Universidad de las Ciencias Informáticas

Facultad 3

Título: CODIS Sistema para la informatización del proceso de Comisión Disciplinaria de la Facultad 3

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

Autor: Rosalia Fernández Castillo

Tutores:

Ing. Adrian Naranjo García

Ing. Fernando Nápoles Gámez

La Habana, junio de 2012

Declaración de autoría

Declaro que soy el único autor del trabajo: Sistema para la informatización del proceso de Comisión Disciplinaria de la Facultad 3 y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste, firmo la presente a los ____ días del mes de _____ del año 2012.

_____

Rosalia Fernández Castillo

Autor

_____ _____

Ing. Adrian Naranjo García

Tutor

Ing.Fernando Nápoles Gámez

Tutor

Datos de contacto

Tutor: Ing. Adrian Naranjo García.

Graduado de Ingeniero en Ciencias informáticas en el 2010. Actualmente desempeña el rol de Arquitecto de Datos en el Departamento de Soluciones para la Aduana.

Tutor: Ing. Fernando Nápoles Gámez

Graduado de Ingeniero en Ciencias informáticas en el 2010. Actualmente desempeña de Arquitecto de Datos en el Departamento de Soluciones para la Aduana.

Agradecimientos

A mi familia por apoyarme tanto durante toda mi vida y durante estos cinco años llenos de emociones.

A mis tutores por no pensarlo dos veces cuando fui a verlos llorando porque no tenía tutor, por haberme apoyado y ayudado en todo lo que les fue posible.

A mis grandes amigos Gretel, Dayaris y Roberto que estuvieron para mí en los momentos más difíciles de mi vida en la Universidad.

A Leodan y Eddy por dedicarme su tiempo cuando les pedí ayuda.

A Liu y Lisandra por ser también amigas y por apoyarme.

A todos mis compañeros de aula y del apartamento por haber compartido con ellos esta experiencia maravillosa que es la vida en la Universidad.

A todos muchas gracias.

Dedicatoria

A mi mamita Maira por darme la vida, por ser mi amiga, mi apoyo, mi universo, por darme tanto cariño y amor y por ser este nuestro sueño, donde quiera que esté, te amo.

A mi abuela Migdalia por haber estado en todo momento en mi vida, por ser tan fuerte, por ayudarme a afrontar mis complejos, mis miedos, por ser como mi segunda madre.

A mi abuelo Silvio, por hacer hasta lo imposible por mí, y por hacerme reír.

A mi papá por ser mi gran amigo, por darme tantos consejos, por enseñarme como es la vida, por quererme tanto y estar para mí cuando lo necesite.

A mi tía Mariela por quererme y ayudarme tanto y a mi primo Alejandro porque lo adoro.

A mi tía Raisa y a mis primas Lidice y Lianet, por acogerme y ayudarme.

A mi novio Miguel Alejandro, por quererme, por sorprenderme en cada momento, por aguantarme en estos días tan estresantes, por darme cariño.

A todos mis amigos, los de la Universidad, los del grupo, los que están y los que ya no están, los llevo en mi corazón.

A todos muchas gracias por ser parte de este momento tan importante en mi vida.

Resumen

En la Universidad de Ciencias Informáticas se cuenta con diferentes Comisiones Disciplinarias por cada facultad, para analizar las indisciplinas cometidas por algunos de los estudiantes que radican en la misma. Este proceso actualmente no se desarrolla con la calidad ni el tiempo requerido. Se generan excesivos volúmenes de planillas, además la información de los casos no se encuentra siempre disponible, no se sigue en ocasiones el mismo formato para las actas, no siempre se cuenta con los datos necesarios de cada implicado, no se cumple con los tiempos estipulados en el Reglamento Universitario para la culminación de cada subproceso y además la información de cada caso cuando es cerrado, se archiva impidiendo el acceso a la misma de manera ágil y sencilla.

En el presente trabajo se realiza el análisis, diseño e implementación de un sistema que permita gestionar el proceso asociado a la Comisión Disciplinaria en la Facultad 3 de forma eficiente, garantizando que se encuentre disponible la información completa de cada uno de los casos, las operaciones de adicionar denuncia, adicionar caso a una comisión, adicionar miembro a comisión, entre otras se realizan de forma ágil, porque no se hace necesario introducir muchos datos, con solo un clic se obtiene la mayoría de la información. Como resultado del mismo se obtiene un sistema funcional, que cumple con las necesidades que presenta la Facultad 3, permitiendo que mejore el funcionamiento de las Comisiones Disciplinarias.

Palabras claves:

Comisión Disciplinaria, Facultad 3, Sistema.

Tabla de Contenido

Índice de Figuras

Índice de Tablas

Introducción

En Cuba actualmente se está llevando a cabo la informatización de gran parte de las instituciones existentes. Como apoyo a este enorme reto se crea la Universidad de Ciencias Informáticas (UCI), donde se forman profesionales de esta rama, comprometidos con la Revolución y que se encuentren altamente calificados en su trabajo. A pesar de ello algunos de estos estudiantes no cuentan con los valores necesarios para convertirse en buenos profesionales, cometiendo faltas, ya sean menos o más graves, pero que de una forma u otra afectan su integridad y la de la Universidad.

Para analizar al estudiante que comete una indisciplina se crean las Comisiones Disciplinarias. Las mismas existen con la finalidad de contribuir a la formación de los profesionales y no como una forma de castigo.

Cada una de las facultades con las que cuenta la UCI realiza el proceso de las Comisiones Disciplinarias, pero muchas veces no de la misma forma. En ocasiones no se sigue el mismo formato o forma de proceder.

Este proceso comienza cuando se realiza la denuncia por parte de alguna persona de la Universidad. Luego se envía el caso a una Comisiones Disciplinarias y la misma lo evalúa, se obtiene la opinión de representantes de la FEU y de la UJC, se toman las evidencias; el demandado entrega por escrito su versión de cómo sucedieron los hechos. También se entrevistan a los testigos. Todo esta parte del proceso se realiza en un período de 30 días.

Luego de haber evaluado exhaustivamente todos los elementos, se propone una medida disciplinaria. Posteriormente el caso y la propuesta de medida disciplinaria pasa al decano de la facultad en la que se esté realizando el proceso y este decide si se procede con la medida disciplinaria o se aplicará otra, esto llevará un período de 5 días.

La Comisión Disciplinaria puede solicitar una prórroga del caso por algún motivo, porque alguno de los miembros se debe ausentar de la Universidad, porque se estén procesando más evidencias entre otros. El decano es el encargado de determinar si se aprueba o no. Además la Comisiones Disciplinarias, puede solicitar que se detenga el caso, porque se cuenten con pocas evidencias, o que las mismas estén infundadas o por alguna otra situación.

Si el estudiante desea apelar, se procede a revisar nuevamente el caso. Al culminar la revisión, la

medida que se determine no tiene revocación, es decir si se mantiene la sanción anterior tendrá que cumplirla.

La Facultad 3 cuenta con diferentes Comisiones Disciplinarias que se dividen los casos según su gravedad. Algunas atienden los menos graves, otras los graves y otras los más graves. Todo este proceso se lleva a cabo manualmente. En ocasiones la información que se necesita de la persona no está completa. Por tener todo documentado en planillas manuscritas, los integrantes de la CD no pueden acceder a la información en cualquier momento lo que representa un freno para el desarrollo del proceso. Toda la información de un caso, al cerrar el mismo es archivada, lo que representa un freno si se necesita analizar a algún estudiante, para conocer si ha cometido alguna indisciplina.

Toda la problemática expresada anteriormente da lugar a que la investigación esté enfocada a resolver el siguiente **problema:** La realización manual del proceso asociado a la Comisión Disciplinaria está afectando la eficiencia con que se realiza el mismo. Para ello se tiene como **objeto de estudio:** Los procesos asociados a la Comisión Disciplinaria y como **campo de acción:** El procedimiento asociado a las Comisiones Disciplinarias de la Facultad 3.

Para solucionar la problemática existente se tiene como **objetivo general:** Desarrollar un sistema que permita gestionar la información asociada al proceso de la Comisión Disciplinaria en la Facultad 3.

Para alcanzar el objetivo trazado se dará cumplimiento a los siguientes **objetivos específicos:**

- Realizar el marco teórico de la investigación
- Realizar el análisis y el diseño de la solución
- Implementar un sistema para la gestión de Comisiones Disciplinarias
- Evaluar técnicamente el sistema implementado

Estructura del Trabajo:

El presente trabajo está conformado por tres capítulos. En el capítulo 1 se realiza la Fundamentación Teórica. En el capítulo 2 se realiza el Diseño de la propuesta de solución. En el capítulo 3 Implementación y pruebas del sistema.

Capítulo 1: Fundamentación teórica

1.1 Introducción al Capítulo

En el presente capítulo se analizará información actualizada referente a sistemas nacionales e internacionales de justicia, se realizará un estudio del lenguaje de programación seleccionado, de las principales herramientas a utilizar para el desarrollo del sistema, es decir los gestores de bases de datos, los IDEs (Entorno de Desarrollo Integrado) de programación, la metodología que se empleará, entre otras.

1.2 Sistemas judiciales internacionales

Sistema del Poder Judicial de la Provincia de Corrientes

Figura 1 Sistema de Poder Judicial de la Provincia de Corrientes

Poder Judicial de la Provincia de Corrientes, es un sitio web que le permite a sus usuarios mantenerse informados sobre el acontecer judicial de la provincia, el país (Argentina) y del mundo. En el mismo pueden encontrarse datos de las personas que constituyen el Superior Tribunal de Justicia, un mapa con

las principales jurisdicciones, entre otros. Se realizan lanzamientos de concursos de cargos, es decir para que personas capacitadas ocupen un determinado cargo dentro de una Jurisdicción o en alguna oficina jurídica. Se brinda además una guía telefónica con números de teléfonos de las Jurisdicciones, y juzgados. Se proporcionan los correos electrónicos de los Juzgados, de la Biblioteca de Corrientes, entre otros que puedes ser de interés para los usuarios (stjprensa@juscorrientes.gov.ar, 2007).

En este sitio se encuentran las publicaciones jurídicas, publicaciones médicas que se establecen en la provincia y el país, además de las constituciones, los códigos procesales, las leyes entre otras informaciones de interés para los ciudadanos de esa provincia y para toda persona que desee conocer acerca del poder judicial de la misma (stjprensa@juscorrientes.gov.ar, 2007).

Se prestan además varios servicios dentro los que se pueden encontrar las guías de trámites para adopciones, fotos de personas desaparecidas de la provincia, entre muchos otros (stjprensa@juscorrientes.gov.ar, 2007).

European Justice

Figura 2 Portal europeo de justicia

"El portal Europeo de e-Justicia está pensado para ser en el futuro una ventanilla única en el ámbito de la Justicia". Este sistema les facilita la vida a los ciudadanos brindándoles información sobre los sistemas jurídicos y mejorando el acceso a la justicia en la Unión Europea (UE). El mismo está disponible en 22

lenguajes. La información que brinda es la referente a los derechos de la UE, la jurisprudencia, los sistemas judiciales, profesiones jurídicas, redes de la justicia, los derechos de los demandados, ofrece información de las normas de esta nación para el registro de testamentos, los derechos de las personas que han sido víctimas de delitos, los derechos de los acusados, entre otros datos de interés para sus usuarios. European Justice, es un sitio interactivo e informativo que hace posible el mantenerse informado acerca de los sistemas judiciales de la UE (Europea, 2007).

1.3 Sistemas Judiciales en Cuba

Sistema de Informatización de la Gestión de las Fiscalías II (SIGEF II)

Figura 3 Sistema de Informatización de la Gestión de Fiscalías

La Fiscalía General de la República (FGR), luego de iniciado el proceso de informatización que se lleva a cabo en todo el país, identificó la necesidad de informatizar sus principales procesos, con lo que podrá controlar las informaciones obtenidas y generadas, será más fácil su almacenamiento, optimización del tiempo, además de otras ventajas que mejorarán la gestión interna del trabajo a realizar. Los procesos no informatizados actualmente se realizan manualmente, generándose un alto volumen de información, siendo necesario utilizar gran cantidad de tiempo para generar todos los documentos pertinentes, obteniéndose como resultado que al finalizar la investigación los expedientes excedan, generalmente, las 100 páginas. (Machado., 2011)

El SIGEF II desarrollado en la Facultad 3 de la UCI está dirigido a adicionar nuevas funcionalidades a la anterior versión del mismo, lo que propiciará el aumento del nivel de informatización de los procesos fiscales, condicionando el incremento de la calidad de la tramitación, supervisión y control en tiempo real de los procesos fiscales, la reducción de los términos de las actividades y diligencias practicadas, se controlará mejor el cumplimiento de la garantía de los procesos, la fuerza fiscal será utilizada de forma más óptima y disminuirá la utilización de documentos en formato duro, todo esto trae consigo una mayor economía procesal. (Machado., 2011)

El SIGEF II hará posible que los procesos judiciales del país se realicen con mayor eficiencia y calidad, evitando el largo y tediosos trabajo manual. Además ayudará a controlar mejor el cumplimiento de la Constitución, las leyes y demás disposiciones legales, por los organismos del Estado, entidades económicas y sociales y por los ciudadanos; y la promoción y el ejercicio de la acción penal pública en representación del Estado. El lenguaje que se utilizó para el desarrollo del mismo es PHP. (Machado., 2011)

Con el estudio realizado a los sistemas jurídicos internacionales y de Cuba, se determinó que no pueden utilizarse los mismos para gestionar los procesos asociados a la Comisión Disciplinaria, porque los sistemas internacionales, son sitios informativos, es decir el usuario no interactúa con el mismo para realizar alguna operación, solamente puede mantenerse informado con el mismo. Y el SIGEF II, presenta diferencias de versiones en las herramientas a utilizar por cada uno en cuanto al lenguaje PHP y el framework symfony. En el SIGEF II se utiliza la versión 5.2 de PHP y la que se necesita en el sistema a implementar es 5.3 o superior. Con respecto al framework Symfony, la versión utilizada en el Sistema SIGEF II es 1.3.11 y la que debe utilizarse en el sistema es 1.4 o superior.

1.4 Lenguaje de programación

PHP 5.3

Para desarrollar el sistema se seleccionó el lenguaje PHP en su versión 5.3, producto de que el sistema debe regirse por la plataforma tecnológica de la Facultad 3, en la que el único lenguaje que puede usarse es PHP y con una versión 5.3 o superior.

PHP (cuyo acrónimo es *Hypertext Preprocessor*) es un lenguaje de programación de código abierto muy utilizado principalmente en el desarrollo web y puede ser incrustado en HTML. En este lenguaje en lugar

de utilizar muchos comandos para mostrar HTML (como en C o Perl) contienen el código HTML incluido. El código PHP, se encuentra en medio de etiquetas de comienzo y final <? php y?>, que permiten entrar y salir del modo PHP (Achour, et al., 2001).

Una de las grandes ventajas de PHP es que es muy simple para la persona que va programar con él por primera vez y brinda además muchas funcionalidades avanzadas para los programadores profesionales. Aunque el desarrollo de PHP está centrado en programación de scripts del lado servidor, se puede utilizar para cualquier cosa que se pueda hacer con un script CGI[1], como procesar la información de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies (Achour, et al., 2001).

Este lenguaje puede ser utilizado en cualquiera de los principales sistemas operativos del mercado incluyendo Linux, muchas variantes Unix (HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS, entre otros. Soporta a la mayoría de los servidores web que existen actualmente, como es el caso de Apache, IIs, y otros más (Achour, y otros, 2001).

PHP también cuenta con soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) entre otros (Achour, y otros, 2001).

1.5 Análisis de los Frameworks para PHP

Un framework (marco de desarrollo o de trabajo) no es más que una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado. Puede incluir soporte de programas, librerías y un lenguaje de scripting entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa básicamente una arquitectura de software que modela las relaciones generales de las entidades del dominio (Tupe, y otros, 2008).

Cada día se hace más necesario la utilización de los frameworks para el desarrollo de aplicaciones web de forma rápida, estructurada y eficiente. Durante mucho tiempo prevalecieron los frameworks asociados con los lenguajes Ruby y Java, pero los desarrollados en PHP han tenido mucho auge en los últimos años. Entre los más populares se encuentran: CodeIgniter y Symfony (Tupe, y otros, 2008).

Los frameworks que se estudiarán son los dos mencionados anteriormente por ser los que se

[1] Es el medio de comunicación que emplea un servidor Web para enviar información útil en ambos sentidos, entre el visualizador y su propio programa de cómputo.

encuentran en la plataforma tecnológica de la facultad.

CodeIgniter

CodeIgniter es un framework de PHP de código libre con tamaño pequeño, realizado para programadores que necesiten un conjunto de herramientas simples y elegantes, para desarrollar las funciones de una aplicación web, versátil y segura (Alvarez, 2009).

Contiene varias librerías y propone una forma de desarrollarlas, y además define una arquitectura que ayuda a los programadores a desarrollar de una forma más estructurada. Implementa el patrón arquitectónico MVC. Muchas de sus utilidades y formas de funcionamiento son opcionales, lo que le proporciona al desarrollador más libertad (Alvarez, 2009).

CodeIgniter se caracteriza por ser *versátil*, es decir a diferencia de otros frameworks, es capaz de trabajar la mayoría de los entornos o servidores. También es *compatible* con PHP 4, por lo que se puede utilizar en cualquier servidor. Funciona además con PHP 5. Es flexible, cuenta con una peculiar forma de trabajar, la cual se puede seguir o no, al igual que se pueden saltar sus reglas de codificación, según el criterio de cada desarrollador. Su núcleo es bastante ligero, protegiendo de una sobrecarga al servidor. Y su documentación está disponible y en forma de tutoriales, es muy fácil de aprender (Alvarez, 2009).

La mayor ventaja de este framework es su accesibilidad, ya que puede utilizarse en gran cantidad de entornos. (Alvarez, 2009)

Symfony

Symfony es un framework para PHP, desarrollado íntegramente con PHP 5 y diseñado para optimizar el desarrollo de aplicaciones web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación (Potencier, y otros, 2008).

Este popular framework se ha probado en diferentes proyectos reales y se utiliza en sitios web de comercio electronico de primer nivel. Además es compatible con la mayoría de los gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. (Potencier & Zaninotto, 2008).

Se basa en el patrón de diseño MVC (Modelo Vista Controlador), por lo que separa la lógica del negocio,

la lógica del servidor y la presentación de la aplicación web (Potencier, y otros, 2008).

Symfony cuenta con una de las mayores comunidades de usuarios, dado su madurez en el mercado. Utiliza las líneas de comando (Interfaz de Línea de Comandos (CLI), para la creación de proyectos. Esto puede mostrar un poco de confusión al principio, pero luego los usuarios se darán cuenta de que posibilita el ahorro en tiempo de muchas de las tareas repetitivas de una aplicación web, por ejemplo en la administración de tablas, módulos de seguridad, archivos de configuración, entre otras (Potencier & Zaninotto, 2008).

Una de las principales ventajas de Symfony es que hace uso del framework nativo de PHP, PEAR (PHP Extension and Application Repository). Este es un proyecto creado por la comunidad PHP y es incluido en la distribución estándar de PHP. El mismo es un marco y un sistema de distribución para componentes PHP reutilizables (Potencier & Zaninotto, 2008).

Para la instalación de Symfony, solo es necesario la configuración de pocos archivos en el servidor Web, en el caso del Apache: httpd.conf y activar el mod_rewrite del mismo.

Otros motivos por los cuales escoger Symfony (Ecured, 2007)*:*

- *Escalable:* Symfony es escalable si se dispone de los recursos necesarios. Yahoo utiliza Symfony para programar aplicaciones con 20 millones de usuarios y 12 idiomas.
- *Probado:* Symfony ha sido probado con éxito durante varios años en aplicaciones muy diferentes. Desde sitios web con millones de usuarios hasta aplicaciones más pequeñas (del.icio.us, Yahoo Bookmarks, Yahoo Answers).
- *Soporte:* Symfony sigue una política de tipo LTS (long term support). Las versiones estables se mantienen durante 3 años sin cambios pero con una continua corrección de los errores conocidos.
- *Licencia:* Symfony utiliza una licencia MIT, con la que puedes hacer aplicaciones web comerciales, gratuitas y/o de software libre.
- *Compromiso:* la empresa que ha creado Symfony no vive del framework, sino de las aplicaciones que hace con él. Esto significa que a ellos les interesa aspectos como el rendimiento, la buena documentación, el soporte muy largo, entre otros.

- *Código:* Desde su primera versión Symfony ha sido creado para PHP 5, desechando la versión PHP 4 (que ha sido declarada obsoleta recientemente).
- *Seguro:* Se puede controlar hasta el último acceso a la información e incluye por defecto protección contra ataques XSS y CSRF.
- *Documentado:* se trata del framework PHP mejor documentado: miles de páginas en el wiki oficial, tutoriales de hasta 250 páginas y un libro gratuito de casi 500 páginas. Además, el libro está completamente traducido al español.
- *Calidad:* su código fuente incluye más de 8.000 pruebas unitarias y funcionales.
- *Internacionalización:* Se pueden crear aplicaciones en varios idiomas. La internacionalización está integrada en el framework, funciona bien, sigue los estándares (XLIFF), es muy completa y está probada en aplicaciones reales.

Luego del estudio realizado a los frameworks anteriores, se decide utilizar en el sistema a desarrollar Symfony, por todas las ventajas y características mostradas.

1.6 Herrramienta CASE[2]

Visual Paradigm 3.4

En la actualidad las herramientas CASE[3], proporcionan gran ayuda para las empresas, para el proceso de desarrollo de un sistema, desde su inicio hasta su fin. La mayoría de estas herramientas son muy costosas y se necesita además invertir en el entrenamiento del personal de la empresa (Angeles Lara, Santoyo Figueroa, & Sierra Cruz, 2007).

La herramienta CASE a utilizar en el desarrollo del sistema CODIS es el Visual Paradigm For UML, la cuál se clasifica como CASE Cruzado de Ciclo de Vida. Estas herramientas se caracterizan porque apoyan actividades que tienen lugar a lo largo de todo el ciclo de vida de un proyecto e incluyen

[2] ***C****omputer* ***A****ided* ***S****oftware* ***E****ngineering*, Ingeniería de Software Asistida por Computadora (http://es.wikipedia.org/wiki/Herramienta_CASE)

actividades como la gestión de proyectos y la estimación (Angeles Lara, Santoyo Figueroa, & Sierra Cruz, 2007).

El Visual Paradigm se integra con las herrramientas java (Angeles Lara, Santoyo Figueroa, & Sierra Cruz, 2007):

- ✓ Eclipse/ IBM Web Sphere
- ✓ JBuilder
- ✓ Netbeans IDE
- ✓ Oracle JDeveloper
- ✓ BEA Weblogic

¿Por qué elegir Visual Paradigm for UML?

Es un producto que facilita a las organizaciones la diagramación visual y el diseño de sus proyectos de sistema y les brinda la posibilidad de integrar y desplegar sus aplicaciones empresariales de mision crítica, y de sus bases de datos subyacentes. Proporciona el código y compatibilidad hasta con diez lenguajes, como es el caso de Java, C++, CORBA IDL, entre otros (Angeles Lara, Santoyo Figueroa, & Sierra Cruz, 2007).

Entre sus principales ventajas se pueden encontrar (Angeles Lara, Santoyo Figueroa, & Sierra Cruz, 2007):

- ➢ Navegación intuitiva entre código y modelo.
- ➢ Poderoso generador de documentación y Reportes UML/PDF/HTML/MS Word.
- ➢ Demanda en tiempo real, modelo incremental del viaje redondo y sincronización de código fuente.
- ➢ Superior entorno de modelado visual.
- ➢ Soporte completo de notaciones UML.
- ➢ Diagramas de diseño automático sofisticado. Análisis de texto y soporte de tarjeta CRC.

Entre los diagramas que se pueden realizar con esta herramienta están:

- ✓ De clases

- ✓ De paquetes
- ✓ De Objetos
- ✓ De estructura de compuesto.
- ✓ De componentes
- ✓ De despliegue
- ✓ De casos de uso
- ✓ De actividades
- ✓ Entre otros

1.7 Servidor web

Apache 2.2.17

Apache es uno de los servidores web más utilizados en el mundo, por su configurabilidad y robustez. Es prácticamente universal, producto de que se encuentra en una gran cantidad de Sistemas Operativos (UNIX, Linux, Windows, entre otros). Su licencia es descendiente de la licencia BSD, lo que proporciona facilidad para estudiar el código fuente, modificarlo y distribuirlo. Trabaja con lenguajes como Perl, PHP, Java entre otros. Por estar diseñado en módulos resulta sencillo ampliar sus capacidades. Existen diferentes módulos que se van instalando cuando son necesarios (editorbfb, 2011).

Representa una ventaja para poder personalizar la respuesta ante los posibles errores que se den en el servidor (editorbfb, 2011).

En la versión 2.2.17 de este potente servidor Web se integran parches de seguridad para vulnerabilidades que podían llegar a permitir ataques de tipo DoS (Denegación de Servicios) (Desarrollo Web, 2010).

1.8 Análisis de Gestores de Bases de Datos

A continuación se realiza un análisis de los gestores de bases de datos MySQL y PostgreSQL para decidir cuál se empleará en el sistema a desarrollar.

MySQL

El gestor de bases de datos MySQL, cuenta con diferentes características como son (MySQL, 1997-2011):

- ✓ Interioridades y portabilidad.
- ✓ Escrito en C y en C++.
- ✓ Probado con un amplio rango de compiladores diferentes.
- ✓ Funciona en diferentes plataformas.
- ✓ Utiliza GNU Automake, Autoconf, y Libtool para portabilidad.
- ✓ APIs disponibles para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, y Tcl.
- ✓ Proporciona sistemas de almacenamiento transaccional y no transaccional.
- ✓ Usa tablas en disco B-tree (MyISAM) muy rápidas con compresión de índice.
- ✓ Relativamente sencillo de añadir otro sistema de almacenamiento. Esto es útil si desea añadir una interfaz SQL para una base de datos propia.
- ✓ Un sistema de reserva de memoria muy rápido basado en threads.
- ✓ Joins muy rápidos usando un multi-join de un paso optimizado.
- ✓ Tablas hash en memoria, que son usadas como tablas temporales.

Su última versión, trae consigo nuevas ventajas para sus usuarios, como son (Robles, 2007):

- Particionamiento de tablas: lo que significa separa la tabla de datos en particiones pequeñas para acelerar las consultas.
- Events Scheduling o Programación de Eventos: con esta versión se pueden programar eventos por fecha y hora para lanzar Procedimientos almacenados o Sentencias SQL
- Row Replication: es un nivel mayor en la replicación de datos.
- Pluggins: se pueden desarrollar pluggins externos para entender el motor de bases de datos.

La arquitectura de MySQL se caracteriza principalmente por separar el motor de almacenamiento del resto de los componentes de la arquitectura. Es decir el diseño del gestor está preparado para que se pueda cambiar el gestor de almacenamiento. Con esto se pueden crear nuevos motores de

almacenamiento especializados para ciertas tareas o tipos de aplicaciones.

PostgreSQL 9.1

PostgreSQL cuenta con un grupo de características que lo hacen prevalecer ante los demás gestores de bases de datos. Ejemplo de las mismas son:

1. Posee una gran escalabilidad. Es capaz de ajustarse al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta (en algunas bibliografías se dice que ha llegado a soportar el triple de carga de lo que soporta MySQL).

2. Implementa el uso de rollback's[4], subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz, y ofreciendo soluciones en campos en las que MySQL no podría.

3. Tiene la capacidad de comprobar la integridad referencial, así como la de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como Oracle.

Esta última actualización de este gestor de base de datos de código abierto líder ofrece tecnología innovadora, extensibilidad sin igual, y nuevas características como replicación sincrónica, método de indexado y conectores de datos foráneos. Provee además algunas de las más avanzadas características empresariales de cualquier base de datos de código abierto, y es soportado por una entusiasta e innovadora comunidad con probados casos de éxito (PostgreSQL, 2008).

PostgreSQL 9.1 brinda a los usuarios diferentes características que habían estado solicitando hacía años, quitando obstáculos para el despliegue de aplicaciones nuevas o migradas en PostgreSQL. Estas incluyen la replicación sincrónica: que permite alta disponibilidad con consistencia sobre múltiples servidores, la regionalización por columna: soportando correctamente el ordenamiento por lenguaje en las bases de datos, tablas o columnas y las tablas unlogged, que representan un importante incremento del rendimiento para los datos efímeros (PostgreSQL, 2008).

La presente versión 9.1 incluye muchas ventajas que son nuevas en la industria de las bases de datos.

[4] Operación que devuelve a la base de datos a algún estado previo (http://es.wikipedia.org/wiki/Rollback)

Ejemplo de ellas son (PostgreSQL, 2008):

✓ Indexamiento de los K vecinos más próximos (K-Nearest-Neighbor): índices sobre "distancia" para consultas rápidas de ubicación y búsquedas de texto.

✓ Nivel de Aislamiento Serializable a través de "Snapshots": mantiene consistentes múltiples transacciones concurrentes sin el uso de bloqueos, usando "verdadera serialización".

✓ Writeable Common Table Expressions: ejecuta actualizaciones multi-fases complejas en una simple consulta.

✓ Security-Enhanced Postgres: despliega seguridad de nivel militar y control de acceso mandatorio.

A lo largo de los años, PostgreSQL ha sido el líder en sistemas de base de datos de código abierto. Su comunidad rebasa los mil usuarios y contribuyentes, además de docenas de empresas y organizaciones. Este sistema tiene una experiencia de 25 años de ingeniería. Las características que ha ido desarrollando PostgreSQL, lo hace competitivo entre los demás sistemas de base de datos propietarios (PostgreSQL, 1996).

Luego de haber realizado un estudio de estos gestores de base de datos, se determinó que se utilizará en el sistema a desarrollar PostgreSQL porque el uso de MySQL hoy en día es riesgoso, por el hecho de que pertenece a la compañía privativa Oracle (luego de ser comprada Sun por Oracle) y existe la incertidumbre de que si algún dia decide privatizar sus productos habría que pagar la licencia por el uso de los mismos. (Espacio Linux, 2010)

1.9 Análisis de IDEs (Entorno de desarrollo integrado) de Programación

Un IDE es un entorno de desarrollo integrado o de programación que ha sido empaquetado como un programa de aplicación, o sea, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Puede ser una aplicación por si sola o parte de una aplicación o aplicaciones ya existentes (Ecured, 2012).

Eclipse

Eclipse no es más que un armazón sobre el que se pueden montar herramientas de desarrollo para cualquier lenguaje, mediante la implementación de los complementos adecuados. La arquitectura de pluggins de Eclipse permite además de integrar diversos lenguajes sobre un mismo IDE, introducir otras

aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, entre otras (Chavarría, 2006).

Está desarrollado por completo en el lenguaje Java por lo que presenta un problema habitual en las herramientas Java y es que consumen bastante memoria RAM (Random Access Memory o Memoria de Acceso Aleatorio).

Además se caracteriza por su forma peculiar de compilar, ya que no se puede compilar individualmente un fichero, el compila automáticamente cuando se guardan los cambios realizados. Incluye un depurador potente, sencillo y cómodo de utilizar. Y permite generar automáticamente documentación del propio proyecto.

Aptana

Aptana es un entorno de desarrollo integrado (IDE) para crear aplicaciones web Ajax[5]. Incluye soporte para JavaScript, HTML, DOM y CSS. Este IDE está basado en Eclipse y se encuentra disponible como independiente en Microsoft Windows, Mac OS X y Linux, o como un complemento para Eclipse. Entre los requisitos que tiene que tener un sistema para utilizarlo están (Ecured, 2011):

- Windows - 512 MB RAM,
- Pentium 4-level processors Windows - 512 MB de RAM, Pentium 4 a nivel de procesador.
- Mac OS X - 512 MB RAM, PowerPC G4/G5, Intel or Mac OS X 10.4+
- Mac OS X - 512 MB de RAM, PowerPC G4/G5, Intel o Mac OS X 10.4 +
- Linux - 512 MB RAM, Pentium 4-level processor Linux - 512 MB de RAM, Pentium 4 a nivel de procesador.

Está disponible como una edición de comunidad de código abierto. Todas las funciones son ahora parte de la edición comunitaria de Estudio (Ecured, 2011).

NetBeans IDE 7.0

[5] Acrónimo de ***A****synchronous* ***J****avaScript* ***A****nd* ***X****ML* (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). (http://es.wikipedia.org/wiki/AJAX)

El IDE NetBeans es un entorno premiado de desarrollo integrado disponible para Windows, Mac, Linux y Solaris. El proyecto con este mismo nombre no es más que un IDE de código abierto y una plataforma de aplicaciones que permite a los desarrolladores crear aplicaciones web, empresariales, de escritorio y aplicaciones móviles de forma rápida, utilizando la plataforma Java, así como PHP, JavaScript y Ajax, Groovy y Grails, y C / C + +.

La versión 7.0 del NetBeans IDE trae consigo soporte de idiomas para la codificación a la propuesta de especificación de Java SE (Standard Edition o Edición estándar) 7 con la vista previa para desarrolladores JDK 7. Esta versión brinda una integración mejorada de Oracle WebLogic, así como soporte para base de datos de Oracle y GlassFish 3.1 (Webmasters, 2011).

Entre las principales ventajas que presenta este IDE están (Tecno, 2011):

• Auto-completa el código que se digita: Ante la falta de inicialización de algún argumento, le sugiere al programador la declaración automática del mismo; propone las características disponibles para los elementos, cuando se intenta acceder a estas mediante el punto después de la variable o argumento.

• Función de Importar Clases: Si se hace uso de una clase para la cual se ha hecho previamente la declaración de importación al código, permite con un simple CRTL + SHIFT+I, resolver todos esos errores.

• Diseño Visual: Se pueden crear formularios y ventanas de forma visual, en diferentes plataformas como. Para Java SE, permite utilizar toda la librería Swing en la creación visual.

• Integración de Servidores: Como se pueden crear diferentes aplicaciones al trabajar en diferentes lenguajes, Netbeans trae en su plataforma servidores Web y de aplicaciones (su instalación es opcional).

Luego de haber realizado un estudio de los IDEs de desarrollo, se determinó utilizar el NetBeans 7.0, por sus características, por sus grandes ventajas, además por ser uno de los IDEs más utilizados en el mundo para el desarrollo de aplicaciones web y por ser una herramienta de código abierto.

1.10 Estudio de Frameworks de Formulario

A continuacion se realiza un estudio detallado de algunos de los frameworks de formularios más utilizados en el mundo, además de ser los que se encuentran en la "*Plataforma tecnológica"* por la que se rige el sistema a implementar.

jQuery

jQuery es una biblioteca de JavaScript rápida y concisa que simplifica el manejo de eventos, animación, y las interacciones Ajax para el desarrollo web rápido. Está diseñado para cambiar la forma en que se escribe JavaScript.

Este framework JavaScript, ofrece una infraestructura con la que el programador tendrá mucha mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Con jQuery se obtendrá ayuda en la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de Ajax, entre otras.

Tiene licencia para uso en cualquier tipo de plataforma, personal o comercial. Para utilizarlo solo se debe incluir en las páginas que se esté programando un script JavaScript que contiene el código de JQuery. Este script se puede descargar de la propia página web del producto (Alvares, 2009).

Sus principales ventajas son (Alvares, 2009):

Se ha ido insertando en el mercado del software por la buena aceptación por parte de los programadores.

Es un producto serio, estable, bien documentado y con un gran equipo de desarrolladores a cargo de la mejora y actualización del framework.

Posee una amplia comunidad de creadores de pluggins o componentes, lo que hace fácil encontrar soluciones ya creadas en jQuery para implementar asuntos como interfaces de usuario, galerías, votaciones, efectos diversos, entre otros.

Su principal desventaja es que aprender a trabajar con él requiere de la preparación del desarrollador, y de que tenga los conocimientos básicos del mismo.

Mootools

Mootools es un conjunto de librerías, también llamado API, que proveen clases de programación orientada a objetos en Javascript, para realizar una amplia gama de funcionalidades en páginas web, como trabajo con capas, efectos diversos, Ajax y mucho más. Con Mootools se puede programar todo tipo de scripts en el cliente rápidamente y sin preocuparse de las distintas particularidades de cada

navegador. Está especialmente indicado para programar scripts complejos, que costarían mucho más trabajo realizarlos si se partiera de cero.

Las ventajas de utilizar este interesante framework son (Alvarez, 2008):

- Ligero: el framework no pesa demasiado en Kb y el procesamiento carga poco al navegador.
- Modular: se compone de diversos módulos y el desarrollador puede seleccionar los que va a utilizar para incorporarlos en sus páginas web, dejando los otros para que no ocupen tiempo de descarga ni procesamiento.
- Libre de errores: las herramientas de Mootools funcionan perfectamente, sin emitir errores en tiempo de ejecución.
- Soportado por una amplia comunidad: existen muchos desarrolladores que lo utilizan con éxito y han creado una serie de componentes adicionales ya listos para usar, como calendarios, editores de texto, entre otros:.
- Gratuito, de código abierto y con licencia MIT: permite usarlo y modificarlo si se desea.

Sus desventajas son:

- La escasa documentación del mismo, pues la que existe es buena pero no se puede encontrar la suficiente que un desarrollador necesita. No se encuentran muchos ejemplos.
- Aprender a trabajar con el framework resulta complicado porque los ejemplos que existen del mismo son complicados.

Ext JS 3.0

Ext JS es una librería JavaScript a través de la cual se pueden construir aplicaciones complejas en internet. Esta librería incluye (Corzo, 2012):

- Componentes UI del alto performance y personalizables.
- Modelo de componentes extensibles.
- Un API fácil de usar.
- Licencias Open Source (GPL) y comerciales

Su principal ventaja radica en que permite crear aplicaciones complejas utilizando componentes predefinidos así como un manejador de layouts, similar al que brinda Java Swing. Esto ayuda a que proporcione una vasta experiencia sobre cualquier navegador, evitando el tedioso problema de validar que el código escrito funcione bien en cada uno (Corzo, 2012).

Usar un motor de render como Ext JS permite tener además estos beneficios (Corzo, 2012):

✓ *Existe un balance entre cliente y servidor*. La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo.

✓ *Comunicación asíncrona*. En este tipo de aplicación el motor de render puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente se dé cuenta.

✓ *Eficiencia de la red*. El tráfico de red puede disminuir al permitir que la aplicación elija que información desea transmitir al servidor y viceversa, sin embargo la aplicación que haga uso de la pre-carga de datos puede que revierta este beneficio por el incremento del tráfico.

En su versión 3.0 introduce varios componentes de interfaz de usuario (RowEditor, ListView, Gráficos, ButtonGroup y GroupTabs) y la mejora en muchos de los componentes existentes como por ejemplo Toolbar Overflow, Menu Overflow, AnchorTips, Buffered GridView y Depurar consola (Conran, 2009).

RowEditor permite editar rápidamente filas completas en una cuadricula. ListView hace posible con la selección, cambiar el tamaño de la columna, la clasificación y otras características DataView. El paquete Ext.char, permite visualizar los datos con gráficos basados en flash y cada gráfico se une directamente a un Ext.data.Store. ButtonGroup permite agrupar los botones de diferentes tamaños para crear barras de herramientas complejas que permite a sus usuarios a encontrar las acciones más comunes en primer lugar. Y GroupTabs representa una ayuda para la creación de diseños de portales similares a iGoogle, o que proporciona una interfaz para acceder a tareas similares con rapidez (Conran, 2009).

Una de las grandes ventajas que posee Ext JS sobre los otros frameworks es que posibilita crear “ventanas” con barras de herramientas y menúes *con* estilo de aplicaciones de escritorio, diálogos modales y eventos. Además puede ser adquirido bajo licencias Libres y Comerciales. La compañía detrás de este framework ofrece cursos de capacitación y un extenso soporte. Y en su sitio web se ofrecen ejemplos de Tabs, Ventanas, Árboles, Menúes y demás componentes de interfaz de usuario

(Tavárez, 2009).

Luego de haber realizado un estudio de los frameworks anteriormente expuestos se determinó utilizar Ext JS, por las ventajas anteriormente expuestas, además de brindarle al programador las herramientas necesarias para construir una interfaz con la calidad requerida, brindandole una ayuda con varios ejemplos y disímiles funcionalidades y eventos, que permiten que la misma sea amigable al usuario.

1.11 Análisis de las metodologías ágiles

El desarrollo de software en estos días ha tomado mucho auge. Las grandes ventajas de informatizar las instituciones de un país ayudan al hombre en su desarrollo. Las metodologías ágiles son una vía de apoyo a este proceso, pues representan una guía de cómo desarrollar un software con calidad y que sea de agrado para el cliente.

XP (Extreme Programming)

XP (Extreme Programming) se encuentra centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. Se basa en la realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Esta metodología se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (Canós, Letelier, & Penadés, 2006).

Los roles de esta metodología son programador, cliente, encargado de pruebas, encargado de seguimiento, entrenador, consultor y gestor. El programador es el encargado de escribir las pruebas unitarias y el código del sistema. El cliente escribe las historias de usuario y las pruebas funcionales para validar su implementación, además se establece la prioridad a las historias de usuarios, decidiéndose cuales se implementan en cada iteración centrándose en aportar mayor valor al negocio. El encargado de pruebas por su parte ayuda al cliente a escribir las pruebas funcionales, ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas (Canós, Letelier, & Penadés, 2006).

El encargado de seguimiento proporciona realimentación al equipo verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones y realiza el

seguimiento del progreso de cada iteración. El entrenador es el responsable del proceso global, debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente. El consultor es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas. Y el gestor es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas, su principal trabajo es de coordinación (Canós, Letelier, & Penadés, 2006).

En el ciclo de desarrollo de XP se siguen los siguientes pasos (Canós, Letelier, & Penadés, 2006):

1ro. El cliente define el valor de negocio a implementar.

2do. El programador estima el esfuerzo necesario para su implementación.

3ro. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.

4to. El programador construye ese valor de negocio.

5to. Se vuelve al 1ro.

El ciclo de vida de esta metodología cuenta con seis fases *Exploración, Planificación de la Entrega (Release), Iteraciones, Producción, Mantenimiento y Muerte del Proyecto.*

Agile UP

El Proceso Unificado Ágil, es un enfoque al desarrollo de software basado en el Rational Unified Process (RUP) de IBM. El ciclo de vida de Agile UP es serial en lo grande, esto significa que se puede mover por alguna de las cuatro fases con que cuenta (Inicio, Elaboración, Construcción y Transición) e iterativo en lo pequeño, es decir las disciplinas son ejecutadas en una forma iterativa definiendo las actividades que el equipo de desarrollo ejecuta para construir, validar y liberar software funcional, el cual cumple con las necesidades del usuario, liberando entregables incrementales en el tiempo (los equipos de proyectos siguiendo AUP entregan versiones incrementales a lo largo del tiempo) (Argüello Oviedo, y otros, 2005).

Los roles de Apile UP son el *DBA Ágil, el Modelador Ágil, Cualquiera, Administrador de la configuración, Implementador, Desarrollador, Especialista del proceso, Administrador del proyecto, Examinador, Involucrado, Documentador técnico, Administrador de pruebas, Equipo de pruebas, y el Especialista en herramientas. El DBA Ágil* es un administrador de bases de datos (DBA) que trabaja de manera colaborativa con los integrantes del equipo del proyecto para diseñar, probar y brindar soporte a los

diferentes esquemas de datos. *El Modelador Ágil* es alguien que cree y desarrolle modelos, ya sean dibujos, tarjetas, o archivos complejos realizados con herramientas CASE, de manera colaborativa y evolutiva. Los modelos ágiles son apenas lo suficientemente buenos. Cualquiera, es otra persona con un rol distinto (Argüello Oviedo, y otros, 2005).

El Administrador de la configuración se encarga de proporcionar la infraestructura y crear el medio ambiente para el equipo de desarrollo. El Implementador es responsable de poner a disposición el sistema en los ambientes de pre-producción y producción. El Desarrollador escribe el código, realiza las pruebas y construye el software. El Especialista del proceso desarrolla, adapta y apoya el material de los procesos de la organización (descripción de procesos, plantillas, guías, ejemplos, entre otros) (Argüello Oviedo, y otros, 2005).

El Administrador del proyecto administra los miembros de los equipos de trabajo, crea relaciones con los involucrados, coordina las interacciones con los involucrados, planea, administra y dispone recursos, enmarca prioridades y mantiene el equipo enfocado. El Examinador Evalúa los productos del proyecto, inclusive "el trabajo en progreso", suministrando retroalimentación al equipo de trabajo (Argüello Oviedo, y otros, 2005).

El Involucrado es cualquiera que sea usuario directo, usuario indirecto, administrador de usuarios, administrador, miembro de equipo de operación o soporte, desarrolladores que trabajan en otros sistemas que se integran o interactúan con el sistema implementado, en fin todo aquel que se vea afectado de una u otra forma con el proyecto. El Documentador técnico es responsable de producir documentación para los involucrados, tal como: materiales de capacitación, documentación de operaciones, documentación de mantenimiento, y documentación de usuario (Argüello Oviedo, y otros, 2005).

El Administrador de pruebas es el responsable del éxito de las pruebas, incluye planificar la administración, y promover las pruebas y las actividades de calidad. El Equipo de pruebas es responsable de ejecutar las pruebas y documentar los resultados que proyecten. El Especialista en herramientas es responsable de seleccionar, adquirir, configurar y brindar mantenimiento al equipo requerido (Argüello Oviedo, y otros, 2005).

En esta metodología se deben generar un grupo de entregables que se clasifican en Entregables Mínimos, Otros Productos de Trabajo del Proyecto y Los Entregables del Negocio (Argüello Oviedo, y

otros, 2005).

Entre los Entregables Mínimos están: *el Sistema*, *el Código Fuente, la Suite de Pruebas de Regresión, los Scripts de Instalación, la Documentación del Sistema, las Notas, el Modelado de requerimientos y el Modelo de Diseño* (Argüello Oviedo, y otros, 2005).

Entre los entregables del negocio están: *Modelo de la Arquitectura del Negocio*, *Orientación del Desarrollo del Negocio, Orientación de la Empresa, Estado de la Misión de la Organización, Visión de la Empresa, Guías de Recursos Humanos, Guías de Modelado, Arquitectura de Referencia y las Guías de Usabilidad* (Argüello Oviedo, y otros, 2005).

Scrum

Scrum es una metodología ágil de desarrollo de proyectos, que fue nombrada por Ikujiro Nonaka e Hirotaka Takeuchi. Su surgimiento fue como modelo para el desarrollo de productos tecnológicos, pero también se utiliza en entornos con requisitos inestables y que requieren además rapidez y flexibilidad, como suele suceder en el desarrollo de software (Palacio, 2006).

Esta metodología requiere un arduo trabajo, porque no funciona siguiendo un plan, más bien se basa en adaptarse continuamente a la evolución del proyecto. Scrum posee un grupo de características que identifican a una metodología ágil. Entre estas se encuentran (Palacio, 2006):

- Es un modo de desarrollo de carácter adaptable más que predictivo.
- Orientado a las personas más que a los procesos.
- Emplea la estructura de desarrollo ágil: incremental basada en iteraciones y revisiones.

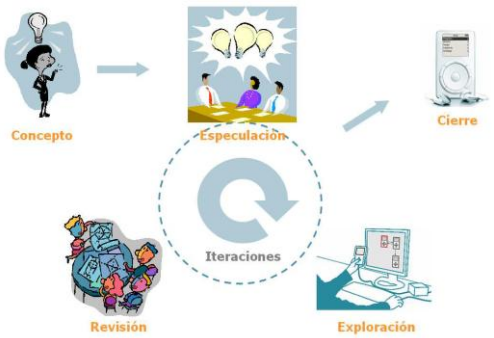

Figura 4 Estructura de un desarrollo ágil

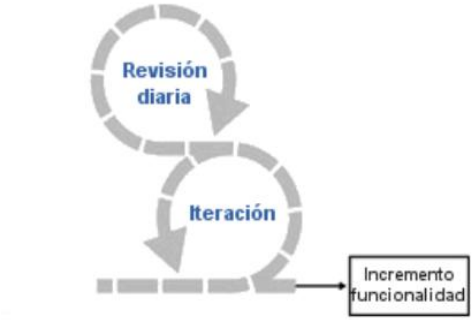

Figura 5 Estructura Central de Scrum

Scrum controla la evolución del proyecto mediante las siguientes prácticas de gestión ágil (Palacio, 2006):

Revisión de las Iteraciones: Este proceso debe tardar un periodo de 30 días, en el cual se realiza una revisión del proyecto con todas las personas implicadas en el mismo.

Desarrollo incremental: al final de cada iteración se cuenta con una parte operativa del producto, la cual puede ser evaluada.

Desarrollo evolutivo: una de las principales premisas que se tiene en Scrum es la inestabilidad, para la cual se adoptan técnicas de trabajo que permiten que el producto evolucione sin perjudicar la calidad de la arquitectura, que se va generando durante el desarrollo. Dicha metodología va generando el diseño y la arquitectura final del proyecto de forma evolutiva. Considerando que no se deben desarrollar en la primera fase del proyecto.

Auto-organización: en Scrum los equipos son auto-organizados y no auto-dirigidos, pues tienen la facultad de tomar la decisión que estimen conveniente.

Colaboración: para que un equipo funcione auto-organizado, debe existir colaboración por parte de todas las personas para con sus compañeros, sin distinción de puesto o rol. En Scrum, se denomina sprint a una iteración de desarrollo, la cual debe ser realizada en un periodo de 15-30 días.

Esta metodología se conforma por tres elementos, las reuniones, los artefactos y los roles (Palacio, 2006).

Las reuniones traen consigo la planificación de sprint, en la misma se determina cual va a ser el trabajo y

los objetivos a cumplir en esta iteración. En la reunión diaria se hace una revisión del equipo de trabajo realizado hasta la fecha y lo que se realizará el día siguiente. Y en la revisión de sprint se analiza y revisa el incremento generado (Palacio, 2006).

Scrum cuenta con tres roles que representan además a los actores de un proyecto*: propietario del producto o Product Owner, equipo de desarrollo o Scrum Team y gestor de Scrum o Scrum Manager o Scrum Master*. Cuenta además con los implicados que serían "otros interesados". Los tres primeros son los responsables del proyecto, es decir los que están comprometidos con el mismo.

El *Product Owner* se responsabiliza de obtener el mayor valor de producto para los clientes, usuarios y resto de implicados, es el que conoce y marca las prioridades del proyecto o producto. El *Scrum Team* es el grupo o grupos de trabajo que desarrollan el producto. El Scrum manager, es el gestor de los equipos que se responsabiliza por el funcionamiento de la metodología Scrum y por la productividad del equipo de desarrollo, su responsabilidad es entre otras, la de hacer de paraguas ante las presiones externas. Los otros interesados serían los que se encuentran implicados con el producto, es decir que aportan ideas para el producto pero no lo desarrollan (Dirección general, Dirección comercial, Marketing Usuarios, entre otros). (Palacio, 2006)

En Scrum se definen un grupo de artefactos para el seguimiento del proyecto y el control de las actividades, relacionadas con el sprint, como son Product Backlog, Sprint Backlog y Gráfica de progreso.

El Product Backlog no es más que un listado con los requisitos del sistema, el cual es mantenido y priorizado por el Product Owner, es un documento dinámico que incorpora constantemente las necesidades del sistema y se mantiene durante todo el ciclo de vida.

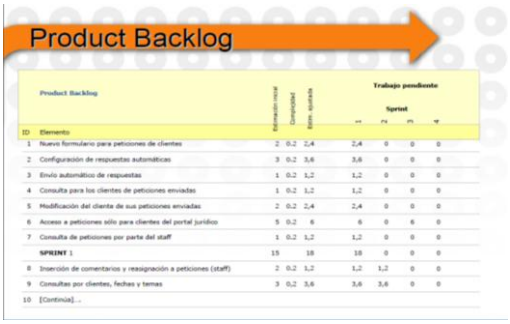

Figura 6 Product Backlog

El Sprint Backlog es una lista de tareas (realista) extraídas del Product Backlog que serán convertidas en un incremento de funcionalidad, las tareas deben tener un tiempo de duración entre 4 y 16 horas y las tareas más grandes se deben descomponer en sub-tareas de ese rango de tiempo.

SPRINT	INICIO	DURACIÓN
1	1-feb-06	12

Tareas pendientes
Horas de trabajo pendientes

PILA DEL SPRINT				
Backlog	Tarea	Tipo	Estad	Responsal
1	Descripción de la tarea 1	Análisis	Terminada	Luis
1	Descripción de la tarea 2	Prototipado	Terminada	Luis
1	Descripción de la tarea 3	Pruebas	Terminada	Luis
1	Descripción de la tarea 4	Codificación	Terminada	Elena
1	Descripción de la tarea 5	Codificación	Terminada	Elena
2	Descripción de la tarea 6	Pruebas	Terminada	Elena
2	Descripción de la tarea 7	Codificación	Terminada	Antonio
2	Descripción de la tarea 8	Codificación	Terminada	Antonio
3	Descripción de la tarea 9	Pruebas	Terminada	Antonio
3	Descripción de la tarea 10	Pruebas	En curso	Luis
3	Descripción de la tarea 11	Codificación	Pendiente	Luis

Figura 7 Sprint Backlog

La gráfica de progreso es una gráfica mediante la cual se mide el resultado de cada sprint, es decir las horas restantes por los días del sprint (Desarrollo Ágil con SCRUM).

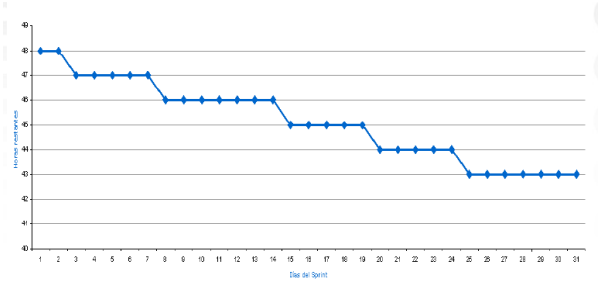

Figura 8 Gráfica de progreso

Se considera oportuno utilizar como guía para el desarrollo del sistema a implementar la metodología Scrum por ser una metodología enfocada en equipos de desarrollo pequeños, al generarse pocos artefactos se tiene la ventaja de contar con más tiempo para desarrollar el sistema y realizarle pruebas al mismo, además se caracteriza por establecer la entrega de una parte funcional del producto al cliente, cuando finaliza un Sprint, brinda la posibilidad de ajustar la funcionalidad en base a la necesidad del

negocio del cliente, proporciona una visualización del proyecto día a día, su alcance es acotado y viable y los equipos están integrados y comprometidos con el proyecto, y ante cualquier discrepancia se auto-administran. Es una de las metodologías ágiles más exitosa y representa una ayuda para organizar a las personas y el flujo de trabajo de un proyecto (Palacio, 2006).

1.12 Conclusiones del capítulo

En el presente capítulo se realizó un estudio de diferentes sistemas judiciales, tanto nacionales como internacionales, determinándose que presentan deficiencias para resolver el problema planteado en este trabajo de diploma. Se hace además un estudio de las herramientas de desarrollo que se utilizarán y el porqué.

Se puede concluir destacando la inconveniencia de aplicar al proceso de las Comisiones Disciplinarias de la Facultad 3 el Sistema desarrollado para la Fiscalía General de la República. Se acentúa la idoneidad de Scrum como Metodología Ágil de desarrollo de Software, el Visual Paradigm for UML como herramienta de modelado, PostgreSQL como gestor de bases de datos y el NetBeans 7.1 como IDE de desarrollo.

Capítulo 2: Análisis y Diseño de la propuesta de solución

2.1 Introducción al Capítulo

En el actual capítulo se aborda acerca del análisis y diseño de la solución propuesta. Tomando como punto de partida un modelo de dominio con las principales clases conceptuales que proporcionará los elementos necesarios para realizar una posterior captura de requisitos, tanto funcionales como no funcionales. Se realizará una descripción del sistema propuesto, abordando la arquitectura del mismo. También se presentaran los artefactos generados según la metodología seleccionada.

2.2 Diagrama de procesos de negocio

BPMN (Notación de Modelado de Procesos de Negocio), proporciona una forma estándar de representar procesos de negocio, tanto para propósitos descriptivos de alto nivel, como para detallados y rigurosos entornos de software orientados a procesos (White, 2009).

El diagrama de proceso, es un diagrama de flujo detallado, con la información necesaria para poder analizar el proceso y simularlo. Para una mejor comprensión del negocio a implementar se realizaron los diagramas de proceso del negocio, pertenecientes a los principales requisitos del sistema (White, 2009).

A continuación se muestra el diagrama perteneciente al requisito ¨Realizar denuncia¨.

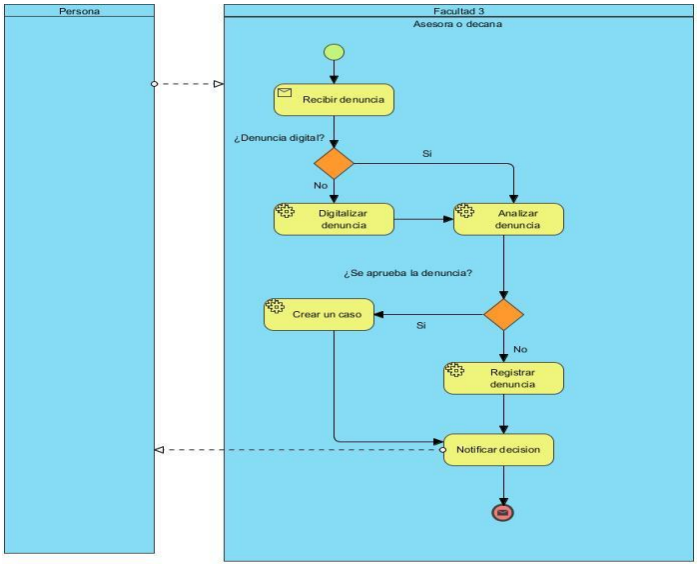

Figura 9 Diagrama de proceso del negocio ¨Realizar denuncia¨.

2.3 Requisitos del Sistema

Según el diccionario Manual de la Lengua Española, un requisito es una *"condición necesaria para una cosa"*. En ingeniería de software se le atribuyen varias definiciones también como son (Slideshare, 2010):

- *"Una condición o necesidad de un usuario para resolver un problema o alcanzar un objetivo."*
- *"Una condición o capacidad que debe estar presente en un sistema o componentes de sistema para satisfacer un contrato, estándar especificación u otro documento formal".*
- *"Una representación documentada de una condición o necesidad".*

Tomando como base las anteriores definiciones, se realizó la captura de los requisitos del sistema a implementar, clasificándolos en Funcionales y No Funcionales.

Requisitos Funcionales

RF 1-Gestionar comisiones

RF 2-Gestionar casos

RF 3-Analizar denuncias

RF 4-Realizar denuncia

RF 5-Solicitar prórroga de casos

RF 6-Solicitar detener casos

RF 7-Gestionar opiniones

RF 8-Gestionar expedientes

RF 9-Gestionar implicados

RF 10-Realizar acta de comisión

RF 11-Mostrar reportes de casos por cursos

Requisitos no Funcionales

Usabilidad: El sistema debe brindar facilidades de uso para sus usuarios, debe contar con un menú que le permita a los mismos acceder a las funciones de su interés y a las que tengan acceso. Estos tendrán además la posibilidad de interactuar con el sistema sin tener amplios conocimientos de informática, esto no incluye a los administradores del negocio, pues ellos si deberán tener conocimientos para poder manipular el software.

Rendimiento: El sistema debe ser eficiente, porque la información que se contendrá debe enviarse de manera ágil, para que el proceso de las Comisiones Disciplinarias concluya en el tiempo establecido.

Soporte: La aplicación debe ser multiplataforma. Para los servidores (aplicaciones y base de datos) se recomienda utilizar Linux por brindar más seguridad, además de las prestaciones que posee para este tipo de tareas. La aplicación debe ser programada en PHP 5.3, y con un gestor de base datos PostgreSQL 9.1 o superior.

Hardware:

Cliente: la aplicación fue desarrollada sobre la base de que las PC cliente de los usuarios puedan utilizarla con el menor requerimiento posible, es decir:

- ✓ Procesador Pentium IV o superior, 333 MHz mínimo pero recomendado 1.8 GHz
- ✓ 40 GB o más de capacidad.
- ✓ Mínimo de memoria RAM 128 DIM.
- ✓ Adaptador de Red y conectividad.

Servidor: Los servidores de aplicación y de base de datos se encontraran en un mismo servidor dado que la cantidad que datos que manejan no es grande.

CPU = 2 Dual Core AMD Opteron 64 bits a 2.2 GHz.

RAM = 1 GB.

Almacenamiento = 1 Disco SATA de 73 GB en RAID 1.

Software:

Las PC cliente de los usuarios deben tener instalados los navegadores Mozilla Firefox en su versión 3.4 o superior o Internet Explorer 7 o superior y el Sistema Operativo Windows XP o Linux. En el servidor (Aplicaciones y Base de Datos) debe tener instalado PHP 5.3 o superior, Servidor Web Apache 2.2.17 o superior y como gestor de bases de datos PostgreSQL 9.1 o superior.

Portabilidad:

El sistema debe ser multiplataforma, teniendo en cuenta usuarios que utilicen Linux o Windows.

Seguridad:

CODIS será integrado al sistema SO3 (Directorio de la Facultad), manejando este último la seguridad del mismo. En SO3 se aplica en el modelo RBAC (Modelo de Control de Acceso basado en Roles), éste es el resultado de un intento de unir los modelos DAC (Control de acceso Discrecional) y MAC (Control de Acceso Obligatorio). Con ello se consigue contar con un sistema que impone el control de accesos, pero sin las restricciones rígidas impuestas por las etiquetas de seguridad. En este modelo de control de acceso los usuarios son asignados a uno o varios roles, mientras que los privilegios y permisos se asignan a estos roles (Ferraiolo, 1995).

Las políticas de control de accesos basado en roles regulan el acceso de los usuarios a la información en términos de sus actividades y funciones de trabajo (roles), representándose así de forma natural la estructura de las organizaciones (Ferraiolo, 1995).

En el sistema a implementar, la seguridad se establecerá utilizando este modelo. Se cuenta con cuatro roles: los *administradores* (persona encargada de que funcione correctamente el sistema, es decir quien tenga acceso al código fuente para modificarlo o cambiarlo si se necesita), los *usuarios avanzados (decano(a) y* asesor(a)), los *miembros de la comisión* (personas que forman parte de una comisión, es decir, dos profesores y un estudiante) y el *usuario* (cualquiera persona de la universidad).

2.4 Descripción del Sistema Propuesto

Para darle cumplimiento al objetivo propuesto al inicio de este trabajo, se realizará una descripción del mismo, tomando como referencia la arquitectura que lo conforma, es decir los patrones con los que

cuenta, también se propondrá una forma de desplegar la aplicación, se analizará la seguridad que debe poseer la misma y se describirán los artefactos del análisis de la solución.

Arquitectura del Sistema

La arquitectura del sistema se conforma por el patrón arquitectónico MVC y algunos de los patrones de diseño GoF (General Responsibility Assignment Software Patterns o Patrones Generales de Software para Asignación de Responsabilidades), y Grasp (Gang of Four o Grupo de cuatro)), que se muestran a continuación.

Patrón Arquitectónico

El Patrón Arquitectónico es el nivel en el cual la arquitectura de software define la estructura básica del sistema, pudiendo estar relacionado con otros patrones y representa una plantilla de construcción que provee un conjunto de subsistemas aportando las normas para su organización. Ejemplo de estos patrones son: Capas, MVC, Tuberías y Filtros, Pizarra, entre otros (Fabien Potencier, 2008).

Patrón MVC

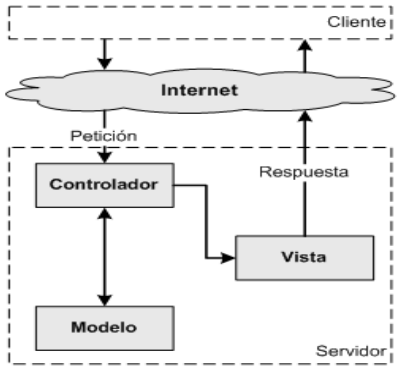

Figura 10 Patrón Modelo Vista Controlador

La utilización de un framework que se base en el patrón arquitectónico Modelo Vista Controlador (MCV), hace necesario que se divida y organice el código de acuerdo a las convenciones establecidas por el framework. El código perteneciente a la presentación se guarda en la vista, el referente a los datos en el modelo y la lógica del procesamiento de las peticiones se encuentra en el controlador. Es muy ventajosa la utilización de este patrón en una aplicación por todas las utilidades que contiene que le facilitan el

trabajo al programador. La implementación que realiza Symfony de la arquitectura MVC incluye varias clases como son (Fabien Potencier, 2008):

✓ sfController: Es la clase del controlador y se encarga de decodificar la petición y transferirla a la acción correspondiente.

✓ sfRequest: Guarda todos los elementos que integran la petición (parámetros, cookies, cabeceras, etc.)

✓ sfResponse: Posee las cabeceras de la respuesta y los contenidos. El contenido de este objeto se convierte en la respuesta HTML que se remite al usuario.

✓ El singleton de contexto (que se obtiene mediante sfContext::getInstance ()): Guarda una referencia a todos los objetos que constituyen el núcleo de Symfony y puede ser accedido desde cualquier parte de la aplicación.

Symfony se apropia de lo mejor de este patrón MVC para que el desarrollo de aplicaciones sea rápido y sencillo. En la capa del controlador, se encuentran las acciones que contienen toda la lógica de la aplicación y representan el núcleo de la misma. Dichas acciones utilizan el modelo y precisan las variables para la vista. Cuando se lleva a cabo una petición web en una aplicación Symfony, la URL define una acción y los parámetros de la petición (Fabien Potencier, 2008).

La vista a su vez genera las páginas que son mostradas como resultado de las acciones, en ella se encuentra el layout que es común para todas las páginas de la aplicación. En Symfony la vista se conforma por varias partes, que a su vez se encuentran preparadas para ser transformadas por la persona que normalmente trabaja con cada aspecto del diseño de las aplicaciones (Fabien Potencier, 2008).

El modelo se compone por las clases, que se generan automáticamente según la estructura de la base de datos. En Symfony, el acceso y la modificación de los datos que se almacenan en la base de datos, se realiza mediante objetos. Propel es el motor generador que se encarga de esta generación automática para construir sus clases, creando la estructura y generando el código de las mismas(Fabien Potencier, 2008).

Con la evolución de un proyecto puede ser necesario agregar métodos y propiedades personalizadas en los objetos del modelo, lo que trae como consecuencia que aumenten las tablas o columnas. Y cada vez

que se realice alguna modificación se deben regenerar las clases del modelo de objeto (Fabien Potencier, 2008).

Patrones de Diseño

Los patrones de diseño surgen por la necesidad de transmitir experiencia. Esta es precisamente la diferencia entre un programador brillante experto y un programador brillante e inexperto. Los programadores que conocen estos patrones son capaces de identificar las situaciones en las que se pueden aplicar los mismos y utilizarlos, sin tener que analizar el problema. Dichos patrones se clasifican en dos tipos, GRASP y GoF.

Patrones GRASP:

Los patrones GRASP, son considerados más que patrones, como una serie de buenas prácticas de aplicación recomendable en el diseño del software (Ecured, 2009)**.**

Experto: En el patrón experto, la responsabilidad de realizar una tarea es de la clase que tiene o puede tener datos involucrados. El mismo representa uno de los patrones más utilizados por Symfony, como por ejemplo en la inclusión de las librerías Propel para mapear la Base de Datos. Esta librería es utilizada para realizar su capa de abstracción en el modelo, encapsular toda la lógica de los datos y generar las clases con todas las funcionalidades comunes de las entidades, las clases de abstracción de datos (Peer del Modelo) poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen la información necesaria de la tabla que representan (Canales Mora, 2003) (Ecured, 2009)**.**

Creador: En este patrón se asigna la responsabilidad de que una clase B cree un objeto de la clase A, solamente cuando: B contiene a A, B es una agregación (o composición) de A, B almacena a A, B tiene los datos de inicialización de A (datos que requiere su constructor) y B usa a A. En Symfony se evidencia esto con la clase Actions. En la misma se encuentran las acciones definidas para el sistema y se ejecutan en cada una de ellas. En estas acciones se crean los objetos de las clases que representan las entidades. Esto demuestra que la clase Actions es creador de dichas entidades. Algunas de estas funciones son utilizadas en la clase Actions son: doSelect (), retrieveByPK (), doSelectOne () (Canales Mora, 2003) (Ecured, 2009).

Alta Cohesión: Este patrón se basa en que cada elemento del diseño debe realizar una labor única dentro del sistema, que no sea desarrollada por el resto de los elementos y además lo identifique. El mismo está presente en Symfony, puesto que este framework permite organizar el trabajo tomando como referencia la estructura del proyecto y la asignación de responsabilidades con una alta cohesión. Un ejemplo de ello es la clase Actions, la cual está formada por varias funcionalidades que están estrechamente relacionadas, siendo la misma la responsable de definir las acciones para las plantillas y colaborar con otras para realizar diferentes operaciones, instanciar objetos y acceder a las properties (Canales Mora, 2003) (Ecured, 2009).

Bajo Acoplamiento: En este patrón deben existir pocas dependencias entre las clases. Ejemplo de ello en Symfony lo representa la clase *actions*, que solamente hereda de la clase *sfAction*, esto hace posible que el acoplamiento sea bajo. Como se explicó anteriormente en Symfony se separan las capas del modelo, la vista y el controlador, por lo que las clases del negocio y la de acceso a datos están en el modelo y no se asocian con las de la vista y el controlador, con ello también existe baja dependencia, resultando bajo el acoplamiento (Canales Mora, 2003) (Ecured, 2009).

Controlador: Con este patrón es posible asignar la responsabilidad de controlar el flujo de eventos del sistema a clases específicas. Con esto se hace más fácil la centralización de las actividades. Es importante destacar que el controlador no realiza estas actividades sino que las delega en otras clases, con las que mantiene un modelo de *Alta Cohesión.* En Symfony, todas las peticiones Web, las maneja un solo controlador frontal (sfActions), el cual representa un único punto de entrada de toda la aplicación, en un determinado entorno. Otros ejemplos de donde se utiliza este patrón en Symfony son en las clases sfFrontController, sfWebFrontController, sfContex, los “actions” y el index.php del ambiente (Canales Mora, 2003) (Ecured, 2009).

Patrones GOF:

Los patrones GoF llamados así porque son cuatro autores del libro Design Patterns son los patrones de diseño, en el campo del diseño Orientado a Objetos más utilizados en la actualidad. (Angel, 2011) (Ecured, 2009).

Singleton (Instancia única): Este patrón garantiza que exista una sola instancia para una clase y crea un mecanismo de acceso global a dicha instancia. En el framework Symfony se puede evidenciar este patrón en la clase sfRouting específicamente en el método getInstance. Esta clase la utiliza el

controlador frontal (sfWebFrontController) y se encarga de enrutar todas las peticiones que se hagan a la aplicación. El singleton sfRouting precisa otros métodos muy útiles para la gestión manual de las rutas: ClearRoutes (), hasRoutes (), getRoutesByName () (Angel, 2011) (Ecured, 2009).

2.5 Artefactos

Product Backlog o Lista de Requisitos

En el Product Backlog se encuentran enumerados los requisitos del sistema. El Product Owner o Propietario del Producto es el responsable de priorizar y mantener este artefacto .

ID	Estimado	Requisitos Funcionales	Requisitos Asociados
1	**6**	**Gestionar comisiones**	**Adicionar comisiones**
2	**2**		**Adicionar miembro comisión**
3	**1**		**Eliminar comisiones**
4	**1**		**Buscar comisiones**
5	**2**		**Listar miembros por comisión**
6	**5**	**Gestionar casos**	**Adicionar caso a comisión**
6	**5**		**Modificar casos**
7	**2**		**Eliminar casos**
8	**1**		**Buscar casos**
9	**1**		**Listar casos**
10	**2**		**Modificar porciento caso**
11	**7**	**Analizar denuncias**	

12	**2**	**Realizar denuncia**	
13	**2**	**Solicitar prorroga de casos**	
14	**3**	**Solicitar detener casos**	
12	**1**	**Gestionar opiniones**	**Adicionar opiniones**
13	**1**		**Gestionar cargos personas**
14	**2**		**Listar opiniones por denunciado**
15	**4**	**Gestionar expedientes**	**Adicionar expedientes**
16	**2**		**Modificar expedientes**
17	**1**		**Archivar expedientes**
18	**1**		**Listar expedientes**
19	**2**	**Gestionar implicados**	**Adicionar involucrados**
20	**5**		**Modificar involucrados**
21	**4**		**Eliminar involucrados**
22	**3**		**Listar involucrados**
23	**6**	**Realizar acta de comisión**	
24	8	**Mostrar reportes de casos por cursos.**	

Tabla 1 Product Backlog

Tiempo total: 81 días **Total de requisitos: 28**

Diseño del modelo de datos

A continuación se muestra el diagrama entidad-relación del sistema CODIS con las clases y atributos que se cuentan en la misma.

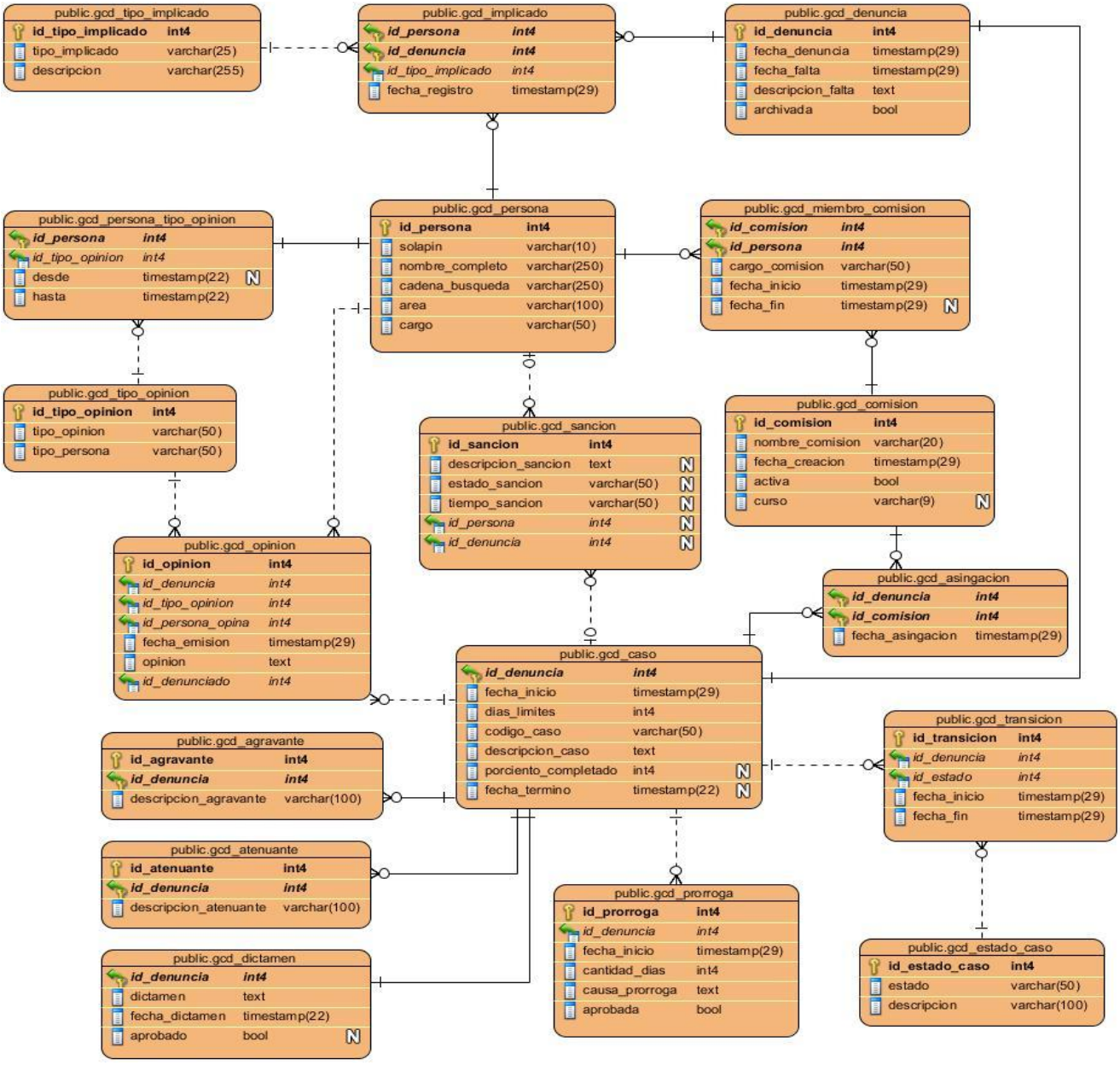

Figura 11 Diagrama entidad relación del sistema CODIS

Conclusiones del capítulo

En el desarrollo del capítulo se describieron detalladamente los elementos a tratar a lo largo del análisis y diseño de la solución planteada. Se evidencia desde el levantamiento de requisitos funcionales y no funcionales que significan una parte fundamental para la posterior implementación del sistema, hasta el diseño de la base de datos y de las interfaces visuales con las que tendrá interacción el usuario. Con este capítulo se pretende lograr un mejor entendimiento del sistema por parte de los futuros usuarios del mismo.

Capítulo 3: Implementación y pruebas

3.1 Introducción al capítulo

En el tercer y último capítulo de este trabajo de diploma se realizará la implementación y pruebas del sistema. Se mostrará el diagrama de componentes. Se presentarán los Sprint para tener evidencia de cómo va el desarrollo del sistema. Además se describirán las interfaces de usuario. Posteriormente, al concluir la implementación del sistema se le realizarán pruebas funcionales y unitarias al mismo.

3.2 Diagrama de componentes

Un diagrama de componentes describe los elementos físicos de un sistema y las relaciones entre los mismos. Estos elementos representan todos los tipos de elementos software que entran en la fabricación de aplicaciones informáticas. Ejemplos de estos son los archivos, paquetes, librerías, entre otros.

La representación gráfica de un componente es la siguiente:

Figura 12 Representación gráfica de un componente

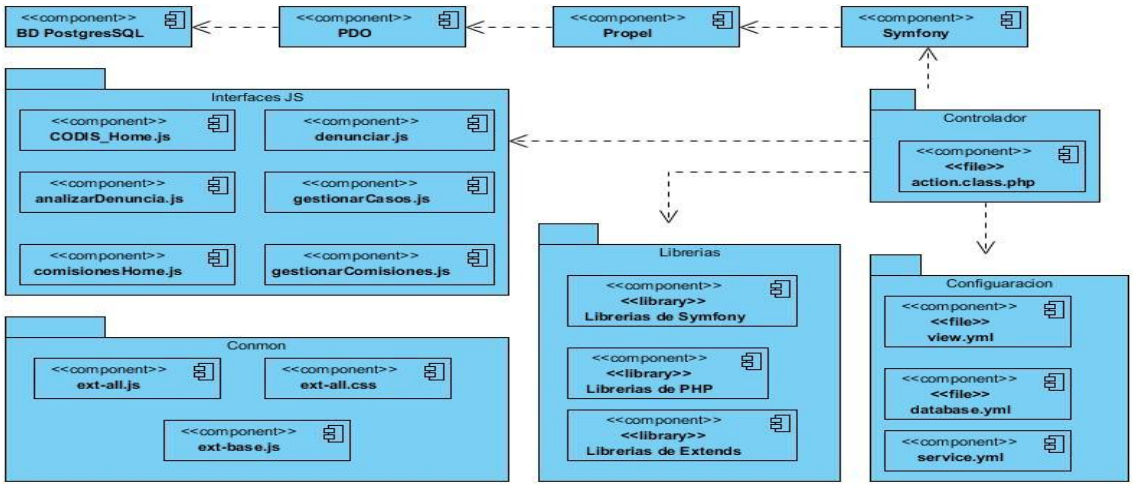

Figura 13 Diagrama de Componentes

3.3 Sprint Backlog/ Pila de tareas del Sprint

El Sprint es la lista de tareas que descompone las funcionalidades mostradas anteriormente en el Product Backlog. En el mismo se determina el estado de cada tarea, así como el tiempo estimado y real en horas, para la culminación de la misma. Una de las principales ventajas que trae esto consigo es que con la culminación de un Sprint, se le entregará al cliente una parte funcional del producto. Las tareas deben contar con un tiempo de 4 a 16 horas, las que tengan más de 16, deberán descomponerse en tareas más pequeñas. Con Scrum se tiene también la posibilidad de revisar diariamente como va avanzando el producto y así determinar si existe atraso o no. A continuación se presentan las planificaciones de los Sprint que han sido creados para el desarrollo del sistema.

Pila del Sprint I				
Backlog	Tarea	Estado	Estimado	Real
	Diseño y generación de la base de datos	100%	6	5
	Implementación de la capa de acceso a datos.	100%	15	10

Tabla 2: Pila del Sprint I

Luego de culminar este Sprint, se cuenta con un avance de un 20% de la implementación del sistema, puesto que el diseño y generación de la base de datos constituye un parte imprescindible de la misma para poder realizar las posteriores consultas que se necesitaran.

Pila del Sprint II				
Backlog	Tarea	Estado	Estimado	Real
	Implementación de la interfaz de usuario “Denunciar”	100%	5	6
	Implementación de la interfaz de usuario “Analizar denuncia”	100%	4	4

	Implementación de la interfaz de usuario “Gestionar casos”.	100%	3	3
	Implementación de la interfaz de usuario “Comisiones Home”.	100%	7	8
	Implementación de la interfaz de usuario “Gestionar comisiones”	100%	6	5
	Implementación de la interfaz de usuario “CODIS Home”	100%	10	10

Tabla 3: Pila del Sprint II

Con la culminación del segundo Sprint, se tiene un avance de un 40% de la implementación del sistema pues ya se cuenta con las interfaces necesarias para la posterior implementación de los requisitos funcionales presentados.

3.4 Interfaces de Usuario

Se muestran a continuación una selección de las interfaces de usuarios del sistema y se brinda una breve explicación de cada una para un mejor entendimiento del sistema. El usuario que accede al sistema luego de loguearse, en dependencia de su rol accederá o no a los servicios que se brindan.

Interfaz de usuario “Denunciar”

La persona que desee denunciar a un estudiante (es) de la Facultad 3, cuando selecciona la opción “Denunciar”, se le muestra una ventana en la cual debe introducir su solapín.

Figura 14 Interfaz de usuario Denunciar, introduciendo el solapín

Luego cuando de presionar el botón buscar, se cargará la interfaz Denunciar con los datos de la persona que desea denunciar. De ahí busca al o los estudiantes que desea denunciar. Si solo desea denunciar a uno, debe seleccionar el radio “Denunciar estudiante”, donde puede buscarlo por el solapín, si desea denunciar a varios estudiantes de distintos grupos, selecciona el radio denunciar grupo y busca a los estudiantes por grupo y si desea denunciar a un grupo entero, los selecciona a todos.

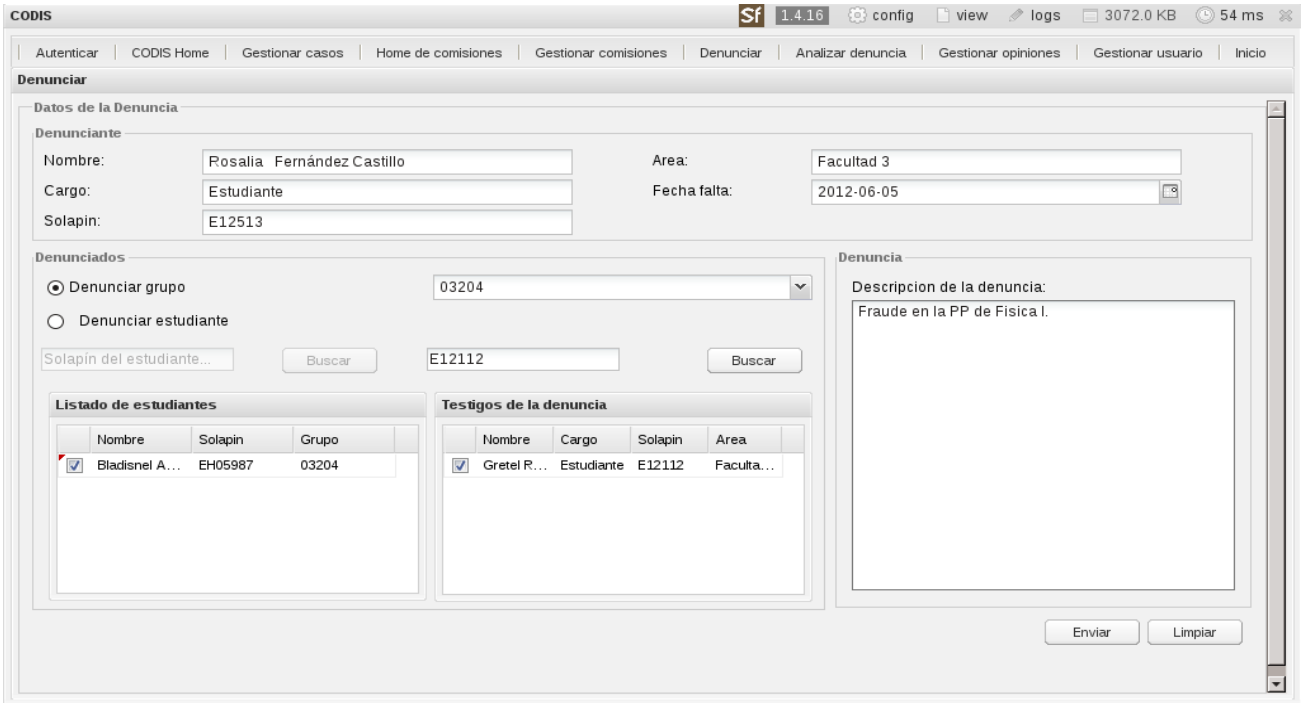

Figura 15 Interfaz de usuario Denunciar

Interfaz de usuario “Analizar denuncia”

La decana o asesora selecciona cuál de las denuncias realizadas va a analizar y luego determina si procede o no, en caso de que proceda se convierte en un caso.

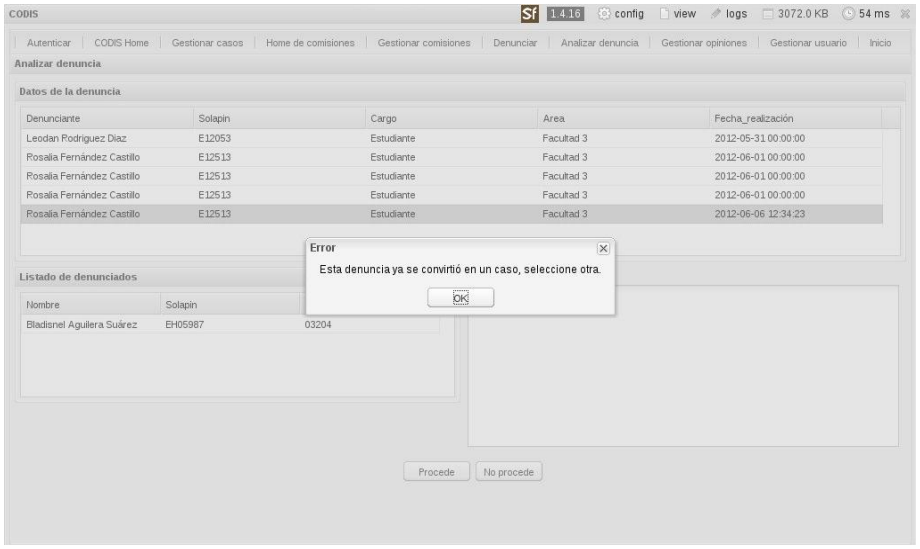

Figura 16 Interfaz de usuario Analizar denuncia

Si una denuncia no procede es archivada en la base de datos.

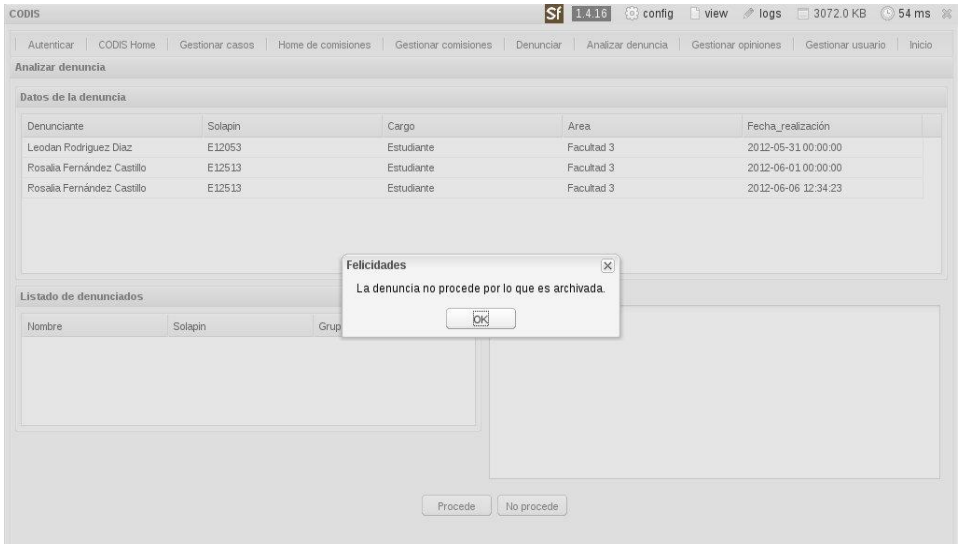

Figura 17 Interfaz de usuario Analizar denuncia, cuando no procede

3.5 Validación de la solución

La industria del software en estos tiempos se encuentra muy revolucionada, las empresas cada vez desarrollan sistemas más complejos y de gran tamaño, siendo también mayor el riesgo de que presenten

errores en cuanto al cumplimiento de los requisitos funcionales, viéndose afectadas las necesidades de los clientes y de esta forma es poco probable asegurar que un software sea seguro y sin errores.

Para comprobar que un software es óptimo existen diversas formas y métodos. Para validar la solución propuesta se aplican distintas métricas de diseño, para poder valorar en qué medida se garantiza la calidad y complejidad de la solución.

Métricas de diseño

Estas métricas se enfocan en las características internas de los componentes de software, contando con medidas que pueden ayudar al desarrollador a valorar la calidad del diseño (Driggs Vélez, 2011).

Las métricas se basan en tratar con datos cuantitativos tanto la complejidad, como la funcionalidad y la eficiencia, inmersas en el desarrollo del software. Sus objetivos se dirigen a mejorar la comprensión de la calidad del producto, a estimar la efectividad del proceso y a mejorar la calidad del trabajo. Los atributos de calidad que utilizan las métricas de diseño son (Driggs Vélez, 2011):

Atributos	Significado
Responsabilidad	Es la responsabilidad que se le asigna a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
Complejidad de implementación	Es el grado de dificultad que conlleva realizar un diseño de clases determinado.
Reutilización	Es el grado de reutilización con que cuenta una clase o estructura de clase dentro del diseño.
Acoplamiento	Es el grado de dependencia o interconexión de una clase o estructura de clases con otras, se encuentra vinculada a las características de la reutilización.
Complejidad de mantenimiento	Es el grado de esfuerzo que se necesita para realizar un arreglo, una mejora o una rectificación de algún error de un diseño de software. Puede influir indirecta, pero fuertemente en los costos y la planificación del proyecto.

Cantidad de pruebas	Es el grado de esfuerzo para realizar las pruebas de calidad, (Unidad) del producto (componente, módulo, clase, conjunto de clases, etc.) diseñado.

Tabla 4 Atributos de calidad que utilizan las métricas de diseño

Las métricas que serán utilizadas para medir la calidad del diseño del sistema CODIS son:

Tamaño operacional de clase (TOC): el tamaño operacional de una clase esta dado por el número de funcionalidades con que cuenta la misma y los atributos de calidad que evalúa son (Driggs Vélez, 2011):

Tamaño Operacional de clase (TOC)	
Atributos	**Afectación**
Responsabilidad	Si aumenta el TOC, aumenta la responsabilidad que se le asigna a la clase.
Complejidad de implementación	Si aumenta el TOC, aumenta la complejidad de la implementación de la clase o clases.
Reutilización	Si aumenta el TOC, disminuye el grado de reutilización de la clase.

Tabla 5 Atributos de la métrica Tamaño operacional de clase (TOC)

Para evaluar los atributos de esta métrica se establece un rango de valores que ayudan a determinar la complejidad en la aplicación (Driggs Vélez, 2011).

	Categoría	Criterio
Responsabilidad	Baja	< =Prom.
	Media	Entre Prom. y 2* Prom.
	Alta	> 2* Prom.

Complejidad implementación	Baja	< =Prom.
	Media	Entre Prom. y 2* Prom.
	Alta	> 2* Prom.
Reutilización	Baja	> =Prom.
	Media	Entre Prom. y 2* Prom.
	Alta	< 2* Prom.

Tabla 6 Rango de valores para evaluar técnicamente los atributos de calidad de la métrica TOC.

Luego de evaluar esta métrica se obtuvieron los siguientes resultados:

En el sistema se cuenta con un 90% de clases que cuentan con entre 1 y 5 procedimientos, un 5% de clases que cuentan con entre 6 y 10 procedimientos, y un 5% que cuenta con más de 26 procedimientos.

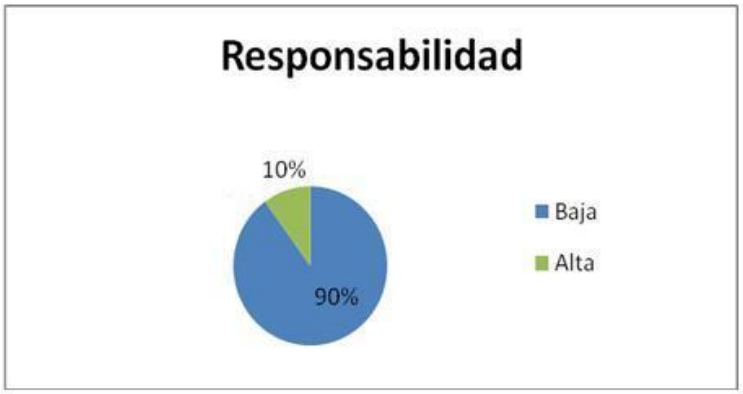

Figura 18 Responsabilidad de las clases

La responsabilidad que presentan las clases del sistema se puede concluir diciendo que es baja, porque un 90% de las mismas cuenta con entre 1 y 5 procedimientos y solo un 10% tiene responsabilidad alta.

Figura 19 Complejidad de las clases

La complejidad de las clases es baja porque el 90% de las mismas presenta una complejidad baja y solo un 10% presenta complejidad alta.

Figura 20 Reutilización de las clases

La reutilización de las clases es alta, porque un 90% de las clases presenta una reutilización alta y un 10% presenta reutilización baja.

Relaciones entre clases (RC): es el número de relaciones de uso de una clase con otra. Esta métrica evalúa los siguientes atributos de calidad (Driggs Vélez, 2011).

Relaciones entre clases (RC)

Atributos	Afectación
Acoplamiento	Un aumento del RC implica un aumento del Acoplamiento de la clase.
Complejidad de mantenimiento	Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
Reutilización	Un aumento del RC implica una disminución en el grado de reutilización de la clase.
Cantidad de pruebas	Un aumento del RC implica un aumento de la Cantidad de pruebas de unidad necesarias para probar una clase.

Tabla 7 Atributos de la métrica Relaciones entre clases (RC)

Para evaluar los atributos de esta métrica se establece un rango de valores que ayudan a determinar la complejidad en la aplicación (Driggs Vélez, 2011).

	Categoría	Criterio
Acoplamiento	Ninguno	0
	Bajo	1
	Medio	>2

	Categoría	Criterio
Complejidad Mant.	Baja	<= Prom.
	Media	Entre Prom. y 2*Prom.
	Alta	> 2*Prom.

	Categoría	Criterio
Reutilización	Baja	>2* Prom.
	Media	Entre Prom. y 2*Prom.
	Alta	<= Prom.
	Categoría	Criterio
Cantidad de Pruebas	Baja	<= Prom..
	Media	Entre Prom. y 2*Prom.
	Alta	> 2*Prom.

Tabla 8 Rango de valores para evaluar técnicamente los atributos de calidad de la métrica Relaciones entre clases (RC).

Luego de evaluar esta métrica se obtuvieron los siguientes resultados:

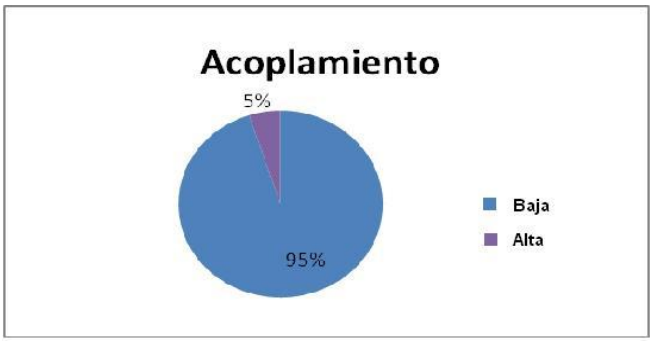

Figura 21 Acoplamiento de las clases

El acoplamiento es bajo porque un 95% de las clases presenta un bajo acoplamiento y solo un 5% presenta un alto acoplamiento.

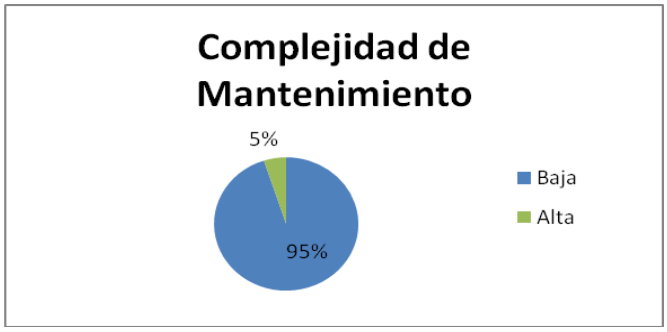

Figura 22 Complejidad de mantenimiento de las clases

La complejidad de mantenimiento es baja pues un 95% de las clases presenta una baja complejidad de mantenimiento, mientras un 5%, alta.

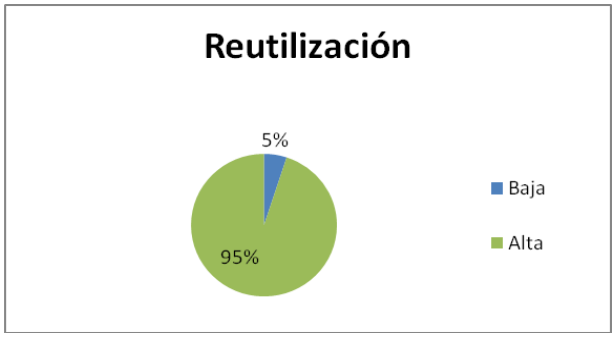

Figura 23 Reutilización de las clases del sistema

La reutilización de las clases del sistema es alta, porque el 95% de las clases presenta una alta complejidad, mientras que un 5%, alta.

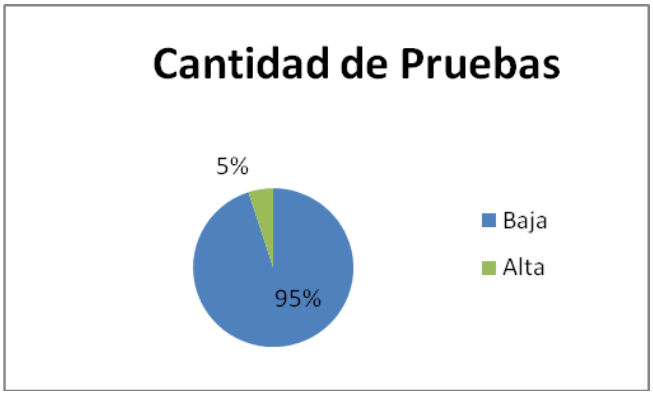

Figura 24 Cantidad de pruebas que se le pueden hacer al sistema

La cantidad de pruebas que se le pueden hacer al sistema es baja porque un 95% de sus clases presenta un criterio bajo para realizar pruebas.

3.6 Pruebas de software

Una de las fases más importantes en el desarrollo de un software es la de pruebas, pues con el desarrollo de las mismas se podrá valorar la calidad de un producto y determinar si posee o no la calidad requerida. La realización de una prueba se basa en ejecutar un programa en busca de errores. Para que un caso de prueba sea satisfactorio debe tener una alta probabilidad de mostrar un error que no haya sido descubierto hasta el momento. El principal objetivo de una prueba de software es diseñar casos de prueba, que muestren sistemáticamente diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo posible (Pressman, 2005).

Pruebas de caja negra

Las pruebas de caja negra son pruebas funcionales. Las mismas se aplican sobre el sistema empleando un determinado conjunto de datos de entrada y observando las salidas que se producen para determinar si la función se está desempeñando correctamente por el sistema bajo prueba. Las herramientas básicas son observar la funcionalidad y contrastar con la especificación (Pressman, 2005).

Al realizar pruebas de caja negra a un sistema se intenta encontrar errores en las siguientes categorías (Pressman, 2005):

- Funciones Incorrectas o Ausentes.
- Errores de Interfaz.
- Errores en estructuras de datos o acceso a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

A continuación se muestran dos diseños de casos de pruebas realizados al sistema CODIS.

Requisito funcional: *"Gestionar casos"*

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Adicionar datos del caso.	Adicionar la información del caso en el SGBD PostgreSQL.	El sistema muestra la interfaz “Analizar denuncia”, que contiene un grid con todas las denuncias realizadas hasta el momento. El sistema adiciona el nuevo caso y muestra el mensaje "La denuncia procede por lo que se convertirá en un nuevo caso."	Seleccionar la opción ¨Analizar denuncia¨ y seleccionar una de las denuncias que se encuentran en el grid Denuncias¨. Se presiona botón Procede y se inserta el caso.
EC 1.2 Adicionar datos del caso ya existente en SGBD	Adicionar la información del caso en el SGBD PostgreSQL.	El sistema muestra la interfaz “Analizar denuncia”, que contiene un grid con todas las denuncias realizadas hasta el momento. El sistema muestra el mensaje "Esta denuncia ya se convirtió en un caso, seleccione otra.”	Seleccionar la opción ¨Analizar denuncia¨ y seleccionar una de las denuncias que se encuentran en el grid Denuncias¨. Se presiona botón Procede y se muestra mensaje.

Tabla 9 Escenario: Adicionar datos del caso

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.3 Modificar información “porciento completado” de caso.	Adicionar la información del caso en el SGBD PostgreSQL	El sistema muestra la interfaz “Gestionar casos”. El sistema modifica el dato porciento y se muestra el mensaje “Se modificó el	Seleccionar la opción ¨Gestionar casos¨ y seleccionar una de las comisiones que se encuentran en el grid Comisiones¨. Al mostrarse un grid con los casos que tiene la

		porciento exitosamente".	comisión seleccionada, se da doble clic en campo "porciento", se selecciona un dato y se presiona el botón "Guardar cambios".

Tabla 10 Escenario: Modificar datos del caso.

Requisito funcional: Gestionar comisiones.

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Adicionar datos de la comisión introduciendo campos validos	Adicionar la información de la comisión, en el SGBD PostgreSQL.	El sistema muestra la ventana "Adicionar comisión", que contiene el formulario. El sistema adiciona la nueva comisión, cierra la ventana "Adicionar comisión" y muestra el mensaje "La comisión fue insertada satisfactoriamente."	Seleccionar la opción ¨Gestionar comisiones¨ y la acción "Adicionar comisión". Dejar campos en blanco y presiona botón Adicionar.
EC 1.2 Adicionar datos de comisión introduciendo campos en blanco		El sistema muestra la ventana "Adicionar comisión", que contiene el formulario. El sistema muestra el mensaje "Error en el formulario, revise si existe algún campo en blanco."	Seleccionar la opción ¨Gestionar comisiones¨ y la acción "Adicionar comisión". Dejar campos en blanco y presiona botón Adicionar.
EC 1.3 Cancelar		El sistema cierra la ventana "Adicionar comisión"	Seleccionar la opción ¨Gestionar comisiones¨ y la acción "Adicionar comisión" y presiona

			botón Cancelar.

Tabla 11 Escenario: Adicionar datos del caso.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.4 Eliminar datos de la comisión.	Eliminar información almacenada en SGBD de la Comisión.	El sistema muestra la ventana “Eliminar comisión”. El sistema elimina la comisión seleccionada, cierra la ventana “Eliminar comisión” y muestra el mensaje "La comisión ha sido eliminada."	Seleccionar la opción ¨Gestionar comisiones¨ y la acción “Eliminar comisión”. Se muestra la ventana “Eliminar comisión” y presiona el botón Aceptar
EC 1.5 Cancelar		El sistema cierra la ventana “Eliminar comisión”.	Seleccionar la opción ¨Gestionar comisiones¨ y la acción “Eliminar comisión” y presiona botón Cancelar.

Tabla 12 Escenario: Eliminar datos de la comisión.

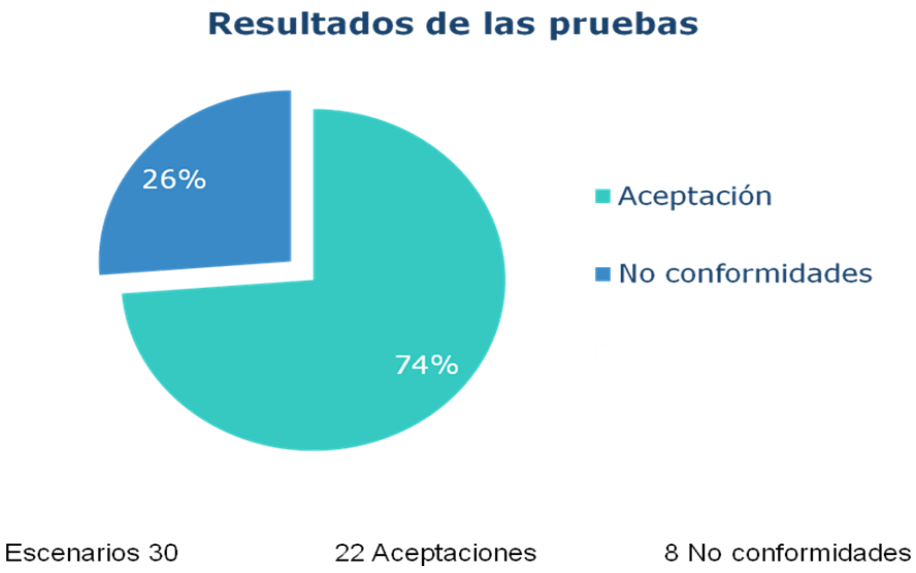

Figura 25 Resultados de las pruebas de Caja Negra

Luego de realizar las pruebas de caja negra, se encontraron un grupo de no conformidades, referentes a validaciones en la introducción de datos y errores de interfaz, siendo corregidas en el momento de la ejecución, por lo que se puede afirmar que el sistema cuenta con la calidad requerida y no presenta no conformidades.

Pruebas de caja blanca

Las pruebas de caja blanca o de caja de cristal como también suelen llamarse, se realizan sobre las funciones internas de un software. Estas sirven de ayuda al ingeniero de software para que el mismo pueda desarrollar casos de pruebas que garanticen cuatro puntos importantes (Pressman, 2005):

1. Que todas las rutas independientes dentro del módulo se han realizado por lo menos una vez.
2. Que se ejerciten los lados verdadero y falso de todas las decisiones lógicas.
3. Que se ejecuten todos los bucles en sus límites y dentro de sus límites operacionales.
4. Que se ejerciten estructuras de datos internos para asegurar su validez.

La técnica de prueba de caja blanca que se determinó utilizar, es la prueba de la *ruta básica*. Mediante la misma se puede obtener una medida de complejidad lógica de un diseño procedimental y puede utilizarse

dicha medida como guía para definir un conjunto básico de rutas de ejecución (Pressman, 2005).

Para el trabajo con la técnica anteriormente presentada, debe mostrarse una notación simple para la representación del flujo de control, llamado gráfica de flujo (o gráfica del programa), que describe un flujo de control lógico empleando las notaciones que se muestran a continuación (Pressman, 2005):

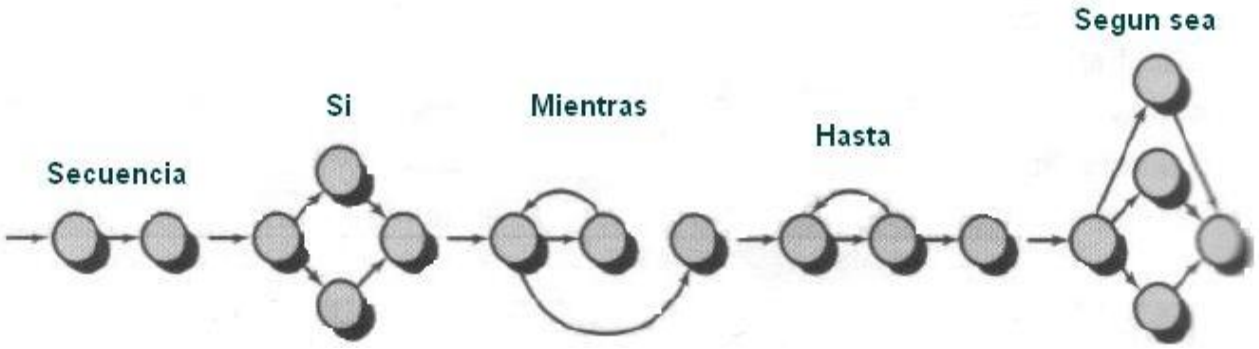

Figura 26 Gráfica de flujo para las instrucciones Secuenciales, If, While y Case

En la siguiente figura se pueden observar, el significado de cada elemento representado en la gráfica. Los círculos también nombrados *nodos de gráficas de flujo,* representan una o más instrucciones procedimentales. Las flechas de color negro son las *aristas o enlaces*, estas simbolizan el flujo de control y son semejantes a las flechas de los diagramas de flujo. Una arista siempre tiene que terminar en un nodo, aunque este no represente ninguna instrucción procedimental. Las áreas que delimitan aristas y nodos se denominan *regiones* y estas incluyen las áreas que están fuera de la gráfica (Pressman, 2005).

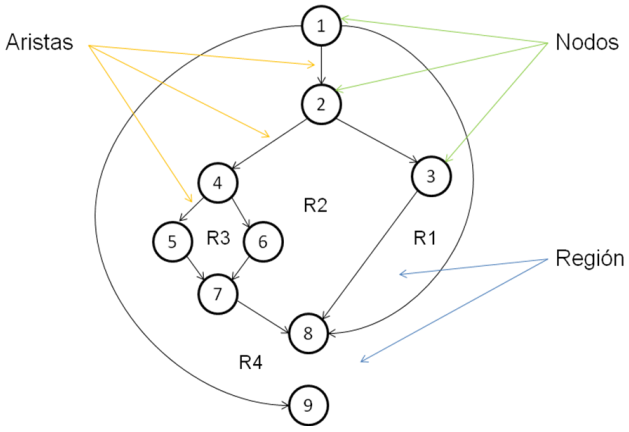

Figura 27 Representación de nodos, aristas y regiones

La generación de una gráfica de flujo se torna más complicada cuando se tienen condiciones compuestas. Una condición de este tipo ocurre cuando hay uno o más operadores booleanos (OR, AND, NAND, NOR) en una instrucción condicional (Pressman, 2005).

Un *nodo predicado* es un nodo que contiene una condición y su principal característica es que de él emanan dos o más aristas (Pressman, 2005).

Una *ruta independiente*, es cualquier ruta del programa que ingresa por lo menos un nuevo conjunto de instrucciones de procesamiento o una nueva condición. Vista desde la gráfica de flujo, una ruta independiente debe recorrer por lo menos una arista que no se halla recorrido anteriormente. Si se diseñan casos de prueba para cada una de estas rutas, se habrán ejecutado los lados verdadero y falso de cada instrucción del programa (Pressman, 2005).

Para conocer cuantas rutas se deben buscar se utiliza la *complejidad ciclomática*. La complejidad ciclomática constituye una métrica de software que proporciona una medida cuantitativa de la complejidad lógica de un programa. El valor obtenido del cálculo de la complejidad ciclomática define el número de rutas independientes en el conjunto básico de un programa y el límite superior para el numero de pruebas que deben aplicarse para asegurar que todas las instrucciones se hayan ejecutado por lo menos una vez (Pressman, 2005).

Formas de calcular la complejidad ciclomática (Pressman, 2005):

- El número de regiones corresponde a la complejidad ciclomática.
- La complejidad ciclomática, V (G), de una gráfica de flujo G, se define como V (G)= E-N+2, donde E es el número de aristas y N el número de nodos de la gráfica de flujo G.
- La complejidad ciclomática, V (G), de una gráfica de flujo G, se define como V (G) = P+1, donde P es el número de nodos predicados incluidos en la gráfica de flujo G.

Resultados obtenidos de las pruebas

A continuación se muestran los resultados obtenidos de la aplicación de las pruebas anteriormente analizadas correspondientes a la funcionalidad InsertarComision () de la clase action, que se encarga de insertar una comisión en la base datos.

```
public function executeInsertarComision(sfWebRequest $request) {
      $nombre = $request->getParameter('nombre_comision');
      $fecha_creacion = date('Y-m-d H:i:s');
 1    $fecha = array();
      $fecha [] = $fecha_creacion;
      $activa = TRUE;
      $curso = $request->getParameter('curso');

2    if (GcdComisionPeer::insertarComision($nombre, $fecha_creacion,    $activa, $curso)) {
         3 return $this->renderText(json_encode(array("success" => true, "msg" => "Se ha
insertado la comision")));
      }
4     return $this->renderText(json_encode(array("success" => false, "msg" => "No ha podido
insertarse la comision, ocurrio un error")))
5 }
```

Figura 28 Funcionalidad InsertarComision ()

Luego de determinar los nodos que se encuentran en la funcionalidad se procede a construir la gráfica de flujo correspondiente. Importante destacar que el nodo 5 es el que culmina la funcionalidad, es decir no representa un procedimiento por eso es distinto a los demás.

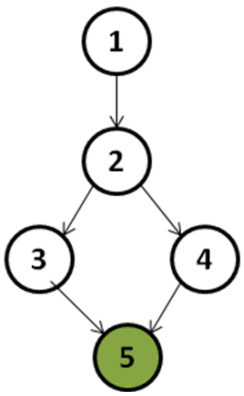

Figura 29 Gráfico de flujo correspondiente al método InsertarComision ().

Para conocer la cantidad de rutas independientes que se van a buscar, debe calcularse primero la complejidad ciclomática.

1. Número de regiones encontradas= 2
2. V (G)= E-N+2

   V (G)= 5-5 +2

V (G)= 2

3. V (G)= P+1

V (G)= 1+1

V (G)= 2

Se tienen solamente dos rutas independientes:

Primera ruta: 1, 2, 3, 5.

Segunda ruta: 1, 2, 4, 5.

A continuación se detallan los casos de prueba a realizar para comprobar que los resultados obtenidos concuerden con los resultados esperados y de este modo evaluar el funcionamiento del método InsertarComision ().

Se determinaron para el diseño de los casos de prueba los siguientes parámetros a medir:

Descripción: una breve reseña de lo que debe hacer la funcionalidad.

Entrada: son los parámetros que debe recibir la funcionalidad.

Salida: respuesta que devuelve el sistema, al ejecutarse la funcionalidad.

Resultados esperados: resultado que se espera retorne la funcionalidad ejecutada.

Diseño de caso de prueba para la ruta 1, 2, 3, 5.	
Descripción	El sistema debe adicionar una comisión con los valores pasados por parámetro.
Entrada	$request
Salida	Se muestra la ventana de información “La comisión ha sido insertada”
Resultados esperados	Se debe adicionar la comisión.

Tabla 13 Diseño de caso de prueba para la ruta 1, 2, 3, 5

Diseño de caso de prueba para la ruta 1, 2, 4, 5.	
Descripción	El sistema debe mostrar un mensaje de Error, ya sea porque ya existe la comisión o porque se insertaron mal los datos.
Entrada	$request
Salida	Se muestra la ventana de información “Error, esta comisión ya existe o se insertaron datos erróneos de la misma.”
Resultados esperados	No se inserta la comisión.

Tabla 14 Diseño de caso de prueba para la ruta 1, 2, 4, 5.

Fueron realizados un total de 100 pruebas de caja blanca mediante la técnica de ruta básica, a las 21 clases con las que se cuenta en el sistema. Del total presentado, 40 se le realizaron a la clase action y 60 a las clases Peer, representando cada una de las mismas 40% y 60% respectivamente.

3.7 Conclusiones del capítulo

Con el desarrollo del presente capítulo se puede concluir que, el diseño elaborado cumple con los aspectos medidos para su validación, mediante las métricas de diseño utilizadas. Además las pruebas aplicadas permitieron evaluar los elementos del software. También las funcionalidades implementadas mostraron el correcto funcionamiento, respondiendo a los requisitos. Y se evidencia que el contar con la metodología de desarrollo de software Scrum, representa una enorme ventaja a para los equipos de desarrollo pequeños.

Conclusiones generales

Como resultado del cumplimiento los objetivos planteados, se obtuvo un sistema que permitirá gestionar de una mejor forma el proceso de la Comisión Disciplinaria, siendo menor el tiempo necesario para realizar las distintas operaciones y obteniéndose los datos necesarios para cada caso.

Se realizó un estudio del lenguaje y las herramientas a utilizar, analizando sus principales características y ventajas, para de esta forma determinar las herramientas idóneas para el desarrollo del sistema. Además se investigaron otros sistemas judiciales nacionales e internacionales.

Se realizó una buena captura de requisitos, se generaron todos los artefactos correspondientes a la metodología Scrum, fueron desarrollados todos los prototipos de interfaz de usuario, el modelo de datos, los diagramas de procesos de negocio y se analizaron también los patrones de diseño utilizados en la solución.

Con la aplicación de distintas métricas de diseño y pruebas de caja negra, se demostró que el diseño e implementación del sistema, cumplen con los parámetros establecidos y dan solución a los requisitos funcionales del sistema.

El uso de herramientas libres, contribuye a una mejor distribución del sistema.

Recomendaciones

Luego de darle cumplimiento a los objetivos propuestos y de implementar el sistema se recomienda:

- ✓ Seguir desarrollando el sistema CODIS a partir de algunas recomendaciones o sugerencias.
- ✓ Extender la utilización del sistema a todas las facultades de la UCI.

Referencias bibliográficas

1. Adictos al Trabajo2003
2. Angel2011Geek the Planet
3. Apache Camel2004
4. Cronicas de un desarrollador y como no morir en el intento
5. DBA support2011
6. Desarrollo Ágil con SCRUM
7. Desarrollo en Web2012
8. Desarrollo Web
9. DesarrolloWeb.com
10. Ecured2009
11. Ecured2011
12. Ecured2012
13. editorbfbPrograma cion Desarrollo.es
14. El modelo Scrum2006
15. El Proceso Unificado Ágil v1.12005
16. Eugenia Bahit
17. Métodologías Ágiles en el Desarrollo de Software2006 8
18. MySQL1997-2011
19. NetBeans2012
20. PHP Manual2001
21. Poder Judicial de la Provincia de Corrientes2007
22. PostgreSQL1996

23. Rector2009Reglamento UniveritarioLa Habana
24. Sencha2009
25. Slideshare2006
26. Slideshare2007
27. Slideshare2010
28. Symfony 1.2, la guia definitiva2008
29. Tupe , C., & Cisneros, J. (2008). Evaluación y Selección de Framework de Desarrollo PHP: Symfony, Kumbia, CakePHP y Zend.
30. 2007TUTORIAL de cakePHP
31. Webmaster
32. Webmaster2007Ucipedia

Anexos

Anexo 1: Sprints de Scrum

Pila del Sprint III				
Backlog	Tarea	Estado	Estimado	Real
Implementación de la funcionalidad Denunciar				
	Implementación de la función Listar grupos	100%	1	2
	Implementación de la window (ventana) Estudiantes Grupo	100%	3	4
	Implementación de la función Adicionar denuncia.	100%	8	8
Implementación de la funcionalidad Analizar denuncia				
	Implementación de la función Listar denuncias	100%	3	2
	Implementación de la función Analizar denuncia	100%	4	4

Tabla 15 Pila del Sprint III

Pila del Sprint IV				
Backlog	Tarea	Estado	Estimado	Real
Implementación de la funcionalidad Gestionar comisiones				
	Implementación de la función Listar comisiones	100%	1	2
	Implementación de la función adicionar comisión.	100%	3	4

	Implementación de la función Adicionar miembro comisión	100%	8	8
	Implementación de la función Eliminar comisión	100%	1	1
	Implementación Obtener miembro comisión	100%	3	2

Tabla 16 Pila del Sprint IV

Pila del Sprint V				
Backlog	Tarea	Estado	Estimado	Real
Implementación de la funcionalidad Gestionar casos				
	Implementación de la función adicionar caso a comisión	100%	2	3
	Implementación de la función modificar porciento caso	100%	1	1
Implementación de la funcionalidad Gestionar cargo				
	Implementación de la función Asignar cargo a persona	100%	4	5
	Implementación Obtener miembro comisión	100%	3	2

Tabla 17 Pila del Sprint V