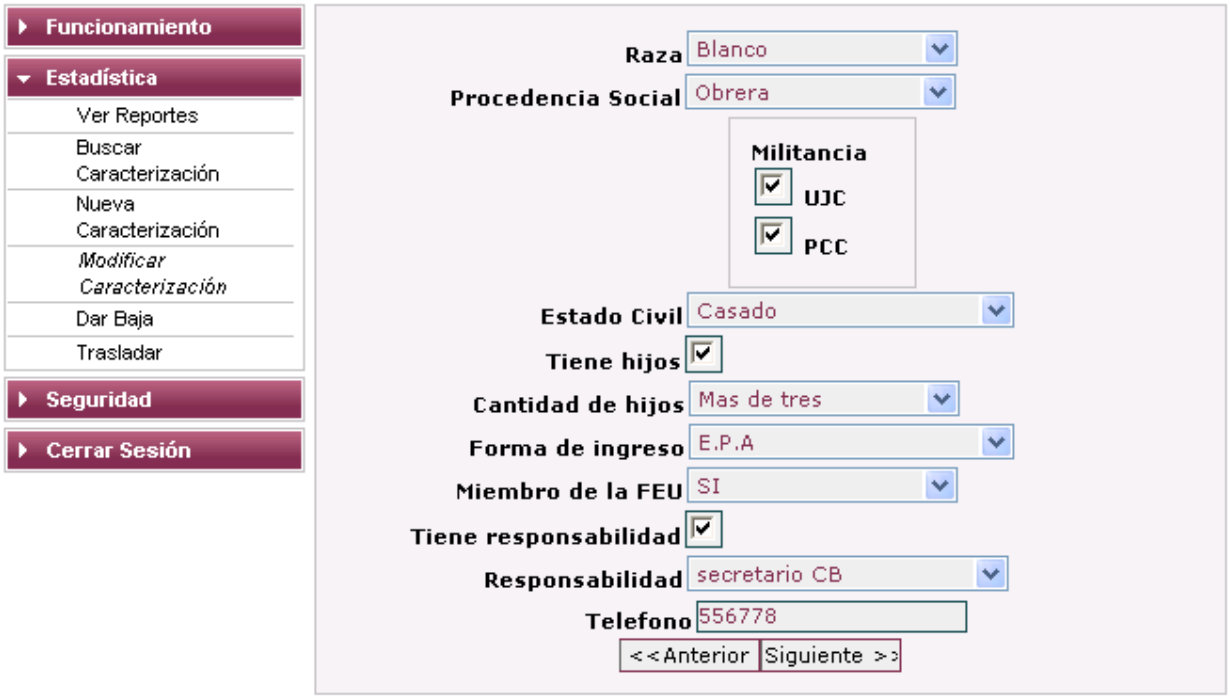


Figura 11. Modificar Caracterización Estadística 2.

Figura 12. Modificar Caracterización Estadística 3.

2.2.3 Módulo Seguridad

Las funcionalidades del módulo “Seguridad” que se tratan son las de adicionar una nueva estructura y modificar una ya existente.

En las figuras 13 y 14 se muestran dos escenarios para adicionar una estructura a diferentes niveles de usuario. En el primer caso se encuentra autenticado un usuario a nivel nacional, y en el segundo uno a nivel de facultad. En ambos casos después de llenar los campos se hace clic en el botón **Adicionar** y se crea la estructura.

Figura 13. Crear Nueva Estructura 1.

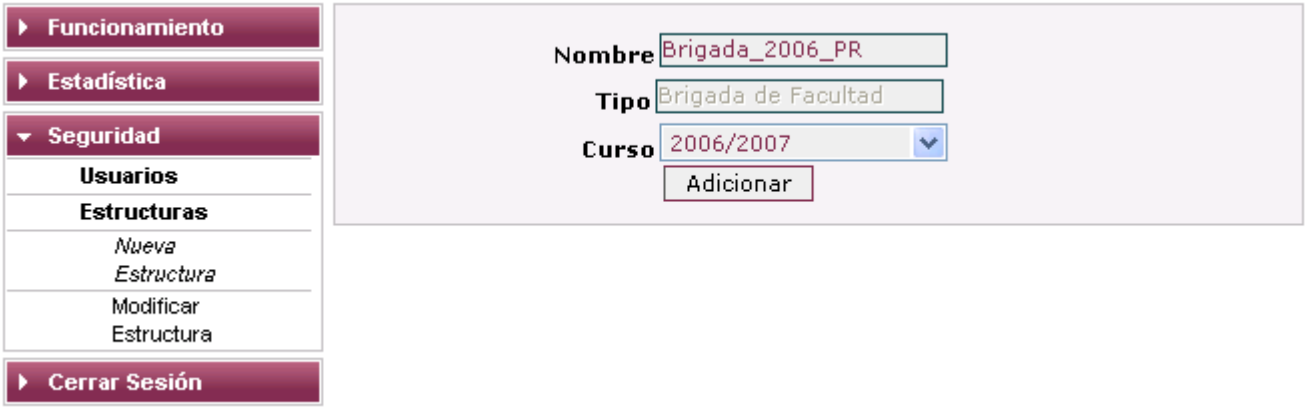


Figura 14. Crear Nueva Estructura 2.

Si en algún momento se desea modificar una de las estructuras existentes, se accede a la opción de **Modificar Estructura**, se escoge la que se desee cambiar y una vez realizados estos cambios se pulsa el botón **Actualizar**, actualizándose los campos de la estructura. (Figura 15)

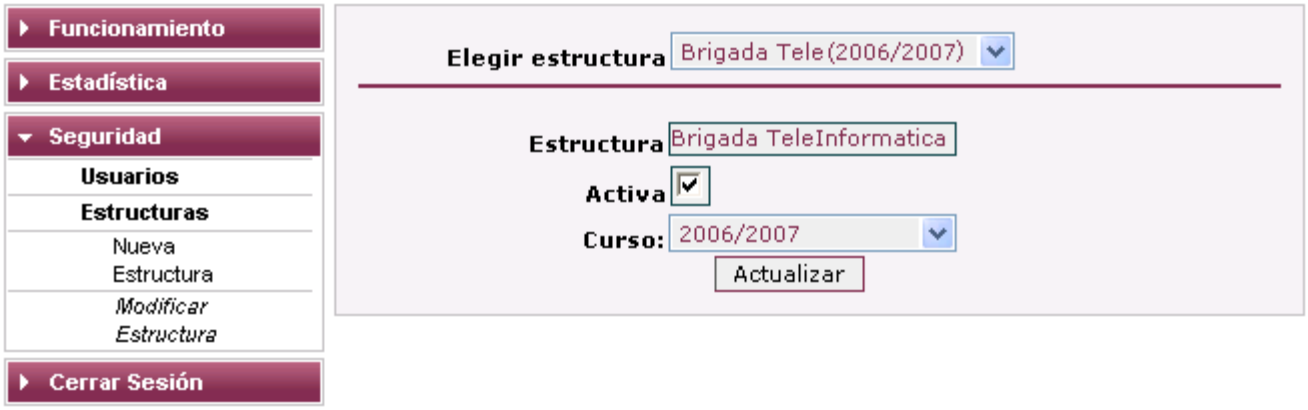


Figura 15. Modificar Estructura.

2.3 Patrones de diseño

Patrones de diseño, del inglés *Design Patterns*, son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Son soluciones basadas en la experiencia y que se ha demostrado que funcionan. (11)

2.3.1 Patrón DAO

El patrón DAO (*Data Acces Object*) encapsula la forma de acceder a la fuente de datos. Este patrón surge históricamente de la necesidad de gestionar una diversidad de fuentes de datos. Su uso se extiende al problema de encapsular no sólo la fuente de datos, sino además ocultar la forma de acceder a los datos. Se trata de que el software cliente se centre en los datos que necesita y se olvide de cómo se realiza el acceso a los datos o de cuál es la fuente de almacenamiento. El uso de este patrón permite hacer la aplicación lo más independiente posible de una base de datos concreta.

2.3.2 Patrón Facade

En ocasiones una librería o serie de librerías contienen un conjunto de funcionalidades para realizar diversas acciones. Si se analiza, en la gran mayoría de los proyectos de software, se observará que la mayor parte de las llamadas que recibe la librería es a las mismas funciones o a una combinación de ellas. Es decir, existe una gran cantidad de peticiones comunes y que satisfacen las necesidades del cliente de la librería. En estos casos, es conveniente crear un objeto que simplifica el *API* de la librería, unificando y simplificando ciertos procesos del *API*.

El patrón Facade simplifica el acceso a un conjunto de clases proporcionando una única clase que todos utilizan para comunicarse con dicho conjunto de clases. Las ventajas fundamentales del uso de este patrón son que los clientes no necesitan conocer las clases que hay tras la clase Facade y se pueden cambiar las clases "ocultadas" sin necesidad de cambiar los clientes. Sólo hay que realizar los cambios necesarios en Facade.

En la implementación del sistema se evidencia el uso del patrón mencionado, por ejemplo en la capa de negocio con la clase ServiciosFuncionamiento.java. Esta clase recoge todas las funcionalidades que ofrecen las clases DAO del módulo Funcionamiento.

2.3.3 Patrón MVC

El patrón MVC es un patrón de diseño de software en el cual todo el proceso está dividido en 3 capas, típicamente estas capas son el Modelo, la Vista y el Controlador. Es un patrón recomendado para aplicaciones interactivas Java. MVC separa los conceptos de diseño, y por lo tanto hay un decremento en la duplicación de código, el centralizamiento del control y hace que la aplicación sea más extensible. Este patrón de diseño arquitectural se aplica en la capa de presentación.

El **Modelo** incorpora la capa del dominio y persistencia, es el encargado de guardar los datos en un medio persistente. Es el proveedor de los recursos, independiente al sistema de almacenamiento de datos. En nuestro caso esto es así, debido a que la forma de acceder a los datos en nuestro sistema no viene condicionada por la forma en la que se encuentren almacenados los mismos.

La **Vista** se encarga de presentar la interfaz al usuario en sistemas *web*. En esta capa solo se deben de hacer operaciones simples, condicionales, ciclos, formateo, etc. Recibe los datos del modelo y los muestra al usuario. Normalmente, tienen un registro del controlador asociado. En nuestro caso, estas vistas son instanciadas por el controlador para la presentación de los datos al usuario.

El **Controlador** es el que escucha los cambios en la vista y se los envía al modelo, el cual le regresa los resultados del envío a la vista, es un ciclo donde cada acción del usuario causa que se inicie un nuevo ciclo.

Desde la perspectiva de la aplicación, los *bean*[5] de *Java Server Face* forman parte de la capa de Modelo, las páginas *JSP* con etiquetas *JSF* son la capa de la Vista y el *Servlet Face*[6] proporciona la funcionalidad de controlador.

2.4 Estándares de codificación

Un estándar de codificación es “Un conjunto de reglas de notación y nomenclatura, específicas de cada lenguaje de programación, que se usan y se siguen durante la fase de implementación (codificación) de una aplicación y reducen perceptiblemente el riesgo de que los desarrolladores introduzcan errores que no son detectados por los compiladores, reduciendo el tiempo y coste de las actividades de depuración y pruebas necesarias para la detección y corrección de los mismos.” (10)

Las convenciones de nombres hacen los programas más entendibles haciéndolos más fácil de leer. También pueden dar información sobre la función de un identificador, por ejemplo, cuando es una constante, un paquete, o una clase, que puede ser útil para entender el código. En la siguiente tabla se especifican las convenciones de nombres utilizadas.

Tabla 1. Convenciones de nombres utilizados.

Tipos de identificadores	**Reglas para nombres**	**Ejemplos**
Paquetes	El prefijo del nombre de un paquete se escribe siempre con letras *ASCII*[22] en minúsculas, y debe ser uno de los nombres de dominio de alto nivel, actualmente com, edu, gov, mil, net, org, o uno de los códigos ingleses de dos letras que identifican cada país como se especifica en el ISO Standard 3166, 1981. Los subsecuentes componentes del nombre del paquete variarán de acuerdo a las convenciones de nombres internas de cada organización. Dichas convenciones pueden especificar que algunos nombres de los directorios correspondan a divisiones, departamentos, proyectos o máquinas.	cu.uci.kainos.feu.estadistica.modelo.dao cu.uci.kainos.feu.seguridad.vista.bean
Clases	Los nombres de las clases deben ser sustantivos, cuando son compuestos tendrán la primera letra de cada palabra que lo forma en mayúsculas. Intentar mantener los nombres de las clases simples y descriptivos. Usar palabras completas,	class Persona; class ReunionPolitica;

	evitar acrónimos y abreviaturas (a no ser que la abreviatura sea mucho más conocida que el nombre completo, como *URL*[23] o HTML). Nota: El estándar de codificación para los nombres de las clases será especificado con mayor detalle a continuación de esta tabla.	
Interfaces	Los nombres de las interfaces siguen la misma regla que las clases.	interface ReunionBase; interface Almacen;
Métodos	Los métodos deben ser verbos. Cuando son compuestos tendrán la primera letra en minúscula, y la primera letra de las siguientes palabras que lo forma en mayúscula.	realizarTraslado(); dibujar();
Variables	Excepto las constantes, todas las instancias y variables de clase o método empezarán con minúscula. Las palabras internas que lo forman (si son compuestas) empiezan con su primera letra en mayúsculas. Los nombres de variables no deben empezar con los caracteres subguión "_" o signo del dólar "$", aunque ambos están permitidos por el lenguaje. Los nombres de las variables deben ser cortos pero con significado. La elección del nombre de una variable debe ser un mnemónico, designado para indicar a un observador casual su función. Los nombres de variables de un solo caracter se deben evitar, excepto para variables índices temporales. Nombres comunes para variables temporales son i, j, k, m, y n para enteros; c, d, y e para caracteres.	int i; char c; float miAnchura;
Constantes	Los nombres de las variables declaradas como constantes deben ir totalmente en mayúsculas separando las palabras con un subguión ("_").	static final int ANCHURA_MINIMA =4; static final int COGER_LA_CPU = 1;

Se definen convenciones de nombres para las distintas clases java dependiendo de las funciones que tengan cada una de estas en la aplicación:

- Clases de la capa de Acceso a Datos:
  - Las interfaces que representan las operaciones sobre los objetos de acceso a datos, correspondientes al patrón de diseño Data Access Object (DEEPAK ALUR 2003) terminan con la palabra “Dao”. Ejemplo: ReunionDao.
  - Las implementaciones reales de las interfaces Dao comienza con el nombre de la interfaz correspondiente y terminan con la palabra “Impl”. Ejemplo: ReunionDaoImpl.
- Clases de la capa de Negocio:
  - Las interfaces que representan las funcionalidades de los DAO se declaran comenzando con la palabra “Servicios” seguido del nombre del módulo donde están ubicadas. Ejemplo: ServiciosFuncionamiento.
  - Las implementaciones reales de las interfaces que representan las funcionalidades de los DAO comienzan con el nombre de la interfaz y terminan con la palabra “Impl”. Ejemplo: ServiciosFuncionamientoImpl.
- Clases de la capa de Presentación:
  - En esta capa se definen los beans. Su nombre se define de la siguiente forma: [nombre_clase]+ “Bean”. Ejemplo: ReunionBean.
  - Cada clase bean que haga relación a una clase persistente tiene que tener un builder, su nombre será el del bean al que construye + “Builder”. Ejemplo: ReunionBuilder.
- Recursos
  - Las páginas HTML que constituyen recursos estáticos tendrán la extensión “.html”.
  - Las imágenes que son estáticas en la aplicación terminarán “.jpg”.
  - Los documentos constituyen recursos estáticos tendrán la extensión “.doc”.

2.5 Conclusiones

En este capítulo se expusieron las funcionalidades de los módulos que fueron implementadas. Se definieron los estándares de codificación puestos en práctica en la implementación de la solución propuesta, lo que ayudó a otros desarrolladores a entender mejor el código. Además se describieron los patrones de diseño usados.

Validación de la solución propuesta

3.1 Introducción

En el presente capítulo se comienza la validación de la solución propuesta a través de las diferentes pruebas de unidad. Dentro de estas pruebas, se utilizan los métodos de pruebas de caja blanca y pruebas de caja negra, definiéndose el objetivo de cada una de estas, así como su alcance y detalles de las mismas. Se analizan los defectos encontrados con el objetivo de que sean corregidos, para aumentar la fiabilidad del software, dando una medida de la calidad del mismo y su correspondencia con los requerimientos establecidos.

3.2 Tipos de pruebas

Garantizar la calidad de un producto de software implica varios tipos de pruebas. Cada tipo de prueba tiene sus propios requisitos y estrategias. El producto se va perfeccionando conforme se avanza de un tipo de prueba a otro, hasta que se ha probado por completo y está listo para su introducción en el entorno de producción. La prueba no puede asegurar la ausencia de defectos; sólo puede demostrar que existen errores en el software. Dentro de los diferentes tipos de pruebas se aplicarán las pruebas de unidad para la validación del producto.

3.2.1 Pruebas de unidad

Se centran en el análisis de un solo componente de la solución. Para crear la solución general, los miembros del equipo de cada grupo de trabajo empiezan a analizar los componentes de los que son responsables. En este momento, a menudo instalan dichas unidades en entornos aislados para validar las capacidades de estos. Este tipo de prueba, se realiza en un solo equipo.

Las Pruebas de Unidades se plantean a pequeña escala, y consisten en ir probando uno a uno los diferentes módulos que constituyen una aplicación para garantizar que cada uno, funcione correctamente de forma independiente, o sea, sin tener en cuenta las relaciones que pueda tener con otras partes del sistema. Las ventajas de usar este tipo de pruebas son muchas, entre ellas se pueden mencionar:

1. **Fomentan el cambio**: las pruebas unitarias facilitan que el programador cambie el código para mejorar su estructura (refactorización), puesto que permiten hacer pruebas sobre los cambios y así asegurarse de que los nuevos cambios no han introducido errores.
2. **Los errores son más fáciles de localizar**: bastará con ejecutar las pruebas unitarias confeccionadas y ver qué módulos no las pasan.
3. **Documenta el código**: las propias pruebas son documentación del código, puesto que ahí se puede ver cómo utilizarlo.
4. **Se reducen los "efectos secundarios"**: muchas veces, cuando se está corrigiendo algún error, se cometen otros. Aplicando las pruebas unitarias, es más fácil controlar esto, ya que pasando de nuevo las pruebas anteriores se asegura de que todo funciona tal y como se espera.
5. **Los errores están más acotados:** al tener las pruebas unitarias, se podrán probar los distintos módulos e identificar errores.
6. **Las pruebas funcionales se hacen más sencillas:** la mayoría de los aspectos individuales de cada unidad ya han sido probados a través de las pruebas unitarias. Así, las pruebas funcionales deben centrarse sólo en verificar la correcta cooperación de las distintas unidades, y en los funcionamientos generales del programa.

La realización de las pruebas unitarias no revelarán todos los errores. Por definición, sólo prueban las unidades por sí solas. Por lo tanto, no descubrirán errores de integración, y otros problemas que afectan al sistema en conjunto.

Dentro de las pruebas de unidades se encuentran las pruebas de caja blanca y las pruebas de caja negra.

3.3 Métodos de pruebas

3.3.1 Pruebas de Caja Blanca

Objetivo

El objetivo de realizar este tipo de prueba al sistema es garantizar que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo o método, todos los bucles (ciclos) en sus límites operacionales, así como las estructuras internas de datos para asegurar su validez.

Alcance

El proceso de pruebas de caja blanca se va a concentrar principalmente en validar que cada uno de los módulos o segmentos de códigos funcione apropiadamente.

Descripción

Las pruebas de caja blanca permiten examinar la lógica interna del programa sin considerar los aspectos de rendimiento. Se diseñan casos de prueba para examinar la lógica del programa. Es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivar casos de prueba que garanticen que:

- Se ejerciten todos los caminos independientes de cada módulo.
- Se ejerciten todas las decisiones lógicas.
- Se ejecuten todos los bucles.
- Se ejecuten las estructuras de datos internas.

La prueba de Caja Blanca es considerada como uno de los tipos de pruebas más importantes que se le aplican a un software, logrando como resultado que disminuya en un gran por ciento el número de errores existentes en los sistemas y por ende una mayor calidad y confiabilidad (6).

Métrica de la complejidad ciclomática

La complejidad ciclomática es una métrica de software extremadamente útil, pues proporciona una medición cuantitativa de la complejidad lógica de un programa. El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y nos da un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez. Esta prueba se conoce como Prueba del Camino Básico.

Un camino independiente es cualquier camino del programa que introduce por lo menos un nuevo conjunto de sentencias de procesamiento o una nueva condición. El camino independiente se debe mover por lo menos por una arista que no haya sido recorrida anteriormente.

Se le realizó el cálculo de la complejidad ciclomática en todos los bloques de código dentro del sistema, la cual ayudó a reflejar una medida de la complejidad del código escrito, teniendo en cuenta el número de destinos posibles. Además ayudó a conocer el esfuerzo a realizar en cada una de las pruebas, que exactamente el valor de la complejidad ciclomática en cada elemento o módulo. A partir de esta medida, se diseñaron pruebas que forzaron el recorrido de estos caminos, lo cual garantiza que se ejecute al

menos una vez cada sentencia del programa y que cada condición se ejecute en sus variantes verdadera y falsa.

Ejemplos de algunos de los segmentos de código analizados.

```
public void adicionarReunion() {
    this.mostrarErrorAsistencia = false;
    this.mostrarErrorFecha = false;
    if (this.cantidadMiembros < this.asistencia) {
        this.mostrarErrorAsistencia = true;
        FacesContext.getCurrentInstance().addMessage(null, new FacesMessage(
                "La asistencia no puede ser mayor que la cantidad de miembros."));
    } else if (pfecha.before(fecha)) {
        this.mostrarErrorFecha = true;
        FacesContext.getCurrentInstance().addMessage(null, new FacesMessage(
                                "La fecha de la próxima reunión no puede ser anterior a la reunión actual."));
    } else {
        UsuarioSessionBean user = (UsuarioSessionBean) FacesUtils.getManagedBean(
                BeanNames.USUARIO_SESSION_BEAN);
        Integer id = user.getEstructura().getIdEstructura();
        Estructura est = user.getServiceLocator().getServicioSeguridad().obtenerEstructura(id);
        for (int i = 0; i < marcados.length; i++) {
            Integer a = new Integer(marcados[i]);
            invitadoSet.add(this.serviceLocator.getServiciosFuncionamiento().obtenerTipoInvitado(a));
        }
        if (tipoReunion.intValue() == 1) {
            OrdenDelDia orden = new OrdenDelDia();
            analisisAsistencia = this.obtenerOrdenDia(0);
            orden.setAnalisisAsistencia(analisisAsistencia);
            acta = this.obtenerOrdenDia(1);
            orden.setActa(acta);
            analisisFuncionamiento = this.obtenerOrdenDia(2);
            orden.setAnalisisDeFuncionamiento(analisisFuncionamiento);
            chequeoAcuerdos = this.obtenerOrdenDia(3);
            orden.setChequeoAcuerdos(chequeoAcuerdos);
            debatePolitico = this.obtenerOrdenDia(4);
            orden.setDebatePolitico(debatePolitico);
            respuestaPlanteamientosEstudiantes = this.obtenerOrdenDia(5);
            orden.setRespuestaPlanteamientosEstudiantiles(respuestaPlanteamientosEstudiantes);
            temaDeInteres = this.obtenerOrdenDia(6);
            orden.setTemaDeInteres(temaDeInteres);
            orden.setTotalDeAcuerdos(this.totalAcuerdos);
            this.serviceLocator.getServiciosFuncionamiento().crearOrdenDelDia(orden);
            ReunionNormal normal = new ReunionNormal();
            normal.setAsistencia(this.asistencia);
            normal.setCantidadMiembros(this.cantidadMiembros);
```

Figura 16. Código para guardar una reunión 1.

```
            if (mostrarBrig == false)
                normal.setEstructura(est);
            else {
                Estructura a = this.serviceLocator.getServicioSeguridad().obtenerEstructura(this.idBrigada);
                normal.setEstructura(a);
            }
            normal.setLugarActual(this.lugar);
            normal.setFechaActual(fecha);
            normal.setTipoInvitadoReunionSet(this.invitadoSet);
            normal.setOrdenDelDia(orden);
            normal.setLugarProximo(this.plugar);
            normal.setFechaProxima(pfecha);
            this.serviceLocator.getServiciosFuncionamiento().crearReunionNormal(normal);
            FacesUtils.resetManagedBean(BeanNames.NUEVA_REUNION_BEAN);
            FacesContext.getCurrentInstance().addMessage(null, new FacesMessage(
                                    "La reunión ha sido guardada satisfactoriamente."));
        } else if (tipoReunion.intValue() == 2) {
            ReunionPPolitica politica = new ReunionPPolitica();
            politica.setAsistencia(this.asistencia);
            politica.setCantidadMiembros(this.cantidadMiembros);
            if (mostrarBrig == false)
                politica.setEstructura(est);
            else {
                Estructura a = this.serviceLocator.getServicioSeguridad().obtenerEstructura(this.idBrigada);
                politica.setEstructura(a);
            }
            politica.setLugarActual(this.lugar);
            politica.setFechaActual(fecha);
            politica.setTipoInvitadoReunionSet(this.invitadoSet);
            politica.setTemaAbordado(this.temaAbordado);
            politica.setLugarProximo(this.plugar);
            politica.setFechaProxima(pfecha);
            this.serviceLocator.getServiciosFuncionamiento().crearReunionPPolitca(politica);
            FacesUtils.resetManagedBean(BeanNames.NUEVA_REUNION_BEAN);
            FacesContext.getCurrentInstance().addMessage(null, new FacesMessage(
                                    "La reunión ha sido guardada satisfactoriamente."));
        }
    }
}
```

Figura 17. Código para guardar una reunión 2.

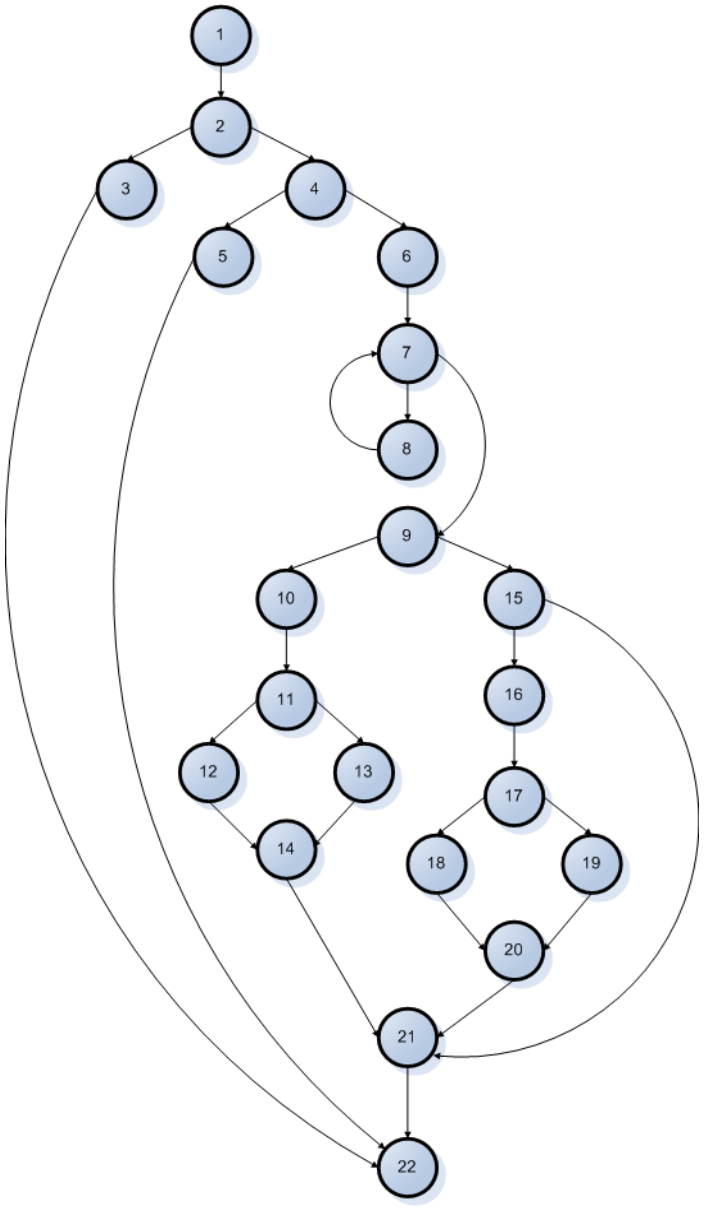


Figura 18. Grafo de flujo del código para guardar una reunión.

Complejidad Ciclomática V (G) para la Figura 18

V (G) = A - N + 2

V (G) = *28 - 22 + 2*

V (G) = 8

```
public void adicionarCaracterizacionEstadistica() {
    if (comprobarCI() == false){
        FacesContext.getCurrentInstance().addMessage(null, new FacesMessage("Número CI no válido"));
    }
    else if (!this.comprobarTelef()){
        FacesContext.getCurrentInstance().addMessage(null, new FacesMessage("Número teléfono no válido"));
    }else
    {
    CaracterizacionEstadistica caracterizacionEstadistica = new CaracterizacionEstadistica();
    caracterizacionEstadistica.setTipoCaracterizacion(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerTipoCaracterizacion(idtipocaract));
    Date x = new Date();
    caracterizacionEstadistica.setFechaRegistro(x);
    caracterizacionEstadistica.setNombre(this.nombre);
    caracterizacionEstadistica.setApellidos(this.apellidos);
    caracterizacionEstadistica.setCarnetIdentidad(this.carnetIdentidad);
    caracterizacionEstadistica.setAnno(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerAnno(this.idAnno));
    caracterizacionEstadistica.setEstadoCivil(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerEstadoCivil(this.idEstadoCivil));
    caracterizacionEstadistica.setCarrera(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerCarrera(idCarreras));
    caracterizacionEstadistica.setNacionalidad(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerNacionalidad(idnacionalidad));
    caracterizacionEstadistica.setSexo(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerSexo(idsexo));
    if (this.muestratg == false) {
        caracterizacionEstadistica.setTiempoEmbarazo(this.serviceLocator.
                getServiciosCaracterizacionEstadistica().obtenerTiempoEmbarazo(this.idTiempoEmbarazo));
    }
    if (this.tieneHijo) {
        caracterizacionEstadistica.setCantidadDeHijos(this.serviceLocator.
                getServiciosCaracterizacionEstadistica().obtenerCantidadHijos(this.idCantidadHijos));
    }
    caracterizacionEstadistica.setTieneComputadora(this.tieneComputadora);
    if (this.mostrarComp == false) {
        caracterizacionEstadistica.setFrecuencia(this.serviceLocator.
                getServiciosCaracterizacionEstadistica().obtenerFrecuencia(this.idFrecuencia));
        caracterizacionEstadistica.setIdentificacionMedio(this.serviceLocator.
                getServiciosCaracterizacionEstadistica().obtenerIdentificacionMedio(
                        this.idIdentificacionMedio));
    }
    for (int i = 0; i < mseleccionados.length; i++) {
        Integer id = new Integer(mseleccionados[i]);
        militanciaSet.add(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerTipoMilitancia(id));
    }
    caracterizacionEstadistica.setMilitanciaSet(this.militanciaSet);
    for (int i = 0; i < vseleccionados.length; i++) {
        Integer id = new Integer(vseleccionados[i]);
        viveConSet.add(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerTipoFamiliarVive(id));
    }
    caracterizacionEstadistica.setViveConSet(this.viveConSet);
    for (int j = 0; j < seleccionados.length; j++) {
        Integer id1 = new Integer(seleccionados[j]);
        familiaFallecidoSet.add(this.serviceLocator.getServiciosCaracterizacionEstadistica()
                .obtenerTipoFamiliarFallecido(id1));
    }
```

Figura 19. Código para crear una caracterización estadística 1.

```
    caracterizacionEstadistica.setFamiliaFallecidoSet(this.familiaFallecidoSet);
    caracterizacionEstadistica.setPadresDivorciados(this.padresDivorciados);
    if (tieneCargo) {
        caracterizacionEstadistica.setCargo(this.serviceLocator.
                getServiciosCaracterizacionEstadistica().obtenerCargo(idCargo));
    } else
        caracterizacionEstadistica.setCargo(null);
    caracterizacionEstadistica.setBaja(false);
    caracterizacionEstadistica.setColorDePiel(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerColorPiel(idcolordepiel));
    caracterizacionEstadistica.setProcedenciaSocial(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerProcedenciaSocial(idpsocial));
    caracterizacionEstadistica.setFormaIngreso(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().obtenerFormaIngreso(idingreso));
    Localizacion localizacion = new Localizacion();
    localizacion.setDireccion(this.dparticular);
    localizacion.setDireccionBeca(this.dbeca);
    localizacion.setTelefonos(this.telefono);
    localizacion.setProvincia(this.serviceLocator.getServiciosCaracterizacionEstadistica().
            obtenerProvincia(idprovincia));
    this.muestrarmunicipio = true;
    localizacion.setMunicipio(this.serviceLocator.getServiciosCaracterizacionEstadistica().
            obtenerMunicipio(idmunicipio));
    localizacion.setConsejoPopular(this.serviceLocator.getServiciosCaracterizacionEstadistica().
            obtenerConsejoPopular(idcpopulares));
    caracterizacionEstadistica.setLocalizacion(this.serviceLocator.
            getServiciosCaracterizacionEstadistica().guardarLocalizacion(localizacion));
    caracterizacionEstadistica = this.serviceLocator.getServiciosCaracterizacionEstadistica().
            guardarCaracterizacion(caracterizacionEstadistica);
    Estructura est = this.serviceLocator.getServicioSeguridad().obtenerEstructura(identradabrigada);
    BrigadaEstudiante brigadaEstudiante = new BrigadaEstudiante();
    brigadaEstudiante.setEstructura(est);
    brigadaEstudiante.setCaracterizacionEstadistica(caracterizacionEstadistica);
    brigadaEstudiante.setActivo(true);
    this.serviceLocator.getServiciosCaracterizacionEstadistica().guardarBrigadaEstudiante(brigadaEstudiante);
    FacesUtils.resetManagedBean(BeanNames.CARACTERIZACION_ESTADISTICA_BEAN);
    FacesContext.getCurrentInstance().addMessage(null,
            new FacesMessage( "La caracterización ha sido guardada satisfactoriamente."));
    }
}
```

Figura 20. Código para crear una caracterización estadística 2.

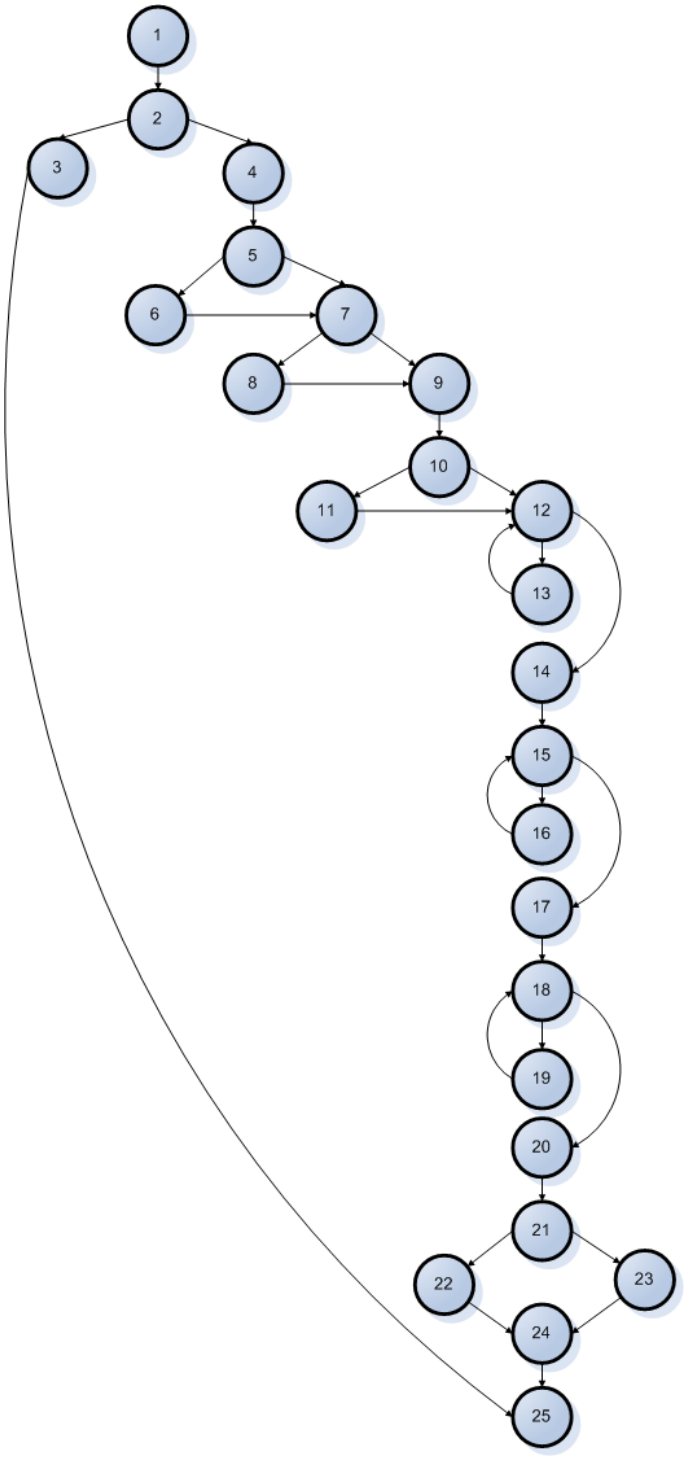


Figura 21. Grafo de flujo para el código de crear una caracterización estadística.

Complejidad Ciclomática V (G) para la Figura 21

V (G) = A - N + 2

V (G) = *32 - 25 + 2*

V (G) = 9

```
public void modificarCaracterizacionEstadistica() {
    if (comprobarCI() == false) {
        FacesContext.getCurrentInstance().addMessage(null,new FacesMessage("Número CI no valido"));
    } else if (!this.comprobarTelef()) {
        FacesContext.getCurrentInstance().addMessage(null,new FacesMessage("Número telefono no valido"));
    } else {
        CaracterizacionEstadistica caracterizacion = this.serviceLocator.
                getServiciosCaracterizacionEstadistica().obtenerCaracterizacion(this.idCaracterizacion);
        caracterizacion.setTipoCaracterizacion(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerTipoCaracterizacion(this.idTipoCaracterizacion));
        caracterizacion.setNombre(this.nombre);
        caracterizacion.setApellidos(this.apellidos);
        caracterizacion.setCarnetIdentidad(this.noIdentidad);
        caracterizacion.setCarrera(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerCarrera(this.idCarrera));
        caracterizacion.setAnno(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerAnno(this.idAnno));
        caracterizacion.setNacionalidad(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerNacionalidad(this.idPais));
        caracterizacion.setSexo(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerSexo(this.idSexo));
        if (this.estaEmbarazada) {
            caracterizacion.setTiempoEmbarazo(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                    obtenerTiempoEmbarazo(this.idTiempoEmbarazo));
        } else {
            caracterizacion.setTiempoEmbarazo(null);
        }
        caracterizacion.setColorDePiel(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerColorPiel(this.idColorPiel));
        caracterizacion.setEstadoCivil(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerEstadoCivil(this.idEstadoCivil));
        caracterizacion.setProcedenciaSocial(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerProcedenciaSocial(this.idProcedenciaSocial));
        Set militanciaSet = new HashSet();
        for (int i = 0; i < militanciaSeleccionada.length; i++) {
            Integer id = new Integer(militanciaSeleccionada[i]);
            militanciaSet.add(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                    obtenerTipoMilitancia(id));
        }
```

Figura 22. Código para modificar una caracterización estadística 1.

```
        caracterizacion.setMilitanciaSet(militanciaSet);
        if (this.tieneHijos)
            caracterizacion.setCantidadDeHijos(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                    obtenerCantidadHijos(this.idCantidadHijos));
        else
            caracterizacion.setCantidadDeHijos(null);
        caracterizacion.setFormaIngreso(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerFormaIngreso(this.idFormaIngreso));
        Set viveConSet = new HashSet();
        for (int i = 0; i < familiaresVivoSeleccionados.length; i++) {
            Integer id = new Integer(familiaresVivoSeleccionados[i]);
            viveConSet.add(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                    obtenerTipoFamiliarVive(id));
        }
        caracterizacion.setViveConSet(viveConSet);
        Set fallecidosSet = new HashSet();

        for (int i = 0; i < familiaresFallecidosSeleccionados.length; i++) {
            Integer id = new Integer(familiaresFallecidosSeleccionados[i]);
            fallecidosSet.add(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                    obtenerTipoFamiliarFallecido(id));
        }
        caracterizacion.setFamiliaFallecidoSet(fallecidosSet);
        caracterizacion.setPadresDivorciados(this.tienePadresDivorciados);
        if (this.tieneResponsabilidad)
            caracterizacion.setCargo(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                    obtenerCargo(this.idCargo));
        else
            caracterizacion.setCargo(null);
        Localizacion localizacion = this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerLocalizacion(caracterizacion.getLocalizacion().getIdLocalizacion());
        localizacion.setProvincia(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerProvincia(this.idProvincia));
        localizacion.setMunicipio(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerMunicipio(this.idMunicipio));
        localizacion.setConsejoPopular(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerConsejoPopular(this.idConsejoPopular));
        localizacion.setDireccion(this.direccionParticular);
        localizacion.setDireccionBeca(this.direccionBeca);
        localizacion.setTelefonos(this.telefono);
        this.serviceLocator.getServiciosCaracterizacionEstadistica().actualizarLocalizacion(localizacion);
        if (this.tieneAcceso) {
            caracterizacion.setTieneComputadora(this.tieneAcceso);
            caracterizacion.setFrecuencia(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                    obtenerFrecuencia(this.idFrecuencia));
            caracterizacion.setIdentificacionMedio(this.serviceLocator.
                    getServiciosCaracterizacionEstadistica().obtenerIdentificacionMedio(this.idIdentificacion));
        } else {
            caracterizacion.setTieneComputadora(this.tieneAcceso);
            caracterizacion.setFrecuencia(null);
            caracterizacion.setIdentificacionMedio(null);
        }
        this.serviceLocator.getServiciosCaracterizacionEstadistica().actualizarCaracterizacion(caracterizacion);
        FacesUtils.resetManagedBean(BeanNames.MODIFICAR_CARACTERIZACION_ESTADISTICA_BEAN);
        FacesUtils.resetManagedBean(BeanNames.LIST_CARACTERIZACION_ESTADISTICA_BEAN);
        FacesContext.getCurrentInstance().addMessage(null,new FacesMessage(
                                "La caracterización ha sido guardada satisfactoriamente."));
    }
}
```

Figura 23. Código para modificar una caracterización estadística 2.

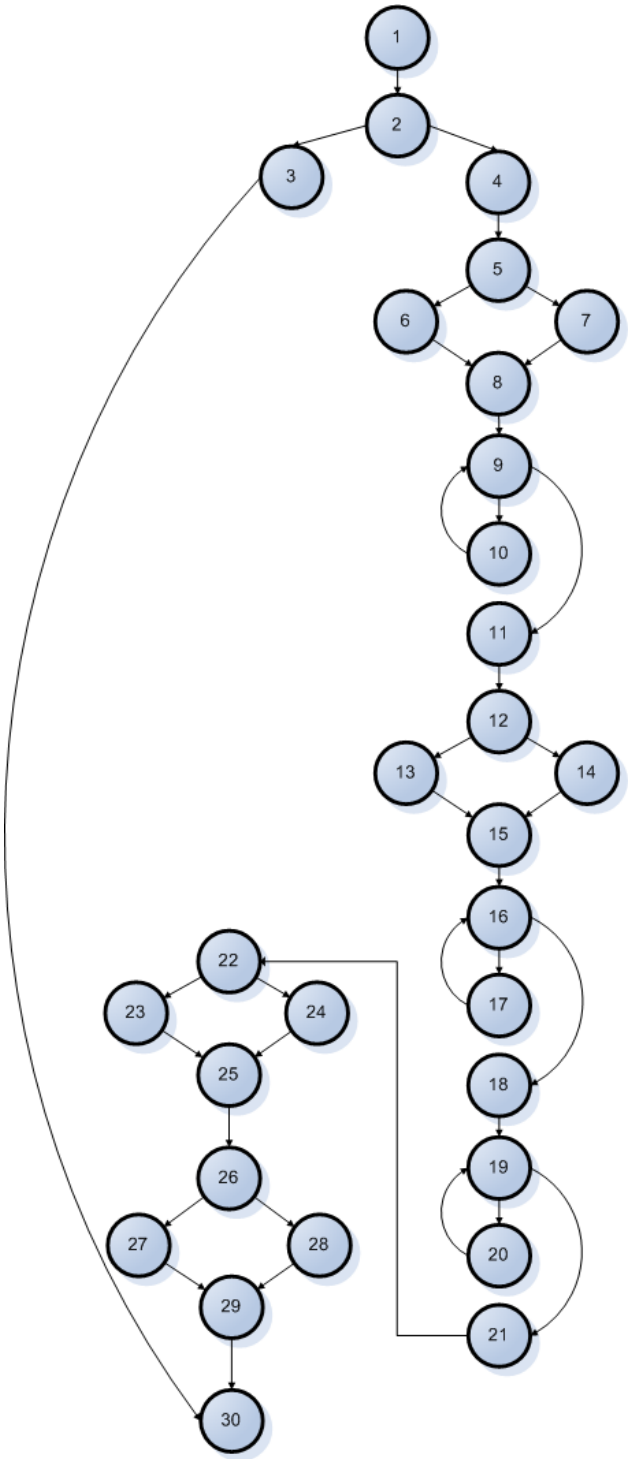


Figura 24. Grafo de flujo del código para modificar una caracterización estadística.

Complejidad Ciclomática V (G) para la Figura 24

V (G) = A - N + 2

V (G) = *37 - 30 + 2*

V (G) = 9

```
public void darBaja() {
    Baja baja = new Baja();
    baja.setCaracterizacionEstadistica(this.serviceLocator.getServiciosCaracterizacionEstadistica()
            .obtenerCaracterizacion(this.idCaracterizacion));
    Motivo motivo = this.serviceLocator.getServiciosCaracterizacionEstadistica().obtenerMotivo(this.idMotivo);
    baja.setMotivo(motivo);
    baja.setPropuesta(false);
    baja = this.serviceLocator.getServiciosCaracterizacionEstadistica().guardarBaja(baja);
    CaracterizacionEstadistica c = this.serviceLocator.getServiciosCaracterizacionEstadistica()
            .obtenerCaracterizacion(this.idCaracterizacion);
    Iterator ite = c.getBrigadas().iterator();
    while (ite.hasNext()) {
        BrigadaEstudiante brigadaEstudiante = (BrigadaEstudiante) ite.next();
        if (brigadaEstudiante.getActivo().booleanValue() == true) {
            brigadaEstudiante.setActivo(false);
            this.getServiceLocator().getServiciosCaracterizacionEstadistica().
                    actualizarBrigadaEstudiante(brigadaEstudiante);
        }
    }
    if (this.mostrarUbicacion == true) {
        UbicacionLaboral ubicacionLaboral = new UbicacionLaboral();
        ubicacionLaboral.setBaja(baja.getIdBaja());
        Provincia provincia = this.serviceLocator.getServiciosCaracterizacionEstadistica().
            obtenerProvincia(this.idprovincia);
        ubicacionLaboral.setProvincia(provincia);
        ubicacionLaboral.setMunicipio(this.serviceLocator.getServiciosCaracterizacionEstadistica().
                obtenerMunicipio(this.idmunicipio));
        ubicacionLaboral.setNombreEntidad(this.nombreEntidad);
        ubicacionLaboral.setNombrePlaza(this.plaza);
        this.serviceLocator.getServiciosCaracterizacionEstadistica().guardarUbicacionLaboral(ubicacionLaboral);
    }
    FacesUtils.resetManagedBean(BeanNames.BAJA_BEAN);
    FacesContext.getCurrentInstance().addMessage(null, new FacesMessage("Baja satisfactoria"));
}
```

Figura 25. Código para dar baja.

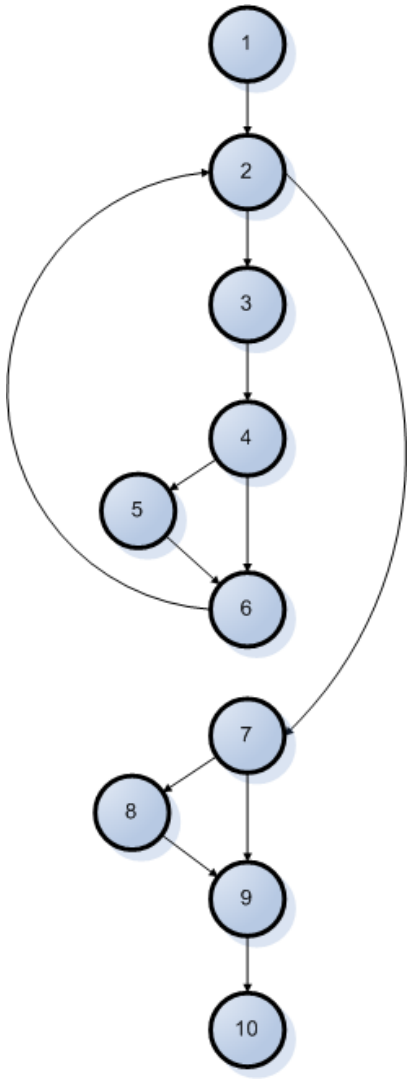


Figura 26. Grafo de flujo para el código de dar baja.

Complejidad Ciclomática V (G) para le Figura 26

V (G) = A - N + 2

V (G) = *12 - 10 + 2*

V (G) = 4

Tabla 2. Posiciones de valores.

Valor de V(G)	Evaluación
1-10	Un programa simple sin mucho riesgo
11-20	Medianamente complejo
21-50	Un programa complejo
50+	Programa inestable

A partir del análisis del cálculo de la complejidad ciclomática en la propuesta de solución, el cual se encuentra entre los valores 1-10, y del resultado de cada una de las pruebas, se llega a la conclusión de que de forma general los resultados obtenidos fueron positivos teniendo en cuenta que el sistema presenta un bajo riesgo de fallos.

3.3.2 Pruebas de Caja Negra

Objetivos

El objetivo de realizar este tipo de prueba al sistema es para revelar el incorrecto o incompleto funcionamiento de este, así como los errores de interfaz, rendimiento y errores de inicialización y terminación.

Alcance

El proceso de pruebas de caja negra se va a centrar principalmente en los requisitos funcionales del software para verificar el comportamiento de la unidad observable externamente y la calidad funcional.

Descripción

Se llevan a cabo sobre la interfaz del software, y es completamente indiferente el comportamiento interno y la estructura del programa. Los casos de prueba de la caja negra pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma adecuada.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

La prueba de la caja negra intenta encontrar errores de las siguientes categorías:

- Funciones incorrectas o ausentes.

- Errores de interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales del programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos.

Dentro de las técnicas de Prueba de Caja Negra se utilizaron la Partición de Equivalencia y el Análisis de Valores Límite (AVL).

Pressman (7), presenta la partición equivalente como un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.

Por su parte, el Análisis de Valores Límite es una técnica de diseño de casos de prueba que completa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el AVL lleva a la elección de casos de prueba en los extremos de la clase. En lugar de centrarse solamente en las condiciones de entrada, el AVL obtiene casos de prueba también para el campo de salida. (7)

Descripción de los casos de prueba

Se definieron los casos de pruebas para los casos de uso Gestionar Caracterización Estadística, Gestionar Estructura, Gestionar Reunión y Buscar Caracterización Estadística.

Se debe tener en cuenta a la hora de utilizar los casos de prueba que no a todos los niveles de usuario (nacional, provincial, municipal, etcétera), se pueden realizar todas las funcionalidades. A continuación se listan los usuarios que se utilizan para las pruebas y los casos que pueden probar.

En algunos, según el nivel del usuario serán los campos se tendrán que llenar a la hora de realizar alguna actividad.

Tabla 3. Usuarios y contraseñas para las pruebas.

Usuario	Contraseña	Descripción	Casos que prueba
nacional	nacional	Usuario nacional.	1. Buscar Caracterización Estadística. 2. Gestionar Estructura.
provinciapr	provinciapr	Usuario de la provincia Pinar del Río (PR).	1. Buscar Caracterización Estadística. 2. Gestionar Estructura. 3. Gestionar Reunión.
Cespr	cespr	Usuario de un Centro de Enseñanza Superior de PR.	1. Buscar Caracterización Estadística. 2. Gestionar Estructura. 3. Gestionar Reunión.
munpr	munpr	Usuario de un Municipio de PR.	1. Buscar Caracterización Estadística. 2. Gestionar Estructura. 3. Gestionar Reunión.
facpr	facpr	Usuario de una facultad de PR	1. Buscar Caracterización Estadística. 2. Gestionar Estructura. 3. Gestionar Reunión. 4. Gestionar Caracterización Estadística.
sedepr	sedepr	Usuario de una sede universitaria de PR.	1. Buscar Caracterización Estadística. 2. Gestionar Estructura. 3. Gestionar Reunión. 4. Gestionar Caracterización Estadística.

Leyenda para la realización de las pruebas:

N/A = no aparece ese campo en ese caso o no está habilitado.

- = no se llenó el campo en ese caso.

-seleccione- = no se escogió ninguna opción de las que brinda el componente.

Nombre del Caso:

Gestionar Estructura

Descripción General:

Este caso de uso le permite al usuario, adicionar y modificar una estructura en el sistema con su respectivo tipo.

Condiciones de Ejecución:

Que el usuario esté autenticado y tenga permisos.

Tabla 4. Secciones a probar en el Caso de Uso Gestionar Estructura.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC1: Adicionar Estructura.	EC 1.1: Adicionar Estructura.	El actor solicita adicionar una estructura y el sistema le provee una interfaz para hacerlo.	1. El actor entra en la opción nueva estructura. 2. El sistema muestra un formulario para adicionar estructura con todos los campos necesarios a llenar para ingresar la nueva estructura. 3. El actor introduce los datos y pulsa el botón Adicionar. 4. El sistema verifica que los campos se hayan llenado y adiciona la nueva estructura mostrando un mensaje que indica que se realizó la acción satisfactoriamente.
	EC 1.2: Datos Obligatorios.	El actor no introduce todos los datos obligatorios y el sistema muestra un mensaje.	A partir de la acción 3. 1. El sistema informa que se dejó algún campo obligatorio sin llenar. 2. El actor llena los campos obligatorios que faltaban y da clic en el botón Adicionar. 3. El sistema verifica que los campos se hayan llenado y crea la nueva estructura mostrando un mensaje que indica que se adicionó satisfactoriamente.
	EC 1.3: Datos Incorrectos.	El actor introduce datos incorrectos y el sistema señala estos datos.	A partir de la acción 3. 1. El sistema verifica que existen datos incorrectos. 2. El sistema señala los datos incorrectos. Ir a la acción 3 del EC 1.1.
SC 2: Modificar estructura.	EC 2.1: Modificar estructura.	El actor selecciona Modificar Estructura y el sistema muestra la interfaz con los campos correspondientes.	1. El actor selecciona la opción modificar estructura. 2. El sistema muestra el formulario para modificar una estructura existente con todos los campos necesarios para actualizar la estructura. 3. El actor selecciona la estructura que desee modificar, cambia los datos que desea, que estarán en dependencia del nivel que tenga el usuario y pulsa el botón de guardar cambios. 4. El sistema verifica que los campos se hayan llenado y actualiza la estructura mostrando un mensaje que indica que se modificó la estructura satisfactoriamente.
	EC 2.2: Datos Obligatorios.	El actor no introduce todos los datos obligatorios y el sistema muestra un mensaje de error.	A partir de la acción 3 del EC 2.1. 1. En caso de no haber llenado uno de los campos obligatorios el sistema mostrará un mensaje de error al lado de dicho campo. 2. El actor llena el campo obligatorio de la estructura y pulsa la opción de guardar cambios. 3. El sistema modifica la estructura en la base de datos y muestra un mensaje de que se modificó la estructura satisfactoriamente.
	EC 2.3: Datos Incorrectos.	El actor introduce datos incorrectos y el sistema señala estos datos.	A partir de la acción 3 del EC 2.1. 1. El sistema verifica que existen datos incorrectos. 2. El sistema señala los datos incorrectos. Ir a la acción 3 del EC 2.1.

Tabla 5. SC 1 Adicionar Estructura.

Id	EC	Nombre	Tipo	Curso	Respuesta del Sistema	Resultado esperado
EC 1.1	Adicionar Estructura.	Nota: para probar estos datos debe estar autenticado un usuario de nivel nacional.				
		Camagüey	Consejo Provincial	N/A	El sistema muestra un mensaje indicando que se adicionó correctamente la estructura.	Satisfactorio.
		UCI	Centro de Educación Superior	N/A	El sistema muestra un mensaje indicando que se adicionó correctamente la estructura.	Satisfactorio.
		Nota: para probar estos datos debe estar autenticado un usuario de nivel de facultad o sede universitaria.				
		Brigada 1504	Brigada de Facultad	2006/2007	El sistema muestra un mensaje indicando que se adicionó correctamente la estructura.	Satisfactorio.
		Brigada San Antonio	Brigada de Sede Universitaria	2006/2007	El sistema muestra un mensaje indicando que se adicionó correctamente la estructura.	Satisfactorio.
		Nota: para probar estos datos debe estar autenticado un usuario de nivel nacional.				
EC 1.2	Datos Obligatorios.	-	-seleccione-	N/A	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.
		UCI	-seleccione-	N/A	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.
		Nota: para probar estos datos debe estar autenticado un usuario de nivel de facultad o sede universitaria.				
		-	Brigada de Facultad	-seleccione-	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.
		Brigada 1506	Brigada de Sede Universitaria	-seleccione-	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.
EC 1.2	Datos Incorrectos.	Nota: para probar estos datos debe estar autenticado un usuario de nivel de facultad o sede universitaria.				
		/*`/%	Brigada de Sede Universitaria	2006/2007	El sistema muestra un mensaje indicando que hay datos incorrectos.	Satisfactorio.

Tabla 6. SC 2 Modificar Estructura.

Id	EC	Elegir Brigada	Nombre	Activa	Curso	Respuesta del Sistema	Resultado esperado
EC 2.1	Modificar estructura.	Nota: para probar estos datos debe estar autenticado un usuario de nivel nacional.					
		Camagüey	Camagüey	Si	N/A	El sistema muestra un mensaje indicando que se modificó correctamente la estructura.	Satisfactorio.
		Nota: para probar estos datos debe estar autenticado un usuario de nivel de facultad o sede universitaria.					
		Brigada 1504-4	Brigada 1504-5	Si	20/07/2008	El sistema muestra un mensaje indicando que se modificó correctamente la estructura.	Satisfactorio.
EC 2.2	Datos Obligatorios.	Nota: para probar estos datos debe estar autenticado un usuario de nivel nacional.					
		Camagüey	-	No	N/A	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.
		Nota: para probar estos datos debe estar autenticado un usuario de nivel de facultad o sede universitaria.					
		Brigada 1504	-	No	-seleccione-	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.
EC 2.3	Datos Incorrectos.	Nota: para probar estos datos debe estar autenticado un usuario de nivel de facultad o sede universitaria.					
		Brigada 1504	¡$%&//´´	Si	20/07/2008	El sistema muestra un mensaje indicando que hay datos incorrectos.	Satisfactorio.

Nombre del Caso:

Gestionar Reunión.

Descripción General:

Permite al usuario crear una reunión, ya sea ordinaria o de preparación política.

Condiciones de Ejecución:

Que el usuario esté autenticado y tenga permisos.

Tabla 7. Secciones a probar en el Caso de Uso Gestionar Reunión.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Crear Reunión Ordinaria.	EC 1.1: Crear Reunión Ordinaria.	El actor solicita crear una reunión y el sistema le provee una interfaz para hacerlo.	1. El actor selecciona la opción para crear una nueva reunión. 2. El sistema muestra el formulario para crear una reunión con todos los campos necesarios a la hora de llenar la planilla de reuniones. 3. Según el nivel del usuario serán los campos que tendrá que llenar este para confeccionar la planilla de la reunión. En este caso deberá seleccionar previamente el campo Reunión Ordinaria. Después de llenado los campos obligatorios se pulsa en el botón Adicionar. 4. El sistema verifica que los campos se hayan llenado y crea la reunión mostrando un mensaje que indica que se creó una nueva reunión satisfactoriamente.
	EC 1.2: Datos Obligatorios.	El actor no introduce todos los datos obligatorios y el sistema muestra un mensaje.	A partir de la acción 3. 1. El sistema informa que se dejó algún campo obligatorio sin llenar. 2. El actor llena los campos obligatorios que faltaban y da clic en el botón Adicionar. 3. El sistema verifica que los campos se hayan llenado y crea la nueva reunión mostrando un mensaje que indica que se creó la reunión satisfactoriamente.
	EC 1.3: Datos Incorrectos.	El actor introduce datos incorrectos y el sistema señala estos datos.	A partir de la acción 3. 1. El sistema verifica que existen datos incorrectos. 2. El sistema señala los datos incorrectos. Ir a la acción 3 del EC 1.1.
SC 2: Crear Reunión de Preparación Política.	EC 2.1: Crear Reunión de Preparación Política.	El actor selecciona Crear Reunión y el sistema muestra la interfaz con los campos correspondientes.	1. El actor selecciona la opción para crear una nueva reunión. 2. El sistema muestra la interfaz para crear una reunión con todos los campos necesarios a llenar a la hora de llenar la planilla de reuniones. 3. Según el nivel del usuario serán los campos que tendrá que llenar este para confeccionar la planilla de la reunión. En este caso deberá seleccionar previamente el campo

			Reunión de Preparación Política. Después de llenado los campos obligatorios se pulsa en el botón Adicionar. 4. El sistema verifica que los campos se hayan llenado y crea la reunión mostrando un mensaje que indica que se creo una nueva reunión satisfactoriamente.
	EC 2.2: Datos Obligatorios.	El actor no introduce todos los datos obligatorios y el sistema muestra un mensaje de error.	A partir de la acción 3 del EC 2.1. 1. En caso de no haber llenado uno de los campos obligatorios el sistema mostrará un mensaje de error al lado de dicho campo. 2. El actor llena el campo obligatorio de la nueva reunión y pulsa la opción guardar. 3. El sistema almacena la reunión creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.
	EC 2.3: Datos Incorrectos.	El actor introduce datos incorrectos y el sistema señala estos datos.	A partir de la acción 3 del EC 2.1. 1. El sistema verifica que existen datos incorrectos. 2. El sistema señala los datos incorrectos. Ir a la acción 3 del EC 2.1.

Nota: En la Tabla 3.7 existen dos respuestas del sistema y dos resultados esperados pues corresponden a los valores 1 y 2 de los datos de entrada.

Tabla 8. SC 1 Crear Reunión Ordinaria.

Id	EC	Datos de entrada	Valores 1	Valores 2	Respuesta del Sistema 1	Resultado esperado 1	Respuesta del Sistema 2	Resultado esperado 2
EC 1.1	Crear Reunión Ordinaria.	**Estructura**	Facultad o Sede	Brigada	El sistema almacena la reunión en la BD y muestra un mensaje informando que se adicionó correctamente.	Satisfactorio.	El sistema almacena la reunión en la BD y muestra un mensaje informando que se adicionó correctamente.	Satisfactorio.
		Brigada	N/A	Brigada Sede Mun PR				
		Lugar	Salón Reuniones UJC	Aula 15 B				
		Fecha y Hora	05/23/2008 18:30	05/15/2008 17:00				
		Cantidad de Miembros	27	23				
		Asistencia	25	22				
		Tipo de Reunión	Reunión Ordinaria	Reunión Ordinaria				
		Tipo de	Consejo	-				

		Invitado	Provincial, PCC, Otros					
		Orden del Día	Acta, Debate Político	-				
		Total de Acuerdos	2	3				
		Lugar Próxima Reunión	Salón Reuniones UJC	Aula 15 B				
		Fecha y Hora Próxima Reunión	06/24/2008 18:30	06/17/2008 19:00				
EC 1.2	Datos Obligatorios	**Estructura**	Facultad o Sede	Brigada	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.
		Brigada	N/A	-seleccione-				
		Lugar	-	-				
		Fecha y Hora	-	-				
		Cantidad de Miembros	-	32				
		Asistencia	-	-				
		Tipo de Reunión	Reunión Ordinaria	Reunión Ordinaria				
		Tipo de Invitado	-	PCC				
		Orden del Día	-	-				
		Total de Acuerdos	1	-				
		Lugar Próxima Reunión	-	Aula 15 B				
		Fecha y Hora Próxima	-	06/17/2008 19:00				

		Reunión						
EC 1.3	Datos Incorrectos	**Estructura**	Brigada	Facultad o Sede	El sistema muestra un mensaje indicando que hay datos incorrectos.	Satisfactorio.	El sistema muestra un mensaje indicando que hay datos incorrectos.	Satisfactorio.
		Brigada	Brigada Sede Mun PR	N/A				
		Lugar	Aula 15 B	Aula 15 B				
		Fecha y Hora	05/15/2008 17:00	05/15/2008 17:00				
		Cantidad de Miembros	2	Gato				
		Asistencia	Tres	3				
		Tipo de Reunión	Reunión Ordinaria	Reunión Ordinaria				
		Tipo de Invitado	UJC	-				
		Orden del Día	Acta	-				
		Total de Acuerdos	pepe	3				
		Lugar Próxima Reunión	Salón Reuniones UJC	Salón Reuniones UJC				
		Fecha y Hora Próxima Reunión	05/15/2008 17:00	06/17/2008 19:00				
Nota: Hacer diferentes combinaciones entre los atributos para EC 1.1, 1.2 y 1.3								

Nota: En la Tabla 9 existen dos respuestas del sistema y dos resultados esperados pues corresponden a los valores 1 y 2 de los datos de entrada.

Tabla 9. SC 2 Crear Reunión de Preparación Política.

Id	EC	Datos de entrada	Valores 1	Valores 2	Respuesta del Sistema 1	Resultado esperado 1	Respuesta del Sistema 2	Resultado esperado 2
EC 2.1	Crear Reunión de Preparación Política.	**Estructura**	Facultad o Sede	Brigada	El sistema almacena la reunión en la BD y muestra un mensaje informando que se adicionó correctamente.	Satisfactorio.	El sistema almacena la reunión en la BD y muestra un mensaje informando que se adicionó correctamente.	Satisfactorio.
		Brigada	-	Brigada Sede Mun PR				
		Lugar	Salón Reuniones UJC	Aula 15 B				
		Fecha y Hora	05/23/2008 18:30	05/15/2008 17:00				
		Cantidad de Miembros	28	20				
		Asistencia	23	15				
		Tipo de Reunión	Preparación Política	Preparación Política				
		Tipo de Invitado	PCC, Otros.	-				
		Tema abordado	Discurso de Raúl 26-Julio	Reflexiones de Fidel.				
		Lugar Próxima Reunión	Salón Reuniones UJC	Aula 15 B				
		Fecha y Hora Próxima Reunión	06/24/2008 18:30	06/17/2008 19:00				
EC 2.2	Datos Obligatorios.	**Estructura**	Facultad o Sede	Brigada	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.	El sistema muestra un mensaje indicando que faltan datos obligatorios por llenar.	Satisfactorio.
		Brigada	N/A	- seleccione -				
		Lugar	-	-				
		Fecha y	06/17/20	06/17/200				

		Hora	08 18:00	8 18:30				
		Cantidad de Miembros	-	32				
		Asistencia	-	-				
		Tipo de Reunión	Preparación Política	Preparación Política				
		Tipo de Invitado	-	PCC				
		Tema abordado	-	-				
		Lugar Próxima Reunión	-	Aula 15 B				
		Fecha y Hora Próxima Reunión	07/17/2008 19:00	07/17/2008 19:00				
EC 2.3	Datos Incorrectos.	**Estructura**	Brigada	Facultad o Sede	El sistema muestra un mensaje indicando que hay datos incorrectos.	Satisfactorio.	El sistema muestra un mensaje indicando que hay datos incorrectos.	Satisfactorio.
		Brigada	Brigada Sede Mun PR	N/A				
		Lugar	Aula 15 B	Aula 15 B				
		Fecha y Hora	05/15/2008 17:00	05/15/2008 17:00				
		Cantidad de Miembros	2	Gato				
		Asistencia	Tres	3				
		Tipo de Reunión	Preparación Política	Preparación Política				
		Tipo de Invitado	UJC	-				
		Tema abordado	Discurso de Raúl 26-Julio	-				
		Lugar	Salón	Salón				

		Próxima Reunión	Reuniones UJC	Reuniones UJC				
		Fecha y Hora Próxima Reunión	05/15/2008 17:00	06/17/2008 19:00				
Nota: Hacer diferentes combinaciones entre los atributos para EC 2.1, 2.2 y 2.3								

Nombre del Caso:

Gestionar Caracterización Estadística

Descripción General

Este caso de uso le permite al usuario llenar una caracterización estadística, modificarla, hacer un traslado o darle de baja a la misma.

Condiciones de Ejecución:

El actor que desea crear la Caracterización Estadística debe tener permiso.

Nota: debe ser un usuario a nivel de facultad o sede para poder realizar estos casos.

Tabla 10. Secciones a probar en el Caso de Uso Gestionar Caracterización Estadística.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Crear Caracterización Estadística.	EC 1.1: Crear Caracterización Estadística.	El actor solicita llenar una caracterización estadística y el sistema le provee una interfaz para hacerlo.	1. El actor solicita llenar una nueva caracterización estadística. 2. El sistema visualiza el formulario en la cual se van a introducir los datos. 3. El actor introduce todos los datos obligatorios y pulsa la opción guardar. 4. El sistema almacena la planilla creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.
	EC 1.2: Datos Obligatorios.	El actor no introduce todos los datos obligatorios y el sistema muestra un mensaje.	A partir de la acción 2. 1. En caso de no haber llenado uno de los campos obligatorios el sistema mostrará un mensaje de error. 2. El actor llena el campo obligatorio de la nueva caracterización y pulsa la opción guardar. 3. El sistema almacena la caracterización estadística creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.

	EC 1.3: Datos Incorrectos.	El actor introduce datos incorrectos y el sistema señala estos datos.	A partir de la acción 3. 1. El sistema verifica que existen datos incorrectos. 2. El sistema señala los datos incorrectos. Ir a la acción 3 del EC 1.1.
SC 2: Modificar Caracterización Estadística.	EC 2.1: Modificar Caracterización Estadística.	El actor selecciona modificar una caracterización estadística y el sistema muestra las opciones de búsqueda. El actor escoge los datos y luego cambia los datos deseados de la caracterización buscada.	1. El caso de uso comienza cuando el actor selecciona la opción modificar caracterización estadística. 2. EL sistema muestra las opciones de búsqueda de caracterización. 3. El actor selecciona la brigada y el estudiante que desea buscarle la caracterización. 4. El sistema muestra la caracterización que cumple con los requisitos de la búsqueda. 5. El actor modifica los campos que desee y pulsa la opción Actualizar. 6. El sistema actualiza la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.
	EC 2.2: Datos Obligatorios.	El actor no introduce todos los datos obligatorios y el sistema muestra un mensaje de error.	A partir de la acción 3 del EC 2.1. 1. En caso de no haber llenado uno de los campos obligatorios el sistema mostrará un mensaje de error. 2. El actor llena el campo obligatorio de la nueva caracterización y pulsa la opción Actualizar. 3. El sistema actualiza la caracterización estadística creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.
	EC 2.3: Datos Incorrectos.	El actor introduce datos incorrectos y el sistema señala estos datos.	A partir de la acción 5. 1. El sistema verifica que existen datos incorrectos. 2. El sistema señala los datos incorrectos. Ir a la acción 5 del EC 2.1.
SC 3: Dar Baja Caracterización estadística.	EC 3.1: Dar Baja Caracterización estadística.	El actor pulsa la acción dar baja, el sistema muestra las opciones de búsqueda, el actor selecciona los datos y el sistema hace un conjunto de acciones, elimina la caracterización estadística y muestra un mensaje.	1. El caso de uso comienza cuando el actor pulsa la opción Dar de baja. 2. El sistema muestra las opciones de búsqueda. 3. El actor selecciona la brigada y la caracterización que desea darle de baja. 4. El sistema muestra el menú con los motivos para dar baja. 5. El actor selecciona el motivo y pulsa la opción Aceptar. 6. El sistema da baja a la caracterización estadística y muestra un mensaje de que se cumplió el proceso satisfactoriamente.
	EC 3.2: Motivo	El actor seleccionó	A partir de la acción 3 del EC 3.1.

	Seleccionado Graduación.	el motivo graduación, el actor introduce los datos, el sistema da baja a la caracterización y muestra un mensaje.	1. En caso que el motivo seleccionado sea graduación, se mostrarán los campos provincia, municipio, entidad y plaza a la que va a ser asignado el estudiante. 2. El actor llena los campos y pulsa el botón Aceptar. 3. El sistema da baja a la caracterización estadística y muestra un mensaje de que se cumplió el proceso satisfactoriamente.
	EC 3.3: Datos Incorrectos.	El actor introduce datos incorrectos y el sistema señala estos datos.	A partir de la acción 2 del EC 3.2. 1. El sistema verifica que existen datos incorrectos. 2. El sistema señala los datos incorrectos. Ir a la acción 2 del EC 3.2.
	EC 3.4: Datos Obligatorios.	El actor no introduce todos los datos obligatorios y el sistema muestra un mensaje de error.	A partir de la acción 3 del EC 3.1. 1. En caso de no haber llenado uno de los campos obligatorios el sistema mostrara un mensaje de error. 2. El actor llena el campo obligatorio de la nueva baja y pulsa la opción Aceptar. 3. El sistema da baja a la caracterización estadística y muestra un mensaje de que se cumplió el proceso satisfactoriamente.
SC 4: Hacer traslado Caracterización Estadística.	EC 4.1: Hacer traslado Caracterización Estadística.	El actor selecciona la opción de hacer un traslado, el sistema muestra los datos a seleccionar y el actor elige la brigada origen y a la que desea trasladar el estudiante. El sistema realiza el traslado y muestra un mensaje.	1. El caso de uso comienza cuando el actor pulsa la opción Trasladar. 2. EL sistema muestra la brigada de origen y la brigada de destino 3. El actor escoge la brigada origen y destino para realizar el traslado. 4. El sistema muestra las caracterizaciones estadísticas de la brigada origen. 5. El actor selecciona la caracterización que desea trasladar y pulsa el botón Trasladar. 6. El sistema realiza el traslado y muestra un mensaje de que se cumplió el proceso satisfactoriamente.
	EC 4.2: Datos Obligatorios.	El actor no introduce todos los datos obligatorios y el sistema muestra un mensaje de error.	A partir de la acción 4 del EC 4.1. 1. En caso de no haber llenado uno de los campos obligatorios el sistema mostrará un mensaje de error. 2. El actor llena el campo obligatorio de la nueva baja y pulsa la opción Aceptar. 3. El sistema elimina la caracterización estadística y muestra un mensaje de que se cumplió el proceso satisfactoriamente.

Nota: en la Tabla 11 existen tres respuestas del sistema y tres resultados esperados pues corresponden a los valores 1, 2 y 3 de los datos de entrada.

Tabla 11. SC 1 Crear Caracterización Estadística.

Id	EC	Datos de entrada	Valores 1	Valores 2	Valores 3	Respuesta del Sistema 1	Resultado esperado 1	Respuesta del Sistema 2	Resultado esperado 2	Respuesta del Sistema 3	Resultado esperado 3
EC 1.1	Crear Caracterización Estadística	**Brigada**	brigada eco 1	brigada eco 1	brigada eco 1	El sistema almacena la planilla creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.	Satisfactorio.	El sistema almacena la planilla creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.	Satisfactorio.	El sistema almacena la planilla creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.	Satisfactorio
		Nombre	Roberto	Celia María	Jorge Enrique						
		Apellidos	Arjona Miranda	De La Osa Ester	Hurtado Gutiérrez						
		CI	84092858741	83110958344	84022154737						
		Año	Quinto	Tercero	Quinto						
		Carrera	C. Computación	Economía	C. Computación						
		País	Cuba	Mongolia	Cuba						
		Sexo	Masculino	Femenino	Masculino						
		Está embarazada	N/A	Sí	N/A						
		Tiempo de embarazo	N/A	5 meses	N/A						
		Raza	Blanca	Blanca	Blanca						
		Procedencia social	Obrera	Campesina	Profesional						
		Militancia	UJC	-	-						
		Estado civil	Soltero	Casado	Soltero						
		Hijos	No	Sí	Sí						
		Cantidad hijos	N/A	3	1						

		Forma de ingreso	Preuniversitario	Otra	Preuniversitario						
		Miembro FEU	Sí	No	Sí						
		Tiene alguna resp.	No	No	Sí						
		Responsabilidad	N/A	N/A	Secretario CB						
		Familiares vive	Madre, Otros	Otros	Madre, Padre, Otros						
		Padres fallecidos	-	-	-						
		Padres divorciados	Sí	No	No						
		Tiene acceso comp.	Sí	No	Sí						
		Frecuencia	Diariamente	N/A	Diariamente						
		Identific. del medio	Estatal	N/A	Estatal						
		Dirección particular	Comunidad gran panel, edificio 12, apartamento 34	Carretera central KM 3	-						
		Provincia	Camagüey	Camagüey	Villa Clara						
		Municipio	Nuevitas	Camagüey	Remedios						
		Consejo popular	Los Micros	San Blas	Remate						
		Dirección beca	Edificio 87 apartamento 104	Edificio 123 apartamento 305, Cama	-						

				güey, Camagüey.							
		Teléfono	8353005	8372290	-						
EC1.2	Datos Obligatorios	**Brigada**	Brigada eco 1	-seleccione-	Brigada eco 1	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.
		Nombre	-	Antonio	Katia						
		Apellidos	-	Marrero Palomino	Reyes Uno						
		CI	84011717482	84011717482	-						
		Año	Quinto	-seleccione-	Tercero						
		Carrera	Informática	-seleccione-	Economía						
		País	Cuba	-seleccione-	Venezuela						
		Sexo	Masculino	-seleccione-	Femenino						
		Está embarazada	N/A	N/A	Sí						
		Tiempo de embarazo	N/A	N/A	-selecione-						
		Raza	-seleccione-	-seleccione-	-seleccione-						
		Procedencia social	-seleccione-	-seleccione-	-seleccione-						
		Militancia	-	-	-						
		Estado civil	-	-	Soltero						

			seleccione-	seleccione-							
		Hijos	Sí	No	No						
		Cantidad hijos	2	N/A	N/A						
		Forma de ingreso	-seleccione-	-seleccione-	-seleccione-						
		Miembro FEU	-seleccione-	-seleccione-	-seleccione-						
		Tiene alguna resp.	No	No	No						
		Responsabilidad	N/A	N/A	N/A						
		Familiares vive	-	-	-						
		Padres fallecidos	-	-	-						
		Padres divorciados	-	-	-						
		Tiene acceso comp.	Sí	No	No						
		Frecuencia	-seleccione-	N/A	N/A						
		Identific. del medio	-seleccione-	N/A	N/A						
		Dirección particular	-	-	-						
		Provincia	-seleccione-	-seleccione-	-seleccione-						
		Municipio	-seleccione-	-seleccione-	-seleccione-						
		Consejo popular	-seleccione-	-seleccione-	-seleccione-						
		Dirección beca	-	Edificio 87,	-						

				apto 203							
		Teléfono	-	8353010	-						
EC1.3	Datos Incorrectos	**Brigada**	brigada eco 1	brigada eco 1	brigada eco 1	El sistema muestra un mensaje de error al lado del campo CI.	Satisfactorio.	El sistema muestra un mensaje de error informando que hay campos incorrectos.	Satisfactorio.	El sistema almacena la planilla creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.	No Satisfactorio.
		Nombre	Yeimys William	$&/¬€#	Dariena						
		Apellidos	Seife Pérez	Ramírez	Ramírez						
		CI	840928587	Ppeeeepppee	8402215737						
		Año	Quinto	Cuarto	Quinto						
		Carrera	C. Computación	C. Computación	Economía						
		País	Cuba	Cuba	Cuba						
		Sexo	Masculino	Femenino	Femenino						
		Está embarazada	N/A	N/A	Sí						
		Tiempo de embarazo	N/A	N/A	6 meses						
		Raza	Blanca	Blanca	Blanca						
		Procedencia social	Campesina	Obrera	Campesina						
		Militancia	PCC	UJC	PCC						
		Estado civil	Casado	Casado	Soltero						
		Hijos	No	No	No						
		Cantidad hijos	-	-	-						
		Forma de	O-18	O-18	CSI						

		ingreso									
		Miembro FEU	Sí	Sí	Sí						
		Tiene alguna resp.	No	No	No						
		Responsabilidad	N/A	N/A	N/A						
		Familiares vive	Madre, Otros	-	Madre						
		Padres fallecidos	-	-	-						
		Padres divorciados	Sí	Sí	Sí						
		Tiene acceso comp.	Sí	No	Sí						
		Frecuencia	Diariamente	N/A	Diariamente						
		Identific. del medio	Personal	N/A	Personal						
		Dirección particular	Comunidad # 1, Calle la Pescadería # 178 D.	Comunidad # 1, Calle la Pescadería # 178 D.	Comunidad # 1, Calle la Pescadería # 178 D.						
		Provincia	Cienfuegos	Matanzas	Villa Clara						
		Municipio	Rodas	Matanzas	Cifuentes						
		Consejo popular	Medidas	Versalles	Mata						
		Dirección beca	Edificio 87 apartamento 203	Edificio 87 apartamento 203	Edificio 87 apartamento 203, San Antonio,						

					Habana.						
		Teléfono	8353010	8353010	ddddddd						
Nota: hacer pruebas con distintas combinaciones para los escenarios EC 1.1, 1.2 y 1.3											

Tabla 12. SC 1 Modificar Caracterización Estadística.

Id	EC	Datos de entrada	Valores 1	Valores 2	Valores 3	Respuesta del Sistema 1	Resultado esperado 1	Respuesta del Sistema 2	Resultado esperado 2	Respuesta del Sistema 3	Resultado esperado 3
EC 2.1	Modificar Caracterización Estadística	**Brigada**	brigada eco 1			El sistema actualiza los datos en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.	Satisfactorio.				
		Estudiantes	Roberto Arjona Miranda								
		Nombre	Roberto								
		Apellidos	Arjona Miranda								
		CI	84092858741								
		Año	Quinto								
		Carrera	C. Computación								
		País	Cuba								
		Sexo	Masculino								
		Está embarazada	N/A								
		Tiempo de embarazo	N/A								
		Raza	Mestizo								

		Procedencia social	Obrera								
		Militancia	UJC								
		Estado civil	Soltero								
		Hijos	No								
		Cantidad hijos	N/A								
		Forma de ingreso	Preuniversitario								
		Miembro FEU	Sí								
		Tiene alguna responsabilidad.	No								
		Responsabilidad	N/A								
		Familiares vive	Madre, Padre								
		Padres fallecidos	-								
		Padres divorciados	Sí								
		Tiene acceso comp.	No								
		Frecuencia	N/A								
		Identific. del medio	N/A								
		Dirección particular	Comunidad gran panel, edificio 12, apartamento 34								
		Provincia	Camagüey								
		Municipio	Nuevitas								
		Consejo popular	Los								

			Micros								
		Dirección beca	Edificio 87 apartamento 104								
		Teléfono	8353005								
EC 2.2	Datos Obligatorios	**Brigada**	brigada eco 1	-seleccione-	brigada eco 1	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.
		Estudiantes	Celia María	Celia María	Roberto Arjona						
		Nombre	-	María	-						
		Apellidos	De La Osa Estany	-	Arjona						
		CI	83110958344	-	84011717482						
		Año	Tercero	-seleccione-	Quinto						
		Carrera	Economía	-seleccione-	Informática						
		País	Mongolia	-seleccione-	Cuba						
		Sexo	Femenino	-seleccione-	Masculino						
		Está embarazada	Sí	N/A	N/A						
		Tiempo de embarazo	5 meses	N/A	N/A						
		Raza	Blanca	-seleccione-	-seleccione-						
		Procedencia social	Campesina	-seleccione-	-seleccione-						

		Militancia	-seleccione-	No	Si						
		Estado civil	Casado	-seleccione-	Casado						
		Hijos	Sí	No	Sí						
		Cantidad hijos	3	N/A	2						
		Forma de ingreso	Otra	O-18	-seleccione-						
		Miembro FEU	No	Si	-seleccione-						
		Tiene alguna resp.	No	No	Si						
		Respons abilidad	N/A	N/A	-seleccione-						
		Familiares vive	Otros	-	-						
		Padres fallecidos	-	-	-						
		Padres divorciados	No	-	-						
		Tiene acceso comp.	No	No	Sí						
		Frecuencia	N/A	N/A	-seleccione-						
		Identific. del medio	N/A	N/A	-seleccione-						
		Dirección particular	Comunidad el Pastel, No 345	-	-						
		Provincia	La Habana	Granma	-seleccione-						
		Municipio	-seleccione-	Yara	-seleccione-						
		Consejo popular	-seleccione-	-seleccione-	-seleccione-						
		Dirección beca	Edificio 123 apartamento 305	-	-						

		Teléfono	8372290	-	-						
EC 2.3	Datos Incorrectos	**Brigada**	brigada eco 1	brigada Tele		El sistema muestra un mensaje de error al lado del campo CI.	Satisfactorio	El sistema almacena la planilla creada en la base de datos y muestra un mensaje de que se cumplió el proceso satisfactoriamente.	No Satisfactorio.		
		Estudiantes	Yeimys William Seife Pérez	Darién López							
		Nombre	Yeimys William	Darién							
		Apellidos	Seife Pérez	López							
		CI	sdfsdsfsfd	86020412348							
		Año	Quinto	Tercero							
		Carrera	C. Computación	Economía							
		País	Cuba	Cuba							
		Sexo	Masculino	Masculino							
		Está embarazada	N/A	N/A							
		Tiempo de embarazo	N/A	N/A							
		Raza	Blanca	Blanca							
		Procedencia social	Campesina	Campesina							
		Militancia	PCC	UJC							
		Estado civil	Casado	Soltero							
		Hijos	No	Si							
		Cantidad hijos	N/A	3							
		Forma de ingreso	O-18	O-18							
		Miembro FEU	Sí	No							
		Tiene alguna responsa	No	No							

		bilidad.									
		Respons abilidad	N/A	N/A							
		Familiares vive	Madre, Otros	-							
		Padres fallecidos	-	-							
		Padres divorciados	Sí	No							
		Tiene acceso comp.	Sí	No							
		Frecuencia	Diariamente	N/A							
		Identific. del medio	Personal	N/A							
		Dirección particular	Comunidad # 1, Calle la Pescadería # 178 D.	Comunidad # 1, Calle la Pescadería # 178 D.							
		Provincia	Cienfuegos	Cienfuegos							
		Municipio	Rodas	Rodas							
		Consejo popular	Medidas	Medidas							
		Dirección beca	Edificio 87 apartamento 203	-							
		Teléfono	2234567	sdehfgh							
Nota: Hacer combinaciones de datos de entrada nulos e incorrectos en el caso que corresponda. Tener en cuenta dar clic en Aceptar con todos los campos vacios.											

Tabla 13. SC 3 Dar Baja Caracterización estadística.

Id	EC	Brigadas	Estudiante	Motivo	Provincia	Municipio	Entidad	Plaza	Respuesta del Sistema	Resultado esperado
EC 3.1	Dar Baja Caracterización estadística.	Brigada Fac Tele PR	Antonio Marrero Palomino	Traslado	N/A	N/A	N/A	N/A	El sistema realiza la baja y muestra un mensaje de que se realizó satisfactoriamente	Satisfactorio.
EC 3.2	Motivo Seleccionado Graduación.	Brigada Fac Tele PR	Yeimys W. Seife	Graduación	Cienfuegos	Rodas	PCC Rodas	Informático	El sistema realiza la baja y muestra un mensaje de que se realizó satisfactoriamente	Satisfactorio.
EC 3.3	Datos Incorrectos.	Brigada Fac Tele PR	Yahima Vigo	Graduación	Camagüey	Camagüey	$%&] {[€€¬	€#|\ K?¬	El sistema muestra un mensaje indicando que hay datos incorrectos. Pero no dice que el teléfono esta mal.	Satisfactorio.
SC 3.4	Datos Obligatorios.	Brigada Fac Tele PR	Yahima Vigo	Graduación	Camagüey	Camagüey	-	-	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.
		Brigada Fac Tele PR	Roberto Arjona	Graduación	-seleccione-	-seleccione-	Joven Club	-	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.
		Brigada Fac Tele PR	Roberto Arjona	Graduación	Camagüey	-seleccione-	-	Cantinero	El sistema muestra un mensaje de que se deben llenar los campos obligatorios que faltan.	Satisfactorio.
Hacer combinaciones de datos para los escenarios 3.2, 3.3 y 3.4.										

Tabla 14. SC 4 Hacer traslado Caracterización estadística.

Id	EC	Brigada	Nueva Brigada	Estudiantes	Respuesta del Sistema	Resultado esperado
EC 4.1	Hacer traslado Caracterización estadística	Brigada Fac Tele PR	Brigada Fac Teleinformática PR	Roberto Arjona	El sistema realiza el traslado y muestra un mensaje de que se realizó la acción satisfactoriamente	Satisfactorio.
		Brigada 2B3 PR	Brigada Fac Tele PR	Roberto Arjona, Jorge E. Hurtado	El sistema realiza el traslado y muestra un mensaje de que se realizó la acción satisfactoriamente	Satisfactorio.
EC 4.2	Datos Obligatorios.	Brigada Fac Tele PR	-seleccione-	-	El sistema muestra un mensaje informando que faltan datos obligatorios.	Satisfactorio.
		Nota: Hacer combinaciones entre datos obligatorios de EC 4.2				

Nombre del Caso:

Buscar Caracterización Estadística

Descripción General

El caso de uso se inicia cuando el usuario desea acceder a una Caracterización Estadística.

Condiciones de Ejecución:

Tiene que existir la caracterización estadística y solamente puede acceder a ella el personal autorizado.

Tabla 15. Secciones a probar en el Caso de Uso Buscar Caracterización Estadística.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Buscar Caracterización Estadística.	EC 1.1: Buscar Caracterización Estadística.	El actor solicita buscar una caracterización estadística y el sistema le provee una interfaz para hacerlo.	1. El caso de uso se inicia cuando cualquiera de los actores desea acceder a la Caracterización Estadística. 2. El actor selecciona las opciones de búsqueda que desee. 3. El sistema muestra la(s) Caracterización(es) Estadística(s) encontradas. 4. El actor da clic encima de uno de los nombres mostrados. 5. El sistema muestra los detalles de la caracterización.

Tabla 16. SC 1 Buscar Caracterización Estadística.

Id	EC	Nombre	Apellidos	Provincia	Municipio, CES o CESE	Facultad o Sede Univ.	Brigada	Respuesta del Sistema	Resultado esperado
EC 1.1	Buscar Caracterización Estadística	-	-	-	-	-	-	El sistema muestra todas las caracterizaciones a las que el usuario tiene permiso ver. Al dar clic sobre uno de los resultados se muestran los detalles del mismo.	Satisfactorio
		Antonio	-	-	-	-	-	El sistema muestra todas las caracterizaciones a las que el usuario tiene permiso ver que tienen como nombre “Antonio”. Al dar clic sobre uno de los resultados se muestran los	Satisfactorio

								detalles del mismo. Al dar clic sobre uno de los resultados se muestran los detalles del mismo.	
		-	Marrero	Camagüey	-	-	-	El sistema muestra todas las caracterizaciones a las que el usuario tiene permiso ver que tienen como apellido “Marrero” y es de la provincia “Camagüey”. Al dar clic sobre uno de los resultados se muestran los detalles del mismo.	Satisfactorio
		-	-	Camagüey	Universidad de Camagüey	Facultad Economía	Brigada econ_6	El sistema muestra todas las caracterizaciones a las que el usuario tiene permiso ver que tienen como apellido “Marrero” y es de la provincia “Camagüey”. Al dar clic sobre uno de los resultados se muestran los detalles del mismo.	Satisfactorio
		Nota: Probar con todas las combinaciones posibles a todos los niveles de usuarios.							

Tabla 17. Registro de defectos y dificultades detectados.

No	Elemento	No conformidad	Etapa de detección	Nivel de importancia	Estado NC	Respuesta de Equipo Desarrollo
1	Crear Caracterización estadística	Al poner en el número de teléfono una cadena de letras el sistema adiciona la caracterización estadística.	Primera iteración	Significativo	18/03/08 PD	Aceptada. Se corrigió.
2	Modificar Caracterización estadística	Al poner en el número de teléfono una cadena de letras el sistema adiciona la caracterización estadística.	Primera iteración	Significativo	18/03/08 PD	Aceptada. Se corrigió.

3.4 Conclusiones

Los casos de prueba son el mecanismo usado para asegurar que el sistema cumple con los requerimientos funcionales, asegurando así la calidad del software. Se incluyeron las descripciones de los principales casos de prueba a los que se sometió el sistema, indicando para cada uno su respuesta, donde en pocos casos la prueba no arrojó el resultado esperado. De manera general las pruebas fueron satisfactorias, ya que arrojaron pocas no conformidades, además de que fueron corregidas por el equipo de desarrollo de manera rápida y eficiente. Contribuyó a obtener un producto con mayor calidad.

Conclusiones

Con la realización de este trabajo se logró:

- Describir las funcionalidades de los módulos Funcionamiento, Estadística y parte del módulo Seguridad.
- Dar cumplimiento de forma satisfactoria al objetivo general, implementándose los módulos de Funcionamiento, Estadística y parte del módulo de Seguridad como parte del sistema informático para la automatización de la gestión de la información en la FEU cubana.
- Aplicar patrones de diseños en la implementación de los módulos.
- Realizar pruebas unitarias a la solución.

Los resultados del trabajo de diploma y la utilización de los módulos implementados serán de mucha ayuda a la Federación Estudiantil Universitaria, ya que contribuyen a una mejor gestión y control de la información y las estadísticas generadas en los diferentes niveles de la organización, agilizan las búsquedas de reportes estadísticos y caracterizaciones a todos los niveles de la FEU en el país, aportando cifras exactas en los reportes estadísticos.

Recomendaciones

Al terminar el desarrollo de este documento se recomienda:

- Realizar pruebas de integración a la propuesta de solución para verificar que el testeo de los módulos, como conjunto, funcione correctamente.
- Implementar, en futuras versiones, una capa de presentación integrada a Ajax para agilizar la navegación y gestión de la información en el sistema.

Referencias Bibliográficas

1. IEEE Std, IEEE Software Engineering Standard: Glossary of Software Engineering Terminology. IEEE Computer Society Press, 1993
2. DNV. [Online] [Cited: Diciembre 8, 2007.] http://www.dnv.es/certificacion/sistemasdegestion/index.asp.
3. Bustelo C, Amarilla R. *Gestión del Conocimiento y Gestión de la Información.* 2001. VIII(34): 226-230.
4. JACOBSON, I. B., GRADY Y RUMBAUGH, JAMES. El Proceso Unificado de Desarrollo de Software. Editorial Félix Varela ed. La Habana: 2000.
5. Reyes-Gavilán, I. G. (n.d.). *Cliente / Servidor*. Retrieved Abril 2, 2008, from http://igrgavilan.iespana.es/doc/MA_20070512_MT_ClienteServidor.pdf
6. Pressman, R. S. (2005). *Ingeniería del Software.* La Habana: Félix Varela.
7. Roger S. Pressman. Ingeniería del software, un enfoque práctico, páginas 281-322. McGrawHill, quinta edición, 2002.
8. JIMÉNEZ, L. E. y ICESI, U. [Consultado el: 5 de marzo de 2008]. Disponible en: http://dspace.icesi.edu.co/dspace/bitstream/item/391/1/ljimenez-javaypuntonet.pdf.
9. [Consultado el: 5 de marzo de 2008]. Disponible en: http://www.manual-java.com/manualjava/caracteristicas-java.html.
10. Yaquelin Y. Morales Rodríguez, A. C. (2007). Propuesta de una Guía para estandarizar la codificación en la Universidad de las Ciencias Informáticas. Disponible en: http://bibliodoc.uci.cu/TD/TD_0222_07.pdf
11. Gracia, P. J. (27 de Mayo de 2005). *Patrones de diseño*. Recuperado el 20 de Mayo de 2008, de http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php

Bibliografía

- Sun. (s.f.). *JavaServer Faces Technology Overview*. Recuperado el 2 de abril de 2008, de Sun Developer Network: http://java.sun.com/javaee/javaserverfaces/overview.html
- Johnson, R. (Mayo de 2005). *Introduction to The Spring Framework*. Recuperado el 2 de Abril de 2008, de http://www.theserverside.com/tt/articles/article.tss?l=SpringFramework
- Desing Patterns. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns*. Addison Wesley, 1995.
- Adobe Systems Incorporated. (2008). *Funciones*. Recuperado el 01 de 05 de 2008, de Adobe: http://www.adobe.com/es/products/dreamweaver/features/
- Ben Collins-Sussman, B. W. (s.f.). *Version Control with Subversion*. Recuperado el 27 de 11 de 2007, de http://svnbook.red-bean.com/
- Díaz, J. F. (s.f.). *Métodos de Prueba de Programas*. Recuperado el 23 de 11 de 2007, de http://www.galeon.com/neoprogramadores/met_test.htm
- Eclipse Foundation, Inc. (2008). *Eclipse Newcomers FAQ*. Recuperado el 14 de 04 de 2008, de Eclipse: http://www.eclipse.org/home/newcomers.php
- Gracia, P. J. (27 de Mayo de 2005). *Patrones de diseño*. Recuperado el 20 de Mayo de 2008, de http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php
- Heiss, J. J. (10 de 2006). *Step up the Java Technology Ladder*. Recuperado el 23 de 03 de 2008, de http://java.sun.com/developer/technicalArticles/Interviews/elliott_qa.html
- Lago, R. (04 de 2007). *Patrón "Data Access Object"*. Recuperado el 03 de 12 de 2007, de http://www.proactiva-calidad.com/java/patrones/DAO.html
- Microsoft Corporation. (2008). *Ayuda y procedimientos de Visio 2007*. Recuperado el 03 de 05 de 2008, de http://office.microsoft.com/es-es/visio/FX100649213082.aspx?CTT=96&Origin=CL100636313082
- Microsoft Corporation. (30 de 11 de 2006). *Manual del grupo de trabajo de pruebas*. Recuperado el 23 de 02 de 2008, de http://www.microsoft.com/spain/technet/desktopdeployment/bdd/2007/testguide/testguide_3.mspx
- PostgreSQL Global Development Group. (2008). *Advantages*. Recuperado el 01 de 05 de 2008, de PostgreSQL: http://www.postgresql.org/about/advantages
- Springframework.org. (2008). *Springframework*. Recuperado el 2008 de 03 de 22, de http://www.springframework.org/
- The Apache Software Foundation. (2007). *Apache Tomcat*. Recuperado el 2 de 10 de 2007, de http://tomcat.apache.org/
- VisualBuilder.com. (2008). *Hibernate Tutorial*. Recuperado el 05 de 2 de 2008, de http://www.visualbuilder.com/java/hibernate/tutorial/

- Yaquelin Y. Morales Rodríguez, A. C. (2007). Propuesta de una Guía para estandarizar la codificación en la Universidad de las Ciencias Informáticas.

Glosario

[1] **Web**, (World Wide Web) o Red Global Mundial es un sistema de documentos de hipertexto y/o hipermedios enlazados y accesibles a través de Internet.

[2] **IEEE**, (The Institute of Electrical and Electronics Engineers) o El Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización.

[3] **Framework**, es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado.

[4] **JavaServer Pages** (JSP) es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.

[5] **API**, (Application Programming Interface) o Interfaz de Programación de Aplicaciones es el conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta librería para ser utilizado por otro software como una capa de abstracción.

[6] **Bean**, es un componente software que tiene la particularidad de ser reutilizable

[7] **Plain Old Java Object**, enfatiza el uso de clases simples y que no dependen de un framework en especial.

[8] **XML** (Extensible Markup Language) o lenguaje de marcas extensible, es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium.

[9] **SQL** (Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

[10] **Servlets,** objetos que corren dentro del contexto de un contenedor de servlets (ejemplo: Tomcat) y extienden su funcionalidad.

[11] **JavaMail**, es una expansión de Java que facilita el envío y recepción de e-mail desde código java.

[12] **RMI**, es un mecanismo ofrecido en Java para invocar un método remotamente.

[13] **TCP/IP**, es un conjunto de protocolos de red que permiten la transmisión de datos entre redes de computadoras.

[14] **HTTP** (HyperText Transfer Protocol), protocolo de transferencia de hipertexto, es el protocolo usado en cada transacción de la Red Global Mundial.

[15] **FTP** (File Transfer Protocol) es un protocolo de transferencia de archivos entre sistemas conectados a una red TCP/IP basado en la arquitectura cliente-servidor.

[16] **IBM** (International Business Machines), es una empresa que fabrica y comercializa herramientas, programas y servicios relacionados con la informática.

[17] **Visual Paradigm**, es una herramienta que apoya notaciones de UML, ingeniería inversa, generación de código.

[18] **Licencia BSD** es la licencia de software otorgada principalmente para los sistemas BSD (*Berkeley Software Distribution*). Pertenece al grupo de licencias de Software Libre.

[19] **World Wide Web Consortium** (W3C), es un consorcio internacional que produce estándares para las principales tecnologías sobre las que se basa la web, como HTML y HTTP.

[20] **HTML** (HyperText Markup Language), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.

[21] **JavaScript** es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web.

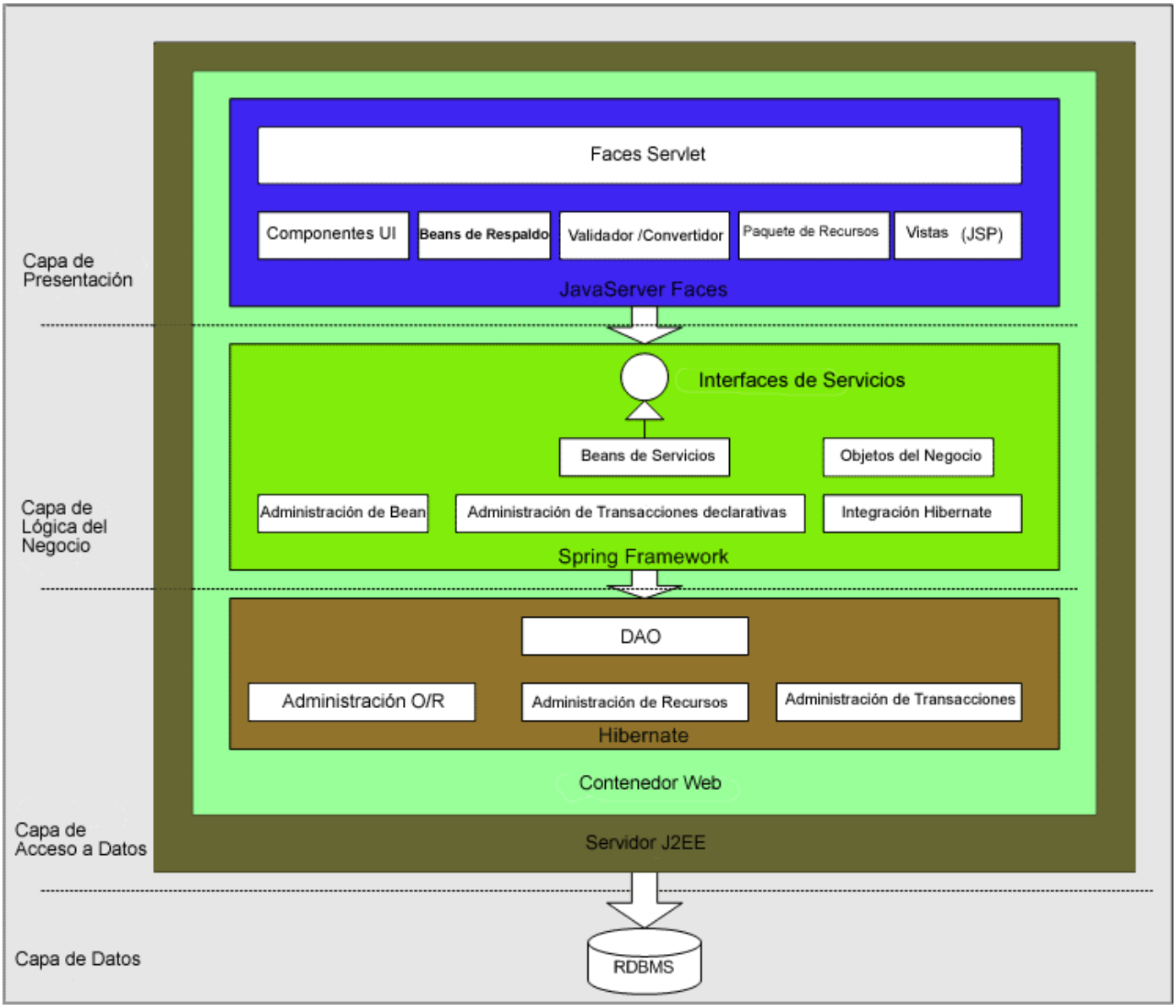
[22] **ASCII**, (American Standard Code for Information Interchange) Código Estadounidense Estándar para el Intercambio de Información; es un código de caracteres basado en el alfabeto latino tal como se usa en inglés moderno y en otras lenguas occidentales.

[23] **URL**, (Uniform Resource Locator) Localizador Uniforme de Recurso. Es una secuencia de caracteres, de acuerdo a un formato estándar, que se usa para nombrar recursos, como documentos e imágenes en Internet**.**

Anexos

Anexo 1

Arquitectura basada en Hibernate+Spring+JSF.



Anexo 2

Proceso Unificado de Desarrollo de Software.