PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

ESCUELA DE SISTEMAS

DISERTACIÓN PREVIA A LA OBTENCIÓN DE TÍTULO DE

INGENIERO DE SISTEMAS Y COMPUTACIÓN

“DESARROLLO DE UN DSS DE LA CARRERA DE INGENIERÍA EN SISTEMAS”

AUTORES:

Andrea Benalcázar

Paúl Manzano

DIRECTOR:

Damián Nicolalde R.

Quito, Abril, 2016

Dedicatoria

Dedico esta tesis a mi familia, especialmente a mis padres Gladis y Eduardo ya que gracias a ellos he tenido toda la fuerza necesaria y el apoyo suficiente para realizar mis objetivos, dedico a mi sobrinita Simone ya que ella ha sido mi inspiración para ser un ejemplo y por ello tener la necesidad de sobresalir, dedico también a mis sobrinos Ian y Gianny que han sabido brindarme su amor desde muy lejos y en consecuencia me han sabido dar fortaleza para acabar con mis objetivos y por ultimo pero no menos importante dedico este trabajo a mi hermano para que se sepa que todo es posible y él también logre cumplir sus objetivos en esta vida.

Andrea Benalcázar

Dedico esta tesis a mi madre Rosita que me ha sabido inculcar responsabilidad y liderazgo, a mi hermana que desde lejos me ha sabido brindar sus consejos y ayuda, a mi familia que siempre se ha enorgullecido de mis logros y a mis maestros que gracias al apoyo de ellos me sido posible culminar este gran paso en mi vida.

Paúl Manzano

Agradecimientos

Agradezco a mi madre Gladis ya que ella ha sabido siempre guiarme en el camino y siempre apoyándome en cada paso que he dado.

Agradezco a mi padre Eduardo ya que aunque a su manera me apoyado y me ha dado consejos para que siga luchando por lo que he querido.

Agradezco a mi esposo Paul ya que siempre me ha cuidado y ayudado en todo este camino que hemos recorrido juntos.

Agradezco a mi director de tesis Ing. Damián por tenernos paciencia y brindarnos su dirección para que esto sea posible.

Y finalmente agradezco a Dios que ha permitido que esto sea posible, brindándonos inteligencia y salud.

Andrea Benalcázar

Agradezco al apoyo de toda mi familia mi madre, mi hermana Jimena y mis sobrinos Ian y Gianny.

Agradezco a mis amigos y mi esposa Andrea que han sido incondicionales y gracias a ellos he podido culminar con gran alegría mi etapa universitaria.

Agradezco a mis maestros que han logrado enseñarme conocimientos importantes que me servirán como guía en mi vida profesional.

Agradezco a dios que siempre ha estado ayudándome y guiándome en cada una de las etapas de mi vida.

Paúl Manzano

CONTENIDO

Índice de Figuras

Índice de Tablas

Resumen

La Carrera de Ingeniería en Sistemas de la Pontificia Universidad Católica del Ecuador se ha visto en la necesidad de desarrollar un sistema capaz de servir de apoyo en la toma de decisiones, esto debido a los cambios recientes en las leyes de Educación Superior que han sido implantadas por el gobierno de turno, y la competencia generada por otros Institutos de Educación Superior en la carrera de Ingeniería en Sistemas-

El sistema desarrollado, será una aplicación web y estará basado en una estructura en capas, que será capaz de generar reportes estáticos y dinámicos, para servir de soporte en la toma de decisiones a nivel gerencial y así saber cómo se encuentra la carrera con respecto a otras Universidades y poder prepararse para nuevos cambios a futuro.

1 Capítulo 1: Marco teórico

Este capítulo contiene los conceptos teóricos de toda la disertación, principalmente trata sobre los conceptos teóricos del sistema que se van a desarrollar, los conceptos teóricos de las herramientas que se van a utilizar para el desarrollo de la aplicación y los conceptos teóricos de la metodología de programación que se usará.

1.1 DSS[1]

1.1.1 Sistema de información

Según Peralta, "un sistema de información es un conjunto de elementos que interactúan entre sí con el fin de apoyar a las actividades de una empresa o negocio" (Peralta, 2008).

Un sistema de información tiene cuatro actividades básicas o principales, que son:

Entrada de información: se refiere al proceso en el cual el sistema de información obtiene los datos que necesita para procesar la información, dichas entradas puede ser manuales o automáticas. Las entradas manuales son las que se toman del usuario directamente y las automáticas son las que provienen de otros sistemas o módulos (Peralta, 2008).

Almacenamiento de información: esta actividad es una de las más importantes de una computadora, porque a través de ella la computadora puede recodar la información guardada de una sesión o proceso anterior, generalmente esta información suele ser guardada en una estructura de información denominada archivos (Peralta, 2008).

Procesamiento de la información: Corresponde a la capacidad del sistema de información de realizar cálculos mediante un orden de operaciones preestablecidas. Dichos cálculos pueden realizarse mediante datos introducidos al sistema o con datos almacenados. Esta característica de los sistemas faculta la transformación de datos que puede ser usada como ayuda para la toma de decisiones (Peralta, 2008).

Salida de información: es la habilidad de un sistema de información para mostrar la información ya procesada o bien mostrar la de entrada (Peralta, 2008).

1.1.2 Tipos de sistemas de información

Según Peralta, los sistemas de información se clasifican en forma general, en tres diferentes formas fundamentales que son: Sistemas transaccionales; Sistemas de Soporte a la Toma de Decisiones y Sistemas estratégicos (Peralta, 2008).

[1] Decision support system o sistema de soporte a la toma de decisiones

Sistemas Transaccionales.

Los sistemas transaccionales están diseñados para recopilar, guardar y modificar cualquier tipo de información que ha sido generada por las diferentes transacciones de una organización. Una transacción es el proceso o evento que ha creado o modificado la información que esta guardada dentro del sistema de información de la organización (Alegsa, 2010).

Sistemas de Apoyo de las Decisiones.

Los sistemas de apoyo de las decisiones se pueden definir como un conjunto de herramientas y sistemas, los cuales permiten obtener información necesaria para realizar un proceso de discernimiento en la toma de decisiones, por tal razón dichos sistemas dan un soporte o apoyo al momento de tomar decisiones en aspectos gerenciales de distintos tipos (SISTEMA DE APOYO A LA TOMA DE DECISIONES, s.f.).

Sistemas Estratégicos.

Un sistema de información estratégico, es un sistema de información que brinda un apoyo a las estrategias, acciones y planes de la organización, esto ayuda a lograr ventajas realmente competitivas en el mercado a nivel global (LOS SISTEMAS DE INFORMACIÓN ESTRATEGICOS, 2012).

1.1.3 Sistemas de soporte a la toma de decisiones

1.1.3.1 Definición

Los DSS son sistemas de información, que realizan el procesamiento de datos para poder mostrar información relevante para el usuario y que dicha información le ayude a la correcta toma de decisiones. Estos sistemas tienen como principal objetivo proporcionar la mayor cantidad de información necesaria en el menor tiempo permitido, para que el usuario del sistema pueda realizar una toma de decisiones acertadas y en menor tiempo (Sistemas de Soporte a la toma de Decisiones, 2012).

1.1.3.2 Objetivo

Los objetivos fundamentales de un sistema de apoyo a la toma de decisiones según (Medina, 2014) son tres:

- Ayudar a los usuarios a tomar decisiones para resolver problemas semi estructurados.
- Apoyar el juicio del usuario en lugar de tratar de reemplazarlo
- Mejorar la eficacia del usuario en la toma de decisiones, más que su eficiencia.

Estos tres objetivos están relacionados con los siguientes tres principios básicos:

Estructura del problema:

Para poder saber la estructura del problema lo fundamental es saber cómo se encuentra estructurado el problema, si sabemos eso podremos saber cuál es el punto de inicio del problema. La mayoría de problemas a los que nos enfrentamos son problemas semi estructurados, ya que resulta difícil localizar problemas totalmente estructurados o sin estructura alguna; esto nos conlleva a dirigir al DSS al área don existe el mayor grado de dificultad (Medina, 2014).

Los problemas estructurados son aquellos donde sus tres primeras fases las cuales son: La obtención de información estratégica o inteligencia, diseño y selección están bien estructuradas, por tal razón se puede especificar las reglas de decisión para poder identificar, entender de qué se trata el problema y así evaluar todas las soluciones posibles y poder tomar una decisión. En cambio un problema no estructurado no dispone ninguna estructura en las tres fases mencionadas anteriormente. Y finalmente un problema semi-estructurado puede tener una o dos fases estructuradas (Medina, 2014).

Apoyo a decisiones:

Para realizar el apoyo a la toma de decisiones el sistema no pretende reemplazar al usuario, sino que quiere ser un apoyo para este. Ya que el sistema ayuda a estructurar el problema, para el usuario se ocupa de la parte no estructurada y aplica su razonamiento, análisis y juicio para poder llevar a cabo la toma de decisiones. Por eso en conclusión se puede decir que el usuario y el sistema trabajan conjuntamente como un equipo centrado en la resolución de problemas (Medina, 2014).

Eficacia de las decisiones:

El principal objetivo de un DSS no es hacer que el proceso de toma de decisiones sea más óptimo, aunque el tiempo del usuario es único y no debe tomarse a la ligera, el principal objetivo del DSS es procurar ayudar a tomar las mejores decisiones posibles. Ya que al fin y al cabo el usuario es el utiliza su análisis y juicio para resolver un problema, y lo que se necesita es la mejor elección (Medina, 2014).

1.1.3.3 Beneficios

Los beneficios que nos brinda un DSS son:

Mejorar la eficiencia personal, Proporcionar soluciones a problemas, Facilitar la comunicación entre varias personas, Fomentar el aprendizaje o formación, Mejorar el control de la organización, Obtener nueva información que ayudan a la toma de decisiones, Crear ventajas competitivas sobra la competencia, Incita a explorar y descubrir cómo se toman las decisiones, Proporciona nuevos puntos de vista de cómo pensar sobre el espacio del problema (Unidad de Desarollo Tecnológico en Inteligencia Artificial, s.f.).

1.1.3.4 Estructura, elementos y componentes

Según Hättenschwiler, se pueden identificar cinco componentes en un DSS:

Usuarios.- Son los usuarios del sistema, estos pueden tener diferentes roles o permisos para poder realizar el proceso de toma de decisiones, esto depende de la organización (Isuba, 2014).

Contexto de decisión.- El contexto de decisión es un marco que debe poder ser definido y especifico (Isuba, 2014).

Sistema de destino.- En el sistema de destino es el componente del sistema donde se pueden describir las preferencias a llevar a cabo para las decisiones (Isuba, 2014).

Bases de conocimiento.- Las bases de conocimientos están constituidas de bases de conocimientos, fuentes de datos externas, bases de datos de trabajo, meta-bases de datos, almacenes de datos, archivos, modelos matemáticos, motores de búsquedas entre otros tipos donde pueda ser almacenada la información (Isuba, 2014).

1.2 Estilo arquitectural en N capas

1.2.1 Estilo arquitectural

El estilo arquitectural puede definirse como un conjunto de varios principios que son definidos en alto nivel en la aplicación. El estilo arquitectural puede definirse como un conjunto de componentes y restricciones conectados entre sí (De La Torre LLorente, Zorilla Castro, Ramos Barros , & Calvarro Nelson, 2010).

1.2.2 Definición del estilo arquitectural en N capas

El estilo arquitectural en N capas se basa en una distribución jerárquica de los roles y las responsabilidades para proporcionar un división efectiva de los problemas a resolver. Los roles indican el tipo y la forma de la interacción con otras capas y las responsabilidades de la funcionalidad que implementan (De La Torre LLorente, Zorilla Castro, Ramos Barros , & Calvarro Nelson, 2010).

1.2.3 Características

- Descomposición de los servicios de forma que la mayoría de las interacciones ocurre solo entre capas vecinas
- Las capas de una aplicación pueden residir en la misma maquina o pueden estar distribuidos en varios equipos
- Los componentes de cada capa se comunican con los componentes de otras capas a través de interfaces bien conocidos
- Cada nivel agrega las responsabilidades y abstracciones del nivel inferior
- Muestra una vista completa del modelo y a la vez proporciona suficientes detalles para entender las relaciones entre capas
- No realiza ninguna suposición sobre los tipos de datos, métodos, propiedades y sus implementaciones
- Separa de forma clara la funcionalidad de cada capa (De La Torre LLorente, Zorilla Castro, Ramos Barros , & Calvarro Nelson, 2010).

1.2.4 Principios clave

- Cada capa contiene la funcionalidad relacionadas con la tarea de esa capa
- Las capas inferiores no tienen dependencias de las capas superiores
- La comunicación entre capas está basada en una abstracción que proporciona un bajo acoplamiento entre capas (De La Torre LLorente, Zorilla Castro, Ramos Barros , & Calvarro Nelson, 2010).

1.2.5 Beneficios

- **Abstracción**, ya que los cambios se realizan a alto nivel y se puede incrementar o reducir el nivel de abstracción que se usa en cada capa del modelo
- **Aislamiento,** ya que se pueden realizar actualizaciones en el interior de las capas sin que esto afecte al resto del sistema
- **Rendimiento** ya que distribuyendo las capas en distintos niveles físicos se puede mejorar la escalabilidad, la tolerancia a fallos y el rendimiento
- **Testeabilidad**, ya que cada capa tiene una interfaz bien definida sobre la que se realizan las pruebas y la habilidad de cambiar entre diferentes implementaciones de una capa
- Independencia ya que elimina la necesidad de considerar el hardware y el despliegue así como las dependencias con interfaces externas (De La Torre LLorente, Zorilla Castro, Ramos Barros , & Calvarro Nelson, 2010).

1.2.6 Tipos de capas

Capa de presentación

La capa presentación es la capa que tiene como propósito interactuar con el usuario, por tal razón, son las ventanas, mensajes, diálogos o páginas que sirve al usuario para poder comunicarse con el sistema. Mediante la capa presentación el usuario puede ordenar que se realicen las tareas, ingresando parámetros de entrada para que el usuario pueda obtener una respuesta del sistema. Esta capa relaciona con la capa Lógica de negocio, la capa de servicio y la capa de entidades, estas capas se comunican entre sí para poder enviar y recibir información necesaria para el funcionamiento del sistema (García Bautista, 2014).

Capa de servicio

La capa de servicios es una capa de abstracción que se encuentra entre la capa de presentación y la capa lógica de negocio, uniendo las funcionalidades de la capa lógica de negocio para que sean presentadas en la capa de presentación. Cualquier comunicación que se origine en la capa de presentación, encuentra una respuesta en la capa de servicios (EL framework de coco, 2009).

Aparte de todo la capa de servicio, puede actuar de una manera independiente de la capa de presentación, ya que uno de sus propósitos es poder ser una capa consumible para diferentes aplicaciones.

Capa Lógica de Negocio

La capa lógica de negocio como su nombre lo dice es la capa encargada de la implementación de lógica del negocio, esto quiere decir, que en esta capa se implementa todo lo que el sistema debe saber o considerar antes de realizar el proceso o la acción en la capa de datos (García Bautista, 2014).

> Por ejemplo: Antes de solicitar a la capa de Datos la inserción de un grupo de registros en una tabla, valida que vayan todos los campos mandatorios dentro de esa solicitud si esta condición no se cumple entonces rechaza la inserción e informa del usuario del status de su solicitud; otro ejemplo podría ser, solicitar a la base de datos que valide la presencia de un registro antes de insertar el siguiente, validar los tipos de datos, etc. esos ejemplos por mencionar los más básicos y generales (García Bautista, 2014).

En conclusión la capa lógica de negocio es la encargada de recoger todas las solicitudes de la capa de presentación, comprobar que las condiciones del negocio se cumplan y si se cumplen dichas condiciones realizar la solicitud a la capa de datos (García Bautista, 2014).

Capa de Datos

La capa de datos, es la capa que se encarga en la comunicación de la base de datos con la capa de la lógica de negocio. En esta capa es donde se encuentran todas las acciones CRUD[2] de la base de datos, esta capa es la única que conoce cuál es el motor de la base de datos se está usando dentro de la aplicación, pero esta capa no puede saber si la aplicación donde se encuentra es una aplicación web o desktop (García Bautista, 2014).

Capa de Entidades

La capa entidades aparentemente puede ser vista como la cuarta capa, pero está en realidad no lo es, ya que la capa entidades tiene como función abarcar todos las clases u objetos que pueden representar al negocio. La capa entidades es la única capa que puede ser llamada por las tres capas anteriores, por tal razón esta única capa es la que tiene comunicación con el resto de capas, pero su función solo está limitada a ser un puente de transmisión de datos, ya que esta es una capa que complementa a la capa lógica de negocio (García Bautista, 2014).

1.2.7 Conexión de las capas en la aplicación

Según la figura 1-01 se puede ver las conexiones de las capas que tiene la aplicación:

Figura 1-1 Conexión entre capas (Benalcázar & Manzano, 2016)

[2] CRUD: Create, Read, Update y Delete

1.3 Metodología de desarrollo (Programación extrema XP)

1.3.1 Metodologías Agiles de software

Las metodologías agiles de software son paradigmas de desarrollo basados en los procesos agiles. Estos procesos agiles para el desarrollo de software, eran conocidos antes como metodologías livianas, dichas metodologías intentan evadir los tormentosos y largos caminos de las metodologías comunes, estas metodologías lo logran enfocándose en los usuarios y los resultados (Universidad Simón Bolívar, 2009).

Las metodologías agiles, son un marco de trabajo conceptual de la ingeniería de software que impulsa iteraciones en el desarrollo en toda la vida del desarrollo del ciclo del proyecto. Existen diferentes formas de metodologías agiles, estas metodologías en su mayoría minimizan los riesgos logrando un desarrollo de software en menor lapso de tiempo (Universidad Simón Bolívar, 2009).

Una de las principales fortalezas de las metodologías agiles es que ponen primero a las personas que a la documentación.

1.3.2 Metodologías Ágiles populares

A continuación se presenta un resumen de cada una de las metodologías ágiles más usadas:

SCRUM

Scrum es una metodología ágil desarrollada el año 1986 por Ken Schwaber, Jeff Sutherland y Mike Beedle. Esta metodología especifica un marco para la gestión de proyectos, dicho marco se ha utilizado con gran éxito durante ya los últimos 10 años. Scrum está sugerida especialmente para proyectos que contienen un rápido cambio de Requisitos. Las principales características que tiene esta metodología son: la implementación del software se realiza por medio de iteraciones que se llaman Sprints, que duran 30 días y el resultado de cada iteración es un incremento ejecutable que se tiene que mostrar al cliente. Otra característica importante es que deben existir reuniones a lo largo del proyecto, la reunión más importante es la reunión diaria del equipo de desarrollo que dura 15 minutos (Canós, Letelier, & Penadés).

Crystal Methodologies

Cristal Methodologies es un conjunto de metodologías de desarrollo de software, con la característica de centrar las personas que componen el equipo y reducir lo más posible el número de artefactos producidos. Esta metodología fue creada en el año 2001 por Alistair Cockburn. Lo más importante de esta metodología es el equipo de desarrollo, por tal razón se invierte en mejorar sus destrezas y habilidades teniendo políticas de trabajo en equipo bien definidas (Canós, Letelier, & Penadés).

Dynamic Systems Development Method

Dynamic Systems Development Method se crea en 1994, con el propósito de crear una metodología RAD unificada. Sus principales factores son: Tener un proceso iterativo e incremental, el usuario y el equipo de desarrollo trabajan juntos (Canós, Letelier, & Penadés).

Desarrollo de software adaptable

El responsable del desarrollo de software adaptable es Jim Highsmith Sam Bayer a comienzos de 1990. Las principales características de esta metodología son: es iterativa, orientada a los componentes software más que a las tareas y tolerante a los cambios. El ciclo de vida que propone tiene tres fases esenciales: especulación, colaboración y aprendizaje. En la fase de especulación se inicia el proyecto y se planifican las características del software; en la fase de colaboración se desarrollan las características del sistema y en la última fase de aprendizaje se revisa la calidad, y se entrega al cliente. La revisión de todos los componentes sirve para aprender de los errores y volver a iniciar el ciclo de desarrollo (Canós, Letelier, & Penadés).

Lean Development

Lean development es una metodología de desarrollo ágil realizada por Bob Charette's a partir de su experiencia en los años 80, en proyectos con la industria japonesa automovilística y utilizada en varios proyectos de telecomunicaciones en Europa. En esta metodología, los cambios están considerados como riesgos, pero si se da la posibilidad de manejarlos adecuadamente pueden cambiar a oportunidades que mejoraran la productividad del cliente. Su principal función y ventaja es obtener un proceso para desarrollar esos cambios (Canós, Letelier, & Penadés).

1.3.3 Programación Extrema

La programación extrema[3] es una metodología de la ingeniería de software formulado en 1999 por Kent Beck, quien fue el autor del primer libro sobre la materia, esta metodología es la más destacada de los procesos ágiles de desarrollo de software. Al igual que la mayoría de metodologías agiles, la programación extrema tiene bastantes diferencias con las metodologías tradicionales, ya que se enfoca en la adaptabilidad que en la previsibilidad. Esta metodología puede ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto, ya que cree que el cambio de requisitos es una aproximación más cerca de la realidad que intentar obtener todos los requisitos al inicio del proyecto e poner esfuerzos después en controlar los cambios en los requisitos (Universidad Simón Bolívar, 2009).

1.3.3.1 Principios

Según (Programacion extrema), la Programación Extrema se basa en 12 principios básicos agrupados en cuatro categorías:

Retroalimentación a escala fina.

- **El principio de pruebas:** Aquí es donde se definen las pruebas para llegar a la aceptación del programa.
  Proceso de planificación: En esta etapa es donde se recopilan los requerimientos del usuario, para poder recopilar estos requerimientos el usuario expondrá sus necesidades en un documento denominado historias de usuario, lo ideal sería tener en 20 y 80 historias de usuario esto dependería del problema al cual se enfrenten. Cada historia de usuario pueden necesitar entre 1 a 3 semanas de implementación. Estas historias son de gran importancia pero tienen que estar respaldadas con reuniones periódicas tanto con el cliente como con el equipo de desarrollo, esto servirá para detectar problemas a tiempo y generar soluciones de dichos problemas (Universidad Simón Bolívar, 2009).
  El cliente en el sitio: El cliente es alguien importante dentro de esta metodología, ya que se le dará el poder para determinar los requerimientos, definir la funcionalidad, señalar las prioridades y responder las preguntas de los programadores. Esta es una fuerte interacción cara a cara con el programador, por ende disminuye el tiempo de comunicación y la cantidad de documentación, junto con los altos costes de su creación y mantenimiento. El cliente o representante del cliente estará con el equipo de desarrollo durante toda la implementación del proyecto (Universidad Simón Bolívar, 2009).

[3] Extreme programing o XP

- **Programación en parejas:** Este principio es uno de los más radicales, ya que necesita que todos los programadores que realizan la metodología de programación extrema realicen su código en una sola máquina, esto producirá aplicaciones más consistentes y de igual o menor costo (Universidad Simón Bolívar, 2009).

Proceso continuo en lugar de por lotes.

- **Integración continua:** Mediante un cronograma pre establecido permite al equipo de desarrollo implementar rápidamente nuevas características de software, eso se logra porque los programadores reúnen su código y reconstruyen la aplicación varias veces en el día, así se reducen los problemas al realizar la integración (Universidad Simón Bolívar, 2009) .
- **Refactoring:** Es el proceso que revisa continuamente el diseño y vuelve a codificar solo lo necesario, esto permite que el sistema se encuentre enfocado en el valor del negocio y elimina código duplicado (Universidad Simón Bolívar, 2009).

  Entregas pequeñas: Se realiza un desarrollo pequeño que cumpla con los primeros requisitos del usuario y poder poner en prueba dicho desarrollo con el cliente en un ambiente real, estas entregas suelen duran un máximo de tres semanas (Universidad Simón Bolívar, 2009)

Entendimiento compartido.

Diseño simple: Está basado en que el mayor valor del negocio es un programa sencillo pero que cumpla con los requerimientos, esto se enfoca en brindar un sistema que cubra las necesidades del cliente ni más, ni nada menos. Esto ayudara a eliminar rejuvenecer diseños obsoletos de forma sencilla y eliminar redundancias. (Universidad Simón Bolívar, 2009)

- **Metáfora:** Es desarrollada al inicio del proyecto, crea la historia de cómo funcionara el sistema completo. La programación extrema estimula historias de usuario, que son breves descripciones de los requerimientos de un sistema en vez de los tradicionales diagramas y modelos UML[4]. La metáfora muestra la visión evolutiva del proyecto que define el alcance y propósito del sistema. Las tarjetas CRC[5] ayudarán al equipo a definir las actividades del diseño del sistema. Cada tarjeta representa una clase en la POO[6] y define sus responsabilidades lo que ha de hacer y las relaciones entre otras clases (Universidad Simón Bolívar, 2009) .

[4] Unified Modeling Language o Lenguaje de modelado unificado

[5] Clase, Responsabilidad y Colaboración

[6] Programación orientada a objetos

- **Propiedad colectiva del código:** En esta metodología dice que el código creado mediante ella, no tiene ninguna propiedad individual, y que todos los que intervinieron en el proyecto son propietarios del código generado, ya que si no hubieran intervenido todos en el código aparecerían más errores (Universidad Simón Bolívar, 2009).
- **Estándar de codificación:** Los estándares de codificación ayudan a definir las normas, reglas para implementar el código, esto ayudara de manera extraordinaria ya que ayudara a la comunicación entre las diferentes partes y el código final parecerá haber sido escrito por una sola persona. Dichos estándares serán definidos por el equipo de desarrollo (Universidad Simón Bolívar, 2009) .

Bienestar del programador.

- **La semana de 40 horas:** El bienestar del programador es muy importante dentro de la programación extrema ya que esta afirma que un programador que ya está cansado escribe código de menor calidad, mientras que cuando están descansados estos generaran código de mayor calidad lo que es de mayor utilidad al momento del desarrollo, por eso esta metodología defiende una semana de máximo 40 horas y si se debe trabajar horas extras no hacerlo durante dos semanas seguidas (Universidad Simón Bolívar, 2009).

1.3.3.2 Ciclo de vida de un proyecto de programación extrema

El ciclo de vida de un proyecto de desarrollo con programación extrema consiste en seis fases:

Exploración

Esta fase consiste en el planteamiento de historias de usuario por parte del cliente, también en esta fase los desarrolladores se adaptan a las tecnologías y prácticas que se utilizaran para el desarrollo del sistema, para ello se realizaran pruebas de la tecnología y de la arquitectura del sistema mediante un prototipo. Esta fase debería durar entre algunas semanas a pocos meses dependiendo del tamaño del sistema y los conocimientos de la tecnología que se aplicara (Universidad Simón Bolívar, 2009).

Fase de Planeamiento o Planificación de la Entrega

Esta es la fase donde se prioriza cada historia de usuario dada por el cliente, con el fin de que el equipo de desarrollo haga una estimación del tiempo y esfuerzo que se necesitara en cada una de ellas, con esto realizado se hacen acuerdos sobre cuándo será la primera entrega y las próximas entregas en conjunto con el cliente. Una entrega se debería realizar en un máximo de tres semanas. Dicha fase es corta, ya que dura pocos días (Universidad Simón Bolívar, 2009).

Iteraciones

Después de la fase de planeación, se realizan las interacciones ya programadas en dicha fase. Se debe procurar hacer en la primera iteración la arquitectura del sistema, esto se puede lograr escogiendo las historias de usuario que ayuden a esto, se dice que se debe procurar ya que el que decide que historias de usuario se harán no son los desarrolladores si no el cliente. Cada iteración debe tener una duración máxima de tres semanas. El trabajo de cada iteración debe ser formulada en tareas de programación, cada tarea debe ser asignada a una pareja de programadores para ser llevadas a cabo (Universidad Simón Bolívar, 2009).

Producción

En esta fase se realizan diferentes pruebas que no se hayan realizado en cada iteración, para poder observar el rendimiento completo del sistema antes de que sea trasladado al entorno final que es el del cliente. Simultáneamente se tomaran decisiones de si se incluyen o no nuevas funciones al sistema esto se debería ya que se pudieron haber realizado cambios durante la fase (Universidad Simón Bolívar, 2009).

Mantenimiento

Si el cliente lo requiere, en esta fase mientras el sistema ya se encuentra en funcionamiento, se pueden realizar nuevas iteraciones para generar tareas de soporte para el cliente. En esta fase se puede bajar la velocidad producción y esta fase podría necesitar nuevas personas en el equipo y realizar cambios en su estructura (Universidad Simón Bolívar, 2009).

Muerte del Proyecto

La muerte del proyecto ocurre cuando ya no existen más historias de usuario a desarrollar al sistema, cuando no se satisfacen las necesidades del cliente o cuando se carece de presupuesto para mantener el sistema. Al terminar el proyecto se debe realizar la documentación final del sistema y ya no realizar más cambios a su arquitectura (Universidad Simón Bolívar, 2009).

1.4 Herramientas utilizadas

Las herramientas utilizadas en el desarrollo del sistema de apoyo en la toma de decisiones son las siguientes:

1.4.1 .Net Framework

"… es el conjunto de nuevas tecnologías en las que han estado trabajando a lo largo de dos años" (Sotomayor, 2002, pág. 2).

Este conjunto de nuevas tecnologías podrían resumirse, según (Sotomayor, 2002, pág. 2) en las siguientes:

- Plataforma .NET
- SDK de la plataforma .NET
- Visual Studio.NET
- Servicios Web

1.4.1.1 La Plataforma .NET

La plataforma .net es la capa de software que se encuentra localizada entre el sistema operativo y la persona que programa, esta capa ayuda a abstraer las características internas del sistema operativo (Sotomayor, 2002). Esta plataforma consta de tres características fundamentales, las cuales son: Portabilidad, multilenguaje e interoperabilidad.

Portabilidad: La portabilidad se da gracias a la abstracción que tiene la persona que programa con el SO[7], por tal razón lo único que se necesita para correr la aplicación en cualquier maquina con cualquier sistema operativo es tener disponible la versión de la plataforma. En la actualidad la versión solo está disponible para Windows (Sotomayor, 2002).

Multilenguaje: Es multilenguaje ya que este lenguaje de programación puede ser adaptado a este y así poder ejecutarse (Sotomayor, 2002).

Interoperabilidad: Esta característica ayuda a que cuando se escribe varios tipos de código escritos en diferentes lenguajes la interoperabilidad es total (Sotomayor, 2002).

[7] SO: Sistema operativo

1.4.1.2 Visual Studio .NET

Visual Studio .net es una herramienta de Microsoft, distribuida junto a la plataforma. Net, esta plataforma nos ayuda a crear nueva aplicaciones con .net. Esta herramienta soporta los siguientes lenguajes de programación:

(Sotomayor, 2002)

- Visual Basic.NET
- Visual C++.NET
- Visual C#.NET
- Visual J#.NET

1.4.1.3 C#.NET

C#[8]es un lenguaje de programación que ha sido introducido por Microsoft a la plataforma .NET. Este lenguaje ha sido creado y adaptado a dicha plataforma, para poder trabajar exclusivamente en ella (Sotomayor, 2002).

El lenguaje C# es conceptualizado como un sub conjunto de visual C++, pero es más simple y seguro al momento de generar código gestionado, al igual que todos los lenguajes que conforman la plataforma .net el código fuente de este lenguaje es compilado con MSIL y es ejecutado por el CLR[9] (Sotomayor, 2002).

> Según Microsoft, C# es «un lenguaje de programación con la potencia de C, la productividad de Visual Basic y la elegancia de Java». Ciertamente, si vemos un trozo de código de C# nos daremos cuenta del asombroso parecido con Java. A su vez tenemos la sintaxis utilizada en Visual C++ y determinadas características que lo hacen muy potente (como la sobrecarga de operadores) combinada con la sencillez y facilidad de Visual Basic que hace que sea un lenguaje muy productivo. (Sotomayor, 2002, pág. 4)

"Desde otro punto de vista, Microsoft ha creado un lenguaje completo, orientado a objetos, que se acopla perfectamente con el desarrollo Web" (Sotomayor, 2002, pág. 4).

[8] C sharp o C números

[9] CLR significa common language runtime y permite las aplicaciones de .net en su entorno gestionado

1.4.1.4 ADO.NET

ADO.net es un conjunto de productos que brinda la plataforma de .Net para poder hacer uso de las bases de datos, estos productos evolucionaron de ADO y han sido pensados especialmente para aplicaciones con una arquitectura en n capas, que usan XML como su núcleo central (Gallegos, 2015).

1.4.1.5 Linq

LINQ[10] es nuevo lenguaje introducido en Visual Studio 2008 con .net framework 3.5, este lenguaje elimina la distancia entre el mundo de los objetos y el mundo de los datos, ya que antiguamente las consultas de datos se expresaban solo como cadenas de texto sencillas, estas cadenas de texto no se podían comprobar si estaban correctas en tiempo de compilación ya que no tenían compatibilidad con IntelliSense[11], además también era tedioso tener que aprenderse un lenguaje para cada origen de datos utilizado. Con LINQ ahora ya no hay necesidad de eso, ya que convierta una consulta de datos en C# o Visual Basic. Este lenguaje utiliza colecciones de objetos fuertemente tipiados, palabras claves y operadores del lenguaje con el que ya se está acostumbrado (Microsofth, s.f.).

> En Visual Studio se pueden escribir consultas LINQ en Visual Basic o en C# con bases de datos SQL Server, documentos XML, conjuntos de datos ADO.NET y cualquier colección de objetos que admita IEnumerable o la interfaz genérica IEnumerable<T>. También se ha previsto la compatibilidad de LINQ con ADO.NET Entity Framework, y otros fabricantes se encuentran escribiendo proveedores LINQ para muchos servicios Web y otras implementaciones de bases de datos. (Microsofth, s.f., pág. 1)

1.4.1.6 Servicios web ASP.NET

Definición de un servicio web XML

Un servicio web XML es un componente de una aplicación, pequeño y reusable, que esta contado como un bloque de construcción para mejorar el rendimiento de las tareas y procesos a beneficio de los usuarios. Microsoft y otros están desarrollando un conjunto básico de estos servicios (Gallegos, 2015).

[10] Language-Integrated Query

[11] IntelliSense: Ayuda a generar automáticamente código en el Editor de código de Visual Studio

Servicios Web basados en XML

Son los bloques de construcción de la tercera generación de Internet. Algunas de sus características son:

- Permiten a las aplicaciones compartir datos.
- Son componentes. Es decir, unidades de código discretas, cada una haciendo una tarea en particular.
- Están basados en el lenguaje universal de intercambio de datos de Internet: XML.
  Pueden ser llamados desde distintos sistemas operativos, plataformas de hardware y lenguajes de programación (Gallegos, 2015).

1.4.2 SQL Server 2012

1.4.2.1 El motor de base de datos de SQL Server 2012

El motor de base de datos de SQL Server 2012 es un servicio de la aplicación central de SQL Server, este servicio se utiliza para guardar, procesar y proteger información en un entorno relacional (Herrarte, 2007). Este motor de base de datos se encarga principalmente de:

- Brinda un almacenamiento confiable para datos
- Brinda un entorno rápido de recuperación de datos
- El acceso es consistente a los datos
- Brinda un control al acceder datos
- Cumple reglas de integridad de datos confirmando así que los datos sean viables y consistentes
- Registra cada cambio que se haga a una base de datos (Herrarte, 2007).

1.4.2.2 SQL Server Management Studio

SSMS[12] es la herramienta en la cual se realiza la gestión de las bases de datos. Esta herramienta brinda una única interfaz donde se pueden manejar todos los servidores de una empresa (Herrarte, 2007).

Las tareas principales que realiza esta herramienta son las siguientes:

- Gestiona todos los servidores en una única interfaz
- Configura las opciones del servidor, estas pueden ser la capacidad del procesador, la capacidad de memoria entre otros
- Administra los usuarios y roles así como accesos a la herramienta
- Realiza copias de seguridad

[12] SSMS: SQL Server Managment Studio

- Restaura base de datos
- Define planes para el mantenimiento de las base de datos
- Realiza la creación de nuevas bases de datos
- Realiza consultas, inserciones, modificaciones y eliminaciones de las tablas de base de datos (Herrarte, 2007)

1.4.2.3 Transact SQL

SQL se define como un lenguaje de consultas que se usa para las bases de datos relaciónales, este no es un lenguaje de programación ya que no tiene la potencia de este ya que solo es un lenguaje de consulta, porque no permite el uso de variables, estructuras de control de flujo, los bucles entre otros. Pero no por ello se debe menospreciar al lenguaje ya que este es una herramienta perfecta para trabajar con una base de datos. Por lo tanto Transact SQL es un lenguaje de programación creado para poder extender las funcionalidades estándares del SQL normal y así poder ser un lenguaje de programación (Herrarte, 2007).

Con la ayuda de Transact SQL podemos realizar las siguientes actividades: Procedimientos almacenados, funciones, triggers y ejecutar scripts (Herrarte, 2007).

Procedimientos almacenados

Los procedimientos almacenados son los encargados de ejecutar una o varias acciones específicas, a los procedimientos almacenados se les debe dar un nombre, opcionalmente uno o varios parámetros y un bloque donde se le diga que acción realizar, con esa estructura los procedimientos almacenados son capaces de devolver valores o realizar la acción requerida (Herrarte, 2007).

Funciones

Las funciones brindar al usuario la opción de crear nuevas funciones propias, estas funciones son conocidas comúnmente como UDF[13] (Herrarte, 2007).

Las funciones que puede crear el usuario son de tres tipos: Escalares, en línea y en línea múltiples.

Las funciones escalares son las funciones que devuelven solamente un valor, este valor puede ser de cualquier tipo como por ejemplo int, decimal, char entre otros (Herrarte, 2007).

[13] User defined functions

Las funciones en línea son aquellas funciones que devuelven conjuntos de datos, estos conjuntos de datos son el resultado de ejecutar consultas con el select (Herrarte, 2007).

Las funciones en línea múltiple son parecidas a las en línea ya que también devuelven como resultado un conjunto de datos, pero este conjunto de datos son el resultado de consultas anidadas select, por ende estas funciones son más complejas de hacer y requieren mucha más lógica (Herrarte, 2007).

Triggers

Los triggers o desencadenadores son un tipo de procedimientos almacenados que se ejecutan automáticamente apenas se realiza un evento dentro del servidor de una base de datos (Herrarte, 2007).

En Transact SQL existen las siguientes clases de triggers:

Trigger DML: Estos trigger se ejecutan cuando se ejecutan eventos DML, estos eventos DML son las instrucciones INSERT, UPDATE, DELETE de una tabla o de una vista (Herrarte, 2007).

Trigger DDL: Estos triggers se ejecutan cuando se producen eventos DDL, estos eventos pueden ser CRÉATE, DROP, ALTER entre otros (Herrarte, 2007).

1.4.2.4 Administrador de configuración de SQL Server

El administrador de configuración de SQL Server o en ingles llamado SQL Server Config Manager es la herramienta en la que se gestiona los servicios relacionados con SQL Server donde pueden ser configurados los protocolos de red del SQL Server, también se utiliza como administrador del servidor para poder iniciar, pausar, reanudar y detener las tareas de SQL (Ramonmorillos Weblog, 2012).

2 Capítulo 2: Situación activa y perspectiva de la carrera

Este capítulo tiene el propósito de realizar un análisis exhaustivo de la situación de la carrera de Ingeniería en Sistemas de Pontificia Universidad Católica del Ecuador, así como también tiene la intención de realizar una investigación sobre las leyes y los organismos públicos que se encargan de los Institutos de Educación Superior.

Estos análisis servirán para tener un conocimiento previo del problema planteado para poder realizar la implementación del sistema.

2.1 Perspectiva de la carrera

A continuación se presenta la misión, visión y objetivos de la Carrera de Ingeniería en Sistemas y Computación de la Pontificia Universidad Católica del Ecuador.

2.1.1 Misión

> *La carrera de Ingeniería de Sistemas y Computación de la PUCE[14] es un conjunto de estudios integrados sistémicamente, que permite formar a los estudiantes en un marco de principios cristianos, con pedagogía ignaciana e investigación formativa, habilitando a sus egresados para el ejercicio de la ingeniería en las áreas de las tecnologías de la información y la comunicación; de una manera profesional, responsable, buscando la innovación y el servicio a la sociedad (Puceing, s.f., pág. 1).*

2.1.2 Visión

"En el año 2017 la carrera de Ingeniería de Sistemas y Computación de la PUCE mantiene su acreditación académica nacional e internacional" (Puceing, s.f., pág. 1).

2.1.3 Objetivos

"Formar Ingenieros en Sistemas con sólidos conocimientos técnicos en diferentes áreas de: telecomunicaciones, sistemas de información, permitiendo cubrir la demanda de profesionales en el ámbito privado y público" (Puceing, s.f., pág. 1).

"Formar profesionales con capacidad objetiva, analítica y crítica, comprometidos en mejorar el nivel de vida en el País" (Puceing, s.f., pág. 1).

[14] Pontificia Universidad Católica del Ecuador

2.2 Malla

Podemos observar en la figura 02-01 la estructura de la malla curricular de la carrera de Ingeniería en Sistemas de la PUCE.

En dicha figura se puede observar la catedra "Cálculo integral" que pertenece al segundo nivel, cuyos prerrequisitos son "Cálculo diferencial" y "Álgebra lineal" ambas cátedras de primer nivel, el número de créditos que tiene la catedra "Cálculo integral" son cuatro.

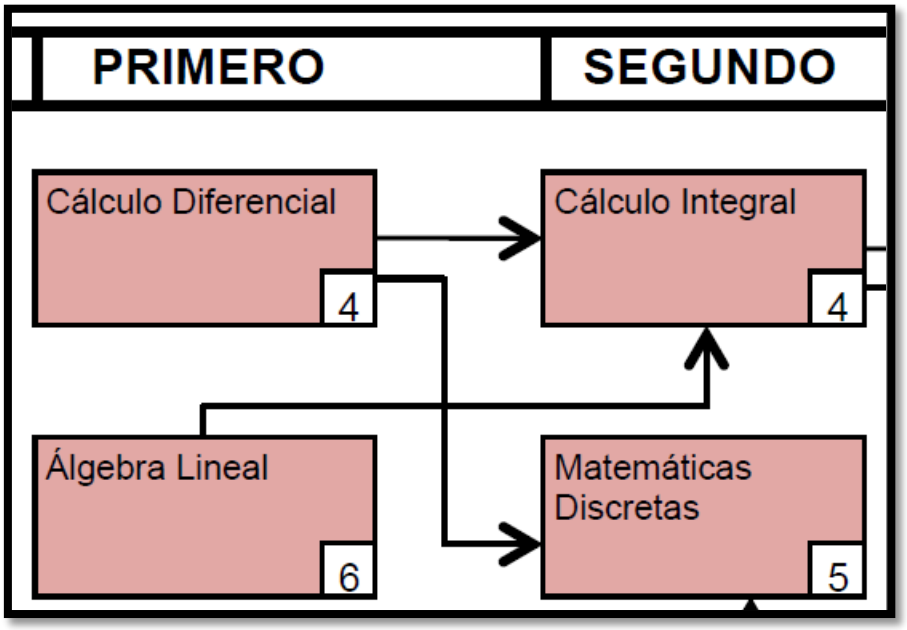

Figura 2-1 Estructura de la malla curricular (Malla curricular, 2015)

Para poder observar toda la estructura de la malla curricular ver Anexo1.

2.3 CES

> El Consejo de Educación Superior (CES) tiene como su razón de ser planificar, regular y coordinar el Sistema de Educación Superior, y la relación entre sus distintos actores con la Función Ejecutiva y la sociedad ecuatoriana; para así garantizar a toda la ciudadanía una Educación Superior de calidad que contribuya al crecimiento del país (CES, 2012).

"El CES trabajará en coordinación con el Consejo de Evaluación, Acreditación y Aseguramiento de la Calidad de la Educación Superior- CEAACES para continuar con la Revolución en el Conocimiento de una forma integral y profunda" (CES, 2012).

2.3.1 Misión

El Consejo de Educación Superior, como uno de los dos organismos que rigen el sistema, tiene como misión la planificación, regulación y coordinación interna del Sistema de Educación Superior del Ecuador, y la relación entre sus distintos actores con la Función Ejecutiva y la sociedad ecuatoriana (CES, 2012).

2.3.2 Visión

Ser el organismo público referente para los procesos que consoliden el Sistema de Educación Superior, ejerciendo sus competencias constitucionales y legales, de forma que incidan decisivamente en el logro de la excelencia de la educación superior mediante la formación académica y profesional, con visión científica y humanística que contribuya con soluciones a los problemas del país articulados al régimen de desarrollo y al del buen vivir; respetando los principios constitucionales que rigen a las Instituciones y al Sistema de Educación Superior (CES, 2012).

2.3.3 Objetivos Estratégicos

a) "Resolver sobre la planificación, regulación y coordinación del Sistema de Educación Superior" (CES, 2012).
b) "Administrar el Sistema de Licenciamiento de la Educación Superior, en conformidad a las normas establecidas en la ley;" (CES, 2012).
c) "Monitorear, controlar y sancionar a las Instituciones de Educación Superior según lo establecido en la ley;" (CES, 2012).
d) "Formular, en coordinación con los otros organismos que rigen el Sistema de Educación Superior y con el órgano que tiene por objeto ejercer la rectoría de la política pública de educación superior, las políticas de Estado y la planificación del Sistema de Educación Superior;" (CES, 2012).
e) "Aprobar y formular la normativa requerida para el funcionamiento del Sistema de Educación Superior y para el ejercicio de sus competencias;" (CES, 2012).
f) "Gestionar la información remitida por el órgano que tiene por objeto ejercer la rectoría de la política pública de educación superior y otras entidades como sustento para las resoluciones" (CES, 2012).

2.4 LOES (Ley Orgánica de Educación Superior)

A continuación se muestra cómo funciona el personal académico según la Ley Orgánica de Educación Superior.

Sobre el personal académico el artículo 147 de (Ley Orgánica de Educación Superior, 2010) dice:

> *El personal académico de las universidades y escuelas politécnicas está conformado por profesores o profesoras e investigadores o investigadoras. El ejercicio de la cátedra y la investigación podrán combinarse entre sí. Lo mismo que con actividades de dirección, si su horario lo permite, sin perjuicio de lo establecido en la Constitución en esta Ley, y el Reglamento de Carrera y Escalafón del Profesor e Investigador del Sistema de Educación Superior.*

Sobre los tipos de docentes y su tiempo de dedicación el artículo 149 de (Ley Orgánica de Educación Superior, 2010) dice:

> *Los profesores o profesoras e investigadores o investigadoras serán: titulares, invitados, ocasionales u honorarios. Los profesores titulares podrán ser principales, agregados o auxiliares. El reglamento del sistema de carrera del profesor e investigador regulará los requisitos y sus respectivos concursos. El tiempo de dedicación podrá ser exclusiva o tiempo completo, es decir, con cuarenta horas semanales: semiexclusiva o medio tiempo, es decir, con veinte horas semanales: a tiempo parcial, con menos de veinte horas semanales. Ningún profesor o funcionario administrativo con dedicación exclusiva o tiempo completo podrá desempeñar simultáneamente dos o más cargos de tiempo completo en el sistema educativo, en el sector público o en el sector privado. El Reglamento de Carrera y Escalafón del Profesor e Investigador del Sistema de Educación Superior. Normará esta clasificación, estableciendo las limitaciones de los profesores. En el caso de los profesores o profesoras de los institutos superiores y conservatorios superiores públicos se establecerá un capítulo especial en el Reglamento de Carrera y Escalafón del Profesor e Investigador del Sistema de Educación Superior (Ley Orgánica de Educación Superior, 2010).*

Los requisitos de los docentes principales la (Ley Orgánica de Educación Superior, 2010) dice:

> *Para ser profesor o profesora titular principal de una universidad o escuela politécnica pública o particular del Sistema de Educación Superior se deberá cumplir con los siguientes requisitos:*
>
> - *Tener título de posgrado correspondiente a doctorado (PhD o su equivalente) en el área afín en que ejercerá la cátedra;*
> - *Haber realizado o publicado obras de relevancia o artículos indexados en el área afín en que ejercerá la cátedra, individual o colectivamente, en los últimos cinco años;*
> - *Ser ganador del correspondiente concurso público de merecimientos y oposición; y.*
> - *Tener cuatro años de experiencia docente, y reunir los requisitos adicionales, señalados en los estatutos de cada universidad o escuela politécnica, en ejercicio de su autonomía responsable, los que tendrán plena concordancia con el Reglamento de Carrera y Escalafón del Profesor e Investigador del Sistema de Educación Superior.*
> - *Los profesores titulares agregados o auxiliares deberán contar como mínimo con título de maestría afín al área en que ejercerán la cátedra, los demás requisitos se establecerán en el reglamento respectivo.*

Para saber cómo se realizara el concurso de merecimientos para obtener una catedra como docente titular el artículo 152 de la (Ley Orgánica de Educación Superior, 2010) dice:

> *En las universidades y escuelas politécnicas públicas, el concurso público de merecimientos y oposición para acceder a la titularidad de la cátedra deberá ser convocado a través de al menos dos medios de comunicación escrito masivo y en la red electrónica de información que establezca la Secretaría Nacional de Educación Superior y Ciencia. Tecnología e Innovación, a través del Sistema Nacional de Información de la Educación Superior del Ecuador y en los medios oficiales de la universidad o escuela politécnica convocante. Los miembros del jurado serán docentes y deberán estar acreditados como profesores titulares en sus respectivas universidades y estarán conformados por un 40% de miembros externos a la universidad o escuela politécnica que está ofreciendo la plaza titular. En el caso de las universidades y escuelas politécnicas particulares, su estatuto establecerá el procedimiento respectivo.*

Para saber sobre los requisitos para docentes no titulares, el artículo 153 de la (Ley Orgánica de Educación Superior, 2010) dice: "Los requisitos para ser profesor o profesora invitado, ocasional u honorario serán establecidos en el Reglamento de Carrera y Escalafón del Profesor e Investigador del Sistema de Educación Superior".

Para obtener más conocimiento sobre la Ley Orgánica de Educación superior ver anexo 2.

2.5 Reglamento de régimen académico

El reglamento de régimen académico nos muestra la estructuración de los niveles de formación de la educación superior.

Según el Artículo 4 la "Organización académica de los niveles de formación de la educación superior. Los diversos niveles de formación de la educación superior responden a necesidades específicas de profundización y diversificación académica y profesional, acorde a los objetos de conocimiento e intervención" (Reglamento de régimen académico codificado, 2013, pág. 4).

Según el Artículo 5 del (Reglamento de régimen académico codificado, 2013, pág. 4)

> *Niveles de formación de la educación superior.- El sistema de educación superior se organiza a partir de los siguientes niveles de formación:*
>
> *a) Educación técnica superior y sus equivalentes;*
> *b) Educación tecnológica superior y sus equivalentes;*
> *c) Educación superior de grado o de tercer nivel; y,*
> *d) Educación superior de posgrado o de cuarto nivel.*

Es necesario también conocer la estructura curricular de una catedra; lo cual dice:

Según el Artículo 20 los Componentes de la estructura curricular son "Los conocimientos disciplinares, interdisciplinares, transdisciplinares, profesionales, investiga ti vos, de saberes integrales y de comunicación, necesarios para desarrollar el perfil profesional y académico del estudiante se organizarán en asignaturas, cursos o sus equivalentes" (Reglamento de régimen académico codificado, 2013, pág. 12).

"A su vez estos componentes de la estructura curricular se organizan a [o largo del proceso de aprendizaje a través de las unidades de organización curricular y de los campos de formación del currículo" (Reglamento de régimen académico codificado, 2013, pág. 12).

"Las unidades de organización curricular son formas de ordenamiento de las asignaturas, cursos o sus equivalentes a lo largo de la carrera o programa, que permiten integrar el aprendizaje en cada período académico, articulando los conocimientos de modo progresivo" (Reglamento de régimen académico codificado, 2013, pág. 12).

"Los campos de formación son formas de organización de los conocimientos en función de sus propósitos y objetivos" (Reglamento de régimen académico codificado, 2013, pág. 12).

"Las carreras y programas deberán incluir en la planificación de las unidades de organización curri cular y de los campos de formación, redes, adaptaciones y vínculos transversales, que permitan abordar el aprendizaje de modo integrado e innovador" (Reglamento de régimen académico codificado, 2013, pág. 12).

Según el Artículo 28 de (Reglamento de régimen académico codificado, 2013, págs. 16-17) los campos de formación de la educación superior o de tercer nivel se deben organizar de la siguiente manera:

1. **Fundamentos teóricos.-** Integra el conocimiento de los contextos, principios, lenguajes, métodos de la o las disciplinas que sustentan la profesión, estableciendo posibles integraciones de carácter multi e inter disciplinar.
2. **Praxis profesional.-** Integra conocimientos teóricos metodológicos y técnicos instrumentales de la formación profesional e incluye las prácticas pre profesionales, los sistemas de supervisión y sistematización de las mismas.
3. **Epistemología y metodología de la investigación.-** Integra los procesos de indagación, exploración y organización del conocimiento profesional cuyo estudio está distribuid o a lo largo de la carrera. Este campo genera competencias investigativas que se desarrollan en los contextos de práctica de una profesión. En este campo formativo se incluirá el trabajo de titulación.
4. **Integración de saberes, contextos y cultura.-** Comprende las diversas perspectivas teóricas, culturales y de saberes que complementan la formación profesional, la educación en valores y en derechos ciudadanos, así como el estudio de la realidad socioeconómica cultural y ecológica del país y el mundo. En este campo formativo se incluirán además, los itinerarios multiprofesionales, multidisciplinares, interculturales e investigativos.
5. **Comunicación y lenguajes.-** Comprende el desarrollo del lenguaje y de habilidades para la comunicación oral, escrita y digital necesarios para la elaboración de discursos y narrativas académicas y científicas. Incluye, además aquí ellas asignaturas, cursos, o sus equivalentes, orientados al dominio de la ofimática (manejo de nuevas tecnologías de la información y la comunicación), y opcionalmente, de lenguas ancestrales. Las asignaturas destinadas al aprendizaje de la ofimática serán tomadas u homologadas necesariamente desde el inicio de la carrera, pudiendo los estudiantes rendir una prueba de suficiencia y exoneración, general o por niveles al inicio de cada período académico.

Para mayor información sobre el régimen académico por favor consulte el anexo 2.

2.6 CEAACES

Es un organismo técnico, público y autónomo encargado de ejercer la rectoría política para la evaluación, acreditación y el aseguramiento de la calidad de las Instituciones de Educación Superior, sus programas y carreras. Para ello, realizan procesos continuos de evaluación y acreditación que evidencien el cumplimiento de las misiones, fines y objetivos de las mismas (CEAACES, 2014).

2.6.1 Misión

"Ejercer la rectoría de la política pública para el aseguramiento de la calidad de la educación superior del Ecuador a través de procesos de evaluación, acreditación y categorización en las IES[15]" (CEAACES, 2014).

2.6.2 Visión

"Ser un referente nacional y regional en la creación e implementación de metodologías integrales, articuladas y transparentes de evaluación, acreditación y aseguramiento de la calidad de la educación superior" (CEAACES, 2014).

2.6.3 Proceso de Evaluación, acreditación y categorización institucional

El proceso de evaluación institucional externa del CEAACES se inició con la construcción del modelo en abril de 2012, que posteriormente fue modificado recogiendo las observaciones y recomendaciones hechas por las universidades y escuelas politécnicas a través de los procesos de socialización llevados a cabo en las ciudades de Quito, Guayaquil, Cuenca, Manta y Ambato entre el 22 de Octubre y el 5 de Noviembre de 2012 (CEAACES, 2014).

El proceso ha finalizado en noviembre de 2013 con todas sus etapas, a saber: evaluación documental, visita in situ, informe preliminar, fase de rectificaciones, fase de apelaciones y audiencias públicas. Como consecuencia de este proceso de evaluación, y conforme al artículo 97 de la LOES, el CEAACES ha determinado la acreditación de las IES que cumplen con los criterios y estándares básicos de calidad definidos por el Consejo, y la nueva categorización de las universidades y escuelas politécnicas del sistema de educación superior del Ecuador (CEAACES, 2014).

[15] Instituto de Educación Superior

2.6.4 Categorización de Universidades

"La categorización de universidades y escuelas politécnicas es el resultado de la aplicación de tres modelos diferentes y específicos aplicados según la oferta académica de las IES…" (CEAACES, 2014).

Las tres ofertas son las siguientes:

"IES con oferta académica de grado, IES con oferta académica de grado y posgrado e IES con oferta académica de posgrado" (CEAACES, 2014).

> *En el año 2013 la clasificación de las IES responde a la aplicación conjunta de las metodologías de análisis multicriterio y el análisis de conglomerados a los resultados obtenidos por las instituciones en los respectivos modelos, los que estaban constituidos por cinco criterios: academia, eficiencia académica, investigación, organización e infraestructura. (CEAACES, 2014)*

3 Capítulo 3: Desarrollo de la aplicación

Este capítulo contiene la documentación necesaria para el desarrollo de la aplicación, que fue implementada con la metodología de programación extrema.

3.1 Análisis

3.1.1 Matriz técnica de requerimientos funcionales

En las tablas 3-1A y 3-1B se presentan los requerimientos funcionales del sistema, donde se indica la prioridad y una descripción breve de cada requerimiento.

Requerimiento funcional	Descripción	Prioridad
Módulo docente	El sistema será capaz de manejar un módulo docente.	Alta
Administrar docentes	El sistema será capaz de insertar, modificar, eliminar, consultar docentes y también realizar una importación de docentes de un archivo Excel a la base de datos.	Alta
Administrar cátedras por docente	El sistema será capaz de insertar, modificar, eliminar, consultar las cátedras que son dictadas por cada docente existente dentro de la base de datos.	Alta
Administrar IES[16] por docente	El sistema será capaz de insertar, modificar, eliminar, consultar las IES donde trabaja cada docente existente dentro de la base de datos.	Alta
Administrar títulos por docentes	El sistema será capaz de insertar, modificar, eliminar, consultar los títulos obtenidos por cada docente existente dentro de la base de datos.	Alta
Módulo IES	El sistema será capaz de manejar un módulo de IES	Alta
Administrar IES	El sistema será capaz de insertar, modificar, eliminar y consultar IES.	Alta
Administrar Mallas	El sistema será capaz de insertar, modificar, eliminar y consultar Mallas.	Alta
Administrar Cátedras por malla	El sistema será capaz de insertar, modificar, eliminar y consultar cátedras que dependen de la malla y la IES.	Alta
Administrar capítulos por cátedras por malla	El sistema será capaz de insertar, modificar, eliminar y consultar capítulos por catedra por malla.	Alta
Administrar prerrequisitos por cátedras por malla	El sistema será capaz de insertar, modificar, eliminar y consultar capítulos por catedra por malla.	Alta
Administrar RDA por cátedras por malla	El sistema será capaz de insertar, modificar, eliminar y consultar RDA por catedra por malla.	Alta

Tabla 3-1A Matriz técnica de requerimientos (Benalcázar & Manzano, 2016)

[16] Instituto de Educación Superior

Requerimiento funcional	Descripción	Prioridad
Módulo cátedras	El sistema será capaz de manejar un módulo cátedras	Alta
Administrar cátedras	El sistema será capaz de insertar, modificar, eliminar y consultar cátedras	Alta
Administrar cátedras por nivel	El sistema será capaz de mostrar todas las cátedras ordenadas por nivel, filtradas por malla. También mostrara el syllabus de cada catedra.	Alta
Comparativa de cátedras	El sistema será capaz de mostrar simultáneamente los RDAs y capítulos de dos cátedras que el usuario requiera.	Alta
Modulo usuarios y roles	El sistema será capaz de manejar un módulo de usuarios y roles que servirá pasara para la autenticación del sistema.	Alta
Administrar usuarios	El sistema será capaz de insertar, modificar, eliminar y consultar usuarios.	Alta
Administrar roles	El sistema será capaz de insertar, modificar, eliminar y consultar roles.	Alta

Tabla 3-1B Matriz técnica de requerimientos (Benalcázar & Manzano, 2016)

3.1.2 Diagramas generales de caso de uso del sistema

Diagrama general

En la figura 3-1 se puede observar el comportamiento del sistema a nivel general, el sistema consta de 4 módulos, en los cuales los actores Gerente y Técnico tienen acceso a los módulos Docente, IES y Cátedra mientras que el actor Administrador solo tiene acceso al Modulo Usuarios Y Roles.

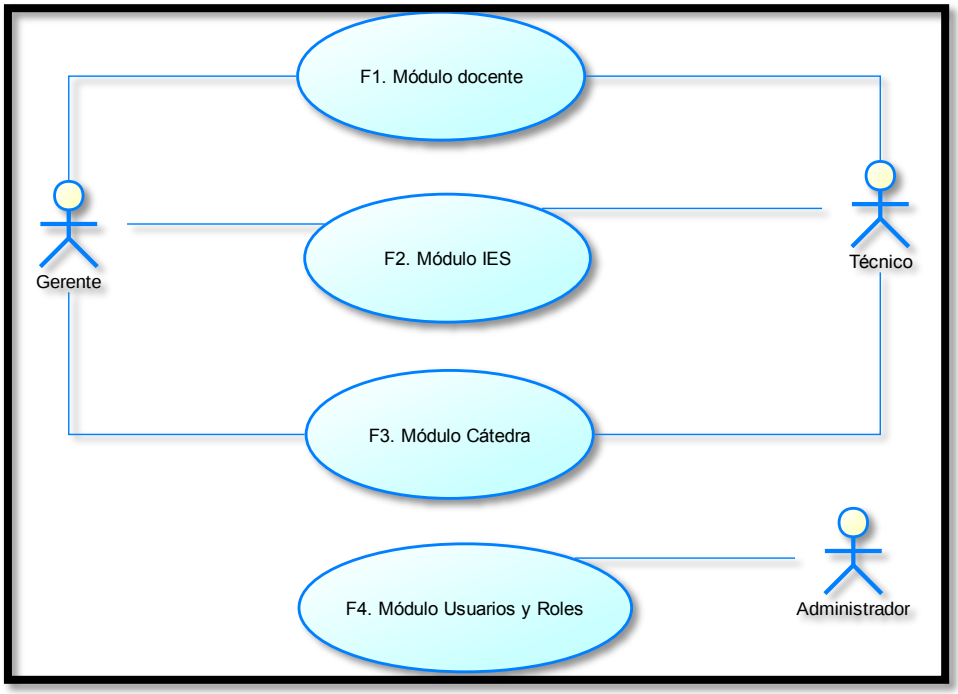

Figura 3-1 Diagrama General de Casos de Uso (Benalcázar & Manzano, 2016)

F1. Modulo docente

En la figura 3-2 se puede ver la estructura de lo que será el modulo docente, este módulo contara con las funcionalidades de administrar docente, docente por catedra, docente por IES y docentes por títulos. En el cual el actor Gerente tiene acceso de vista a todo el modulo mientras que el actor técnico tiene un acceso total al módulo docente.

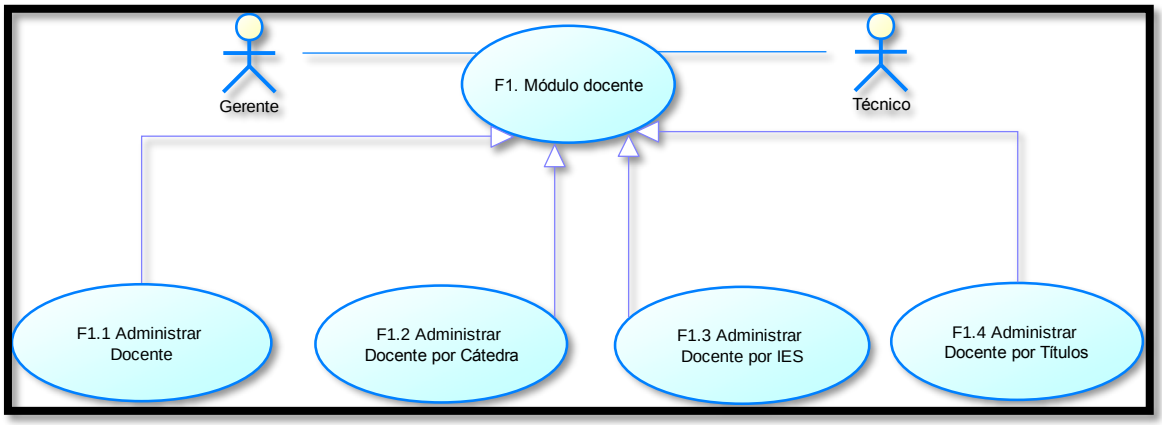

Figura 3-2 Casos de uso módulo docente (Benalcázar & Manzano, 2016)

F2. Módulo IES

La figura 3-3 nos muestra que el módulo IES presenta las funcionalidades administrar IES, mallas, cátedras por malla, capítulos por catedra por malla, prerrequisitos por catedra por malla, RDA por catedra por malla las tres últimas dependen de cátedras por malla. El actor Gerente tiene accesos de vista de las tres primeras funcionas y el actor técnico tiene un acceso total al módulo.

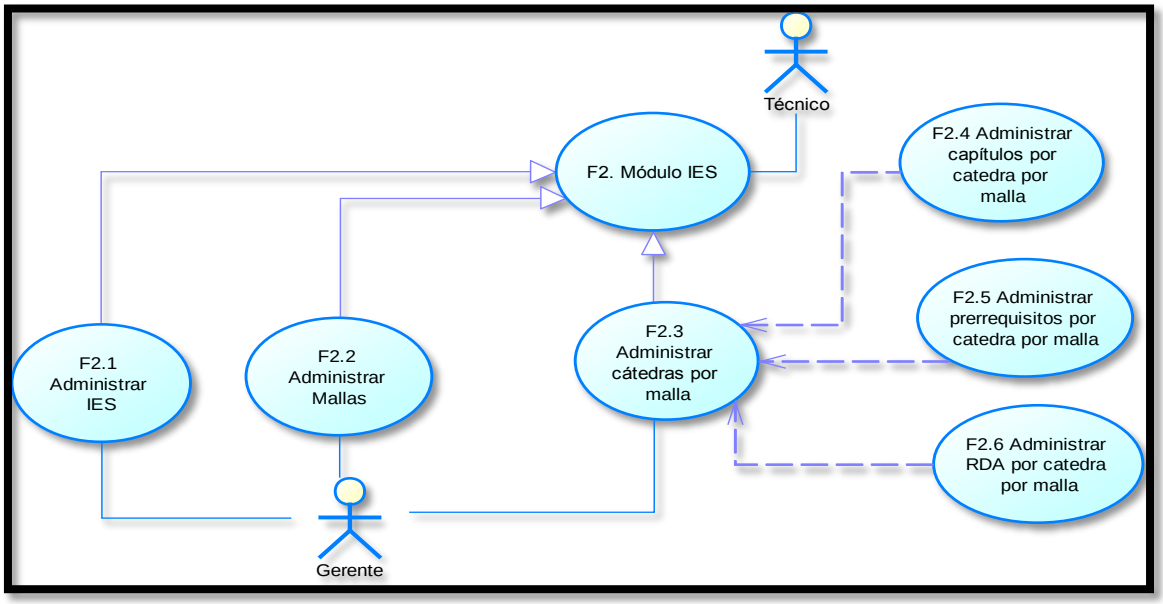

Figura 3-3 Casos de uso módulo IES (Benalcázar & Manzano, 2016)

F3. Módulo cátedras

En módulo cátedras se encarga de la administración de cátedras, ver cátedras por nivel y comparativo cátedras, el actor Gerente tiene acceso a las dos últimas funciones, mientras que el actor técnico tiene acceso a todo el modulo. Esto se puede mirar en la figura 3-4.

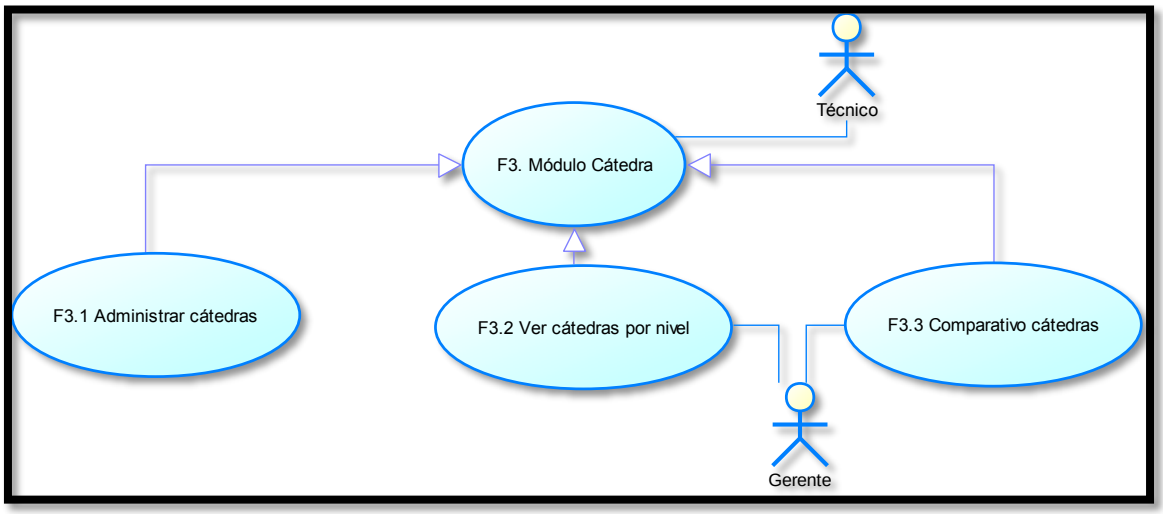

Figura 3-4 Casos de uso módulo Cátedra (Benalcázar & Manzano, 2016)

F4. Módulo Usuarios y Roles

Según la figura 3-5 el modulo usuario consta de las funcionalidades Administrar usuario y roles, donde el actor Administrador es el único que puede acceder a este módulo.

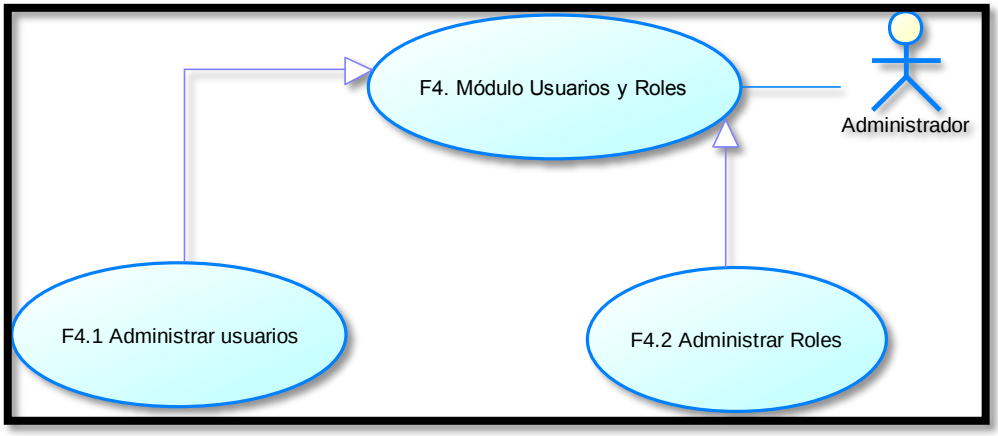

Figura 3-5 Casos de uso módulo Usuarios y Roles (Benalcázar & Manzano, 2016)

3.1.3 Historias de usuario

Se realizaron en total 23 historias de usuario, donde fueron divididas en un máximo de 4 historias de usuario por iteración, las historias de usuario fueron respaldadas también con el diseño de las pantallas de requerimientos que se pueden observar en el anexo 6 adjuntado.

Las iteraciones se realizaron en un tiempo máximo de tres semanas, esto dependía de la complejidad de las historias de usuario que se desarrollaban.

Cronograma de planificación

En el cronograma de planificación, se resume mediante la tabla 3-2 la planificación establecida de entregas de iteraciones y el modulo al que pertenecen.

Fecha de entrega	Numero de iteración	Historias de usuario realizadas en la iteración	Modulo al que pertenece la iteración	Desarrolladores responsables
04/01/2016	1	RF1,RF2,RF3,RF6	Docentes	Andrea Benalcázar y Paúl Manzano
25/01/2016	2	RF5,RF6,RF7,RF8	Docentes	Andrea Benalcázar y Paúl Manzano
08/02/2016	3	RF9,RF10,RF11,RF12	Cátedras	Andrea Benalcázar y Paúl Manzano
29/02/2016	4	RF13,RF14,RF15,RF16	IES	Andrea Benalcázar y Paúl Manzano
14/03/2016	5	RF17,RF18,RF19	IES	Andrea Benalcázar y Paúl Manzano
28/03/2016	6	RF20,RF21,RF22,RF23	Usuarios y roles	Andrea Benalcázar y Paúl Manzano

Tabla 3-2 Cronograma de planificación (Benalcázar & Manzano, 2016)

Iteraciones

Las iteraciones se realizaron según el cronograma de planificación (Tabla 3-02), a continuación se relata un pequeño resumen de lo que se hizo en cada iteración, para mayor información se puede ver el Anexo 4, Anexo 5 y Anexo 6 relacionado con las iteraciones de las historias de usuario.

Iteración 1

En esta iteración se hicieron varios diseños de la base de datos para la aplicación, una vez obtenido el diseño más óptimo se empezó a desarrollar el modulo docentes con las primeras funcionalidades de este módulo.

Iteración 2

En la iteración dos se hicieron pequeñas modificaciones en la base de datos para mejorar su funcionamiento, y se siguió implementando las últimas funcionalidades del módulo docente.

Iteración 3

En esta iteración se desarrollaron todas las funcionalidades del módulo cátedras, y se hizo un gran cambio en la base para un mejor cumplimiento de la lógica del negocio del sistema. Para observar las versiones de la base de datos ver el Anexo 7.

Iteración 4

En la iteración número cuatro se implementaron las primeras funcionalidades del módulo IES.

Iteración 5

En la iteración número cinco se terminó de implementar las funcionalidades del módulo IES y también se diseñó la base de datos para implementar roles y permisos al sistema.

Iteración 6

En esta iteración se implementó los permisos de acceso a cada página del sistema y se desarrolló las funcionalidades del módulo usuario y roles.

3.2 Diseño

3.2.1 Entorno del software

Plataforma SO

La plataforma del Sistema Operativo usada para desarrollar el sistema es Windows 8.1.

Base de datos

La base de datos que se usó para crear el aplicativo es SQL server 2012, express edition.

Lenguaje de programación

El lenguaje de programación usado es C#, y .net con el entorno de programación de Visual Studio 2012.

Metodología de programación

La metodología utilizada para la implementación de la aplicación es programación extrema.

Modelo y arquitectura

El modelo y arquitectura utilizada para la creación del sistema es una arquitectura n-capas donde se utilizan 5 capas que son:

- Capa de entidades
- Capa de acceso a datos
- Capa lógica
- Capa presentación
- Capa de servicio

3.2.2 Diagrama de clases

A continuación se presenta la clase cátedras por malla de la lógica de negocio:

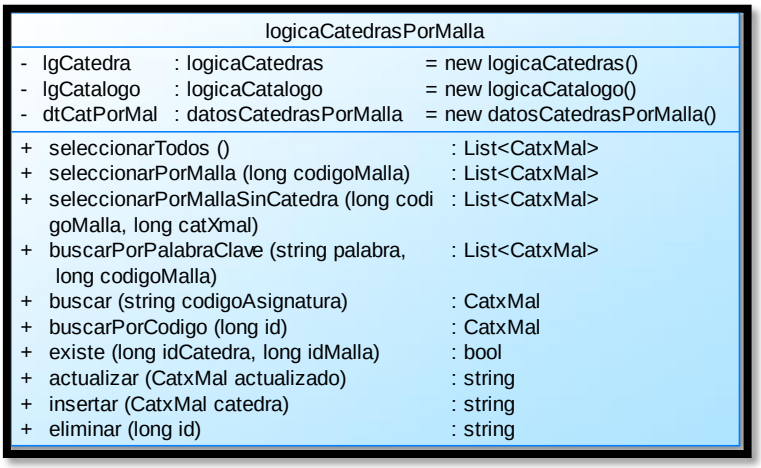

Figura 3-6 Diagrama de clases cátedras por malla(Benalcázar & Manzano, 2016)

Se puede observar según la figura 3-6, que la clase cátedras por malla, que tiene dependencia de catedra y catálogo.

Los métodos de esta clase son:

- SeleccionarTodos(): Este método devuelve en una lista todas las cátedras por malla existentes dentro de la base de datos.
- seleccionarPorMalla(long codigoMalla): Este método devuelve en una lista las cátedras por malla de la malla seleccionada.
- seleccionarPorMallaSinCatedra(long codigoMalla, long catXmal): Este método devuelve una lista de cátedras por malla filtradas por malla y sin mostrar la catedra requerida.
- buscarPorPalabraClave(string palabra, long codigoMalla): Devuelve una lista de cátedras por malla filtradas por malla y que cumplan con la condición de tener la palabra clave en cualquier campo tipo texto.
- Buscar(string códigoAsignatura): Devuelve la catedra por malla que tenga el código de asignatura.
- buscarPorCodigo(long id): Devuelve una catedra por malla que tenga el código ingresado.
- Existe(long idCatedra, long idMalla): Devuelve verdadero o falso dependiendo si existe una asociación anterior entre la malla y la catedra.
- Actualizar(): Actualiza en la base de datos una catedra por malla.
- Insertar(): Inserta en la base de datos una catedra por malla
- Eliminar(): elimina de la base de datos una catedra por malla

Para observar el diagrama de clases completo por favor mirar Anexo8.

3.2.3 Esquema de datos

Modelo conceptual

En la figura 3-7 podemos observar las tablas de Docente que hereda de persona, es decir que los atributos que tiene persona también va a tener docente.

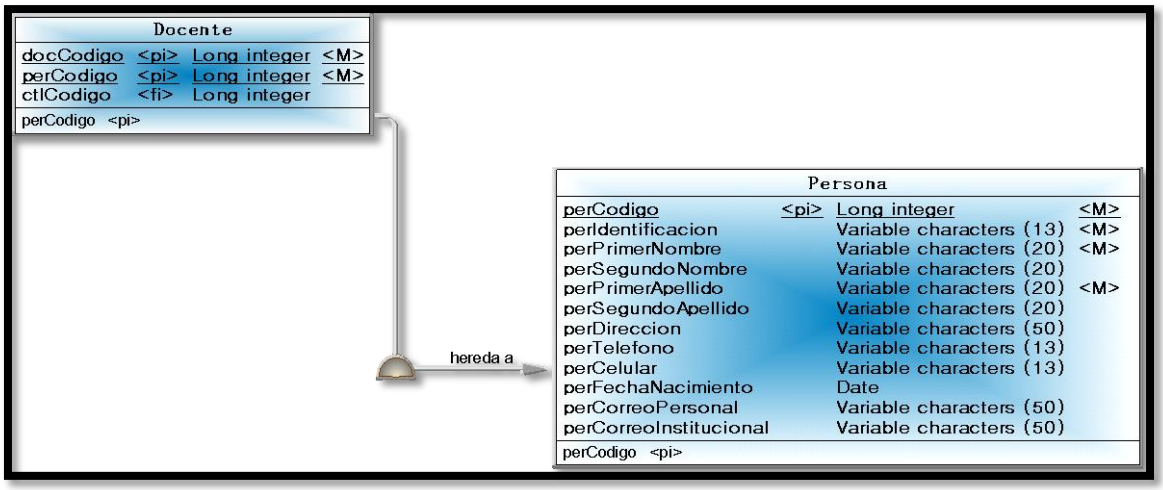

Figura 3-7 Modelo Conceptual de Base de datos (Benalcázar & Manzano, 2016)

Según la figura los atributos de la tabla Docente son:

- docCodigo: Este es un código secuencial numérico propio de la tabla
- perCodigo: Permite la herencia entre la tabla docente y persona
- ctlCodigo: Es un código que permite la asociación entre la tabla catalogo

Los atributos de la tabla persona que también comparte la tabla docente son:

- perCodigo: Es el código secuencial numérico propio de la tabla persona
- perIdentificacion: El número de identificación de una persona es un campo de caracteres , puede ser la el número de cedula o el número de pasaporte
- perPrimerNombre, perSegundoNOmbre:El primer y segundo nombre de una persona respectivamente, es un campo de caracteres
- perPrimerApellido, perSegundoApellido: El primer y segundo apellido respectivamente de una persona, son campos de caracteres
- perDireccion: La dirección de la persona, es un campo de caracteres
- perTelefono, perCelular: Aquí se guarda el teléfono y celular de la persona, campo de caracteres
- perFechaDeNacimiento: Campo tipo fecha, para almacenar la fecha de nacimiento de la persona
- perCorreoIntitucional, perCorreoPersonal: Almacena el correo personal e institucional respectivamente, son campos caracteres.

Para mirar el modelo conceptual completo ir a Anexo9,

Modelo físico

En la figura 3-8 podemos observar la tabla IES, y sus atributos.

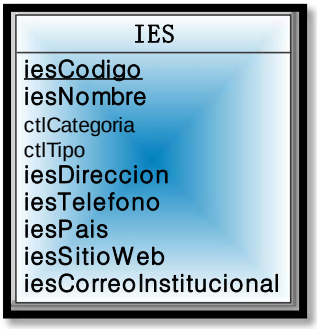

Figura 3-8 Modelo Físico de la base de datos (Benalcázar & Manzano, 2016)

Los atributos de la tabla IES son:

- IesCodigo: Es un campo numero secuencia propio de la tabla
- iesNombre: representa el nombre del IES y es un campo tipo texto
- ctlCategori: representa el código de la categoría que tiene la IES , es un campo numérico
- ctlTipo: representa el código del tipo de IES que tiene la IES, es un campo numerico
- iesDireccion: representa la dirección de la IES, es un campo tipo texto
- iesTelefonp: representa el teléfono que pertenece a la IES, es un campo tipo texto
- iesPais: representa al país que pertenece la IES, es un campo tipo texto
- iesSitioWeb: representa el sitio web de la IES, es un campo tipo texto
- iesCorreoInstitucional: representa el correo de información que tiene cada IES, es un campo tipo texto

Para observar el modelo físico completo de la base de datos mirar el Anexo10.

Y para observar todos los atributos de todas las tablas del sistema, se puede mirar el Diccionario de datos en el Anexo11.

3.2.4 Diseño de las interfaces

Para el diseño de las interfaces se realizó los siguientes prototipos pantallas:

La siguiente figura 3-9 muestra la pantalla general de cómo se mostrarán todas las pantallas en el sistema.

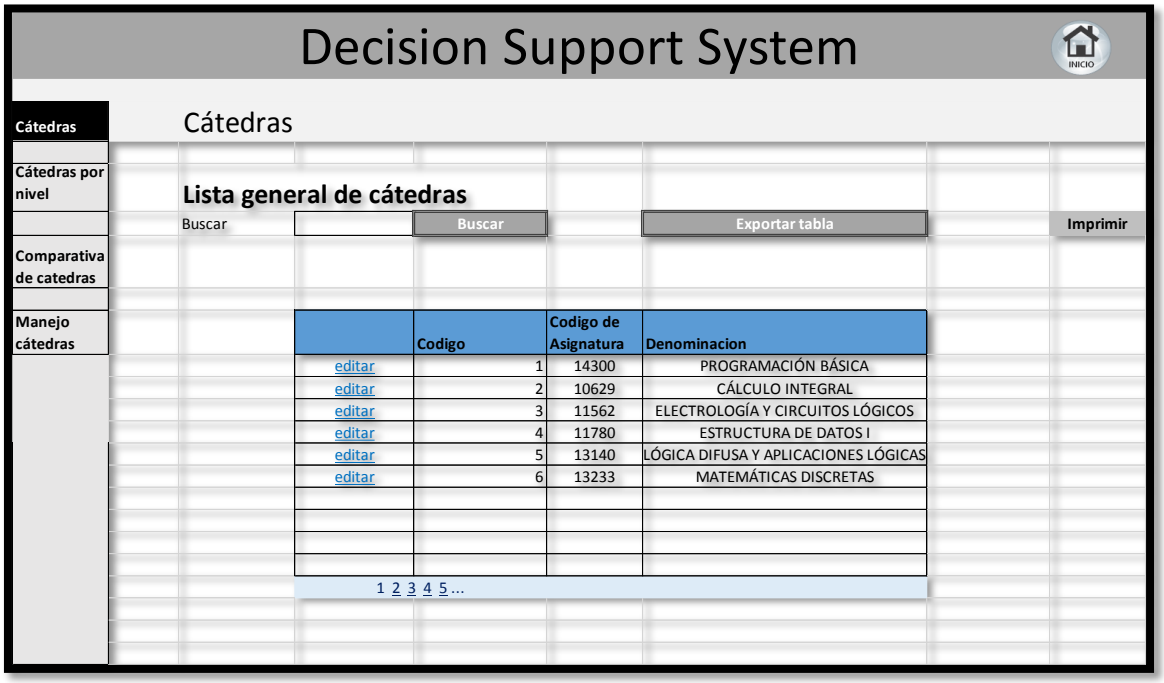

Figura 3-9 Pantalla general (Benalcázar & Manzano, 2016)

La figura 3-10 muestra el diseño de la cabecera de la pantalla, donde se puede observar que en la parte superior se encuentra el nombre del sistema, abajo en la parte inferior de la cabecera se encuentra el nombre del módulo donde se encuentra el usuario y finalmente en la parte derecha superior se encuentra el botón inicio para regresar al inicio de la aplicación.

Figura 3-10 Cabecera de pantalla (Benalcázar & Manzano, 2016)

La figura 3-11 nos muestra un diseño prototipo de cómo se van a observar las opciones por modulo en la pantalla.

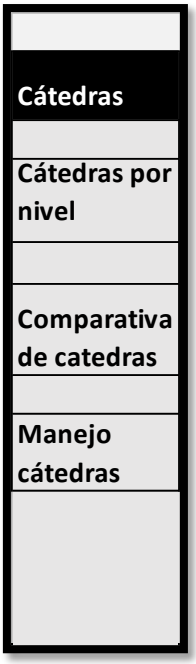

Figura 3-11 Lista de opciones diseño de pantallas (Benalcázar & Manzano, 2016)

La figura 3-12 muestra la pantalla de vista de datos, donde el usuario podrá ver en una tabla los datos que él requiera, así como imprimir o exportar.

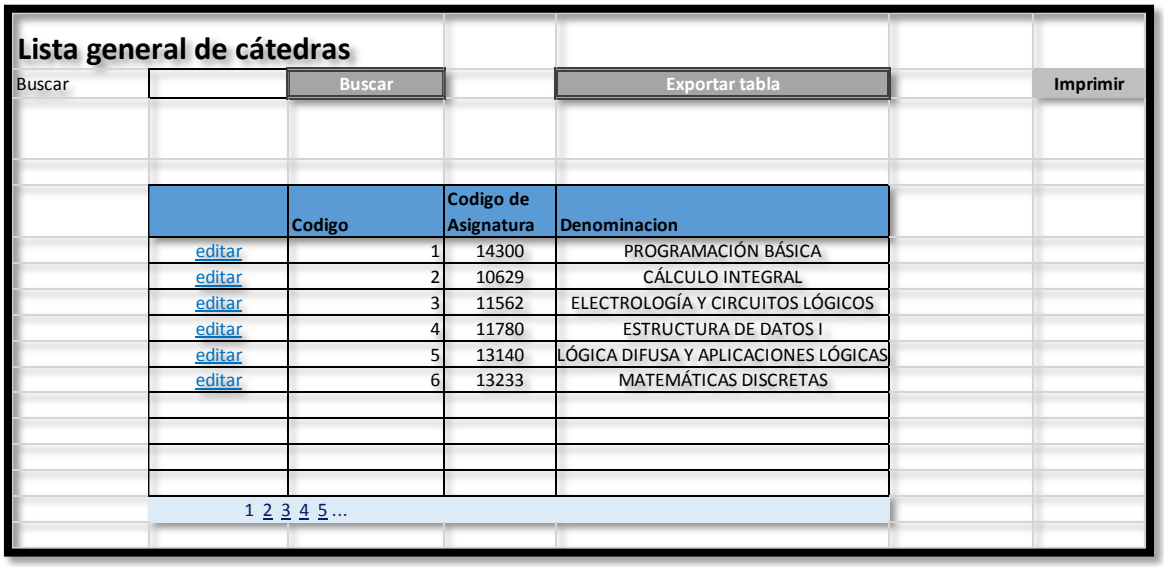

Figura 3-12 Pantalla vista de datos (Benalcázar & Manzano, 2016)

Si se quiere observar el desarrollo de todas las pantallas por favor mirar el Anexo6.

3.3 Implementación

3.3.1 Diagrama de componentes

La figura 3-13 muestra el diagrama de componentes del sistema, se puede observar que el sistema está compuesto por cinco componentes que son: Acceso a datos, datos, lógica de negocio, servicioDSS, presentación.

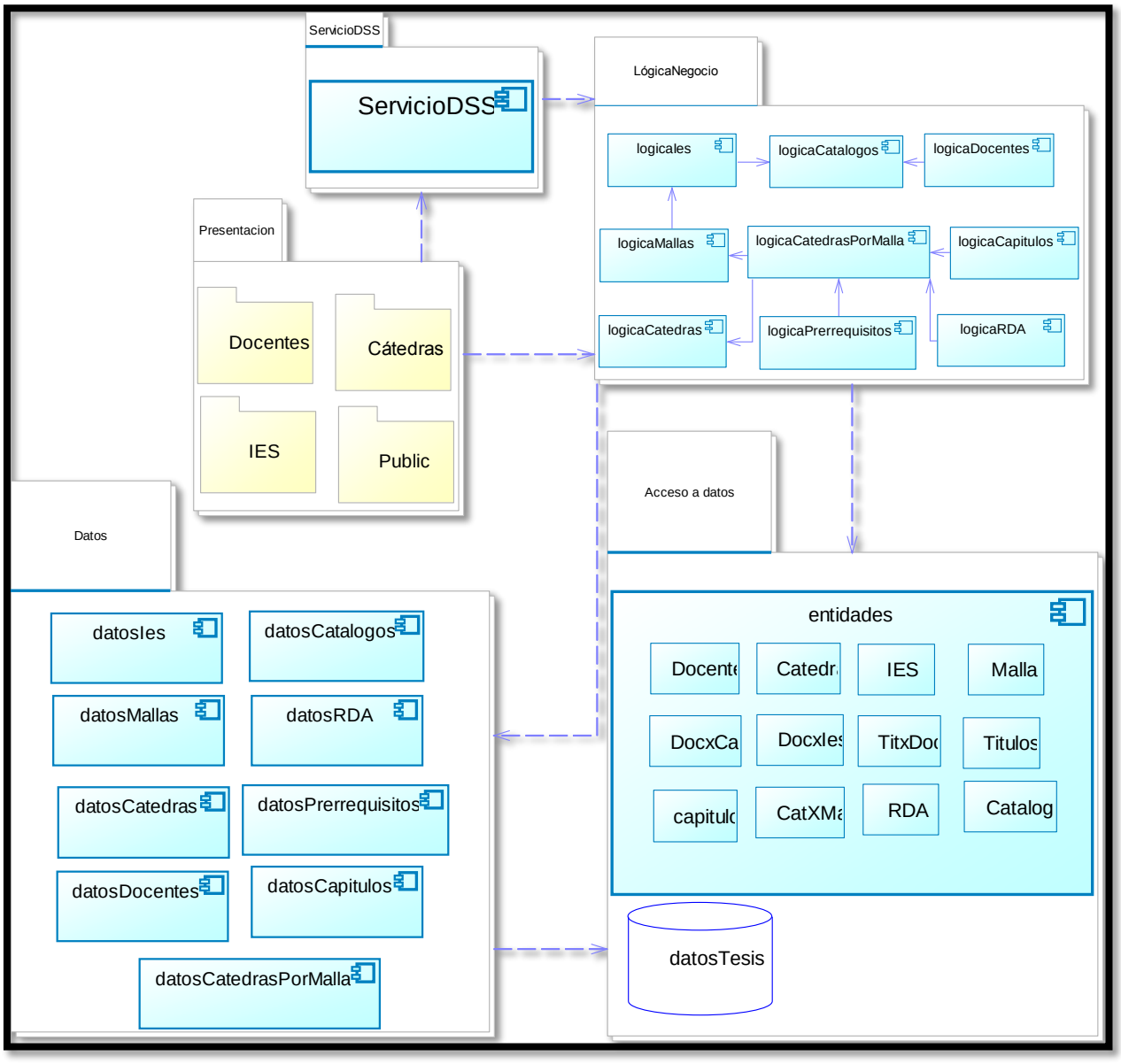

Figura 3-13 Diagrama de componentes (Benalcázar & Manzano, 2016)

3.3.2 Diagrama de despliegue

La figura 3-14 muestra el diagrama de despliegue de la aplicación, donde se muestra como se despliega el sistema. Se puede observar que disponemos de tres nodos el servicio de negocio donde se encuentra toda la programación del sistema, el servicio web donde se han publicado todos los métodos del sistema y la aplicación web que consume al servicio.

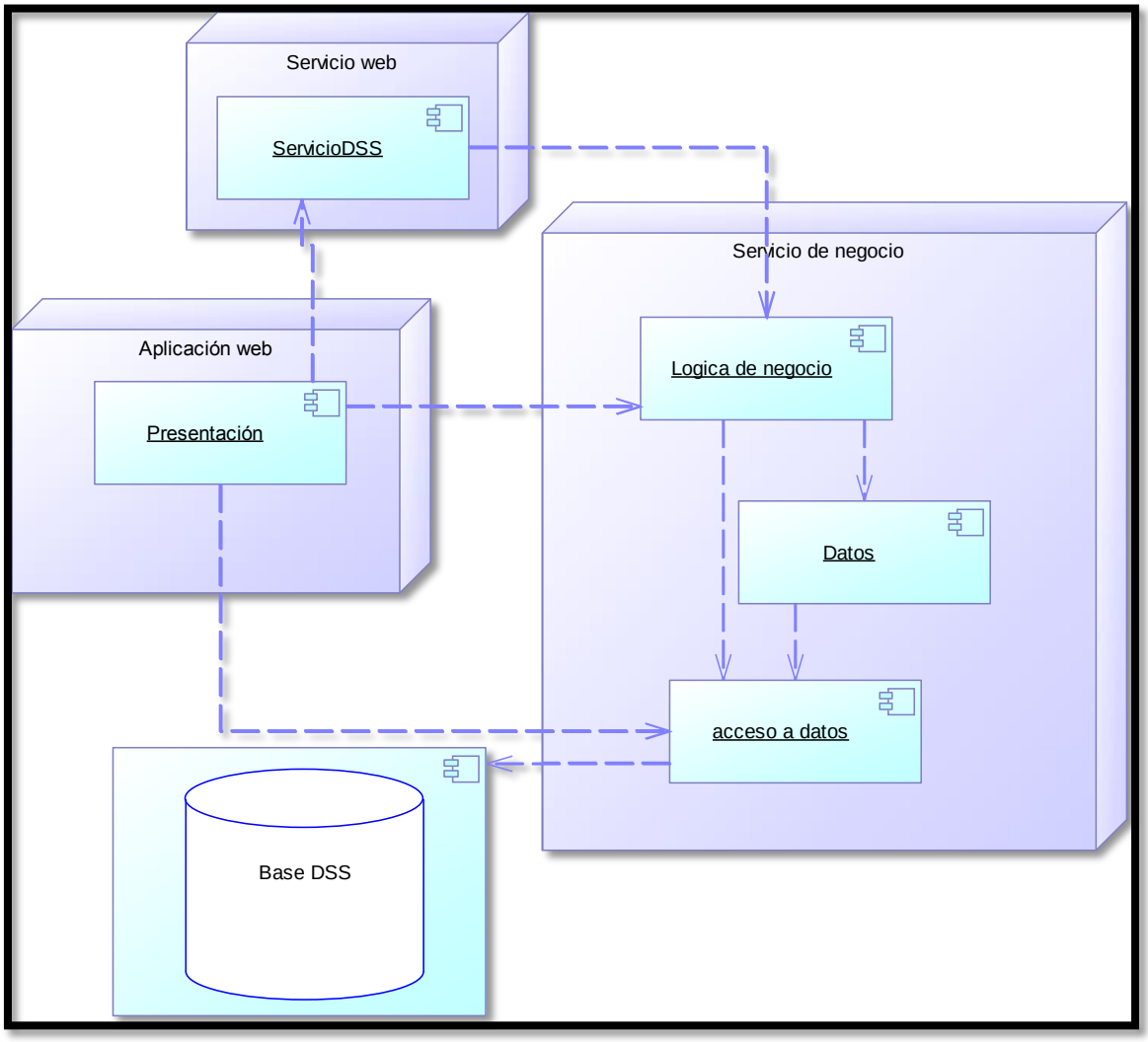

Figura 3-14 Diagrama de despliegue (Benalcázar & Manzano, 2016)

3.3.3 Codificación (Estándares de codificación)

Los estándares de codificación utilizada para el desarrollo del sistema se basaran en los siguientes lineamientos detallados a continuación.

Reglas de nombrado

- Se usara la primera letra de cada palabra en mayúsculas y el resto en minúsculas para nombrar nuevas clases o páginas.
- Se usara la primera letra de cada palabra en mayúsculas y el resto en minúsculas, tomando en cuenta que la primera palabra siempre será minúscula, esto se usara para nombrar atributos de clases, variables de distintos tipos, el nombre de la variable debe tener sentido con lo que se está programando.
- Evitar el uso de abreviaturas para nombrar variables o nuevas clases.
- No usar guiones de ningún tipo para nombrar variables o clases.
- Los métodos o eventos se nombraran en forma verbal es decir que indiquen acción.
- Para nombrar propiedades se nombraran como sustantivos
- Evitar el uso de comentarios
- Usar nombres genéricos, por ejemplo el método insertar puede existir en todas las clases.

Reglas de nombrado de Controles web

Para nombrar controles web se utilizara los siguientes prefijos antes de cada control, estos están resumidos en la tabla 3-3:

Prefijo	Control	Ejemplo
lbl	Label	lblNombre
Btn	Button	btnNombre
Drp	dropDownList	drpNombre
txt	TextBox	txtNombre
tabla	GridView	tablaNombre

Tabla 3-3 Reglas de nombrado de controles web (Benalcázar & Manzano, 2016)

3.4 Pruebas

3.4.1 Pruebas de caja blanca

Las pruebas de caja blanca del sistema se centraron en comprobar el correcto funcionamiento del código y su estructura lógica, comprobándolas en varias situaciones diferentes, basándose en un nivel bajo y no necesariamente en la especificación de requerimientos dados por los usuarios. Dichas pruebas se realizaron continuamente al momento de implementar el código.

3.4.2 Pruebas caja negra

Las pruebas de caja negra que se realizaron en el sistema, se basaron en comprobar el correcto funcionamiento de cada uno de los requerimientos dados por los usuarios. Se verifico si lo que hacía el sistema, era lo que realmente el usuario requería, también se comprobó que todas las excepciones fueran manejadas correctamente al presentarlas al usuario y que la base de datos este acorde de los requerimientos de los usuarios. Dichas pruebas se realizaron basando en los casos de uso obtenidos de los requerimientos de usuario

3.5 Prototipo

Se generó un prototipo funcional, a continuación se presentan las pantallas del sistema:

3.5.1 Inicio de sesión

La figura 3-15 muestra como se ve el inicio de sesión en el sistema, el usuario tendrá que ingresar su usuario y contraseña para poder acceder al sistema.

Figura 3-15 Prototipo-Inicio de sesión (Benalcázar & Manzano, 2016)

3.5.2 Pantalla de inicio

La figura 3-16 nos muestra como se muestra la pantalla de inicio, esta pantalla se muestra al usuario una vez que ingresa al sistema. En la figura podemos observar los botones de los 4 módulos del sistema, pero los botones solo son visibles según el rol del usuario.

Figura 3-16 Prototipo – Inicio (Benalcázar & Manzano, 2016)

3.5.3 Estructura del prototipo

La figura 3-17 nos muestra que nuestra aplicación dispone de una cabecera igual en todas las pantallas de la aplicación. Esta cabecera tiene en la parte superior el nombre del sistema, y en la parte inferior se observa el nombre del módulo donde el usuario se encuentra, en la parte derecha superior dispone un botón para regresar al inicio y debajo de este se muestra el nombre de usuario y a lado la opción de cerrar sesión.

Figura 3-17 Prototipo – Cabecera (Benalcázar & Manzano, 2016)

En la figura 3-18 se puede observar un ejemplo la estructura de las pantallas de la aplicación, en la parte superior derecha se puede ver el nombre del módulo donde el usuario ingreso, en este caso Cátedras, en la parte derecha podemos ver las opciones que nos permite hacer este módulo, y por último en la parte derecha abajo del nombre del módulo, es donde se muestran todas las opciones según el usuario escoja.

Figura 3-18 Prototipo – Estructura ejemplo (Benalcázar & Manzano, 2016)

4 Capítulo 4: Conclusiones y recomendaciones

4.1 Conclusiones

- ✓ Se ha podido desarrollar un DSS[17] capaz de generar reportes estáticos y dinámicos para dar soporte y ayuda a la toma de decisiones de la carrera de ingeniería en sistemas
- ✓ Gracias al uso de c# como lenguaje de programación, se ha podido obtener los beneficios de la programación orientada a objetos en un ambiente de desarrollo web.
- ✓ El uso de visual studio 2012 como plataforma optimizo el tiempo de codificación, ya que ayudo en la búsqueda de errores tanto en tiempo de ejecución como de compilación.
- ✓ El uso de ADO.NET para el uso de la base de datos, simplifico el proceso de conexión con la base de datos y a la vez la codificación, ya que ayudo a la creación de la capa acceso a datos.
- ✓ El uso de ingeniería de software, principalmente el análisis de requerimientos es una base fundamental para el desarrollo de cualquier aplicación.
- ✓ Las metodologías agiles, son herramientas que ayudan a simplificar el proceso de documentación del desarrollo de software.
- ✓ Usar un estilo arquitectural en n-capas ayudo a que el desarrollo del sistema obtenga un mayor rendimiento, fácil manejo de errores y un manejo eficiente del código.
- ✓ El usar LINQ[18] como lenguaje para realizar las consultas a la base de datos, ayudo de manera considerable a optimizar tiempo ya que se podían detectar errores en tiempo de ejecución.
- ✓ El uso de roles dentro de la aplicación, hace que la seguridad sea más optima
- ✓ Debido a las nuevas leyes de educación superior, la carrera de Ingeniería en Sistemas y Computación de la Pontificia Universidad Católica del Ecuador sigue innovando en sus estrategias.
- ✓ La carrera de Ingeniería en sistemas y computación de la Pontificia Universidad Católica del Ecuador está enfocada en mantener su acreditación, tanto a nivel nacional como internacional.
- ✓ El Consejo de Educación Superior y el Consejo de Evaluación, Acreditación y Aseguramiento de la Calidad de la Educación Superior son organismos públicos que trabajan conjuntamente para asegurar la calidad de la educación superior.
- ✓ Un DSS puede ser aplicado no solo en áreas orientadas a la educación si no también áreas industriales, gubernamentales entre otras.

[17] Decision Support System o Sistema de ayuda a la toma de decisiones

[18] Language Integrated Query

4.2 Recomendaciones

- Se recomienda a la Carrera de Ingeniería en Sistemas y Computación de la PUCE,[19] seguir formando profesionales con conocimientos altos en Ingeniería de Software ya que es una base fundamental para la vida profesional.
- Se recomienda seguir fomentando el desarrollo de nuevos DSS para otras áreas gerenciales de la carrera, una de ellas podría ser para ayudar en las decisiones al momento de crear nuevos horarios.
- Se recomienda seguir ampliando este aplicativo con más funciones, como por ejemplo aumentar un modulo estudiantes.
- Para el correcto funcionamiento del DSS es necesario adquirir información real y clara, ya que con información no concisa y errónea no es posible llevar a cabo una buena toma de decisiones
- Se recomienda hacer una recolección de datos real, para que toda la información ingresada al sistema sea válida y ayude a la toma de decisiones.
- Se recomienda reuniones periódicas con los clientes para poder realizar una continua revisión de los sistemas y así poder detectar errores a tiempo.
- Se recomienda realizar un buen análisis de requerimientos y documentarlos para que luego no exista mal entendidos con los clientes.
- Debido a los continuos cambios de las leyes de la educación superior, se recomienda tener los datos del sistema actualizados, para que cuando se haga uso de este tengamos una ayuda más certera para tomar decisiones.
- Debido a las nuevas leyes de educación superior, se recomienda que la carrera de Ingeniería en Sistemas y Computación de la Pontificia Universidad Católica del Ecuador siga preparándose para nuevos cambios

[19] Pontificia Universidad Católica del Ecuador

5 Bibliografía

Alegsa, L. (12 de Mayo de 2010). *Definición de Sistema transaccional (sistema de procesamiento de transacciones)*. Obtenido de Alegsa: http://www.alegsa.com.ar/Dic/sistema%20transaccional.php

Benalcázar, A. E., & Manzano, P. G. (14 de Julio de 2016). *Disertación Desarollo de un DSS de la carrera de Ingeniería en Sistemas.* Quito, Chimborazo, Ecuador: PUCE.

Canós, J., Letelier, P., & Penadés, C. (s.f.). *Universidad Politécnica de Valencia.* Obtenido de http://www.yises.com/pdfs/agil/TodoAgil.pdf

CEAACES. (2014). Recuperado el 02 de Septiembre de 2015, de http://www.ceaaces.gob.ec/

CES. (12 de Julio de 2012). Recuperado el 02 de Septiembre de 2015, de Consejo de Educación Superior: http://www.ces.gob.ec/institucion/mision-vision-y-objetivos

De La Torre LLorente, C., Zorilla Castro, U., Ramos Barros , M. A., & Calvarro Nelson, X. (2010). *Guía de Arquitectura N-Capas orientada al dominio con .NET* . España: Microsoft Ibérica S.R.I.

EL framework de coco. (05 de Diciembre de 2009). Recuperado el 06 de Julio de 2015, de http://kartones.net/blogs/coco/la-capa-de-servicios-conceptos-b-225-sicos.aspx

Gallegos, M. (2015). *Estudio de la tecnología .Net para el desarrollo de aplicaciones e implementación de servicios Web XML.* Repositorio Digital. Recuperado el 17 de 08 de 2015, de http://repositorio.utn.edu.ec/handle/123456789/1116

García Bautista, J. L. (24 de Septiembre de 2014). *Parvulos.net*. Recuperado el 06 de Julio de 2015, de http://joseluisgarciab.blogspot.com/2014/09/programacion-en-3-capas.html

Herrarte, P. (2007). *Programación con Transact SQL*. Recuperado el 17 de Agosto de 2015, de http://www.devjoker.com/gru/Tutorial-Transact-SQL/TSQL/Tutorial-Transact-SQL.aspx

Isuba. (21 de Enero de 2014). *Sistemas UBA*. Recuperado el 26 de Junio de 2015, de http://isuba.blogspot.es/1390301965/arquitectura-de-los-dss/

Jiménez, D. (s.f.). *jordai.com.* Obtenido de http://jordai.com/master/fundamentos/2-NET_Framework.pdf

Ley Orgánica de Educación Superior. (12 de 10 de 2010). Obtenido de CES: http://www.ces.gob.ec/doc/gaceta_ces/loes/loes.pdf

LOS SISTEMAS DE INFORMACIÓN ESTRATEGICOS. (20 de Octubre de 2012). Obtenido de Gerenciasg.blogspot: http://gerenciasig.blogspot.com/

Luján, S. (s.f.). *Programación en internet: Clientes web.* España: Club Universitario.

Malla curricular. (Junio de 2015). Obtenido de PUCE: http://www.puce.edu.ec/documentos/mallas-curriculares/vigentes/PUCE-ING-IngenierIa-Sistemas.pdf

Manzano, P. (2016). Entorno del software.

Manzano, P. (2016). Estándares de codificación.

Medina, Z. (24 de Mayo de 2014). *Sistemas de Información Gerencial*. Recuperado el 26 de Junio de 2015, de http://g6-sad-unesr-20141.blogspot.com/2014/05/objetivos-del-dss.html

Microsoft. (2007). Recuperado el 16 de Julio de 2015, de https://msdn.microsoft.com/es-es/library/t745kdsh(v=vs.100).aspx

Microsofth. (s.f.). Recuperado el 17 de 08 de 2015, de https://msdn.microsoft.com/es-es/library/bb397897.aspx

Peralta, M. (2008). *Sistema de Información*. Recuperado el 20 de Junio de 2015, de http://www.ilustrados.com/tema/3351/Sistema-Informacion.html

Programacion extrema. (s.f.). Obtenido de Ecured: http://www.ecured.cu/Programacion_extrema

Puceing. (s.f.). Recuperado el 01 de 09 de 2015, de http://www.puceing.edu.ec/ingenieria/index.php/la-facultad/ingenieria-en-sistemas

Ramonmorillos Weblog. (30 de Julio de 2012). Obtenido de SQL Server 2012 Conceptos básicos: Las Herramientas: https://ramonmorillo.wordpress.com/2012/07/31/sql-server-2012-conceptos-basicos-las-herramientas/

Reglamento de régimen académico codificado. (2013). Obtenido de CES: http://www.ces.gob.ec/doc/Reglamentos/reglamentos2016/abril/codificado%20del%20reglamento%20de%20rgimen%20acadmico.pdf

Ruiz, S. (s.f.). *Wikia*. Obtenido de http://es.sandramarramirez.wikia.com/wiki/Programaci%C3%B3n_por_Capas

Scrum Manager. (27 de Abril de 2014). Recuperado el 2015 de Junio de 2015, de http://www.scrummanager.net/bok/index.php?title=El_manifiesto_%C3%A1gil

SISTEMA DE APOYO A LA TOMA DE DECISIONES. (s.f.). Obtenido de Bligoo: http://sistemasdeinformacion.bligoo.com.mx/sistema-de-apoyo-a-la-toma-de-decisiones#.Vw7_x_nhBD8

Sistemas de Soporte a la toma de Decisiones. (Junio de 2012). Recuperado el 26 de Junio de 2015, de https://sites.google.com/a/usb.ve/sistemas-de-soporte-a-decisiones/sistemas-de-informacion/inspiration

Sotomayor, B. (2002). *The university of Chicago.* Obtenido de La plataforma .NET:: http://people.cs.uchicago.edu/~borja/pubs/revistaeside2002.pdf

Unidad de Desarollo Tecnológico en Inteligencia Artificial. (s.f.). Recuperado el 26 de Junio de 2015, de http://www.iiia.csic.es/udt/es/artificialintelligence/sistemas-soporte-decisiones

Universidad Simón Bolívar. (Abril de 2009). Obtenido de Ingeniería de software: http://ldc.usb.ve/~abianc/materias/ci4713/metodologiasagiles.pdf

Universidad Tecnológica Nacional. (s.f.). Obtenido de Ingeniería Industrial: http://www.frlp.utn.edu.ar/materias/info2/SI-Sistemas%20de%20Informacion.pdf

Vázquez, A. S. (s.f.). PROPUESTA DE BASES PARA EL DISEÑO DE UN SISTEMA DE GESTIÓN ESTRATÉGICA DE INFORMACIÓN PARA LA DIRECCIÓN DE ENERGÍA RENOVABLE DEL MINBAS. *Tesis en opción al título de Máster en Gerencia de la Ciencia*. La Habana, Cuba: 2011.

Wikipedia. (19 de Mayo de 2015). Recuperado el 27 de Junio de 2015, de https://es.wikipedia.org/wiki/Manifiesto_%C3%A1gil#cite_note-1

6 Anexos

- Anexo 1. Malla Curricular
- Anexo 2. Ley orgánica de educación superior
- Anexo 3. Codificación del régimen académico modificado
- Anexo 4. Historias de usuario
- Anexo 5. Historias de usuario con la aprobación del cliente
- Anexo 6. Pantallas de requerimientos
- Anexo 7. Versiones del diseño de la base de datos
- Anexo 8. Diagrama de clases
- Anexo 9. Modelo conceptual de la base de datos
- Anexo 10. Modelo físico de la base de datos
- Anexo 11. Diccionario de datos
- Anexo 12. Manual de Usuario