Universidad de las Ciencias Informáticas

Facultad 7

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

Título: Herramienta de apoyo al proceso Administración de Requisitos

Autores: Lyvan Bacallao Matos

Aliesky Lino Alonso

Tutoras: Ing. Darling Darías Pérez

Ing. Kenia Fernández Parra

La Habana, junio de 2011

"Año 53 de la Revolución"

Datos de Contactos

Ing. Kenia Fernández Parra graduada en el año 2007 de Ingeniera en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Profesora de la Facultad # 7. Ha impartido las asignaturas Matemática Discreta, Gestión de Software, Sistemas de Bases de Datos, Ética Informática y Metodología de la Investigación Científica (MIC). Se desempeñó por un período de 6 meses como líder del área temática productiva Gestión Hospitalaria (GEHOS), actualmente se desempeña como analista en la misma.

Correo electrónico: kfernandezp@uci.cu

Ing. Darling Darías Pérez graduada de Ingeniera en Ciencias Informáticas en la UCI en el curso 2008. Ha impartido las asignaturas de Gestión de Software, Sistemas de Bases de Datos y Preparación para las Pruebas de Nivel de Ingeniería de Software. Perteneció a la Subdirección de Gestión de la Calidad como administradora de la misma. Actualmente es subdirectora de la Subdirección de Investigaciones y Postgrado del CESIM.

Correo electrónico: ddarias@uci.cu

Dedicatoria

De Aliesky:

Dedico este trabajo de diploma a mi madre Gladys Alonso por apoyarme a lo largo de todos estos años, a mi novia Zenia por ser mi brazo derecho y a toda mi familia.

De Lyvan:

Dedico este trabajo de diploma a mi Creador Jesucristo, a mis padres Juan Enrique y Mara Matos. A mis abuelos que me han servido de apoyo incondicional siempre. A toda mi familia y a todos los que me aman.

Agradecimientos

De Aliesky:

A mi mamá y mi padrastro por apoyarme a hacer mi sueño realidad, a mi novia por ser lo mejor que me ha pasado a lo largo de la carrera, a mis amigos y amistades, en especial al kuko por estar siempre ahí cuando lo necesitaba, a mi compañero de tesis por siempre darme ánimo para salir adelante y a la revolución por haberme dado la oportunidad de estudiar en la UCI y poder graduarme como teniente de las FAR.

De Lyvan:

Muy especial a mis padres y mi familia en general por su preocupación por mí, por brindarme todo su amor, dedicación y apoyo incondicional durante toda mi vida. A mi abuelo Juan Matos por que fue un ejemplo para mí en todo y me ayudo mucho al guiarme por el camino de Dios. A mi amigo y padre Dios, que bueno tenerlo conmigo siempre .A mi compañero de tesis por su dedicación, ayuda. A mis hermanos en la fe por ser un ejemplo para mí en todo. A toda mi familia por su apoyo durante estos 5 largos años. A mis compañeros de aula y de proyecto, por toda la ayuda brindada y por compartir conmigo todo este tiempo. A todas las personas que de una forma u otra han contribuido a la realización de este trabajo.

A todos MUCHAS GRACIAS.

Resumen

En la actualidad han sido definidos un conjunto de procesos asociados a las áreas de procesos del nivel II de CMMI en la Universidad de las Ciencias Informáticas, como parte del programa de mejora. Uno de estos procesos, el IPP-3510 Administración de Requisitos, no cuenta con una herramienta que maneje los artefactos que genera. El objetivo de este trabajo es implementar una herramienta que apoye la creación de los artefactos que se generan durante este proceso.

La herramienta obtenida está implementada sobre tecnología .NET, utilizando como metodología de desarrollo RUP, como lenguaje de programación orientado a objetos C#, Visual Studio 2008 como IDE de desarrollo, PostgreSQL 8.3 como Sistema Gestor de bases de datos y Enterprise Architect 7.1 como herramienta CASE.

Con el desarrollo de esta herramienta web y su incorporación al proceso IPP-3510 Administración de Requisitos se pretende agilizar dicho proceso. Esto permitirá minimizar la carga de trabajo que realizan los analistas de los equipos de desarrollo de software. Además se contará con una base de datos centralizada en todos los departamentos del CESIM, donde se guardará toda la información de los proveedores de requisitos, de las evaluaciones de los casos de uso, de los requisitos del cliente y de los requisitos producto.

Palabras claves: requisitos, software, procesos, administración de requisitos, herramienta.

Índice

Introducción 1

Capítulo 1: Fundamentación Teórica 6

1.1 Conceptos generales 6

1.1.1 Ingeniería de Requisitos del Software 6

1.1.2 Requisitos 6

1.2 Conceptos relacionados con el dominio del problema 7

1.2.1 Proceso 7

1.2.2 Administración de Requisitos 7

1.2.3 Calidad 7

1.2.4 Capability Maturity Model Integration (CMMI) 8

1.3 Antecedentes del sistema 8

1.3.1 Antecedentes Internacionales 8

1.3.2 Antecedentes de ámbito nacional 14

1.4 Análisis crítico 14

1.5 Herramientas y tecnologías a utilizar 15

1.5.1 Lenguajes 15

1.5.2 Metodologías 18

1.5.3 Herramientas 19

Capítulo 2: Características del Sistema 22

2.1 Objetos de automatización 22

2.2 Información que se maneja 22

2.3 Descripción del Sistema 22

2.4 Modelo de Dominio 23

2.5 Conceptos Fundamentales del dominio 24

2.6 Diagrama de Modelo de Dominio 24

2.7 Especificación de los requisitos de software 24

2.7.1 Requisitos funcionales 24

2.7.2 Requisitos no funcionales 26

2.8 Diagrama de Casos de Uso del Sistema 28

2.9 Actores del Sistema 28
2.10 Especificación de los Casos de Uso 29
2.10.1 Descripción del caso de uso: Autenticar 29
2.10.2 Descripción del caso de uso: Crear catálogo de proveedores de requisitos 31
2.10.3 Descripción del caso de uso: Crear proveedores de requisitos válidos 33
2.10.4 Descripción del caso de uso: Crear matriz de disponibilidad de proveedores válidos 35
2.10.5 Descripción del caso de uso: Crear criterios para validar requisitos del cliente 37
2.10.6 Descripción del caso de uso: Crear criterios para validar requisitos del producto 40
2.10.7 Descripción del caso de uso: Crear catálogo de casos de uso 42
2.10.8 Descripción del caso de uso: Crear criterios de prioridad para validar casos de uso 43
Capítulo 3: Diseño del Sistema 46
3.1 Descripción de la arquitectura 46
3.2 Modelo de diseño 47
3.3 Diagramas de clases del diseño 47
3.3.1 CUS Crear criterios para validar requisitos del producto 48
3.3.2 CUS Crear criterios para validar requisitos del cliente 48
3.3.3 CUS Crear catálogo de proveedores de requisitos válidos 49
3.3.4 CUS Crear catálogo de casos de uso 49
3.3.5 CUS Crear matriz de disponibilidad de proveedores 50
3.4 Diagramas de secuencia 50
3.4.1 CUS Crear catálogo de casos de uso 50
3.4.2 CUS Crear criterios para validar requisitos del cliente 51
3.5 Descripción de las clases 51
3.5.1 Clase Crear catálogo de proveedores de requisitos 51
3.5.2 Clase Crear catálogo de casos de uso 52
3.6 Modelo de Datos 52
Capítulo 4: Implementación y Prueba 54
4.1 Modelo de implementación 54
4.2 Diagrama de componentes 54
4.3 Diagrama de despliegue 57

4.4 Pruebas 58
4.5 Pruebas de Caja Negra 58
4.6 Casos de prueba (CP) 59
4.7 Diseño de CP para el CU Crear catálogo de proveedores de requisitos 59
4.8 Diseño de CP para el CU Ver detalles de proveedores de requisitos 60
Conclusiones 62
Recomendaciones 63
Referencias Bibliográficas 64
Bibliografía 67
Glosario de Términos 70

Introducción

La informática tiene como propósito convertirse en una de las ramas más productivas para Cuba. Para lograr esta meta es trascendental el lugar que ocupa la Universidad de las Ciencias Informáticas (UCI).Que tiene por objetivo convertirse una universidad de excelencia, como dijera el Comandante Fidel Castro Ruz; con la misión de perfeccionar y automatizar los procesos en diferentes sectores del país como la salud, la educación, el deporte, el turismo y la economía.

Su principal misión es ser una universidad innovadora de excelencia científica, académica y productiva que forme de manera continúa profesionales integrales comprometidos con la patria, soporte de la informatización del país y la competitividad internacional de la industria cubana del software. (1)

Sobre el objetivo de este centro Fidel Castro Díaz-Balart planteó:

“El propósito fundamental es lograr un centro de excelencia para la formación masiva de profesionales de nivel superior. Ello debe alcanzarse con la ejecución de ambiciosos programas curriculares y de producción y con la aplicación de las más modernas tecnologías en la docencia”. (2)

La Universidad de las Ciencias Informáticas (UCI), además de su perfil académico, encaminado a preparar profesionales revolucionarios en el campo de la informática, tiene como segundo perfil la investigación y producción de Software donde los estudiantes están vinculados a proyectos reales desde tercer año de la carrera. Por la magnitud que han alcanzado estos proyectos y el peso que tienen dentro de la economía del país es preciso contar con procesos bien definidos que ayuden a lograr una alta calidad en los productos desarrollados.

Fieles a su compromiso de innovación y mejora continua la UCI desde hace algún tiempo comenzó los trabajos para alcanzar el nivel II de CMMI, el cual es un modelo para la mejora de los procesos de desarrollo y mantenimiento de sistemas y clientes de software. CMMI se organiza en cinco niveles que garantizan la existencia de procesos descritos y específicos para los diferentes proyectos de la empresa, así como la transparencia en el seguimiento de los mismos y el cumplimiento de sus plazos. La obtención del nivel II de CMMI aportará a los proyectos desarrollados por la UCI y a sus clientes mayor efectividad en la detección temprana de errores y una reducción en las desviaciones

en plazo. Además, favorecerá un incremento en la capacidad de adopción y adaptación de nuevas tecnologías, mayor rapidez y efectividad de respuesta ante las exigencias del negocio y un resultado predecible para los proyectos. (3)

Como parte del programa de mejora en que está inmersa la UCI han sido definidos un conjunto de procesos asociados a las áreas del nivel II de CMMI. Uno de estos procesos lo constituye el IPP-3510 Administración de Requisitos.

Como proceso, la administración de requisitos es fundamental en todo proyecto de desarrollo de software, ya que se debe de contar con una especificación clara y completa desde las fases iniciales para no tener problemas posteriores que implicarían un retraso en el cronograma, un presupuesto erróneo, o hasta la posible cancelación del proyecto. Es importante que el documento que se obtenga de esta etapa sea un reflejo real del acuerdo de las partes involucradas.

Al ser los requisitos las capacidades que los usuarios plantean que necesitan que tenga un software. También podría ser para integrar un objetivo que debiera tener algún componente del software, todo para cumplir con las exigencias del contrato hecho entre los desarrolladores y los usuarios o clientes.

Uno de los procesos claves en el desarrollo de software es la gestión de requisitos, la cual garantiza que se cumplan las necesidades del cliente, a través de la búsqueda de la solución correcta y libre de ambigüedades, para finalmente transformarse en el sistema operacional que espera el cliente. El proceso de gestión de requisitos se describe en cinco pasos fundamentales: (4)

- Identificación de requisitos.
- Análisis de requisitos y negociación.
- Especificación de requisitos.
- Modelado del sistema.
- Validación de requisitos y gestión de requisitos.

Realizar una adecuada gestión de requisitos posibilita que el desarrollo del proyecto se tarde menos, se optimicen los recursos, se reutilicen los requerimientos y que el usuario final participe en todo el proceso de desarrollo.

Con la ejecución del programa de mejora y del proceso IPP-3510 Administración de Requisitos se ha provocado un incremento del número de artefactos antes generados durante el desarrollo del software, entre los que se encuentran los siguientes documentos: Evaluación de Casos de Uso, Criterios para definir proveedores válidos de requisitos, Matriz de disponibilidad de proveedores, Criterios para validar requisitos del cliente y Criterios para validar requisitos del producto.

Para llenar las plantillas que se generan, se realiza prácticamente de forma manual, provocándose un gran aumento en la carga de trabajo de todos los involucrados en este proceso, además se efectúan cálculos que dilatan tanto el cumplimiento de las tareas como la calidad de las mismas. Todo esto induce un atraso en la entrega de toda la documentación perteneciente al proceso IPP-3510 Administración de Requisitos, y por consiguiente tardará más tiempo en realizar el software en cuestión, por parte de los desarrolladores.

Con respecto a la situación existente en la Universidad de las Ciencias Informáticas específicamente en la facultad 7 se presenta el siguiente **problema a resolver:** ¿Cómo viabilizar la generación de los artefactos definidos en el proceso IPP-3510 Administración de Requisitos?

Se define como **objeto de estudio** el proceso de desarrollo de software en el CESIM y como **campo de acción** el proceso IPP-3510 Administración de Requisitos.

Como **objetivo general** de la investigación se plantea: desarrollar una aplicación web que apoye la generación de los artefactos definidos en el proceso IPP-3510 Administración de Requisitos durante el desarrollo de software en el CESIM.

Para lograr cubrir las necesidades planteadas se establecen las siguientes actividades como las **tareas de la investigación**:

1. Analizar el proceso IPP-3510 Administración de Requisitos.
2. Evaluar las tendencias actuales sobre herramientas de apoyo a la administración de requisitos.

3. Definir los conceptos que constituyen el modelo de dominio.
4. Determinar la especificación de los requerimientos funcionales y no funcionales.
5. Especificar los casos de usos de la propuesta.
6. Evaluar la arquitectura a utilizar para el desarrollo de la aplicación.
7. Especificar los diagramas de clases e interacción del diseño.
8. Realizar la descripción de las clases del diseño.
9. Obtener el modelo de datos y descripción de sus entidades.
10. Implementar las funcionalidades definidas.

Se pretende con el desarrollo de esta aplicación web que se apoye la ejecución del proceso IPP-3510 Administración de Requisitos alcanzando como **resultados**:

- La obtención de una aplicación web para apoyar el proceso de Administración de Requisitos que se realiza en el Centro de la Informática Médica de la UCI.
- Determinar si los requisitos de los clientes y de los productos son válidos.
- Evaluar los casos de uso por prioridad y por complejidad.
- Definir los proveedores válidos de requisitos de software.
- Minimizar la carga de trabajo que realizan los analistas de los equipos de desarrollo de software.
- Agilizar el proceso IPP-3510 Administración de Requisitos en los proyectos del CESIM.
- Contar con una base de datos centralizada en todos los departamentos del CESIM, donde se guarde toda la información de los proveedores de requisitos, de las evaluaciones de los casos de uso, de los requisitos del cliente y de los requisitos producto.

El presente trabajo de diploma está estructurado en cuatro capítulos, a continuación se brinda una breve descripción de cada uno de ellos:

Capítulo 1. Fundamentación teórica: Contendrá el estudio del estado del arte de las herramientas de administración de requisitos, abarcando la esfera internacional, nacional y de la universidad. Se tendrán en cuenta las tecnologías, las metodologías, las herramientas de software y el lenguaje de programación que se propone para dar solución al problema planteado, junto con la justificación de su uso, buscando obtener el cumplimiento de los objetivos propuestos.

Capítulo 2. Características del sistema: Se definen los actores, trabajadores y casos de uso de la herramienta. Además se realiza la descripción del modelo del dominio y detallan sus principales conceptos. También se determinan los requisitos funcionales y no funcionales.

Capítulo 3. Diseño del sistema: Se crean los artefactos del flujo de trabajo análisis y diseño, como los diagramas de clases del diseño, los diagramas de secuencia, la descripción de las clases y de la arquitectura a utilizar en la aplicación.

Capítulo 4: Implementación y Prueba: Se basa en la implementación de la aplicación. Aquí es donde se representa el modelo de implementación, además se especifica el diagrama de componentes y de despliegue.

Capítulo 1: Fundamentación Teórica

En este capítulo se abordarán los principales conceptos generales y del dominio como por ejemplo: Requisitos, Administración de Requisitos, CMMI y Calidad del Software. También se tratará el estado del arte de las herramientas de administración de requisitos tanto nacional como internacionalmente y se dará una descripción sobre las herramientas y tecnologías que se utilizarán en el desarrollo de la herramienta que se propone.

1.1 Conceptos generales

1.1.1 Ingeniería de Requisitos del Software

Es un proceso durante el cual se hace una compilación, análisis y verificación de las necesidades del cliente para un sistema. Cuya meta es entregar una especificación de requisitos de software correcta y completa.

"Ingeniería de Requerimientos ayuda a los ingenieros de software a entender mejor el problema en cuya solución trabajarán. Incluye el conjunto de tareas que conducen a comprender cuál será el impacto del software sobre el negocio, qué es lo que el cliente quiere y cómo interactuarán los usuarios finales con el software". (5)

Durante el proceso de Ingeniería de Requerimientos (IR) se realizan de forma cíclica las actividades de:

- Identificación y consenso.
- Análisis y negociación.
- Documentación (especificación).
- Validación de requisitos.

1.1.2 Requisitos

Es una condición o capacidad que un sistema o un componente de un sistema debe satisfacer o poseer de acuerdo con un contrato, estándar, especificación u otro documento impuesto formalmente.

- **Requisitos del Cliente**

Resultado de la obtención, consolidación y resolución de conflictos entre las necesidades, expectativas, limitantes e interfaces de los involucrados relevantes del cliente, presentados de forma que sea aceptable por el cliente.

Refinamiento de los requisitos del cliente al lenguaje de los desarrolladores, haciendo los requisitos implícitos en requisitos derivados explícitos. El desarrollador utiliza los requisitos del cliente para guiar el diseño y construir el cliente.

- **Requisitos del Producto**

Los requisitos del producto son la expresión técnica de las necesidades establecidas por el cliente en términos no técnicos y que son utilizadas como base para las decisiones relacionadas con el diseño y que implica cuestiones tecnológicas. (6)

1.2 Conceptos relacionados con el dominio del problema

1.2.1 Proceso

Es un conjunto de prácticas realizadas para obtener un resultado, incluye técnicas, materiales, herramientas y personas. (7)

1.2.2 Administración de Requisitos

Es la administración de todos los requisitos recibidos o generados por el proyecto, incluyendo tanto requisitos técnicos como no técnicos así como aquellos requisitos impuestos en el proyecto por la organización. (8)

1.2.3 Calidad

Conjunto de propiedades y características de un cliente o servicio, que le confieren aptitud para satisfacer una serie de necesidades explícitas o implícitas. (9)

1.2.4 Capability Maturity Model Integration (CMMI)

CMMI está entre los principales estándares de evaluación y mejora del proceso de desarrollo de software junto con ISO/IEC 15504. Con su utilización vincula una serie de prácticas y principios con la madurez de los procesos en una organización, con el fin de mejorarlos. (10)

- **Prácticas Específicas (SP):**

Los requisitos están presentes en muchas áreas de proceso, CMMI en el nivel II define un área expresamente relacionada con los requisitos: Gestión de Requisitos (GR, Requirements Management).Teniendo como objetivo específico (SG por sus siglas en inglés: SG1 Administrar los requisitos).

1.3 Antecedentes del sistema

1.3.1 Antecedentes Internacionales

Actualmente en el mundo de la producción de software la Ingeniería de Requisitos está muy presente, al cumplir un papel importante en el proceso de desarrollo de los mismos. La Administración de Requisitos es una de las actividades de vital importancia que ofrece la IR. Últimamente en el mercado de software se ha dado un gran avance en la implementación de herramientas de gestión de requisitos. Con el uso de estas herramientas se disminuye el trabajo en la administración de requisitos.

1.3.1.1 Open Source Requirement Management Tool (OSRMT)

Se trata de una herramienta de gestión de requisitos, que permite la descripción avanzada de diversos tipos de requisitos y garantiza la trazabilidad entre todos los documentos relacionados con la ingeniería de requisitos (funcionalidades, requisitos, casos de uso, casos de prueba).

Ventajas

- La visualización de requisitos en forma jerárquica es intuitiva y fácil de manejar.
- Existen diversas distribuciones, tanto para un equipo en local como para un servidor de aplicaciones J2EE para permitir desarrollo colaborativo.

- Su licencia es GPL.
- Es un desarrollo basado en Java, por lo que es multiplataforma.
- Las nuevas versiones incorporan un cliente Web para permitir accesos desde internet.
- Como herramienta Open Source de gestión de requisitos no tiene mucha competencia en cuanto a la funcionalidad ofrecida.
- Tiene una buena documentación pese a tratarse de una herramienta muy reciente.
- A pesar de ser llevada a cabo por un único desarrollador, el ritmo de mejoras y nuevas versiones es constante a lo largo del último año.
- Existen muchas opciones para configurar y personalizar la herramienta a las necesidades concretas de una organización.
- Lleva incorporado un sistema de gestión de la configuración que permite definir líneas base.
- Existe un gran soporte para mantener la trazabilidad entre los documentos.
- Existen mecanismos que facilitan la importación y exportación de la información en XML.

Desventajas

- Es un desarrollo llevado a cabo por una persona individual, por lo que existe el riesgo de que no sea sostenible a lo largo del tiempo.
- No existe un soporte empresarial.
- Las nuevas versiones no están planificadas ni se anuncian claramente las mejoras que serán incorporadas. Es posible que las nuevas versiones no sean compatibles con las anteriores.
- No es posible generar automáticamente un documento de requisitos para entregar al cliente.
- Algunas funcionalidades no han sido desarrolladas completamente y están a medias.
- La interfaz de usuario es en ocasiones lenta.

- Se ofrecen pocos mensajes de confirmación y aviso al usuario (la interacción con el usuario es pobre). (11)

1.3.1.2 Requisite Management (REM)

Es una herramienta experimental gratuita desarrollada por la Universidad de Sevilla para gestión de requisitos. Cuenta con características y funcionalidades básicas para realizar las siguientes tareas.

- Gestión de requisitos.
- Trazabilidad entre documentos.
- Generación de informes (sólo en HTML).

Ventajas

- La visualización de requisitos en forma jerárquica es intuitiva y fácil de manejar.
- A la vez que se introduce la información va generando un documento html.
- Tiene una interfaz muy intuitiva, por lo que la curva de aprendizaje es mínima.
- Su sencillez conceptual (plantillas y patrones lingüísticos).
- Es una muy buena herramienta para comenzar el hábito de realizar la gestión de requisitos en aquellas organizaciones que no estuvieran acostumbradas a hacerlo.
- Aunque no es Open Source, sí es de uso gratuito.
- Permite incluir trazabilidad entre los requisitos.
- Da soporte a dependencias entre requisitos.
- Almacena los autores de cada requisito.
- Es posible vincular la definición de los requisitos con reuniones o con solicitudes de cambios realizadas por el cliente.
- Dispone de una herramienta de análisis de impacto de cambios a los requisitos.

- En realidad almacena la información en una base de datos Access, por lo que es posible importar y exportar la información (aunque se hace preciso conocer en detalle el modelo de datos).

Desventajas

- Requiere de las ADO por lo que sólo funciona en entornos con el sistema operativo Windows.
- No es una herramienta Open Source y en principio el autor no plantea liberar su código.
- No se pueden generar líneas base.
- Tiene algunos errores.
- No tiene soporte (no es una herramienta comercial).
- No es eficiente (lenta cuando hay muchos objetos).
- La calidad del HTML generado es mejorable y el desarrollo de nuevas hojas de estilo XSLT requiere amplios conocimientos de XML, XSLT, HTML.
- El metamodelo interno no se puede cambiar (añadir nuevas propiedades a los objetos o nuevos tipos de objetos).
- El metamodelo de casos de uso dificulta la especificación de caminos alternativos en los casos de uso.
- No es multiusuario.
- No incorpora gestión de versiones.
- No es posible controlar la numeración de los requisitos.
- No se puede añadir formato al texto. (12)

1.3.1.3 Rational RequisitePro

Es una solución fácil de usar, es una herramienta de administración de requerimientos que le permite al equipo crear y compartir sus requerimientos utilizando métodos familiares basados en documentos potenciados por la aplicación de las capacidades de una base de datos, tales como la trazabilidad y

análisis de impacto. El resultado es una mejor comunicación y administración de requerimientos con una mayor probabilidad de completar los proyectos en tiempo, dentro del presupuesto y superando las expectativas. Los proyectos exitosos comienzan con una buena administración de requerimientos cuanto más efectiva sea su ejecución, mayor será el resultado en calidad y satisfacción del cliente.

- **Integración de Word y Base de Datos**

Soporta Microsoft Word para creación y comunicación de los requerimientos; complementa las entradas en forma de documentos con una base de datos comercial para agregar capacidades de organización, seguimiento y administración.

- **Potente motor de requerimientos**

Se pueden configurar fácilmente los tipos de requerimientos, atributos y tipo de documentos. Definir consultas y filtros para encontrar rápidamente la información de interés.

- **Trazabilidad**

Se pueden configurar y dar seguimiento a las relaciones entre requerimientos para verificar que los requerimientos de alto nivel están representados dentro de las especificaciones detalladas de requerimientos de software.

- **Acceso vía Web mediante RequisiteWeb**

Todo aquel que posea acceso Web, independientemente de la plataforma, puede ver y modificar rápida y eficientemente los requerimientos sin necesidad de tener cargado el Rational RequisitePro en su máquina.

Ventajas

- Propicia una mejor comunicación, mejoras en el trabajo en equipo y reduce el riesgo de los proyectos.
- Combina la interfaz conocida y fácil de utilizar de los documentos de Microsoft Word con potentes funciones de base de datos para conseguir la máxima eficacia en análisis y consulta de requerimientos.

- Proporciona a los equipos la posibilidad de comprender el impacto de los cambios.
- Garantiza que todos los componentes del equipo estarán informados de los requisitos más actuales para asegurar la coherencia.
- Proporciona acceso basado en web para los equipos distribuidos.

Desventajas

- El programa es sólo para Windows y es privativa.
- No ofrece soporte a pruebas, es necesario utilizar herramientas externas. (13)

1.3.1.4 GatherSpace

Es una herramienta entorno web para administración de requisitos de software y de gran utilidad para trabajo de equipos distribuidos.

Ventajas

- La herramienta cuenta con un módulo de requisitos donde se gestionan los paquetes, características, requisitos de software, iteraciones, actores, casos de uso y glosarios.
- Módulo para gestionar proyectos y para explotar la información contenida en el proyecto.
- Módulo de generación de reporte (Visión y Requisitos, Modelo de Casos de Uso, Detalles de Características y Requisitos Asociados, Casos de Uso, Detalles de Requisitos, Trazabilidad, Cambios y Errores).

Desventajas

- La aplicación no es un paquete de instalación por lo tanto corre en servidores fuera del lugar de trabajo.
- Es una herramienta privativa. (14)

1.3.2 Antecedentes de ámbito nacional

1.3.2.1 Subsistema Editor de Plantillas

Durante el curso 2008 – 2009 en la Facultad 7 de la UCI se decidió implementar un Subsistema Editor de Plantillas, perteneciente al Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana, para la edición de los modelos de flujo que contienen los datos estadísticos que se generan en los diferentes procesos referentes al sector de la salud. En dependencia de las necesidades planteadas el subsistema proporcionaría autonomía a los usuarios para establecer sus definiciones, sin necesidad de volver a iniciar un nuevo ciclo de desarrollo en la edición de una plantilla, el mismo tiene como objetivo automatizar la gestión y edición de las plantillas para la captura de la información estadística en el sector de la salud. (15)

1.3.2.2 Generador Automático de Plantillas en la Web

Durante el curso 2009 – 2010 en la Facultad 1 de la UCI se desarrolló un Generador automático de plantillas en la Web, que genera plantillas a través de la Web, logrando que los usuarios creen los formularios deseados, para poder recoger y consultar la información necesaria de una actividad. El sistema automatiza la creación de plantillas de datos a través de la Web para cualquier usuario pueda crear sus formularios para la recogida de los datos que considere necesarios, brinda la posibilidad de modificarlos y eliminarlos, si se desea, también automatiza la disponibilidad de cada plantilla creada para cualquier usuario que se interese en llenar sus datos, así como el almacenamiento de los datos de los interesados. (16)

1.4 Análisis crítico

Todas las herramientas descritas anteriormente se pueden utilizar durante el proceso de administración de requisitos y pueden resultar de gran ayuda. Presentan una interfaz amigable y fácil de entender por parte de los usuarios, además de la gran cantidad de funcionalidades que ofrecen.

Sin embargo a pesar de los muchos beneficios que prestan, ninguna de estas herramientas cuenta entre sus funcionalidades con algún editor de plantillas que brinde la posibilidad de crear los artefactos que se generan durante el proceso IPP-3510 Administración de Requisitos, además de que la mayoría de estas herramientas son propietarias lo que representa un inconveniente para su utilización y soporte.

1.5 Herramientas y tecnologías a utilizar

En este epígrafe se realiza un análisis detallado de las principales herramientas y tecnologías que se utilizarán en el desarrollo de la herramienta. Se describe la metodología para el análisis y diseño, el lenguaje de programación que se empleará, así como la plataforma de desarrollo en que se va a trabajar.

1.5.1 Lenguajes

1.5.1.1 Lenguaje Unificado de Modelado (UML 2 .0)

UML (Lenguaje Unificado de Modelado) es un lenguaje para especificar, visualizar, construir y documentar las diferentes etapas del desarrollo de un software, así como para el modelado de procesos de negocio u otros sistemas. UML reúne una colección de las mejores prácticas en la ingeniería que han sido utilizadas con éxito para modelar sistemas grandes y complejos porque cubre tanto objetos conceptuales como los procesos de negocio y funciones del sistema, cubre también objetos concretos como clases en un lenguaje de programación, esquemas de bases de datos y componentes reusables de software.

En las versiones previas se hacía un fuerte hincapié en que UML no era un lenguaje de programación, un modelo creado con él no podía ejecutarse. En UML 2.0 esta asunción cambió en forma drástica y el lenguaje se modificó de manera tal que permitiera capturar mucho más comportamiento, con el fin de permitir la creación de herramientas que brinden la automatización y la generación de código ejecutable a partir de modelos UML.

Ventajas de UML 2.0:

- Produce un aumento en la calidad del desarrollo.
- Reduce los costos del proyecto.
- Mejora en un 50% o más los tiempos totales de desarrollo.
- Permite especificar la estructura y el comportamiento del sistema y comunicarlo a todos los integrantes del proyecto.
- Brinda la posibilidad de obtener un "plano" del sistema.

- Permite dimensionar mejor los riesgos de un proyecto, tener un mejor rendimiento ante de construir el sistema.
- Facilita la documentación de las decisiones de la arquitectura del proyecto.
- Ofrece un mejor soporte a la planificación y control del proyecto.
- Permite realizar una verificación y validación del modelo realizado. (17)

1.5.1.2 CSharp (C#)

C# es un lenguaje orientado a objetos, con seguridad de tipos que permite a los desarrolladores crear una amplia gama de aplicaciones sólidas y seguras que se ejecutan en .NET Framework. Se puede utilizar este lenguaje para crear aplicaciones clientes para Windows, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, entre otras.

Como lenguaje orientado a objetos C# admite los conceptos de encapsulación, herencia y polimorfismo. El proceso de generación de C# es simple en comparación con el de C y C++ y es más flexible que en Java.

No hay archivos de encabezado independientes ni se requiere que los métodos y los tipos se declaren en un orden determinado. Un archivo de código fuente de C# puede definir cualquier número de clases, estructuras, interfaces y eventos.

Entre sus principales características se destacan:

Sencillez: C# elimina muchos elementos que otros lenguajes incluyen y que son innecesarios en .NET. Por ejemplo:

- El código escrito en C# es autocontenido, lo que significa que no necesita de ficheros adicionales al código fuente tales como ficheros de cabecera o ficheros IDL.
- El tamaño de los tipos de datos básicos es fijo e independiente del compilador, sistema operativo o máquina para quienes se compile, lo que facilita la portabilidad del código.
- Modernidad: C# incorpora en el propio lenguaje elementos que a lo largo de los años ha ido demostrándose son muy útiles para el desarrollo de aplicaciones y que en otros lenguajes

como Java o C++ hay que simular, como la inclusión de una instrucción foreach que permita recorrer colecciones con facilidad y es ampliable a tipos definidos por el usuario.

- Orientación a objetos: Como todo lenguaje de programación de propósito general actual, C# es un lenguaje orientado a objetos. Una diferencia de este enfoque respecto al de otros lenguajes como C++, es que el de C# es más puro en tanto que no admiten ni funciones ni variables globales, sino que todo el código y datos han de precisarse dentro de definiciones de tipos de datos, lo que reduce problemas por conflictos de nombres y facilita la legibilidad del código.

- Orientación a componentes: La propia sintaxis de C# incluye elemento propios del diseño de componentes que otro lenguaje tiene que simular mediante construcciones más o menos complejas. Es decir la sintaxis de C# permite definir cómodamente propiedades, eventos o atributos. (18)

Ventajas frente a C++.

- Compila a código intermedio independiente del lenguaje en que haya sido escrita la aplicación e independiente de la máquina donde vaya a ejecutarse.
- Realiza la recolección automática de basura.
- Elimina el uso de punteros, en C# no son necesarios aunque permite utilizarlos.
- Posee capacidades de reflexión.
- No hay que preocuparse por archivos de cabecera ".h".
- Es flexible en cuanto al orden de definición de las clases y las funciones.
- No hay necesidad de declarar funciones y clases antes de invocarlas.
- No existen las dependencias circulares.
- Soporta definición de clases dentro de otras.
- No existen funciones ni variables globales, todo pertenece a una clase.

- Todos los valores son inicializados antes de ser usados (automáticamente por defecto, o manualmente desde constructores estáticos).
- No se pueden utilizar valores no booleanos (enteros, coma flotante) para condicionales.
- Es mucho más limpio y menos propenso a errores.

Ventajas frente a Java.

- El rendimiento es, por lo general, mucho mejor.
- Soporta más tipos primitivos incluyendo tipos numéricos sin signo.
- Se usan indizadores que permiten acceder a cualquier objeto como si se tratase de un arreglo.
- Compilación condicional.
- Aplicaciones multi-hilo más sencillas de manejar.
- Soporta la sobrecarga de operadores, que aunque pueden complicar el desarrollo son opcionales y algunas veces muy útiles.
- Permite el uso de punteros cuando realmente se necesiten, como al acceder a librerías que no se ejecuten sobre la máquina virtual. (19)

1.5.2 Metodologías

1.5.2.1 Rational Unified Process (RUP)

Un Proceso de Desarrollo de Software es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto a modo de plantilla que explica los pasos necesarios para termina el proyecto. RUP es más que un proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, tipos de organizaciones, niveles de aptitud y tamaño de proyecto. Está formado por componentes interconectados a través de interfaces y junto con el Lenguaje Unificado de Modelado UML. Constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Es un refinamiento realizado por Rational Software del más genérico Proceso Unificado.

El ciclo de vida de RUP se caracteriza por:

- Guiado por los casos de uso.
- Centrado en la arquitectura.
- Iterativo e incremental.

RUP para organizar el proceso de construcción cuenta con cuatro fases, seis flujos de trabajos fundamentales y tres de apoyo.

Fases de desarrollo: Inicio, Elaboración, Construcción y Desarrollo.

Flujos de trabajo fundamentales: Modelado de negocio Requerimientos, Análisis y Diseño, Implementación, Prueba, Despliegue.

Flujos de trabajo de apoyo: Administración de Configuración y Cambios, Administración del proyecto, Ambiente. (20)

1.5.3 Herramientas

1.5.3.1 Visual Studio 2008

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros.

Visual Studio permite a los desarrolladores crear aplicaciones desktop, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles. (21)

1.5.3.2 Enterprise Architect 7.1 (EA)

Enterprise Architect es una herramienta CASE (Computer Aided Software Engineering) para el diseño y construcción de sistemas de software. EA soporta la especificación de UML 2.0 que describe un lenguaje visual por el cual se pueden definir mapas o modelos de un proyecto. EA es

una herramienta progresiva que cubre todos los aspectos del ciclo de desarrollo y proporciona una trazabilidad completa desde la fase inicial del diseño a través del despliegue y mantenimiento. También provee soporte para pruebas, mantenimiento y control de cambio.

Algunas de las características claves de Enterprise Architect son:

- Crea elementos del modelo UML para un amplio alcance de objetivos.
- Ubica esos elementos en diagramas y paquetes.
- Documenta los elementos que se han creado. La documentación de alta calidad puede ser rápidamente exportada desde sus modelos en formato RTF e importar a Word para una personalización y presentación final.
- Se puede realizar ingeniería reversa del código existente en varios lenguajes tales como: C++, C#, Delphi, Java, Python, PHP, VB.NET y clases de Visual Basic.
- Permite estimar el tamaño de un proyecto en esfuerzo de trabajo en horas. (22)

1.5.3.3 Sistemas Gestores de Bases de Datos (SGBD ó DBMS).

Un Sistema Gestor de Bases de Datos (SGBD) o DBMS (Database Management System) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos.

1.5.3.4 PostgreSQL

PostgreSQL es un potente sistema de base de datos objeto relacional, libre (su código fuente está disponible). Tiene más de 15 años de activo desarrollo y arquitectura probada que se ha ganado una muy buena reputación por su confiabilidad e integridad de datos. Funciona en todos los sistemas operativos importantes, incluyendo Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows.

Entre sus principales características se encuentran:

- Soporta casi toda la sintaxis SQL pues tiene soporte total para llaves extranjeras, joins, vistas, disparadores y procedimientos almacenados (en múltiples lenguajes).
- Interfaces con lenguajes de programación: La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, Pike, etc.
- Incluye la mayoría de los tipos de datos SQL92 y SQL99 (INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, y TIMESTAMP), soporta almacenamiento de objetos grandes binarios, además de tipos de datos y operaciones geométricas.
- Posee un sofisticado analizador/optimizador de consultas.
- Soporta juegos de caracteres internacionales, codificación de caracteres multibyte.
- Ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros clientes, conservando todas las características de estabilidad y rendimiento.
- PostgreSQL soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario. (23)

En el este capítulo se realizó un estudio sobre los conceptos fundamentales para el entendimiento de la presente investigación, se abordaron los temas relacionados con la administración de requisitos. Se estudiaron sistemas vinculados al campo de acción en el ámbito nacional e internacional. También se agrupan en este capítulo las fundamentales herramientas, tecnologías y metodologías utilizadas en el desarrollo de la herramienta.

Capítulo 2: Características del Sistema

El presente capítulo tiene como objetivo presentar la propuesta de solución para la Herramienta de Apoyo al Proceso Administración de Requisitos. Al existir poca claridad en los procesos del negocio se sugiere un Modelo de Dominio, y teniendo en cuenta las necesidades del cliente se crean los requisitos funcionales y no funcionales.

2.1 Objetos de automatización

Se desea automatizar la creación y edición de las plantillas que se generan en el proceso IPP-3510 Administración de Requisitos, donde se recoge la información referente a los requisitos, a los proveedores de requisitos y a los casos de uso.

2.2 Información que se maneja

La información empleada en las plantillas a automatizar en algunos casos es numérica y en otros, es información específica referente a los criterios para validar requisitos de los clientes y de los productos. También se maneja información correspondiente a la especificación de requisitos de software, a la de los casos de usos y a la de la matriz de disponibilidad de proveedores de requisitos.

Toda la información que se recoge en las plantillas es de vital importancia para el diseño e implementación del sistema, por lo que la misma debe ser tratada y analizada correctamente, garantizando un mejor resultado en cuanto a la calidad del sistema desarrollado.

2.3 Descripción del Sistema

Para la propuesta que se desea, el usuario que interactúe con la herramienta, después de una autenticación previa, será el responsable de realizar todas las funcionalidades que esta brinda, como son Crear matriz de disponibilidad de proveedores, Definir proveedores válidos de requisitos, Evaluar casos de uso por complejidad y prioridad y Validar requisitos del cliente y del producto, para cada una de estas funcionalidades la herramienta cuenta con un Ver detalles, el cual contiene un eliminar, un modificar y un listar donde se pueden buscar los datos previamente creados.

2.4 Modelo de Dominio

El Modelo de Dominio es presentado como uno o más diagramas de clases y que contiene, no conceptos propios de un sistema de software sino de la propia realidad física, es una representación de las clases conceptuales del mundo real, no de componentes de software. No se trata de un conjunto de diagramas que describen clases de software ni objetos de software con responsabilidades, sino más bien representa las clases conceptuales u objetos del mundo real en un dominio de interés. (24)

El modelo de dominio se debe concebir como un diccionario visual de abstracciones que será utilizado en fases posteriores y cuya función principal es ayudar a comprender el problema a tratar.

El principal objetivo del Modelo de Dominio es ayudar a comprender los conceptos que utilizan los usuarios, los conceptos con los que trabajan y con los que deberá trabajar la aplicación.

Para su elaboración se tiene tres pasos fundamentales:

- Identificar las Clases Conceptuales.
- Dibujarlas en un Diagrama de Clases.
- Añadir Relaciones y Atributos.

En el Modelo de Dominio no se muestra comportamiento. Las clases conceptuales pueden tener atributos pero no métodos. Cualquiera sea la solución de casos de uso que se haya elegido, los conceptos e ideas propias del dominio del problema son las mismas; un mismo modelo de dominio contempla cualquiera de las soluciones analizadas. El modelo de dominio es global, es decir se realiza para todos los casos de uso y no para uno en particular. (25)

El modelo de dominio se realiza si no se logra determinar los procesos de negocio con fronteras bien establecidas donde se logren ver claramente, quienes son las personas que realizan cada proceso de negocio, así como las actividades pertinentes.

La Herramienta de Apoyo al Proceso Administración de Requisitos no responde a las exigencias de un cliente, sino que se desarrolla para satisfacer las necesidades de procesamiento de información sin seguir necesariamente la estructura de ningún organismo o entidad en específico.

2.5 Conceptos Fundamentales del dominio

Usuario: Representa la persona encargada/o de crear las plantillas y realizar las diferentes funcionalidades con las que cuenta la herramienta.

Plantilla: La clase plantilla es el formato digital, encargada de captar la información.

Información: La clase información es todo lo que se encuentra en un plantilla, es decir, todo lo que contienen los campos de la plantilla.

Formato: Una clase formato es la forma en que información se encuentra en una plantilla, es decir, el estilo, el tamaño y el color de la fuente.

2.6 Diagrama de Modelo de Dominio

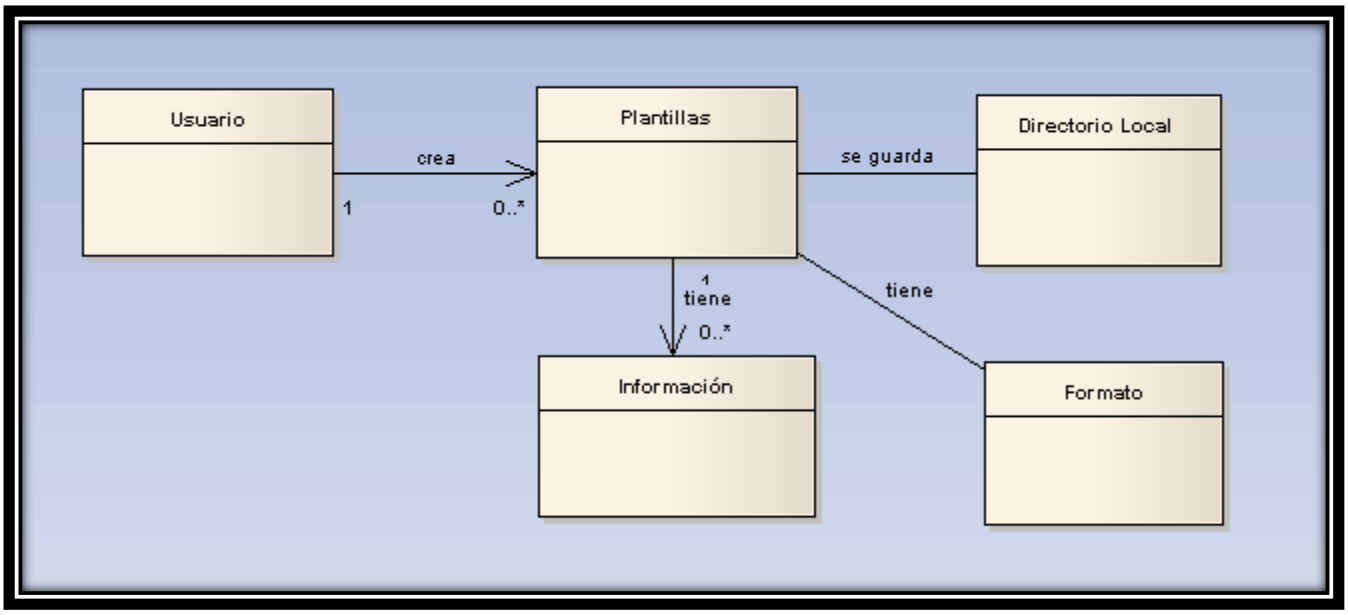


Figura1: Diagrama de Modelo de Dominio

2.7 Especificación de los requisitos de software

2.7.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. Se mantienen invariables sin importar con qué propiedades o cualidades se relacionen. (26)

Listado de requisitos funcionales	
RF1 Autenticar RF2 Crear usuario RF3 Listar o buscar usuario RF4 Eliminar usuario RF5 Crear rol RF6 Listar o buscar rol RF7 Eliminar rol RF8 Asignar rol RF9 Crear catálogo de proveedores de requisitos. RF10 Ver detalle catálogo de proveedores de requisitos. RF11 Modificar catálogo de proveedores de requisitos. RF12 Eliminar catálogo de proveedores de requisitos. RF13 Listar o buscar catálogo de proveedores de requisitos. RF14 Crear proveedores de requisitos válidos. RF15 Ver detalle proveedores de requisitos válidos. RF16 Modificar proveedores de requisitos válidos. RF17 Eliminar proveedores de requisitos válidos. RF18 Listar o buscar proveedores de requisitos válidos.	RF19 Crear matriz de disponibilidad de proveedores válidos. RF20 Ver detalle matriz de disponibilidad de proveedores válidos. RF21 Modificar matriz de disponibilidad de proveedores válidos. RF22 Eliminar matriz de disponibilidad de proveedores válidos. RF23 Listar o buscar matriz de disponibilidad de proveedores válidos. RF24 Crear catálogo de requisitos del cliente. RF25 Ver detalle catálogo de requisitos del cliente. RF26 Modificar catálogo de requisitos del cliente. RF27 Eliminar catálogo de requisitos del cliente. RF28 Listar o buscar catálogo de requisitos del cliente. RF29 Crear criterios para validar requisitos del cliente. RF30 Ver detalle criterios para validar requisitos del cliente. RF31 Eliminar criterios para validar requisitos del cliente. RF32 Listar o buscar requisitos del cliente aprobados. RF33 Crear catálogo de requisitos del producto. RF34 Ver detalle catálogo de requisitos del producto.

RF35 Modificar catálogo de requisitos del producto. RF36 Eliminar catálogo de requisitos del producto. RF37 Listar o buscar catálogo de requisitos del producto. RF38 Crear criterios para validar requisitos del producto. RF39 Ver detalle criterios para validar requisitos del producto. RF40 Eliminar criterios para validar requisitos del producto. RF41 Listar o buscar criterios para validar requisitos del producto aprobados. RF42 Crear catálogo de casos de uso. RF43 Ver detalle catálogo de casos de uso. RF44 Modificar catálogo de casos de uso. RF45 Eliminar catálogo de casos de uso. RF46 Listar o buscar catálogo de casos de uso.	RF47 Crear criterios de prioridad para validar casos de uso. RF48 Ver detalle criterios de prioridad para validar casos de uso. RF49 Modificar criterios de prioridad para validar casos de uso. RF50 Eliminar criterios de prioridad para validar casos de uso. RF51 Listar o buscar catálogo de casos de uso validados por criterios de prioridad. RF52 Crear criterios de complejidad para validar casos de uso. RF53 Ver detalle criterios de complejidad para validar casos de uso. RF54 Modificar criterios de complejidad para validar casos de uso. RF55 Eliminar criterios de complejidad para validar casos de uso. RF56 Listar o buscar catálogo de casos de uso validados por criterios de complejidad.

2.7.2 Requisitos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el cliente debe tener. Debe pensarse en estas propiedades como las características que hacen al cliente atractivo, usable, rápido o confiable. (27)

Usabilidad

- RNF1 La herramienta podrá ser utilizada por las personas encargadas de realizar el proceso IPP-3510 en todos los proyectos del centro CESIM.
- RNF2 La herramienta deberá garantizar un acceso fácil y rápido, podrá ser usado por usuarios con conocimientos básicos en informática.

Seguridad

Confidencialidad

- RNF3 La información estará protegida contra accesos no autorizados, utilizando mecanismos de autenticación y autorización.

Integridad

- RNF4 La información podrá ser modificada solo por personal autorizado.
- RNF5 Se realizarán las operaciones de cálculo y validaciones automáticamente, reduciendo la posibilidad de registrar datos erróneos producidos por cálculos manuales.

Restricciones de diseño

- RNF8 Para el desarrollo se utilizó C# como leguaje de programación.
- RNF9 Se utilizó Visual Studio como entorno de desarrollo.
- RNF10 Para realizar el modelado del software se utilizó la herramienta Enterprise Architect.

Interfaces de usuario

- RNF11 La herramienta debe tener una interfaz sencilla, agradable, legible y de fácil uso para el usuario.

Interfaces Hardware

- RNF13 Ordenador Pentium o superior, 512 MB de memoria RAM, Disco Duro de 40 GB.

Interfaces Software

- RNF15 La Herramienta se implementará sobre .NET Framework 3.5.

2.8 Diagrama de Casos de Uso del Sistema

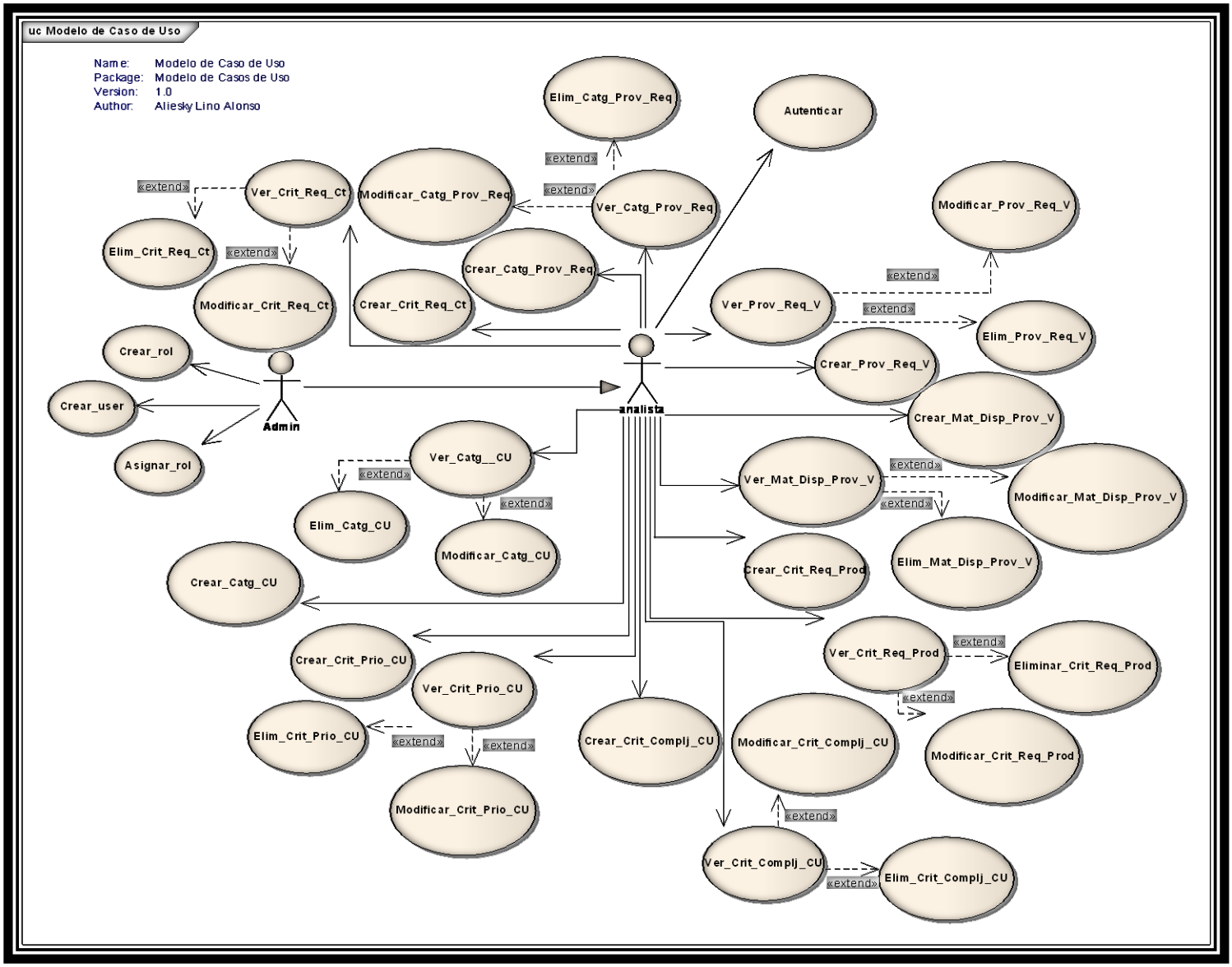


Figura2: Diagrama de Caso de Uso del Sistema

2.9 Actores del Sistema

A continuación se define y describen los actores del sistema.

Actor	Descripción
Analista	Es el encargado de realizar las operaciones de la herramienta, no tiene acceso a la parte de la configuración de los usuarios y roles.
Admin	Es el que realiza todas las operaciones con que cuenta la herramienta.

Figura3: Descripción de los Actores del Sistema.

2.10 Especificación de los Casos de Uso

2.10.1 Descripción del caso de uso: Autenticar

Caso de Uso:	Autenticar
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el usuario accede a la aplicación y finaliza cuando cierra la sesión.
Precondiciones:	El usuario accede a la aplicación.
Referencias	
Prioridad	Media

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario introduce usuario y contraseña y acepta enviar la información.	2. Se muestra una interfaz para que el usuario introduzca los datos necesarios para autenticarse, estos datos son usuario y contraseña.
	3. El sistema verifica que el usuario y contraseña sean correctos. Si la notificación informa que los datos introducidos por el usuario son incorrectos ver flujo alterno 1

	4. Autentica el usuario

Prototipo de Interfaz

Flujos Alternos 1

Acción del Actor	Respuesta del Sistema
	1. Muestra un mensaje de error y brinda la opción al usuario de volver a autenticarse.

Prototipo de Interfaz

Poscondiciones	El usuario es registrado y se le da acceso a trabajar con la aplicación.

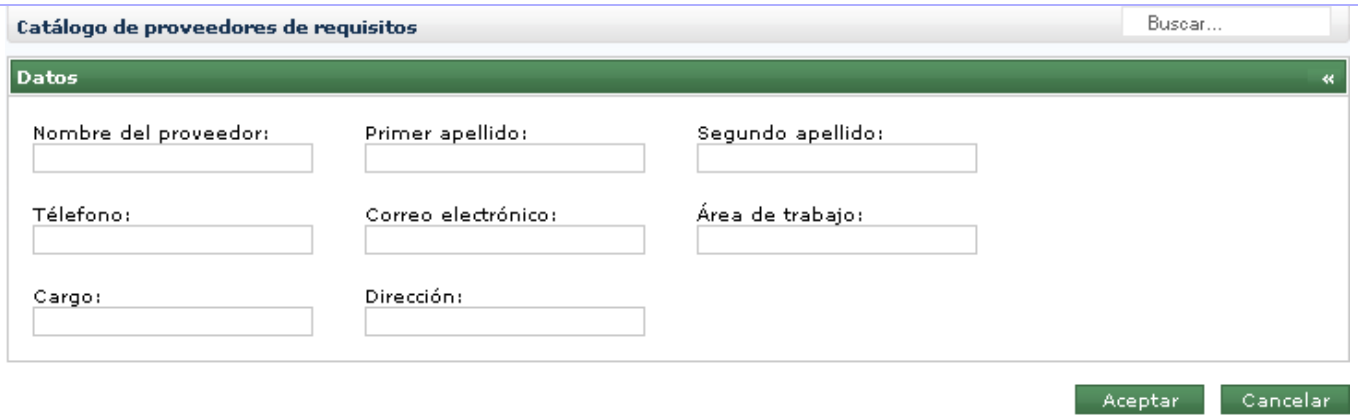
Figura4: Descripción del caso de uso: Autenticar

2.10.2 Descripción del caso de uso: Crear catálogo de proveedores de requisitos

Caso de uso:	Crear catálogo de proveedores de requisitos
Resumen:	El caso de uso inicia cuando el Admin accede a la opción Crear catálogo de proveedores de requisitos, el sistema brinda la posibilidad de introducir los datos para crear el catálogo de proveedores de requisitos, el actor introduce los datos de los proveedores de requisitos, el sistema crea el catálogo de proveedores de requisitos, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico
Precondiciones:	
REFERENCIAS	
Actores:	Admin
Requisitos:	
Entidades:	Catálogo de proveedores de requisitos
Casos de Uso:	Crear catálogo de proveedores de requisitos
FLUJO NORMAL DE EVENTOS	

Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el Admin accede a la opción Crear catálogo de proveedores de requisitos.	

	2. Brinda la posibilidad de introducir los datos del catálogo de proveedores de requisitos tales como: • Nombre • Primer Apellido • Segundo Apellido • Teléfono • Correo electrónico • Área de Trabajo • Cargo • Dirección • Departamento Y permite: • Aceptar crear catálogo de proveedores de requisitos • Cancelar operación, **ver Alternativa 1**: "Cancelar operación."
3. Introduce los datos del catálogo de proveedores de requisitos.	
4. Selecciona la opción de aceptar crear catálogo de proveedores de requisitos.	
	5. Valida los datos. Si hay datos incompletos, **ver Alternativa 2**: "Existen datos incompletos.". Si hay datos incorrectos, **ver Alternativa 3**: "Existen datos incorrectos."
	6. Crea catálogo de proveedores de requisitos.
	7. Muestra los datos del catálogo de proveedores de requisitos, **ver CU**: Ver catálogo de proveedores de requisitos.
	8. El caso de uso termina.
Prototipo de Interfaz	



FLUJOS ALTERNOS	
Alternativa 1. "Cancelar operación."	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de Cancelar operación.	
	2. Regresa a la vista anterior.
	3. El caso de uso termina.
Alternativa 2. "Existen datos incompletos."	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incompletos.
Alternativa 3. "Existen datos incorrectos"	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incorrectos.
Poscondiciones	Se crea el Catálogo de proveedores de requisitos.

Figura5: Descripción del caso de uso: Crear catálogo de proveedores de requisitos

2.10.3 Descripción del caso de uso: Crear proveedores de requisitos válidos

Caso de Uso	Crear proveedores de requisitos válidos
Resumen:	El caso de uso inicia cuando el Usuario accede a la opción Crear proveedores de requisitos válidos, el sistema brinda la posibilidad de introducir los datos para crear los proveedores de requisitos válidos, el actor introduce los datos de los proveedores de requisitos, el sistema crea el catálogo de proveedores de requisitos, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico
Precondiciones:	El catálogo de proveedores de requisitos ya este creado.

<table>
<tr><th colspan="3">REFERENCIAS</th></tr>
<tr><td>Actores:</td><td colspan="2">Admin</td></tr>
<tr><td>Requisitos:</td><td colspan="2"></td></tr>
<tr><td>Entidades:</td><td colspan="2">Proveedores de requisitos válidos</td></tr>
<tr><td>Casos de Uso:</td><td colspan="2">Crear catálogo de proveedores de requisitos</td></tr>
<tr><th colspan="3">FLUJO NORMAL DE EVENTOS</th></tr>
<tr><th colspan="2">Acción del Actor</th><th>Respuesta del Sistema</th></tr>
<tr><td colspan="2">1. El caso de uso inicia cuando el Admin accede a la opción Crear proveedores de requisitos válidos.</td><td></td></tr>
<tr><td colspan="2"></td><td>2. Brinda la posibilidad de introducir los valores numéricos que se le otorgarán a los siguientes criterios:

• Conocimiento del negocio
• Conocimiento del mercado
• Disponibilidad de tiempo
• Habilidades de comunicación
• Interacción con el sistema
• Grado de autoridad
• Conocimientos de informática
• Compromiso con el proyecto
• Departamento
Y permite:
• Aceptar crear proveedores de requisitos válidos
• Cancelar operación, ver Alternativa 1: “Cancelar operación.”</td></tr>
<tr><td colspan="2">3. Introduce los valores numéricos en los diferentes criterios.</td><td></td></tr>
<tr><td colspan="2">4. Selecciona la opción de aceptar.</td><td></td></tr>
<tr><td colspan="2"></td><td>5. Valida los datos. Si hay datos incompletos, ver Alternativa 2: “Existen datos incompletos.”. Si hay datos incorrectos, ver Alternativa 3: “Existen datos incorrectos.”</td></tr>
<tr><td colspan="2"></td><td>6. Crea o calcula proveedores de requisitos válidos.</td></tr>
<tr><td colspan="2"></td><td>7. Muestra los datos de proveedores de requisitos válidos, ver CU: Ver proveedores de requisitos válidos.</td></tr>
<tr><td colspan="2"></td><td>8. El caso de uso termina.</td></tr>
<tr><td colspan="3">Prototipo de Interfaz</td></tr>
</table>

FLUJOS ALTERNOS	
Alternativa 1. "Cancelar operación."	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de Cancelar operación.	
	2. Regresa a la página inicio.
	3. El caso de uso termina.
Alternativa 2. "Existen datos incompletos."	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incompletos.
Alternativa 3. "Existen datos incorrectos"	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incorrectos.
Poscondiciones	Se crean los proveedores de requisitos válidos.

Figura6: Descripción del caso de uso: Crear proveedores de requisitos válidos

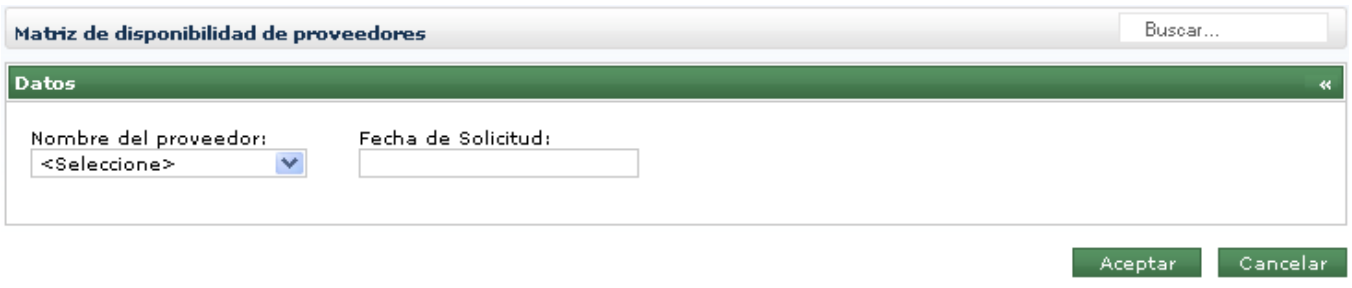
2.10.4 Descripción del caso de uso: Crear matriz de disponibilidad de proveedores válidos

Caso de Uso:	Crear matriz de disponibilidad de proveedores válidos.
Resumen:	El caso de uso inicia cuando el Admin accede a la opción Crear matriz de disponibilidad de proveedores válidos, el sistema brinda la posibilidad de introducir los datos para crear la Matriz de disponibilidad de proveedores válidos, el actor introduce los datos de la matriz de disponibilidad de proveedores válidos, el sistema crea la matriz de disponibilidad de proveedores válidos, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico

Precondiciones:	Los proveedores de requisitos válidos ya estén creados.
REFERENCIAS	
Actores:	Admin
Requisitos:	
Entidades:	Disponibilidad de proveedores válidos.
Casos de Uso:	Crear matriz de disponibilidad de proveedores válidos.

FLUJO NORMAL DE EVENTOS	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el Admin accede a la opción Crear matriz de disponibilidad de proveedores válidos.	
	2. Brinda la posibilidad de introducir los datos de la matriz de disponibilidad de proveedores válidos tales como: • Fecha • Nombre(seleccionar) Y permite: • Aceptar crear matriz de disponibilidad de proveedores válidos • Cancelar operación. Ver Alternativa 1: "Cancelar operación."
3. Introduce los datos de la matriz de disponibilidad de proveedores válidos.	
4. Selecciona la opción de aceptar.	
	5. Valida los datos. Si hay datos incompletos, **ver Alternativa 2**: "Existen datos incompletos.". Si hay datos incorrectos, **ver Alternativa 3**: "Existen datos incorrectos."
	6. Crea la matriz de disponibilidad de proveedores válidos.
	7. Muestra los datos de disponibilidad de proveedores válidos, **ver CU**: Ver matriz de disponibilidad de proveedores válidos.
	8. El caso de uso termina.

Prototipo de Interfaz



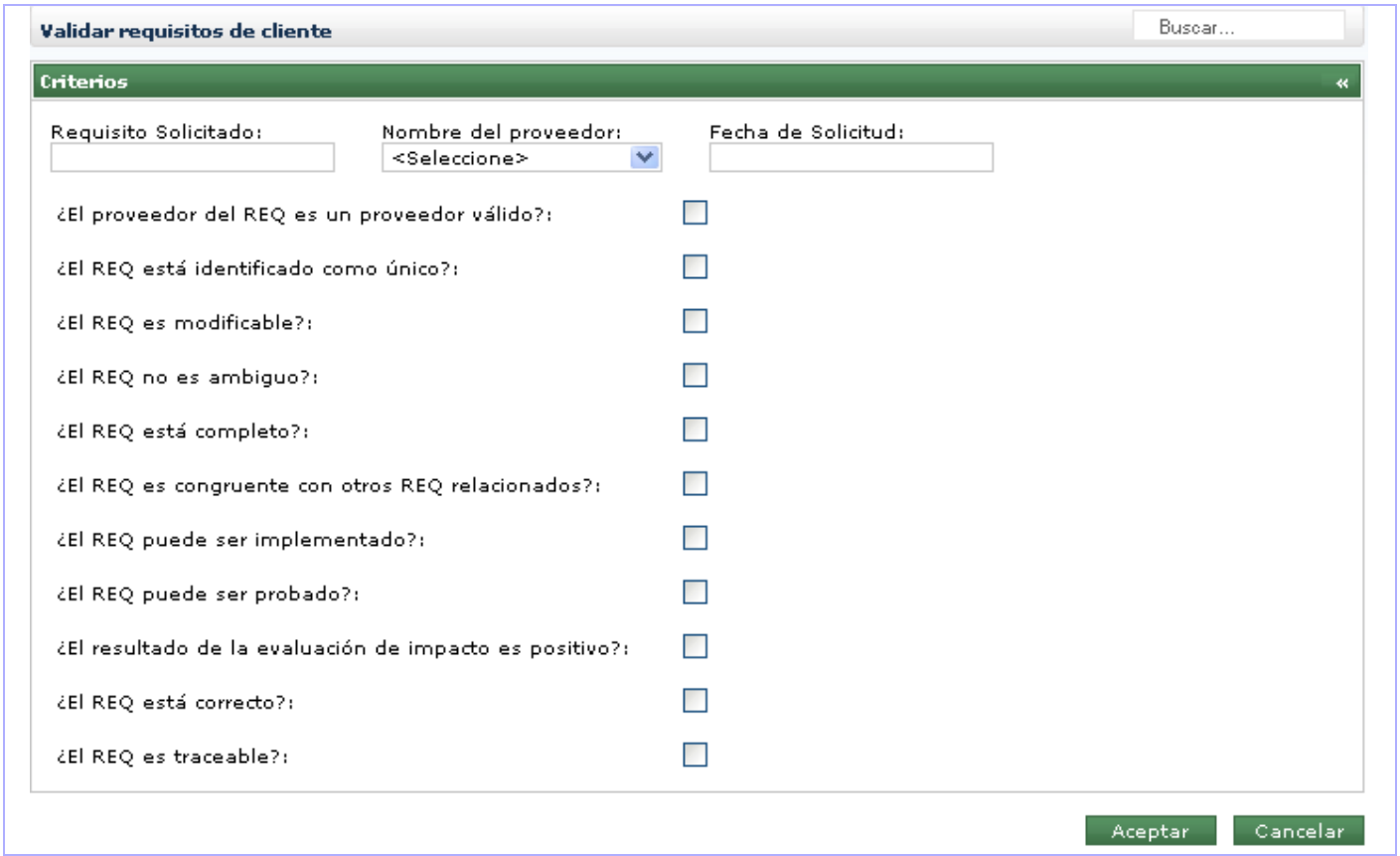
FLUJOS ALTERNOS	
Alternativa 1. "Cancelar operación."	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de Cancelar operación.	
	2. Regresa a la página inicio.
	3. El caso de uso termina.
Alternativa 2. "Existen datos incompletos."	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incompletos.
Alternativa 3. "Existen datos incorrectos"	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incorrectos.
Poscondiciones	Se crea la matriz de disponibilidad de proveedores válidos.

Figura7: Descripción del caso de uso: Crear matriz de disponibilidad de proveedores válidos

2.10.5 Descripción del caso de uso: Crear criterios para validar requisitos del cliente

Caso de Uso:	Crear los criterios para validar requisitos del cliente.
Resumen:	El caso de uso inicia cuando el Admin accede a la opción Crear los criterios para validar requisitos del cliente, el sistema brinda la posibilidad de introducir los datos para crear los criterios para validar requisitos del cliente, el actor introduce los datos de los criterios para validar requisitos del cliente, el sistema crea los criterios para validar requisitos del cliente, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico
Precondiciones:	El catálogo de proveedores de requisitos ya este creado.
REFERENCIAS	
Actores:	Admin
Requisitos:	
Entidades:	Criterios para validar requisitos del cliente.
Casos de Uso:	Crear criterios para validar requisitos del cliente.
FLUJO NORMAL DE EVENTOS	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el Admin accede a la opción Crear criterios para validar requisitos del cliente.	

	2. Brinda la posibilidad de seleccionar los criterios para validar requisitos del cliente tales como: • Requisito Solicitado • Solicitado Por • Fecha de Solicitud • ¿El proveedor del REQ es un proveedor válido? • ¿El REQ está identificado como único? • ¿El REQ es modificable? • ¿El REQ no es ambiguo? • ¿El REQ está completo? • ¿El REQ es congruente con otros REQ relacionados? • ¿El REQ puede ser implementado? • ¿El REQ puede ser probado? • ¿El resultado de la evaluación de impacto es positivo? • ¿El REQ está correcto? • ¿El REQ es traceable? Y permite: • Aceptar crear los criterios para validar requisitos del cliente. • Cancelar operación. **Ver Alternativa 1**: "Cancelar operación."
3. Introduce los datos de los criterios para validar requisitos del cliente.	
4. Selecciona la opción de aceptar.	
	5. Valida los datos. Si hay datos incompletos, **ver Alternativa 2**: "Existen datos incompletos.". Si hay datos incorrectos, **ver Alternativa 3**: "Existen datos incorrectos."
	6. Crean los criterios para validar requisitos del cliente.
	7. Muestra los datos de los criterios para validar requisitos del cliente, **ver CU**: Ver los criterios para validar requisitos del cliente.
	8. El caso de uso termina.
Prototipo de Interfaz	



FLUJOS ALTERNOS	
Alternativa 1. "Cancelar operación."	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de Cancelar operación.	
	2. Regresa a la página inicio.
	3. El caso de uso termina.
Alternativa 2. "Existen datos incompletos."	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incompletos.
Alternativa 3. "Existen datos incorrectos"	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incorrectos.
Poscondiciones	Se crea la los criterios para validar requisitos del cliente.

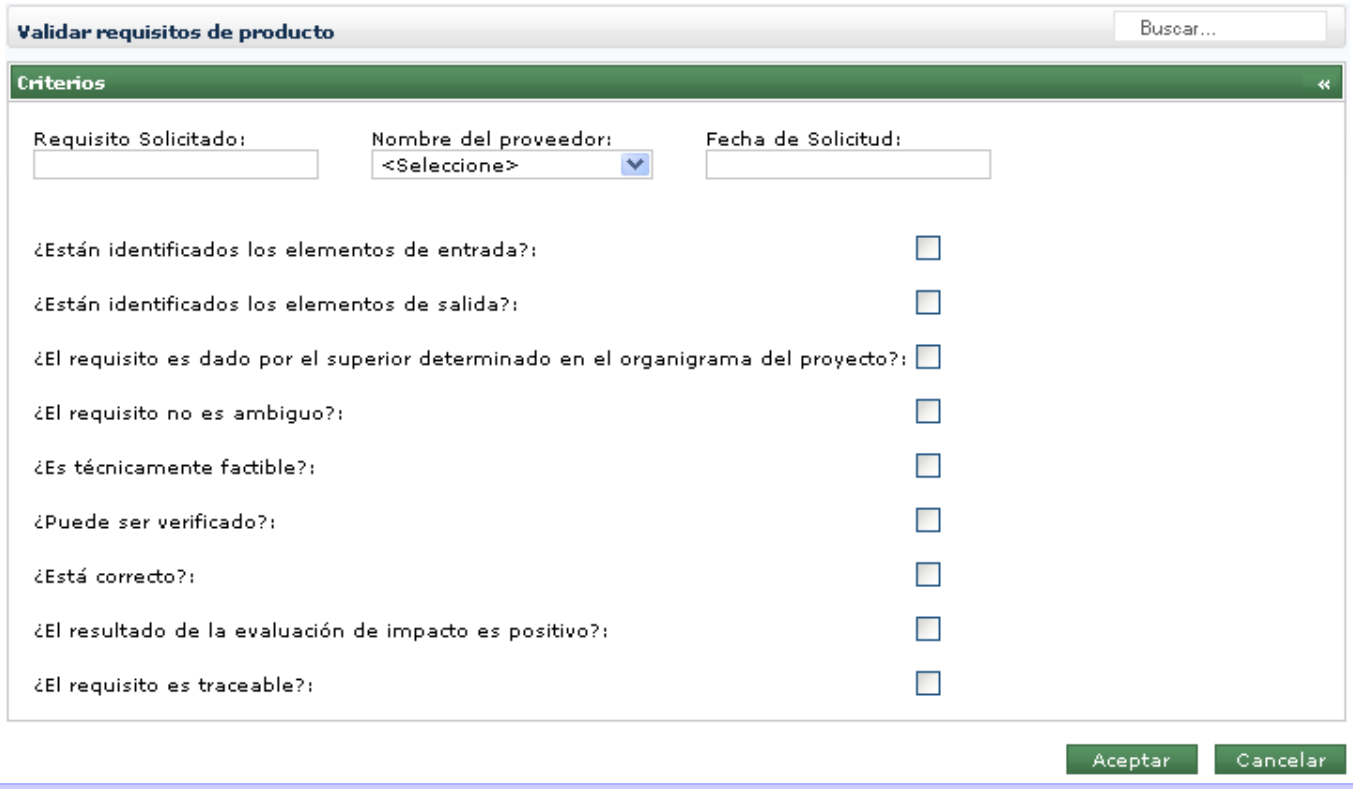
Figura8: Descripción del caso de uso: Crear criterios para validar requisitos del cliente

2.10.6 Descripción del caso de uso: Crear criterios para validar requisitos del producto

Caso de Uso:	Crear criterios para validar requisitos del producto.
Resumen:	El caso de uso inicia cuando el Admin accede a la opción Crear criterios para validar requisitos del producto, el sistema brinda la posibilidad de introducir los datos para Crear criterios para validar requisitos del producto, el actor introduce los datos de los criterios para validar requisitos del producto, el sistema crea los criterios para validar requisitos del producto, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico
Precondiciones:	El catálogo de proveedores de requisitos ya este creado.
REFERENCIAS	
Actores:	Admin
Requisitos:	
Entidades:	Criterios para validar requisitos del producto.
Casos de Uso:	Crear criterios para validar requisitos del producto.
FLUJO NORMAL DE EVENTOS	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el Admin accede a la opción Crear criterios para validar requisitos del producto.	
	2. Brinda la posibilidad de seleccionar los criterios para validar requisitos del producto tales como: • Requisito Solicitado • Solicitado Por • Fecha de Solicitud • ¿Están identificados los elementos de entrada? • ¿Están identificados los elementos de salida? • ¿El requisito es dado por el superior determinado en el organigrama del proyecto? • ¿El requisito no es ambiguo? • ¿Es técnicamente factible? • ¿Puede ser verificado? • ¿Está correcto? • ¿El resultado de la evaluación de impacto es positivo? • ¿El requisito es traceable?

	Y permite: • Aceptar Crear criterios para validar requisitos del producto. • Cancelar operación. **Ver Alternativa 1**: "Cancelar operación."
3. Introduce los datos de los criterios para validar requisitos del producto.	
4. Selecciona la opción de aceptar.	
	5. Valida los datos. Si hay datos incompletos, **ver Alternativa 2**: "Existen datos incompletos.". Si hay datos incorrectos, **ver Alternativa 3**: "Existen datos incorrectos."
	6. Crea los criterios para validar requisitos del producto.
	7. Muestra los datos de los criterios para validar requisitos del producto, **ver CU**: Ver los criterios para validar requisitos del producto.
	8. El caso de uso termina.

Prototipo de Interfaz



FLUJOS ALTERNOS

Alternativa 1. "Cancelar operación."

Acción del Actor	Respuesta del Sistema
1. Selecciona la opción de Cancelar operación.	

	2. Regresa a la página inicio.
	3. El caso de uso termina.
Alternativa 2. "Existen datos incompletos."	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incompletos.
Alternativa 3. "Existen datos incorrectos"	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incorrectos.
Poscondiciones	Se crean los criterios para validar requisitos del producto.

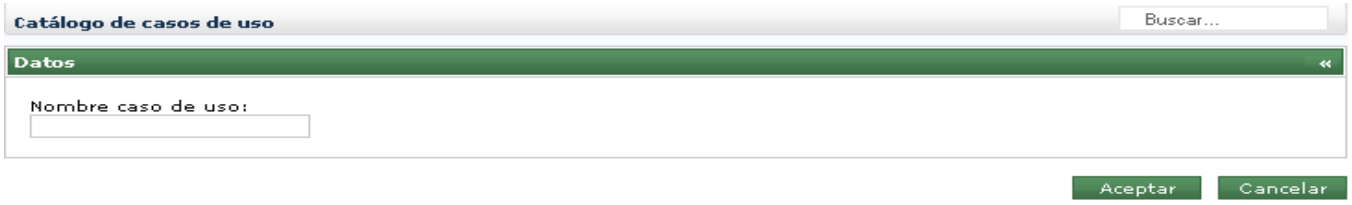
Figura9: Descripción del caso de uso: Crear criterios para validar requisitos del producto

2.10.7 Descripción del caso de uso: Crear catálogo de casos de uso

Caso de uso:	Crear catálogo de casos de uso
Resumen:	El caso de uso inicia cuando el Admin accede a la opción Crear catálogo de casos de uso, el sistema brinda la posibilidad de introducir los datos para crear el catálogo de casos de uso, el actor introduce los datos de los proveedores de requisitos, el sistema crea el catálogo de casos de uso, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico
Precondiciones:	
REFERENCIAS	
Actores:	Admin
Requisitos:	
Entidades:	Catálogo de casos de uso
Casos de Uso:	Crear catálogo de casos de uso
FLUJO NORMAL DE EVENTOS	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el Admin accede a la opción Crear catálogo de casos de uso.	
	2. Brinda la posibilidad de introducir los datos del catálogo de casos de uso tales como: • Nombre Y permite: • Aceptar Crear catálogo de casos de uso. • Cancelar operación. **Ver Alternativa 1**: "Cancelar operación."
3. Introduce los datos del catálogo de casos de uso.	
4. Selecciona la opción de aceptar Crear catálogo de casos de uso.	

	5. Valida los datos. Si hay datos incompletos, **ver Alternativa 2**: "Existen datos incompletos.". Si hay datos incorrectos, **ver Alternativa 3**: "Existen datos incorrectos."
	6. Crea catálogo de casos de uso.
	7. Muestra los datos del catálogo de casos de uso, **ver CU**: Ver catálogo de casos de uso.
	8. El caso de uso termina.

Prototipo de Interfaz



FLUJOS ALTERNOS	
Alternativa 1. "Cancelar operación."	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de Cancelar operación.	
	2. Regresa a la página inicio.
	3. El caso de uso termina.
Alternativa 2. "Existen datos incompletos."	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incompletos.
Alternativa 3. "Existen datos incorrectos"	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incorrectos.
Poscondiciones	Se crea el Catálogo de casos de uso.

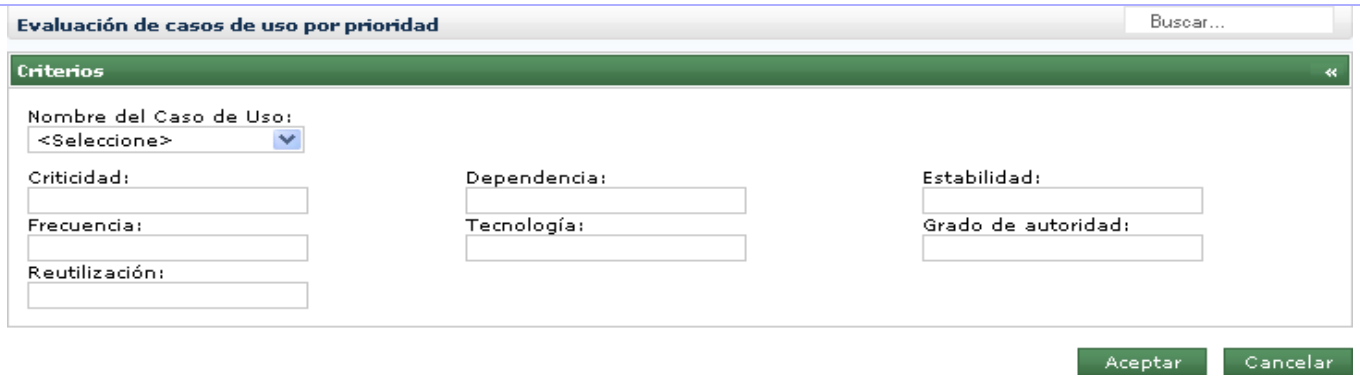
Figura10: Descripción del caso de uso: Crear catálogo de casos de uso

2.10.8 Descripción del caso de uso: Crear criterios de prioridad para validar casos de uso

Caso de Uso:	Crear los criterios de prioridad para validar casos de uso.
Resumen:	El caso de uso inicia cuando el Admin accede a la opción Crear los criterios de prioridad para validar casos de uso, el sistema brinda la posibilidad de introducir los datos para crear los criterios de prioridad para validar casos de uso, el actor introduce los datos de los criterios de prioridad para validar

	casos de uso, el sistema crea los criterios de prioridad para validar casos de uso, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico
Precondiciones:	El catálogo de casos de uso.
REFERENCIAS	
Actores:	Admin
Requisitos:	
Entidades:	Criterios de prioridad para validar casos de uso.
Casos de Uso:	Crear criterios de prioridad para validar casos de uso.

FLUJO NORMAL DE EVENTOS	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el Admin accede a la opción Crear criterios de prioridad para validar casos de uso.	
	2. Brinda la posibilidad de seleccionar los criterios de prioridad para validar casos de uso tales como: • Criticidad • Dependencia • Estabilidad • Frecuencia Y permite: 1. Aceptar crear los criterios de prioridad para validar casos de uso. 2. Cancelar operación. **Ver Alternativa 1**: "Cancelar operación."
3. Introduce los datos de los criterios de prioridad para validar casos de uso.	
4. Selecciona la opción de aceptar.	
	5. Valida los datos. Si hay datos incompletos, **ver Alternativa 2**: "Existen datos incompletos.". Si hay datos incorrectos, **ver Alternativa 3**: "Existen datos incorrectos."
	6. Crean los criterios de prioridad para validar casos de uso.
	7. Muestra los datos de los criterios de prioridad para validar casos de uso, **ver CU**: Ver los criterios de prioridad para validar casos de uso.
	8. El caso de uso termina.
Prototipo de Interfaz	



FLUJOS ALTERNOS	
Alternativa 1. "Cancelar operación."	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de Cancelar operación.	
	2. Regresa a la página inicio.
	3. El caso de uso termina.
Alternativa 2. "Existen datos incompletos."	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incompletos.
Alternativa 3. "Existen datos incorrectos"	
Acción del Actor	**Respuesta del Sistema**
	1. Muestra un indicador sobre los campos incorrectos.
Poscondiciones	Se crea la los criterios de prioridad para validar casos de uso.

Figura11: Descripción del caso de uso: Crear los criterios de prioridad para validar casos de uso

En este capítulo se identificaron los principales conceptos del modelo del dominio, mostrándose además las descripciones de cada una de las clases que intervienen. Se presenta también un listado de los requerimientos funcionales y no funcionales que indican las capacidades o condiciones que el sistema debe cumplir y las propiedades o cualidades que el cliente debe tener. Por último se seleccionan los requerimientos o grupos de requerimientos funcionales con los que se va a conformar los casos de uso del sistema, y estos son descritos al igual que sus actores.

Capítulo 3: Diseño del Sistema

En el presente capítulo se realiza una descripción detallada del sistema propuesto, donde se describe la arquitectura a utilizar, además en vista a lograr una correcta implementación se realizan los diagramas de clases de diseño, los diagramas de secuencia y las descripciones de las clases.

3.1 Descripción de la arquitectura

Para el desarrollo del sistema se debe tener en cuenta que la arquitectura es una vista estructural de alto nivel, ocurre muy tempranamente en el ciclo de vida y define los estilos o grupos de estilos adecuados para cumplir con los requerimientos no funcionales, por lo que es un elemento de vital importancia para el desarrollo del sistema.

El desarrollo del sistema se basa en una vista modular, cuyo objetivo es la separación de las capas de presentación, de negocio y de acceso a datos, que logra la comunicación entre ellas, se usa porque en caso de que haya algún cambio solo se cambia el nivel en el que sucedió el cambio.

Otros patrones utilizados son los patrones GRASP que describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. A las clases se le asignaron las responsabilidades que podían realizar según la información contenida.

La Capa Intermedia (Middleware) forma parte de la solución del problema, a través de la misma se accede al repositorio. Esta subcapa tiene a su cargo la ejecución la lógica de acceso a datos y contiene dos paquetes de clases los repositorios y fábricas de objetos. Los repositorios son los encargados de manejar las colecciones de Objetos Comunes (Entidades de la BD representados como objetos) y realizar operaciones sobre ellas. Las fábricas de objetos son paquetes de clases que contiene la funcionalidad necesaria para acceder a la base de datos y realizar las operaciones que se explican a continuación.

Por cada entidad mapeada desde la base de datos, existen cuatro clases que heredan de las interfaces:

IInsertFactory: encargada del proceso de inserción.

IDeleteFactory: Encargada de suprimir una entidad determinada.

IUpdateFactory: encargada de actualizar atributos de los objetos en la BD.

ISelectFactory: encargada del proceso de selección.

Es válido mencionar que existen por cada entidad otra clase que hereda de la clase interfaz IDomainObjectFactory, encargada de realizar el proceso de mapeo de las entidades (convertir tupla a tupla el resultado de un proceso de selección en la entidad a la que corresponde). Para enlazar la capa de Middleware con el Acceso a Datos que está en el servidor de BD se requiere el uso la NpgSQL. Interfaz de Programación de Aplicación (API), encargada de la comunicación de una aplicación .Net con servidores de Bases de Datos PostgreSQL. Este Acceso a Datos está conformado por un conjunto de funciones programadas en lenguaje PlpgSQL. (28)

3.2 Modelo de diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso, centrándose en los requisitos no funcionales y otras restricciones del entorno de implementación que tienen impacto en la herramienta, por tanto, define las clases del diseño que conformarán la herramienta a implementar.

3.3 Diagramas de clases del diseño

Los diagramas de clases del diseño describen gráficamente las especificaciones de las clases del software y contienen las clases, atributos, métodos, navegabilidad y dependencias existentes entre ellas. (29)

A continuación se representan los diagramas de clase del diseño por casos de uso de la herramienta.

3.3.1 CUS Crear criterios para validar requisitos del producto

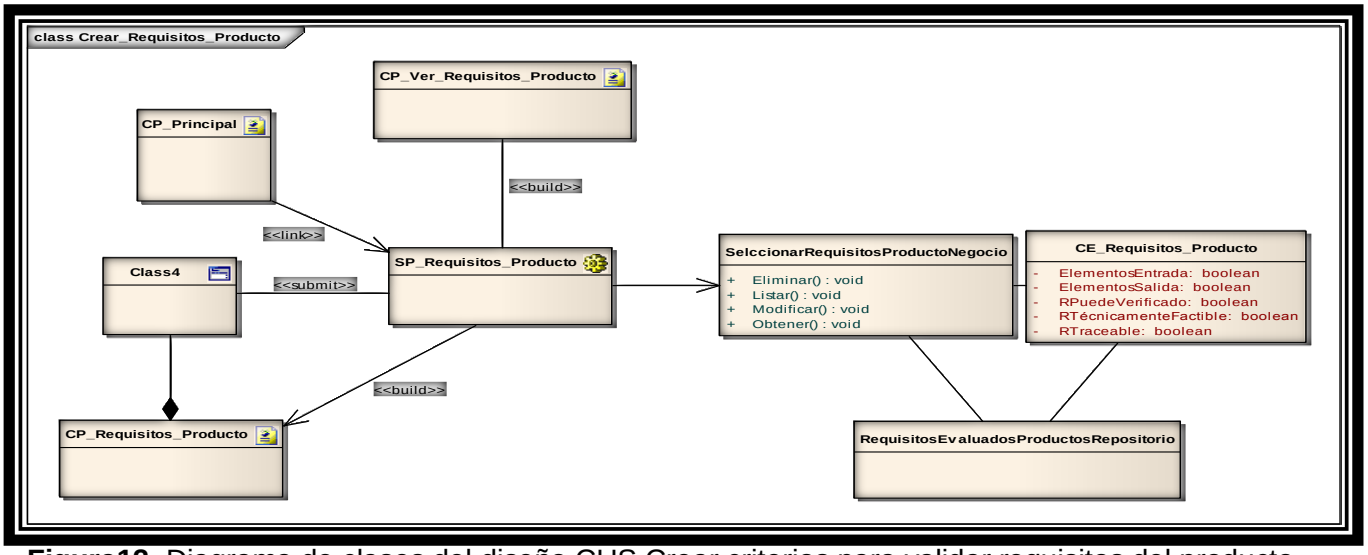


Figura12. Diagrama de clases del diseño CUS Crear criterios para validar requisitos del producto

3.3.2 CUS Crear criterios para validar requisitos del cliente

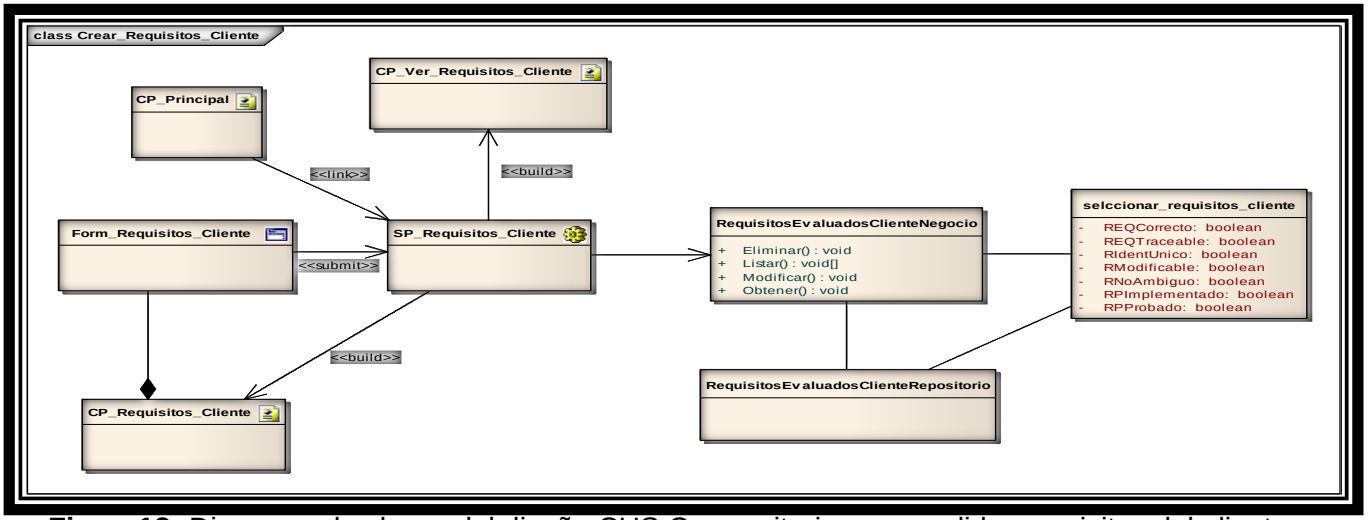


Figura13: Diagrama de clases del diseño CUS Crear criterios para validar requisitos del cliente

3.3.3 CUS Crear catálogo de proveedores de requisitos válidos

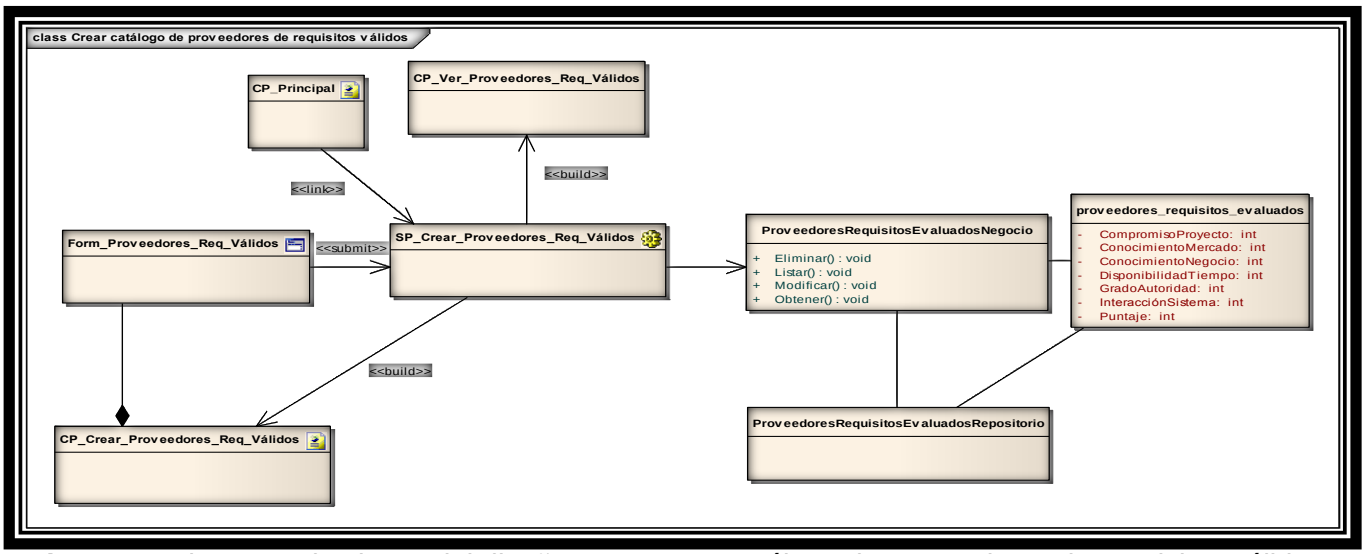


Figura14. Diagrama de clases del diseño CUS Crear catálogo de proveedores de requisitos válidos

3.3.4 CUS Crear catálogo de casos de uso

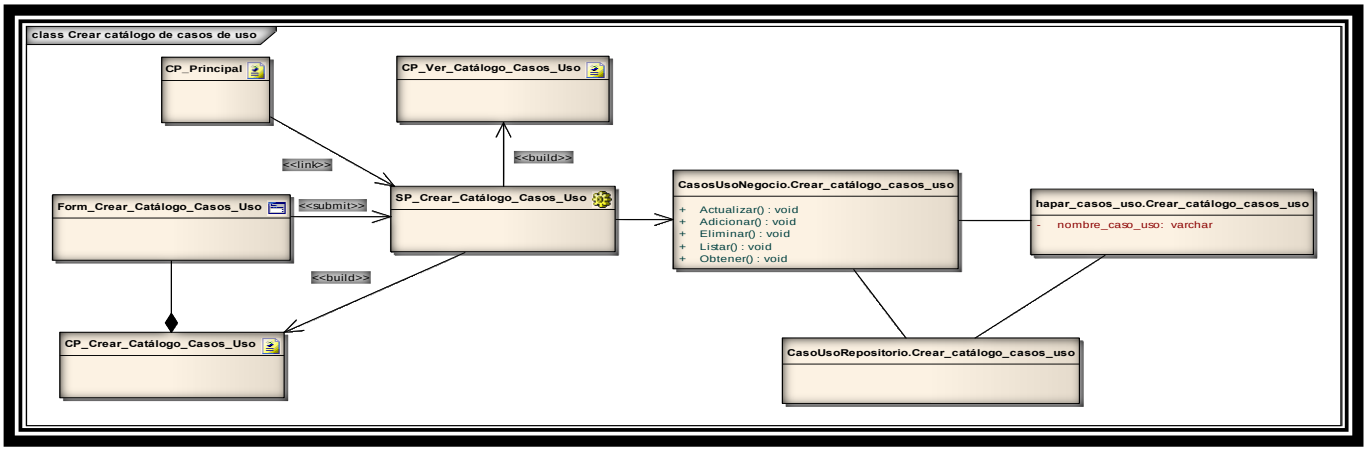


Figura15. Diagrama de clases del diseño CUS Crear catálogo de casos de uso

3.3.5 CUS Crear matriz de disponibilidad de proveedores

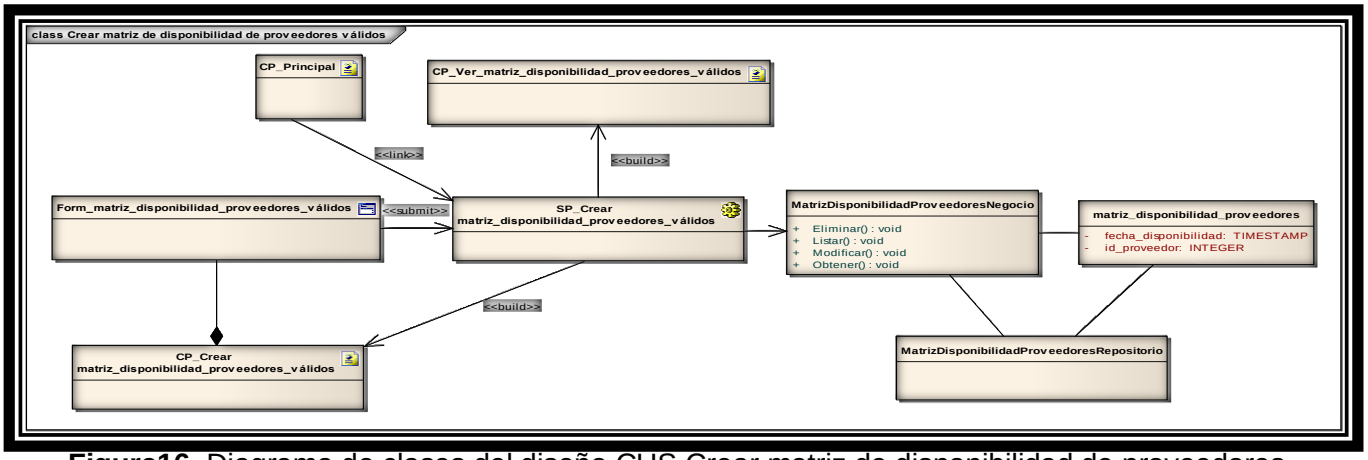


Figura16. Diagrama de clases del diseño CUS Crear matriz de disponibilidad de proveedores

3.4 Diagramas de secuencia

Los diagramas de secuencia muestran los objetos y sus relaciones, incluyen los mensajes que pueden ser enviados entre ellos, además, se observan las interacciones entre objetos ordenadas en una secuencia de tiempo. (30)

3.4.1 CUS Crear catálogo de casos de uso

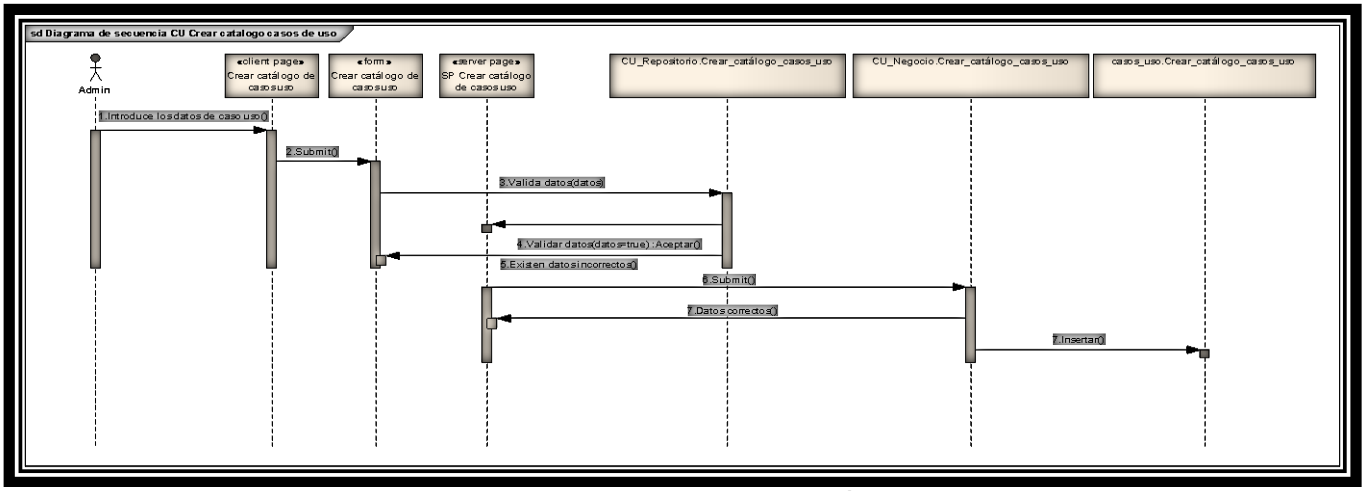


Figura17: Diagrama de secuencia CUS Crear catálogo de casos de uso

3.4.2 CUS Crear criterios para validar requisitos del cliente

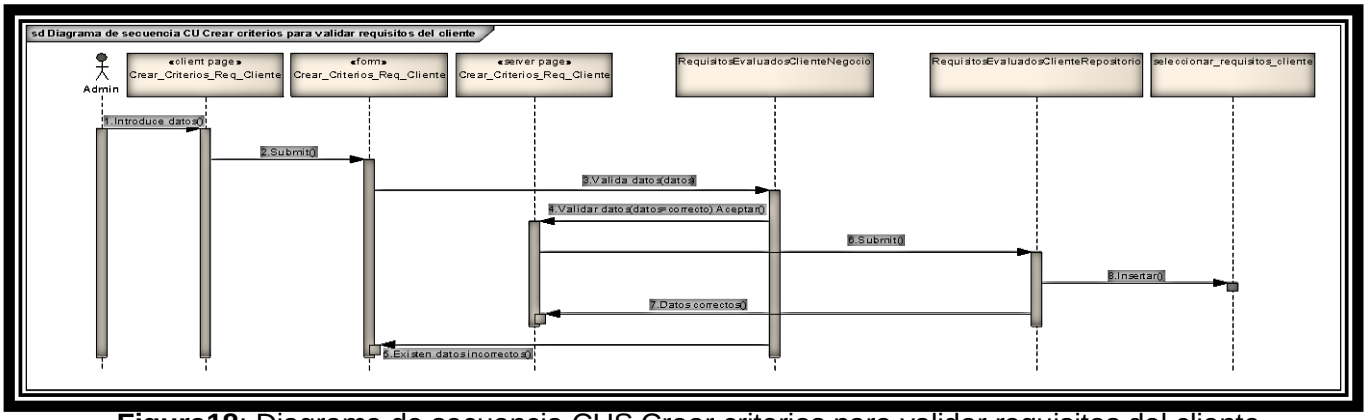


Figura18: Diagrama de secuencia CUS Crear criterios para validar requisitos del cliente

3.5 Descripción de las clases

3.5.1 Clase Crear catálogo de proveedores de requisitos

Nombre: Crear catálogo de proveedores de requisitos	
Tipo de clase: Controladora	
Atributo	**Tipo**
Nombre	String
Primer apellido	String
Segundo apellido	String
Teléfono	String
Correo electrónico	String
Área de trabajo	String
Cargo	String
Dirección	String
Aceptar	Botón
Cancelar	Botón

Para cada responsabilidad:	
Nombre:	Crear catálogo de proveedores de requisitos.
Descripción:	Permite crear el catálogo de proveedores de requisitos.

Tabla1: Descripción de la clase Crear catálogo de proveedores de requisitos

3.5.2 Clase Crear catálogo de casos de uso

Nombre: Crear catálogo de casos de uso	
Tipo de clase: Controladora	
Atributo	**Tipo**
Nombre_CU	String
Aceptar	Botón
Cancelar	Botón
Para cada responsabilidad:	
Nombre:	Crear catálogo de casos de uso
Descripción:	Permite crear catálogo de casos de uso

Tabla3: Descripción de la clase Crear catálogo de casos de uso

3.6 Modelo de Datos

Un modelo de datos es un lenguaje utilizado para la descripción de una base de datos. Por lo general, un modelo de datos permite describir las estructuras de datos de la base (el tipo de los datos que incluye la base y la forma en que se relacionan), las restricciones de integridad (las condiciones que los datos deben cumplir para reflejar correctamente la realidad deseada) y las operaciones de manipulación de los datos (agregado, borrado, modificación y recuperación de los datos de la base). (31)

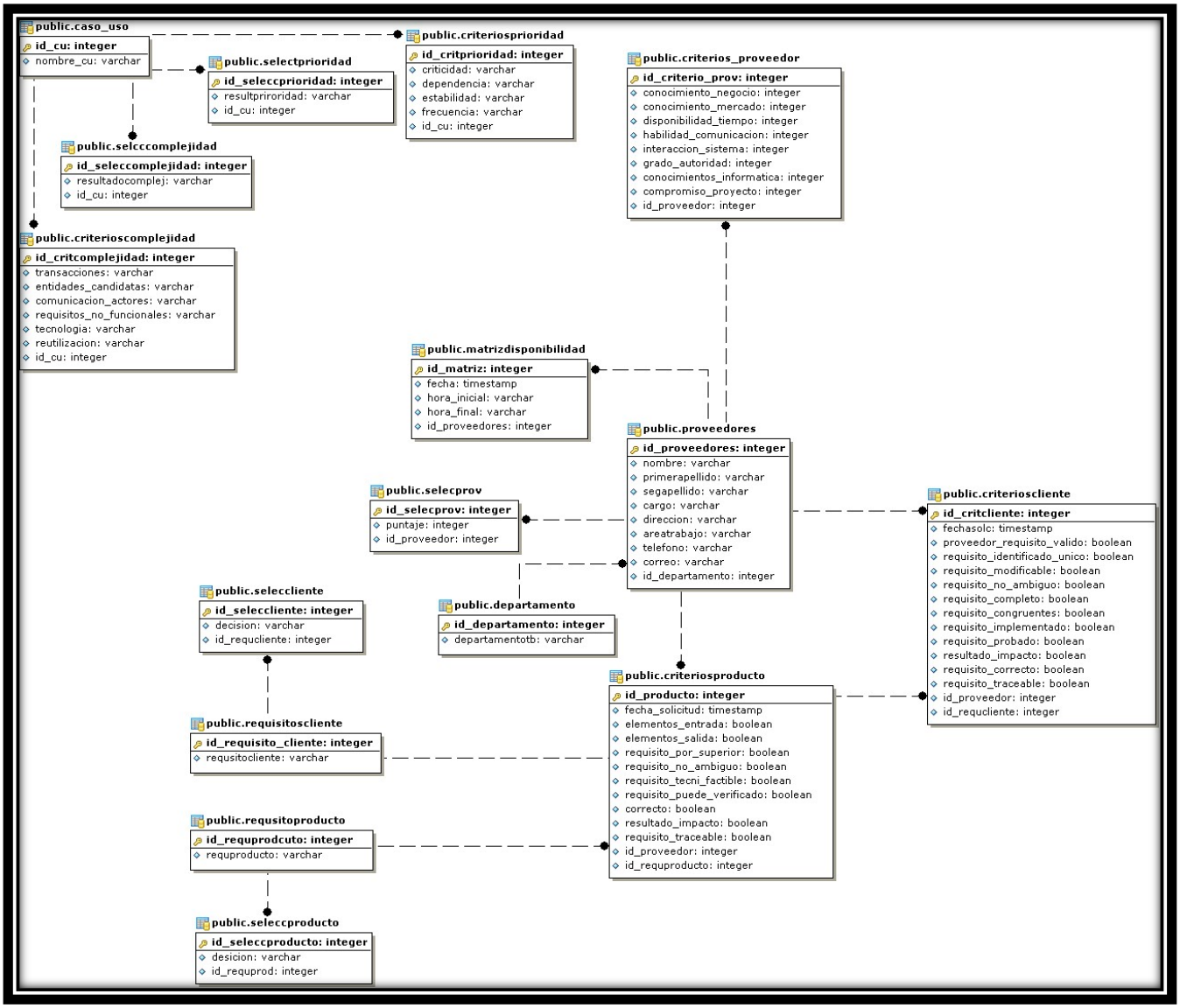


Figura19: Diagrama del Modelo de Datos

En el presente capítulo se obtuvo el diagrama de clases del diseño para cada caso de uso del sistema, donde se representaron las relaciones que se establecen entre las clases. Además, quedaron definidos los diagramas de secuencia, las descripciones de las clases y la arquitectura de la aplicación, también se realizó el Modelo de Datos, en el cual se observan todas las tablas de la base de datos con sus atributos, así como la relación entre ellas.

Capítulo 4: Implementación y Prueba

4.1 Modelo de implementación

El modelo de implementación describe cómo los elementos de diseño se implementan en componentes, ficheros de código fuente, ejecutables, entre otros. Los diagramas de despliegue y componentes conforman lo que se conoce como un modelo de implementación. (32)

4.2 Diagrama de componentes

Los diagramas de componentes ilustran las piezas del software que conformarán un sistema. Un diagrama de componentes tiene un nivel más alto de abstracción que un diagrama de clase, usualmente un componente se implementa por una o más clases en tiempo de ejecución. (33)

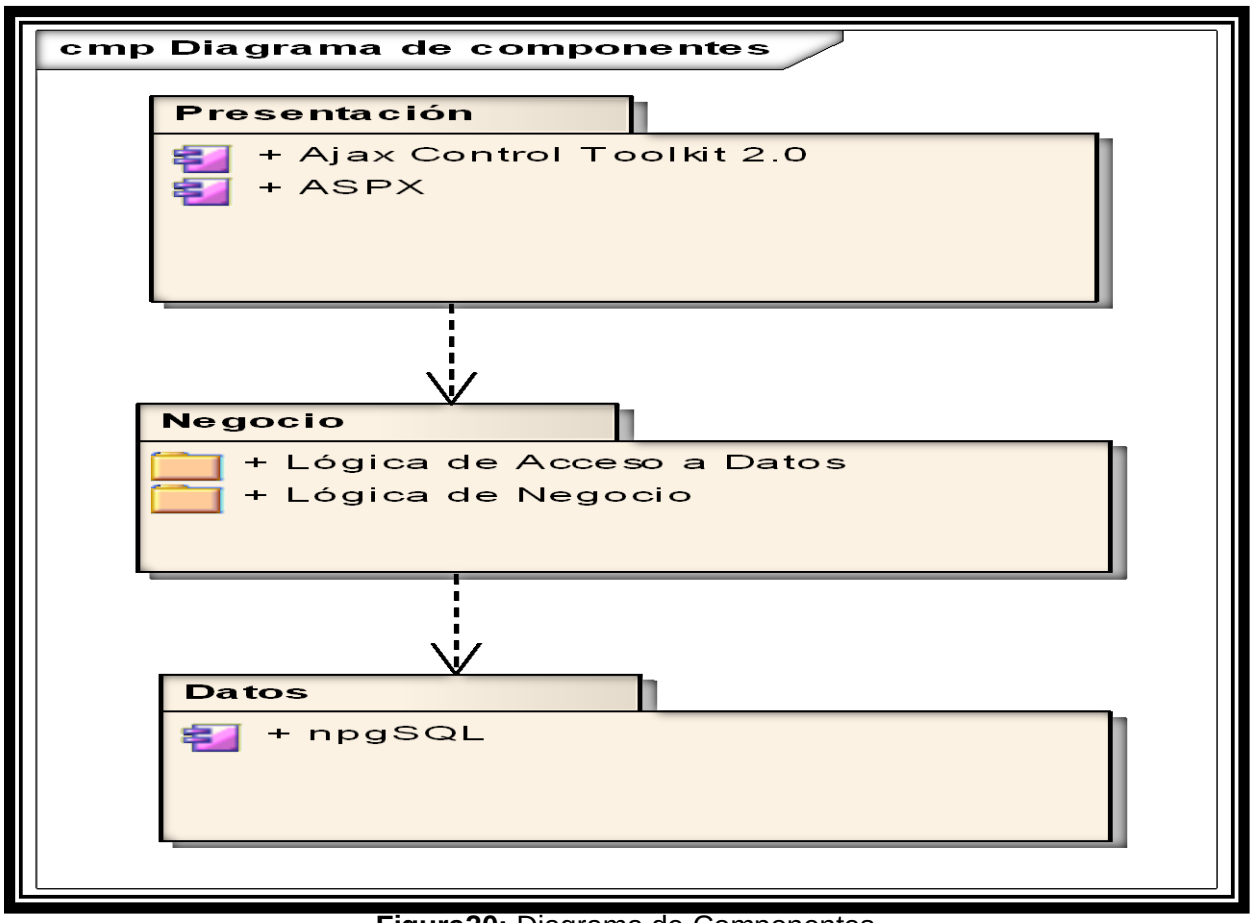


Figura20: Diagrama de Componentes

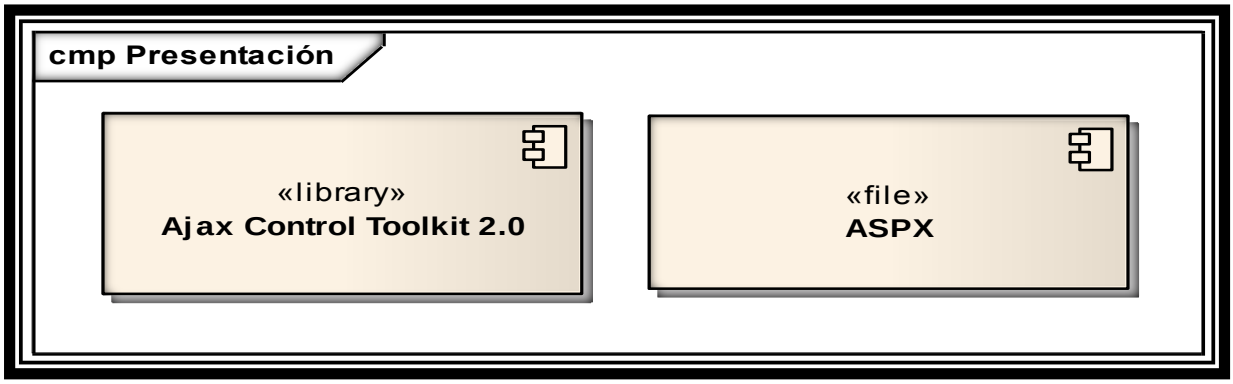


Figura21: Vista detallada del paquete Presentación

Figura22

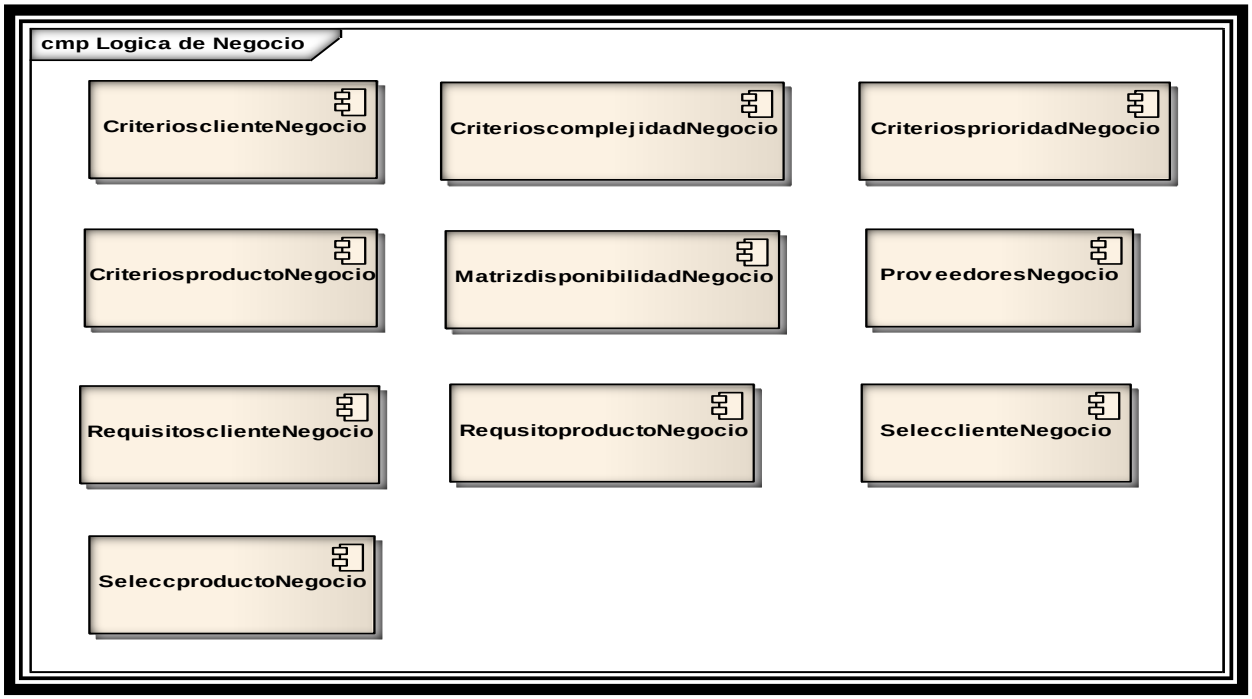


Figura23

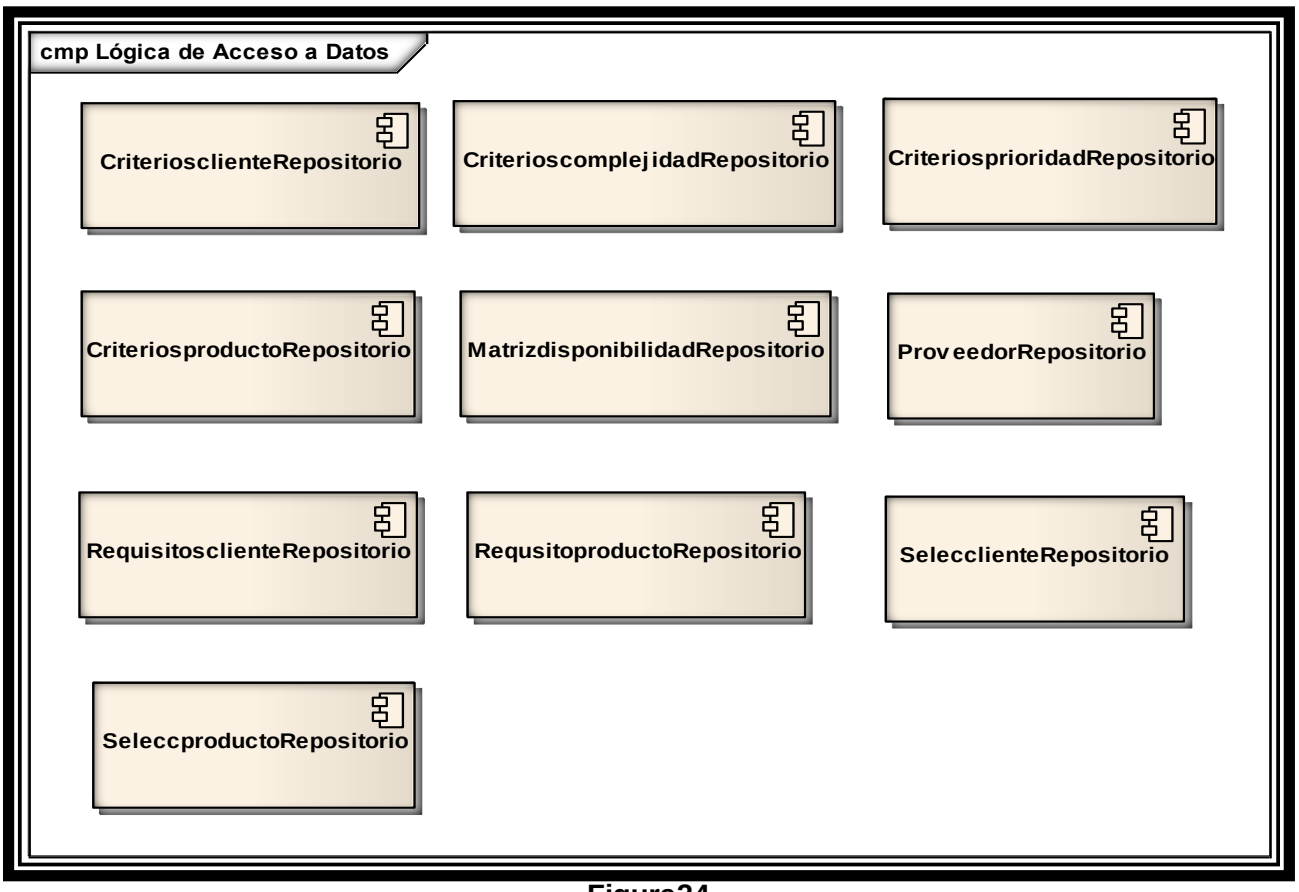


Figura24

Figura22, 23,24: Vista detallada del paquete Negocio

Figura25: Vista detallada del paquete Datos

4.3 Diagrama de despliegue

Los diagramas de despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. (34)

A continuación se representa el diagrama de despliegue de la herramienta desarrollada el cual consta con una computadora cliente, en la cual los usuarios pueden visualizar e interactuar con la aplicación que se encuentra en el servidor de aplicación. Este servidor es el encargado de responder las peticiones de las computadoras clientes y este a su vez está conectado al servidor de base de datos, el cual es el encargado de administrar los datos de la herramienta alasHapar.

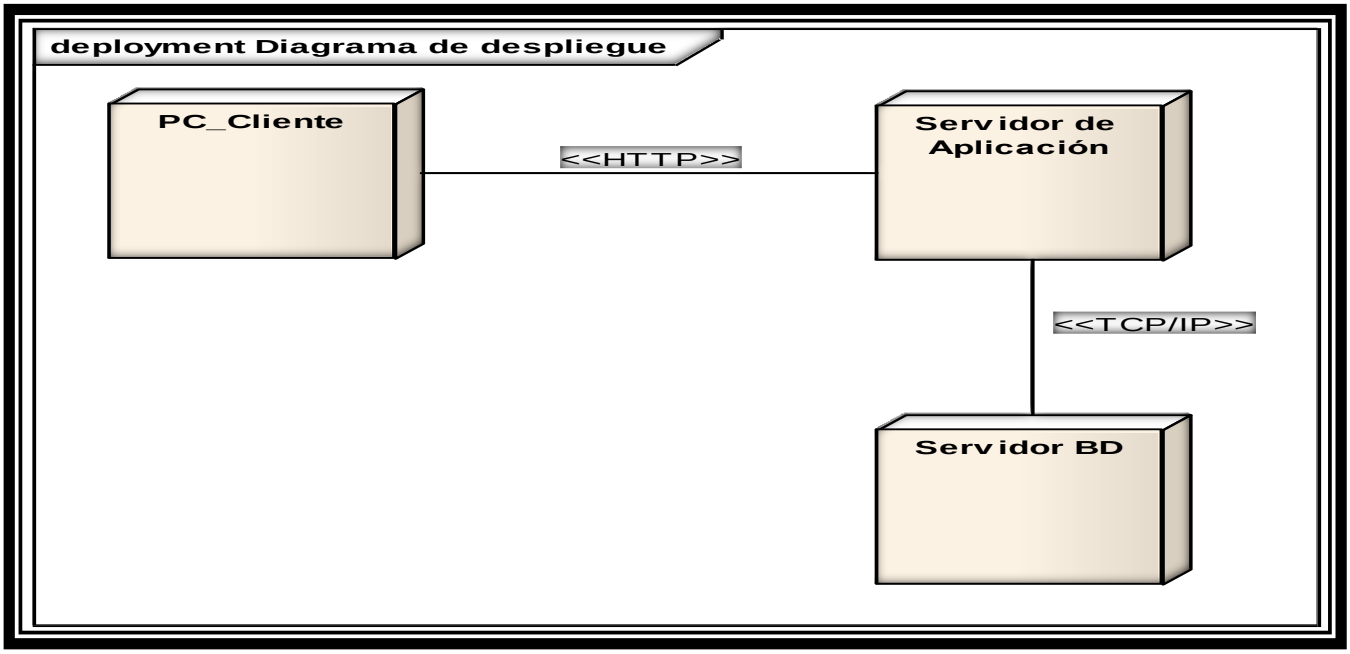


Figura26: Diagrama de Despliegue

4.4 Pruebas

Es el proceso de ejecutar un programa con el fin de encontrar errores, en la cual un sistema o uno de sus componentes se ejecutan en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto. (35)

La prueba es un elemento crítico para la calidad del software. La importancia de los costos asociados a los errores promueve la definición y aplicación de un proceso de pruebas minuciosas y bien planificadas. Las pruebas permiten validar y verificar el software, entendiendo como validación del software el proceso que determina si el software satisface los requisitos, y verificación como el proceso que determina si los productos de una fase satisfacen las condiciones de dicha fase. (36)

4.5 Pruebas de Caja Negra

Las pruebas de caja negra se llevan a cabo sobre la interfaz del software, obviando el comportamiento interno y la estructura del programa. (37)

Los casos de prueba de la caja negra pretenden demostrar que:

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

Las pruebas de caja negra pretenden encontrar estos tipos de errores:

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

4.6 Casos de prueba (CP)

Los casos de pruebas tienen como objetivo fundamental determinar si los requisitos d la aplicación son parcial o completamente satisfactorios a través de funciones determinadas. Estos casos de pruebas proporcionan una descripción de puntos importantes de observación, con pruebas positivas y negativas para lograr que la mayoría de los requisitos de la aplicación sean debidamente probados. (38)

4.7 Diseño de CP para el CU Crear catálogo de proveedores de requisitos

Descripción General: El caso de uso inicia cuando el Admin accede a la opción Crear catálogo de proveedores de requisitos.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad.	Flujo Central
C1: Crear catálogo de proveedores de requisitos.	EC 1.1: Crear catálogo de proveedores de requisitos.	Se crea un catálogo de proveedores de requisitos si la información es correcta.	• El administrador selecciona la opción **Catálogo de proveedores**. • El sistema muestra los campos necesarios para que el administrador introduzca los datos del catálogo. • El administrador introduce los datos del catálogo y selecciona la opción **Aceptar**. • El sistema verifica que los datos estén correctamente y crea el catálogo.

	EC 1.2: Existen datos incompletos.	Se verifica que los campos estén completos.	• El sistema muestra un indicador sobre los campos incompletos. • El administrador llena los campos vacíos. • El sistema crea el catálogo de proveedores y muestra el catálogo creado.
	EC 1.3: Existen datos incorrectos.	Se verifica que la información sea correcta.	• El sistema muestra un indicador sobre los campos incorrectos. • El administrador arregla la información. • El sistema crea el catálogo de proveedores y muestra el catálogo creado.

Figura18: Descripción del Caso de prueba Crear catálogo de proveedores de requisitos.

4.8 Diseño de CP para el CU Ver detalles de proveedores de requisitos

Descripción General: El caso de uso inicia cuando el Admin acepta la opción Crear catálogo de proveedores de requisitos.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad.	Flujo Central
SC1: Ver detalles de proveedores de requisitos.	EC 1.1: Ver detalles de proveedores de requisitos.	Se muestra el catálogo con toda la información creada por el administrador.	• El administrador selecciona la opción **Aceptar**. • El sistema muestra los datos del catálogo.

	EC 1.2: Modificar del catálogo de proveedores de requisitos.	Permite modificar la información del catálogo.	• El administrador selecciona la opción de **Modificar catálogo de proveedores de requisitos**. • El sistema muestra el formulario de **Catálogo de proveedores de requisitos**. • El administrador modifica la información.
	EC 1.3: Eliminar catálogo de proveedores de requisitos.	Permite eliminar un catálogo seleccionado por el administrador.	• El administrador selecciona la opción **Eliminar catálogo de proveedores de requisitos** • El sistema muestra un mensaje de confirmación *¿Está seguro que desea eliminar el proveedor de requisito seleccionado?* • El administrador selecciona la opción **SI**. • El sistema elimina el catálogo de la base de datos de la aplicación.

Figura29: Descripción del Caso de prueba Ver detalles de proveedores de requisitos.

En este capítulo se obtuvo el diagrama de componentes, que muestra cómo se encuentra dividido por paquetes de clases la herramienta alasHapar, agrupando las clases en 3 grupos fundamentales: la de Presentación, Negocio y Datos. También se realizó el diagrama de despliegue representando los distintos nodos que componen el sistema. Además se describieron los casos de prueba de los diferentes casos de uso de la herramienta, lo que trajo consigo la obtención de un producto final con mayor calidad.

Conclusiones

Con la investigación se logró desarrollar la herramienta alasHapar, y se ha cumplido con el objetivo y las tareas planteadas. A continuación se plantean los principales resultados:

- El análisis del estado del arte reveló que existen herramientas de apoyo a la administración de requisitos, pero la mayoría tienen la desventaja de no contar con funcionalidades referente a los proveedores válidos de requisitos, a las matrices de disponibilidad de los proveedores válidos de requisitos y a las evaluaciones de casos de uso, ya sea evaluación por prioridad o por complejidad, además algunas eran herramientas propietarias. Por lo que se determinó que no satisfacían las necesidades del CESIM.

- Después de analizar las posibles herramientas y tecnologías para el desarrollo de la herramienta se decidió implementar la herramienta basándose en el patrón de arquitectura tres capas, donde se encuentran la capa de Presentación, la capa de Negocio y la capa de Datos. Se utilizó CSharp como lenguaje de programación y Visual Studio 2008 como IDE (Entorno de Desarrollo Integrado) sobre .NET Framework y como gestor de base de datos el PostgreSQL.

- Como resultado de los flujos que propone la metodología RUP, se obtuvieron varios artefactos necesarios para obtener una visión detallada de la herramienta desarrollada.

- Se desarrolló una herramienta web que satisface las principales necesidades referentes a la creación de proveedores válidos de requisitos. Así como, a la evaluación de caso de uso por prioridad y por complejidad, a la creación de matrices de disponibilidad de proveedores válidos y a la validación de requisitos del cliente y requisitos del producto.

- Se desarrolló una herramienta que cumple con los requisitos de seguridad necesarios para garantizar la confidencialidad de la información que se manipula en la misma.

Recomendaciones

Luego de haber concluido la herramienta propuesta y haber cumplido los objetivos trazados, se plantean las siguientes recomendaciones:

- Durante el desarrollo de la aplicación se realizaron pruebas de compatibilidad con el Framework Mono las cuales resultaron positivas, por lo que se recomienda usar ese framework a la hora de efectuar una migración a entornos basados en el sistema operativo GNU/Linux.

- Incorporar nuevas funcionalidades a la herramienta como pueden ser crear lista de chequeo para detectar inconsistencias o crear registro de revisiones de inconsistencias, todas ellas vinculadas al proceso de administración de requisitos.

- La herramienta desarrollada está implementada para trabajar a nivel de departamentos, por lo cual se recomienda hacer algunas modificaciones que permitan soportar un flujo de trabajo de varios departamentos simultáneamente, y se logre un mayor alcance de la aplicación, necesitándose solamente una instancia a nivel de centro.

- Integrar el software desarrollado con la herramienta Gestión de la Información del Expediente de Proyecto, con el objetivo de correlacionar toda la información al realizar el cierre del expediente del proyecto en desarrollo.

Referencias Bibliográficas

1. eumed. [En línea] [Citado el: 12 de 11 de 2010.] http://www.eumed.net/libros/2009a/514/Caracterizacion%20de%20la%20Universidad%20de%20las%20Ciencias%20Informaticas.htm.

2. **Días, Andy y Ruíz, Osmel.** *Sistema Integrado para Bibliotecas.* Ciudad de la Habana : s.n., 2007.

3. germinus. [En línea] [Citado el: 14 de 12 de 2010.] http://www.germinus.com/htm/02/CasosExito/IntegracionyDesarrollodeSistemas/CMMI1PARTE.pdf.

4. calisof. [En línea] [Citado el: 14 de 11 de 2010.] calisof.uci.cu.

5. infor. [En línea] [Citado el: 14 de 11 de 2010.] http://www.infor.uva.es/~jsalama1/calsoft/Tema1.pdf.

6. ines. [En línea] 2007. [Citado el: 16 de 11 de 2010.] http://www.ines.org.es/vulcano/wp-content/uploads/2007/09/d6-estudio-de-herramientas-de-certificacion-tid.pdf.

7. rational. [En línea] [Citado el: 17 de 11 de 2010.] http://www.rational.com.ar/herramientas/requisitepro.html.

8. **Perez, Yunelis y Rosabal, Antonio.** *Perez, Yunelis & Rosabal, Antonio. . Sistema para la Gestión y Análisis de la Información Estadística en la Salud Pública Cubana: Subsistema Editor de Plantillas.* Ciudad de la Habana : UCI, 2009.

9. **García Ferrero, Adrián y Fernández Castellano, Yasiel.** *Generador automático de plantillas en la Web.* Ciudad de La Habana : UCI, 2010.

10. usmp. [En línea] [Citado el: 19 de 11 de 2010.] www.usmp.edu.pe/publicaciones/boletin/fia/info21/uml.htm.

11. msdn. [En línea] [Citado el: 20 de 11 de 2010.] http://msdn.microsoft.com/es-es/library/z1zx9t92%28VS.80%29.aspx..

12. **Seguí, Arian y Fernández, Dariel.** *Herramienta para la gestión de la información.* Ciudad de la Habana : UCI, 2009.

13. elartedeprogramar. [En línea] 2008. [Citado el: 22 de 11 de 2010.] http://www.elartedeprogramar.cl/csharp-developed-visual-studio-2008-monodevelop.

14. sparxsystems. [En línea] [Citado el: 23 de 11 de 2010.] http://www.sparxsystems.com.ar.

15. bdigital. [En línea] [Citado el: 24 de 11 de 2010.] http://bdigital.eafit.edu.co/bdigital/PROYECTO/P005.12CDT629/capitulo6.pdf.

16. migueljaque. [En línea] [Citado el: 25 de 11 de 2010.] http://migueljaque.com/index.php/tecnicas/tecnicasmodnegocio/37-modelado_negocio/46-modelo-de-dominio?tmpl=component&print=1&page.

17. petra. [En línea] [Citado el: 11 de 1 de 2011.] http://petra.euitio.uniovi.es/~i1643233/Analisis_de_herramientas-v03.pdf.

18. **Quiñones, Ernesto.** www.postgresql.org.pe. *www.postgresql.org.pe.* [En línea] [Citado el: 20 de 1 de 2011.] http://www.postgresql.org.pe/articles/introduccion_a_postgresql.pdf.

19. www.eumed.net. *www.eumed.net.* [En línea] [Citado el: 25 de 1 de 2011.] http://www.eumed.net/libros/2010b/698/Requisitos%20funcionales.htm.

20. zoftwar.blogspot.com. *zoftwar.blogspot.com.* [En línea] [Citado el: 26 de 1 de 2011.] http://zoftwar.blogspot.com/2006/11/requisitos-no-funcionales-parte-1-con.html.

21. monografias.com. *monografias.com.* [En línea] [Citado el: 15 de 2 de 2011.] www.monografias.com/trabajos-pdf4/diagramas-secuencia/diagramas-secuencia.pdf.

22. **Visconti, Marcello y Astudillo, Hernán.** www.inf.utfsm.cl. *www.inf.utfsm.cl.* [En línea] [Citado el: 5 de 4 de 2011.] http://www.inf.utfsm.cl/~visconti/ili236/Documentos/15-Implementacion.pdf.

23. www.itescam.edu.mx. *www.itescam.edu.mx.* [En línea] 2000. [Citado el: 6 de 4 de 2011.] http://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r55051.PDF.

24. www.sparxsystems.com.ar. *www.sparxsystems.com.ar.* [En línea] [Citado el: 7 de 4 de 2011.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_componentdiagram.html.

25. definicion.de. [En línea] 2008. [Citado el: 27 de 4 de 2011.] http://definicion.de/modelo-de-datos/.

26. asprotech.blogspot.com. *asprotech.blogspot.com.* [En línea] 2011. [Citado el: 8 de 12 de 2010.] http://asprotech.blogspot.com/2011/03/requisitos-del-cliente-producto.html.

27. alarcos.inf-cr.uclm.es. *alarcos.inf-cr.uclm.es.* [En línea] [Citado el: 31 de 5 de 2011.] http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema09.pdf.

28. www.willydev.net. [En línea] [Citado el: 31 de 5 de 2011.] http://www.willydev.net/descargas/oguzman-diseno_pruebas.pdf.

29. indalog.ual.es. *indalog.ual.es.* [En línea] [Citado el: 31 de 5 de 2011.] http://indalog.ual.es/mtorres/LP/Prueba.pdf.

30. **Ramos Carrillo,Anay .** eumed. [En línea] 2009. [Citado el: 21 de 11 de 2010.] http://www.eumed.net/libros/2009c/587/indice.htm..

31. **lópez,Lara.** ines. [En línea] 2009. http://www.ines.org.es/vulcano/wp-content/uploads/2010/04/d36.pdf.

32. **Rigoni Brualla,Cecilia .** mityc. [En línea] [Citado el: 14 de 11 de 2010.] http://www.mityc.es/dgdsi/es-ES/Servicios/Biblioteca%20Jornadas/Jornadas/s01CeciliaRigoni.pdf.

33. **Arias Chaves, Michel.** *La ingeniería de requerimientos y su importancia en el desarrollo de sofware.* s.l. : Universidad de Costa, 2005.

34. **González Méndez, Bradier.** *Sistema de Gestión de Datos Geológicos. Módulo: Inventario de Minerales Sólidos.* Ciudad de la Habana : UCI, 2010.

35. **Mendoza Navarro, Javier.** sisbib.unmsm.edu.pe. *sisbib.unmsm.edu.pe.* [En línea] [Citado el: 10 de 2 de 2011.] http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/basic/mendoza_nj/cap5.pdf.

36. **Rabelo Rodriguez, Ing.Alexander.** *CESIM_BQO_0120_Arquitectura de software v2.0.* 2010.

37. *Ingeniería Informática.* **Zulueta Véliz, Ing. Yeleny.** 14, 2007.

38. *LOS MAPAS CONCEPTUALES EN LA MEJORA DE PROCESOS EN EL DESARROLLO DE SOFTWARE.* **Estrada Sent, Vivian y Febles Estrada, Ailin.** 2010.

Bibliografía

1. eumed. [En línea] [Citado el: 12 de 11 de 2010.] http://www.eumed.net/libros/2009a/514/Caracterizacion%20de%20la%20Universidad%20de%20las%20Ciencias%20Informaticas.htm.

2. **Días, Andy y Ruíz, Osmel.** *Sistema Integrado para Bibliotecas.* Ciudad de la Habana : s.n., 2007.

3. germinus. [En línea] [Citado el: 14 de 12 de 2010.] http://www.germinus.com/htm/02/CasosExito/IntegracionyDesarrollodeSistemas/CMMI1PARTE.pdf.

4. calisof. [En línea] [Citado el: 14 de 11 de 2010.] calisof.uci.cu.

5. infor. [En línea] [Citado el: 14 de 11 de 2010.] http://www.infor.uva.es/~jsalama1/calsoft/Tema1.pdf.

6. ines. [En línea] 2007. [Citado el: 16 de 11 de 2010.] http://www.ines.org.es/vulcano/wp-content/uploads/2007/09/d6-estudio-de-herramientas-de-certificacion-tid.pdf.

7. rational. [En línea] [Citado el: 17 de 11 de 2010.] http://www.rational.com.ar/herramientas/requisitepro.html.

8. **Perez, Yunelis y Rosabal, Antonio.** *Perez, Yunelis & Rosabal, Antonio. . Sistema para la Gestión y Análisis de la Información Estadística en la Salud Pública Cubana: Subsistema Editor de Plantillas.* Ciudad de la Habana : UCI, 2009.

9. **García Ferrero, Adrián y Fernández Castellano, Yasiel.** *Generador automático de plantillas en la Web.* Ciudad de La Habana : UCI, 2010.

10. usmp. [En línea] [Citado el: 19 de 11 de 2010.] www.usmp.edu.pe/publicaciones/boletin/fia/info21/uml.htm.

11. msdn. [En línea] [Citado el: 20 de 11 de 2010.] http://msdn.microsoft.com/es-es/library/z1zx9t92%28VS.80%29.aspx..

12. **Seguí, Arian y Fernández, Dariel.** *Herramienta para la gestión de la información.* Ciudad de la Habana : UCI, 2009.

13. elartedeprogramar. [En línea] 2008. [Citado el: 22 de 11 de 2010.] http://www.elartedeprogramar.cl/csharp-developed-visual-studio-2008-monodevelop.

14. sparxsystems. [En línea] [Citado el: 23 de 11 de 2010.] http://www.sparxsystems.com.ar.

15. bdigital. [En línea] [Citado el: 24 de 11 de 2010.] http://bdigital.eafit.edu.co/bdigital/PROYECTO/P005.12CDT629/capitulo6.pdf.

16. migueljaque. [En línea] [Citado el: 25 de 11 de 2010.] http://migueljaque.com/index.php/tecnicas/tecnicasmodnegocio/37-modelado_negocio/46-modelo-de-dominio?tmpl=component&print=1&page.

17. petra. [En línea] [Citado el: 11 de 1 de 2011.] http://petra.euitio.uniovi.es/~i1643233/Analisis_de_herramientas-v03.pdf.

18. **Quiñones, Ernesto.** www.postgresql.org.pe. *www.postgresql.org.pe.* [En línea] [Citado el: 20 de 1 de 2011.] http://www.postgresql.org.pe/articles/introduccion_a_postgresql.pdf.

19. www.eumed.net. *www.eumed.net.* [En línea] [Citado el: 25 de 1 de 2011.] http://www.eumed.net/libros/2010b/698/Requisitos%20funcionales.htm.

20. zoftwar.blogspot.com. *zoftwar.blogspot.com.* [En línea] [Citado el: 26 de 1 de 2011.] http://zoftwar.blogspot.com/2006/11/requisitos-no-funcionales-parte-1-con.html.

21. monografias.com. *monografias.com.* [En línea] [Citado el: 15 de 2 de 2011.] www.monografias.com/trabajos-pdf4/diagramas-secuencia/diagramas-secuencia.pdf.

22. **Visconti, Marcello y Astudillo, Hernán.** www.inf.utfsm.cl. *www.inf.utfsm.cl.* [En línea] [Citado el: 5 de 4 de 2011.] http://www.inf.utfsm.cl/~visconti/ili236/Documentos/15-Implementacion.pdf.

23. www.itescam.edu.mx. *www.itescam.edu.mx.* [En línea] 2000. [Citado el: 6 de 4 de 2011.] http://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r55051.PDF.

24. www.sparxsystems.com.ar. *www.sparxsystems.com.ar.* [En línea] [Citado el: 7 de 4 de 2011.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_componentdiagram.html.

25. definicion.de. [En línea] 2008. [Citado el: 27 de 4 de 2011.] http://definicion.de/modelo-de-datos/.

26. asprotech.blogspot.com. *asprotech.blogspot.com.* [En línea] 2011. [Citado el: 8 de 12 de 2010.] http://asprotech.blogspot.com/2011/03/requisitos-del-cliente-producto.html.

27. alarcos.inf-cr.uclm.es. *alarcos.inf-cr.uclm.es.* [En línea] [Citado el: 31 de 5 de 2011.] http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema09.pdf.

28. www.willydev.net. [En línea] [Citado el: 31 de 5 de 2011.] http://www.willydev.net/descargas/oguzman-diseno_pruebas.pdf.

29. indalog.ual.es. *indalog.ual.es.* [En línea] [Citado el: 31 de 5 de 2011.] http://indalog.ual.es/mtorres/LP/Prueba.pdf.

30. **Ramos Carrillo,Anay .** eumed. [En línea] 2009. [Citado el: 21 de 11 de 2010.] http://www.eumed.net/libros/2009c/587/indice.htm..

31. **lópez,Lara.** ines. [En línea] 2009. http://www.ines.org.es/vulcano/wp-content/uploads/2010/04/d36.pdf.

32. **Rigoni Brualla,Cecilia .** mityc. [En línea] [Citado el: 14 de 11 de 2010.] http://www.mityc.es/dgdsi/es-ES/Servicios/Biblioteca%20Jornadas/Jornadas/s01CeciliaRigoni.pdf.

33. **Arias Chaves, Michel.** *La ingeniería de requerimientos y su importancia en el desarrollo de sofware.* s.l. : Universidad de Costa, 2005.

34. **González Méndez, Bradier.** *Sistema de Gestión de Datos Geológicos. Módulo: Inventario de Minerales Sólidos.* Ciudad de la Habana : UCI, 2010.

35. **Mendoza Navarro, Javier.** sisbib.unmsm.edu.pe. *sisbib.unmsm.edu.pe.* [En línea] [Citado el: 10 de 2 de 2011.] http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/basic/mendoza_nj/cap5.pdf.

36. **Rabelo Rodriguez, Ing.Alexander.** *CESIM_BQO_0120_Arquitectura de software v2.0.* 2010.

37. *Ingeniería Informática.* **Zulueta Véliz, Ing. Yeleny.** 14, 2007.

38. *LOS MAPAS CONCEPTUALES EN LA MEJORA DE PROCESOS EN EL DESARROLLO DE SOFTWARE.* **Estrada Sent, Vivian y Febles Estrada, Ailin.** 2010.

Glosario de Términos

Administración de Requisitos: Es la administración de todos los requisitos recibidos o generados por el proyecto.

Caso de Uso: Secuencias de acciones que el sistema puede llevar a cabo interactuando con sus actores.

Clases: Conjunto de objetos que comparten atributos, operaciones, relaciones y semántica, las mismas representan los conceptos fundamentales del sistema.

CMMI (Capability Maturity Model Integration): está entre los principales estándares de evaluación y mejora del proceso de desarrollo de software.

Calidad: Conjunto de propiedades y características de un cliente o servicio, que le confieren aptitud para satisfacer una serie de necesidades explícitas o implícitas.

Diagramas: Es la representación gráfica de un conjunto de elementos, visualizan un sistema desde diferentes perspectivas.

Enterprise Architect: Es una herramienta CASE (Computer Aided Software Engineering) para el diseño y construcción de sistemas de software.

GatherSpace: Es una herramienta entorno web para administración de requisitos de software. Anexos

GRASP: Patrones de software para la asignación general de responsabilidades.

OSRMT (Open Source Requirement Management Tool): Es una herramienta de gestión de requisitos.

RF (requisitos funcionales): Son capacidades o condiciones que el sistema debe cumplir.

RFN (requisitos no funcionales): Son propiedades o cualidades que el sistema debe tener.

Rational RequisitePro: Es una herramienta de administración de requerimientos.