UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 1

Implementación del módulo de Reportes y Estadísticas del subsistema de Gestión Universitaria Pregrado.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE

INGENIERO EN CIENCIAS INFORMÁTICAS

Autores: Yeny Lauris Milanés Regalado.

Yoander Peña Pérez.

Tutor: Ing. Jorge Luis Tamarit Cutiño.

Ciudad de La Habana, Junio de 2010

"Año 52 de la Revolución".

"Los hombres son como los astros, que unos dan luz de sí y otros brillan con la que reciben".

José Martí.

Declaración de autoría

Declaramos ser autores de la presente tesis y concedemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de_ junio_ del 2010.

Yeny Lauris Milanés Regalado	Yoander Peña Pérez

Ing. Jorge Luis Tamarit Cutiño

Datos de contacto

Jorge Luis Tamarit Cutiño (jtamarit@uci.cu)

Ingeniero en Ciencias Informáticas, graduado en la Universidad de las Ciencias Informáticas en el curso correspondiente a los años 2006-2007. Actualmente se desempeña como especialista superior en el Centro de Informatización Universitaria.

Agradecimientos

Agradecemos a nuestro Comandante en Jefe Fidel Castro Ruz y a la Revolución Cubana por darnos la oportunidad de formar parte de este proyecto. Agradecemos a todas las personas que de alguna forma han colaborado con la realización del presente trabajo de diploma, Ronny Zamora, Diana Rosa Pérez, Javier González de La Nuez y en especial al tutor Jorge Luis Tamarit Cutiño por ayudarnos y guiarnos.

Yeny

A mi papá Julio por ser el mejor de todos los hombres de la tierra, por ser junto a mi mamá los seres que más quiero y respeto, por enseñarme que para lograr cosas hay que dedicar mucho tiempo, esfuerzo y sacrificio y que al final todo se puede. Te amo papi.

A mi mamá Mariela que me ha aguantado todas mis malcriadeces, sacrificando todo lo que tiene para que a mí y a mi hermana no nos falte nunca nada. Ella ha sido la luz que guía mi camino y sin su apoyo incondicional no hubiera podido llegar hasta aquí.

A mi abuela por estar presente en cada paso que doy en mi vida alumbrándome con su cariño y su amor maternal.

A mi hermana Yeilín por ser la mejor de las hermanas, porque aunque siempre estemos fajadas, yo sé que ella me quiere tanto como yo a ella.

A mis primas Dalita, Ochín, Nani, a las cuales considero otras hermanas por habernos criado juntas y estar conmigo siempre en las buenas y malas. Las quiero con todo mi corazón.

A mi primito Rubencito y mi tía Radeunda que aunque ya no estén en este mundo siempre los llevaré presente en mi mente y mi corazón.

A mis tíos Jorge, Rubén, Antonio y Oscar que siempre que he necesitado de ellos nunca me han fallado y siempre me han aconsejado y querido como una hija más.

A mis tías Adis y Gisela que han sido mis segundas mamás desde que era un bebé y han estado cuidándome y dándome su comprensión en todo momento de mi vida.

A mis mejores amigas Eli, Sailín, Yanet, Vivi y Gabi y mis grandes amigos Yeset y Enrique por demostrarme que en esta vida existen las buenas amistades, ustedes son el ejemplo fehaciente de ello.

A mi compañero de tesis Yoander por haber sido el mejor de los compañeros sin él esto no hubiera sido posible.

A mi novio Roberto por haber estado para mí en los momentos más duros y brindarme su apoyo en todo momento.

A los amigos de mi municipio Yailín, Marlies, Marlis, Leidis, Adrian y Quintero por ser tan chéveres y estar tan unidos estos 5 años.

Yoander

A la Revolución por haberme dado la posibilidad de llegar a ser alguien en la vida.

A mis padres por su amor incondicional, apoyo y confianza en todo momento aun cuando las cosas no iban bien.

A mis hermanos y familiares por sus consejos y confianza en mí.

A mi tutor porque a pesar de su escaso tiempo, siempre estaba ahí para cuando lo necesitaba.

A Diana Carrillo por haber compartido conmigo tres años, que me marcaron por el resto de

la vida.

A mi amiga Imilsy por todo su apoyo y confianza en todo momento.

A Rubén, María Luisa, Diana por todo su cariño brindado en todo este tiempo.

A mi compañera de tesis Yeny por su esmero y dedicación en todo el transcurso del trabajo de diploma.

A Javier, Aylien, Alexander y Diana Pérez por toda su ayuda ante todas mis dudas.

A todos mis amigos, los que vinieron conmigo y los que conocí aquí, gracias por estar ahí.

Dedicatoria

Yeny

A mis queridos padres por ser lo más importante que tengo en la vida.

A mi hermana Yeilín por ser la mejor amiga y estar ahí siempre para mí.

A mi abuela y mis tías que han sabido educarme y darme todo el amor del mundo.

A mi primo Rubencito que aunque ya no está aquí en este mundo, mi tesis va dedicada a él como un homenaje por haber sido el mejor primo del mundo.

Dedicatoria

Yoander

A mi abuelo Eugenio que aunque no haya podido verme realizando uno de mis sueños esto es para él. Gracias por todo el amor brindado.

A mis padres por ser la razón de mi vida y una guía a seguir.

A mis hermanos para que les sirva de ejemplo en un futuro no muy lejano.

A Diana por haberse ganado un lugar muy especial en mi corazón.

Resumen

La Universidad de las Ciencias Informáticas (UCI) cuenta con un Sistema de Gestión Académica denominado Akademos 1, que es el encargado de llevar todo el control de la actividad docente en la institución. El módulo de Reportes de dicho sistema no brinda la posibilidad de que los usuarios puedan generar sus propios reportes, pues no cuenta con reportes dinámicos, sino estáticos. Está implementado con herramientas privativas lo cual entra en conflicto con la política de migración a software libre de la UCI y del país. Además, cuenta con un mal diseño de su base de datos impidiendo que se puedan realizar modificaciones al sistema para solucionar un determinado problema.

En el presente trabajo se realiza la implementación del módulo de Reportes y Estadísticas del nuevo subsistema de Gestión Universitaria Pregrado con tecnologías libres, para darle solución a algunas de las deficiencias del sistema actual, anteriormente mencionadas.

Se implementan las Historias de Usuario entregadas por los analistas, para lo cual se emplea el estándar de codificación seleccionado por el Departamento de Gestión Universitaria de la UCI para el tratamiento de los algoritmos más importantes implementados. Se detallan además cada una de las clases y operaciones fundamentales del sistema.

Se especifican de forma detallada las funcionalidades de la propuesta de solución, la cual es validada a través de las pruebas de Aceptación.

Palabras claves:
Reportes, estadísticas, Gestión Universitaria Pregrado.

Índice de contenido

ÍNDICE DE TABLAS

Índice de figuras

Introducción

Las Tecnologías de la Información y las Comunicaciones (TIC) han marcado grandes avances en la sociedad actual. En Cuba han surgido nuevos proyectos que pretenden dar un impulso al desarrollo económico, político y social alcanzado. Uno de estos proyectos es la Universidad de las Ciencias Informáticas (UCI), nacida bajo el calor de la Batalla de Ideas para informatizar el país y desarrollar la industria del software cubana.

En la UCI se han creado sistemas de gestión académica para la automatización de diversos procesos. Un sistema de gestión académica es aquel que provee de información útil y necesaria a una Dirección Docente. Entre las tareas fundamentales de dicho sistema, está la generación de reportes en diferentes formatos para posibilitar una mejor visualización de los datos disponibles y así tener todos los elementos necesarios para una correcta toma de decisiones.

Muchos sistemas de gestión tienen el punto débil de contar sólo con un conjunto cerrado de reportes, pero a medida que va evolucionando un proceso de negocio, aparecen nuevas necesidades de información que ameritan incorporar nuevos tipos de reportes a las aplicaciones.

Entre las soluciones de software que se han implementado en la Universidad de las Ciencias Informáticas, se encuentra el Sistema de Gestión Académica conocido como Akademos 1, su misión fundamental es gestionar toda la información de los elementos que intervienen en el proceso docente. Está compuesto por una serie de módulos que permiten registrar premios, asistencias, evaluaciones, la gestión del plan de estudio, guardar el registro de los profesores, realizar la matrícula y los reportes de los datos almacenados.

En la **situación problémica** se identificaron las siguientes deficiencias del sistema Akademos 1:

Las herramientas que se utilizaron para su diseño fueron propietarias, por lo cual no se cumple con los planes de migración a software libre que se ha trazado la UCI y el país. Además, no posee una documentación consistente que respalde su desarrollo, dificultando en gran medida la realización de nuevas funcionalidades e incluso la posibilidad de hacer cambios al sistema para solucionar problemas.

Cuenta con un mal diseño de su base de datos debido a una implementación en un corto período de tiempo, los constantes cambios y el gran volumen de información que almacena; todo lo cual provoca problemas de rendimiento y escalabilidad.

Además, su estructura está diseñada solamente para realizar reportes estáticos, por lo que no le permite al usuario confeccionar sus propios reportes, es decir, reportes dinámicos.

Por todo lo antes mencionado y al no cumplir el módulo de Reportes del Akademos 1 actual con las necesidades de la gama de usuarios a la que está dirigido se plantea el siguiente **problema científico**: ¿Cómo facilitar el proceso de toma de decisiones en la gestión académica del subsistema de Gestión Universitaria Pregrado en la UCI?

Luego de un análisis de la situación actual, la investigación enmarca su **objeto de estudio** en los procesos de generación de reportes y estadísticas para los sistemas de gestión académica, centrando su **campo de acción** en los procesos de generación de reportes y estadísticas para el subsistema de Gestión Universitaria Pregrado en la UCI.

Teniendo en cuenta lo planteado anteriormente, el **objetivo general** de este trabajo es: Implementar el módulo de Reportes y Estadística del subsistema de Gestión Universitaria Pregrado.

Para cumplir con el objetivo general expuesto, se trazan varios objetivos específicos que facilitan y organizan el trabajo:

- Estudio de las técnicas de programación, plataformas, herramientas y metodologías de desarrollo de software que faciliten la implementación del módulo de Reportes y Estadísticas.
- Implementación del módulo de Reportes y Estadísticas del subsistema de Gestión Universitaria Pregrado.
- Validación de la propuesta de solución mediante las pruebas de funcionalidad.

Tras definir los objetivos y analizar el problema se plantea como **idea a defender**: La implementación del módulo de Reportes y Estadística del subsistema de Gestión Universitaria Pregrado posibilitará gestionar de manera eficiente la información referente a los procesos de salida del mismo.

Para hacer efectivo el cumplimiento de cada uno de los objetivos específicos mencionados, se orienta un conjunto de **tareas científicas** que aseguran el éxito de estos:

- Describir las tecnologías y herramientas a utilizar en la realización de la propuesta de solución.
- Implementar la propuesta de solución.
- Realizar pruebas de funcionalidad tales como: las Unitarias o de Aceptación.

Los **métodos teóricos** utilizados fueron el **histórico-lógico** y **analítico-sintético**. El primero, para estudiar formas de solución a problemas similares sobre la gestión de procesos de reportes en todo el mundo, ya que permitirá constatar teóricamente cómo ha evolucionado el fenómeno que se estudia en un período de tiempo determinado. El segundo, para el análisis de teorías y documentos partiendo de la extracción de los elementos más importantes de cada uno de los aspectos esenciales de las herramientas y la literatura seleccionada para el tema.

El **método empírico** utilizado es la **Observación**: Debido a que se pudo constatar los problemas que existían en la Universidad producto al proceso de gestión de reportes en Akademos 1.

Capítulo 1. Fundamentación teórica

Este capítulo incluye un estado del arte de las distintas técnicas de programación que rigen en el mundo, el país y en la Universidad. Se hace referencia a los lenguajes de programación del lado del cliente y del servidor, a las diferentes herramientas y tecnologías seleccionadas por el Departamento de gestión Universitaria de la UCI, tales como, Sistema Gestor de Base de Dato, framework, entorno de desarrollo y plataforma sobre la que se realiza la propuesta de solución. Por último, se fundamenta brevemente la metodología de desarrollo de software seleccionada para el trabajo.

Capítulo 2. Descripción y análisis de la solución propuesta

En este capítulo se describen las clases que forman parte de la propuesta de solución, sus principales funcionalidades y el estándar de codificación empleado para la implementación de las mismas. De igual forma se realizan las tareas de ingeniería correspondiente a cada una de las Historias de Usuario (HU).

Capítulo 3. Validación de la solución propuesta

En este capítulo se analizan algunas de las características que presentan las pruebas de Aceptación y Unitarias. A partir de esto se realiza la validación de la propuesta de solución a través de las pruebas de aceptación. Se comprueban los resultados obtenidos y de esta forma se llega a conclusiones sobre lo logrado en el trabajo.

CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se analizan las principales técnicas de programación que existen en la actualidad, las ventajas de las aplicaciones web y del software libre. Además, se hace un breve resumen acerca de la plataforma, framework, metodología, gestor de base de datos, entorno de desarrollo, servidor web y los lenguajes de programación del lado del cliente y del servidor seleccionados por el Departamento de Gestión Universitaria de la UCI para la propuesta de solución.

1.2 Técnicas de programación

Las técnicas de programación constituyen una parte fundamental en el proceso de desarrollo e ingeniería del software dentro del ámbito informático. Cada técnica tiene sus propias características y distintos métodos para darle solución a determinados problemas, es de gran importancia aprender a implementarlas a la hora de realizar cualquier proyecto de desarrollo de software.

1.2.1 Programación modular

La programación modular es una técnica donde los procedimientos con una funcionalidad común son agrupados en diferentes módulos. Cada módulo contendrá sus propios datos, lo que permitirá que cada cual maneje un estado interno que será modificado por las llamadas a procedimientos de estos.

Existe un estado para cada módulo y estos existen aunque sea una vez en todo el programa. Por lo que un programa ya no consistirá en una sección, sino que estará dividido en varias secciones más pequeñas que interactúan a través de llamadas a procedimientos y que integran el programa en su totalidad. (1)

El programa principal coordina las llamadas a procedimientos en módulos separados y pasa los datos apropiados en forma de parámetros, tal como se muestra en la figura 1.

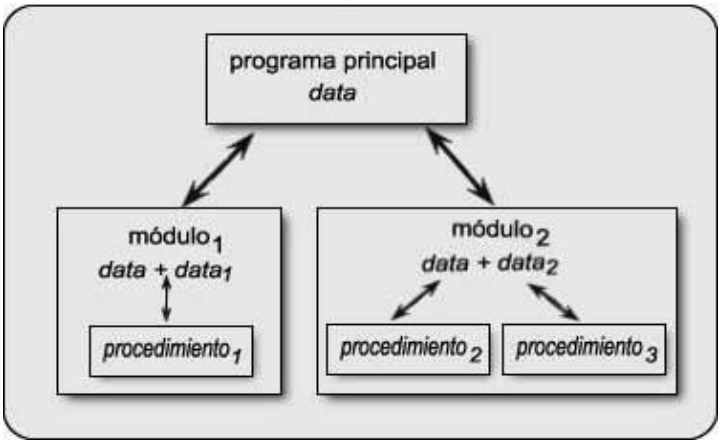

Figura 1 Programación Modular

Cuando la programación se realiza internamente y se hace un enfoque ascendente, es difícil llegar a integrar los subsistemas al grado tal que el desempeño global sea fluido. Los problemas de integración entre los subsistemas son sumamente costosos y muchos de ellos no se solucionan hasta que la programación no alcanza la fecha límite para la integración total del sistema. En esta fecha ya se cuenta con muy poco tiempo, presupuesto o paciencia de los usuarios, como para corregir aquellas delicadas interfaces, que en un principio, se ignoran. (1)

1.2.2 Programación Orientada a Objetos (POO)

En la POO se debe tener tantos objetos-lista como sea necesario. En esta programación se manda un mensaje directamente al objeto-lista en cuestión. Ver figura 2. Cada objeto implementa su propio módulo, permitiendo por ejemplo que coexistan muchas listas. Cada objeto es responsable de inicializarse y destruirse en forma correcta. No existe la necesidad de llamar explícitamente al procedimiento de creación o de terminación. (2)

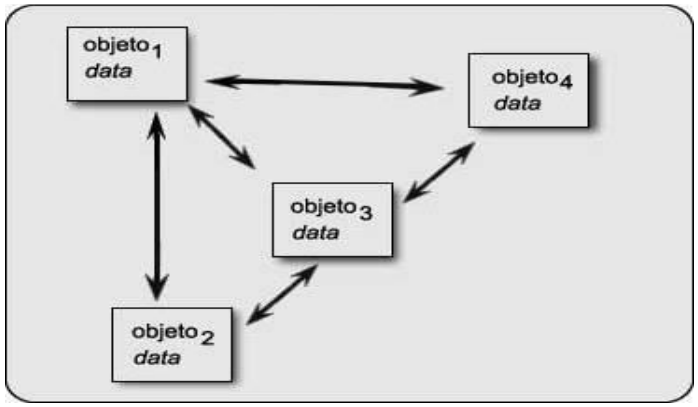

Figura 2 Programación Orientada a Objetos

La POO está basada en cuatro aspectos: definición de tipos de datos abstractos, herencia, encapsulamiento y polimorfismo.

En la POO encapsular significa que se reúne y controla todo el grupo resultante en un conjunto y no de forma individual. La abstracción es un término externo al objeto, que controla la forma en que es visto por los demás. La herencia se define como una jerarquía de clases derivadas y la relación entre estas, donde se comparte la estructura y el comportamiento de una o más clases consideradas como clases padres. El polimorfismo constituye la definición de múltiples clases con funcionalidades diferentes, pero con métodos o propiedades denominados de forma idéntica.

Los programas OO (Orientado a Objetos) suelen ser más fáciles de escribir, mantener y reutilizar que programas escritos en un lenguaje del paradigma procedural (como C, Pascal, Fortran o Basic). (2)

La popularidad de la POO viene dada por la fácil implementación de interfaces gráficas de usuario, en las cuales los diferentes elementos de la interfaz son objetos que interactúan por eventos.

1.2.3 Programación procedimental

Con esta técnica el programa se divide en partes independientes, cada una de las cuales ejecuta una única actividad o tarea. Estas partes se analizan, se codifican separadas de las demás y se ponen aparte. Siguiendo un método ascendente o descendente de desarrollo se llegará a la descomposición final del problema en módulos en forma jerárquica. Las descomposiciones resultantes reciben luego el refinamiento progresivo del repertorio de instrucciones que van a formar parte de cada pieza del programa. (3)

Si la tarea asignada a cada procedimiento es demasiado compleja, éste deberá descomponerse en otros más pequeños. Este proceso de subdivisión continúa hasta que cada uno tenga solamente una tarea específica que realizar. Dichas tareas pueden ser de entrada, salida, procesamiento de datos, control de otros métodos o una combinación de éstos.

Al introducir parámetros, así como procedimientos de procedimientos (subprocedimientos) los programas pueden ser escritos de forma más estructurada y libre de errores. Si un procedimiento es correcto, cada vez que es usado produce resultados correctos. En caso de errores, se puede reducir la búsqueda a aquellos lugares que aún no han sido revisados.

De este modo, un programa puede ser visto como una secuencia de llamadas a procedimientos, tal como se muestra en la figura 3. El programa principal es responsable de pasar los datos a las

llamadas individuales, los datos son procesados por los procedimientos y, una vez que el programa ha terminado, los datos resultantes son presentados.

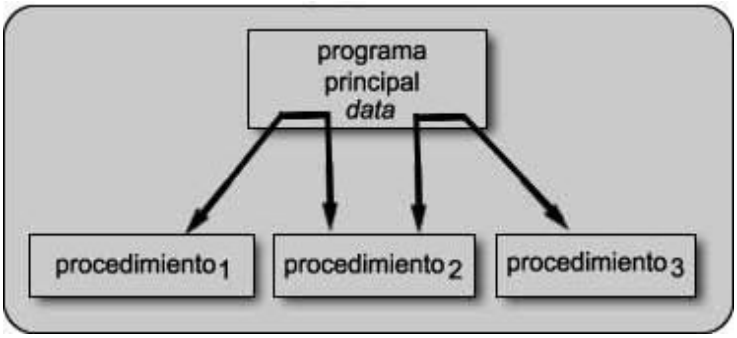

Figura 3 Programación Procedimental

Un procedimiento puede transferir temporalmente el control a otro y a la vez ser independiente ya que el acceso no se realiza de manera directa. Un procedimiento puede utilizar los resultados de otro siempre y cuando se le transfiera el control de estos.

La programación procedimental es rígida e inflexible, con ella hay pérdida excesiva de tiempo en la corrección de errores y la documentación es deficiente e insuficiente. (3)

1.2.4 Programación Concurrente

La programación concurrente aborda las notaciones y técnicas de programación, que se usan para expresar el paralelismo potencial para resolver los problemas de comunicación y sincronización entre procesos.

En la programación concurrente se supone que hay un procesador utilizable por cada tarea; no se hace ninguna suposición de si el procesador será una unidad independiente para cada una de ellas o si será un solo CPU, que comparte el tiempo entre las tareas.

Independientemente de cómo se vaya a ejecutar realmente el programa, en un sistema uniprocesador o multiprocesador, el resultado debe ser el correcto. Por ello, se supone que existe un conjunto de instrucciones primitivas que son parte del sistema operativo o de un lenguaje de programación, y cuya correcta implementación y corrección está garantizada por el sistema. (3)

1.2.5 Programación Funcional

Es un paradigma independiente que surge a principios de los sesenta, su creación se debió a las necesidades de los investigadores en el campo de la inteligencia artificial y los asociados a este como

el cálculo simbólico, pruebas de teoremas, sistemas basados en reglas y procesamiento del lenguaje natural.

Estas necesidades no estaban cubiertas por los lenguajes imperativos de la época. La programación funcional es uno de los paradigmas fundamentales entre los de programación declarativa[1]. La misma permite unir los componentes de especificación y programación en la tarea de solución automática de problemas.

La característica principal de la programación funcional es que los cálculos se ven como una función matemática que hacen corresponder entradas y salidas. No hay noción de posición de memoria y por tanto, no hay necesidad de una instrucción de asignación. Los bucles se modelan a través de la recursividad ya que no hay manera de incrementar o disminuir el valor de una variable. Como aspecto práctico casi todos los lenguajes funcionales soportan el concepto de variable, asignación y bucle. (3)

Los lenguajes funcionales ofrecen al programador recursos expresivos que permiten resolver problemas complejos mediante programas pequeños y robustos. Entre ellos el sistema de tipo polimórfico que permite definir estructuras de datos de uso genérico.

1.2.6 Programación Lógica

La programación lógica, junto con la funcional, forma parte de la programación declarativa. En los lenguajes tradicionales, la programación consiste en indicar cómo resolver un problema mediante sentencias; la programación lógica se realiza de forma descriptiva, estableciendo relaciones entre entidades, indicando no cómo, sino qué hacer.

La ecuación de Robert Kowalski [2](Universidad de Edimburgo) establece la idea esencial de la programación lógica: algoritmos = lógica + control. Es decir, un algoritmo se construye especificando conocimiento en un lenguaje formal (lógica de primer orden) y el problema se resuelve mediante un mecanismo de inferencia (control) que actúa sobre aquél. (4)

La programación lógica construye una base de conocimientos mediante reglas y hechos. Un lenguaje por excelencia de programación lógica es el Prolog.

[1] Es un paradigma de programación que está basado en el desarrollo de programas especificando o "declarando" un conjunto de condiciones, proposiciones, afirmaciones, restricciones, ecuaciones o transformaciones que describen el problema y detallan su solución.

[2] Robert Kowalski: Fue uno de los creadores de la programación lógica en 1970. Escribió el libro “Cómo ser artificialmente inteligente”.

1.3 Tendencias actuales

La UCI ha incrementado considerablemente el desarrollo de aplicaciones webs encaminadas a la informatización de sus procesos. La tendencia a la hora de desarrollar estas aplicaciones es la utilización de software libre. El mismo ofrece un gran número de ventajas, por ejemplo, se le pueden realizar las modificaciones que se crean necesarias, reutilizando códigos que resuelvan los problemas a los que se intenta dar solución.

1.3.1 Software *Libre*

La Fundación de Software Libre es una organización creada en octubre de 1984 a partir del esfuerzo de Richard Matthew Stallman[3] y otros entusiastas del software libre con el propósito de difundir este movimiento. El “Software Libre'' es un asunto de libertad, no de precio. Para entender el concepto, se debe pensar en “libre" como en “libertad de expresión", lo que ha dado lugar a cierta confusión. “Software Libre" se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. (5)

De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito.
- La libertad de estudiar cómo funciona el programa, y adaptarlo a necesidades específicas. El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias.
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie.

3 Gurú, hacker y filósofo del movimiento del Software Libre. A mediados de los años 80 fundó la *Free* Software *Foundation* y el proyecto GNU. En los últimos años, su actividad principal es viajar por el mundo y dar conferencias sobre Software Libre y en contra de la patentabilidad del software.

Se puede decir que el software libre tiene muchísimos beneficios estos son:

Razones económicas

Ahorros importantes al liberarse del pago de licencias y especialmente por la replicación casi gratuita de aplicaciones comunes a toda la administración pública.

Independencia tecnológica

No depende de terceros para el diseño, desarrollo y mantenimiento de sus sistemas de información, retomando el control total de sus procesos.

Control de la información

El acceso al código fuente, la libertad de inspeccionar el funcionamiento del software, la libertad de decidir la manera en que se almacenan los datos y la posibilidad de modificar cualquiera de estos, permitiendo el control total de la información.

Confiabilidad y estabilidad

El software libre realizado por comunidades está sometido a la inspección de un importante número de personas. Estas personas identifican los problemas, los resuelven, y comparten las soluciones con los demás.

Desarrollo del país

Se genera transferencia tecnológica hacia los actores nacionales productores de software, acelerando el desarrollo endógeno y reforzando la soberanía nacional.

1.3.2 Aplicaciones web

Una aplicación web es un sistema informático que los usuarios utilizan accediendo a un servidor web a través de un navegador. Las aplicaciones webs son populares debido a la practicidad del navegador web como cliente ligero. La facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software en miles de potenciales clientes es otra razón de su popularidad. Proporcionan un lenguaje y una sintaxis independiente de la plataforma para intercambiar datos complejos mediante mensajes. (6)

Además de posibilitar el acceso a varios usuarios simultáneamente al sistema, estas poseen otras ventajas tales como:

- Tienen un precio menor que las aplicaciones de escritorio, pues no requieren la infraestructura de distribución, soporte técnico y marketing requerido por las aplicaciones descargables de software tradicionales. Esto permite que las aplicaciones en línea cuesten una fracción de sus contrapartes descargables.
- Los datos están en línea, es decir, no es necesario encontrarse precisamente en la computadora en la que se instaló el software para tener acceso al mismo, sin necesidad de compartir escritorios.
- En el caso de las personas o instituciones que usan las aplicaciones web, encuentran en las mismas un medio de promoción rápido y eficiente.
- Cuenta con una mayor inmediatez de acceso, estas no necesitan ser descargadas, instaladas, ni configuradas.

1.4 Lenguajes de programación

Un lenguaje de programación es aquel que es capaz de ordenar un conjunto de instrucciones realizadas por una computadora. El uso más común para un lenguaje de este tipo consiste en la creación de programas, secuencias de órdenes para lograr determinadas aplicaciones prácticas: por ejemplo el navegador que se utiliza es un programa, así como los procesadores de texto y planillas de cálculo. (7)

Es decir, los lenguajes de programación permiten crear programas y a su vez resultan independientes del modelo de computador a utilizar. Actualmente existen diferentes lenguajes de programación para desarrollar en la web, estos han ido surgiendo debido a las tendencias y necesidades de las plataformas. A continuación una panorámica de cada uno de los utilizados en la solución propuesta.

1.4.1 Lenguaje al lado del cliente

Un lenguaje del lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio. Pero la página no se verá bien si el ordenador cliente no tiene instalados los *plug-ins* adecuados. El código, tanto del hipertexto como de los *scripts*, es accesible por cualquier cliente y ello puede afectar a la seguridad. (8)

Los lenguajes del lado del cliente, entre los cuales no sólo se encuentra el HTML[4], sino también JavaScript, el cual es simplemente incluido en el código HTML, son aquellos que pueden ser directamente asimilados por el navegador.

HTML

Todos los sitios web se realizan en base a dos lenguajes de marcación, HTML o XHTML[5]. Un lenguaje de marcación permite estructurar y codificar el contenido de un sitio web, para lo cual emplea marcas o etiquetas en las que se incluye información sobre la estructura del mismo. Por lo que se denomina lenguaje de marcación, a aquel que emplea marcas que indican de qué manera organizar el contenido de un sitio web.

HTML es un lenguaje simple utilizado para crear documentos de hipertexto para la WWW[6]. No es un lenguaje de descripción de página como Postscript y el PCL[7], ya que no permite definir de forma estricta la apariencia de este, aunque una utilización algo desviada hace que se utilice en ocasiones como un lenguaje de presentación. Además, la presentación de la página es muy dependiente del *browser* (programa o navegador) utilizado. HTML se limita a describir la estructura y el contenido de un documento, y no el formato de la página y su apariencia. (9)

El lenguaje de HTML es uno de los más populares en la creación de páginas webs dinámicas. Algunas de sus ventajas principales son:

- Texto presentado de forma estructurada y agradable.
- Despliegue rápido.
- Lenguaje de fácil aprendizaje.
- Lo admiten todos los navegadores.

Hojas de Estilo en Cascada (CSS)

El CSS en un lenguaje de estilo que se emplea para la presentación de documentos escritos en HTML o XHTML. Especifica cómo se mostrará cada parte del texto; color, fuente, cuerpo, alineación y fondo.

[4] Lenguaje de Marcas Hipertextuales.

[5] Lenguaje extensible de marcado de hipertexto o en inglés *Extensible Hypertext Markup Language.*

[6] Red Global Mundial o en inglés *World Wide* Web.

[7] Lenguaje de órdenes de impresión o en inglés *Printer Command Language*.

El CSS se organiza en capas que coinciden con las divisiones del texto en HTML. Cada capa tiene indicaciones específicas. Incluye las propiedades de: fuentes, color y fondo, texto (alineación, espaciado de palabras), de caja (margen, borde, relleno), de clasificación (listas, visualización). (9)

Los beneficios de utilizar este lenguaje de estilo son varios.

- Actualización tanto de los contenidos como de los aspectos gráficos, pues es más sencillo si el texto y la presentación están separados.
- Se pueden hacer modificaciones fácilmente, por ejemplo, de color y de fondo.
- Un sitio web puede disponer de diferentes hojas de estilo, una para cada tipo de dispositivo (computadora, teléfono celular e impresora).
- Al emplear el lenguaje de estilo CSS, el archivo de HTML tiene un tamaño menor y los navegadores pueden leer más sencillamente un sitio web.

JavaScript

JavaScript es un lenguaje de programación para crear sitios web con presentaciones dinámicas. Fue creado por Brendan Eich[8] en la empresa *Netscape Communications*. Se trata de una aplicación "del lado del cliente", dado que es el navegador del usuario el que interpreta y ejecuta el lenguaje JavaScript, es decir, sus instrucciones.

Si el navegador no soporta a JavaScript el usuario no podrá visualizar las presentaciones dinámicas de un sitio web. De todos modos, los navegadores más populares, tales como Internet Explorer o Netscape (especialmente en sus últimas versiones), leen JavaScript sin problemas. (10)

En JavaScript se pueden crear sitios con efectos especiales, elementos con movimiento, cambios de color, sonido, y también sitios interactivos, con tablas de cálculo, personalización, agendas, calculadoras y otras formas de comunicación con el usuario. El lenguaje JavaScript es relativamente sencillo de utilizar. Las instrucciones se suelen incluir en el documento HTML, con la etiqueta <*script*>.

JavaScript siempre fue utilizado de forma masiva por la mayoría de los sitios de Internet. La aparición de Flash disminuyó su popularidad, ya que este realizaba algunas acciones que a JavaScript le eran

[8] Programador de computadoras, más conocido por inventar el lenguaje de programación JavaScript.

imposibles de llevar a cabo. La aparición de las aplicaciones AJAX[9] programadas con este lenguaje, le devolvió su popularidad sin igual. (10)

Ventajas:

- Es un lenguaje de *scripting* seguro y fiable, ya que está en claro y hay que interpretarlo y la seguridad es casi total pues se corrigieron la mayoría de los fallos como; una leve entidad a la hora de realizar una lectura de los sitios visitados, de la dirección e-mail y de los archivos presentes en el disco, que existían en las versiones de Netscape sucesivas a la 2.0.
- El código de Javascript es ejecutado en el cliente, lo que conlleva a que el servidor no es solicitado más de lo debido. Un *script* ejecutado en el servidor se sometería a una dura prueba y los servidores con capacidades más limitadas podrían debilitarse a causa de una continua solicitud por un mayor número de usuarios.
- El código es visible y puede ser leído por cualquiera, incluso si está protegido con las leyes del copyright.

En la implementación del módulo se utilizó jQuery, un framework de JavaScript y una técnica para implementar páginas webs más dinámicas llamada AJAX.

jQuery

Cuando un desarrollador tiene que utilizar JavaScript, generalmente tiene que preocuparse por hacer s*cripts* compatibles con varios navegadores como Explorer, Firefox y Opera. En jQuery se implementa una serie de clases que van a permitir programar sin que haya que preocuparse por el navegador que está utilizando el usuario, ya que funcionan de la misma forma en todas las plataformas más habituales. (11)

Este framework ofrece una infraestructura con la que se tendrá 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 y aplicaciones que hacen uso de AJAX.

Además de todas estas ventajas, jQuery se obtiene de manera gratuita, ya que el framework tiene licencia para uso en cualquier tipo de plataforma, personal o comercial. Esto se logra incluyendo en las

[9] Acrónimo para *Asynchronous* JavaScript + XML.

páginas un *script* JavaScript que contiene el código de jQuery para descargar de la propia página web del producto y comenzar a utilizar el framework.

El archivo del framework ocupa unos 56 KB, lo que es bastante razonable y no retrasará mucho la carga de la página. Esta carga de la página sólo se verá afectada por el peso de este framework una vez por usuario. Sin embargo, las ventajas a la hora de desarrollo de las aplicaciones, así como las puertas que abre jQuery compensan extraordinariamente el peso de dicho paquete.

Ajax

AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las sobrecargas constantes de una determinada página, ya que el intercambio de información con el servidor se produce en un segundo plano.

Una aplicación AJAX elimina la naturaleza "arrancar-frenar- arrancar-frenar" de la interacción en la web introduciendo un intermediario -un motor AJAX- entre el usuario y el servidor. Sumar una capa a la aplicación puede que la haga a esta poco reactiva, pero la verdad es todo lo contrario. (12)

En vez de cargar una página web al inicio de la sesión, el navegador carga al motor AJAX. Este es el responsable de cargar la interfaz que el usuario ve y comunicarse con el servidor en nombre del usuario. El motor AJAX permite que la interacción del usuario con la aplicación suceda independientemente de la comunicación con el servidor. Así el usuario nunca estará mirando una ventana en blanco del navegador, ni esperando que el servidor haga algo.

1.4.2 Lenguaje del lado del servidor

Los lenguajes del lado del servidor son aquellos lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él. En el dominio de la red, los lenguajes del lado del servidor más ampliamente utilizados para el desarrollo de páginas dinámicas son el ASP, PHP y PERL. (8)

En el trabajo de investigación se utiliza PHP, específicamente su versión *PHP 5.*

PHP

PHP[10] se trata de un lenguaje de programación que se emplea para la creación de sitios webs dinámicos. Su nombre proviene de Hypertext Pre-processor. El código PHP se incluye dentro del

[10] Páginas Iniciales Personales o en inglés *Personal Home Page*.

HTML, con etiquetas especiales. Es una tecnología del lado del servidor, ya que éste es el que interpreta y ejecuta el código PHP de un sitio dinámico.

Entre las principales ventajas que lo convierten en la herramienta ideal para la creación de páginas web dinámicas se tiene que:

- Es un lenguaje que puede ejecutarse en diferentes sistemas operativos (Linux, Windows y Macintosh) y en los servidores más populares como Apache.
- Permite programar sitios con contenido dinámico, combinándolo con los principales servidores de bases de datos, entre ellos PostgreSQL. De esta manera los sitios desarrollados con PHP presentan un gran dinamismo y un excelente manejo de datos.
- Es ideal para crear sitios web catálogos.
- Es un lenguaje libre y, por ende, accesible a todas las personas.
- Tiene gran cantidad de funciones ejemplificadas y una amplia documentación.

Con PHP se puede realizar cualquier cosa con un *script* de Interfaz Común de Puerta de Enlace (CGI por sus siglas en inglés), como el procesamiento de información en formularios, manipulación de páginas dinámicas y potentes herramientas empleadas por los servidores web para almacenar y recuperar información acerca de sus visitantes. (13)

También ofrece la integración con varias bibliotecas externas, que permiten que el desarrollador haga casi cualquier cosa, desde generar documentos en Portable Document Format (PDF, por sus siglas en inglés) hasta analizar código XML[11].

PHP 5

En versiones previas de PHP, los objetos eran manejados como tipos primitivos, por ejemplo enteros y cadenas. En la versión 5 el manejo de objetos en PHP ha sido re-escrito por completo, permitiendo una mejora en rendimiento y muchas características nuevas. En esta versión se incluyen modificadores de control de acceso para implementar el encapsulamiento, lo cual no existía en otras anteriores. Además, tiene capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad. (14)

[11] Lenguaje de Marcas Extensible o en inglés *Extensible Markup Language*.

1.5 Sistema de Gestión de Base de Datos

Un Sistema Gestor o Manejador de Bases de Datos (SGBD) es un conjunto de programas que permite a los usuarios crear y mantener una base de datos (BD), por lo tanto, el SGBD es un software de propósito general que facilita el proceso de definir, construir y manipular la BD para diversas aplicaciones.

Los sistemas de bases de datos están diseñados para gestionar grandes volúmenes de información. Algunas de las acciones que realizan son:

- Definición de los datos.
- Mantenimiento de la integridad de los datos dentro de la base de datos.
- Control de la seguridad y privacidad de los datos.

1.5.1 PostgreSQL

Es un servidor de base de datos de objeto relacional libre. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo, dicha comunidad es denominada el PGDG (Grupo de desarrollo global de PostgreSQL).

Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido más tarde en otros sistemas de gestión comerciales. Es un sistema objeto-relacional, ya que incluye características propias de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones disparadoras, reglas e integridad transaccional. A pesar de esto, PostgreSQL no es un sistema de gestión de bases de datos puramente orientado a objetos. (15)

Ventajas:

- Soporte para los lenguajes más populares del medio: PHP, C, C++, Perl, Python entre otros.
- Puede ser utilizado en los principales sistemas operativos: Linux, Unix, Mac OS, Beos y Windows.
- Extensiones para alta disponibilidad, nuevos tipos de índices, datos espaciales y minería de datos.
- Implementa el estándar SQL92/SQL99.

- Incorpora una estructura de datos *array*.
- Incorpora funciones de diversa índole: manejo de fechas, reglas y soporte para el Protocolo de Capa de Conexión Segura (SSL, por sus siglas en inglés).
- Soporta el uso de índices, reglas y vistas.
- Aprovecha la potencia de sistemas multiprocesadores, gracias a su implementación multihilo.
- Permite la gestión de diferentes usuarios y los permisos asignados a cada uno de ellos.

1.6 Servidor Web

Un servidor web es un programa que se ejecuta continuamente en un computador, manteniéndose a la espera de peticiones de ejecución que le hará un cliente o un usuario de Internet. El servidor web contesta estas peticiones de una forma adecuada, entregando una página web o información de acuerdo a los comandos solicitados.

1.6.1 Apache

Apache es el servidor web por excelencia, su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. (16)

Dentro de sus puntos fuertes se encuentran:

- Tiene interfaz de autenticación con Windows 9x/NT, Macintosh, Novell NetWare, OS/2, Linux y la mayoría de los Unix existentes: IRIX, Solaris, HPUX, SCO, FreeBSD, NetBSD, AIX y Digital Unix.
- Facilita la integración como *"plug-ins"* de lenguajes de programación de páginas web dinámicas.
- Provee interfaz a todas las bases de datos.
- Servidor altamente configurable de diseño modular: Se pueden escribir módulos para realizar determinadas funciones.
- Es flexible, rápido y eficiente.
- Se desarrolla de forma acelerada estimulando la retroalimentación desde sus usuarios a través de nuevas ideas, reportes de errores y parches.

- Multiplataforma.
- Es una tecnología gratuita de código fuente abierto. Esto le da una transparencia a este software de manera que si se quiere ver que es lo que se está instalando como servidor, se puede saber, sin ningún secreto.

1.7 Framework

Es un producto que sirve como base para la programación avanzada de aplicaciones, que aporta una serie de funciones o códigos para realizar tareas habituales. Un framework es un conjunto de librerías de código que contienen procesos o rutinas ya listos para usar. Los programadores utilizan los frameworks para no tener que desarrollar ellos mismos las tareas más básicas, puesto que en el propio framework ya hay implementaciones que están probadas, funcionan y no se necesitan volverse a programar. (17)

El concepto de framework viene aparejado con el desarrollo del software, es una estructura de soporte robusta y bien definida en la cual otro proyecto de software puede ser organizado y desarrollado. Los frameworks se han convertido en la piedra angular de la moderna ingeniería del software.

Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. El framework seleccionado para la propuesta de solución es CodeIgniter, el cual presenta las diversas características que se mencionan a continuación.

1.7.1 Framework *CodeIgniter*

CodeIgniter es un programa o aplicación web desarrollada en PHP para la creación de cualquier tipo de aplicación web. Es un producto de código libre. Como cualquier otro framework, contiene una serie de librerías que sirven para el desarrollo de aplicaciones web y propone una manera de desarrollarlas que se debe seguir para obtener un buen provecho de la aplicación. Este framework marca una manera específica de codificar las páginas web y clasificar sus diferentes *scripts*, que sirven para que el código esté organizado y sea más fácil de crear y mantener. (17)

Características generales de CodeIgniter**:**

- Compatibilidad: CodeIgniter es compatible con la versión *PHP 4*, lo que hace que se pueda utilizar en cualquier servidor, incluso en algunos antiguos. Funciona correctamente también en *PHP 5.*

- Versatilidad: Es quizás la característica fundamental que lo distingue de otros frameworks. Es capaz de trabajar la mayoría de los entornos o servidores, incluso en sistemas de alojamiento compartido, donde sólo hay un acceso por Protocolo de Transferencia de Archivos (FTP, por sus siglas en inglés) para enviar los archivos al servidor y donde no se tiene acceso a su configuración.
- Ligereza: Es ligero, lo que permite que el servidor no se sobrecargue interpretando o ejecutando grandes porciones de código, ya que la mayoría de los módulos o clases que ofrece se pueden cargar cuando se van a utilizar realmente.
- Flexibilidad: CodeIgniter es menos rígido que otros frameworks a pesar de que tiene una manera de trabajar específica, en muchos de los casos se puede o no seguir las reglas de codificación posibilitando trabajar como más a gusto se encuentre uno.
- Documentación tutorializada: La documentación de CodeIgniter es fácil de seguir y de asimilar. Posibilita la referencia rápida para consultar una función o un método en concreto.

Lo más destacable de CodeIgniter es su accesibilidad, ya que se puede utilizar en la mayor gama de entornos. Si lo que se busca es un rendimiento excepcional, este es el framework ideal. Contiene un marco que requiere casi cero configuraciones.

Al usar este framework se evita la complejidad, a favor de soluciones simples y se cuenta con la documentación completa. Además, implementa el proceso de desarrollo llamado Modelo Vista Controladora (MVC, por sus siglas en inglés), utilizado tanto para hacer sitios web, como programas tradicionales.

MVC

El Modelo Vista Controlador es interesante porque separa en varios grupos las complejidades de las distintas partes que componen una página web tales como; la interfaz de usuario, la lógica del negocio y los datos de una aplicación. (18)

Algunas ventajas de este estilo de arquitectura de software para los programas son:

- Más entendibles por los desarrolladores.
- Más reutilizables.
- De más fácil mantenimiento.

El MVC es un patrón de desarrollo o estilo de arquitectura de software que separa el código fuente de las aplicaciones en tres grupos:

Modelo:

En el modelo se preocupa de cosas como el tipo de base de datos, las tablas y sus relaciones, pero desde las otras partes del programa simplemente se llamará a las funciones del modelo sin importar qué tiene que hacer éste para conseguir realizar las acciones invocadas.

Vista:

La vista codifica y mantiene la presentación final de la aplicación de cara al usuario. Es decir, en la vista se colocará todo el código HTML, CSS, JavaScript que se tiene que generar para producir la página tal cual se quiere que la vea el usuario..

Controlador:

El controlador es la parte más importante, porque hace de enlace entre el modelo, la vista y cualquier otro recurso que se tenga que procesar en el servidor para generar la página web.

En el caso en que no se quiera seguir el desarrollo atendiendo a esta arquitectura, se puede tener simplemente controladores y dentro de ellos realizar todas las acciones de acceso a la base de datos directamente, sin hacer llamadas al modelo o escribir texto en la salida sin utilizar las vistas. (18)

Obviamente, esta opción no aprovechará las ventajas de separación de código entre presentación, lógica y modelo de base de datos.

1.8 Entornos de desarrollo

Un IDE[12] es un programa de aplicación. Consiste en un editor de código, compilador, depurador y constructor de interfaz gráfica. Pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes.

Los IDEs proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, C#, Delphi, Visual Basic y PHP. En algunos lenguajes, un IDE puede funcionar como un sistema en tiempo de ejecución, en donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto. (19)

[12] Entorno de Desarrollo Integrado

El IDE seleccionado por el Departamento de Gestión Universitaria de la UCI para la propuesta de solución es Netbeans por las ventajas que se mencionan a continuación.

1.8.1 NetBeans

Netbeans es un IDE sumamente completo, fácil de usar, cómodo, de excelente calidad y es completamente gratis. Es muy famoso entre los programadores de Java hoy en día, por lo que hay mucha información al respecto. Es un proyecto de código abierto de gran éxito, con una gran comunidad en constante crecimiento y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó este proyecto en junio 2000 y continúa siendo el patrocinador principal.

La plataforma Netbeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos por lo que pueden ser extendidas fácilmente por otros desarrolladores de software. (19)

NetBeans IDE es una aplicación de código abierto diseñada para el desarrollo de aplicaciones fácilmente portables entre las distintas plataformas. Hace uso de la tecnología Java y dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones y creación de aplicaciones compatibles con teléfonos móviles.

Es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Aunque está escrita en Java puede servir para cualquier otro lenguaje de programación. Como existe un gran número de módulos para extender el NetBeans IDE, se puede decir que esta herramienta sea la única que se necesite a la hora de comenzar un proyecto. (19)

1.9 Plataforma de Desarrollo.

Una plataforma de desarrollo es el entorno de software común en el cual se desenvuelve la programación de un grupo definido de aplicaciones. Comúnmente se encuentra relacionada directamente a un sistema operativo. Además, es posible encontrarla ligada a una familia de lenguajes de programación o a un API[13].

[13] Interfaz de programación de aplicaciones.

1.9.1 GNU/Linux.

GNU/Linux es un sistema operativo de software libre que cumple las normas de POSIX[14], su base es un núcleo o Kernel monolítico llamado Linux combinado con un grupo de librerías y herramientas. Su estructura general es la típica de cualquier sistema UNIX (núcleo, "intérprete de comandos", aplicaciones). (20)

GNU/Linux tiene todas las características que se pueden esperar de un moderno y flexible sistema operativo. Incluye multitarea real, memoria virtual, librerías compartidas, dirección y manejo propio de memoria. Es sin lugar a dudas uno de los ejemplos más prominentes del software libre y del desarrollo del código abierto.

La plataforma Linux ofrece potencia, estabilidad y seguridad contra virus. Además cuenta con las siguientes características:

- Facilidad de uso.
- Alta fiabilidad.
- Mayor velocidad.
- Alta Funcionalidad.
- Es gratis, ya que al ser un software libre no requiere licencia, lo que permite abaratar costos.

Linux es mucho más que una plataforma económica en el mercado, dado que es el precursor de las premisas fundamentales sobre cómo se construye el software hoy en día. Desde sus humildes orígenes como un experimento técnico a su actual función como base de un enorme ecosistema de tecnologías libres y comerciales.

La recesión económica está obligando a las empresas a consolidar su infraestructura técnica, y Linux, que trabaja en una serie de contextos más amplios que sus competidores, puede ser potencialmente usado para reducir los costos. Linux está ganando mucho terreno debido a esto y se piensa que poco a poco vaya mejorando su posición para convertirse en líder a largo plazo en el mercado de sistemas operativos. (20)

[14] Interfaz para Sistemas Operativos migrables basados en UNIX.

Varias son las distribuciones de GNU/Linux que se han hecho en el mundo gracias al trabajo constante de los desarrolladores y promotores del software libre en el mundo entero. Ejemplo de estas: Debian, Mandriva, Ubuntu, Novel/Suse, Red Hat y Gentoo.

Para el desarrollo del módulo se seleccionó la plataforma de desarrollo Ubuntu Linux.

1.10 Metodologías de desarrollo de software

Todo desarrollo de software es riesgoso y difícil de controlar, pero si no se sigue una metodología de por medio, lo que se obtiene son clientes insatisfechos con el resultado y desarrolladores aún más insatisfechos.

En la solución propuesta se utilizó una metodología ágil definida por el Departamento de Gestión Universitaria de la UCI.

1.10.1 Metodologías ágiles

Las metodologías ágiles intentan evitar los tortuosos y burocráticos caminos de las metodologías tradicionales enfocándose en la gente y los resultados. Promueven iteraciones en el desarrollo a lo largo de todo el ciclo de vida del proyecto, logrando que se minimicen los riesgos a la hora de desarrollar un software en corto tiempo. El software desarrollado en una unidad de tiempo es llamado iteración. Cada iteración del ciclo de vida incluye: planificación, análisis de requerimientos, diseño, codificación, revisión y documentación. (21)

Una iteración no debe agregar demasiada funcionalidad para justificar el lanzamiento del producto al mercado. La meta al final de cada iteración, es lograr un demo sin errores para que el equipo vuelva a evaluar las prioridades del proyecto.

SXP (Scrum -XP)

Programación Extrema (XP)

XP es un proceso ligero, de bajo riesgo, flexible, predecible y científico para desarrollar un software. Algunos conocedores y desarrolladores lo describen como pura lógica. Las prácticas de este método, están basadas en la simplicidad, la comunicación y el reciclado continúo de código. (22)

Esta metodología propone un cambio revolucionario de llevar las prácticas al extremo, tiene como fin la integración del cliente en el equipo de trabajo, adaptación a los cambios ya sean de tecnología o de

metáforas dentro del negocio y la satisfacción del cliente. Además, es la integración de las prácticas de métodos tradicionales, logrando que se resuman y utilice lo más práctico y eficaz.

Scrum

Scrum es una metodología para el desarrollo ágil de productos, expuesta por Hirotaka Takeuchi[15] e Ikujiro Nonaka[16], los cuales exponen según sus ideas que en el mercado competitivo de los productos tecnológicos, además de los conceptos básicos de calidad, coste y diferenciación, se exige también rapidez y flexibilidad. Esta metodología surge por la exigencia del mercado de ciclos de desarrollo más cortos.

Aunque surgió como modelo para el desarrollo de productos tecnológicos, también se emplea en entornos que trabajan con requisitos inestables y que requieren rapidez y flexibilidad; situaciones frecuentes en el desarrollo de determinados sistemas de software.

Scrum es una metodología de desarrollo muy simple, que requiere trabajo duro, porque la gestión no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto. (23)

La metodología seleccionada por el Departamento de Gestión Universitaria es SXP, la cual es la mezcla de Scrum y XP.

Dicha mezcla ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva, fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo. (24)

Consta de 4 fases principales:

- **Planificación-Definición**: se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.

[15] Es decano en la Universidad Hitotsubashi en Tokio. Fue profesor visitante en la Escuela de Negocios Harvard entre 1989 y 1990. Obtuvo una Maestría en Administración de Negocios (MBA, por sus siglas en inglés) en la Escuela de Negocios Haas de la Universidad de California, en Berkeley.

[16] Escribió el libro "La Luz y la Sombra". Nació el 10 de mayo de 1935. Es profesor emérito de la Universidad Hitotsubashi. En el 2008 el *Wall Street Journa*l lo listó como una de las personas con el pensamiento más influyente en el área de negocios.

- **Desarrollo**: se realiza la implementación del sistema hasta que esté listo para ser entregado.
- **Entrega**: puesta en marcha.
- **Mantenimiento**: se realiza el soporte para el cliente.

De cada una de estas fases se realizan numerosas actividades tales como el levantamiento de requisitos, la priorización de la Lista de Reserva del Producto, definición de las HU, Diseño, Implementación, Pruebas, entre otras; de donde se generan artefactos para documentar todo el proceso. Las entregas son frecuentes, y existe una refactorización continua, que permite mejorar el diseño cada vez que se añade una nueva funcionalidad.

SXP está especialmente indicada para proyectos de pequeños equipos de trabajo, rápido cambio de requisitos o requisitos imprecisos, muy cambiantes, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar, de forma que los jefes pueden ver día a día cómo progresa el trabajo. (24)

1.10 Conclusiones

En el desarrollo del capítulo se especifica la selección por parte del Departamento de Gestión Universitaria de la UCI, de las herramientas y tecnologías utilizadas para la implementación de la propuesta de solución. Como lenguaje de programación del lado del cliente HTML y JavaScript y del lado del servidor PHP, utilizando como framework CodeIgniter el cual se complementa muy bien con dicho lenguaje.

Toda la implementación fue sobre la plataforma de desarrollo Ubuntu GNU/Linux, como sistema gestor de base de datos PostgreSQL, el entorno de desarrollo utilizado fue NetBeans y la metodología de desarrollo del software SXP. Dichas herramientas y tecnologías se seleccionan teniendo en cuenta las políticas de migración del país y de la Universidad hacia el software libre; por la flexibilidad, potencia, y múltiples posibilidades que este ofrece.

CAPÍTULO 2 DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA

2.1 Introducción

En este capítulo se describen las funcionalidades de las HU[17] implementadas y sus respectivas tareas de ingenierías. Se utiliza el estándar de codificación definido por el Departamento de Gestión Universitaria para el tratamiento de los algoritmos más importantes implementados. Además, se presentan las descripciones de las clases fundamentales con sus métodos esenciales.

2.2 Diseño propuesto por el analista

Al utilizar la metodología SXP se generaron diferentes artefactos. Unos de los más importantes son las HU, pues son la forma en que se especifican los requisitos del sistema. Se redactan desde la perspectiva del cliente aunque los desarrolladores pueden brindar también su ayuda en la identificación de las mismas. El contenido que ellas abarcan debe ser concreto y sencillo, que permitan hacerse una ligera idea de cuánto tiempo llevará implementar el sistema. Emplean terminología del cliente sin lenguaje técnico. Se realiza una por cada característica principal del sistema.

Se emplean para hacer estimaciones de tiempo y para el plan de lanzamientos. Reemplazan un gran documento de requisitos y presiden la creación de las pruebas de aceptación. Estas deben proporcionar sólo el detalle suficiente y difieren de los casos de usos porque son escritos por el cliente, no por los programadores. Cada HU debe contar con las interfaces correspondientes para la solución de las mismas.

Para la implementación del módulo Reportes y Estadísticas se han definido las siguientes HU:

HU Reportes de Matrícula

- Cantidad de estudiantes por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.
- Listar estudiantes matriculados por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.

[17] Historias de Usuario.

HU Reportes de Bajas

- Cantidad de estudiantes de bajas por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.
- Listar bajas por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.

HU Reportes de Licencias

- Cantidad de estudiantes de licencias por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.
- Listar licencias por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.

HU Reportes de Traslados

- Cantidad de estudiantes de traslados por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.
- Listar traslados por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.

HU Reportes de Egresados

- Listar egresados por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.
- Cantidad de estudiantes egresados por provincia, tipo de curso, plan de estudio, situación militar, año académico, municipio, sexo, carrera, modalidad de estudio y vía de ingreso.

Se identificaron las funcionalidades a implementar para que el módulo funcione correctamente, basándose en la descripción detallada de las HU. Además, se tomó como base la propuesta de los prototipos de interfaz, las cuales cumplen con las exigencias del cliente, pues hacen más fácil y cómodo el trabajo y sirven de guía para la implementación de la propuesta de solución.

2.3 Descripción de las HU

2.3.1 HU Reportes de Matrícula

La HU permite obtener los reportes de manera general referente a todos los estudiantes matriculados en la Universidad.

Estudiantes matriculados

Al seleccionar en el menú principal del módulo la opción de Estudiantes matriculados, se muestra una cuadrícula con dos pestañas llamadas Nominales y Cuantitativos. Dentro de la pestaña Nominales existe un buscador de búsqueda rápida “Buscar y seleccionar estudiantes” por los criterios (Nombre, usuario, CI o Apellidos) y un filtrar “Filtros de Búsqueda” que contiene determinados nomencladores (Carrera, Modalidad, Tipo de curso, Plan de Estudio, Sexo, Municipio, Provincia, Situación militar, Vía de Ingreso y Año académico). Cuenta además con un botón “Cancelar” para no seguir realizando la búsqueda. Ver Anexo 1.

El usuario para realizar la búsqueda de estudiantes selecciona uno de los filtros o escribe el criterio que desee en el buscador y se listarán los estudiantes que tengan el estado activo de matriculado en la base de datos, en dependencia de lo escogido. El listado de los estudiantes se mostrará en un paginado donde se verán siempre datos fijos como el CI, Primer Nombre, Primer Apellido y Segundo Apellido. Ver Anexo 2.

Al final de dichos datos aparece una columna llamada opción, en la cual se encuentra un botón en forma de lupa, que permitirá cargar la hoja de ratificación de matrícula del estudiante seleccionado. Ver Anexo 3.

Estadísticas de los estudiantes matriculados

En la pestaña de Cuantitativos se filtrará la búsqueda de la misma forma que en la de Nominales, con la diferencia que ahora se mostrará la cantidad de estudiantes matriculados activos en dependencia del filtro que se escoja. Por ejemplo si el usuario selecciona el filtro sexo entonces saldrá una tabla l “Cantidad de matriculados por Sexo” que contiene las columnas “Masculino”, “Femenino” y “Cantidad”, donde las dos primeras tienen la cantidad de estudiantes que almacena la base de datos con esas características y la última el total de la suma de estas. También cuenta con el botón “Cancelar” para no continuar la búsqueda. Ver Anexo 4.

Exportar a PDF

Luego de haber generado los reportes de matrícula, si el usuario lo desea los puede exportar a PDF. Existe un ícono en la cuadrícula “Estudiantes matriculados” que permite realizar esta opción. Cuando el documento ya esta en PDF se le da la posibilidad de guardarlo o imprimirlo al usuario, lo cual facilita y cómodo a las secretarias que constantemente trabajan con estas informaciones en la Universidad.

2.3.2 HU Reportes de Movimiento

La HU se divide en cuatro:

- HU Reportes de Bajas
- HU Reportes de Licencias
- HU Reportes de Egresados
- HU Reportes de Traslados

Estas HU permiten conocer la información general referente a los estudiantes que son bajas, licencias, traslados o egresados de la Universidad.

HU Reportes de Movimiento

Se selecciona en el menú principal del módulo la opción que se desea saber acerca de algunos de los reportes de movimiento. Por ejemplo si desea conocer los estudiantes de baja debe seleccionar “Estudiantes de baja” y lo mismo pasaría para los demás Reportes de Movimiento. A continuación se mostraría la cuadrícula correspondiente de “Buscar y seleccionar estudiantes de baja”. Dicha cuadrícula contiene las pestañas Nominal y Cuantitativo, se hace la misma operación que para los estudiantes matriculados, pero ahora solo se listarán los que tengan uno de los estados (baja, licencia, egresado o traslado).

Estadísticas de los Reportes de Movimiento

Para estos reportes pasa lo mismo que para las estadísticas de los estudiantes matriculados. La diferencia es que ahora se mostrará la cantidad de estudiantes baja, traslado, egreso o licencia, en dependencia del filtro que se seleccione.

Exportar a PDF

Permite llevar a formato PDF y guardar e imprimir los reportes generados por el usuario sobre un determinado estudiante (baja, traslado, licencia o egresados).

2.4 Descripción de las clases de las HU implementadas y sus operaciones necesarias

Tabla 2. 1 Clase matrícula

Nombre de la clase: matrícula	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al tab matrícula que se encuentra dentro de las vistas.

Tabla 2. 2 Clase matrícula_cuantitativos

Nombre de la clase: matrícula_cuantitativos	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al método listar() que se encuentra dentro de la misma clase. Además, carga la barra de botones y el *index Ajax*.
Nombre:	listar()
Descripción:	Crea los filtros y hace un llamado a los diferentes métodos que permiten obtener dichos filtros, los cuales se encuentran dentro de una librería. Además, carga la vista de matrícula_cuantitativos.
Nombre:	BuscarEstudiante()
Descripción:	Carga un componente *grid json* el cual hace un llamado a un método que está dentro de una librería obtenerCantidadEstudiantes().

Tabla 2. 3 Clase matrícula_nominales

Nombre de la clase: matrícula_nominales	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al método listar() que se encuentra dentro de la misma clase. Además, carga la barra de botones y el *index Ajax*.
Nombre:	listar()
Descripción:	Crea los filtros y hace un llamado a los diferentes métodos que permiten obtener dichos filtros, los cuales se encuentran dentro de una librería. Además, carga la vista de matrícula nominales.
Nombre:	Buscar_estudiante()
Descripción:	Carga un componente grid json el cual hace un llamado a dos métodos que están dentro de una librería obtenerCantidadEstudiantes() y obtenerEstudiantesMatriculados().
Nombre:	exportarPDF()
Descripción:	Exporta a PDF los resultados de los reportes de la búsqueda de estudiantes matriculados.

Tabla 2. 4 Baja.

Nombre de la clase: Baja	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al tab baja que se encuentra dentro de las vistas.

Tabla 2. 5 Baja_nominales.

Nombre de la clase: baja_nominales	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al método listar() que se encuentra dentro de la misma clase. Además carga la barra de botones y el *index Ajax*.
Nombre:	listar()
Descripción:	Crea los filtros y hace un llamado a los diferentes métodos que permiten obtener dichos filtros los cuales se encuentran dentro de una librería. Además carga la vista de listar_baja_nominales_*view*.
Nombre:	Buscar_estudiante()
Descripción:	Carga un *grid json* el cual hace un llamado a dos métodos que están dentro de una librería obtenerCantidadEstudiantesBaja() y obtenerEstudiantesBaja ().
Nombre:	exportarPDF()
Descripción:	Exporta a PDF los resultados de los reportes de la búsqueda de estudiantes de bajas.

Tabla 2. 6 Baja_cuantitativos.

Nombre de la clase: baja_cuantitativos	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al método listar() que se encuentra dentro de la misma clase. Además carga la barra de botones y el *index Ajax*.
Nombre:	listar()
Descripción:	Crea los filtros y hace un llamado a los diferentes métodos que permiten obtener dichos filtros, los cuales se encuentran dentro de una librería. Además carga la vista de listar_baja_cuantitativos_*view*.

Nombre:	BuscarEstudiante()
Descripción:	Carga un *grid json* el cual hace un llamado a un método que está dentro de una librería obtenerCantidadEstudiantesBaja().

Tabla 2. 7 Traslado.

Nombre de la clase: Traslado	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al tab traslado que se encuentra dentro de las vistas.

Tabla 2. 8 Traslados_nominales

Nombre de la clase: traslados_nominales	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al método listar() que se encuentra dentro de la misma clase. Además carga la barra de botones y el *index Ajax*.
Nombre:	listar()
Descripción:	Crea los filtros y hace un llamado a los diferentes métodos que permiten obtener dichos filtros, los cuales se encuentran dentro de una librería. Además carga la vista de listar_traslados_nominales_*view*.
Nombre:	Buscar_estudiante()
Descripción:	Carga un *grid json* el cual hace un llamado a dos métodos que están dentro de una librería obtenerCantidadEstudiantesTraslados() y obtenerEstudiantesTraslados().
Nombre:	exportarPDF()
Descripción:	Exporta a PDF los resultados de los reportes de la búsqueda de estudiantes de traslados.

Tabla 2. 9 Traslados_cuantitativos.

Nombre de la clase: traslados_cuantitativos	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al método listar() que se encuentra dentro de la misma clase. Además carga la barra de botones y el *index* Ajax.
Nombre:	listar()
Descripción:	Crea los filtros y hace un llamado a los diferentes métodos que permiten obtener dichos filtros, los cuales se encuentran dentro de una librería. Además carga la vista de listar_traslados_cuantitativos_*view*.
Nombre:	BuscarEstudiante()
Descripción:	Carga un *grid json* el cual hace un llamado a un método que está dentro de una librería obtenerCantidadEstudiantesTraslados().

Tabla 2. 10 Licencia.

Nombre de la clase: Licencia	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al tab licencia que se encuentra dentro de las vistas.

Tabla 2. 11 Licencias_nominales.

Nombre de la clase: licencia_nominales	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al método listar() que se encuentra dentro de la misma clase. Además carga la barra de botones y el *index Ajax*.

Nombre:	listar()
Descripción:	Crea los filtros y hace un llamado a los diferentes métodos que permiten obtener dichos filtros, los cuales se encuentran dentro de una librería. Además carga la vista de listar_licencia_nominales_*view*.
Nombre:	Buscar_estudiante()
Descripción:	Carga un *grid json* el cual hace un llamado a dos métodos que están dentro de una librería obtenerCantidadEstudiantesLicencia() y obtenerEstudiantesLicencia().
Nombre:	exportarPDF()
Descripción:	Exporta a PDF los resultados de los reportes de la búsqueda de estudiantes de licencias.

Tabla 2. 12 Licencia_cuantitativos.

Nombre de la clase: licencia_cuantitativos	
Tipo de clase: controladora	
Para cada responsabilidad:	
Nombre:	*index*()
Descripción:	Hace un llamado al método listar() que se encuentra dentro de la misma clase. Además carga la barra de botones y el *index Ajax*.
Nombre:	listar()
Descripción:	Crea los filtros y hace un llamado a los diferentes métodos que permiten obtener dicho filtro los cuales se encuentran dentro de una librería. Además carga la vista de listar_licencia_cuantitativos_*view*.
Nombre:	BuscarEstudiante()
Descripción:	Carga un *grid json* el cual hace un llamado a un método que está dentro de una librería obtenerCantidadEstudiantesLicencias().

Se describen las funcionalidades de las clases vistas baja, egresado, licencia y traslado con sus métodos fundamentales. Ver Anexo 5.

2.5 Estilos de programación

La motivación que tienen los estilos de programación esta relacionada con la comunicación entre personas que facilitan los lenguajes de programación. Las reglas de los estilos son muy flexibles, lo cual no significa que se va a estar cambiando el estilo en cada momento, todo lo contrario, es importante mantener las reglas rígidas dentro de un proyecto. No basta con escribir un programa que funcione, el código tiene que estar bien escrito para que se construya un buen programa. (25)

Diversas son las cualidades que se benefician por el buen uso de un estilo de programación, entre las cuales se encuentran:

- **Extensibilidad**: La facilidad con que se adapta el software a cambios de especificación. Un buen estilo de código fomenta programas que no sólo resuelven el problema, sino que también reflejan claramente la relación problema/solución. Esto tiene como efecto que muchos cambios simples en el problema reflejen de forma obvia los cambios a hacer en el programa.
- **Verificabilidad**: La facilidad con que pueden comprobarse propiedades de un sistema. Si el estilo de código hace obvia la estructura del programa, eso ayuda a verificar que el comportamiento sea el esperado.
- **Reparabilidad**: La posibilidad de corregir errores sin demasiado esfuerzo.
- **Capacidad de evolución**: La capacidad de adaptarse a nuevas necesidades.
- **Comprensibilidad**: La facilidad con que el programa puede ser comprendido.

2.6 Estándares de codificación

2.6.1 Identación, llaves de apertura y cierre, y tamaño de las líneas.

En el desarrollo del sistema de Gestión Universitaria se ha establecido un estándar de codificación, como usar una identación sin tabulaciones, con un equivalente a 4 espacios, para mantener integridad en las revisiones svn. El uso de las llaves” {}” es en una nueva línea. La longitud de las líneas de código es aproximadamente de 75-80 caracteres para mantener la legibilidad del código. (26)

Ejemplo:

```
1 ....$a = $b;
2
3 ....function ejemplo()
4 ....{
5     ....//BI
6 ....}
```

Figura 4 Longitud de las líneas de un código.

2.6.2 Conversión de nomenclatura

Las variables se rigen por la nomenclatura camelCase. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula.

Ejemplo:

```
1 ....$variable
2 ....$variableNombreCompuesto
```

Figura 5 Nomenclatura camelCase con letra minúscula.

Las constantes son escritas en mayúsculas, con caracteres de subrayado "_" para separar palabras en caso de nombres compuestos.

Ejemplo:

```
1 ....define(CONSTANTE,valor);
2 ....define(CONSTANTE_COMPUESTO,valor);
```

Figura 6 Nomenclatura camelCase con letra mayúscula.

Los nombres de las clases siempre comienzan con mayúscula, en caso de nombre compuesto las palabras se separan con el caracter subrayado "_" y el resto en minúscula.

Ejemplo:

```
1 ....class Clase
2 ....{
3     ....//BI
4 ....}
5
6 ....class Clase_nombre_compuesto
7 ....{
8     ....//BI
9 ....}
```

Figura 7 Nomenclatura camelCase para una clase.

Las funciones se rigen por la nomenclatura camelCase. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Los parámetros son separados por espacio luego de la coma que los separa. (26)

Ejemplo:

```
1 ....function funcion($parametro1,.$parametro2)
2 ....{
3     ....//BI
4 ....}
5
6 ....function funcionNombreCompuesto($parametro1,.$parametro2)
7 ....{
8     ....//BI
9 ....}
```

Figura 8 Nomenclatura camelCase para una función.

Los ficheros siempre se escriben con minúscula y en caso de nombres compuestos se usa el caracter subrayado"_".

Vistas: intuitivo y relacionado con el formulario y/o vista que representa.

Modelos: con el mismo nombre de la clase que representa que contiene en el nombre el sufijo _mdl o _base en caso de ser modelos base.

Librerías: con el mismo nombre de la clase que representa que contiene en el nombre el sufijo: _lib.

Controladoras: con el mismo nombre de la clase que representa.

Manager: con el mismo nombre de la clase que representa que contiene en el nombre el sufijo: _mng.

2.6.3 Estructuras de control

Se incluye un espacio entre las estructuras de control (if, for, foreach, while, switch) y los paréntesis. Se recomienda utilizar siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales. Esto aumenta la legibilidad y disminuye la probabilidad de errores lógicos.

Ejemplos:

```
1 ....if.(condicion)
2 ....{
3     ....//BI
4 ....}
5 ....elseif(condicion)
6 ....{
7     ....//BI
8 ...}
9 ....else
```

Figura 9 Ejemplo 1 de estructuras de control.

```
10....{
11        ....//BI
12....}
13
14....switch.(valor)
15....{
16        ....case valor1:
17            ....//BI para valor1
18        ....break;
19        ....case valor2:
20            ....//BI para valor2
21        ....break;
22        ....default:
23            ....//BI por defecto
24....}
```

Figura 10 Ejemplo 2 de estructuras de control.

Si las condiciones son muy largas que sobrepasan el tamaño de la línea, éstas se dividen en varias líneas.

Ejemplo:

```
1 ....if.(condicion1
2 ....|| condicion2)
3 ....||(condicion3
4 ....&& condicion4))
5 ....{
6         ....//BI
7 ....}
```

Figura 11 Ejemplo 3 de estructuras de control.

En el mejor de los casos cuando la condición es muy extensa, se puede dividir en variables y compararlas dentro de la estructura de control. (26)

Ejemplo:

```
1 ....$variableCondicion1 = condicion1 || condicion2;
2 ....$variableCondicion2 = condicion3 && condicion4;
3
4 ....if.($variableCondicion1 || $variableCondicion2)
5 ....{
6     ....//BI
7 ....}
```

Figura 12 Ejemplo 4 de estructuras de control.

2.6.4 Documentación

Todos los archivos deben de tener la documentación asociada al mismo. Para esto se debe cumplir con el siguiente bloque al principio de cada clase. (26)

Clase:

```
1 /**
2 *Breve descripción de la clase
3 *
4 *PHP versión #
5 *
6 *@category Categoría de la clase implementada "Libreria,
7 * Controladora, Modelo"
8 *@package  Nombre del paquete o módulo al que pertence
9 *@author  Nombre y Apellidos del autor y  correo electrónico
10*/
```

Figura 13 Documentación de una clase.

Funciones:

```
1 /**
2 *Breve descripción de la función
3 *
4 *@param tipo y nombre del paremetro(por cada parametro que
5 * recibe la función)
6 *@return  tipo que retorna
7 *@author  Nombre y Apellidos del autor y  correo electrónico
8 */
```

Figura 14 Documentación de una función.

2.6.5 Buenas prácticas

Los valores booleanos y nulos siempre se escriben con mayúscula, para facilitar la legibilidad del código usar una línea en blanco antes de las estructuras de control y definición de las funciones. (26)

```
1 ....$variableBooleana = FALSE;
2 ....$variableNula = NULL;
3 ....
4 ....if(condicion)
5 ....{
6     ....//BI
7 ....}
```

Figura 15 Valores booleanos y nulos.

2.6.6 Pautas para la implementación de las HU.

El desarrollo en la metodología utilizada por el sistema de Gestión Universitaria está guiado por las HU y a su vez el desarrollo de estas HU estará guiado y dirigido por las tareas de ingeniería que se generan de una HU.

Para la implementación de las tareas de ingeniería se definen algunos patrones a seguir:

1. Regirse por el estándar de código definido para la implementación del proyecto.

2. Todas las interfaces interactúan mediante Ajax con el servidor.

3. Las acciones se dividen generalmente en dos métodos en las controladoras para lograr una mayor legibilidad del código:

- Para obtener los datos necesarios en caso de que lo necesite y mostrar la vista. El nombre usado para estos métodos se escribe en infinitivo, ejemplo: crear, listar, modificar, detallar, asociar.
- Para obtener y validar los datos recibidos desde la vista, interactúa con la librería asociada y crear el mensaje que se envía a la vista. El nombre usado es el mismo infinitivo usado en el otro método con el o los elementos afectados en la acción ejemplo: crearGrupo, modificarGrupo, asociarTrabajadorGrupo, en el caso de que no sea necesario mostrar una vista, éste se escribe en infinitivo ejemplo: eliminar, activar.

4. En los métodos implementados en las clases modelo, las consultas no se hacen con código SQL directamente, se realiza utilizando el *active record* de CodeIgniter.

Ejemplo:

```
1 ....$this->db->select('sq_esquema.tb_ddatos.id_datos');
3 ....$this->db->where ($key, $value);
4 ....$this->db->get ('sq_esquema.tb_ddatos')->result();
```

Figura 16 Métodos con active record de CodeIgniter.

5. Para la creación de las interfaces se utilizarán las funciones del helper form de CodeIgniter para garantizar homogeneidad en el html, ejemplo: form_open, form_dropdown, form_input.

6. Los mensajes de error en la vista se lanzan con un throw Exception (“mensaje”).

7. No se utiliza el .load de jQuery pues el envío de los datos los hace por get y para garantizar un poco de seguridad se deben enviar por post siempre, para esto hay que utilizar $.Ajax.

8. En los métodos de las controladoras que serán encuestados mediante Ajax, se utiliza echo y no die para mostrar los datos o la vista cargada, esto garantiza el buen funcionamiento de la programación orientada a aspectos.

9. Todas las implementaciones se realizan en español. (26)

2.7 Tareas de ingeniería.

Las tareas de ingeniería son actividades que los programadores conocen que el sistema debe hacer. Deben ser estimables, y poder ser implementadas entre uno y tres días ideales. La mayoría de estas tareas se derivan directamente de las HU. Existen dos tipos de tareas de ingeniería las que provienen de las historias de usuario y las técnicas.

Cada historia de usuario puede ser dividida en varias tareas de ingeniería sencillas. Para determinar las tareas que componen a una historia de usuario se propone hacer una reunión con todos los miembros del equipo de desarrollo y obtener una buena lista con todas.

Las tareas de ingeniería técnicas son aquellas que no son resultado del análisis de ninguna historia de usuario pero deben ser realizadas para que el sistema funcione. Cada tarea de ingeniería será comprobada a través de los casos de prueba y no tienen por qué ser comprendidas por el cliente.

2.7.1 Descripción de las tareas de ingeniería.

Tabla 2. 13 Desarrollar formulario para los Reportes de Matrícula

Tarea de ingeniería	
Número tarea: HU1-1	Número historia: 1
Nombre tarea: Implementar el código para la vista de los reportes de los estudiantes matriculados.	
Tipo de tarea: Desarrollo	Puntos estimados:
Fecha de inicio: 25/03/2010	Fecha de fin: 9/04/2010
Programador responsable: Yoander Peña Pérez y Yeny Lauris Milanés Regalado.	
Descripción: Se seleccionan los filtros que desee el usuario y se realiza la búsqueda de estudiantes matriculados.	

Tabla 2. 14 Desarrollar botón para la vista de detalles.

Tarea de ingeniería	
Número tarea: HU1-2	Número historia: 1
Nombre tarea: Implementar el código para la funcionalidad que muestra la vista de detalles.	
Tipo de tarea: Desarrollo	Puntos estimados:
Fecha de inicio: 10/04/2010	Fecha de fin: 24/04/2010
Programador responsable: Yoander Peña Pérez y Yeny Lauris Milanés Regalado.	
Descripción: Se selecciona el botón en forma de lupa y se muestra la hoja de matrícula del estudiante seleccionado por el usuario.	

Tabla 2. 15 Desarrollar botón para graficar.

Tarea de ingeniería	
Número tarea: HU1-3	Número historia: 1
Nombre tarea: Implementar el código para exportar a PDF los reportes generados.	
Tipo de tarea: Desarrollo	Puntos estimados:

Fecha de inicio: 25/04/2010	Fecha de fin: 8/05/2010
Programador responsable: Yoander Peña Pérez y Yeny Lauris Milanés Regalado.	
Descripción: Se selecciona el ícono para la opción de exportar a PDF y luego se puede imprimir el mismo.	

Tabla 2. 16 Desarrollar formulario para los Reportes de Bajas.

Tarea de ingeniería	
Número tarea: HU2.1-1	Número historia: 2.1
Nombre tarea: Implementar el código para la vista de los reportes de estudiantes de bajas.	
Tipo de tarea: Desarrollo	Puntos estimados:
Fecha de inicio: 9/05/2010	Fecha de fin: 16/05/2010
Programador responsable: Yoander Peña Pérez y Yeny Lauris Milanés Regalado.	
Descripción: Crear el código que permita realizar reportes de bajas. Se especifica la carrera, año, sexo, entre otros filtros.	

Tabla 2. 17 Desarrollar formulario para los Reportes de Licencias.

Tarea de ingeniería	
Número tarea: HU2.2-1	Número historia: 2.2
Nombre tarea: Implementar el código para la vista de los reportes de estudiantes de licencias.	
Tipo de tarea: Desarrollo	Puntos estimados:
Fecha de inicio: 17/05/2010	Fecha de fin: 22/05/2010
Programador responsable: Yoander Peña Pérez y Yeny Lauris Milanés Regalado.	
Descripción: Crear el código que permita realizar reportes de licencias. Se especifica la carrera, año, sexo, entre otros filtros.	

Tabla 2. 18 Desarrollar formulario para los Reportes de Traslados.

Tarea de ingeniería	
Número tarea: HU2.3-1	Número historia: 2.3
Nombre tarea: Implementar el código para la vista de los reportes de estudiantes de traslados.	
Tipo de tarea: Desarrollo	Puntos estimados:
Fecha de inicio: 23/05/2010	Fecha de fin: 30/05/2010
Programador responsable: Yoander Peña Pérez y Yeny Lauris Milanés Regalado.	
Descripción: Crear el código que permita realizar reportes de traslados. Se especifica la carrera, año, sexo, entre otros filtros.	

Tabla 2. 19 Desarrollar formulario para los Reportes de Egresados.

Tarea de ingeniería	
Número tarea: HU2.4-1	Número historia: 2.4
Nombre tarea: Implementar el código para la vista de los reportes de estudiantes de traslados.	
Tipo de tarea: Desarrollo	Puntos estimados:
Fecha de inicio: 31/05/2010	Fecha de fin: 6/06/2010
Programador responsable: Yoander Peña Pérez y Yeny Lauris Milanés Regalado.	
Descripción: Crear el código que permita realizar reportes de traslados. Se especifica la carrera, año, sexo, entre otros filtros.	

2.8 Conclusiones

Al finalizar el capítulo queda implementado el sistema y descritas todas las clases utilizadas, cumpliendo con los objetivos específicos planteados y las tareas propuestas para la elaboración de la solución. A partir de una breve descripción de cómo tener acceso a las funcionalidades implementadas dentro del módulo de Reportes y Estadísticas, se le da paso a la etapa de pruebas y se realizan los ensayos necesarios para asegurar que el sistema esté libre de no conformidades y con la debida calidad para ser entregado al cliente.

CAPÍTULO 3 VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

3.1 Introducción

En el presente capítulo se hace una valoración de los principales artefactos de la Fase de Prueba. Se describen las Pruebas Unitarias y de Aceptación que son las que utiliza la metodología SXP. Además, son realizadas las plantillas de las Pruebas de Aceptación para cada una de las HU implementadas, ya que son las seleccionadas por el Departamento de Gestión Universitaria para la validación de la propuesta de solución.

3.2 Pruebas

Un instrumento adecuado para determinar el *status* de la calidad de un producto es el proceso de pruebas. Ver figura 4.

En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos. La prueba no puede asegurar la ausencia de defectos; sólo puede demostrar que existen errores. (27)

El proceso de desarrollo de las pruebas ayuda al cliente a refinar, concretar, identificar, corregir fallos u omisiones en la funcionalidad de las historias de usuarios. Favorece la comunicación entre el cliente y el equipo de desarrollo. Identifica historias adicionales que no fueran obvias para el cliente o en las que cliente no hubiese pensado de no enfrentarse a dicha situación. Además, permite corregir errores en las ideas del cliente, encontrando resultados en la implementación, que ningún camino de ejecución conducía a ello.

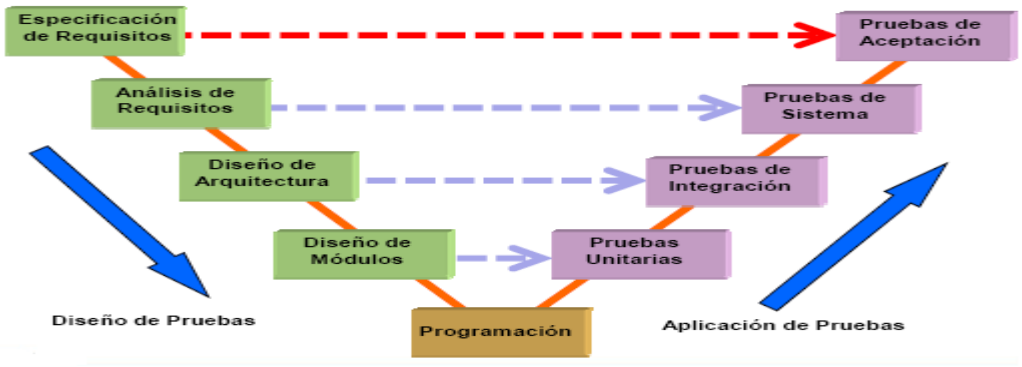

Figura 17 Modelo de Prueba

En la metodología SXP las pruebas de funcionalidad se realizan a través de los casos de prueba. La metodología divide las pruebas en dos grupos: pruebas unitarias, desarrolladas por los programadores, encargadas de verificar el código de forma automática y las pruebas de aceptación que son las destinadas a comprobar que la funcionalidad implementada sea la esperada por el cliente. Las pruebas funcionales se escriben para cada historia de usuario que deba validarse.

3.2.1 Pruebas unitarias

Las pruebas unitarias consisten en comprobaciones manuales o automatizadas que son realizadas para verificar que el código correspondiente a un módulo concreto de un sistema de software funciona de acuerdo con los requisitos del sistema. Son realizadas a pequeña escala y prueban uno a uno los diferentes módulos que constituyen una aplicación, para garantizar que cada uno funcione correctamente de forma independiente. (28)

La producción de código está dirigida por las pruebas unitarias. Éstas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.

La utilización de estas pruebas tiene grandes ventajas pues facilitan que el código sea cambiado por el programador para mejorar su estructura. Además, permiten hacer pruebas sobre los cambios, así como asegurarse de que los nuevos cambios no han introducidos errores y permitir reducir los efectos secundarios.

3.2.2 Pruebas de aceptación

El objetivo de estas pruebas es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema determinar su aceptación, desde el punto de vista de su funcionalidad y rendimiento. (29)

Las pruebas de aceptación son escritas conjuntamente con el cliente y el equipo de desarrollo, aunque la aprobación final corresponde al usuario. Dicho plan está diseñado para asegurar que se satisfacen todos los requisitos funcionales especificados por el usuario, teniendo en cuenta también los requisitos no funcionales relacionados con el rendimiento, seguridad de acceso al sistema, a los datos, procesos y los distintos recursos del sistema.

3.3 Plantillas de casos de prueba de aceptación

En la plantilla casos de prueba de aceptación se escriben las pruebas realizadas según las HU implementadas para realizar la comprobación, validar las funcionalidades del sistema y de esta forma saber si esta apto para ser liberado.

Los casos de pruebas de aceptación se caracterizan por:

- Ser escritos conjuntamente con el cliente.
- Ejecutados por el equipo de desarrollo.
- Difundir los resultados entre el equipo de desarrollo.

3.3.1 Escenario "Reportes de Matrícula"

Tabla 3. 1 Descripción del caso de prueba Reportes de Matrícula

Caso de Prueba
Identificación de la HU a probar: CP1-HU1-Reportes de Matrícula
Descripción de la funcionalidad: Pruebas para realizar a los diferentes reportes de estudiantes matriculados en la Universidad.
Condiciones de Ejecución: Debe seleccionar uno de los filtros o hacer la búsqueda por el buscador rápido. El usuario debe haber sido autenticado en el sistema. Solo tiene acceso a realizar esta acción el administrador del sistema.
Flujo central: Opción del menú "Estudiantes matriculados" / cuadrícula de Buscar y seleccionar estudiantes / botón "Buscar".

Clases Válidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Introducción correcta de los datos en el buscador de búsqueda rápida tales como "nombre", "apellidos" o	1. Tras la introducción correcta de uno de los datos que corresponde en el buscador de búsqueda rápida o la selección de uno de los nomencladores del filtro "Filtrar Búsqueda", se buscará internamente	Satisfactorio	

"carnet de identidad" o se puede además seleccionar un nomenclador en el filtro de "Filtrar Búsqueda". Anexo 2	en la base de datos y se mostrarán los reportes generados.		
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción incorrecta de un dato en el buscador rápido "Buscar y seleccionar estudiante". Anexo 6	1. Se mostrará un mensaje en el paginado "Sin registros que mostrar."	Satisfactorio	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción de campos obligatorios en el buscador rápido "Buscar y seleccionar estudiante" o en el filtrar "Filtrar Búsqueda". Anexo 7	1. Se muestra el mensaje "Debe escoger algún tipo de búsqueda". 2. No se hace ninguna búsqueda en la base de datos.	Satisfactorio	
Evaluación de la Prueba:	Satisfactoria		

3.3.2 Escenario "Reportes de Baja"

Tabla 3. 2 Descripción del caso de prueba para los Reportes de Baja

Caso de Prueba
Identificación de la HU a probar: CP2-HU2-Reportes de Baja
Descripción de la funcionalidad: Pruebas para realizar a los diferentes reportes de estudiantes baja de la Universidad.
Condiciones de Ejecución: Debe seleccionar uno de los filtros o hacer la búsqueda por el buscador rápido. El usuario debe haber sido autenticado en el sistema. Solo tiene acceso a realizar esta acción el administrador del sistema.
Flujo central: Opción del menú "Estudiantes baja" / cuadrícula "Buscar y seleccionar estudiantes baja" / botón "Buscar".

Clases Válidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción correcta de los datos en el buscador de búsqueda rápida tales como "nombre", "apellidos" o "carnet de identidad" o se puede además seleccionar un nomenclador en el filtro de "Filtrar Búsqueda".	1. Tras la introducción correcta de uno de los datos que corresponde en el buscador de búsqueda rápida o la selección de uno de los nomencladores del filtro "Filtrar Búsqueda", se buscará internamente en la base de datos y se mostrarán los reportes generados de estudiantes bajas.	Satisfactorio	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción incorrecta de un dato en el buscador rápido "Buscar y seleccionar estudiante".	1. Se mostrará un mensaje en el paginado "Sin registros que mostrar".	Satisfactorio	

Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Introducción de campos obligatorios en el buscador rápido "Buscar y seleccionar estudiante" o en el filtrar "Filtrar Búsqueda".	1. Se muestra el mensaje "Debe escoger algún tipo de búsqueda". 2. No se hace ninguna búsqueda en la base de datos.	Satisfactorio	
Evaluación de la Prueba:	Satisfactoria		

3.3.3 Escenario "Reportes de Egresados"

Tabla 3. 3 Descripción del caso de prueba para los Reportes de Egresados.

Caso de Prueba
Identificación de la HU a probar: CP3-HU3-Reportes de Egresados
Descripción de la funcionalidad: Pruebas para realizar a los diferentes reportes de estudiantes egresados de la Universidad.
Condiciones de Ejecución: Debe seleccionar uno de los filtros o hacer la búsqueda por el buscador rápido. El usuario debe haber sido autenticado en el sistema. Solo tiene acceso a realizar esta acción el administrador del sistema.
Flujo central: Opción del menú "Estudiantes egresados" / cuadrícula "Buscar y seleccionar estudiantes egresados"/ botón "Buscar".

Clases Válidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Introducción correcta de los datos en el buscador de búsqueda rápida tales como "nombre", "apellidos" o	1. Tras la introducción correcta de uno de los datos que corresponde en el buscador de búsqueda	Satisfactorio	

“carnet de identidad” o se puede además seleccionar un nomenclador en el filtro de “Filtrar Búsqueda”.	rápida o la selección de uno de los nomencladores del filtro ”Filtrar Búsqueda”, se buscará internamente en la base de datos y se mostrarán los reportes generados de estudiantes egresados.		
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción incorrecta de un dato en el buscador rápido “Buscar y seleccionar estudiante”.	1. Se mostrará el mensaje en el paginado “Sin registros que mostrar.”	Satisfactorio	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción de campos obligatorios en el buscador rápido “Buscar y seleccionar estudiante” o en el filtrar “Filtrar Búsqueda”.	1. Se muestra un mensaje en el paginado “Debe escoger algún tipo de búsqueda”. 2. No se hace ninguna búsqueda en la base de datos.		
Evaluación de la Prueba:	Satisfactoria		

3.3.4 Escenario “Reportes de Licencia”

Tabla 3. 4. Descripción del caso de prueba para los Reportes de Licencia

Caso de Prueba
Identificación de la HU a probar: CP4-HU4-Reportes de Licencia
Descripción de la funcionalidad: Pruebas para realizar a los diferentes reportes de estudiantes licencia de la Universidad.

Condiciones de Ejecución: Debe seleccionar uno de los filtros o hacer la búsqueda por el buscador rápido. El usuario debe haber sido autenticado en el sistema. Solo tiene acceso a realizar esta acción el administrador del sistema.
Flujo central: Opción del menú "Estudiantes licencia" / cuadrícula "Buscar y seleccionar estudiantes licencia"/ botón "Buscar".

Clases Válidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción correcta de los datos en el buscador de búsqueda rápida tales como "nombre", "apellidos" o "carnet de identidad" o se puede además seleccionar un nomenclador en el filtro de "Filtrar Búsqueda".	1. Tras la introducción correcta de uno de los datos que corresponde en el buscador de búsqueda rápida o la selección de uno de los nomencladores del filtro "Filtrar Búsqueda", se buscará internamente en la base de datos y se mostrarán los reportes generados de estudiantes de licencias.	Satisfactorio	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción incorrecta de un dato en el buscador rápido "Buscar y seleccionar estudiante".	1. Se mostrará un mensaje en el paginado "Sin registros que mostrar".	Satisfactorio	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción de campos obligatorios en el buscador rápido "Buscar y seleccionar	1. Se muestra el mensaje "Debe escoger algún tipo de búsqueda".		

estudiante" o en el filtrar "Filtrar Búsqueda".	2. No se hace ninguna búsqueda en la base de datos.		
Evaluación de la Prueba:	Satisfactoria		

3.3.5 *Escenario "Reportes de Traslado"*

Tabla 3. 5 Descripción del caso de prueba para los Reportes de Traslado.

Caso de Prueba
Identificación de la HU a probar: CP5-HU5-Reportes de Traslado
Descripción de la funcionalidad: Pruebas para realizar a los diferentes reportes de estudiantes de traslado de la Universidad.
Condiciones de Ejecución: Debe seleccionar uno de los filtros o hacer la búsqueda por el buscador rápido. El usuario debe haber sido autenticado en el sistema. Solo tiene acceso a realizar esta acción el administrador del sistema.
Flujo central: Opción del menú "Estudiantes traslados" / cuadrícula "Buscar y seleccionar estudiantes traslado"/ botón "Buscar".

Clases Válidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción correcta de los datos en el buscador de búsqueda rápida tales como "nombre", "apellidos" o "carnet de identidad" o se puede además seleccionar un nomenclador en el filtro de "Filtrar	1. Tras la introducción correcta de uno de los datos que corresponde en el buscador de búsqueda rápida o la selección de uno de los nomencladores del filtro "Filtrar Búsqueda", se buscará internamente en la base de datos y se mostrarán los reportes generados de estudiantes de traslados.	Satisfactorio	

Búsqueda".			
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción incorrecta de un dato en el buscador rápido "Buscar y seleccionar estudiante".	1. Se mostrará en el paginado el mensaje "Sin registros que mostrar".	Satisfactorio	
Clases Inválidas	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Introducción de campos obligatorios en el buscador rápido "Buscar y seleccionar estudiante" o en el filtrar "Filtrar Búsqueda".	1. Se muestra el mensaje "Debe escoger algún tipo de búsqueda". 2. No se hace ninguna búsqueda en la base de datos.		
Evaluación de la Prueba:	Satisfactoria		

3.3 Conclusiones

Con este capítulo se ha dado fin al flujo de prueba que propone SXP. Los casos de prueba son el mecanismo usado para asegurar que el sistema cumple con los requerimientos funcionales, asegurando así la calidad del software. Se incluyeron las descripciones de los principales casos de prueba de aceptación a los que se sometió el sistema, indicando para cada uno su respuesta. De manera general las pruebas fueron satisfactorias, ya que arrojaron pocas no conformidades que fueron corregidas por el equipo de desarrollo de manera rápida y eficiente, contribuyendo a obtener un producto de mayor calidad.

Conclusiones generales

Con la utilización de las herramientas, lenguajes y tecnologías definidas por el Departamento de Gestión Universitaria de la UCI, como PHP, PostgreSQL, servidor web Apache y el framework CodeIgniter, se garantizó que el módulo implementado fuese visible en cualquier plataforma y que se cumpliera con las políticas de migración a software libre de la Universidad y del país.

Se implementó el módulo de Reportes y Estadísticas para el control y gestión de toda la información del Pregrado de la Universidad de las Ciencias Informáticas, dándole solución a los problemas relacionados con la disponibilidad y búsqueda de información y simplificando de esta forma el trabajo en la Secretaría General y Docente.

El módulo implementado brinda la posibilidad de obtener diferentes reportes asociados a los criterios seleccionados por el usuario.

A través de las pruebas realizadas a la propuesta de solución se obtuvieron resultados satisfactorios que validaron la calidad de la misma y el cumplimiento de los objetivos planteados en el trabajo de investigación.

Recomendaciones

A pesar de haber cumplido los objetivos específicos de este trabajo, a medida que se ha ido desarrollando la propuesta de solución han surgido ideas que podrían implementarse más adelante. Por lo que se recomienda al equipo del proyecto, la completa terminación del módulo de Reportes y Estadísticas para lograr un sistema más provechoso y efectivo. Algunas de las funcionalidades que se recomiendan son:

- Realizar Reportes de Promoción: Están relacionados con la asistencia, registro, evaluación y grupo docente de los estudiantes de la Universidad.
- Realizar Reportes de Balance: Se vinculan al plan de estudio, modalidad de estudio y asignaturas de los estudiantes de la Universidad.
- Realizar gráficas estadísticas de barra, línea y pastel, no sólo para describir y resumir la información, sino también para analizarla y mostrar el avance de la gestión académica en la institución.

Bibliografía

1. Técnicas de programación. [En línea] 22 de mayo de 2008. http://autorneto.com/tecnologia/software/tecnicas-de-programacion/.

2. Programación Orientada a objetos. [En línea] 22 de abril de 2010. http://kaorientadaaobjetos.blogspot.com/.

3. **Casanova, Ing.Diana Cecilia Muñoz.** PROGRAMACION CON ASIGNACIONES. [En línea] 2007. http://www.scribd.com/doc/4750382/PROGRAMACION-CON-ASIGNACIONES.

4. **Rossel, Lic. Gerardo.** Programación logica. [En línea] 2004. http://www.amzi.com/articles/code07_whitepaper.pdf.

5. **Gobierno de Navarra.** *GUIACTIVA TECNICA. SOFTWARE LIBRE.* s.l. : CEIN, S.A., 2005.

6. Login. [En línea] 2010. [Citado el: 20 de febrero de 2010.] http://www.logindesarrollos.com/es/Servicios/Desarrollo-de-aplicaciones-online.

7. Lenguajes de programación. [En línea] 2009. http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml.

8. Lenguajes del lado servidor y del lado cliente. [En línea] 17 de enero de 2007. http://eats.wordpress.com/2007/01/17/lenguajes-del-lado-servidor-y-del-lado-cliente/.

9. **Arteaga, Esther Mª Pulido Alonso y Octavio Domínguez.** *Tutorialde HTML.* Gran Canaria : s.n., 2009.

10. **Pérez, Javier Eguíluz.** *Introducción a JavaScript.* 2009.

11. **Alvarez, Miguel Angel.** Desarrolloweb.com. [En línea] 2009. http://www.desarrolloweb.com/articulos/codeigniter.html.

12. **Pérez, Javier Eguíluz.** *Introducción a AJAX.* 2008.

13. Lenguajes de lado servidor. [En línea] 2010. http://www.desarrolloweb.com/articulos/243.php.

14. Maestros del web. [En línea] [Citado el: 12 de febrero de 2010.] http://www.maestrosdelweb.com/editorial/como-migrar-tus-aplicaciones-de-php4-a-php5/.

15. **Pecos, Daniel.** PostGreSQL vs. MySQL. *PostGreSQL vs. MySQL.* [En línea]

16. **Foundation, Apache Software.** The Apache Software Foundation . [En línea] 2009. [Citado el: 24 de febrero de 2010.] http://www.apache.org..

17. **Rick Ellis, y , Paul Burdick . : Conocimiento Virtual Academia ltda.** MANUAL DE Code Igniter. [En línea] Medellín, Colombia. http://eats.wordpress.com/2007/01/17/lenguajes-del-lado-servidor-y-del-lado-cliente/..

18. **Alvarez, Miguel Angel.** DesarrolloWeb.com. [En línea] 2010. http://www.desarrolloweb.com/articulos/introduccion-jquery.html.

19. **Corredor, Germán.** Sun libera su IDE NetBeans para Java y PHP. [En línea] julio de 2009. http://osum.sun.com/profiles/blogs/sun-libera-su-ide-netbeans-67-1..

20. **Zaballa Coca, Roilán.** *Personalización de distribuciones basadas en la familia SUSE Linux.". 2009.* 2009.

21. **Sanchez, María A. Mendoza.** Metodologías De Desarrollo De Software. [En línea] http://www.willydev.net/InsiteCreation/v1.0/descargas/cualmetodologia.pdf.

22. XP Programación Extrema. Club Developers. [En línea] [Citado el: 7 de febrero de 2010.] http://www.clubdevelopers.com/index.php?p=38..

23. **Chuidiang.** SCRUM. [En línea] 2009. http://www.chuidiang.com/ood/metodologia/scrum.php..

24. **Marsi, Gladys.** Trabajo de Diploma MA-GMPR-UR2 Metodología ágil para proyectos de software libre. La Habana, Cuba : s.n., 2008.

25. **Moisset, Daniel F.** Cómo y porqué programar con buen estilo. [En línea] http://educa.di.unc.edu.ar/mod/resource/view.php?id=2904..

26. **Productiva, Dirección de Calidad de la Infraestructura.** *Estándar de código.* Universidad de las Ciencias Informáticas : s.n., 2009.

27. *El único instrumento adecuado para determinar el status de la calidad.* s.l. : Asociación de Técnicos, 2008.

28. **B., Ing. Alexander Oré.** Pruebas Unitarias CAP 1-Sotware Testing and QA-Pruebas Unitarias. [En línea] 2009. http://www.calidadysoftware.com/testing/pruebas_unitarias1.php.

29. La prueba de aceptación es la prueba más importante para los productos software. [En línea] 2010. http://pruebasdesoftware.com/pruebadeaceptacion.htm.

Glosario de términos

Ajax: Acrónimo de Asynchronous JavaScript And XML (JavaScript y XML asíncronos), es una técnica de desarrollo web para crear aplicaciones interactivas. Éstas se ejecutan en el cliente, es decir, en el navegador del usuario, y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma.

CamelCase: Es un estilo de escritura que se aplica a frases o palabras compuestas. El nombre se debe a que las mayúsculas a lo largo de una palabra en CamelCase se asemejan a las jorobas de un camello. El nombre CamelCase se podría traducir como Mayúsculas/Minúsculas Camello. El término case se traduce como "caja tipográfica", que a su vez implica si una letra es mayúscula o minúscula y tiene su origen en la disposición de los tipos móviles en casilleros o cajas.

CGI: son las siglas en inglés de Common Gateway Interface, en español, Interfaz común de puerta de enlace, y es la interfaz entre un servidor con Protocolo de Transferencia de Hipertexto (HTTP), programa que sirve las páginas de un sitio Web y los demás recursos de la computadora host del servidor.

Cookies: Son fragmentos de información que se almacenan en el disco duro del visitante de una página web a través de su navegador, a petición del servidor de la página. Esta información puede ser luego recuperada por el servidor en posteriores visitas. En ocasiones también se le llama "huella".

Feed RSS: Es un archivo generado por algunos sitios web (y por muchos weblogs) que contiene una versión específica de la información publicada en esa web. Cada elemento de información contenido dentro de un archivo RSS se llama "ítem".

GNU: Es un acrónimo recursivo que significa GNU No es Unix (GNU is Not Unix). Puesto que en inglés "gnu" (en español "ñu") se pronuncia igual que "new", Richard Stallman recomienda pronunciarlo "guh-noo". En español, se recomienda pronunciarlo ñu como el antílope africano o fonéticamente; por ello, el término mayoritariamente se deletrea (G-N-U).

Grid: Es la tecnología que permite que los ordenadores compartan a través de Internet u otras redes de telecomunicaciones no sólo información, sino también poder de cálculo (grid computing) y capacidad de almacenamiento (grid data).

HTML: Acrónimo inglés de HyperText Markup Language (lenguaje de marcas hipertextuales), lenguaje de marcación diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas web. Gracias a Internet y a los navegadores del tipo Internet Explorer, Opera, Firefox o Netscape, el HTML se ha convertido en uno de los formatos más populares que existen para la construcción de documentos y también de los más fáciles de aprender.

HTTP: Es el protocolo de transferencia de hipertexto (HyperText Transfer Protocol) usado en cada transacción de la Web (WWW). El protocolo HTTP está basado en el modelo cliente-servidor y su versión actual la 1.1.

Identación: Anglicismo (de la palabra inglesa indentation) de uso común en informática, que significa mover un bloque de texto hacia la derecha insertando espacios o tabuladores para separarlo del texto adyacente, lo que en el ámbito de la imprenta se ha denominado siempre como sangrado o sangría.

IIS: Es una serie de servicios para los ordenadores que funcionan con Windows. Los servicios que ofrece son FTP, SMTP, NNTP y HTTP/HTTPS., 17

Interfaz: En software, una interfaz es la parte del programa informático que permite el flujo de información entre varias aplicaciones o entre el propio programa y el usuario. En software también se habla de interfaz gráfica de usuario, que es un método para facilitar la interacción del usuario con el ordenador o la computadora a través de la utilización de un conjunto de imágenes y objetos pictóricos (iconos, ventanas, formularios, páginas web.)

JSON: Acrónimo de JavaScript Object Notation, es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.

jQuery: es una biblioteca o framework de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web.

Metodología: (del griego metà "más allá", odòs "camino" y logos "estudio"), hace referencia al conjunto de procedimientos basados en principios lógicos, utilizados para alcanzar una gama de objetivos que rigen en una investigación científica.

Plug-in: Módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

Procedimiento: Es la secuencia de acciones concatenadas entre sí, que ordenadas en forma lógica permite cumplir un fin u objetivo predeterminado.

Reporte: Se refiere a la información lógica, relevante, y organizada, obtenida a partir de la recuperación de datos incluidos en el sistema.

Script : Es un conjunto de instrucciones. Permiten la automatización de tareas, creando pequeñas utilidades. Es muy utilizado para la administración de sistemas UNIX. Son ejecutados por un intérprete de línea de órdenes y usualmente son archivos de texto.

SVN: Subversion es un software de sistema de control de versiones, se le conoce como svn por ser ese el nombre de la herramienta utilizada en la línea de órdenes.

Tabs: Son las etiquetas, tabulaciones o el asistente técnico.

Web: La World Wide Web, cuya traducción podría ser Red Global Mundial o "Red de Amplitud Mundial", es un sistema de documentos de hipertexto o hipermedios enlazados y accesibles a través de Internet.

World Wide Web Consortium: Es un consorcio internacional que produce estándares para la World Wide Web. Esta competencia en exclusiva del W3C para crear estándares abiertos es crucial, pues de ella depende que ningún fabricante alcance nunca el monopolio de explotación de la Web.

XML: son las siglas en inglés de Extensible Markup Language, lenguaje de marcado ampliable o extensible, desarrollado por el World Wide Web Consortium (W3C). No es más que un conjunto de reglas para definir etiquetas semánticas que nos organizan un documento en diferentes partes.