Universidad de las Ciencias Informáticas
Facultad 6

Título: "Sistema Experto para el diagnóstico médico de las enfermedades genéticas con dismorfias (SEGEDIS)"

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

Autores: Marianela Gutiérrez Rodríguez

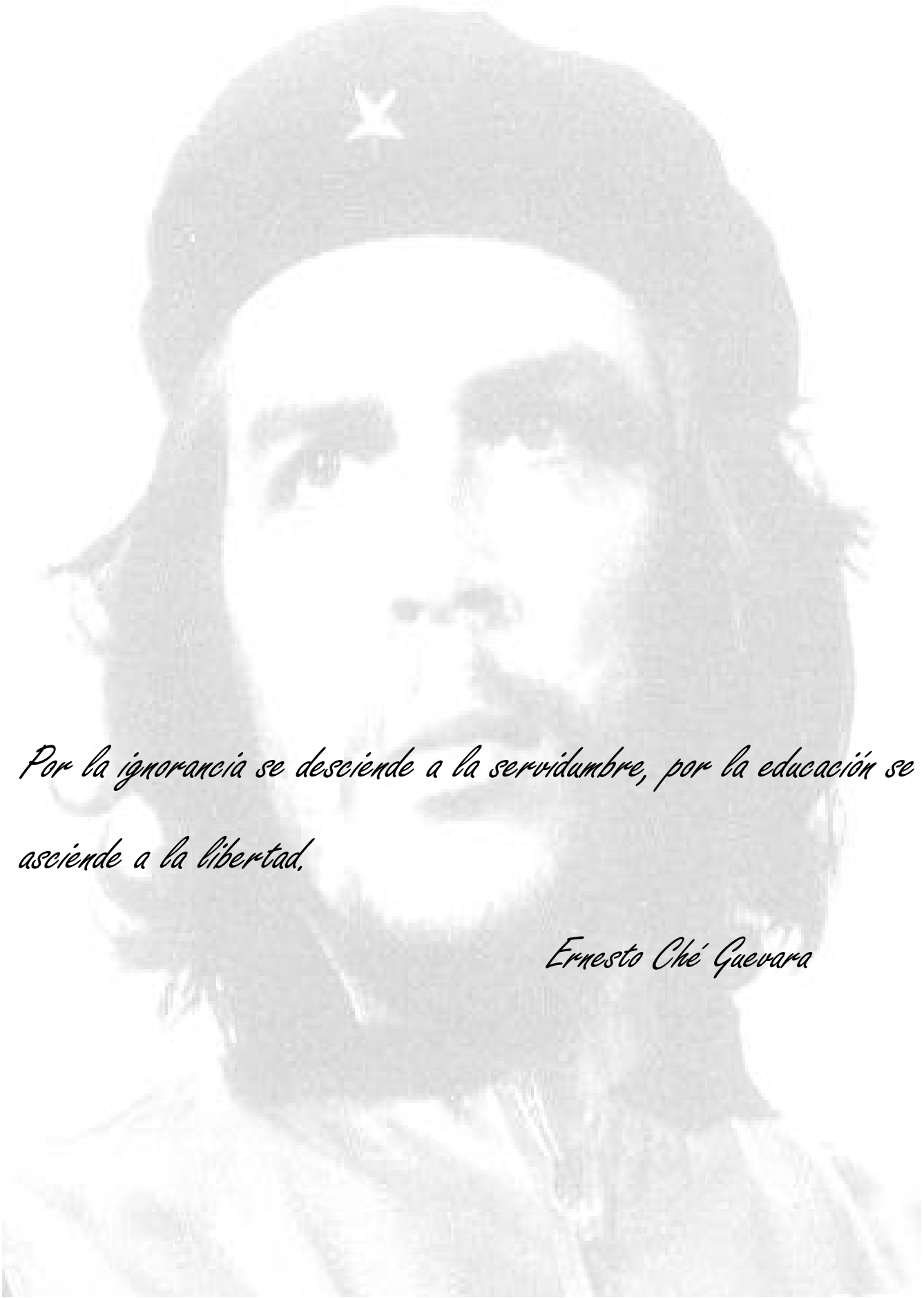
Jorge Bedoya Rusenko

Tutores: MsC. Maikel Yelandi Leyva Vázquez

Ing. Yadira Barroso Rodríguez

Consultante: Dra. Estela Morales Peralta

Ciudad de La Habana, Cuba 2010
"Año 52 de la Revolución"



Por la ignorancia se desciende a la servidumbre, por la educación se asciende a la libertad.

Ernesto Ché Guevara

DECLARACIÓN DE AUTORÍA

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

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

Marianela Gutiérrez Rodríguez

_____

Firma del autor

Jorge Bedoya Rusenko

_____

Firma del autor

Ing. Yadira Barroso Rodríguez

_____

Firma del tutor

MsC. Maikel Yelandi Leyva Vázquez

_____

Firma del tutor

DATOS DE CONTACTO

Tutor:
MsC. Maikel Yelandi Leyva Vázquez
Universidad de las Ciencias Informáticas, Habana, Cuba
Email: mleyvaz@uci.cu

Tutor:
Ing. Yadira Barroso Rodríguez
Universidad de las Ciencias Informáticas, Habana, Cuba
Email: ybarroso@uci.cu

AGRADECIMIENTOS

Son muchos los años que transitamos como estudiantes por diferentes niveles de enseñanzas donde cada uno nos roba un pedacito del corazón porque dejamos atrás profesores, amigos, compañeros, escuelas que contienen recuerdos por cada rincón; pero sin duda alguna este ha sido el lugar donde hemos vivido independientes, donde se nos ha puesto a prueba a cada momento haciéndonos crecer profesionalmente y lo más importante espiritualmente. Hoy nos vamos convertidos en hombres y mujeres de bien, listos para dar lo mejor de sí donde se nos asigne.

Por todo esto agradecemos a la Revolución que nos ha forjado bajo los ideales de Martí, Fidel y el Ché. Por forjarnos como profesionales útiles para retribuirle a la patria lo que ha hecho por nosotros. Por educarnos en el concepto martiano de que "El deber de un hombre está allí donde es más útil", formándonos para servir a nuestro pueblo y a otros pueblos hermanos que lo necesiten. Por habernos guiado por el camino de los que fundan y aman.

Agradecemos a nuestros padres que nos han sabido guiar para alcanzar un sueño que hoy se hace realidad. Gracias por siempre estar ahí para nosotros en cualquier circunstancia.

Agradecemos a los tutores Yadira Barroso Rodríguez y Maikel Yelandi Leyva por su paciencia, dedicación y optimismo.

A Yusdenis Sánchez por siempre estar ahí para nuestras dudas como otro tutor más.

A Asdrubal, Osmany, Francisco, Roberto, Félix, Andrés que nos aportaron sus conocimientos y su ayuda.

A nuestros amigos, compañeros de aula y apartamento y a todas aquellas personas que han hecho posible de una forma u otra que este sueño se hiciera realidad, gracias.

DEDICATORIA

De: Marianela Gutiérrez Rodríguez

Dedico este trabajo especialmente para mi papá y mi mamá. Desde muy chica me exigieron estar entre lo que más brillaba donde me encontrara, al principio pensaba que me exigían mucho, pero hoy comprendo que les debo todo, mi cuerpo, mi conocimiento y mi forma de ser. Gracias papitos por estar presentes hoy y vivir conmigo este sueño, que no es solo mío, es de ustedes también. A mi papito le dedico mis resultados académicos y ese título de oro que tanto querías, para ti papá luché para lograrlo y a mi mamá por su paciencia y dedicación por ser tan linda conmigo porque ante todo es mi amiga y consejera. También quisiera agradecerles a mis padres la decisión de no dejarme sola en el mundo y darme dos hermanas que amo mucho. Agradecerles a ellas Irma y Graciela por ser las amigas que crecieron conmigo y que compartí todas las etapas de mi vida. Pero quisiera dar un agradecimiento especial a mi hermanita Gracielita como siempre la llamo, porque ha seguido mis pasos. Ella se preocupa por no ser capaz de llegar a las metas de su hermana mayor pero hoy mi hermanita te digo que lo has hecho muy bien y que llegarás muy lejos, estoy orgullosa de ti, gracias por existir.

A mis abuelos por ser mis segundos padres, abuelita Pilar, Irma, abuelitos Pucho, Baldo y abuelito Nené que me observa desde el cielo y me da fuerzas para continuar.

A mis tíos, tías y primos que siempre están pendientes de mí y me han hecho gozar de una familia unida y cariñosa.

A mis amigas Haymel, Yadira, Mily, Danelis, Geydi, Yisel, Yailén, Mariela y Yaneysi por ser las personas que me han acompañado y que han hecho florecer lo mejor de mí. Pero en especial quisiera agradecerle a Haymel y Yadira que las conocí en octubre del 2005 y desde entonces hemos sido las mejores amigas para siempre. Gracias a las dos por estar ahí.

A mis amigas de la vocacional Leydis Hidalgo, Leydis Aguilera, Elizabeth y mis otras amigas que aunque no estén aquí como Nelia, Viviana y Yusimí que nunca se han olvidado de mí.

A mi segunda familia habanera; Manuel, Elvira y Raúl que me adoptaron como su segunda hija y me brindaron amor y cariño.

A mis amigos Asencio, Asdrubal, David, Pochy, Frank , Andy y Edgar.

A mi gran tutora que supo romper los límites del profesionalismo para convertirse en una amiga.

A mis suegros y mi cuñada linda por brindarme otra familia.

Y no por ser el último agradecimiento, el menos importante, a mi compañero de tesis, mi novio, mi amigo que supo entregarse en cuerpo y alma para este trabajo y para mí. Gracias al momento en el que te encontré, en que tus ojos verdes me cautivaron. Gracias por tu paciencia, por tu apoyo, por tus conocimientos y por sobre todo, TU AMOR.

De: Jorge Bedoya Rusenko

Dedico este trabajo a mi mamuchi, mi papuchi y mi hermanota. Nunca pensé que llegaría el momento de agradecer, porque estas palabras darían el punto final a este trabajo que me ha crecido como profesional. Siempre quise ser el orgullo de mis padres, todas mis acciones, mis reconocimientos se las dedicaba, pero hoy el mejor regalo que les puedo hacer es esta tesis que le da culminación a mi carrera universitaria. Gracias a mi familia por ser tan pacientes, dedicados y proporcionarme día a día el amor incondicional de la sangre.

A mi mamuchi, papuchi por cuidarse mutuamente mientras yo me forjaba para regalarles un sueño.

A mi tía Svieta y mi primo Andrei por el regalo que posibilitó la realización de esta tesis, gracias los quiero mucho.

A mi hermanota linda por considerarme además de un hermano, un hijo.

A mis suegros y mi cuñadita por ser mi segunda familia y por apoyarme en todas mis decisiones.

A mis amigos Asencio, Asdrubal, David, Daniel, Félix, Andrés, Pochy, Enrique, Silvio, Aparicio, Polivio, Edgar, Haymel, Yadira, Yailén, Mariela, Yaneysi, compañeros de aula y del apartamento, a la gente del Dota y a todos aquellos que me preguntaron en alguna ocasión como iba mi tesis.

Muchas personas piensan que las parejas no pueden llegar a ser compañeros de tesis pero hoy demuestro que si se puede. Nela más que una novia es mi amiga, mi compañera, ella supo combinar el cariño y el estudio dando por resultado este trabajo. Gracias por hacer feliz los momentos más difíciles, gracias por tu perseverancia, comprensión y por demostrar que no estaba equivocado, eres la mujer de mis sueños, la que tanto busqué y un 7 de marzo encontré.

RESUMEN

Las enfermedades genéticas, aunque en su conjunto son abundantes, individualmente son raras, lo que hace difícil su identificación y para los médicos en ocasiones no son suficientes sus conocimientos, la revisión de la literatura, ni el intercambio con otros especialistas. Por estos motivos se desarrolló la versión 1.0 del Sistema Experto de Enfermedades Genéticas con Dismorfias (SEGEDIS) con el objetivo de proporcionarles una herramienta a los genetistas cubanos en el apoyo a sus decisiones.

El presente trabajo contribuye al diagnóstico de pacientes que hoy cuentan con especialistas altamente calificados, pero por el gran número de enfermedades genéticas con dismorfias que existen en el mundo se hace difícil reconocer cada tipo de estas enfermedades que se les presentan en sus consultas. Además permite a los genetistas del Centro Nacional de Genética Médica (CNGM) que poseen los permisos de administración sobre la misma, agregar nuevas enfermedades, actualizando sistemáticamente la aplicación, y de esta forma brindar una mejor atención a los pacientes.

Palabras claves: dismorfias, genética, Sistema Experto.

ÍNDICE

INTRODUCCIÓN

En Cuba, uno de los principales objetivos a cumplir, es brindar una salud gratuita y eficiente al pueblo, a pesar de estar sometida a un bloqueo cruel y genocida se encuentra inmersa en una lucha constante por mejorar sus servicios. Por eso prioriza el desarrollo integral de soluciones informáticas para el Centro Nacional de Genética Médica, con el propósito de garantizar mayor calidad en la atención a pacientes. Muchas de ellas logradas con la colaboración de la Universidad de las Ciencias Informáticas, están en fase de implementación en este campo e incluyen, entre otras, los registros nacionales de discapacitados, enfermedades genéticas, malformaciones congénitas, gemelos, discapacidades mentales, discapacidades físicas y teleconsultas.

La Universidad de las Ciencias Informáticas (UCI) es una idea del Comandante en Jefe Fidel Castro Ruz que trabaja para convertirse en un centro de excelencia. En ella se desarrollan producciones de software para la salud y equipos médicos, educación, telecomunicaciones, bioinformática, entre otras; vinculadas con diferentes instituciones, como el CNGM, sede del Programa de la Revolución para el desarrollo de la genética médica en el país.

El 5 de agosto del 2003 fue inaugurado este centro por el Comandante en Jefe, quien señaló en el propio acto los objetivos de esta institución y su nueva concepción hacia la investigación. Dentro de sus principales funciones tiene las investigaciones básicas y aplicadas en el campo de la genética médica, la inmunología, la bioquímica y otras disciplinas afines dirigidas a la obtención de nuevos conocimientos, evaluación y desarrollo de nuevas tecnologías, productos y procedimientos de trabajo.[1]

En el CNGM se estudian las enfermedades genéticas con dismorfias, que aunque en su conjunto son abundantes, individualmente son raras, lo que hace difícil su identificación y para los médicos en ocasiones no son suficientes sus conocimientos, la revisión de la literatura, ni los medios que emplean para el intercambio con otros especialistas. Por estos motivos se desarrolla la versión 1.0 de la Teleconsulta genética con el objetivo de acelerar y automatizar los procesos que se llevan a cabo en las consultas de referencia nacional.

Esta versión actual solamente se limita al intercambio de opiniones entre los especialistas basándose en su experiencia. Además la evaluación de los genetistas no se hace del todo certera ya que en el mundo

existen aproximadamente 14 000 enfermedades genéticas y gran parte de ellas, son enfermedades genéticas con dismorfias y es imposible que un conjunto de doctores valorando a un paciente haya visto todas las enfermedades. Muchas de estas enfermedades tienen síntomas similares lo cual se hace engorroso un diagnóstico preciso solamente con la experiencia del genetista. El CNGM no dispone con una herramienta que recoja el conocimiento de varios especialistas y que esté disponible para trabajar simultánea y continuamente en un problema, a cualquier hora del día y de la noche. Una herramienta que sea permanente, a diferencia de los especialistas humanos, que pueden retirarse, renunciar o morir. Una herramienta que proporcione una segunda opinión que incremente la confianza en que un genetista ha tomado la decisión correcta.

Por la situación planteada con anterioridad surge **la interrogante**: ¿Cómo contribuir al diagnóstico médico de las enfermedades genéticas con dismorfias?

Con el desarrollo de este trabajo se pretende dar solución al problema antes citado. Para la obtención de estos resultados se plantea como **objeto de estudio**: Los Sistemas Expertos para el diagnóstico médico y como **campo de acción:** Los Sistemas Expertos para el diagnóstico médico de las enfermedades genéticas con dismorfias.

El **objetivo general** de este trabajo**:** Desarrollar un Sistema Experto para el diagnóstico médico de las enfermedades genéticas con dismorfias que permita lograr un acercamiento al dictamen más probable.

Como **objetivos específicos:**

- Realizar la adquisición del conocimiento al experto en enfermedades genéticas con dismorfias.
- Formalizar la representación del conocimiento adquirido.
- Implementar los componentes del SEGEDIS.
- Validar el SEGEDIS implementado.

Para poder cumplir estos objetivos y lograr una solución adecuada a la situación antes expuesta se plantean las siguientes **tareas**:

- Estudio de las tendencias actuales de los Sistemas Expertos para el diagnóstico médico.

- Estudio y selección de las metodologías, técnicas y herramientas necesarias para la construcción de un Sistema Experto.
- Realizar la captura del conocimiento al experto en enfermedades genéticas con dismorfias.
- Realizar la representación del conocimiento capturado.
- Implementación de los componentes del SEGEDIS.
- Validación del SEGEDIS implementado.

Posibles resultados: Prototipo del Sistema Experto para el diagnóstico médico de enfermedades genéticas con dismorfias.

El documento está estructurado de la siguiente manera: introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía, anexos y glosario de términos.

Capítulo 1: Fundamento teórico

En el capítulo se exponen varios conceptos referentes a la inteligencia artificial, los Sistemas Expertos, la arquitectura de estos y sus aplicaciones en la medicina. Brinda una descripción de los Sistemas Expertos en la genética médica, resaltando sus características. Ofrece además un breve estudio acerca de las tecnologías y sus características específicas, que permitan determinar las potencialidades que brindan para el desarrollo del SEGEDIS. Además se realiza una comparación de metodologías aplicables a la construcción de un sistema.

Capítulo 2: Solución propuesta

En este capítulo se pone en práctica la metodología para un Sistema Experto. Se identifica el problema a resolver, así como el equipo de desarrollo que dará progreso ha dicho trabajo. Expone varias técnicas de adquisición del conocimiento y las fuentes donde fue extraído el mismo. Brinda un ejemplo de la forma que se representa el conocimiento y muestra el mecanismo de búsqueda que se utiliza. Se representa la arquitectura del SEGEDIS y los prototipos de interfaz de usuario. Además se expone el estándar y los estilos de codificación que se tuvieron en cuenta en el desarrollo del trabajo, las pautas de diseño y algunos patrones que utiliza Symfony.

Capítulo 3: Resultados y discusión

En este capítulo se exponen los aspectos esenciales de la Validación y Verificación de los Sistemas Expertos. Se ejemplifican las validaciones que se les hace a la aplicación, mediante el framework Symfony. Se muestran algunos ejemplos de estas validaciones tanto en el código en php como en la misma interfaz. Se describen los atributos que componen la facilidad de uso, ilustrándolos en diferentes pruebas aplicadas a la interfaz gráfica del SEGEDIS.

CAPÍTULO 1: FUNDAMENTO TEÓRICO

Introducción

En el capítulo se exponen varios conceptos referentes a la inteligencia artificial, los Sistemas Expertos, la arquitectura de estos y sus aplicaciones en la medicina. Brinda una descripción de los Sistemas Expertos en la genética médica, resaltando sus características. Ofrece además un breve estudio acerca de las tecnologías y sus características específicas, que permitan determinar las potencialidades que brindan para el desarrollo del SEGEDIS. Además se realiza una comparación de metodologías aplicables a la construcción de un sistema.

1.1 Inteligencia Artificial

La inteligencia artificial es un término que ha generado muchas polémicas desde hace varias décadas, pero la mayoría de los autores coinciden en que es, en esencia, lograr que una máquina tenga inteligencia propia, es decir: “La inteligencia artificial es una de las áreas más fascinantes y con más retos de las ciencias de la computación ya que ha tomado a la inteligencia como la característica universalmente aceptada para diferenciar a los humanos de otras criaturas ya sean vivas o inanimadas, para construir programas o computadoras inteligentes.” [2]

Esta es una de las definiciones más completa que ilustra este término, pero no es la única, para otros autores, la Inteligencia Artificial es el estudio de cómo hacer que los ordenadores hagan cosas que, en estos momentos, hace mejor el hombre. [3] Para otros, la Inteligencia Artificial (IA) es una ciencia que intenta la creación de programas para máquinas que imiten el comportamiento y la compresión humana, que sea capaz de aprender, reconocer y pensar. [4]

Pero sea cual sea el concepto, la inteligencia artificial está orientada a conseguir que las máquinas realicen trabajos donde se aplique la inteligencia, el razonamiento y el conocimiento de un ser humano.

La inteligencia artificial posee técnicas que han sido explotadas en la solución de problemas de ayuda a la toma de decisiones en el diagnóstico y tratamiento de pacientes por sus potencialidades. Mediante estas técnicas lo que se pretende es emular la capacidad del ser humano al enfrentarse a una toma de decisión,

imitando tanto su aprendizaje como la manera de llegar a una decisión basándose en sus conocimientos; características que son las bases fundamentales para el diagnóstico y el tratamiento. Para cumplir con estos objetivos la IA hace uso de sus propias ramas, dentro de las cuales se pueden destacar los Sistemas Basados en el Conocimiento (SBC).

Un ***Sistema Basado en el Conocimiento*** es aquel en el que aparece representado el conocimiento de un dominio determinado, de tal forma que dicha representación sea procesable por un programa informático. Un SBC al que se le incorpora conocimiento proveniente de expertos en un dominio se le conoce como Sistema Experto (SE). [5]

Los términos S***istema Experto***, S***istema Basado en Conocimiento***, o ***Sistema Experto Basado en Conocimiento***, se usan como sinónimos. [6] La mayoría utiliza el término Sistema Experto porque es más corto, en la tesis que ocupa se utilizará el término Sistema Experto.

1.2 Sistema Experto

Se puede decir que los ***Sistemas Expertos*** son el primer resultado operacional de la Inteligencia artificial, pues logran resolver problemas a través del conocimiento y raciocinio de igual forma que lo hace el experto humano.

Un Sistema Experto, es básicamente un programa de computadora basado en conocimientos y raciocinio que lleva a cabo tareas que generalmente sólo realiza un experto humano; es decir, es un programa que imita el comportamiento humano en el sentido de que utiliza la información que le es proporcionada para poder dar una opinión sobre un tema en especial. Otros autores lo definen como sigue: un Sistema Experto es un programa de computadora interactivo que contiene la experiencia, conocimiento y habilidad propios de una persona o grupos de personas especialistas en un área particular del conocimiento humano, de manera que permitan resolver problemas específicos de esa área de forma inteligente y satisfactoria. La tarea principal de un SE es tratar de aconsejar al usuario. [6]

Los Sistemas Expertos son utilizados como asesores o consultores de usuarios humanos. Pueden ser utilizados para resolver problemas rutinarios, liberando a los expertos de esas pocas gratificantes tareas de modo que pueden utilizar su conocimiento en otros asuntos más complejos y novedosos. Los Sistemas

Expertos pueden llevar la experiencia a lugares donde no existen expertos humanos, o donde los servicios de ese experto serían muy costosos.

El conocimiento de los Sistemas Expertos puede obtenerse por experiencia o consulta de los conocimientos que suelen estar disponibles en libros, revistas y con usuarios capacitados. Los usuarios que introducen la información al SE son en realidad los expertos humanos, y tratan a su vez de estructurar los conocimientos que poseen para ponerlos entonces a disposición del sistema. Los SE son útiles para resolver problemas que se basan en conocimiento.

El objetivo de un SE es igualar el comportamiento de los expertos humanos. Existe mucha gente que se considera experto en la actualidad como por ejemplo los abogados, economistas, médicos. Todos ellos comparten una característica en común: deben de tomar decisiones acertadas en ambientes rodeados de riesgos e incertidumbre pero poseen la habilidad superior de hacerlo como resultado de su entrenamiento, experiencia y práctica profesional. [7] Los expertos humanos tienen las siguientes características generales:

- Son personas raras, tanto por su escaso número como su comportamiento poco “ortodoxo” y e “incomprensible” frente a los problemas con los que se enfrentan [5].
- Son caros por dos motivos: por su escaso número y por necesitar un largo período de aprendizaje [5].
- No están siempre disponibles, pues son humanos y cuando se jubilan o mueren se llevan con ellos todos sus conocimientos [5].

Como se observó anteriormente los expertos humanos tienen un corto tiempo de existencia, contando a partir de su capacidad para actuar como un experto, lo que le atribuye la desventaja de que su conocimiento sea un caudal enriquecido temporalmente; a diferencia de los Sistemas Expertos que se alimentan de estos pero perduran mucho más en el tiempo. [5] (Ver anexo #1)

Ventajas de los Sistemas Expertos

El acceso al conocimiento y al juicio de un experto es extremadamente valioso en muchas ocasiones, sin embargo, en la mayoría de los campos de actividad existen más problemas por resolver que expertos para

resolverlos. Para solucionar este desequilibrio es necesario utilizar un SE. En general, actuará como ayudante para los expertos humanos y como consultor cuando no se tiene otro acceso a la experiencia.

Un sistema experto, además, mejora la productividad al resolver y decidir los problemas más rápidamente. Esto permite ahorrar tiempo y dinero. A veces sin esa rapidez las soluciones obtenidas serían inútiles.

Los valiosos conocimientos de un especialista se guardan y se difunden, de forma que, no se pierden aunque desaparezca el especialista. En los Sistemas Expertos se guarda la esencia de los problemas que se intenta resolver y se programa cómo aplicar los conocimientos para su resolución. Ayudan a entender cómo se aplican los conocimientos para resolver un problema. Esto es útil porque normalmente el especialista da por ciertos sus conocimientos y no analiza cómo los aplica.

Con un Sistema Experto se obtienen soluciones más fiables gracias al tratamiento automático de los datos, y más contrastadas, debido a que se suele tener informatizado el conocimiento de varios expertos.

Debido a la separación entre la base de conocimiento y el mecanismo de inferencia, los Sistemas Expertos tienen gran flexibilidad, lo que se traduce en una mejor modularidad, modificabilidad y legibilidad del conocimiento.

Otra ventaja es que este tipo de sistemas pueden utilizar razonamiento aproximado para hacer deducciones y que pueden resolver problemas sin solución algorítmica. [8]

Los Sistemas Expertos poseen varias características atractivas que pone en evidencia las ventajas que atribuyen su utilización. Seguido a esto se representan algunas de ellas.

- Mayor disponibilidad: la experiencia está disponible para cualquier hardware de cómputo adecuado.
- Costo reducido: se reduce grandemente el costo cuando ponemos la experiencia a disposición del usuario.
- Permanencia: la experiencia es permanente, a diferencia de los especialistas humanos, que pueden retirarse, renunciar o morir, el conocimiento del sistema experto durará indefinidamente.
- Experiencia múltiple: la combinación del conocimiento de varias especialistas puede estar disponible para trabajar sincrónica y continuamente en un problema a cualquier hora del día o de la

noche. La experiencia combinada de muchos Sistemas Expertos puede exceder el de un solo especialista humano.

- Mayor confiabilidad: los especialistas al apoyarse en los Sistemas de Expertos aumentan su nivel de confiabilidad ya que el sistema proporciona una segunda opinión e indica que el especialista ha tomado la decisión correcta.
- Respuesta rápida: hay situaciones que exigen respuestas rápidas, que a veces los humanos no pueden proporcionar por diferentes razones, de modo que un Sistema Experto en tiempo real resulta una buena elección.
- Respuestas sólidas, completas y sin emociones, en todo momento: esto puede ser muy importante en tiempo real y en situaciones de emergencia, cuando un especialista quizá no funcionaría a toda su capacidad a causa de la presión y la fatiga.

El proceso de desarrollo de un Sistema Experto también tiene un beneficio colateral, ya que el conocimiento que se le introduce a partir de los especialistas humanos es claramente detallado y examinado. Como se dispone explícitamente del conocimiento, en vez de tenerlo implícito en la mente del especialista, puede examinarse para corregirlo, darle más consistencia y completarlo. El conocimiento puede entonces ajustarse o reexaminarse, lo que aumenta su calidad. [5]

Arquitectura de los Sistemas de Expertos

Los Sistemas Expertos son, simultáneamente, un sistema de ejecución y un sistema de transmisión del conocimiento. Asimismo, se definen mediante su arquitectura; obtienen, por lo tanto, una realidad palpable. Mientras que en las operaciones de programación clásicas se diferencia únicamente entre el propio programa y los datos, en el caso de los Sistemas Expertos se diferencian tres componentes principales. [5] (Ver figura 1.1)

- Interfaz con el usuario (IU)
- Motor de inferencia (MI)
- Base de conocimiento (BC)

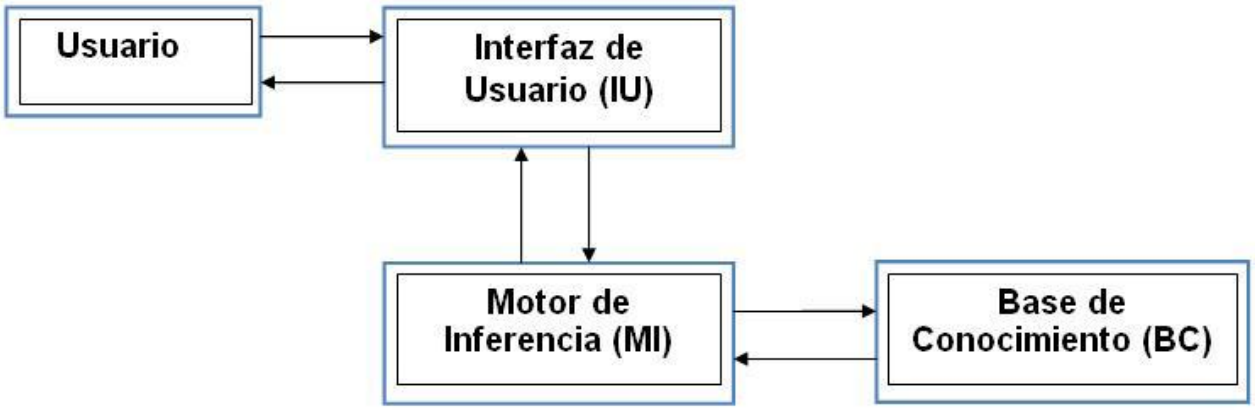


Figura 1.1 Arquitectura de un Sistema Experto.

Mediante la ***interfaz con el usuario***, el mismo plantea los problemas al SE, recibe preguntas del mismo y le son ofrecidas las explicaciones necesarias.

El ***motor de inferencia***, es un programa que emplea conocimientos de un dominio para solucionar un problema dado. Implementa algún método de solución de problema que manipula el conocimiento almacenado en la BC. Por regla general, el tipo de reglas que forman la base de conocimientos es tal que, si A es válido, puede deducirse B como conclusión. En este caso, la tarea que lleva a cabo el motor de inferencias es la de seleccionar, validar y activar algunas reglas que permiten obtener finalmente la solución correspondiente al problema planteado. El motor de inferencia no es un mecanismo universal de deducción, ya que hay dos tipos diversos: los que emplean el razonamiento aproximativo (para el cual el resultado puede ser erróneo) y aquellos que emplean un tipo de razonamiento capaz de obtener un resultado (si llegan a él), con toda seguridad, verdadero.

La ***base de conocimiento*** aloja la totalidad de las informaciones específicas relativas al campo del saber deseado. Está escrita en un lenguaje específico de representación del conocimiento que contiene y en el cual el experto puede definir su propio vocabulario técnico. A la inversa de lo que sucede en los programas clásicos, en la base de conocimientos las informaciones entran tal como llegan, ya que el orden no influye en los resultados obtenidos. Sucede así porque cada elemento de conocimiento es comprensible por sí mismo tomado de forma aislada y, por lo tanto, no es necesario referirse al contexto en el cual está inserto.

La base de conocimiento es el cimiento de un SE, ya que es el depósito donde se almacena el conocimiento del que éste dispone sobre un dominio específico. La cuestión básica de la representación del conocimiento es el desarrollo de una notación suficientemente precisa con la cual representar el mismo. [5] A esa notación se le llama Forma de Representación del Conocimiento (FRC).

Determinado por la Forma de Representación del Conocimiento se da lugar a diferentes variantes de SBC, entre los más conocidos y empleados en labores de clasificación se encuentran: los Sistemas Basados en Reglas (SBR), los Sistemas Basados en Frames (SBF), los Sistemas Basados en Casos (SBC), los Sistemas Basados en Probabilidades (SBP), las Redes Expertas y los Sistemas Basados en Modelos. (Ver anexo #2)

Los ***Sistemas Basados en Reglas*** se caracterizan porque la forma de representación del conocimiento son las reglas de producción y como método de inferencia utilizan la regla de modus ponens. Este tipo de regla expresan siempre una condicional, con antecedentes y un consecuente. La interpretación de una regla surge del hecho que si los antecedentes se satisfacen entonces se logra el consecuente, este tipo de reglas se conoce en la literatura como reglas duras. Como ventaja fundamental muestran su capacidad de interpretación y explicación de la inferencia.

La representación del conocimiento en forma de reglas de producción fue propuesta por Emil Leon Post en 1943. La regla es la forma más común de representar el conocimiento, debido a su gran sencillez ya que es la formulación más inmediata del principio de causalidad. Una regla consta de un conjunto de acciones o efectos (una o más) que son ciertas cuando se cumplen un conjunto de condiciones o causas. La potencia de una regla está en función de la lógica que admita en las expresiones de las condiciones y de las conclusiones.

Las ventajas que representan las reglas de producción son su carácter declarativo, su sencillez, su uniformidad que permite la representación de conocimiento, su independencia que permite la supresión o inclusión sin que se vea afectado el resto de la base de conocimientos y su modularidad al ser fácilmente agrupables.

Las reglas utilizan un formato WHEN - THEN para representar el conocimiento, la parte WHEN de una regla es una condición (también llamada premisa o antecedente), y la parte THEN de la regla es una

acción (también llamada conclusión o consecuente) permite inferir un conjunto de hechos nuevos si se verifican las condiciones establecidas en la parte WHEN.

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

1- Seleccionar una conclusión posible y tratar de probar su validez buscando evidencias que la soporten (encadenamiento hacia atrás).
2- Comenzar con todos los datos conocidos y progresar hacia la conclusión (encadenamiento hacia delante). [9]

Encadenamiento hacia atrás

El encadenamiento hacia atrás no tiene como objetivo inferir el valor de una variable, sino la demostración de una hipótesis. Su desarrollo se presenta como sigue:

1. Se define una hipótesis, es decir, un valor a alcanzar en una variable de salida del sistema.

2. Si la hipótesis se basa en un hecho comprendido en la base de conocimiento, se finaliza el proceso. En caso contrario se continúa al paso 3.

3. Se busca el subconjunto de reglas de la base de conocimiento cuyos consecuentes coincidan con la hipótesis.

4. Se establecen los antecedentes de las reglas del subconjunto seleccionado como nuevas hipótesis a demostrar y se vuelve al paso 2.

Este tipo de inferencia establece (si existe) una cadena de reglas que prueba la veracidad de una hipótesis. MYCIN fue uno de los pioneros en aplicar este tipo de inferencia, con el fin de detectar infecciones bacterianas en pacientes.

Encadenamiento hacia delante

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

1. Se define el hecho a alcanzar, es decir, las variables de salida del sistema cuyo valor se desea inferir.
2. Un conjunto de hechos relativo a las variables de entrada es observado por el sistema, es decir, llega una entrada al sistema.
3. Se busca el subconjunto de reglas de la base de conocimiento cuyos antecedentes son satisfechos por los hechos observados.
4. Si el subconjunto está vacío, se finaliza el proceso. En otro caso se continúa al paso 5
5. El subconjunto de reglas seleccionado se activa y da lugar a un número de hechos nuevos igual al tamaño del subconjunto.
6. La base de hechos se actualiza con los nuevos hechos.
7. Si se ha alcanzado el hecho de salida deseado se finaliza el proceso, en otro caso, se vuelve al paso 3. [10]

Aplicaciones

Los Sistemas Expertos han demostrado ser herramientas muy útiles en gran cantidad de situaciones. En las últimas décadas, se han desarrollado un gran número de Sistemas Expertos en diferentes áreas del conocimiento: Medicina, Geología, Química, Economía, Ingeniería Civil. Estos programas proporcionan la capacidad de trabajar con grandes cantidades de información, que son uno de los grandes problemas que enfrenta el analista humano que puede afectar negativamente a la toma de decisiones, pues el analista humano puede depurar datos que no considere relevantes, mientras un *SE* debido a su gran velocidad de procesamiento analiza toda la información incluyendo las no útiles para de esta manera aportar una decisión más sólida. Por otra parte es importante mencionar que estos seguirán siendo usados en todas y cada una de las áreas y/o campos donde los expertos humanos sean escasos. A continuación la tabla 1.3 presenta de manera resumida los primeros Sistemas Expertos y sus aplicaciones en la Medicina. [11]

SISTEMA	FECHA	AUTOR	APLICACIÓN
Dendral	1965	Stanford	Deduce información sobre estructuras químicas
Mycin	1972	Stanford	Diagnóstico de enfermedades de la sangre
Caduceus	1975	University of Pittsburg	Herramienta de diagnóstico para medicina interna

Tabla 1.1 Primeros Sistemas de Expertos en la Medicina.

1.3 Sistemas Expertos en la Genética Médica

Desde mitad del siglo XX se han venido desarrollando en el mundo el empleo de Sistemas Expertos con gran número de enfermedades genéticas, que proporciona a los doctores un apoyo en el diagnóstico de un paciente, a continuación algunos ejemplos.

- **London Dysmorphology Database, London Neurogenetics Database & Dysmorphology Photo Library on CD-ROM**

Publicado por primera vez, en 1990, como la base de datos de Dismorfología de Londres, la 3ra edición, Londres Neurogenética de base de datos y Dysmorphology Photo Library en CD-ROM, combina dos bases de datos globales, junto con una amplia biblioteca de fotos.

El Dr. Baraitser y el Prof. Winter han incluido más de 3.400 síndromes que no tienen origen cromosómico en la base de datos dismorfología y cerca de 3.300 trastornos neurológicos en la base de datos neurogenética. [12]

- **POSSUM (Physiological and Operative Severity Score for the enumeration of Mortality and Morbidity)**

Lo desarrolló Copeland en 1991 como un método de estratificación de todos los tipos de pacientes quirúrgicos de acuerdo con su grado de riesgo, y en el que únicamente se utilizan hallazgos clínicos,

exámenes de laboratorio y los datos del protocolo operatorio, de modo que puede utilizarse en una gran variedad de situaciones diferentes.

Se desarrolló a partir de un análisis aleatorizado, multivariado, de 62 variables diferentes, durante 6 meses en una etapa inicial y luego con 35 variables durante un período similar, antes de llegar a su actual formulación. Este índice permite evaluar los rangos esperados de morbilidad y mortalidad ajustadas por riesgo para una determinada situación, a partir de la consideración de 2 puntuaciones: una fisiológica, con 12 factores, y la otra de gravedad operatoria, que utiliza 6 factores.

El Possum no se puede utilizar para evitar que un paciente sea sometido a un procedimiento potencialmente curativo. Aunque el sistema fue diseñado no específicamente para ser utilizado como un índice de "inutilidad", es posible seleccionar una combinación de resultados para los cuales se debe buscar una opinión de expertos humanos antes de la intervención quirúrgica.

- **OMIN (Online Mendelian Inheritance in Man)**

Esta base de datos se inició en la década de 1960 por el Dr. Victor A. McKusick como un catálogo de rasgos mendelianos y trastornos, titulado Mendelian Inheritance in Man (MIM). Doce ediciones de libros de MIM fueron publicados entre 1966 y 1998. La versión en línea, OMIM, fue creado en 1985 por una colaboración entre la Biblioteca Nacional de Medicina y la William H. Welch Medical Library en Johns Hopkins. Se hizo generalmente disponible en el Internet a partir de 1987. En 1995, OMIM fue desarrollado para la World Wide Web por NCBI, el Centro Nacional de Información sobre Biotecnología.

OMIM es destinado a ser utilizado principalmente por los médicos y otros profesionales interesados en los trastornos genéticos, la investigación en genética, y por los estudiantes avanzados de la ciencia y la medicina. Si bien la base de datos OMIM es abierto al público, los usuarios que buscan información sobre una condición médica o genética personal es instado a consultar con un médico calificado para el diagnóstico y para obtener respuestas a preguntas personales. [13]

Estas bases de datos pueden ser consultada por personas que no son especialistas, con solo el acceso a internet para evacuar sus dudas o curiosidades, pero una evaluación certera solo puede ser obtenida por genetistas, que cobran por sus servicios y numerosos pacientes se ven limitados a la hora de pagar una

gran suma de dinero por la requerida atención. En Cuba todos los ciudadanos tenemos derecho a una salud gratuita, lo que por el bloqueo económico que el gobierno estadounidense ha impuesto por más de 50 años a Cuba, la internet es limitada y concentrada en los centros estatales, ya que Cuba no puede conectarse a la red a la velocidad que desea, pues sólo puede hacerlo a través de una conexión satelital cara y lenta. Por este motivo a los cubanos se les dificulta consultar éstas grandes bases de datos, que en su totalidad están en inglés y hacen difícil su comprensión. Además estas bases de datos no están diseñadas en una aplicación web, la cual es mucho más fácil de consultar, sino en una aplicación desktop que cuesta aproximadamente $ 2495,00.

La versión 1.0 del SEGEDIS facilitará la mejora de los servicios de atención a la salud. Permitirá a los médicos consultar una base de datos que le proporcionará una segunda opinión acerca de la decisión que ha tomado. Tendrá almacenado miles de enfermedades recogidas de expertos humanos en este dominio, las cuales son imposible de ver todas por un solo profesional. Es un sistema que estará trabajando en tiempo real en soluciones que requieran de rapidez. Facilitará una búsqueda especializada de enfermedades genéticas con dismorfias, tanto por los signos clínicos como directamente por el nombre de la enfermedad. Cada enfermedad contendrá un resumen con la descripción de esta, los signos clínicos que la generan y referencias donde se puede abundar más sobre la misma. Es un sistema que estará disponible en la Red Nacional de Genética Médica para todos los especialistas que le concedan el permiso. Sólo el CNGM otorgará estos permisos y será administrado por el mismo para una mayor seguridad e integridad de la información que contenga el SEGEDIS.

1.4 Metodología

Al igual que para desarrollar un sistema de información convencional existen varias metodologías de desarrollo como la Ingeniería de la Información, así existen varias metodologías para desarrollar un sistema experto. El área de Sistemas Expertos es relativamente joven por lo cual no se dispone de una única metodología sino que cada autor propone una de acuerdo a su forma de desarrollo. Sin embargo existen algunas que han tenido éxito más que otras lo cual ha llevado a su mayor difusión. [14]

Aquí solo se mencionarán algunas y se mostrará un esquema general de la metodología con la cual se trabajará en la tesis que ocupa.

- Metodología KADS
- Metodología de Buchanan
- Metodología de Grover
- Metodología clásica de la Ingeniería del Conocimiento

Metodología KADS

Esta metodología ofrece un marco estructurado de desarrollo en el que se proporcionan diversos modelos para responder a diversos problemas. Estos pueden estar relacionados con el impacto del desarrollo en la organización (*modelo de la organización*), con la naturaleza y la estructura del conocimiento utilizado (*modelo del conocimiento*), con los agentes que ejecutan las tareas (*modelo de agente*), con la comunicación entre los agentes (*modelo de comunicación*).

KADS pretende establecer una distinción clara entre las distintas fases de desarrollo de un Sistema Experto: análisis (modelo conceptual), especificación de distintas capas en el modelo del dominio (estrategia, tarea, inferencia y dominio) y, finalmente, obtención de una descripción formal y reescritura de los términos del modelo usando las primitivas de un lenguaje de implementación concreto. [14]

Metodología de Buchanan

Este método puede esquematizarse en cinco etapas representada en la figura 1.2. A continuación se explican brevemente:

Identificación

– Se reconocen aspectos importantes del problema:

- Participantes.
- Características del problema.
- Recursos disponibles.
- Metas.

Conceptualización

- Organización del conocimiento según un esquema conceptual.
- Búsqueda de conceptos que representen el conocimiento del experto.
- Identificación del flujo de información durante el proceso de resolución de problemas.

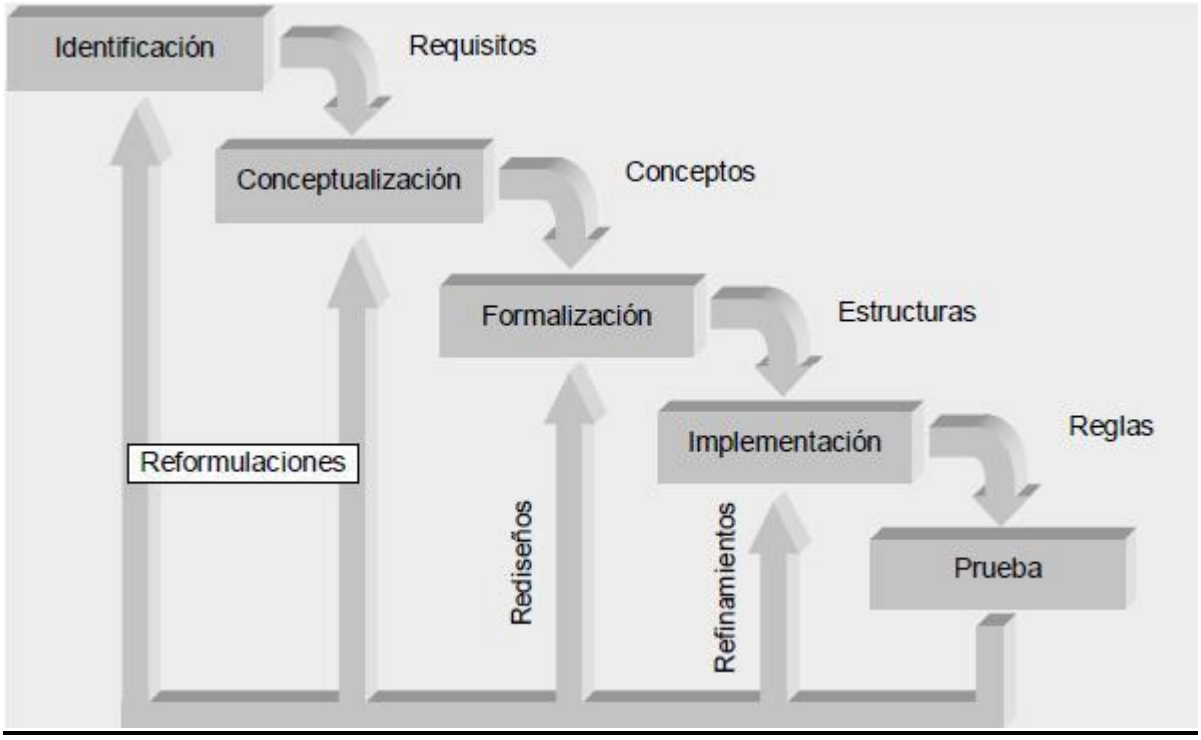


Figura 1.2 Fases de la metodología de Buchanan.

Formalización

- Proceso de traducción de:

  - Conceptos clave.
  - Subproblemas.
  - Características del flujo de información.

- Construcción de representaciones formales basadas en:

  - Herramientas de desarrollo.

- Esquemas de ingeniería del conocimiento.

Implementación

- Formulación de reglas.
- Formulación de estructuras de control.
- Obtención de un prototipo.

Prueba

- Evaluación del rendimiento del prototipo construido.
- Identificación de errores.

Metodología de Grover

La metodología de Grover (1983) se concentra en el la definición del dominio (conocimiento, referencias, situaciones y procedimientos) en la formulación del conocimiento fundamental (reglas elementales, creencias y expectativas) y en la consolidación del conocimiento de base (revisión y ciclos de corrección).

Este método además de definir dos etapas, propone con énfasis una documentación de los procesos: los cuales reemplazan parcialmente al experto y sirven de medio de comunicación y referencia entre los usuarios y los diseñadores. [14]

Etapa 1: *Definición del Dominio*

El objetivo de esta etapa consiste en realizar una cuidadosa interpretación del problema y documentarla, elaborando un Manual de Definiciones del Dominio, el cual debe contener:

- Una descripción general del problema.
- La bibliografía de referencia.
- Un glosario de términos y símbolos.
- La identificación del o los expertos.
- La definición de métricas de performance (parámetros) para evaluar el rendimiento del Sistema Experto.

- La descripción de escenarios para ejemplos posibles.

Etapa 2: *Formulación del Conocimiento Fundamental*

Esta etapa tiene como objetivo examinar los escenarios ejemplo a partir de criterios de evaluación y reclasificarlos según:

- Los más importantes y los más insignificantes.
- Los más esperados.
- Los más arquetípicos.
- Los más comprensibles.

Metodología clásica de la Ingeniería del Conocimiento

La Ingeniería del Conocimiento (IC) es la disciplina de la Inteligencia Artificial encargada de hacer explícitos los conocimientos de un dominio en una BC separada del resto del sistema en el que se integra. El trabajo de los ingenieros del conocimiento consiste en extraer el conocimiento de los expertos humanos en una determinada área, y en codificar dicho conocimiento de manera que pueda ser procesado por un sistema.

La ingeniería del conocimiento engloba a los científicos, tecnología y metodología necesarios para procesar el conocimiento. Su objetivo es extraer, articular e informatizar el conocimiento de un experto. En la figura 1.3 se representa una gráfica del ciclo de vida clásico de la ingeniería del conocimiento. [14]

En la bibliografía consultada no consta que este tema haya sido tratado de una manera sistemática. Cuando se habla de formas de construir Sistemas Expertos, éstas se encuentran muy ligadas a aplicaciones específicas y hay que deducir los pasos seguidos a través del desarrollo que se expone del Sistema Experto. Además las metodologías expuestas anteriormente no cumplen con las expectativas que se persiguen, ya que algunas engloban muchas técnicas y modelos lo cual se necesitaría un largo período de tiempo para su construcción y otras no delimitan las fases esenciales en la elaboración de un sistema experto. Por todo lo dicho anteriormente se escogió el ciclo de vida clásico que propone la ingeniería del conocimiento.

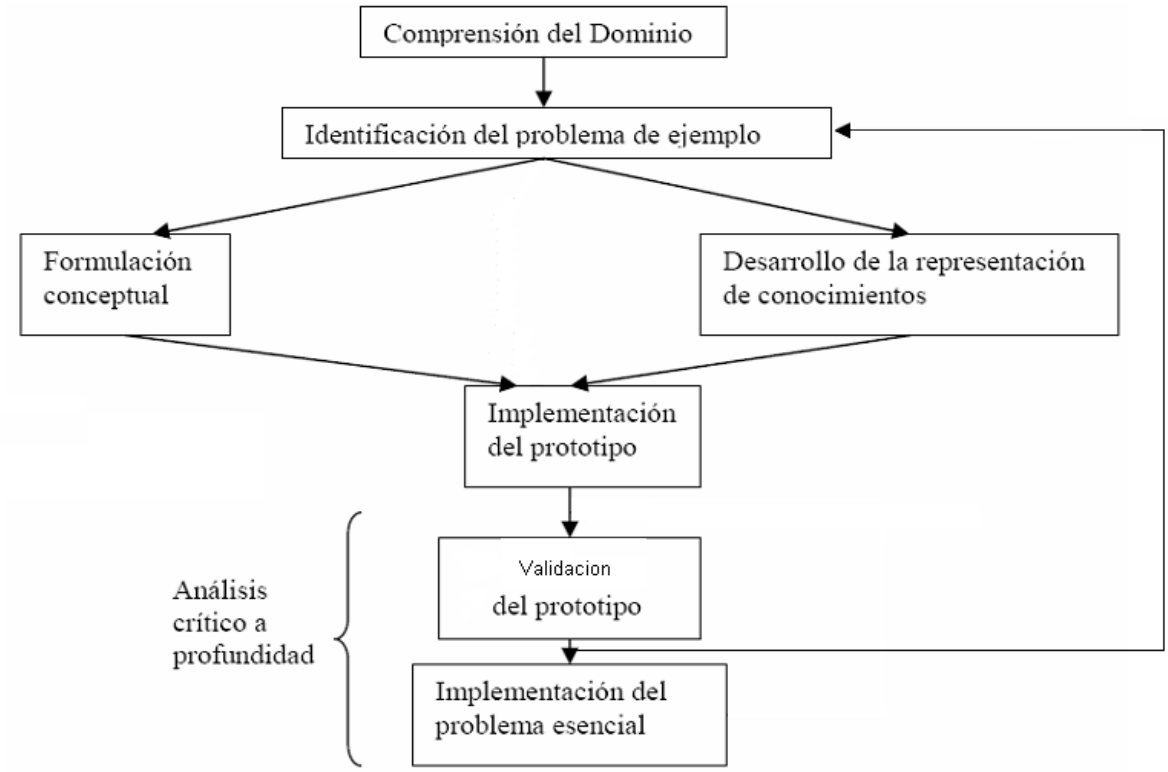


Figura 1.3 Ciclo de vida clásico de la ingeniería del conocimiento.

Vista la figura 1.3 y analizada cada una de sus etapas se determinó una metodología clásica, reducida a cinco etapas ya que no todas se adaptan a las características del problema. Las fases a seguir son: identificación del problema, adquisición del conocimiento, representación del conocimiento, implementación del Sistema Experto y prueba.

Identificación del problema

En esta etapa se determina, básicamente, cuál es el problema que se quiere resolver y sus características, así como las personas que van a participar en el levantamiento de la base de conocimiento y el papel de cada cual en ese proceso. En un trabajo conjunto se deben determinar:

- Cuáles serán los datos iniciales.
- Los posibles subproblemas en que pueda dividirse el problema inicial.

- Cuáles son los objetivos relevantes e irrelevantes, partiendo de la percepción de la realidad y teniendo en cuenta los propósitos del sistema que se quiere construir.
- Los recursos con que se cuenta para la obtención de información, es decir, las posibles fuentes de conocimiento, que pueden ser utilizadas además del conocimiento extraído a partir de los expertos, información acumulada en libros sobre el tema que se trabaja.

Adquisición del conocimiento

La adquisición del conocimiento es la labor de extracción del conocimiento de las fuentes. Existen varias técnicas de adquisición del conocimiento. Todas estas técnicas pueden ser utilizadas en la resolución de un problema, pero también se pueden escoger unas pocas según las particularidades que posea cada Sistema Experto a construir.

- Entrevista directa o formal: La IC establece un plan de reuniones en el que se determina el objetivo principal de la misma, el tema a tratar, los recursos que se necesitan para registrar (guardar) la entrevista, la fecha, la hora y el lugar donde se llevará a cabo dicha entrevista.
- Entrevista informal: Se realiza de forma personal pero no planeada. Es aprovechar la oportunidad del encuentro entre la IC y la persona que tiene el conocimiento, en donde el primero le hace una pequeña entrevista al segundo.
- Observación del trabajo real del experto: Consiste en examinar la labor del experto en su ambiente de trabajo, solucionando un problema como el que se está tratando de simular.

Representación del conocimiento

Este proceso consiste en coger el conocimiento extraído y representarlo en una forma evidente. En esta etapa diseñan las estructuras para organizar el conocimiento. Después de un análisis intensivo, por parte del ingeniero del conocimiento, de los diferentes medios de representación con que se cuenta, se determina cuál se adapta mejor a las condiciones del problema, estableciendo un lenguaje formal que incorpore los conceptos formalizados del tema objeto de representación y describa a la vez el mecanismo de solución.

Para seleccionar un esquema de representación adecuado se deben considerar la generalidad, accesibilidad, posibilidades de prueba y velocidad de razonamiento, el esquema seleccionado depende fundamentalmente de tres aspectos:

- Las características del dominio del problema, que incluye a su vez: la estructura del espacio de búsqueda, es decir la forma en que se puede subdividir y organizar el dominio de solución.
- Las características o propiedades de los datos: es muy importante comprender la naturaleza de los datos del dominio de conocimiento, es decir si son datos exactos, inexactos, completo o incompletos.
- Las características del esquema para resolver el problema, que incluye:

    1. El tipo de búsqueda (backward, forward)
    2. Las características del sistema basado en el conocimiento que se desea construir, que incluye: el tipo de usuario (experimentado, no experimentado) y el método de aplicación del sistema.

Implementación

En esta etapa el ingeniero del conocimiento combina y reorganiza el conocimiento formalizado para hacerlo compatible con las características del flujo de información del problema, obteniéndose un primer prototipo.

El objetivo fundamental de este primer prototipo es obtener una solución inmediata del problema, independientemente de cuan eficiente pueda ser. En esta etapa se debe demostrar que los métodos de solución seleccionados son los más indicados para la exitosa solución de, al menos, un primer grupo de problemas, comenzando por los más generales y urgentes y pasando posteriormente a un refinamiento del conocimiento.

El proceso de refinamiento consiste en el mejoramiento del conocimiento ya existente y la adición de nuevos fragmentos de conocimiento, además de mejorar la estrategia de búsqueda del sistema, lo que contribuye a una solución más efectiva y confiable.

La realización de pruebas al primer prototipo y el análisis de las críticas que se hagan a su funcionamiento servirán como punto de partida para la formalización de una versión más avanzada de la base de conocimiento.

Prueba

Este es un período de validación del conocimiento ya formulado. Se hace una valoración del sistema en su conjunto, probándolo con un grupo bastante amplio de ejemplos, de forma que se cubran todos los casos posibles. Se hace con el objetivo de determinar insuficiencias en la base de conocimiento y/o en las estrategias para la solución del problema.

Para la aplicación efectiva del sistema en la práctica, se hacen pruebas con diferentes expertos que ofrecerán sus puntos de vista acerca de su funcionamiento. Después que se han hecho todas las críticas, se incorporan las correcciones a la formalización final y se pasa a la implantación como tal del sistema. Seguida de un período cíclico de permanente crítica y mejora de la base de conocimiento hasta que se alcance una situación de estabilidad. Entre todas estas etapas, no existe una clara separación e independencia. Por ejemplo, la formalización e implementación están estrechamente relacionadas.

En resumen, un sistema basado en el conocimiento evoluciona a partir de tareas simples hacia propósitos más difíciles, mejorando en la organización y representación del conocimiento. Ocasionalmente, cuando se desean capacidades que exceden el alcance del sistema actual, es necesario remodelarlo.

1.5 Tecnologías y herramientas

Las herramientas y tecnologías que se utilizan en el presente trabajo se dividen en dos partes, las que surgen debido a la utilización de técnicas de IA y las definidas en la arquitectura del producto alasMEDIGEN: Sistema Informático de Genética Médica. Posteriormente se quiere integrar este producto a alasMEDIGEN: Sistema Informático de Genética Médica y por eso se utiliza muchas herramientas definidas en su arquitectura para que sea más flexible a la hora de integrarlo, ya que el SEGEDIS es un sistema que apoya a la toma de decisiones de los genetistas.

Definidas en el producto alasMEDIGEN: Sistema Informático de Genética Médica

Entorno de desarrollo: Eclipse 3.5

Eclipse es un IDE o una plataforma universal para integrar herramientas de desarrollo, con una arquitectura abierta y basada en plug-ins. Además da soporte a todo tipo de proyectos que abarcan todo el ciclo de vida del desarrollo de aplicaciones, incluyendo soporte para modelado. La arquitectura basada en plug-ins permite integrar diversos lenguajes sobre un mismo IDE e introducir otras aplicaciones accesorias. Conservan el registro de las versiones, generan y mantienen la documentación de cada etapa del proyecto.

Las características más importantes de Eclipse son las siguientes:

- Editor visual con sintaxis coloreada.
- Compilación incremental de código.
- Modifica e inspecciona valores de variables.
- Avisa de los errores cometidos mediante una ventana secundaria.
- Depura código que resida en una máquina remota.
- ECLIPSE es soportado por los principales sistemas operativos Linux, Windows, Solaris 8, Mac OSX –Mac/Carbon.
- En Eclipse existen Control de versiones como CVS y Subclipse. [15]

Tecnologías del lado del servidor. Lenguajes de programación: PHP 5.0

Uno de los aspectos más novedosos de PHP 5.0 es Zend Engine II, que entre otras características, presenta un modelo basado en objetos, que mejora la funcionalidad general. PHP 5.0 soporta XML, que para esta versión se ha reescrito íntegramente, también soporta MySQLi, una nueva ampliación de MySQL, la cual además de la interfaz habitual, encierra una interfaz basada en objetos. Estas son algunas de las mejoras que este incluye:

- Zend Engine II, con un nuevo modelo de objetos más avanzado y robusto que su predecesor de PHP 4.
- Soporte para XML reescrito desde cero, todas las extensiones relacionadas están escritas ahora entorno a la excelente librería libxml2.

- Nueva extensión MySQL denominada MySQLi para los desarrolladores que utilicen MySQL 4.1 y versiones posteriores. Esta extensión incluye una interfaz orientada a objetos como adición a la interfaz tradicional; así como soporte para las numerosas nuevas funciones de MySQL.
- El soporte para streams ha sido mejorado, incluyendo soporte para acceder a operaciones de bajo nivel sobre los sockets en streams. [16]

Framework: Symfony 1.0.20

Hoy día el uso de los framework se ha popularizado a gran escala, facilitando a los desarrolladores la creación de aplicaciones con un coste de tiempo considerablemente menor que hace unos años cuando estas no estaban presente. En el caso particular de **Symfony 1.0.20,** seleccionado para el desarrollo de la aplicación, es un completo framework diseñado para optimizar el desarrollo de las aplicaciones web mediante algunas de sus principales características. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. [17]

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

Servidor Web. WampServer

Apache es un servidor Web, flexible, rápido y eficiente, de código fuente abierto, continuamente actualizado y adaptado a los nuevos protocolos. Está disponible para diferentes plataformas como: FreeBSD, NetBSD, OpenBSD, GNU/Linux, Mac OS y Mac OS X Server, Netware, Solaris, Windows, entre otras. Con los diferentes módulos de apoyo que proporciona y con la API de programación de módulos, puede ser adaptado a diferentes entornos y necesidades. [18]

Servidor Web. Tomcat

Apache Tomcat 6.0 es una aplicación de software open source que implementa las especificaciones de Java Servlet 2.5 y JavaServer Pages 2.1. Fue desarrollado bajo el proyecto Jakarta en la Apache Software Fundation, en un entorno abierto y participativo y publicado bajo la licencia del software de Apache. Puede funcionar como servidor HTTP o conectado a otro servidor HTTP como Apache HTTP o IIS (Internet Information Service). Permite ejecutar servicios web mediante Apache Axis. [19]

Gestor de Base Datos. MySQL

MySQL es un sistema de gestión de base de datos relacionales. Opera en una arquitectura cliente/servidor, de tal manera que el servidor sólo tiene que enviarle una cadena de caracteres y esperar la devolución de los datos. Es el sistema de gestión de bases de datos más usado. Todo el mundo puede acceder al código fuente y contribuir con ideas, elementos, mejoras o sugerir optimizaciones. MySQL ha pasado de ser una pequeña base de datos a una completa herramienta. Es una tecnología útil para la creación de bases de datos seguras al poseer un sistema de privilegios y contraseñas que es muy flexible y seguro, que permite verificación basada en el host. Las contraseñas son seguras porque todo el tráfico de ellas a través de la Web está encriptado al conectarse con un servidor.

Las principales características de este gestor de bases de datos son las siguientes:

- Soporta gran cantidad de tipos de datos para las columnas.
- Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP).
- Proporciona sistemas de almacenamiento transaccional y no transaccional.
- Relativamente sencillo de añadir otro sistema de almacenamiento. Esto es útil si desea añadir una interfaz SQL para una base de datos propia.
- Gran portabilidad entre sistemas.

Herramientas CASE: Visual Paradigm 6.1

Es una herramienta CASE que utiliza “UML” como lenguaje de modelado. Está orientada a la creación de diseños usando el paradigma de programación orientada a objetos (POO).

Visual Paradigm para UML es un galardonado producto que facilita las organizaciones del diagrama visual y de diseño, integrar y desplegar sus aplicaciones empresariales de misión crítica y de sus bases de datos subyacentes. Es una poderosa herramienta para visualizar y diseñar elementos de software, utilizando el lenguaje UML. [20]

Las que surgen debido a la utilización de técnicas de IA

Motor de inferencia

Existen varios motores de inferencias escritos en java donde aparecen OFBiz, Jess y Drools. Un detalle importante es que casi todos los motores de reglas utilizan el lenguaje RuleML (Rule Mark-up Language) basado en XML para definir las reglas, excepto Drools, "que no lo estima prioritario". Esto parece un poco contradictorio, ya que la especificación JSR-94 asume la existencia de un esfuerzo paralelo para la especificación de un lenguaje abierto de reglas basado en XML y se refiere explícitamente al proyecto RuleML.

- ***OFBiz***

El OFBiz es un motor de reglas que se basa en tecnologías de programación de la lógica que existen desde hace varias décadas. El lenguaje de programación lógica más notable es Prolog. En la encarnación actual del motor de OFBiz utiliza el algoritmo encadenamiento hacia atrás. En la parte superior del motor de reglas se necesita un método de hacer valer los hechos y se definen las normas. Si está trabajando directamente con la API del motor de reglas, estos objetos se pueden añadir de esa manera. Eso puede ser útil para algunas circunstancias, pero sería tedioso REALMENTE si fuera la única forma de crear hechos y reglas.

- ***Jess***

Es un sistema basado en reglas de producción. Jess permite *encadenamiento de reglas hacia adelante*, emparejando hechos en Memoria Activa con antecedentes, y *hacia atrás*, emparejando hipótesis en Memoria Activa con consecuentes. Además puede manipular y razonar directamente sobre objetos de Java. Se pueden crear objetos, llamar a métodos, o ejecutar interfaces sin tener que compilar ningún

código Java. La desventaja de este motor de reglas del negocio es que no es gratuito lo que dificulta su utilización.

- ***Drools***

Drools es un sistema de administración de reglas de negocio (BRMS) Business Rules Management System con un motor de reglas basado en una adaptación orientada a objetos del algoritmo Rete. Permite expresar de una forma más natural las reglas de negocio interactuando con los objetos de negocio. Provee separación de lógica (reglas) y datos (hechos). También provee soporte para la programación declarativa, y es lo suficientemente flexible para expresar la semántica del problema con un lenguaje específico de dominio (DSL). Drools cuenta con la implementación completa de la JSR-94 Rule Engine API. También existe un plug-in de Eclipse para facilitar el desarrollo con esta herramienta.

Para especificar las reglas Drools utiliza el lenguaje de reglas de drools (DRL) para especificar las condiciones, acciones y funciones de las mismas, las cuales se puede expresar con distintos lenguajes, como Java y MVEL. Entonces las reglas son guardadas en archivos de texto con la extensión drl.

Drools utiliza el algoritmo Rete, ya que es un rápido igualador de patrones que obtiene su velocidad del almacenamiento de información sobre las reglas de una red. En lugar de tener que igualar los hechos con todas las reglas en cualquier ciclo-acto reconocimiento, el algoritmo Rete sólo busca los cambios en las correspondencias de cada ciclo. Esto acelera en gran medida la correspondencia de los hechos con los antecedentes, porque los datos estáticos que no cambiaron de un ciclo a otro pueden pasarse por alto. En el presente trabajo se hará uso del motor de reglas del negocio Drools por las razones antes expuestas. [21]

Frameworks en JavaScript

Para determinar que framework se utiliza en el presente trabajo de diploma se hace una comparación de diferentes framework que se usan en el mundo.

Existen múltiples Frameworks en JavaScript muchos de ellos muy buenos y de código abierto. Con la llegada del web 2.0 y lo que conocemos hoy en día como AJAX hoy en día javascript está más vigente que nunca pues basta con llamar un conjunto de elementos objetos como controles, widgets y en pocos

minutos se tiene una aplicación web. A continuación se presentan algunos con sus respectivas características. [22]

- ***Dojo***

Dojo es un Framework en JavaScript el cual te permite añadir características dinámicas fácilmente a las páginas Web. Puedes utilizar los componentes que incorpora Dojo para mejorar la usabilidad y funcionabilidad. Puedes construir interfaces con degradados de color, widges y transacciones animadas de forma rápida y sencilla. Contiene varias características que trabajan a través de la mayoría de los navegadores, tales como: Menúes, Tabs, Tooltips y Tablas ordenables.

- ***JQuery***

Es una librería liviana que enfatiza la interacción entre Javascript y HTML. Microsoft integra jQuery VisualStudio para el uso en aplicaciones desarrolladas en ASP.NET.

Con facilidad podría ser considerado el framework más utilizado en la actualidad. En adición, en internet existen bastantes plug-ins que agregan funcionalidades extras al framework.

Actualmente su última versión estable, la 1.3.2, viene un solo archivo de 19KB, y su funcionalidad puede ser extendida utilizando plug-ins.

- ***Moo tools***

Liviano, modular y orientado a objetos, la meta es ser un intermediador para los desarrolladores ayudándolos a crear código javascript en una manera elegante, flexible y eficiente. Contiene un gran número de componentes, pero no todos necesitan ser cargados en cada aplicación. Consta de un Core, que es una colección de librerías que el resto de sus componentes necesitan, Class, que es la librería básica.

También provee componentes que enriquecen a los objetos nativos de javascript, para agregar compatibilidad y simplificación de código. Fx es su API avanzado de efectos para animar Elementos.

- ***ExtJS***

Originalmente fue construido como una extensión de YUI. Incluye interporalidad con JQuery y Prototype. Posee controles para Campos de Textos, incluyendo Áreas de Texto. Controladores selectores de fecha, Campos Numéricos, para Radiobox y Checktbox.

También contiene componentes para crear y manipular DataGrids donde goza de cierta ventaja sobre otros frameworks. Posibilita crear “ventanas” con Barras de Herramientas y Menú con estilo de aplicaciones de Escritorio, Diálogos modales y eventos.

ExtJS puede ser adquirido bajo licencias Libres y Comerciales. La compañía detrás de este framework ofrece cursos de capacitación y un extenso soporte. Entre los ejemplos ofrecidos en su sitio web encontramos ejemplos de Tabs, Ventanas, Árboles, Menú y más.

Es difícil determinar cuál framework es mejor. Es importante recordar que estos frameworks son simples herramientas que ayudan a realizar diferentes tareas de diferentes tipos, todos basados en el mismo lenguaje, javascript. Lo correcto es seleccionar uno u otro dependiendo la utilidad y la capacidad de saber cómo utilizarlo.

JQuery ha tenido gran éxito, ya que es fácil de usar y aprender, además de que existe en internet una gran cantidad de documentación y plug-ins que extienden su funcionalidad. Ha sido muy utilizado por Diseñadores con pocos conocimientos en programación. Scriptaculous, basado en Prototype, se integra bastante bien con Ruby OnRails, lo cual ha beneficiado a desarrolladores independientes y empresas al momento de crear aplicaciones de vanguardia.

DojoToolkit, en su incorporación con Zend Framework promete mucho, pero no goza de la popularidad de MooTools, que aunque es liviano y orientado a objetos, en algunas ocasiones resulta complicado realizar simples acciones, que utilizando Prototype suelen ser bastantes sencillas, como Peticiones Periódicas y acceso a diferentes elementos.

No cabe duda de que si lo que se quiere crear es una aplicación web con estilo de aplicación de Escritorio, **ExtJs** es la mejor opción, ya que este framework goza de una gran comunidad de desarrollo dentro de la Universidad de la Ciencias Informáticas lo que facilita un mejor intercambio de conocimientos.

Servidor Web. ApacheTomcat

Tomcat es una aplicación de código abierto que funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en el apache software Foundation. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages o JSP de Sun Microsystems. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. Es desarrollado en un entorno abierto y participativo y publicado bajo la licencia del software de Apache. Además utiliza librerías axis para las aplicaciones de servicios web. [18]

Conclusiones parciales

De acuerdo a lo antes analizado se llegó a la conclusión de que el CNGM no cuenta con una herramienta que recoja el conocimiento de varios especialistas y que esté disponible para trabajar simultánea y continuamente en un problema, a cualquier hora del día y de la noche. Se plasmó una fundamentación teórica, lo que permitió dejar definido la posición de los autores en cuanto a la necesidad de crear un Sistema Experto que apoye al diagnóstico de enfermedades genéticas con dismorfias. Para obtener un buen desarrollo de este trabajo se realizó una comparación de varias metodologías y se determinó que la metodología clásica de la Ingeniería del Conocimiento se adapta a las características del problema. Luego de un estudio de varias técnicas y herramientas existentes, se escogieron las apropiadas para dicho trabajo, con el objetivo de garantizar la adecuada prestación del Sistema Experto, las cuales son: como entorno de desarrollo: Eclipse 3.5, lenguaje de programación: PHP 5.0, framework de desarrollo: Symfony 1.0.20, como gestor de bases de datos: MySQL 5.0, como herramienta CASE: Visual Paradigm 6.1, servidores web: el Tomcat y el Apache, como framework en JavaScript: ExtJS 3.1 y el motor de regla: Drools.

CAPÍTULO 2: SOLUCIÓN PROPUESTA

Introducción

En este capítulo se pone en práctica la metodología para un Sistema Experto. Se identifica el problema a resolver, así como el equipo de desarrollo que dará progreso ha dicho trabajo. Expone varias técnicas de adquisición del conocimiento y las fuentes donde fue extraído el mismo. Brinda un ejemplo de la forma que se representa el conocimiento y muestra el mecanismo de búsqueda que se utiliza. Se representa la arquitectura del SEGEDIS y los prototipos de interfaz de usuario. Además se expone el estándar y los estilos de codificación que se tuvieron en cuenta en el desarrollo del trabajo, las pautas de diseño y algunos patrones que utiliza Symfony.

2.1 Identificación del problema a resolver

La Genética tiene una clara entidad como especialidad médica tanto por el volumen de pacientes y familias a los que debe atender de manera específica, como por la formación que deben tener los profesionales. La asistencia clínica atesora el consejo genético para la planificación familiar, la atención médica durante el desarrollo embrionario y fetal, la atención médica del niño y del adulto con enfermedades genéticas, incluyendo cáncer hereditario, así como los estudios y las medidas epidemiológicos encaminados a disminuir la incidencia de malformaciones congénitas y enfermedades hereditarias.

El diagnóstico médico juega un papel fundamental en la atención que se les brinda a los pacientes, ya que es una de las tareas fundamentales de los doctores y la base para una terapéutica eficaz. En sí mismo no es un fin sino un medio, e indispensable para establecer el tratamiento adecuado. Hay quienes lo señalan como la parte más importante del trabajo médico, pero a pesar de eso conlleva muchas dificultades cuando se trata de enfermedades genéticas con dismorfias.

Las enfermedades genéticas con dismorfias pueden considerarse variaciones de la forma o tamaño normales y a diferencia de las malformaciones, generalmente no implican un mal funcionamiento del órgano, aunque a veces son consecuencia de malformaciones de órganos internos. Las dismorfias se

localizan preferentemente en la cara, los genitales y en la parte distal de las extremidades, manos y pies. En el presente trabajo de diploma la base de conocimiento estará enfocada en la región de la cara.

Son muchas las enfermedades genéticas con dismorfias, las cuales no se manifiestan igual en cada individuo, por lo que se les atribuye numerosos signos clínicos. Es difícil para los propios médicos definir una enfermedad específica cuando analizan un paciente, debido a que varios signos casi nunca arroja una sola enfermedad, sino varias. Estos genetistas son expertos en esta área, pero por sí solos es una fracción muy pequeña de conocimiento en la naturaleza de las enfermedades genéticas con dismorfias; porque en su estudio durante toda la carrera sólo ven algunas, pero no todas.

En el mundo existen aplicaciones que recogen el conocimiento de numerosos expertos en el área de enfermedades genéticas, las cuales disponen de una gran base de datos. Pero Cuba no puede hacer uso de éstas, porque el software y sus actualizaciones requieren de grandes sumas de dinero y su volumen es completamente en inglés.

Los genetistas cubanos no cuentan de una aplicación libre, en español y publicada en la red de genética médica. El presente trabajo de diploma arrojará la solución a dicho problema con la creación de un sistema experto para el diagnóstico médico de las enfermedades genéticas con dismorfias.

Para el progreso del trabajo se determinó un equipo de desarrollo que cumplen ciertas características y cada uno de ellos dentro del equipo desarrolla un papel distinto. A continuación se detalla cada componente del equipo dentro del desarrollo y cuál es la función de cada uno:

- ***El experto***. La función del experto es la de poner sus conocimientos especializados a disposición del Sistema Experto. En la tesis que ocupa el experto es la Dra. Estela Morales Peralta, especialista en la rama de genética médica.
- ***El ingeniero del conocimiento***. Es el ingeniero que plantea las preguntas al experto, estructura sus conocimientos y los implementa en la base de conocimientos. En el presente trabajo los ingenieros del conocimiento son los autores del mismo: Marianela Gutiérrez Rodríguez y Jorge Bedoya Rusenko.

- ***El usuario***. El usuario aporta sus deseos y sus ideas, determinando especialmente el escenario en el que debe aplicarse el Sistema Experto. Este rol se le atribuye al Centro Nacional de Genética Médica.

2.2 Adquisición del conocimiento

La adquisición del conocimiento es la fase que se encarga de exponer las fuentes consultas para la extracción del conocimiento y diferentes técnicas de cómo extraer el mismo a los expertos en el dominio que ocupa.

Las fuentes de conocimiento consultadas para el desarrollo del sistema fueron:

- Expertos humanos en el dominio del problema: en dicho caso la Dra. Estela Morales Peralta.
- Libros y manuales que expongan el problema y técnicas de resolución: Patrones reconocibles de malformaciones humanas, Jones, K.L.: SMITH. 6ta Edición. Barcelona: Editorial Elsevier; 2007.

Para obtener el conocimiento se utilizaron diferentes técnicas que se expondrá a continuación:

- Se realizaron entrevistas formales e informales al experto. Los ingenieros del conocimiento establecieron un plan de reuniones, con sus objetivos y temas a tratar. Guardaron sus entrevistas en formato duro y digital, instauraron la hora, la fecha y el lugar de dicha entrevista. Aprovecharon los encuentros casuales con el experto y las llamadas sin previa planificación.
- Se confeccionó un cuestionario (Ver Anexo #3) para extraer el conocimiento de una forma organizada y completa. Ejemplos de algunas preguntas de este cuestionario:
    1. ¿Qué son las enfermedades genéticas con dismorfias?
    2. ¿Por qué realizar un sistema basado en reglas?
    3. ¿Qué es el código de McKusick?
    4. ¿Qué es la localización en el genoma?
- Se observó el trabajo real del experto visitando su consulta en el pediátrico del municipio Centro Habana. Mediante este encuentro se adquirió la realidad de lo que se va a simular.

2.3 Representación del conocimiento

Después de un análisis intensivo, por parte de los ingenieros del conocimiento, de los diferentes medios de representación con que se cuenta, se determinó cuál se adapta mejor a las condiciones del problema, y el mecanismo de solución que se utilizó. En dicho trabajo la forma de representación del conocimiento se hace a través de reglas.

La producción de reglas se expresan en un pseudocódigo equivalente al formato when - then:

```
rule "Regla1"
      when
         e:ClaseFinal( letra == "Cara amplia" && letra1=="Cara plana"  &&
         letra2=="Hirsutismo facial" && letra3=="Cara redonda" )
      then
          e.AdicionarEnf("Sindrome de Allan-Herdon");
end
```

Cada regla se identifica con un nombre que en este caso sería `"Regla1"`, seguido por la parte when de la regla. A la sección entre la parte when y then de la regla se denomina condiciones, los cuales son los signos clínicos que puedan tener determinadas enfermedades `letra == " Cara amplia "`, `letra1=="Cara plana”, letra2=="Hirsutismo facial”, letra3=="Cara redonda"` y a la parte seguida del then se le llama acciones, la enfermedad que contienen los anteriores signos clínicos, es `" Sindrome de Allan-Herdon".`

En un Sistema Basado en Reglas, el mecanismo de inferencia determina cuáles condiciones de regla, si hay alguno, queda satisfecho por las acciones. Dos métodos generales de inferencia que usan con frecuencia como estrategia para la solución de problemas con los sistemas expertos son el encadenamiento hacia atrás y el encadenamiento hacia delante.

Las reglas que se ejecutan en el motor de regla Drools lo hacen con el método encadenamiento hacia delante. Este método es el razonamiento desde las condiciones hacia las acciones que resultan de estas. El encadenamiento hacia delante es típico de sistemas en los cuales se desea conocer el valor de una variable de salida atendiendo a una serie de valores de entrada.

El problema está en la eficiencia con que se genere las reglas. Si queremos crear sistemas expertos para problemas reales que contengan cientos miles de reglas, sin importar qué tan bueno sea todo lo demás en un sistema, si el usuario tiene que esperar mucho tiempo para una respuesta, este no lo usará. Lo que en verdad es necesario es un algoritmo que conozca todas las reglas y pueda aplicar cualquier regla sin tener que probar cada una en forma secuencial.

Una solución para este problema es el algoritmo Rete desarrollado en 1979 por Charles L. El algoritmo Rete es un rápido igualador de patrones que obtiene su velocidad de almacenamiento de información sobre las reglas de una red. En lugar de tener que igualar la acción con todas las reglas en cualquier ciclo-acto de reconocimiento, el algoritmo Rete sólo busca los cambios en las correspondencias de cada ciclo. Esto acelera en gran medida la correspondencia de las acciones con las condiciones, porque los datos estáticos que no cambiaron de un ciclo a otro pueden pasarse por alto. Los algoritmos de igualado rápido como Rete completaron las bases para la aplicación práctica de los Sistemas Expertos. [5]

2.4 Implementación

En esta etapa se establece la arquitectura por la cual se regirá la implementación de la aplicación. Se combina y reorganiza el conocimiento y se hace compatible con las características de la información proporcionada por la BC obteniéndose una primera versión de un prototipo. Otro de los objetivos de esta fase es el tratamiento de errores y la seguridad que contiene el sistema. Además de exponer el estándar de codificación y el estilo de codificación establecidos en el alasMEDIGEN: Sistema Informático de Genética Médica.

2.4.1 Características del sistema

El sistema debe cumplir ciertas características que se resumen en la siguiente lista para el sistema en cuestión:

- **Apariencia o interfaz externa:** Se deben utilizar imágenes y colores identificados con el negocio del sistema. La interfaz externa debe estar diseñada para verse en cualquier resolución igual o superior a 1024x768.
- **Facilidad de uso:** La aplicación informática debe garantizar un acceso fácil y rápido, contando con un menú que satisfaga las necesidades de los usuarios. Este podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de una computadora y del ambiente web.

- **Hardware:** Para el desarrollo y ejecución de la aplicación que estará corriendo sobre dos servidores web, uno para la aplicación y el otro para brindar servicios web se necesitrá.

  Para los servidores:

  Microprocesador Pentium IV a 3.0 GHz

  1GB de RAM

  Para el Cliente:

  Microprocesador Pentium a 233 MHz (Recomendado: Pentium a 500MHz o superior).

  64 MB de RAM (Recomendado: 512MB o superior).

  52 MB de espacio de disco duro. (Recomendado: 120MB para la instalación completa del Internet Explorer 5.5 o superior).

  Conexión al servidor a través de MODEM o tarjeta de red.

- **Rendimiento:** Los tiempos de respuestas deben ser los más rápidos al igual que la velocidad de procesamiento de la información.
- **Software:**

  Para el servidor web:

  Se requiere para el funcionamiento del sistema disponer de un servidor web que cuente con el apache-tomcat-6.0 o superior y que tenga las librerías necesarias para el funcionamiento del servicio.

  Para el servidor de aplicaciones:

  Se requiere para el funcionamiento del sistema disponer de un servidor que cuente con Apache 2.0.

  Para el cliente:

  Los usuarios del sistema deberán contar con un navegador Internet Explorer 5.5 o Mozilla Firefox 2.0 o superior, para poder acceder a las opciones que brinda el sistema.

- **Disponibilidad:** Se debe garantizar el funcionamiento de la aplicación durante las 24 horas del día y los siete días de la semana, con el menor tiempo posible de recuperación de fallos. Se deben crear copias de respaldo periódicas que puedan restaurar el sistema en caso de fallo crítico o pérdida total de la información.

2.4.2 Arquitectura SEGEDIS

En el proyecto de Genética Médica se establece el framework Symfony que utiliza el patrón Modelo-Vista-controlador (MVC). Este framework obliga a dividir y organizar el código de acuerdo a las convenciones establecidas por el mismo. El código de la presentación se guarda en la vista, el código de manipulación de datos se guarda en el modelo y la lógica de procesamiento de las peticiones constituye el controlador.

Symfony toma lo mejor de la arquitectura MVC y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo. [23]

La vista se encarga de producir las páginas que se muestran como resultado de las acciones, aquí se encuentra el layout, que es común para todas las páginas. La vista en Symfony está compuesta por diversas partes, estando cada una de ellas especialmente preparada para que pueda ser fácilmente modificable por la persona que normalmente trabaja con cada aspecto del diseño de las aplicaciones, asegurando la correcta combinación del layout y la página.

En el controlador se encuentran las acciones, siendo estas el corazón de la aplicación, ya que contienen toda la lógica de la aplicación. Las acciones utilizan el modelo y definen variables para la vista. Cuando se realiza una petición web en una aplicación Symfony, la URL define una acción y los parámetros de la petición.

En el Modelo se encuentran las clases, las cuales se generan de forma automática en dependencia de la estructura de la BD. En Symfony, el acceso a los datos, se realiza mediante objetos. Propel es el motor generador que se encarga de esta generación automática para construir sus clases, creando la estructura y generando el código de las mismas.

El presente trabajo de diploma se rige por la arquitectura antes expuesta, con la adición de otros componentes introducidos a raíz de la construcción de un SE, que inducen ciertas características en la conformación de la arquitectura de un proyecto. La figura 2.1 brinda una visión de ésta arquitectura.

Las peticiones que hacen los usuarios a través de la interfaz de usuario definen la dirección de las mismas, bien sea hacia la base de datos (BD) o hacia un motor de reglas que recibe los parámetros a través de un servicio web. Este servicio web es un archivo de extensión .wsld, el cual contiene un método público que se consume a través del lenguaje php desde Modulo1/Action.Class. El servicio web es generado desde la herramienta Eclipse mediante el lenguaje php.

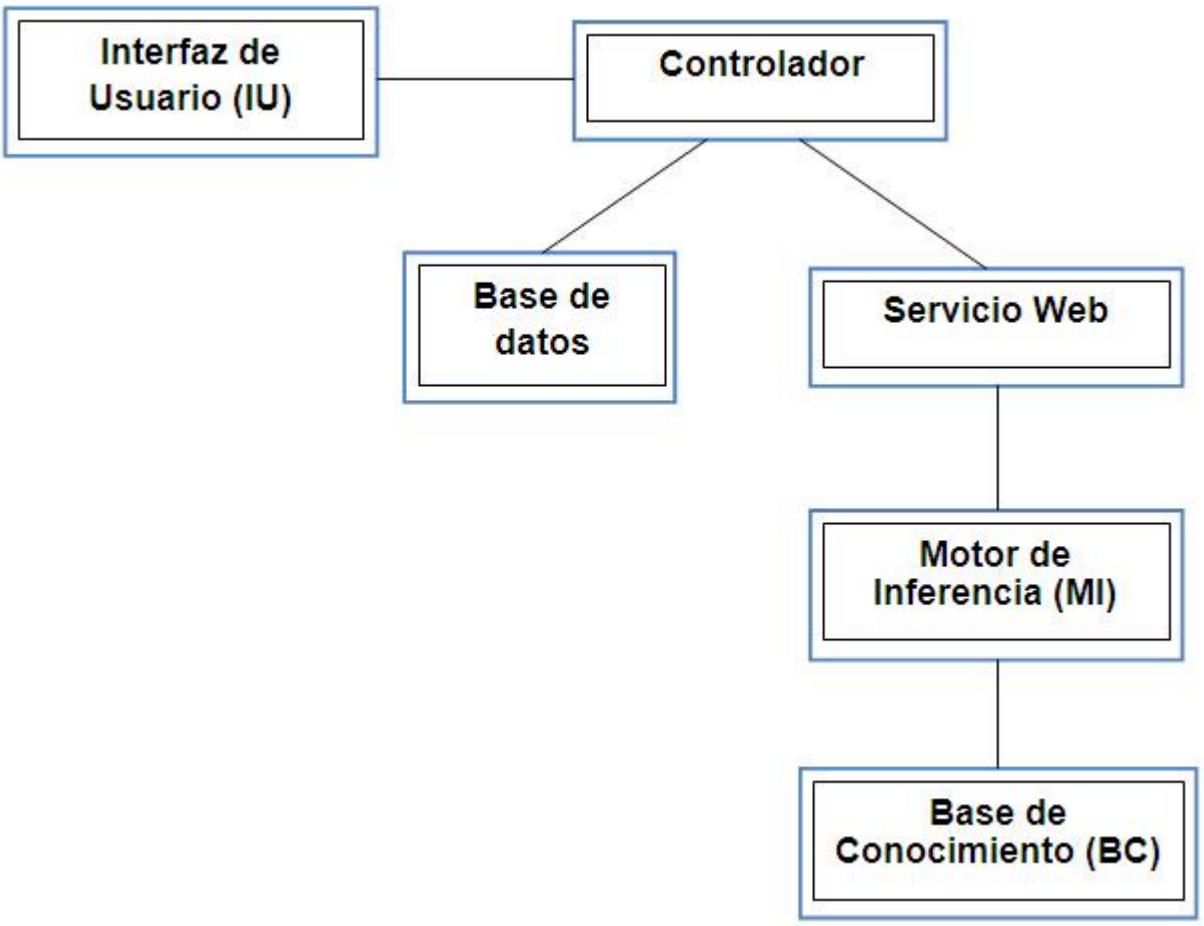


Figura 2.1 Arquitectura de SEGEDIS.

La herramienta Eclipse posee un plug-in que permite trabajar con Drools, en donde se crean las reglas que se utilizan en este Sistema Experto. El motor de reglas Drools, posee un motor de inferencia que compila las reglas para llegar a conclusiones, mediante la búsqueda que hace a la base de conocimiento. La base de conocimiento está compuesta por archivos de reglas en formato (.drl). En estos archivos se especifican las condiciones, acciones y funciones de las reglas expresadas en el lenguaje Java.

La tesis que ocupa posee una pequeña base de datos (BD) que está compuesta por cuatro tablas, las cuales se representan en un modelo entidad-relación (Ver figura 2.2). Esta BD permite poder mostrar todas las enfermedades genéticas con dismorfias y los signos clínicos que están presentes en estas enfermedades con sus respectivas características. La BD también posee una tabla árbol de signos

clínicos con la posibilidad de eliminar y agregar un signo clínico y enviar estos hacia los criterios de búsqueda. En el anexo #4 se muestran las descripciones de cada una de estas tablas.

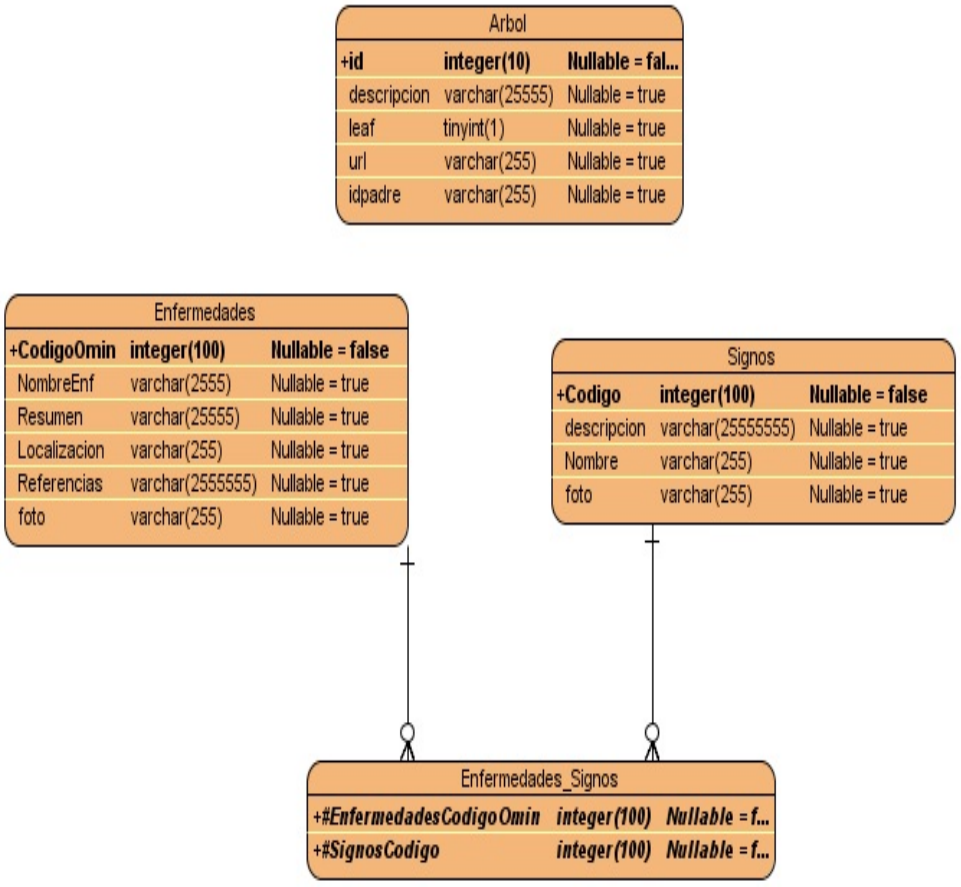


Figura 2.2 Modelo Entidad-Relación.

2.4.3 Pautas del diseño

1. Los grid que se crean, con un ancho de 893 px y un alto de 410 px con un borde de 1 px.
2. La primera fila del grid debe contener el nombre y en la segunda el código.
3. Para visualizar los detalles de cada nombre se utiliza un plug-ins.
4. El filtrado que se realiza en cada grid aparecerá en la parte superior izquierda del grid.
5. Los botones para efectuar operaciones se ubicarán en la parte superior derecha.
6. Para representar campos que deben ser de entrada obligatoria se colocarán un asterisco al lado derecho del componente.

7. Los mensajes de error ocurridos durante la validación de los campos se mostrarán en la parte superior del campo validado en el cual ocurrió el error.
8. Los estilos a usar en los componentes web son los que tiene el framework ExtJs por defecto.
9. Los estilos que se utilizan para el login son:

Componentes	Estilos
TextBox	entradaplana7
Botones	sbttn

2.4.4 Aplicación de algunos patrones GRASP en Symfony

Son muchos los patrones que se aplican en la implementación con Symfony, a continuación se mencionan algunos ejemplos de los evidenciados, ubicándolos en las capas de Modelo y Control que plantea el patrón arquitectónico MVC.

Experto: Es uno de los patrones más utilizados al trabajar con Symfony, con la inclusión de Propel para mapear la BD se justifica de algún modo. Propel genera las clases para la gestión de las entidades con las responsabilidades asignadas correctamente según el patrón Experto, las clases de abstracción de datos (Peer del Modelo) cuentan con un conjunto de funcionalidades relacionadas directamente con la entidad que representan y contienen la información necesaria de la tabla que representan.

Creador: En la clase enfermedadesAction se encuentran las acciones definidas para el módulo1. En dichas acciones se crean los objetos de las clases que son los que instancian a la clase del módulo1 lo que evidencia que la clase enfermedadesAction es “creador” de dichas entidades. Ejemplo del uso de este patrón es en la acción executeGetGrid mediante la creación de las instancias de las clases entidades que contienen los datos de las enfermedades. Ejemplo de una función utilizada en la clase enfermedadesAction: doSelect().

Alta Cohesión: El trabajar con Symfony permite la organización del trabajo en cuanto a la estructura del proyecto, esto proporciona crear y trabajar con clases con una alta cohesión. Ejemplo de esto se evidencia en la clase modulo1Actions, la cual está formada por diferentes funcionalidades que se

encuentran estrechamente relacionadas, teniendo un sentido común y un propósito único, siendo las mismas las encargadas de controlar las acciones de las plantillas.

Bajo Acoplamiento: En el modelo también se encuentran las clases que implementan la lógica de negocio y de acceso a datos, estas clases no tienen asociaciones con las de la vista o el controlador por lo que la dependencia en este caso es baja, poniéndose de manifiesto este patrón.

Controlador: Este patrón se observa en las clases sfFrontController, sfWebFrontController, sfContex, los "actions" y el index.php del ambiente. La arquitectura del framework (MVC) ayuda desde el principio, existiendo una capa específicamente para los controladores, que son el núcleo del mismo, el puesto de mando.

2.4.5 Aplicación de algunos patrones GoF que implementa Symfony

Patrón Decorador

Este método corresponde a la clase abstracta sfView, padre de todas las vistas, las que contienen un decorador para permitir añadir funcionalidades dinámicamente.

El archivo llamado layout.php es el que contiene el Layout de la página. Este archivo, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el layout, o si se mira desde el otro punto de vista, el layout decora la plantilla. Este comportamiento es una implementación del patrón de diseño llamado "Decorator".

Figura 2.3 Planilla decorada con un layout.

Patrón Registry

El patrón Registry, pese a su simplicidad, es un patrón sumamente útil para los desarrolladores en la programación orientada a objetos (POO). En resumidas palabras, el Patrón Registry, es un medio simple y eficiente de compartir datos y objetos en la aplicación sin tener que preocuparse de mantener numerosos parámetros o hacer uso de variables globales. La aplicación de este patrón se ve en la clase sfConfig la cual es la encargada de almacenar todas las variables de uso global en la aplicación, decir también que esta clase aplica el patrón Singleton.

Symfony también aplica el patrón "Front Controller" (Controlador frontal) y por tanto tiene una estructura bien organizada de controladores, que parte desde el "index.php" del ambiente y terminan en los "Actions". Aquí cada clase en esta capa tiene su responsabilidad y es única, hay controladores que se encargan de la seguridad del sistema trabajando con ficheros YML, otros sólo velan por identificar mediante unos datos las clases que deben realizar determinadas tareas (Patrón GoF Command, clase sfRouting), las clases relacionadas con la configuración del sistema (sfConfig, y sfConfigHandler).

2.4.6 Estándar de codificación

Los lenguajes de programación son herramientas que permiten crear programas y software. La manera de escribir código es variable por los programadores, especialmente en PHP, lo que provoca que la próxima persona que trabaje con los módulos, opinará que no es ordenado y de esta forma se atrasará el trabajo. Un estándar de codificación comprende los aspectos de la generación de código y repercute directamente en la legibilidad y la extensibilidad de cualquier proyecto de software, haciendo que nuevos desarrolladores se acoplen rápidamente al proceso de desarrollo. Además son reglas específicas a una lengua que reducen perceptiblemente el riesgo de que los desarrolladores introduzcan errores. Los estándares de codificación no destapan problemas existentes, evitan más bien que los errores ocurran. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico, por estas razones siempre un proyecto debe definir los estilos de codificación, para así tener una estandarización en el proyecto.

Estilo de codificación para el SEGEDIS

1. Todas las etiquetas php deben ser completas (<?php?>)... no reducidas (<? ?>).

2. Todas las variables deberían ser inicializadas o, al menos, comprobada su existencia utilizando isset () antes de ser utilizadas.

3. La sangría del texto debe ser siempre de 4 espacios. No utilices el tabulador (todos los editores no interpretan el Tab de la misma manera).

4. Los nombres de las variables y funciones tienen que ser siempre fáciles de leer, procurando que sean palabras en minúsculas con significado claro. Si realmente necesita más de una palabra, póngalas juntas, poniendo la inicial de cada palabra en mayúscula siempre que no sea la primera, pero procure mantenerlas tan breves como sea posible. Utilice nombres en plural para arreglos o matrices de objetos. Ejemplos: $edad, $fechaNacimiento, $personas.

5. Los bloques de código siempre deben estar encerrados por llaves (incluso si sólo constan de una línea) y correctamente alineados (a 4 espacios).

6. Las cadenas tienen que ser definidas utilizando comillas simples siempre que sea posible, para obtener un mejor rendimiento.

7. Todas las funciones y clases deben estar comentadas. Los comentarios deben ser añadidos de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones o variables. [13]

2.4.7 Seguridad

El framework Symfony, que es el utilizado en el presente trabajo de diploma, garantiza la seguridad al ser ejecutada cada acción a través de un filtro verificando si el usuario autenticado tiene los privilegios necesarios a la acción en cuestión, como por ejemplo:

- Las acciones requieren que los usuarios estén autenticados.
- La autenticación y las credenciales son privilegios agrupados bajo el nombre y simplifican la creación de secciones restringidas y la gestión de la seguridad de usuario en grupos.

Para garantizar la seguridad en el sistema, este contará con un módulo para la gestión de la autenticación. Cuando un usuario se autentica en el sistema se crean automáticamente un grupo de variables temporales que permiten al programador comprobar en cada momento que nivel y que permisos tiene

cada usuario. Para el control de estos permisos se crearon permisos y grupos de usuarios. Específicamente para este trabajo todas las acciones requieren que los usuarios estén autenticados en el sistema para acceder a ella, existen los grupos administrador y genetista que cuentan con los permisos todos los permisos y algunos permisos respectivamente. Los usuarios que pertenezcan al grupo de genetista sólo tienen posibilidad de realizar la búsqueda de enfermedades dado los criterios y tendrán permiso también a visualizar todas las enfermedades y signos que existen. Los usuarios del grupo de administradores tienen la posibilidad agregar un nuevo signo al árbol y la posibilidad de eliminar cualquier signo del árbol, además de todos los permisos que posee los genetistas.

2.4.8 Interfaces de la aplicación

Para la realización de esta aplicación se desarrolló una interfaz amigable y segura, acorde con el personal que trabajará con la misma.

Esta interfaz permite la autenticación del usuario, así sea como administrador o como genetista. (Ver figura 2.4)

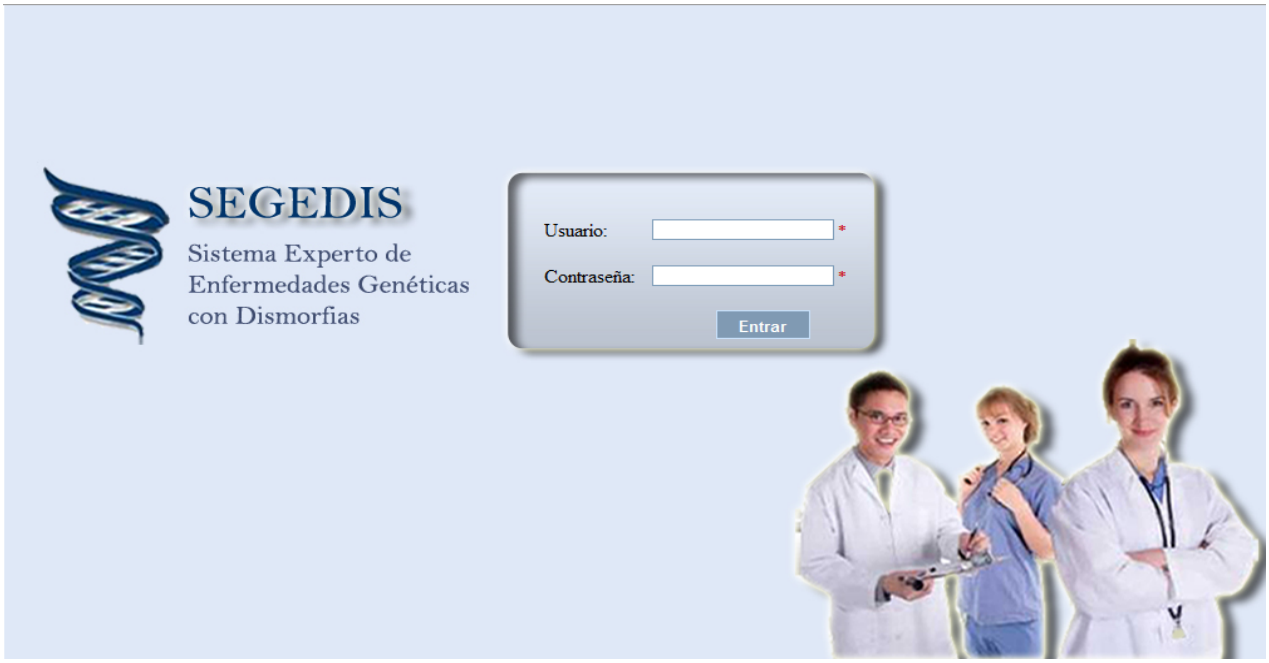


Figura 2.4 Autenticación.

Cuando el usuario es autenticado se le muestra una interfaz principal, por la cual le permite acceder a todos los servicios que este posibilita. Además cada interfaz muestra en la parte izquierda el menú del sistema y en la parte superior derecha el tipo de usuario que es y los permisos que posee.

Si se elige en el menú, “Enfermedades Genéticas con Dismorfias” se muestra un listado de estas enfermedades facilitando ver los detalles de cada una. Cuando se muestran los detalles de cada enfermedad aparece la opción “Sus signos”, la cual genera una ventana con todos los signos que esta contiene. Además permite realizar un filtrado de las enfermedades. (Ver figura 2.5)

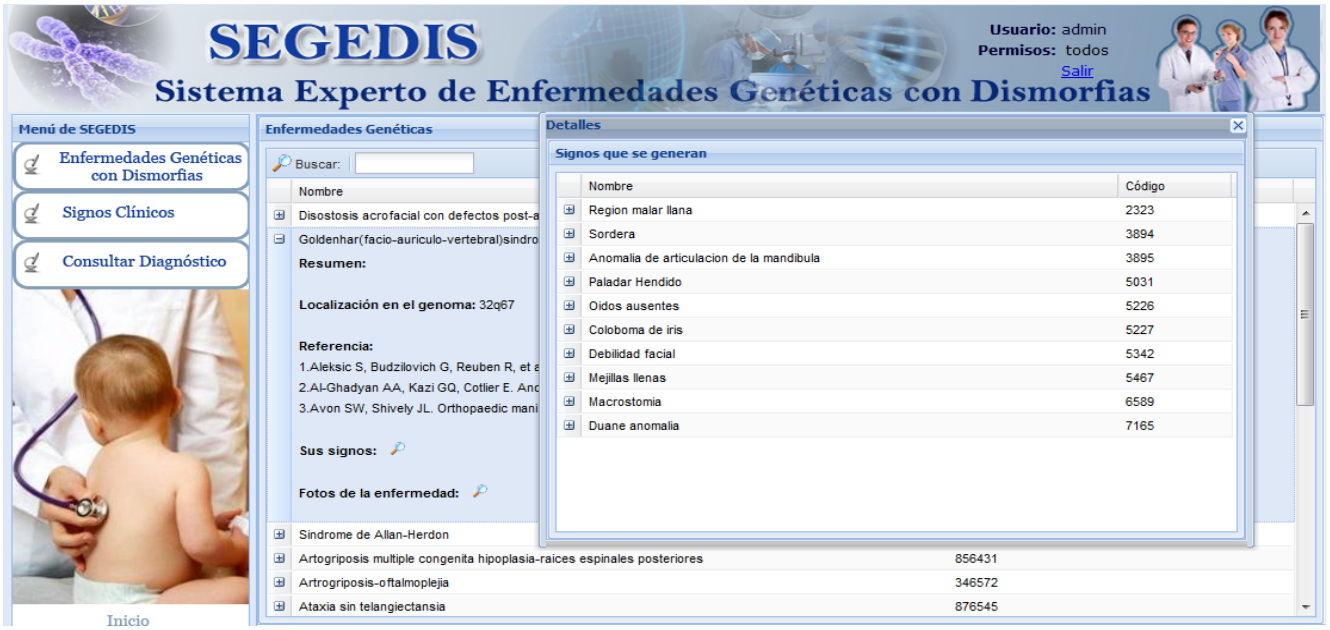


Figura 2.5 Interfaz de enfermedades genéticas con dismorfias.

Seguidamente en la misma interfaz, en los detalles de las enfermedades aparece la opción de “Fotos de las enfermedades”, las cuales se muestran en otra ventana. (Ver figura 2.6)

Figura 2.6 Interfaz de fotos de cada enfermedad.

En la interfaz de “Signos Clínicos” muestra una lista de los signos con sus características y las enfermedades que presentan el marcado signo, con sus respectivos detalles. (Ver figura 2.7)

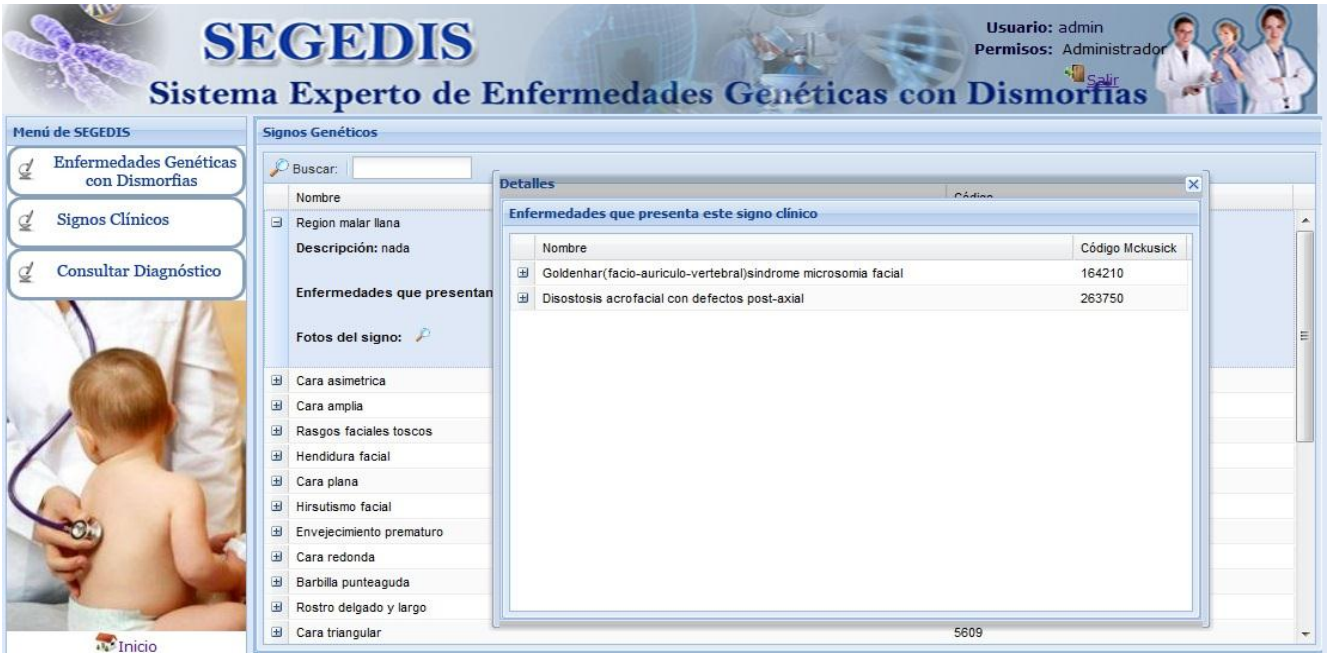


Figura 2.7 Interfaz de signos clínicos.

Esta interfaz corresponde a “Consultar Diagnóstico” en el cual se genera un árbol de signos clínicos que al dar clic derecho sobre ellos permite la opción de agregarlo a los criterios que aparecen en la parte derecha para generar una búsqueda de enfermedades que posean estos signos. (Ver figura 2.8)

Figura 2.8 Interfaz consultar diagnóstico.

Luego se realiza la búsqueda de las enfermedades genéticas con dismorfias y se muestra una ventana con las enfermedades encontradas en la base de conocimiento. (Ver figura 2.9)

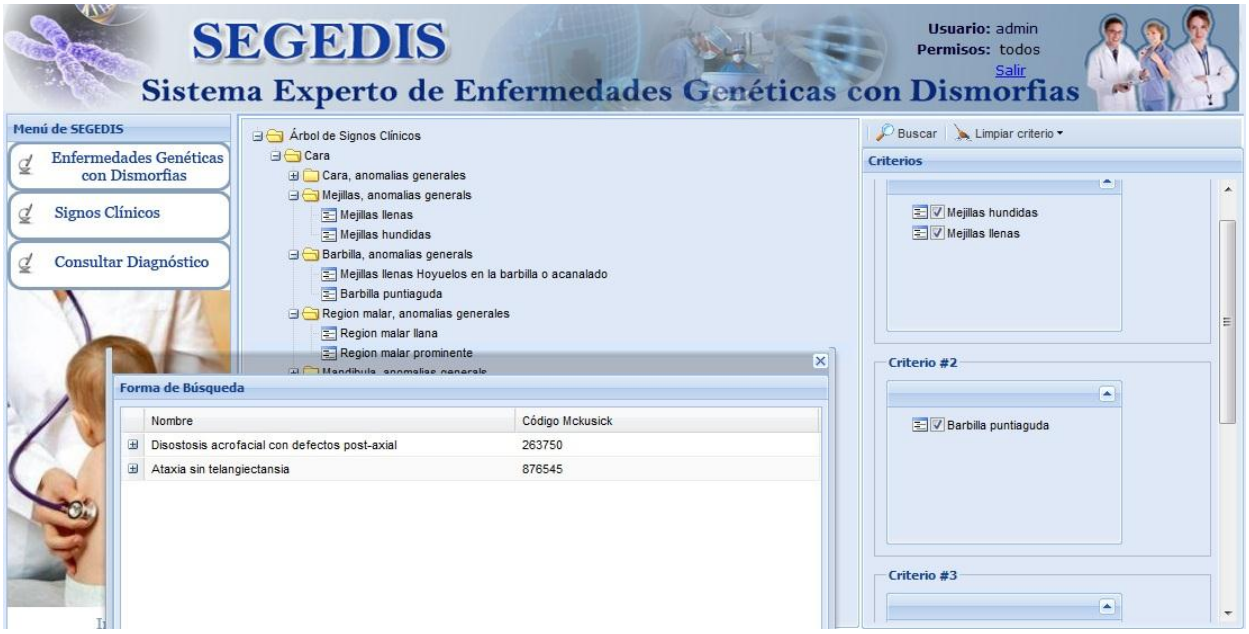


Figura 2.9 Interfaz consultar diagnóstico con el resultado de la búsqueda.

Conclusiones parciales

En este capítulo se puso en práctica la metodología para la construcción de un Sistema Experto, que desglosada por fases, se explicaron las técnicas de adquisición del conocimiento, su forma de representarlo, que en la tesis que ocupa es en forma de reglas. En esta fase se puso ejemplo de la estructura de una regla y se argumentó el algoritmo utilizado por el motor de inferencia. En la fase de implementación se determinaron algunos subepígrafes como pautas del diseño, algunos patrones GRASP y GoF que utiliza Symfony ,estándares de codificación, seguridad e imágenes de los prototipos de interfaz que están contenidos dentro de la arquitectura de SEGEDIS.

CAPÍTULO 3: RESULTADOS Y DISCUSIÓN

Introducción

En este capítulo se exponen los aspectos esenciales de la Validación y Verificación de los Sistemas Expertos. Se ejemplifican las validaciones que se les hace a la aplicación, mediante el framework Symfony. Se muestran algunos ejemplos de estas validaciones tanto en el código en php como en la misma interfaz. Se describen los atributos que componen la facilidad de uso, ilustrándolos en diferentes pruebas aplicadas a la interfaz gráfica del SEGEDIS.

3.1 Validación y Verificación de los Sistemas Expertos

En la medida en que se deseen construir SE cada vez de mayor calidad, será necesario avanzar en el campo de la Validación y Verificación (V&V). El término calidad comprende distintos factores, tales como la fiabilidad, la robustez, la precisión, la eficiencia, la facilidad de uso y otras. Ahora bien, a la hora de evaluar la calidad de un SE, se afronta, muy a menudo, la ausencia de una especificación de requisitos completa y precisa, que marque las pautas de cómo debe ser el producto final. Esto es debido a que los problemas que se tratan de resolver con SE, dado su carácter inédito, suelen ser problemas difícilmente estructurales en las primeras etapas del desarrollo. La estructura de uno de estos problemas se va conociendo sobre la marcha, al mismo tiempo que se va conociendo la estructura de la solución. De ahí que el ciclo de vida en cascada que se utiliza a menudo para el software convencional no sea adecuado para el desarrollo de un SE, y que éste se plantee, más bien, de una manera evolutiva, normalmente en la forma de prototipo rápido.

Existen algunas otras diferencias entre los SE y los sistemas basados en software convencional, que impiden utilizar técnicas de V&V de software convencional con el fin de validar un SE. Por ejemplo, como consecuencia de que no siempre es posible disponer de una especificación de requisitos completa para un SE, las salidas que va a suministrar el sistema no siempre se van a poder licitar con los requisitos. [24]

En el presente trabajo de diploma se expondrán diferentes atributos de la facilidad de uso que determinan el nivel de calidad que posee el mismo y la validación que se le hace al sistema por parte del programador. Facilidad aprendizaje, Eficiencia, Tasa de errores y Accesibilidad a la información

constituyen los atributos de la facilidad de uso que se explicarán más adelante en el epígrafe 3.1. Pero antes se mostrarán los diferentes estados de desarrollo por los cuales transita un SE. Los SE poseen una BC que su tamaño determina el estado desarrollo en que se encuentra el prototipo. A continuación se representan los estados antes dichos:

Estados en el desarrollo de un SBR

1. Prototipo demostrativo: resuelve una porción del problema. La BC tiene entre 50 y 100 reglas y demora aproximadamente 3 meses en realizarla.
2. Prototipo de investigación: El conocimiento almacenado en la BC cubre todo el dominio, aunque no está probado. Contiene entre 200 y 500 reglas en un tiempo de 1 a 2 años.
3. Prototipo de campo: El SE muestra buenos resultados funcionando extensivamente en ambiente del usuario. Posee entre 500 y 1000 reglas en un tiempo de 2 a 3 años.
4. Módulo de producción: El SE muestra una alta calidad, confiabilidad, rapidez y eficiencia en su trabajo. Contienen entre 500 y 1500 reglas en un tiempo de 2 a 4 años.
5. Sistema comercial: Es un SE que está listo para ser comercializado y que tarda 5 o 6 años terminarlo.

Como se observó anteriormente el tamaño de la BC determina el tipo de prototipo en el que se encuentra el sistema. El SEGEDIS posee aproximadamente 80 reglas por lo que se coloca como un prototipo demostrativo. Cuando el sistema pasa a otro estado de desarrollo su BC es más grande por lo que se le realizan diferentes pruebas. Una de ellas es la validación contra el experto humano, que este analiza las salidas del sistema y las compara con su razonamiento, para llegar a comprobar el SE. A veces un solo experto no es suficiente para realizar estas pruebas por lo que se utilizan las validaciones contra un grupo de expertos y contra un conceso de expertos; pero estas tres formas de validar el SE poseen ventajas y desventajas, por lo que la utilización de una en específico debe ser valorado por el Ingeniero del Conocimiento.

Dependiendo del tipo de prototipo, la verificación puede centrarse en la Base de Conocimiento, el motor de inferencia, o la interfaz de usuario. Como la BC del SEGEDIS aún es pequeña no se les hace las pruebas correspondientes con el experto humano por eso es que en posteriores epígrafes la atención se

centra en atributos de calidad que se miden en el sistema implementado y que se centran en los tiempos de respuestas de la interfaz gráfica.

3.2 Validación de la aplicación a nivel de desarrollador

Sin darse cuenta cada desarrollador es un probador en potencia, un especialista en calidad, y lo que logra con ello es validar la aplicación sobre la marcha. Similar a la forma de redactar un pseudocódigo, cada programador tiene su estilo propio para validar sus aplicaciones y comprobar que el camino por el que se conduce lo llevará a una solución satisfactoria.

Teniendo en cuenta el lenguaje de programación utilizado y el framework de desarrollo, los cuales no cuentan con robustos compiladores en tiempo de diseño que permitan la rápida detección y mitigación de los errores se hace necesaria la aplicación de los conocimientos, la experiencia y la perspicacia del programador para detectar estos errores de la forma más rápida.

PHP es un lenguaje interpretado, esto significa que las instrucciones son interpretadas y ejecutadas en tiempo de ejecución, no media un lenguaje de objeto como en los lenguajes C++ y C#. Por tanto de más está decir que depurar una aplicación web es tarea más que compleja. En ocasiones se hace necesario comprobar si determinada función se está ejecutando, o si cierto algoritmo está alcanzando un punto determinado, en estos casos se puede auxiliar el programador de la función renderText() de php que envía en forma de texto al navegador la variable que se le pasa a la función. (Ver figura 3.1)

```
public function executeGetTree() {

    $arr= array();
    $arrTemp= array();

    $c = new Criteria();
    $c->add( TreePeer::IDPADRE, $this->getRequestParameter('node'));

    $tree = TreePeer::doSelect($c);

    foreach ($tree as $obj) {
        $arrTemp['id']          = $obj->getId();
        $arrTemp['text']        = $obj->getDescripcion();
        $arrTemp['leaf']        = $obj->getLeaf();

        $arr[]=$arrTemp;
    }

    return $this->renderText(json_encode($arr));

}
```

Figura 3.1 Código de la función renderText().

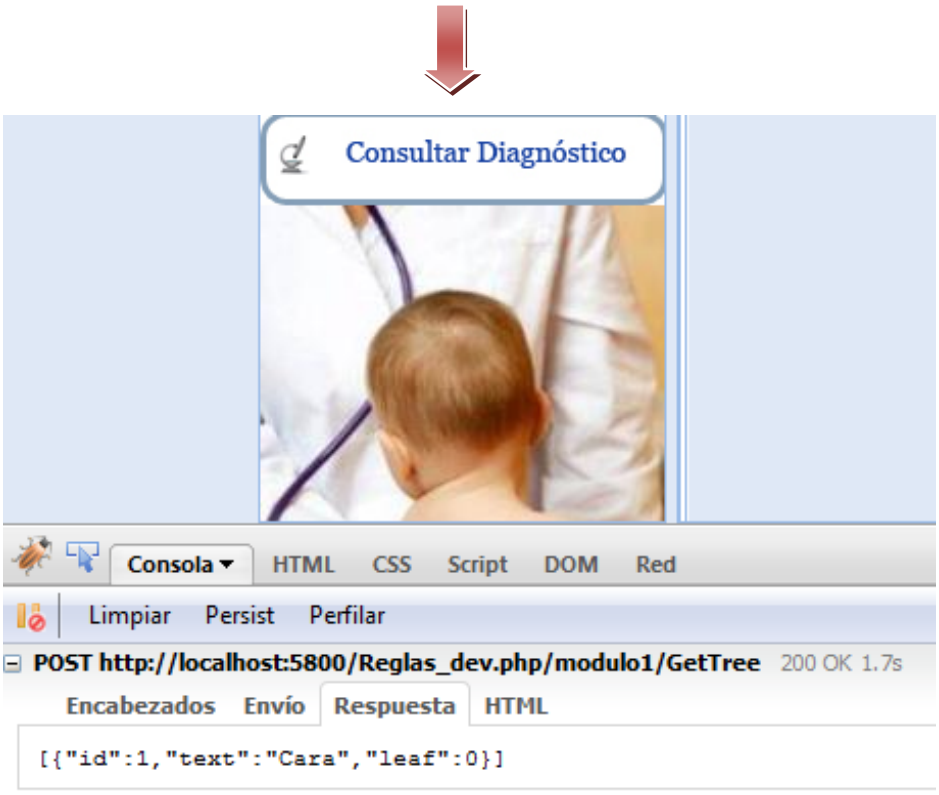


Figura 3.2 Consola de firebug.

El plug-in firebug 1.5.3 que se representa con el icono que aparece en la figura 3.3 permite visualizar tanto los parámetros que se envían al controlador y la respuesta que este proporciona a través de la función renderText() sin tener que modificar la interfaz . Este icono aparece en la parte inferior derecha del Firefox que al ejecutarse aparece la consola del firebug. (Ver figura 3.2)

Figura 3.3 Plug-in firebug 1.5.3.

La introducción de datos incorrectos puede ser otra manera de verificar la consistencia de la aplicación que se está desarrollando, un adelanto y ayuda para la etapa de pruebas. (Ver figura 3.4)

Figura 3.4 Validación de la interfaz.

Desconfigurar un fichero, cambiarle el nombre, incluso eliminar un registro pueden constituir métodos, que aunque poco recomendados pueden aclarar y facilitar la solución de ciertos problemas. (Ver figura 3.5)

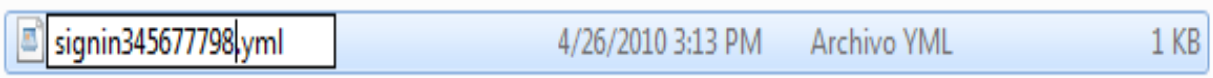


Figura 3.5 archivo .yml.

Cambiar el nombre de un fichero .yml para desvincularlo de su 'template' puede esclarecer el origen de muchos problemas que se dan a la hora de las validaciones.

3.3 Facilidad de uso de la interfaz gráfica

En el desarrollo de software se identifica a menudo la facilidad de uso con las características de los elementos de una interfaz gráfica de usuario basada en ventanas, como puede ser su color, su disposición o el diseño gráfico de los iconos y animaciones. Sin embargo, la facilidad de uso no sólo tiene que ver con la interfaz gráfica de usuario. La facilidad de uso de un sistema está ligada principalmente a la interacción del mismo, al modo en que se realizan las operaciones con el sistema. Esta interacción no está definida en la interfaz gráfica, sino que está imbricada en el código que implementa la funcionalidad del sistema. La interfaz gráfica de usuario es la parte visible de tal interacción.

La facilidad de uso es un factor de calidad demasiado abstracta como para ser medida directamente. Para poder estudiarla se descompone habitualmente en los siguientes cuatro atributos básicos:

- **Facilidad de aprendizaje**: Cuán fácil es aprender a navegar por el sistema, como para ser capaz de realizar correctamente la tarea que desea realizar el usuario. Se mide normalmente por el tiempo empleado con el sistema hasta realizar ciertas tareas en menos de un tiempo dado (el tiempo empleado habitualmente por los usuarios expertos).[25]

En el presente trabajo se puso en práctica este atributo. Se escogió un usuario con poco dominio del problema para que interactuara con la aplicación y en 3 minutos pudo acceder a las tareas deseadas, casi el mismo tiempo que lo hace un usuario experto.

- **Eficiencia**: El número de funcionalidades por unidad de tiempo que el usuario puede realizar usando el sistema. Lo que se busca es la máxima velocidad de realización de tareas del usuario. Cuanto mayor es la facilidad de uso de un sistema, más rápido es el usuario al utilizarlo, y el trabajo se realiza con mayor rapidez. Nótese que eficiencia del software en cuanto su velocidad de proceso no implica necesariamente eficiencia del usuario en el sentido en el que aquí se ha descrito.[25]

En el SEGEDIS se utilizó el plug-in firebug para determinar el tiempo en que se ejecuta una acción. En esta ocasión se utilizó un usuario experto, en la tabla que se muestra a continuación se reflejan el número de funcionalidades por segundos que realizó este usuario.

Funcionalidades	Segundos
Listar enfermedades genéticas con dismorfias	2,70
Listar signos clínicos	2,95
Cargar la página Consultar Diagnóstico	1,43
Hacer una búsqueda de enfermedades a partir de signos clínicos	3,40

Tabla 3.1 Velocidad del SEGEDIS.

Como este sistema es una aplicación web, se da el caso de que varios usuarios estén conectados a la vez y ejecuten diferentes funcionalidades al mismo tiempo, sobrecargando el sistema, seguidamente se exponen varias tablas con diferentes tipos de usuarios realizando diferentes funcionalidades y el tiempo que demoran al ejecutarlas, pero antes se define las diferencias de los usuarios en tres niveles.

Niveles	Usuarios	Descripción
1	Usuario avanzado	Conocimiento del dominio y de informática
2	Usuario medio	Sabe algo del dominio y de informática
3	Usuario bajo	No sabe nada

Tabla 3.2 Niveles de los usuarios.

Ejemplo para usuario de nivel 1.

Funcionalidades	Usuario 1	Usuario 2	Usuario 3
Listar enfermedades genéticas con dismorfias	2,3	2,1	2,6
Listar signos clínicos	2,8	2,3	2,2
Cargar la página Consultar Diagnóstico	0,3	0,5	0,7
Hacer una búsqueda de enfermedades a partir de signos clínicos	5,2	5,1	5,8

Tabla 3.3 Ejemplo para el usuario de nivel 1.

Ejemplo para usuario de nivel 2.

Funcionalidades	Usuario 1	Usuario 2	Usuario 3
Listar enfermedades genéticas con dismorfias	3,5	3,1	3,4
Listar signos clínicos	2,8	3,0	3,7
Cargar la página Consultar Diagnóstico	0,3	0,5	0,7
Hacer una búsqueda de enfermedades a partir de signos clínicos	8,3	8,1	8,7

Tabla 3.4 Ejemplo para el usuario de nivel 2.

Ejemplo para usuario de nivel 3.

Funcionalidades	Usuario 1	Usuario 2	Usuario 3
Listar enfermedades genéticas con dismorfias	5,3	5,1	4,8
Listar signos clínicos	5,6	5,2	5,5
Cargar la página Consultar Diagnóstico	0,3	0,5	0,7
Hacer una búsqueda de enfermedades a partir de signos clínicos	10,3	11,1	10,8

Tabla 3.5 Ejemplo para el usuario de nivel 3.

En las anteriores tablas se evidenció las diferencias de tiempos por cada grupo de usuarios al navegar por diversas funcionalidades del sistema. El usuario de nivel uno que pertenece a un usuario avanzado tiene los mejores tiempos es decir que en menos segundos logró llegar a las funcionalidades deseadas pero estos resultados no están muy alejados de los tiempos del usuario de nivel tres que no tiene conocimiento del dominio. Todo esto se debe a la buena facilidad de uso que posee el SEGEDIS.

- **Tasa de errores**: Este atributo contribuye de forma negativa a la facilidad de uso de un sistema. Se refiere al número de errores cometidos por el usuario mientras realiza una determinada tarea. Un buen nivel de facilidad de uso implica una tasa de errores baja. Los errores reducen la eficiencia y satisfacción del usuario, y pueden verse como un fracaso en la transmisión al usuario del modo de hacer las cosas con el sistema.[25]

En el SEGEDIS se determinan una cierta cantidad de características que le dan las funcionalidades al sistema. A continuación se mostrará una comparación entre la cantidad de características que posee el sistema y la cantidad de errores que el usuario puede cometer al ejecutarlas.

Características del SEGEDIS (# C)

1. Autenticarse.
2. Buscar una enfermedad a partir de signos clínico chequeados.
3. Buscar una enfermedad en la lista de enfermedades genéticas con dismorfias.
4. Buscar un signo clínico.
5. Limpiar todos los criterios.
6. Adicionar un signo clínico a un criterio.
7. Eliminar un signo clínico de un criterio.
8. Adicionar un signo clínico al árbol de signos.
9. Chequear un criterio para buscar enfermedades genéticas con dismorfias.
10. Chequear un criterio para ser eliminado.

Errores que pueden ser cometidos por el usuario al interactuar con el SEGEDIS (# E).

1. Error al autenticarse.
2. Buscar enfermedades sin haber chequeado el signo clínico.
3. Eliminar un signo clínico sin ser chequeado.

La cantidad de características del SEGEDIS es mayor que el número de errores que puede cometer el usuario al navegar por el sistema lo que proporciona una visión de la buena facilidad de uso que el SEGEDIS posee ya que el usuario tiene menos riesgo a equivocarse al navegar por el sistema.

 # C > # E

Un elemento que posee el SEGEDIS para evitar los errores del usuario son los tooltips también llamados descripción emergente, es una herramienta de ayuda visual que funciona al situar o pulsar con el ratón sobre algún elemento gráfico, mostrando una ayuda adicional para informar al usuario de la finalidad del elemento sobre el que se encuentra.

- **Accesibilidad a la información:** La interfaz al usuario es el elemento más débil pero también uno de los más críticos en un SE ya que determina que tan bien los sistemas serán aceptados por los usuarios. Una buena accesibilidad a la información está determinada por la facilidad con que

puedan acceder a cualquier información disponible en el sitio web mediante un máximo de 3 clics desde la portada. Así mismo se deben proporcionar enlaces que permitan volver atrás pasando por las secciones previas, creando de esta forma una lógica estructura de árbol. Su objetivo es proporcionar una navegación simple y efectiva, que no obligue a los usuarios a perderse en un complejo entramado de enlaces sin sentido estructural y mejorar la accesibilidad a la información y minimizar las dificultades a su obtención.[25]

Un ejemplo de esto en el SEGEDIS se muestra en la interfaz “Consultar Diagnóstico” que con un primer clic se adiciona el signo clínico al criterio, con un segundo clic se escoge la opción buscar y se muestra una ventana con los signos clínicos seleccionados. Finalmente dando un tercer clic se escoge la opción aceptar y se muestra las enfermedades resultantes de la búsqueda. (Ver figura 3.5)

Figura 3.5 Principios de utilidad: La regla de los 3 clics.

Conclusiones parciales

En este capítulo se ejemplificaron las validaciones realizadas a la aplicación. Se mostraron las imágenes de segmentos de código en php donde se visualizan mediante la función renderText() estas validaciones. Otro método representado de comprobación del sistema es el plug-in firebug insertado en el navegador firefox, el cual permite evidenciar los parámetros que están siendo enviado. Además se explicaron los atributos que se desglosan de la facilidad de uso de la interfaz gráfica de un sistema, siendo ejemplificados en la aplicación, tales como facilidad de aprendizaje, eficiencia, tasa de errores y accesibilidad a la información.

CONCLUSIONES GENERALES

Una vez culminado el trabajo es posible afirmar que se les dio cumplimiento a los objetivos trazados para el mismo.

- Mediante el uso de técnicas de Ingeniería del Conocimiento se adquirió el conocimiento del experto en enfermedades genéticas con dismorfias.
- Se utilizaron las reglas como forma de representación del conocimiento.
- Se implementó el sistema cumpliendo las herramientas, lenguajes y tecnologías propuestas por el grupo de desarrollo, en su mayoría distribuidas bajo licencias de software libre en correspondencia con las políticas de la Universidad y del país.
- Se validaron los prototipos a través de una función en php mediante un plug-in y se realizaron pruebas de la facilidad de uso de la interfaz gráfica.

RECOMENDACIONES

Teniendo en cuenta el estado que alcanzó el sistema implementado se recomienda:

- Integrar el SEGEDIS con el producto alasMEDIGEN: Sistema Informático de Genética Médica.
- Extender la base de conocimiento hacia las demás regiones del cuerpo humano.
- Realizar las pruebas a la base de conocimiento.

REFERENCIAS BIBLIOGRÁFICAS

1. **Alvarado Garrigó, Marta y Borges Fernández, Nelvis.** Sistema Informático para la Red Nacional de Genética Médica: módulo Teleconsulta versión 2.0. 2008. Tesis de Diploma. UCI. La Habana.
2. **Criado Briz, José Mario**. Introducción a los Sistemas Expertos. [En línea] 2010. [Citado el: 13 de Noviembre de 2010.] http://www.ingenieroseninformatica.org/recursos/tutoriales/sist_exp/index.php.
3. **Hurtado Vega, José de Jesús**. Inteligencia Artificial. [En línea] 2010. [Citado el: 16 de Noviembre de 2010.] http://www.itlp.edu.mx/publica/boletines/actual/inteligencia.html.
4. **Pajares Santos, M**. Inteligencia Artificial e Ingeniería del Conocimiento. Ra-Ma, 2005. [En línea] 2008.
5. **Giarratano, Riley**. Sistema Experto, principios y programación. ISBN 7-111-10844-2.
6. **Rodríguez, Yanet**. Evaluación de un Sistema Basado en Casos Interpretativo. Tesis de Maestría. 1998.
7. **Gutiérrez, José Manuel y S. Hadi, Ali**. Libro Sistema Experto y Redes Probabilísticas.
8. **Gochez, Alejandro.** Sistemas Expertos. Un gran Avance. [En línea] lunes 28 de enero de 2008 http://inteligenciaartificialudb.blogspot.com/2008/01/sistemas-expertos-un-gran-avance.html.
9. **Gutiérrez, José Manuel**, Dpto. de Matemática. Sistemas Expertos Basados en Reglas, Aplicada, Universidad de Cantabria.
10. **Pérez González, Joel**. Sistemas Expertos, Universidad del Valle de México, Febrero 2010.
11. **Font Fernández, José** María. Generación de Sistemas Basados en Reglas mediante Programación Genética, Tesis de maestría.
12. **London Dysmorphology Database**. [En línea] http://www.lmdatabases.com/.
13. **Online Mendelian Inheritance in Man**. http://www.ncbi.nlm.nih.gov/omim.
14. **García Martínez, Ramón**. Metodologías de educción de conocimiento para la construcción de sistemas informáticos expertos.
15. **Sanz Bermejo, Laura y Monreal Gómez, Enrique.** Eclipse como IDE. [En línea] [Citado el: 12 de Febrero de 2010.] http://www.eclpise.org.
16. **Álvarez, Miguel Angel**. Introducción al manual del lenguaje PHP en su versión 5. [En línea] http://www.desarrolloweb.com/articulos/1696.php.

17. **Potencier, Francois Zaninotto Fabien.** Symfony la guía definitiva. 2008. [En línea] http://www.librosweb.es.
18. Servidor Web. WampServer [En línea] http://www.desarrolloweb.com/articulos/wampserver-instalar-php5.html.
19. Servidor Web. Tomcat [En línea] http://tomcat.apache.org/
20. **Hernández, José Alberto**. [En línea] http://www.versioncero.com/noticia/210/visual-paradigm-for-uml.
21. **Gómez Mazza, Ramiro.** Primeros pasos con Drools. [En línea] Martes 26 de Mayo de 2009 http://www.worldlingo.com.
22. Comparación de Frameworks en JavaScript. [En línea] http://www.desarrolloweb.com/articulos/listado-distintos-framework-javascript.html.
23. **Orosco, Eyleen, Molina, Elvismary y Sánchez, Yusdenis**. Documento de Arquitectura de Software. [En línea] http://10.7.9.200:5901/svn/genetica/Ingenieria/Arquitectura/.
24. **Ramírez Rodríguez, Jaime.** Método para la verificación de sistemas híbridos basado en la propagación de etiquetas. Tesis Doctoral. Mayo 2002. Universidad Politécnica de Madrid. Facultad de Informática.
25. **Grau, Xavier Ferré**. Principios Básicos de Usabilidad para Ingenieros Software. Universidad Politécnica de Madrid

BIBLIOGRAFÍA

1. **Alvarado Garrigó, Marta y Borges Fernández, Nelvis.** Sistema Informático para la Red Nacional de Genética Médica: módulo Teleconsulta versión 2.0. 2008. Tesis de Diploma. UCI. La Habana.
2. **Criado Briz, José Mario**. Introducción a los Sistemas Expertos. [En línea] 2010. [Citado el: 13 de Noviembre de 2010.] http://www.ingenieroseninformatica.org/recursos/tutoriales/sist_exp/index.php.
3. **Hurtado Vega, José de Jesús**. Inteligencia Artificial. [En línea] 2010. [Citado el: 16 de Noviembre de 2010.] http://www.itlp.edu.mx/publica/boletines/actual/inteligencia.html.
4. **Pajares Santos, M**. Inteligencia Artificial e Ingeniería del Conocimiento. Ra-Ma, 2005. [En línea] 2008.
5. **Giarratano, Riley**. Sistema Experto, principios y programación. ISBN 7-111-10844-2.
6. **Rodríguez, Yanet**. Evaluación de un Sistema Basado en Casos Interpretativo. Tesis de Maestría. 1998.
7. **Gutiérrez, José Manuel y S. Hadi, Ali**. Libro Sistema Experto y Redes Probabilísticas.
8. **Gochez, Alejandro.** Sistemas Expertos. Un gran Avance. [En línea] lunes 28 de enero de 2008 http://inteligenciaartificialudb.blogspot.com/2008/01/sistemas-expertos-un-gran-avance.html.
9. **Gutiérrez, José Manuel**, Dpto. de Matemática. Sistemas Expertos Basados en Reglas, Aplicada, Universidad de Cantabria.
10. **Pérez González, Joel**. Sistemas Expertos, Universidad del Valle de México, Febrero 2010.
11. **Font Fernández, José** María. Generación de Sistemas Basados en Reglas mediante Programación Genética, Tesis de maestría.
12. **London Dysmorphology Database**. [En línea] http://www.lmdatabases.com/.
13. **Online Mendelian Inheritance in Man**. http://www.ncbi.nlm.nih.gov/omim.
14. **García Martínez, Ramón**. Metodologías de educción de conocimiento para la construcción de sistemas informáticos expertos.
15. **Sanz Bermejo, Laura y Monreal Gómez, Enrique.** Eclipse como IDE. [En línea] [Citado el: 12 de Febrero de 2010.] http://www.eclpise.org.
16. **Álvarez, Miguel Angel**. Introducción al manual del lenguaje PHP en su versión 5. [En línea] http://www.desarrolloweb.com/articulos/1696.php.

17. **Potencier, Francois Zaninotto Fabien.** Symfony la guía definitiva. 2008. [En línea] http://www.librosweb.es.
18. Servidor Web. WampServer [En línea] http://www.desarrolloweb.com/articulos/wampserver-instalar-php5.html.
19. Servidor Web. Tomcat [En línea] http://tomcat.apache.org/
20. **Hernández, José Alberto**. [En línea] http://www.versioncero.com/noticia/210/visual-paradigm-for-uml.
21. **Gómez Mazza, Ramiro.** Primeros pasos con Drools. [En línea] Martes 26 de Mayo de 2009 http://www.worldlingo.com.
22. Comparación de Frameworks en JavaScript. [En línea] http://www.desarrolloweb.com/articulos/listado-distintos-framework-javascript.html.
23. **Orosco, Eyleen, Molina, Elvismary y Sánchez, Yusdenis**. Documento de Arquitectura de Software. [En línea] http://10.7.9.200:5901/svn/genetica/Ingenieria/Arquitectura/.
24. **Ramírez Rodríguez, Jaime.** Método para la verificación de sistemas híbridos basado en la propagación de etiquetas. Tesis Doctoral. Mayo 2002. Universidad Politécnica de Madrid. Facultad de Informática.
25. Motor de inferencia, OFBiz. [En línea] http://ofbiz.sourceforge.net/.
26. **De Ávila Ramos, Jorge.** Sistema Informatizado para la Gestión de Teleconsultas entre Genetistas. Tesis de Doctorado. http://www.lafacu.com/apuntes/informatica/.
27. **Jones, K.L.: SMITH**. Patrones reconocibles de malformaciones humanas. 6ta Edición. Barcelona: Editorial Elsevier; 2007.
28. **Castro, Marcel** (2002). Sistema Experto. Consultado el 6 de abril del 2010 en http://strix.ciens.ucv.ve/~iartific/Material/PP_Sistemas_Expertos.pdf.
29. **Giné Benaiges, R.** LA CITOGENÉTICA EN LA VALORACIÓN DISMÓRFICA. Consultado 12 de abril del 2010 http://www.comtf.es/pediatria/Bol-2004-2_3/CITOGENETICA_RGines_Benaiges.pdf.
30. Apache http server project. [En línea]. http://httpd.apache.org. http://httpd.apache.org/docs/2.0/es/new_features_2_0.html.
31. **Cabezas Granado, Luis Miguel.** PHP 5. [En línea] 2004. España. ISBN: 8441517851.
32. **Soriano, Ing. Fernando.** Diseño Orientado a Objetos Patrones GoF. [En línea] 2010.

33. **Fabien Potencier, Francois Zaninotto.** Symfony la guía definitiva. [En línea] 2008. ISBN-13.

34. **Castillo, Enrique.** Base de Dato, Ingeniería del Software y Sistemas de Información. [En línea] 2007. [En línea] http://kybele.escet.urjc.es/documentos/HC/Exposiciones/EclipseIDE.pdf.

35. **Russell, Stuart J and Norvig, Peter**. Artificial Intelligence. Prentice Hall, Englewood Cliffs, New Jersey 07632.

36. **Gálvez Lio, Dr. Daniel.** Sistemas Basados en el Conocimiento. 1998. Universidad Central "Martha Abreu" de las Villas.