Sistema para la recomendación de costos en proyectos de desarrollo de software

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor:

Sandra Jerónimo Casal

Tutores:

MSc. Delly Lien González Hernández

MSc. Hubert Viltres Sala

Ing. Wendy Rodriguez Muñoz

La Habana, junio de 2018

"Año 60 de la Revolución"

Declaración de autoría

Declaro por este medio que yo Sandra Jerónimo Casal, con carnet de identidad 95071428935, soy el autor principal del trabajo titulado **"Sistema para la recomendación de costos en proyectos de desarrollo de software"** y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Declaro que todo lo anteriormente expuesto se ajusta a la verdad, y asumo la responsabilidad moral y jurídica que se derive de este juramento profesional.

Y para que así conste, firmo la presente declaración de autoría en La Habana a los ___ días del mes de _____ del año 2018.

_____

Firma del autor
Sandra Jerónimo Casal

_____

Firma del tutor
MSc. Delly Lien González Hernández

_____

Firma del tutor
MSc. Hubert Viltres Sala

_____

Firma del tutor
Ing. Wendy Rodriguez Muñoz

Resumen

La presente investigación propone un sistema de recomendación de costos que permita optimizar el proceso de estimación de costos de los proyectos de desarrollo de software en la Universidad de la Ciencias Informáticas. Durante la investigación se analizaron los sistemas de recomendaciones y las herramientas y técnicas para la estimación de los costos. El estudio del estado del arte permitió identificar el algoritmo de recomendación, las funcionalidades y las tecnologías necesarias para implementar la solución propuesta. La solución es un sistema de recomendación basado en conocimiento, que utiliza la técnica de razonamiento basado en casos. La propuesta de solución estuvo guiada por la metodología AUP-UCI y se seleccionó como principales tecnologías el marco de trabajo *Django*, *Python* como lenguaje de programación, el entorno integrado de desarrollo *PyCharm*, el servidor web *Apache*, el Sistema Gestor de Base de Datos *MySQL* y *Visual Paradigm* como herramienta para el modelado. La estrategia de prueba aplicada permitió verificar la seguridad, la calidad y el correcto funcionamiento del sistema. La consulta con expertos mediante el método *Delphi* validó que la propuesta de solución contribuye a mejorar y facilitar la estimación de los costos de los proyectos de software.

Palabras clave: algoritmos, costos, proyectos de software, sistema de recomendación.

Índice

Introducción

El éxito de un proyecto de software depende del personal calificado, los objetivos claros y bien definidos, de una adecuada metodología y de la correcta gestión de la información. En el desarrollo de un proyecto de software se realizan diversas actividades, el levantamiento de requisitos, la gestión de los costos, la implementación de la aplicación. La gestión de los costos permite conocer por adelantado los gastos de un proyecto mediante los procesos de estimación, asignación y control de los costos. El proceso de estimación de costos se enfoca en desarrollar una aproximación de los recursos financieros necesarios para completar las actividades del proyecto software (PMBOOK, 2004). Este proceso emplea diversos métodos, técnicas y herramientas que contribuyen a realizar una estimación más rápida y acertada. Las técnicas empleadas son el juicio de expertos, la estimación análoga, paramétrica, ascendente o por tres valores (Rodríguez y Rojas, 2015).

Las herramientas para la estimación de costos son los softwares de gestión de proyectos y las técnicas grupales de toma de decisiones (Rodríguez y Rojas, 2015). En la estimación de costos se analizan variables, las tecnologías y herramientas, los recursos humanos, los recursos materiales y las necesidades del cliente, que permiten realizar la aproximación del presupuesto del proyecto (Guerrero, 2015).

En Cuba, con la implementación de los Lineamientos del Partido Comunista de Cuba (PCC) relacionados con el proceso de la informatización de la sociedad, se potencia la creación y utilización de aplicaciones informáticas en empresas, organizaciones e instituciones (Comité Central del PCC, 2016). Para cumplir con el lineamiento 108 del PCC que plantea "…avanzar gradualmente en el proceso de informatización de la sociedad, el desarrollo de la infraestructura de telecomunicaciones y la industria de aplicaciones y servicios informáticos...", las empresas y los organismos del sector de la Informática y las Comunicaciones desarrollan diferentes proyectos de software. La Universidad de las Ciencias Informáticas (UCI) con el objetivo de impulsar el avance tecnológico del país trabaja en el desarrollo de aplicaciones y la prestación de servicios. En la universidad se determina la factibilidad de un proyecto mediante la gestión de los costos. Esta efectúa un importante cálculo o aproximación de los recursos necesarios para el cumplimiento de sus objetivos.

En la estimación de costos se analizan las actividades del proyecto y los recursos necesarios (recursos humanos, recursos materiales, tecnologías, herramientas, equipos, etc.). En la UCI los centros de producción de software para la estimación de costos emplean las fichas de costo y analizan únicamente las variables tiempo y recursos humanos, desestimando los recursos materiales (ver anexo 1). Esto puede

ocasionar gastos y retrasos en la obtención de los materiales faltantes. En la gestión de costos, al utilizar la ficha de costos no se analiza la categoría científica- técnica del personal a cargo, factor que puede repercutir en la calidad y en el tiempo de realización de una actividad. El proceso de estimación de costos se realiza de forma manual, esto genera retraso en la aproximación de los costos, gasto de recursos y la posibilidad de errores de cálculos que pueden influir en los costos.

Por lo anteriormente planteado y la necesidad de darle solución a esta problemática, se define como **problema de investigación:** ¿Cómo contribuir a la gestión de costos en los proyectos de desarrollo de software de la Universidad de la Ciencias Informáticas?

El **objeto de estudio** está centrado en la recomendación de costos en los proyectos de desarrollo web.

Se plantea como **objetivo general:** Desarrollar un sistema de recomendación de los costos asociados a proyectos de desarrollo de software.

Se selecciona como **campo de acción:** El proceso de recomendación de costos en proyectos de desarrollo de software en la Universidad de la Ciencias Informáticas.

Por consiguiente, los **objetivos específicos** son los siguientes:

1. Construir los referentes teóricos fundamentales que sustentan la investigación relacionados con el desarrollo del sistema de recomendación de costos.
2. Diagnosticar el estado de la gestión de costos en proyectos de desarrollo de software.
3. Diseñar las funcionalidades del sistema de recomendación de costos en proyectos de desarrollo de software.
4. Implementar las funcionalidades del sistema de recomendación de costos en proyectos de desarrollo de software.
5. Validar las funcionalidades del sistema de recomendación de costos en proyectos de desarrollo de software.

Se plantea como **hipótesis**: El desarrollo de un sistema para la recomendación de costos permitirá mejorar la estimación de los costos de los proyectos de software.

Tareas de investigación:

1. Realización de un diagnóstico sobre la gestión de costos en los proyectos de desarrollo de software.
2. Realización de un estudio sobre las tendencias en actuales en el desarrollo de sistemas de recomendación.
3. Selección de las tecnologías, herramientas y estándares que se necesitan para implementar la propuesta de solución.
4. Definición de los requisitos funcionales y no funcionales de la propuesta de solución.
5. Definición de los requisitos funcionales y no funcionales de la propuesta de solución.
6. Diseño de la propuesta de solución.
7. Implementación de la propuesta de solución.
8. Documentación de las pruebas realizadas a la propuesta de solución (carga y estrés, aceptación, usabilidad y seguridad)

Para cumplir con las tareas antes mencionadas se hace necesaria la utilización de los siguientes **métodos de investigación:**

Métodos teóricos:

Análisis Histórico-Lógico: utilizado para realizar un estudio de cómo se realiza la gestión de costo y las herramientas que se emplean para así posibilitar la creación del marco teórico.

Analítico-Sintético: para el estudio de las fuentes bibliográficas y conceptos existentes sobre la gestión de costos y los sistemas de recomendaciones, con el objetivo de poder demostrar la necesidad de la investigación.

Modelación: para la representación de la solución propuesta a través de la modelación de diagramas que permiten modelar el proceso a seguir para desarrollar la solución creando una abstracción con el objetivo de explicar la realidad.

Métodos Empíricos:

Observación: Se emplea en la adquisición de nuevas ideas con el objetivo de establecer métricas sobre la realización del diseño web para el sistema de recomendación de costo en proyectos de desarrollo software.

Entrevista: Permite identificar las principales funcionalidades a considerar en el desarrollo de la propuesta de solución, así como los contenidos a mostrar.

Estructura del documento.

Capítulo 1: Sistema de recomendación de costos en proyectos de desarrollo de software

Se realiza un análisis de los elementos teóricos que soportan la investigación. Se presenta un estudio asociado a las principales tendencias en la gestión de costos y al empleo de sistemas de recomendación como herramientas de apoyo al proceso de toma de decisiones en las empresas; haciendo especial énfasis en los algoritmos empleados por estos sistemas. Además, se realiza un análisis de las herramientas, tecnologías y metodología para el desarrollo de software.

Capítulo 2: Análisis y diseño del Sistema de recomendación de costos asociado a proyectos de desarrollo de software

Se realiza el análisis y diseño de la propuesta de solución mediante la definición de los requisitos funcionales y no funcionales que debe cumplir el sistema. Se describen a grandes rasgos las características de la solución propuesta a través de los más diversos artefactos que especifica la metodología de desarrollo empleada.

Capítulo 3: Implementación y validación del Sistema de recomendación de costos

Se realiza la implementación del Sistema de recomendación de costo asociado al proceso de desarrollo de software y se especifican las pruebas realizadas al mismo para validar el correcto funcionamiento de la propuesta de solución.

Capítulo 1: Sistema de recomendación de costos en proyectos de desarrollo de software

En el presente capítulo se exponen los fundamentos teóricos asociados al objeto de estudio y al campo de acción. Se realiza un estudio de los principales sistemas de recomendación, se analizan sus características, ventajas y desventajas, se determina el algoritmo de recomendación. Se analiza y se selecciona la metodología, tecnologías y herramientas que sean factibles para la implementación del sistema.

1.1 Gestión de la información

La información constituye un recurso estratégico que interviene en los procesos cognitivos e informacionales en la toma de decisiones de las organizaciones, empresas e instituciones. La toma de decisiones permite identificar la mejor decisión y curso de acción para explotar una oportunidad o solucionar de forma efectiva un problema. Para la correcta gestión de la información es necesario realizar un correcto procesamiento, organización, almacenamiento y recuperación de la información (Rodríguez, 2015).

El procesamiento de la información permite percibir, crear conocimiento y elegir la mejor alternativa de decisión. El objetivo principal de la Recuperación de Información según plantean Bender y Deco (2014) es satisfacer la necesidad de información de un usuario mediante una consulta en lenguaje natural especificada a través de un conjunto de palabras claves, también llamadas descriptores. El procesamiento y la recuperación de la información son elementos relevantes en la toma de decisiones que inciden en seleccionar la decisión y acción adecuada para resolver una situación determinada.

En el desarrollo de un proyecto de software se realizan varias actividades, procesar y recopilar la información para determinar los requisitos, analizar las etapas y gestionar los costos. La gestión de los costos permite determinar la factibilidad del proyecto mediante la estimación de los gastos por cada etapa, aspectos que influyen en el éxito de un proyecto de software. En la estimación de los costos se consideran diversos factores como los riegos, las características, las necesidades del cliente o los costos de proyectos similares. Además, para la adecuada estimación de los costos es necesario gestionar toda la información disponible y realizar una correcta elección de las herramientas o técnicas en la estimación de costos a emplear.

1.2 Herramientas y técnicas empleadas en la gestión de costos

La gestión de los costos del proyecto efectúa el coste de los recursos necesarios para completar las actividades para desarrollar un software. En la gestión de los costos se involucran los procesos de planificación, estimación y control de los costos. En el proceso de planificación se identifican las

herramientas y técnicas para realizar la estimación de los costos. En la estimación de los costos se utiliza diferentes técnicas, el juicio de expertos, la estimación análoga, paramétrica, ascendente o por tres valores. Las herramientas para la estimación de los costos son los softwares de gestión de proyectos, las fichas de costo y las técnicas grupales de toma de decisiones.

1.2.1 Juicio de expertos

Guiado por la información histórica, el juicio de expertos aporta una perspectiva valiosa sobre el ambiente y la información procedentes de proyecto similares anteriores. El juicio de expertos también puede utilizarse para determinar si es conveniente combinar métodos de estimación y cómo conciliar las diferencias entre ellos (PMBOOK, 2004). Este método consiste en reunir un grupo de expertos para realizar la estimación de los costos según su experiencia y conocimiento (Rodríguez, 2015).

1.2.2 Estimación análoga

Estimación *Top-down* o análoga se emplea para estimar un parámetro cuando la información detallada sobre el proyecto es limitada. Esta técnica utiliza el costo real de proyectos similares anteriores como base para estimar el costo del proyecto actual (Guerrero, 2015). Para realizar esta técnica el estimador utiliza de la base histórica organizacional las características técnicas y los costos de proyectos similares y con esta información realiza una comparación y se modifica el proyecto actual (PMBOOK, 2004; Rodríguez, 2015).

1.2.3 Estimación paramétrica

La estimación paramétrica es una técnica que utiliza una relación estadística entre los datos históricos y otras variables para calcular una estimación de costos de un recurso de la actividad del cronograma. Esta técnica puede producir niveles superiores de exactitud según la complejidad, la cantidad subyacente de recursos y la información de costos incorporada al modelo (PMBOOK, 2004).

Para calcular los parámetros de costos, se utilizan las Relaciones de Estimación de Costos (*CERs – Cost Estimating Relationships*), que convierten los costos en variables dependientes de otras variables y que se obtienen a partir de una base empírica y las variables independientes características del diseño o prestaciones del sistema. Estas *CERs* son a menudo métodos de estimación de costos que convergen en funciones ajustadas matemáticamente que describen el coste de una estructura, sistema, o servicio en función de una o más variables independientes.

1.2.4 Estimación ascendente

La estimación ascendente consiste en estimar el costo de paquetes de trabajo individuales o actividades del cronograma individuales con el mayor nivel de detalle. Este costo detallado luego se resume o "acumula" en niveles superiores para fines de información y seguimiento (PMBOOK, 2004; Guerrero, 2015).

El costeo basado en actividades (ABC), es una técnica del tipo estimación ascendente, no obstante, es genérica y aplica no solo para el costo de proyectos, sin para calcular los costos en que se incurre por las actividades de manufactura para un producto (Guerrero, 2015).

1.2.5 Estimación de tres valores

Según Guerrero(2015) el proceso estimar la duración de las actividades, el PERT (Técnica de Revisión y Evaluación de Programas) utiliza tres escenarios para definir un rango aproximado del coste de una actividad y se considera la incertidumbre y el riesgo:

- Más probable (cM): El coste de la actividad se estima sobre la base de una evaluación realista del esfuerzo necesario para el trabajo requerido y de cualquier gasto previsto (Aguilera y Franco,2015).
- Optimista (cO) El coste de la actividad se estima sobre la base del análisis del mejor escenario para esa actividad (Aguilera y Franco,2015).
- Pesimista (cP): El coste de la actividad se estima sobre la base del análisis del peor escenario para esa actividad (Aguilera y Franco,2015).
- Valor Esperado(cE): $cE = (cO + 4cM + cP)/6$ (1)

1.2.6 Técnicas grupales de la toma de decisiones

La toma de decisiones es un conjunto de operaciones que se realizan al detectar una situación (Leyva, 2013). Una de las técnicas grupales para la toma de decisiones es la tormenta de ideas (lluvia de ideas o *brainstorming*). Esta técnica de pensamiento creativo es utilizada para estimular la producción de un elevado número de ideas, por parte de un grupo, acerca de un problema y de sus soluciones o, en general, sobre un tema que requiere de ideas originales (Aiteco.com). La lluvia de ideas permite (Gestiopolis.com, 2017):

- Plantear y resolver los problemas existentes

- Plantear posibles causas
- Plantear soluciones alternativas
- Desarrollar la creatividad
- Discutir conceptos nuevos
- Superar el conformismo y la monotonía

Luego de estudiar las técnicas de estimación de costos se determinaron las siguientes dificultades (ver tabla 1):

Tabla 1. Dificultades de técnicas y herramientas de estimación de costos.

Técnica	Deficiencias
Juicio de Expertos	Requiere de personas con experiencia en el área de la gestión de los costos
Técnica grupal para la toma de decisiones: Tormenta de Ideas	Requiere de personas con experiencia en el área de la gestión de los costos
La estimación análoga	Necesita de bases de datos robustas que contengan la información histórica de proyectos similares
La estimación paramétrica	Requiere de una base de datos amplia y actualizada que contengan los datos históricos de otros proyectos
La estimación por tres valores	Necesita de un estudio de analogía y de colecciones de datos históricos de otros proyectos
La estimación ascendente	Solo calcula el costo de las actividades que se desarrollan en el proyecto, desestimando el consumo de materiales y los recursos humanos

Después de analizar las deficiencias planteadas se determina que las técnicas estimación de costo necesitan de especialistas con experiencia en ciencias empresariales o de sistemas que almacenen toda la información de los proyectos de software.

1.2.7 Software de gestión de costo

El software de gestión de proyectos, las aplicaciones de software de estimación de costos y las hojas de cálculo computarizadas, se emplean para asistir en el proceso de estimación de costos. Estas herramientas pueden simplificar la utilización de algunas de las técnicas de estimación de costos y facilitar la consideración rápida de las diversas alternativas de estimación de costos (PMBOOK, 2004). Algunos de los softwares utilizados para asistir en el proceso de estimación de costos son gesPro, OpenProj y XEDRO GESPRO.

El programa **gesPro** de **Softpoint** es un sistema *web* apoyado en herramientas de software libre que permite organizar y manejar de forma eficiente el ciclo de vida de proyectos de cualquier tipo de organización o empresa. Este sistema permite organizar la información en módulos como proyectos, personas, organizaciones, case management, facturación y cuentas corrientes, pagos, recursos humanos, base de conocimientos, departamento editorial, eventos, etcétera (gesPro.softpoint.com, 2017).

OpenProj, es una aplicación de código libre que compite contra Microsoft Project. Ofrece la posibilidad de trabajar en proyectos de una forma sencilla y fácil. Está disponible para Sistemas Operativos: Windows, Linux, Unix, Mac, entre otros**.** Sus funciones principales, se concentran en el diseño de Gantt, diagramas de redes PERT, y muchas otras gráficas más. También permite hacer un seguimiento de las distintas etapas de un proyecto: presupuesto, definición, estimación, etc. Cuando un proyecto se termina, existe un histórico que refleja cada uno de los anteriores puntos (Lemus y Muñoz, 2009).

XEDRO GESPRO es una Suite orientada a la web que permite la planificación, seguimiento y control de productos en forma de proyectos. Este software tiene herramientas para el apoyo a la toma de decisiones a nivel de proyecto, nivel de entidad ejecutora y nivel gerencial. Se presenta en un modelo de negocios basado en servicios. El modelo de negocio es basado en servicios y el precio del producto varía de acuerdo a los servicios definidos como consultorías sobre gestión de proyectos. Actualmente se utiliza en la Universidad de las Ciencias Informáticas (UCI) (Gespro.uci.cu, 2017).

1.2.8 Ficha de costo

A partir de las encuestas y entrevistas realizadas a los asesores de mercadotecnia en los centros de desarrollo de la Universidad de las Ciencias Informáticas (UCI) se puede definir que estos emplean las fichas de costo para la estimación de los costos de los proyectos de software. La ficha de costo es el

documento donde se refleja la información relacionada con los componentes del costo unitario de la producción o el servicio. El costo unitario constituye un indicador económico de vital importancia en el análisis de los resultados obtenidos, que muestra la efectividad alcanzada en el proceso y la eficiencia en la utilización de los recursos. Las fichas de costo se pueden clasificar en atención al momento de confección de la misma, en función al criterio de los especialistas y a los fines que se persiguen en (GestioPolis.com, 2017):

- Ficha de costo planificada: representa la magnitud máxima de los gastos esperados en la producción de una unidad de producto. Se confeccionan utilizando normas y normativas que garanticen la situación óptima posible de la producción para el año que se planifica y se tienen en consideración las variaciones existentes de calidad, medidas y precios de los materiales y calificación de la fuerza de trabajo cuando sea necesario.
- Ficha de costo normativa: se calcula a partir de las normas vigentes en una ficha determinada y caracteriza la situación técnica –organizativa y económica de la producción–. Estas fichas son más dinámicas a diferencia de las planificadas, se modifican según cambian las normas.
- Ficha de costo presupuestada: es una variante de la planificada y se confecciona para aquellos tipos de productos que su producción no es representativa y generalmente se coordina con el consumidor la fundamentación de los cálculos de los gastos. Esta ficha es necesaria para establecer los precios de estos productos.
- Ficha de costo real: caracteriza el costo real de la producción elaborada en el periodo que se informa. Cuando se confecciona esta ficha es necesario considerar que los objetivos de cálculo, la unidad de cálculo y la clasificación de los gastos van a ser iguales la base para la ficha de costo planificada. La ficha de costo real constituye una fuente importante para el análisis económico y es contentiva de los indicadores que deben ser utilizados para la confección del plan de costo.

Luego de analizar las técnicas y herramientas para la estimación de los costos se concluye que son necesarias para lograr calcular el costo de las actividades del proyecto y determinar su factibilidad. Las técnicas y herramientas, para la gestión de costos manipulan grandes volúmenes de información. Para mejora el procesamiento de la información, existen técnicas de recuperación información, una de estas alternativas son los sistemas de recomendación. En bibliografía analizada no se evidencia la existencia de sistemas de recomendaciones para la gestión de costos.

1.3 Sistemas de recomendaciones

Los sistemas de recomendación (SR) son técnicas aplicadas a la recuperación de información para intentar resolver el problema de sobrecarga de datos en Internet. Estos sistemas permiten a los usuarios encontrar propuestas de productos que pueden ser de utilidad o de su interés en base a sus necesidades y dentro de sus preferencias (Núñez, 2012; Caro, 2017). Un Sistema de recomendación (ver figura 1) tiene como principal función establecer las relaciones entre los elementos (productos), los usuarios y sus preferencias; efectuándose en 3 etapas (captura de las preferencias, procesamiento y recomendación de la información) (Viltres, 2014).

Figura 1. Esquema general de un sistema de recomendación (Formoso, 2013)

El avance científico-técnico alcanzado en la actualidad genera un gran volumen de contenido digital, para ello surgen los SR con el objetivo de procesar, clasificar y recuperar la información. Los SR se emplean en diversas áreas, en el comercio electrónico (Amazon, Ebay), en las redes sociales (Facebook, Twitter), en la publicidad computacional, que proporciona recomendaciones de productos a usuarios al realizar búsquedas (Google, Yahoo!), en plataformas que recomiendan libros, películas, videos y música (YouTube, BookLens, Netflix) o que recomiendan hoteles, viajes, ect (Tripadvisor, Trivago) (Caro, 2017).

Los sistemas de recomendación emplean técnicas basadas en la información que pueden obtener del conocimiento de los productos, de las valoraciones de los usuarios, de la interacción del usuario con el SR

o de la interacción del usuario con los productos. Estas técnicas de recomendación determinan la clasificación de los SR (Caro, 2017) en sistemas de recomendación basados en filtrado colaborativo, en sistemas de recomendación basados en contenido, en sistemas de recomendación basados en conocimiento y en sistemas de recomendación híbridos. A continuación, se realiza una descripción de cada SR.

1.3.1 Sistemas de recomendación basado en filtrado colaborativo

Los sistemas de recomendación de filtrado colaborativo predicen o determinan la utilidad de un *ítem* de un usuario basándose de la información y las calificaciones de otros usuarios respecto a un grupo de *ítems*. Las recomendaciones se basan en el grado de semejanza entre usuarios; los elementos que prefiere un usuario pueden ser favorables para otros usuarios con gustos afines. Los algoritmos de recomendación colaborativos se clasifican en dos categorías generales: algoritmos basado en modelos y basados en memoria. Los algoritmos basados en memoria utilizan todos los datos disponibles de una base de datos del sistema para calcular predicciones y recomendaciones. En contraste, los algoritmos basados en modelos operan derivando primero un modelo a partir de los datos del sistema, y este modelo es posteriormente utilizado en el proceso de recomendación. (Marina, 2014; Viltres,2014).

Los SR basados en filtrado colaborativo tienen una extensa variedad de algoritmos (ver figura 2) para realizar el procesamiento de la información; entre los más empleados se encuentran los algoritmos de los k vecinos (Viltres, 2014).

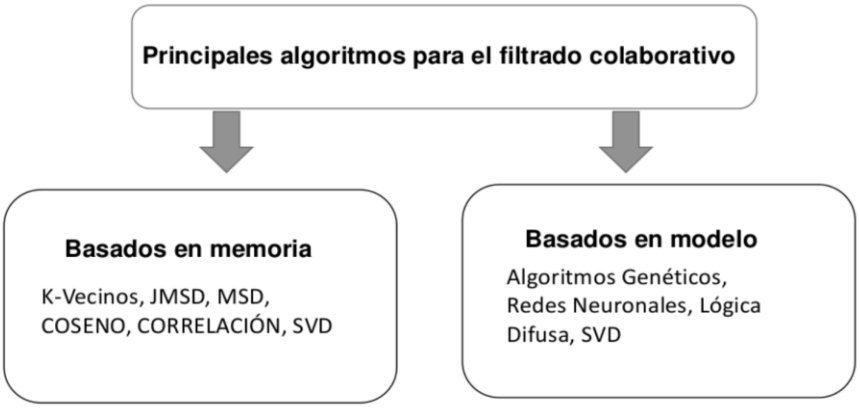

Figura 2. Algoritmos de filtrado colaborativo

Ventajas de los SR basados en filtrado colaborativo (Núñez, 2012; Caro, 2017):

- No necesita modelo detallado de preferencias; solo requiere de un vector valoración de objetos.
- Permite recomendar contenidos difíciles de analizar.
- Recomendar *ítems* basados en las preferencias del usuario.
- Realizar recomendaciones válidas, pero no esperadas.
- Puede aplicarse a cualquier tipo de *ítem* o producto: documentos, música, películas, libros, etc.

Desventajas de los SR basados en filtrado colaborativo (Núñez, 2012; Caro, 2017):

- Requiere mucho espacio de almacenamiento y tiempo de proceso para determinar usuarios similares. El coste computacional es elevado.
- Es imprescindible conocer la valoración de algunos objetos para que el proceso pueda funcionar.
- Problema de *Cold-Start*: Problema del Usuario Nuevo (o *early rater*) y problema de *ítem* Nuevo.
- Problema de Dispersión (*Sparsity*): Si el número de usuarios es pequeño en relación al volumen de información en el sistema, el cubrimiento de las valoraciones se vuelva muy disperso y disminuye la colección de *ítems* recomendables.
- Problema de Escalabilidad: A medida que la cantidad de usuarios y de *ítems* crece, también crece la cantidad de cómputos de vecinos más cercanos para la determinación de usuarios similares, y como los cálculos se hacen en tiempo real, el sistema puede colapsar.

1.3.2 Sistemas de recomendación basado en contenido

Los sistemas de recomendaciones basados en contenido utilizan la recuperación de información para encontrar las preferencias del usuario (perfil del usuario) y la descripción del *ítem* (Viltres, 2014). Rodríguez (2012) plantea que las recomendaciones emplean un perfil creado y los algoritmos "*ítem* a *ítem*" generado a través de la asociación de reglas de correlación entre ellos. Para realizar la recomendación se considera el *ítem*, el perfil del usuario y el algoritmo de recomendación; hacer coincidir los atributos del perfil del usuario con los atributos de los *ítems* a recomendar (Hdioud et al., 2012). Los algoritmos de recomendación basado en contenidos utilizan modelos de recuperación de información relativamente simples, entre estas técnicas según varios autores (Font, 2009; Castro, 2012) las más empleadas son el modelado vectorial, las redes bayesianas y reglas de asociación.

Ventajas de los SR basados en contenido (Núñez, 2012; Caro, 2017):

- Recomendación por contenido y no por opiniones subjetivas de otros usuarios
- El sistema puede generar explicaciones sobre la recomendación que hizo en base al historial del usuario
- No hay Dispersión (*Sparsity*): El modelado de la información está presente en las características del documento y no necesitan proveerlas otros usuarios
- Elemento Nuevo (*New Item*): El sistema es capaz de recomendar elementos que todavía no han sido calificado por ningún usuario

Desventajas de los SR basados en contenidos (Núñez, 2012; Caro, 2017):

- Necesita un modelo detallado de preferencias del usuario, que es complejo de construir y mantener
- Sobre especialización: Al usuario se le recomiendan *ítems* similares a los que recomendó
- Subjetividad de los Contenidos: Dificultad en dominios con contenido difícil de analizar, (audio, gráficos, imágenes, vídeo)
- Problema del Usuario Nuevo: El usuario tiene que puntuar un número suficiente de *ítems* para que el sistema pueda realmente entender sus preferencias
- Problema Estabilidad vs Plasticidad: Es difícil para el sistema aprender a adaptarse, a los cambios en el perfil del usuario hasta no haber recolectado un número suficiente de valoraciones actualizadas

1.3.3 Sistemas de recomendación basado en conocimiento

Las recomendaciones se realizan en base a la utilidad o la necesidad del *ítem* para el usuario. Se recomiendan *ítem* basados en el conocimiento de un dominio específico de cómo ciertas características de productos satisfacen las necesidades y preferencias de los usuarios y, en última instancia, cómo el producto es útil para el usuario (Viltres, 2014). Los SR basados en conocimiento sirven para resolver las restricciones de los anteriores tipos de sistemas, es decir, solucionan el problema de las pocas interacciones o valoraciones con ciertos productos. Estos SR utilizan los requisitos de los usuarios (que realizan de forma explícita) para hacer las recomendaciones más personales. Un factor principal de estos sistemas es que el usuario tiene un gran control sobre las recomendaciones por la necesidad de establecer sus propios filtros

o requisitos sobre el amplio dominio (Caro, 2017). Estos sistemas utilizan para la integración del conocimiento en los procesos de recomendación la técnica Razonamiento Basado en Casos (RBC) o Conocimiento mediante ontologías (Castro, 2012).

Técnica de Razonamiento Basado en Casos

El Razonamiento Basado en Casos (en inglés, *Case-Based Reasoning*) es una técnica de resolución de problemas que utiliza el conocimiento que proporcionan las experiencias previas. Esta técnica realiza una evaluación del problema que se plantea y obtiene de las experiencias almacenada la solución del mismo (Castro, 2012).

En la técnica de RBC para determinar la similitud entre un problema (P) y un caso (C) es necesario definir una función de distancia entre los valores de cada de atributo. La distancia posee relación inversamente proporcional con el concepto de similitud, mientras la distancia existente entre los valores de los atributos del problema y el caso sea menor, mayor es la similitud entre estos (Wong, 2012). Algunas de las funciones de distancia que permiten calcular la similitud entre casos de valores cuantitativos son la distancia euclidiana, la distancia de Manhattan, la distancia de Mahalanobis y la distancia coseno. En la presente investigación la autora determino utilizar la función de distancia de Manhattan:

$Sim(P,C) = \frac{\sum_{i=1}^{n} w_i * |P_i - C_i|}{\sum_{i=1}^{n} w_i}$ (2), donde:

w_i: es la importancia del atributo

P_i y C_i: son los valores que el atributo i tiene en el problema y en el caso respectivamente.

Ventajas de los SR basados en conocimiento (Gómez, 2012; Caro, 2017):

- Facilidad para explicar las recomendaciones
- No necesita que los usuarios hayan puntuado los objetos:
  - No hay "*cold start*"
  - No hay sobre especialización

Desventajas de los SR basados en conocimiento (Gómez, 2012; Caro, 2017):

- La obtención del conocimiento no es sencilla: Necesita interactuar varias veces con el usuario

1.3.4 Sistemas de recomendación híbridos

Los sistemas híbridos mezclan múltiples técnicas para lograr obtener los beneficios de la combinación de más de un método de recomendación. La información para generar las recomendaciones puede obtenerse de las diferentes fuentes de datos y se utiliza la eficiencia de los diferentes algoritmos de recomendación con el objetivo hacer el sistema más robusto. La decisión más importante es decidir qué sistemas mezclar y qué información va a utilizar cada uno, por eso se obtienen mejores resultados cuando se emplean diferentes tipos de métodos (Viltres, 2014; Caro, 2017).

Después de analizadas las clasificaciones de los sistemas de recomendación, se determinó desarrollar un sistema de recomendación basado en conocimiento y se propone como técnica a emplear Razonamiento Basado en Casos. Esta técnica utiliza el conocimiento que proporcionan las experiencias previas almacenadas. Esta decisión de la autora de la presente investigación se basa en las características de la propuesta de solución y en las necesidades del cliente.

1.4 Tecnologías, herramientas y metodología asociadas al desarrollo de la solución

La propuesta de solución es un sistema de recomendación de costos para proyectos de software, al analizar la información actualizada relacionada con este, se determinan las bases tecnológicas y las herramientas que fueron utilizadas para su implementación. Esto garantiza el correcto funcionamiento y utilización del sistema.

Framework

Los marcos de trabajo o *framework* se diseñan para facilitar el desarrollo de softwares. Estos permiten a los diseñadores y programadores pasar más tiempo en identificar requerimientos de software que en realizar los detalles de bajo nivel para proveer un sistema funcional. Un *framework* es una estructura de soporte definida para que los proyectos de softwares puedan ser organizados y desarrollados. Estos marcos de trabajo pueden incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros softwares para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio (Gómez y otros, 2016). Existen diferentes marcos de trabajo, *Symfony, Django, AngularJS.*

Django

Django es un *framework* de alto nivel basado en *Python* que facilita el desarrollo de aplicaciones web dinámicas. Es un *framework* que abstrae a los programadores de los problemas comunes del desarrollo web y acelera las tareas más frecuentes en la programación. Proporciona un método de mapear las URL (del inglés, *Uniform Resource Locator*), al ejecutar un código en especial para cada una. A través de plantillas ayuda a separar el contenido de la presentación evitando tener que manipular la lógica de negocio cuando se necesite realizar cambios de apariencia en la página. Permite lidiar con trescientas peticiones *web* por segundo. *Django* incluye muchas aplicaciones comunes a todos los sitios *web*, como la autenticación de usuarios o la administración del contenido del sitio. Su arquitectura está inspirada en el patrón Modelo-Vista-Controlador (MVC), encargándose en gran parte de los controladores, y proveyendo herramientas para facilitar el desarrollo de las vistas y los modelos. *Django* posee además soporte nativo para MySQL como gestor de bases de datos. Entre los sitios que usan este *framework* actualmente se encuentran, entre otros, Mozilla y OpenStack38 (Django, 2014). Por todas estas características se escoge trabajar con Django en su versión 1.7 como entorno de desarrollo.

Lenguaje *Python*

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90, cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”. Este es un lenguaje de programación multiparadigma, soporta orientación a objetos, programación imperativa, programación orientada a aspectos y en menor medida, programación funcional.

Algunas de las ventajas del uso de *Python* (Dayley, 2007):

- **Multiplataforma:** Hay versiones disponibles de *Python* en muchos sistemas informáticos distintos. Originalmente se desarrolló para Unix, aunque cualquier sistema es compatible con el lenguaje siempre y cuando exista un intérprete programado para él.
- **Sintaxis clara:** Tiene una sintaxis visual, por una notación *identada* (con márgenes) de obligado cumplimiento. Esto ayuda a los programadores adoptar unas mismas notaciones y que los programas de las personas tengan un aspecto muy similar.
- **Mixto:** Se puede integrar de manera "fácil" con otros lenguajes de programación.
- **Gratuito:** Una ventaja de Python es la gratuidad de su intérprete, se puede descargar desde la página web: http://www.Python.org.

El empleo del lenguaje de programación Python en esta investigación está regido principalmente por las potencialidades de este lenguaje y a que tiene librerías cliente para permitir el desarrollo de aplicaciones clientes en este lenguaje.

Lenguaje *HyperText Markup Language* 5 (HTML)

HTML es un lenguaje de marcado de hipertexto. Este es un lenguaje de programación ideado para permitir la creación de sitios web.

Algunas de sus características son:

- Estructura del cuerpo: permite agrupar todas estas partes de una web en nuevas etiquetas que representarán cada una de las partes típicas de una página.
- Etiquetas para contenido específico: utiliza etiquetas específicas para cada tipo de contenido en particular, como audio, video, etcétera.
- Bases de datos locales: permite el empleo de una base de datos local para trabajar en una página web por medio del cliente y a través de un API.

Lenguaje *Cascading Style Sheets* 3 (CSS)

Las hojas de estilo en cascada CSS, por sus siglas en inglés (Cascade Style Sheet) proponen una navegación más rápida, menores tiempo de respuesta producidos por una reducción de imágenes, las cuales serán requeridas para diseñar botones o efectos de textos. También deja atrás una excesiva dependencia de JavaScript para fines de representación visual, como son las animaciones, dando como resultado menos código y mejor rendimiento. Representa una menor dependencia de software para gráficos que resultan bastante caros, como es el caso de Photoshop, Illustrator o Corel. En la actualidad, CSS3 brinda la facilidad de producir diseños más flexibles que permiten optimizar la experiencia del usuario, el uso de herramientas permite mejorar el rendimiento y la visualización en base al dispositivo de navegación que utilice el usuario. Con los nuevos selectores se pueden realizar efectos de animación sin el uso de JavaScript. Una de las propiedades más populares en el uso de CSS3 es aquella que se aplica a la edición y estilización de textos, como son: text-shadow, text-stroke y text-overflow. Box-shadow es otra propiedad que permite a los diseñadores implementar fácilmente sombras sobre cualquier elemento tipo capa (Webera, 2017).

Gestor de base de datos MySQL 5

Para la gestión de la base de datos se utiliza MySQL 5.7, es un sistema de gestión de bases de datos (SGBD) relacional, su diseño multihilo le permite soportar una gran carga de datos de forma eficiente. *MySQL* es el gestor de base de datos de código abierto más utilizado a nivel mundial por su rapidez y facilidad de uso datos ofrecidos por Oracle (2017). Su utilización está enfocada en aplicaciones web dinámicas escritas en *PHP* por la optimización de consultas sencillas y su compatibilidad con el servidor web *Apache*. Este se caracteriza por su nivel de seguridad, es multiplataforma, admite hasta 32 índices por tablas, la variedad de tipos de datos para las columnas y la utilización de *triggers* (Oracle, 2017).

Servidor web Apache

Para el hospedaje del sistema se emplea *Apache web Server* 2.2*,* es el servidor HTTP de código abierto desarrollado por la empresa *Apache Software Fundation* en 1996*.* Es un servidor web flexible, rápido y eficiente, altamente configurable por su diseño modular, esta característica permite ampliar considerablemente sus capacidades pues existe un repositorio extenso completamente gratuito de extensiones y módulos. Es compatible con el lenguaje de programación *PHP,* comparten muchas de sus características (Apache, 2017).

Visual Paradigm for UML 8.0

Visual Paradigm for UML es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, implementación y pruebas. El software ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite construir todos los tipos de diagramas de clases, realizar ingeniería inversa desde gestores de bases de datos, generación de bases de datos, generar código desde diagramas y generar documentación (Visual Paradigm, 2015). Esta herramienta será utilizada para la realización y diseño de todos los diagramas necesarios para documentar el proceso de desarrollo de la propuesta de solución.

PyCharm Professional Edition

Para el desarrollo de la propuesta de solución se utiliza *JetBrains Pycharm* en su versión 10.0, es un Entorno Integrado de Desarrollo (*IDE*, por sus siglas en inglés) que se utiliza para la programación en *Python*. Proporciona análisis de código, un depurador gráfico, un probador de unidad integrada, integración con

sistemas de control de versiones (VCSes), y apoya el desarrollo web con Django. *PyCharm* es desarrollado por la empresa checa JetBrains (JetBrains, 2017).

Lenguaje de modelado UML

El Lenguaje Unificado de Modelado (UML -*Unified Modeling Language*) es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de *software*. UML utiliza diagramas y una semántica bien definida para modelar casos conceptuales, procesos de negocio y funciones de sistema; casos concretos escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de *software* reusables, en las distintas etapas de del ciclo de vida del software, fundamentalmente en el análisis y diseño. Brinda el lenguaje de modelado para:

- Modelo de proceso de negocios con casos de uso.
- Modelado de clases con objetos.
- Modelado de componentes.
- Modelado de distribución y despliegue.

El lenguaje ayuda al usuario a comprender bien el funcionamiento del software y reflexionar antes de invertir grandes cantidades de dinero en un proyecto no viable. El modelado ayuda, a mejorar la capacidad del equipo para gestionar la complejidad del software. Permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma convencional fácil de comprender para comunicarlas a otras personas (STEVENS, 2007; Krall, 2016).

Metodología de desarrollo de software

Una metodología de desarrollo de *software* describe un entorno que se emplea para organizar, planificar, y dirigir un proceso de desarrollo de un *software*. Existen varias metodologías de desarrollo de *software*, todas contienen algunas etapas básicas del ciclo de desarrollo de *software,* la planificación, análisis, diseño, implementación y mantenimiento (Martínez, 2016). Para el desarrollo de la solución se empleará AUP-UCI, variación de la metodología Proceso Unificado Ágil (AUP por sus siglas en inglés).

Metodología AUP-UCI

Para guiar el proceso de desarrollo de la propuesta de solución se decidió utilizar la metodología AUP-UCI, de esta forma se logra estandarizar el proceso de desarrollo de software. La metodología AUP-UCI es una

variación de la metodología de desarrollo AUP (Proceso Unificado Ágil) creada por la UCI. Fue creada con el objetivo de adaptarse al ciclo de vida definido para la actividad productiva de la universidad. Esta metodología es recomendada para proyectos no muy extensos y permite ser adaptada a las peculiaridades de cada proyecto, lográndose así que el proceso sea configurable. Las tres fases que define AUP-UCI para componer el ciclo de vida de un proyecto son

- Inicio: realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo, costo y decidir si se ejecuta o no el proyecto.
- Ejecución: ejecuta las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtiene los requisitos, se elabora la arquitectura y el diseño, se implementa y se libera el producto.
- Cierre: analiza tanto los resultados del proyecto como su ejecución y se realizan las actividades formales del cierre del proyecto.

AUP-UCI define cuatro escenarios para modelar el sistema en dependiendo de la complejidad y características del proceso de desarrollo. El escenario número dos de esta metodología se adapta al desarrollo del componente propuesto: Aplica a los proyectos que evalúan el negocio a informatizar para obtener un negocio bien desarrollado. El cliente siempre acompañará al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos (Universidad de las Ciencias Informáticas, 2015).

Herramientas validación

Las pruebas de validación en la ingeniería de software son el proceso de revisión que verifica que el sistema de software producido cumple con las especificaciones y que logra su cometido. La validación es una parte del proceso de pruebas de software de un proyecto, que también utiliza técnicas tales como evaluaciones, inspecciones y tutoriales. Es el proceso de comprobar que las especificaciones del usuario fueron cumplidas (Pressman, 2002). Para realizar estas pruebas se utilizan herramientas, en el desarrollo de la solución se empleó *Acunetix.*

Acunetix

Acunetix WVS, por sus siglas en inglés (Web Vulnerability Scanner) es una herramienta de seguridad de aplicaciones web automatizada. Es capaz de escanear cualquier sitio o aplicación web que es accesible a través del protocolo HTTP/HTTPS. Sin embargo, no todas las pruebas se pueden realizar de forma automática, por lo tanto, Acunetix WVS proporciona herramientas de penetración manuales para pruebas particulares. Comprueba diferentes vulnerabilidades (por ejemplo, inyección de SQL, Cross Site Scripting). Hasta la fecha Acunetix comprueba sobre más de 500 tipos diferentes de vulnerabilidades (acunetix, 2012).

1.5 Conclusiones del capítulo

Como resultado de la investigación realizada en este capítulo se puede concluir que:

- El estudio de los softwares de gestión de costos demostró que en el proceso estimación de costos, solo se calcula el costo por actividad o por etapa, por lo que se hace necesario el desarrollo de un sistema para la gestión de costos que permita obtener el costo general de un proyecto de software.
- El estudio de los referentes teóricos permitió identificar las funcionalidades básicas del sistema.
- El estudio sobre las metodologías, herramientas y lenguajes permitió definir los componentes base para el desarrollo de la solución que son AUP-UCI como metodología, *Visual Paradigm* 8.0, *Django* en su versión 1.10, *Apache* 2.2, *MySQL* 5.7 y *PyCharm* 10.0.

Capítulo 2: Análisis y diseño del Sistema de recomendación de costos asociados a proyectos de desarrollo de software

En el presente capítulo se describe la propuesta de solución basada en un sistema de recomendación (SR) que permita facilitar la estimación de los costos de los proyectos de software. Además, se identifican a partir del modelo de negocio los requisitos funcionales (RF) y los requisitos no funcionales del sistema (RNF), que generan los casos de uso (CU). También se elaboran los principales artefactos propuestos por la metodología, se modelan los diagramas, y se determina la arquitectura y los patrones de diseño.

2.1 Descripción de la propuesta de solución

El sistema de recomendación basado en conocimiento que se propone solo permite que accedan dos tipos de usuarios, los asesores de mercadotecnia y el administrador, que se encarga de asignar los permisos del sistema. Para solicitar la recomendación el usuario debe introducir los datos del nuevo proyecto software, que serán los parámetros de entrada. Los parámetros introducidos se procesan para seleccionar los datos de mayor importancia para ser utilizado por la técnica de Razonamiento Basado en Casos (RBC). La técnica de RBC realiza una comparación entre los datos seleccionados y los almacenados en el sistema mediante una función de similitud para determinar los proyectos con características similares. Después de analizar el proyecto más semejante según sus características al nuevo proyecto, el sistema muestra como resultado el costo recomendado y otros costos de proyectos similares.

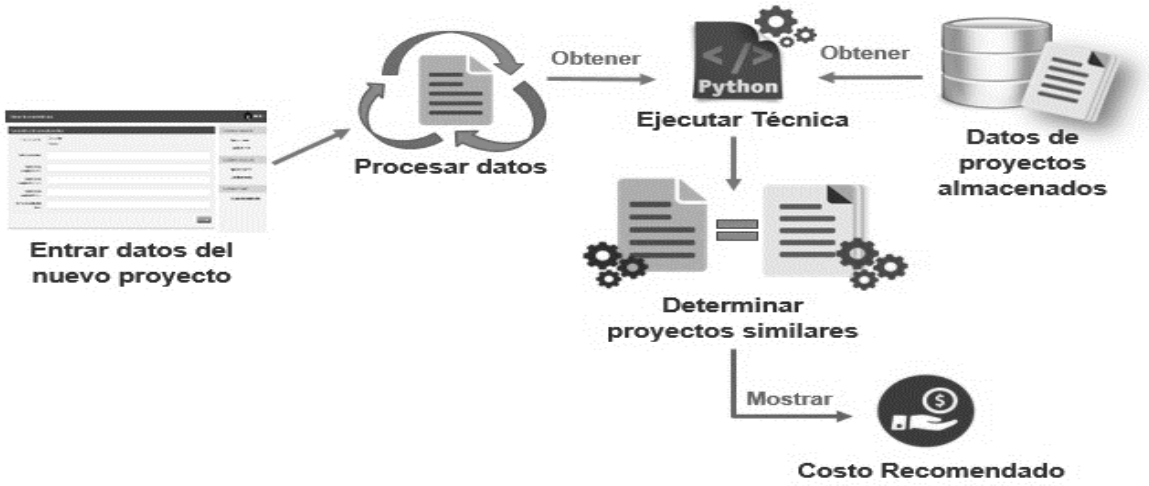

Figura 3. Funcionamiento del Sistema de Recomendación

2.2 Modelo de negocio

En los centros de desarrollo de la UCI, los asesores de mercadotecnia son los encargados de realizar el proceso de estimación de los costos del proyecto de software mediante la ficha de costo. Los asesores de mercadotecnia deben obtener toda la información del nuevo proyecto e introducir los datos en la ficha de costo. La ficha costo calcula y muestra el costo estimado del proyecto de software.

Un modelo de procesos de negocio describe cómo funciona el negocio, es decir, describe las actividades involucradas en el negocio y la manera en que se relacionan entre ellas e interactúan con los recursos necesarios para lograr la meta del proceso (Mora, 2009). A continuación, se muestra el diagrama de proceso de negocio que modela cómo funciona el proceso de estimación de los costos.

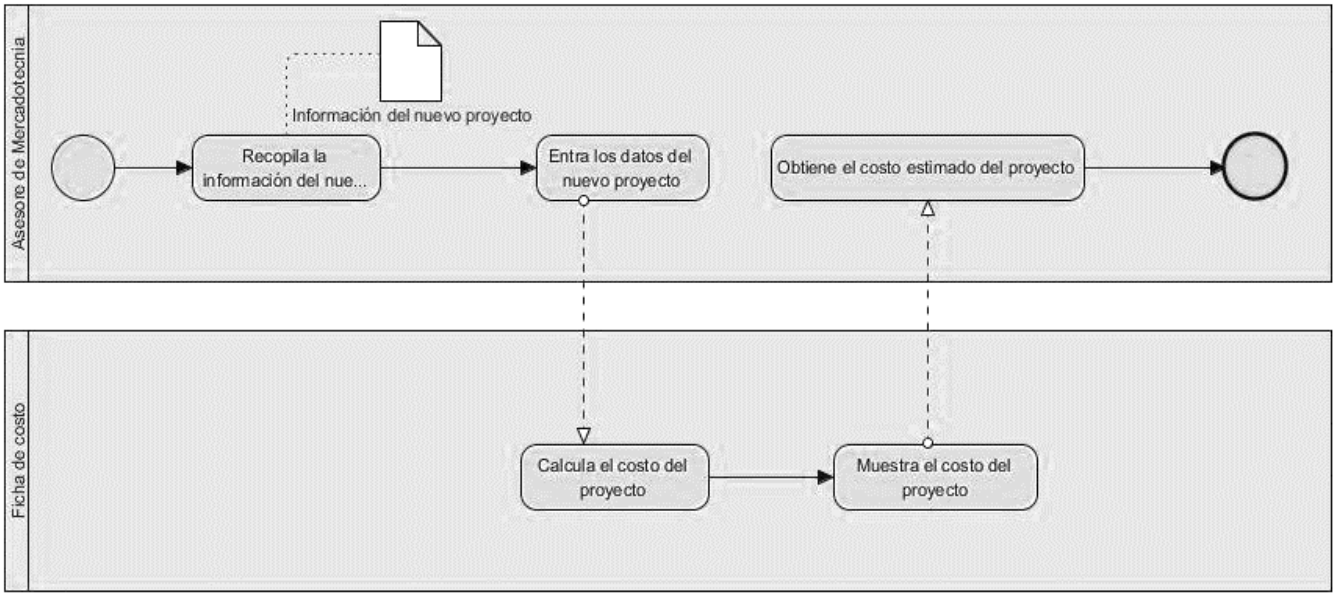

Figura 4. Diagrama de proceso de negocio.

2.3 Especificación de los requisitos del software

Los requisitos de software son las cualidades del sistema, se definen y describen de forma clara, consistente, compacta, y sin ambigüedades. Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir; los no funcionales son propiedades o cualidades del producto. Los requisitos forman parte del modelo del sistema que se desarrolla (Pressman, 2002). A continuación, se presentan los requisitos funcionales y no funcionales extraídos.

2.3.1 Requisitos funcionales

Tabla 2. Requisitos funcionales.

Nº	Nombre	**Descripción**	**Prioridad**	**Complejidad**
RF1	Autenticar usuario	El sistema debe permitir que los usuarios se autentiquen al introducir su usuario y contraseña.	Baja	Media
RF2	Agregar usuario	El sistema debe permitir que los usuarios con rol administrador registren usuarios en el sistema, al introducir sus datos y asignándole roles.	Alta	Media
RF3	Editar usuario	El sistema debe permitir que los usuarios con rol administrador modifiquen los datos.	Media	Baja
RF4	Eliminar usuario	El sistema debe permitir que los usuarios con rol administrador cancelen las cuentas de usuarios existentes.	Media	Alta
RF5	Mostrar datos de un usuario	El sistema debe permitir que los usuarios con rol administrador visualicen los datos de un usuario seleccionado del listado de usuarios existentes	Baja	Alta
RF6	Mostrar listado de usuarios	El sistema debe permitir que los usuarios con rol administrador visualicen el listado de usuarios existentes.	Baja	Baja
RF7	Agregar proyecto	El sistema debe permitir que los usuarios registren proyectos en el sistema, al introducir sus datos.	Alta	Media
RF8	Editar proyecto	El sistema debe permitir que los usuarios con rol administrador modifiquen los datos de los proyectos.	Media	Baja
FR9	Eliminar proyecto	El sistema debe permitir que los usuarios con rol administrador eliminen los proyectos existentes.	Media	Alta
FR10	Mostrar datos de un proyecto	El sistema debe permitir que los usuarios con rol administrador visualicen los datos de un proyecto seleccionado del listado de proyectos existentes	Baja	Alta
RF11	Mostrar listado de	El sistema debe permitir que los	Baja	Baja

	proyectos	usuarios con rol administrador visualicen el listado de proyectos existentes.		
RF12	Generar perfil de recomendación	El sistema debe permitir que los usuarios autenticados soliciten la recomendación del costos del nuevo proyecto.	Alta	Media
RF13	Generar recomendación de costo	El sistema debe buscar los proyectos similares al nuevo proyecto introducido por los usuarios.	Alta	Alta
RF14	Mostrar recomendación de costo	El sistema debe mostrar el costo del proyecto más similar a las características del nuevo proyecto	Alta	Media
RF15	Mostrar costo de proyectos similares	El sistema debe mostrar el costo de los proyectos similares al nuevo proyecto.	Alta	Media

2.3.2 Requisitos no funcionales

- **Seguridad**

  - RNF1. Se garantizará la integridad de la información mediante mecanismos de control de usuario, contraseña y niveles de accesos para cada usuario, de manera que cada uno pueda tener disponible solamente las opciones que se encuentran en correspondencia con su actividad.
  - RNF2. Los errores deben mostrar la menor cantidad de detalles posible, para evitar brindar información que comprometa la seguridad e integridad del sistema.
  - RFN3. Se usarán mecanismos de encriptación de la contraseña que por cuestiones de seguridad no deben estar en texto plano, las contraseñas en la base de datos se almacenarán en forma encriptada.

- **Hardware**

  - RNF4. El servidor de aplicaciones web y de base de datos deben poseer como mínimo un CPU Core2Duo a 2.20 GHz con 2 GB de RAM.
  - RNF5. El servidor de base de datos debe poseer una capacidad mínima de 4 GB.
  - RNF6. El servidor de aplicaciones web debe poseer una capacidad mínima de 5GB.

- **Software**
    - RNF7. El sistema debe funcionar en cualquier entorno de Windows y Linux.
- **Operaciones**

  RNF8. Para la instalación de este producto se necesitará un entorno de trabajo compuesto por:

    - Servidor web: Apache 2.4.14
    - Base de datos: MySQL 5.7
    - Framework: Django 1.10
    - Lenguaje de programación: Python 2.7

2.4 Definición de casos de uso

A continuación, se definen los casos de uso que se corresponden con los requisitos identificados. Los casos de uso son artefactos narrativos que describen, mediante acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Además, son una de las bases durante las fases de análisis, diseño e implementación del sistema. Para consultar la descripción de los casos de uso consultar el anexo 2.

Tabla 3. Casos de uso del sistema

CU	Referencia
CU1: Asignar permisos	RF16
CU2: Autenticar usuario	RF1
CU3: Gestionar usuarios	RF2, RF3, RF4, RF5, RF6
CU4: Gestionar proyectos	RF7, RF8, RF9, RF10, RF11

CU5: Obtener recomendación de costos	RF12, RF13, RF14, RF15,

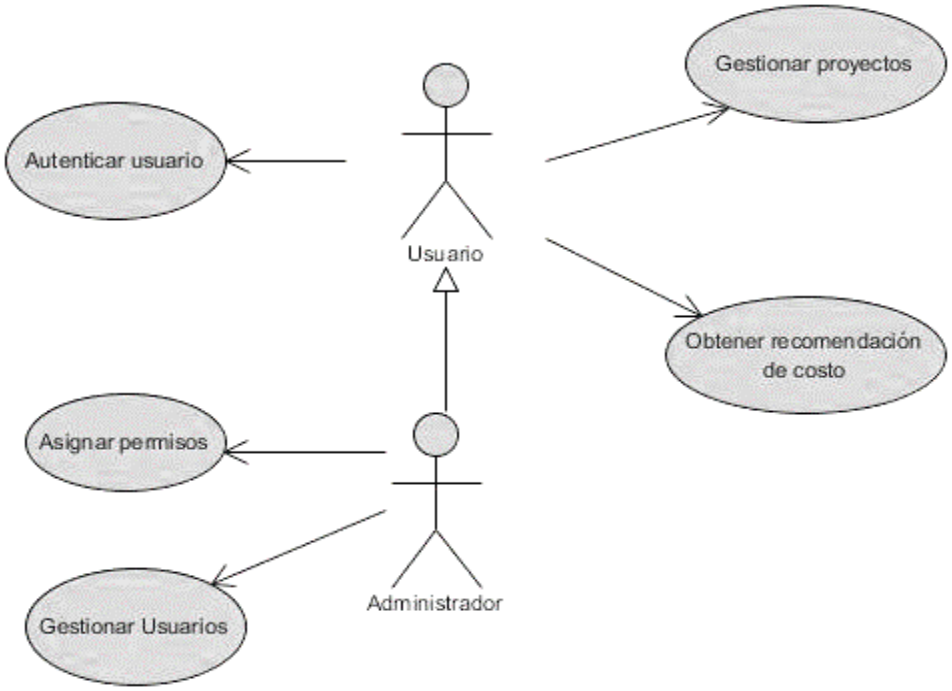

Figura 5. Diagrama de casos de uso del sistema de recomendación de costos.

2.5 Descripción de la arquitectura de software y los patrones de diseño

La arquitectura es un elemento primordial en el éxito o el fracaso de un proyecto, proporciona una visión global del sistema a construir, es una vista estructural de alto nivel que define el estilo arquitectónico o su combinación para la solución de un determinado problema (Rodríguez, 2015).

Arquitectura de software

Django utiliza una modificación de la arquitectura Modelo Vista Controlador (MVC), llamada MTV (*Model Template View*), que sería Modelo Plantilla Vista, en Django la vista se llama plantilla (*Template*) y el controlador se llama vista (*View*) (Infante, 2012). Esta arequitectura permite separar los datos, la lógica de la aplicación y la presentación gráfica con la finalidad de conseguir un código limpio y un fácil mantenimiento. Para una mejor representación de la arquitectura MTV se presenta la siguiente imagen.

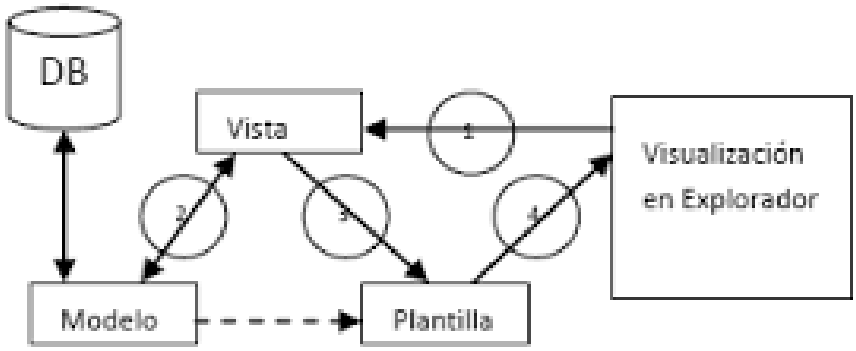

Figura 6. Arquitectura del Sistema. Fuente: Infante, 2012

Funcionamiento del MTV de Django:

1. El navegador manda una solicitud
2. La vista interactúa con el modelo para obtener datos
3. La vista llama a la plantilla
4. La plantilla renderiza la respuesta a la solicitud del navegador

Modelo: El modelo tiene como objetivo mapear la base de datos de tal forma que cree una sincronización entre la base de datos y la aplicación, con el fin de mantener actualizada toda la información de las tablas, campos y datos.

Vista: La vista tiene como objetivo recibir el requerimiento que es enviado por el navegador, procesar la información, si es necesario realizar una petición al modelo para extraer un valor de base de datos u otro de los casos podría ser que sólo muestre un mensaje enviando un requerimiento a la plantilla.

Plantilla: La plantilla es básicamente una página HTML con algunas etiquetas extras propias de Django. Esta recibe los datos de la vista y luego los organiza para la presentación al navegador web.

Patrones de diseño

Los patrones de diseño son descripciones de clases cuyas instancias colaboran entre sí y soluciones simples a problemas específicos y comunes del diseño orientado a objetos. Cada patrón es adecuado para ser adaptado a un cierto tipo de problema. La clave para la reutilización es anticiparse a los nuevos requisitos

y cambios, de modo que los sistemas evolucionen de forma adecuada. Facilitan la reusabilidad, extensibilidad y mantenimiento (genbetadev, 2015).

Patrones de diseño GRASP

Los GRASP (*General Responsibility Assignment Software Patterns*) son patrones generales de software para la asignación de responsabilidades a las clases (Ibáñez, 2014). Los patrones para asignar responsabilidades que a continuación se mencionan son los utilizados en el Sistema.

- **Controlador:** El patrón controlador es el intermediario entre la interfaz y el algoritmo que la implementa, así recibe los datos del usuario y los envía a las clases según el método llamado. Este patrón se evidencia en la clase *view* es quien delega las funcionalidades.
- **Bajo acoplamiento:** Las clases están lo menos ligadas entre sí, en caso de producirse una modificación en alguna de ellas, tenga la mínima repercusión posible en el resto de las clases, esto fomenta la reutilización y disminución de dependencia entre las clases. Se evidencia en la estructura misma del *framework* Django (*models*, *view, template*).
- **Alta cohesión:** Se aplica en la mayoría de las clases del diseño; cada una de las clases solo se implementan las funcionalidades que le corresponden. En la solución este patrón es el resultado de asignar responsabilidades únicas a cada uno de los componentes (*models:* se realizan todas las consultas a la base de datos*, view:* se implementan todas las funcionalidades del sistema*, template:* son las vistas que ve el usuario

Patrones de diseño GoF (*Gang of Four)*

Describen las formas comunes en que diferentes tipos de objetos, pueden ser organizados para trabajar unos con otros. Gestionan la correspondencia entre clases y la creación de estructuras de alta complejidad. Además, permiten formar grupos de objetos para realizar tareas complejas (Craig, 2003). Los patrones GOF utilizados en el sistema son:

Patrones de Comportamiento: son soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan.

- ***Mediator*** *(*mediador*)***:** Objeto que encapsula cómo otro conjunto de objetos interactúa y se comunican entre sí. Las clases mediadoras del sistema son *models*, *view* y *template*

Patrones Creacionales: solucionan problemas de creación de instancias, ayudan a encapsular y abstraer dicha creación

- ***Factory Method*** *(*Método de fábrica*)**:*** Expone un método de creación, delegando en las subclases la implementación de este método. Este patrón se evidencia en la clase *view,* esta delega en diferentes subclases la implementación de una funcionalidad

2.6 Diagrama de clases de diseño

El diagrama de clases del diseño es la representación gráfica de las clases que serán implementadas en el sistema. Este diagrama muestra las especificaciones y detalles más concretos de cada clase del sistema, así como su relación de asociación, composición o agregación existentes entre ellas (Ambler, 2003). A continuación, se observa el diagrama de clases del diseño del CU # 5: Realizar recomendación de costos. Los restantes se pueden consultar en el anexo 3.

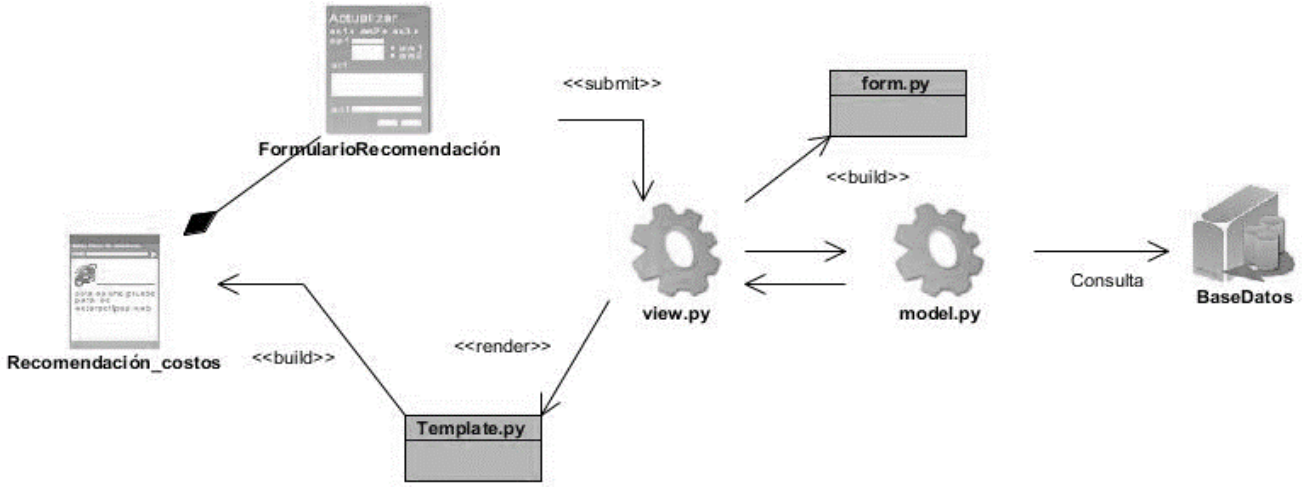

Figura 7. Diagrama de clase de diseño del CU5 Obtener recomendación de costos.

Descripción del diagrama de clase de diseño del CU # 5

Para obtener una recomendación el navegador envía una solicitud a la página servidora view.py, que contiene las funciones de las vistas. El view.py a su vez interactúa con el model.py para obtener los datos de los proyectos en la BaseDatos y ejecuta la función de recomendación. Después la página servidora llama al Template.py para que este renderice la respuesta a la solicitud del navegador y muestre el costo recomendado y los costos de proyectos con características similares.

2.7 Diagrama de secuencia del diseño

Los diagramas de secuencia, describen las interacciones entre un grupo de objetos y muestra de forma secuencial los envíos de mensajes entre estos (docs.kde.org, 2015). A continuación, se presenta el diagrama de secuencia: Realizar recomendación de costos. Los restantes se pueden consultar en el anexo 4.

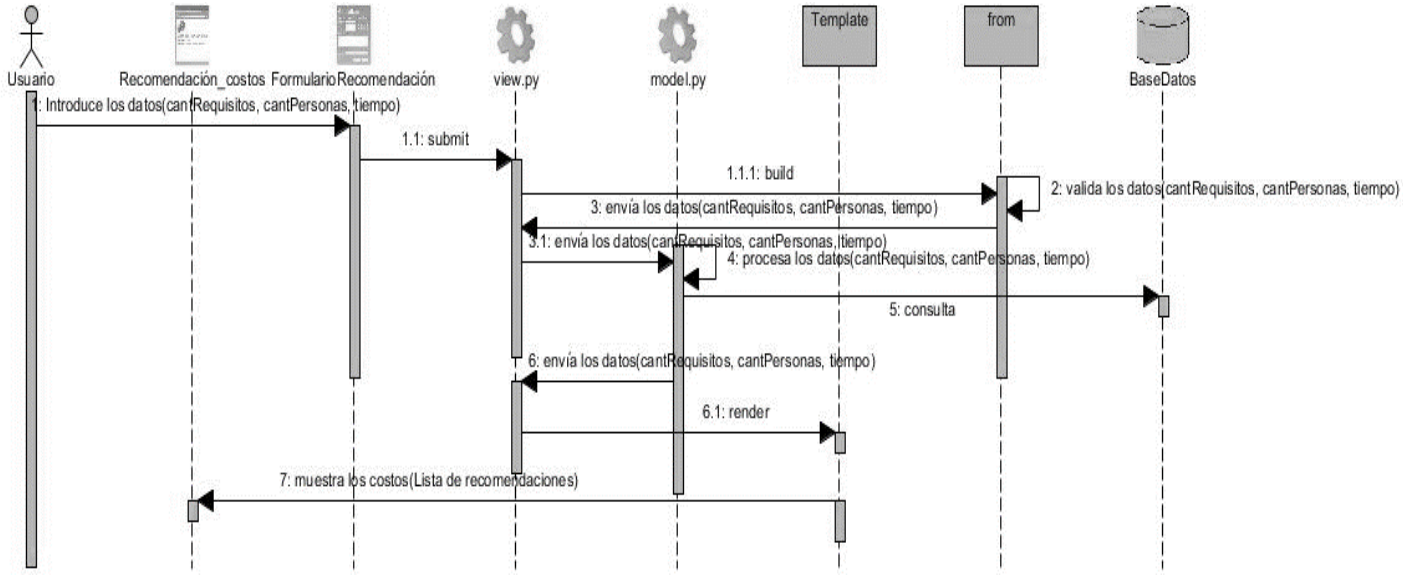

Figura 8. Diagrama de secuencia: Obtener recomendación de costos.

2.8 Diagrama de datos

Un diagrama de datos muestra la estructura lógica de una base de datos, incluidas las relaciones y limitaciones que determinan cómo se almacenan los datos y cómo se accede a ellos. Estos modelos de datos se pueden representar por medio de un diagrama de base de datos (microsoft, 2015).

Modelo de datos entidad relación

Los modelos de datos aportan la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos. Un modelo de datos está orientado a representar los elementos que intervienen en la realidad o en un problema dado y la forma en que se relacionan dichos elementos entre sí (Blázquez, 2014).

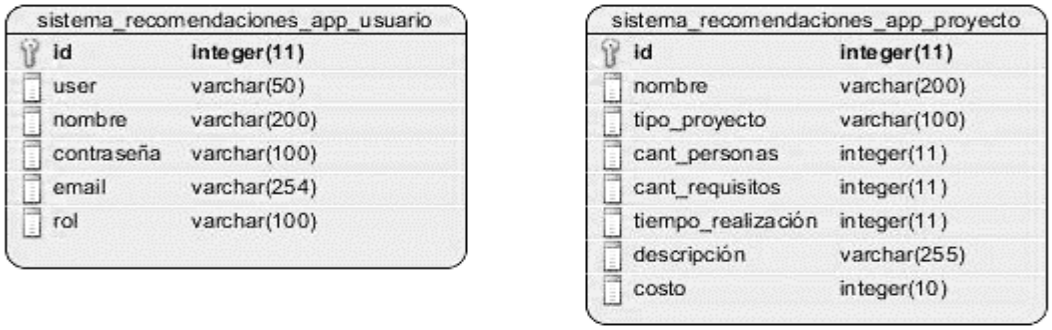

Figura 9. Modelo de Base de Datos.

2.9 Modelo de despliegue

El diagrama de despliegue está compuesto por nodos y sus relaciones, que modelan la distribución física de un sistema. La figura 11 muestra el diagrama de despliegue correspondiente a la propuesta de solución.

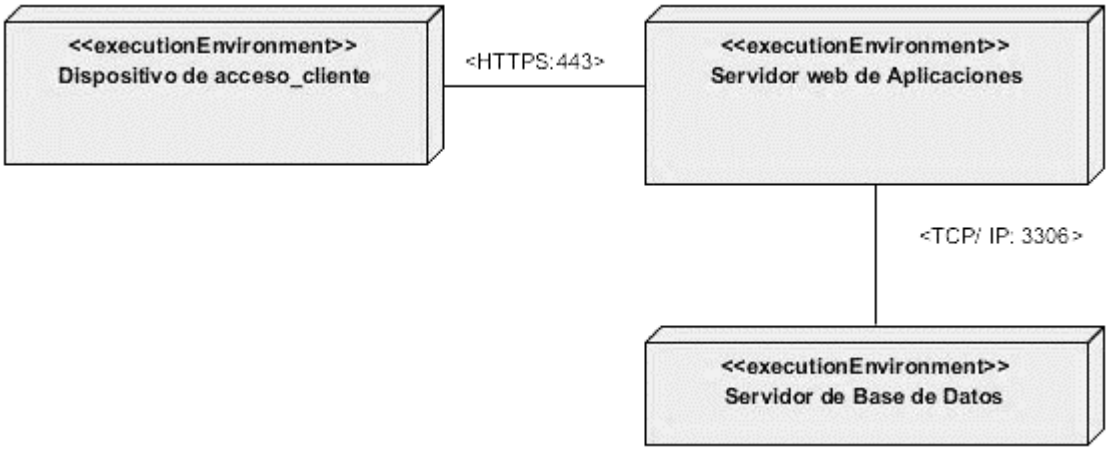

Figura 10. Modelo de despliegue.

Descripción de los elementos de interface y comunicación

<<HTTPS>>: Protocolo para establecer la conexión segura entre el dispositivo de acceso_cliente y el servidor web de aplicaciones a través del puerto.

<<TCP/IP>>: Protocolo para establecer la conexión entre el servidor de aplicaciones y el servidor de base de datos a través del puerto definido para el gestor de base de datos MySQL: 3306. La conexión entre estos servidores permitirá ejecutar un conjunto de órdenes y obtener rápidamente respuesta a las mismas.

Descripción de los elementos que componen el diagrama

Dispositivo de acceso del cliente: Los dispositivos de acceso pueden ser ordenadores o dispositivos móviles y deben tener un navegador para que el cliente realice las peticiones mediante el protocolo *HTTPS* al sistema alojado en el servidor de aplicaciones web.

Servidor web de aplicaciones: El servidor de aplicaciones web debe poseer como mínimo un CPU Core2Duo a 2.20 GHz con 2 GB de RAM y tener una capacidad mínima de almacenamiento de 5Gb.

Servidor de base de datos: debe poseer como mínimo un CPU Core2Duo a 2.20 GHz con 2 GB de RAM y tener una capacidad mínima de almacenamiento de 4Gb.

2.10 Conclusiones parciales

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

- Los requerimientos funcionales y no funcionales obtenidos a partir del proceso de identificación de los requisitos, sirvieron de guía para desarrollar las distintas funcionalidades y de este modo satisfacer las necesidades detectadas.
- Los artefactos generados según la metodología de desarrollo utilizada y los patrones de arquitectura y diseño descritos, constituyeron una guía fundamental para la construcción de la propuesta de solución.

Capítulo 3: Implementación y validación del Sistema de recomendación de costos

En el presente capítulo se presentan y describe los componentes y los estándares de codificación que sustentan la implementación del sistema de recomendación de costos. Se describe y fundamenta el proceso de validación de la solución implementada, mediante la utilización de los casos de pruebas, la herramienta de seguridad, las técnicas cuantitativas y cualitativas.

3.1 Diagrama de componentes

Los diagramas de componentes muestran los elementos de diseño de un sistema de software. Permiten visualizar con mayor facilidad la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de interfaces (Microsoft Developer Network, 2014).

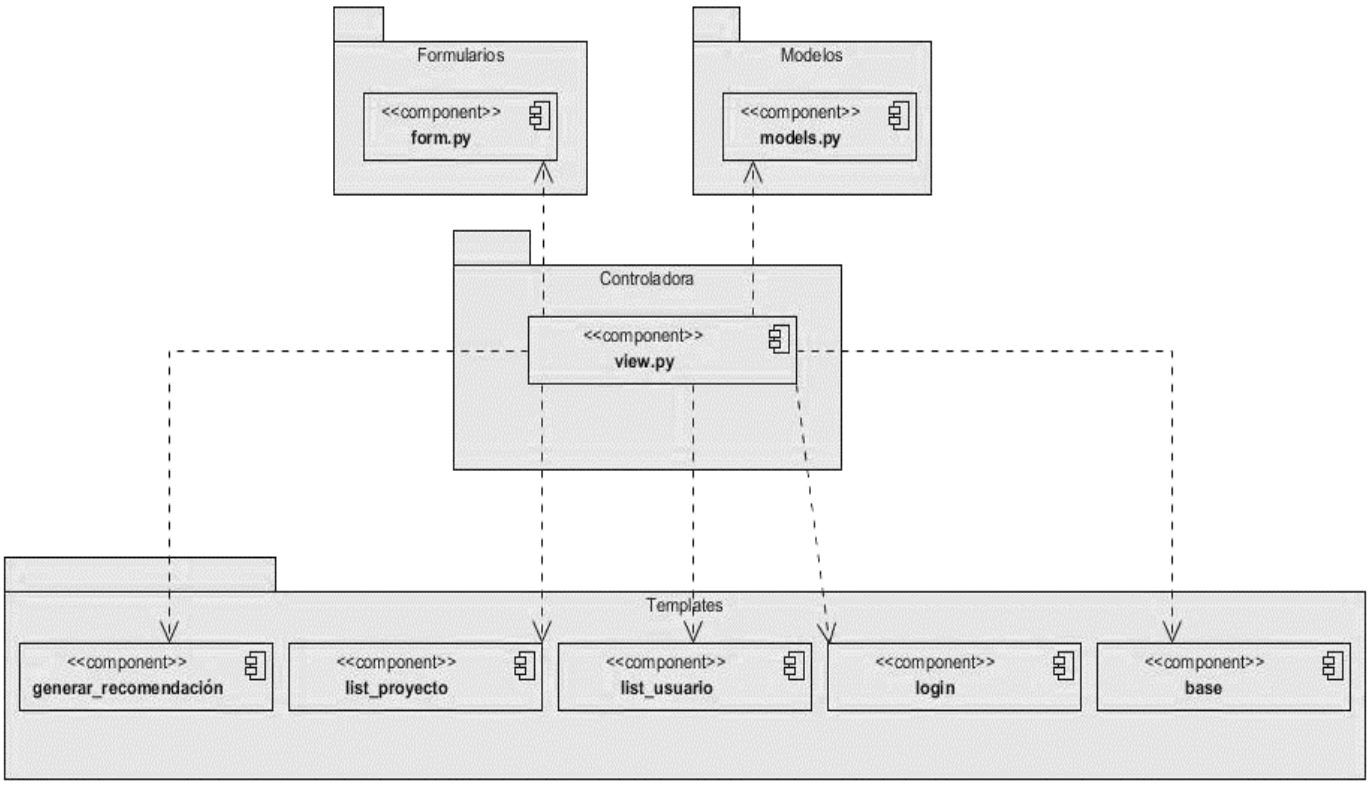

Figura 11. Diagrama de componentes.

A continuación, describe cada uno de los componentes representados en el diagrama:

- view.py: contiene las clases que controlan todo el flujo de los procesos de recomendación de costos, gestión de usuarios, gestión de proyectos y asignación de permisos.
- form.py: genera los formularios del sistema y permite el ingreso de datos para su procesamiento.
- models.py: contiene las clases que permiten la interacción con la base de datos para obtener y actualizar la información almacenada en esta.
- Paquete Templates: contiene todas las interfaces (base, login, generar_recomendación, list_usuario, list_proyecto, etc) que revelan los formularios y elementos para introducir la información necesaria para realizar las peticiones.

3.2 Estándares de codificación

Los estándares de codificación son un elemento fundamental en la implementación de proyectos. Garantizan que el código generado sea fácil de leer, entender y modificar independientemente del desarrollador. Son una guía para el equipo de desarrollo, permiten asegurar que el código presente alta calidad y no contenga errores (recursospython.com, 2012).

A continuación, se muestran los estándares utilizados:

- **Líneas en blanco**: se separan las funciones no anidadas y las definiciones de clases con dos líneas en blanco. Las definiciones de métodos dentro de una misma clase se separan con una línea en blanco.
- **Imports**
  - Normalmente los *imports* deberían colocarse en distintas líneas, por ejemplo:

```
import ListView
import DetailView
```

Figura 12. *Imports.*

  - Los *imports* se colocan siempre en la parte superior del archivo, justo después de cualquier comentario o cadena de documentación del módulo, y antes de las variables globales y las constantes del módulo.
  - Cuando se importe una clase de un módulo, normalmente se realiza.

```
from django.views.generic import ListView
from django.views.generic import DetailView
```

Figura 13. Importar una clase de un módulo.

- **Espacios en blanco en expresiones y sentencias**

Se evita espacios en blanco extra en las siguientes situaciones:

- Inmediatamente después de entrar en un paréntesis o antes de salir de un paréntesis, corchete o llave.
- Inmediatamente antes de una coma, punto y coma, o dos puntos.
- Inmediatamente antes de abrir un paréntesis para una lista de argumentos de una llamada a una función:
- Inmediatamente antes de abrir un paréntesis usado como índice o para particionar (*slicing*).
- Más de un espacio alrededor de un operador de asignación (u otro operador) para alinearlo con otro.

```
if form_generar.is_valid():
    tipo_Proyecto = request.POST['tipo_Proyecto']
    cant_Personas = int(request.POST['cant_Personas'])
    requisitos_Complejidad_Alta = int(request.POST['requisitos_Complejidad_Alta'])
    requisitos_Complejidad_Media = int(request.POST['requisitos_Complejidad_Media'])
    requisitos_Complejidad_Baja = int(request.POST['requisitos_Complejidad_Baja'])
    tiempo_realizacion = int(request.POST['tiempo_realizacion'])
    lista_proyectos = models.Proyecto.objects.filter(tipo_Proyecto=tipo_Proyecto).values()
```

Figura 14. Espacio alrededor de un operador de asignación.

- **Utiliza espacios alrededor de los operadores aritméticos:**

```
d1 = 0.2 * abs(cant_Personas - proyecto['cant_Personas'])
d2 = 0.1 * abs(tiempo_realizacion - proyecto['tiempo_realizacion'])
d3 = 0.3 * abs(requisitos_Complejidad_Alta - proyecto['requisitos_Complejidad_Alta'])
d4 = 0.2 * abs(requisitos_Complejidad_Media - proyecto['requisitos_Complejidad_Media'])
d5 = 0.2 * abs(requisitos_Complejidad_Baja - proyecto['requisitos_Complejidad_Baja'])
desimilitud = (d1 + d2 + d3 + d4 + d5)
```

Figura 15. Espacios alrededor de los operadores aritméticos.

- **Comentarios de bloque**

Los comentarios de bloque generalmente se aplican al código. Cada línea de un comentario de bloque comienza con el símbolo de # y un único espacio. Los párrafos dentro de un comentario de bloque se separan por una línea conteniendo un único símbolo de #.

- **Descriptivo: Estilos de Nombrado**
  - Minúsculas (para las funciones cuando su nombre es una sola palabra)
  - minúsculas_con_guiones_bajos (para las funciones: no anidadas o métodos de clases)
  - PalabrasEnMayúsculas (para las clases)
- **No comparar valores booleanos con True o False usando ==**

3.3 Validación de la propuesta de solución

Las pruebas de software (*Software Testing*) comprenden el conjunto de actividades que se realizan para identificar posibles fallos de funcionamiento, configuración o usabilidad de un programa o aplicación, por medio de pruebas sobre el comportamiento del mismo (PMOinformatica.com, 2017). Con el objetivo de comprobar la calidad del software se efectúan diferentes pruebas (funcionales y de seguridad) que miden el cumplimiento de los requisitos funcionales y no funcionales del sistema.

3.3.1 Funcionales

Las pruebas funcionales son realizadas en la interfaz de usuario y permiten valorar el funcionamiento de la aplicación según la interacción del usuario con el sistema. Estas pruebas tienen como objetivo, asegurar el funcionamiento apropiado de los requisitos funcionales, que incluyen la navegación, la entrada de datos, el procesamiento y obtención de resultados (Globe, 2017).

Tabla 4. Variables empleadas en el CU2 Autenticar usuario.

No.	Nombre del campo	Clasificación	Valor nulo	Descripción
1	Usuario	Campo texto	No	Se inserta una cadena de texto de no más de 15 caracteres, puede tener valores alfanuméricos, pero no caracteres extraños

2	Contraseña	Campo texto	No	Se inserta la contraseña de usuario, que no puede tener menos de 8 caracteres

Las celdas de la tabla contienen V (indica válido), I (indica inválido), N/S (No es necesario llenar).

Tabla 5. Descripción de prueba para el CU 2 Autenticar usuario.

Escenario	Descripción	V1	V2	Respuesta del sistema	Flujo central
EC 1.1 Autenticar usuario correctamente	El usuario introduce los datos correctamente	V	V	El sistema entra a la página inicio	1- El usuario accede al sistema 2- Llenar los campos correspondientes y seleccionar la opción entrar
		admin	admin1991		
EC 1.2 Campos vacíos	El usuario deja uno o más campos vacíos	V	I	Se muestra el mensaje: * Rellenar este campo	
		admin	Vacío		
		V	I		
		Vacío	admin1991		
		I	I		
		Vacío	Vacío		
EC 1.3 Autenticar usuario incorrectamente	El usuario introduce los datos incorrectamente	V	I	El sistema no deja entrar	
		admin	Ad1569		
		I	V		
		Admin_5	admin1991		
		I	I		
		Admin_5	Ad1569		

Tabla 6. Variables empleadas en el CU5 Obtener recomendación de costos.

No.	Nombre del campo	Clasificación	Valor nulo	Descripción

1	Tipo de proyecto	Botón de selección	No	Se selecciona el tipo de proyecto
2	Cantidad de requisitos	Campo texto	No	Se inserta la cantidad de requisitos que tendrá el proyecto, pueden ser solo números enteros
3	Cantidad de personas	Campo texto	No	Se inserta la cantidad de personas que tendrá el proyecto, pueden ser solo números enteros
4	Tiempo de realización	Campo texto	No	Se inserta el tiempo de realización en días del proyecto, pueden ser solo números enteros

Tabla 7. Descripción de prueba para el CU 5 Obtener recomendación de costos.

Escenario	**Descripción**	**V1**	**V2**	**V3**	**V4**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Obtener recomendación de costo	El usuario introduce los datos correctamente	V	V	V	V	Muestra los costos de proyectos similares en una nueva interfaz.	1- El usuario accede al menú lateral izquierdo. 2- Selecciona la opción Obtener recomendación. 3- Llenar los campos correspondientes y seleccionar la opción obtener.
		Desarrollo	15	4	60		
	El usuario deja uno o más campos vacíos	V	I	V	V	Se muestra el mensaje: * Rellenar este campo	
		Desarrollo	Vacío	4	60		
		I	V	V	V		
		Vacío	15	4	60		
		V	V	I	V		

		Desarrollo	15	Vacío	60		
		V	V	V	I		
		Desarrollo	15	4	Vacío		

Resultado de las pruebas funcionales

Después del análisis de los escenarios de pruebas se detectaron en las 3 iteraciones realizadas un total de 20 no conformidades. En la primera iteración se obtuvieron 15 no conformidades que fueron resueltas, la segunda iteración arrojó 5 no conformidades que fueron solucionadas y para una tercera iteración y final no se detectaron no conformidades. En la Figura 13 se muestran los resultados por cada iteración realizada al sistema.

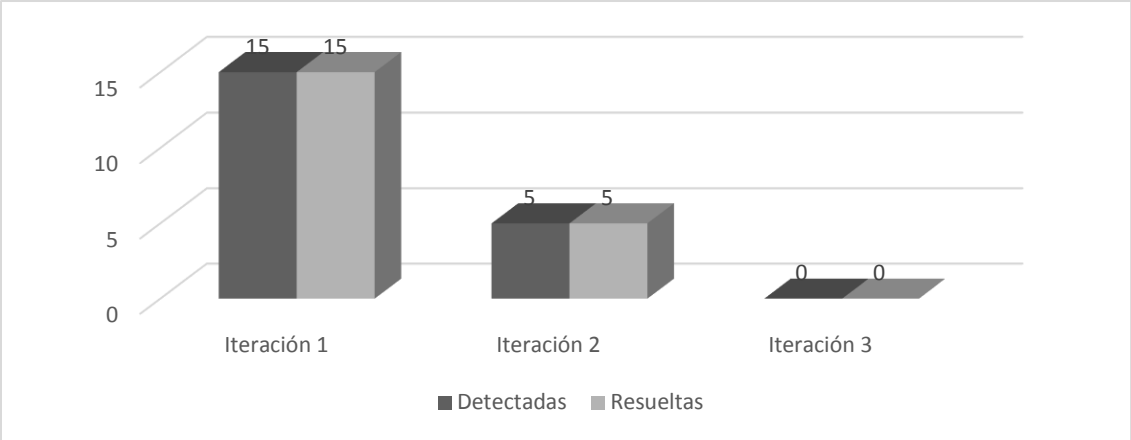

Figura 16. Resultado de las pruebas funcionales.

Las no conformidades identificadas durante el proceso de pruebas son:

- Los datos incorrectos introducidos por los usuarios son guardados en las bases de datos sin realizar una validación previa (cantidad 3). Esta no conformidad se obtenía por la validación de campos en blanco y fue solucionada al añadir una aserción para que estos campos no validaran y guardaran campos vacíos.
- Presencia de errores ortográficos (cantidad 15). Esta no conformidad fue resuelta al corregir los problemas de acentuación en las etiquetas de los campos y en las notificaciones de usuario.

- Los mensajes de error que devuelve el sistema no corresponden con los errores que ocurren (cantidad 2). La presencia de esta no conformidad es causada por la ausencia de notificaciones en los métodos de la clase controladora. Para solucionar este problema fueron incluidas las notificaciones faltantes en el código fuete.

3.3.2 Seguridad

Las pruebas de seguridad se aplican para garantizar la confidencialidad, integridad y disponibilidad de los datos. Constituye una de las principales pruebas que permiten identificar ataques de seguridad y posibles amenazas mediante vulnerabilidades existentes en el sistema (vyvquality.com, 2017).

Para evaluar la seguridad de la aplicación se empleó la herramienta *Acunetix Web Vulnerabitily Scanner 9.0*. Esta es una herramienta que permite escanear el sistema en busca de posibles fallos de seguridad que ponen en peligro su integridad. A continuación, se muestran las vulnerabilidades identificadas con el empleo de la herramienta seleccionada:

Tabla 8. Resultados de las pruebas de seguridad empleando Acunetix

Categoría de vulnerabilidades	Cantidad de errores	Nivel de riesgo	Problema	Solución
Vínculos rotos	2	Bajo	Los enlaces son inaccesibles.	Eliminar los enlaces rotos
Mensajes de error de la aplicación	2	Medio	Los mensajes de error pueden revelar información sensitiva como aspectos del código o configuración que pueden ser de utilidad a un atacante	No hay porque estos mensajes de error son un falso positivo debido a que se encuentran documentados
Total	4			

La prueba realizada mediante la herramienta Acunetix WVS v9 de nivel 2, de los 3 establecidos por los especialistas de seguridad de la Universidad de las Ciencias Informáticas, arrojó que en la primera iteración se detectaron 4 no conformidades, de ellas 2 alertas de riesgo medio, 2 de riesgo bajo y ninguna alerta de

riesgo alto. En la primera iteración se solucionan todos los problemas encontrados que pudieran comprometer la seguridad e integridad de la aplicación y en la segunda iteración no se detectaron no conformidades.

3.4 Validación de la hipótesis de la investigación

Hipótesis: El desarrollo de un sistema para la recomendación de costos permitirá mejorar la estimación de los costos de los proyectos de software.

Variable independiente: Sistema para la recomendación de costos.

Variable dependiente: Estimación de los costos de los proyectos de software.

Para la validación de la hipótesis de investigación se emplea el método criterio de expertos en su variante Delphi. Este método consiste en una técnica de obtención de información, basada en la consulta de expertos en un área de estudio, con el fin de obtener la opinión más fiable proveniente del conocimiento y la experiencia de los participantes del grupo (Sampieri, 2014; Reguant, 2016; Gestiopolis.com, 2018) mediante los siguientes pasos:

- Identificación de los posibles expertos
- Selección de los expertos
- Realización de la consulta a los expertos y procesamiento y valoración de la información obtenida

Para identificar los posibles expertos se consideró la experiencia profesional en relación a la gestión de los costos en proyectos de desarrollo. Con el objetivo de valorar el nivel de experiencia que poseen, se aplica un cuestionario de autovaloración (ver Anexo 5) de los niveles de información y argumentación que tienen sobre el tema en cuestión. En la tabla 8 se muestran los expertos seleccionados.

Tabla 9. Expertos en la validación de la propuesta de solución

No.	Experto	Entidad
1	Yasmani Joaquín Álvarez	Dirección de negocio
2	Ihoandra Sotolongo Carballo	Dirección de negocio
3	Adyana Pileta	CIDI
4	Yoandy Pérez Villazón	CESOL

5	Osay González Fuentes	CISED

Después de seleccionar los expertos, se sometió a su consideración un instrumento para la validación del sistema de recomendación de costos. El instrumento se compone de 5 sentencias (ver tabla 9), así como 5 categorías evaluativas que son: muy adecuado (MA), bastante adecuado (BA), adecuado (A), poco adecuado (PA) e inadecuado (I).

Tabla 10. Sentencias a evaluar por los expertos para validar la hipótesis

No.	Sentencias plasmadas en la consulta realizada a los expertos
1	¿Cómo se obtiene la información de los proyectos?
2	¿Cómo se procesa la información de los proyectos?
3	¿Cómo se realiza la estimación de costos de los proyectos?
4	Selección del tipo de sistema de recomendación
5	Selección del enfoque de la recomendación

Se calcula el coeficiente de Kendall para analizar la concordancia en las evaluaciones realizadas por los expertos (Sampieri, 2014). El coeficiente de concordancia de Kendall (W) será un índice de la divergencia del acuerdo efectivo entre los expertos, que fluctúa entre 0 y 1 (1 significa una concordancia de acuerdos total y el valor de 0 un desacuerdo total) y se obtiene de la expresión:

$$W = 12S/K2\ (N3 - N)_{(3)}$$

Donde las variables representan:

- S: cuadrado de las desviaciones medias
- K: número de expertos
- N: el número total de aspectos a evaluar

Se realiza también la Prueba de Significación de Hipótesis para comprobar el grado de significación de Kendall, planteándose la hipótesis nula (H0: no existe concordancia entre los expertos) y la alternativa (H1: existe concordancia entre los expertos). Para determinar que hipótesis es válida se efectúan los pasos siguientes:

1. Calcular X^2:

$X^2 = K(N-1)W$ (4)

X^2=0.352

2. Comparar el X^2 con el tabulado en la tabla del percentil de la distribución X^2:

   X^2 calculada < X^2 (α, N-1); se empleará α=0.05 para tener un 95% de confianza

   0.352<9,4877

Si se cumple que el resultado de X2 es menor que X2(α, N-1) entonces se valida la hipótesis alternativa H1, de que existe concordancia entre los expertos.

Los criterios aportados por los expertos se someten a una prueba estadística no paramétrica que determina la valoración final de cada aspecto a evaluar (Castro, 2014). En la tabla 10 se muestra distribución de frecuencia a partir de los datos primarios para cada uno de los aspectos sometidos a consulta.

Tabla 11. Distribución de frecuencia para los datos primarios obtenidos

Categorías evaluativas	Frecuencia absoluta	Frecuencia Relativa
Muy adecuado	21	0.84
Bastante adecuado	3	0.12
Adecuado	1	0.04
Poco adecuado	0	0
Inadecuado	0	0

A continuación, se puede observar en la figura 14 los resultados obtenidos en la validación.

Figura 17. Comportamiento de la valoración de los expertos por categorías evaluadas

Según los datos analizados el 84% de los aspectos fueron valorados de muy adecuado, el 12% de bastante adecuado y un 4% de adecuado. Además de los criterios sobre el sistema, los expertos recomendaron considerar la complejidad de los requisitos. Estos resultados favorables obtenidos de la consulta de expertos demostraron que existe una concordancia en las valoraciones realizadas sobre el alto valor y calidad que presenta el sistema de recomendación de costos. Los indicadores fueron evaluados satisfactoriamente lo que demuestra la validez de la hipótesis científica planteada en la presente investigación, evidenciándose la optimización la estimación de los costos de los proyectos de software.

3.5 Conclusiones parciales

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

- La elaboración de los diagramas y la descripción de los componentes que lo conforman, permitió una mejor comprensión de la estructura del sistema implementado.
- Aplicar los estándares de codificación permitió obtener en el sistema para la recomendación de costos un código legible, estándar y fácil de comprender.

- El proceso de validación de la solución propuesta permitió demostrar la calidad del sistema desarrollado.

Conclusiones

- La confección del marco teórico permitió identificar la necesidad de crear el sistema de recomendación de costos para los proyectos de software, determinando la selección de las herramientas, metodología y tecnologías viables a emplear en el desarrollo del sistema.
- El análisis de la descripción de la propuesta de solución, el modelo conceptual y la elaboración de los artefactos generados durante la fase análisis y diseño, permitieron la implementación de los requisitos de software identificados.
- Las pruebas funcionales y de seguridad permitieron solucionar los errores detectados en el sistema y demuestran que es una solución funcional y segura.
- El criterio de expertos demostró con un alto nivel de concordancia la valoración de muy adecuado que el sistema desarrollado mejora la estimación de los costos de los proyectos de software.

Recomendaciones

Luego de concluir la investigación y el desarrollo de la propuesta de solución, se recomienda para futuras versiones:

- Agregar en la obtención de la recomendación el costo en otras monedas.
- Considerar la experiencia del personal a cargo del proyecto para obtener la recomendación del costo.
- Considerar los recursos materiales utilizados en el desarrollo del proyecto para obtener la recomendación del costo.

Bibliografía

- acunetix. 2012**.** [En línea] [consulta: 7 enero 2018]. Disponible en: https://www.acunetix.com/wp-content/uploads/2012/10/wvsmanual.pdf.
- Aguilera, F. y Franco, J.C. Procedimiento para estimar el extractable del níquel con una red neuronal artificial, en sustitución de los modelos estadísticos tradicionales*.* Tesis doctoral. España: Instituto Universitario General Gutiérrez Mellado. 2015. [En línea] [consulta: 5 octubre 2017]. Disponible en: http://www.redalyc.org/html/1815/181517930009/.
- APACHE. What is the Apache HTTP Server Project?. Apache Software Foundation. 2017. [En línea] [consulta: 7 enero 2018]. Disponible en: https://httpd.apache.org/ABOUT_APACHE.html.
- Blázquez, M. Modelo entidad-relación ER. Fundamentos y Diseño de Bases de Datos, 2014. [En línea] [consulta: 15 de febrero del 2018]. Disponible en: http://ccdoc-basesdedatos.blogspot.com/2013/02/modelo-entidad-relacion-er.html
- Caro, M. Sistemas de Recomendación basados en técnicas de predicción de enlaces para jueces en línea. 2017.
- Castro, J. Un nuevo modelo ponderado para Sistemas de Recomendación Basados en Contenido con medidas de contingencia y entropía. 2012. [En línea] [consulta: 26 noviembre 2017]. Disponible en: http://sinbad2.ujaen.es/sites/default/files/publications/TTII_JorgeCastro.pdf
- Comité Central del PCC, 2016. Actualización de los lineamientos de la política económica y social del partido y la revolución para el periodo 2016-2021 aprobados en el 7mo. Congreso del Partido y por la Asamblea Nacional del Poder Popular. [en línea] [consulta: 5 octubre, 2017]. Disponible en: http://www.granma.cu/file/pdf/gaceta/01Folleto.Lineamientos-4.pdf
- Craig, L. UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado*.* Madrid: Pearson Educación, 2003.
- Diseño de datos . [En línea] [consulta: 21 enero, 2018.] https://msdn.microsoft.com/es-es/library/aa290752%28v=vs.71%29.aspx .
- Doria, J. y González, JC. Sistema de recomendación de problemas para el Juez en Línea Caribeño. Tesis de Diplomado. Universidad de las Ciencias Informáticas, La Habana, 2014.

- DSA. Distribuidora de Servicios Antivirus, S.L. [En línea] [consulta: 27 de marzo del 2018]. Disponible en: https://seaq.co/wp-content/uploads/2017/08/Acunetix-WVS-castellano.pdf
- Elementos de UML [En línea] [consulta: 15 de febrero del 2018]. Disponible en: https://docs.kde.org/stable4/es/kdesdk/umbrello/uml-elements.html.
- Globe. [En línea] [consulta: 27 de marzo del 2018]. Disponible en: https://www.globetesting.com/pruebas-funcionales/.
- Gómez, E., Sartorio, A. y Vaquero, M. EasyCard.js Framework. Tesis (Ingeniería en Sistemas Informáticos). Santa Fe: Universidad Abierta Interamericana. 2016.
- Guerrero, D. Estimación de costos. 2015. [En línea] [consulta: 5 octubre, 2017]. Disponible en: https://pirhua.udep.edu.pe/bitstream/handle/11042/2398/7.2%20Estimacion%20de%20costos.pdf?sequence=1
- Guía de estilo para el código Python - Recursos Python. [En línea] [consulta: 22 de marzo del 2018]. Disponible en: http://recursospython.com/pep8es.pdf.
- Ibañez, L.B. 2014. *Desarrollo del portal web Observatorio Político Cubano en Internet.* Pregrado. La Habana, Cuba: Universidad de las Ciencias Informaticas.
- JETBRAIN. Enjoy Productive PHP. 2017. [En línea] [consulta: 7 enero 2018]. Disponible en: https://www.jetbrains.com/phpstorm/.
- MICROSOFT DEVELOPER NETWORK 2014. Diagramas de componentes de UML: Referencia. [En línea] [consulta: 22 de marzo del 2018]. Disponible en: https://msdn.microsoft.com/es-es/library/dd409390.aspx.
- Mora, B; Ruiz, F; García, F y Piattini, F. Experiencia en transformación de modelos de procesos de negocios desde BPMN a XPDL. 2009. [En línea] [consulta: 24 febrero, 2018] Disponible en: http://www.researchgate.net/profile/Felix_Garcia2/publication/221561481_Experiencia_en_Transformacin_de_Modelos_de_Procesos_de_Negocios_desde_BPMN_a_XPDL/file/d912f50c71bd937e3d.pdf.
- Núñez, E. Sistemas de Recomendación de Contenidos para Libros Inteligentes. 2012.
- ORACLE. Getting Started with MySQL. 2017. [En línea] [consulta: 10 enero 2018]. Disponible en: https://dev.mysql.com/doc/mysql-getting-started/en/.

- Patrones de diseño: qué son y por qué debes usarlos. [en linea] [consulta: 26 febrero, 2018.] Disponible en: http://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos.
- PMBOOK. Guía de los fundamentos de la dirección de proyectos. Tercera edición. 2004. [En línea] [consulta: 24 febrero, 2018]
- PMOinformatica. La oficina de proyectos de informática. [En línea] [consulta: 27 de marzo del 2018]. Disponible en: http://www.pmoinformatica.com/p/pruebas-de-software.html.
- Pressman, R.S. Ingeniería del Software Un enfoque práctico: Quinta edición. 2002. [En línea] [consulta: 24 febrero, 2018] Disponible en: http://unpocodejava.wordpress.com/2013/05/09/tecnicas-para-la-captura-de-requisitos/.
- PyCharm: El mejor IDE para tus proyectos en Python. [En línea] [consulta: 10 de enero del 2018]. Disponible en: http://www.cristalab.com/tutoriales/pycharm-el-mejor-ide-para-tus-proyectos-en-python-c114084l/.
- Reguant, M. El método Delphi. Revista d'Innovació i Recerca en Educacio. 2016. [En línea] [consulta: 5 abril 2018].
- Rodríguez, H. y Rojas, P. Técnicas de estimación de costos para proyectos. 2015. [En línea] [consulta: 5 octubre, 2017]. Disponible en: http://repository.udistrital.edu.co/bitstream/11349/2874/1/Rodr%C3%ADguezGuti%C3%A9rrezHern%C3%A1nGuillermo2015.pdf
- Rodríguez, W. Portal Web de la Facultad 1 de la Universidad de las Ciencias Informáticas. Tesis de Diplomado. Universidad de las Ciencias Informáticas, La Habana, 2015.
- Viltres, H. Procedimiento para recomendar problemas en el Juez en Línea Caribeño. Tesis de Maestría. Universidad de las Ciencias Informáticas, La Habana, 2014.
- vyvquality. Quality performace, security and process solutions. [En línea] [consulta: 27 de marzo del 2018]. Disponible en: http://vyvquality.com/pruebas-seguridad/.
- Webera. 2017**.** [En línea] [consulta: 5 diciembre 2017]. Disponible en : https://www.lawebera.es/maquetacion-web/darle-uso-apropiado-css3.php#.

- Wong, L. Un modelo de Razonamiento Basado en Casos para la Captación de Requisitos en el desarrollo de proyectos de software. 2012.

Anexos

Anexo 1: Modelo de entrevista

Tabla 12: Plan de preguntas. Entrevista a los principales especialistas del proyecto.

No.	Preguntas
1	¿Cómo recuperan la información?
2	¿Cómo se realiza el proceso de estimación de los costos?
3	¿Qué herramienta o técnica utilizan para realizar la estimación de los costos en los proyectos de software?
4	¿Qué deficiencias presenta la herramienta o técnica empleada para realizar la estimación de los costos?

Anexo 2: Descripción de los casos de uso

Tabla 13. Descripción del CU5 Obtener recomendación de costos

Objetivo	Obtener recomendación de costos de proyectos similares	
Actores	Usuario básico	
Resumen	El CU se inicia cuando un usuario introduce los datos de un nuevo proyecto, el CU termina cuando se muestra el resultado de costos de proyectos similares al usuario.	
Complejidad	Alta	
Prioridad	Alta	
Precondiciones	Debe ser un usuario básico que esté operando en el sistema.	
Postcondiciones		
Flujo de eventos		
Flujo básico <Realizar recomendación de costos>		
	Actor	**Sistema**
1.	Introduce los datos de un nuevo proyecto.	
2.		Muestra el resultado de costos de proyectos similares en una nueva interfaz.
3.		Termina el CU.

Flujos alternos			
1. Los datos no son válidos.			
	Actor		**Sistema**
1.			Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
Relaciones	**CU Incluidos**		
	CU Extendidos		
Requisitos no funcionales			
Asuntos pendientes			

Tabla 14. Descripción del CU1 Asignar permisos

Objetivo	Asignar los permisos que posee el sistema para limitar el acceso a sus funcionalidades.
Actores	Administrador
Resumen	
Complejidad	Alta
Prioridad	Alta
Precondiciones	El usuario que debe estar autenticado con permisos de administrador
Postcondiciones	
Flujo de eventos	
Flujo básico <Asignar permisos>	

	Actor	**Sistema**
1.	El usuario selecciona personas/permisos	El sistema muestra el listado de permisos existentes.

2.	El usuario asigna los permisos deseados.	El sistema almacena los cambios realizados en los permisos. El sistema muestra un mensaje de notificación de cambios.
3.	Termina el CU.	
Flujos alternos		
1. Los datos no son válidos.		
	Actor	**Sistema**
2.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.

Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes		

Tabla 15. Descripción del CU2 Autenticar usuario

Objetivo	Autenticar usuario en el sistema
Actores	Usuario anónimo
Resumen	El CU se inicia cuando un usuario intenta autenticarse en el sistema introduciendo sus datos, el CU termina cuando el usuario es autenticado.
Complejidad	Media
Prioridad	Alta
Precondiciones	El usuario debe ser anónimo.
Postcondiciones	El usuario es redireccionado hacia la interfaz correspondiente a su rol
Flujo de eventos	
Flujo básico <Autenticar usuario>	

	Actor	Sistema
1.	Accede al sistema	
2.		Muestra la interfaz de autenticación.
3.	Introduce los datos.	
4.		Comprueba que los datos cumplan con los requisitos necesarios para que sea válido.
5.		Comprueba los datos del usuario con la base de datos.
6.		El usuario es autenticado y redireccionado.
7.		Termina el CU.
Flujos alternos		
1. Los datos no son válidos.		
	Actor	**Sistema**
3.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.

Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales		
Asuntos pendientes		

Tabla 16. Descripción del CU3 Gestionar usuarios

Objetivo	Crear, eliminar, modificar, mostrar usuario y listar usuarios
Actores	Administrador
Resumen	Este CU describe cómo crear, eliminar, modificar, mostrar usuario y listar usuarios
Complejidad	Media

Prioridad	Media	
Precondiciones	Que el usuario esté autentificado en el sistema.	
Postcondiciones		
Flujo de eventos		
Flujo básico <Gestionar usuario>		
	Actor	**Sistema**
1.	Se autentica en el sistema.	
2.		Muestra una interfaz para que escoja la opción de: • Crear usuario • Eliminar usuario • Modificar usuario • Mostrar usuario • Listar usuarios
3.	Si se escoge la opción: • Crear usuario ir a la interfaz crear usuario • Eliminar usuario ir a la interfaz eliminar usuario. • Modificar ir a la interfaz modificar usuario. • Mostrar ir a la interfaz mostrar usuario. • Listar ir a la interfaz listar usuarios.	
Flujos alternos		
1. No se puede autenticar usuario		
	Actor	**Sistema**
4.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
• **Sección 2: "**Crear usuario**"**		
Flujo básico <Gestionar usuario: Crear usuario>		
	Actor	**Sistema**

1.	Selecciona la opción **“**Crear usuario**”.**	
2.		Muestra la interfaz crear usuario**.**
3.	Introduce los datos correspondientes.	
4.		Comprueba que los datos cumplan con los requisitos necesarios para que sea válido.
5.		Comprueba que no existan los datos en la base de datos.
Flujos alternos		
4. Los datos no son válidos.		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
Flujos alternos		
5. Los datos existen.		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
• **Sección 3: “**Eliminar usuario**”**		
Flujo básico < Gestionar usuario: Eliminar usuario >		
	Actor	**Sistema**
1.	Selecciona la opción **“**Eliminar usuario**”.**	
2.		Muestra la interfaz eliminar usuario.
3.	Selecciona usuario a eliminar y hace clic en el botón “Eliminar”.	
4.		Busca usuario seleccionado en la base de datos y lo elimina.
Flujos alternos		
3. Hace clic en el botón “Eliminar” sin haber seleccionado un usuario.		
	Actor	**Sistema**

1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
• **Sección 4: "**Modificar usuario**"**		
Flujo básico < Gestionar usuario: Modificar usuario>		
	Actor	**Sistema**
1.	Selecciona la opción **"**Modificar usuario**".**	
2.		Muestra la interfaz modificar usuario.
3.	Introduce los datos correspondientes.	
4.		Busca usuario seleccionado en la base de datos y lo modifica.
Flujos alternos		
3. Hace clic en el botón "Modificar" sin haber seleccionado un usuario.		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
• **Sección 5: "**Mostrar usuario**"**		
Flujo básico < Gestionar usuario: Mostrar usuario >		
	Actor	**Sistema**
1.	Selecciona la opción **"**Mostrar usuario**".**	
2.		Muestra la interfaz mostrar usuario.
Flujos alternos		
3. Hace clic en el botón "Mostrar" sin haber seleccionado un usuario.		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
• **Sección 6: "**Listar usuarios**"**		
Flujo básico < Gestionar usuario: Listar usuarios >		
	Actor	**Sistema**

1.	Selecciona la opción **“**Listar usuarios**”.**	
2.		Muestra la interfaz listar usuarios.
Flujos alternos		
5. Los datos no son válidos		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.

Tabla 17. Descripción del CU4 Gestionar proyectos

Objetivo	Crear, eliminar, modificar, mostrar usuario y listar proyectos	
Actores	Usuario básico	
Resumen	Este CU describe cómo crear, eliminar, modificar, mostrar proyecto y listar proyectos	
Complejidad	Media	
Prioridad	Media	
Precondiciones		
Postcondiciones		
Flujo de eventos		
Flujo básico <Gestionar proyectos >		
	Actor	**Sistema**
1.	Se autentica en el sistema.	
2.		Muestra una interfaz para que escoja la opción de: • Crear proyecto • Eliminar proyecto • Modificar proyecto • Mostrar proyecto • Listar proyectos
3.	Si se escoge la opción:	

	• Crear usuario ir a la interfaz crear proyecto • Eliminar usuario ir a la interfaz eliminar proyecto. • Modificar ir a la interfaz modificar usuario. • Mostrar ir a la interfaz mostrar proyecto. • Listar ir a la interfaz listar proyectos.	

Flujos alternos		
1. Los datos no son válidos		
	Actor	**Sistema**
6.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.

- **Sección 2: "**Crear proyecto**"**

Flujo básico <Gestionar usuario: Crear proyecto >		
	Actor	**Sistema**
1.	Selecciona la opción **"**Crear proyecto**".**	
2.		Muestra la interfaz crear proyecto**.**
3.	Introduce los datos correspondientes.	
4.		Comprueba que los datos cumplan con los requisitos necesarios para que sea válido.
5.		Comprueba que no existan los datos en la base de datos.
Flujos alternos		
4. Los datos no son válidos.		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al proyecto las causas del mismo.
Flujos alternos		

5. Los datos existen.		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al proyecto las causas del mismo.
• **Sección 3: "**Eliminar proyecto**"**		
Flujo básico < Gestionar usuario: Eliminar proyecto >		
	Actor	**Sistema**
1.	Selecciona la opción **"**Eliminar proyecto**".**	
2.		Muestra la interfaz eliminar proyecto.
3.	Selecciona usuario a proyecto y hace clic en el botón "Eliminar".	
4.		Busca proyecto seleccionado en la base de datos y lo elimina.
Flujos alternos		
3. Hace clic en el botón "Eliminar" sin haber seleccionado un proyecto.		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
• **Sección 4: "**Modificar proyecto**"**		
Flujo básico < Gestionar usuario: Modificar proyecto >		
	Actor	**Sistema**
1.	Selecciona la opción **"**Modificar proyecto**".**	
2.		Muestra la interfaz modificar proyecto.
3.	Introduce los datos correspondientes.	
4.		Busca proyecto seleccionado en la base de datos y lo modifica.
Flujos alternos		
3. Hace clic en el botón "Modificar" sin haber seleccionado un proyecto.		

	Actor	Sistema
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
• **Sección 5: "**Mostrar proyecto**"**		
Flujo básico < Gestionar usuario: Mostrar proyecto >		
	Actor	**Sistema**
1.	Selecciona la opción **"**Mostrar proyecto**".**	
2.		Muestra la interfaz mostrar proyecto.
Flujos alternos		
3. Hace clic en el botón "Mostrar" sin haber seleccionado un proyecto.		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.
• **Sección 6: "**Listar proyectos**"**		
Flujo básico < Gestionar usuario: Listar proyectos>		
	Actor	**Sistema**
1.	Selecciona la opción **"**Listar proyectos**".**	
2.		Muestra la interfaz listar proyectos.
Flujos alternos		
7. Los datos no son válidos		
	Actor	**Sistema**
1.		Muestra un mensaje de error donde se le comunica al usuario las causas del mismo.

Anexo 3: Diagramas de clase de diseño

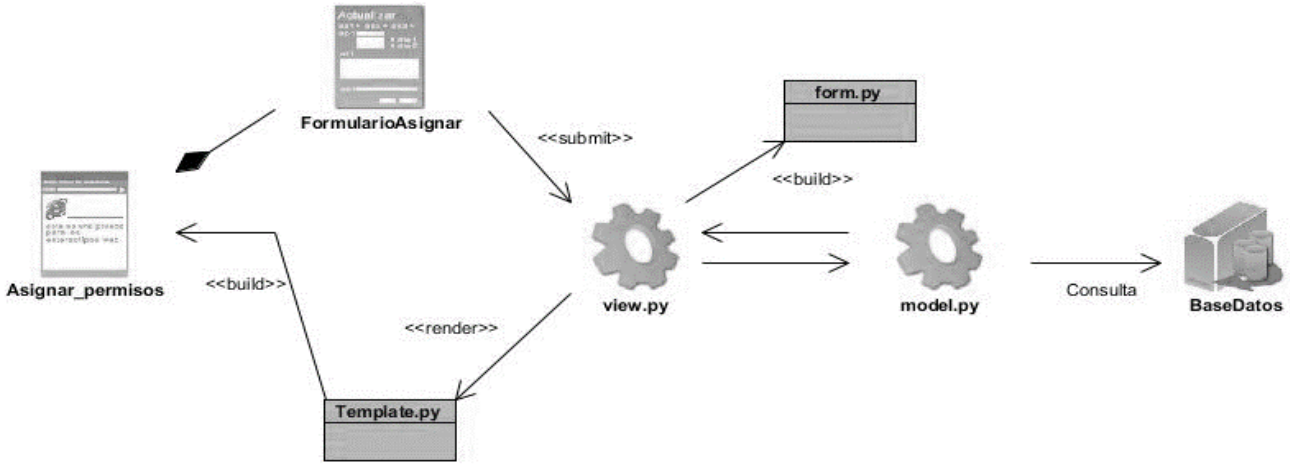

Figura 18. Diagrama de clase de diseño del CU1 Asignar permisos

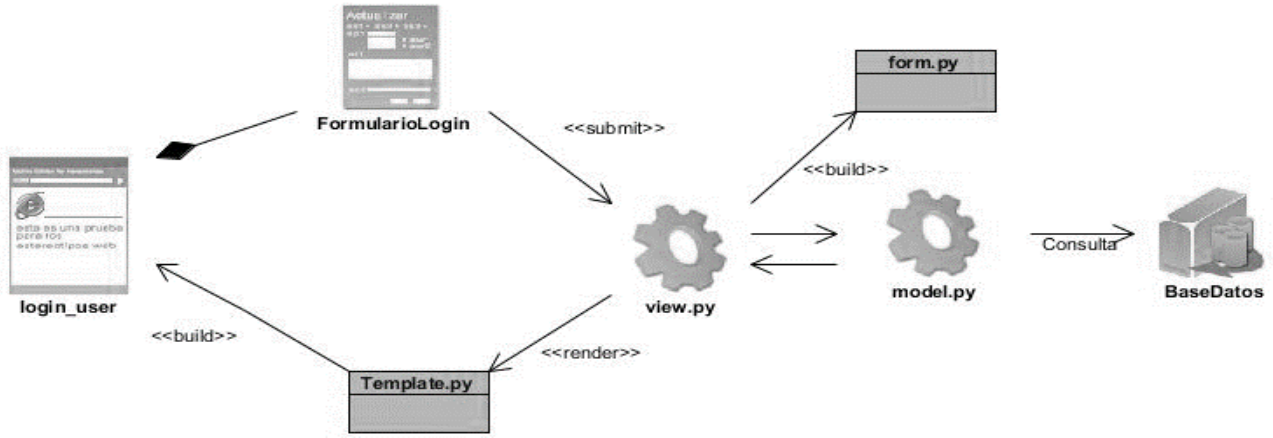

Figura 19. Diagrama de clase de diseño del CU2 Autenticar usuario

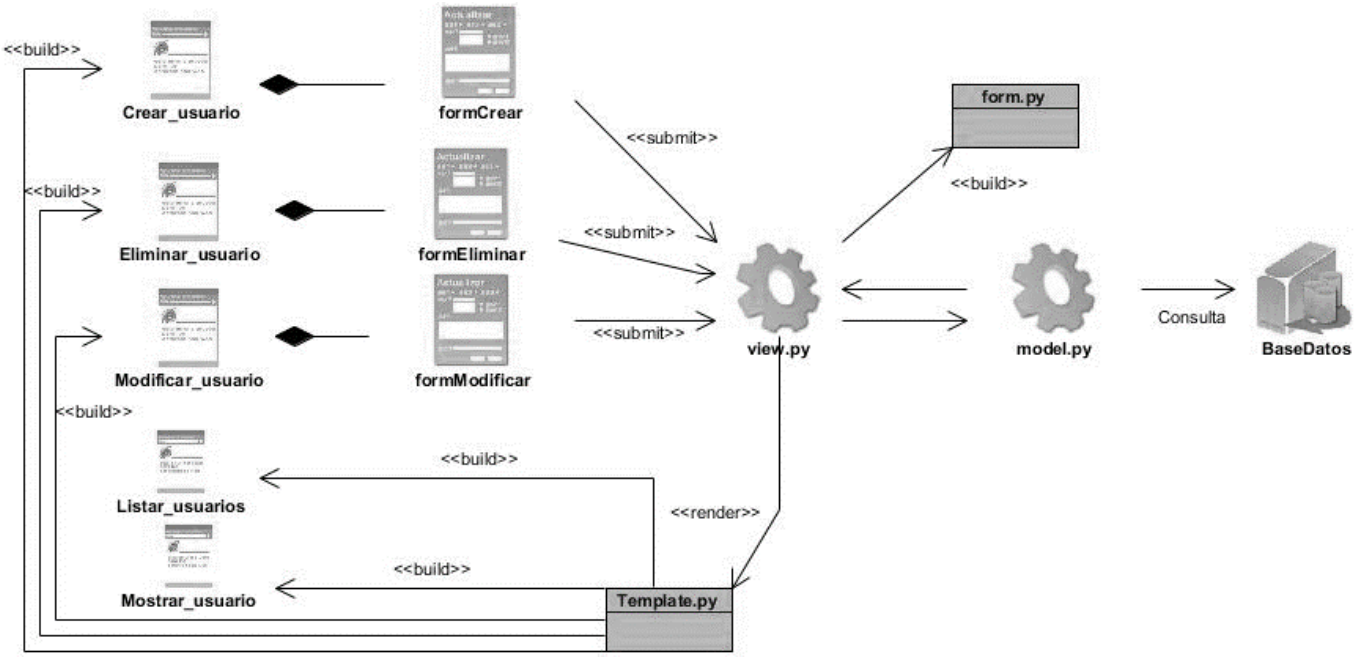

Figura 20. Diagrama de clase de diseño del CU3 Gestionar usuarios

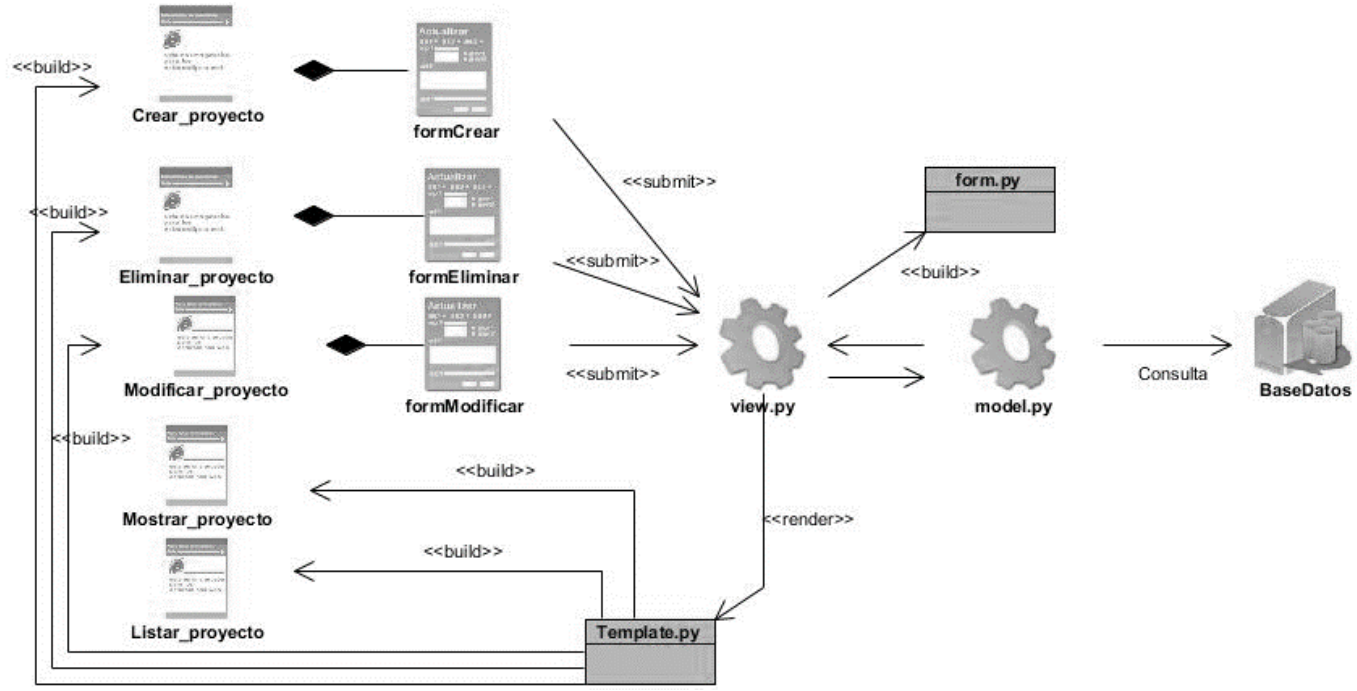

Figura 21. Diagrama de clase de diseño del CU4 Gestionar proyectos

Anexo 4: Diagramas de secuencia

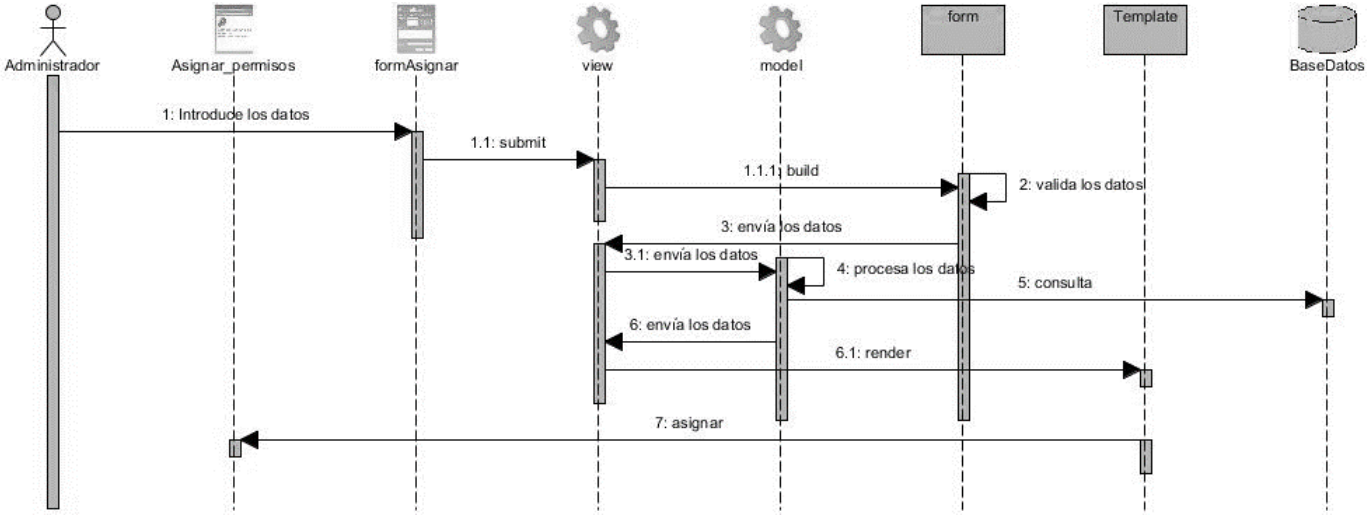

Figura 22. Diagrama de secuencia del CU1 Asignar permisos

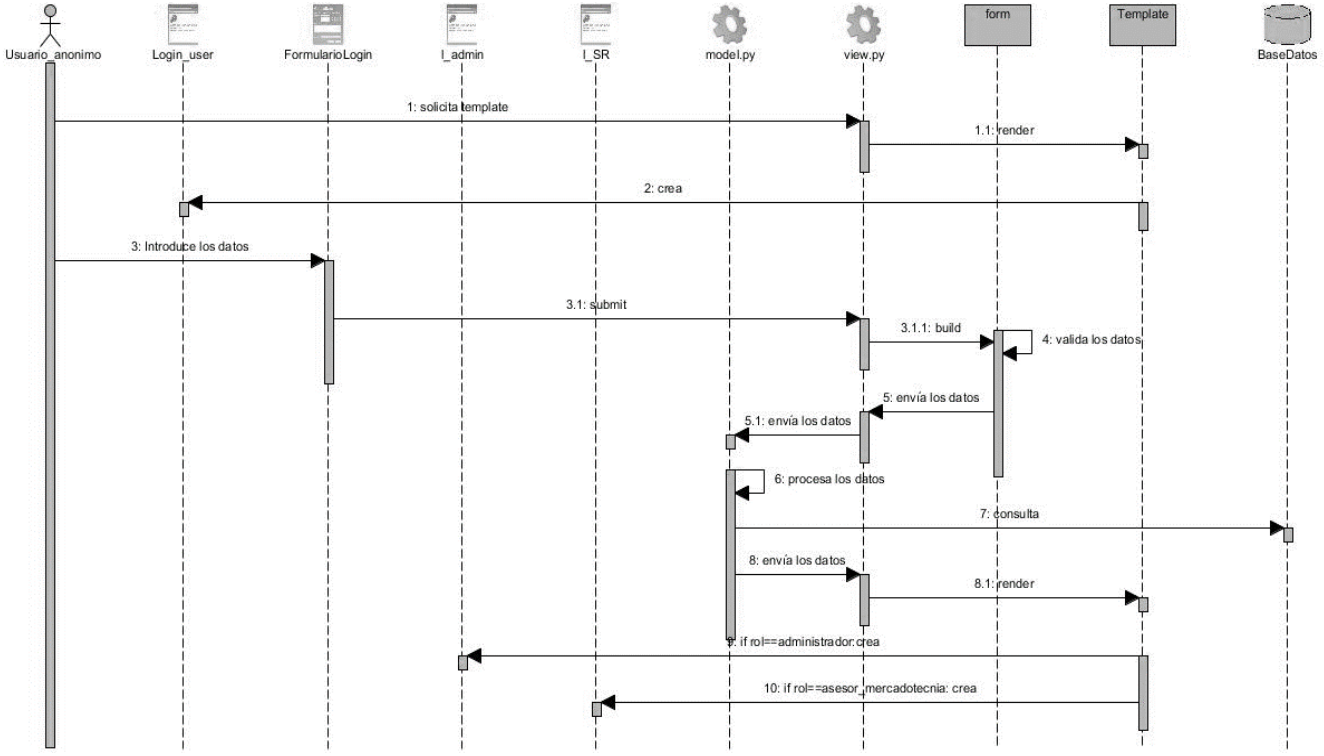

Figura 23. Diagrama de secuencia del CU2 Autenticar usuario

Anexo 5: Encuesta para la identificación de posibles expertos

Este material constituye un instrumento para la investigación del trabajo de diploma en opción al título de Ingeniero en Ciencias Informáticas: **Sistema para la recomendación de costos en proyectos de desarrollo de software** de la autora: **Sandra Jerónimo Casal**, para el Centro de Ideoinformática (CIDI) de la Facultad 1. Usted ha sido identificado como posible experto en el tema y su colaboración es importante para la investigación.

1.- Nombre y Apellidos: _____

2.- Cargo que ocupa: _____

3.- Años de experiencia: ___

4.- Marque con una cruz (X):

Nivel Escolar: ___ Técnico Medio ___ Enseñanza Media Superior ___ Enseñanza Superior

Grado Científico: ___ Master en Ciencias ___ Doctor en Ciencias ___ Ninguno

5.- Valore en la siguiente tabla con una escala del 1 al 10, el grado de conocimiento que usted posee sobre cada uno de los temas. Considere que la escala es ascendente donde el 1 es que no posee ningún conocimiento y 10 posee el máximo conocimiento del tema:

Tema	Escala de conocimiento
Procesamiento y recuperación de la información	
Estimación de los costos de los proyectos de software	
Técnicas y herramientas para la estimación de los costos	

6.- Realice una autoevaluación del grado de influencia que cada una de las fuentes, que se le presenta a continuación, ha tenido en su conocimiento y criterio para la estimación de los costos de los proyectos de software. Para ello marque con una cruz (X) según corresponda:

Fuentes de Argumentación	Alto	Medio	Bajo
Análisis teóricos realizados por usted			
Experiencia obtenida			

Trabajos de autores nacionales			
Trabajos de autores extranjeros			
Su conocimiento del estado del problema en el extranjero			
Su Intuición			

Gracias por su colaboración.

Anexo 6: Casos de prueba

Tabla 18. Descripción de prueba para el CU4 Gestionar proyecto.

Escenario	Descripción	Nombre del proyecto	Descripción del proyecto	Tipo de proyecto	Cantidad de requisitos	Cantidad de personas	Tiempo de realización	Costo	Respuesta del sistema	Flujo central
EC 1.1 Insertar proyecto	El sistema permite insertar un proyecto.	Portal UCI	El Portal UCI es para mostrar la información y las noticias de la universidad	Desarrollo	16	5	64	200	Se muestra los datos del nuevo proyecto en una nueva interfaz.	1-El usuario accede a Gestionar proyectos que se encuentra en el menú lateral izquierdo. 2- Seleccionar la
		Portal UCI	El Portal UCI es para mostrar la información y las	Desarrollo	16	5		200	Se muestra el mensaje: * El campo de tiempo	

			noticias de la universidad						de realización es obligatorio.	opción añadir proyecto. 3-Llenar los campos correspondientes y seleccionar la opción agregar.
		Portal UCI	El Portal UCI es para mostrar la información y las noticias de la universidad	Desarrollo		5	64	200	Se muestra el mensaje: * El campo cantidad de requisitos es obligatorio.	
			El Portal UCI es para mostrar la información y las	Desarrollo	16	5	64	200	Se muestra el mensaje: * El campo nombre	

			noticias de la universidad						del proyecto es obligatorio.	
		Portal UCI	El Portal UCI es para mostrar la información y las noticias de la universidad	Desarrollo	16		64	200	Se muestra el mensaje: * El campo cantidad de personas es obligatorio.	

		Portal UCI	El Portal UCI es para mostrar la información y las noticias de la universidad.	Desarrollo	16	5	64		Se muestra el mensaje: * El campo costo es obligatorio.	
		Portal UCI		Desarrollo	16	5	64	200	Se muestra el mensaje: * El campo descripción es obligatorio.	
		Portal UCI	El Portal UCI es para mostrar la información		16	5	64	200	Se muestra el mensaje: * El campo	

			y las noticias de la universidad.						tipo de proyecto es obligatorio.	
EC 1.2 Modificar proyecto	El sistema debe permitir modificar un proyecto.				20		75	214	Se muestra los datos del proyecto en una nueva interfaz.	1-El usuario accede a Gestionar proyectos que se encuentra en el menú lateral izquierdo. 2-Seleccionar la opción

										lista de proyectos. 3-Seleccionar del listado de proyectos existentes el proyecto que desea modificar. 4-Llenar los campos correspondientes y selecciona la opción actualizar.

EC 1.3 Eliminar proyecto	El sistema debe permitir eliminar un proyecto				N/A		N/A		El sistema muestra el mensaje proyecto eliminado	1-El usuario accede a Gestionar proyectos que se encuentra en el menú lateral izquierdo. 2-Seleccionar la opción lista de proyectos. 3-Seleccionar del listado de

										proyectos existentes el proyecto que desea eliminar. 4- Seleccionar la opción confirmar
EC 1.4 Mostrar proyecto	El sistema debe permitir mostrar los datos de un proyecto.				N/A		N/A		Se muestran todos los proyectos existentes en el sistema	1-El usuario accede a Gestionar proyectos que se encuentra en el menú

										lateral izquierdo. 2- Seleccionar la opción lista de proyectos. 3- Seleccionar del listado de proyectos existentes el proyecto que desea ver.
EC 1.5 Listar proyectos	El sistema debe permitir				N/A		N/A		Se muestran todos los	1-El usuario accede a

	mostrar los datos de todos los proyectos.								proyectos existentes en el sistema	Gestionar proyectos que se encuentra en el menú lateral izquierdo. 2- Seleccionar la opción lista de proyectos.