



Facultad 2

SISTEMA EXPERTO PARA LA GENERACIÓN DE ESTRATEGIAS INDIVIDUALES DE SUPERACIÓN PEDAGÓGICA DEL CLAUSTRO DE LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Trabajo de diploma para optar por el título de ingeniero en ciencias informáticas

Autores: Harold García Alfonso
Yuniel Alamino Placeres

Tutores: Dr.C. Febe Angel Ciudad Ricardo
Ing. Walfrido Serrano Pérez

La Habana, junio de 2014
“Año 56 de la Revolución”

Harold

A la Revolución y a Fidel, por darme la oportunidad de ser mejor cada día.

A mi mamá y mi papá, por su amor incondicional, dedicación y sacrificio.

A mi hermano, por ser aquella persona que siempre está dispuesto ayudarme.

A mi novia, Yuliet, por todo su amor, comprensión y apoyo inquebrantable durante este último período de estudios intensos.

A mis primos y primas que siempre han estado atento a mis estudios, en especial a Daniel, Oscarito y Maggie.

A mis abuelos, abuelas, tías, especialmente a Maritza y Marisita.

A mi suegra, Idalmis, que me ha acogido como otro hijo.

A mi tutor y profesor, Febe, al que le debo gran parte de mi preparación como ingeniero y el espíritu de superación.

A mis amigos de la universidad y del barrio.

A mi amigo y compañero de tesis, Yuniel, ejemplo de consagración y sacrificio en la búsqueda de un objetivo.

A todos,

¡Muchas Gracias!

Yuniel

De corazón quiero agradecer:

A mi mamá, por brindarme todo su apoyo, cariño y amor desde el primer día de mi vida.

A mi papá, mi abuela y mi hermana por todo el esfuerzo y sacrificio que han hecho para lograr mi sueño.

A mi esposa Yisel por su apoyo incondicional durante todo este período y por traer al mundo a mi bebita hermosa.

A mis suegros que me han aceptado en su familia como otro hijo más. A mis cuñados por su apoyo durante todos estos días.

A mí cuñado Juan por estar presente y cuidar de mi hermana en los momentos que más necesitaba.

A la Revolución y al Comandante Fidel por brindarme la posibilidad de estudiar en esta especial Universidad.

A la Federación Estudiantil Universitaria por aceptarme como miembro y darme la posibilidad de dirigirla a distintos niveles. Al Secretariado de la FEU de la facultad 7 y al Consejo FEU-UCI.

Al Consejo de dirección de la Facultad 7 y en especial a mi Decana Niurys.

A mi tutor Febe, ejemplo de profesor, de profesional, de superación y de persona que me ha sabido guiar para ser un gran profesional.

A mi tutor Walfrido por su apoyo y guía durante la investigación.

A mi compañero de tesis y amigo, Harold, por su trabajo perfeccionista, por ser ejemplo de estudiante, de persona, de profesional y de amigo.

A mis amigos y compañeros de la vida que siempre estuvieron ahí para apoyarme.

A todos aquellos que de una forma u otra ayudaron en el desarrollo de esta investigación.

¡Muchas gracias!

Harold

A quienes me enseñaron a nadar contra la corriente, y a luchar para poder alcanzar mis metas: Mis padres, Ana y Antonio. ¡Mi triunfo es el de ustedes!
A mi hermano por confiar en mí y siempre darme todo el apoyo que necesité.
A toda mi familia que me han entregado su amor y apoyo siempre.
A Yuliet, mi novia, por entregarme todo su amor cada día y convertirse en una de las personas más importante en mi vida.

Yuniel

Le dedico el trabajo de diploma.

A mi mamá, por existir en todo momento en mi corazón y por haber depositado en mí todo su amor, fuerza, cariño y confianza y por convertirme en el hombre que soy.
A mi hermosa hija Vero, por darme la posibilidad de conocer lo hermoso de ser padre y por hacerme sacar las fuerzas para poder seguir adelante con esta investigación.
A mi hermana, mi esposa, mi papá y mi abuela por estar conmigo en todo momento, apoyándome y dándome fuerzas para realizar esta gran tarea, el trabajo de diploma, sin ellos, no hubiese sido posible convertirme en Ingeniero en Ciencias Informáticas.
¡Los quiero mucho y gracias a todos!

Declaramos ser autores de este trabajo y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los 26 días del mes de junio del año 2014.

Harold García Alfonso

Firma del Autor

Yuniel Alamino Placeres

Firma del Autor

Dr.C. Febe Angel Ciudad Ricardo

Firma del Tutor

Ing. Walfrido Serrano Pérez

Firma del Tutor

Febe Angel Ciudad Ricardo (fciudad@uci.cu): Graduado como Ingeniero Informático en el año 2004 por la Universidad de Holguín “Oscar Lucero Moya” (UHOLM) y el Instituto Superior Politécnico “José Antonio Echeverría” (CUJAE). Titulado como Máster en Informática Aplicada en el año 2007 por la Universidad de las Ciencias Informáticas (UCI) y obtuvo el grado científico de Doctor en Ciencias de la Educación – Especialidad Tecnología Educativa en el año 2012 por la Universidad de La Habana (UH). Imparte su docencia de pregrado en las disciplinas de Ingeniería y Gestión de Software, Metodología de la Investigación Científica y Formación Pedagógica. Es miembro de los claustros de las maestrías de Informática Aplicada, Informática Avanzada, Gestión de Proyectos y Educación a Distancia de la UCI. Desarrolla sus investigaciones en las temáticas de Ingeniería y Gestión de Software, con énfasis en el área del Software Educativo; así como en la Tecnología e Informática Educativas. Ha publicado diversos artículos científicos y ha participado en diferentes eventos nacionales e internacionales en estas áreas del conocimiento. Ha sido arquitecto, analista y líder de proyectos de desarrollo de software, jefe de departamento docente y asesor técnico – docente. Actualmente se desempeña como Director del Centro de Innovación y Calidad de la Educación (CICE) de la UCI.

Ing. Walfrido Serrano Pérez (wserrano@uci.cu): Graduado como Ingeniero en Ciencias Informáticas (3ra graduación, 2009). Trabaja en el CICE desde el 2009. Posee la categoría de Profesor Instructor. Se desempeña como profesor en la facultad 1 e imparte la asignatura de Programación Web y el curso optativo Elementos de Hardware. Trabajó además vinculado al proyecto de Gestión de Archivos de la misma facultad. Ha participado como expositor de los productos de la UCI en los eventos internacionales Universidad 2010 y Universidad 2012, así como en el salón de exposición de la UCI. También trabajó en la administración del Sistema de Encuestas UCI apoyando el proceso de caracterización de los estudiantes. Ha tutorado varias tesis en años anteriores. Por su ardua labor como profesor ha obtenido siempre la evaluación de excelente.

La generación de estrategias individuales de superación pedagógica del claustro de la Universidad de las Ciencias Informáticas constituye un proceso complejo, depende de las habilidades y experiencias adquiridas por los J.Dpto durante su elaboración. El objetivo de esta investigación se centró en el desarrollo de un sistema experto para la generación de la estrategia individual de superación pedagógica de un profesor.

Para el cumplimiento del objetivo, se establecieron los fundamentos teóricos de los principales conceptos del desarrollo de sistemas expertos para la toma de decisiones. Se analizaron las tendencias actuales de estos sistemas en el ámbito nacional e internacional. Además se realizó un estudio de las tecnologías, herramientas y lenguajes utilizados en el desarrollo de aplicaciones web, seleccionándose el uso del lenguaje Java, el framework Seam y como gestor de base de datos PostgreSQL.

Como resultado del proceso ingenieril, para la obtención de la documentación, se utilizó la metodología de desarrollo de software OpenUp, el lenguaje de modelado UML y la herramienta CASE Visual Paradigm. Se empleó la técnica de Razonamiento Basado en Casos (RBC) como forma de representar el conocimiento, se calculó la importancia de los rasgos mediante la teoría de testores típicos y como paradigma clasificatorio el algoritmo K-NN (*K-NearestNeighbour*).

La implementación del sistema experto proporcionó una herramienta fiable para la toma de decisiones, durante el proceso de generación de estrategias individuales de superación pedagógica. Para obtener un producto con tales características se realizaron las pruebas unitarias y la validación cruzada.

Palabras clave: estrategia de superación pedagógica, sistema experto, razonamiento basado en casos, testores típicos.

The generation of individual strategies for the pedagogical improvement of the teaching staff at the University of Informatics Science represents a complex process depending on the skills and experience acquired by Head of Departments while working with it. The objective of this research focused in the development of an expert system to generate an individual strategy to determine the needs of each professor training scheme.

To accomplish this objective, the theoretical foundations of the main concepts of the expert system development for decision-making were established. The current trends of these systems at the national and international level were analyzed. In addition a study of technologies, tools and languages utilized in web application development was carried out, being chosen for this work the use of Java language, the Seam framework and as database manager PostgresSQL.

As a result of the engineering process, for obtaining documentation, was used the OpenUp as software development methodology, the language modeling UML as well as the CASE tool Visual Paradigm. In addition the technique of Case Based Reasoning (CBR) was used as a way of representing knowledge; the importance of each feature was calculated using the theory of typical testors and as a qualifying paradigm the algorithm K-NN (*K-Nearest Neighbour*).

The implementation of the expert system provided a reliable tool for decision-making during the process of generating individual strategies of pedagogical improvement for the teaching staff. To obtain this product with such features, unit tests and cross-validation were performed.

Keywords: *case-based reasoning, expert system, strategy of pedagogical improvement, typical testors.*

INTRODUCCIÓN	1
CAPÍTULO 1. FUNDAMENTOS TEÓRICOS DE LOS SISTEMAS EXPERTOS	7
1.1. Inteligencia artificial	7
1.2. Sistemas Expertos	8
1.3. Sistema Basado en Casos. Fundamento de sus componentes	14
1.4. Sistemas Expertos para la toma de decisiones	22
1.5. Metodologías de desarrollo de software	25
1.6. Naturaleza del software	27
1.7. Tecnologías y Herramientas de desarrollo	28
CAPÍTULO 2. ANÁLISIS Y DISEÑO DE LA SOLUCIÓN PROPUESTA	36
2.1. Definición conceptual y operacional de las variables	36
2.2. Caracterización del objeto de estudio	37
2.3. Modelo de dominio	38
2.4. Requisitos funcionales y no funcionales de software	39
2.5. Modelo de casos de uso del sistema	41
2.6. Modelo del diseño	44
CAPÍTULO 3. IMPLEMENTACIÓN Y VALIDACIÓN DE LA SOLUCIÓN PROPUESTA	50
3.1. Especificación de los componentes del SBC propuesto	50
3.2. Diagrama de Despliegue	58
3.3. Tratamiento de excepciones	58
3.4. Seguridad	60
3.5. Principales funcionalidades	60
3.6. Estándar de codificación	63
3.7. Prueba y validación	64
CONCLUSIONES FINALES	69
RECOMENDACIONES	70
REFERENCIAS BIBLIOGRÁFICAS	71
GLOSARIO DE TÉRMINOS	77
ANEXOS	78

Tabla 1: Muestra Probabilística Estratificada de los J.Dpto.....	4
Tabla 2: Comparación entre experto humano y SE. [Tomado de (Tapia, 1996)].....	10
Tabla 3: Tabla de decisión que representa una BC. [Tomado de (Gutiérrez, y otros, 2002)].....	16
Tabla 4: Matriz de semejanza entre casos. [Tomado de (Bergmann, 1999)].....	21
Tabla 5: Comparación entre los SE para la toma de decisiones.	25
Tabla 6: Comparación entre las metodologías ágil y pesada. [Tomado de: (Letelier, y otros, 2006)]	26
Tabla 7: Comparación entre las principales metodologías ágiles. [Tomado de: (Letelier, y otros, 2006)] ...	27
Tabla 8: Comparación entre software WEB vs. Software de Escritorio. [Tomado de (OSD, 2005)].....	28
Tabla 9: Operacionalización de la variable dependiente.	36
Tabla 10: Descripción de los actores del sistema.....	41
Tabla 11: Descripción de casos de uso del sistema.....	42
Tabla 12: Descripción de los estereotipos web de las clases del diseño.....	46
Tabla 13: Categoría de los rasgos predictivos y rasgos objetivos.	51
Tabla 14: Dominio de definición de cada categoría.....	51
Tabla 15: Peso informacional inicial de cada rasgo.....	53
Tabla 16: Tratamiento de situaciones anómalas.	60
Tabla 17: Métodos para las pruebas unitarias.....	65
Tabla 18: Resultados de la validación cruzada.	67
Tabla 19: Resultados de los indicadores de fiabilidad.....	68
Tabla 20: Entrevista para los J.Dpto del claustro de la UCI.....	78
Tabla 21: CU: Generar estrategia individual de superación pedagógica.	78
Tabla 22: CU: Buscar caso.	81
Tabla 23: CU: Atender peticiones	84
Tabla 24: Descripción de la tabla tb_caso.....	86
Tabla 25: Descripción de la tabla tr_caso_caracteristica.....	86
Tabla 26: Descripción de la tabla tr_caso_curso.....	87

Figura 1: Áreas de estudio e investigación de la Inteligencia Artificial. [Tomado de (León, 2007)]	8
Figura 2: Arquitectura de los SE. [Tomado de (Ayala, 2006)]	10
Figura 3: Estructura de un Sistema Basado en Casos. [Modificado de (Moya, y otros, 2012)]	15
Figura 4: Estructura general de un caso. [Tomado de (Pérez, 1997)]	16
Figura 5: Diagrama de clases del dominio.	38
Figura 6: Diagrama de casos de uso del sistema.	42
Figura 7: DCD: Generar estrategia individual de superación pedagógica.	47
Figura 8: DCD: Buscar Caso.	48
Figura 9: DCD: Atender peticiones.	49
Figura 10: Modelo de datos.	52
Figura 11: Pseudocódigo para el clasificador K-vecinos próximos. [Modificado de (Moujahid, y otros, s.f)]	56
Figura 12: Diagrama de Despliegue.	58
Figura 13: Tratamiento de Excepciones.	59
Figura 14: Funcionalidad. Generar estrategia individual de superación pedagógica.	61
Figura 15: Funcionalidad. Crear nueva propuesta de superación pedagógica.	62
Figura 16: Funcionalidad. Atender peticiones.	63
Figura 17: Código generado con JUnit en Eclipse.	66
Figura 18: Pruebas unitarias con JUnit en Eclipse.	66
Figura 19: Validación cruzada de 5 iteraciones.	67

INTRODUCCIÓN

Las universidades destacan en el empeño por garantizar la educación continua y el postgrado, desarrollando habilidades que sirvan como base para la creación científica. Es por esto que las actividades de extensión, capacitación, grado y postgrado, se convierten en eslabones fundamentales del proceso de formación integral de estudiantes y profesores.

Es una prioridad en Cuba la superación de los docentes en cualquier nivel de enseñanza, con el objetivo de mantener la calidad de la educación. La educación superior con su órgano rector, el Ministerio de Educación Superior (MES), se plantea un reto mayor, elevar la superación pedagógica. Según (Ciudad, y otros, 2013), *«la mayoría de los profesionales que laboran en este nivel, en general poseen un conocimiento intuitivo y práctico heredado de sus antiguos profesores»*. Esta situación impacta negativamente en la relación superación - calidad de los docentes.

En un análisis realizado a diferentes diseños de superación pedagógica en el país por (Ciudad, y otros, 2013), se constata:

- Diferencias en el reconocimiento de necesidades en el área pedagógica.
- Variedad de contextos en que se desarrolla la preparación.
- Diversidad de objetivos y contenidos declarados en el área pedagógica para organizar la superación profesional y la formación académica.

La Universidad de las Ciencias Informáticas (UCI), perteneciente al MES, manifiesta similares condiciones. Hoy día, la superación pedagógica del claustro de la UCI, no está en correspondencia con las necesidades de la universidad, lo cual queda evidenciado en varios documentos, disponibles en IAI-UCI¹ (2010), IFAI-UCI-MES² (2010) y CD-UCI³ (2013). En estos documentos, también se constata que a pesar de las acciones de superación pedagógica realizadas, todavía resulta insuficiente la preparación pedagógica del claustro.

El Centro de Innovación y Calidad de la Educación (CICE) es el responsable de coordinar el diseño de acciones para elevar la calidad del proceso de formación integral de los estudiantes y profesores de la UCI.

¹ Informe de Autoevaluación Institucional de la UCI.

² Informe final de evaluación institucional del MES a la UCI.

³ Presentación del Rector al claustro UCI.

En dicho centro se elaboró en el año 2013 la Estrategia de Superación Pedagógica del Claustro de la UCI (ESPC-UCI). Esta tributa a la formación académica y profesional de los docentes, al establecer cómo elevar en gran medida la cultura profesional pedagógica en la universidad.

Según sus creadores, la ESPC-UCI consta de cuatro etapas que conforman un ciclo: diagnóstico, organización y planificación, ejecución y acreditación. En igual documento se describe que en la etapa de diagnóstico, se analizan los resultados de las evaluaciones de cada profesor, con el objetivo de generar las estrategias individuales de superación pedagógica de cada uno de ellos. Esta estrategia contiene tanto las áreas en las que cada profesor tiene una elevada preparación demostrada y aquellas en las que debe superarse pedagógicamente, con una propuesta de las formas de superación profesional pedagógica a las que debe incorporarse para mitigar las necesidades de superación.

Los jefes de departamento docente (J.Dpto) de cada una de las facultades y de otras áreas docentes de la universidad, son los máximos responsables del proceso de generación de la estrategia individual de superación pedagógica de cada profesor, que debe estar contenida en su plan de trabajo anual. Para esto es necesario recopilar información sobre el desempeño de un profesor. Un estudio realizado (ver **Anexo 1**) evidencia que las vías de obtención de conocimiento, comúnmente utilizadas, resultan insuficientes para garantizar la fiabilidad del proceso, entre las que destacan los controles a clases y la autoevaluación del profesor.

La generación de las estrategias individuales de superación pedagógica de los profesores resulta complejo, depende de las habilidades desarrolladas por los J.Dpto y las experiencias adquiridas durante su ejecución; destacando lo difícil que resulta sugerir un conjunto de acciones de postgrado en correspondencia con sus necesidades pedagógicas. Cada una de estas características, propias de los J.Dpto, limita el cúmulo de información a procesar sobre un profesor y a la vez diferencian la superación pedagógica del claustro de cada departamento.

Situación Problemática

A partir de lo expuesto anteriormente se puede concluir que, en la UCI, el proceso de generación de la estrategia individual de superación pedagógica de cada profesor se caracteriza por presentar:

- Insuficientes vías para la obtención del conocimiento.
- Identificación de las necesidades de superación pedagógica y acciones de postgrado limitadas por las habilidades y experiencias de los J.Dpto.

- Insuficiente homogeneidad en el diseño de las estrategias para profesores de diferentes departamentos con similares niveles de desarrollo de su cultura profesional pedagógica.

Esta situación impacta en la calidad de la superación pedagógica del claustro de la UCI, debido a que la generación de la estrategia individual de superación pedagógica de cada profesor, no constituye un proceso fiable que permita satisfacer sus necesidades de superación pedagógica. Lo anterior evidencia la necesidad de crear un mecanismo que reúna el conocimiento de los J.Dpto y de expertos en el tema de superación pedagógica; con el objetivo de: (1) procesar tanta información de un profesor como se necesite, (2) utilizar diversas vías de obtención del conocimiento sobre un profesor para el diseño de su superación pedagógica y (3) aplicar el mecanismo de forma homogénea en todos los departamentos de la universidad.

Por lo antes expuesto se identifica el **problema a resolver** ¿Cómo elevar la fiabilidad en el proceso de generación de estrategias individuales de superación pedagógica del claustro de la UCI?

En correspondencia con el problema planteado, el **objeto de estudio** lo constituye el proceso de generación de estrategias individuales de superación pedagógica del claustro de la UCI, así como el **campo de acción** está en los sistemas expertos para la definición de estrategias individuales de superación pedagógica.

El **objetivo** de la investigación consiste en desarrollar un sistema experto para elevar la fiabilidad del proceso de generación de estrategias individuales de superación pedagógica del claustro de la UCI.

Para dar cumplimiento al objetivo anteriormente planteado se definen las siguientes **tareas de investigación**:

1. Establecimiento de los fundamentos teórico-metodológicos acerca del desarrollo de sistemas expertos.
2. Caracterización del proceso de generación de estrategias individuales de superación pedagógica del claustro de la UCI.
3. Desarrollo de un sistema experto para elevar la fiabilidad del proceso de generación de estrategias individuales de superación pedagógica del claustro de la UCI.
4. Validación de la contribución lograda a través de la comprobación de la fiabilidad del sistema experto para la generación de estrategias individuales de superación pedagógica del claustro de la UCI.

Para guiar la investigación se plantea la siguiente **hipótesis**: la utilización de un sistema experto elevará la fiabilidad en el proceso de generación de estrategias individuales de superación pedagógica del claustro de la UCI.

Tomando como base lo planteado por (Hernández, y otros, 2006), la **población** de la investigación tuvo las siguientes características:

- Jefes de departamento docente de facultad (J.Dpto-DF) que tengan al menos un año de experiencia en el cargo.
- Jefes de departamento de centros productivos (J.Dpto-CP) que tengan al menos un año de experiencia en el cargo en centros de desarrollo con al menos 5 años de experiencia.
- Jefes de departamento docente central (J.Dpto-DC) que tengan al menos un año de experiencia en el cargo.

Siguiendo los criterios de (Hernández, y otros, 2006), se tomó una **muestra** probabilística estratificada dividida en los siguientes estratos: J.Dpto-DF, J.Dpto-CP y J.Dpto-DC.

La **técnica de muestro** utilizada fue aleatoria simple para cada estrato. Para garantizar que los resultados obtenidos en la muestra seleccionada puedan ser generalizados, se utilizó un **error estándar** de 0.05 (5%) para un **nivel de confianza** del 95%. Con una población de 37 J.Dpto, se necesitan 13 J.Dpto para conformar la muestra. Luego la muestra se divide por estratos de forma homogénea. El porcentaje de representatividad de la muestra con respecto a la población es del 35% y queda conformada de la siguiente manera:

Tabla 1: Muestra Probabilística Estratificada de los J.Dpto.

Estratos	J.Dpto	Población	Muestra
1	Docentes de Facultades	21	7
2	Centros Productivos	11	4
3	Docentes Centrales	5	2
Total		37	13

En el desarrollo de la investigación se utilizaron un conjunto de **métodos científicos**. Como parte de los **métodos teóricos** utilizados se encuentran:

- Histórico – lógico: para la determinación de antecedentes y tendencias actuales del objeto de estudio y el campo de acción.
- Análisis – síntesis e Inducción – deducción: para la determinación de las generalidades y especificidades en el objeto de estudio y el campo de acción; así como en la fundamentación

teórica de los sistemas expertos para la generación de estrategias individuales de superación pedagógica.

- Modelación: para sustituir el objeto de la investigación a través de abstracciones con vistas a explicar la realidad.
- Sistémico: para modelar el objeto mediante la determinación de sus componentes, así como las relaciones entre ellos. Estas relaciones determinan por un lado la estructura del objeto y por otro su dinámica.

Se utilizaron como **métodos empíricos**:

- Entrevista: con el objetivo de obtener información mediante la interacción con los entrevistados que conformaron la muestra.
- Análisis documental: con el objetivo de obtener información mediante la recolección y selección de documentos relacionados con el tema que se estudia.
- Experimento: para comprobar la validez de la hipótesis planteada.

El presente trabajo está compuesto por tres capítulos que se estructuran de la forma descrita a continuación:

En el **Capítulo 1. Fundamentos teóricos de los sistemas expertos** se describen los términos más importantes asociados al surgimiento de la Inteligencia Artificial, así como elementos esenciales de los Sistemas Expertos, teniendo en cuenta sus características, arquitectura y clasificación. Por otra parte se caracterizan los Sistemas Basados en Casos, fundamentalmente sus componentes. Además, se realiza un estudio de los sistemas expertos existentes a nivel nacional e internacional en el dominio que se investiga. Al final, se define la metodología para guiar el desarrollo de la investigación, así como las tecnologías y herramientas que serán utilizadas para la construcción del sistema.

El **Capítulo 2. Análisis y diseño de la solución propuesta** aborda la definición conceptual y operacional de las variables, así como la caracterización del objeto de estudio. Se identifican los requisitos funcionales y no funcionales, y se construye el modelo de dominio. Además se realiza el modelo de casos de uso del sistema, donde se describen los casos de uso identificados. Posteriormente se define el estilo arquitectónico, la estructura y las propiedades de los componentes para el desarrollo del sistema.

Por último en el **Capítulo 3. Implementación y validación de la solución propuesta** se describen los métodos y técnicas de los componentes del SBC propuesto. Se identifican los criterios de evaluación y acciones a realizar sobre un profesor. También se construye el modelo de datos, así como el diagrama de

despliegue. Además, se explican los estándares de diseño, estilos, codificación y tratamiento de errores utilizados y se exponen las principales funcionalidades desarrolladas en el sistema. Se concluye el capítulo con la realización de las pruebas y validación de la solución propuesta.

CAPÍTULO 1. FUNDAMENTOS TEÓRICOS DE LOS SISTEMAS EXPERTOS

1.1. Inteligencia artificial

El término Inteligencia Artificial (IA), según (Krishnamoorthy, y otros, 1996), fue acuñado formalmente en 1956 en el instituto de tecnología de Massachusetts por John McCarthy donde se celebró la conferencia de Dartmouth en Hanover (Estados Unidos). En este certamen McCarthy, Marvin Minsky, Nathaniel Rochester y Claude E. Shannon establecieron las bases de la inteligencia artificial como un campo independiente dentro de la informática.

La IA posee diversas definiciones. Autores como (Rich, y otros, 1994), la definen como *«la capacidad que tienen las máquinas para realizar tareas que, por el momento, los humanos hacen mejor»*. Esta definición es algo efímera debido a que hace referencia al momento histórico de la ciencia de la computación.

Otros autores coinciden con la definición de (Nebendah, 1988) y (Delgado, 1998), y la entienden cómo *«el campo de estudio que se enfoca en la explicación y emulación de la conducta inteligente en función de procesos computacionales basados en la experiencia y el conocimiento continuo del ambiente»*.

De igual forma, (Tapia, 1996) describe la IA como *«la rama de la ciencia de la computación que estudia la resolución de problemas no algorítmicos mediante el uso de cualquier técnica de computación disponible, sin tener en cuenta la forma de razonamiento subyacente a los métodos que se apliquen para lograr esa resolución»*. Para completar esta definición, algunas consideraciones de menor grado de formalidad, emitidas por diferentes investigadores de la IA que consideran otros puntos de vista:

- (Kurzweil, 1990), *«la IA es el arte de crear máquinas con capacidad de realizar funciones que realizadas por personas requieren de inteligencia»*.
- (Schalkoff, 1990), *«la IA es el campo de estudio que se enfoca a la explicación y emulación de la conducta inteligente en función de procesos computacionales»*.
- (Stubblefield, y otros, 1993), *«la IA es la rama de la ciencia de la computación que se ocupa de la automatización de la conducta inteligente»*.

Tomando como fundamento los criterios planteados, en la investigación se entiende por *Inteligencia Artificial* la definición dada por la Encyclopedia of Artificial Intelligence (2008) y que la define como *«el campo de la ciencia y la ingeniería que se ocupa de la comprensión, desde el punto de vista informático, de lo que denomina comúnmente comportamiento inteligente. También se ocupa de la creación de artefactos que exhiben este comportamiento»*. Esta definición es la más aceptada por la mayoría de los científicos e investigadores de la rama.

Según (León, 2007) «la IA se divide en campos de estudio, además se deriva en una extensa variedad de áreas, siendo las más relevantes el procesamiento de lenguaje natural, la simulación sensorial, la robótica y los sistemas expertos» (Figura 1). La simulación sensorial persigue la imitación de las capacidades o habilidades sensoriales humanas tales como vista, oído, habla y tacto. El estudio de la imitación del movimiento humano a través de robots, los cuales son creados con el fin de apoyar procesos mecánicos repetitivos que requieren gran precisión, constituye el objeto fundamental de la robótica. Los lenguajes naturales se enfocan en el diseño y desarrollo de software capaz de aceptar, interpretar y ejecutar instrucciones dadas por los usuarios en su lenguaje nativo. Los sistemas expertos constituyen igualmente un área dentro de la IA.

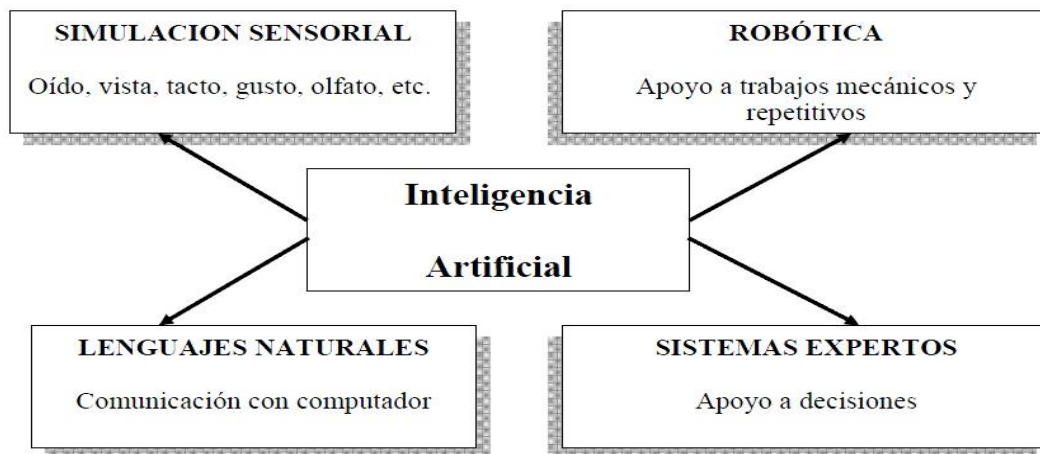


Figura 1: Áreas de estudio e investigación de la Inteligencia Artificial. [Tomado de (León, 2007)]

1.2. Sistemas Expertos

Los Sistemas Expertos (SE), de acuerdo con (León, 2007), «*simulan el proceso de aprendizaje, memorización, razonamiento, comunicación y acción, de un experto humano en cualquier rama de la ciencia. Estas características le permiten almacenar datos y conocimiento, obtener conclusiones lógicas, tomar decisiones, aprender de la experiencia y los datos existentes, comunicarse con expertos humanos, explicar el porqué de las decisiones tomadas y realizar acciones en consecuencia de lo anterior*». Autores como (Parsaye, y otros, 1990) reducen los SE al primer resultado operacional de la IA, pues «*logran resolver problemas a través del conocimiento y razonamiento de igual forma que lo hace el experto humano*».

Según (Gálvez, 1998) generalmente los términos Sistemas Basados en Conocimientos (SBCs) y SE se usan indistintamente, aunque algunos autores limitan el uso del término SE a aplicaciones donde el conocimiento al nivel de experto es requerido. (Ayala, 2006) refiere que «*los SE son empleados para ejecutar una variedad de tareas que en el pasado solamente podían realizarse por un número limitado de personas expertas. A través de la aplicación de las técnicas de IA, los SE captan el conocimiento básico que permite a una persona desempeñarse como un experto frente a problemas complicados*».

(León, 2007) define un SE como un «*programa de computadora interactivo que contiene la experiencia, conocimiento y habilidad propios de una persona o grupo de personas especialistas de un área particular del conocimiento*». Además resuelve problemas específicos de esa área de manera inteligente y satisfactoria. Por otra parte (Giarratano, y otros, 2005) precisa que un SBCs es aquel en el que aparece representado el conocimiento de un dominio determinado, de tal forma que dicha representación sea procesable por un programa informático. Un SBCs al que se le incorpora conocimiento proveniente de expertos en un dominio se le conoce como SE.

En resumen, cuando el conocimiento se extrae del experto para el desarrollo de un sistema que simule su comportamiento e intente emitir respuestas a determinadas situaciones, se constata la presencia de un SE. De igual forma los autores defienden que, es un programa de ordenador que intenta imitar e incluso superar a un experto humano en un ámbito determinado, capaz de facilitar recomendaciones deducidas a partir de su conocimiento.

Según plantea (Ayala, 2006), entre las características más relevantes de los SE, que los distingue de la mayoría de las aplicaciones tradicionales de la computación, se encuentra su capacidad para enfrentar problemas que constituyen un reto del mundo real, por medio de la aplicación de procesos que reflejan el juicio y la intuición humana. Autores como (Rolston, 1998) y (Gálvez, 1998) plantean que un SE debe cumplir con las siguientes características:

- Conocimiento específico del dominio.
- Naturaleza heurística, en lugar de algorítmica, del conocimiento empleado.
- Distinción entre conocimiento y estrategia de control.
- Capacidad de procesar símbolos, así como explicar su propio razonamiento.
- Alto nivel de experticia.

En la Tabla 2 se muestra una comparación, respecto a la experiencia, entre un experto humano y un SE, donde se puede constatar las facilidades y ventajas del segundo con respecto al primer elemento.

Tabla 2: Comparación entre experto humano y SE. [Tomado de (Tapia, 1996)]

Experto Humano	Sistema Experto
Tiempo limitado	Perdura en el tiempo
Difícil de transferir	Fácil de transferir
Difícil de documentar	Fácil de documentar
Impredecible	Consistente
Alto costo	Alcanzable
Experiencia personal	Casos similares
Conocimiento del sentido común	Conocimiento técnico

Como se observa, el SE resulta ser más conveniente de usar y fácil de entender. El proceso de desarrollo de un SE también tiene un beneficio colateral, ya que el conocimiento almacenado a partir de los especialistas humanos es claramente detallado y examinado. Asimismo, se dispone explícitamente del conocimiento, en vez de tenerlo implícito en la mente del especialista, puede examinarse para corregirlo, darle más consistencia y completarlo. El conocimiento puede entonces ajustarse, lo que aumenta su calidad.

Autores como (Ayala, 2006) y (León, 2007) plantean que «no existe una estructura de SE común, aunque la mayoría de los SE tienen componentes básicos como son la base de conocimientos, la máquina de inferencia, la base de datos y la interfaz de usuario». Otros autores agregan además, un módulo de explicación y un módulo de adquisición del conocimiento. En la Figura 2 se muestran los componentes antes mencionados.

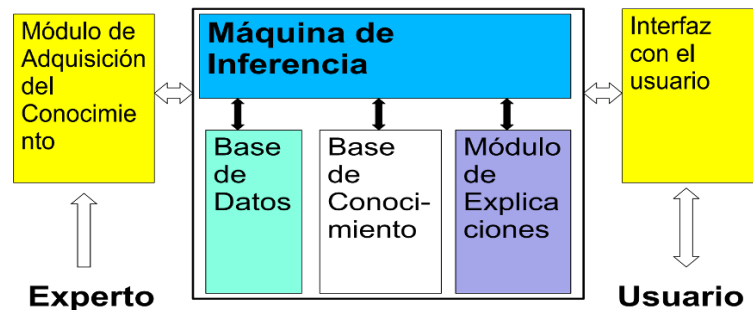


Figura 2: Arquitectura de los SE. [Tomado de (Ayala, 2006)]

Base de conocimiento: es la parte del SE que contiene el conocimiento sobre el dominio. Está escrita en un lenguaje específico de representación del conocimiento que domina. Obtiene el conocimiento del experto y lo codifica, es decir el experto puede definir su propio vocabulario técnico.

Interfaz con el usuario: es el subsistema responsable de interactuar con el mismo. Permite al SE realizar preguntas al usuario y viceversa. Ofrece explicaciones de por qué hace una pregunta.

Máquina de inferencia: conocida como motor, contempla un conjunto de algoritmos que posibilitan el análisis de la información contenida en la base de conocimiento. Es decir, se encarga de representar los mecanismos de solución, implementar los criterios de búsqueda y fundamentar la inferencia. También es el motor es considerado el núcleo del SE donde descansa la parte procedural.

Base de Datos: contiene toda la información dada para el usuario sobre el problema actual y todas las conclusiones que el SE ha inferido. Es decir, contiene los hechos sobre un problema y en otro espacio, el usuario actualiza la información en la base de hechos, el SE equilibra esta información y deduce nuevos hechos.

Subsistema de explicación: puede proporcionar al usuario una explicación de por qué está haciendo una pregunta y cómo ha llegado a una conclusión. Se utiliza para la detección de errores y para la transparencia del sistema.

Módulo de adquisición del conocimiento: almacena solamente la nueva información para la base de datos y es el encargado de controlar que esta información no sea redundante.

El análisis documental realizado, expone diferentes puntos de vista de cómo clasificar los SE. Se defiende la clasificación utilizada por (Rojas, y otros, 2002).

Razonamiento Basado en Reglas

El Razonamiento Basado en Reglas o también conocido como Sistema Basado en Reglas (SBR), funciona mediante la aplicación de reglas, comparación de resultados y aplicación de las nuevas reglas basadas en una situación modificada. La aplicación de estas reglas se debe fundamentalmente a la expresividad para representar un rango útil de reglas de inferencia y su amplia capacidad de explicación. Se utilizan para expresar conocimiento deductivo, orientado a objetivos y relaciones causales. Generalmente son convenientes cuando el conocimiento a representar aparece naturalmente en forma de reglas, cuando el control del programa es muy complejo o cuando se espera que el programa sea modificado reiteradamente.

Las reglas utilizan un formato *IF - THEN* para representar el conocimiento, la parte *IF* de una regla es una condición en la que algunos autores la llaman premisa o antecedente, y la parte *THEN* de la regla llamada acción, conclusión o consecuente permite inferir un conjunto de hechos nuevos si se verifican las condiciones establecidas en la parte *IF*.

El proceso de solución de problemas en un SBR, es crear una cadena de inferencias que constituye un camino entre la definición del problema y su solución. Esta cadena de inferencias puede construirse por dos vías (direcciones de búsqueda):

- Seleccionar una conclusión posible y tratar de probar su validez buscando evidencias que la soporten (encadenamiento hacia atrás).
- Comenzar con todos los datos conocidos y progresar hacia la conclusión (encadenamiento hacia delante).

El **encadenamiento hacia atrás** tiene como objetivo la demostración de una hipótesis. Su desarrollo se presenta como sigue:

- Se define una hipótesis, es decir, un valor a alcanzar en una variable de salida del sistema.
- Si la hipótesis se basa en un hecho comprendido en la base de conocimiento, se finaliza el proceso. En caso contrario se continúa al paso 3.
- Se busca el subconjunto de reglas de la base de conocimiento cuyos consecuentes coincidan con la hipótesis.
- Se establecen los antecedentes de las reglas del subconjunto seleccionado como nuevas hipótesis a demostrar y se vuelve al paso 2.

Este tipo de inferencia establece una cadena de reglas que prueba la veracidad de una hipótesis.

El **encadenamiento hacia delante** parte de la observación de hechos en las variables de entrada para, mediante el encadenamiento de reglas, alcanzar un hecho de salida deseado. El proceso es el siguiente:

- Se define el hecho a alcanzar, es decir, las variables de salida del sistema cuyo valor se desea inferir.
- Un conjunto de hechos relativo a las variables de entrada es observado por el sistema, es decir, llega una entrada al sistema.
- Se busca el subconjunto de reglas de la base de conocimiento cuyos antecedentes son satisfechos por los hechos observados.
- Si el subconjunto está vacío, se finaliza el proceso. En otro caso se continúa al paso 5.

- El subconjunto de reglas seleccionado se activa y da lugar a un número de hechos nuevos igual al tamaño del subconjunto.
- La base de hechos se actualiza con los nuevos hechos.
- Si se ha alcanzado el hecho de salida deseado se finaliza el proceso, en otro caso, se vuelve al paso 3.

Razonamiento Basado en Redes Bayesianas

Una red bayesiana es un modelo probabilístico multivariado que relaciona un conjunto de variables aleatorias mediante un grafo dirigido, que indica explícitamente influencia causal. Por su parte, gracias a su motor de actualización de probabilidades, las redes bayesianas son una herramienta extremadamente útil en la estimación de probabilidades, pueden llegar a trabajar en la búsqueda de datos que representen un valor (porcentaje de estudiantes aprobados en exámenes, la cantidad de misiles certeros al blanco en el primer intento, una muestra de las personas consumidoras de carbohidratos en exceso, etc.). Además son usadas en distintos aplicaciones tales como:

- Prevención del fraude.
- Prevención del abandono de clientes.
- Blanqueo de dinero.
- Estimación de cuentas.
- Representación de ventas.
- Porcentajes, muestras, comparaciones, entre muchas otras aplicaciones.

Razonamiento Basado en Casos

El Razonamiento Basado en Casos (RBC) o también conocidos como Sistema Basado en Casos está inspirado, fundamentalmente, en el papel que juega el recuerdo en el razonamiento humano. Es una técnica de IA muy usada y significa razonar sobre la base de experiencias o "casos" previos. El RBC indica un método para la solución de un nuevo problema, donde la misma surge a partir de las soluciones anteriores de un conjunto de problemas del dominio de aplicación.

El RBC es una manera de razonar haciendo ciertas similitudes, por tanto se argumenta que es más que un método de solución de problemas cotidianos; para hacerlo necesita de un proceso el cual se resume en:

- **Recordar o Recuperar:** dado un determinado problema, se recuerdan los casos relevantes que pueden solucionarlo. Un caso consiste en un problema, una solución y la explicación sobre como la solución fue llevada a cabo.
- **Reutilizar o Adaptar:** adaptar la solución del problema anterior a uno nuevo. Se reutilizan casos anteriores con características similares que puedan ser utilizados como base para adaptarlo a situaciones específicas del nuevo problema.
- **Revisar:** una vez adaptado el problema, se valida el nuevo caso con el usuario o en una simulación, en caso de no resolver el problema se modifica el recuperador de casos y se inicia de nuevo el proceso.
- **Retener:** después de que la solución ha sido adaptada satisfactoriamente para solucionar el problema dado, se almacena la experiencia resultante como un nuevo caso en la memoria.

En resumen, los tipos de SE responden a distintas situaciones, en dependencia del tipo de problema y del conocimiento adquirido sobre el dominio en cuestión. Luego de un estudio a las principales formas de representación del conocimiento de los SE en la actualidad, se decide desarrollar en la investigación un sistema basado en casos atendiendo a las siguientes características:

- Al experto le resulta más fácil proporcionar ejemplos, que crear reglas de comportamiento.
- Se puede obtener un nuevo caso cada vez que se resuelve un problema, se indica su resultado y se explica. De esta manera se garantiza que el SE incorpore nuevo conocimiento del dominio.
- Dada las características del problema a resolver, es posible definir con facilidad una función de comparación para recuperar los casos similares y adaptarlos de manera efectiva.
- Los casos mantienen su vigencia durante bastante tiempo, es decir, los problemas tienden a repetirse.

1.3. Sistema Basado en Casos. Fundamento de sus componentes

En Sistema Basado en Casos (SBC) el conocimiento se representa a través de una base de casos (BC). Cuando surge un nuevo problema a resolver, éste es descrito para el módulo de recuperación (Recuperar), que realiza una búsqueda en la BC y encuentra problemas o casos similares. Estos problemas o casos similares son recuperados (soluciones similares) y enviados al módulo de adaptación (Adaptar), donde se analizan para construir la solución del nuevo problema. Una vez hallada la solución, se revisa por el experto (Revisar), el cual determina si se almacena o no (Retener) en la BC. Como resultado del proceso se obtiene un nuevo caso.

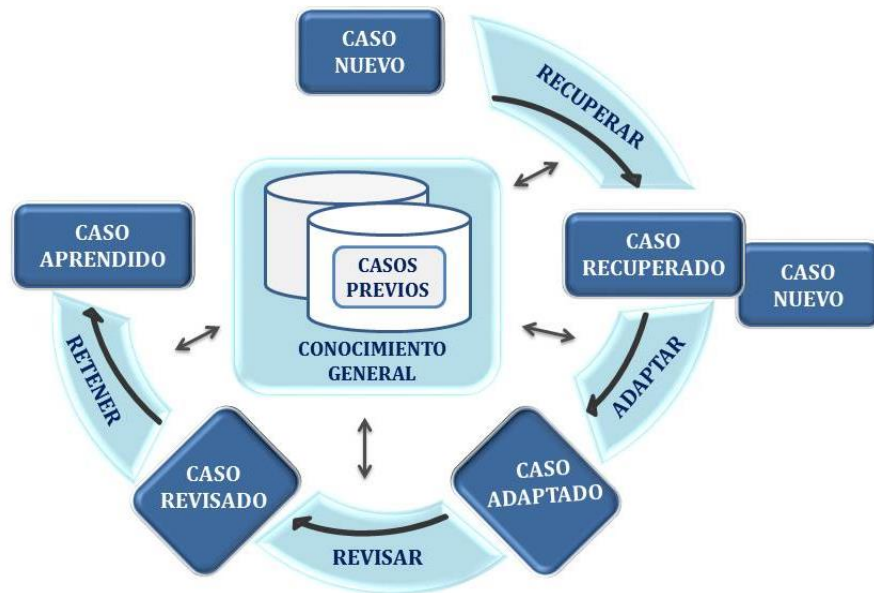


Figura 3: Estructura de un Sistema Basado en Casos. [Modificado de (Moya, y otros, 2012)]

Base de casos

Según (Agudo, 2002) «el componente principal de un SBC es la BC, la cual constituye la memoria del sistema y está formada por una colección de casos. Para comprender la BC, es necesario establecer o definir un caso». (Kolodner, 1996), define **caso** como «una pieza de conocimiento contextualizado que representa una experiencia, en la que se muestra una lección fundamental para consolidar los objetivos de un razonamiento». Por su parte, (Pérez, 1997), lo define como «la descripción detallada de una experiencia del pasado sobre una situación particular, formada por la descripción del problema y la solución tomada para resolverlo».

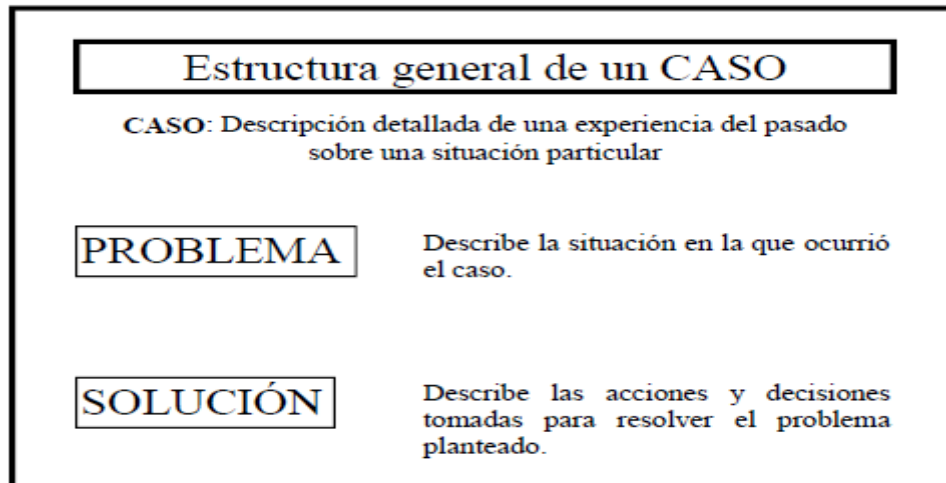


Figura 4: Estructura general de un caso. [Tomado de (Pérez, 1997)]

La BC está formada por un conjunto de casos y puede ser representada a través de una tabla de decisión, en la cual las columnas son etiquetadas por variables que representan los rasgos predictivos y los rasgos objetivos (decisiones) y las filas representan los casos.

La tabla siguiente muestra un ejemplo de una BC. Los casos están representados por O_1, \dots, O_m , los rasgos predictivos por X_1, X_2, \dots, X_n y los rasgos objetivos (decisiones) por Y_1, \dots, Y_p .

Tabla 3: Tabla de decisión que representa una BC. [Tomado de (Gutiérrez, y otros, 2002)]

Casos	X_1	...	X_n	Y_1	...	Y_p
O_1	$X_1(O_1)$...	$X_n(O_1)$	$Y_1(O_1)$...	$Y_p(O_1)$
...
O_m	$X_1(O_m)$...	$X_n(O_m)$	$Y_1(O_m)$...	$Y_p(O_m)$

Módulo de Recuperación

El módulo de recuperación consta de dos etapas fundamentales: acceder a la BC y recuperar los casos más similares al nuevo caso. Esta última etapa a su vez incluye el cálculo de la semejanza entre los casos de la BC y el nuevo caso.

Etapas: acceso a la base de casos

Para lograr acceso rápido y eficiente a la BC es necesario definir el modelo de memoria para organizar los casos. Dado que los casos constituyen el elemento principal del SBC, la manera de acceder a ellos repercutirá directamente en el rendimiento del sistema. En la selección del tipo de memoria que

organizará la BC, (Pérez, 2002) plantea tener en cuenta aspectos básicos como: el tamaño deseado de la BC, los requisitos que se imponen en el dominio específico de trabajo y la inserción eficiente de nuevos casos. El propio autor define dos tipos de memoria: plana y jerárquica.

- **Memoria plana:** se presentan los casos completos de forma secuencial utilizando la indexación manual. Los casos se almacenan secuencialmente en una lista simple, un arreglo o un fichero. Para lograr una recuperación eficiente, se indexan los casos de la base. Los índices se eligen para representar los aspectos importantes del caso y la recuperación involucra comparar las características del nuevo problema, con cada caso de la BC.

Según (Amandi, 2012), esta estructura tiene el inconveniente de que *«la búsqueda de casos es menos eficiente, por lo que no es recomendable utilizarla en sistemas con una BC de gran tamaño y necesiten respuesta en tiempo real»*. Por otra parte, la propia autora agrega que *«la inserción de nuevos casos es muy sencilla, ya que basta incluir un nuevo registro con el nuevo caso, respetando el método de indexación que se esté utilizando»*.

- **Memoria jerárquica:** se utilizan representaciones en forma de árbol, en los que cada nodo interior representa un atributo del caso y en las hojas se almacenan las soluciones a los mismos. Cada recorrido desde la raíz hasta las hojas del árbol representa un caso completo. Para obtener mejores soluciones, los atributos más discriminantes deben estar almacenados en los niveles superiores del árbol, empezando desde la raíz.

«La gran ventaja de este tipo de almacenamiento, [según (Amandi, 2012)], es la eficiencia en la búsqueda, pero a cambio se sacrifica la sencillez de inserción de nuevos casos, se dificulta mantener el árbol en forma óptima cuando se agregan nuevos casos y el consumo de memoria aumenta considerablemente».

El modelo de memoria jerárquica está en posición ventajosa respecto modelo memoria plana, teniendo en cuenta rendimiento y eficiencia, sin embargo no siempre es rentable utilizarlo; por su propia naturaleza nunca se garantiza que el caso recuperado sea el óptimo, pues no se recorre toda la BC. Por el contrario la memoria plana hace un recorrido exhaustivo por toda la BC y garantiza que el caso recuperado sea el óptimo, aunque el costo computacional aumente considerablemente. Para la propuesta se selecciona la memoria plana, ya que la BC es de tamaño medio, por lo que analizar todos los casos no será muy costoso; además se garantizan los elementos descritos sobre el modelo de memoria.

Etapa: recuperar casos similares

La selección de un caso semejante a un nuevo problema entre un grupo de casos almacenados, exige el uso de una función que mida cuan semejantes son los casos analizados. Una vez definida y calculada la función de semejanza se procede a la elección de los casos más similares.

Función de semejanza

Para el cálculo de la función de semejanza, es necesario tener en cuenta dos aspectos fundamentales. El primero refiere definir la función de comparación de similitud entre casos, este viene dado por el tipo de variable que toman los valores de los rasgos predictivos. El segundo aspecto, se trata de determinar cuáles características tienen mayor relevancia (peso informacional) en el dominio del problema. La función de semejanza proporciona como resultado un valor entre cero y uno, que expresa el grado de semejanza entre el problema y cada uno de los casos contenidos en la BC. El modelo matemático que sustenta lo descrito, según (Uddin, y otros, 2010), se muestra a continuación:

Función de semejanza entre un nuevo problema a resolver O_0 y un caso O_t de la base.

$$\beta(O_0, O_t) = \frac{\sum_{i=1}^n p_i \cdot \delta_i(O_0, O_t)}{\sum_{i=1}^n p_i}$$

Donde:

n: Número de rasgos predictivos.

p_i : Peso informacional o relevancia del rasgo i.

$\delta_i(O_0, O_t)$: Función de comparación entre los casos O_0 y O_t atendiendo al rasgo i. Esta función puede estar definida de diferentes formas, por ejemplo:

$$\delta_i(O_0, O_t) = \begin{cases} 1 & \text{si } x_i(O_0) = x_i(O_t) \\ 0 & \text{e.o.c} \end{cases}$$

$$\delta_i(O_0, O_t) = \begin{cases} 1 & \text{si } |x_i(O_0) - x_i(O_t)| < \varepsilon \\ 0 & \text{e.o.c} \end{cases}$$

ε : Valor umbral definido por el usuario.

$$\delta_i(O, O_t) = 1 - \frac{|x_i(O) - x_i(O_t)|}{\max_i - \min_i}$$

\max_i, \min_i : Valores máximo y mínimo respectivamente que alcanza el rasgo i .

Determinar cuál o cuáles serán las funciones de comparación, contenidas en la función de semejanza, depende de los valores que tomen los rasgos predictivos. En el ejemplo anterior se mostraron las funciones de comparación más comunes para valores numéricos registradas en la literatura.

Una vez que se define la función de comparación, es necesario asignarle un peso aquellos rasgos que a los expertos por sus experiencias consideran más importantes. La literatura consultada (Álvarez, y otros, 2012), (Ruiz-Shulcloper, 2009) y otros, evidencia principalmente dos tipos de enfoques para la asignación de pesos. El primero de ellos se realiza de acuerdo al criterio de expertos. Para ello se necesita tener bien claro el dominio de aplicación y contar con varios expertos en el tema, para identificar los rasgos más importantes, menos importantes e incluso aquellos que carecen de importancia a la hora de tomar una decisión. Este escenario resulta idóneo cuando se cumplen las restricciones anteriores, no obstante su aplicación en algunos contextos, por lo complejo que resulta, es muy difícil. El segundo enfoque manifiesta otorgar la responsabilidad al mecanismo de inferencia (MI) de asignarle pesos a cada rasgo.

Existen varias formas para el cálculo de los pesos, en un estudio sobre el tema, (Ruiz-Shulcloper, 2009) y (Zhuravliov, y otros, 1972) destacan el cálculo basado en la teoría de testores. Un testor típico, según los propios autores, «es un conjunto de columnas (rasgos) de la matriz de aprendizaje, para las cuales se cumple que no existen filas iguales en clases diferentes, pero además si se elimina cualquiera de estas columnas las restantes perderán esta propiedad». De esta manera se llega a un conjunto irreducible de rasgos, donde cada rasgo resulta imprescindible para mantener las diferencias entre las clases. Es natural suponer, que si un rasgo aparece en muchos conjuntos irreducibles, resulta más difícil prescindir de él para describir de manera diferenciada las clases. Sobre la base de esta idea, Zhuravliov formula su definición de *peso informacional de un rasgo*, como la frecuencia relativa de aparición de ese rasgo en la familia de los testores típicos.

$$P(x_i) = \frac{\tau(i)}{\tau}, \text{ para } i=1, \dots, n, x_i \in R.$$

Donde τ es el número de testores típicos de la matriz de aprendizaje y $\tau(i)$ el número de testores típicos en los que aparece la columna correspondiente al rasgo X_i . Sin embargo, la expresión anterior propuesta por

Zhuravliov para la evaluación del peso informacional de los rasgos, considera más importante los rasgos que más veces aparecen en la familia de los testores típicos, sin tener en cuenta las longitudes de los testores que contienen al rasgo en cuestión. De acuerdo con ello, (Ruiz-Shulcloper, 2009) define que el peso informacional de un rasgo esta dado por la suma de la frecuencia relativa y la longitud del testor que contiene el rasgo. La longitud se determina de la siguiente manera:

$$L(x) = \frac{\sum_{t \in \Psi^*(x)} \frac{1}{|t|}}{|\Psi^*(x)|}$$

Una vez conocido los valores $P(x)$ y $L(x)$ la fórmula propuesta por Shulcloper resulta de la forma siguiente:

$$D(x) = \alpha P(x) + \beta L(x) \text{ con } \alpha, \beta > 0 \text{ y } \alpha + \beta = 1$$

Donde las variables α y β son dos parámetros que ponderan la participación o influencia de $P(x)$ y de $L(x)$ respectivamente en D . Esto es, la importancia que se le concede a la frecuencia de aparición y a la longitud de los testores.

Elección de los casos más similares

Luego de conocer la similitud entre el caso a resolver y cada caso de la BC, es necesario determinar cuál o cuáles de esos casos semejantes son los que se recuperarán. Se puede simplemente recuperar todos los casos, pero ello no garantiza que la solución brindada sea la mejor. Para eliminar esta desventaja, (Ruiz-Shulcloper, 2009) plantea, determinar un umbral de recuperación, que representa el mínimo valor de semejanza establecido para un caso. Con este valor se garantizará, al menos, la recuperación de los casos más similares en la BC.

El propio autor plantea que el proceso de determinar el umbral puede ser realizado de dos maneras; determinado por el usuario o el propio MI puede calcularlo. Si el umbral lo define el usuario, como consecuencia se obtiene un valor elevado, por tanto los casos recuperados tendrán un alto nivel de semejanza, pero como inconveniente si ninguno de los casos sobrepasa este valor, el usuario no obtiene una respuesta al problema planteado. Por otra parte, esta situación se resuelve si el valor del umbral es calculado por el MI, ya que garantiza al menos una solución al usuario similar al caso nuevo aceptable. Este cálculo se basa en una matriz cuadrada donde las filas y las columnas están representadas por los casos almacenados en la BC; además en la intersección se encuentra el valor de semejanza (β), como se muestra en la tabla 4.

Tabla 4: Matriz de semejanza entre casos. [Tomado de (Bergmann, 1999)]

	O ₁	O ₂	O ₃	O _n
O ₁	$\beta (O_1, O_1)$	$\beta (O_1, O_2)$	$\beta (O_1, O_3)$	$\beta (O_1, O_n)$
O ₂	$\beta (O_2, O_1)$	$\beta (O_2, O_2)$	$\beta (O_2, O_3)$	$\beta (O_2, O_n)$
O ₃	$\beta (O_3, O_1)$	$\beta (O_3, O_2)$	$\beta (O_3, O_3)$	$\beta (O_3, O_n)$
O _n	$\beta (O_n, O_1)$	$\beta (O_n, O_2)$	$\beta (O_n, O_3)$	$\beta (O_n, O_n)$

Una vez construida la matriz, mediante el cálculo de μ (Ruiz-Shulcloper, 2009) se obtiene el valor umbral que depende de los valores de semejanza entre los casos contenidos en la matriz. Donde m es el número de casos y $\beta (O_i, O_j)$ representa la semejanza entre el caso i y el caso j .

$$\mu = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^m \beta(O_i, O_j)$$

Módulo de Adaptación

La etapa de adaptación es el proceso de transformar la(s) solución(es) recuperada(s), en una solución apropiada para resolver el problema. Según (Navarro, 2011), «*la adaptación es una etapa importante dentro del ciclo de RBC, al añadir inteligencia a lo que de otra manera sería un simple modelo de emparejamientos*». La propia autora define dos formas generales de realizar la adaptación:

- Mediante la sustitución de aquellos valores que aparecen en el(los) caso(s) recuperado(s) de la memoria, por aquellos valores del caso actual, de esta forma la nueva solución hace uso de la situación actual a resolver.
- Mediante la aplicación del mismo conjunto de procedimientos, reglas o inferencias utilizados como solución del caso recuperado, al caso para resolver.

La adaptación que se propone fundamenta su aplicación en la segunda forma descrita. Si en el proceso de recuperación se obtienen un conjunto de casos candidatos, se puede, seleccionar uno de ellos con algún criterio específico o, hacer uso de las soluciones conjuntas en la búsqueda de una solución apropiada para el caso actual. En la propuesta se utiliza un procedimiento particular para la definición de un criterio específico, la cual será abordada en el epígrafe 1 del capítulo 3.

Módulo de Revisión

Los SBC, como se ha planteado anteriormente aprenden de las experiencias y de los nuevos casos que aporten un enriquecimiento del sistema. Así, después de adaptar una posible solución esta puede ser correcta o no, si es correcta la nueva solución será almacenada en el sistema, pero si la solución no ha

sido satisfactoria entonces se debe revisar. La revisión de casos, según (Giménez, 2006), comprende básicamente dos fases:

- **Evaluar la solución:** decidir si la solución dada es la correcta al problema planteado. Esta fase normalmente se realizará por algún método externo al SBC, como por ejemplo el experto.
- **Reparar los fallos:** Si no es correcta la solución, se detectan y corrigen los fallos modificando el mecanismo de inferencia.

La propia autora plantea que esta etapa posee pocas referencias por la mayoría de los autores, centrando el sistema a las etapas de recuperación y adaptación. Para el desarrollo de la propuesta solo se tendrá en cuenta la primera fase, además se desarrolló un algoritmo para ayudar al experto en la revisión del caso adaptado. En futuras versiones del sistema se desarrollará la fase de reparar los fallos.

Módulo de Retención

Una de las principales características de un SBC es la capacidad de recordar nuevos casos y su solución aplicada, para ello es fundamental poder retener estos casos en la BC. Asimismo, la BC crecerá y se enriquecerá a partir de la solución de problemas basados en la experiencia. La forma de estructuración de la BC y las políticas de retención del SBC facilitarán el buen funcionamiento. Por ello el primer problema a tratar por un sistema de aprendizaje es, decidir qué casos se aprenden. La eficiencia de un SBC se puede degradar cuando el número de casos crece excesivamente y por tanto, se debe evitar incluir casos que no aporten información nueva al sistema. El rango de posibilidades se encuentra desde los sistemas que, de forma autónoma deciden qué casos deben incluir hasta los que delegan esta posibilidad en el mismo experto. En la solución propuesta el experto decidirá qué caso se almacena, pero de igual forma que en el módulo de revisión, se desarrollará un algoritmo de apoyo a la etapa de retención de casos.

1.4. Sistemas Expertos para la toma de decisiones

Hoy día *«la utilización de SE para la toma de decisiones, [como plantea (Pinochet Olave, 2003)], es un hecho generalizado, siendo múltiples las empresas y sectores productivos que los tienen incorporados dentro de su operación cotidiana»*. Con el objetivo de fundamentar las tendencias de tales SE, se realiza un análisis de su desarrollo a nivel internacional y nacional, identificándose los exponentes descritos a continuación.

Sistema de RBC para el Soporte a la Toma de Decisiones (1)

Los autores de este artículo (Laza, y otros, 2012) muestran cómo el agente consejero del Sistema de Ayuda a la Elección de Asignaturas Optativas (SAEAO), empleando un sistema de RBC puede aconsejar

a estudiantes de forma autónoma. Finalmente muestra los resultados obtenidos con él y los compara con los resultados arrojados por una red de funciones de base radial (RBF – Radial Basis Functions). Se presenta un método para la toma de decisiones sobre el problema expuesto anteriormente. La utilización del RBC muestra una mayor precisión a la hora de seleccionar el conjunto de asignaturas que la opción basada exclusivamente en la aplicación de una red neuronal artificial.

Sistema experto para la enseñanza: ENS-AI (2)

Según (Barraso, 1994), ENS-AI, acrónimo de ENSeñanza - Inteligencia Artificial es un sistema de soporte y guía para la práctica de la educación en España. Desarrollado en el Instituto de Investigación en Inteligencia Artificial del Consejo Superior de Investigaciones Científicas, utilizando el lenguaje Milord II. Es un sistema basado en reglas; esta facilidad permite representar pequeños dominios de conocimiento estructurados, como en el caso de la educación.

El sistema posee las siguientes entradas de datos fundamentales (1) el alumno con sus características individuales y sociales, (2) se entorno de enseñanza-aprendizaje con las características externas (entorno familiar y social) e internas (entorno escolar) y (3) su profesor con sus características de formación, habilidades, destreza, actitudes y expectativas. Con todos estos datos realiza el diagnóstico psicopedagógico del alumno, así como una valoración de las técnicas y habilidades del profesor y del entorno de enseñanza-aprendizaje. Como resultado se obtiene la recomendación de una intervención pedagógica (asociada a un valor de certeza), que le permite enfrentarse a la solución del problema diagnosticado.

Sistema experto de orientación vocacional – profesional (3)

Sistema desarrollado en la Universidad Complutense de Madrid, España, que según (Martin, 1996) a partir del conocimiento de las aptitudes, personalidad e intereses del estudiante el sistema realiza un diagnóstico, como resultado aporta dos tipos de informaciones. En primer lugar le acercará a la profesión o profesiones (universitarias o no universitarias) que considera más adecuada en función de sus características personales. En segundo lugar le aporta conocimientos en relación a la profesión elegida (duración de los estudios, lugares donde puede estudiarse, salidas laborales, etc.). Es un SE basado en reglas desarrollado en el lenguaje VP-Expert.

Sistema de RBC para la identificación de riesgos de software (4)

Los autores de esta investigación, (Cordero, y otros, 2013), proponen una herramienta que utiliza el RBC para facilitar la identificación de riesgos en los proyectos de desarrollo de software. Asimismo, el sistema

brinda la posibilidad de que a partir de un conjunto de características relevantes del proyecto que le son introducidas, se logre obtener de forma automática los riesgos potenciales a incidir a lo largo del ciclo de desarrollo, así como una posible mitigación o aprovechamiento de los mismos. Esto permite tener una visión adelantada, agilizando el proceso de toma de decisiones. La solución desarrollada combina principalmente un sistema de gestión de información con las técnicas de RBC. Constituye una aplicación web, basada en el modelo cliente - servidor y desarrollada sobre la plataforma Java.

Sistema Inteligente de Soporte a la Toma de Decisiones (5)

En esta investigación, (Rodríguez, 2010), propone un producto de software desarrollado completamente en software libre, que constituye una solución genérica, reutilizable y multiplataforma. Emplea técnicas de inteligencia artificial, específicamente el razonamiento basado en casos, permite a los dirigentes de cualquier institución u organismo tomar una decisión correcta en un tiempo mínimo, ante una situación dada. Con una arquitectura cliente servidor, este sistema, se consume desde la web como un servicio y puede ser perfectamente integrable tanto con un sistema de gestión como con un sistema de información geográfica que apoye también el proceso a informatizar.

El RBC en el ámbito de la Enseñanza/Aprendizaje (6)

En este artículo, según (Martínez, y otros, 2010), se describe un modelo para la enseñanza / aprendizaje, sustentado en la aplicabilidad del RBC para obtener una adecuada representación de la Modelación del Estudiante e inferir las estrategias de aprendizaje. Desarrollado en la Universidad Central de Las Villas, mediante el uso de la herramienta computacional HESEI, la cual ha sido aplicada exitosamente en la toma de decisiones en tareas de enseñanza/ aprendizaje. Se utiliza como estructura de datos una BC donde los rasgos predictivos de los casos se corresponden con el módulo Conocimiento del Alumno, y el rasgo objetivo con el módulo Conocimiento de los Contenidos, con el objetivo de identificar los propósitos que debe alcanzar un alumno al finalizar una unidad básica de aprendizaje.

El estudio comparativo de los sistemas descritos teniendo en cuenta los parámetros (1) multiplataforma, (2) interoperabilidad (3) tipo de licencia, (4) tipo de razonamiento, (5) dominio de aplicación y (6) configurable, arrojó los resultados que se muestran en la tabla 5.

El estudio de los SE para el apoyo a la toma de decisiones permitió identificar un conjunto de funcionalidades básicas, así como componentes esenciales que debe poseer el sistema a desarrollar. Luego del análisis realizado se concluye que estos SE constituyen una base para el diseño e implementación de la propuesta, no para su utilización, ya que son aplicaciones específicas para una

universidad o empresa determinada, que presentan características particulares en el dominio de aplicación. Incluso cuando (Rodríguez, 2010) propone un sistema con elementos significativos, no posee la opción de ser configurable, por tanto resulta limitado para la solución del problema de la investigación. Por otra parte, aunque algunos resuelven problemáticas en el campo de la educación, pero no lo realizan directamente en el área de la superación pedagógica.

Tabla 5: Comparación entre los SE para la toma de decisiones.

SE	Multiplata forma	Interoperabilidad	Tipo de licencia	Configurable	Tipo de razonamiento	Dominio de aplicación
1	Si	No	Pagada	No	RBC	Educación
2	No	No	Pagada	No	RBR	Educación
3	No	No	Pagada	No	RBR	Educación
4	Si	Si	Pagada	No	RBC	Empresarial
5	Si	Si	Libre costo	No	RBC	Militar
6	Si	No	Libre costo	No	RBC	Educación

1.5. Metodologías de desarrollo de software

Según (Avison y Fitzgerald, 1995), «una metodología de desarrollo de software es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información». A grandes rasgos, considerando su filosofía de desarrollo se pueden agrupar en: metodologías ágiles o ligeras y metodologías pesadas.

Para la selección de uno de estos enfoques se identificaron los siguientes indicadores:

- Capacidades y habilidades de desarrollo del equipo de trabajo
- Comunicación entre los miembros del equipo
- Trabajo en equipo
- Tiempo de desarrollo
- Tamaño del equipo
- Recursos materiales
- Compromiso del cliente
- Disposición del equipo ante los cambios durante el proyecto

Tabla 6: Comparación entre las metodologías ágil y pesada. [Tomado de: (Letelier, y otros, 2006)]

Indicadores	Metodología Ágil	Metodología Pesada
Capacidades y habilidades de desarrollo	Alta	Baja
Comunicación entre los miembros del equipo	Alta	Baja
Trabajo en equipo	Si	No
Tiempo de desarrollo	Limitado	Amplio
Tamaño del equipo	Pequeño	Grande
Recursos materiales	Pocos	Muchos
Compromiso del cliente	Alto	Bajo
Disposición ante los cambios	Altamente preparados	Muestran cierta resistencia

La investigación responde a necesidades y condiciones reales del equipo de desarrollo, determinadas por sus amplias habilidades y capacidades en el desarrollo soluciones de software, dominio de un lenguaje común, considerable experiencia y medios de colaboración entre los miembros del equipo lo que contribuye a la confianza y al respeto mutuo, destreza para la toma de decisiones y organización propia, demuestran una adecuada dinámica de funcionamiento. Además, el cliente está altamente comprometido y motivado con el desarrollo del SE y el equipo de trabajo está caracterizado por una elevada disposición y capacidad ante los nuevos cambios surgidos durante la investigación. Asimismo, el equipo de trabajo es pequeño y dispone de tiempo limitado para su desarrollo.

Las características personalógicas que presenta el equipo de trabajo, determinan en gran medida la utilización de una metodología ágil, pues según (Pressman, 2005), los defensores del desarrollo ágil del software resaltan la importancia de los factores de las personas en un desarrollo ágil exitoso. Como establecen (Cockburn y Highsmith, 2001); el desarrollo ágil se centra en los talentos y las habilidades de los individuos, puesto que el proceso se ajusta a personas y equipos específicos.

A partir de la selección del tipo de metodología ágil, se analizan varios exponentes de la misma, XP (eXtreme Programming), Scrum y OpenUp, por su gran aceptación en el desarrollo de proyecto de software. Para ello se tomaron en cuenta un conjunto de indicadores:

- Participación del cliente
- Realización de Pruebas
- Nivel de documentación
- Nivel de flexibilidad para el equipo de trabajo

Tabla 7: Comparación entre las principales metodologías ágiles. [Tomado de: (Letelier, y otros, 2006)]

Indicadores	XP	Scrum	OpenUp
Participación del cliente	Activa, el cliente tiene que estar presente y disponible todo el tiempo para el equipo	En cada iteración	Pertenece al equipo de trabajo
Pruebas	Ejecutadas constantemente ante cada modificación del sistema	En cada iteración	Muchas veces en cada iteración
Documentación	Se sustituye la documentación por la comunicación	Poca	Suficiente y necesaria
Centra la atención	Programadores	Gestión del proyecto	Ciclo completo
Flexible para el equipo	Poco	Poco	Mucho

El equipo de desarrollo seleccionó la metodología OpenUp por las ventajas que ofrece con respecto a las demás, evidenciado en la documentación que esta genera, la esencial para guiar el desarrollo. Además, esta metodología brinda la posibilidad de organizar el trabajo y determinar la mejor forma de alcanzar los objetivos, motivando a los miembros del equipo de desarrollo. También permite realizar varias pruebas en cada iteración, lo cual garantiza la elaboración de una solución estable y disponible durante su desarrollo. Por otra parte el cliente, a diferencia de las restantes metodologías analizadas, puede ser consultado tantas veces como sea necesario aunque no esté disponible a tiempo completo para el equipo de desarrollo.

1.6. Naturaleza del software

Según (Pressman, 2007) existen siete grandes categorías de software. El sistema propuesto es un híbrido entre las categorías de “Software de Inteligencia Artificial” y “Aplicaciones basadas en Web”. Para la selección de que fuera una aplicación web y no una aplicación de escritorio se tomaron los indicadores utilizado por (OSD, 2005), (1) personalización, actualización y soporte, (2) accesibilidad y cobertura, (3) capacidad de usuarios concurrentes, (4) portabilidad e (5) infraestructura y movilidad. Los resultados arrojados en la tabla comparativa realizada por el mismo autor favorecen el uso de una aplicación web. Todas estas características que dan respuesta a los indicadores que defienden el uso de una aplicación web se adapta satisfactoriamente en la UCI.

Tabla 8: Comparación entre software WEB vs. Software de Escritorio. [Tomado de (OSD, 2005)]

Características	Aplicación web (Navegador)	Escritorio
Personalización, actualización y soporte	Es suficiente con realizar los cambios en el servidor web.	Hay que realizarlos en cada estación de trabajo (PC) donde se tenga la aplicación.
Accesibilidad y cobertura	Cualquier lugar con acceso a la red.	Solo en el computador donde se haya instalado previamente el software.
Capacidad de usuarios concurrentes	Alta debido a la arquitectura de clientes livianos que la pueden usar.	Baja ya que la forma de diseño es centrada en un único usuario local.
Portabilidad	El sistema puede ser usado con cualquier navegador de Internet.	Solo funciona en el sistema operativo para el cual fue creado.
Infraestructura y movilidad	Solo se tiene que conectar a la red.	Está restringido a la ubicación del computador local.

1.7. Tecnologías y Herramientas de desarrollo

Para el desarrollo de un proyecto es necesario el estudio de las tecnologías y herramientas a utilizar, así como definir lenguaje de modelado y programación, herramientas de modelado e implementación y técnicas para elevar la calidad del producto a desarrollar.

En el SE para la generación de estrategias individuales de superación pedagógica del claustro UCI se realizó un estudio de las herramientas con el objetivo de obtener las más factibles para el desarrollo de aplicaciones web y SE. Para estar en correspondencia con estándares de la organización se definieron tres indicadores generales para seleccionar las herramientas: (1) herramientas de software libre o que según su licenciamiento sean de libre de costo, (2) tecnologías que garanticen que el sistema sea configurable y multiplataforma. Además, la selección de las herramientas se centra en el (3) tiempo limitado que presenta el equipo de desarrollo, así como los conocimientos y habilidades del mismo.

Lenguaje de Modelado

Los lenguajes de modelado, según (Magaña, 2009), «*permiten representar un software y describir las actividades a realizar para desarrollarlo*». En la actualidad existen dos lenguajes que son los más populares y ampliamente utilizados, BPMN y UML. BPMN, de acuerdo a los criterios de (Figuerola, 2014), está dirigido a los analistas de negocio, arquitectos de sistemas e ingenieros de software y fue desarrollado para mejorar el ciclo de vida del desarrollo de procesos. Por otra parte, el lenguaje UML facilita a los desarrolladores la especificación, visualización y documentación de modelos de sistemas de

software. La propia (Magaña, 2009) plantea, que «UML está dirigido a los arquitectos e ingenieros de software, como un medio para mejorar el proceso de desarrollo de software».

BPMN está relacionado con UML por el hecho que ambos definen una notación gráfica para los procesos de negocio, sin embargo, usan enfoques diferentes para modelarlos. UML en general ofrece un enfoque orientado a objetos para modelar aplicaciones, mientras que BPMN toma un enfoque centrado en los procesos. En la investigación se utiliza UML, ya que se desea desarrollar un sistema en base a diferentes componentes. Además se tuvo en cuenta la experiencia del equipo de trabajo y las posibilidades de extensión del lenguaje.

Herramienta de modelado

La calidad de un proyecto está determinada en gran medida por una adecuada decisión a la hora de escoger una herramienta de modelado con UML. Es por ello que el equipo de desarrollo utilizó una serie de indicadores para la selección de la herramienta de modelado con UML. Los elementos a valorar fueron (1) el modelado de datos y manejo de diagramas, (2) el apoyo metodológico y soporte completo UML y (3) la tendencia de utilización en la comunidad de desarrolladores en la UCI. Como resultado se obtienen: Visual Paradigm, Rational Rose y Enterprise Architect.

La herramienta de modelado que responde con mayor potencial a los indicadores evaluados anteriormente, así como los indicadores generales, es **Visual Paradigm**. Esta herramienta de Ingeniería de Software Asistida por Computadora (CASE, por sus siglas en inglés), es eficiente, fácil de usar y rica en características. Se integra con varios entornos de desarrollo como Eclipse, NetBeans, VisualStudio, entre otros. Suministra un generador de mapeo de objetos relacionales para los lenguajes de programación Java, PHP y algunos lenguajes del ambiente .Net. (Visual Paradigm For UML, 2014)

Lenguaje de Programación

En los últimos años la programación orientada a objetos (POO) se ha convertido en una de las formas más populares para el desarrollo de proyectos de software, de acuerdo con los criterios dados por (Joyanes Aguilar, 1996). Este estilo de programación fundamenta su desarrollo en la creación de un sistema de clases y objetos a partir de ejemplos del mundo real, en el que los objetos realizan acciones o tienen mecanismos de interacción. En la actualidad existen diversos lenguajes que poseen este paradigma, en los que destacan PHP, Python y Java, según las consideraciones del propio autor.

A partir de los indicadores generales se seleccionó el lenguaje **Java**, ya que su diseño, su robustez, el respaldo de la industria y su fácil portabilidad, lo han convertido en uno de los lenguajes con un mayor

crecimiento y amplitud de uso en distintos ámbitos de la industria de la informática. Este lenguaje ofrece aplicaciones para todos los sistemas operativos y si no es la versión adecuada para dicho sistema, la misma aplicación java se encarga de descargas o actualizaciones (Tocolapa, 2013). Además, posee valor agregado, al constituir el lenguaje de programación de mayor conocimiento y experiencia del equipo de trabajo.

Entorno de desarrollo integrado

A partir de la selección del lenguaje de programación, es necesario definir un entorno de desarrollo integrado (IDE, por sus siglas en inglés) que permita desarrollar aplicaciones utilizando el lenguaje Java. La revisión de la literatura (Tocolapa, 2013) en correspondencia con los indicadores generales, determinó que dentro de los principales entornos de desarrollo para este lenguaje, se encuentran NetBeans y Eclipse. Los autores basándose en estos propios indicadores, principalmente en el tiempo limitado que presenta el equipo de desarrollo, así como los conocimientos y habilidades del mismo, seleccionaron **Eclipse** como IDE.

Eclipse es un IDE multiplataforma, de código abierto, extensible. Está construido sobre un mecanismo para descubrir, integrar y ejecutar módulos (arquitectura basada en *plug-ins*), que facilita el desarrollo de aplicaciones con RichFaces, Seam, Hibernate y el JBoss Application Server. Permite integrar diversos lenguajes de programación sobre un mismo IDE e introducir otras aplicaciones accesorias (Tocolapa, 2013)

Framework de desarrollo

Un framework (marco de trabajo) según (Riehle, 2000) promete una mayor productividad y un menor tiempo en el desarrollo de aplicaciones a través del diseño y la reutilización de código. Aunque lo antes mencionado, solamente no lo garantiza la utilización de un framework, para el desarrollo del sistema si se utilizaron. A partir de indicadores como (1) integridad, (2) persistencia de los datos, (3) desarrollo de interfaces se seleccionaron los siguientes framework:

Según (ORACLE, 2010) **Java Server Faces (JSF)** *«fue desarrollado para aplicaciones Java basadas en web, permite la simplificación y agiliza el desarrollo de interfaces de usuario en aplicaciones Java Enterprise Edition. JSF emplea Facelets como la tecnología para hacer el despliegue de las páginas, aunque se puede adaptar a otras tecnologías como XUL (XML para la interfaz de usuario). El uso de JSF permite el manejo de estados y eventos, la asociación entre los datos de la interfaz y los datos de la aplicación web, la especificación de la navegación del usuario, entre otras funcionalidades».*

Teniendo en cuenta los elementos definidos por (Yuan, y otros, 2009), **Enterprise JavaBeans (EJB)** «proporciona un modelo de componentes informáticos distribuidos del lado del servidor. Encapsula la lógica del negocio de una aplicación y permite construir aplicaciones de negocio portables, reutilizables y escalables usando el lenguaje de programación Java. Permite abstraerse de los problemas generales de una aplicación (conurrencia, transacciones, persistencia, seguridad, etc.), para centrarse en el desarrollo de la lógica de negocio en sí». La ventaja de estar basado en componentes informáticos permite que éstos sean flexibles y sobre todo reutilizables. EJB a partir de la versión EJB 3.0 convierte los Plain Old Java Objects (POJOs) en objetos de persistencia de base datos.

Según el propio autor, **EJB3** y **JSF** son tecnologías que se complementan entre sí, aunque están diseñados como framework separados, cada uno con su propia filosofía. Para complementar el trabajo de estos framework y ajustarse a los indicadores, se decidió por el equipo de desarrollo utilizar otra tecnología que facilitara la integración de JSF y EJB3. A partir de lo antes mencionado se utiliza para la integración el framework Seam.

Seam colapsa la capa artificial entre EJB3 y JSF. Proporciona un enfoque coherente, basado en anotaciones para integrar EJB3 y JSF. Con unas pocas simples anotaciones, los componentes de negocio EJB3 en Seam se pueden utilizar directamente para respaldar los formularios web JSF o controlar los eventos web de interfaz de usuario. Seam permite a los desarrolladores utilizar POJOs anotados para todos los componentes de la aplicación. En comparación con las aplicaciones desarrolladas en otros framework web, las aplicaciones Seam son conceptualmente simples y requieren mucho menos código (en Java y XML) para la misma funcionalidad. Además, el propio autor plantea que, Seam provee una alta granularidad de contextos de estado. Es el contexto conversacional así como el asociado a procesos del negocio.

Para el correcto funcionamiento de todas estas herramientas y tecnologías es necesario la utilización de otras tecnologías que complementan de una forma u otra el trabajo de las anteriores.

El consorcio internacional que produce recomendaciones para la World Wide Web (W3C), en el año 2010, expone que el **Lenguaje de Mercado Extensible (XML)** «es un sencillo formato de texto para representar información estructurada: los documentos, los datos, la configuración, los libros, las transacciones, facturas y mucho más». Fue derivado de un formato estándar antiguo llamado SGML (ISO 8879), con el fin de ser más adecuado para uso de la Web. XML es uno de los formatos más utilizados para el

intercambio de información estructurada hoy: entre los programas, entre las personas, entre los equipos y las personas, tanto a nivel local como a través de redes.

La propia comunidad internacional expone el **Lenguaje de Marcado de Hipertexto Extensible** (XHTML, por sus siglas en inglés), como una versión más estricta y limpia del Lenguaje de Marcado de Hipertexto (HTML, por sus siglas en inglés), que nace ciertamente con el objetivo de remplazar a HTML ante su limitación de uso con las herramientas basadas en XML. XHTML es compatible con los principales navegadores.

La compañía Red Hat defiende **RichFaces** como una librería de componentes enriquecidos para JSF. Tiene un framework avanzado para la integración sencilla de las funcionalidades Ajax dentro del desarrollo de las aplicaciones del negocio. **RichFaces** contiene ciclo de vida, validaciones, conversiones y la gestión de recursos estáticos y dinámicos. Los componentes de RichFaces están construidos con soporte Ajax y puede ser fácilmente incorporado dentro de las aplicaciones JSF.

Según (Yuan, y otros, 2009), **Ajax4jsf** es una librería de código abierto, que se integra totalmente en la implementación de JSF. Extiende la funcionalidad de sus etiquetas, dotándolas con tecnología Ajax sin código JavaScript. Con este framework se puede variar el ciclo de vida de una petición JSF, recargar determinados componente informáticos de la página sin necesidad de refrescarla en su totalidad, realizar peticiones al servidor de forma automática, controlar cualquier evento de usuario.

El propio autor manifiesta que **Facelets** es un framework para plantillas, de código abierto bajo la licencia Apache y tecnología alternativa de controlador de vista de JSF. Facelets soporta los componentes informáticos de interfaz de usuario JSF y construye su propio árbol de componente informáticos, lo que refleja el punto de vista de una aplicación de este tipo. Permite que Java Server Pages (JSP) y JSF puedan trabajar en conjunto en una misma aplicación web. Incluye funciones para expresiones, así como también permite la creación de librerías de componente informáticos. Facelets permite en general el desarrollo amigable para el diseñador gráfico.

Java Persistence API (JPA), según (ORACLE, 2010), es la interfaz de programación de aplicaciones de persistencia desarrollada para la plataforma Java EE y está incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que provee un mapeo objeto-relacional. El objetivo que persigue su diseño es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, como sucedía con EJB2, y permitir el uso de POJOs.

(Bauer, y otros, 2007) manifiestan que **Hibernate** es una herramienta de mapeo objeto-relacional para la plataforma java, que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación mediante archivos declarativos xml. Cuenta con el lenguaje de consulta hibernate (hql, por sus siglas en inglés), que es un lenguaje de consulta similar al sql. Además brinda, de una manera muy rápida y mejorada, la posibilidad de generar bases de datos en cualquiera de los entornos soportados: postgresql, oracle, db2, mysql, entre otros, también es una herramienta de código abierto.

Tecnologías Horizontales

La **Máquina Virtual Java**, según (García, y otros, 1999), es el núcleo del lenguaje de programación Java. De hecho, es imposible ejecutar un programa Java sin ejecutar alguna implantación de la misma. Es la clave de muchas de las características principales de Java, como la portabilidad, la eficiencia y la seguridad. Se encarga de interpretar todo el código java y convertirlo al lenguaje nativo del sistema operativo en uso.

Autores como (Yuan y otros, 2009) definen **Java Enterprise Edition 5 (Java EE 5)**, como una tecnología que permite al desarrollador crear una aplicación de empresa portable entre plataformas y escalable, a la vez que se integra con tecnologías anteriores. Otros beneficios añadidos son, por ejemplo, que el servidor de aplicaciones puede manejar transacciones, la seguridad, escalabilidad, concurrencia y gestión de los componente informáticos desplegados, significando que los desarrolladores pueden concentrarse más en la lógica de negocio de los componente informáticos en lugar de en tareas de mantenimiento de bajo nivel.

JBoss Application Server (JBoss AS) es uno de los servidores de aplicaciones de código abierto más usado en el mercado. Se encuentra bajo la licencia GNU LGPL, por lo que puede libremente usarse sin costo alguno en cualquier aplicación comercial o ser redistribuido. Por ser una plataforma certificada JEE 5, soporta todas las especificaciones correspondientes, incluyendo servicios adicionales como clustering, caching y persistencia. JBoss AS uno de los servidores de aplicaciones recomendado para aplicaciones Java, pues está basado en este lenguaje de programación, y también es propicio para aplicaciones basadas en la web. Según (Yuan, y otros, 2007) JBoss AS es sin dudas el mejor servidor para aplicaciones basadas en el framework Seam.

Sistema gestor de base de datos (SGBD)

Según (Cobo, 2007) «*un SGBD permite el almacenamiento, manipulación y consulta de datos pertenecientes a una base de datos organizada en uno o varios ficheros. En el modelo más extendido*

[base de datos relacional] *la base de datos consiste, de cara al usuario, en un conjunto de tablas entre las que se establecen relaciones*». Existen SGBD como son PostgreSQL y MySql que son los más utilizados en la universidad de acuerdo a los estándares de organización. La selección del SGBD se centró en (1) la escalabilidad y (2) identidad referencial, (3) transacciones que estos poseen. El resultado estos indicadores, así como los indicadores generales, arrojaron la necesidad de utilizar PostgreSQL.

PostgreSQL, según (Pecos, 2002), *«implementa el uso de rollback's, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz. También tiene la capacidad de comprobar la integridad referencial, posee una gran escalabilidad, utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema*». Ofrece una potencia adicional al incorporar los conceptos de clases, herencia, tipos y funciones; de modo que los usuarios puedan extender fácilmente el sistema. Además de todas estas características técnicas, el equipo de desarrollo se destaca en el conocimiento y la experiencia en esta herramienta.

PgAdmin III es la herramienta utilizada como cliente para la gestión de la base de datos. *«Es una herramienta de código abierto que usa la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda usar en Linux, FreeBSD, Solaris, Mac OS X y Windows*». Incluye un editor SQL con resaltado de sintaxis, un editor de código en el servidor, un agente para lanzar script programados y mucho más. La conexión al servidor puede hacerse mediante conexión TCP/IP o Unix Domain Sockets, y puede cifrarse mediante Capa de Conexión Segura (SSL) para mayor seguridad.

Conclusiones parciales

- Los SE están compuestos principalmente por la base de conocimientos, la máquina de inferencia, la base de datos y la interfaz de usuario.
- Existen diferentes formas de representación del conocimiento para el diseño de un SE, siendo el RBC la que mejor se adapta a las particularidades de la investigación.
- Los SE analizados para la toma de decisiones, poseen limitaciones para darle solución al problema de la presente investigación, al ser aplicaciones específicas, carecer de opciones de configuración y emplear un tipo de razonamiento no idóneo.
- Se desarrolló la investigación con la metodología OpenUp porque brinda la posibilidad de organizar el trabajo y determinar la mejor forma de alcanzar los objetivos; además permite, realizar varias pruebas en cada iteración y contactar con el cliente tantas veces como sea necesario.

- Las características del SE a desarrollar, determinaron el uso de las tecnologías EJB, JSF, XHTML y el framework Seam; los lenguajes Java y UML y las herramientas PostgreSQL, PgAdmin y Visual Paradigm.

CAPÍTULO 2. ANÁLISIS Y DISEÑO DE LA SOLUCIÓN PROPUESTA

2.1. Definición conceptual y operacional de las variables

Según lo que establecen (Nocedo, 1984), (Blanco, 2003), (Campistrous, 2003) y (Hernández, y otros, 2006), en el proceso investigativo se tuvo en consideración la definición conceptual y operacional de las variables. La variable independiente la constituye los sistemas expertos, cuya definición fue ofrecida en el epígrafe dos del capítulo uno. Por otra parte la variable dependiente se identifica como la fiabilidad en la generación de estrategias individuales de superación pedagógica del claustro de la UCI.

Tomando como fundamento los criterios planteados, en esta investigación se entiende por fiabilidad «la capacidad de obtener resultados consistentes en mediciones sucesivas del mismo fenómeno» (Jacob, 1994). De igual forma (Lamy, y otros, 2010) definen fiabilidad como «la capacidad de dar respuestas similares a las de un experto o grupo de expertos». La escala de valoración propuesta para cada indicador, fue producto de estudios donde la variable fiabilidad ha sido un éxito, entre los que destacan (Martí, 2012), (Vallejo, 2007) y (Monroy, 2003).

Tabla 9: Operacionalización de la variable dependiente.

Variable	Dimensiones	Indicadores	Escala de valoración
Fiabilidad en la generación de estrategias individuales de superación pedagógica del claustro de la UCI.	Precisión	Tasa de aciertos (TA)	Alta: Por encima del 84% de los casos analizados. Cuestionable: Entre el 60% y el 84% de los casos analizados. Baja: Por debajo del 60% de los casos analizados.
		Tasa de errores (TE)	Baja: Por debajo del 15% de los casos analizados. Cuestionable: Entre el 15% y el 39% de los casos analizados. Alta: Por encima del 39% de los casos analizados.
	Estabilidad	Tasa de dispersión (TD)	Alta: Por debajo del 15% de diferencia en cada medición de precisión con respecto a la precisión final. Baja: Por encima del 14% de diferencia en cada medición de precisión con respecto a la precisión final.

2.2. Caracterización del objeto de estudio

Desde su creación, en la UCI se ha prestado atención prioritaria a la formación de los profesores noveles que integran mayoritariamente su claustro, lo que se recoge en el Sistema de Superación de Profesores establecido por el MES en el año 2008. Dentro de este proceso, resulta un factor determinante la creación de estrategias individuales de superación pedagógica para los profesores, que tiene como máximos responsables a los J.Dpto. El resultado de las entrevistas, a los J.Dpto que conformó la muestra, constató lo siguiente:

- El J.Dpto-DF genera la estrategia individual de superación pedagógica de cada profesor de su claustro.
- Los J.Dpto-DC y el J.Dpto-CP no realizan este proceso; los profesores, con vínculo docente, asociados a cada uno de estos departamentos también pertenecen a una facultad, y por consiguiente su estrategia de superación pedagógica es planificada por su J.Dpto-DF.

En lo adelante se le atribuye el término J.Dpto a los J.Dpto-DF, ya que el objetivo es describir el proceso donde intervienen los J.Dpto que generan la estrategia individual de superación pedagógica de los profesores. Para la realización de este proceso se tienen en cuenta diversas vías de obtención del conocimiento sobre un docente (Ciudad, y otros, 2013), entre las que destacan la autoevaluación del docente, la evaluación externa, la evaluación de los controles a clases y la evaluación profesoral, bien fundamentadas en el documento ESPC-UCI.

A partir de la información recopilada sobre el profesor, el J.Dpto posee los elementos necesarios para identificar las áreas en las que el profesor tiene una elevada preparación demostrada y aquellas en las que debe superarse pedagógicamente. Finalmente propone un conjunto de acciones de postgrado encaminadas a mitigar las deficiencias detectadas. Lo antes descrito resulta un proceso complejo, depende en gran medida de las habilidades y experiencias adquiridas por los J.Dpto en la elaboración de la estrategia individual de superación pedagógica de cada profesor.

Con el objetivo de evaluar el desarrollo de este proceso se realizaron entrevistas a los J.Dpto. El análisis evidenció que más del 75% de los encuestados, solo tienen en cuenta la evaluación de los controles a clases y la autoevaluación del profesor. Asimismo, menos del 25% incorporan la evaluación profesoral como vía de obtención del conocimiento. Por otra parte, según datos registrados en el documento CD-UCI en el año 2013, el 73% de los J.Dpto posee la categoría docente de Instructor o Asistente y solo el 27% es profesor Auxiliar o Titular. De igual forma, la propia fuente muestra que el 72% es menor de 35 años.

Luego de realizar una valoración del proceso se concluye que, la mayoría de los J.Dpto emplean insuficientes vías para la obtención del conocimiento sobre un profesor, poseen índices bajos en las categorías docentes Auxiliar o Titular, así como poca experiencia en la generación de la estrategia de superación pedagógica; como consecuencia se obtienen diferencias en la superación pedagógica de cada departamento del claustro de la universidad, lo que evidencia baja fiabilidad en la generación de las estrategias.

2.3. Modelo de dominio

Comprender y tener un firme conocimiento del contexto en el que se trabaja, representa uno de los pilares fundamentales para garantizar el éxito del desarrollo de un sistema. El modelo de dominio o modelo conceptual, de acuerdo con (Larman, 1999), constituye una de las formas de representar los conceptos fundamentales del dominio de la investigación y la relación que existe entre sí, de forma tal, que facilite el entendimiento y contribuya a la solución del problema. En la presente investigación lo constituye la generación de estrategias individuales de superación pedagógica del claustro de la UCI, el cual fue caracterizado en el epígrafe anterior. El diagrama que se presenta a continuación describe su funcionamiento.

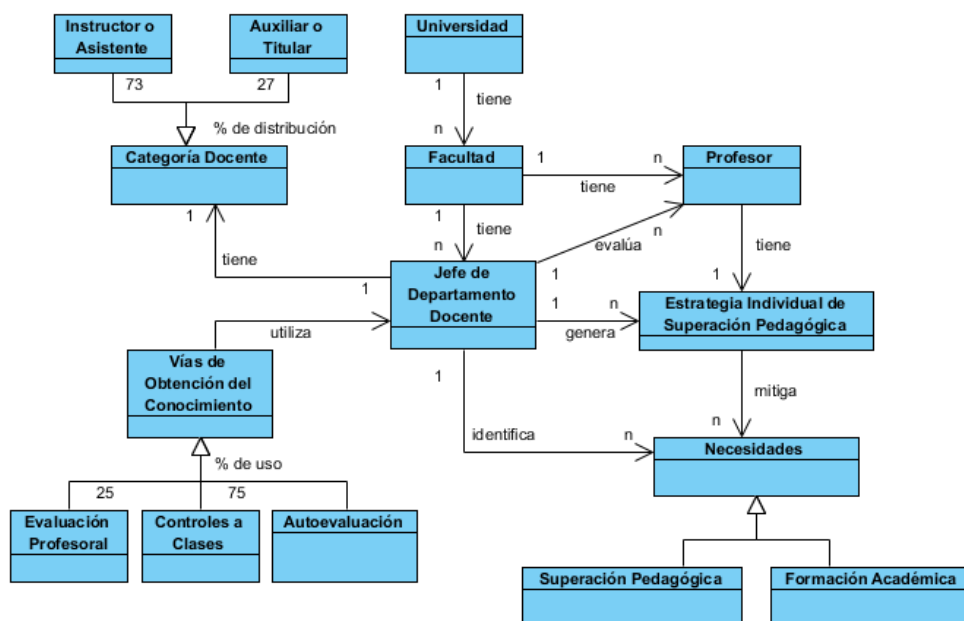


Figura 5: Diagrama de clases del dominio.

2.4. Requisitos funcionales y no funcionales de software

Según (Martínez, 2004) la ingeniería de requisitos es la etapa más crucial del desarrollo de un proyecto de software. Los requisitos son el conjunto de técnicas y procedimientos que permiten conocer los elementos necesarios para definir un proyecto de software, es la ingeniería de requisitos, donde se lleva a cabo el proceso de descubrir, analizar, escribir y verificar los servicios y restricciones, del sistema que se desea producir; este proceso se realiza mediante la obtención, el análisis, la especificación, la validación y la administración de los requisitos del software. Los requisitos de software son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema y las restricciones en las que debe operar.

Requisitos funcionales

Según (Pressman, 2005) los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir, incluyendo las acciones. Para la construcción del SE se definieron los requisitos funcionales siguientes:

RF 1: Adicionar caso: se adicionan los casos que conforman la base de conocimiento.

RF 2: Modificar caso: se modifican los casos que conforman la base de conocimiento.

RF 3: Eliminar caso: se eliminan los casos que conforman la base de conocimiento.

RF 4: Visualizar casos: se visualizan los casos que conforman la base de conocimiento.

RF 5: Ver detalles del caso: se visualizan los elementos que componen el caso seleccionado.

RF 6: Visualizar peticiones: se visualizan los casos creados por los J.Dpto a espera de aprobación por los expertos.

RF 7: Atender petición: se visualiza la propuesta del J.Dpto y el experto la aprueba o no como nuevo conocimiento.

RF 8: Diagnosticar profesor: se valoran los resultados por las distintas vías de obtención del conocimiento y se emite un diagnóstico, que contendrá tanto las áreas en las que el docente tiene una elevada preparación demostrada y aquellas en las que debe superarse pedagógicamente.

RF 9: Clasificar profesor: se clasifica un profesor de acuerdo a los programas que existen en el sistema.

RF 10: Sugerir acciones de postgrado: se sugiere un conjunto de acciones que deben ser culminadas para mitigar las necesidades de superación pedagógica.

RF 11: Justificar propuesta: se exponen al usuario las causas que dieron lugar a los requisitos funcionales 8,9 y 10, de acuerdo a la opinión del experto.

RF 12: Crear nueva propuesta: el usuario puede crear una nueva propuesta, si no coincide con la sugerencia dada por el sistema.

RF 13: Calcular dispersión de la nueva propuesta: se calcula el grado de dispersión que existe entre la nueva propuesta, creada por el usuario, y la propuesta del SE.

Requisitos no Funcionales

Según (Somerville, 2005) los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Son las características que hacen al producto atractivo, usable, rápido y confiable. Siguiendo la clasificación del propio autor, se definieron los siguientes requisitos no funcionales:

Requisitos del producto.

RNF 1: El sistema debe permitir la fácil interacción y llegar de manera rápida y efectiva a la opción deseada. Debe poseer además una interfaz de manejo cómodo que posibilite a los usuarios sin experiencia una rápida adaptación.

RNF 2: El sistema debe estar disponible las 24 horas del día.

RNF 3: El sistema adoptará buenas prácticas de programación para incrementar el rendimiento en operaciones costosas para la máquina virtual como la creación de objetos entre otras.

Requisitos organizacionales

RNF 4: En la base de datos es necesario que esté instalada una versión de PostgreSQL 9.1 o superior. Tecnologías necesarias: Máquina Virtual de Java (JVM, por sus siglas en inglés), Java Runtime Environment (JRE, por sus siglas en inglés), el Servidor de aplicaciones JBoos 4.2.2 y deberá disponer de navegadores web como Internet Explorer 7, Opera 9, Google Chrome 1 y Mozilla Firefox 2 o versiones superiores.

RNF 5: El sistema debe seguir estándares para mantener una coherencia y estilo común entre todas las páginas. Interfaz sencilla con colores suaves a la vista y sin cúmulo de imágenes u objetos que distraigan al cliente del objetivo.

RNF 6: El sistema debe implementarse siguiendo los estándares de codificación y diseño definidos por el equipo de desarrollo.

RNF 7: El sistema como mínimos recomienda, tanto para servidores como para estaciones de trabajo:

Equipo Servidor

- Procesador Core™2 Duo a 3 GHz

- Memoria RAM 2 GB. La cantidad de memoria RAM varía según la cantidad de usuarios Web, así como de la cantidad de tareas extras que ejecute el servidor.
- Disco Duro con 80 Gb libres para datos.

Equipo Cliente

- Procesador Core™2 Duo 1.5GHz.
- Memoria RAM 512 MB.

RNF 8: El sistema debe correr sobre las plataformas Linux y Windows.

Requisitos externos

RNF 9: La tabla *tn_curso* debe ser gestionada por otro sistema y sincronizada con la base de datos del sistema propuesto.

RNF 10: El sistema debe contar con un mantenimiento periódico a su servidor, garantizando la integridad de los datos almacenados en el mismo.

2.5. Modelo de casos de uso del sistema

Según (Jacobson, y otros, 2000), «un caso de uso especifica una secuencia de acciones, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto. Los diagramas de casos de uso especifican las funcionalidades y el comportamiento de un sistema mediante la interacción con los usuarios y/u otros sistemas, muestra la relación entre los actores y los casos de uso». De acuerdo con lo antes descrito, se realiza a continuación una descripción de los actores que intervienen en el sistema y se conforma el modelo de casos de uso.

Tabla 10: Descripción de los actores del sistema.

Actor	Descripción
Experto	Persona experta en el área de superación pedagógica, que posee permisos en el sistema para introducir los datos en la base de conocimiento.
Directivo	Persona que interactúa con el sistema con el objetivo de obtener estrategias individuales de superación pedagógica para los profesores. Entre este grupo de persona destacan los J.Dpto.

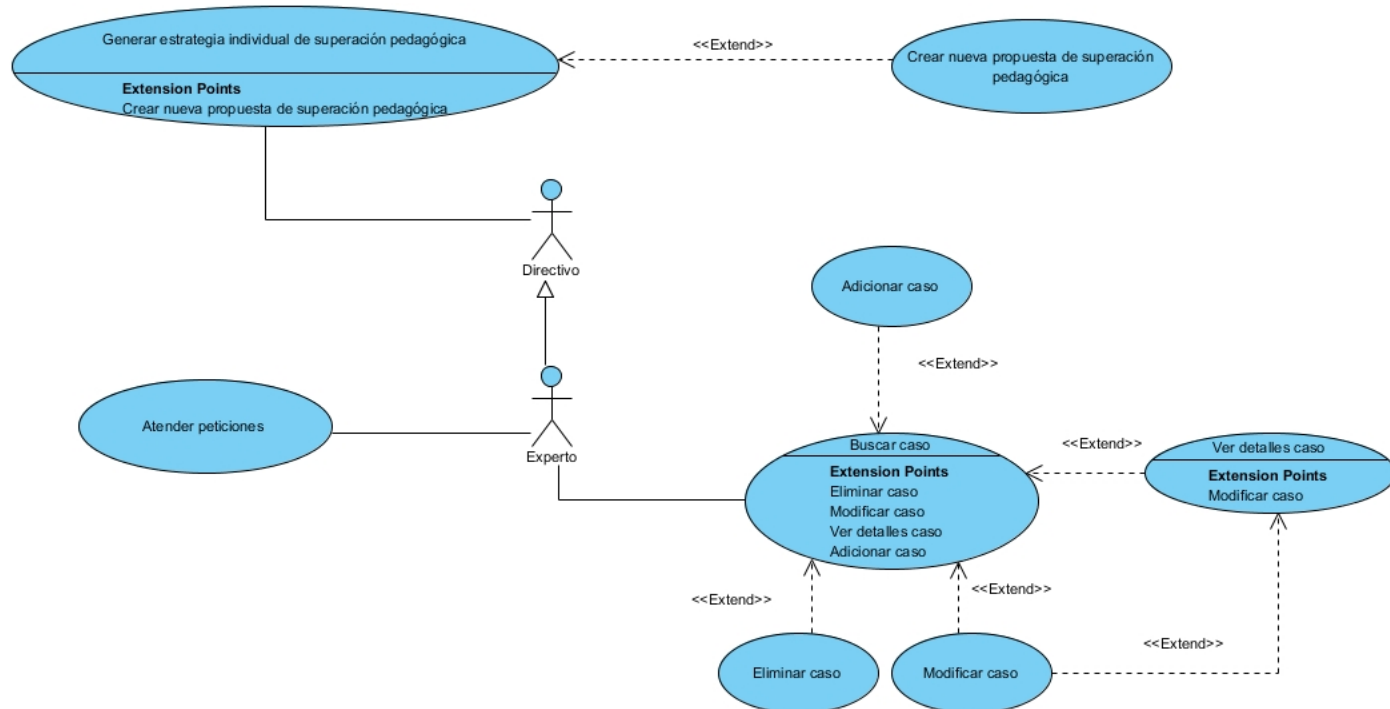


Figura 6: Diagrama de casos de uso del sistema.

A continuación se muestra un resumen de los casos de uso del sistema. En el **anexo 2** se exponen las descripciones textuales ampliadas de los casos de uso arquitectónicamente significativos: Generar estrategia individual de superación pedagógica, Buscar caso y Atender peticiones.

Tabla 11: Descripción de casos de uso del sistema.

Caso de Uso (CU)	Actor	Descripción	Referencia
Generar estrategia individual de superación pedagógica	Directivo Experto	El caso de uso inicia cuando el actor selecciona un profesor del listado de profesores, el sistema muestra los datos generales y las evaluaciones del profesor. Si selecciona la opción Aceptar, el sistema asigna la propuesta de superación pedagógica al profesor seleccionado. Si no coincide con la propuesta determinada por el sistema, selecciona la opción Crear nueva propuesta de superación pedagógica (Ver CU: Crear nueva Propuesta de Superación Pedagógica). Si accede a la opción Cancelar, el sistema regresa a la vista anterior. En cualquiera de los flujos el caso de uso termina.	RF 8 RF 9 RF 10 RF 11

Caso de Uso (CU)	Actor	Descripción	Referencia
Crear nueva propuesta de superación pedagógica	Directivo Experto	El caso de uso inicia cuando el actor selecciona la opción Nueva sugerencia, el sistema muestra la dispersión entre la propuesta de superación pedagógica emitida y la nueva propuesta de superación pedagógica, actualizándose continuamente, además brinda la posibilidad de introducir los datos de la nueva propuesta de superación pedagógica y de seleccionar las opciones de Aceptar o Cancelar, el caso de uso termina.	RF 12 RF 13
Buscar caso	Experto	El caso de uso inicia cuando el actor accede a la opción Buscar caso, el sistema brinda la posibilidad de introducir criterios de búsqueda para localizar el Caso deseado, el actor introduce los datos que considera como criterios para realizar una búsqueda, el sistema busca y muestra los Casos que cumplen con los criterios de búsqueda y permite añadir Caso (Ver CU: Adicionar caso), modificar Caso (Ver CU: Modificar caso), ver datos del Caso (Ver CU: Ver detalles del caso), eliminar Caso (Ver CU: Eliminar caso) y Cancelar operación. En cualquiera de los flujos el caso de uso termina.	RF 4
Adicionar caso	Experto	El caso de uso inicia cuando el actor accede a la opción Añadir, el sistema brinda la posibilidad de introducir los datos para crear el Caso, el actor introduce los datos del Caso, el sistema crea el Caso, el caso de uso termina.	RF 1
Modificar caso	Experto	El caso de uso inicia cuando el actor selecciona un Caso y accede a la opción Modificar caso o Ver detalles del caso (Ver Cu: Ver detalles del caso), el sistema muestra los datos del Caso y brinda la posibilidad de cambiar sus valores ya sea introduciendo nuevos o seleccionando diferentes, el actor modifica los datos que necesita, el sistema actualiza los datos del Caso, el caso de uso termina.	RF 2
Ver detalles del caso	Experto	El caso de uso inicia cuando el actor accede a la opción Ver detalles del caso, el sistema muestra los datos del Caso y permite modificar caso (Ver CU: Modificar caso), el caso de uso termina.	RF 5
Eliminar caso	Experto	El caso de uso inicia cuando el actor selecciona un Caso y accede a la opción Eliminar caso, el sistema elimina el Caso, el caso de uso termina.	RF 3

Caso de Uso (CU)	Actor	Descripción	Referencia
Atender peticiones	Experto	El caso de uso inicia cuando el actor accede a la opción Atender peticiones, el sistema muestra un listado de Peticiones, el actor selecciona una petición, el sistema carga los datos de la petición seleccionada, permite aceptar o rechazar petición y posibilita Aceptar cambiar el estado de la petición o Cancelar operación, el caso de uso termina.	RF 6 RF 7

2.6. Modelo del diseño

Según (Pressman, 2005), luego de analizar y especificar los requisitos, el diseño de software es la última acción de la ingeniería dentro de la actividad de modelado, la cual establece una plataforma para la construcción del sistema (generación de código y pruebas).

Diseño arquitectónico

La estructura de los datos y los componentes del programa, según (Pressman, 2005), requeridos para construir un software se representa en el diseño arquitectónico. El mismo autor establece el estilo arquitectónico, la estructura y las propiedades de los componentes que comprende, y las interrelaciones que tienen lugar entre los componentes arquitectónicos del sistema.

El sistema propuesto, según su naturaleza de software (Pressman, 2005), forma parte de la categoría de las *Aplicaciones basadas en Web*. El mismo autor plantea que el estilo arquitectónico ideal utilizado para estas aplicaciones es el cliente/servidor. En el cliente se encuentra el navegador, que es el encargado de enviar una solicitud o hacer una petición al servidor; en este último se encuentra el sistema, el cual envía uno o varios mensajes con la respuesta.

El autor mencionado, sugiere para las aplicaciones web, el patrón de diseño arquitectónico Modelo-Vista-Controlador (MVC), encargado de desacoplar la interfaz de usuario de la funcionalidad y el contenido de la aplicación. El MVC divide una aplicación en tres componentes claramente identificables y con funcionalidad bien definida: Modelo, Vista y Controlador.

El **Modelo** contiene el contenido específico de la aplicación y la lógica de procesamiento, e incluye todos los objetos de contenido, el acceso a fuentes de datos/información externas y toda la funcionalidad de procesamiento, específicos de la aplicación. Se identifican como tecnologías utilizadas: OMR Hibernate, EJB encargada de encapsular la lógica de negocio y el API de persistencia de java JPA.

La **Vista** contiene las funciones específicas de la interfaz y habilita la presentación del contenido y la lógica de procesamiento, e incluye todos los objetos de contenido, acceso a fuente de datos/información externa y a toda funcionalidad de procesamiento requerida por el usuario final (maneja la presentación visual de los datos representados por el modelo). Se utilizan las páginas XHTML desarrolladas básicamente con JSF, utilizando librerías y componentes como AJAX y Richfaces, y Facelets como motor de plantillas.

El **Controlador** gestiona el acceso al **Modelo** y la **Vista**, y coordina el flujo de datos entre ellos. Interpreta las acciones del usuario sobre el sistema, informando al modelo y/o vista, para que cambien según resulte apropiado. Este patrón arquitectónico se ve claramente evidenciado en el framework JBoss Seam, el cual integra los framework JSF y EJB.

Patrones de diseño




Según (Jiménez, 2003), «*los patrones de diseño cambian la perspectiva acerca de las posibilidades de diseño de la solución de un problema, permiten que estos sean más flexibles, modulares, reutilizables y comprensibles*». Estos patrones ayudan al diseñador a lograr un buen diseño rápidamente. Para el desarrollo del sistema se utilizaron los patrones GRASP. Estos patrones describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Según (Larman, 1999), constituyen un apoyo para la enseñanza, ayuda a entender el diseño de objeto esencial y emplea el razonamiento para el diseño de una forma sistemática, racional y explicable.

En la presente investigación se utilizó el patrón **Experto**, como principio básico de asignación de responsabilidades, permitió mantener el encapsulamiento, y que los objetos utilicen su propia información para llevar a cabo sus tareas, además de distribuir el comportamiento entre las clases que contienen la información requerida. Asimismo se empleó el patrón **Creador** con el objetivo de ayudar a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases; una de las consecuencias de usar este patrón es la visibilidad entre la clase creada y la clase creador. Permitted el bajo acoplamiento, lo cual supone facilidad de mantenimiento y reutilización. Por último, se utilizaron los patrones **Alta cohesión** y **Bajo acoplamiento** con el objetivo de tener menor dependencia y especificar los propósitos de cada objeto en el sistema, igualmente lograr la coherencia en la información que almacena una clase y tener las clases lo menos relacionadas entre sí.

Diagrama de clases del diseño

Los diagramas de clases del diseño (DCD), de acuerdo con (Larman, 1999), describen gráficamente las clases del software y las interfaces de una aplicación. Además muestran el flujo de cómo una página servidora construye una página cliente, la cual contiene un formulario que puede actualizar directamente a las entidades o enviar las peticiones a la página servidora. Estas últimas invocan métodos y responsabilidades de las clases controladoras, las cuales pueden consultar o modificar las entidades. Los diagramas de clases del diseño utilizados para el desarrollo del sistema, se realizaron mediante estereotipos web que son una representación gráfica de los componentes a los cuales hacen referencia. Se definen 3 tipos fundamentales de estereotipos para clasificar las clases del diseño de la aplicación:

Tabla 12: Descripción de los estereotipos web de las clases del diseño.

 Páginas servidoras	Se ejecutan en el servidor web, procesan las peticiones de los usuarios, generando páginas clientes que pueden ser representadas por los navegadores web.
 Páginas clientes	Presenta al usuario la interfaz con la que puede realizar peticiones e interactuar con el sistema.
 Formularios	Están contenidos en las páginas clientes. Representan la entrada y salida de los datos en el sistema. Envían los datos a las páginas servidoras para poder procesar la solicitud realizada.

A continuación se muestran los diagramas de clases de diseño pertenecientes a los casos de uso, Generar estrategia individual de superación pedagógica, Buscar Caso y Atender peticiones.

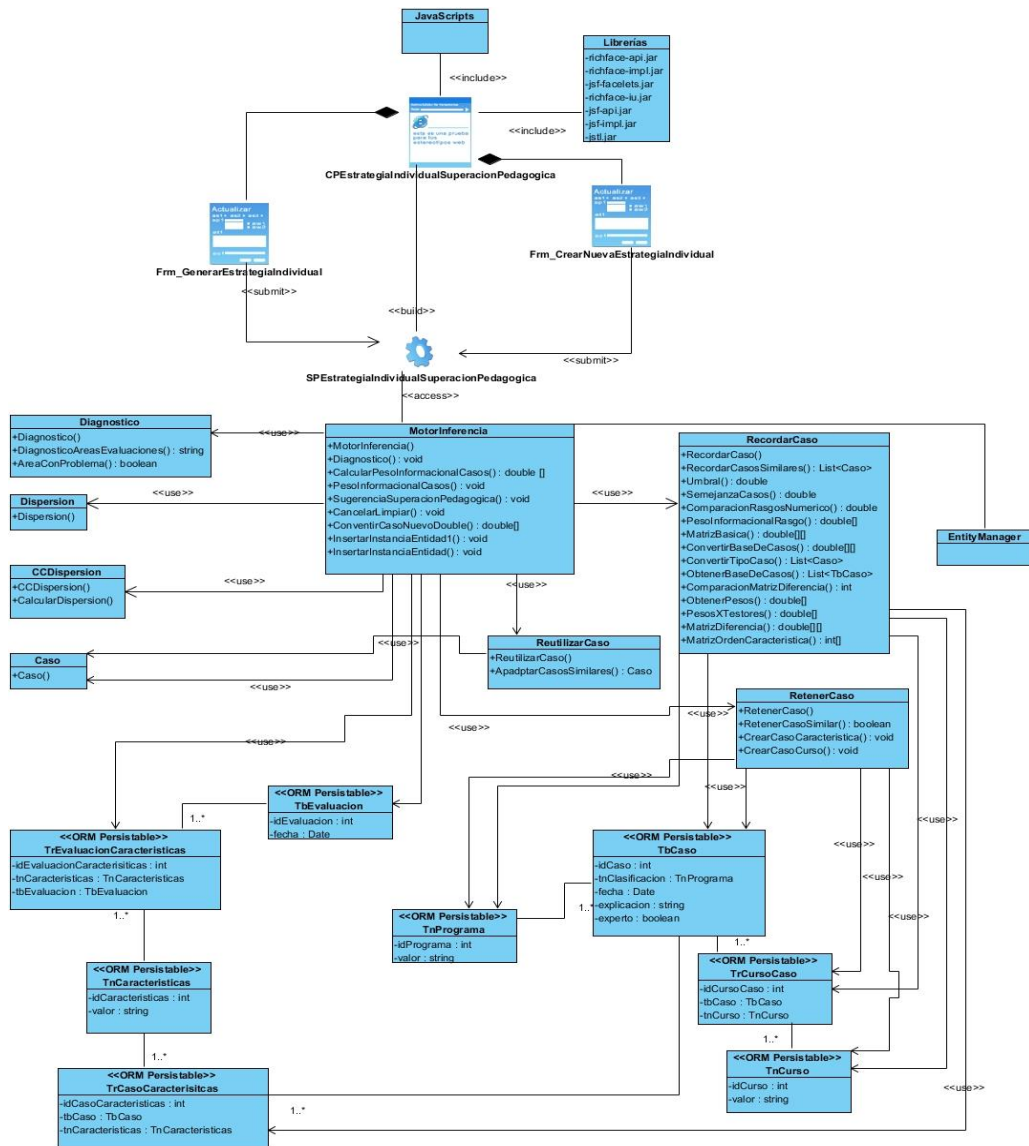


Figura 7: DCD: Generar estrategia individual de superación pedagógica.

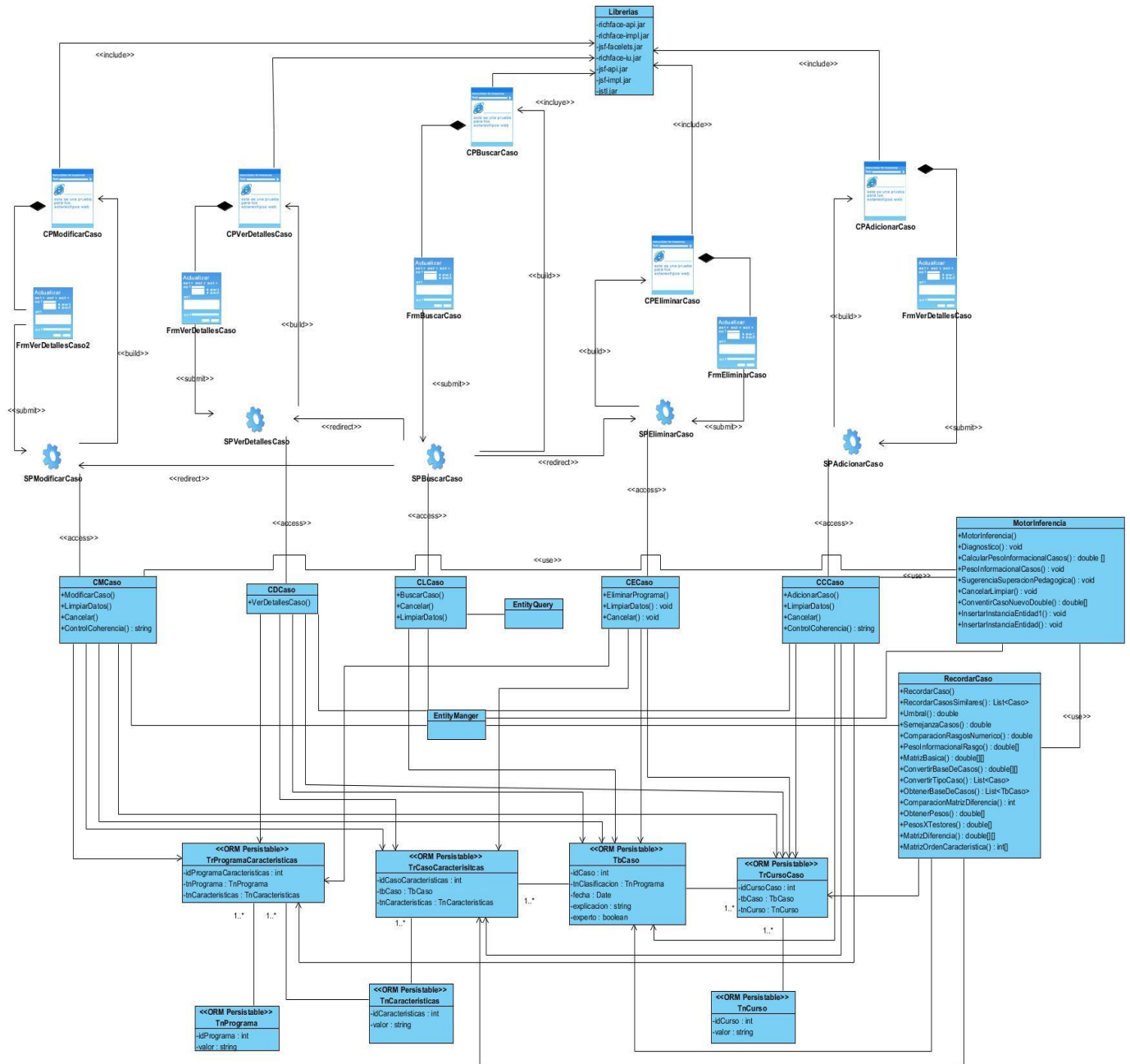


Figura 8: DCD: Buscar Caso.

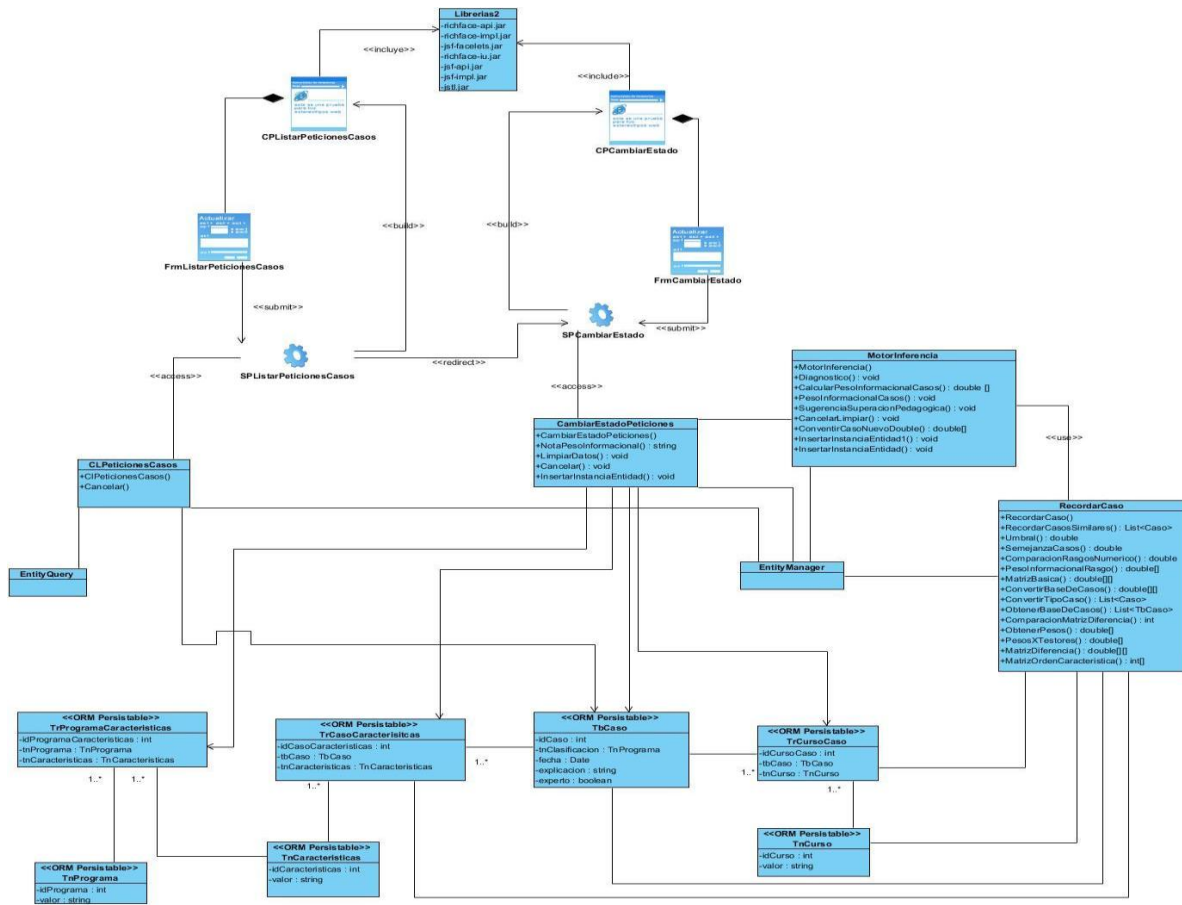


Figura 9: DCD: Atender peticiones.

Conclusiones parciales

- La caracterización de la generación de estrategias individuales de superación pedagógica refleja que la mayoría de los J.Dpto emplean insuficientes vías para la obtención del conocimiento sobre un profesor, poseen índices bajos en las categorías docentes Auxiliar o Titular, así como poca experiencia durante este proceso.
- El estudio del dominio de aplicación junto a las demandas específicas del cliente, permitió identificar 12 requisitos funcionales y 10 no funcionales, para garantizar el correcto funcionamiento del SE.
- El análisis por caso de uso del sistema, junto a los representación de los clases del diseño constató el funcionamiento del SE, el cual se favorece con el estilo arquitectónico cliente/servidor y el patrón MVC.

CAPÍTULO 3. IMPLEMENTACIÓN Y VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

3.1. Especificación de los componentes del SBC propuesto

Sobre la base de los fundamentos teóricos, definidos en el epígrafe 1.3, se especifican los componentes del SBC propuesto. A continuación se expone la secuencia de pasos para cada componente, con el objetivo de guiar el desarrollo del SBC.

Base de casos

1. Definir los rasgos predictivos y los rasgos objetivos.
2. Determinar el dominio de definición de cada rasgo.
3. Representar los casos.

Recuperar

4. Determinar el peso informacional de cada rasgo.
5. Definir las funciones de comparación de rasgos.
6. Definir la función de semejanza entre casos.
7. Obtener los casos más semejantes.

Adaptar

8. Determinar el peso informacional de cada caso.
9. Obtener la clase que predomina en la lista de los casos más similares.
10. Obtener una solución para el nuevo caso.

Revisar

11. Coherencia del nuevo caso respecto a la BC.

Retener

12. Almacenar el nuevo caso como nuevo conocimiento.
13. Actualizar los pesos informacionales de los rasgos y los casos.

Se procede a continuación a explicar los resultados de la aplicación de cada uno de los pasos descritos anteriormente.

Paso 1. Definición de los rasgos predictivos y objetivos

Para definir los rasgos predictivos y objetivos, se utilizó el conocimiento de los expertos en cuanto a su criterio de evaluación para la generación de la estrategia individual de superación pedagógica de un profesor. De igual forma fue de vital importancia, el estudio del documento ESPC-UCI, ya que constituye un referente teórico para esta investigación.

Con la cohesión del conocimiento adquirido por las fuentes consultadas se definieron 34 rasgos predictivos, agrupados en 5 categorías, y como rasgos objetivos se obtuvo, el tipo de programa al cual debe incorporarse el profesor y el conjunto de acciones de postgrado que garantizan su superación.

Tabla 13: Categoría de los rasgos predictivos y rasgos objetivos.

Categoría de los rasgos predictivos	Rasgos Objetivos
Socio-Afectiva	Tipo de Programa
Pedagogía y Didáctica General	Acciones de Postgrado
Conocimientos del área	
Controles a clases	
Evaluación profesoral	

Paso 2. Dominio de definición de cada rasgo

En este paso se procede a determinar el dominio de definición de cada rasgo, se muestran en la siguiente tabla agrupados en las 5 categorías establecidas, en la tabla 3.2 se puede observar el dominio de definición para cada rasgo predictivo.

Tabla 14: Dominio de definición de cada categoría.

Categoría de los rasgos predictivos	Tipo de valor	Dominio
Socio-Afectiva	Entero y Booleano	[2 , 5] y [Si o No]
Pedagogía y Didáctica General	Entero y Booleano	[2 , 5] y [Si o No]
Conocimientos del área	Booleano	[Si o No]
Controles a clases	Entero y Booleano	[2 , 5] y [Si o No]
Evaluación profesoral	Entero	[2 , 5]

Paso 3. Representación de los casos

El almacenamiento de los casos se realiza en una BC, la cual se nutre de los casos nuevos generados por el SE, con previa revisión del usuario experto, y los casos previamente almacenados en ella. Coexistiendo entonces los conocimientos, las experiencias brindadas por los expertos y las recreadas por el sistema. En esta base de conocimiento se almacena el conocimiento necesario para resolver los problemas del dominio de aplicación.

La BC propuesta se encuentra almacenada en una base de datos relacional, lo cual permite nutrirse de las ventajas que trae consigo este tipo de estructura. Por lo particular de la BC del SE, en el **anexo 3** se encuentran las descripciones de las principales tablas del modelo de datos. El modelo propuesto se muestra a continuación:

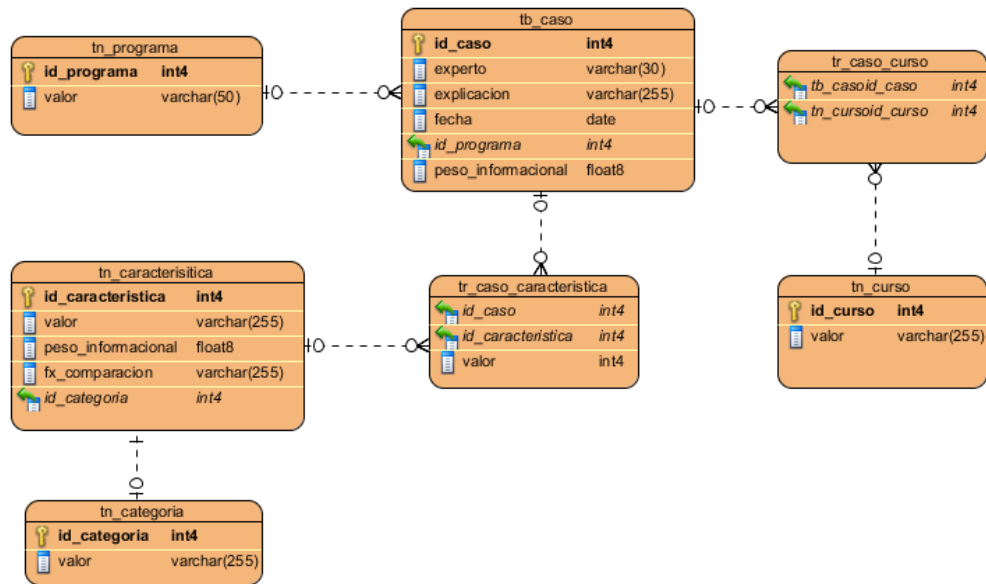


Figura 10: Modelo de datos.

Paso 4. Peso informacional de cada rasgo

Después de identificar los rasgos y su dominio de definición, constituye un elemento fundamental definir el peso informacional de cada uno de estos rasgos, siendo algunos más significativos que otros. En la propuesta se procedió a obtener la importancia de cada rasgo mediante la utilización del cálculo de los testores típicos.

La BC del SE, según (Zhuravliov, y otros, 1972), constituye la matriz de aprendizaje. A esta matriz se le aplica un criterio de comparación de disimilitud tipo *booleano*. El propio autor define la siguiente función de comparación, utilizada en la propuesta.

$$\delta_i(O_0, O_t) = \begin{cases} 0 & \text{si } x_i(O_0) = x_i(O_t) \\ 1 & \text{e.o.c} \end{cases}$$

Luego de aplicar la función de comparación para cada par de casos de diferentes clases, se obtiene una matriz *booleana* llamada matriz diferencia. Mediante esta matriz, siguiendo el procedimiento propuesto por (Ruiz-Shulcloper, 2009), (Zhuravliov, y otros, 1972), se obtiene la matriz básica y de ella los valores de frecuencia relativa $P(x)$ y longitud $L(x)$ de cada rasgo. El peso $D(x)$ para cada uno de los rasgos se obtiene a partir de la fórmula siguiente:

$$D(x) = \alpha P(x) + \beta L(x) \text{ con } \alpha, \beta > 0 \text{ y } \alpha + \beta = 1$$

El significado de α y β se explicó en el epígrafe 1.3. Para la propuesta, se determina el mismo valor de importancia a la frecuencia relativa y a la longitud del testor donde se encuentra el rasgo, o sea $\alpha = \beta = 0,5$. Para una BC compuesta por 50 casos se obtuvieron los siguientes pesos por rasgo.

Tabla 15: Peso informacional inicial de cada rasgo.

Categorías	Rasgos predictivos	Tipo de valor	Dominio	Peso [0-1]
Socio-Afectiva	Comunicación con los estudiantes	Entero	[2 , 5]	0.423
	Visita a la residencia	Booleano	Si o No	0.235
	Participación en la ejecución del proyecto educativo	Booleano	Si o No	0.365
	Apoyo incondicional a las misiones de la universidad	Booleano	Si o No	0.523
Pedagogía y Didáctica General	Participación en el diseño del proyecto educativo	Booleano	Si o No	0.351
	Impartición de actividades metodológicas	Entero	[2 , 5]	0.483
	Liderazgo de proyecto de innovación educativa	Booleano	Si o No	0.124
	Membresía a proyecto de innovación educativa	Booleano	Si o No	0.231
	Dirección de estructuras docentes	Booleano	Si o No	0.362
	Premios en Ciencia e Innovación Tecnológica en las Ciencias de la Educación	Booleano	Si o No	0.413
	Obtención de titulaciones científicas en Ciencia e Innovación Tecnológica en las Ciencias de la Educación	Booleano	Si o No	0.423
	Obtención de titulaciones académicas en Ciencia e Innovación Tecnológica en las Ciencias de la Educación	Booleano	Si o No	0.396
	Participación como jurado en tribunales de categorías docentes	Booleano	Si o No	0.365
	Gestión eficiente y racional del conocimiento y la información científica	Entero	[2 , 5]	0.69
Uso eficiente de las TICs	Entero	[2 , 5]	0.736	
Conocimientos del área	Tutorías de tesis de diploma en su perfil de graduado	Booleano	Si o No	0.339
	Tutorías de tesis de maestría en su perfil de graduado	Booleano	Si o No	0.348
	Tutorías de tesis de doctorado en su perfil de graduado	Booleano	Si o No	0.375
	Impartición de acciones de postgrado en su perfil de graduado	Booleano	Si o No	0.422
	Dirección científica en grupos de investigación	Booleano	Si o No	0.521

Categorías	Rasgos predictivos	Tipo de valor	Dominio	Peso [0-1]
	Membresía a grupos de investigación	Booleano	Si o No	0.425
	Premios en Ciencia e Innovación Tecnológica en su perfil de graduado	Booleano	Si o No	0.381
	Obtención de titulaciones científicas en Ciencia e Innovación Tecnológica en su perfil de graduado	Booleano	Si o No	0.376
	Obtención de titulaciones académicas en Ciencia e Innovación en su perfil de graduado	Booleano	Si o No	0.396
	Participación como jurado en tribunales de grado científico, estudiantiles y del Fórum de Ciencia y Técnica	Booleano	Si o No	0.511
Controles a clases	Cumplimiento del objetivo	Booleano	Si o No	0.823
	Evaluaciones de los controles a clases	Entero	[2, 5]	0.863
	Trabajo político ideológico	Booleano	Si o No	0.723
	Selección y utilización de los medios	Entero	[2, 5]	0.293
	Selección y utilización de los métodos	Entero	[2, 5]	0.316
	Dominio del contenido	Entero	[2, 5]	0.913
	Control de la disciplina	Booleano	Si o No	0.423
Evaluación profesoral	Evaluación de la FEU	Entero	[2, 5]	0.823
	Evaluación de la UJC o PCC	Entero	[2, 5]	0.853

Paso 5. Selección de las funciones de comparación de rasgos

En correspondencia al tipo de valor que toman los rasgos y las características propias del dominio de aplicación, se definen las siguientes funciones de comparación:

Rasgos con valores enteros

$$\delta_i(O_0, O_t) = \begin{cases} 1 & \text{si } |x_i(O_0) - x_i(O_t)| = 0 \\ 0.75 & \text{si } |x_i(O_0) - x_i(O_t)| = 1 \text{ y } [x_i(O_0) \& x_i(O_t)] \in [4-5] \\ 0.50 & \text{si } |x_i(O_0) - x_i(O_t)| = 1 \text{ y } [x_i(O_0) \& x_i(O_t)] \in [3-4] \\ 0.25 & \text{si } |x_i(O_0) - x_i(O_t)| = 1 \text{ y } [x_i(O_0) \& x_i(O_t)] \in [2-3] \\ 0 & \text{e.o.c} \end{cases}$$

Rasgos con valores booleanos

$$\delta_i(O_0, O_t) = \begin{cases} 1 & \text{si } x_i(O_0) = x_i(O_t) \\ 0 & \text{e.o.c} \end{cases}$$

$\delta_i(O_0, O_t)$: función de comparación entre los casos O_0 y O_t atendiendo al valor X del rasgo i .

Paso 6. Selección de la función de semejanza entre casos

Después de haber definido el peso informacional de cada rasgo y la función de comparación para cada uno de ellos, se procede a definir una medida de semejanza entre el caso nuevo y cada caso de la BC, teniendo en cuenta los dos factores antes descrito.

$$\beta(O_0, O_t) = \frac{\sum_{i=1}^n p_i \cdot \delta_i(O_0, O_t)}{\sum_{i=1}^n p_i}$$

$\beta(O_0, O_t)$: función de semejanza entre los casos O_0 y O_t .

p_i : peso informacional del rasgo i .

Paso 7. Obtención de los casos más semejantes

Luego de obtener la similitud entre cada caso de la BC y el nuevo caso, se seleccionan los casos más similares al nuevo caso. Para ello se define un umbral de semejanza, si es superado por un caso se considera una solución aceptable. La fórmula de umbral propuesta fue definida por (Ruiz-Shulcloper, 2009).

$$\mu = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^m \beta(O_i, O_j)$$

μ : umbral de semejanza.

m : cantidad de casos de la BC.

En la medida que se recuperen más casos, el costo computacional aumenta considerablemente. Por esta razón, si la cifra de casos recuperados supera los 10, sólo se recuperan los 10 casos más similares al nuevo caso.

Paso 8. Peso informacional de cada caso

De manera similar a la importancia del cálculo del peso informacional de cada rasgo para la función de semejanza entre casos, resulta significativo definir el peso informacional para cada caso u objeto respecto a la clase en la que fueron clasificados. Sobre el tema, las fórmulas más usadas en la bibliografía consultada se manifiestan en la propuesta de (Zhuravliov, y otros, 1972), siendo esta la utilizada en el sistema.

$$P(O_k) = \frac{\sum_{i=1}^{n_x} \beta(O_k, O_i)}{n_x}$$

$P(O_k)$: peso informacional del caso k en la clase x .

$\beta(O_k, O_i)$: semejanza entre el O_k y el O_i .

n_x : cantidad de casos en la clase x .

Paso 9. Obtención de la clase predominante entre la lista de los casos más similares

Como resultado del paso 7, se obtienen los casos más semejantes al nuevo caso. En este grupo resulta factible analizar, cómo ha sido clasificado cada uno, con el objetivo de seleccionar solo aquellos casos más veces clasificados en una misma clase. A este paradigma clasificador se le conoce como K vecinos próximos.

COMIENZO

Entrada: $D = \{(x_1, c_1), \dots, (x_N, c_N)\}$

$x = (x_1, \dots, x_n)$ nuevo caso a clasificar

PARA todo objeto ya clasificado (x_i, c_i)

 calcular Función de semejanza

Quedarnos con los K casos D_x^K ya clasificados más cercanos a x

FIN

Figura 11: Pseudocódigo para el clasificador K -vecinos próximos. [Modificado de (Moujahid, y otros, s.f)]

Paso 10. Obtención de la solución para el nuevo caso

Después de filtrar los casos más similares en el paso anterior y obtener la lista de casos más similares de la clase predominante, sólo resta definir de este conjunto el caso que puede convertirse en solución al nuevo caso. Para ello se tendrá en cuenta, tanto la semejanza de cada caso con el nuevo, como el nivel de representación de cada caso respecto a su clase, o sea el peso informacional del caso. Esta fórmula fue demostrada y aplicada en escenarios reales por (Ruiz-Shulcloper, 2009).

$$D(O_n) = \alpha \beta(O_n, O_k) + \beta P(O_k) \text{ con } \alpha, \beta > 0 \text{ y } \alpha + \beta = 1$$

Las variables α y β obtienen el mismo significado de la fórmula del peso informacional de cada rasgo, pero en esta ocasión refiere a concederle importancia a la semejanza entre casos y al peso informacional de los mismos. Como ya se planteó anteriormente, para la propuesta se trabajará con α y β igual a 0,5. De esta manera se obtiene un caso que representa bien su clase, y a la vez se asemeja en gran medida al nuevo caso, lo que significa que el nuevo caso debe pertenecer a dicha clase.

Paso 11. Control de coherencia del nuevo caso respecto a la BC

Una vez propuesta la solución al usuario, este puede aceptarla o bien crear una nueva. En cualquier caso, este nuevo conocimiento no se almacena directamente en la BC, sino que es revisado por los expertos. El sistema propuesto le brinda a los expertos una sugerencia basada en el peso informacional del nuevo caso respecto a la clase que fue clasificado. Según (Ruiz-Shulcloper, 2009) «*el caso de mayor peso informacional, es aquel que mejor representa una clase determinada y viceversa*». Basándose en esta idea, se puede inferir que, si el caso que se desea insertar tiene menos peso informacional que el caso que menos representa la clase, entonces no se recomienda incluir el caso en esta clase.

Paso 12. Almacenamiento del nuevo caso como nuevo conocimiento

Se debe comprobar que el nuevo conocimiento no entre en contradicción o sea redundante con el resto, aun cuando el experto esté de acuerdo con insertar el nuevo caso como nuevo conocimiento. Para ello el sistema propuesto verifica, (1) que no exista un caso exactamente igual en la BC y (2) que no exista un caso con los mismos rasgos predictivos clasificado en otra clase (rasgo objetivo). Para los casos anteriores el sistema emite una alerta al experto informando lo que pretende insertar; en cualquier otro caso el sistema sitúa el caso en estado de espera de asignación de peso informacional, para luego ser usado como nuevo conocimiento.

Paso 13. Actualización de los pesos informacionales de los rasgos y los casos

Como parte del módulo de retención o almacenamiento, es necesario asignar el peso informacional a cada caso revisado por los expertos; de igual modo al agregar un nuevo conocimiento a la BC se debe actualizar los pesos de los rasgos. Para realizar este proceso se espera a que exista una cantidad considerable de casos para insertar, ya que uno a uno implica un costo computacional alto en todo momento. Para la propuesta se realizará el proceso cada vez que existan 50 casos nuevos revisados y aprobados por los expertos.

3.2. Diagrama de Despliegue

El Diagrama de despliegue describe la distribución física de los distintos nodos que componen el sistema, y la relación que se establece entre ellos. Estos nodos representan los elementos de software que entran en la realización del sistema y se relacionan mediante soportes bidireccionales. Su objetivo es mostrar cómo y dónde se desplegará el sistema. En la Figura 12 se muestra el Diagrama de despliegue perteneciente al sistema propuesto.

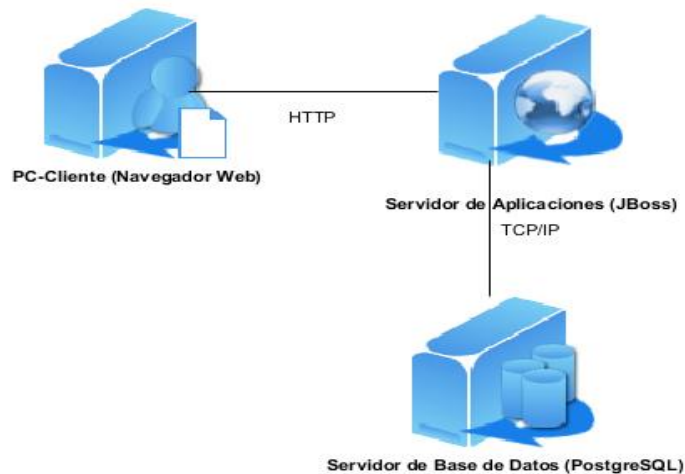


Figura 12: Diagrama de Despliegue.

Descripción de los elementos

- **Estaciones Clientes (Navegador Web):** serán las estaciones de los usuarios, las cuales permitirán acceder al SE mediante el navegador.
- **Servidor de Aplicaciones (Jboss):** certificado por SUN para el estándar JEE5, hospedará la solución integrada, proveerá un clúster para balancear la carga de peticiones hechas por los usuarios, garantizando la disponibilidad de la información mostrada.
- **Servidor de Bases de Datos (PostgreSQL):** residirá toda la información operacional (BC).

3.3. Tratamiento de excepciones

Las excepciones son definidas como errores que interrumpen el flujo normal de las sentencias de un programa. Realizar el tratamiento de estas excepciones le brinda robustez y flexibilidad al sistema, ya que permite que el programa siga ejecutándose aun teniendo un error, cuando este error ocurre dentro de un método Java, automáticamente se crea un objeto “Excepción”, el cual es tratado en el sistema de

ejecución. Este objeto contiene información sobre la excepción, incluyendo su tipo y el estado del programa cuando ocurrió el error. Para cada fragmento de código donde se espere una situación anómala, se definen las excepciones correspondientes, para luego ser tratadas evitando la interrupción del sistema. La siguiente figura hace referencia a su uso en una de las clases del sistema desarrollado:

```
public void insertarInstanciaEntidad() {
    try {
        Caso objCasoAux = new Caso();
        objRetenerCaso = new RetenerCaso(entityManager);

        objCasoAux = casoExperto;
        if (objCasoAux.getConfianza() < 100.0) {

            if (objRetenerCaso.RetenerCasoSimilar(objCasoAux, objEvaCarac,
                "Espera")) {
                facesMessages
                    .add("Se insertado una nueva sugerencia dada por el sistema");
            } else {
                facesMessages
                    .add("Ha ocurrido un error en el almacenamiento de los datos");
            }
        }

        CancelarLimpiar();
    } catch (Exception e) {

        Object obj = e.getClass();
        String mensaje = obj.toString().substring(6);
        facesMessages.addFromResourceBundle(mensaje);
    }
}
```

Figura 13: Tratamiento de Excepciones.

En caso de ocurrir una excepción que implique una redirección, se maneja mediante los *.pages.xml*, los mismos se encargan de capturar globalmente las excepciones y ejecutar las instrucciones determinadas. Incorporado a esto se utiliza el componente *Seam FacesMessages*, propio del framework Seam, que captura el mensaje de excepciones provenientes de las clases controladoras, y permite mostrarlo en la interfaz de usuario.

Por otra parte, durante el proceso de ejecución del sistema pueden ocurrir otras situaciones imprevistas. Resulta impracticable obtener una respuesta del sistema para cada situación, pero si se pueden controlar algunas de ellas. A continuación se muestra una tabla con algunos ejemplos de situaciones detectadas:

Tabla 16: Tratamiento de situaciones anómalas.

Situación anómala	Respuesta del sistema
No se supera el umbral de aceptación de casos similares.	Muestra el mensaje: “El sistema no pudo generar una estrategia para el profesor.”
No se genera una estrategia por parte del sistema y se crea una nueva estrategia.	Muestra el mensaje: “Los valores de dispersión no están disponibles porque no existe una estrategia por parte del sistema.”
Se desea insertar un caso nuevo con campos obligatorios vacíos.	Muestra el mensaje encima del campo obligatorio: “*, Valor requerido”


3.4. Seguridad

La seguridad de sistema de información constituye un elemento de gran importancia. Para obtener un sistema seguro se debe garantizar la integridad, confidencialidad y disponibilidad de la información. La mezcla de estas características trae consigo que la información sea modificada y vista por el personal autorizado. El SE propuesto se adjudicará la seguridad proporcionada por el sistema de gestión al cual se integre.

3.5. Principales funcionalidades

Para tener una mejor comprensión sobre el sistema desarrollado, se muestran a continuación las funcionalidades más importantes implementadas. La figura 14 muestra la funcionalidad **Generar estrategia individual de superación pedagógica**. Se observa la propuesta de superación pedagógica que proporciona el SE. De igual forma permite crear una nueva propuesta de superación pedagógica, en caso de no coincidir con la emitida por el sistema.

Datos generales del docente



Nombre: Leonardo Yuniel **Apellidos:** Palmero Remón
Carnet: 88021320566 **Sexo:** Masculino

Evaluaciones del docente

Socio-Afectiva	Pedagogía y Didáctica General	Conocimientos del Área	Controles a Clases	Evaluación Profesional
Áreas evaluadas				Evaluación
Comunicación con los estudiantes				2
Visita a las residencias				No
Participación en la ejecución del proyecto educativo				No
Apoyo incondicional a las misiones de la universidad				No

[+ Nueva sugerencia](#)

Sugerencia de superación pedagógica.

Tipo de programa	Cursos
Programa completo	Entrenamiento "La clase en la Educación Superior"
	Entrenamiento "Formación a través del vínculo universidad – industria"
	Entrenamiento "Formación a través de las TIC"
	Diplomado "Formación Pedagógica Básica"
	Curso de postgrado "El Trabajo Metodológico en la Educación Superior"
	Maestría en Educación a Distancia
	Maestría en Procesos educativos sustentados en el vínculo universidad – industria y las TIC
	Doctorado en Ciencias Pedagógicas
	Doctorado en Ciencias de la Educación – Especialidad Tecnología Educativa
Diplomado "Formación Pedagógica Superior"	

El profesor presenta insuficiente preparación en cada una de las categorías evaluadas.

Aceptar **Cancelar**

Figura 14: Funcionalidad. Generar estrategia individual de superación pedagógica.

La funcionalidad **Crear nueva propuesta de superación pedagógica**, brinda la opción al usuario de confeccionar su propia solución. Por otra parte, el sistema muestra en todo momento la dispersión que existe entre la nueva propuesta y la propuesta dada por el sistema. En siguiente figura se observa lo antes descrito.

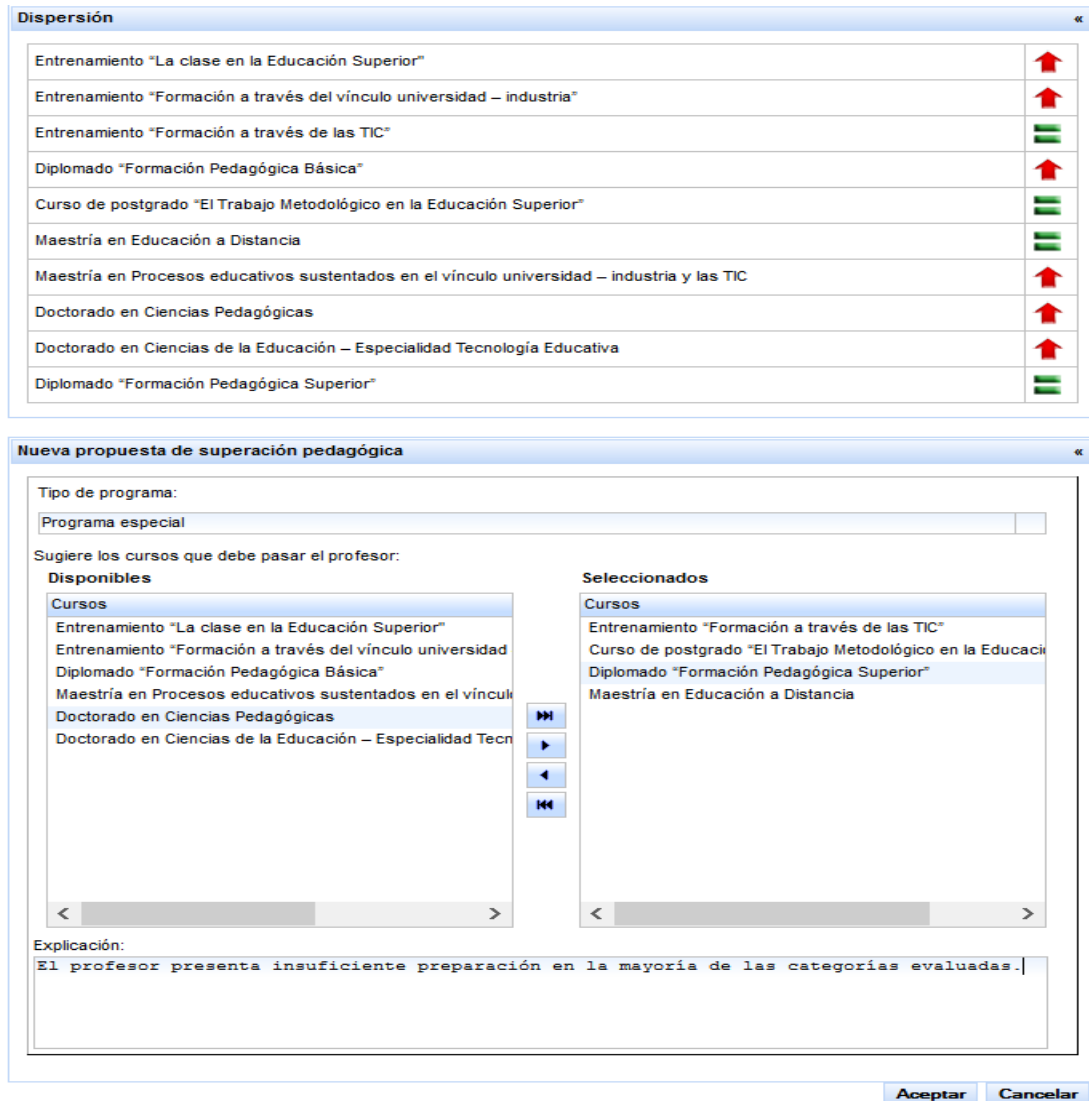


Figura 15: Funcionalidad. Crear nueva propuesta de superación pedagógica.

En la figura 16 se muestra la funcionalidad **Atender peticiones**, esta opción permite al usuario (experto) aceptar o rechazar la propuesta de superación pedagógica creada por otro usuario (directivo).

Tipo de programa				
Programa completo				
Socio-Afectiva	Pedagogía y Didáctica General	Conocimientos del Área	Controles a Clases	Evaluación Profesoral
Áreas evaluadas				Evaluación
Comunicación con los estudiantes				4
Visita a las residencias				No
Participación en la ejecución del proyecto educativo				No
Apoyo incondicional a las misiones de la universidad				No
Sugerencia de cursos				
Curso de postgrado "El Trabajo Metodológico en la Educación Superior"				
Diplomado "Formación Pedagógica Superior"				
Diplomado "Formación Pedagógica Básica"				
Entrenamiento "Formación a través del vínculo universidad – industria"				
Explicación				
El profesor presenta insuficiente preparación en la mayoría de las categorías evaluadas.				
Nota				
Se acepta revisado				
Estado				
<input type="text"/>				
<input type="button" value="Aceptar"/> <input type="button" value="Cancelar"/>				

Figura 16: Funcionalidad. Atender peticiones.

3.6. Estándar de codificación

Los estándares de codificación, de acuerdo con (Smania, y otros, 2008), «se definen por el equipo de desarrollo para lograr estandarización en la programación del software. Se centran en el aspecto y la estructura física y no en la lógica del programa. El uso de estándares ofrece ventajas ya que, contribuyen a facilitar la lectura, comprensión, mantenimiento y reutilización del código a lo largo del proceso de desarrollo de un software». A continuación se describen varias estrategias de codificación, utilizadas en la implementación del sistema, definidas por el equipo de desarrollo.

Se debe usar siempre una línea en blanco en las siguientes circunstancias

- Entre métodos.

- Entre las variables locales de un método y su primera sentencia.
- Antes de un comentario de bloque o un comentario de una línea.
- Entre las distintas secciones lógicas de un método.

Respecto a la longitud de la línea

- Evitar las líneas de más de 80 caracteres, ya que no son bien manejadas por la mayoría de las terminales y herramientas.

Respecto a las normas de inicialización, declaración y colocación de variables, constantes, clases y métodos

- Los nombres de las variables deben ser cortos pero con significado.
- Los nombres de las clases deben tener la primera letra de cada palabra que lo forma en mayúscula. Mantener los nombres de las clases simples y descriptivas.
- Los métodos deben ser verbos y tener la primera letra de cada palabra que lo forma en mayúscula
- Las variables deben tener la primera letra en minúscula y si son compuestas, el resto de las palabras que la conforman, su primera letra en minúscula.

3.7. Prueba y validación

Una estrategia de prueba, según (Pressman, 2005), debe incluir pruebas de bajo nivel (necesarias para confirmar la correcta implementación de un pequeño segmento de código) y de alto nivel (que validen las principales funciones del sistema a partir de los requisitos del cliente). El propio autor lo reafirma cuando expresa que las pruebas comienzan a nivel de componentes y trabaja hacia afuera, hacia la integración de todo el software. Para el sistema propuesto se defendieron los criterios de Pressman antes descritos, primeramente se realizarán pruebas unitarias a las principales funcionalidades de la aplicación y luego se probará el mecanismo de inferencia como un todo, mediante la técnica de validación cruzada.

Pruebas unitarias

El objetivo de las pruebas unitarias, según (Barnes, y otros, 2007), es el aislamiento de partes del código y la demostración de que estas partes no contienen errores. Las pruebas unitarias son una forma de probar el correcto funcionamiento de un módulo de código, esto permite asegurar el funcionamiento correcto de cada uno de los módulos.

Para realizar las pruebas unitarias se utilizó el framework JUnit, el cual brinda un conjunto de librerías que se integran fácilmente al IDE de desarrollo seleccionado, Eclipse. JUnit permite la realización de las

pruebas a los métodos de las clases implementadas. Para hacer uso de este framework se definieron los siguientes pasos:

Paso 1. Definición de los métodos a probar

Según (Pressman, 2005), realizar pruebas a todos los caminos lógicos del software resulta impracticable. El propio autor propone seleccionar y comprobar un número limitado de rutas lógicas importantes. De acuerdo con lo antes descrito, (Tabla 17) se seleccionaron sólo aquellos métodos que resultan de vital importancia para el funcionamiento del sistema.

Tabla 17: Métodos para las pruebas unitarias.

Métodos
+PesoInformacionalCasos();
+Umbral();
+SemejanzaCasos();
+MatrizBasica();
+PesoInformacionalRasgos();
+ComparacionRasgosNumericos();
+MatrizDiferencia();

Paso 2. Adaptación de los métodos al framework JUnit

Para la utilización de los métodos con JUnit, es preciso realizarles algunas modificaciones que no cambian la lógica del negocio. Asimismo es necesario definir que, (1) los métodos testados sean de tipo *void*, (2) identificar juegos de datos para probar cada método y (3) agregar en la parte superior del método la anotación `@test`.

```
package test.com.programarenjava.tutoriales.junit;  
  
import static org.junit.Assert.*;  
  
public class JUnitTest {  
  
    @Test  
    public void testRecordarCaso() {  
        fail("Lógica de negocio");  
    }  
  
}
```

Figura 17: Código generado con JUnit en Eclipse.

Paso 3. Realización de las pruebas

El resultado de la prueba realizada a los métodos principales del mecanismo de inferencia fue satisfactorio (Figura 18). Además se muestran los tiempos de ejecución para un total de 100 casos iniciales.

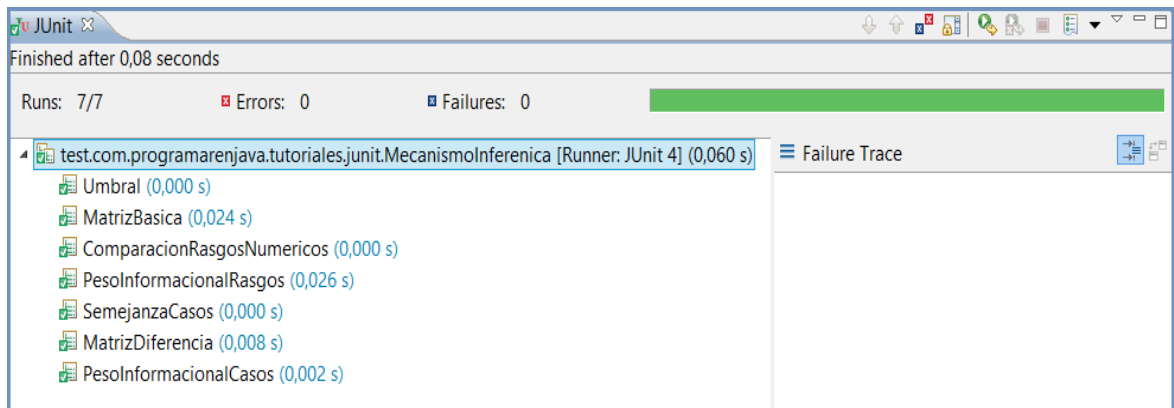


Figura 18: Pruebas unitarias con JUnit en Eclipse.

Validación cruzada

La validación cruzada, según (Devijver, y otros, 1982), «es una técnica utilizada para evaluar los resultados de análisis estadístico y garantizar que son independientes de la partición entre datos de

entrenamiento y prueba. Se utiliza en entornos donde el objetivo principal es la predicción y se quiere estimar cómo de preciso y fiable es un modelo, mecanismo o procedimiento que se llevará a cabo a la práctica». Esta técnica es la comúnmente utilizada para validar mecanismos de inferencia dentro del área de la IA.

Para la validación de la hipótesis planteada se utiliza la validación cruzada de K iteraciones. Para ello se dividen los datos de muestra en K subconjuntos, donde uno de los subconjuntos se utiliza como dato de prueba y el resto (K-1) como dato de entrenamiento. A continuación, el modelo se entrena con los (K-1) subconjuntos de entrenamiento y se evalúa con el subconjunto de prueba. Este proceso se repite hasta que cada subconjunto sea utilizado exactamente una vez como prueba. Para evaluar la fiabilidad del mecanismo de inferencia se promedia la precisión obtenida en cada iteración. La muestra seleccionada fue de 30 casos, con validación cruzada de 5 iteraciones. La figura muestra la distribución de los subconjuntos de entrenamiento y prueba.

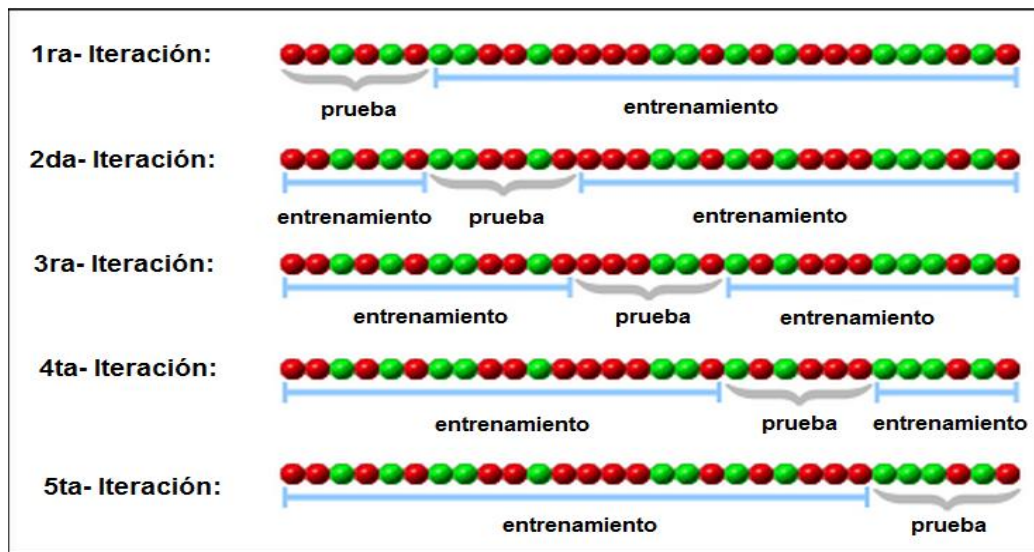


Figura 19: Validación cruzada de 5 iteraciones.

Los datos de prueba se conformaron por 6 casos en cada iteración. Los resultados obtenidos se muestran en la tabla 18.

Tabla 18: Resultados de la validación cruzada.

Iteración	1	2	3	4	5	Total
Cantidad de aciertos	6/6	5/6	6/6	6/6	6/6	29/30

Cantidad de fallos	0/6	1/6	0/6	0/6	0/6	1/30
---------------------------	-----	-----	-----	-----	-----	------

Tasa de aciertos = Cantidad de aciertos / tamaño del subconjunto de prueba

Tasa de errores = Cantidad de fallos / tamaño del subconjunto de prueba

Tasa de dispersión = (Tasa de aciertos – media (Tasa de aciertos de cada iteración)) / tamaño del subconjunto de prueba

Los resultados que se muestran a continuación pertenecen a la evaluación de la fiabilidad del mecanismo de inferencia en cada iteración y la media aritmética.

Tabla 19: Resultados de los indicadores de fiabilidad.

Iteración	1	2	3	4	5	Media
Tasa de aciertos (%)	100	83.33	100	100	100	96.66
Tasa de errores (%)	0	16.67	0	0	0	3.33
Tasa de dispersión (%)	3.34	13.33	3.34	3.34	3.34	5.34

Una vez terminado el proceso de validación cruzada, se concluye que los indicadores Tasa de aciertos y Tasa de dispersión, de la variable Fiabilidad, en la generación de estrategias individuales de superación pedagógica del claustro de la UCI toman valores altos, de igual forma el indicador Tasa de errores de la propia variable se comporta con índices bajos. Los resultados de estos indicadores permiten inferir que el mecanismo utilizado es fiable. Lo anterior constata la validez de la hipótesis planteada y responde al problema científico propuesto.

Conclusiones parciales

- Las entrevistas a los J.Dpto y el estudio del documento ESPC-UCI permitieron identificar 34 rasgos predictivos, divididos en 5 categorías.
- Se definieron dos tipos de función de comparación (numérico y booleano) por el tipo de valor que toman los rasgos predictivos.
- El procedimiento utilizado, descrito en 13 pasos, permitió garantizar la fiabilidad del SE.
- La técnica validación cruzada, junto a la realización de las pruebas unitarias constató la pertinencia de la investigación y la veracidad de la hipótesis planteada.

CONCLUSIONES FINALES

- Los SE han evolucionado en las últimas décadas, siendo múltiples las empresas y sectores productivos que los tienen incorporados dentro de su operación cotidiana. Su funcionamiento, se basa en simular el comportamiento e intentar emitir respuestas a determinadas soluciones como las daría un experto humano en un dominio de conocimiento específico.
- En el proceso de generación de estrategias individuales de superación pedagógica es necesario elevar la fiabilidad, que de acuerdo con las condiciones actuales de dicha variable, puede favorecerse a partir de la utilización de los SE para responder a las insuficiencias en las vías para la obtención del conocimiento, la identificación de las necesidades de superación pedagógica y la necesidad de homogenizar el diseño de las estrategias para los profesores de diferentes departamentos con similares niveles de desarrollo en su cultura profesional pedagógica.
- El SE desarrollado para la generación de estrategias individuales, constituye una herramienta práctica y fiable durante la etapa de diagnóstico de la superación pedagógica, demostrada a través de las pruebas unitarias y la técnica de validación cruzada.

RECOMENDACIONES

- Incluir la fase de recuperación de fallos del módulo de revisión de casos para el tratamiento y corrección del mecanismo de inferencia antes soluciones incorrectas.
- Estructurar la BC de forma jerárquica, con el objetivo de disminuir los tiempos de respuesta del recuperador de casos sin afectar la fiabilidad del mecanismo de inferencia.
- Implementar el método Delphi para lograr un criterio común entre los expertos, durante el proceso de adquisición de nuevo conocimiento para garantizar la experticia y calidad de la BC.

REFERENCIAS BIBLIOGRÁFICAS

- **Aamodt, A. 1994.** *Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches.* United States : AI Communicatios, 1994.
- **Agudo, B. 2002.** *Una aproximación ontológica al desarrollo de Sistemas de Razonamiento Basado en Casos.* Madrid. España.Facultad de Informática.Universidad Complutense de Madrid. : s.n., 2002. 84-669-1851-5.
- **Amandi, Analía. 2012.** *Razonamiento Basado en Casos.* Facultad de Ciencias Exactas. Universidad Nacional del centro de la provincia de Buenos Aires. : s.n., 2012.
- **Ayala, Alejandro Peña. 2006.** *Sistemas Basados en Conocimientos: Una Base para su concepción y desarrollo.* México : s.n., 2006.
- **Barraso, Clara. 1994.** *ENS-AI: UN SISTEMA EXPERTO PARA LA ENSEÑANZA.* España : Teoría de la Educación. Revista Interuniversitaria, 1994. Vol. 6. ISSN: 1130-3743.
- **Bauer, Christian y King, Gavin. 2007.** *Java Persistence with Hibernate.* 2007.
- **Bello, R. E, y otros. 2002.** *Aplicaciones de la Inteligencia Artificial.* Guadalajara.Jalisco.México : Ediciones de la noche, 2002. 970-27-0177-5.
- **Bergmann, R and Goker,Mehmet. 1999.** *Developing Industrial Case Based Reasoning Applications.* Universidad de Trier : The INRECA Methodology, 1999.
- **Bergmann, R, Kolodner, J y Plaza, E. 2006.** *Representation in case-based reasoning.* 2006.
- **Blanco, A. 2003.** *Hipótesis, variables y dimensiones en la investigación educativa.*En COLECTIVO DE AUTORES, *Metodología de la investigación educacional. Desafíos y polémicas actuales.* La Habana : Félix Varela, 2003.
- **Bregón, Anibal. 2005.** *Un sistema de razonamiento basado en casos para la clasificación de fallos en sistemas dinámicos.* Sevilla. España : s.n., 2005.
- **Brito, Mateo Lezcano. 2001.** *Prolog y los Sistemas Expertos.* 2001.
- **Brulé, J. y Bount, A. 1989.** *Knowledge Acquisition.* New York : McGraw-Hill, 1989.
- **Campistrous, L. y Rizo, C. 2003.** *Indicadores e investigación educativa.* En COLECTIVO DE AUTORES, *Metodología de la investigación educacional. Desafíos y polémicas actuales.* La Habana : Félix Varela, 2003.
- **CD-UCI.** *Presentación del Rector al claustro UCI al concluir el 1er semestre del curso escolar 2012 – 2013 [Presentación digital].* La Habana: Universidad de las Ciencias Informáticas, Cuba : s.n.

- **Ciudad, F., y otros. 2013.** *ESTRATEGIA DE SUPERACIÓN PEDAGÓGICA DEL CLAUSTRO*. La Habana : Universidad de las Ciencias Informáticas - Centro de Innovación y Calidad de la Educación, 2013.
- **Cordero, Dasiel, Ruiz, Yadira y Torres, Yoanny. 2013.** Sistema de RBC para la identificación de riesgos de software. *Revista Cubana de Ciencias Informáticas*. 2013, Vol. Vol. 7, 2.
- **Debenham, J. 1989.** *Knowledge System Design*. Sidney : Prentice Hall, 1989.
- **Delgado, Alberto. 1998.** *Inteligencia Artificial y Mini robots. Segunda Edición*. 1998.
- **Figuerola, Noberto. 2014.** *BPMN vs UML. Los Requerimientos y el Modelo del Negocio*. Barcelona,España : PMQuality, 2014.
- **Gálvez, Daniel. 1998.** *Sistemas Basados en el Conocimiento*. Cuba : s.n., 1998.
- **García, J., y otros. 1999.** *Aprenda Java como si estuviera en primero*. Escuela Superior de Ingenieros Industriales de la Universidad del País Vasco (UPV), San Sebastián. : s.n., 1999.
- **García, L. 2006.** *Una concepción pedagógica para la formación ideológica y en valores de los futuros maestros*. Guantánamo: Universidad de Ciencias Pedagógicas. : s.n., 2006.
- **García, R. 2009.** *Metodologías de educación de conocimiento para la construcción de sistemas informáticos expertos*. 2009.
- **Giarratano, Joseph C. y Riley, Gary D. 2005.** *Sistemas Expertos: Principios y Programación*. s.l. : 4ta Ed, 2005.
- **Greenwell, M. 1988.** *Knowledge Engineering for Expert Systems*. Chichester : Ellis Horwood, 1988.
- **Hernández, R., Fernández-Collado, C. y Batptista, P. 2006.** *Metodología de la Investigación. 4ta Edición*. México : McGraw-Hill Interamericana, 2006.
- **IAI-UCI. 2010.** *Informe de autoevaluación institucional de la UCI [Documento interno UCI]*. La Habana: Universidad de las Ciencias Informáticas, Cuba. : s.n., 2010.
- **IFAI-UCI-MES. 2010.** *Informe final de evaluación institucional del MES a la UCI [Documento interno UCI]*. La Habana: Universidad de las Ciencias Informáticas, Cuba : s.n., 2010.
- **Jacobson, Ivan, Booch, Grady y Rumbaugh, James. 2000.** *Proceso unificado de desarrollo de software*. México : Pearson Educacion, 2000.
- **Joyanes Aguilar, Luis. 1996.** *Programación Orientada a Objetos*. España : McGRAW-HILL, 1996. ISBN: 84-481-0590-7.
- **Kolodner, J. 1996.** *Tutorial introduction to case based reasoning. Case based reasoning: Experiences, Lesson and future directions*. Menlo Park AAI Press/MIT : s.n., 1996.

- **Krishnamoorthy, C.S. y Rajeev, S. 1996.** *Artificial Intelligence and Expert Systems for Engineers.* EE.UU : s.n., 1996.
- **Kurzweil, Raymond. 1990.** *The Age of Intelligent Machines.* EE.UU : s.n., 1990.
- **Lamy, J.B, y otros. 2010.** *Testing Methods for Decision Support Systems. Decision Support Systems.* Vukovar,Crotia : CS Jao(Ed.), 2010.
- **Larman, Craig. 1999.** *UML y Patrones, Introducción al análisis y diseño orientado a objetos.* México : Prentice Hall, 1999.
- **Laza, R, Fernández, F y Corchado, J.M. 2012.** *Sistema de RBC para el Soporte a la Toma de Decisiones.* Ourense, España : E.U.I.T. Informática de Gestión, Universidad de Vigo, 2012.
- **León, Maikel. 2007.** *La Inteligencia Artificial en la Informática Educativa.* 2007.
- **León, Tomás Quintanar. 2007.** *Sistemas Expertos y sus aplicaciones.* Pachuca de Soto,Hidalgo : s.n., 2007.
- **Letelier, Patricio y Penadés, M. Carmen. 2006.** *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* Valencia : Universidad Politécnica de Valencia, 2006.
- **Magaña, Sara. 2009.** *Estudio comparativo de Lenguajes de Modelado de Procesos de Negocio para su integración en Procesos de Desarrollo de Software dirigido por modelos.* Universidad Carlos III, Madrid : s.n., 2009.
- **Martí, Ileana Pérez. 2012.** *Sistema para la ayuda a la toma de decisiones en el diagnóstico y tratamiento de dislipoproteinemias basado en la ingeniería del conocimiento.* Habana,Universidad De Las Ciencias Informáticas, Dirección De Formación Posgraduada : s.n., 2012.
- **Martin, Maria José. 1996.** *SISTEMA EXPERTO DE ORIENTACIÓN VOCACIONAL PROFESIONAL.* Madrid, España : s.n., 1996.
- **Martínez, Juan Carlos Medina. 2004.** *Análisis Comparativo de Técnicas, Metodologías y Herramientas de Ingeniería de Requisitos.* D.F. México : s.n., 2004.
- **Martínez, Natalia, y otros. 2010.** *El RBC en el ámbito de la Enseñanza/Aprendizaje.* Santa Clara : Departamento de Computación, Universidad Central de Las Villas , 2010.
- **Martínez, P. Díaz. 2011.** *Estrategia extensionista en el ISP "Enrique José Varona".* La Habana: Instituto Superior Pedagógico "Enrique José Varona". : s.n., 2011.
- **Meyer, M. y Booker, J. 1991.** *Eliciting and Analyzing Expert Judgement.* Londres : Academic Press, 1991.

- **Monroy, Fernández Margarita. 2003.** *Modelo de comportamiento de la organización virtual: una aplicación empírica a los sistemas de franquicia.* España, Universidad de Las Palmas de Gran Canaria : s.n., 2003.
- **Moujahid, Abdelmalik, INZA, Inaki y LARRANAGA, Pedro. s.f.** *Tema 5. Clasificadores K-NN.* s.f.
- **Moya, J, Becerra-Ferreiro, A y Chagoyén-Méndez, C. 2012.** *Utilización de Sistemas Basados en Reglas y en Casos para diseñar transmisiones por tornillo sinfín.* Universidad Central Marta Abreu de Las Villas Facultad de Ingeniería Mecánica. Santa Clara. Cuba : s.n., 2012. Vol. 15. ISSN 1815-5944.
- **Mustelier, K. García. 2010.** *Estrategia de superación para los tutores sobre la actividad científica estudiantil en la carrera de Tecnología de la Salud.* s.l. : La Habana:Universidad de Ciencias Médicas de la Habana. Centro de Desarrollo Académico en Salud (CEDAS)., 2010.
- **Navarro, M.I. 2002.** *Una nueva perspectiva para recuperación en Razonamiento Basado en Casos: mejora de la adecuación usando funciones de riesgo.* Granada : DIGIBUG, 2002. 978-84-695-1188-6.
- **Nebendah, Dieter. 1988.** *Sistemas Expertos. Ingeniería y Comunicación.* . Barcelona : s.n., 1988.
- **Nocedo, I. y Abreu,E. 1984.** *Metodología de la investigación pedagógica y psicológica.* La Habana : Pueblo y Educación, 1984.
- **Núñez, J. 2006.** *Postgrado, gestión del conocimiento y desarrollo social (Conferencia ofrecida)* . 2006.
- **ORACLE. 2010.** The Java EE 5 Tutorial for Sun Java System Application Server 9.1. ORACLE. [En línea] 2010. <http://docs.oracle.com/javase/5/tutorial/doc/javaeetutorial5.pdf>.
- **OSD. 2005.** Open Systems Development. Comparativo software WEB vs. software de Escritorio. [En línea] 2005. <http://www.osdglobal.com/faq/desarrollo-software/comparativo-web-vs-escritorio>.
- **Parsaye, Kamran, y otros. 1990.** *AI Experts.* EE.UU : s.n., 1990.
- **Pecos, D. (6 de 2002).** *Daniel Pecos.* Recuperado el 5 de 2014, de POSTGRESQL VS. MYSQL: <http://danielpecos.com/documents/postgresql-vs-mysql/>
- **Pérez, Alonso. 2002.** *Memoria Organizacional Basada en Casos.* Recife. Brasil : Revista de Ciencia, 2002. 1415-3262.
- **—. 1997.** *Modelo para la Representación de una Memoria.* Universidad de Sonora : ITESM, 1997.

- **Pinochet Olave, Ruperto. 2003.** *Los sistemas informáticos expertos de toma de decisiones y la voluntad como elemento de validez del negocio jurídico.* Chile : Ius et Praxis, 2003. ISSN 0718-0012.
- **Pressman, R. 2005.** *Ingeniería de Software. Un enfoque práctico. Sexta Edición.* s.l. : Mc Graw Hill, 2005.
- **Rich, Elaine y Knight, Kevin. 1994.** *Inteligencia Artificial. Segunda Edición.* México : s.n., 1994.
- **Rodríguez, Kathrin. 2010.** *Sistema Inteligente de Soporte a la Toma de Decisiones.* La Habana : Universidad de las Ciencias Informáticas (UCI), 2010.
- **Rojas, L y Leiva, A. 2002.** *Sistemas Expertos.* 2002.
- **Rolston, David W. 1998.** *Principles of artificial intelligence and expert systems development.* EE.UU : s.n., 1998.
- **Ruiz-Shulcloper, José. 2009.** *Reconocimiento Lógico Combinatorio De Patrones: Teoría Y Aplicaciones.* Habana.Cuba : Centro de Aplicaciones de Tecnologías de Avanzada, 2009.
- **Rusell, Stuart y Norving, Peter. 1996.** *Inteligencia Artificial un Enfoque Moderno.* México : s.n., 1996.
- **Sarabia, Esther G. 2002.** *Three-Dimensional Robotic Vision Using Ultrasonic Sensors.* 2002.
- **Schalkoff, Robert J. 1990.** *Artificial Intelligence Engine.* EE.UU : s.n., 1990.
- **Smania, Duarte Sofía y Noelia. 2008.** *Tecnologías en Desarrollo de Software IDE -UTN.* 2008.
- **Solano, A, Yong, G y Camacho, A. 2007.** *Introducción a los Lenguajes de Cuarta Generación (4GL).* 2007.
- **Stubblefield, W.A. y G.F, Luger. 1993.** *Artificial Intelligence: Structures and Strategies for Complex Problem Solving.* s.l. : 2nd Ed, 1993.
- **Tapia, F. 1996.** *Inteligencia Artificial.* España : s.n., 1996.
- **Uddin, M. Ahmed, y otros. 2010.** *Case-Based Reasoning for Medical and Industrial Decision Support Systems.* Germany : Springer-Verlag, 2010. ISBN 978-3-642-14077-8.
- **Vallejo, Morales Pedro. 2007.** *La fiabilidad de los tests y escalas.* Madrid, Universidad Pontificia Comillas, Facultad de Ciencias Humanas y Sociales : s.n., 2007.
- **W3C. 2010.** W3C. [En línea] 2010. <http://www.w3.org>.
- **Yuan, Michael y Heute, Thomas . 2007.** *JBoss Seam. Simplicity and Power Beyond Java EE.* 2007.

- **Yuan, Michael Juntao, Orshalick, Jacob y Heute, Thomas. 2009.** *Seam Framework. Experience the Evolution of Java.* 2009.

GLOSARIO DE TÉRMINOS

Ajax: Asynchronous JavaScript And XML. Técnica de desarrollo web para crear aplicaciones web interactivas.

Algoritmo: Del nombre del matemático árabe Al-Khwarizmi (780 - 850 aprox.). Define el conjunto de instrucciones que sirven para ejecutar una tarea o resolver un problema. Los motores de búsqueda usan algoritmos para mostrar los resultados de búsquedas.

API: Del inglés Application Programming Interface, Interfaz de Programación de Aplicaciones. Serie de rutinas usadas por una aplicación para gestionar generalmente servicios de bajo nivel, realizados por el sistema operativo de la computadora.

GRASP: Patrones generales de software para asignación de responsabilidades, es el acrónimo de GRASP (*General Responsibility Assignment Software Patterns*). Aunque se considera que más que patrones propiamente dichos, son una serie de buenas prácticas de aplicación recomendable en el diseño de software.

Subsistema: Conjunto de elementos interrelacionados que, en sí mismo, es un sistema, pero a la vez es parte de un sistema superior.

Inferencia: Deducción de una cosa a partir de otra, conclusión.

Framework: Conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar.

POJO: Instancia de una clase que no extiende, ni implementa de otra clase o interfaz.

Tecnología: Conjunto de conocimientos técnicos, científicamente ordenados, que permiten diseñar y crear bienes y servicios, para facilitar la adaptación al medio ambiente y satisfacer tanto las necesidades esenciales como los deseos de la humanidad.

ANEXOS

Anexo 1. Encuesta para los J.Dpto del claustro UCI

Marca con una X las vías de obtención de la información sobre un profesor, que utiliza para conocer las áreas con preparación y aquellas en la que debe superarse pedagógicamente.

Tabla 20: Entrevista para los J.Dpto del claustro de la UCI.


Vías de obtención de la información	Marque con una X
Autoevaluación del docente	
Evaluación externa	
Evaluaciones de los controles a clases	
Evaluación profesoral del docente	
Otras	

Anexo 2. Especificación de casos de uso

Tabla 21: CU: Generar estrategia individual de superación pedagógica.

CASO DE USO:	Generar estrategia individual de superación pedagógica
Resumen:	El caso de uso inicia cuando el actor selecciona un Profesor del listado de profesores, el sistema muestra los datos generales y evaluaciones del profesor, y permite generar la estrategia de superación pedagógica, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico
Precondiciones:	El profesor seleccionado debe tener al menos una evaluación.
REFERENCIAS	
Actores:	Experto Directivo
Requisitos:	RF8, RF9, RF10, RF11
Casos de Uso:	Crear nueva Estrategia Individual de Superación Pedagógica.
FLUJO NORMAL DE EVENTOS	
Acción del Actor	Respuesta del Sistema
1. El caso de uso inicia cuando el actor selecciona un Profesor del listado de profesores evaluados.	
	2. Muestra los datos: Generales del profesor: <ul style="list-style-type: none"> • Nombre • Apellidos

	<ul style="list-style-type: none"> • Carnet • Sexo <p>Evaluaciones del profesor:</p> <ul style="list-style-type: none"> • Áreas evaluadas • Evaluación <p>Sugerencia de Superación Pedagógica:</p> <ul style="list-style-type: none"> • Tipo de programa • Cursos • Justificación <p>y permite:</p> <ul style="list-style-type: none"> • Aceptar generar Propuesta de Superación Pedagógica. • Cancelar operación. Ver Alternativa 1: “Cancelar operación.” • Crear nueva Propuesta de Superación Pedagógica. Ver Alternativa 2: “Crear nueva Propuesta de Superación Pedagógica”.
3. Selecciona la opción Aceptar generar Propuesta de superación pedagógica.	
	4. Asigna la propuesta de superación pedagógica al profesor seleccionado.
	5. El caso de uso termina.

Datos generales del docente				
	Nombre: Leonardo Yuniel Apellidos: Palmero Remón Carnet: 88021320566 Sexo: Masculino			
Evaluaciones del docente				
Socio-Afectiva	Pedagogía y Didáctica General	Conocimientos del Área	Controles a Clases	Evaluación Profesional
Áreas evaluadas				Evaluación
Comunicación con los estudiantes				2
Visita a las residencias				No
Participación en la ejecución del proyecto educativo				No
Apoyo incondicional a las misiones de la universidad				No
+ Nueva sugerencia				
Sugerencia de superación pedagógica.				
Tipo de programa	Cursos			
Programa completo	Entrenamiento "La clase en la Educación Superior"			
	Entrenamiento "Formación a través del vínculo universidad – industria"			
	Entrenamiento "Formación a través de las TIC"			
	Diplomado "Formación Pedagógica Básica"			
	Curso de postgrado "El Trabajo Metodológico en la Educación Superior"			
	Maestría en Educación a Distancia			
	Maestría en Procesos educativos sustentados en el vínculo universidad – industria y las TIC			
	Doctorado en Ciencias Pedagógicas			
	Doctorado en Ciencias de la Educación – Especialidad Tecnología Educativa			
Diplomado "Formación Pedagógica Superior"				
El profesor presenta insuficiente preparación en cada una de las categorías evaluadas.				
<input type="button" value="Aceptar"/> <input type="button" value="Cancelar"/>				
FLUJOS ALTERNOS				
Alternativa 1. "Cancelar operación."				
Acción del Actor	Respuesta del Sistema			
1. Selecciona la opción de Cancelar operación.				
	2. Regresa a la vista anterior.			
	3. El caso de uso termina.			
Alternativa 2. "Crear nueva Propuesta de Superación Pedagógica."				

Acción del Actor	Respuesta del Sistema
1. Selecciona la opción Nueva sugerencia.	
	2. Muestra los datos para crear la nueva Propuesta de Superación Pedagógica. Ver caso de uso: Crear nueva Propuesta de Superación Pedagógica.
	3. El caso de uso termina.
Poscondiciones	Se asignó la propuesta de superación pedagógica al profesor seleccionado.

Tabla 22: CU: Buscar caso.

CASO DE USO:	Buscar caso
Resumen:	El caso de uso inicia cuando el actor accede a la opción Buscar caso, el sistema brinda la posibilidad de introducir criterios de búsqueda para localizar el Caso, el actor introduce los datos que considera como criterios para realizar una búsqueda, el sistema busca y muestra los Casos que cumplen con los criterios de búsqueda y permite realizar varias opciones, el caso de uso termina.
Complejidad:	Alta
Prioridad:	Crítico
Precondiciones:	No existen
REFERENCIAS	
Actores:	Experto
Requisitos:	RF1, RF2, RF3, RF4, RF5
FLUJO NORMAL DE EVENTOS	
Acción del Actor	Respuesta del Sistema
1. El caso de uso inicia cuando el actor accede a la opción Buscar caso.	
	2. Muestra un listado de Casos con los siguientes atributos: <ul style="list-style-type: none"> • Identificador • Programa • Explicación • Fecha Mostrando la cantidad de elementos configurada para mostrar por página, permitiendo navegar por el resultado. Si no se encuentra ningún Caso. Ver Alternativa 1 : “No se encuentra información.”

	<p>Permite:</p> <ul style="list-style-type: none"> • Realizar una búsqueda simple. Ver Alternativa 2: “Realizar una búsqueda simple.” • Cancelar operación. Ver Alternativa 3: “Cancelar operación.” • Crear caso. Ver Alternativa 4: “Crear caso”. • Ver detalles del caso. Ver Alternativa 5: “Ver detalles del caso”. • Modificar caso. Ver Alternativa 6: “Modificar caso”. • Eliminar caso. Ver Alternativa 7: “Eliminar caso”. <p>3. El caso de uso termina.</p>
--	--

Buscar Caso

Programa, explicación

Buscar **Cancelar**

[+ Añadir](#)

Listado de casos			
Programa	Explicación	Fecha	
Programa completo	El profesor presenta insuficiente preparación en cada una de las categorías evaluadas.(menos una)	2014-05-31	
Programa completo	El profesor presenta insuficiente preparación en cada una de las categorías evaluadas.(menos dos)	2014-05-31	
Programa completo	El profesor presenta insuficiente preparación en cada una de las categorías evaluadas.(menos dos y otras dos con regular)	2014-05-31	
Programa completo	"El profesor presenta insuficiente preparación en cada una de las categorías evaluadas.(menos cuatro)"	2014-05-31	
Programa promedio	El profesor presenta insuficiente preparación en la mayoría de las categorías evaluadas.	2014-05-31	
Programa promedio	El profesor presenta insuficiente preparación en la mayoría de las categorías evaluadas.	2014-05-31	
Programa promedio	El profesor presenta insuficiente preparación en la mayoría de las categorías evaluadas.	2014-05-31	
Programa especial	El profesor presenta buena preparación en gran parte de las categorías evaluadas.	2014-05-31	
Programa especial	El profesor presenta buena preparación en gran parte de las categorías evaluadas.	2014-05-31	
Programa especial	El profesor presenta buena preparación en gran parte de las categorías evaluadas.	2014-05-31	

1 - 10/17

Cancelar

FLUJOS ALTERNOS

Alternativa 1: “No se encontró información que cumpla con los criterios de búsqueda.”

Acción del Actor	Respuesta del Sistema
	1. Muestra el mensaje de información “No se encontró información que cumpla con los criterios de búsqueda.”
	2. Regresa al paso 2 del Flujo Normal de Eventos .
Alternativa 2: “Realizar una búsqueda simple.”	
Acción del Actor	Respuesta del Sistema
	1. Brinda la posibilidad de introducir los criterios de búsqueda simple: <ul style="list-style-type: none"> • Identificador • Programa • Explicación
2. Introduce los datos que considera como criterios para realizar una búsqueda simple y selecciona la opción de Buscar caso.	
	3. Busca los datos de los Casos que cumplen con los criterios de búsqueda.
	4. Regresa al paso 2 del Flujo Normal de Eventos .
Alternativa 3: “Cancelar operación.”	
Acción del Actor	Respuesta del Sistema
1. Selecciona la opción de Cancelar operación.	
	2. Regresa a la vista anterior.
	3. El caso de uso termina.
Alternativa 4: “Adicionar caso.”	
Acción del Actor	Respuesta del Sistema
1. Selecciona la opción Adicionar caso.	2. Muestra los datos para adicionar el caso. Ver caso de uso: Adicionar caso .
	3. Regresa al paso 2 del Flujo Normal de Eventos .
Alternativa 5: “Ver detalles del caso.”	
Acción del Actor	Respuesta del Sistema
1. Selecciona la opción Ver detalles del caso.	2. Muestra los datos del caso. Ver caso de uso: Ver detalles del caso .
	3. Regresa al paso 2 del Flujo Normal de Eventos .
Alternativa 6: “Modificar caso.”	
Acción del Actor	Respuesta del Sistema
1. Selecciona la opción Modificar caso.	2. Muestra los datos para modificar el caso. Ver caso de uso: Modificar caso .
	3. Regresa al paso 2 del Flujo Normal de

Eventos.	
Alternativa 7: "Eliminar caso."	
Acción del Actor	Respuesta del Sistema
1. Selecciona la opción Eliminar caso.	2. Muestra un mensaje para eliminar el caso. Ver caso de uso: Adicionar caso.
	3. Regresa al paso 2 del Flujo Normal de Eventos.
Poscondiciones	Se gestionaron los datos del caso.

Tabla 23: CU: Atender peticiones

CASO DE USO:	Atender peticiones
Resumen:	El caso de uso inicia cuando el actor accede a la opción Atender peticiones, el sistema muestra un listado de Peticiones, el sistema carga en la vista anterior la Petición seleccionada y permite cambiar el estado de la petición, el caso de uso termina.
Complejidad:	Media
Prioridad:	Secundario
Precondiciones:	Deben estar creadas las Peticiones en el sistema.
REFERENCIAS	
Actores:	Experto
Requisitos:	RF6, RF7
FLUJO NORMAL DE EVENTOS	
Acción del Actor	Respuesta del Sistema
1. El caso de uso inicia cuando el actor accede a la opción Atender peticiones.	
	2. <i>Muestra un listado de Peticiones con los siguientes atributos:</i> <ul style="list-style-type: none"> • Identificador • Programa • Explicación • Fecha y permite: <ul style="list-style-type: none"> • Cancelar operación. Ver Alternativa 1: "Cancelar operación." • Seleccionar una Petición.
3. Selecciona la Petición deseada.	
	4. De cada la Petición seleccionada, carga los datos : <ul style="list-style-type: none"> • Tipo de programa • Características • Valor • Sugerencia de cursos • Explicación

	<ul style="list-style-type: none"> • Nota Brinda la posibilidad de seleccionar: • Estado de la petición. y permite: • <i>Aceptar cambiar el estado de la petición.</i> • Cancelar operación. Ver Alternativa 1: “Cancelar operación.”
	5. El caso de uso termina.

Tipo de programa	
Programa completo	
Socio-Afectiva	Pedagogía y Didáctica General
Conocimientos del Área	Controles a Clases
Evaluación Profesional	
Áreas evaluadas	
Comunicación con los estudiantes	4
Visita a las residencias	No
Participación en la ejecución del proyecto educativo	No
Apoyo incondicional a las misiones de la universidad	No
Sugerencia de cursos	
Curso de postgrado “El Trabajo Metodológico en la Educación Superior”	
Diplomado “Formación Pedagógica Superior”	
Diplomado “Formación Pedagógica Básica”	
Entrenamiento “Formación a través del vínculo universidad – industria”	
Explicación	
El profesor presenta insuficiente preparación en la mayoría de las categorías evaluadas.	
Nota	
Se acepta revisado	
Estado	
<input type="text"/>	
<input type="button" value="Aceptar"/> <input type="button" value="Cancelar"/>	

FLUJOS ALTERNOS

Alternativa 1: “Cancelar operación.”

Acción del Actor	Respuesta del Sistema
1. Selecciona la opción de Cancelar operación.	
	2. Regresa a la vista anterior.

	3. El caso de uso termina.
Poscondiciones	Se atendió una petición seleccionada.

Anexo 3. Descripción de las principales tablas del modelo de datos

Tabla 24: Descripción de la tabla tb_caso

Nombre:	tb_caso	
Descripción:	La tabla representa la unidad principal del caso. Donde se guarda el identificador del caso entre otros datos.	
Atributo	Tipo	Descripción
id_caso	integer	Identificador de la tabla
fecha	date	Este atributo almacena la fecha exacta en la que se creó el caso.
explicación	varchar	En este atributo se da la explicación de la solución del caso respecto a un problema determinado
estado	varchar	En este atributo se especifica el estado del caso. En caso de que el atributo este en Pendiente, significa que está a espera de revisión por el experto. Si se encuentra en estado de Revisado, significa que ha sido aprobado y está en espera de asignación de peso para ser usado. Los otros estados que existen son Aceptado y Rechazado, que son los que se usan o no (respectivamente) para resolver una situación.
tn_programa	integer	Este atributo es la llave foránea que absorbe de la tabla tn_programa. Representa el id del tipo de clasificación del caso.
Tablas relacionadas	Descripción de la relación	
tn_programa	Uno a muchos.	

Tabla 25: Descripción de la tabla tr_caso_caracteristica

Nombre:	tr_caso_caracteristica	
Descripción:	La tabla almacena el valor de cada característica asociada a cada caso.	
Atributo	Tipo	Descripción
id_caso_caracteristica	integer	Identificador de la tabla
valor	integer	Este atributo almacena el valor que toman las características de un caso. Los valores que pueden tomar se encuentran en el intervalo de 2 a 5.
tb_caso	integer	Este atributo es la llave foránea que absorbe de la tabla tb_caso. Relaciona el id del caso con las características o rasgos.

tn_caracteristica	integer	Este atributo es la llave foránea que absorbe de la tabla tb_caracteristica. Representa el id de la características asociadas al caso.
Tablas relacionadas	Descripción de la relación	
tn_caracteristica	Uno a muchos.	

Tabla 26: Descripción de la tabla tr_caso_curso

Nombre:	tr_caso_curso	
Descripción:	La tabla almacena los cursos que componen un caso.	
Atributo	Tipo	Descripción
id_caso_curso	integer	Identificador de la tabla
tb_caso	integer	Este atributo es la llave foránea que absorbe de la tabla tb_caso. Relaciona el id del caso con los cursos que debe pasar.
tn_curso	integer	Este atributo es la llave foránea que absorbe de la tabla tb_curso. Representa el id del curso asociado al caso.
Tablas relacionadas	Descripción de la relación	
tn_curso	Uno a muchos.	
tb_caso	Uno a muchos.	