

Universidad de las Ciencias Informáticas

Facultad 4



Título: Sistema para gestionar las lecciones aprendidas en proyectos productivos.

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

Autor(es): Leiny Evelin Alonso Lamí.
Yanetsy Aragón Pérez.

Tutor(es): Ing. Lizandra Arza Pérez

Ciudad de La Habana, Cuba

Junio 2008

Haz primero lo que es necesario, luego lo que es posible y de pronto te encontrarás haciendo lo imposible.

San Francisco de Asís.

Declaración de Autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas para que haga el uso que estime pertinente con este trabajo.

Para que así conste firmamos la presente a los ____ días de mes de _____ del año _____

Leiny Evelin Alonso Lamí

Yanetsy Aragón Pérez

Lizandra Arza Pérez

*A mi mamá, mi papá y mis abuelos
Por el inmenso amor que siempre
he recibido de ellos.*

Yanetsy

*A mi mamá, mi papá, mi tata y mis abuelas
Por todo el amor y cariño que me han dado
en cada minuto de mi vida.*

Leiny Evelin

Agradecimientos

- ♥ *A Mario por su ayuda incondicional en todo el proceso de la tesis.*

- ♥ *A todas las personas que de alguna forma nos ayudaron (Adriana, Liliannis, Zuzel, Vilmavis, Yiliexy, Danay, Cobo, Mandy, Alexander, Carlos Abel)*

- ♥ *A nuestro tribunal, en especial a Yenín y a Liván por las ideas que siempre nos dieron para mejorar la tesis.*

- ♥ *A Julio por guiarnos por el camino correcto cuando empezamos la tesis.*

- ♥ *A Lizandra, nuestra tutora, que a pesar de no contar con mucho tiempo nos ayudó y guió.*

Leiny Evelin

- ♥ *A mi mamá y a mi papá por ser ejemplo, ayudarme, aconsejarme, estar siempre a mi lado guiándome para que cada día sea mejor y por darme todo su amor. Porque su sueño de verme graduada, esta por hacerse realidad. Los amo.*
- ♥ *A mi tata por la alegría y cariño que siempre me brinda.*
- ♥ *A Dios por la vida, su bendición, por mi familia, mis amistades y sobre todo por darme la dicha de disfrutar de estos 5 años de universidad.*
- ♥ *A mi abuelita Librada por la confianza que siempre ha tenido en mí y por los deseos que me inspira de luchar por la vida. Te adoro.*
- ♥ *A mi abuelita Dulce, que esta en el cielo porque su sueño era verme graduada, gracias por todo.*
- ♥ *A mi abuelito Berto por ser tan especial, te quiero.*
- ♥ *A mi segundo papá, mi tío Onofre, por su cariño y apoyo.*
- ♥ *A Mary, Omar Yanelys, Pedrito, Amparo y Raúl por la acogida que me dieron en estos 5 años de Universidad.*
- ♥ *A mis amistades (Ivette, Mandy, Danay, Adriana, Alexander, Yily, Zuzi, David y Dayannis) por todos los momentos que vivimos, por sus consejos y comprensión. Porque sin ellos hubiese sido imposible llegar al final. Siempre van a estar en mi corazón. Los quiero mucho.*
- ♥ *A mi amiga Yanetsy que desde primer año ha estado a mi lado, por su comprensión, cariño y por todos los momentos lindos que pasamos juntas, gracias por dejarme ser tu amiga. Recuerda que siempre serás " My Best Friend of the University ".*
- ♥ *A mis amigos de siempre Grette, Isairys, Donybel, Ana lindy, Herlys, Shael, Annister, Yuriam, Rafelito y Edel por estar siempre a mi lado y por su preocupación porque todo me saliera bien. Gracias.*

Yanetsy

En la vida pasamos por momentos buenos y malos, en cada uno de ellos he tenido a mi alrededor personas maravillosas, personas a las cuales no tengo como agradecerles todo lo que han hecho por mí, hoy de alguna manera les hago saber que todo ese apoyo que recibí no fue en vano, gracias a ustedes soy la persona que soy...

- ♥ *A mami y papi: Gracias por darme la vida. A ustedes jamás podre agradecerles con palabras todo lo que han hecho por mí. Gracias por su inmenso amor, por la educación que me supieron dar, por sus consejos sabios y por estar presente en cada momento junto a mí. Uno no escoge los padres, pero si volviera a nacer, y me dieran esa oportunidad los escogería a ustedes de nuevo. Son lo más grande que tengo....Los adoro.....*
- ♥ *A Dios: Gracias por la vida, por estar presente en todo momento y por dejarme alcanzar todos mis sueños...*
- ♥ *A mis abuelos: Gracias por su cariño, apoyo y amor. Los llevare siempre en mi corazón...*
- ♥ *A mi tati: A ti mi vida, gracias por tu amor, apoyo y comprensión. Por todos los momentos maravillosos que pasamos juntos y gracias a la vida por ponerte en mi camino. Hiciste de mí la persona más feliz en estos dos últimos años. Te amo mucho y te llevo en mi corazón...*
- ♥ *A mis tias y tios: Gracias por quererme y considerarme como una hija más. Los quiero mucho...*
- ♥ *A mis primos: A ustedes gracias por tenerlos ahí, por confiar tanto en mí, por esa alegría y amor que siempre han sabido darme y por más que ser primos ser mis hermanos....*
- ♥ *A Kendry: A ti mi niñito lindo, gracias por llegar a este mundo a darme tanta felicidad, por querer tanto a mis padres y a mí, por decirme TATA y por darme toda la alegría del mundo...Te quiero mucho mi amor...*
- ♥ *A leiny: Mi mejor amiga de la universidad. Gracias por estar siempre a mi lado, en los momentos buenos y malos. Por ser desde el primer día que llegamos a esta universidad la mejor amiga y por ser la mejor compañera de tesis. Nunca te olvidare...*

- ♥ *A (Zuzel, Yily, Danay, Adriana, Ivette, y Jasel): A ustedes gracias por ser mis amigas, por estar a mi lado en los momentos de alegría y tristeza, por nuestras travesuras y por la felicidad en estos años de universidad. Las llevo en el corazón...*
- ♥ *A (Lili, Yabseny, Aniuska, Aliuska, Leiny, Nelson y Anders): Gracias por ser los mejores compañeros de apto, por ser tan buenos y por los momentos inolvidables que pasamos en el 7105. Nunca los olvidare...*
- ♥ *A Yandy: Gracias por ser el mejor amigo de la infancia, por nuestras travesuras y maldades y por ser el mejor vecino que tengo... Te quiero como un hermano...*
- ♥ *A todos mis amigos: Gracias por estos 5 años tan inolvidables y por ser todos como una gran familia.*
- ♥ *A toda mi familia: Gracias por ser la mejor familia del mundo...*

Resumen:

La Gestión del Conocimiento es un recurso importante para compartir, desarrollar y administrar las experiencias que posee una organización y los individuos que en ella trabajan, de manera que puedan ser empleadas para aumentar la productividad y el desarrollo económico de la misma. En la Universidad de las Ciencias Informáticas, el actualmente no se llevan a cabo procesos claros y establecidos para gestionar el conocimiento.

Existe la trasmisión intencionada de los conocimientos relacionados con los resultados técnicos, vinculados fundamentalmente a la experiencia en el uso de determinadas tecnologías, no siendo así de espontánea la trasmisión de los conocimientos y experiencias que se adquieren de la práctica diaria de cada una de las actividades. Es por ello que en el presente trabajo se lleva a cabo el análisis, diseño e implementación de un sistema que soporte la socialización del conocimiento de las personas que laboran en los proyectos productivos de la universidad, con el propósito de que la información almacenada pueda ser utilizada para la toma de decisiones y definición de nuevos proyectos.

Par realizar el diseño del sistema se utiliza la Metodología RUP, el lenguaje de modelado UML, que permite visualizar, especificar, construir y documentar los artefactos de un sistema y como herramienta de modelado el Visual Paradigm. En la fase de implementación, se utiliza como lenguaje de programación PHP y el Gestor de Base de Datos PostgreSQL.

Palabras Clave

Conocimiento, Gestión del Conocimiento, Experiencia, Gestión de Experiencias, Lecciones Aprendidas.

Introducción	1
CAPITULO 1: FUNDAMENTACION TEORICA	3
1.1 Introducción	3
1.2 Conceptos de Conocimiento.....	3
1.3 Clasificación del conocimiento	4
1.4 ¿Qué es la Gestión del Conocimiento?	6
1.5 Objetivos de la Gestión del Conocimiento	7
1.6 Proceso de Gestión del Conocimiento	8
1.7 Herramientas tecnológicas para la administración de conocimiento:	9
1.8 Tecnologías y Herramientas Utilizadas	11
1.8.1 ¿Por qué utilizar tecnologías Web?.....	11
1.8.2 Metodologías de desarrollo de software	12
1.8.3 El Lenguaje Unificado del Modelado (UML).....	15
1.8.4 Herramientas CASE	15
1.8.5 Herramienta de modelado	16
1.8.6 Lenguajes de Programación.	17
Lenguaje HTML.....	17
Lenguaje Java Script	17
Lenguaje PHP 5.0.....	18
1.8.7 Ajax	18
1.8.8 DOM	19
1.8.9 CSS.....	19
1.8.10 Zend Studio para Eclipse	20
1.8.11 Servidores de Aplicación	20
1.8.12 Sistemas Gestores de Bases de Datos (SGBD).....	21
PostgreSQL.....	21
1.9 Ext JS	22
1.10 Arquitectura Cliente /Servidor	23
1.11 Framework	24
1.11.1 Symfony.....	25
1.12 Patrón Modelo Vista Controlador (MVC)	28
1.13 Conclusiones del Capítulo.....	28
CAPITULO 2: CARACTERISTICAS DEL SISTEMA.....	30
2.1 Introducción	30
2.2 Descripción del proceso a implementar	30
2.3 Propuesta del sistema.....	31

2.4 Definición de requerimientos funcionales	32
2.4.1 Paquete Cliente	32
2.4.2 Paquete Administración	33
2.5 Definición de requerimientos no funcionales	34
2.5.1 Requerimientos de Apariencia o interfaz externa	34
2.5.2 Requerimientos de Usabilidad	35
2.5.3 Requerimientos de Rendimiento	35
2.5.4 Requerimientos de Seguridad	35
2.5.5 Requerimientos de Software	36
2.5.6 Requerimientos de Hardware	36
2.5.6 Requerimientos de Portabilidad	36
2.5.7 Restricciones en el diseño y la implementación	36
2.6 Definición general de los casos de usos.....	37
2.6.1 Identificación de los actores	37
2.6.2 Casos de usos del Paquete Cliente.....	37
2.6.3 Casos de usos del Paquete Administración.....	39
2.7 Diagramas de Casos de Usos y paquetes	40
2.8 Descripción Textual de los Casos de Uso.....	42
2.9 Conclusiones	42
Capítulo 3: Análisis y diseño del sistema.....	43
3.1 Introducción	43
3.2 Análisis.	43
3.2.1 Clases del análisis.	43
3.3 Diseño	43
3.3.1 Diagramas de interacción.....	44
3.4 Diseño de la base datos	44
Modelo Entidad Relación	44
3.5 Tratamiento de Errores	45
3.6 Concepción de la Ayuda.	46
3.7 Seguridad	46
3.8 Principios de diseño del sitio	47
3.9 Conclusiones	47
Capítulo 4: Implementación y prueba	48
4.1 Introducción	48
4.2 Implementación.....	48
4.3 Diagramas de Componentes	48

4.4 Diagrama de despliegue.....	51
Figura 4.7 Diagrama de Despliegue	51
4.5 Pruebas	52
4.5.1 Métodos de Pruebas	52
4.5.2 Prueba de Sistema	52
4.6 Conclusiones	52
CONCLUSIONES	53
RECOMENDACIONES.....	54
REferencias bibliográficas.....	55
ANEXOS.....	57
GLOSARIO DE TERMINOS.....	117

INTRODUCCIÓN

En las últimas décadas en el mundo se ha presentado un salto significativo en el desarrollo de las industrias que hacen un uso intensivo del capital intelectual, cuyo negocio fundamental se basa en la creación y distribución del conocimiento. Este auge ha sido centro de atención de empresarios e investigadores que han conceptualizado y puesto en práctica los elementos vinculados al conocimiento. Entre ellos ha surgido la Gestión del Conocimiento, que se identifica como un nuevo enfoque empresarial capaz de generar importantes ventajas competitivas.

La Gestión del Conocimiento es un conjunto de prácticas que incluye la identificación de recursos intangibles o intelectuales dentro de las organizaciones, captura, almacenamiento y generación de nuevos conocimientos. Tiene como base el intercambio, por lo que necesita de una gestión correcta de los recursos humanos y se apoya en las tecnologías de la informática y las comunicaciones para su desarrollo. En esencia se trata de definir, capturar, almacenar, analizar, difundir y mantener el conocimiento de una organización.

La Universidad de las Ciencias Informáticas dados los procesos docentes-educativos, las investigaciones, la producción, el uso intensivo de la tecnología para el aprendizaje y otras características, necesita que se realice una eficiente gestión de los conocimientos. Sin embargo, aunque se pueden identificar algunos métodos de trabajo que tributan a que se gestione el conocimiento, resultan insuficientes, teniendo en cuenta las actividades que se realizan y las amplias posibilidades de intercambio que facilita un centro como este.

En la actividad productiva que se realiza en la universidad es en una de las áreas se evidencia con mayor fuerza la carencia de definiciones establecidas para la gestión del conocimiento. En los mismos se genera una gran cantidad de conocimientos a partir de la experiencia práctica de las personas, dicho conocimiento ni se socializa ni se gestiona de manera que la organización, es decir las facultades y demás estructuras de producción aprendan de estas experiencias. A este conocimiento se le puede llamar lecciones aprendidas, estas lecciones a su vez pueden ser positivas o negativas para la organización.

Al finalizar el proyecto o en cierres de determinadas etapas se debe hacer el análisis de estas lecciones, registrarlas y que de esta manera sirvan para utilizarlas como experiencias, para la gestión de riesgos y la utilización de soluciones en futuros proyectos. Como en la actualidad esto no se pone en práctica en los proyectos, este trabajo estará enmarcado en como darle solución a este problema en los proyectos productivos. Teniendo en cuenta toda esta situación, se ha determinado darle solución al siguiente

problema científico: ¿Cómo gestionar las lecciones aprendidas de los proyectos productivos de la Universidad de las Ciencias Informáticas para lograr la socialización de las experiencias? Se plantea como **objeto de estudio** la Gestión del Conocimiento, haciendo necesario establecer como **campo de acción** la gestión del conocimiento a partir de las lecciones aprendidas en los proyectos productivos de la Universidad.

Se propone como **objetivo general:**

Desarrollar un sistema informático que permita gestionar las lecciones aprendidas en los proyectos productivos de la Universidad.

Para un mejor entendimiento del objetivo general se determinaron las siguientes **tareas específicas:**

- Definir las bases teóricas del desarrollo que se realiza.
- Definir las metodologías, tecnologías y herramientas para el desarrollo de la aplicación.
- Modelar el sistema en cuestión utilizando una metodología de desarrollo.
- Realizar el análisis y el diseño del sistema.
- Generar el código necesario para la implementación del sistema.
- Realizar las pruebas de sistema a la aplicación desarrollada.
- Realizar la documentación del software.

El presente documento se estructura en resumen, introducción y cuatro capítulos de contenidos, conclusiones, recomendaciones, referencias bibliográficas, anexos y glosario de términos. En el capítulo 1 se desarrolla la fundamentación teórica, donde están presentes conceptos características, herramientas más utilizadas para Gestión de Conocimiento. Además de un análisis de las diferentes metodologías, artefactos y herramientas utilizadas para desarrollar el sistema.

En el Capítulo 2 se tratan los aspectos relacionados con el proceso de registro y uso de las lecciones aprendidas, los requisitos tanto funcionales, como no funcionales, los diagramas de casos del sistema y la descripción textual de los casos de usos.

En el Capítulo 3 se lleva a cabo el análisis y diseño del sistema, utilizando la metodología RUP y el lenguaje de modelación UML y como herramienta de modelado el Visual Paradigm.

En el capítulo 4 se aborda lo referente a los flujos de trabajo, implementación y prueba de la primera versión del sistema.

CAPITULO 1: FUNDAMENTACION TEORICA

1.1 Introducción

La Gestión del Conocimiento es capaz de captar, almacenar y difundir el conocimiento de una organización, es por ello que para el logro de los objetivos propuestos en el presente trabajo se hace necesario realizar estudios sobre el conocimiento y las tendencias actuales de la gestión del conocimiento. Por lo que se hace necesario entender bien sus conceptos, objetivos y características fundamentales.

Es en este capítulo donde se describen las bases teóricas de la investigación y se establecen las definiciones significativas para el trabajo a desarrollar. Además se expone un análisis de las tecnologías, artefactos y software seleccionados para desarrollar el sistema.

1.2 Conceptos de Conocimiento

En estos tiempos se exige ser más creativos en las organizaciones para lograr el desarrollo de las mismas. De aquí que el conocimiento en la producción de bienes se ha convertido en un recurso fundamental para las empresas. Por lo que ha surgido la necesidad de conceptualizarlo, de esta forma lograr un mejor entendimiento del tema.

Para Muñoz Seca el **conocimiento es la capacidad de resolver un determinado conjunto de problemas con una efectividad determinada**. Es un conjunto integrado por información, reglas, interpretaciones y conexiones puestas dentro de un contexto y de una experiencia, que ha sucedido dentro de una organización, bien de una forma general o personal. El conocimiento sólo puede residir dentro de un conocedor, una persona determinada que lo interioriza racional o irracionalmente. (Muñoz, 1997)

Agustí Canals menciona que **el conocimiento es un recurso que no sólo permite interpretar nuestro entorno, sino que da la posibilidad de actuar. Es un recurso que se halla en las personas y en los objetos físicos o no que estas personas utilizan**. Agrega que también se encuentra en las organizaciones a las que pertenecen estas personas, en los procesos y en los contextos de dichas organizaciones. (Canals, 2003)

El conocimiento es identificar, estructurar y sobre todo utilizar la información para obtener un resultado. Requiere aplicar la intuición y la sabiduría, propios de la persona a la información. La capacidad de interpretar esos datos es lo que provoca que la información se convierta en conocimiento. (Figuroa 2006)

En el presente trabajo se estudiaron los conceptos establecidos por diferentes autores y de todos ellos se tomaron los aspectos que para esta investigación son relevantes, por lo que se definirá conocimiento como: El activo más valioso de cualquier organización en la Sociedad de la Información, no es más que el conjunto organizado de datos e información destinado a resolver un determinado problema.

El conocimiento tiene diferentes orígenes y características, es por ello que existen estudios realizados por varios autores que definen clasificaciones del conocimiento de acuerdo a determinados aspectos.

1.3 Clasificación del conocimiento

Según Nonaka y Takeuchi existen **dos tipos de conocimiento**: el conocimiento tácito y el conocimiento explícito.

El conocimiento tácito: Es aquel que una persona, comunidad, organización o país, tiene incorporado o almacenado en su mente, en su cultura y es difícil de explicar.

Para un mejor entendimiento el conocimiento tácito o implícito tiene un carácter muy personal, puede estar compuesto por: Ideas, experiencias, destrezas, habilidades, costumbres, valores, historia, creencias.

El conocimiento explícito: Puede ser expresado con palabras, números, fórmulas (Nonaka y Takeuchi 1995)

Mientras que Israel Núñez en su estudio *Propuesta de clasificación de las herramientas de software para la Gestión del Conocimiento* clasifica el conocimiento en generativo, productivo, representativo y organizacional y describen su carácter individual y social. Describe:

El conocimiento generativo como el proceso y resultado de la creación del nuevo conocimiento durante la solución de problemas o la identificación de nuevas propuestas o alternativas para nuevas oportunidades.

El conocimiento productivo como el proceso y resultado de aplicar el conocimiento generativo en los procesos productivos, es un tipo de conocimiento, que se concreta en los procesos, en los procedimientos, en la tecnología y en los productos o resultados y, por ello, es de carácter explícito y con valor de uso.

El conocimiento representativo como el proceso y resultado de transferir conocimiento explícito a los interlocutores del entorno de la organización: clientes, proveedores, competidores, reguladores.

El conocimiento organizacional como sedimento en la organización. Se presenta en sus ideas rectoras - misión, visión, objetivos estratégicos, reglamentos, políticas, así como en la estructura funcional, flujos de información y de trabajo, nivel de desarrollo y uso de la tecnología, resulta de los procesos de

transformación dialéctica entre los diferentes tipos de conocimiento tácito, explícito, generativo, productivo y representativo, a partir de la comunicación corporativa, y que debe reflejarse en la memoria corporativa. Por su forma, es conocimiento explícito, consciente y perceptible. (Núñez, 2005)

Estos autores explican que cada persona le da un significado propio a lo que aprende, por que el conocimiento es resultado de las experiencias de cada cual, de su inteligencia y sus capacidades para el aprendizaje. Pero el individuo también tiene presente el significado que determinada información alcanza para los demás. Esta condición permite el entendimiento, la valoración de diferentes puntos de vista y el aprendizaje colectivo.

En esta investigación se quiere lograr una herramienta que permita gestionar las experiencias que adquieren cada una de las personas vinculadas a los proyectos productivos de la universidad. Dichas experiencias son las ideas, destrezas y habilidades que surgen como resultado del trabajo diario, puestas en prácticas para darle solución a un problema determinado. Al revisar las diferentes clasificaciones estudiadas lo que se pretende está en línea con las definiciones de conocimiento tácito y generativo, por lo que para la presente investigación trabajaremos con el concepto Lección aprendida, que unifique las características de las dos definiciones vistas.

En los últimos años el conocimiento ha ocupado un importante lugar en las empresas. Las industrias más desarrolladas son: Software, Biotecnología, Microelectrónica, Computación, Telecomunicaciones, Farmacéutica, Aeroespacial. Industrias donde precisamente el conocimiento es el recurso fundamental. Sobre las características fundamentales del conocimiento Andreu y Sieber plantean que existen básicamente tres:

- El conocimiento es personal, en el sentido de que se origina y reside en las personas, que lo asimilan como resultado de su propia experiencia (es decir, de su propio hacer, ya sea físico o intelectual) y lo incorporan a su acervo personal estando convencidas de su significado e implicaciones, articulándolo como un todo organizado que da estructura y significado a sus distintas piezas.
- Su utilización, que puede repetirse sin que el conocimiento se consuma como ocurre con otros bienes físicos, permite entender los fenómenos que las personas perciben (cada una a su manera, de acuerdo precisamente con lo que su conocimiento implica en un momento determinado), y también evaluarlos, en el sentido de juzgar la bondad o conveniencia de los mismos para cada una en cada momento.

- Sirve de guía para la acción de las personas, en el sentido de decidir qué hacer en cada momento porque esa acción tiene en general por objetivo mejorar las consecuencias, para cada individuo, de los fenómenos percibidos (incluso cambiándolos si es posible). (Andreu y Sieber 2000)

Una vez identificado como el conocimiento ha pasado a ser uno de los factores fundamentales de las empresas el punto es: si las personas son las que poseen el conocimiento y este marca considerables diferencias competitivas, cómo hacer que tan importante recurso mantenga su valor estratégico y más aún cómo la empresa puede mantenerlo. Ante estas interrogantes surge el término Gestión del Conocimiento, que hoy ocupa el centro de la atención de los directivos y algunos investigadores.

1.4 ¿Qué es la Gestión del Conocimiento?

Pérez Rodríguez y Coutin Domínguez la definen como un proceso mediante el cual se desarrolla, estructura y mantiene la información, con el objetivo de transformarla en un activo crítico y ponerla a disposición de una comunidad de usuarios, definida con la seguridad necesaria. Incluye el aprendizaje, la información, las aptitudes y la experiencia desarrollada durante la historia de la organización. (Pérez, 2005)

Según Balasubramanian es la capacidad organizativa que permite a las personas de las organizaciones, que trabajan como individuos (trabajadores del conocimiento), o en grupos, proyectos u otras comunidades de interés, a crear, capturar, compartir e influir su conocimiento colectivo para mejorar el desempeño. (Balasubramanian, 1999)

Para Agustí-Cullel es la gestión del conocimiento organizativo para crear valor de negocio y generar una ventaja competitiva. La esencia de la Gestión del Conocimiento es cómo las organizaciones generan y adquieren; almacenan y guardan; acceden a y utilizan; distribuyen y diseminan su conocimiento. (Cullel, 2003)

Gunnlaugsdottir menciona este fenómeno como el modo de descubrir, recoger, documentar y organizar una base de conocimiento que los empleados de la organización puedan recuperar más tarde, distribuir y utilizar en su trabajo diario individual y en sus colaboraciones con sus compañeros. (Gunnlaugsdottir, 2003)

Carrión lo define como el conjunto de procesos y sistemas que permiten que el Capital Intelectual de una organización aumente de forma significativa, mediante la gestión de sus capacidades de resolución de problemas de forma eficiente (en el menor espacio de tiempo posible), con el objetivo final de generar ventajas competitivas sostenibles en el tiempo.

Gestión de conocimiento es organizar recursos intangibles y capacidades organizativas para obtener objetivos empresariales. La gestión del conocimiento se basa fundamentalmente en compartir conocimiento y permitir el uso de dicho conocimiento. Tomado de (www.knowledgeboard.com)

La Gestión del Conocimiento se encarga de resolver cómo utilizar mejor uno de los valores más importantes de las instituciones, organizaciones o empresas, los recursos humanos con sus conocimientos, apoyándose para ellos en las Tecnologías de la Información y las Comunicaciones (TIC).

La GC exige utilizar eficientemente los recursos tecnológicos en la comunicación dentro de la organización. Una buena comunicación permitirá la creación, transformación y transmisión del conocimiento. Por otro lado gestionar bien los recursos humanos es primordial para la GC, porque en definitiva es el recurso humano el que posee y crea conocimiento.

Por esos sus pilares fundamentales son precisamente: los recursos humanos (RRHH), las TIC y el conocimiento. (Ver figura 2)

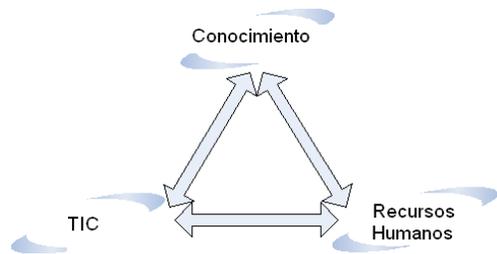


Figura 2: Relación entre los pilares de la GC.

La Gestión del Conocimiento a decir de Agustí Canals consiste en optimizar la utilización de este recurso mediante la creación de las condiciones necesarias para que los flujos de conocimiento circulen mejor. Lo que gestionamos en realidad, pues, no es el conocimiento en sí mismo, sino las condiciones, el entorno y todo lo que hace posible y fomenta dos procesos fundamentales: la creación y la transmisión de conocimiento. (Canals, 2003)

Para esta investigación se tomará como definición de Gestión del Conocimiento, este último definido por Agustí Canals, por ser considerado por las autoras como el más completo.

1.5 Objetivos de la Gestión del Conocimiento

Díaz Muñante A ha resumido perfectamente los objetivos de la Gestión de Conocimiento en:

- Formular una estrategia de alcance organizacional para el desarrollo, adquisición y aplicación del conocimiento.
- Implantar estrategias orientadas al conocimiento.
- Promover la mejora continua de los procesos de negocio, enfatizando la generación y utilización del conocimiento.
- Monitorear y evaluar los logros obtenidos mediante la aplicación del conocimiento.
- Reducir los tiempos de ciclos en el desarrollo de nuevos productos, mejoras de los ya existentes y la reducción del desarrollo de soluciones a los problemas.
- Reducir los costos asociados a la repetición de errores.

1.6 Proceso de Gestión del Conocimiento

El proceso de Gestión del conocimiento debe entenderse como los subprocesos necesarios para el desarrollo de soluciones orientadas a generar las bases del conocimiento de valor para la organización. Representa la cadena de agregación de valor a cada una de las instancias de conocimiento existentes en la organización. Cabe destacar que el proceso de Gestión del conocimiento se centra en el concepto de generación de valor asociado al negocio, el cual ayudará a descartar las instancias de conocimiento que sean no-relevantes. Los subprocesos que de manera general se definen se muestran en la Figura 1:

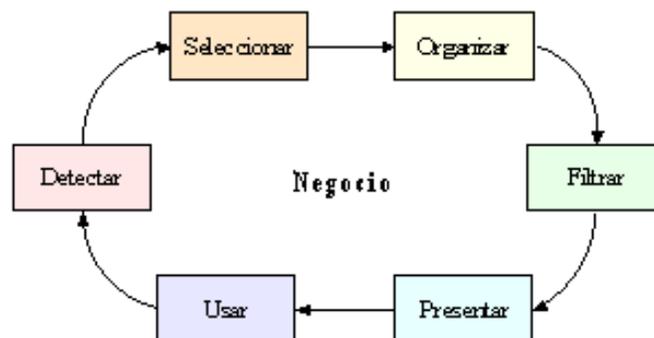


Figura 1 - El proceso de Gestión del conocimiento

La Gestión del conocimiento puede ser descrita como el proceso sistemático de detectar, seleccionar, organizar, filtrar, presentar y usar la información por parte de los participantes de la organización, con el objetivo de explotar cooperativamente los recursos de conocimiento basados en el capital intelectual

propio de las organizaciones, orientados a potenciar las competencias organizacionales y la generación de valor, donde:

- * **Detectar:** Es el proceso de localizar modelos cognitivos y activos de valor para la organización, el cual radica en las personas. Son ellas, de acuerdo a sus capacidades cognitivas (modelos mentales, visión sistémica, etc.), quienes determinan las nuevas fuentes de conocimiento de acción.
- * **Seleccionar:** Es el proceso de evaluación y elección del modelo en torno a un criterio de interés. Los criterios pueden estar basados en criterios organizacionales, comunales o individuales, los cuales estarán divididos en tres grandes grupos: Interés, Práctica y Acción.
- * **Organizar:** Es el proceso de almacenar de forma estructurada la representación explícita del modelo.
- * **Filtrar:** Una vez organizada la fuente, puede ser accedida a través de consultas automatizadas en torno a motores de búsquedas. Las búsquedas se basarán en estructuras de acceso simples y complejas, tales como mapas de conocimientos, portales de conocimiento o agentes inteligentes.
- * **Presentar:** Los resultados obtenidos del proceso de filtrado deben ser presentados a personas o máquinas. En caso que sean personas, las interfaces deben estar diseñadas para abarcar el amplio rango de comprensión humana. En el caso que la comunicación se desarrolle entre máquinas, las interfaces deben cumplir todas las condiciones propias de un protocolo o interfaz de comunicación.
- * **Usar:** El uso del conocimiento reside en el acto de aplicarlo al problema objeto de resolver. De acuerdo con esta acción es que es posible evaluar la utilidad de la fuente de conocimiento a través de una actividad de retroalimentación.

En la presente investigación se pretende automatizar una parte de este proceso de gestión del conocimiento, fundamentalmente la relacionada con los subprocesos de organizar, filtrar y presentar.

1.7 Herramientas tecnológicas para la administración de conocimiento:

Las herramientas tecnológicas para administración de conocimiento, como cualquier otra herramienta, están diseñadas para facilitar el trabajo y permitir que los recursos sean aplicados eficientemente intercambiando información y conocimiento dentro y fuera de las organizaciones.

Las herramientas para administración de conocimiento permiten a las organizaciones generar, acceder, almacenar y transferir el conocimiento existente en la empresa.

Muchas de las herramientas identificadas como herramientas de administración de conocimiento son muy sencillas, una simple base de datos que administre lecciones aprendidas, mejores prácticas o competencias que cumplan estrictamente con la definición de una herramienta de administración de conocimiento.

Herramientas tecnológicas de soporte a la administración del conocimiento:

Infraestructura

Es la base sobre la cual las soluciones o aplicaciones de administración de conocimiento son construidas, es decir, todos aquellos repositorios de información estructurada (bases de datos) o no estructurada (correos, documentos); el grupo de almacenamiento que provee el soporte necesario a las actividades de colaboración para compartir conocimientos, así como los servicios de redes y telecomunicaciones que permiten hacer el enlace para los grupos de trabajo

Repositorios de almacenamiento

En esta categoría de tecnologías se ubican a todos los medios de almacenamiento de información comúnmente encontrados con el nombre de bases de datos, bases de conocimiento e, incluso, repositorios de almacenamiento. Estos elementos sirven como bodegas de almacenamiento de información. El almacenamiento no es por sí mismo lo principal de estos repositorios sino su explotación.

Plataformas de aplicaciones

Las plataformas de aplicación son un elemento clave y muchas veces determinante en la selección de una herramienta de administración de conocimiento ya que éstas son la base sobre las que corren las aplicaciones. Se habla básicamente de Lotus Notes y Microsoft como los dos grandes rivales.

Aplicaciones

Se construyen sobre la infraestructura para proveer acceso a las bases de conocimiento. Muchas veces utilizan las tecnologías de red para lograr la colaboración a través de portales de información empresarial, avanzados sistemas de búsqueda y consultas basadas en Web. Estas aplicaciones para los usuarios son sólo la interface con la cual ellos interactúan para lograr acceso al conocimiento.

Algunos de los beneficios del uso de tecnologías en la administración de conocimiento que han sido identificados son los siguientes:

a) Aseguramiento de la memoria organizacional

Gracias a que la mayoría de estas herramientas utilizan formas de almacenamiento como bases de datos o simplemente a través de archivos planos como lo hacen algunos sistemas de mensajería síncrona y los manejadores electrónicos, se va conformando una creciente memoria organizacional corporativa, ya sea de manera digital, visual o auditiva. Así, la experiencia ganada por la empresa a través de los años puede ser explotada.

b) Aseguramiento del flujo de conocimiento

Uno de los grandes beneficios de la relación entre las tecnologías de información y de comunicaciones es precisamente la posibilidad de poder transferir información a través de medios electrónicos; con esto, se brinda un medio a través del cual no sólo se registre el conocimiento sino además se permita transferir ese conocimiento a todas las entidades que lo necesiten.

c) Facilitación del trabajo colaborativo

Todos conocen las ventajas que trae consigo el trabajo colaborativo; para la empresa es primordial proporcionar un contexto en que los individuos puedan interactuar unos con otros. Las herramientas tecnológicas tales como portales, el correo y chats corporativos pueden suministrar este medio y además integrar estas formas de colaboración a los procesos de negocio.

1.8 Tecnologías y Herramientas Utilizadas

Una vez estudiado y analizado los aspectos esenciales de la Gestión del Conocimiento se tiene un dominio de los términos que se manejarán en esta investigación para el desarrollo del sistema a implementar, por lo que se hace necesario un estudio de las tecnologías y herramientas más utilizadas en la actualidad para lograr un sistema que cumpla con todos los objetivos propuestos.

1.8.1 ¿Por qué utilizar tecnologías Web?

Las ventajas que proporcionan las aplicaciones Web se concentran en su mantenimiento, puesto que el sostenimiento del sistema se concentra en el servidor. En general, es el proveedor del servicio quien se preocupa de tener la aplicación siempre disponible y actualizada. Se podrían encontrar numerosas ventajas más, pero probablemente, la propiedad más destacada sea la conectividad que proporciona las aplicaciones Web, permitiendo el acceso a la aplicación desde cualquier punto. Esto permite ahorrar invertir en costosas infraestructuras de comunicaciones que en muchos casos podrían ser sencillamente imposibles.

Las tecnologías Web sirven para acceder a los recursos de conocimiento disponibles en Internet o en las intranets utilizando un navegador. Están muy extendidas por muchas razones: facilitan el desarrollo de sistemas de Gestión del Conocimiento, su flexibilidad en términos de escalabilidad, es decir, a la hora de expandir el sistema; su sencillez de uso y que imitan la forma de relacionarse de las personas, al poner a disposición de todos el conocimiento de los demás, por encima de jerarquías, barreras formales u otras cuestiones. Estas tecnologías pueden llegar a proporcionar recursos estratégicos, pero, evidentemente, no por la tecnología en sí misma, que está disponible ampliamente, sino por lo fácil que es personalizarla y construir con ella sistemas de Gestión de Conocimiento.

1.8.2 Metodologías de desarrollo de software

En la actualidad existen **dos tipos de metodologías de desarrollo del software**, las metodologías ágiles y las metodologías tradicionales.

Metodología ágiles: Son adaptativas más que predictivas. Ejemplo de ella son:

Programación Extrema (XP): La programación extrema concede una gran importancia a las pruebas del software. Aunque la mayoría de los procesos las tienen en cuenta, generalmente lo contemplan de una forma demasiado ligera y superficial. Sin embargo, XP lo toma como base para el desarrollo y cada programador que escribe código también escribe los casos de prueba. Estos forman parte del proceso continuo de generación de código y se integra continuamente con ello, lo que garantiza una plataforma estable para el futuro desarrollo. Sobre dicha plataforma se genera un proceso de diseño evolutivo, que es la base del sistema y que se enriquece con cada iteración. Nunca se generan diseños futuros. Además en esta metodología el cliente es parte del equipo de desarrollo, no existen prácticamente los analistas sino que los clientes escriben sus requisitos, además está concebida para equipos donde las personas sean capaces de cumplir varios roles.

Código Abierto (OS): Apuesta por la distribución de trabajo entre diferentes equipos, al igual que ocurre con la mayoría de los procesos adaptativos. La mayoría de los proyectos de código abierto cuentan con supervisores de código. Estos supervisores de código, son las únicas personas autorizadas para realizar un cambio en el repositorio del código fuente. Por otra parte, el resto del personal puede realizar cualquier cambio en el código base. Sin embargo, el supervisor del código es la persona responsable de coordinar y de mantener la consistencia del diseño del software. Una de las principales ventajas de los desarrollos de código abierto es que la depuración es altamente paralelizable, aunque un gran número de personas puedan verse involucradas. Cuando se soluciona un error, se envía la solución al supervisor de código, lo

que garantiza que alguien realiza la modificación de forma fiable mientras otra parte del personal se dedica a las tareas de depuración.

Metodologías Tradicionales: Potencian la planificación detallada de prácticamente todo el desarrollo de software a largo plazo. Ejemplo de ella es Proceso Unificado de Desarrollo (RUP), dicha metodología es la que será utilizada en este trabajo, debido a las características y ventajas que a continuación se mencionan.

La metodología RUP divide en 4 fases el desarrollo del software:

- **Inicio:** El Objetivo en esta etapa es determinar la visión del proyecto.
- **Elaboración:** En esta etapa el objetivo es determinar la arquitectura óptima.
- **Construcción:** En esta etapa el objetivo es llevar a obtener la capacidad operacional inicial.
- **Transición:** El objetivo es llegar a obtener una versión del proyecto.

Cada una de estas etapas es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes.

El ciclo de vida que se desarrolla por cada iteración, es llevada bajo dos disciplinas:

Disciplina de Desarrollo

- Ingeniería de Negocios: Entendiendo las necesidades del negocio.
- Requerimientos: Trasladando las necesidades del negocio a un sistema automatizado.
- Análisis y Diseño: Trasladando los requerimientos dentro de la arquitectura de software.
- Implementación: Creando software que se ajuste a la arquitectura y que tenga el comportamiento deseado.
- Pruebas: Asegurándose que el comportamiento requerido es el correcto y que todo lo solicitado esta presente.

Disciplina de Soporte

- Configuración y administración del cambio: Guardando todas las versiones del proyecto.
- Administrando el proyecto: Administrando horarios y recursos.
- Ambiente: Administrando el ambiente de desarrollo.

Principales características de RUP:

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo)
- Pretende implementar las mejores prácticas en Ingeniería de Software

- Desarrollo iterativo
- Administración de requisitos
- Uso de arquitectura basada en componentes
- Control de cambios
- Modelado visual del software
- Verificación de la calidad del software

El ciclo de vida de RUP se caracteriza por:

Dirigido por casos de usos: Los casos de usos reflejan lo que los usuarios necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de usos guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajos, representan la realización de los casos de usos.

Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo del proyecto y los usuarios deben de estar de acuerdo, por lo que describe los elementos del modelo que son mas importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlos , desarrollarlos y producirlos económicamente. RUP se desarrolla mediante iteraciones , comenzando por los casos de usos relevante desde el punto de vista de la arquitectura.

Iterativo e Incremental: RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividad de todos los flujos de trabajo, aunque desarrolla fundamentalmente unos mas que otros. Las iteraciones hacen referencias a los pasos en los flujos de trabajos, y los incrementos, al crecimiento del producto.

RUP es un proceso de desarrollo de software 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. Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso). En resumen una de las grandes ventajas de la metodología para

nosotros son los casos de uso y casos de prueba, los primeros te facilitan las tareas de programación y los últimos te garantizan un plan de pruebas bastante robusto aunque a veces tedioso.

1.8.3 El Lenguaje Unificado del Modelado (UML).

Desde los inicios de la informática se han estado utilizando distintas formas de representar los diseños de una manera más bien personal o con algún modelo gráfico, La falta de estandarización en la representación gráfica de un modelo impedía que los diseños gráficos realizados se pudieran compartir fácilmente entre distintos diseñadores, con este objetivo se creó el Lenguaje Unificado de Modelado (UML), es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad.

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

UML sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas software como para la arquitectura hardware donde se ejecuten.

UML es además un método formal de modelado. Esto aporta las siguientes ventajas:

- Mayor rigor en la especificación.
- Permite realizar una verificación y validación del modelo realizado.
- Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener la visión en el diseño, de más alto nivel, de la estructura de un proyecto.

De forma general las principales características de UML son:

1. Lenguaje unificado para la modelación de sistemas
2. Tecnología orientada a objetos
3. El cliente participa en todas las etapas del proyecto
4. Corrección de errores viables en todas las etapas
5. Aplicable para tratar asuntos de escala inherentes a sistemas complejos de misión crítica, tiempo real y cliente/servidor.

1.8.4 Herramientas CASE

Las Herramientas de Ingeniería de Software Asistida por Ordenador (CASE) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Se puede definir a una herramienta CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software.

Ventajas con la utilización de las herramientas CASE:

- Permiten el incremento en la velocidad de desarrollo de los sistemas.
- Permiten a los analistas tener más tiempo para el análisis y diseño y minimizar el tiempo para codificar y probar.
- En las etapas del proceso de desarrollo de software permiten:
 - Automatizar el dibujo de diagramas.
 - Ayudar en la documentación del sistema.
 - Ayudar en la creación de relaciones en la base de datos.
 - Generar estructuras de código.
- Aumentan la productividad. Esto se consigue a través de la automatización de determinadas tareas, como la generación de código y la reutilización de objetos o módulos.

1.8.5 Herramienta de modelado

El Visual Paradigm es una poderosa herramienta CASE de modelación visual. Utiliza UML para el modelado, permitiendo crear tipos diferentes de diagramas en un ambiente totalmente visual. Es muy sencillo de usar, fácil de instalar y actualizar. Genera código para varios lenguajes. Es una herramienta que puede ser utilizada en la creación de software libre, es nombrada por muchas bibliografías como la herramienta CASE por excelencia del software libre, sin dejara de mencionar que el Visual Paradigm, posee además:

- ↻ Un entorno de creación de diagramas para UML.
- ↻ Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- ↻ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ↻ Capacidades de ingeniería directa e inversa.
- ↻ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- ↻ Disponibilidad de múltiples versiones, para cada necesidad.
- ↻ Disponibilidad de integrarse en los principales IDEs.

↳ Disponibilidad en múltiples plataformas.

Por otra parte, posibilita la representación gráfica de los diagramas permitiendo ver el sistema desde diferentes perspectivas, como el de componentes, despliegue, secuencia, casos de uso, clase, actividad, entre otros. Además, se centra en cómo los componentes del sistema interactúan entre ellos, sin entrar en detalles excesivos, también, permite ver las relaciones entre los componentes del diseño y mejora la comunicación entre los miembros del equipo usando un lenguaje gráfico.

En resumen el Visual Paradigm es una herramienta multiplataforma dotada de una buena cantidad de productos o módulos para facilitar el trabajo durante la confección del software, reduciendo el tiempo de desarrollo del mismo y garantizando la calidad del producto final. Ayuda a la comunicación entre los miembros de equipo, a monitorear el tiempo de desarrollo y a entender el entorno de los sistemas.

1.8.6 Lenguajes de Programación.

Actualmente existen diferentes lenguajes de programación para desarrollar en la web, estos han ido surgiendo debido a las tendencias y necesidades de las plataformas. A continuación daremos una introducción a los diferentes lenguajes de programación para la web que se utilizará en el desarrollo del sistema.

Lenguaje HTML

HTML, no es un lenguaje de programación, es un lenguaje de especificación de contenidos para un tipo específico de documentos. Es decir, mediante HTML podemos especificar, usando un conjunto de etiquetas, como va a representarse la información en un navegador. Se centra en la representación en la pantalla de la información. [Rodríguez]

Son sistemas complejos de descripción de información, normalmente documentos, que se pueden controlar desde cualquier editor de código binario. [Rodríguez]

Se puede decir que existen tres utilidades básicas de los lenguajes de marcas: los que sirven principalmente para describir su contenido, los que sirven más que nada para definir su formato y los que realizan las dos funciones indistintamente. Las aplicaciones de bases de datos son buenas referencias del primer sistema, los programas de tratamientos de textos, son ejemplos típicos del segundo tipo, y el HTML es la muestra más conocida del tercer modelo. [Huidobor]

Lenguaje Java Script

Este es un lenguaje interpretado, no requiere compilación. Utilizado principalmente en páginas web. Es similar a Java, aunque no es un lenguaje orientado a objetos, el mismo no dispone de herencias. La

mayoría de los navegadores en sus últimas versiones interpretan código JavaScript. Tienen como ventajas Lenguaje de scripting seguro y fiable, los script tienen capacidades limitadas, por razones de seguridad, el código JavaScript se ejecuta en el cliente. JavaScript es un lenguaje orientado a eventos. Cuando un usuario pincha sobre un enlace o mueve el puntero sobre una imagen se produce un evento. Mediante JavaScript se pueden desarrollar Script que ejecuten acciones en respuesta a estos eventos.

Lenguaje PHP 5.0

Es un lenguaje de programación utilizado para la creación de sitio web. Surgió en 1995, es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, embebidas en páginas HTML y ejecutadas en el servidor.

Es un lenguaje de programación orientado a objetos. Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones. Es libre, por lo que se presenta como una alternativa de fácil acceso para todos. Es un lenguaje multiplataforma, esto hace que cualquier sistema puede ser compatible con el lenguaje, por ejemplo: Linux, Windows, entre otros. Incluye gran cantidad de funciones. Capacidad de conexión con la mayoría de los manejadores de base de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, entre otras. PHP es un poderoso lenguaje e intérprete, es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor. [PHP]

1.8.7 Ajax

AJAX, acrónimo de JavaScript asíncrono y XML, es una técnica de desarrollo web para crear aplicaciones interactivas. Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma.

AJAX es una combinación de tres tecnologías ya existentes:

- XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.
- Modelo de Objetos de Documento (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.

- El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto iframe en lugar del XMLHttpRequest para realizar dichos intercambios.
- XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML pre formateado, texto plano, JSON.

AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente.

1.8.8 DOM

El Modelo de Objetos de Documento, frecuentemente abreviado DOM, es una forma de representar los elementos de un documento estructurado (tal como una página web HTML o un documento XML) como objetos que tienen sus propios métodos y propiedades.

En efecto, el DOM es una API para acceder, añadir y cambiar dinámicamente contenido estructurado en documentos con lenguajes como ECMAScript (JavaScript).

1.8.9 CSS

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas.

La separación de los contenidos y su presentación presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo. Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Mientras que el lenguaje HTML/XHTML se utiliza para *marcar* los contenidos, es decir, para designar lo que es un párrafo, lo que es un titular o lo que es una lista de elementos, el lenguaje CSS se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista, entre otros.

Las ventajas de utilizar CSS son:

- Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.

- Los Navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario. Por ejemplo, para ser impresa, mostrada en un dispositivo móvil, o ser "leída" por un sintetizador de voz.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.
- Se trata de un programa de la casa Zend, impulsores de la tecnología de servidor PHP, orientada a desarrollar aplicaciones web, como no, en lenguaje PHP. El programa, además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código.
- El programa entero está escrito en Java, lo que a veces supone que no funcione tan rápido como otras aplicaciones de uso diario. Sin embargo, esto ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para Windows, Linux.

1.8.10 Zend Studio para Eclipse

Zend Studio consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Las dos partes se instalan por separado, la del cliente contiene el interfaz de edición y la ayuda. Permite además hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración habrá que disponer de la parte del servidor, que instala Apache y el módulo PHP o, en caso de que estén instalados, los configura para trabajar juntos en depuración.

Si deseamos aumentar la productividad en nuestros desarrollos PHP no cabe duda que este programa puede ayudarnos seriamente. Todas las opciones que dispone están pensadas con acierto por personas que conocen como nadie la tecnología.

1.8.11 Servidores de Aplicación

El servidor de aplicaciones fue introducido como parte del diseño de tres capas. Es relativamente nuevo y aún no está bien definido. Las empresas del mundo entero están esforzándose para producir su propia versión de lo que creen que es un servidor de aplicaciones. La definición más común de un servidor de aplicaciones es la de software corriendo en una capa intermedia entre un cliente pequeño y una base de

datos. Generalmente se acepta que un servidor de aplicaciones maneja todas las transacciones lógicas y de conectividad que históricamente compartían el cliente y el servidor en un diseño cliente/servidor. (James 2005)

La propuesta se desarrollará utilizando el servidor HTTP Apache, por las siguientes características:

- Constituye parte de la amplia gama de software libre, proporcionado código abierto para plataformas Unix (BSD, GNU/Linux, entre otras), Windows entre otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.
- Constituye el complemento perfecto para la creación de aplicaciones Web dinámicas con PHP.
- Presenta una gran sencillez en el manejo. (Luke and Laura 2003)
- A pesar de haber sido criticado por la falta de una interfaz gráfica que ayude en su configuración brinda la posibilidad de obtener mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido.

Funciona sobre muchas plataformas (diferentes versiones de Unix, Linux).

1.8.12 Sistemas Gestores de Bases de Datos (SGBD).

Un Sistema Gestor de base de datos (SGBD) es un conjunto de programas que permiten crear y mantener una Base de datos, asegurando su integridad, confidencialidad y seguridad. Es un software de propósito general que facilita el proceso de definir, construir y manipular la base de datos para diversas aplicaciones. Actualmente el gestor de base de datos juega un rol central en la informática, como única utilidad, o como parte de otra aplicación. Características de los sistemas gestores de basas de datos:

[Matos]

- Independencias de los datos y los programas de aplicación.
- Minimización de la redundancia.
- Integración y sincronización de las bases de datos.
- Integridad de los datos.
- Seguridad y protección de los datos.
- Facilidad de manipulación de la información.
- Control centralizado.

PostgreSQL

Debido a las características y las ventajas que ofrece este gestor de base de datos, es el que se utilizará en esta investigación.

PostgreSQL se diseñó como una base de datos orientada a objetos. Esto significa, que las tablas no son tablas, sino objetos, y las tuplas son instancias de ese objeto. Puedes crear nuevos tipos de datos, hacer herencias entre objetos, PostgreSQL tiene transacciones, integridad referencial, vistas, y multitud de funcionalidades. Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, cadenas de bits, etc. También permite la creación de tipos propios. Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc. Permite la declaración de funciones propias, así como la definición de disparadores. Soporta el uso de índices, reglas y vistas. Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

PostgreSQL, permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Los bloques de código que se ejecutan en el servidor, pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos da, desde las operaciones básicas de programación, tales como bifurcaciones y ciclos, hasta las complejidades de la programación orientada a objetos o la programación funcional.

1.9 Ext JS

Ext JS, se comenzó a desarrollar a principios del 2006 por Jack Slocum como un conjunto de extensiones), estas extensiones recibieron el nombre de yui-ext. El otoño del 2006 la librería gana tanta popularidad que cambia su nombre a Ext como un reflejo de su madurez e independencia como framework. En 2007 se forma una empresa y se liberan versiones.

Ext JS es una librería para interfaces avanzadas, es completo y extremadamente avanzado, que contiene casi todo lo que se pueda imaginar y nace como solución a tareas comunes pero complejas. Está completamente orientado a objetos. Cada objeto contiene lo típico: propiedades, métodos, eventos...etc.

Ext basa toda su funcionalidad en JS a través de librerías ya muy conocidas: YUI, jQuery y Prototype/Script.aculo.us. Así, en tiempo de ejecución carga y crea todos los objetos HTML a través del uso intenso de DOM.

Ventanas, mensajes emergentes, grids, date pickers y un sin número de utilidades son todas creadas en tiempo de ejecución. Los datos son obtenidos con mucho AJAX a través de XML y/o JSON.

Ventajas:

- La orientación a objetos intensa te hará modular todos tus scripts.
- El diseño está completamente separado de la funcionalidad.

- Funciones comunes como validación, comboboxes editables, ventanas arrastables (con minimizar y maximizar), grillas editables, son muy fáciles de implementar.
- Permite la reutilización de código.
- Buena y amplia documentación, así como también su comunidad.

1.10 Arquitectura Cliente /Servidor

¿Que es una arquitectura?

Una arquitectura es un entramado de componentes funcionales que aprovechando diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios informáticos, de manera que pueden ser utilizados eficazmente dentro de la organización. Se debe señalar que para seleccionar el modelo de una arquitectura, hay que partir del contexto tecnológico y organizativo del momento y, que la arquitectura Cliente/Servidor requiere una determinada especialización de cada uno de los diferentes componentes que la integran.

¿Que es un cliente?

Es el que inicia un requerimiento de servicio. El requerimiento inicial puede convertirse en múltiples requerimientos de trabajo a través de redes de área local y redes de área extensa o extendida (LAN o WAN). La ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente.

¿Qué es un servidor?

Es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LAN o WAN, para proveer de múltiples servicios a los clientes y ciudadanos tales como impresión, acceso a bases de datos, fax, procesamiento de imágenes, etc.

La arquitectura de tres capas se refiere a un diseño reciente que introduce una capa intermedia al proceso. Cada capa es un proceso separado y bien definido corriendo en plataformas separadas. La parte funcional de la arquitectura de tres capas generalmente es conocida como la capa intermedia o el servidor de aplicaciones. En éste ocurren la mayoría de los procesos. (Enrique. 2005)

Estas son las razones por lo que sea asumen la arquitectura de tres capas:

- Las llamadas de la interfaz del usuario en la estación de trabajo, al servidor de capa intermedia, son más flexibles que en el diseño de dos capas, ya que la estación sólo necesita transferir parámetros a la capa intermedia.

- Con la arquitectura de tres capas, la interfaz del cliente no es requerida para comprender o comunicarse con el receptor de los datos. Por lo tanto, esa estructura de los datos puede ser modificada sin cambiar la interfaz del usuario en la PC (Computadora Personal).
- El código de la capa intermedia puede ser reutilizado por múltiples aplicaciones si está diseñado en formato modular. Esto puede reducir los esfuerzos de desarrollo y mantenimiento, así como los costos de migración.
- La separación de roles en tres capas, hace más fácil reemplazar o modificar una capa sin afectar a los módulos restantes.
- Separando la interfaz del usuario de la aplicación, libera de gran procesamiento a la estación de trabajo y permite que las actualizaciones de la aplicación sean centralizadas en el servidor de aplicaciones. (James 2005)

1.11 Framework

Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. En el desarrollo de software, un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Un framework representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio. Los Frameworks son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional. Un frameworks proporciona una estructura al código y hace que los desarrolladores escriban código mejor, más entendible y manejable. Además hace la programación más fácil, convirtiendo complejas funciones en sencillas instrucciones. Está usualmente escrito en el lenguaje que extiende.

Un frameworks permite separar en capas la aplicación. En general, divide la aplicación en tres capas:

- La lógica de presentación que administra las interacciones entre el usuario y el software.
- La Lógica de Datos que permite el acceso a un agente de almacenamiento persistente u otros.

- La lógica de dominio o de negocio, que manipula los modelos de datos de acuerdo a los comandos recibidos desde la presentación.

Características de los frameworks:

Abstracción de URLs y sesiones: No es necesario manipular directamente las URLs ni las sesiones, el framework ya se encarga de hacerlo.

Acceso a datos: Incluyen las herramientas e interfaces necesarias para integrarse con herramientas de acceso a datos, en XML.

Controladores: La mayoría de frameworks implementa una serie de controladores para gestionar eventos, como una introducción de datos mediante un formulario o el acceso a una página. Estos controladores suelen ser fácilmente adaptables a las necesidades de un proyecto concreto.

Autenticación y control: Incluyen mecanismos para la identificación de usuarios de acceso, mediante usuario y contraseña y permiten restringir el acceso a determinadas páginas a determinados usuarios.

Las principales ventajas de la utilización de un framework son:

1. El desarrollo rápido de aplicaciones. Los componentes incluidos en un framework constituyen una capa que libera al programador de la escritura de código de bajo nivel.
2. La reutilización de componentes software al por mayor. Los frameworks son los paradigmas de la reutilización.
3. El uso y la programación de componentes que siguen una política de diseño uniforme. Un framework orientado a objetos logra que los componentes sean clases que pertenezcan a una gran jerarquía de clases, lo que resulta en bibliotecas más fáciles de aprender a usar.
4. La portabilidad de aplicaciones de una arquitectura a otra.

1.11.1 Symfony

Symfony es un completo frameworks diseñado para optimizar el desarrollo de las aplicaciones web mediante algunas de sus principales características. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web.

Está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server. Se puede ejecutar tanto en plataformas (Unix, Linux, etc.) como en plataformas Windows.

Características

Symfony se diseñó para que se ajustara a los siguientes requisitos:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y estándares).
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de convenir en vez de configurar, en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con las librerías de otros fabricantes.

Automatización de características de proyectos web

Symfony automatiza la mayoría de elementos comunes de los proyectos web, como por ejemplo:

- La capa de internacionalización que incluye Symfony permite la traducción de los datos y de la interfaz, así como la adaptación local de los contenidos.
- La capa de presentación utiliza plantillas y layouts que pueden ser creados por diseñadores HTML sin ningún tipo de conocimiento del frameworks. Los helpers incluidos permiten minimizar el código utilizado en la presentación, ya que encapsulan grandes bloques de código en llamadas simples a funciones.
- Los formularios incluyen validación automatizada y relleno automático de datos lo que asegura la obtención de datos correctos y mejora la experiencia de usuario.

- Los datos incluyen mecanismos de escape que permiten una mejor protección contra los ataques producidos por datos corruptos.
- La gestión de la caché reduce el ancho de banda utilizado y la carga del servidor.
- La autenticación y la gestión de credenciales simplifican la creación de secciones restringidas y la gestión de la seguridad de usuario.
- El sistema de enrutamiento y las URL *limpias* permiten considerar a las direcciones de las páginas como parte de la interfaz, además de estar optimizadas para los buscadores.
- El soporte de e-mail incluido y la gestión de APIs permiten a las aplicaciones web interactuar más allá de los navegadores.
- Los listados son más fáciles de utilizar debido a la paginación automatizada, el filtrado y la ordenación de datos.
- Las interacciones con Ajax son muy fáciles de implementar mediante los helpers que permiten encapsular los efectos JavaScript compatibles con todos los navegadores en una única línea de código.

Entorno de desarrollo y herramientas

Symfony puede ser completamente personalizado para cumplir con los requisitos de las empresas que disponen de sus propias políticas y reglas para la gestión de proyectos y la programación de aplicaciones. Por defecto incorpora varios entornos de desarrollo diferentes e incluye varias herramientas que permiten automatizar las tareas más comunes de la ingeniería del software:

- Las herramientas que generan automáticamente código han sido diseñadas para hacer prototipos de aplicaciones y para crear fácilmente la parte de gestión de las aplicaciones.
- El frameworks de desarrollo de pruebas unitarias y funcionales proporciona las herramientas ideales para el desarrollo basado en pruebas.
- La barra de depuración web simplifica la depuración de las aplicaciones, ya que muestra toda la información que los programadores necesitan sobre la página en la que están trabajando.
- La interfaz de línea de comandos automatiza la instalación de las aplicaciones entre servidores.
- Es posible realizar cambios de la configuración (sin necesidad de reiniciar el servidor).

¿Por qué usar Symfony?

Porque permite asegurar el mantenimiento y las ampliaciones futuras de la aplicación, es necesario que el código sea ligero, legible y efectivo. Como se quiere incorporar los últimos avances en interacción con

usuarios (como por ejemplo Ajax), se puede escribir cientos de líneas de JavaScript. Se puede desarrollar aplicaciones de forma divertida y muy rápida, no es aconsejable utilizar solo PHP. En todos estos casos, se utilizaría Symfony pues es un frameworks maduro, bien documentado y con una gran comunidad.

1.12 Patrón Modelo Vista Controlador (MVC)

El patrón Modelo-Vista-Controlador es una guía para el diseño de arquitecturas de aplicaciones que ofrezcan una fuerte interactividad con usuarios. Este patrón organiza la aplicación en tres modelos separados, el primero es un modelo que representa los datos de la aplicación y sus reglas de negocio, el segundo es un conjunto de vistas que representa los formularios de entrada y salida de información, el tercero es un conjunto de controladores que procesa las peticiones de los usuarios y controla el flujo de ejecución del sistema. La mayoría, por no decir todos, de los frameworks para Web implementan este patrón.

Esta arquitectura de aplicaciones otorga varias ventajas clave al desarrollo de aplicaciones web, destacando:

- Al separar de manera clara la lógica de negocio (modelo) de la vista permite la reusabilidad del modelo, de modo que la misma implementación de la lógica de negocio que maneja una aplicación pueda ser usado en otras aplicaciones, sean éstas web o no.
- Permite una sencilla división de roles, dejando que sean diseñadores gráficos sin conocimientos de programación o desarrollo de aplicaciones los que se encarguen de la realización de la capa vista, sin necesidad de mezclar código Java entre el código visual que desarrollen (tan sólo utilizando algunos tags, no muy diferentes de los usados en el código HTML).

1.13 Conclusiones del Capítulo

Con el desarrollo de este capítulo, se logra tener una mayor comprensión del tema central de la tesis, la gestión de conocimiento, desde sus objetivos, hasta herramientas que existe para así llegar a lo que se define como lección aprendida. También se analizan las metodologías y herramientas informáticas más utilizadas en la actualidad, y las razones por las que se escogió como metodología de desarrollo RUP, UML como lenguaje de modelado y Visual Paradigm como herramienta para la modelación del análisis, diseño e implementación del sistema. Además de utilizarse como lenguaje de programación PHP, aplicando el Framework Symfony aprovechando las amplias ventajas que el mismo ofrece y que fueron

descritas en este capítulo. Además se utilizará como gestor de base de datos PostgreSQL por ser compatible con el lenguaje seleccionado y brindar diferentes funcionalidades descritas anteriormente.

CAPITULO 2: CARACTERISTICAS DEL SISTEMA

2.1 Introducción

Una vez analizadas las bases teóricas de la investigación, y de haber llegado a la conclusión que tecnologías se utilizarán para la realización de la herramienta, en el presente capítulo se realiza la propuesta del sistema, se analizan los requerimientos funcionales y no funcionales, además de los casos de uso a implementar y la descripción textual de cada uno de ellos.

2.2 Descripción del proceso a implementar

El proceso que se debe implementar en la aplicación define que en los proyectos productivos u organizaciones (equipos dentro del proyecto, proyecto, polo) al concluir determinadas iteraciones o fases debe realizar una actividad que esté dirigida por el líder de proyecto y en la que participen especialistas que conformen el proyecto con el fin de hacer un encuentro, donde en forma de debate se analicen todos los problemas que se presentaron a lo largo del proceso de desarrollo del proyecto y cuales fueron las soluciones y alternativas que se buscaron para solucionarlos, de esta forma se obtiene las lecciones aprendidas que pueden ser tanto positivas como negativas.

Una vez definidas estas lecciones cada una de los roles del proyecto pueden registrar sus lecciones. Estas lecciones para ser publicadas deben pasar por un proceso de aprobación que es donde se revisa que esta experiencia pueda ser útil para próximos proyectos, y que cumplan con todos los parámetros definidos, publicándose luego de esta aprobación la lección que fue registrada. Luego de pasar por estos procesos la información puede servir para ser consultada por los usuarios que lo requieran. A continuación se muestra el diagrama del proceso descrito anteriormente. (Figura 2.1)

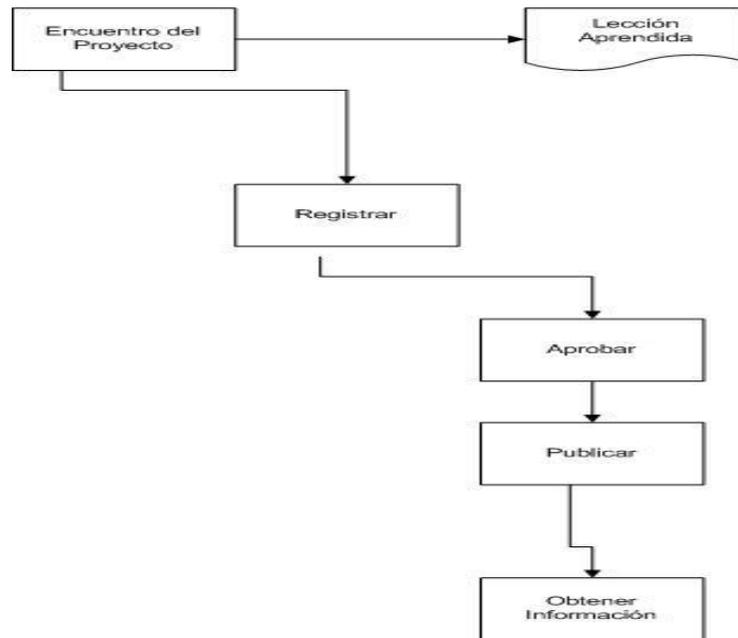


Figura 2.1 Proceso de registro y uso de la lección aprendida

2.3 Propuesta del sistema

La herramienta que se propone desarrollar, automatiza la gestión de lecciones aprendidas en los proyectos productivos de la Universidad. Para una mayor eficiencia en la implementación se organizó en tres módulos fundamentales.

El módulo de autenticación que es el encargado de verificar que el usuario que desea visitar el sitio pertenezca al dominio UCI , además de definir el nivel de acceso que va a tener a las diferentes opciones que brinda la herramienta, contribuyendo así a la seguridad del sitio.

El módulo Gestor es donde se gestionan las lecciones aprendidas esta opción podrá ser realizada por el cliente, que va a ser cualquier persona que pertenezca a un proyecto, una vez registradas estas lecciones podrán ser evaluadas por el revisor, es aquí donde las lecciones son aprobadas y publicadas o son rechazadas. A partir de toda esta información los usuarios que entren al sitio podrán realizar búsquedas y mostrar reportes.

El módulo de Administración es el encargado de gestionar las áreas, los proyectos, los roles, los usuarios y los tipos de usuarios en cada uno de ellos permite adicionar, listar y eliminar, además de permitir asignarle a un proyecto los roles que se desempeñan en el mismo, de editar los datos del usuario y de

asignarle a un cliente el proyecto al que pertenece y el rol que realiza en el mismo. Todo esto posibilita que el sitio este actualizado.

Esta herramienta se implementará utilizando el lenguaje PHP y aplicando el Framework Symfony, aprovechando las amplias ventajas que el mismo ofrece y que fueron descritas en el capítulo 1. Además se utilizará como gestor de base de datos PostgreSQL por ser compatible con el lenguaje seleccionado y brindar diferentes funcionalidades descritas anteriormente.

El sistema constará de políticas de seguridad, otorgando a cada usuario los derechos que le corresponden. Existen 4 tipos de usuarios: administrador, revisor, cliente, usuario. Los usuarios para poder realizar las opciones que brinda el sitio deberán pertenecer al dominio UCI y autenticarse.

2.4 Definición de requerimientos funcionales

Los requerimientos funcionales del proyecto definen las funciones que el sistema será capaz de realizar. [Herrera]

Los requerimientos funcionales de un sistema se corresponden con:

- Futuras opciones del software
- Acciones ocultas del software
- Condiciones extremas a determinar por el software

Para el cumplimiento de los objetivos específicos del proyecto: Proporcionar la forma de registrar las lecciones aprendidas de los proyectos productivos, a través de un sitio web y proporcionar las facilidades para la administración del sitio, se plantean los siguientes requerimientos funcionales:

2.4.1 Paquete Cliente

R1: Autenticar usuario

R1.1: Verificar usuario

R1.2: Verificar contraseña

R1.3: Definir niveles de acceso por roles

R2: Gestionar lecciones aprendidas

R2.1: Adicionar lecciones aprendidas

R2.2: Modificar lecciones aprendidas

R2.2: Eliminar lecciones aprendidas

R3: Evaluar lección aprendida

R3.1: Aprobar lección aprendida

R3.2: Publicar lección aprendida

R4: Realizar Búsquedas Avanzadas

R4.1: Buscar por palabras claves

R4.2: Buscar por un área

R4.3: Buscar por un proyecto

R4.4: Buscar por un rol

R5: Mostrar Reporte

R5.1: Mostrar las lecciones aprendidas de un cliente que han sido publicadas

R5.2: Mostrar las lecciones aprendidas por tipo.

R5.3: Mostrar las lecciones aprendidas que han sido publicadas por un revisor

2.4.2 Paquete Administración

R1: Gestionar Usuario

R1.1: Crear un nuevo usuario

R1.2: Editar datos de usuario

R1.3 Mostrar Usuario por Proyecto

R1.4 Asignar Usuario a Proyecto

R1.5 Eliminar un usuario de un proyecto.

R1.6: Eliminar usuario

R2: Gestionar Área

R2.1: Adicionar área

R2.2: Listar área

R2.3: Eliminar área

R3: Gestionar Proyecto

R3.1: Adicionar proyecto

R3.2: Listar proyectos

R3.3: Eliminar proyecto

R4: Gestionar Roles de Proyectos

R4.1: Adicionar roles

R4.2: Asignar rol a proyecto

R4.3: Listar roles

R4.4 Mostrar Roles por Proyecto

R4.5 Eliminar un rol de un proyecto.

R4.6: Eliminar roles

R5: Gestionar Tipo de Usuario

R5.1 Adicionar tipo de usuario

R5.2 Listar tipo de usuario

R5.3 Eliminar tipo de usuario

2.5 Definición de requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe de pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. [Herrera]

Los requerimientos no funcionales incluyen:

- Conjunto de facilidades
- Capacidades
- Seguridad

2.5.1 Requerimientos de Apariencia o interfaz externa

- ✓ La aplicación se desarrollara en un ambiente Web.

- ✓ Por el uso diario y constante que tendrá el software, la interfaz debe ser agradable, que favorezca el estado de ánimo del cliente y que combine correctamente los colores, tipo de letra y tamaño y que los iconos estén en correspondencia con lo que representan.
- ✓ Deben utilizarse plantillas con un mismo estilo.
- ✓ Cada página no debe exceder los 300 Kb en imágenes.
- ✓ El sistema utilizará el framework Symfony para la programación en php y ExtJS para las validaciones y la programación en el cliente.

2.5.2 Requerimientos de Usabilidad

- ✓ El sistema podrá ser utilizado por cualquier persona que este registrado en dominio UCI los cuales tendrán la opción de realizar búsquedas.
- ✓ El sitio tendrá la opción de registrar las lecciones aprendidas, pero para esto es necesario que el usuario sea integrante de uno de los proyectos productivos de la facultad 4.
- ✓ Las lecciones aprendidas deben ser aprobadas, para luego ser publicadas y los autorizados para estas tareas son los revisores.
- ✓ El administrador es el encargado de mantener actualizado el sitio haciendo los cambios necesarios en lo referente a la gestión de usuario, área, proyecto, rol y tipo de usuario.

2.5.3 Requerimientos de Rendimiento

- ✓ La eficiencia del producto estará determinada en gran medida por el aprovechamiento de los recursos que se disponen en el modelo Cliente/Servidor, y la velocidad de las consultas en la Base de Datos.
- ✓ La aplicación debe estar concebida para el consumo mínimo de recursos.
- ✓ El sistema debe ser capaz de formular la respuesta lo más rápido posible.

2.5.4 Requerimientos de Seguridad

- ✓ El sistema debe comunicarse usando un protocolo seguro, (https).
- ✓ Mantener la integridad de la información, es decir que no se pierda durante su almacenamiento o transporte.
- ✓ Permitir que cuando se borre cualquier documento o información pueda existir una opción de advertencia antes realizar la acción.

Confiabilidad: Al sistema tendrán acceso todas las personas que estén registradas en el dominio UCI y todos estos usuarios tendrán un nivel de acceso determinado, por lo que es necesaria una autenticación previa, estando la información a la que estos acceden protegida del acceso no autorizado y la divulgación.

Integridad: La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción y estados inconscientes.

Disponibilidad: El sistema deberá estar disponible las 24 horas del día para todos los usuarios.

2.5.5 Requerimientos de Software

Para la máquina cliente del usuario:

- ✓ Un navegador para visualizar la aplicación e interactuar con el sistema en cuestión
- ✓ Sistema Operativo Linux o Windows

Para la máquina servidor:

- ✓ Sistema Operativo Windows XP Professional
- ✓ Servidor apache con módulo PHP5 disponible.
- ✓ Un servidor de Base de Datos PostgreSQL

2.5.6 Requerimientos de Hardware

Para la máquina cliente del usuario:

- ✓ Requerimientos Mínimos: Procesador Pentium III a 1GHz de velocidad de procesamiento y 256 de memoria RAM.
- ✓ Tarjeta de red

Para la maquina servidor:

- ✓ Requerimientos Mínimos: Procesador Pentium IV a 2.8 GHz de velocidad de procesamiento y a 256 de memoria RAM.
- ✓ Tarjeta de red

2.5.6 Requerimientos de Portabilidad

- ✓ La herramienta propuesta podrá ser usada bajo cualquier sistema operativo, para su implementación se usaron herramientas de programación y Gestión de Base de Datos que son multiplataforma.

2.5.7 Restricciones en el diseño y la implementación

Es necesario usar para la implementación de la herramienta lenguajes de programación que sean multiplataforma, en este caso PHP, que además ha ido en ascenso en los últimos tiempos. Por esta

misma razón es necesario utilizar un Sistema Gestor de Base Datos multiplataforma, en este caso PostgreSQL. Para garantizar una mejor documentación del sistema, así como el uso de últimas tecnologías, se utiliza para el análisis y el diseño del sistema UML y como herramienta de apoyo a este lenguaje de programación se utiliza el Visual Paradigm.

2.6 Definición general de los casos de usos

Un caso de usos es un documento narrativo que describe la secuencia de un actor (agente externo) que utiliza un sistema para completar un proceso. [Alvarez]

2.6.1 Identificación de los actores

Un actor no es parte del sistema, es un rol de un usuario que puede intercambiar información o puede ser un recipiente pasivo de información y representa a un ser humano, a un software o a una máquina que interactúa con el sistema. [Alvarez]

A continuación se definen los actores del sistema propuesto:

Actor	Descripción
Usuario	Es aquella persona autorizada a entrar al sistema, es decir que estén registradas en el dominio UCI y según su nivel podrá llevar a cabo acciones dentro del mismo
Administrador	Es el encargado de mantener actualizado el sitio haciendo los cambios necesarios en lo referente a la gestión de usuario, área, proyecto, rol y tipo de usuario.
Cliente	Es cualquier integrante de un proyecto productivo, de la universidad. Encargado de registrar las experiencias adquiridas en el proyecto.
Revisor	Es aquella persona que se encarga de aprobar y publicar, las lecciones aprendidas que han sido registradas por los clientes.

2.6.2 Casos de usos del Paquete Cliente

CU-1	Autenticar Usuario
Actor	Usuario
Descripción	El Caso de Uso se inicia cuando el usuario introduce los datos que se le piden para acceder a la aplicación, estos se verifican y finaliza dándole los permisos y habilitándole la entrada.

Referencia	R1
-------------------	----

CU-2	Gestionar Lecciones Aprendidas
Actor	Cliente
Descripción	El Caso de Uso se inicia cuando el cliente desea registrar, modificar o eliminar una lección aprendida, finaliza cuando la información queda actualizada.
Referencia	R2

CU-3	Evaluar Lecciones Aprendidas
Actor	Revisor
Descripción	El Caso de Uso se inicia cuando el revisor, evalúa la lección aprendida registrada por el cliente, finaliza cuando la lección es aprobada y publicada, o es rechazada.
Referencia	R3

CU-4	Realizar Búsquedas Avanzadas
Actor	Usuario
Descripción	El Caso de Uso se inicia cuando el usuario, selecciona una de las opciones de búsquedas, finaliza cuando el usuario obtiene el resultado de la búsqueda seleccionada.
Referencia	R4

CU-5	Mostrar Reportes
Actor	Usuario
Descripción	El Caso de Uso se inicia cuando el usuario, selecciona una de las opciones de mostrar reportes, finaliza cuando el usuario obtiene el resultado del reporte seleccionado.
Referencia	R5

2.6.3 Casos de usos del Paquete Administración

CU-1	Gestionar Usuario
Actor	Administrador
Descripción	<p>El caso de uso comienza cuando el Administrador va a realizar alguna de las siguientes operaciones:</p> <ul style="list-style-type: none"> • Crear nuevo usuario. • Editar datos de usuario. • Mostrar Usuario por Proyecto. • Asignar Usuario a Proyecto. • Eliminar un usuario de un proyecto. • Eliminar usuario. <p>El sistema le muestra la interfaz correspondiente según su solicitud y ejecuta las acciones necesarias. El caso de uso finaliza cuando se emite el resultado de la operación solicitada.</p>
Referencia	R1

CU-2	Gestionar Área
Actor	Administrador
Descripción	<p>El Caso de Uso se inicia cuando el administrador selecciona una de las opciones, ya sea adicionar, listar o eliminar un área y finaliza cuando la información queda registrada en la base de datos.</p>
Referencia	R2

CU-3	Gestionar Proyecto
Actor	Administrador
Descripción	<p>El Caso de Uso se inicia cuando el administrador selecciona una de las</p>

	opciones ya sea adicionar, asignar, listar o eliminar un proyecto y finaliza cuando la información queda registrada en la base de datos.
Referencia	R3

CU-4	Gestionar Roles de Proyecto
Actor	Administrador
Descripción	El Caso de Uso se inicia cuando el administrador selecciona una de las opciones ya sea adicionar rol, asignar rol a un proyecto, listar rol, mostrar rol por proyecto, eliminar un rol de un proyecto o eliminar un rol y finaliza cuando la información queda registrada en la base de datos.
Referencia	R4

CU-5	Gestionar Tipo de Usuario
Actor	Administrador
Descripción	El Caso de Uso se inicia cuando el administrador selecciona una de las opciones ya sea adicionar, asignar, listar o eliminar un tipo de usuario y finaliza cuando la información queda registrada en la base de datos.
Referencia	R5

2.7 Diagramas de Casos de Usos y paquetes

Los paquetes de casos de usos son las formas de agrupar a estos últimos respondiendo a algún criterio, se representan a través de los diagramas que evidencian gráficamente la relación entre los actores y los casos de usos. se definieron dos paquetes: Paquete Cliente y paquete Administración, en el paquete Cliente contiene los casos de usos que se referencian con los requerimientos funcionales que representan las opciones del sitio web del cliente, en el Paquete Administración están los casos de usos que representan las opciones de administración, mostrándose en los diagramas que recogen las figuras 2.1 y 2.2, respectivamente.

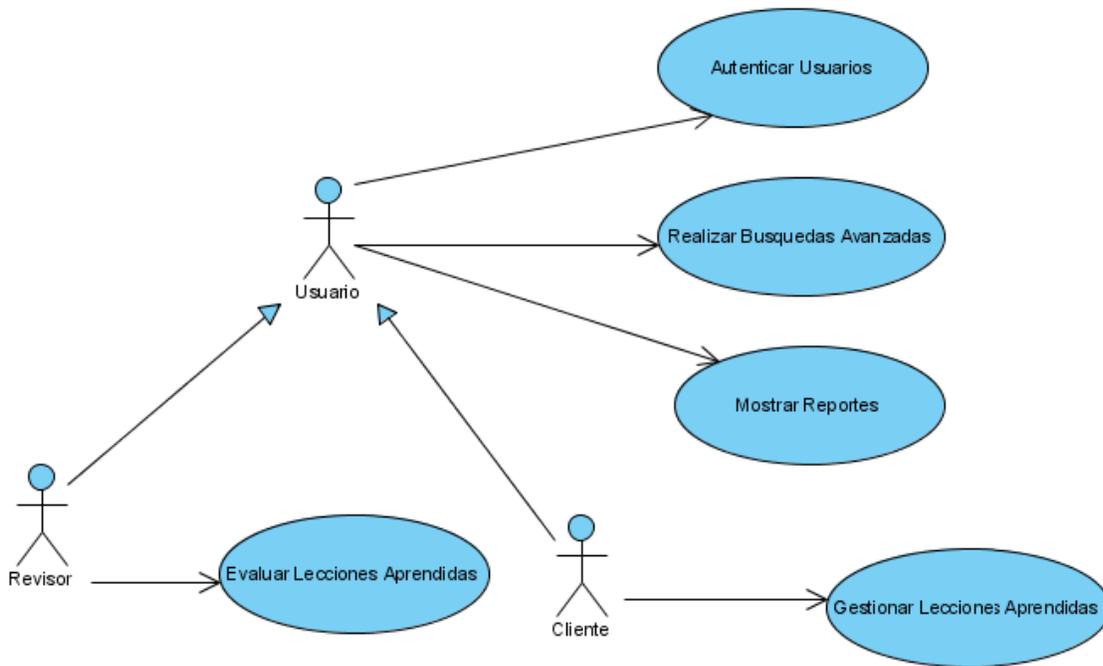


Figura 2.1 Diagrama de caso de usos del Paquete Cliente

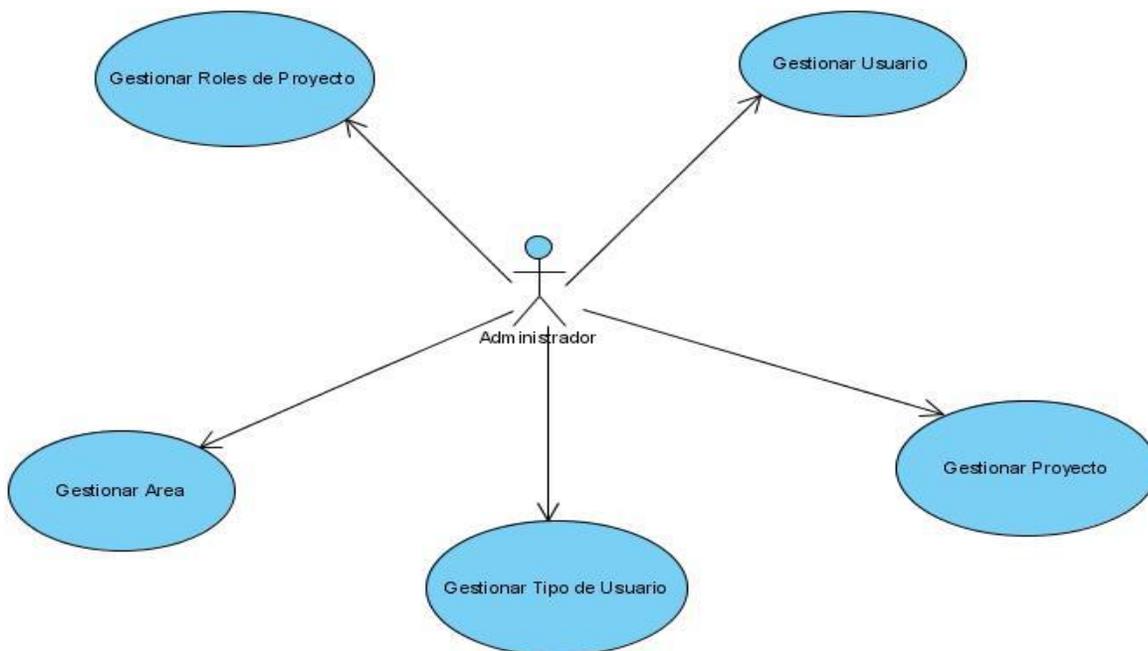


Figura 2.2 Diagrama de caso de usos del Paquete Administración.

2.8 Descripción Textual de los Casos de Uso

Mediante la descripción de los caso de usos, se describen paso a paso la secuencia de eventos que los actores utilizan para completar un proceso a través del sistema [Álvarez]. En el anexo 1 se muestra las descripciones de cada uno de los casos de usos que serán implementados.

2.9 Conclusiones

Se determinaron las funcionalidades necesarias con que la solución debe cumplir y modelar una propuesta en términos de casos de uso, así como los actores que estarán involucrados con este sistema. Concluye así este capítulo, dando paso a la construcción del sistema.

CAPITULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

3.1 Introducción

En este capítulo se muestran los resultados obtenidos en el desarrollo de los procesos de análisis y diseño del sistema, así como los diagramas que fueron necesarios utilizar para obtener una mayor claridad a la hora de elaborar la solución que se propone y para lograr una implementación acorde a las necesidades de los usuarios.

3.2 Análisis.

Durante este flujo de trabajo se analizan los requisitos que se describieron en la captura de requisitos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que ayude a estructurar el sistema entero, incluyendo su arquitectura. El modelo de análisis está escrito en el lenguaje del desarrollador, es la vista interna del sistema. El análisis se basa en un modelo de objetos conceptual, que es llamado modelo de análisis. (Pressman 1998)

3.2.1 Clases del análisis.

Una clase de análisis representa una abstracción de una o varias clases y/o subsistemas del diseño del sistema. Las clases de análisis se basan en tres estereotipos básicos: interfaz, entidad y control.

Clases de Interfaz: se utilizan para modelar la interacción entre el sistema y sus actores (usuarios y sistemas externos). Modelan las partes del sistema que dependen de sus actores. Representan abstracciones de ventanas, formularios, paneles, interfaces de comunicación, interfaces de impresora, sensores y terminales.

Clases de Entidad: se utiliza para modelar información que posee una vida larga y que es a menudo persistente.

Clases controladoras: Representan coordinación, secuencia, transacciones, y control de otros objetos y se usa con frecuencia para encapsular el control de un caso de uso en concreto, modelan los aspectos dinámicos del sistema. (Pressman 1998)

En anexo 2 se describen los diagramas de clases del análisis de los casos que conforman la solución propuesta.

3.3 Diseño

El propósito del diseño es modelar el sistema y encontrar su forma para que soporte todos los requisitos. Es un modelo físico que crea una entrada apropiada y un punto de partida para la implementación, es decir es una abstracción a la implementación del sistema. (Date 2003)

Anexo 3 Diagramas de clases del diseño del sistema a implementar.

3.3.1 Diagramas de interacción

Los diagramas de interacción describen el modo en que cada operación detectada en los diagramas de secuencia y colaboración, lleva a cabo sus responsabilidades y modifica el estado del sistema.

Diagramas de secuencia: Se muestra la interacción entre objetos mediante transferencia de mensajes entre objetos o subsistemas.

Diagramas de colaboración: Se muestra la interacción entre objeto creando enlaces entre ellos y añadiendo mensajes a esos enlaces. (Pressman 1998)

Anexo 4 Diagramas de secuencia de los casos de usos que conforman la propuesta del sistema.

3.4 Diseño de la base datos

Mediante este diseño se representa las tablas de la base de datos. Además de describirse las tablas y cada uno de sus atributos.

Modelo Entidad Relación

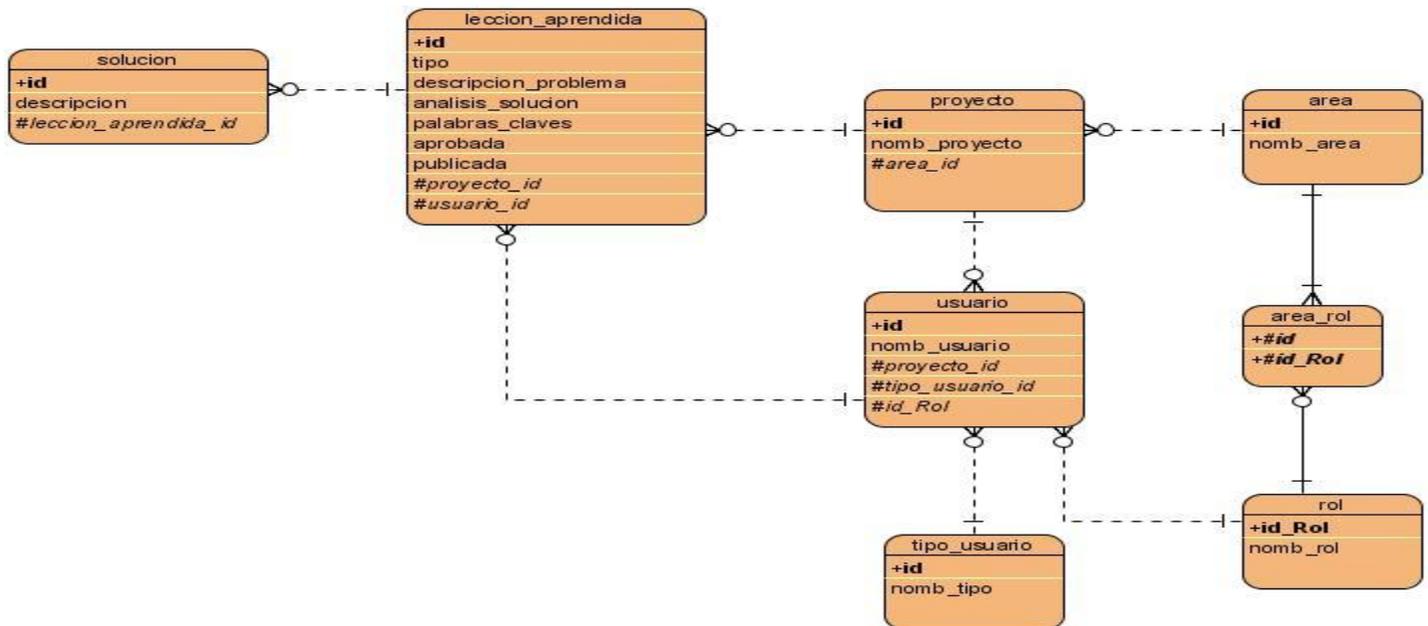


Figura 3.1 Diagrama de entidad relación. Del sistema "Gestión de Lecciones Aprendidas".

Anexo 5 Descripción de las tablas de la Base de Datos

3.5 Tratamiento de Errores

Se contará con un sistema de tratamiento de errores para disminuir la posibilidad de cometerlos. Para esto se contará con la validación de la información introducida en el sistema por la validación de los formularios utilizando funciones Java Scripts.

Mediante la interfaz Web se evitará que el usuario asuma un papel activo en la introducción de la información, para esto se contará con un menú de selección lo cual facilitará la entrada de datos. La información que requiera ser adicionada por el usuario se validará mediante funciones que garanticen que sea válida y que el cuadro de texto no esté vacío si es obligatorio llenarlo. Si hay un error en la información le saldrá al usuario un mensaje en pantalla indicándole el error, al oprimir Aceptar el mensaje desaparece y el usuario podrá seguir introduciendo los datos en el formulario.

También se validarán las opciones correspondientes a la extracción o modificación de datos del servidor de base datos. Si se desea eliminar algún elemento de la BD se preguntará al usuario si está seguro de realizar dicha acción, al igual que cuando desee modificar alguna información, antes de actualizarla se le preguntará si desea realizarla o no. Así se logra que se realicen las operaciones que se desean y que se rectifique al cometer un error.

A continuación se muestran algunos de estos mensajes:

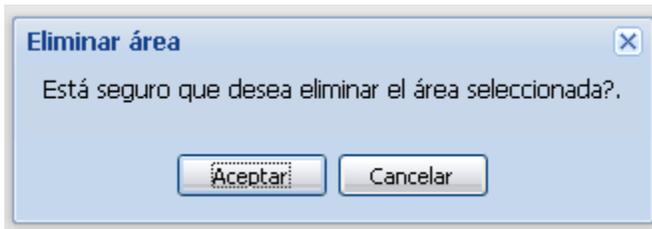


Figura 3.2 Mensaje de verificación. Eliminar área.

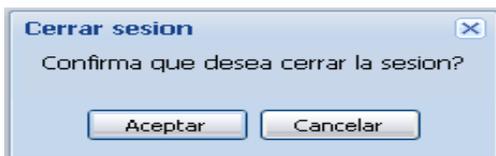


Figura 3.3 Mensaje de verificación. Cerrar sección

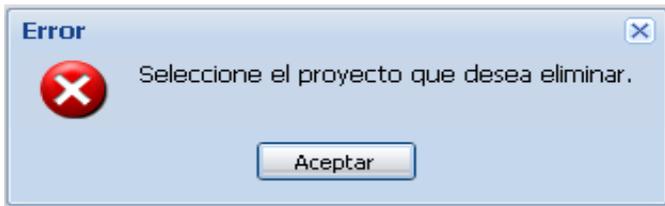


Figura 3.4 Mensaje de validación. Eliminar proyecto.

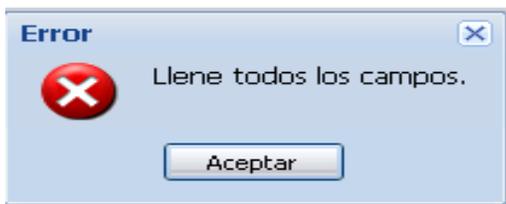


Figura 3.5 Mensaje de validación. Autenticar Usuario.

3.6 Concepción de la Ayuda.

La Ayuda es un componente necesario en todo software que esté diseñado para cumplir los intereses del usuario. Esta debe poseer informaciones generales de la aplicación y orientaciones detalladas de las funcionalidades del sistema. La ayuda debe formar parte del menú en todas las pantallas del sistema, con el objetivo fundamental de que el usuario pueda concluir aquellas operaciones que esté realizando en dicho momento.

Además se debe tener en cuenta la confección de manuales de usuario que deberán hacérsele llegar a los usuarios que trabajan con el sistema. Dichos manuales explicarán de forma detallada las principales funcionalidades y bondades que brinda el software.

3.7 Seguridad

Para seguridad del sitio utilizamos la clase LDAP que permite verificar que el usuario esta en el dominio UCI y que los datos que introduce sean correctos, en caso de no ser así no tendrá acceso a la aplicación. Además se crearon varios niveles de seguridad, definidos como tipos de usuario, que pueden ser: Usuario, Cliente, Revisor o Administrador. Este último es el encargado del buen funcionamiento del sistema por lo que tendrá derecho al control total del mismo. Los demás usuarios no tendrán acceso a la información restringida para ellos, para esto utilizamos el manejador de permisos integrado que tiene Symfony, de forma tal que siempre se sabe qué usuario intenta visitar dichas páginas y estas se muestran solo para aquellos que pueden tener acceso a ellas.

A continuación se muestra la interfaz de autenticación del sitio:

Una interfaz de usuario de autenticación con un título "Bienvenido" en la parte superior izquierda. Hay dos campos de entrada de texto: "Usuario:" y "Contraseña:". Debajo de los campos de texto hay un botón que dice "Aceptar".

Figura 3.6 Interfaz de Autenticación.

3.8 Principios de diseño del sitio

Para garantizar un vínculo entre el usuario y el sistema realizado se ha hecho un esfuerzo por garantizar la máxima claridad posible en la interfaz procurando que sea intuitiva y que garantice un empleo óptimo del sistema y una comodidad en el trabajo de los usuarios finales. Se trabajó con formularios Web, para lograr una mayor visualización de la información, se utilizó un mismo color en todas las páginas y un mismo tipo y tamaño de letra con textos claros, el color predominante es el azul. El idioma utilizado es el español.

El envío de la información debe ser lo mas rápida posible por lo que no se utilizaron muchas imágenes ni funciones que atenten contra esto y que posibiliten también una navegación rápida y eficiente. Cada página está representada por un título acorde a su funcionalidad.

El sitio tiene un baner que esta compuesto por el logotipo de la universidad, el nombre del mismo y una imagen representativa. En todas las páginas se mantiene un menú en la parte inferior al baner, que permite el acceso a todas las opciones que se brindan en el sitio, en dependencia del tipo de usuario. En cada momento en la parte superior se muestra el usuario que esta autenticado al sistema.

3.9 Conclusiones

En el capítulo se realizaron las definiciones del Modelo de Análisis, Modelo de clases de análisis así como los diagramas de análisis. Y se realizaron los diferentes diagramas de diseño, diagramas de secuencia, se describieron las clases utilizadas en el diseño, así como el modelo entidad relación de la base de datos y las descripciones de las tablas de la base de datos. También se explicó la definición del diseño que se aplicó, la forma de tratar los errores, se definió la seguridad del sistema y la interfaz.

CAPITULO 4: IMPLEMENTACIÓN Y PRUEBA

4.1 Introducción

En el capítulo se muestran las definiciones de los diferentes modelos utilizados en la implementación para la solución del problema y se procede a la construcción de los modelos especificados en los requisitos. Se realiza el modelo de despliegue y el modelo de componentes.. El modelo de implementación parte del diseño.

4.2 Implementación

Con el flujo de trabajo Implementación se le está dando comienzo a la fase de Construcción, el propósito de esta fase es dejar listo un producto software en su versión operativa inicial (versión beta), el cual debe tener la calidad adecuada para su aplicación y cumplir con los requisitos de software establecidos en el Capítulo II.

Al final de la fase de Elaboración se ha llevado el producto software al estado de una línea base de la arquitectura ejecutable. Las fases previas han reducido los riesgos críticos y significativos a niveles rutinarios que pueden ser gestionados durante el plan de construcción. Durante esta fase, el equipo del proyecto estableció los fundamentos de los elementos arquitectónicamente significativos de los modelos de diseño y despliegue.

Los objetivos principales del flujo de trabajo Implementación son:

- Definir la organización del código en términos de Subsistemas de Implementación organizados en capas.
- Implementar los elementos de diseño en términos de elementos de implementación (ficheros fuentes, binarios, ejecutables y otros).
- Probar los componentes desarrollados independientemente como unidades.
- Integrar los resultados producidos por desarrolladores independientes o equipos en sistema ejecutable.

4.3 Diagramas de Componentes

Se representa como un grafo de componentes software unidos por medio de relaciones de dependencia, pudiendo mostrarse las interfaces que estos soporten. Es un diagrama que muestra un conjunto de elementos del modelo tales como componentes, subsistemas de implementación y sus relaciones. Se utiliza para modelar la vista estática de un sistema. Muestra la organización y las dependencias lógicas

entre un conjunto de componentes software, los cuales pueden ser código fuente, librerías, binarios o ejecutables.

A continuación se muestra el diagrama de componentes del sistema organizado en paquetes.

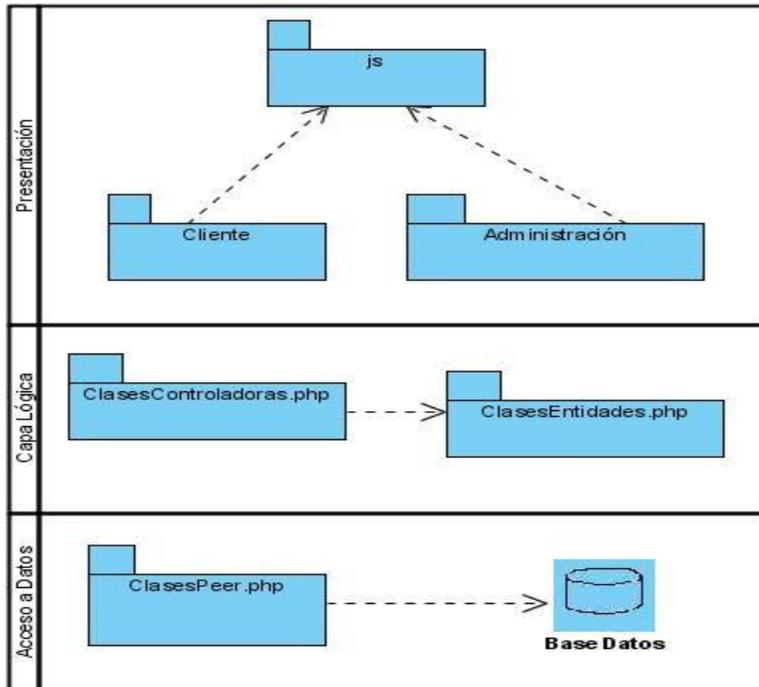


Figura 4.1 Diagrama de Componentes en Paquetes y por capas.

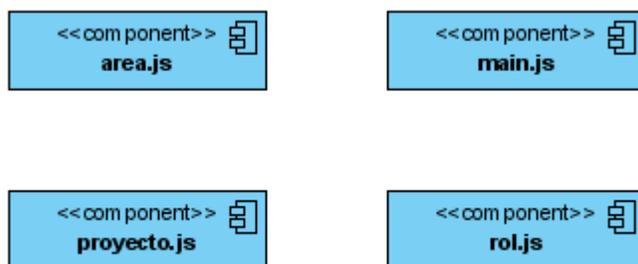


Figura 4.2 Diagrama de componente del paquete js de Administración.

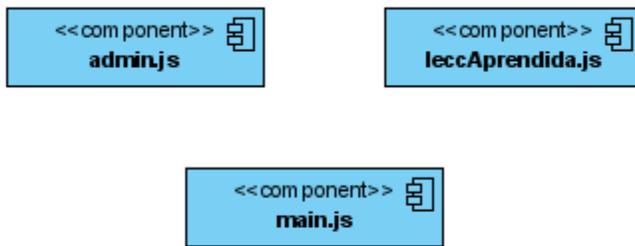


Figura 4.3 Diagrama de componente del paquete js del Cliente

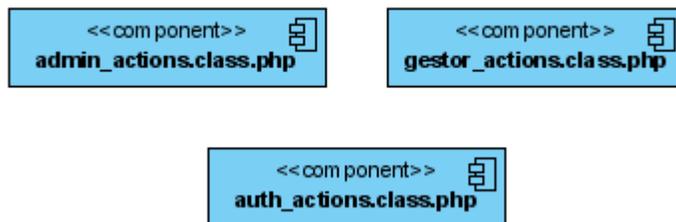


Figura 4.4 Diagrama de componente de Clases Controladoras.

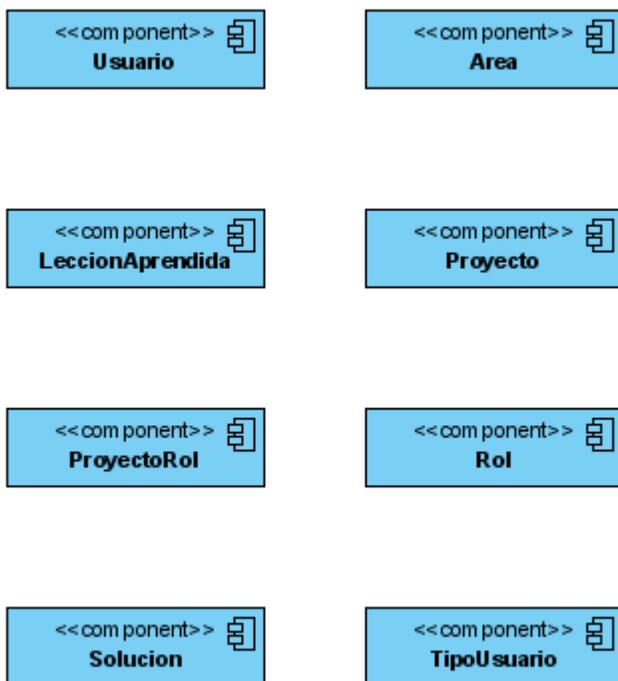


Figura 4.5 Diagrama de componente de las Clases Entidades.

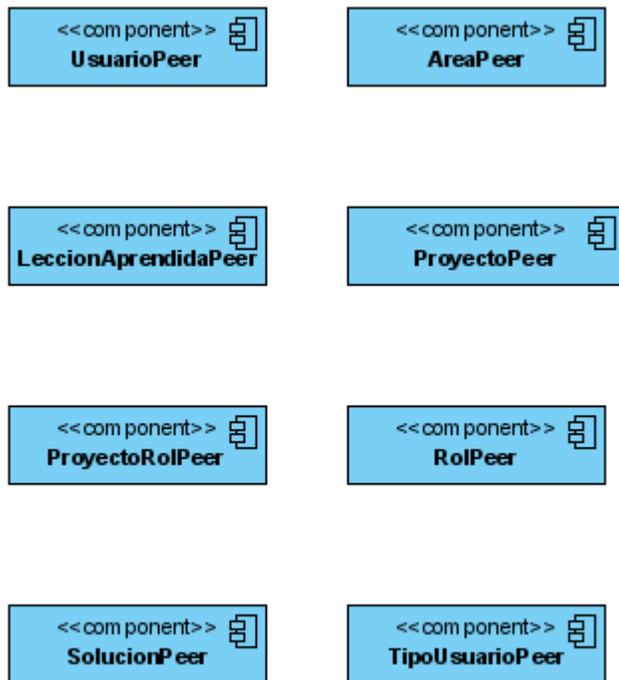


Figura 4.6 Diagrama de componente de las Clases Peer.

4.4 Diagrama de despliegue

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados, es decir, se sitúa el software en el hardware que lo contiene; cada hardware se representa como un nodo.

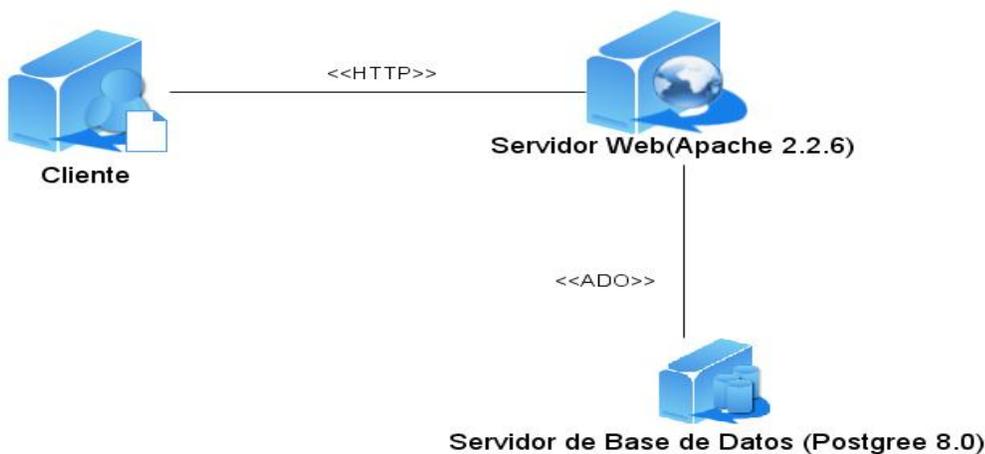


Figura 4.7 Diagrama de Despliegue

4.5 Pruebas

El proceso de pruebas de software es un elemento crítico para la garantía de la calidad del software. Es el conjunto de actividades y entregables que se sigue para realizar pruebas de software garantizando un producto confiable y sin errores. Se enfoca sobre la lógica interna del software y las funciones externas. No va orientado a demostrar la ausencia de fallos en el software, sino más bien a todo lo contrario: encontrar cuantos fallos existan, por escondidos que se encuentren. El proceso de pruebas de software toma como punto de partida los requisitos y a partir de ahí realiza las pruebas.

4.5.1 Métodos de Pruebas

Existen diferentes métodos para realizar las pruebas pero los más significativos y usados son las pruebas de caja blanca y las de caja negra. En este trabajo se aplicó el método de prueba de caja negra. Esta prueba se denomina prueba de comportamiento y básicamente se centra en los requisitos funcionales del software, lo que le permite al ingeniero de software obtener un conjunto de entradas que ejerciten todos los requisitos funcionales de este y así obtener un conjunto de errores para poder corregirlos.

4.5.2 Prueba de Sistema

Dentro de las pruebas de caja negra, las que se realizaron fueron las pruebas de sistema. Son las pruebas que se hacen cuando el software está funcionando como un todo. Es la actividad de prueba dirigida a verificar el programa final, después que todos los componentes de software y hardware han sido integrados. Las pruebas de sistema principalmente se centran en verificar la interacción de los actores con el sistema, por lo que a menudo los casos de pruebas se obtienen a partir de las descripciones de los casos de uso. Aunque también se le aplican pruebas al sistema como un todo.

En el Anexo 6 los casos de pruebas del sistema propuesto:

4.6 Conclusiones

En este capítulo se abordaron los conceptos fundamentales asociados con las fases de Implementación y Prueba en el desarrollo del software. Se realizó el modelo de despliegue y los distintos modelos de componentes que dan cumplimiento a los requisitos planteados en el sistema, así como el diseño del caso de prueba que demuestra la funcionalidad del sistema como un todo y la calidad de este.

CONCLUSIONES

En la Universidad de las Ciencias Informáticas no se contaba con ningún sistema que permitiera gestionar las lecciones aprendidas en los proyectos productivos y con este trabajo se logró desarrollar una herramienta que da respuesta a este problema, permitiendo la socialización del conocimiento.

Se realizó una exhaustiva búsqueda bibliográfica acerca de las últimas tendencias y tecnologías que a nivel internacional se están utilizando en el mundo de la informática, para definir aquellas que mejor respuesta darían al problema al cual se quería dar solución. Como conclusión, se optó por la utilización del PHP como lenguaje de programación y como framework Symfony, PostgreSQL como gestor de base de datos, RUP como metodología de desarrollo de software y el Visual Paradigm como herramienta de modelado por formar parte de la amplia gama de software libre, solución de gran importancia para Cuba actualmente.

RECOMENDACIONES

- ❖ Continuar con la investigación sobre el tema de la gestión de conocimiento en los proyectos productivos de la universidad.
- ❖ Mejorar las tecnologías y herramientas utilizadas en la implementación de la primera versión del sistema.
- ❖ Poner a prueba la herramienta implementada en los proyectos productivos de la Universidad.

REFERENCIAS BIBLIOGRÁFICAS

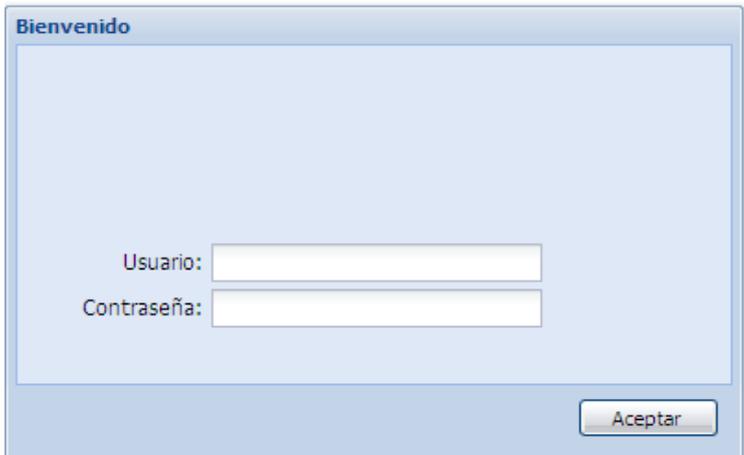
- **Álvarez**, Sofía, Hernandez Anaisa. Metodología para el desarrollo de aplicaciones con tecnologías Orientada a Objetos utilizando notación UML, La Habana.
- **Alarcos, G.** Visual Paradigm. [En línea] enero de 2007.
<http://www.macupdate.com/info.php/id/16945/visual-paradigm-for-uml>.
- **Agustí-Cullel, J. (2003):** "Issues on knowledge management", Knowledge and Process Management, vol. 10, nº 1, pp. 37-59.
- **Balasubramanian, P. (2003):** "KnowledgeScope: managing knowledge in context", Decision Support Systems, vol.35, nº 4, pp. 467-486.
Caballero, I., "Visual Paradigm", in
http://alarcos.infcr.uclm.es/per/fgarcia/isoftware/doc/LabTr1_VP.pdf. 2007.
- **Canals, Agustí. 2003.** La Gestión del Conocimiento. [En línea] 2003. [Citado el: 23 de enero de 2008.] <http://www.uoc.edu/dt/20251/index.html>.
- **Carrión, Juan.** Conocimiento. [En línea] [Citado el: 22 de enero de 2008.]
http://www.gestiondelconocimiento.com/conceptos_conocimiento.htm.
—, Gestión del Conocimiento. [En línea] [Citado el: 20 de enero de 2008.]
http://www.gestiondelconocimiento.com/conceptos_gestion_del_conocimiento.htm.
- **Date, C. J. (2003).** Introducción a los Sistemas de Bases de Datos. Ciudad de La Habana, Editorial Félix Varela.
- **Díaz Muñante, Jorge. 2003.** Modelo de Gestión del Conocimiento aplicado a la Universidad Pública de Perú. [En línea] 2003. [Citado el: 17 de enero de 2007.]
http://sisbib.unmsm.edu.pe/BibVirtual/Tesis/Basic/Diaz_MJ/Contenido.htm.
- **E. Hernández**, J. Hernández, C. Lizandra, "C++ Es-tandar", ITP Paraninfo 2001.
- **Enrique., P. R. (2005).** Herramienta para la estructuración y la gestión de contenidos de aplicaciones multimedia educativas. Facultad de ingeniería Industrial. Ciudad de La Habana, Instituto Superior Politécnico "José Antonio Echeverría"_ 128.
- **Figuroa**, Liliana; Palavecino, Rosa. Aproximación a la diferencia entre Gestión de la Información y la Gestión del Conocimiento. (2006). Disponible en:
<http://www.cibersociedad.net/congres2006/gts/comunicacio.php?id=618&llengua=es>.

- **G. Booch**, J. Rumbaugh y I. Jacobson, "El Lenguaje Unificado de Modelado", Addison Wesley, 1999
- **Gunlaugsdottir, J. (2003)**: "Seek and you will find, share and you will benefit: organising knowledge using groupware systems", *International Journal of Information Management*, vol.23, nº5, pp. 363-380.
- **Herrera J.**, Lizka Johany. "Ingeniería de requerimientos – Ingeniería de Software", <http://www.monografias.com/trabajos6/resof> (8/1/2008)
- **Hiodobro Moya**, Jose Manuel. Redes y Servicios de telecomunicaciones. Segunda Edición.
- **Jacobson**, G. Booch, J. Rumbaugh , "El Proceso Unificado de Desarrollo", Addison Wesley, 2000
- **James**, G. G. (2005). "Taller de sistemas informáticos." 2de noviembre del 2006, from http://www.pcm.gob.pe/portal_ongei/publicaciones/cultura/Lib5038.
- **Luke, W. and T. Laura (2003)**. PHP and MySQL Web Development. Llegó la hora del PostgreSQL <http://www.http-peru.com/postgresql1.php>
- **María Luisa Garzón Villar** Informática: cuerpo de profesores de enseñanza secundaria: temario 2004 ISBN 8466501339
- **Nonaka, I., Takeuchi, H.** "The Knowledge Creating Company", Oxford Universit Press, 1995. Citado por Lara, José Luis en: "Diez respuestas a las preguntas más frecuentes sobre gestión del conocimiento", Instituto Nacional de Estadística y Censos de Argentina, Consultado el 21 de enero del 2001. <http://www.gestiondelconocimiento.com>
- **Núñez, Israel. 2005.** Propuesta de clasificación de herramientas de software para conocimiento. [Citado el: 29 de enero de 2007.] http://eprints.rclis.org/archive/00004134/01/Propuesta_de_clasificacion.pdf
- **Pérez, Daniel y Dresler, Matthias 2006.** Tecnologías de la Información para la Gestión del Conocimiento. [En línea] diciembre de 2006. [Citado el: 7 de febrero de 2008.] <https://upcommons.upc.edu/revistes/bitstream/2099/2945/1/Tecnologias%20de%20la%20informacion.pdf>
- **PHP-** Introducción a php. <http://www.ciberteca.net/webmaster/php>. (17/1/2008)
- **Pressman**, R. S. (1998). "Ingeniería de Software.", from <http://www.exit109.com/~leebee/>. (15/1/08)
- **Rodriguez**, Daniel y Bravo, Joaquín. Tutorial de HTML <http://html.programacion.net>. (17/1/2008)
- **[Symfony_Guia_Definitiva]**
- **UML- a** Tutorial de UML <http://www.dcc.uchile.cl/~psalinas/uml>. (10/1/2008)
- **UML- b-** Salinas Caro, Patricio y Histchfeld K, Nancy Tutorial de UML.

ANEXOS

Anexo 1: Descripción Textual de los Casos de Uso

Casos de Usos del Paquete Cliente

Caso de Uso	Autenticar Usuario
Actor	Usuario (inicia)
Propósito	Verificar que el usuario esta autorizado a entrar a determinada sección del sistema
Resumen	El caso de uso se inicia cuando el usuario introduce su nombre de usuario y contraseña del dominio en el sistema. Este verifica que los datos son correctos y da acceso a las secciones correspondientes finalizando así el caso de uso
Precondiciones	El usuario debe haber seleccionado la opción de acceder al sitio Web
Poscondiciones	El usuario queda autenticado
Referencia	R1.
Interfaz	
	
Descripción de las entradas de datos	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1.El usuario introduce: <ul style="list-style-type: none"> • Usuario • Contraseña 2. selecciona la opción "Aceptar"	

	<p>3. El sistema verifica que ningún campo este vacío</p> <p>4. El sistema verifica que el usuario y la contraseña introducidos tengan acceso al sistema</p> <p>5. El sistema se configura según los niveles de acceso del usuario</p> <p>6. El sistema le permite la entrada al usuario mostrando las opciones que le corresponden</p>
Flujos Alternos	
	3. Si el sistema verifica que algún campo esta vacío emite un mensaje de error
	4. Si el sistema verifica que el usuario y la contraseña introducidos no tienen acceso al sistema emite un mensaje de error
Prioridad	

Caso de Uso	Gestionar Lecciones Aprendidas
Actor	Cliente (inicia)
Propósito	Permitir que los clientes, es decir los integrantes de lo proyectos productivos puedan registrar o modificar las lecciones aprendidas que han sido registradas por ellos.
Resumen	El Caso de Uso se inicia cuando el cliente desea registrar o modificar una lección aprendida ya registrada por el mismo usuario, finaliza cuando la información queda actualizada en la Base de Datos
Precondiciones	El usuario autenticado debe ser un cliente
Poscondiciones	Queda actualizada la información relacionada con las lecciones aprendidas.
Referencia	R2

Descripción de las entradas de datos

Flujo Normal de Eventos

Acción del Actor	Respuesta del Sistema
1. El cliente selecciona la opción de gestionar lecciones aprendidas.	2. El sistema muestra las opciones relacionadas con la

<p>3. El cliente selecciona una de las opciones:</p> <p>a) solicita adicionar una lección aprendida, ver sección adicionar.</p> <p>b) solicita modificar una lección aprendida, ver sección modificar.</p> <p>c) solicita eliminar una lección aprendida, ver sección eliminar.</p>	<p>gestión de lecciones aprendidas: adicionar, modificar y eliminar las lecciones aprendidas.</p>
---	---

Sección “Adicionar Lección Aprendida”

Interfaz

The screenshot shows a web interface for adding a lesson. On the left, under 'Nueva Lección Aprendida', there is a 'Tipo' dropdown menu, a 'Descripción del problema' text area, an 'Análisis de la solución' text area, and a 'Palabras claves' text field. On the right, under 'Soluciones', there is a large empty text area, an 'Adicionar' button, and a table titled 'Soluciones a la lección' with columns 'No.' and 'Solución', and an 'Eliminar' button. At the bottom are 'Aceptar' and 'Cancelar' buttons.

<p>2. El cliente introduce la información relacionada con la lección aprendida que desea adicionar y solicita que se guarde.</p>	<p>1. El sistema muestra el formulario de datos necesarios: Proyecto, Área, Rol, Tipo, Descripción del problema, Solución1, Solución2, Análisis de las soluciones, Palabras claves.</p> <p>3. El sistema verifica que ningún campo obligatorio este vacío y que los datos a entrar sean correctos.</p>
--	--

	4. El sistema adiciona la lección aprendida.
Flujos Alternos	
	3. Si el sistema verifica que algún campo obligatorio esta vacío o el tipo de alguno de los datos a entrar son incorrectos emite un mensaje de error
Sección “Modificar Lección Aprendida”	
2. El cliente selecciona la lección aprendida que desea modificar. 4. El cliente modifica la información de la lección aprendida y solicita que se guarde.	1. El sistema muestra las lecciones aprendidas que han sido registradas por ese cliente. 3. El sistema muestra la información actual de la lección aprendida seleccionada. 5. El sistema verifica que ningún campo obligatorio este vacío y que los datos a entrar sean correctos. 6. El sistema guarda la información modifica de la lección aprendida.
Flujos Alternos	
	5. Si el sistema verifica que algún campo obligatorio esta vacío o el tipo de alguno de los datos a entrar son incorrectos emite un mensaje de error
Sección “Eliminar Lección Aprendida”	
2. El administrador selecciona la lección aprendida que desea eliminar. 4. Acepta el mensaje.	1. Muestra la interfaz, con todas las lecciones aprendidas que hay en la base de datos. 3. Muestra un mensaje de ratificación “Esta seguro que desea eliminar la lecciones aprendidas seleccionado” 5. El sistema eliminar la lecciones aprendidas seleccionado.
Flujos Alternos	
4. Cancela el mensaje.	

Prioridad	Crítico
------------------	---------

Caso de Uso	Evaluar Lecciones Aprendidas
Actor	Revisor (inicia)
Propósito	Permitir que el revisor apruebe y publique las lecciones aprendidas registradas por lo clientes
Resumen	El Caso de Uso se inicia cuando el revisor desea aprobar o publicar las lecciones aprendidas registradas, finaliza cuando las lecciones aprendidas son aprobada y publicadas, o son rechazadas.
Precondiciones	El usuario autenticado debe ser un revisor y además deben existir lecciones aprendidas registradas.
Poscondiciones	Quedan las lecciones aprendidas aprobadas y publicadas, o son rechazadas.
Referencia	R3
Descripción de las entradas de datos	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
<p>1. El revisor selecciona la opción de evaluar lecciones aprendidas.</p> <p>3. El revisor selecciona una de las opciones:</p> <p>a) solicita aprobar una lección aprendida, ver sección aprobar.</p> <p>a) solicita publicar una lección aprendida, ver sección publicar.</p>	<p>2. El sistema muestra las opciones relacionadas con la evaluación de lecciones aprendidas: aprobar y publicar las lecciones aprendidas.</p>
Sección "Aprobar Lección Aprendida"	
<p>2. El revisor selecciona la lección aprendida que desea aprobar.</p>	<p>1. El sistema muestra las lecciones aprendidas que han sido registradas.</p> <p>3. El sistema muestra la información actual de la lección</p>

4. El revisor la evalúa y aprueba la lección aprendida.	aprendida seleccionada. 4. El sistema guarda la lección aprendida como aprobada, para que luego pueda ser publicada.
Flujos Alternos	
4. El revisor cuando evalúa la lección aprendida, no esta correcta, la rechaza y le envía un correo al cliente.	
Sección “Publicar Lección Aprendida”	
2. El revisor selecciona la lección aprendida que desea publicar. 4. El revisor solicita la opción de publicar.	1. El sistema muestra las lecciones aprendidas que han sido aprobadas. 3. El sistema muestra la información actual de la lección aprendida seleccionada. 4. El sistema guarda la lección aprendida como publicada.
Prioridad	Crítico

Caso de Uso	Realizar Búsquedas Avanzadas
Actor	Usuario(inicia)
Propósito	Permitir que los usuarios, puedan hacer búsquedas, referentes a las lecciones aprendidas que han sido publicadas.
Resumen	El Caso de Uso se inicia cuando el usuario, selecciona una de las opciones de búsquedas, finaliza cuando el usuario obtiene el resultado de la búsqueda seleccionada.
Precondiciones	El usuario autenticado debe estar autorizado a utilizar el sistema.
Poscondiciones	El usuario obtiene la información que desea.
Referencia	R4
Descripción de las entradas de datos	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario selecciona la opción de búsqueda avanzada.	2. El sistema muestra las opciones relacionadas con las

<p>3. El usuario selecciona una de las opciones:</p> <p>a) solicita buscar lecciones aprendidas por palabras claves, ver sección Buscar por Palabras Claves.</p> <p>b) solicita buscar lecciones aprendidas por Área, ver sección Buscar por Área.</p> <p>c) solicita buscar lecciones aprendidas por Rol, ver sección Buscar por Rol.</p> <p>d) solicita buscar lecciones aprendidas por Proyecto, ver sección Buscar por Proyecto.</p>	<p>búsquedas avanzadas: Buscar por Palabras Claves, por Área, por Rol o por Proyecto.</p>
<p>Sección “Buscar por Palabras Claves”</p>	
<p>2. El usuario introduce las palabras claves.</p>	<p>1. El sistema muestra el formulario de datos necesarios: Palabras claves.</p> <p>3. El sistema verifica si existe alguna lección aprendida con esas palabras claves.</p> <p>4. El sistema muestra las lecciones aprendidas que hayan cumplido con la condición.</p>
<p>Flujos Alternos</p>	
	<p>3. Si el sistema verifica y no encuentra ninguna lección aprendida con esas palabras claves, manda un mensaje que no ha encontrado ninguna lección aprendida.</p>
<p>Sección “Buscar por Área”</p>	
<p>2. El usuario introduce el área de la que desea la lección aprendida.</p>	<p>1. El sistema muestra el formulario de datos necesarios: Área.</p> <p>3. El sistema verifica si existe alguna lección aprendida que pertenezca a esa área.</p> <p>4. El sistema muestra las lecciones aprendidas de esa área.</p>
<p>Flujos Alternos</p>	

	3. Si el sistema verifica y no encuentra ninguna lección aprendida que pertenezca a esa área, manda un mensaje que no ha encontrado ninguna lección aprendida.
Sección “Buscar por Rol”	
2. El usuario introduce el rol del que desea las lecciones aprendidas.	<p>1. El sistema muestra el formulario de datos necesarios: Rol.</p> <p>3. El sistema verifica si existe alguna lección aprendida referente a ese rol.</p> <p>4. El sistema muestra las lecciones aprendidas que hayan cumplido con la condición.</p>
Flujos Alternos	
	3. Si el sistema verifica y no encuentra ninguna lección aprendida referente a ese rol, manda un mensaje que no ha encontrado ninguna lección aprendida.
Sección “Buscar por Proyecto”	
2. El usuario introduce el proyecto del que desea la lección aprendida.	<p>1. El sistema muestra el formulario de datos necesarios: Proyecto.</p> <p>3. El sistema verifica si existe alguna lección aprendida que pertenezca a ese proyecto.</p> <p>4. El sistema muestra las lecciones aprendidas de ese proyecto.</p>
Flujos Alternos	
	3. Si el sistema verifica y no encuentra ninguna lección aprendida que pertenezca a ese proyecto, manda un mensaje que no ha encontrado ninguna lección aprendida.
Prioridad	

Caso de Uso	Mostrar Reportes
Actor	Usuario(inicia)
Propósito	Permitir que los usuarios, puedan obtener reportes, referentes a las lecciones aprendidas que han sido publicadas.

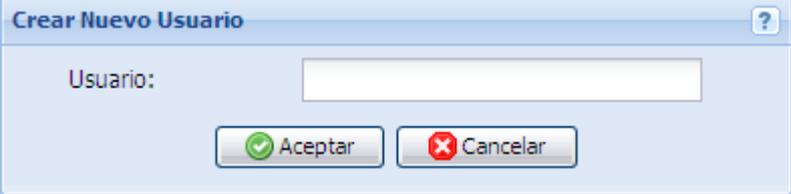
Resumen	El Caso de Uso se inicia cuando el usuario, selecciona una de las opciones de mostrar reporte, finaliza cuando el usuario obtiene el resultado del reporte seleccionado.
Precondiciones	El usuario autenticado debe estar autorizado a utilizar el sistema.
Poscondiciones	El usuario obtiene la información que desea.
Referencia	R5
Descripción de las entradas de datos	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
<p>1. El usuario selecciona la opción de mostrar reporte.</p> <p>3. El usuario selecciona una de las opciones:</p> <p>a) solicita lecciones aprendidas de un cliente que han sido publicada, ver sección Mostrar lecciones aprendidas de un cliente, que han sido publicadas.</p> <p>b) solicita lecciones aprendidas por tipo, ver sección Mostrar lección aprendida por tipo.</p> <p>c) solicita lecciones aprendidas que han sido publicada por un revisor, ver sección Mostrar lecciones aprendidas que han sido publicadas por un revisor</p>	<p>2. El sistema muestra las opciones relacionadas con los reportes: lecciones aprendidas de un cliente que han sido publicas, lecciones aprendidas por tipo, lecciones aprendidas que han sido publicas por un revisor.</p>
Sección “Mostrar lecciones aprendidas de un cliente, que han sido publicadas”	
<p>2. El usuario introduce el nombre del cliente que desea obtener las lecciones aprendidas.</p>	<p>1. El sistema muestra el formulario de datos necesarios: nombre de cliente.</p> <p>3. El sistema verifica si existe el cliente y si tiene lecciones aprendidas publicadas.</p>

	4. El sistema muestra las lecciones aprendidas de ese cliente que han sido publicadas.
Flujos Alternos	
	3. Si el sistema verifica y el cliente no existe, manda un mensaje, no existe cliente con ese nombre. También si no encuentra ninguna lección aprendida de este cliente que haya sido publicada, manda un mensaje que no ha encontrado ninguna lección aprendida.
Sección “Mostrar lección aprendida por tipo”	
2. El usuario el tipo de lección aprendida que desea obtener.	<p>1. El sistema muestra el formulario de datos necesarios: Tipo (negativa o positiva).</p> <p>3. El sistema verifica si existe el cliente y si tiene lecciones aprendidas publicadas.</p> <p>4. El sistema muestra las lecciones aprendidas que cumplieron con la condición.</p>
Flujos Alternos	
	3. Si el sistema verifica y no encuentra ninguna lección aprendida de este tipo, manda un mensaje que no ha encontrado ninguna lección aprendida.
Sección “Mostrar lecciones aprendidas que han sido publicadas por un revisor ”	
2. El usuario introduce el nombre del revisor del que desea las lecciones aprendidas.	<p>1. El sistema muestra el formulario de datos necesarios: nombre del revisor.</p> <p>3. El sistema verifica si existe alguna lección aprendida referente a ese rol.</p> <p>4. El sistema muestra las lecciones aprendidas publicadas por ese revisor.</p>
Flujos Alternos	
	3. Si el sistema verifica y el revisor no existe, manda un mensaje, no existe revisor con ese nombre. También si no encuentra ninguna lección aprendida publicada por este

	revisor, manda un mensaje que no ha encontrado ninguna lección aprendida.
Prioridad	

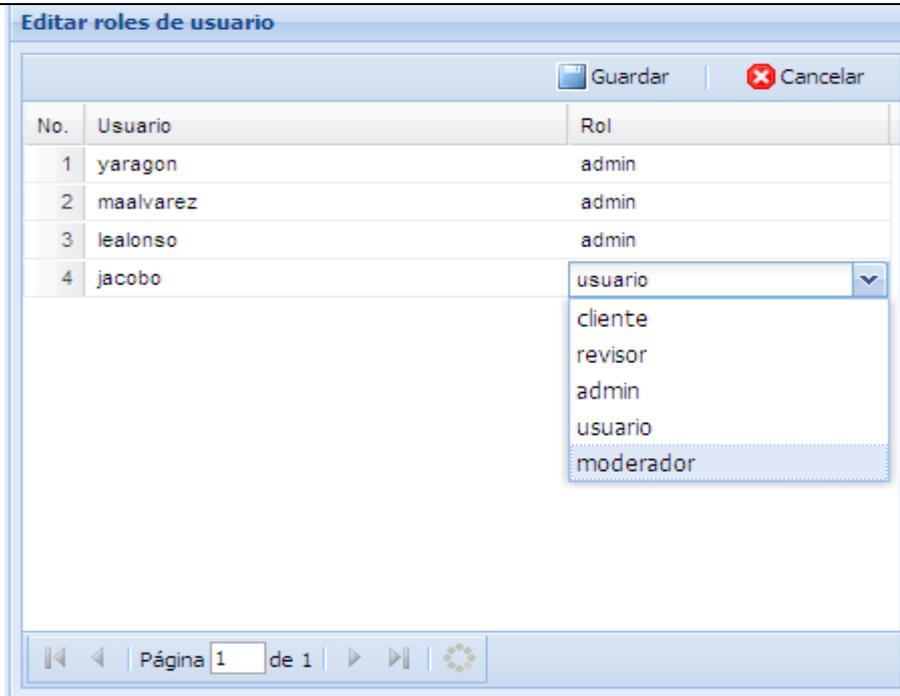
2.8.2 Casos de Usos del Sitio Administración

Caso de Uso	Gestionar Usuario
Actor	Administrador(inicia)
Propósito	Crear un nuevo usuario, editar roles del usuarios, Mostrar usuario por proyecto, eliminar un usuario de un proyecto y eliminar usuarios.
Resumen	<p>El caso de uso comienza cuando el Administrador va a realizar alguna de las siguientes operaciones:</p> <ul style="list-style-type: none"> • Crear nuevo usuario. • Editar datos de usuario. • Mostrar Usuario por Proyecto • Asignar Usuario a Proyecto • Eliminar un usuario de un proyecto. • Eliminar usuario. <p>El sistema le muestra la interfaz correspondiente según su solicitud y ejecuta las acciones necesarias. El caso de uso finaliza cuando se emite el resultado de la operación solicitada.</p>
Precondiciones	Que el Administrador se haya autenticado en el sistema.
Poscondiciones	El sistema crea un nuevo usuario, edita datos de usuario y elimina usuarios.
Referencia	R5
Descripción de las entradas de datos	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El Administrador quiere realizar alguna de las operaciones siguientes:	2. El sistema muestra las opciones de gestionar usuario , ya sea , crear nuevo usuario, editar roles del usuario o eliminar

<ul style="list-style-type: none"> • Si desea crear un nuevo usuario ir a la Sección "Crear Nuevo Usuario" • Si desea editar datos de un usuario ir a la Sección "Editar datos de usuario" • Si desea asignar usuario a proyecto ir a la Sección "Asignar Usuario a Proyecto" • Si desea mostrar usuario por proyecto ir a la Sección "Mostrar usuario por Proyecto" • Si desea eliminar un usuario de un proyecto ir a la Sección "Eliminar un usuario de un Proyecto" • Si desea eliminar un usuario ir a la Sección "Eliminar usuario" 	<p>usuario</p>
<p>Sección "Crear un usuario"</p>	
<p>Interfaz</p>	
	
<p>2. Provee los datos necesarios para crear un nuevo usuario:</p> <ul style="list-style-type: none"> • Usuario 	<ol style="list-style-type: none"> 1. Muestra la interfaz correspondiente para la creación de un nuevo usuario. 3. El sistema verifica que ningún campo obligatorio este vacío y que los datos a entrar sean correctos. 4. El usuario queda guardado en la Base Datos.
<p>Flujos Alternos</p>	
	<ol style="list-style-type: none"> 3. Si el sistema verifica que algún campo obligatorio esta vacío o el tipo de alguno de los datos a entrar son incorrectos emite un mensaje de error

Sección “Editar Datos de Usuario”

Interfaz



- 2. Asignas o modificas roles a los usuarios que desees.
- 3. Presiona el botón Guardar

- 1. Muestra la interfaz correspondiente a editar roles de usuario.
- 4. Quedan guardados en la base de datos los cambios realizados.

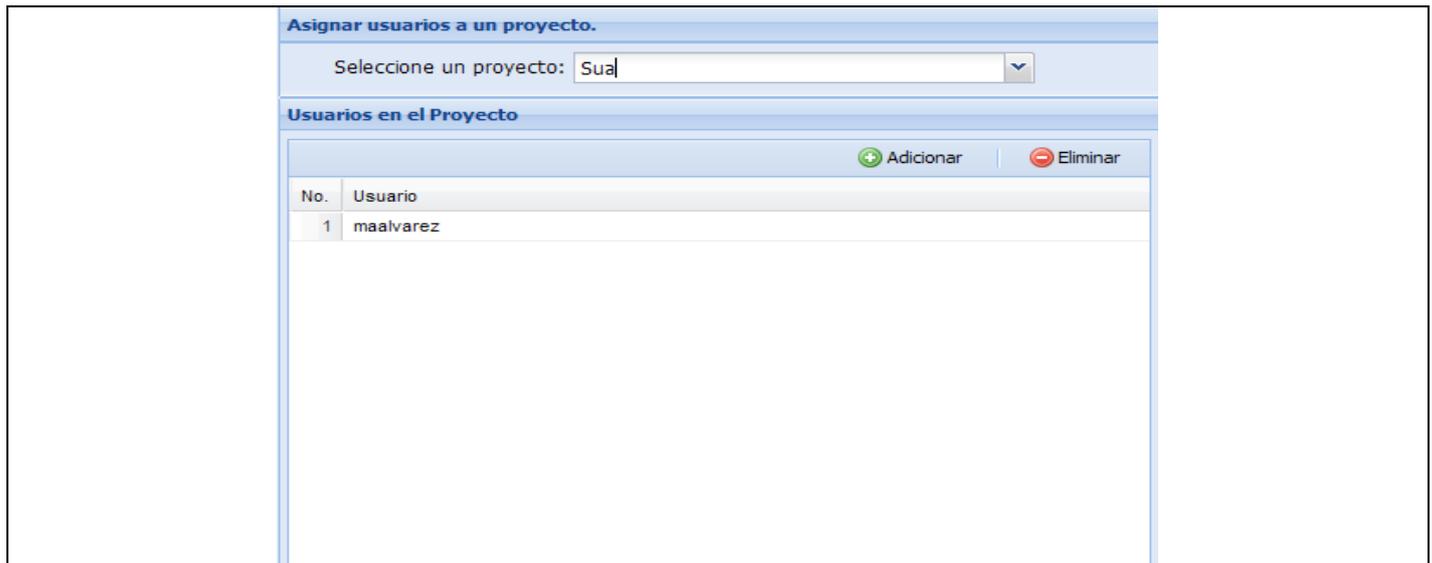
Flujos Alternos

- 3. Presiona el botón Cancelar

- 4. La información queda igual que como estaba en la Base de Datos.

Sección “Mostrar Usuario por Proyecto”

Interfaz



2. El administrador selecciona el proyecto que desea conocer los usuarios que están en dicho proyecto

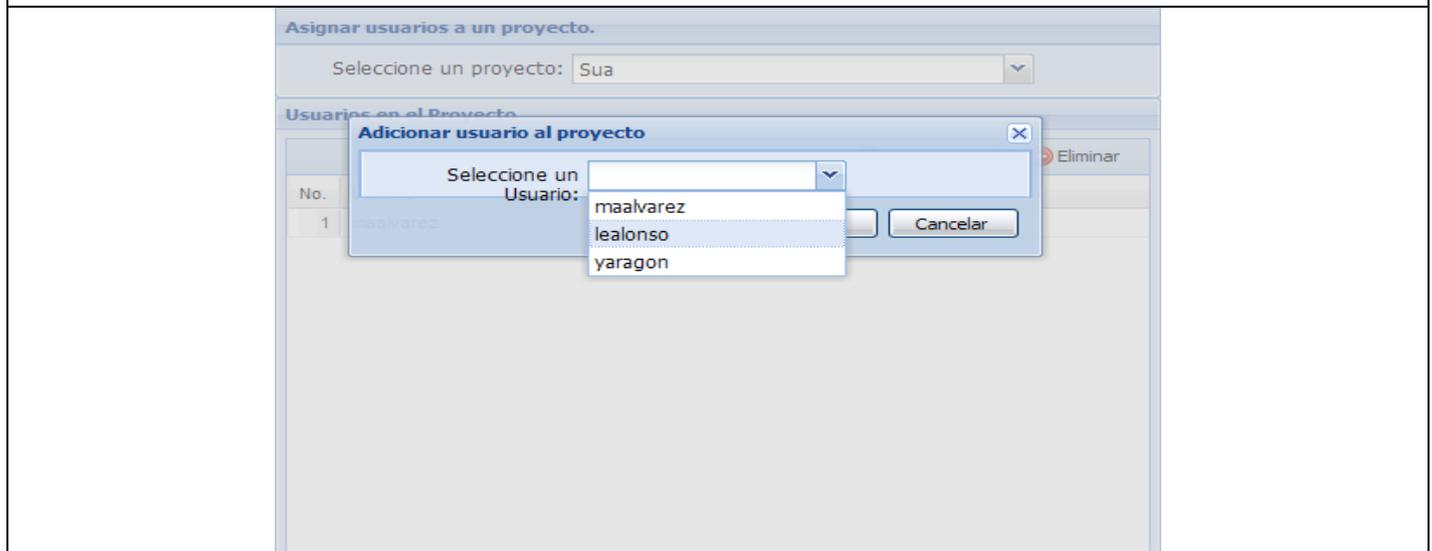
1. Se muestran todos los proyectos
3. Se muestran todos los usuarios que pertenecen al proyecto seleccionado.

Flujos Alternos

3. Si no hay usuarios almacenados, se muestra no hay usuarios en la base de datos

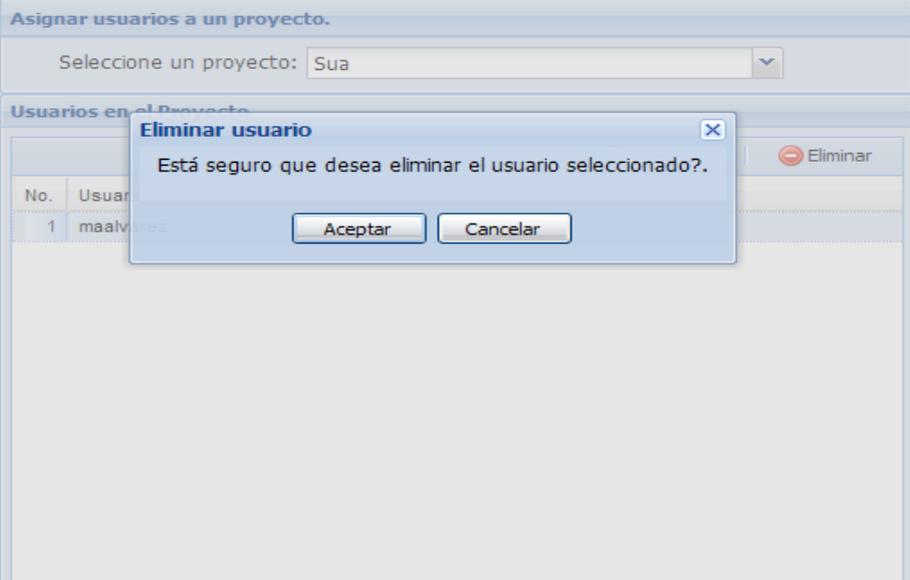
Sección "Asignar Usuario a Proyecto"

Interfaz



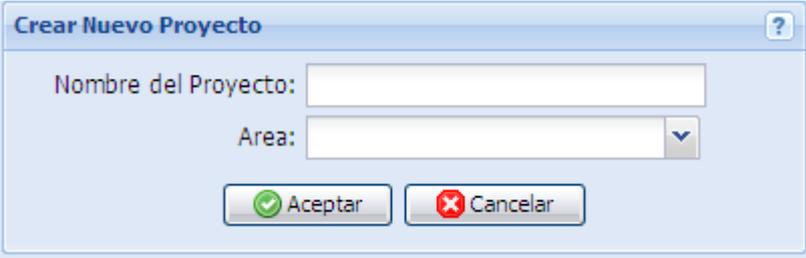
2. El administrador selecciona el proyecto que desea asignarle un nuevo usuario
3. Selecciona la opción de adicionar un nuevo

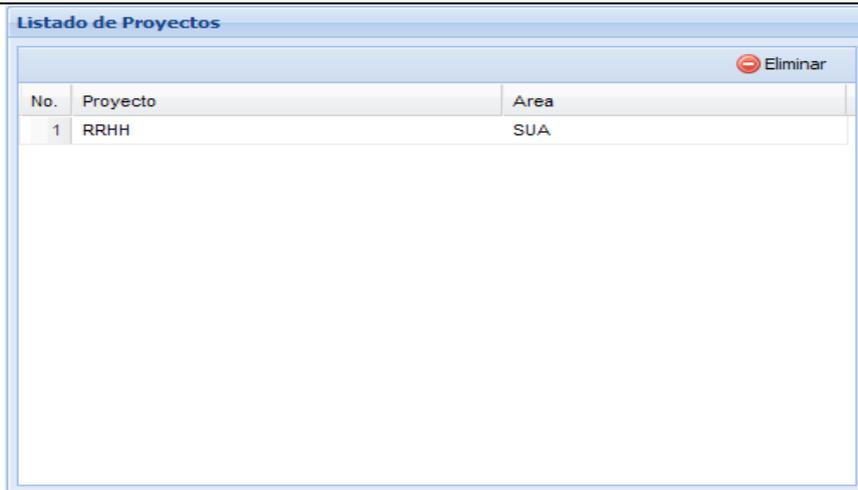
1. Se muestran todos los proyectos

<p>usuario al proyecto</p> <p>5. Selecciona el usuario que desea asignar al proyecto y acepta.</p>	<p>4. Se muestran todos los usuarios que hay almacenados en la base de datos</p> <p>6. Se muestra como un usuario del proyecto.</p>
<p>Flujos Alternos</p>	
<p>5. Si da cancelar, la información en la base de datos se mantiene igual</p>	
<p>Sección “Eliminar un Usuario de un Proyecto”</p>	
<p>Interfaz</p>	
	
<p>2. El administrador selecciona el proyecto del que desea eliminar un usuario</p> <p>4. Selecciona el usuario que desea eliminar</p> <p>6. Acepta el mensaje.</p>	<p>1. Se muestran todos los proyectos</p> <p>3. Se muestran todos los usuarios que hay almacenados en la base de datos que pertenecen a dicho proyecto</p> <p>5. Muestra un mensaje de ratificación “Esta seguro que desea eliminar el usuario”</p> <p>5. El sistema elimina el usuario seleccionado.</p>
<p>Flujos Alternos</p>	
<p>4. Si el administrador no selecciona el usuario que desea eliminar, se muestra un mensaje de error.</p> <p>6. Cancela el mensaje.</p>	

Sección “Eliminar Usuario”	
Interfaz	
2. El administrador selecciona el usuario que desea borrar.	1. Muestra la interfaz, con todos los usuarios que hay en la base de datos.
4. Acepta el mensaje.	3. Muestra un mensaje de ratificación “Esta seguro que desea eliminar el usuario”
	5. El sistema elimina el usuario seleccionado.
Flujos Alternos	
2. Si el administrador no selecciona el usuario que desea eliminar, se muestra un mensaje de error.	
4. Cancela el mensaje.	

Caso de Uso	Gestionar Proyecto
Actor	Administrador (inicia)
Propósito	Adicionar, listar o eliminar un proyecto.
Resumen	El Caso de Uso se inicia cuando el administrador selecciona la opción de adicionar, listar o eliminar un proyecto y finaliza cuando la información queda registrada en la base de datos.
Precondiciones	El usuario autenticado debe ser administrador
Poscondiciones	La información relacionada con los proyectos queda registrada en la Base Datos.
Referencia	R2
Descripción de las entradas de datos	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El administrador selecciona la opción de Gestionar proyecto.	2. El sistema muestra las opciones de gestionar proyecto,

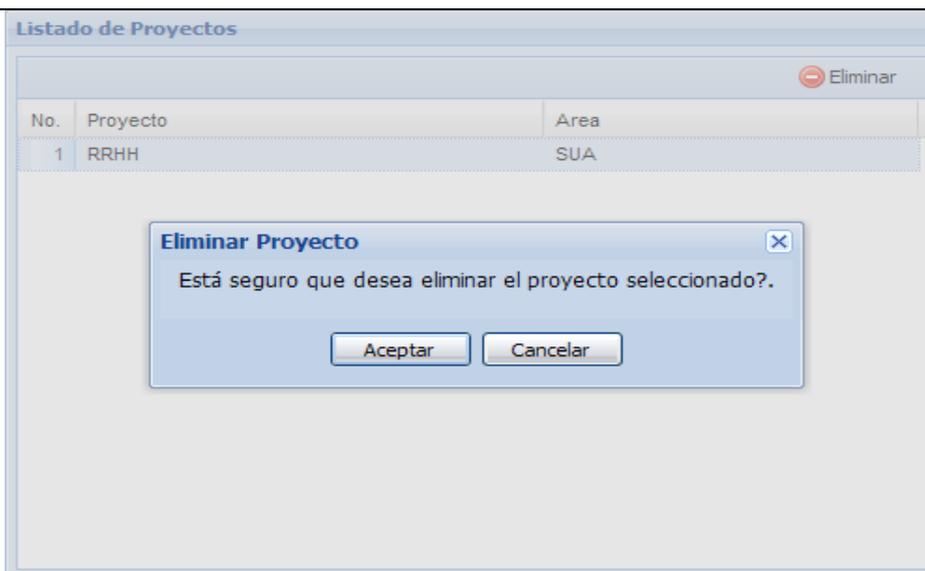
<p>3. El usuario selecciona una de las opciones:</p> <p>a) solicita adicionar un proyecto, ver sección Adicionar.</p> <p>b) solicita listar un proyecto, ver sección Listado.</p> <p>c) solicita eliminar un proyecto, ver sección Eliminar.</p>	<p>adicionar, listar, eliminar un proyecto.</p>
<p>Sección "Adicionar proyecto"</p>	
<p>Interfaz</p>	
	
<p>2. El administrador llena los campos y selecciona Aceptar</p>	<p>1. El sistema muestra el formulario necesario:</p> <ul style="list-style-type: none"> • Nombre del Proyecto • Área. <p>3. El sistema verifica que ningún campo obligatorio este vacío y que los datos a entrar sean correctos.</p> <p>4. El proyecto queda guardado en la Base Datos.</p>
<p>Flujos Alternos</p>	
	<p>3. Si el sistema verifica que algún campo obligatorio esta vacío o el tipo de alguno de los datos a entrar son incorrectos emite un mensaje de error</p>
<p>Sección "Listar Proyectos"</p>	
<p>Interfaz</p>	



2. El administrador obtiene la información deseada.	1. El sistema muestra el Listado de Proyectos
---	---

Sección "Eliminar Proyecto"

Interfaz



2. El administrador selecciona el proyecto que desea eliminar.	1. Muestra la interfaz, con todos los proyectos que hay en la base de datos.
4. Acepta el mensaje.	3. Muestra un mensaje de ratificación "Esta seguro que desea eliminar el proyecto seleccionado"
	5. El sistema eliminar el proyecto seleccionado.

Flujos Alternos	
4. Cancela el mensaje.	

Caso de Uso	Gestionar Área
Actor	Administrador (inicia)
Propósito	Adicionar, listar o eliminar área.
Resumen	El Caso de Uso se inicia cuando el administrador selecciona una de las opciones, ya sea adicionar, listar o eliminar un área y finaliza cuando la información queda registrada en la base de datos.
Precondiciones	El usuario autenticado debe ser administrador
Poscondiciones	La información relacionada con las áreas queda registrada en la Base Datos.
Referencia	R3

Descripción de las entradas de datos

Flujo Normal de Eventos

Acción del Actor	Respuesta del Sistema
1. El administrador selecciona la opción de Gestionar Área. 3. El usuario selecciona una de las opciones: a) solicita adicionar un área, ver sección Adicionar. b) solicita listar las áreas, ver sección Listado. c) solicita eliminar área, ver sección Eliminar.	2. El sistema muestra las opciones de gestionar área: adicionar, listar o eliminar área.

Sección "Adicionar Área"

Interfaz

1. El sistema muestra el formulario necesario:

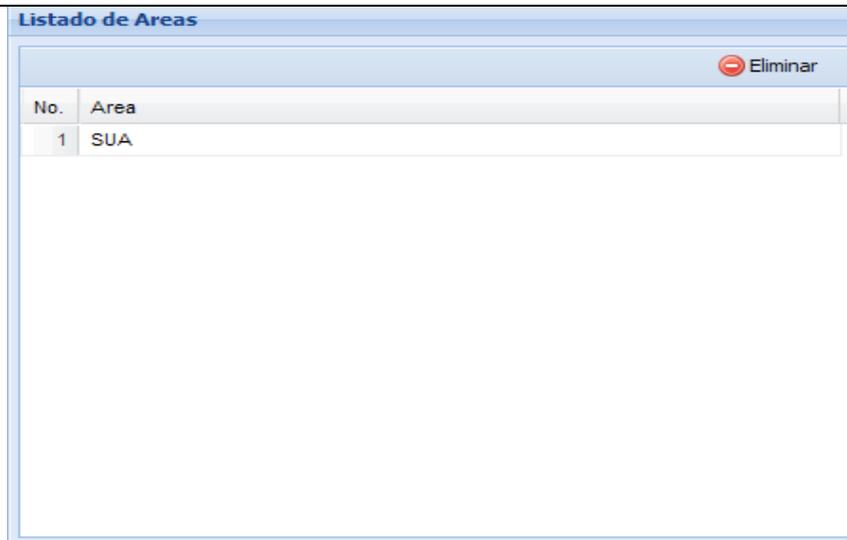
<p>2. El administrador llena los campos y selecciona Aceptar</p>	<ul style="list-style-type: none"> Nombre del Área. <p>3. El sistema verifica que ningún campo obligatorio este vacío y que los datos a entrar sean correctos.</p> <p>4. El área queda guardado en la Base Datos.</p>
--	--

Flujos Alternos

	<p>3. Si el sistema verifica que algún campo obligatorio esta vacío o el tipo de alguno de los datos a entrar son incorrectos emite un mensaje de error</p>
--	---

Sección "Listar Área"

Interfaz



<p>2. El administrador obtiene la información deseada.</p>	<p>1. El sistema muestra el Listado de Proyectos</p>
--	--

Sección "Eliminar Área"

Interfaz

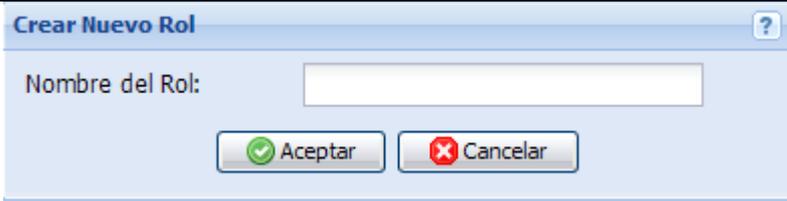


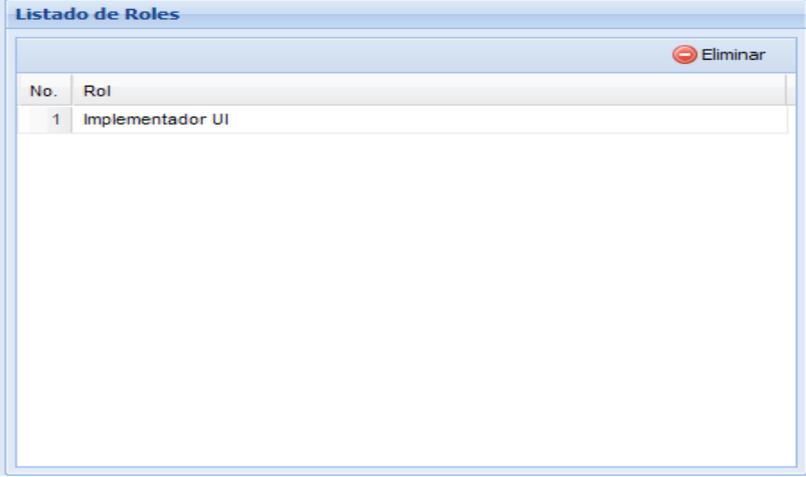
<p>2. El administrador selecciona el área que desea eliminar.</p> <p>4. Acepta el mensaje.</p>	<p>1. Muestra la interfaz, con todas las áreas que hay en la base de datos.</p> <p>3. Muestra un mensaje de ratificación “Esta seguro que desea eliminar el área seleccionado”</p> <p>5. El sistema eliminar el proyecto seleccionado.</p>
--	--

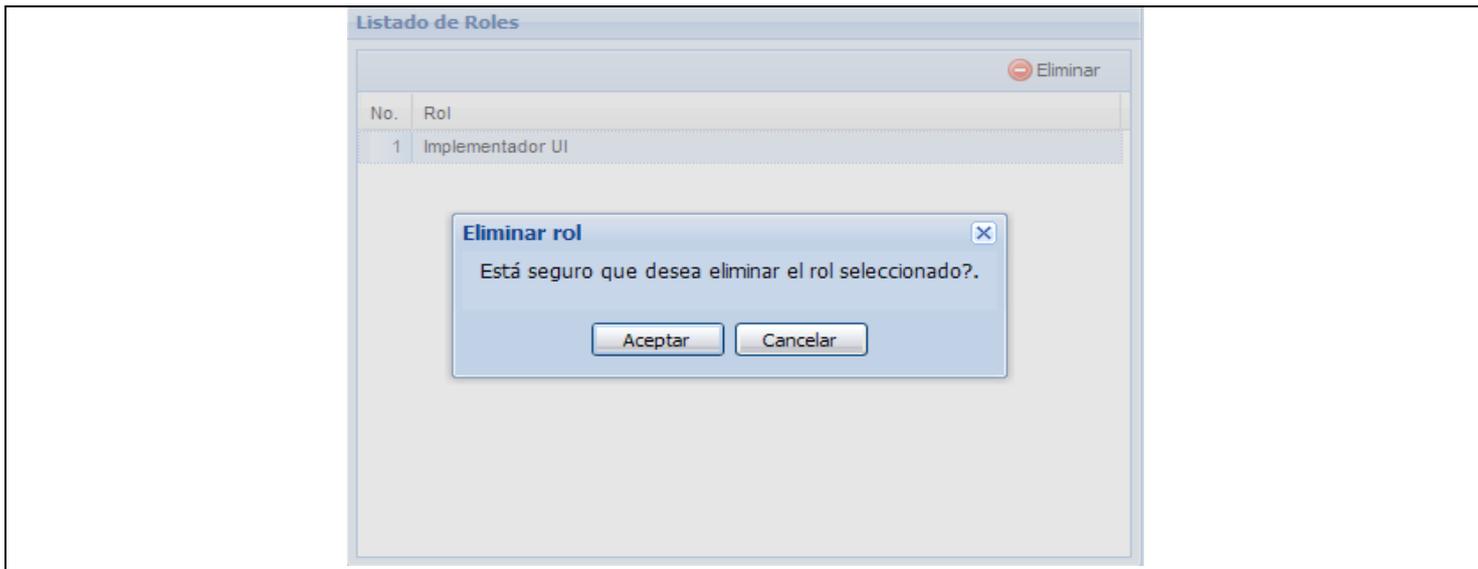
Flujos Alternos

<p>4. Cancela el mensaje.</p>	
<p>Prioridad</p>	<p>Critico</p>

<p>Caso de Uso</p>	<p>Gestionar Roles de Proyecto</p>
<p>Actor</p>	<p>Administrador (inicia)</p>
<p>Propósito</p>	<p>Adicionar, listar, eliminar, asignar roles a un proyecto, mostrar rol por proyecto, eliminar un rol de un proyecto.</p>
<p>Resumen</p>	<p>El Caso de Uso se inicia cuando el administrador selecciona una de las opciones ya sea adicionar rol,</p>

	asignar rol a un proyecto, listar rol, mostrar rol por proyecto, eliminar un rol de un proyecto o eliminar un rol y finaliza cuando la información queda registrada en la base de datos.
Precondiciones	El usuario autenticado debe ser administrador
Poscondiciones	La información sobre el rol queda actualizada en la base de datos.
Referencia	R2
Descripción de las entradas de datos	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
<p>1. El administrador selecciona la opción de Gestionar proyecto.</p> <p>3. El usuario selecciona una de las opciones:</p> <p>a) solicita adicionar un rol, ver sección Adicionar.</p> <p>b) solicita listar un rol, ver sección Listado de Roles.</p> <p>c) solicita eliminarr un rol, ver sección Eliminar.</p> <p>d) solicita asignar roles a un proyecto, ver sección Asignar Rol a Proyecto.</p> <p>e) solicita mostrar roles por proyecto, ver Mostrar Rol por Proyecto.</p> <p>f) solicita eliminar un rol de un proyecto, ver sección Eliminar un Rol de un Proyecto.</p>	<p>2. El sistema muestra las opciones de gestionar rol: adicionar, listar, eliminar o asignar un rol a un proyecto.</p>
Sección "Adicionar rol"	
Interfaz	
	
	1. El sistema muestra el formulario necesario:

<p>3. El administrador llena los campos y selecciona Aceptar.</p>	<ul style="list-style-type: none"> Nombre del Rol <p>4. El sistema verifica que ningún campo obligatorio este vacío y que los datos a entrar sean correctos.</p> <p>5. El rol queda guardado en la Base Datos.</p>
<p>Flujos Alternos</p>	
	<p>4. Si el sistema verifica que algún campo obligatorio esta vacío o el tipo de alguno de los datos a entrar son incorrectos emite un mensaje de error</p>
<p>Sección "Listar roles"</p>	
<p>Interfaz</p>	
	
<p>2. El administrador obtiene la información deseada.</p>	<p>1. El sistema muestra el Listado de Roles</p>
<p>Sección "Eliminar rol"</p>	
<p>Interfaz</p>	



2. El administrador selecciona el área que desea eliminar.

4. Acepta el mensaje.

1. Muestra la interfaz, con todos los roles que hay en la base de datos.

3. Muestra un mensaje de ratificación "Esta seguro que desea eliminar el área seleccionado"

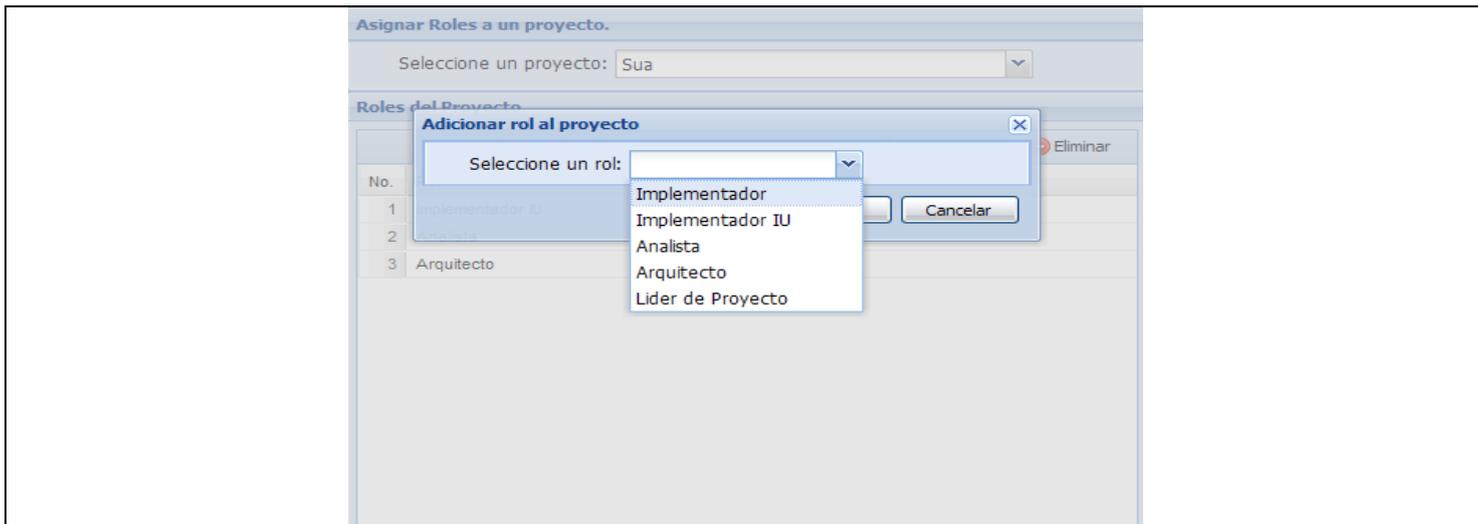
5. El sistema eliminar el proyecto seleccionado.

Flujos Alternos

4. Cancela el mensaje.

Sección "Asignar rol a proyecto"

Interfaz



- 2. El administrador selecciona el proyecto que desea asignarle un nuevo rol
- 3. Selecciona la opción de adicionar un nuevo rol al proyecto
- 5. Selecciona el rol que desea asignar al proyecto y acepta.

1. Se muestran todos los proyectos

4. Se muestran todos los roles que hay almacenados en la base de datos

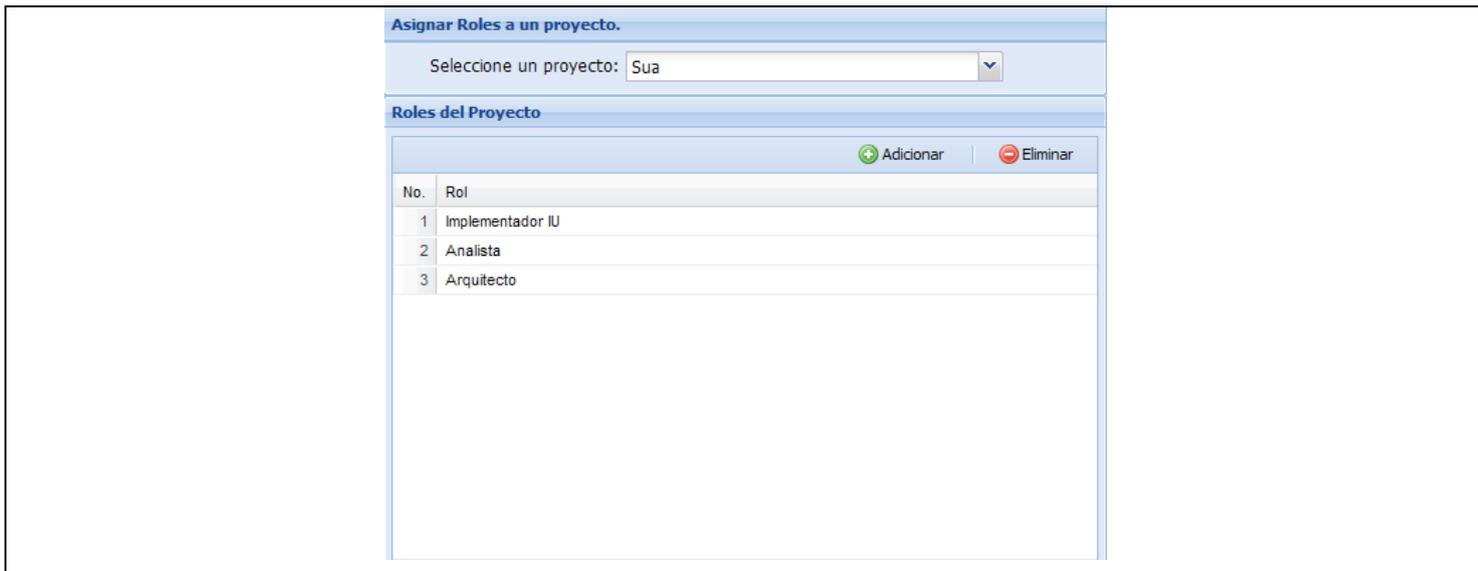
6. Se muestra como un rol del proyecto.

Flujos Alternos

- 5. Si da cancelar, la información en la base de datos se mantiene igual.

Sección "Mostrar Rol por proyecto"

Interfaz



1. Se muestran todos los proyectos

3. Se muestran todos los usuarios que pertenecen al proyecto seleccionado.

2. El administrador selecciona el proyecto que desea conocer los roles que están en dicho proyecto

1. Se muestran todos los proyectos

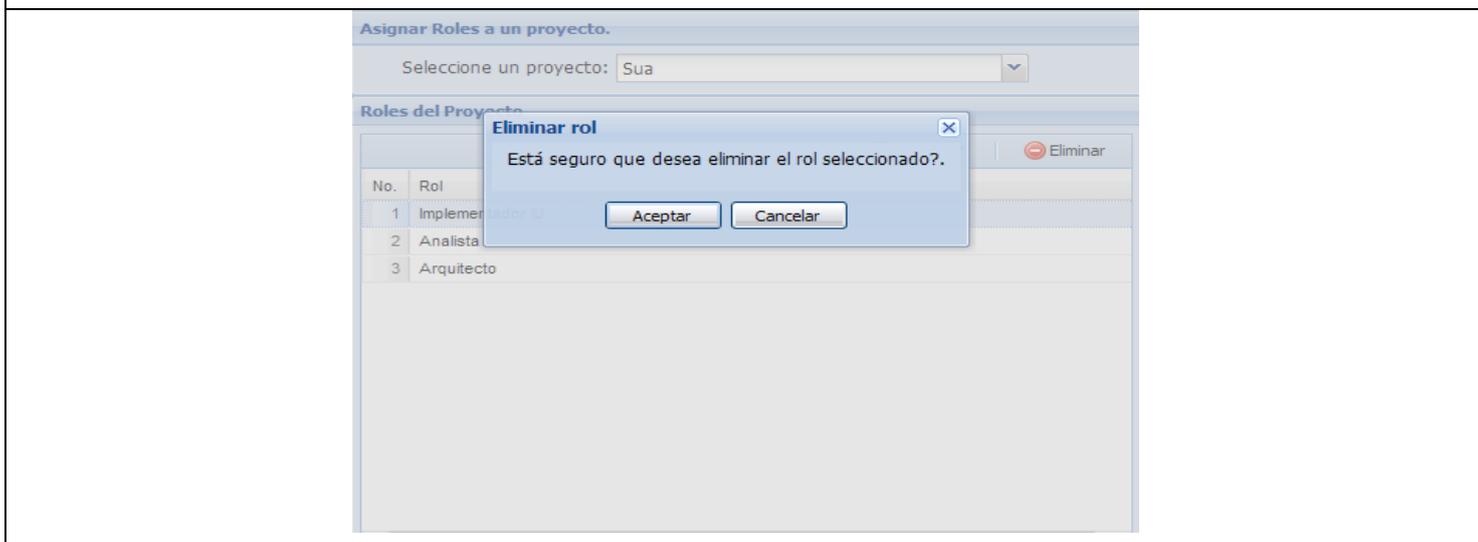
3. Se muestran todos los roles que pertenecen al proyecto seleccionado.

Flujos Alternos

3. Si no hay roles almacenados, se muestra no hay roles en la base de datos

Sección "Eliminar un Rol de un proyecto"

Interfaz

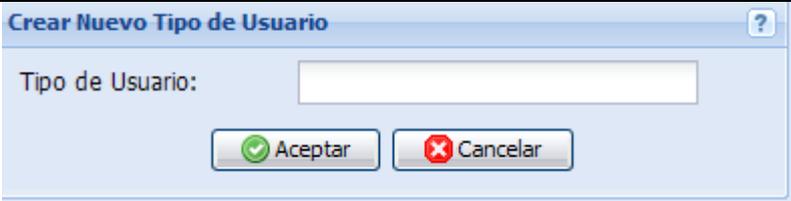


<p>1. Se muestran todos los proyectos</p> <p>3..Se muestran todos los usuarios que hay almacenados en la base de datos que pertenecen a dicho proyecto</p> <p>5. Muestra un mensaje de ratificación “Esta seguro que desea eliminar el usuario”</p> <p>5. El sistema elimina el usuario seleccionado.</p>	<p>2. El administrador selecciona el poyecto del que desea eliminar un rol</p> <p>4.Selecciona el rol que desea elminar</p> <p>6. Acepta el mensaje.</p>	<p>1. Se muestran todos los proyectos</p> <p>3..Se muestran todos los roles que hay almacenados en la base de datos que pertenecen a dicho proyecto</p> <p>5. Muestra un mensaje de ratificación “Esta seguro que desea eliminar el rol”</p> <p>5. El sistema elimina el rol seleccionado.</p>
---	--	--

Flujos Alternos

<p>4. Si el administrador no selecciona el usuario que desea elminar, se muestra un mensaje de error.</p> <p>6. Cancela el mensaje.</p>	
<p>Prioridad</p>	<p>Critico</p>

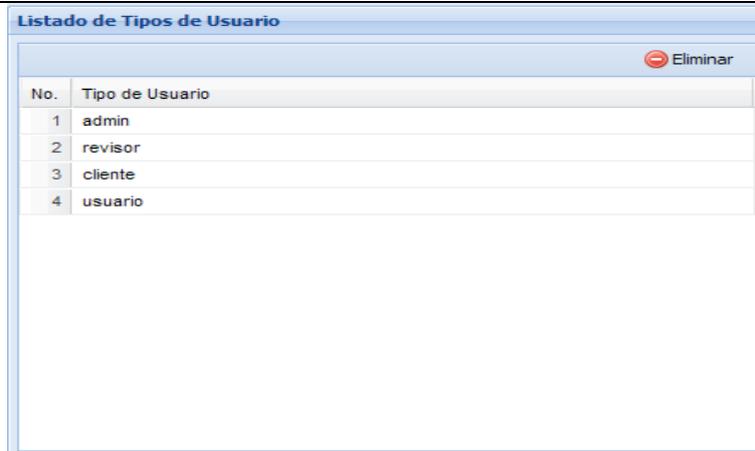
<p>Caso de Uso</p>	<p>Gestionar Tipo de Usuario</p>
<p>Actor</p>	<p>Administrador (inicia)</p>
<p>Propósito</p>	<p>Adicionar, listar o eliminar tipo de usuario.</p>
<p>Resumen</p>	<p>El Caso de Uso se inicia cuando el administrador selecciona la opción de adicionar, listar o eliminar un tipo de usuario y finaliza cuando la información queda registrada en la base de datos.</p>
<p>Precondiciones</p>	<p>El usuario autenticado debe ser administrador</p>
<p>Poscondiciones</p>	<p>La información relacionada con los tipos de usuarios queda registrada en la Base Datos.</p>
<p>Referencia</p>	<p>R2</p>

Descripción de las entradas de datos	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
<p>1. El administrador selecciona la opción de Gestionar Tipo de Usuario.</p> <p>3. El usuario selecciona una de las opciones:</p> <p>a) solicita adicionar un tipo de usuario, ver sección Adicionar.</p> <p>b) solicita listar un tipo de usuario, ver sección Listado.</p> <p>c) solicita eliminarr un tipo de usuario, ver sección Eliminar.</p>	<p>2. El sistema muestra las opciones de gestionar tipo de usuario: adicionar, listar, eliminar un tipo de usuario.</p>
Sección "Adicionar Tipo de usuario"	
Interfaz	
	
<p>2. El administrador llena los campos y selecciona Aceptar</p>	<p>1. El sistema muestra el formulario necesario:</p> <ul style="list-style-type: none"> Tipo de Usuario <p>3. El sistema verifica que ningún campo obligatorio este vacío y que los datos a entrar sean correctos.</p> <p>4. El tipo de usuario queda guardado en la Base Datos.</p>
Flujos Alternos	
	<p>3. Si el sistema verifica que algún campo obligatorio esta vacío o el tipo de alguno de los datos a entrar son incorrectos</p>

emite un mensaje de error

Sección "Listar Tipo de Usuario"

Interfaz

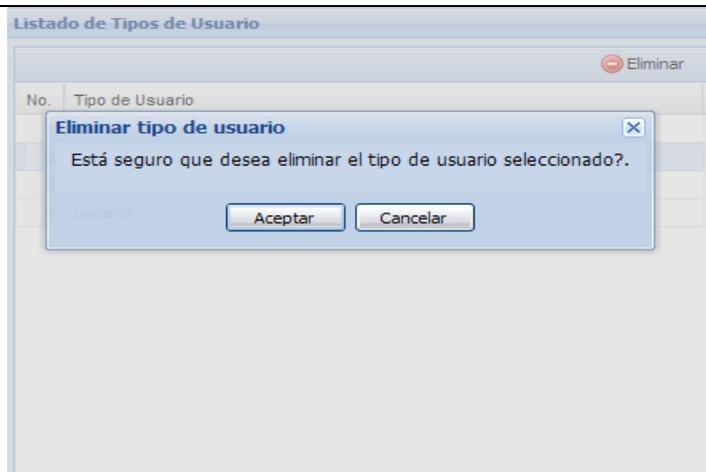


2. El administrador obtiene la información deseada.

1. El sistema muestra el Listado de Tipos de Usuario.

Sección "Eliminar Tipo de Usuario"

Interfaz



2. El administrador selecciona el tipo de usuario que desea eliminar.

1. Muestra la interfaz, con todos los tipos de usuario que hay en la base de datos.

3. Muestra un mensaje de ratificación "Esta seguro que desea eliminar el tipo de usuario seleccionado"

4. Acepta el mensaje.	5. El sistema eliminar el tipo de usuario seleccionado.
Flujos Alternos	
4. Cancela el mensaje.	

Anexo 2 Diagramas de clases del análisis de los casos que conforman la solución propuesta.



Figura 2.1 Diagrama de clases del análisis del CU Autenticar Usuario

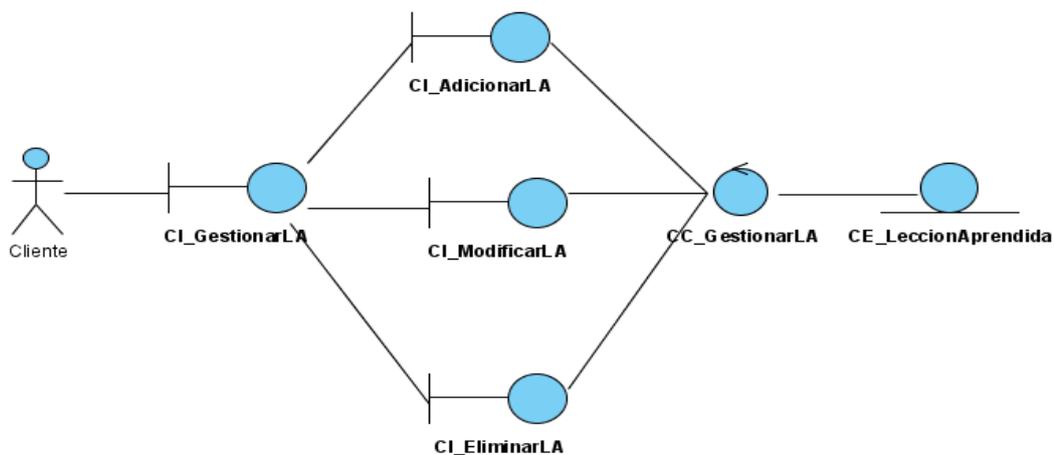


Figura 2.2 Diagrama de clases del análisis del CU Gestión Lección Aprendida

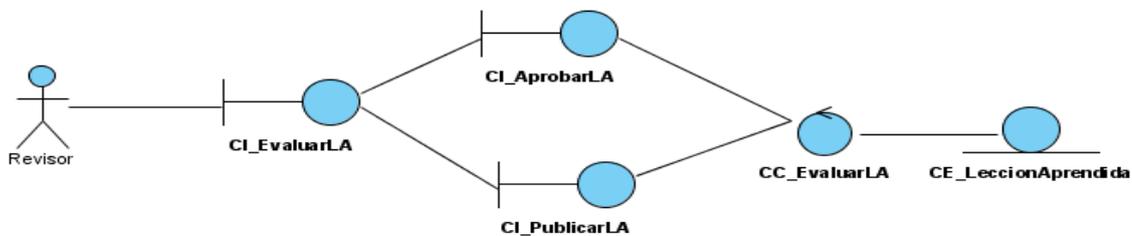


Figura 2.3 Diagrama de clases del análisis del CU Evaluar Lección Aprendida

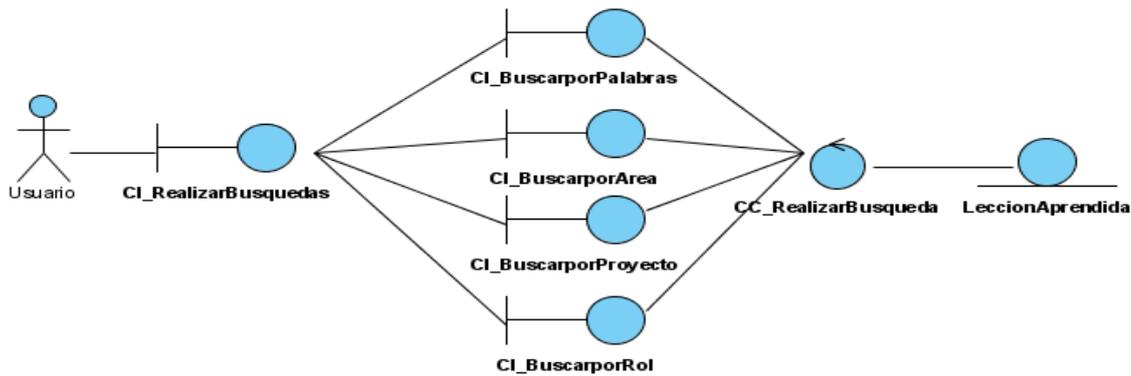


Figura 2.4 Diagrama de clases del análisis del CU Realizar Búsquedas

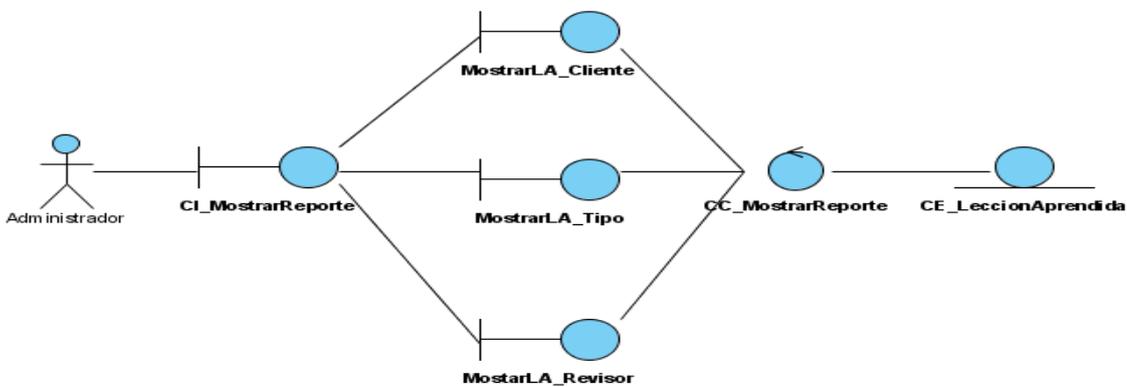


Figura 2.5 Diagrama de clases del análisis del CU Motrar Reporte

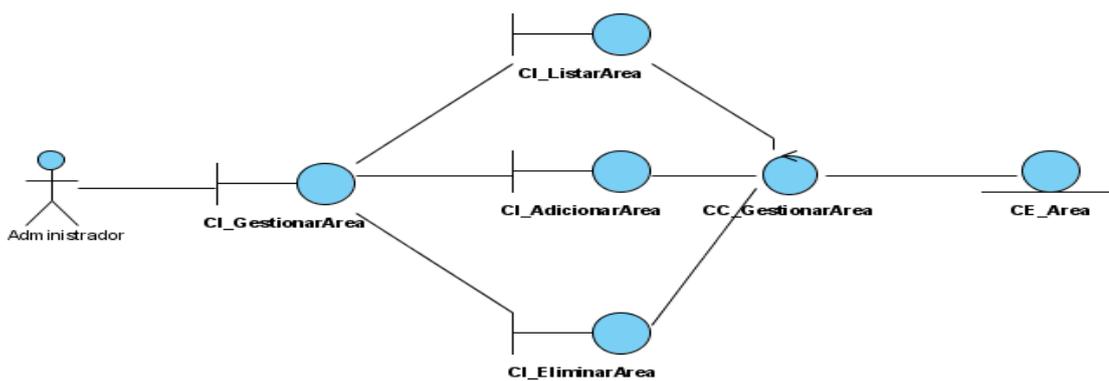
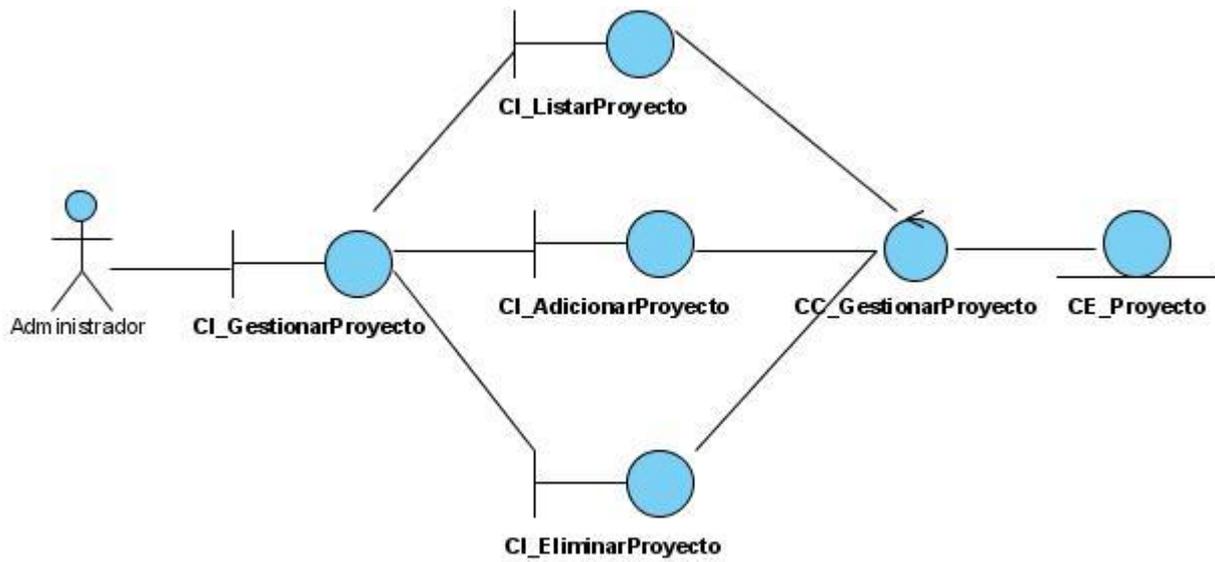


Figura 2.6 Diagrama de clases del análisis del CU Gestionar Área



V

Figura 2.7 Diagrama de clases del análisis del CU Gestionar Proyecto

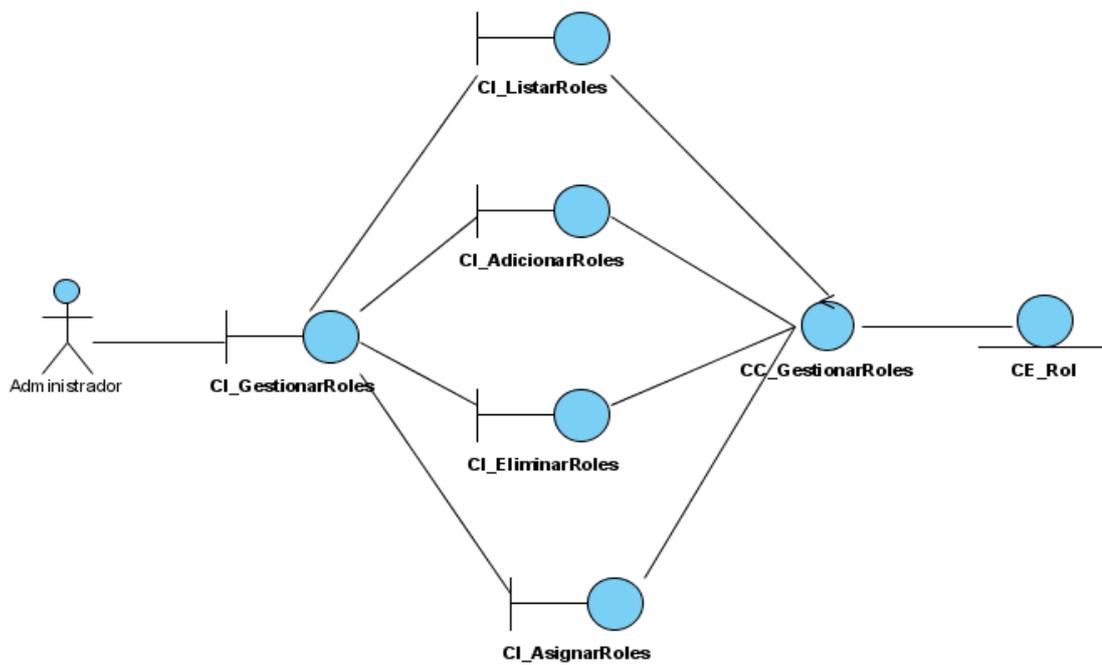


Figura 2.8 Diagrama de clases del análisis del CU Gestionar Roles.

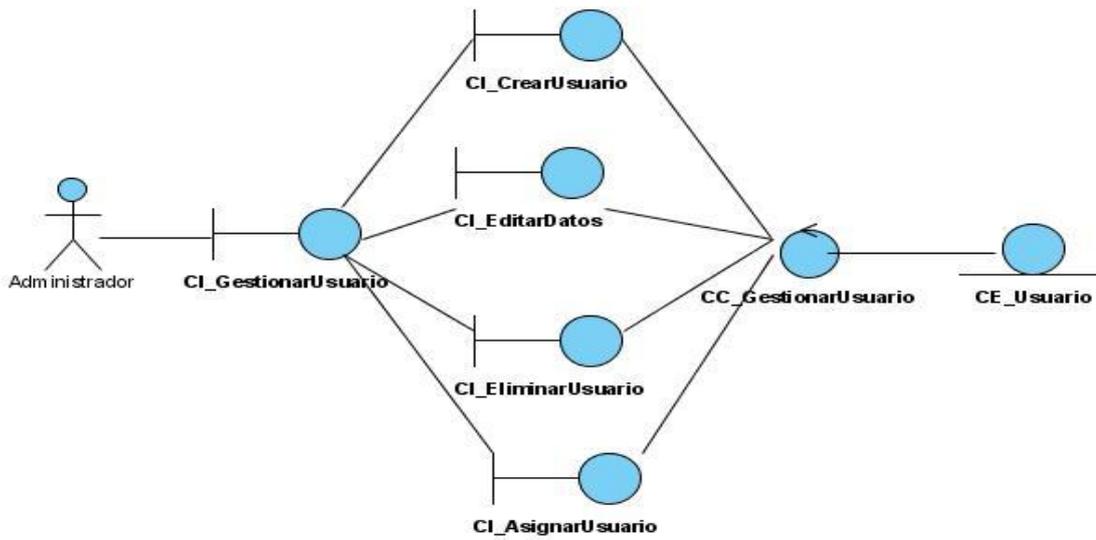


Figura 2.9 Diagrama de clases del análisis del CU Gestionar Usuario

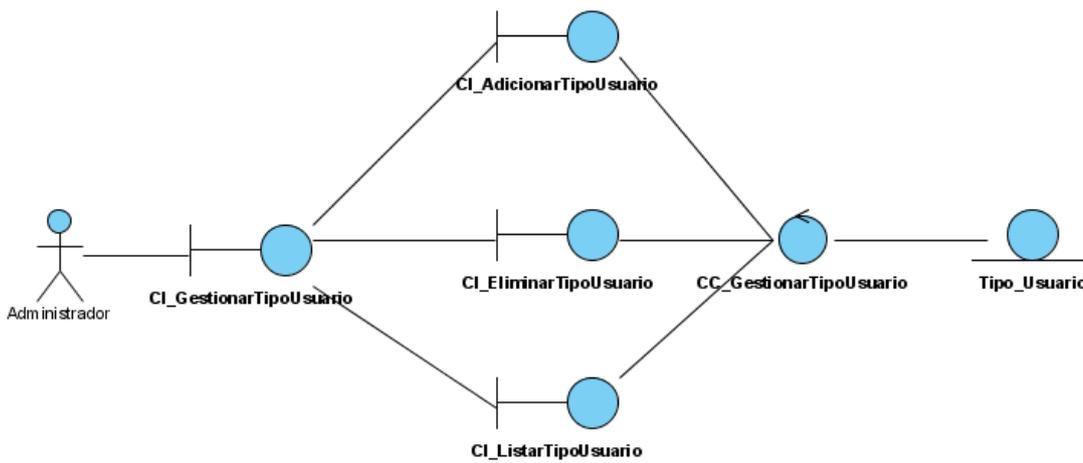


Figura 2.10 Diagrama de clases del análisis del CU Gestionar Tipo Usuario

Anexo 3 Diagramas de clases del diseño del sistema a implementar.

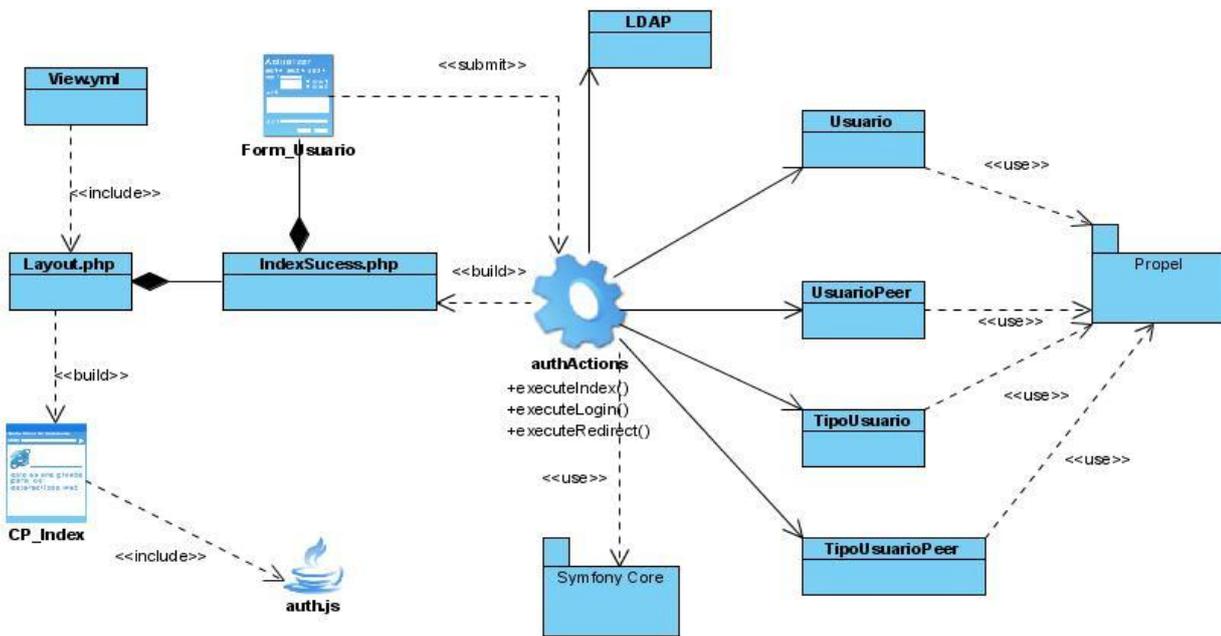


Figura 3.1 Diagrama de Diseño Web del CU Autenticar Usuario.

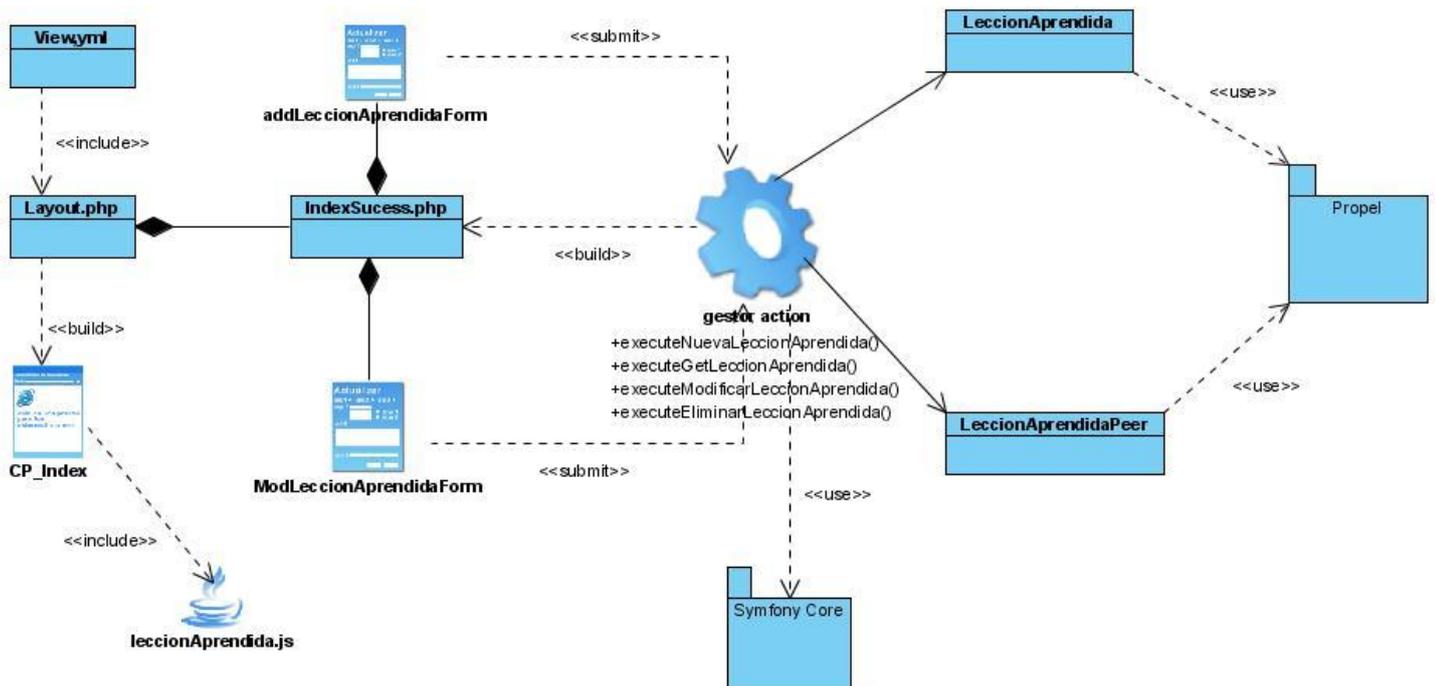


Figura 3.2 Diagrama de Diseño Web del CU Gestionar Lección Aprendida.

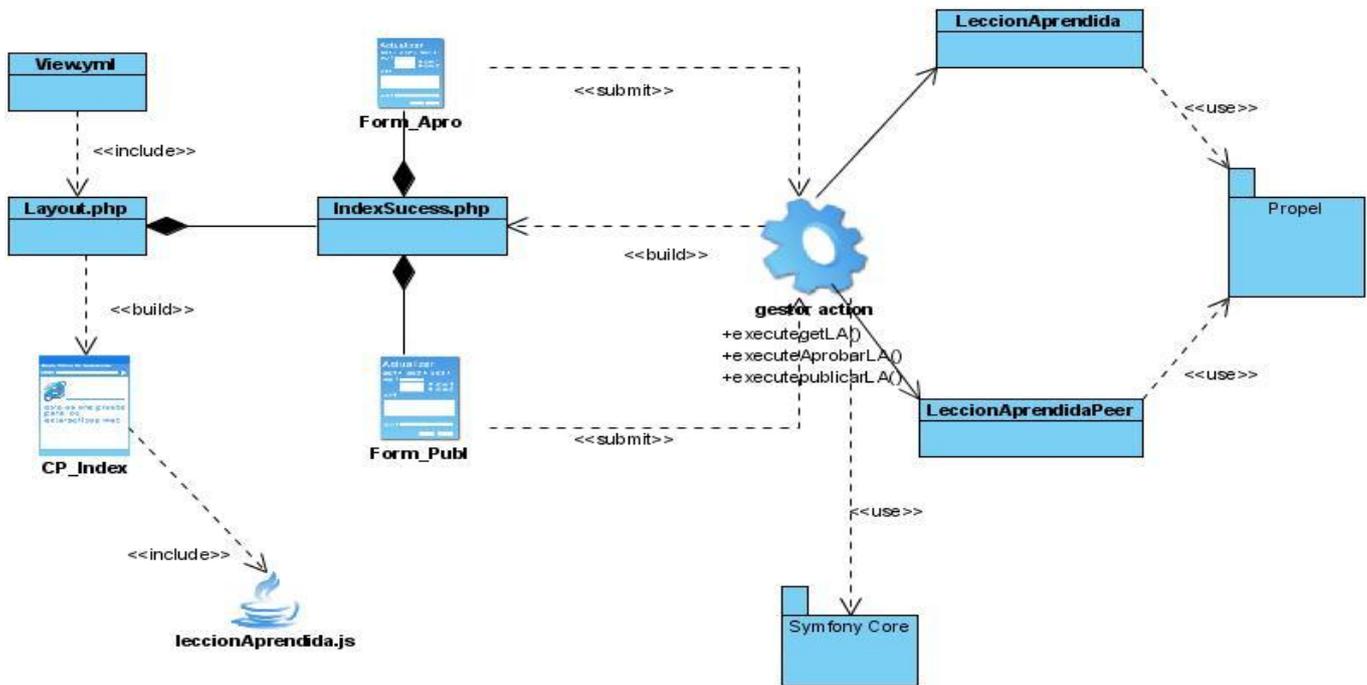


Figura 3.3 Diagrama de Diseño Web del CU Evaluar Lección Aprendida.

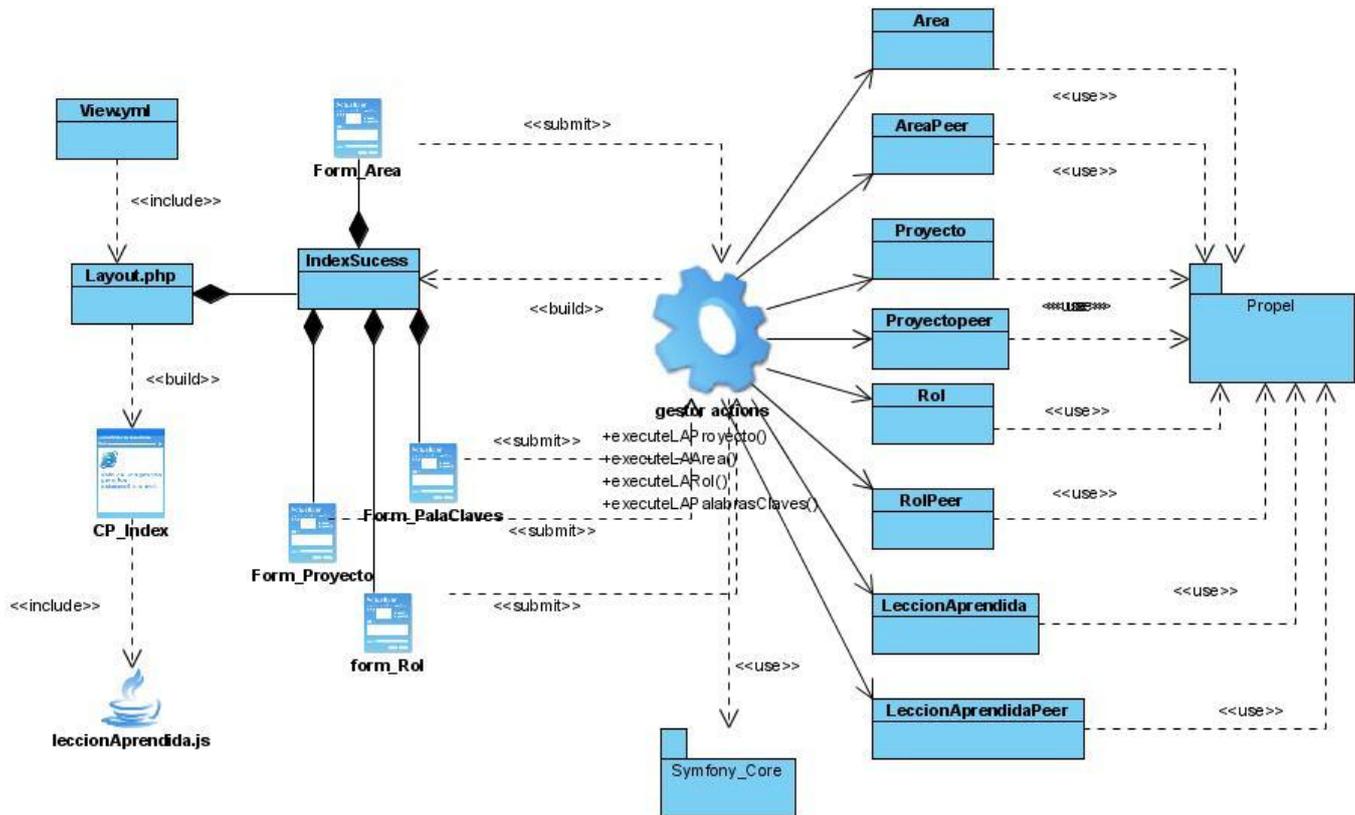


Figura 3.4 Diagrama de Diseño Web del CU Realizar Búsquedas

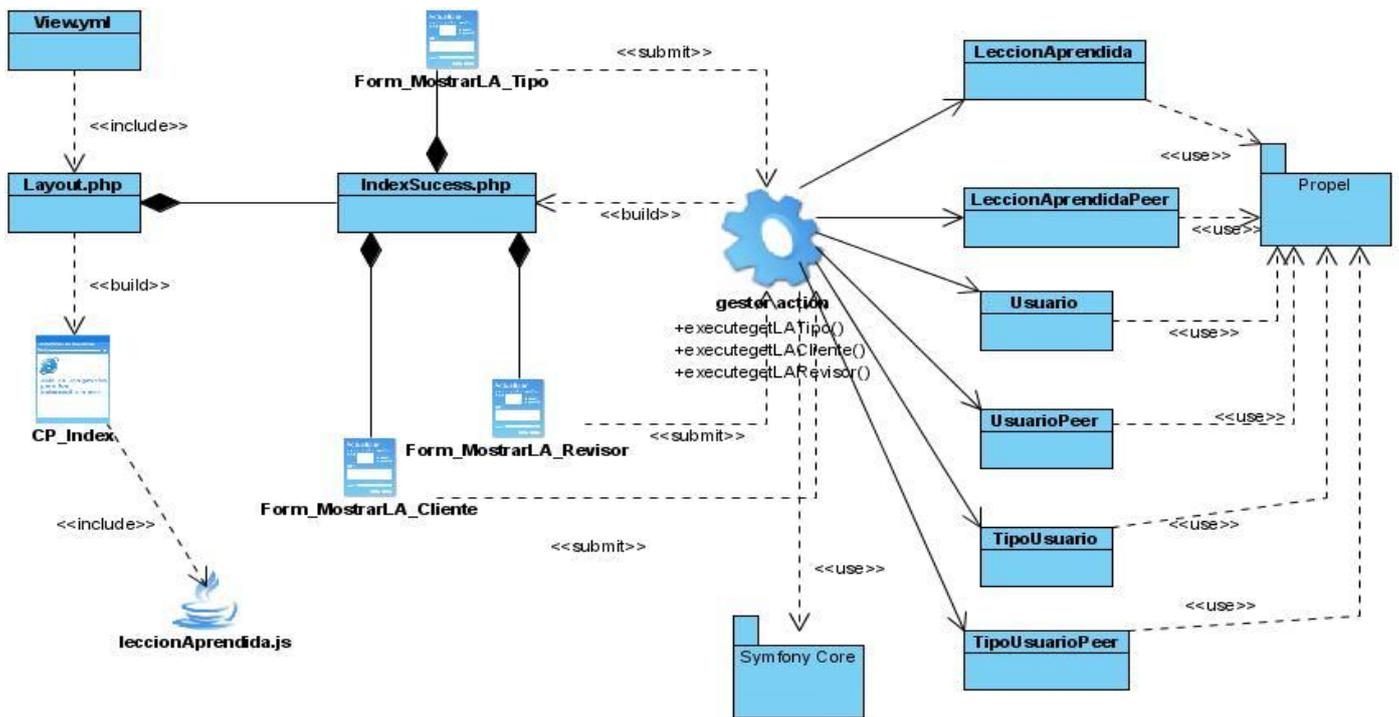


Figura 3.5 Diagrama de Diseño Web del CU Mostrar Reportes.

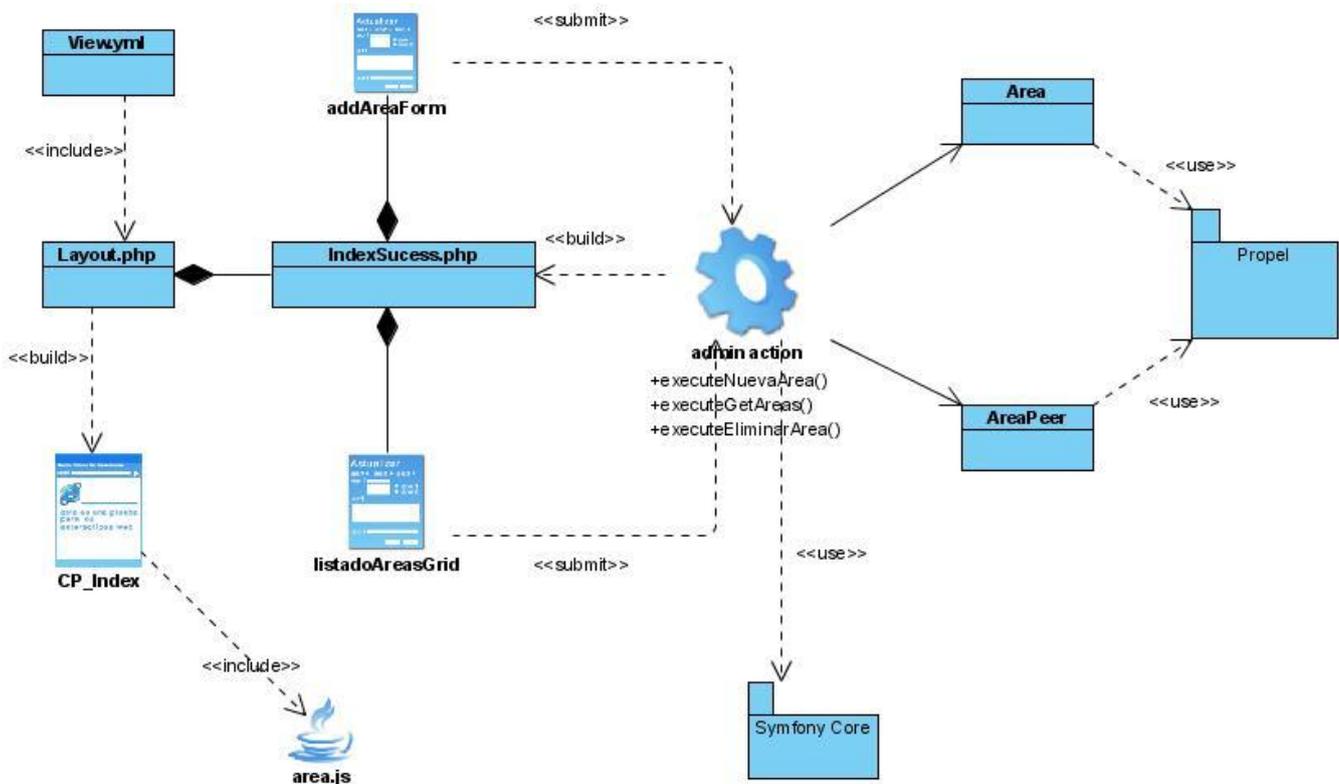


Figura 3.6 Diagrama de Diseño Web del CU Gestionar Área.

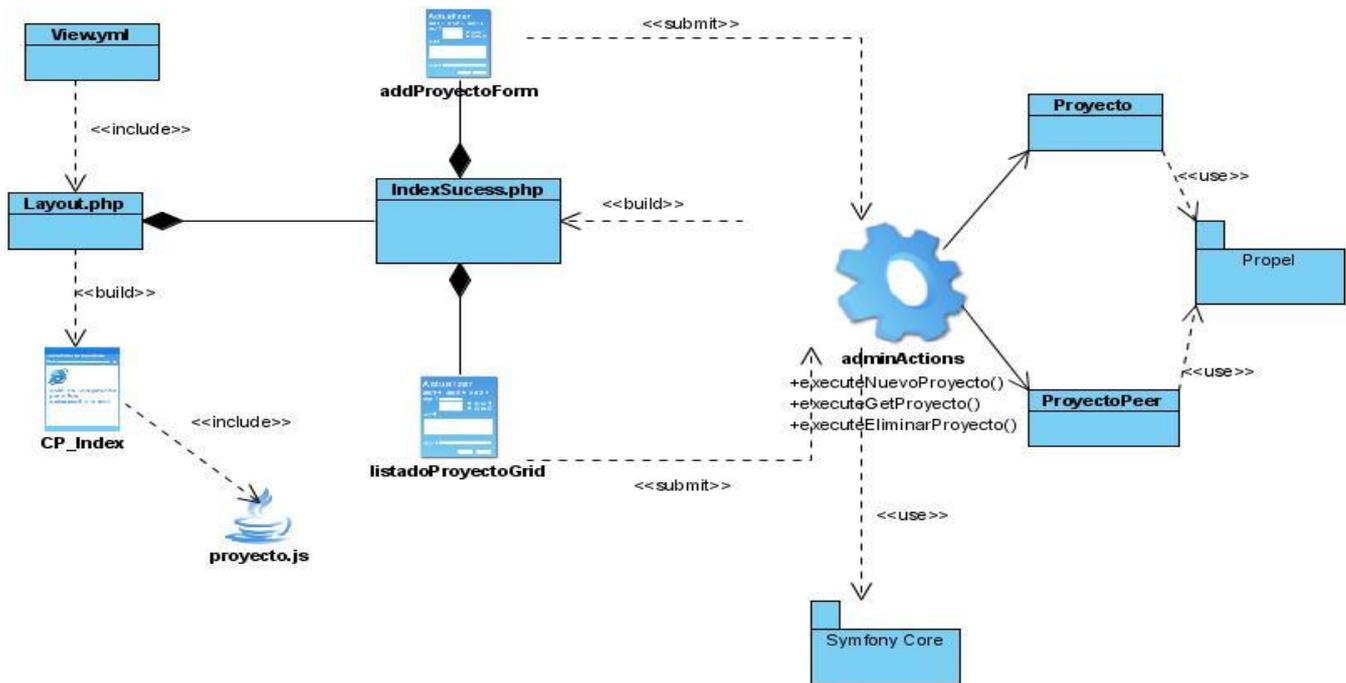


Figura 3.7 Diagrama de Diseño web del CU Gestionar Proyecto.

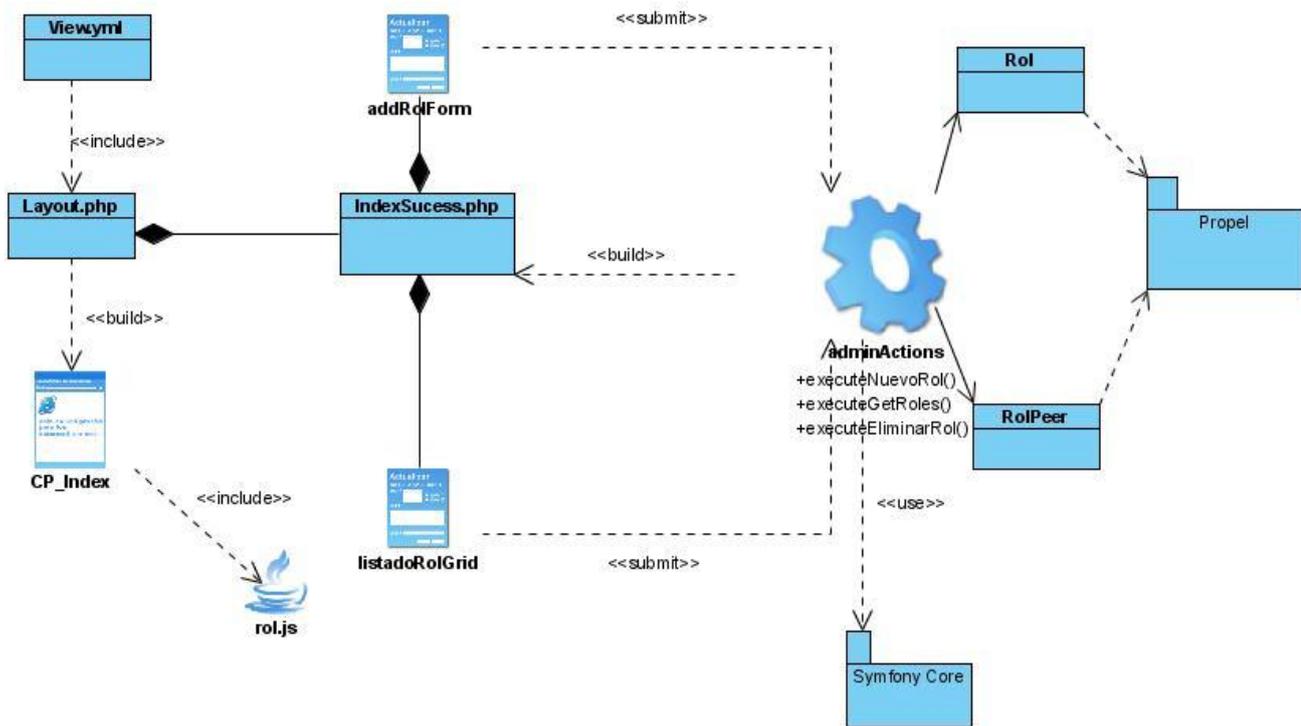


Figura 3.8 Diagrama de Diseño Web del CU Gestionar Roles.

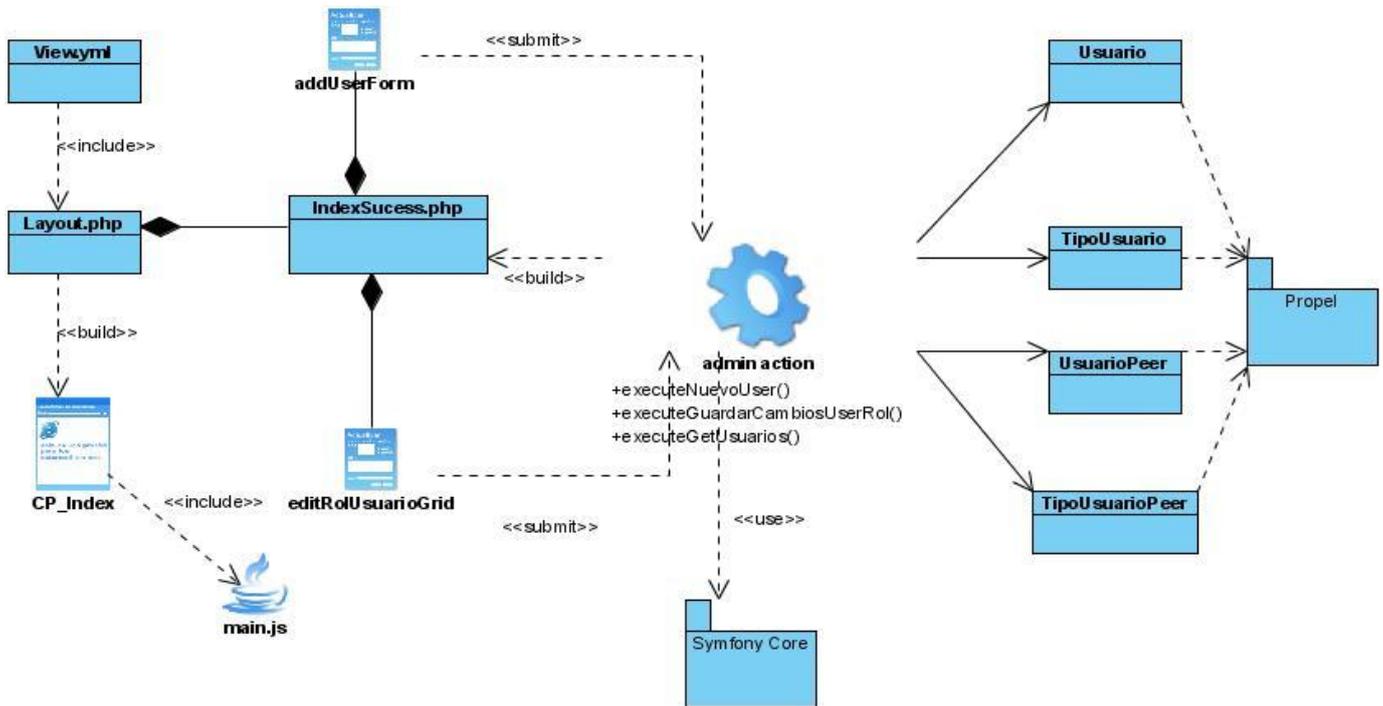


Figura 3.9 Diagrama de Diseño Web del CU Gestionar Usuario.

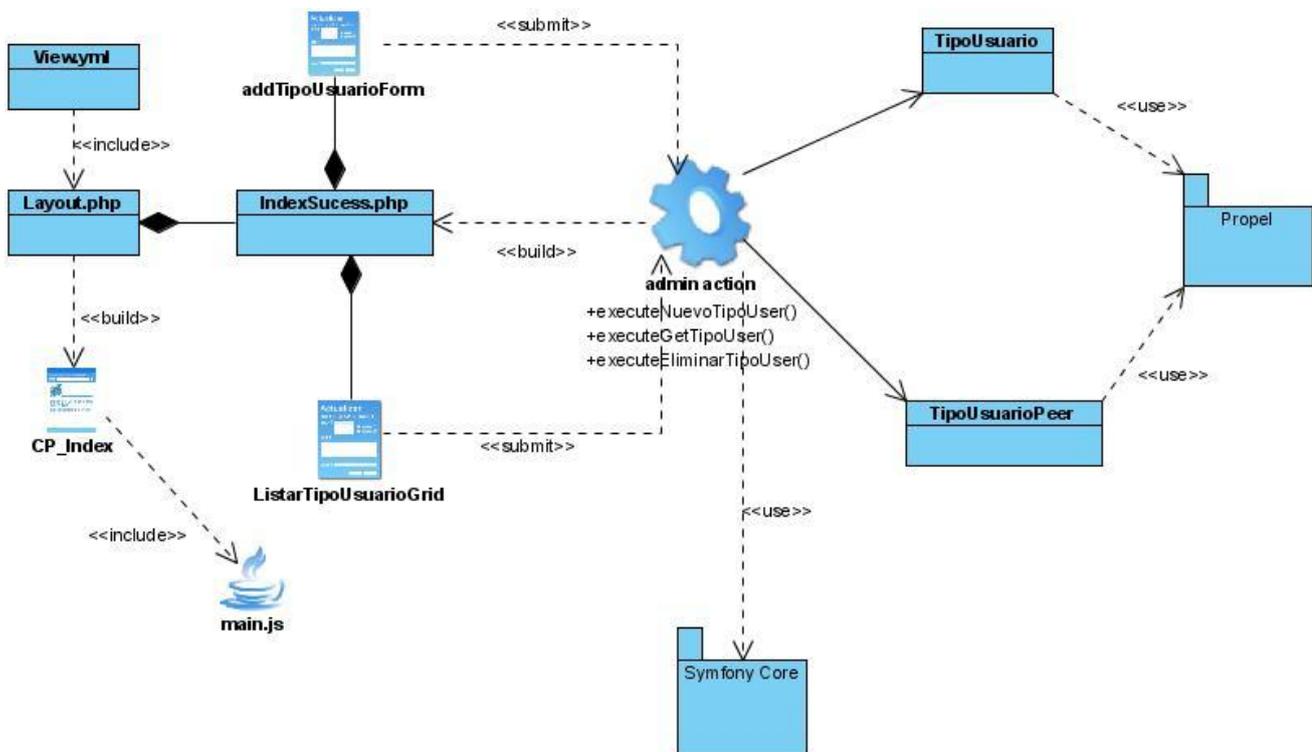


Figura 3.10 Diagrama de Diseño Web del CU Gestionar Tipo de Usuario
 Anexo 4 Descripciones de las clases del diseño.

Nombre: addAreaForm	
Tipo de clase (interfaz)	
Atributo	Tipo
areaField	textfield
Aceptar	button
Cancelar	button
Para cada responsabilidad:	
Nombre:	
Descripción:	
Nombre: addUserForm	
Tipo de clase (interfaz)	
Atributo	Tipo
userField	textfield
Aceptar	button
Cancelar	button
Para cada responsabilidad:	
Nombre:	
Descripción:	

Nombre: addTipoUserForm	
Tipo de clase (interfaz)	
Atributo	Tipo
tipoUserField	textfield
Aceptar	button
Cancelar	button
Para cada responsabilidad:	
Nombre:	
Descripción:	

Nombre: addProyectoForm	
Tipo de clase (interfaz)	
Atributo	Tipo
proyectoField	textfield
areaProyecto	combo
Aceptar	button
Cancelar	button
Para cada responsabilidad:	
Nombre:	
Descripción:	

Nombre: addRolForm	
Tipo de clase (interfaz, controladora, entidad--)	
Atributo	Tipo
rolField	textfield
Aceptar	button
Cancelar	button
Para cada responsabilidad:	
Nombre:	
Descripción:	

Nombre: admin actions	
Tipo de clase (servidora)	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	executeNuevaArea()
Descripción:	Este método es para adicionar las nuevas áreas a la base de dato.
Nombre:	executeGetAreas()
Descripción:	Este método me devuelve todas las áreas que hay guardadas en la base de dato.
Nombre:	executeEliminarArea()
Descripción:	Este método me elimina el área que el administrador selecciona, de la base de dato.
Nombre:	executeNuevoRol()
Descripción:	Este método es para adicionar los nuevos roles a la base de dato.
Nombre:	executeGetUsuarios()
Descripción:	Este método me devuelve todos los usuarios que hay guardados en la base de dato.
Nombre:	executeRoles()
Descripción:	Este método me devuelve todos los roles del sistema , es decir los tipos de usuarios que hay guardados en la base de dato.
Nombre:	executeGetRoles()
Descripción:	Este método me devuelve todos los roles que hay guardados en la base de dato.
Nombre:	executeEliminarRol()
Descripción:	Este método me elimina el rol que el administrador selecciona, de la base de dato.
Nombre:	executeNuevoProyecto()
Descripción:	Este método es para adicionar los nuevos proyectos a la base de dato.
Nombre:	executeGetProyectos()
Descripción:	Este método me devuelve todos los proyectos que hay guardados en la base de dato.
Nombre:	executeEliminarProyecto()
Descripción:	Este método me elimina el proyecto que el administrador selecciona, de la base de dato.
Nombre:	executeGuardarCambiosUserRole()
Descripción:	Este método es para asignar o cambiar el rol del sistema que va a desempeñar un usuario determinado, en la base de dato.
Nombre:	executeNuevoUser()
Descripción:	Este método es para adicionar los nuevos usuarios a la base de dato.
Nombre:	executeNuevoTipoUser()
Descripción:	Este método es para adicionar los nuevos tipos de usuarios a la base de dato.
Nombre:	executeGetRolesPorProyecto()
Descripción:	Este método devuelve todos los roles que pertenecen a un proyecto.
Nombre:	executeEliminarProyectoRol()
Descripción:	Este método me elimina un rol de un proyecto.
Nombre:	executeAddProyectoRol()
Descripción:	Este método me adiciona un rol a un proyecto.
Nombre:	executeGetUsersPorProyecto()
Descripción:	Este método me devuelve todos los usuarios que pertenecen a un proyecto determinado.
Nombre:	executeEliminarProyectoUser()
Descripción:	Este método elimina un usuario de un proyecto.
Nombre:	executeUsuariosSinProyecto()
Descripción:	Este método me devuelve los usuarios que no están en proyecto.
Nombre:	executeAddProyectoUser()
Descripción:	Este método me adiciona un usuario a un proyecto.
Nombre:	executeValidadorCliente()
Descripción:	Este método es para saber si un usuario esta en un proyecto, por lo tanto es cliente.

Nombre:gestor actions	
Tipo de clase (servidora)	
Atributo	Tipo
Para cada responsabilidad:	
Nombre:	executeAddLeccAprendida()
Descripción:	Este método es para adicionar las nuevas lecciones aprendidas a la base de dato.
Nombre:	executeLeccionesPorUsuario()
Descripción:	Este método me devuelve todas las lecciones que pertenecen a un usuario determinado.
Nombre:	executeSolucionesDeLeccion()
Descripción:	Este método me devuelve las soluciones de una lección determinada.
Nombre:	executeGetLeccionById()
Descripción:	Este método me devuelve las lecciones dado un id.
Nombre:	executeEditarLeccion()
Descripción:	Este método me permite modificar los datos de las lecciones aprendidas.
Nombre:	executeEliminarSolucion()
Descripción:	Este método elimina las soluciones de una lección.
Nombre:	executeAddSolucion()
Descripción:	Este método adiciona soluciones
Nombre:	executeLeccionesAprobar()
Descripción:	Este método devuelve las lecciones aprobadas
Nombre:	executeLeccionesPublicar()
Descripción:	Este método devuelve las lecciones publicadas.
Nombre:	executeAprobarLeccion()
Descripción:	Este método cambia la lección de registrada a aprobada.
Nombre:	executePublicarLeccion()
Descripción:	Este método cambia la lección de aprobada a publicada.
Nombre:	executeAllPalabras()
Descripción:	Este método devuelve todas las palabras claves
Nombre:	executeLeccionesPorPalabra()
Descripción:	Este método devuelve las lecciones que tengan determinadas palabras claves
Nombre:	executeAllProyectos()
Descripción:	Este método devuelve todos los proyectos
Nombre:	executeLeccionesPorProyecto()
Descripción:	Este método devuelve las lecciones de determinados proyectos
Nombre:	executeAllClientes()
Descripción:	Este método devuelve todos los clientes
Nombre:	executeLeccionesPorCliente()
Descripción:	Este método devuelve las lecciones de determinado cliente

Anexo 5 Diagramas de secuencia de los casos de usos que conforman la propuesta del sistema

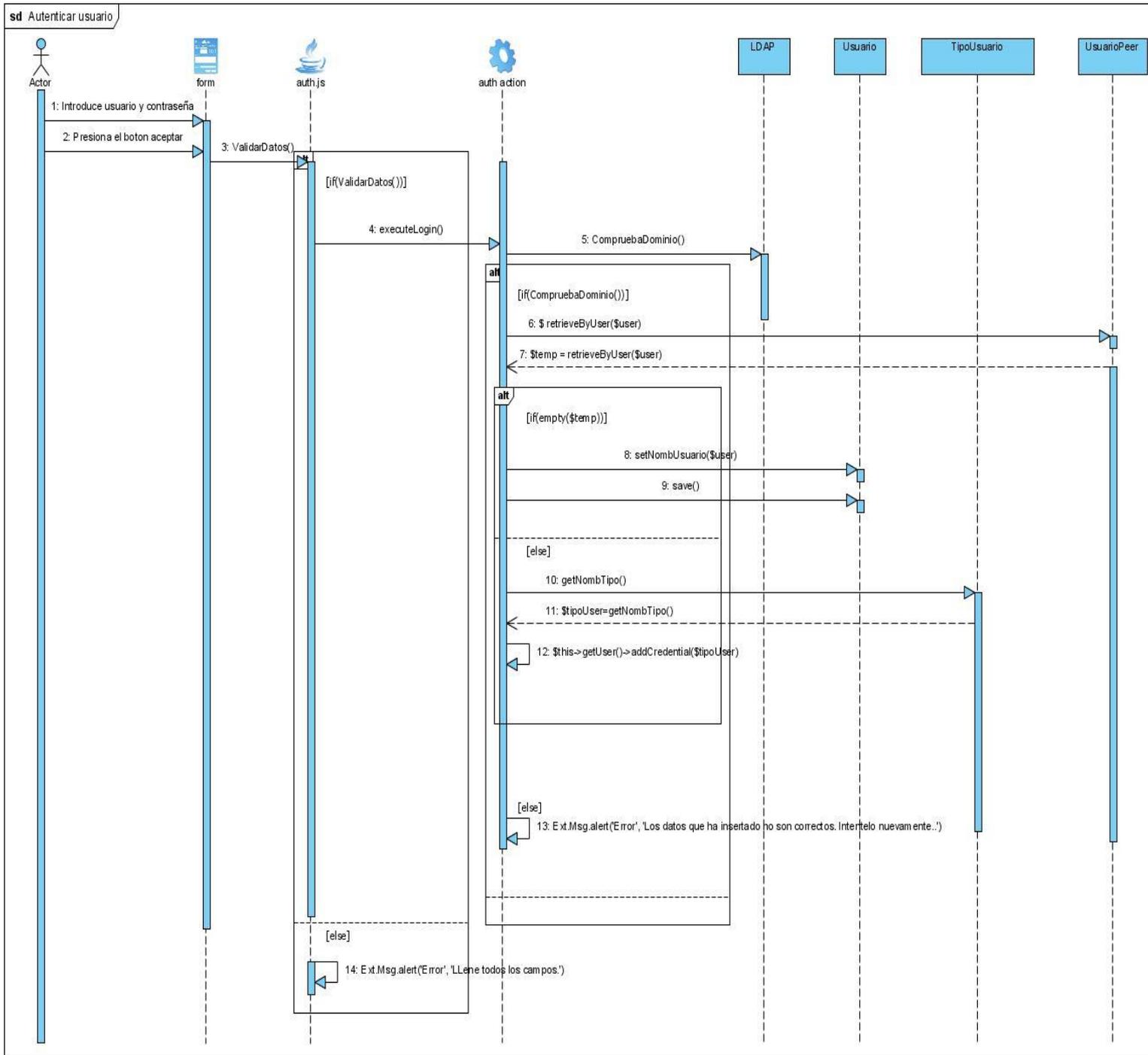


Figura 5.1 Diagramas de secuencia Autenticar

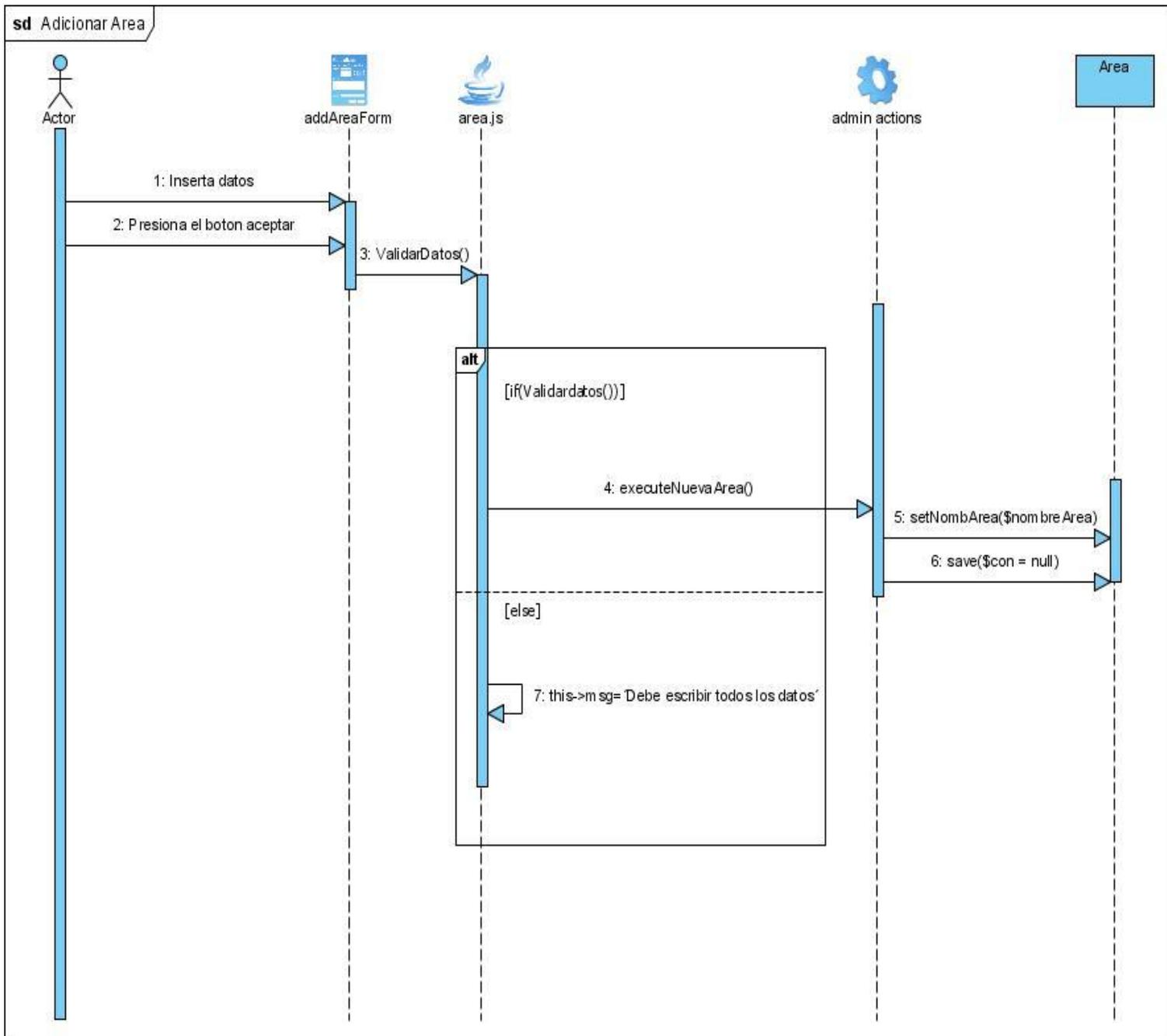


Figura 5.2 Diagramas de secuencia Adicionar Área

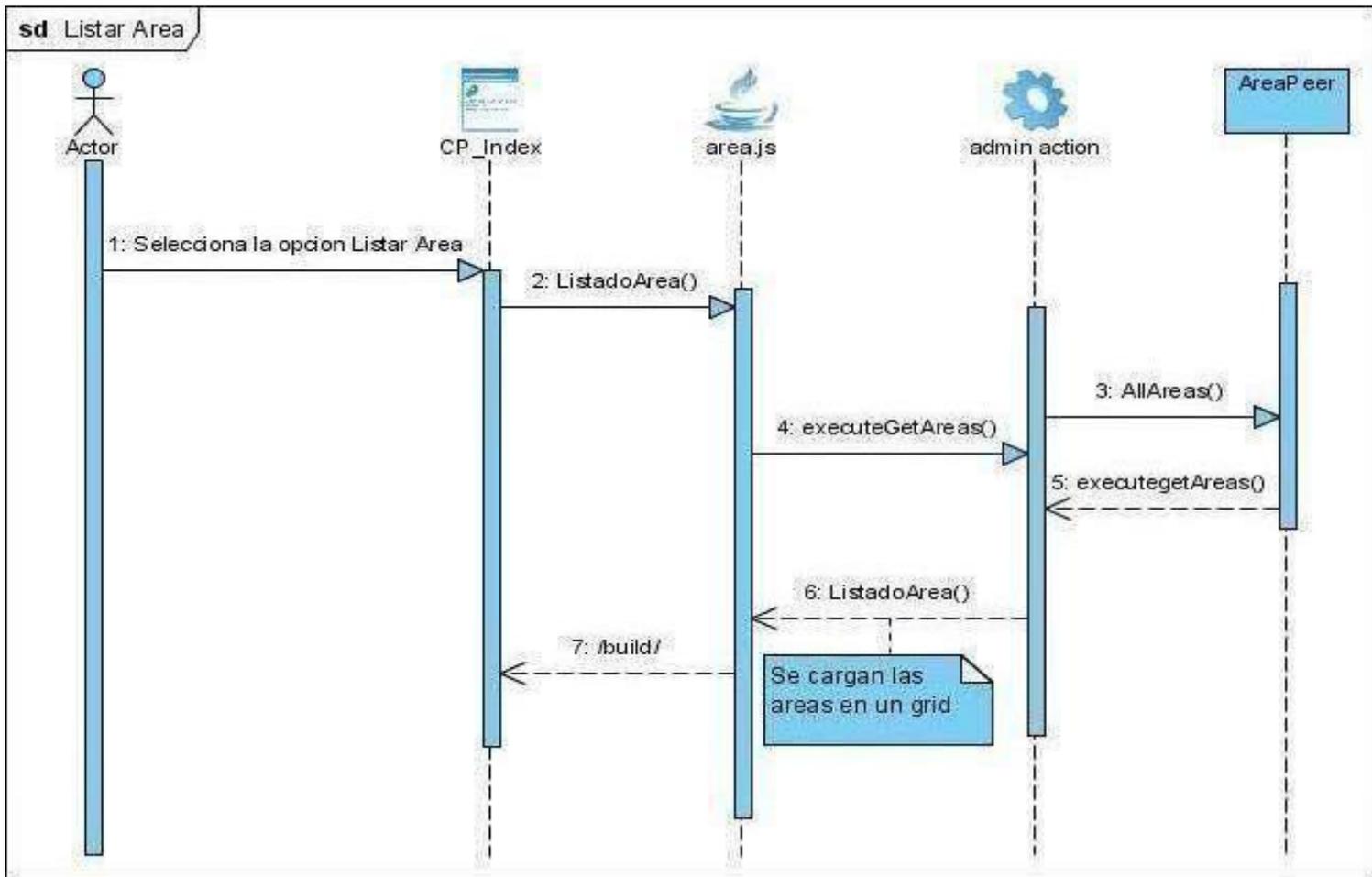


Figura 5.3 Diagramas de secuencia Listar Área

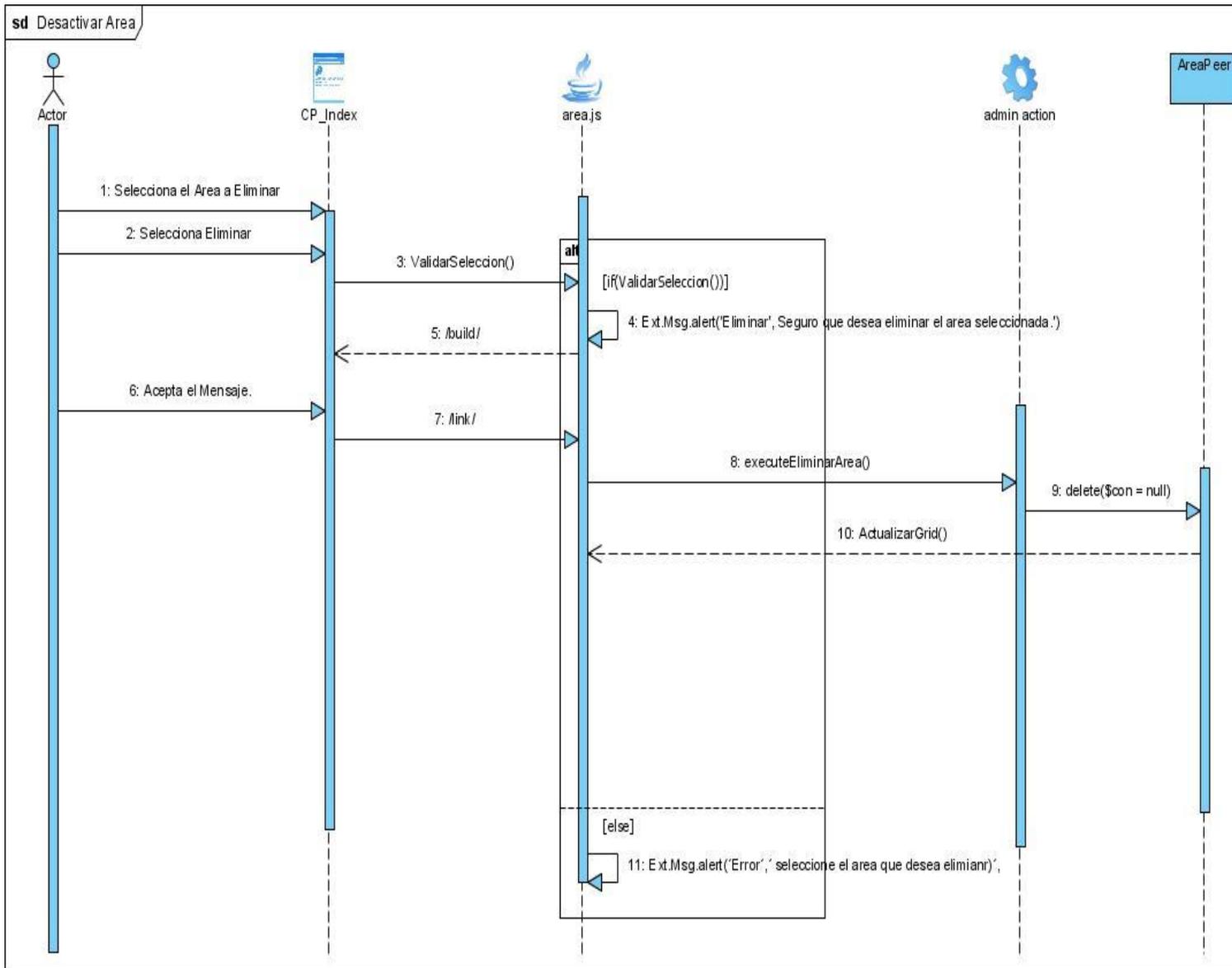


Figura 5.4 Diagramas de secuencia Eliminar Área

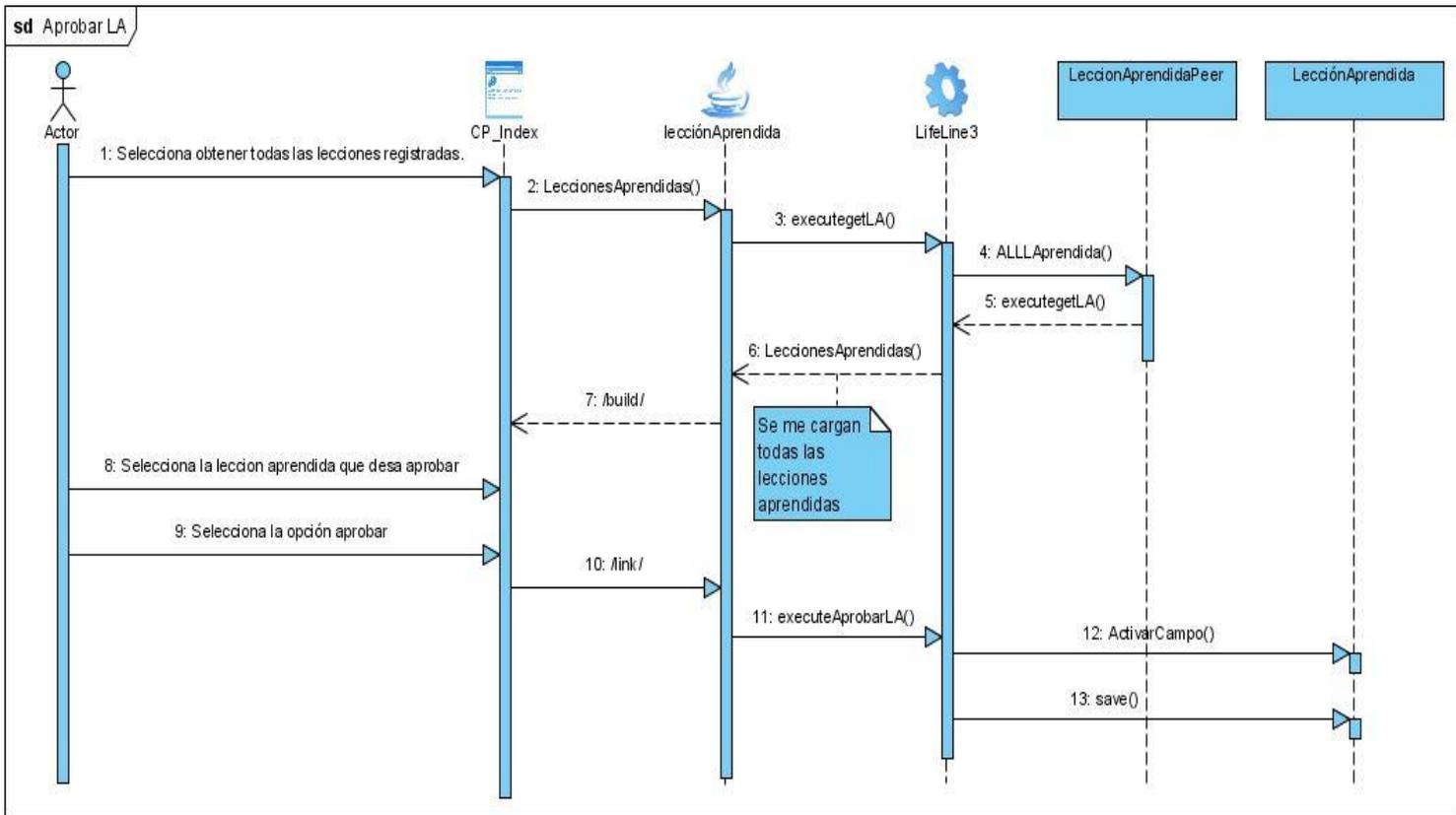


Figura 5.5 Diagramas de secuencia Aprobar Lección Aprendida

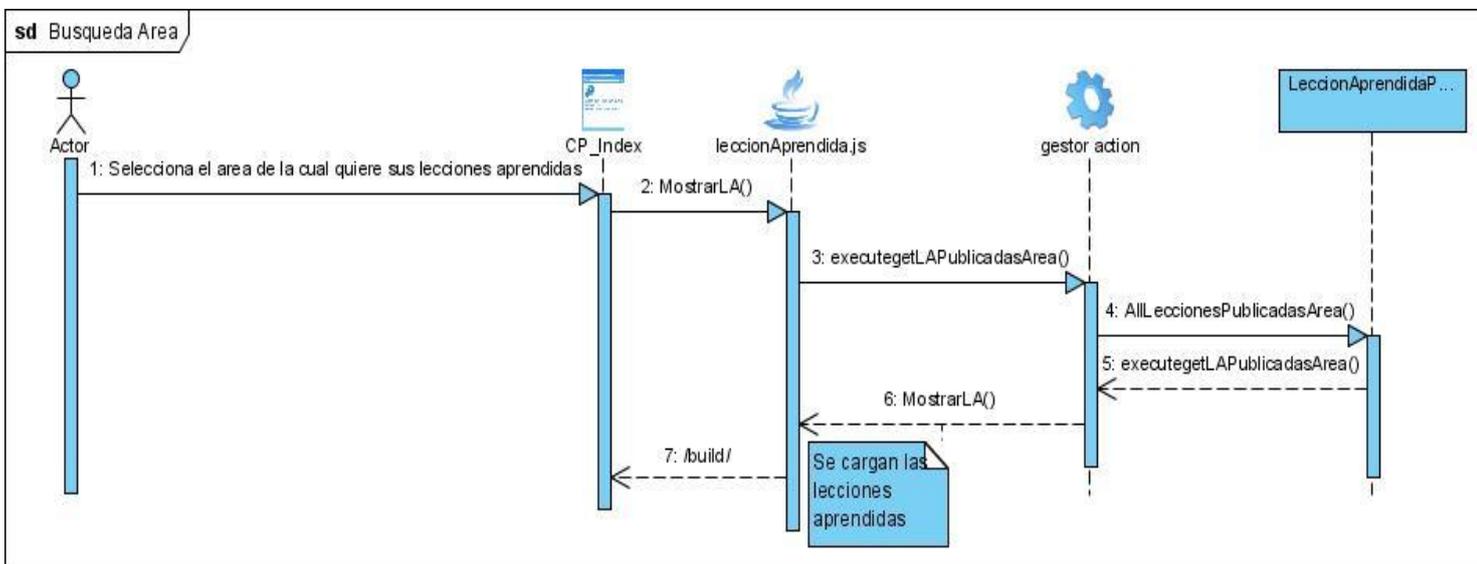


Figura 5.6 Diagramas de secuencia Buscar Lección Aprendida por Area

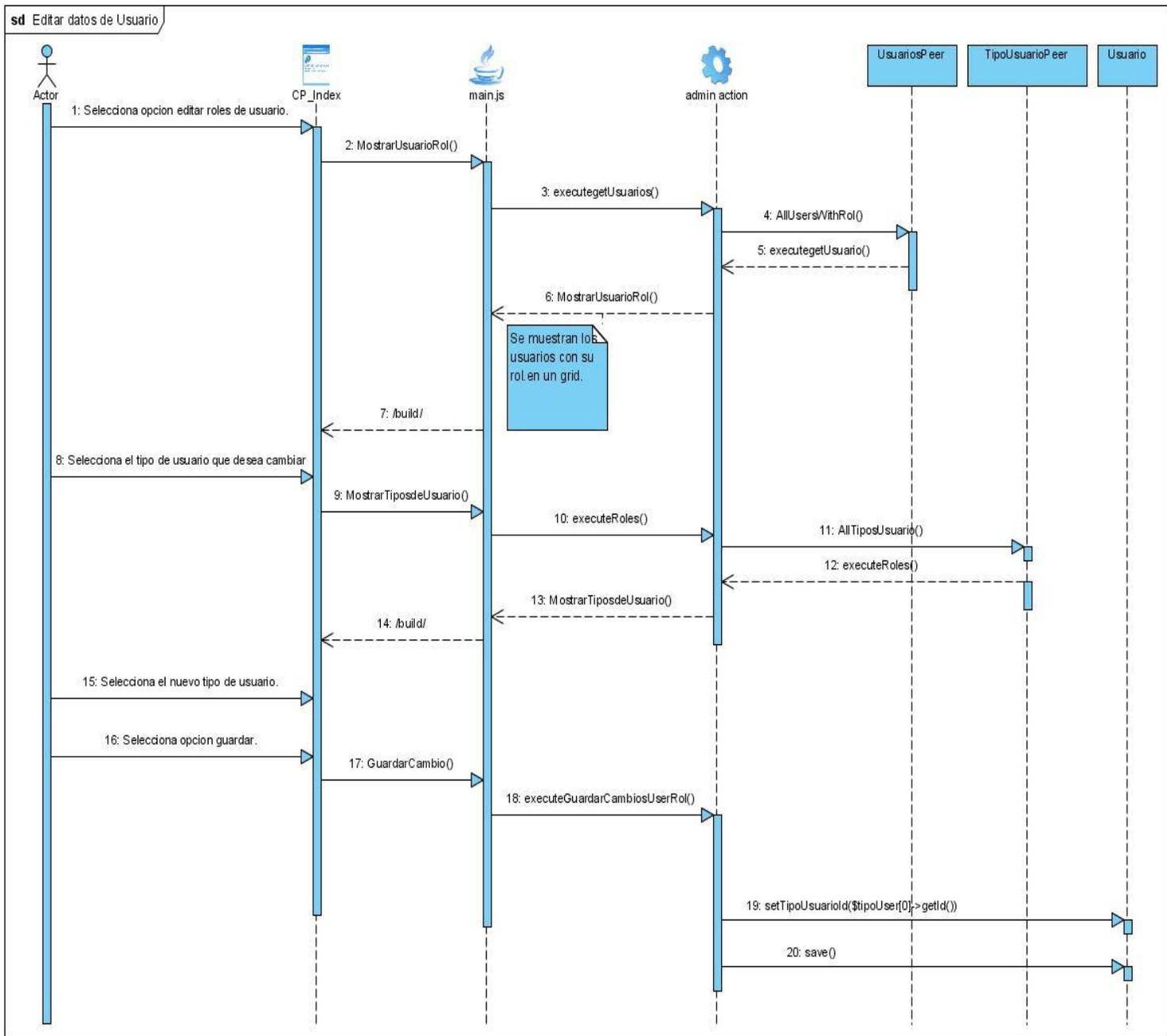


Figura 5.7 Diagramas de secuencia Editar Datos de Usuario.

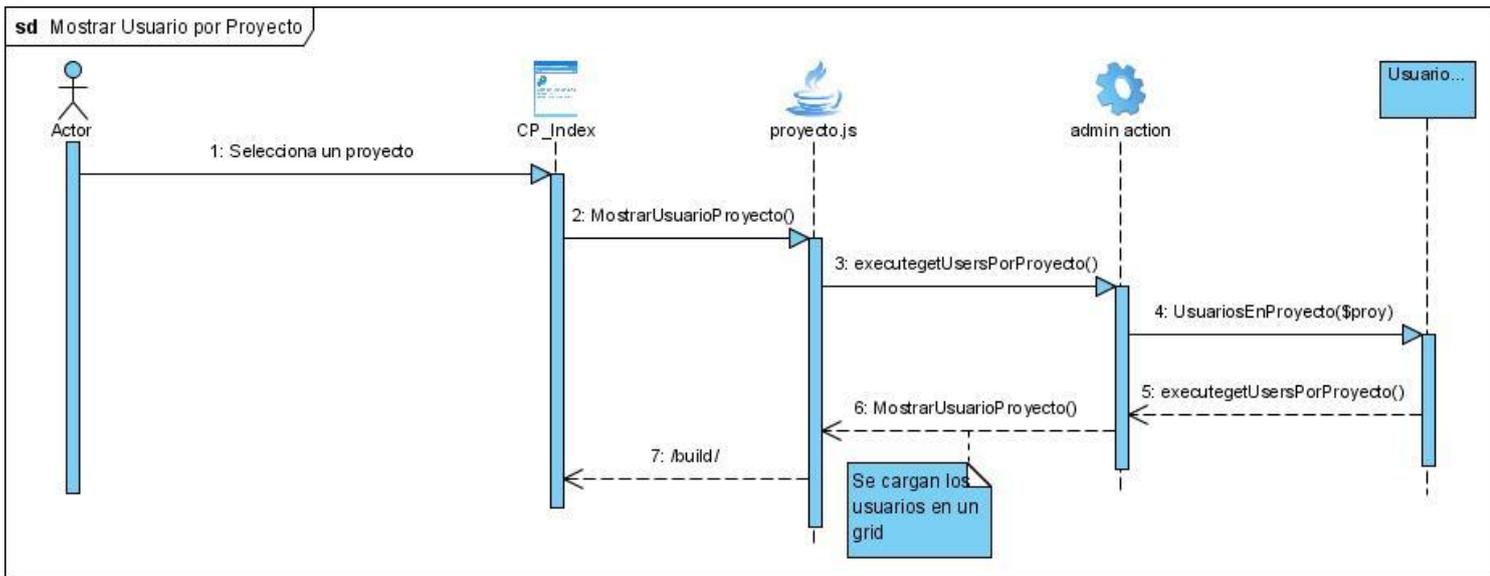


Figura 5.8 Diagramas de secuencia Mostrar Usuario por Proyecto.

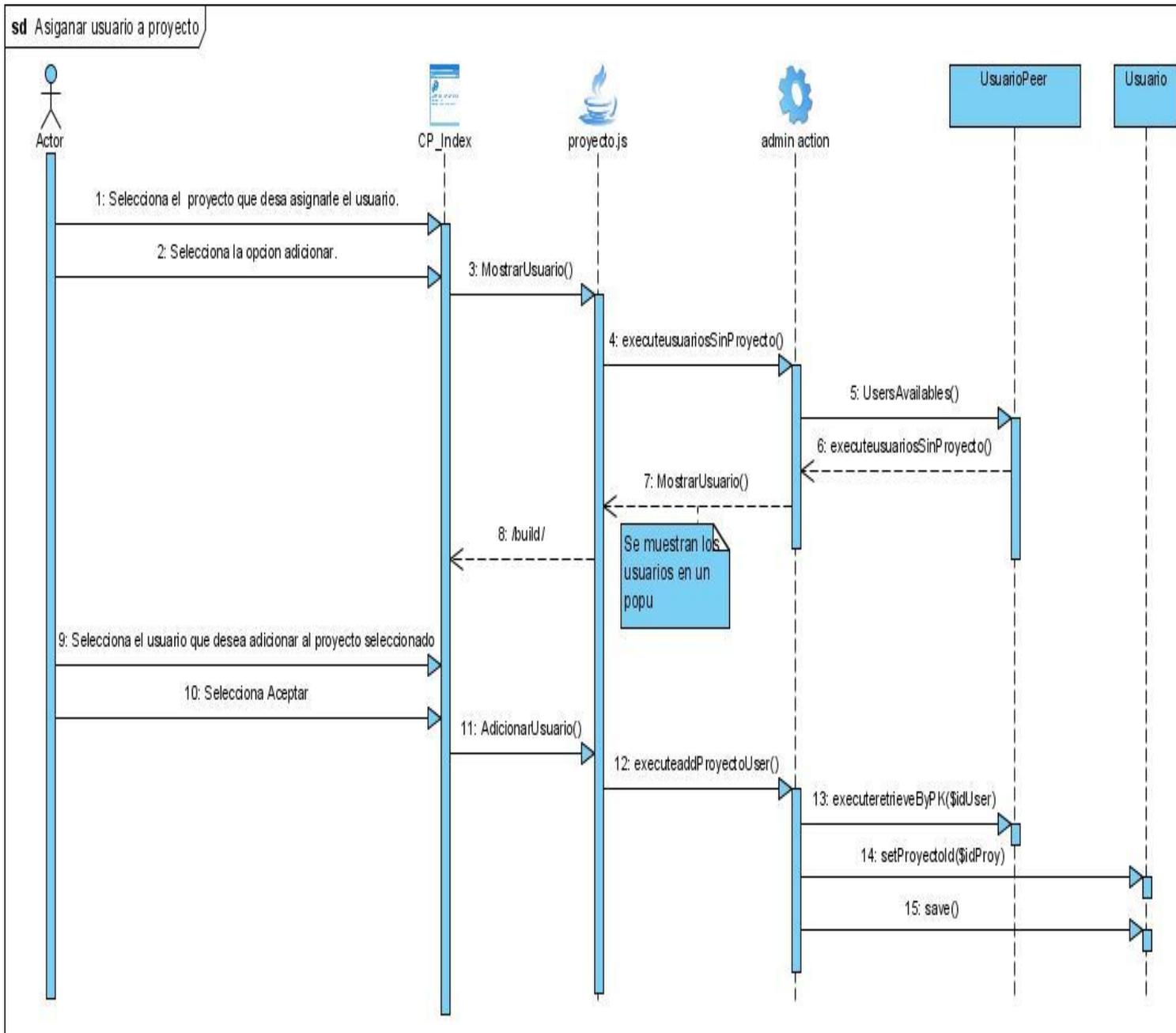


Figura 5.9 Diagramas de secuencia Asignar Usuario a un Proyecto.

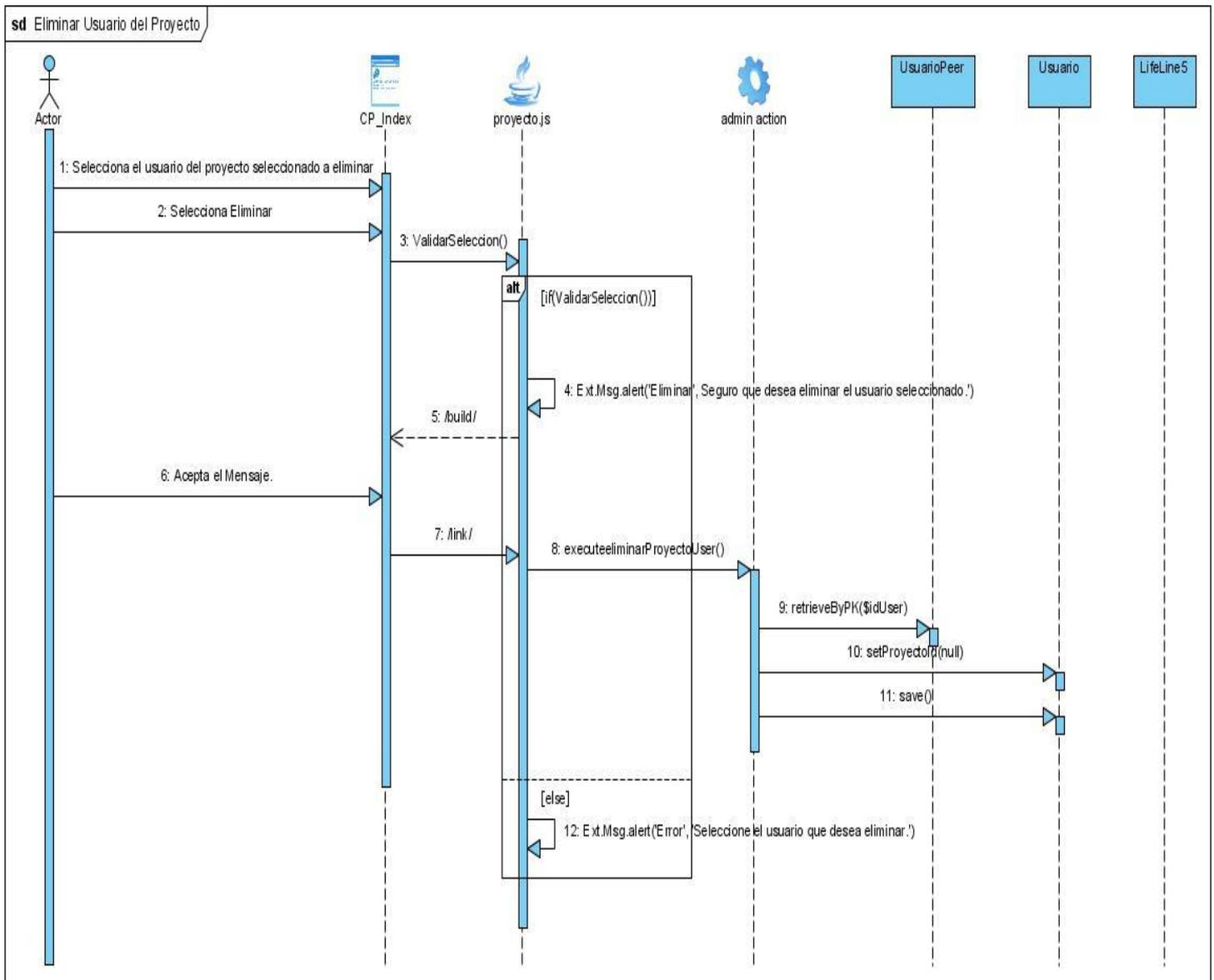


Figura 5.10 Diagramas de secuencia Eliminar Usuario del Proyecto.

Anexo 6 Descripción de las tablas de la Base de Datos

Nombre: **Usuario**

Descripción: En esta tabla se almacenan los nombres de los usuarios que visiten este sitio, ya sean usuarios, clientes, revisores o administradores.

Atributo	Tipo	Descripción
Id	Integer	Este campo es una llave primaria y almacena el identificador que distingue a esta tabla.
nomb_usuario	Varchar	Este campo se almacena el nombre del usuario
proyecto_id	Integer	Este campo es una llave foránea y almacena el identificador

		de la tabla proyecto
tipo_usuario_id	Integer	Este campo es una llave foránea y almacena el identificador de la tabla Tipo de Usuario
Id_Rol		Este campo es una llave foránea y almacena el identificador de la tabla Rol

Tabla 3.1 Descripción de la tabla Usuario

Nombre: Tipo Usuario		
Descripción: En esta tabla se almacenan los tipos de usuario que pueden visitar el sitio.		
Atributo	Tipo	Descripción
Id	Integer	Este campo es una llave primaria y almacena el identificador que distingue a esta tabla.
nomb_tipo	Varchar	En este campo se almacenan los diferentes tipos de usuarios que van a utilizar el sitio.

Tabla 3.2 Descripción de la tabla Tipo Usuario.

Nombre: Área		
Descripción: En esta tabla se almacenan las áreas a las que pertenecen los proyectos productivos de la Facultad.		
Atributo	Tipo	Descripción
Id	Integer	Este campo es una llave primaria y almacena el identificador que distingue a esta tabla.
nomb_area	Varchar	En este campo se almacenan las diferentes áreas a las que pertenecen los proyectos.

Tabla 3.3 Descripción de la tabla Área.

Nombre: Proyecto		
Descripción: En esta tabla se almacenan los proyectos productivos de la Facultad.		
Atributo	Tipo	Descripción
Id	Integer	Este campo es una llave primaria y almacena el identificador que distingue a esta tabla.
nomb_proyecto	Varchar	En este campo se almacenan los nombres de los proyectos.
Area_id	Integer	Este campo es una llave foránea y almacena el identificador de la tabla Área.

Tabla 3.4 Descripción de la tabla Proyecto.

Nombre: Rol		
--------------------	--	--

Descripción: En esta tabla se almacenan los diferentes roles que van a estar presente en cada uno de los proyectos productivos de la Facultad.

Atributo	Tipo	Descripción
id_rol	Integer	Este campo es una llave primaria y almacena el identificador que distingue a esta tabla.
nomb_rol	Varchar	En este campo se almacenan los nombres de los roles que están presente en los proyectos.

Tabla 3.5 Descripción de la tabla Rol.

Nombre: **Lección Aprendida**

Descripción: En esta tabla se almacena la información referente a las lecciones aprendidas.

Atributo	Tipo	Descripción
Id	Integer	Este campo es una llave primaria y almacena el identificador que distingue a esta tabla.
Tipo	Varchar	En este campo se almacena el tipo de lección aprendida (positiva o negativa)
descripcion_problema	Varchar	En este campo se almacena la descripción del problema que dio lugar a esta lección aprendida
analisis_solucion	Varchar	En este campo se almacena el análisis de las soluciones que se le hicieron al problema que existió.
palabras_claves	Varchar	En este campo se almacena las palabras claves de la lección aprendida.
aprobada	Varchar	En este campo se almacena si la lección aprendida esta o no aprobada.
publicada	Char	En este campo se almacena si la lección aprendida esta o no publicada.
proyecto_id	Integer	Este campo es una llave foránea y almacena el identificador de la tabla Proyecto.
Usuario_id	Integer	Este campo es una llave foránea y almacena el identificador de la tabla Usuario.

Tabla 3.6 Descripción de la tabla Lección Aprendida.

Nombre: **Solución**

Descripción: En esta tabla se almacenan las soluciones que se le dan a los problemas que se presentan en los proyectos productivos.

Atributo	Tipo	Descripción
----------	------	-------------

Id	Integer	Este campo es una llave primaria y almacena el identificador que distingue a esta tabla.
descripcion	Integer	En este campo se almacenan las soluciones que se le dan a los problemas que se presentan en los proyectos productivos.
lección_aprendida_id	Integer	Este campo es una llave foránea y almacena el identificador de la tabla Lección Aprendida.

Tabla 3.7 Descripción de la tabla Solucion.

Anexo 7 los casos de pruebas del sistema propuesto

CU Autenticar Usuario

1. Usuario y contraseña correcto

Entrada	Resultados	Condiciones
El usuario introduce su nombre de usuario y contraseña correctos	El sistema muestra satisfactoriamente la página a la que tiene acceso el usuario según su rol	El usuario autenticado debe pertenecer al dominio UCI.

2. Usuario o contraseña incorrectos.

Entrada	Resultados	Condiciones
El usuario introduce su nombre de usuario o contraseña incorrectamente	El sistema muestra el siguiente mensaje de error "Los datos que ha insertado no son correctos"	La contraseña o el usuario introducido no se corresponden con el dominio UCI.

3. Campos vacíos.

Entrada	Resultados	Condiciones
El usuario deja algún campo vacío	El sistema muestra el siguiente mensaje de error "Llene todos los campos"	El usuario autenticado no llena todos los campos e intenta entrar al sistema

CU Gestionar Área

Sección Adicionar Área.

1. Los datos introducidos son correctos.

Entrada	Resultados	Condiciones
El administrador introduce el nombre del área correctamente.	El sistema adiciona a la base de datos el área.	Los campos se hayan llenado correctamente.

2. Campos vacíos

Entrada	Resultados	Condiciones
El administrador deja algún campo vacío	El sistema muestra el siguiente mensaje de error "Debe escribir todos los datos"	El administrador no llena todos los campos e intenta adicionar el área.

3. Área existente.

Entrada	Resultados	Condiciones
El administrador introduce el nombre de un área que ya existe.	El sistema muestra el siguiente mensaje de error "Existe un área con ese nombre"	El área que desean adicionar ya esta guardada en la base de datos.

Sección Eliminar Área.

1. El usuario escoge un área

Entrada	Resultados	Condiciones
El administrador selecciona un área.	El sistema elimina el área seleccionada y actualiza la información.	El administrador seleccione una de las áreas que están guardadas en la bases de datos.

2. El usuario no escoge ningún área

Entrada	Resultados	Condiciones
El administrador no selecciona ningún área.	El sistema muestra el siguiente mensaje de error "Selecciona el área que desea eliminar"	El administrador no seleccione ninguna de las áreas que están guardadas en la bases de datos

CU Gestionar Proyecto

Sección Adicionar Proyecto.

1. Los datos introducidos son correctos.

Entrada	Resultados	Condiciones
El administrador introduce el nombre del proyecto correctamente.	El sistema adiciona a la base de datos el proyecto.	Los campos se hayan llenado correctamente.

2. Campos vacíos.

Entrada	Resultados	Condiciones
El administrador deja algún campo vacío	El sistema muestra el siguiente mensaje de error	El administrador no llena todos los campos e intenta adicionar el

	“Debe escribir todos los datos”	proyecto.
--	---------------------------------	-----------

3. Proyecto Existente.

Entrada	Resultados	Condiciones
El administrador introduce el nombre de un proyecto que ya existe.	El sistema muestra el siguiente mensaje de error “Existe un proyecto con ese nombre”	El proyecto que desean adicionar ya esta guardado en la base de datos.

4. No selecciona el área.

Entrada	Resultados	Condiciones
El administrador no selecciona el área.	El sistema muestra el siguiente mensaje de error “Debe seleccionar un área”	El administrador no selecciona una de las áreas que están guardadas en la base de datos.

Sección Eliminar Proyecto.

1. El usuario escoge un proyecto

Entrada	Resultados	Condiciones
El administrador selecciona un proyecto.	El sistema elimina el proyecto seleccionado y actualiza la información.	El administrador seleccione uno de los proyectos que están guardados en la bases de datos.

2. El usuario no escoge ningún proyecto

Entrada	Resultados	Condiciones
El administrador no selecciona ningún proyecto.	El sistema muestra el siguiente mensaje de error “Selecciona el proyecto que desea eliminar”	El administrador no seleccione ninguno de los proyecto que están guardado en la bases de datos

Sección Asignar Proyecto.

1. El administrador escoge un área y un proyecto.

Entrada	Resultados	Condiciones
El administrador selecciona un área y los proyectos que van a estar presentes en esta área.	El sistema le asigna al área los proyectos escogidos por el administrador y se actualiza la información.	El administrador seleccione un área y los proyectos que estén guardados en la bases de datos.

2. El administrador no escoge ningún rol

Entrada	Resultados	Condiciones

El administrador no selecciona ningún proyecto.	El sistema muestra el siguiente mensaje de error "Selecciona los proyectos que desea asignar al área"	El administrador no seleccione ninguno de los proyectos que están guardados en la base de datos.
---	---	--

CU Gestionar Roles**Sección Adicionar Roles.****1. Los datos introducidos son correctos.**

Entrada	Resultados	Condiciones
El administrador introduce el nombre del rol correctamente.	El sistema adiciona a la base de datos el rol.	Los campos se hayan llenado correctamente.

2. Campos vacíos.

Entrada	Resultados	Condiciones
El administrador deja algún campo vacío	El sistema muestra el siguiente mensaje de error "Debe escribir todos los datos"	El administrador no llena todos los campos e intenta adicionar el rol.

3. Rol Existente.

Entrada	Resultados	Condiciones
El administrador introduce el nombre de un rol que ya existe.	El sistema muestra el siguiente mensaje de error "Existe un rol con ese nombre"	El rol que desean adicionar ya esta guardado en la base de datos.

4. No selecciona el Rol.

Entrada	Resultados	Condiciones
El administrador no selecciona el proyecto.	El sistema muestra el siguiente mensaje de error "Debe seleccionar un proyecto"	El administrador no selecciona uno de los proyectos que están guardados en la base de datos.

Sección Eliminar Rol.**1. El usuario escoge un rol**

Entrada	Resultados	Condiciones
El administrador selecciona un rol.	El sistema elimina el rol seleccionado y actualiza la	El usuario seleccione un rol que está guardado en la bases de

	información.	datos.
--	--------------	--------

2. El usuario no escoge ningún rol

Entrada	Resultados	Condiciones
El administrador no selecciona ningún rol.	El sistema muestra el siguiente mensaje de error "Selecciona el rol que desea eliminar"	El administrador no seleccione ninguno de los roles que están guardado en la bases de datos

Sección Asignar roles.

3. El administrador escoge un proyecto y roles.

Entrada	Resultados	Condiciones
El administrador selecciona un proyecto y los roles que van a estar presentes en ese proyecto.	El sistema le asigna al proyecto los roles escogidos por el administrador y se actualiza la información.	El administrador seleccione un proyecto y roles que estén guardados en la bases de datos.

4. El administrador no escoge ningún rol

Entrada	Resultados	Condiciones
El administrador no selecciona ningún rol.	El sistema muestra el siguiente mensaje de error "Selecciona los roles que desea asignar al proyecto"	El administrador no seleccione ninguno de los roles que están guardado en la base de datos.

CU Gestionar Usuario

Sección Adicionar Usuario.

1. Los datos introducidos son correctos.

Entrada	Resultados	Condiciones
El administrador introduce los datos del usuario que desea adicionar.	El sistema adiciona a la base de datos el usuario.	Los campos se hayan llenado correctamente.

2. Campos vacios.

Entrada	Resultados	Condiciones
El administrador deja algún campo vacío	El sistema muestra el siguiente mensaje de error "Debe escribir todos los datos"	El administrador no llena todos los campos e intenta adicionar el usuario.

3. Usuario Existente.

Entrada	Resultados	Condiciones
El administrador introduce el nombre de un usuario que ya existe.	El sistema muestra el siguiente mensaje de error "Existe un usuario con ese nombre"	El usuario que desean adicionar ya esta guardado en la base de datos.

4. No selecciona el Usuario.

Entrada	Resultados	Condiciones
El administrador no selecciona el tipo de usuario.	El sistema muestra el siguiente mensaje de error "Debe seleccionar el tipo de usuario "	El administrador no selecciona uno de los tipos de usuarios que están guardados en la base de datos.

Sección Eliminar Usuario.

1. El administrador escoge un usuario

Entrada	Resultados	Condiciones
El administrador selecciona un usuario	El sistema elimina el usuario seleccionado y actualiza la información.	El administrador seleccione un usuario que está guardado en la bases de datos.

2. El administrador no escoge ningún usuario

Entrada	Resultados	Condiciones
El administrador no selecciona ningún usuario.	El sistema muestra el siguiente mensaje de error "Selecciona el usuario que desea eliminar"	El administrador no seleccione ninguno de los usuarios que están guardado en la bases de datos

Sección Editar Datos.

1. El administrador escoge un usuario y un tipo de usuario.

Entrada	Resultados	Condiciones
El administrador selecciona un usuario y el tipo de usuario.	El sistema le asigna al usuario el tipo de usuario escogido por el administrador y se actualiza la información.	El administrador seleccione un usuario y un tipo de usuario que estén guardados en la bases de datos.

2. El administrador no escoge ningún tipo de usuario

Entrada	Resultados	Condiciones
---------	------------	-------------

El administrador no selecciona ningún tipo de usuario.	El sistema muestra el siguiente mensaje de error "Selecciona el tipo de usuario que desea asignar al usuario"	El administrador no seleccione ninguno de los tipos de usuarios que están guardado en la bases de datos
--	---	---

CU Gestionar Lección Aprendida**Sección Adicionar Lección Aprendida.****1. Los datos introducidos son correctos.**

Entrada	Resultados	Condiciones
El cliente introduce los datos de la lección aprendida que desea adicionar.	El sistema adiciona a la base de datos la lección aprendida.	Los campos se hayan llenado correctamente.

2. Campos vacios.

Entrada	Resultados	Condiciones
El cliente deja algún campo vacío	El sistema muestra el siguiente mensaje de error "Debe escribir todos los datos"	El cliente no llena todos los campos e intenta adicionar el usuario.

Sección Modificar Lección Aprendida.**1. Los datos introducidos son correctos.**

Entrada	Resultados	Condiciones
El cliente selecciona la lección aprendida que desea modificar.	El sistema actualiza la información que ha sido modificada en la base de datos.	El cliente selecciona una lección aprendida.

2. Campos vacios.

Entrada	Resultados	Condiciones
El cliente deja algún campo vacío	El sistema muestra el siguiente mensaje de error "Debe escribir todos los datos"	El cliente no llena todos los campos e intenta guardar la información modificada.

Sección Eliminar Lección Aprendida.**1. El cliente escoge una lección aprendida**

Entrada	Resultados	Condiciones
El cliente selecciona una lección aprendida	El sistema elimina la lección aprendida seleccionada y	El cliente seleccione una lección aprendida que está guardada en

actualiza la información.	la bases de datos.
---------------------------	--------------------

2. El administrador no escoge ninguna lección aprendida

Entrada	Resultados	Condiciones
El cliente no selecciona ninguna lección aprendida	El sistema muestra el siguiente mensaje de error "Selecciona la lección aprendida que desea eliminar"	El cliente no seleccione ninguno de las lecciones aprendidas que están guardado en la bases de datos

CU Realizar Búsquedas

Sección Realizar Búsquedas (condición)

1. Selección correcta de la búsqueda.

Entrada	Resultados	Condiciones
El usuario selecciona la búsqueda que desea realizar.	El sistema realiza la búsqueda que desea el usuario y muestra la información.	El usuario selecciona la condición de búsqueda que desea realizar

2. Selección incorrecta correcta de la búsqueda

Entrada	Resultados	Condiciones
El usuario no selecciona la búsqueda que desea realizar.	El sistema muestra el siguiente mensaje de error "Debe seleccionar alguna condición de búsqueda"	El usuario no selecciona la condición de búsqueda que desea realizar

3. No existe lección aprendida que cumpla con la condición.

Entrada	Resultados	Condiciones
El usuario selecciona la búsqueda que desea realizar.	El sistema muestra el siguiente mensaje de error ""No existen lecciones aprendidas que cumplan esa condición."	Que no existan en la bases datos lecciones aprendidas que cumplan con esa condición.

GLOSARIO DE TERMINOS

Actor: Alguien o algo, fuera del sistema o negocio que interactúan con el sistema o negocio.

Artefacto: Una parte de la información que (1) es producida, modificada, o usada por un proceso, (2) define un área de responsabilidad, y (3) es candidata a ser tenido en cuenta para el control de la configuración. Un artefacto puede ser un modelo, un elemento de un modelo o un documento.

Información: Conjunto de datos experiencia.

Base de datos: Conjunto de datos organizados de modo tal que resulte fácil acceder a ellos, gestionarlos y actualizarlos.

Interfaz: Una colección de operaciones que se usan para especificar el servicio de una clase o de un componente. Un juego nombrado de operaciones que caracterizan la conducta de un elemento. La interfaz hombre-máquina es un canal comunicativo entre el usuario y el ordenador.

Internet: Es la mayor red de computadoras que existe, ya que se componen de infinitas redes pequeñas a lo largo del planeta, interconectadas entre sí. Esta permite a los usuarios compartir información de todo tipo.

Intranet: Es una misma adaptación de las mismas tecnologías que existe en Internet, para que sean utilizadas dentro de la red interna de una empresa u organización de forma tal que sus miembros puedan intercambiar información de todo tipo, utilizando el Web como interfaz común.

Método: Es una operación o función que está asociada a un objeto y que tiene permiso para manipular los datos del objeto.

Página web: Documento en formato ASCII que puede ser visualizado en distintos navegadores Web sin importar la plataforma. Este documento en su forma más sencilla permite presentar información textual, gráficos, tablas, etc. Permite el uso de enlaces o hipervínculos para saltar a otras páginas o referenciar a otros archivos.

Patrón: Solución común a un problema común de un determinado contexto.

Plataforma: Base, elemento de apoyo.

Proceso: Es un conjunto de actividades o eventos que se realizan o suceden con un determinado fin.

Proyecto: Esfuerzo de desarrollo para llevar un sistema a lo largo de un ciclo de vida.

Rol: Papel, cometido o función que tiene o desempeña que interpreta un actor.

Servidor: Es un ordenador que trata las peticiones de datos, el correo electrónico, la transferencia de ficheros, y otros servicios de red realizados por otros ordenadores (clientes).

Usuario: Humano que interactúa con un sistema.