UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 2

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

Título: Propuesta de motor de inferencia de una base de casos para la estimación de la duración de un proyecto de desarrollo de software.

Autora: Karina Reyna Pupo

Tutora: MsC. Yadira Ruíz Constanten

Co-tutor: MsC. Darian Horacio Grass Boada

La Habana, junio del 2012.

Año 54 de la Revolución.

… las cosas simples deben ser simples, las cosas complejas deben ser posibles...

Alan Kay

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas y a la Facultad 2 para que hagan el uso que estimen pertinente con este trabajo.

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

_____

Karina Reyna Pupo

Autor

_____ _____

MsC. Yadira Ruíz Constanten **MsC. Darian H. Grass Boada**

Tutora **Co-tutor**

Dedico esta este trabajo a mi familia, que le sirva de orgullo de la misma manera que cada uno de sus triunfos me sirvió de ejemplo y guía.

A mi hermanita Elizabeth sobre todo, por ser la más pequeña y a la que más camino la falta por recorrer. Ojalá y vea en mí un ejemplo, así como lo fue para mí, nuestra hermana mayor Aniuska.

A la Revolución Cubana por darme la oportunidad de hacer realidad el sueño de convertirme en una profesional. A la Universidad de las Ciencias Informáticas por formarme como ingeniera.

Agradezco infinitamente a mi madre, por ser la guía y luchar con tantas fuerzas por esta familia tan linda de la que formo parte, donde hay tanto amor y con la que soy tan feliz, y todo gracias a ella, que desde que desde que decidió formar su propio jardín, no ha hecho otra cosa más que luchar por la felicidad y el bienestar de sus flores. Es muy fácil hacer buenas cosas teniendo tanto amor. Para ella todo mi orgullo, mi amor y respeto. Qué buena fue la vida conmigo al darme el regalo de tener una madre como tú.

A papuso por ser mi ejemplo y mi lucero, por servirme de inspiración y porque querer todos los días del mundo saber como sabe él. No hay un día que no me levante que no me compare con él y sienta lo mucho que me falta para alcanzarlo, pero lucho y seguiré luchando.

A mi hermana Aniuska le agradezco mucho lo que soy hoy, la veo como el ejemplo de la mujer fuerte, bella, inteligente, que logra todo lo que se propone, por eso me esfuerzo en ser como ella. Y a ella tengo que agradecerle la dicha de tener a mi sobrinita Ana Karla, que me inunda con amor y cariño, a ella la doy las gracias por existir.

A mi hermanita por ser tan buena, tan cariñosa, tan preocupada y tan atenta conmigo, a ella le doy las gracias por darme tanto cariño y hacerme que me supere para que vea en mí un ejemplo.

A mi abuela y a mi tío, que aunque estén lejos de nosotros, son muy importantes para mí, y sé que siempre rezan por mí, para que me vaya bien, esté saludable y sea muy feliz.

A mi novio Pedro tengo mucho que agradecerle, por su amor, su comprensión, su apoyo y por hacerme la persona más querida de la tierra. Hoy gran parte de mi felicidad, se la debo a él. Al igual que a su familia por acogerme con tanto cariño.

A mi cuñado Robert también le doy las gracias porque desde que forma parte de nuestra familia siempre ha sido muy atento y preocupado.

A mi tutora Yadira por ser tan magnífica, tan paciente y tan preocupada, por guiarme para lograr hacer este trabajo con calidad. A ella le doy las gracias y me considero privilegiada de haber tenido el placer de contar con su ayuda.

A mi tutor Darian por sus buenas ideas y consejos, fueron muy útiles y aprendí mucho.

Un agradecimiento especial para el profe Dasiel por su apoyo incondicional, sin su ayuda no creo que lo hubiera logrado.

A mis amistades de la Universidad que se mantuvieron junto a mí estos 5 años, compartiendo los buenos y los malos momentos: a Yuliani, Yamila, Yolanda, Ángel, Miguel Ángel, Tony, Alejandro y el Chino, a ellos mil gracias. Al resto de mis buenas amistades, les agradezco también el haberme dado la oportunidad de formar parte de sus vidas.

A todos mis profesores que me enseñaron, me educaron y creyeron en mí. A ellos les agradezco el poder haberme convertido hoy en una profesional.

Resumen

El trabajo en proyectos es un esfuerzo llevado a cabo por un grupo de personas regidas por una idea común, para crear un producto o servicio único. Surge con el objetivo de satisfacer una necesidad y desarrollar una capacidad, resolviendo un problema dentro de un contexto específico, labor que variará en dependencia de las circunstancias y situaciones en cada organización, razón por la que cada proyecto, dependerá de una gestión apropiada, adaptada a sus características, para alcanzar sus objetivos.

El uso de herramientas inteligentes que permitan aprovechar el conocimiento acumulado por especialistas tras su experiencia en proyectos, puede ser utilizado para obtener mejores resultados en proyectos futuros, contribuyendo a la toma de decisiones en los mismos.

Partiendo del conocimiento almacenado en una base de casos se precisa de un mecanismo que permita interpretar los datos que en ella existen. El diseño de un motor de inferencia para esta base de casos ayudará a predecir, a partir de datos de proyectos con estimaciones de tiempo correctas y validados por expertos, los resultados de estimación de un nuevo proyecto, partiendo de casos similares conocidos.

La descripción de este nuevo problema será la entrada para empezar a inferir sobre la base de casos, utilizando para ello métodos inductivos de recuperación que obtengan los casos más similares, luego serán evaluados en una función de semejanza de donde se obtendrá el grado de similitud. El caso escogido para la adaptación que dará la solución al problema planteado será el de mayor utilidad esperada.

Introducción

Desde el surgimiento del hombre, hace miles de años, este ha tenido la capacidad de evolucionar sus métodos de trabajo acorde a sus necesidades. Al inicio el trabajo se realizaba individualmente pero con el transcurso del tiempo se fueron juntando y trabajando en grupos, descubriendo así, las facilidades que tiene el trabajo en equipo. Pero no fue hasta la segunda mitad del siglo XX donde las situaciones existentes requerían de la coordinación del trabajo de equipos en diferentes disciplinas para lograr la construcción de un sistema único, materializándose de esta manera el trabajo en proyectos.

La acumulación de experiencias en el trabajo en proyectos y el surgimiento de organizaciones y estándares dedicados a la planificación de los mismos, han permitido que su desarrollo se haya convertido en una actividad planificada y controlada (1).

El PMI[1] es una organización internacional sin fines de lucro que asocia a profesionales para la gestión de proyectos (2). Actualmente, es la más grande del mundo en su rubro; dado que se encuentra integrada por más de 260.000 miembros alrededor de 171 países. El mismo, define que: un proyecto es un esfuerzo temporal, único y progresivo, emprendido para crear un producto o un servicio también único (3).

El proceso de gestión de proyectos de desarrollo de software comienza con un conjunto de actividades que, globalmente se denominan planificación del proyecto. Esta es fundamental en desarrollo del software ya que define el qué, el cuándo y los recursos de las tareas que se van a realizar (4). Tiene como finalidad la planificación, el control y el seguimiento de recursos humanos, tiempo y esfuerzo que se necesitan antes y durante el desarrollo del software. Los proyectos de software deben iniciarse con un plan definido, al igual que el resto, pero en esta esfera, la planificación es un poco más compleja.

Los proyectos bien ejecutados pasan por tres etapas básicas para crear una planificación de software. Primero, se estima el tamaño del producto, luego el esfuerzo necesario para construir un producto con este tamaño y por último la duración cronológica del proyecto (5).

[1]Instituto de Gestión de Proyectos por sus siglas en inglés Project Management Institute PMI®

El desarrollo de software requiere de la estimación como una herramienta para controlar y administrar los recursos que se necesitan y que se utilizan durante la concepción y desarrollo del proyecto. La estimación no puede ser considerada como una ciencia exacta ya que existen numerosas variables humanas, técnicas, del entorno y políticas, que intervienen en su proceso y que pueden afectar los resultados finales (6). Sin embargo, cuando es llevada a cabo en forma sistemática, se pueden lograr resultados con un grado aceptable de riesgo y convertirla en un instrumento útil para la toma de decisiones. De manera general se puede decir que la estimación brinda la medición de un determinado tipo de software.

Con el desarrollo de las ciencias y las tecnologías, han surgido disciplinas capaces de predecir e identificar un resultado determinado o un evento que puede ocurrir simulando el razonamiento humano. A los sistemas que son capaces de tener este tipo de comportamientos se les denomina de manera general sistemas inteligentes, ya que son capaces de deducir soluciones a partir de un conocimiento previamente almacenado.

Es posible y resultaría ventajoso, utilizar las facilidades que brindan estos sistemas inteligentes para optimizar los resultados de la estimación de la duración de proyectos de desarrollo de software.

Resulta importante para las empresas de desarrollo de software contar con modelos y métodos de estimación capaces de asegurar el prestigio de la institución. Para ello, es vital, poder entregar al cliente, el producto terminado con la calidad requerida en el tiempo acordado. Pero esto, es algo que no sucede muy a menudo, se pueden dar varias situaciones que impidan satisfacer el cliente: la entrega del producto en tiempo, pero no con todas las funcionalidades acordadas; la entrega fuera de tiempo del producto terminado; la entrega fuera de tiempo del producto sin terminar. Entre los factores que más inciden en la ocurrencia de este fenómeno, se encuentra el problema que presenta el equipo de trabajo para realizar una estimación lo más real posible del software.

La UCI [2] es una institución dónde se realizan proyectos de software para su comercialización dentro y fuera del país. Esta, se ha visto afectada con la entrega de productos en tiempo y con las funcionalidades requeridas. Varios son los elementos que

[2] Universidad de las Ciencias Informáticas

inciden, muchos de los cuales se derivan de problemas en la planificación, dados por el proceso de estimación que realizan. En la mayoría de los proyectos, durante el desarrollo de software, la estimación se lleva a cabo solo una vez y utilizando un único método de estimación, lo cual imposibilita realizar comparaciones que aseguren que las estimaciones realizadas sean reales y confiables.

Como elemento distintivo se debe señalar que a pesar de los problemas que se dan en la estimación en los proyectos, no se recogen los detalles de cada una de estas estimaciones realizadas, en aras de tenerlas como base para futuras valoraciones, aprovechando la experiencia adquirida por muchos de los especialistas para estimar la duración de los mismos.

Entre los métodos que se utilizan para realizar la estimación se encuentra el método Puntos por Casos de Uso, que se basa en el cálculo a partir de la clasificación de las transacciones obtenidas tras la descripción detallada de los Casos de Uso (CU) que han sido considerados como críticos. Esto da paso a la aparición de otro problema: ya le estimación no solo dependerá de la persona encargada de estimar, sino también de la quien redacta las descripciones detallada de los CU, determinando la cantidad de transacciones existentes. Teniendo en cuenta además el hecho de que se deba incluir o no algún CU que en el primer instante no se consideraba como crítico y después se decide modificar su clasificación.

Es importante señalar también que el proceso de estimación muchas veces no es realizado por una persona especializada en el tema, por lo que los resultados no son lo más fieles posibles.

En la UCI se está desarrollando una aplicación que permitirá almacenar los indicadores necesarios de un proyecto para predecir el tiempo que demoraría la construcción del producto. La manera de representar este conocimiento previo que se obtiene de un experto, es mediante casos, los cuales están compuestos por un conjunto de rasgos. Cada uno de ellos define un indicador en específico necesario para estimar el producto de software. Al conjunto de casos que representan ejemplos de proyectos se le denomina Base de Casos.

Este sistema no incluye el análisis de los datos almacenados, por lo que no se provee de un mecanismo para interpretar los resultados y llegar a nuevas soluciones de proyectos que se deseen estimar, utilizando como base los ejemplos anteriores recolectados.

Tras la situación problemática expuesta surge como **problema a resolver**: ¿Cómo inferir la estimación de la duración de proyectos de desarrollo de software a partir de los datos almacenados en una base de casos con indicadores de tamaño y esfuerzo?

Definiéndose en la investigación como ***objeto de estudio*** los métodos de obtención de información a partir de una base de casos y el ***campo de acción*** se enmarcará en los fundamentos matemáticos del motor de inferencia de un sistema basado en casos para la estimación de la duración de proyectos de desarrollo de software.

En aras de solucionar el problema planteado se tiene como ***objetivo general*** definir el motor de inferencia de una base de casos con indicadores de tamaño y esfuerzo, para la estimación de la duración de proyectos de desarrollo de software en la Universidad de las Ciencias Informáticas.

Teniendo en cuenta lo anterior se tiene como ***idea a defender*** que la definición del motor de inferencia permite la obtención de la duración de proyectos de desarrollo de software a partir de una base de casos con indicadores de tamaño y esfuerzo.

Del mismo se derivan los siguientes ***objetivos específicos***:

1. Analizar los sistemas basados en el conocimiento como técnica de inteligencia artificial en la solución de problemas relacionados con la estimación de proyectos de desarrollo de software.
2. Analizar los fundamentos matemáticos necesarios para formalizar la propuesta del motor de inferencia.
3. Validar la solución propuesta.

Para lograr los objetivos trazados, se definieron las siguientes ***tareas de investigación***:

1. Elaboración del marco teórico de la Investigación.
2. Revisión de los elementos generales de estimación de la duración de proyectos de desarrollo de software al ser el dominio de la base de casos.
3. Caracterización de los sistemas basados en conocimiento para definir el uso de la técnica de razonamiento basado en casos.

4. Análisis de las características y particularidades de los motores de inferencia o mecanismos de control.

En el presente trabajo se adoptó como estrategia de investigación la descriptiva debido a que en la misma se pretende solucionar el problema en cuestión y realizar una descripción del fenómeno, reflejando lo más importante del mismo, teniendo en cuenta además las relaciones entre las partes involucradas y sus aspectos generales.

Para darle cumplimiento a la investigación a través de la mencionada estrategia se utilizaron los siguientes métodos de investigación:

Métodos Teóricos:

- Analítico–Sintético: Mientras que el análisis permite el estudio de cada uno de los factores que influyen en la realización de la estimación del proceso de desarrollo de software en su relativa independencia uno de otro, la síntesis permite descubrir las relaciones existentes entre dichos factores, así como la interacción dialéctica que se establece entre ellos y el condicionamiento mutuo que ejercen sobre la estimación del proceso de desarrollo de software**.**
- Inductivo-Deductivo: Se utiliza para el planteamiento del objetivo y la extracción de las ideas fundamentales para la elaboración y fundamentación del trabajo de diploma.
- Histórico-Lógico: El método histórico permitió estudiar la trayectoria real de la situación actual de la estimación del proceso de desarrollo de software, además de herramientas inteligentes que permitan su tratamiento y acontecimientos fundamentales en el de cursar de la historia de ambos. El método lógico permitió investigar las leyes generales del tema y las peculiaridades de los marcos estudiados. La utilización de este método posibilitó que el estudio no se limitara a una simple descripción de los hechos sino que facilitó el descubrimiento de la lógica objetiva del desarrollo histórico de la estimación del proceso de desarrollo de software y su integración con herramientas inteligentes para su tratamiento.

Métodos Empíricos:

- Entrevista: Se hizo uso de la entrevista para la obtención de datos lógicos para la BC usada como prueba así como para distinguir aspectos a incluir en el modelo para mejorar su aplicación y efectividad.

Capítulo 1: Estimación en Proyectos de Software

Introducción del Capítulo

En el presente capítulo se hace un análisis de algunos de los principales métodos de estimación de procesos de desarrollo de software y de las principales técnicas de los sistemas basados en conocimiento de la Inteligencia Artificial. Se incluyen también conceptos teóricos referentes a las dos disciplinas tratadas.

1.1 Gestión de Proyectos

Para entender la labor que realiza la gestión de proyectos se debe definir primeramente a qué se hace referencia con la palabra **gestión** (7):

Gestión: Acción y efecto de gestionar. ...

Gestionar: Hacer diligencias conducentes al logro de un negocio o de un deseo cualquiera.

La **gestión de proyectos**, por su parte, es la disciplina que se encarga de organizar y de administrar los recursos de manera tal que se pueda concretar todo el trabajo requerido por un proyecto dentro del tiempo y del presupuesto definido. Este aplica los conocimientos, las técnicas y herramientas en aras de satisfacer una necesidad y cumplir sus objetivos. La gestión de proyectos es una de las áreas clave dentro del proceso de desarrollo de software y su tiempo de vida se extiende a lo largo de todo el desarrollo.

El inicio de la gestión de proyectos en los años 60 estuvo dado por el auge de implementación de sistemas complejos que requerían el trabajo en conjunto y sincronizado de varias disciplinas, pues se necesitaba diseñar algún método de organización para lograr mejorar el trabajo y de esta forma evitar problemas que se repetían con gran frecuencia como lo era el desbordamiento de la agenda de los miembros, un aumento exhaustivo de los costos y un descenso de la calidad del resultado obtenido. Estos parámetros afectaban en gran medida el éxito del producto final, pues el tiempo, el costo y la calidad, son elementos deterministas de esta cualidad, los cuales se incluyen dentro de la razón de ser de la gestión de proyectos: crear un producto dentro del plazo establecido, con el presupuesto acordado y con las especificaciones pactadas con el cliente (1).

Para lograr que un proyecto de desarrollo de software se lleve a cabo con éxito es necesario pensar en todo: las etapas que ha de transitar, las tareas a realizar, los riesgos que se tendrán para poder evitarlos. Para ello se hace imprescindible contar con una buena planificación la cual se considera la base de la gestión de software.

1.1.1 Planificación de Proyectos de Software

El proceso de gestión de proyectos de desarrollo de software comienza con un conjunto de actividades que globalmente se denominan: planificación del proyecto. Esta es fundamental en el desarrollo del software ya que define, entre otras cosas, el qué, el cuándo y los recursos de las tareas que se van a realizar. El objetivo específico de la planificación del proyecto de software es proporcionar un marco de trabajo que permita al gestor hacer estimaciones razonables de recursos y costos, además de definir una planificación temporal. Estas estimaciones se hacen dentro de un marco de tiempo limitado: al comienzo de un proyecto de software, y deberían actualizarse regularmente a medida que progresa.

Para lograr una buena planificación del proyecto se debe (8):

1. Conceptualizar el producto y estimar el tamaño.
2. Realizar un estudio de factibilidad (evaluar viabilidad).
3. Estimar los recursos.

1.1.2 Estimación de software

Estimar consiste en determinar el valor de una variable desconocida a partir de otras conocidas, o de una pequeña cantidad de valores conocidos de esa misma variable. Es importante pensar en una predicción como en un rango más que como en un simple número. Una estimación o predicción no es un objetivo, sino una valoración probabilística. El valor que se obtiene de una estimación es el centro del rango (5).

La estimación en los proyectos de software tiene dificultades particulares si se compara con la predicción en otras industrias, donde es habitual producir el mismo tipo de producto una y otra vez, con los mismos métodos. Esta centra su atención en reducir los costos e incrementar los niveles de servicio y de calidad. Midiendo determinados aspectos del

proceso de desarrollo de software se puede tener una visión amplia de lo que sucederá durante su desarrollo.

Cuando se habla de estimación de un proyecto se puede hacer referencia a tres grandes áreas: estimación de tiempo, de recursos y de costo, que aunque se trate de verlas a cada una de ellas de manera individual, la relación que existe es muy estrecha.

Se han desarrollado varias técnicas de estimación para el desarrollo de software, y aunque cada una tiene sus puntos fuertes y sus puntos débiles, todas tienen en común los siguientes atributos (9):

1. Se ha de establecer de antemano el ámbito del proyecto.
2. Como bases para la realización de estimaciones se usan métricas del software de proyectos pasados.
3. El proyecto se desglosa en partes más pequeñas que se estiman individualmente.

El proceso para crear una planificación de desarrollo exacta consta de tres pasos (5):

- Estimar el tamaño del producto (generalmente a través del número de líneas de código o puntos de función).
- Estimar el esfuerzo (personas-mes).
- Estimar la duración (meses).

1.2 Inteligencia Artificial

A la rama de las Ciencias de la Computación dedicada al desarrollo o uso de los ordenadores, con los que se intenta reproducir los procesos de la inteligencia humana se le denomina IA. Su objetivo principal es lograr un sistema totalmente autónomo (10). Los sistemas de IA incluyen a programas capaces de hallar soluciones en dominios con altos niveles de complejidad, aprender con la experiencia, comprender un lenguaje natural y en general, comportarse de alguna forma que sea considerada inteligente.

Según la bibliografía consultada, la IA se divide en dos escuelas de pensamiento (11):

La inteligencia artificial convencional: Se conoce también como IA simbólico-deductiva. Está basada en el análisis formal y estadístico del comportamiento humano ante diferentes problemas. Entre los diferentes tipos de sistemas que se incluyen dentro de esta escuela de pensamiento se pueden mencionar:

- Razonamiento basado en casos: Ayuda a tomar decisiones mientras se resuelven ciertos problemas además de ser muy importantes requieren de un buen funcionamiento.
- Sistemas expertos: Infieren una solución a través del conocimiento previo del contexto en que se aplica y se ocupa de ciertas reglas o relaciones.
- Redes bayesianas: Propone soluciones mediante inferencia probabilística.
- Inteligencia artificial basada en comportamientos: que tienen autonomía y pueden auto-regularse y controlarse para mejorar.
- Gestión de procesos inteligentes: facilita la toma de decisiones complejas, proponiendo una solución a un determinado problema al igual que lo haría un especialista en la actividad.

La inteligencia computacional: También conocida como IA simbólica-inductiva, implica desarrollo o aprendizaje interactivo (modificaciones interactivas de los parámetros en sistemas conexionistas). El aprendizaje se realiza basándose en datos empíricos.

En esta investigación, se hace referencia solamente a la inteligencia artificial convencional, ya que la solución se centra en un sistema basado en conocimiento, específicamente un sistema basado en casos. Esta selección está determinada por varios factores. El primero de ellos está dado porque se modela de la manera más sencilla posible el conocimiento, se pueden devolver determinadas respuestas y conocer los detalles del por qué del resultado y por último, simula de manera similar el comportamiento del razonamiento humano, comparando los nuevos casos, conocimiento y experiencia acumulada.

1.2.1 Sistemas Basados en el Conocimiento (SBC)

En la década del 70 se reconoció que los métodos de solución de problemas generales eran insuficientes para resolver los problemas orientados a aplicaciones. Se determinó que era necesario tener un conocimiento específico sobre el problema, limitado a los dominios de aplicación de interés, en lugar de tener un conocimiento general aplicable a muchos dominios. Este reconocimiento condujo al desarrollo de Sistemas Basados en el Conocimiento (SBC) (10).

Los SBC, típicos del campo de la IA, no son más que programas para computadoras que simulan las cadenas de razonamiento que realiza un experto para resolver un problema de su dominio. Para conseguirlo, se dota al sistema de un conjunto de principios o reglas que infieren nuevas evidencias a partir de la información previamente conocida. En términos generales, un SBC puede ser definido como un sistema que utiliza el conocimiento acumulado sobre un dominio en específico para obtener resultados y llegar a conclusiones en problemas del mismo dominio.

En (11) se encuentran reflejadas una serie de ventajas y desventajas que presentan los SBC. En los siguientes subepígrafes se pretende detallar cómo se ven reflejadas sus características y de qué manera se resolverán las desventajas más significativas que poseen.

1.2.1.1 Ventajas de los SBC

- **Mayor disponibilidad, accesibilidad y permanencia:** La experiencia se encuentra disponible todo el tiempo al almacenarla en un mismo sistema. Esto permite poder consultarla en cualquier momento. Ya no sería necesario esperar por la disposición de un experto para poder realizar la estimación de un proyecto ni sería imprescindible su conocimiento, pues toda la experiencia se encuentra almacenada en una base de conocimientos.
- **Experiencia múltiple:** Al contar con un sistema que almacene conocimiento, se pueden hacer converger el criterio de varios especialistas para atacar a un mismo problema de estimación. El nivel de experiencia combinada de muchos sistemas expertos puede mejorar la calidad de las respuestas.
- **Respuestas no subjetivas:** El sistema basado en conocimientos ofrece respuestas sólidas y completas que no se ven afectada por la apreciación personal que pudiera representar una mala conclusión, como indicadores que se obviaran o el asumir valores irreales, elementos que pudieran desviar el resultado final de una estimación.
- **Respuesta rápida:** Algunas situaciones de emergencia pueden exigir respuestas más rápidas que las de un humano, y la estimación que realizan los expertos muchas veces es un proceso que demora. La creación de un sistema inteligente para estimar software es capaz de devolver respuestas rápidas y fundamentadas.
- **Tutoría inteligente:** asegura que se den respuestas consistentes y que el conocimiento crezca en experiencia con el transcurso del tiempo, mejorando la calidad

de sus respuestas sin necesitar inferir un nuevo conocimiento por parte de los expertos. Para ello, los primeros proyectos almacenados en el sistema, son validados y aprobados por expertos, pues no se puede empezar a derivar conclusiones de estimación sombre datos que no sean sólidos.

- **Preservación de la experticidad:** los SBC constituyen una memoria asociativa y poseen la capacidad para adquirir nuevo conocimiento y perfeccionar el que poseen. Lo que posibilita el poder realizar estimaciones más precisas a medida que mejore el entrenamiento del sistema inteligente.

1.2.1.2 Desventajas de los SBC

Los SBC también presentan una serie de desventajas:

- La primera está dada por la adquisición del conocimiento y cómo representarlo de forma abstracta efectiva. Esto se debe a que no existe un mecanismo definido que explique de qué manera es más conveniente representar los datos que luego serán inferidos. Pero como esta propuesta parte de la definición de la estructura de los datos que serán almacenados en una BC, esta no es una desventaja que afecte el objetivo que se persigue con este trabajo, ya que fue solucionada con la creación del sistema que almacena los indicadores de estimación de un proyecto en forma caso, que luego son acumulados en una BC.
- La generación de inferencias o cómo hacer uso de esas estructuras abstractas para generar información útil para el contexto de un caso específico, es también una desventaja que presenta los SBC, ya que lo más difícil de definir es de qué manera se realizará la inferencia, cuyo problema resuelve la realización de este trabajo.

1.2.1.3 Arquitectura de los SBC

Los SBC pueden ser diferenciados del resto de las técnicas de la IA ya que: en primer lugar, ejecutan tareas que de alguna forma resultan complicadas o simplemente difíciles para cualquier ser humano y que están reservadas solamente a unos pocos en un dominio en particular. Para ejecutar estas tareas o resolver los problemas que solucionan los expertos, los SBC utilizan estrategias de búsqueda que comúnmente se denominan heurísticas. Además, emplean conocimientos para razonar acerca de sus propios métodos de inferencia y por último proporcionan explicaciones o justificación de la

solución obtenida. Los SBC pueden representarse de manera general de la siguiente manera:

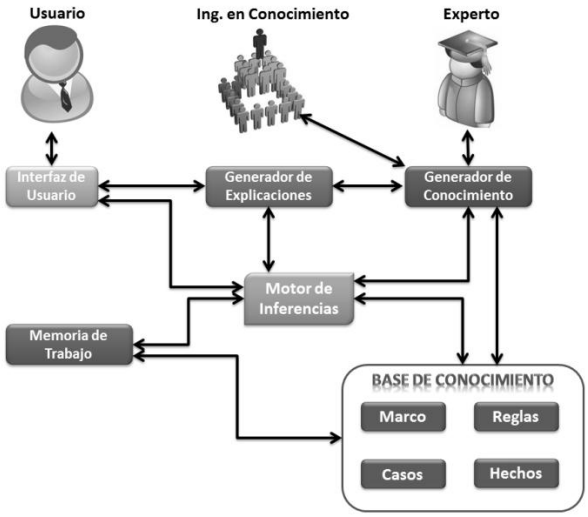

Figura 1.1: Representación general de los SBC.

Como se ilustra en la figura 1.1, en la base de conocimientos se albergan los conocimientos relativos a la tarea que resuelve el experto. El motor de inferencia es el encargado de aplicar esos conocimientos. La memoria de trabajo contiene la información de datos de entrada y conclusiones intermedias que se generan en el proceso de razonamiento. El generador de explicaciones es el módulo que proporciona una explicación coherente con la solución encontrada. El subsistema de adquisición de conocimientos es el medio que facilita tanto la inclusión como la actualización del conocimiento. Por último la interfaz del usuario permite establecer una comunicación entre el SBC y el usuario.

De manera general, los elementos fundamentales que componen la arquitectura de los SBC son:

- Interfaz
- Máquina o Motor de Inferencia (MI)
- Base de Conocimientos

La forma en que se relacionan estos elementos se muestra en la figura 1.2.

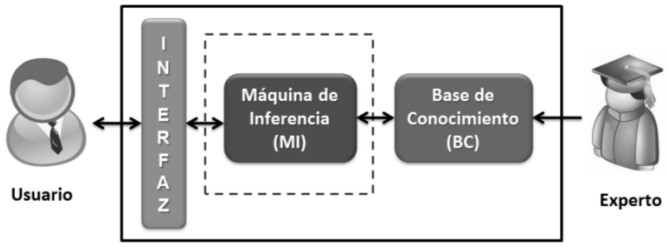

Figura 1.2: Arquitectura de los SBC

La interfaz es el componente que facilita la interacción con el sistema. La máquina de inferencia es quién realiza la obtención y transformación del conocimiento previo almacenado en la base de conocimientos por un experto, para dar solución a un nuevo problema planteado.

1.2.1.4 Tipos de SBC

Diferentes tipos de conocimiento dan lugar a diferentes SBC, entre ellos se encuentran los sistemas basados en reglas, los sistemas basados en probabilidades, y los sistemas basados en casos:

- **Sistemas de Razonamiento Basados en Reglas (SRBR):** Son sistemas que utilizan para el proceso de inferencia un conjunto de reglas que constituyen la base de conocimiento del experto.

  Los SBR presentan la facilidad de ser un formato muy fácil para expresar el conocimiento. Son altamente modulares pues cada regla es una unidad de conocimiento y estas pueden ser añadidas, modificadas y removidas independientemente de las otras reglas existentes. También poseen una uniformidad pues todo el conocimiento del sistema se expresa en el mismo formato.

  Sin embargo sería muy poco probable lograr estructurar el conocimiento referente a los datos de un proyecto con muchos indicadores que después no se dificulte en gran medida la recuperación de un nuevo proyecto similar, esto se debe la gran cantidad de reglas que se necesitarían para poder lograr almacenar sus datos, el tamaño de la BC crecería notablemente con cada proyecto nuevo que se pudiera almacenar.

  Aplicar este tipo de sistema de conocimiento produce un encadenamiento infinito debido a que la mayoría realizan una búsqueda primero en profundidad y este

método produce ese inconveniente. El almacenamiento de tantas reglas trae consigo incoherencia entre ellas, y que la adición de una nueva, pueda entrar en contradicción con alguna que ya esté registrada, pues es muy engorroso verificar esto. También hay que señalar que para crear un SRBR indispensable la presencia de un experto que generen estas instrucciones de alto nivel.

- **Sistemas de Razonamiento Basados en Frames (SRBF):** Un *frame*[3] es una estructura de datos compleja que contiene un agregado de información acerca de un objeto, ofreciendo una representación estructurada de este objeto. Este SBC aunque representa una mejora con respecto a SRBR, ya que el conocimiento está significativamente bien estructurado y organizado, permitiendo al sistema rápidamente dirigir la atención a los aspectos apropiados, cuando aparece nuevos objetos no tiene prototipos nuevos para guiar la representación de estas instancias, inconveniente que imposibilita almacenar nuevos indicadores de proyectos en caso de requerirse para actualizar el sistema y este no llegue a ser obsoleto en algún momento. En los SRBF no se aplican mecanismos de aprendizaje automático razón por la no aumentan su conocimiento.
- **Sistemas de Razonamiento Basados en Probabilidades (SRBP):** Los sistemas probabilísticos se consideran otro de los tipos más importantes de sistemas basados en el conocimiento. Este método es utilizado mayormente por sistemas donde la naturaleza de las inferencias sean probabilísticas, como lo son: inferencias por diagnóstico, inferencias causales, inferencias intercausales e inferencias mixtas. También se utiliza en sistemas donde existe una falta de formalismo para representar cierta clase de conocimiento, o donde la información provenga de múltiples fuentes, algunas fiables y otras no tanto, o conjuntos cuyas cotas no estén definidas rigurosamente.

  Los indicadores de estimación de software, que son los atributos mediante los cuales se representan los datos que se necesita de un proyecto para estimar el tiempo que necesitará para su confección, se obtuvieron de una misma fuente de información, que no es más que las metodologías existentes que se especializan en estimar los procesos de desarrollo de software, por lo que todos tienen el mismo

[3] marco

nivel de análisis. Es por esta razón que se aparta los SRBP como posible solución para construir el sistema inteligente que permitirá estimar la duración de proyectos de desarrollo de software.

- **Sistemas de Razonamiento Basados en Casos (SRBC):** Representa un nuevo método para resolver problemas no estructurados, en el cual el razonamiento se realiza a partir de una memoria asociativa que usa un algoritmo para determinar una medida de semejanza entre dos objetos. En este paradigma la base del comportamiento inteligente de un sistema radica en recordar situaciones similares existentes en el pasado.

  Los SRBC presentan una serie de ventajas que al explotarlas darán una buena visión para mejor soluciones alcanzadas, ya que este utiliza las experiencias previas que almacena en forma de ejemplos concretos que hayan sido exitosas, para llegar a nuevas soluciones o anticipar problemas. Su aprendizaje es incremental pues este asimila nuevos ejemplos a medida que vaya entrenando su conocimiento sin la necesidad de un experto y también ayuda a resolver problemas de forma más rápida que analizando el problema de forma trivial por su capacidad en derivar sus respuestas a partir de otras.

  Todas estas ventajas y teniendo en cuenta la manera en que se modeló el conocimiento en la base de conocimientos, se decidió usar u SRBC a pesar de las desventajas que puede presentar la dificultad de encontrar una función de semejanza que se adecúe con el dominio del problema y que sea capaz de encontrar la similitud entre los casos a través de sus rasgos.

1.3 Sistema de Razonamiento Basado en Casos

Dentro de las técnicas de IA, el Razonamiento Basado en Casos (RBC) es un paradigma de solución de problemas que difiere de otros enfoques y técnicas por su capacidad de utilizar el conocimiento específico adquirido en situaciones previas y utilizarlo en la situación presente. El problema nuevo se resuelve buscando en su memoria, un caso similar resuelto en el pasado. Además incrementa su conocimiento, almacenando el nuevo caso para ser usado en situaciones futuras (12).

El proceso que realiza el RBC es análogo al razonamiento humano. Cuando una persona se enfrenta a un problema por simple que sea, empieza por buscar en su memoria experiencias similares a esta, combinando una serie de semejanzas y diferencias que

ayude a obtener la nueva solución. Este es un proceso intuitivo que las personas lo realizan prácticamente sin darse cuenta. En este principio se basa el RBC.

La elaboración de un sistema que emplea el RBC presenta dos problemas principales: el primero, saber cómo almacenar la experiencia de tal forma que esta pueda ser recuperada en forma adecuada y el segundo, conseguir utilizar la experiencia previa en un problema actual (13).

La forma de representar y almacenar los indicadores de los proyectos para estimar los procesos de desarrollo de software se realiza a través de casos. Un caso es constituido por dos elementos generales: los rasgos predictores y los rasgos objetivos (figura 1.3). La especificación de los primeros dará lugar a la salida o solución del problema, representado por el segundo. Cuando se encuentra un caso declarado en la BC se da por hecho que este poseerá un cierto grado de riqueza en la información que contiene, es decir, la experiencia está perfectamente descrita y se procura que no falte información en su descripción, además es necesario que toda esta información que contiene posea un cierto grado de organización que permita su rápido entendimiento y sobre todo llegar a la información necesaria rápidamente, minimizando acceder a información innecesaria.

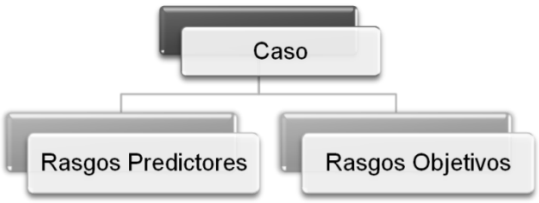

Figura 1.3: Elementos generales de un caso

La calidad de un RBC está dada por (13):

1. La experiencia que tiene.
2. La habilidad para entender situaciones nuevas en términos de experiencias pasadas.
3. Su capacidad de adaptación.
4. Su habilidad para integrar nuevas experiencias en su memoria adecuadamente.

1.3.1 Arquitectura y Componentes

Los SRBC contienen dos componentes fundamentales para su funcionamiento: una Base de Casos (BC) y un componente encargado de solucionar los problemas. La BC es quien

contiene todas las descripciones que han sido obtenidas previamente; cada caso representa los indicadores específicos de un proyecto. En el ciclo de solución del problema se recupera un caso semejante al nuevo y la solución del problema que fue recuperado es propuesta como solución potencial del nuevo problema, luego el nuevo caso obtenido podrá formar parte de la BC. Esto se deriva de un proceso de adaptación en el cual se indexa la vieja solución a la nueva situación.

El funcionamiento del RBC parte del principio de simular el razonamiento humano detallado anteriormente y para ello comprende cuatro actividades principales.

- **Recuperar los casos más parecidos:** esta recuperación tendrá lugar en la BC donde está almacenada la experiencia obtenida en forma de casos.
- **Reutilizar el o los casos para tratar de resolver el nuevo problema:** esta tarea está centrada en dos aspectos: las diferencias entre el caso pasado y el actual, y qué parte o partes del caso recuperado pueden ser transferidas al nuevo caso.
- **Revisar y adaptar la solución propuesta, en caso de ser necesario:** Una vez analizada la tarea de reutilización, la solución del nuevo problema tiene que ser probada; este proceso de prueba se hace durante la tarea de revisión.
- **Almacenar la nueva solución como parte de un nuevo caso:** El nuevo caso es almacenado con vista a un posible uso futuro. Durante este proceso de aprendizaje, el sistema tiene que seleccionar qué información del caso almacenar y cómo indexar el caso en la estructura de memoria para una posterior recuperación.

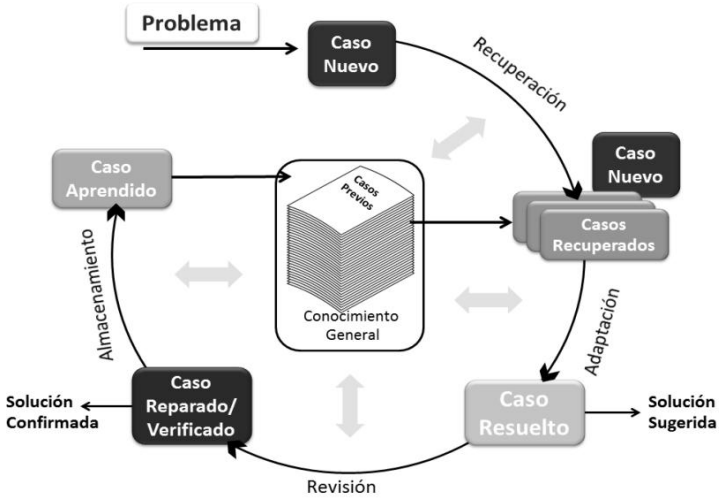

Figura 1.4: Ciclo básico de un sistema de RBC

En la figura 1.4 se puede observar el ciclo básico de un SRBC. La descripción de un nuevo problema da lugar a la aparición de un nuevo caso que se desea resolver. Este pasa a la fase de recuperación donde se obtienen los casos más similares a él, registrados en la BC. Luego en una fase de adaptación el nuevo caso es combinado con él/los casos recuperados de la BC para dar solución al nuevo problema planteado. En una fase de evaluación se verifica el éxito de solución, la mayoría de las veces esta etapa es llevada a cabo por un agente humano. Durante el almacenamiento, la experiencia útil se guarda para una futura reutilización y el caso base se actualiza como un nuevo caso aprendido, o se modifican algunos casos existentes, evitando así, tener datos redundantes en BC.

Conclusiones Parciales

En este capítulo se realizó un estudio del proceso de estimación de software y las técnicas de la IA, mostrando los elementos necesarios para demostrar que una posible mejora en el proceso de estimación de los proyectos de desarrollo de software en la UCI, puede ser el uso de un sistema inteligente que ayude a la toma de decisiones. Se realizó un estudio de algunos de los sistemas de IA, mostrando sus principales características y señalando las razones más significativas por las cuales se escogió los SRBC, ya que permite estructurar los datos de una manera lógica, haciendo más organizada la información que se necesita registrar para estimar los procesos de desarrollo de software, del cual parte la propuesta del mecanismo de inferencia que se obtendrá con este trabajo. También se describió la arquitectura de los SBC detallando su funcionamiento en todas las etapas por las que este puede incurrir. De igual manera se especificó el ciclo básico del RBC.

Capítulo 2: Análisis y definición del motor de inferencia

Introducción

En este capítulo se analizan los módulos que están contenidos dentro del motor de inferencia: el módulo de recuperación y el módulo de adaptación. Dentro del módulo de recuperación de casos se definirá el algoritmo de recuperación y la función de semejanza a partir de un análisis crítico de los ya existentes. Para la definición del proceso de adaptación se analizarán los distintos métodos para escoger el/los que más se ajustan. También se explicará la necesidad de llevar a cabo el tratamiento de la incertidumbre y de qué manera incluirla dentro del funcionamiento del motor de inferencia.

2.1 El Motor de Inferencia

Los SRBC modelan el proceso de razonamiento humano con un componente conocido como el Motor de Inferencia (MI). Este trabaja con los datos contenidos en la BC para interpretarlos y deducir resultados para un nuevo caso. La tarea fundamental del MI es obtener un nuevo conocimiento que no exista explícitamente en la BC, infiriéndolo a partir de lo que sí está representado a través del módulo recuperador de casos y luego obtener la solución mediante el módulo de adaptación.

Los SRBC se clasifican en uno de los tres niveles siguientes, según el conjunto de facilidades que proporcionen (14):

- Sistemas consejeros que tan sólo recuperan casos: Sin las facilidades de adaptación, evaluación y reparación. El usuario describe una situación y el sistema devuelve casos previos relevantes de los que el usuario extraerá sus propias conclusiones.
- Sistemas con recuperación y adaptación: Sin evaluación ni reparación. El usuario describe una situación, y el sistema encuentra y adapta casos previos similares.
- Sistemas con recuperación, adaptación y reparación: El usuario describe una situación y el sistema encuentra y adapta casos previos relevantes. Si la solución propuesta falla, el sistema la repara y también modifica el mecanismo de recuperación para evitar el mismo error en el futuro.

La utilización del motor de inferencia que se plantea en la propuesta, se centra hasta la recuperación y la adaptación de los casos por no ser objetivo de este trabajo definir el mecanismo de aprendizaje automático, encargado de realizar, como una más de sus tareas la reparación del caso.

2.1.1 Funcionamiento del Motor de Inferencia

La función del MI se puede detallar en una serie de pasos lógicos y secuenciales, empezando por la detección de casos aplicables al nuevo problema que se desea solucionar. Como puede haber más de un caso aplicable, es necesaria la elección de uno de ellos, para lo cual se establece algún criterio de selección que sea capaz de escoger el caso con mayor grado de similitud. En la adaptación se modifica el nuevo caso y se indexa la solución al problema que este plantea, quedando el caso listo para ser almacenado en la BC si fuese necesario, evitando siempre el establecimiento de datos redundantes, lo que vuelve ineficiente la BC.

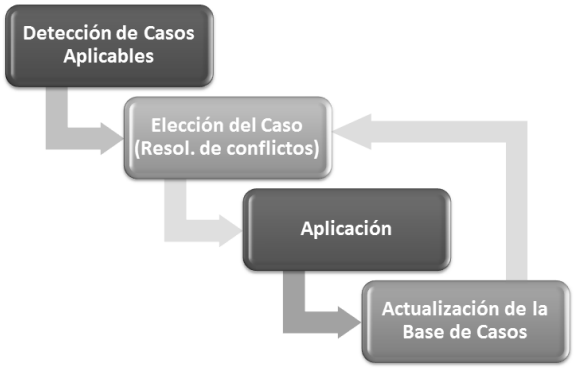

Figura 2.1: Ciclo base del MI

2.2 Módulo de Recuperación

El módulo de recuperación está dividido en dos etapas fundamentales para lograr una mayor organización: la etapa de acceso y la recuperación propiamente dicha.

Entre las técnicas de acceso se puede encontrar (15) (16):

- **Acceso exacto:** Se busca el caso que encaje exactamente con el caso nuevo.
- **Acceso Jerárquico:** Básicamente se buscan los casos en un árbol, tomando decisiones en cada nodo hasta que ya no se pueda avanzar. Si se encuentra una

hoja se muestran los casos correspondientes, si se llegó hasta un nodo, se muestra todos los casos que se deriven de él.

La segunda etapa del módulo de recuperación es quien selecciona los casos más similares al nuevo problema. Para saber qué tan similar es un caso a otro ya existente en la BC, según la investigación realizada, la técnica más sencilla consiste en contar la cantidad de características similares que hay entre un caso y otro. Esta técnica no toma en cuenta la importancia que puede tener una característica con respecto a la otra, la cual varía de un contexto a otro. Para dar solución a esta problemática se crean un conjunto de heurísticas que son las que permiten determinar cuáles características tienen mayor relevancia y así poder definir una función de semejanza que tenga en cuenta la similitud entre los diferentes rasgos (δ) teniendo en cuenta la relevancia (p). Es fundamental que el procedimiento de selección sea rápido de modo que no se pierda la eficiencia del razonador basado en casos.

2.2.1 Algoritmos de recuperación

Los algoritmos de recuperación más investigados según Laura Lozano (16) son: k-vecinos más cercanos, aplicable a la técnica de acceso exacto y árboles de decisión, aplicable el acceso jerárquico.

K-vecinos más cercano

El caso recuperado es aquel en el que la suma de los pesos de las características que se ajustan con las del caso nuevo es mayor que la de otros casos que ajustan. Esto quiere decir, que si los pesos de las características son iguales, un caso que ajusta con n características será recuperado antes que otro caso que empareja con k características, siendo k<n. Se puede asignar un peso mayor a las características consideradas más importantes (16).

Árboles de decisión

Los árboles de decisión representan una estructura de datos que organiza eficazmente los descriptores (17); dichos árboles son construidos de forma tal que en cada nodo se realiza una prueba sobre el valor de los descriptores y de acuerdo con la respuesta se va

descendiendo en las ramas, hasta llegar al final del camino donde se encuentra el valor del objetivo.

En los SRBC la BC es analizada por un algoritmo de inducción que determina qué características son las que mejor diferencian los casos entre sí para producir un árbol de decisión que clasifica los casos. Este enfoque es muy útil cuando se requiere un único caso como solución y cuando las características del caso dependen de otras.

El vecino más cercano es una técnica muy simple que proporciona una medida de cuán similar es un caso objetivo a un caso de la BC. Pero su principal desventaja es la velocidad de recuperación ya que para encontrar el caso que mejor ajusta, el caso nuevo debe compararse con cada caso de la BC. Esto quiere decir que las comparaciones de similitud, se realizarán para cada rasgo predictor indexado al caso.

Para una base de 100 casos con 10 características indexadas habría que realizar 1000 cálculos de similitud. Si el tamaño de la base aumenta a 10000 casos, se tendrían que realizar 100000 cálculos de similitud (16). Esto hace que el algoritmo sea ineficiente a medida que aumenta la BC.

La BC para cual se está diseñando el MI, es de notable tamaño, contando inicialmente con 113 rasgos. Para poder garantizar que a medida que el sistema crezca no se vea afectado el rendimiento de la aplicación, es necesario contar con algoritmos y métodos de acceso que sean flexibles al tamaño. Razón por la cual el algoritmo de recuperación escogido es árboles de decisión y con ello la técnica de acceso a la cual pertenece; el acceso jerárquico.

El árbol de decisión es construido antes de que la recuperación comience. Aunque es también un proceso que consume recursos para una BC grande y tiene que repetirse el proceso cada vez que un nuevo caso es añadido, los tiempos de recuperación, comparado con el método k-vecinos son más rápidos. Una de las desventajas que presenta, al ser una técnica inductiva, es que si los datos de un caso se pierden o son desconocidos no se puede recuperar el caso.

2.2.1.1 Características de los árboles de decisión

Los árboles de decisión son una forma estructurada y secuencial de representar reglas subyacentes en los datos (18). Una de las muchas ramas en las que se pueden explotar los árboles de decisión, es en la clasificación de instancias. Para ello verifican si los datos contienen clases de objetos bien definidas y diferenciadas (excluyentes), de tal forma que dichas clases puedan ser inferidas para clasificar nuevas instancias.

Entre las ventajas más significativas que presentan los árboles de decisión con respecto a otros mecanismos se pueden encontrar que (18):

- Desarrollan la clasificación en una secuencia de preguntas simples y fáciles de entender, cuya semántica es claramente intuitiva para los expertos en el dominio.
- Pueden ser utilizados tanto en problemas determinísticos como en problemas incompletos.
- La descomposición jerárquica implica el mejor uso de las características disponibles y una eficacia computacional en la clasificación.

La representación gráfica de una BC en un árbol de decisión permite clasificar y evaluar un nuevo caso para determinar la clase a la que pertenece, sin tener en cuenta todos los rasgos. Para la construcción del árbol de decisión se hizo un análisis de los principales algoritmos de IA aplicables a un número elevado de datos para obtener información de ellos. Entre los algoritmos más conocidos se encuentra el ID3 utilizado para la construcción de árboles de decisión en función de un conjunto de datos previamente clasificados. Este algoritmo puede aplicarse a cualquier contexto de datos, siempre y cuando las variables tengan valores discretos, pues no presenta facilidad para el trabajo con valores continuos ni numéricos. Esta problemática fue corregida por el algoritmo C4.5 desarrollado por JR Quinlan en 1993, como una mejora del algoritmo ID3 que desarrolló en 1986, cronología mostrada en la figura 2.2:

Figura 2.2: Cronología de los algoritmos desarrollados por Quinlan

Algunos de los rasgos que conforman los casos almacenados en la BC tienen dominio numérico [Anexo 9] razón que hace significativa la limitante que presenta el algoritmo ID3 en el trabajo con este tipo de variables. Por lo tanto, para la construcción del árbol de decisión que permitirá la recuperación de los casos más similares en la BC, se empleará el algoritmo C4.5.

2.2.1.1.1 Algoritmo C4.5

Englobado dentro del llamado aprendizaje inductivo y supervisado, C4.5 es un algoritmo que pretende modelar los datos mediante un árbol de decisión. Tanto éste como su antecesor, el ID3, pertenecen a la familia de métodos de inducción TDIDT[4] y ambos generan árboles usando el método de aprendizaje "divide y vencerás".

Como ventajas de este algoritmo se puede decir que es capaz de determinar qué tan profundo debe crecer el árbol y reduce los errores mediante la poda que implementa de las ramas que brinden información irrelevante. Esta poda es realizada luego que es construido el árbol para mejorar el rendimiento y a su vez obtener un árbol más sencillo y cómodo de inferir. Soporta el trabajo con valores discretos, continuos, numéricos y faltantes, ventaja significativa que marca la principal diferencia con respecto al ID3.

El procedimiento para la construcción del árbol consiste en seleccionar cuál prueba se usará para dividir los datos. En (19) aparecen tres tipos de posibles propuestas que usa el algoritmo C4.5 de las cuales una de ellas es para el tratamiento de valores continuos que como ya se dijo anteriormente no se tienen registrados rasgos con este tipo de dominio en la BC, por lo tanto no se tomará en cuenta. Las otras dos propuestas restantes son para el trabajo con variables discretas que son las que se muestran a continuación:

- La prueba estándar: se obtiene un resultado y una rama para cada valor posible de la variable.
- La prueba compleja: los valores posibles son asignados a un número variable de grupos con un resultado posible para cada grupo, en lugar de para cada valor.

El C4.5 genera un árbol de decisión a partir de los datos mediante particiones realizadas recursivamente, según la estrategia de primero en profundidad. Ante cada partición de datos, el algoritmo considera todas las pruebas posibles que pueden dividir el conjunto de

[4]Top Down Induction Trees: Árbol de inducción con recorrido de arriba hacia abajo

datos y selecciona la prueba en que resulta mayor la proporción de ganancia de información. Este criterio evita beneficiar las variables con mayor cantidad de posibles valores en caso de exista una igualdad de ganancia de la información. Para cada atributo discreto, se considera una prueba con n resultados, siendo n el número de valores posibles que puede tomar el atributo. El orden de los rasgos en el árbol de decisión lo obtendrá el rasgo que mayor valor proporción de ganancia de la información tenga.

Para obtener la proporción de ganancia se parte del cálculo de la entropía para cada uno de los rasgos (fórmula 1), estas magnitudes tienen un comportamiento inversamente proporcional ya que mientras menor sea el valor de la entropía, menor será la incertidumbre del rasgo y más útil para la clasificación, maximizando la ganancia de la información. Luego se calcula la entropía de los subconjuntos en los cuales se dividen cada rasgo (fórmula 2). Con ambos resultados se calcula la ganancia de información para cada rasgo luego de haber sido divididos por varios subconjuntos (fórmula 3) y la de cada uno de los subconjuntos divididos (fórmula 4). Como resultado final se obtiene la proporción de ganancia (fórmula 5).
A continuación se describe una explicación más detallada (20):
Para cada uno de los rasgos se calcula la entropía:

$$\boldsymbol{H(S) = info(S) = -\sum_{j=1}^{k} p_j * \log_2 p_j} \qquad \text{(fórmula 1)}$$

H(S): entropía del rasgo S
p_j: probabilidad de los valores que puede tomar el rasgo S

Luego se calcula la entropía que tendrán los conjuntos resultantes de la división del rasgo según los posibles valores que pueda tomar.

$$\boldsymbol{H((S,X)) = \sum_{i=1}^{n} P(S_i) * H(S_i)} \qquad \text{(fórmula 2)}$$

H(S, X): entropía de los subconjuntos n, en los que son divididos el rasgo X.
P (S_i): probabilidad de ocurrencia del subconjunto i.
H(S_i): entropía del subconjunto i.

Luego se calcula la ganancia resultante de dividir el rasgo E en n subconjuntos:

$$\boldsymbol{Ganancia(E,S) = H(S) - H((S,X))} \quad \text{(fórmula 3)}$$

H(S): entropía del rasgo S

H (S, X): Entropía de los subconjuntos n, en los que son divididos el rasgo X.

Para calcular la proporción de la ganancia de debe conocer además del valor de la ganancia, la división realizada, la cual se calcula:

$$\boldsymbol{I\_división(S) = -\sum_{i=1}^{n} \frac{|S_i|}{|S|} * \log_2\left(\frac{|S_i|}{|S|}\right)} \quad \text{(fórmula 4)}$$

S_i: cantidad de valores del subconjunto i

S: total de valores del rasgo

$$\boldsymbol{Proporción\_de\_Ganancia(S) = \frac{Ganancia(S)}{I\_división(S)}} \quad \text{(fórmula 5)}$$

Para cada rama resultante (nuevo nodo del árbol), se realiza el mismo proceso, se selecciona otro rasgo y se generan los subconjuntos que se expresan en ramas para cada posible valor del rasgo. Este procedimiento continúa hasta que los ejemplos se clasifiquen a través de uno de los caminos del árbol, llegando a un nodo hoja. En la figura 2.3 se muestra el algoritmo C4.5.

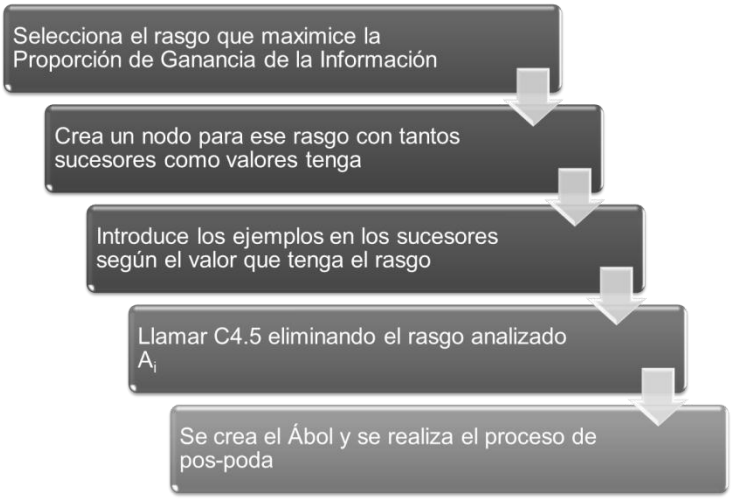

Figura 2.3: Algoritmo C4.5.

2.2.2 Tratamiento de la incertidumbre

Toda la información que se tiene del mundo es incierta, incompleta o imprecisa; nada es totalmente cierto. El propósito de los sistemas de información es modelar el mundo real, por eso, dichos sistemas tienen que ser capaces de modelar la incertidumbre. De forma empírica, la incertidumbre es casi una medida inseparable de casi cualquier media resultante de una combinación de inevitables errores de medición y límites de resolución de los instrumentos de medición utilizados. De forma cognitiva, esta emerge de la vaguedad y la ambigüedad inherente al lenguaje natural (15).

Muchos de los SRBC pueden aplicarse solo a situaciones deterministas. Sin embargo, hay casos prácticos que implican incertidumbre. La estimación de software generalmente conlleva un gran riesgo, y una buena estimación tiene que ser capaz de medirla. En un SRBC para la estimación de software hay posibilidades de imprecisión tanto en el conocimiento implícito en la BC como en el planteamiento del problema. Los rasgos que se tienen en cuenta no afectan con un mismo grado de certeza la obtención de un resultado determinado. Por ello, es útil extender la lógica clásica para incorporar incertidumbre.

Existen algunos momentos del RBC donde resulta necesario tener en cuenta la manipulación de la incertidumbre (21):

- Cuando se tiene un rasgo con un alto nivel de relevancia, pero no se está seguro del valor que debe tomar en ese momento, situación que le atribuye un alto nivel de incertidumbre.
- Cuando se tiene dos rasgos con valores diferentes y el grado de certidumbre de los dos es distinto a 1. Para considerar estos valores de certeza es necesario redefinir la función de semejanza definida originalmente.
- Cuando se tiene dos casos recuperados y ambos tienen el mismo nivel de semejanza, sin embargo el valor que alcanza el rasgo objetivo es diferente para ambos.

Los rasgos que componen los casos almacenados en la BC, están formados por tres elementos: el nombre, el valor y la relevancia; pero la necesidad de representar una información adicional que indique el grado de certeza del valor de un rasgo da lugar a la inserción de un nuevo elemento, la incertidumbre.

Cada valor de incertidumbre representará un conjunto difuso continuo ubicado en un intervalo de [0; 1] donde el 0 representa el desconocimiento total que se tiene del rasgo mientras que 1 el conocimiento pleno del mismo. Los valores ubicados dentro de este intervalo se traducen en el grado de incertidumbre que se tiene sobre el rasgo, donde los valores que tienden a 0 representan un mayor grado de incertidumbre mientras los que tienden a 1 un mayor grado de certeza.

Las incertidumbres de los rasgos predictores pueden ser especificadas en caso de que el especialista la conozca, pero, previendo que no siempre son expertos los que realizarán esta labor, el módulo recuperador propone un valor de incertidumbre para cada uno de los rasgos predictores haciendo uso de la ingeniería del conocimiento, partiendo de las previas incertidumbres que este tiene ya registrado en la BC para cada uno de los rasgos y utilizando un enfoque probabilístico. El resultado describirá el comportamiento más probable que puedan tener las incertidumbres desconocidas para cada rasgo respectivamente.

$$\boldsymbol{I_{O_i}} = \frac{\sum_{j=1}^{n} \boldsymbol{I_j}}{\boldsymbol{n}} \qquad \text{(fórmula 6)}$$

$\mathbf{I_{O_i}}$: Incertidumbre no especificada para el rasgo i del nuevo caso O.

$\mathbf{I_j}$: Incertidumbre del rasgo i para el caso j de la BC

$\boldsymbol{n}$: Cantidad de casos de la BC.

Teniendo en cuenta el nuevo elemento que se le adiciona al rasgo, la estructura real de la los rasgo predictores de la BC sería como se visualiza en la figura 2.4:

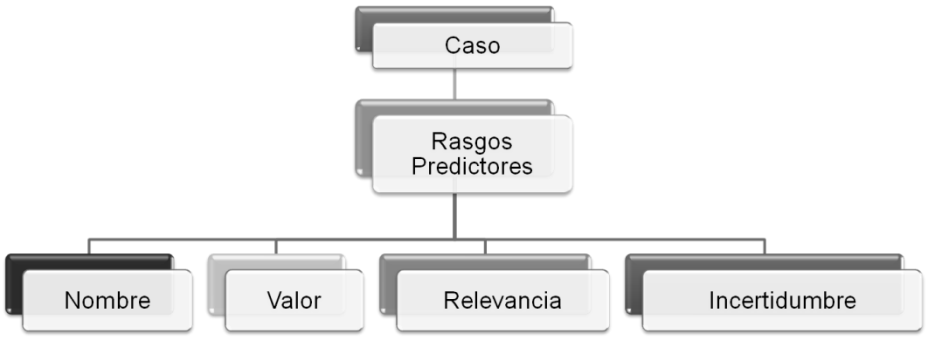

Figura 2.4: Estructura de los rasgos predictores

Luego de la elección del algoritmo de recuperación basado en árboles de decisión C4.5, es necesaria la definición de una función de semejanza, que sea capaz de obtener el caso más semejante de una serie de casos recuperados, teniendo en cuenta la incertidumbre.

2.2.3 Función de semejanza

La función de semejanza es la encargada de determinar el grado de igualdad que tiene el caso a resolver con respecto a los casos semejantes obtenidos de la BC. No existe una medida de semejanza única para cualquier dominio, de ahí que la eficiencia del sistema radica en la función de semejanza que se defina (15).

En la búsqueda por semejanza, el objetivo es recuperar el elemento que más se parece al nuevo problema a partir de calcular una medida que cuantifique ese grado de similitud. La función de semejanza puede incluir entre sus argumentos, la importancia o relevancia de cada rasgo al integrar los valores obtenidos en la comparación rasgo a rasgo, de modo que el resultado de la comparación de rasgos más relevantes, tenga más peso en el cálculo del grado de similitud entre el patrón de búsqueda y un elemento de la memoria permanente.

Existe una fórmula básica para calcular la semejanza entre un nuevo problema a resolver O_0 y un caso O_t de la BC (21).

$$\boldsymbol{\beta(O, O_t)} = \frac{\sum_{i=1}^{n} \boldsymbol{p_i} * \boldsymbol{\delta(O, O_t)}}{\sum_{i=1}^{n} \boldsymbol{p_i}} \qquad \text{(fórmula 7)}$$

n: Número de rasgos predictores.

P_i: Relevancia del rasgo i.

$s_i(O, O_t)$: Función de comparación entre los casos O y O_t del al rasgo i.

Pero esta función no contempla el tratamiento de la incertidumbre en los parámetros que mide para la obtención de su resultado, razón por la cual necesita ser modificada. Para ello es necesario agregarle alguna función matemática que exprese las relaciones entre las incertidumbres del rasgo i para ambos casos que se analizan.

Esta función tiene que mantener el sentido de la función de semejanza y ser totalmente lógica su afectación en la comparación de los rasgos (22):

$$\beta(O, O_t) = \frac{\sum_{i=1}^{n} p_i * \delta(O, O_t) * (1 - |I_o - I_{o_t}|)}{\sum_{i=1}^{n} p_i} \quad \text{(fórmula 8)}$$

I_o: Incertidumbre del caso O para el rasgo i.

I_{o_t}: Incertidumbre del caso O_t para el rasgo i.

Esta fórmula matemática que se le agrega a la función de semejanza, representa la varianza entre las incertidumbres. Su resultado afectará a medida que dichos valores sean más dispersos. Cuando el resultado tienda a 1, afectará con menor medida el resultado de la función de similitud con respecto al rasgo i de los casos correspondientes, mientras que si tiende a 0 significa una mayor dispersión entre las incertidumbres, lo que se traduce en un baja confiabilidad de los valores, por lo que necesariamente el resultado de la similitud entre rasgos se verá más afectado. Su valor neutro estará dado en caso de que las incertidumbres de ambos casos sean iguales.

2.2.3.1 Similitud entre rasgos

La función de similitud entre rasgos (δ) puede tener diferentes definiciones, en dependencia del tipo de dato que represente el rasgo. Los casos que están registrados en la BC están definidos por indicadores específicos que se necesitan para poder dar una medida de estimación de la duración para un proyecto de software. Para poder calcular la similitud entre los rasgos, es necesario definir la función que mejor se ajuste al conjunto de valores admisibles en que se enmarcan los valores de cada rasgo (dominio). Para hallar la semejanza de cada rasgo predictor con los datos insertados del nuevo caso, fueron utilizadas diferentes funciones según los valores de dominio que abarcan.

2.2.3.1.1 Similitud entre rasgos numéricos

Para valores numéricos [Anexo 2] existen una serie de fórmulas que determinan la similitud entre dos rasgos, derivadas de basamento en principios geométricos. Cada elemento de memoria se puede considerar un punto en un espacio n-dimensional (donde n es la cantidad de rasgos). Frecuentemente la semejanza entre dos elementos se calcula como el grado de cercanía entre los puntos correspondientes. La cercanía entre dos puntos es inversamente proporcional a la distancia que los separa. Asumiendo que un objeto se puede representar como una lista de N pares (atributo, valor), donde cada valor

es numérico, el mismo representa un punto en el espacio de dimensión N. La definición geométrica de distancia cuantifica la similitud entre dos objetos. La distancia es una magnitud que mide la relación de lejanía o cercanía entre dos cuerpos, objetos o individuos (15). Partiendo de la definición planteada anteriormente se analizan las funciones de distancia euclidiana (fórmula 9), la distancia absoluta o Manhattan (fórmula 10) y la Manhattan ajustada (fórmula 11), tomadas de (23).

Distancia euclidiana

$$\boldsymbol{\delta(O_o, O_t)} = \sqrt{(\boldsymbol{X_i(O_0)} - \boldsymbol{X_i(O_{0_t})})^2}$$ (fórmula 9)

Distancia absoluta o Manhattan

$$\boldsymbol{\delta(O_o, O_t)} = |\boldsymbol{X_i(O_0)} - \boldsymbol{X_i(O_{0_t})}|$$ (fórmula 10)

Distancia Manhattan ajustada

$$\boldsymbol{\delta(O_o, O_t)} = \frac{|\boldsymbol{X_i(O_0)} - \boldsymbol{X_i(O_{0_t})}|}{\boldsymbol{Max_i - Min_i}}$$ (fórmula 11)

$\boldsymbol{\delta(O_o, O_t)}$: semejanza del rasgo i para caso O_0 y el caso O_t.
X_i (O_0): valor del rasgo i para el caso O_0.
X_i (O_t): valor del rasgo i para el caso O_t.

Todas las fórmulas expuestas anteriormente dan una buena medida de distancia y por ende de semejanza entre 2 valores numéricos, por ello para su elección, se tendrán en cuenta las características de los rasgos numéricos almacenados en la BC escogiendo la que mejor se ajusta.

Los rasgos numéricos que se analizan, están definidos por valores mayores que 1, y sin un intervalo específico. Esto puede representar un inconveniente a la hora de realizar las comparaciones, ya que estas pueden estar favorecidas erróneamente por aquellos rasgos que tienen altos valores numéricos y una varianza significativa entre ellos, obteniendo como resultados valores elevados, opacando así, los efectos que se pudieran obtener luego de comprar el resto de los rasgos. Como acción para mejorar esta problemática se hace uso de la de la distancia Manhattan ajustada (fórmula 11), ya que esta hace posible, como bien dice su nombre, de ajustar los valores de los rasgos dentro del intervalo

acotado por el mayor y el menor valor que toma el rasgo en la BC. El resultado luego de aplicar esta fórmula estará establecido en el intervalo [0; 1].

Luego de definida la fórmula 11 para comparar los rasgos numéricos, es necesario mejorar el sentido del resultado obtenido. Si se tiene en cuenta que para dos rasgos totalmente iguales el valor de la semejanza es 1 y entre más tienda a 1 este valor significa un mayor grado de similitud, entonces el complemento de del resultado obtenido por la fórmula 11 sería el valor más coherente que definiría la semejanza entre dos valores numéricos. Siendo la fórmula 12 finalmente la que se utilizará para comprar dos rasgos numéricos, tomada de (23).

$$\boldsymbol{\delta(O_o, O_t) = 1 - \frac{|X_i(O_0) - X_i(O_{0_t})|}{Max_i - Min_i}}$$ (fórmula 12)

2.2.3.1.2 Similitud entre rasgos ordinales

Los valores ordinales son aquellos que las categorías están ordenadas en una secuencia significativa de valores. Estos son muy útiles para el registro de las evaluaciones subjetivas de cualidades que no pueden ser medidas objetivamente (23). Para el tratamiento de estas variables se definirán valores numéricos para los diferentes estados ordenados para la clasificación. El conjunto quedaría conformado {1,…, M}, donde M, es el último valor ordenado. El rasgo que se analiza del nuevo caso, tendrá un valor perteneciente a este intervalo. En los anexos [Anexo 3 - Anexo 7] se define el valor numérico que estará en correspondencia con cada categoría del rasgo ordinal, en dependencia de la cantidad de estados que tenga dicho rasgo. Es muy conveniente llevar el valor de esas variables aun intervalo de [0.0; 1.0], realizando un proceso de normalización (fórmula 13), de modo que cada variable tenga el mismo peso. Esto se puede lograr evaluando los valores en la siguiente fórmula tomada de (23):

$$\boldsymbol{V_{iO} = \frac{R_{iO} - 1}{M_{iO} - 1}}$$ (fórmula 13)

$\boldsymbol{V_{iO}}$: valor normalizado del rasgo i para el caso O.

$\boldsymbol{R_{iO}}$: valor numérico del rasgo i para el caso O.

$\mathbf{M_{iO}}$: último valor del orden perteneciente al intervalo numérico del rasgo ordinal i para del caso O.

Esta fórmula es aplicada al rasgo i del caso O (del nuevo caso) y O_t (del caso recuperado). Como los valores obtenidos son numéricos, para hallar la semejanza entre ellos se podrá aplicar cualquier fórmula definida para los rasgos numéricos. En el proceso de normalización se definieron los valores para un intervalo ya especificado, por lo que no es necesario usar la fórmula 11 como lo fue en caso de los rasgos numéricos. En este caso pueden ser aplicables cualquiera de las otras dos fórmulas, la distancia euclidiana (fórmula 9) o la distancia Manhattan (fórmula 10). Ambas arrojan el mismo resultado, pero por la sencillez que la de define, es escogida para la propuesta la distancia Manhattan (fórmula 10), a la cual se le realiza la misma modificación hecha para la Manhattan ajustada, donde se toma como el valor real de la comparación, el complemento de su resultado, como se evidencia en la fórmula 14 tomada de (23).

$$\boldsymbol{\delta(O_o, O_t) = 1 - |V_i(O_0) - V_i(O_t)|} \quad \text{(fórmula 14)}$$

$\boldsymbol{V_{i_O}}$: valor normalizado del rasgo i para el caso O_0.
$\boldsymbol{V_{i_{O_t}}}$: valor normalizado del rasgo i para el caso O_t.

Para los valores nominales [Anexo 8] y binarios [Anexo 9], se utilizará para cuantificar su similitud una medida absoluta (fórmula 15), la cual devolverá el valor de 1 si son exactamente iguales, 0 en cualquier otro caso. Esta fórmula fue tomada de.

$$\boldsymbol{\delta(O_o, O_t) = \begin{cases} 1 \; si \;\; X_i(O_0) = X_i(O_t) \\ 0 \;\; e.o.c \end{cases}} \quad \text{(fórmula 15)}$$

$\boldsymbol{X_i(O_0)}$: valor del rasgo i para el caso $\boldsymbol{O_0}$.
$\boldsymbol{X_i(O_t)}$: valor del rasgo i para el caso $\boldsymbol{O_t}$.

De esta forma, cada caso nuevo será comparado con los existentes. Este proceso se realiza de forma iterativa para los casos recuperados de la BC obteniendo así el caso que exprese mayor semejanza a la descripción especificada.

2.2.4 Valor Umbral

El umbral es la cota inferior de los posibles valores de semejanza existente para cada uno de los casos recuperados con respecto al nuevo caso. Su verdadera función consiste en restringir el intervalo en caso de que se quiera lograr una mayor precisión en la obtención de los casos más semejantes. Este umbral podrá ser definido por el usuario que interactúe con el sistema, y este puede variarlo en medida de qué tan exigente pueda ser con el resultado. Un umbral muy elevado puede representar la ventaja de que solo se tengan en cuenta en la fase de adaptación los casos con un alto nivel de semejanza, pero puede presentar la desventaja de que ninguno de los casos recuperados lleguen o sobrepasen este valor, lo que traería consigo no tener al menos un caso para poder realizar la adaptación. Si el usuario presenta problemas con la definición del umbral, el módulo de recuperación le brindará el más adecuado según los datos que tiene registrados en la BC.

Para ello construye una matriz cuadrada donde la fila y las columnas están representadas por los casos que se encuentran almacenados en la BC y en la intersección está el valor de semejanza ($\boldsymbol{\beta}$).

	Caso$_1$	**Caso$_2$**	**Caso$_3$**	**Caso$_n$**
Caso$_1$	$\boldsymbol{\beta}(C_1, C_1)$	$\boldsymbol{\beta}(C_1, C_2)$	$\boldsymbol{\beta}(C_1, C_3)$	$\boldsymbol{\beta}(C_1, C_n)$
Caso$_2$	$\boldsymbol{\beta}(C_2, C_1)$	$\boldsymbol{\beta}(C_2, C_2)$	$\boldsymbol{\beta}(C_2, C_3)$	$\boldsymbol{\beta}(C_2, C_n)$
Caso$_3$	$\boldsymbol{\beta}(C_3, C_1)$	$\boldsymbol{\beta}(C_3, C_2)$	$\boldsymbol{\beta}(C_3, C_3)$	$\boldsymbol{\beta}(C_3, C_n)$
Caso$_n$	$\boldsymbol{\beta}(C_n, C_1)$	$\boldsymbol{\beta}(C_n, C_2)$	$\boldsymbol{\beta}(C_n, C_3)$	$\boldsymbol{\beta}(C_n, C_n)$

Tabla 2.7: Matriz de semejanza entre casos

Luego de construida la matriz, mediante el cálculo de μ se obtiene el valor umbral (fórmula 16) que dependerá de los valores de semejanza entre los casos contenidos en dicha matriz. Tomado de (24):

$$\boldsymbol{\mu} = \frac{\mathbf{2}}{\boldsymbol{m(m-1)}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{m-1}} \sum_{\boldsymbol{j=i+1}}^{\boldsymbol{m}} \boldsymbol{\beta}(\boldsymbol{O_0}, \boldsymbol{O_t}) \qquad \text{(fórmula 16)}$$

m: número de casos
i: valor que recorrerá las filas
j: valor que recorrerá las columnas
$\boldsymbol{\beta}(\boldsymbol{O_0}, \boldsymbol{O_t})$:semejanza entre el caso 0 y el caso t

Luego de definido el umbral y comparar los valores de semejanza de los casos recuperados, solamente pasan a la fase de adaptación aquellos casos que sean iguales o estén por encima.

2.3 Módulo de Adaptación

Una vez que se selecciona los casos similares, se efectúa el procedimiento de adaptación, que consiste en la modificación y combinación de las soluciones de los casos similares para formar una nueva solución. Las técnicas desarrolladas para la realización de la adaptación en forma automática han dado en general buenos resultados. Un proceso de adaptación está compuesto por: identificación de aquello que hay que adaptar y aplicación de un método de adaptación. Es importante señalar, algo que Rafael E. Bello define en su publicación (15), que cuando se cuenta con una BC lo suficientemente grande, no es necesario aplicar métodos de adaptación muy complejos.

Existen varias definiciones de métodos de adaptación los cuales están agrupados en dos colecciones, los métodos de sustitución y los de transformación (15) (16). Estos pueden aplicarse de manera individual o aprovechar las facilidades que brindan más de uno y hacerlas converger para mejorar el resultado.

Los métodos de sustitución dan valores apropiados para la nueva situación dependiendo de los valores de la solución previa. Se puede sustituir sólo un componente de la solución, varios de ellos, o todos los componentes de una solución. Estos son usados cuando se cuenta con una BC bastante sólida. Cuando no se cuenta con una BC lo suficientemente sólida *los métodos de transformación* son una alternativa muy viable, pues generan una nueva solución basándose en las restricciones y características de la solución requerida.

Los métodos de transformación de una forma u otra realizan modificaciones en los rasgos predictores. Para los casos que se encuentran almacenados en la BC, realizar una modificación significa cambiar los indicadores reales que definen las características de un

proyecto. De la misma manera pasa con los rasgos del nuevo caso que se quiere analizar, ajustar algunos de sus rasgos para lograr una mayor semejanza con el/los casos recuperados, significaría cambiar la situación real que tiene el proyecto que se quiere estimar y el resultado que arrojaría no es lo que se pretende. La necesidad es poder estimar el tiempo con los indicadores reales del proyecto, no ajustar el proyecto para cumplir con un determinado tiempo. Partiendo de la necesidad planteada, los métodos de transformación no representan una posible solución para la adaptación, lo cual conlleva al análisis de los métodos de sustitución.

Agrupados dentro de los métodos de sustitución se pueden encontrar la reinstanciación y el ajuste de parámetros. Ambos métodos no necesitan realizar ningún cambio en los rasgos, ya que la solución se obtiene indexando el rasgo objetivo del caso recuperado al nuevo caso. La diferencia entre estos métodos radica en el criterio de selección que utilizan para escoger, de los casos recuperados, el más adecuado para realizar la adaptación.

El ajuste de parámetros propone tener en cuenta además del grado de similitud entre los casos, la semejanza que existe entre los principales rasgos que son capaces de definir el rasgo objetivo. Para ello, compara estos rasgos más importantes, y escoge el caso que tenga una mayor coincidencia entre ellos. Este criterio de elección se puede ver afectado con la aparición de la incertidumbre para cada rasgo predictor, pues puede ocurrir que los rasgos definidos como imprescindibles para obtener el resultado de la estimación, presenten valores muy bajos de certeza. Este tipo de inconveniente no es tenido en cuenta por el ajuste de parámetros, por lo que también es desechado como posible vía para la adaptación.

La reinstanciación o adaptación nula sin embargo, es más flexible a la hora de definir un criterio de selección, ya que no especifica un criterio estándar. Esta adaptación se pone en práctica en la mayoría de los sistemas de clasificación. Para establecerlo se necesita una BC que tenga una gran cantidad de casos almacenados, ya que la variedad garantizará la precisión del resultado del problema. Esta característica es heredada del grupo al que pertenece. En (21) Rafael Bello propone el cálculo de la utilidad esperada como criterio de selección para llevar a cabo la adaptación nula, en la siguiente sección se detalla en que consiste.

2.3.1 Utilidad esperada del Caso

La utilidad esperada del caso (fórmula 17), es una fórmula matemática, donde se expresa un criterio entre la semejanza del caso devuelta por la función de semejanza y la incertidumbre del caso similar (15).

La BC tiene solo un rasgo objetivo, el tiempo que duraría el proyecto descrito por los rasgos que definen al nuevo caso. Pero además de este rasgo, el caso tiene un valor de incertidumbre, el cual es representado con un número que expresa el grado de veracidad del resultado con respecto al valor de los rasgos predictores. En caso de que el módulo de recuperación devuelva un solo caso (situación que se da poco cuando se cuenta con una BC amplia), la adaptación solo pudiera estar dada por este caso, pero sin embargo, si el resultado fuese más de uno, entonces sería necesario basarse en algún criterio para escoger el caso con el cual se realizará la adaptación.

2.3.2 Utilidad esperada del Caso

La utilidad esperada del caso (fórmula 17), es una fórmula matemática, donde se expresa un criterio entre la semejanza del caso devuelta por la función de semejanza y la incertidumbre del caso similar (15).

$$\boldsymbol{\mu}(\boldsymbol{O_t}) = \boldsymbol{\alpha}\,\boldsymbol{\beta}(\boldsymbol{O}, \boldsymbol{O_t}) + (\mathbf{1} - \boldsymbol{\alpha}) * \boldsymbol{\vartheta}(\boldsymbol{O_t}) \qquad \text{(fórmula 17)}$$

$\boldsymbol{\mu}(\boldsymbol{O_t})$: utilidad del caso O con respecto al caso recuperado O_t.
$\boldsymbol{\beta}(\boldsymbol{O}, \boldsymbol{O_t})$: valor de la semejanza entre el nuevo caso O y el caso recuperado O_t.
$\boldsymbol{\vartheta}(\boldsymbol{O_t})$: valor de la incertidumbre del caso recuperado O_t.

El valor de α es un parámetro que a medida que tiende a 1 le dará más importancia a la semejanza que a la certidumbre de la solución. Es muy común que este valor sea dado por un experto que sea capaz de identificar para su problema que es lo que más necesita, si un valor más cierto, o uno más semejante. Si el usuario sabe decidir qué es lo que quiere obtener, podrá especificar el valor de α con un número ubicado en el intervalo [0; 1].

En caso de no ser un experto o se desconozca el posible valor que pueda tomar α, se hará la propuesta de un valor calculado aplicando la ingeniería del conocimiento. Este

valor deducido partirá del principio de que lo más importante para los sistemas inteligentes son valores que ellos mismos son capaces de definir, confiando más en lo que él obtiene de su conocimiento que en cualquier dato introducido. De este modo el valor de α dependerá de las incertidumbres que son especificadas por el usuario; si la cantidad de incertidumbres introducidas sobrepasan a la cantidad calculada se le estará dando más importancia a la semejanza entre rasgos, en caso contrario, sería la incertidumbre la favorecida. Para ello se utilizará la siguiente función definida por la investigadora:

$$\alpha = \frac{Cant\_Incert\_Usuario}{Cant\_Rasgos} \qquad \text{(fórmula 18)}$$

$Cant\_Incert\_Usuario$: la cantidad de valores de incertidumbre que el usuario introdujo
$Cant\_Rasgos$: cantidad de Rasgos que tiene el caso

Luego de haber obtenido el valor de la utilidad, el caso seleccionado para realizar la adaptación y con ello dar respuesta al problema planteado en la descripción del nuevo caso, será el que mayor utilidad tenga. Pero el nuevo caso que ya ha sido adaptado, todavía no posee un valor de incertidumbre, pues este caso no tiene que tener necesariamente la misma incertidumbre del caso que se adaptó la solución.

2.3.3 Incertidumbre del nuevo Caso

Recordando que la incertidumbre del caso está en correspondencia de qué tan cierto es el valor de los rasgos, para obtener la conclusión del rasgo objetivo, su cálculo dependerá del grado de semejanza entre el caso con mayor utilidad esperada y el nuevo caso, además del valor de incertidumbre del primero. Partiendo del análisis que en el mejor de los casos la semejanza entre ellos haya dado 1, esto significa que los casos son completamente iguales, por lo tanto se tendría el mismo nivel de certeza con respecto al resultado. Sin embargo, si la semejanza entre los casos no alcanzara su valor máximo, aunque el resultado que se utilice sea el mismo por la gran similitud que existe entre ellos, el grado de certeza disminuirá, lo cual es una buena manera de significar la diferencia entre estos casos a través de la incertidumbre, representada por la varianza entre ambos casos. La fórmula matemática que propone la investigadora, para hallar dicha relación y que cumpla con los principios planteados es la siguiente:

$$\boldsymbol{I_c = \beta(O, O_u) * I_u}$$ (fórmula 19)

$\boldsymbol{I_c}$: el valor de incertidumbre del nuevo caso que se analiza *O.*
$\boldsymbol{\beta(O, O_u)}$: valor de la semejanza entre el nuevo caso O y el caso de mayor utilidad esperada O_u.
$\boldsymbol{I_u}$: incertidumbre del caso con mayor utilidad esperada.

Una vez definida la manera del calcular la incertidumbre del nuevo caso, este estará definido en su totalidad. Obteniendo como resultado, el valor de la estimación de la duración del proyecto de software descrito mediante los rasgos, con un grado de incertidumbre.

Conclusiones Parciales

En este capítulo se realizó un análisis de los principales módulos que componen el motor de inferencia que permitió seleccionar los mecanismos de trabajo que mejor se ajustan a la solución que se quiere brindar. En el módulo de recuperación se escogió como técnica de acceso la jerárquica, y como algoritmo de recuperación, los árboles de decisión específicamente el algoritmo C4.5 por superar en eficiencia los resultados que se pudieran obtener aplicando otros métodos y algoritmos en una BC de gran tamaño. La función de semejanza quedó definida utilizando diferentes funciones de similitud en dependencia del dominio de los rasgos y se modificó para que soportara el tratamiento de la incertidumbre. En el módulo de adaptación se definió a la adaptación nula, escogiendo como criterio de selección al caso de mayor utilidad esperada.

Capítulo 3: Validación de la propuesta

Introducción

En este capítulo se analizarán los resultados obtenidos tras las validaciones realizadas a la propuesta del MI utilizando como herramientas de apoyo para la visualización de los resultados RapidMiner. Se mostrarán las descripciones detalladas de los datos que forman la BC, los resultados obtenidos del proceso de clasificación utilizando el algoritmo C4.5 para la construcción del árbol de decisión. Además utilizando una BC reducida teniendo en cuenta el criterio de experto para la asignación de la clase, el peso y la incertidumbre a cada uno de los rasgos, se muestra el resultado de la validación del proceso de inferencia.

3.1. Herramienta de apoyo. RapidMiner

Incluido entre las herramientas del género de aprendizaje automático, RapidMiner es una poderosa herramienta de código abierto utilizada mayormente para el análisis y la minería de datos. Este proporciona más de 500 operadores orientados al análisis de datos, incluyendo los necesarios para realizar operaciones de entrada/salida, procesamiento de datos, visualización y clasificación. Posee las características de estar programado en java y ser multiplataforma. Su proceso de descubrimiento de conocimientos es modelado por árboles de operación, lo que permite definir en una jerarquía la secuencia lógica que debe tener los procesos que se llevarán a cabo, para la obtención de resultados específicos.

En la figura 3.1 se muestra el flujo diseñado que permitió la validación de la clasificación y la relación entre los componentes. El componente *Retrieve*, es el encargado de cargar los datos provenientes de un Excel donde se almacenaron los 40 casos. Luego en el componente *Entrenamiento,* se especificó el modelo a construir, en este caso un árbol de decisión mediante el algoritmo C4.5. El componente *Prueba* permitió especificar como se realizará la prueba, donde se puntualiza que para llevar a cabo la validación, el conjunto de prueba será el mismo que el de entrenamiento y en el componente *Evaluación*, se especificaron las diferentes estadísticas que se deseaban evaluar, estas coinciden con las mostradas en la tabla 3.4. Estos tres últimos componentes están contenidos dentro de *XVal.*

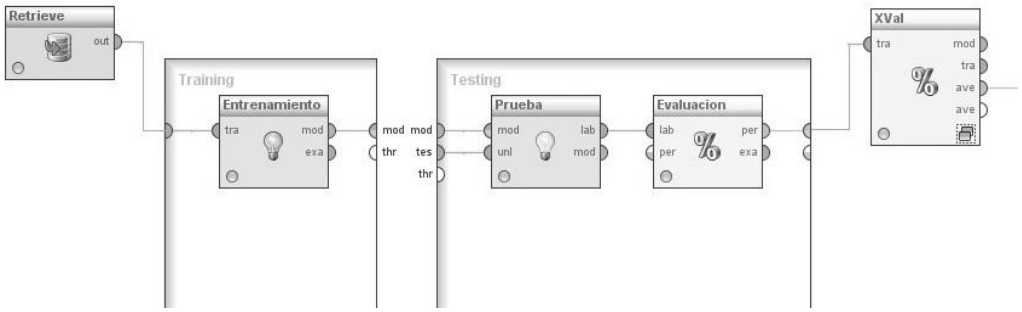

Figura 3.1: Flujo del proceso ejecutado en el RapidMiner para realizar la prueba a la clasificación y la obtención del modelo.

3.2. Descripción de los datos

Para la validación del clasificador se tomará una BC confeccionada con el criterio de experto. Para ello se hizo uso solamente de los rasgos que este consideró más relevantes a la hora de realizar una estimación de software (Tabla 3.1). Quedando un total de 48 rasgos significativos que compondrán el total de rasgos predictores. Los valores fueron tomados aleatoriamente y luego ajustados para evitar contradicciones que podrían dificultar la clasificación del tiempo de duración (rasgo objetivo) que el experto definiera para cada proyecto hipotético (caso). La BC contará con una cantidad de 40 casos y cada rasgo tendrá además del valor, el peso y la incertidumbre, obtenida igualmente por el criterio del experto.

a7	a36	a67	a96
a8	a40	a70	a97
a9	a43	a71	a98
a14	a44	a74	a100
a15	a45	a75	a101
a17	a53	a86	a102
a21	a54	a88	a103
a22	a55	a90	a104
a24	a56	a91	a107
a25	a59	a92	a108
a33	a64	a93	a109
a34	a66	a95	a111

Tabla 3.1: Rasgos seleccionados por el criterio de experto

De la selección de los rasgos predictores más representativos que se muestra en la tabla 3.1, se realizó un resumen agrupando los rasgos en dependencia del dominio al que pertenecen. (Figura 3.2), especificando en la tabla 3.2 los valores para cada rasgo:

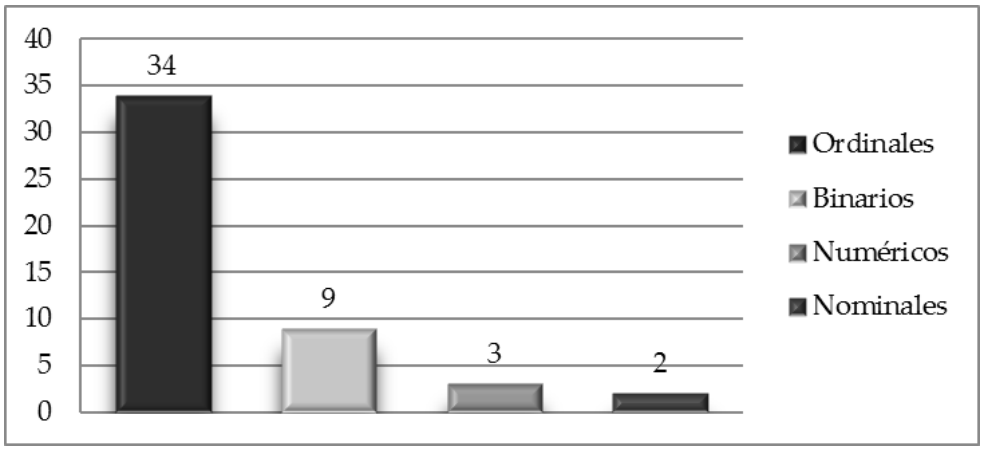

Figura 3.2: Dominio de los Rasgos

Rasgo	Tipo de Valor	Valor/Cantidad
a7	Ordinal	Esencial (6), Significativo (14), Medio (7), Incidental (8), Moderado (5)
a8	Ordinal	Sin_influencia (10), Significativo (8), Moderado (13), Incidental (9)
a9	Ordinal	Esencial (13), Incidental (12), Moderado (15)
a14	Binario	Si (18), No (22)
a15	Binario	Si (18), No (22)
a17	Binario	Si (19), No (21)
a21	Binario	Si (13), No (27)
a22	Binario	Si (19), No (21)
a24	Nominal	Java (19), Php (3), Python (6), C++ (8), Charp (3), PHP (1)
a25	Binario	Si (21), No (19)
a33	Binario	Si (24), No (16)
a34	Binario	Si (13), No (27)
a36	Ordinal	Eclipse (17), visual_estudio (12), Netbean (6), Zend_Estudio (5)
a40	Numérico discreto	[21.000 ; 41.000]
a43	Ordinal	Alto (18), Bajo (4), Medio_alto (8), Medio_bajo (8), Medio (2)
a44	Ordinal	Medio (7), Medio_alto (14), Medio_bajo (3), Alto (15), Bajo (1)
a45	Ordinal	Alto (13), Medio (19), Bajo (5), Medio_alto (3)
a53	Binario	Si (16), No (24)
a54	Ordinal	Ninguna (23), General (5), Completa (3), Pequeña (5), Algo (4)
a55	Ordinal	Algo (23), General (5), Completa (12)

a56	Ordinal	Normal (10), Alto (7), Extra (6), Muy_bajo (9), Bajo (5), Muy_alto (3)
a59	Ordinal	Medio (11), Extremo (13), Significativo (6), Considerable (3), Muy_Poco (4), Poco (3)
a64	Ordinal	Poca (8), Ninguna (19), Basica (7), Vasto (5), Considerable (1)
a66	Ordinal	No_se_conoce (18), Casi_siempre (18), No_se_aplica (2), Ocasionalmente (2)
a67	Ordinal	La_mitad_de_las_veces (14), No_se_aplica (18), Casi_siempre (8)
a70	Ordinal	A_menudo (14), Casi_siempre (23), La_mitad_de_las_veces (3)
a71	Ordinal	La_mitad_de_las_veces (8), A_menudo (14), Casi_siempre (18)
a74	Ordinal	A_menudo (6), No_se_aplica (3), Ocasionalmente (11), Casi_siempre (14), No_se_conoce (6)
a75	Ordinal	Ocasionalmente (11), A_menudo (16), Casi_siempre (6), No_se_aplica (3), No_se_conoce (4)
a86	Ordinal	Alguna (20), Considerable (20)
a88	Ordinal	Considerable (12), Total (16), Basica (12)
a90	Ordinal	Extra (24), Muy_alto (14), Alto (2)
a91	Ordinal	Muy_alto (13), Alto (16), Normal (11)
a92	Ordinal	Extra (8), Bajo (20), Muy_bajo (5), Alto (7)
a93	Ordinal	Muy_alto (17), Alto (6), Muy_bajo (6), Bajo (7), Extra (4)
a95	Ordinal	Muy_alto (13), Alto (16), Normal (11)
a96	Ordinal	Normal (17), Alto (13), Bajo (7), Extra (3)
a97	Ordinal	Alto (21), Muy_alto (9), Extra (2), Normal (7), Bajo (1)
a98	Ordinal	Muy_alto (10), Alto (11), Bajo (12), Normal (6), Muy_bajo (1)
a100	Ordinal	Muy_alto (24), Alto (3), Nominal (13)
a101	Ordinal	Bajo (24), Nominal (10), Muy_alto (5), Alto (1)
a102	Ordinal	Bajo (12), Muy_alto (7), Normal (14), Muy_bajo (4), Alto (3)
a103	Ordinal	Alto (17), Normal (10), Bajo (5), Muy_bajo (8)
a104	Ordinal	Normal (28), Muy_alto (6), Muy_bajo (6)
a107	Ordinal	Alto (15), Muy_alto (10), Bajo (10), Muy_bajo (5)

a108	Numérico discreto	[5.000 ; 15.000]
a109	Numérico discreto	[15.000 ; 29.000]
a111	Nominal	Criptografía (8), Gestion (9), Realidad_virtual (8), Empotrados (8), Multimedia (7)

Tabla 3.2: Valores de los rasgos predictores

El rasgo objetivo está definido como un valor nominal y los distintos valores que toma son los siguientes:

Tiempo	Nominal	34meses (1), 24meses (2), 22meses (3), 32meses (3), 28meses (2), 31meses (1), 13meses (3), 11meses (1), 16meses (1), 9meses (2), 8meses (2), 27meses (1), 25meses (1), 12meses (2), 29meses (1), 21meses (1), 39meses (1), 33meses (2), 14meses (2), 30meses (2), 35meses (1), 17meses (2), 18meses (2), 20meses (1)

Tabla 3.3: Valores de los rasgos objetivos

3.3 Prueba para la clasificación

La estrategia para validar un sistema de aprendizaje depende esencialmente de la manera que la división es realizada. Para esta prueba de clasificación el conjunto de casos de entrenamiento y el conjunto de casos de prueba coinciden con los 40 casos clasificados y validados por el experto. Para ello se tratará de clasificar cada uno de los casos, obteniendo como instancias clasificadas correctamente, si la clasificación es capaz de arrojar al mismo caso como solución. Esta clasificación se realizará utilizando la herramienta RapidMiner.

En la tabla 3.4 se muestra una serie de resultados estadísticos de las pruebas realizadas al modelo de clasificación construido.

Estadísticas del Árbol C4.5	
Instancias clasificadas correctamente	27 (67.5%)
Instancias clasificadas Incorrectamente	13 (32.5%)
Estadístico Kappa	0.65
Error medio absoluto	0.02
Raíz del error cuadrático	0.10
Error absoluto relativo	34.76%
Número total de instancias	40

Tabla 3.4: Estadísticas del árbol de clasificación

Como se evidencia en la tabla anterior, de un total de 40 instancias, el 67.5% fueron clasificadas correctamente, no siendo así para el 32.5% restante. Estos errores cometidos, pueden estar dados por la falta de entrenamiento que tenga la BC, resultado que mejorará con una mayor cantidad de casos, o definiendo un aprendizaje automático capaz de deducir un conjunto de casos abstractos que permita a la BC alcanzar un mayor nivel de conocimiento.

3.4 Validación del Motor de Inferencia

Partiendo de la BC confeccionada y evaluada por el criterio de experto, se realizará el ciclo del MI teniendo en cuenta todas las fases propuestas en el capítulo anterior. Los casos escogidos para el conjunto de prueba (tabla 3.5) son también casos validados por el experto, para una vez finalizado el procesamiento del MI poder realizar comparaciones. Los grupos para la validación de este parte del motor de inferencia estarán distribuidos como se muestra en la figura 3.3.

Nuevo Caso	Tiempo estimado por el experto
C1	19meses
C2	18meses
C3	10meses
C4	10meses
C5	22meses

Tabla 3.5: Clasificación del conjunto de prueba

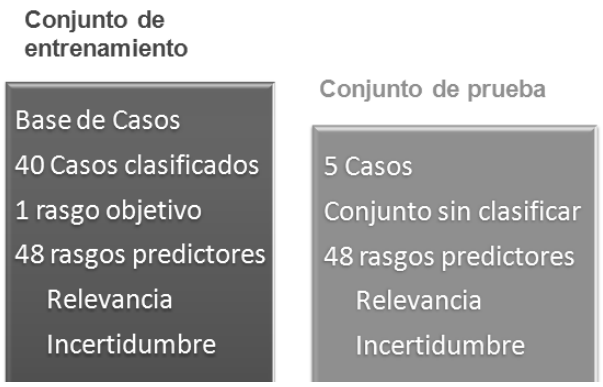

Figura 3.3: Grupos distribuidos para la validación del MI

3.4.1 Validación del modelo de recuperación

Para la creación del árbol en el que se hará la inferencia (Figura 3.4) se utilizó la herramienta RapidMiner obteniendo una representación gráfica que facilita la compresión

del modelo creado con el algoritmo C4.5. Las hojas están representadas por nodos con colores, cada color representa una clase distinta que se define con ese nodo.

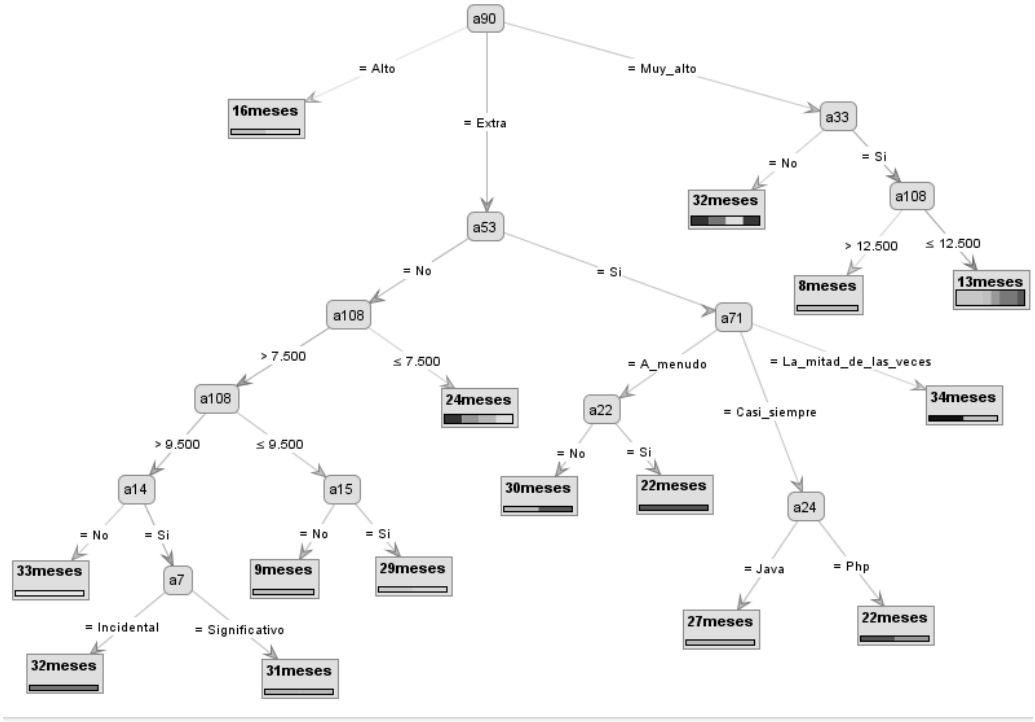

Figura 3.4: Árbol construido por el algoritmo C4.5 con pos-poda en RapidMiner

Luego de construido el árbol se pasará a la clasificación de los 5 casos que no tienen definido ninguna clase (tabla 3.6). Los valores de incertidumbre que se señalan en rojo fueron calculados mediante el comportamiento más probable (fórmula 6) que tiene la incertidumbre de su respectivo rasgo para cada caso en específico. Este fue el método propuesto para cuando se desconoce la incertidumbre.

C	P	Caso 1	I_{c1}	Caso 2	I_{c2}	Caso 3	I_{c3}	Caso 4	I_{c4}	Caso 5	I_{c5}
a7	7	Moderado	0,5	Moderado	0.6	Incidental	0.6	Medio	0.6	Medio	0.6
a8	6	Moderado	0,5	Moderado	0.5	Sin_influencia	0.9	Moderado	0.9	Moderado	0.5
a9	8	Moderado	0.6	Esencial	0.9	Moderado	0.8	Incidental	0.8	Esencial	0.6
a14	6	No	0.5	Si	0.5	No	0.5	Si	0.5	Si	0.7
a15	7	No	0.6	No	0.8	Si	0.6	Si	0.6	Si	1
a17	7	Si	0,6	No	0.5	Si	0.5	Si	0.5	No	0.7
a21	7	Si	0.6	Si	0.8	Si	0.6	Si	0.7	Si	0.9

a22	8	No	0.9	Si	0.5	No	0.9	Si	0.5	Si	0.5
a24	6	Java	0.9	Php	0.6	C++	0.9	Php	0.6	Java	0.7
a25	6	No	0.6	No	0.6	No	0.6	Si	1	No	0.6
a33	2	No	0.8	Si	0.7	Si	0.5	Si	0.5	Si	0.5
a34	4	No	0.9	Si	0.6	No	0.9	Si	0.6	No	0.6
a36	4	Eclipse	0.5	visual_estudio	0.5	Eclipse	0.9	visual_estudio	0.5	Eclipse	0.7
a40	5	32	0.6	27	0.5	32	0.8	36	0.6	30	0.8
a43	5	Medio_alto	0.4	Alto	0.5	Medio_bajo	0.9	Bajo	0.6	Alto	0.5
a44	**6**	Medio_alto	0.7	Alto	0.5	Medio	0.6	Medio_bajo	0.5	Medio	0.7
a45	**6**	Medio	0.9	Bajo	0.5	Bajo	0.9	Bajo	0.6	Medio	1
a53	7	Si	0.9	No	0.5	No	0.9	Si	0.5	Si	0.6
a54	**8**	General	0.5	Ninguna	0.5	Algo	1	Ninguna	0.5	General	0.6
a55	**8**	Algo	0.6	Algo	0.6	Algo	0.9	General	0.8	Algo	1
a56	7	Alto	0.7	Extra	0.9	Muy_bajo	0.6	Bajo	1	Alto	0.5
a59	**8**	Poco	1	Extremo	0.8	Considerable	0.6	Extremo	0.7	Significativo	0.9
a64	**8**	Ninguna	0.9	Basica	0.5	Vasto	0.5	Vasto	0.5	Ninguna	0.8
a66	8	Casi_siempre	0.9	No_se_conoce	0.5	No_se_conoce	0.5	Casi_siempre	0.6	Casi_siempre	0.8
a67	**6**	Casi_siempre	0.5	No_se_aplica	0.8	La_mitad_de_las_veces	0.5	La_mitad_de_las_veces	0.8	La_mitad_de_las_veces	0.5
a70	**7**	A_menudo	0.7	Casi_siempre	0.9	A_menudo	0.6	Casi_siempre	0.8	A_menudo	0.6
a71	7	La_mitad_de_las_veces	0.8	Casi_siempre	0.7	La_mitad_de_las_veces	0.5	Casi_siempre	1	Casi_siempre	1
a74	**4**	Ocasionalmente	0.5	Ocasionalmente	0.5	No_se_conoce	1	Casi_siempre	0.8	Ocasionalmente	0.6
a75	**6**	Casi_siempre	0.6	A_menudo	0.6	A_menudo	1	A_menudo	0.8	Ocasionalmente	0.5
a86	7	Alguna	0.6	Alguna	0.9	Considerable	0.6	Alguna	0.8	Alguna	0.6
a88	**6**	Basica	0.6	Total	0.7	Total	0.6	Considerable	0.8	Considerable	0.5
a90	**6**	Muy_alto	0.7	Extra	0.9	Muy_alto	0.6	Extra	0.8	Extra	0.6

a91	8	Alto	0.5	Normal	0.7	Alto	0.6	Muy_alto	0.8	Normal	0.5
a92	**9**	Bajo	0.9	Bajo	0.6	Alto	1	Extra	0.8	Bajo	0.6
a93	**9**	Alto	0.8	Muy_alto	0.6	Muy_bajo	0.7	Muy_alto	0.7	Bajo	0.9
a95	9	Alto	1	Normal	0.8	Alto	0.6	Muy_alto	0.7	Normal	0.6
a96	**9**	Normal	0.8	Normal	0.6	Alto	0.7	Alto	0.9	Bajo	0.6
a97	**7**	Alto	0.8	Extra	0.6	Normal	0.8	Alto	0.6	Normal	0.7
a98	6	Normal	0.6	Alto	0.7	Bajo	0.6	Muy_bajo	0.5	Muy_alto	1
a100	**7**	Muy_alto	0.8	Nominal	0.6	Muy_alto	0.7	Muy_alto	0.8	Muy_alto	0.6
a101	**9**	Nominal	0.7	Bajo	1	Nominal	0.6	Nominal	0.6	Bajo	0.8
a102	9	Alto	0.6	Muy_bajo	0.7	Normal	0.9	Muy_bajo	0.6	Bajo	0.5
a103	**9**	Normal	0.7	Alto	0.9	Normal	0.6	Normal	0.8	Normal	0.6
a104	**9**	Normal	0.9	Muy_bajo	0.6	Muy_alto	1	Alto	0.6	Normal	0.5
a107	9	Bajo	0.6	Alto	0.7	Alto	0.6	Alto	0.8	Bajo	0.5
a108	**6**	7	0.8	5	0.7	7	0.6	13	0.8	5	0.6
a109	**5**	25	0.6	22	0.6	25	0.8	23	0.7	25	0.5
a111	**6**	Empotrados	0.7	Multimedia	0.8	Gestion	0.7	Gestion	0.8	Multimedia	0.5

Tabla 3.6: Casos sin clasificar

En la figura 3.5 se muestra el recorrido que hace el árbol en la búsqueda de los casos semejantes a cada uno de los casos sin clasificar. Los caminos recorridos están representados por un color definido para cada caso en específico (Ver tabla 3.6).

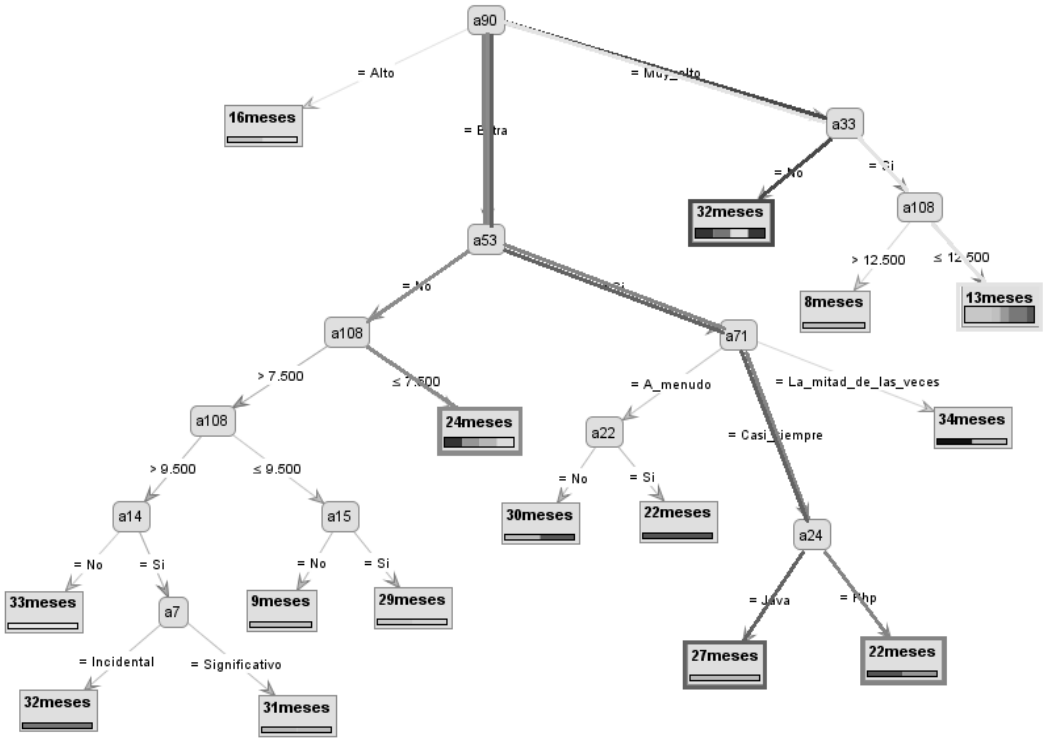

Figura 3.5: Recorrido de cada clasificación

Como resultado de la obtención de los casos más semejantes encontrados al realizar la búsqueda en el árbol de decisión se obtuvo la tabla 3.7.

Casos sin clasificar	Casos obtenidos
Caso1	C7 - 32meses
	C11 - 24meses
	C20 - 39meses
	C40 - 20meses
Caso2	C2 - 24meses
	C15 - 25meses
	C21 - 28meses
	C25 - 14meses
Caso4	C3 – 11meses
	C6 – 22meses

Casos sin clasificar	Casos obtenidos
Caso3	C9 – 13 meses
	C16 – 12meses
	C22 – 13meses
	C29 – 35meses
	C30 – 17meses
	C33 – 13meses
	C37 - 17meses
	C39 – 18meses
Caso5	C14 – 27meses
	C38 – 27meses

Tabla 3.7: Resultados de la clasificación para los casos sin clasificar

Luego de obtenido los casos recuperados mediante la inferencia en el árbol de decisión, es necesario calcular el grado de similitud entre el caso sin clasificación y los casos recuperados, resultado que arroja la función de semejanza definida por la fórmula 8. Para

ello primero se calcula la semejanza entre los rasgos, utilizando las funciones establecida en dependencia del dominio del rasgo.

El resultado que se obtuvo se visualiza en la tabla 3.8:

Semejanza		Resultado
C1	c7	0.51246659
	c11	0.46090815
	c20	0.50662705
	c40	0.53363104
C2	c2	0.50807301
	c15	0.51728361
	c21	0.51524502
	c25	0.57120341
C3	c9	0.52481594
	c16	0.51428955
	c22	0.47209539
	c29	0.51068598
	c30	0.44452318
	c33	0.55753671
	c37	0.43655764
	c39	0.54012587
C4	c3	0.55502386
	c6	0.54775496
C5	c14	0.4136086
	c38	0.53979036

Tabla 3.8: Grado de semejanza entre los casos sin clasificación y los respectivos casos recuperados

El valor umbral que acota las soluciones y asegura pasar a la fase de adaptación los casos más semejantes fue definido por el experto ya que calcularlo sería bastante engorroso para esta validación que se realiza, pues no se cuenta con un sistema informático que facilite esta tarea. Por el origen de los datos que componen la BC el umbral propuesto por el experto es de 0.5, lo que se traduce en decantar como posibles casos para la adaptación a todos los grados de similitud por debajo de este valor. De los casos recuperados: para el C2 y C4 no se elimina ninguno; para el C1 se elimina el c11; para C3 se elimina c22, c30 y c37; y para C5 se elimina c14. El resto se analizarán en la próxima fase.

3.4.2 Validación del modelo de adaptación

Como entrada a este modelo se reciben los casos recuperados que estuvieron por encima del valor umbral. Para cada juego de casos se calculó la utilidad esperada (fórmula 17) partiendo primeramente del cálculo del parámetro de medida α (fórmula 18). Los resultados obtenidos se muestran en la tabla 3.9.

Semejanza		Incertidumbre del Caso Recuperado	Semejanza	α	Utilidad Esperada
C1	7	0.6	0.51246659	0.41666667	0.56352775
	20	0.5	0.50662705		0.50276127
	40	0.9	0.53363104		**0.74734627**
C2	2	0.6	0.50807301	0.52083333	0.55212136
	15	0.2	0.51728361		0.36525188
	21	0.4	0.51524502		0.46002345
	25	0.9	0.57120341		**0.72875178**
C3	9	0.9	0.52481594	0.52083333	**0.70459163**
	16	0.8	0.51428955		0.65119247
	29	0.5	0.51068598		0.50556561
	33	0.8	0.55753671		0.67371704
	39	0.7	0.54012587		0.61673222
C4	3	0.8	0.55502386	0.375	**0.70813395**
	6	0.5	0.54775496		0.51790811
C5	14	0.8	0.4136086	0.60416667	0.5665552
	38	0.6	0.53979036		**0.56362334**

Tabla 3.9: Utilidad esperada de los casos *c*.

Luego de haber obtenido el valor de la utilidad, el caso seleccionado para realizar la adaptación y con ello dar respuesta al problema planteado en la descripción del nuevo caso, será el que mayor utilidad tenga. Los resultados fueron resaltados en la tabla anterior.

Quedando definida la clase a la que pertenecen cada uno de los casos que inicialmente no tenían clasificación, es necesario calcular la incertidumbre del nuevo caso que queda conformado utilizando la fórmula 16 propuesta anteriormente:
Los resultados se visualizan en la tabla 3.10.

C	c	Incertidumbre de c	Semejanza	I(C)
C1	40	0.9	0.53363104	0.5
C2	25	0.9	0.57120341	0.5
C3	9	0.8	0.51524502	0.4
C4	3	0.8	0.54775496	0.4
C5	38	0.6	0.53979036	0.3

Tabla 3.10: Incertidumbre del nuevo caso

Como se puede apreciar los resultados de la incertidumbre que están en función de la semejanza y la incertidumbre del caso recuperado, es un índice de certeza por debajo del caso con el que se compara. Resultado coherente teniendo en cuenta que la semejanza entre estos casos está por debajo del valor máximo, esta diferencia se refleja en el valor de incertidumbre.

Como resultado se obtiene los casos completados con la incertidumbre y la clase a la que pertenecen mostrados en la tabla 3.11.

Nuevo Caso	Tiempo	I(C)
C1	20meses	0.5
C2	14meses	0.5
C3	13meses	0.4
C4	11meses	0.4
C5	27meses	0.3

Tabla 3.11: Nuevos casos

En una comparación que se realizó entre el tiempo estimado por el experto y los resultados que se obtenidos luego de aplicar los modelos de recuperación y adaptación de los casos definidos como no clasificados, se generó el gráfico representativo que se muestra en la figura 3.6:

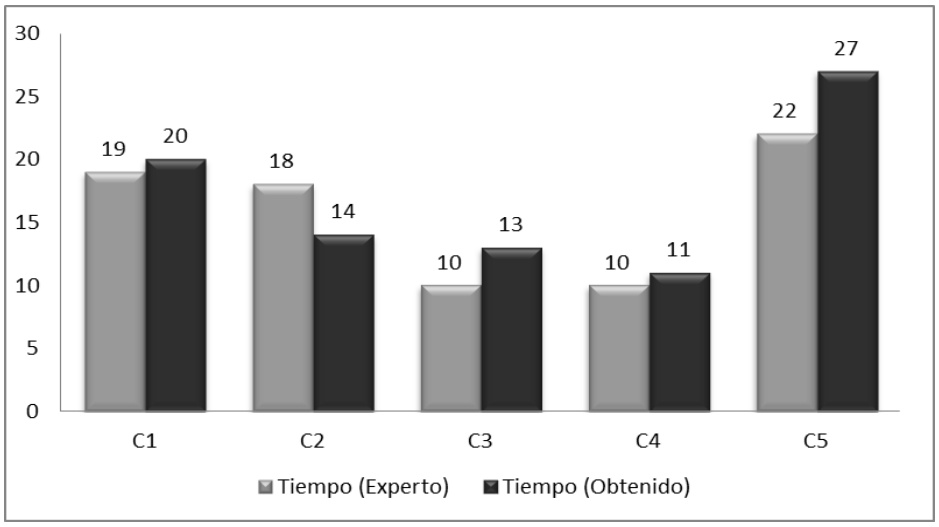

Figura 3.6: Comparación de los resultados

Como se puede evidenciar los resultados obtenidos están bastante cercanos al tiempo definido por el experto, con un margen de error relativo de 0.1, -0.2, 0.3, 0.1 y 0.2 respectivamente acotados en un intervalo de [-0.2; 0.3]. Por lo que se puede afirmar que los resultados arrojan niveles bastante similares de precisión.

Conclusiones parciales

En este capítulo se analizaron los resultados tras las validación realizada a la propuesta del MI utilizando como herramientas de apoyo para la visualización de los árboles de decisión RapidMiner. Tras la clasificación de una BC supervisada por el criterio de experto se mostraron los resultados usando para la clasificación el algoritmo C4.5 obteniendo un 67.5% de instancias clasificadas correctamente. También se validaron las diferentes fases del MI descrito en el capítulo anterior arrojando como resultados niveles elevados de similitud para los resultados obtenidos con respecto a la clasificación previa hecha por el experto. Esto permite deducir que si se cuenta que si contando con una BC de pequeño tamaño se obtuvieron índices aceptables, por hipótesis se puede inducir, que con una BC más grande, los resultados serán mejores.

Conclusiones Generales

Con la culminación de esta investigación se propuso un motor de inferencia para obtener información de los datos almacenados en una base de casos donde se registran los indicadores necesarios para estimar la duración de proyectos de desarrollo de software. Tras analizar los componentes de los sistemas basados en conocimiento, se corroboró que los SRBC son la solución más adecuada para representar esta información.

Se detallaron los módulos que componen dicho motor que permitió seleccionar los mecanismos de trabajo que mejor se ajustan a la solución que se quiere brindar a partir de fundamentos matemáticos. En el módulo de recuperación se escogió como técnica de acceso la jerárquica, y como algoritmo de recuperación, los árboles de decisión específicamente el algoritmo C4.5 porque permite evaluar todos los posibles valores de dominio que pueden tener los rasgos asociados a la investigación. La función de semejanza quedó definida utilizando diferentes funciones de similitud en dependencia de los tipos de datos de los rasgos y se modificó para que soportara el tratamiento de la incertidumbre. En el módulo de adaptación se definió a la adaptación nula, escogiendo como criterio de selección al caso de mayor utilidad esperada.

Luego de definida la propuesta de solución se realizó un conjunto de pruebas al algoritmo de recuperación creando una BC supervisada por el criterio de experto donde se obtuvo de un total de 40 instancias evaluadas, 67.5% de instancias clasificadas correctamente, arrojando resultados con un alto nivel de similitud. Concluyendo que el MI propuesto permite la obtención de resultados sobre un error admisible en el intervalo de [-0.2; 0.3] para una BC de solo 40 rasgos. Resultado que podrá ser mejorado si se entrena la BC con un algoritmo de aprendizaje automático que permita al sistema auto extenderse.

Recomendaciones

Se recomienda:

- Implementar la propuesta del motor de inferencia realizada.
- Definir el mecanismo de aprendizaje automático que permitirá a la BC auto extenderse.

Referencias Bibliográficas

1. **Palacio, Juan.** *Origen de la Gestión de Proyectos.* 2006.

2. **PMI - Project Management Intitute.** *Una Guía a los Fundamentos de la Dirección de Proyectos.* s.l. : Project Management Intitute, Inc, 2004.

3. **Proyecto, Lider de.** Lider de Proyecto. [En línea] 2010. http://liderdeproyecto.com/manual/que_es_el_pmbok.html..

4. **Navarro, A.** *Planificación de Proyectos de Software.* 2005.

5. **Cerrillo, David.** *Planificación y Gestión de Sistemas de información. 1999.* 1999.

6. **Ovejero, Daniel. Jose.** *Estimación de Proyectos Para Sistemas Basados en Conocimiento.* Buenos Aires : s.n., 2006.

7. **Real Academia de la Lengua Española.** Real Academia de la Lengua Española. [En línea] www.rae.es.

8. **Ruíz, Yadira.** *Elementos a considerar en la integración de los métodos de Boehm y Humphrey para la estimación en la duración de un proyecto de software.* 2011.

9. **Giraldo, Otoniel Perez.** *Métricas, Estimación y Planificación en Proyectos de Software.* 2004.

10. **Gálvez, Daniel.** *Curso de Sistemas Basados en el Conocimiento.* 1998.

11. **Cordero, Dasiel y Torres, Yoanny.** *Sistema Inteligente de Mitigación de Riesgos para el Centro ISEC.* 2011.

12. **Febles, Dr. Juan P.** *La inteligencia artificial en la enseñanza de la medicina.* 2000.

13. **Bergmann, R.** *Developing Industrial Case Based Reasoning Applications. The INRECA Methodology.* Berlín, Alemania : s.n., 1999.

14. **Turban, E.** *Decision Support Systems and Intelligent Systems.* New Jersey, Estados Unidos. : s.n., 1998. Quinta edición.

15. *Un Sistema Basado en Casos para la toma de desiciones en condiciones de incertidumbre.* **Gutiérrez, Iliana, Bello, Rafael E. y Tellería, Andrés.** 2, Santa Clara, Cuba : s.n., 2002, Vol. 23.

16. **Lozano, Laura y Fernández, Javier.** *Razonamiento Basado en Casos: Una Visión General.* 2005.

17. **García Martínez, Ramón y López Nocera, Marcelo Uvaldo .** *Descubrimiento de conocimiento basado en la integracón de algoritmos de explotación de la información.* Buenos Aires : s.n., 2009.

18. **Schulz, Gastón.** *Un ambiente integrado de clasificación, selección y ponderación de reglas basado en sistemas inteligentes.* Buenos Aires : s.n., 2007.

19. **Quinlan, J.R.** *Programs for machine learning. Morgan Kaufmann publishers.* San Mateo, California, EE.UU : s.n., 1993.

20. **Moro, Dr. Quiliano Isaac y Hurtado, Dra. Aránzazu Simón .** *Introducción al Diseño de Experimentos para el Reconocimiento de Patrones. Inducción de árboles de decisión.* 2004.

21. **Bello, Rafael.** *Solución de problemas bajo incertidumbre.* 1998.

22. **Martínez, MSc. Natalia, y otros, y otros.** *El Razonamiento Basado en Casos en el ámbito de la Enseñanza/Aprendizaje.* 2008.

23. **Han , Jiawei y Kamber, Micheline.** *Data mining: Concepts and Techniques.* 2006. Second Edition.

24. *Modelo para diseñar sistemas de enseñanza-aprendizaje inteligentes utilizando un razonamiento basado en casos.* **Martínez Sánchez, Dra. Natalia, García Lorenzo, Dra. Maria Matilde y García Valdivia , Dra. Zenaida.** 2009.

25. **Martínez, Natalia, García, María M y Zenaid, Zoila.** *El Paradigma del Razonamiento Basado en Casos en el Ámbito de los Sistemas de Enseñanza/Aprendizaje Inteligentes.* Santa Clara, Cuba : s.n., 2009. No. 30.

26. *Sistemas de razonamiento basado en casos aplicado a sistemas de líneas de productos software.* **Cortez Vásquez, Augusto , Navarro Depaz, Carlos y Pariona Quispe, Jaime .** 43-48, 2010.

27. **RapidMiner.** [En línea] [Citado el: 8 de 6 de 2012.] http://rapid-i.com.

Bibliografía

Cerrillo, David. *Planificación y Gestión de Sistemas de información. 1999.* 1999.

Gálvez, Daniel. *Curso de Sistemas Basados en el Conocimiento.* 1998.

Giraldo, Otoniel Perez. *Métricas, Estimación y Planificación en Proyectos de Software.* 2004.

Gómez, Fernando.*Aplicación para la estimación software basada en el modelo SLIM.* Madrid : s.n., 2004.

Gyepro®. *Grupo de Investigación en Gestión y Evaluación de Programas y Proyectos.* 2005.

Han , Jiawei y Kamber, Micheline. *Data mining: Concepts and Techniques.* 2006. Second Edition.

Jorgensen, M y S. *A systematic review of software development cost estimation studies.* s.l. : IEEE Transactions on Software Engineering 33(No 1), 2007.

Martínez, Natalia, García, María M y Zenaid, Zoila. *El Paradigma del Razonamiento Basado en Casos en el Ámbito de los Sistemas de Enseñanza/Aprendizaje Inteligentes.* Santa Clara, Cuba : s.n., 2009. No. 30.

Navarro, A.*Planificación de Proyectos de Software.* 2005.

Ovejero, José Daniel. *Estimación de Proyectos para Sistemas Basados en Conocimiento.* 2006.

Palacio, Juan. *Origen de la Gestión de Proyectos.* 2006.

PMI - Project Management Intitute. *Una Guía a los Fundamentos de la Dirección de Proyectos.* s.l. : Project Management Intitute, Inc, 2004.

Pow, Jose Antonio and Portillo, Sang. *Estudio de Técnicas basadas en Puntos de Función para la Estimación en Proyectos de Software.* 2002.

Pressman, R.S. *Ingeniería de Software. Un enfoque práctico.* 2002. 5ta edición.

Proyectics. *Los cambios en las Áreas de conociemiento se enfocan en nuevos procesos y un mejor alineamiento.* 2004.

Robiolo, M. G.*Transacciones, Objetos de Entidad y Caminos: métricas de software basadas en casos de uso, que mejoran la estimación temprana de esfuerzo.* 2009.

Solano, Alcides A., Yong, Gustavo A. y Camacho, Andrés S. *Introducción a los Lenguajes de Cuarta Generación (4GL).* 2007.

Fuente, Antonio de la.*COCOMO v2. Modelo de Estimación de Costes para proyectos software.* 1999.

Ogáyar, Diego Jódar y Casals Castells, Anna. *Project Manager Handbook. Manual del Jefe de Proyectos.* 2004.

Ordieres Meré, J.B, Torralba Martínez, J. Mª y Chiner Dasi, M. *Estimación del presupuesto del proyecto de software.* 2004.

Proyecto, Lider de. Lider de Proyecto. [En línea] 2010. http://liderdeproyecto.com/manual/que_es_el_pmbok.html..

Cuadrado, J., Sicilia, Miguel Angel y Garre, Miguel. *Segmentación Recursiva de Proyectos Software para la Estimación del Esfuerzo de Desarrollo de Software.* 2008.

King, D. *Project Management Made Simple. A guide to successful of computer Systems projects.* 1992.

Iglesias, Carlos A. *Arquitectura de los Sistemas Basados en Conocimiento.* 1999.

Bergmann, R. *Developing Industrial Case Based Reasoning Applications. The INRECA Methodology.* Berlín, Alemania : s.n., 1999.

Turban, E. *Decision Support Systems and Intelligent Systems.* New Jersey, Estados Unidos. : s.n., 1998. Quinta edición.

Ovejero, Daniel. Jose. *Estimación de Proyectos Para Sistemas Basados en Conocimiento.* Buenos Aires : s.n., 2006.

Bregón, Anibal, y otros, y otros. *Un sistema de razonamiento basado en casos para la clasificación de fallos en sistemas dinámicos.* 2005.

Morales, Mauricio. Líder de Proyecto. *Estructura del PMBOK®.* [En línea] 2009. [Citado el: 16 de 02 de 2012.] http://www.liderdeproyecto.com/manual/estructura_del_pmbok.html.

Febles, Dr. Juan P. *La inteligencia artificial en la enseñanza de la medicina.* 2000.

Un Sistema Basado en Casos para la toma de desiciones en condiciones de incertidumbre. **Gutiérrez, Iliana, Bello, Rafael E. y Tellería, Andrés.** 2, Santa Clara, Cuba : s.n., 2002, Vol. 23.

Lozano, Laura y Fernández, Javier. *Razonamiento Basado en Casos: Una Visión General.* 2005.

Bello, Rafael. *Solución de problemas bajo incertidumbre.* 1998.

Instituto Superior Politécnico José Antonio Echeverría. *Razonamiento basado en casos . Información Tecnológica.* 4, 2003, Vol. 14.

Molina López, José Manuel y García Herrero, Jesús. *Técnicas de análisis de datos.* Madrid : s.n., 2006.

García Martínez, Ramón y López Nocera, Marcelo Uvaldo. *Descubrimiento de conocimiento basado en la integracón de algoritmos de explotación de la información.* Buenos Aires : s.n., 2009.

Schulz, Gastón. *Un ambiente integrado de clasificación, selección y ponderación de reglas basado en sistemas inteligentes.* Buenos Aires : s.n., 2007.

Ruíz, Yadira. *Elementos a considerar en la integración de los métodos de Boehm y Humphrey para la estimación en la duración de un proyecto de software.* 2011.

Cordero, Dasiel y Torres, Yoanny. *Sistema Inteligente de Mitigación de Riesgos para el Centro ISEC.* 2011.

Quinlan, J.R. *Programs for machine learning. Morgan Kaufmann publishers.* San Mateo, California, EE.UU : s.n., 1993.

Anexos

Anexo 1: Representación simplificada para cada uno de los rasgos

Categorías	Indicadores de Tamaño	Código
Factores de complejidad técnica	Mecanismos de recuperación y back-up confiables	a1
	Comunicación de Datos	a2
	Funciones de Procesamiento Distribuido	a3
	Configuración usada rigurosamente	a4
	Entrada de datos on-line	a5
	Actualización de archivos on-line	a6
	Interfaces Complejas	a7
	Procesamiento Interno Complejo	a8
	Reusabilidad	a9
	Facilidad de cambios y amigabilidad	a10
	Tiempo de respuesta	a11
	Eficiencia por el usuario	a12
	Portabilidad	a13
	Concurrencia	a14
	Objetivos especiales de seguridad	a15
Factor Cliente	Existencia de un sistema anterior.	a16
	Existencia de infraestructura tecnológica en la organización.	a17
	Existencia de una base legal en la organización.	a18
	Estructura clara en la organización.	a19
	Disponibilidad de un especialista para atender al proyecto	a20
	Definidas las funciones de las áreas.	a21
	Estabilidad de los requisitos.	a22
	Nacionalidad del cliente.	a23
Líneas de código	Lenguaje de implementación	a24
	Uso de frameworks	a25
	Porcentaje del diseño del software que requiere modificación para alcanzarlos objetivos del nuevo software a desarrollar	a26
	Porcentaje del código del software que requiere modificación para lograr los objetivos del nuevo software a desarrollar.	a27
	Porcentaje del esfuerzo requerido para integrar y testear el software adaptado al producto global.	a28
	Grado de Evaluación y Asimilación.	a29
	Nivel de familiaridad del programador con el software.	a30
Reingeniería y conversión	Porcentaje de código que sufre un proceso de reingeniería mediante el uso de una herramienta de traducción automática	a31
	Tecnologías de traducción automática	a32
Recursos	Estilo Arquitectónico	a33
	Patrón de Arquitectura	a34

	Herramienta de Modelado Visual	a35
	Herramienta de Programación	a36
	Herramienta Sistema Gestor Base de Datos	a37
	Herramientas utilizadas en la Planificación	a38
	Herramientas utilizadas en la Gestión de Configuración	a39
	Herramientas utilizadas en la Estimación	a40
	Cantidad de PC	a41
Categorías	**Indicadores de Esfuerzo**	**Código**
Factores de complejidad técnica	Rendimiento de la Aplicación.	a42
	Rendimiento de la Plataforma.	a43
	Seguridad.	a44
	Integración.	a45
	Asimilación de Sistemas.	a46
	Asimilación de Dispositivos.	a47
	Asimilación de Estándares.	a48
	Dominio del líder sobre gestión de proyectos.	a49
	Proyecto de continuidad.	a50
	Características del líder de proyecto.	a51
	Familiaridad con el modelo del proyecto usado.	a52
	Motivación.	a53
	Personal tiempo parcial.	a54
Arquitectura y resolución de riesgos	Planificación de la administración de riesgo, identificando todos los ítems de riesgo y estableciendo hitos de control para su solución por medio de la revisión del diseño del producto (PDR)	a55
		a56
	Cronograma, presupuesto e hitos internos especificados en el PDR, compatibles con el plan de administración de riesgo	a57
	Porcentaje del cronograma dedicado a la definición de la arquitectura de acuerdo a los objetivos generales del producto	a58
	Porcentaje de arquitecturas de software disponibles para el proyecto	a59
	Herramientas disponibles para resolver ítems de riesgo, desarrollando y verificando las especulaciones de arquitecturas	a60
	Nivel de incertidumbre en factores claves de la arquitectura: interface de usuario, hardware, tecnología, performance	a61
	Cantidad y grado de criticidad de ítems de riesgo	a62
Cohesión de equipo	Compatibilidad entre los objetivos y culturas de los integrantes del equipo	a63
	Habilidad y predisposición para conciliar objetivos	a64
	Buena comunicación entre los miembros del equipo	a65
	Experiencia en el trabajo en equipo	a66
	Visión compartida de objetivos y compromisos compartidos	a67
Madurez del proceso	Administración de Requerimientos	a68
	Planificación del Proyecto de Software	a69
	Seguimiento y supervisión del Proyecto de Software	a70

	Administración de Subcontratos	a71
	Aseguramiento de la Calidad	a72
	Administración de la Configuración	a73
	Definición del Proceso de Organización	a74
	Programa de Entrenamiento	a75
	Administración Integrada de Software	a76
	Ingeniería del Producto	a77
	Coordinación entre Grupos	a78
	Revisión por Pares	a79
	Administración Cuantitativa	a80
	Administración de la Calidad	a81
	Prevención de Defectos	a82
	Administración de las Tecnologías de Cambio	a83
	Administración de los Procesos de Cambio	a84
Factor de precedencia	Entendimiento organizacional de los objetivos del producto	a85
	Experiencia en el trabajo con software relacionado	a86
	Desarrollo concurrente de un nuevo hardware y procedimientos operacionales	a87
	Necesidad de innovación en el procesamiento de datos, arquitectura y algoritmos	a88
Factor de flexibilidad de desarrollo	Necesidad de conformar requerimientos pre-establecidos	a89
	Necesidad de conformar especificaciones externas de interface	a90
	Estímulo por terminación temprana	a91
Factores del producto	Confiabilidad requerida	a92
	Tamaño de la base de datos	a93
	Complejidad del control producto	a94
	Complejidad computacional del producto	a95
	Complejidad dependientes de los dispositivos del producto	a96
	Complejidad de administración de datos computacional del producto	a97
	Complejidad de administración de interfaz de usuario computacional del producto	a98
	Requerimientos de reusabilidad	a99
	Documentación acorde a las diferentes etapas del ciclo de vida	a100
Factores de la plataforma	Volatilidad de la plataforma	a101
	Restricción del almacenamiento principal	a102
	Restricción del tiempo de ejecución	a103
Factores del personal	Capacidad del analista	a104
	Capacidad del programador	a105
	Continuidad del personal	a106
	Experiencia en la aplicación	a107
	Experiencia en la plataforma	a108
	Experiencia en el lenguaje y las herramientas	a109

	Cantidad de especialistas	a110
	Cantidad de estudiantes	a111
Factores del proyecto	Cronograma requerido para el desarrollo	a112
	Tipo de proyecto	a113

Anexo 2: Rasgos con valores numéricos

Tamaño	
Líneas de Código	Cantidad de líneas de código nuevas
	Cantidad de líneas de código modificadas
Componentes reusables	Cantidad de líneas de código fuente del software existente usadas para desarrollar el nuevo producto
Recursos	Cantidad de PC
Factores del personal	Cantidad de especialistas
	Cantidad de estudiantes
Esfuerzo	
Factores de complejidad técnica	Dominio del líder sobre gestión de proyectos.
	Proyecto de continuidad.
	Características del líder de proyecto.
	Familiaridad con el modelo del proyecto usado.
	Motivación.
	Personal tiempo parcial.

Anexo 3: Rasgos ordinales con tres posibles valores

Tamaño		**1**	**2**	**3**
Factor de Complejidad Técnica	Tiempo de respuesta	Esencial	Medio	Irrelevante
	Eficiencia por el usuario	Esencial	Medio	Irrelevante
	Portabilidad	Esencial	Medio	Irrelevante
	Concurrencia	Esencial	Medio	Irrelevante
	Objetivos especiales de seguridad	Esencial	Medio	Irrelevante
Esfuerzo		**1**	**2**	**3**
Factor de precedencia	Entendimiento organizacional de los objetivos del producto	Total	Considerable	General
	Experiencia en el trabajo con software relacionado	Amplia	Considerable	Moderada
	Desarrollo concurrente de un nuevo hardware y procedimientos operacionales	Abundante	Moderado	Escaso
	Necesidad de innovación en el procesamiento de datos, arquitectura y algoritmos	Considerable	Alguna	Mínima
Factor de flexibilidad de desarrollo	Necesidad de conformar requerimientos pre-establecidos	Total	Considerable	Básica
	Necesidad de conformar especificaciones externas de interface	Total	Considerable	Básica

	Estímulo por terminación temprana	Elevado	Medio	Bajo
Factores del producto	Tamaño de la base de datos	Muy alto	alto	Normal

Anexo 4: Rasgos ordinales con cuatro posibles valores

Tamaño		1	2	3	4
Componentes reusables	Porcentaje del diseño del software que requiere modificación para alcanzarlos objetivos del nuevo software a desarrollar	Alto	Medio Alto	Medio	Bajo
	Porcentaje del código del software que requiere modificación para lograr los objetivos del nuevo software a desarrollar.	Alto	Medio Alto	Medio	Bajo
	Porcentaje del esfuerzo requerido para integrar y testear el software adaptado al producto global.	Alto	Medio Alto	Medio	Bajo
Reingeniería y conversión	Porcentaje de código que sufre un proceso de reingeniería mediante el uso de una herramienta de traducción automática	Alto	Medio Alto	Medio	Bajo
Esfuerzo		**1**	**2**	**3**	**4**
Factores de la plataforma	Volatilidad de la plataforma	Extra Alto	Muy alto	Alto	Nominal
	Restricción del almacenamiento principal	Extra Alto	Muy alto	Alto	Nominal
	Restricción del tiempo de ejecución	Muy alto	Alto	Nominal	Bajo

Anexo 5: Rasgos ordinales con cinco posibles valores

Tamaño		**1**	**2**	**3**	**4**	**5**
Factores de complejidad técnica	Rendimiento de la Aplicación.	Alto	Medio Alto	Medio	Medio Bajo	Bajo
	Rendimiento de la Plataforma.	Alto	Medio Alto	Medio	Medio Bajo	Bajo
	Seguridad.	Alto	Medio Alto	Medio	Medio Bajo	Bajo
	Integración.	Alto	Medio Alto	Medio	Medio Bajo	Bajo
	Asimilación de Sistemas.	Alto	Medio Alto	Medio	Medio Bajo	Bajo
	Asimilación de Dispositivos.	Alto	Medio Alto	Medio	Medio Bajo	Bajo
	Asimilación de Estándares.	Alto	Medio Alto	Medio	Medio Bajo	Bajo
Componentes reusables	Grado de Evaluación y Asimilación	**Ver Anexo 5.1**				
	Nivel de familiaridad del programador con el software	**Ver Anexo 5.2**				
Factores del producto	Requerimientos de reusabilidad	Extra	Muy alto	alto	Normal	Bajo

	Documentación acorde a las diferentes etapas del ciclo de vida	Muy alto	alto	Normal	Bajo	Muy Bajo
Factores del personal	Capacidad del analista	Muy alto	alto	Normal	Bajo	Muy Bajo
	Capacidad del programador	Muy alto	alto	Normal	Bajo	Muy Bajo
	Continuidad del personal	Muy alto	alto	Normal	Bajo	Muy Bajo
	Experiencia en la aplicación	Muy alto	alto	Normal	Bajo	Muy Bajo
	Experiencia en la plataforma	Muy alto	alto	Normal	Bajo	Muy Bajo
	Experiencia en el lenguaje y las herramientas	Muy alto	alto	Normal	Bajo	Muy Bajo
Factores del proyecto	Cronograma requerido para el desarrollo	Muy alto	alto	Normal	Bajo	Muy Bajo

Anexo 5.1: Dominio del indicador Grado de Evaluación y Asimilación

	Nivel de esfuerzo
Componentes reusables	Ninguno
	Búsqueda del módulo básico y documentación
	Algunos módulos de testeo y evaluación, documentación
	Considerable cantidad de módulos de testeo y evaluación, documentación
	Gran cantidad de módulos de testeo y evaluación, documentación

Anexo 5.2: Dominio del indicador Nivel de familiaridad del programador con el software

	Nivel de esfuerzo
Componentes reusables	Completamente familiar
	Familiar en su mayor parte
	Algo familiar
	Considerablemente familiar
	Desconocido en su mayor parte

Anexo 6: Rasgos ordinales con seis posibles valores

Tamaño	
Factor de Complejidad Técnica	Mecanismos de recuperación y back-up confiables
	Comunicación de Datos
	Funciones de Procesamiento Distribuido
	Configuración usada rigurosamente
	Entrada de datos on-line
	Actualización de archivos on-line
	Interfaces Complejas
	Procesamiento Interno Complejo

	Reusabilidad
	Facilidad de cambios y amigabilidad
Esfuerzo	
Arquitectura y resolución de riesgos	Planificación de la administración de riesgo, identificando todos los ítems de riesgo y estableciendo hitos de control para su solución por medio de la revisión del diseño del producto (PDR)
	Cronograma, presupuesto e hitos internos especificados en el PDR, compatibles con el plan de administración de riesgo
	Porcentaje del cronograma dedicado a la definición de la arquitectura de acuerdo a los objetivos generales del producto
	Porcentaje de arquitecturas de software disponibles para el proyecto
	Herramientas disponibles para resolver ítems de riesgo, desarrollando y verificando las especulaciones de arquitecturas
	Nivel de incertidumbre en factores claves de la arquitectura: interface de usuario, hardware, tecnología, performance
	Cantidad y grado de criticidad de ítems de riesgo
Cohesión de equipo	Compatibilidad entre los objetivos y culturas de los integrantes del equipo
	Habilidad y predisposición para conciliar objetivos
	Experiencia en el trabajo en equipo
	Visión compartida de objetivos y compromisos compartidos
Factores del producto	Confiabilidad requerida
	Complejidad del control producto
	Complejidad computacional del producto
	Complejidad dependientes de los dispositivos del producto
	Complejidad de administración de datos computacional del producto
	Complejidad de administración de interfaz de usuario computacional del producto

Anexo 6.1: Valores ordinales de las variables con 6 posibles valores.

1	2	3	4	5	6
Esencial	Significativo	Medio	Moderado	Incidental	Sin influencia

Anexo 7: Rasgos ordinales con siete posibles valores

Esfuerzo	
Madurez del proceso	Administración de Requerimientos
	Planificación del Proyecto de Software
	Seguimiento y supervisión del Proyecto de Software
	Administración de Subcontratos
	Aseguramiento de la Calidad
	Administración de la Configuración
	Definición del Proceso de Organización

	Programa de Entrenamiento
	Administración Integrada de Software
	Ingeniería del Producto
	Coordinación entre Grupos
	Revisión por Pares
	Administración Cuantitativa
	Administración de la Calidad
	Prevención de Defectos
	Administración de las Tecnologías de Cambio
	Administración de los Procesos de Cambio

Anexo 7.1: Valores ordinales de las variables con 7 posibles valores.

1	2	3	4	5	6	7
Casi siempre	A menudo	La mitad de las veces	Ocasionalmente	Casi nunca	No se aplica	No se conoce

Anexo 8: Rasgos con valores nominales

Tamaño	
Factor Cliente	Nacionalidad del cliente.
Reingeniería y conversión	Tecnologías de traducción automática
Recursos	Herramienta de Modelado Visual
	Herramienta de Programación
	Herramienta Sistema Gestor Base de Datos
	Herramientas utilizadas en la Planificación
	Herramientas utilizadas en la Gestión de Configuración
Líneas de código	Lenguaje de implementación

Anexo 9: Rasgos con valores binarios

Tamaño	
Factor Cliente	Existencia de un sistema anterior.
	Existencia de infraestructura tecnológica en la organización.
	Existencia de una base legal en la organización.
	Estructura clara en la organización.
	Disponibilidad de un especialista para atender al proyecto
	Definidas las funciones de las áreas.
	Estabilidad de los requisitos.
Líneas de Código	Uso de frameworks
Recursos	Estilo Arquitectónico
	Patrón de Arquitectura

Esfuerzo	
Factores de complejidad técnica	Dominio del líder sobre gestión de proyectos.
	Proyecto de continuidad.
	Características del líder de proyecto.
	Familiaridad con el modelo del proyecto usado.
	Motivación.
	Personal tiempo parcial.