Universidad de las Ciencias Informáticas

Facultad 6

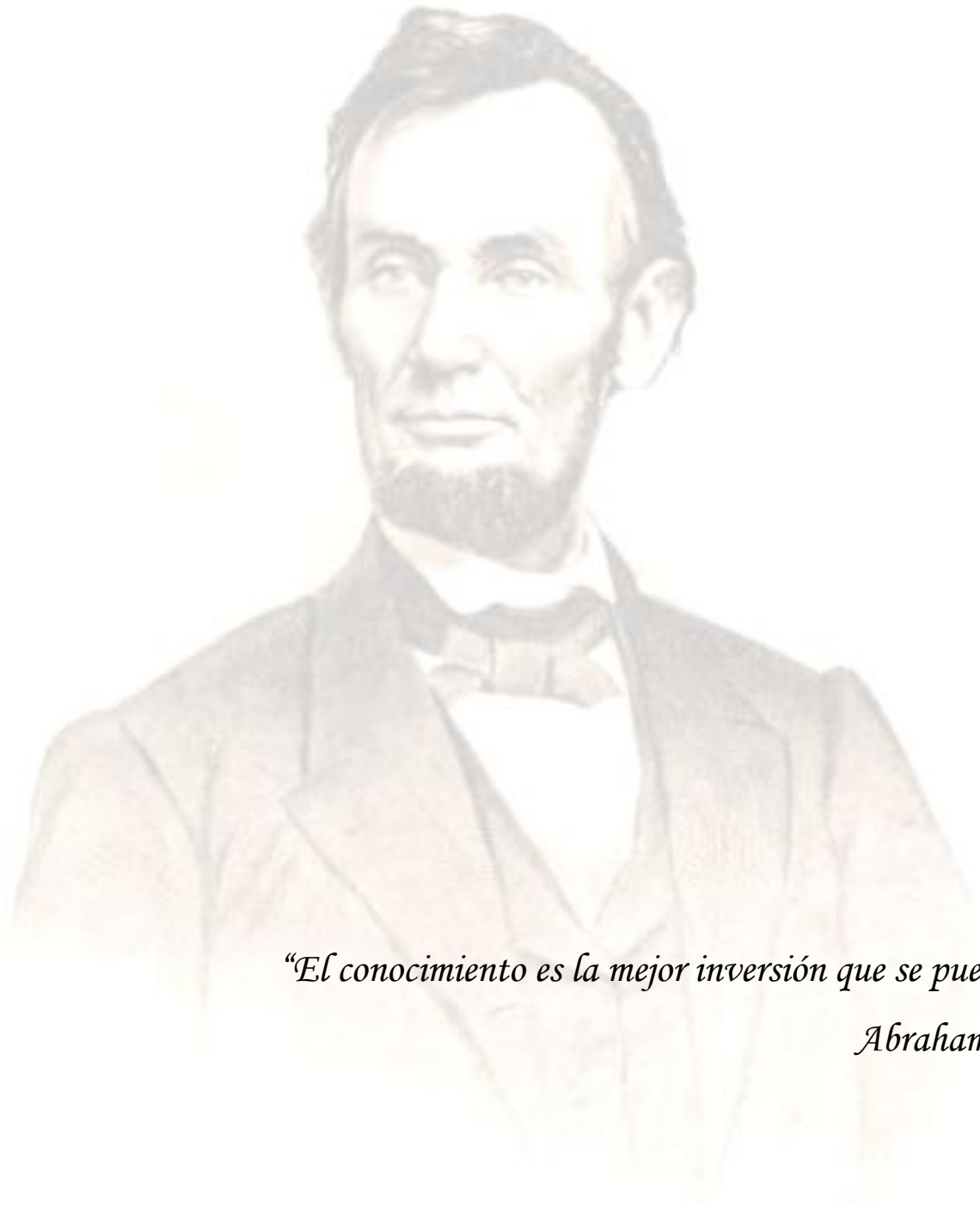
Título: “Módulo para la visualización geográfica de la información que se gestiona en alasMEDIGEN”

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

Autores: Odaimy Barreto Ruiz
Ivet González Palmero

Tutores: Ing. Yosuán Crespo García
Ing. Randy Ibarrola Suárez

La Habana,
junio 2012
“Año 54 de la Revolución”



"El conocimiento es la mejor inversión que se puede hacer"

Abraham Lincoln

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 se firma la presente a los ____ días del mes de _____ del año _____.

Odaimy Barreto Ruiz

_____

Firma del autor

Ivet González Palmero

_____

Firma del autor

Ing. Yosuán Crespo García

_____

Firma del tutor

Ing. Randy Ibarrola Suárez

_____

Firma del tutor

DATOS DE CONTACTO

Tutor:
Ing. Yosuán Crespo García
Universidad de las Ciencias Informáticas, Habana, Cuba
Email: ycgarcia@uci.cu

Tutor:
Ing. Randy Ibarrola Suárez
Universidad de las Ciencias Informáticas, Habana, Cuba
Email: 1

AGRADECIMIENTOS

A Fidel y la Revolución por permitir que nuestro sueño de ser profesionales se haga realidad.

A nuestros tutores, los Ingenieros Yosuán Crespo y Randy Ibarrola. En especial a Randy por ser realmente especial como tutor y una excelente persona, por su apoyo incondicional, su paciencia, su confianza y colaboración para poder lograr la realización de este trabajo lleno de experiencias y satisfacciones memorables; por ayudarnos a convertirnos en mejores personas.

A los Ingenieros Adrián Gracia, Yunier Santana, Yudiel la Rosa, Keidy García. Yasel Almenares, Velmour Muñoz, por su apoyo incondicional.

A todas aquellas personas que de una u otra forma, colaboraron o participaron en nuestra formación como persona y profesional, hacemos extensivo nuestro más sincero agradecimiento.

Odaimy Barreto

Llega el momento más difícil para mi, y es el de expresar lo agradecida que en este momento me siento con todas las personas que de una forma u otra me han dado su apoyo, comprensión y ayuda desinteresada para cumplir mi sueño, y si existe alguien que no sienta su presencia en mis palabras, reciba mis más sinceras disculpas y tenga la certeza que siempre le estaré agradecida.

Quiero agradecer a mi madre, por estar junto a mí en cada paso que he dado de mi carrera, por ser la principal fuente de inspiración para seguir adelante sin rendirme. Porque sin escatimar esfuerzo alguno, ha sacrificado gran parte de su vida para formarme en la profesional que me he convertido. A quien la ilusión de su vida ha sido convertirme en persona de bien. A quien nunca podré pagar su sacrificio ni aún con el tesoro más grandes del mundo.

A mi abuela Aracelis, por su cariño, por comprenderme y por confiar siempre en mí, por estar presentes en esta etapa de mi vida tan importante.....la quiero mucho.

A toda mi familia por siempre confiar en mi capacidad de poder llegar hasta el lugar en el que hoy me encuentro, especialmente a mi hermano Dariel, mi padrasto Daniel y mi prima Arletys.

A mis viejos y nuevos amigos, porque siempre han estado ahí para mí, por apoyarme, por dejarme formar parte de sus vidas, en especial a Ismaray por ser mi amiga de toda la vida y a Yailenis porque durante 5 largos años ha estado a mi lado en las buenas y en las malas, a todos los adoro y nunca los olvidaré.

A mi compañera de tesis, Ivet, por su optimismo y porque juntas hicimos un gran esfuerzo por lograr esta anhelada realidad.

A todos los profesores que han contribuido con mi formación profesional.

Y a todo aquel que luego del abrazo y del saludo, preguntó ¿Cómo va la Tesis?

A todos muchísimas gracias.....

Ivet González

A mis padres por amarme, educarme y hacerme una persona de bien para con la sociedad. Por ser mi apoyo y mi motor impulsor en cada uno de mis sueños y metas, sobre todo en este de convertirme en Ingeniera. Por aguantar mis maclriadeces y malos humores.

A mi abuela preciosa, mis primos, tías y tío, por ser la mejor familia del mundo. Por ser también un apoyo incondicional en todas las etapas de mi vida.

A mi compañera de tesis por pasar juntas malas noches, por aguantar mi mal humor y comprenderme.

A mis amigos de la casa, a Yelen, Annie, Richard, Jarold, Yunior, por ser los mejores amigos del mundo. Por estar siempre ahí aunque estemos lejos.

A Yamirka por ser una segunda madre, por quererme como su hija.

A mis amigos de la universidad, en especial a Yaima, Yudelis, Leovan, Yeicy, Geidy, Adis y Dary, a los muchachos del grupo 1 de Artemisa; que a pesar que no pudimos pasar juntos este último año, han sido más que amigos hermanos. Agradecerle a los que estuvieron conmigo la mayor parte del tiempo en este año, especialmente a Yehimy, Eddy, Dayleé, Sadys, Grether, Osmil, Leisy, Yoandy, Nily, Héctor Alejandro Rodríguez, Leiny, Ada, Carlos Arce y a todos los demás muchachos del 6510 y 6511, que me acogieron en su grupo como si estuviéramos juntos desde primer año.

DEDICATORIA

Dedico este trabajo a la personas que le debo mi vida:

A mi madre, que tanto se sacrificó todos estos años para que pudiera estudiar y ser la persona que soy, quien hubiera dado su vida si fuese necesario para que terminara la carrera,

A mi abuela por su preocupación y por estar siempre pendiente de mí,

A mi hermano por ser la persona que es,

A mi padrasto por ser como un padre para mí,

A mis amigos por consentirme muchas veces y porque con sus consejos supe tomar las decisiones correctas en momentos cruciales,

A mi familia que siempre me deseó los mejores resultados,

A todos los que formaron parte de mi vida en esta etapa de mi carrera profesional y contribuyeron de alguna forma a que pudiera alcanzar mis metas.

Odaimy Barreto Ruiz

Dedico este trabajo a mis padres y a mi abuela preciosa, por ser los mejores padres y la mejor abuela del mundo, por su apoyo, su confianza y su amor,

A mis hermanos, que aunque no los tengo cerca ni crecí junto a ellos, siempre han sido una guía para la más pequeña,

A Doña Eglys de Lozada, que Dios la tenga en la Gloria, por ser una extraordinaria amiga, por hacerme parte de su familia, por tenerme como una hija,

A mi familia y amigos, que son muy especiales en mi vida.

Ivet González Palmero

RESUMEN

El Centro Nacional de Genética Médica tiene como objetivo principal desarrollar proyectos de investigación e innovación en el campo de la Genética Médica y otras disciplinas afines. Actualmente el Sistema Informático para la red Nacional de Genética Médica: alasMEDIGEN, contempla una gran cantidad de información procedente de los estudios genéticos realizados en todo el territorio nacional. Debido a esto, el proceso de obtención de los reportes de los diferentes registros con los que cuenta se hace engorroso, máxime cuando el sistema no cuenta con un mecanismo que le represente a los genetistas la información de manera clara y precisa. Esto trae como consecuencias, que el proceso de análisis de la información de cada registro sea costoso en tiempo y esfuerzo, debido al cúmulo de información que estos contienen, que además impiden que las investigaciones llevadas a cabo por los especialistas, tengan la eficiencia debida. Dicha abundancia de información atenta contra el proceso de toma de decisiones certeras, de la que en muchos casos dependen los especialistas para diagnosticar el tratamiento adecuado a pacientes con cierta discapacidad genética. Como solución a esta problemática la presente investigación tiene como propósito, el diseño, implementación y prueba del Módulo para la visualización geográfica de la información que se gestiona en el Sistema Informático para la red Nacional de Genética Médica. Este módulo aportará eficiencia y rapidez al análisis de la información de los pacientes por parte de los especialistas, permitiendo representar la información de manera clara y precisa.

Palabras claves: análisis de la información, genética, módulo, reportes, toma de decisiones.

TABLA DE CONTENIDOS

Tabla de contenidos

Tabla de contenidos

Índice de Figuras

Índice de Tablas

INTRODUCCIÓN

La Genética Médica (GM) se define como la especialidad médico-sanitaria que aplica los conocimientos de la genética a la práctica médica, ocupándose de las enfermedades de origen genético, incluyendo patologías hereditarias y malformativas de la especie humana. (1)

Su aparición data de mediados del siglo XIX cuando el monje Gregor Mendel investiga sobre la hibridación en guisantes y que más tarde se conoce como las leyes de Mendel, que plantean la transmisión por herencia de las características de los organismos padres a hijos. A partir de entonces el conocimiento genético ha permitido la mejora extensa en productividad de plantas, desarrollo de animales, pero sobre todo se aplica a seres humanos para evitar y combatir malformaciones en el ADN. El conocimiento genético también ha sido un componente dominante de la revolución en salud y asistencia médica en este siglo.

El Centro Nacional de Genética Médica (CNGM), órgano rector al respecto en Cuba, tiene dentro de sus funciones principales 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, con el fin de mejorar los niveles de salud del pueblo cubano, y disminuir el impacto de las enfermedades con implicación genética, en el cuadro de la morbimortalidad del país, así como realizar aportes al desarrollo de estas ramas de la ciencia, teniendo en cuenta las potencialidades que se derivan de su integración.

En el año 2008 la Universidad de las Ciencias Informáticas (UCI), que tiene como misión, entre otras, la de generar productos y servicios informáticos; conjuntamente con el CNGM, comienzan el desarrollo del Sistema Informático para la Red Nacional de Genética Médica (alasMEDIGEN), el cual tendría como objetivo fundamental la gestión de la información de las consultas de genética médica y de los estudios que en el CNGM se realizan. Actualmente el sistema alasMEDIGEN, contempla una gran cantidad de información procedente de los estudios genéticos realizados en todo el territorio nacional, tales como: Registro Cubano de Malformaciones Congénitas, Registro Cubano de Discapacitados, Registro Cubano de Gemelos, Registro Cubano de las Personas con Retraso Mental, Registro Cubano de Historias Clínicas, Registro Cubano de Enfermedades Genéticas, Registro Genético Preventivo de Familias con Enfermedades Comunes y el Registro Cubano de Anomalías Cromosómicas, de los cuáles se deriva un gran cúmulo de información. Debido a esto, el proceso de obtención de los reportes de los diferentes registros se hace engorroso, máxime cuando el sistema no cuenta con un mecanismo que le represente a los genetistas la información de manera clara y precisa. Esto trae

como consecuencias, que el proceso de análisis de la información de cada registro sea costoso en tiempo y esfuerzo, debido al cúmulo de información que estos contienen, que además impiden que las investigaciones llevadas a cabo por los especialistas, tengan la eficiencia debida. Dicha abundancia de información atenta contra el proceso de toma de decisiones certeras, de la que en muchos casos dependen los especialistas para diagnosticar el tratamiento adecuado a pacientes con cierta discapacidad de tipo genética.

Por lo anteriormente expuesto se define como **problema de la investigación** ¿Cómo contribuir a la visualización de la información que se gestiona en alasMEDIGEN?

El **objeto de estudio** se centra en la visualización geográfica de la información enmarcado en el **campo de acción**: Visualización geográfica de la información que se gestiona en alasMEDIGEN.

El **objetivo general** que se persigue es**:** Desarrollar el módulo para la visualización geográfica de la información que se gestiona en alasMEDIGEN.

Del objetivo general se desglosan los siguientes objetivos específicos:

- Seleccionar las herramientas, metodologías y tecnologías para el desarrollo del módulo de visualización geográfica.
- Realizar análisis y diseño del módulo de visualización geográfica.
- Implementar el módulo de visualización geográfica.
- Realizar pruebas al módulo de visualización geográfica.

Para solucionar el problema planteado y lograr con éxito el cumplimiento de los objetivos, se definieron las siguientes tareas de la investigación:

- Estudio del estado del arte de las principales tecnologías *web* para el desarrollo de localización y visualización geográfica de la información que se gestiona en alasMEDIGEN.
- Análisis y selección de las tecnologías a utilizar para la localización y visualización geográfica de la información que se gestiona en alasMEDIGEN.
- Selección de los componentes de la arquitectura a utilizar.
- Caracterización de los componentes de la arquitectura que serán empleados durante el desarrollo.
- Evaluación de la arquitectura a utilizar para su posible despliegue en los servidores de Infomed.

- Desarrollo de los componentes del diseño.
- Selección del tipo de prueba para la validación del módulo.
- Aplicación del tipo de prueba seleccionada.

El desarrollo del módulo para la visualización geográfica de la información que se gestiona en alasMEDIGEN permitirá que los especialistas visualicen geográficamente la incidencia y nivel de prevalencia de los diferentes estudios genéticos en todo el país, facilitando de esta forma la toma de decisiones.

La estructura del trabajo de diploma queda definida de la siguiente forma:

Capítulo 1: Fundamentos teóricos.

En este capítulo se hace un estudio de los sistemas de visualización geográfica de la información existentes a nivel internacional y nacional. Se analizan la metodología, tecnologías y herramientas a utilizar para el desarrollo del módulo de visualización.

Capítulo 2: Análisis y diseño del sistema.

Se definen los requerimientos funcionales y no funcionales. De la misma forma se definen los actores y trabajadores y los casos de uso del sistema y se analiza la arquitectura establecida para alasMEDIGEN. Se refleja todo lo referente a las clases, diagramas y detalles propios del diseño.

Capítulo 3: Implementación y prueba.

Se realiza la implementación del módulo, se analizan los estilos de codificación mostrándose una solución a los requisitos especificados. También se hacen referencias a los diagramas de componentes y el código fuente de las clases más importantes. Se ilustran los principales resultados obtenidos y los casos de prueba para los principales casos de uso.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

En el capítulo se hace una breve investigación de los Sistemas de Información Geográfica (SIG) a nivel nacional e internacional. Se analizan las principales herramientas y tecnologías que se utilizarán para el desarrollo del módulo.

1.1 Sistemas de visualización geográfica

Un Sistema de Información Geográfica (SIG) es una integración organizada de *hardware*, *software* y datos geográficos diseñado para capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada, con el fin de resolver problemas complejos de planificación y gestión.

La evolución de los Sistemas de Información Geográfica en la actualidad ha tenido un desarrollo considerable en todas las ramas de la sociedad. Los SIG resultan de gran importancia debido a que ayudan a optimizar el proceso de toma de decisiones tanto de entidades como de negocios. Permiten además relacionar información de cualquier tipo con una localización geográfica. Los ejemplos de utilización de los SIG son virtualmente ilimitados.

- Instituciones gubernamentales o empresas de mercadeo pueden relacionar información demográfica de censos con mapas políticos.
- Médicos y hospitales pueden relacionar mapas de enfermedades con condiciones de salubridad.
- Autoridades y legisladores pueden relacionar mapas de lugares donde se cometieron crímenes con patrones de criminalidad.
- Personal de servicios de emergencia puede relacionar mapas de áreas de riesgo con información sobre inundaciones o incendios forestales. (2)

1.1.1 JGISView-Leader

Es un programa que accede a la base de datos de proyectos LEADER[1], hace consultas a las mismas, a partir de las informaciones generadas construye una cobertura geográfica que puede ser vista tanto desde la aplicación o desde cualquier visualizador capaz de trabajar con coberturas en el formato *Shapefile* de ESRI (*Enviromental Systems Research Institute*). Este programa no solo trabaja con

[1] *Integración de capacidades de visualización geográfica en el software de gestión de proyectos LEADER.* **Latre, M. Á., y otros, y otros.** 72, Universidad de Zaragoza. España: s.n., 2001.

tablas de la base de datos del proyecto LEADER sino también con otras bases de datos siempre y cuando pueda ser referenciada con el proyecto y/o por el tipo de medida del mismo. (3)

1.1.2 Sistema de Información Geográfica de Correos de la compañía Telecom de España

Es un SIG que se utiliza para la planificación de la distribución de los correos, la ubicación de las oficinas de repartos, análisis de rutas de cualquier tipo de reparto, una mejor asignación de recogida de buzones, una gestión controlada del transporte y la concentración de los servicios tanto en zonas rurales como urbanas. (4)

1.1.3 Sistema de Información Turístico Geo Referenciado (SIGTUR)

En América Latina se utiliza SIGTUR de Honduras, es un SIG que se encarga de brindar detallada información turística nacional de todas las regiones del país así como de las empresas prestadoras de servicios turísticos para facilitar la toma de decisiones de los inversionistas tanto nacionales como extranjeros. (5)

1.1.4 SectorModel

En Cuba la empresa de Acueducto y Alcantarillado de Aguas de la Habana cuenta con un departamento SIG para la gestión de los recursos hídricos, en el cual se trabaja en pos del desarrollo, se gana en eficiencia, calidad de los servicios, así como confiabilidad en las decisiones tomadas. Se implementa la herramienta SectorModel para el trabajo con los sectores hidráulicos donde se maneja toda la información cartográfica para el análisis y la toma de decisiones mediante consultas SQL (lenguaje de consulta estructurado). (6)

1.1.5 El Sistema de Información Geográfica para el control de las rutas y paradas del Transporte Obrero de la Universidad de las Ciencias Informáticas (SIG-Rutas v1.0)

SIG-Rutas es una aplicación desarrollada por el Proyecto Productivo Aplicativos SIG del Departamento Geoinformática del Centro GEYSED. Tiene como objetivo la representación y gestión de las rutas y paradas de la transportación obrera del centro que se ejecuta a diario en distintos horarios y direcciones sobre la capital cubana. Es una herramienta de tipo aplicación *Web* que brinda una interfaz cómoda a los usuarios desde el punto de vista de funcionamiento. Su implementación está basada en la Plataforma GeneSIG y está soportada por una cartografía ligera de La Habana con capas especializadas de información del transporte obrero de la UCI. (7)

1.2 Metodología y lenguaje de desarrollo de software

1.2.1 RUP

Las metodologías y estándares utilizados en el desarrollo de *software* proporcionan las guías para conocer el camino a recorrer, desde antes de iniciar la implementación, asegurando de esta forma la calidad del producto final y el cumplimiento de su entrega en el tiempo estipulado.

Para el desarrollo de alasMEDIGEN se definió usar la metodología *Rational Unified Process* (RUP por sus siglas en inglés) que consiste en una metodología del proceso de ingeniería de *software* que proporciona un enfoque disciplinado para asignar tareas y responsabilidades dentro de una organización del desarrollo. Tiene como meta asegurar la producción del *software* de alta calidad que resuelva las necesidades de los usuarios dentro de un presupuesto y tiempo establecidos.

Es de vital importancia el seguir la metodología y las herramientas de implementación adecuadas siendo por ello que la metodología RUP proporciona las bases para llevar al éxito la elaboración del *software*.

En RUP existen dos dimensiones:

- Un eje vertical que representa las disciplinas, y agrupan actividades definidas lógicamente por la naturaleza y que representa el aspecto estático del proceso; como describe en términos de componentes de proceso, las disciplinas, actividades, flujos de trabajo artefactos y roles.
- Un eje horizontal que representa tiempo y demuestra los aspectos del ciclo de vida del proceso, representando el aspecto dinámico y que se expresa en términos de fases, de iteraciones y la finalización de las fases como se representa en la figura 1.

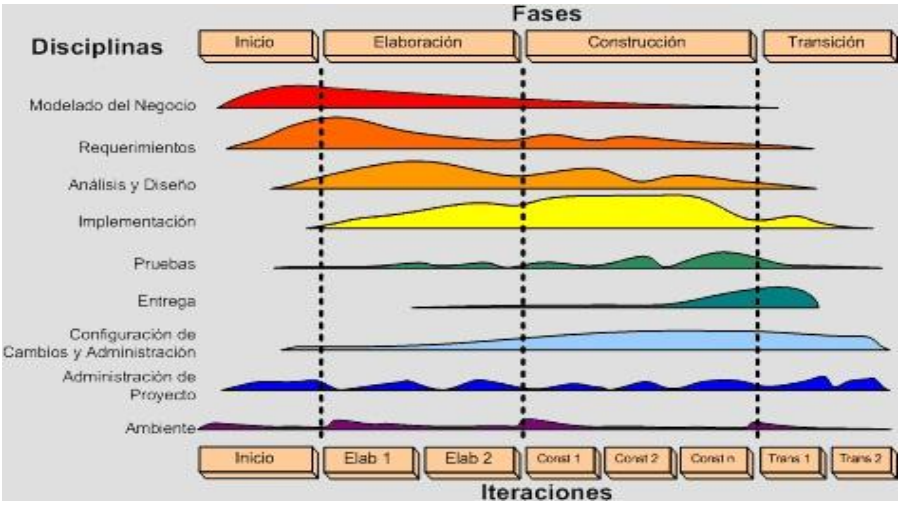


Fig 1. Disciplinas, fases e iteraciones de RUP

RUP utiliza el Lenguaje Unificado de Modelado (UML por sus siglas en inglés) para definir los modelos de *software* y consta de tres características esenciales que lo definen como un proceso dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental Se divide en cuatro fases: Inicio, Elaboración, Construcción y Transición.

Inicio: Se definen los objetivos del proyecto, funcionalidad y capacidad del producto.

Elaboración: Se define una arquitectura básica, se planifica el proyecto considerando recursos disponibles.

Construcción: Se desarrolla el producto a través de iteraciones, proporciona un producto con capacidad operacional inicial.

Transición: Se libera el producto y se entrega al usuario para un uso real.

Todas las fases no son idénticas en cuanto a tiempo y esfuerzos. Cada paso a través de las cuatro fases produce una generación del *software*. Se desarrollará nuevamente repitiendo la misma secuencia de cada una de las fases pero con diversos énfasis en cada una de ellas.

Esta metodología se ha agrupado en dos disciplinas: primaria o de ingeniería y de apoyo, las cuales se dirigen a una secuencia de pasos para la culminación de cada disciplina, estos pasos son conocidos como flujos de trabajo. Entre los flujos de ingeniería se tienen: Modelado del Negocio, Requerimientos, Análisis y Diseño, Implementación, Pruebas, Despliegue y entre los flujos de apoyo se tienen: Ambiente, Administración del Proyecto, Administración de Configuración y Cambios. (8)

1.2.2 UML

La metodología RUP utiliza UML, el cual es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de *software*. UML entrega una forma de modelar los procesos de negocio y las funciones de sistema, además permite escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de *software* reusables. Es un lenguaje gráfico para especificar, construir, visualizar y documentar las partes o artefactos (información que se utiliza o produce mediante un proceso de *software*). Pueden ser artefactos: un modelo, una descripción que comprende el desarrollo de *software* que se basen en el enfoque orientado a objetos, utilizándose también en el diseño *Web*. UML usa procesos de otras metodologías, aprovechando la experiencia de sus creadores, eliminó los componentes que resultaban de poca utilidad práctica y añadió nuevos elementos. UML es un lenguaje más expresivo, claro y uniforme que los anteriores definidos para el diseño orientado a objetos, que no garantiza el éxito de los proyectos pero si mejora sustancialmente el desarrollo de los mismos, al permitir una nueva y fuerte integración entre las

herramientas, los procesos y los dominios. Su principal característica es que se utiliza para el modelado de sistemas, con tecnología orientada a objetos, en donde el cliente participa en todas las fases del proyecto, con corrección de errores viables en todas las fases, aplicable para tratar asuntos de escala inherentes a sistemas complejos de misión crítica, tiempo real y cliente/servidor. (9)

1.3 Herramientas y tecnologías

Para el desarrollo del Módulo para la Visualización Geográfica de la información que se gestiona en alasMEDIGEN es muy importante cumplir con determinados requisitos que posibiliten su inclusión en el sistema. Para ello es imprescindible el uso de ciertas tecnologías y herramientas que faciliten la implementación.

1.3.1 PHP 5.0

En el presente trabajo se utilizará el lenguaje de programación interpretado PHP 5.0 (del inglés *Hypertext Pre Processor*). El mismo es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor, el cual brinda las siguientes ventajas:

PHP puede ser utilizado en cualquiera de los principales sistemas operativos del mercado, incluyendo Linux, muchas variantes Unix, Microsoft Windows, Mac OS X, RISC OS y probablemente alguno más. PHP soporta la mayoría de servidores *web* de hoy en día, incluyendo Apache, IIS, y muchos otros.

De modo que, con PHP se tiene la libertad de elegir el sistema operativo y el servidor *web* de su gusto. Además, tiene la posibilidad de utilizar programación por procedimientos, programación orientada a objetos (POO), o una mezcla de ambas.

Una de las características más potentes y destacables de PHP es su soporte para una gran cantidad de bases de datos. Escribir una página *web* con acceso habilitado a una base de datos es increíblemente simple utilizando una de las extensiones específicas (por ejemplo, para mysql), o utilizar una capa de abstracción, o conectarse a cualquier base de datos que soporte el estándar de Conexión Abierta a Bases de Datos (ODBC).

PHP también cuenta con soporte para comunicarse con otros servicios usando protocolos tales como LDAP (del inglés *Lightweight Directory Access Protocol*), IMAP (del inglés *Internet Message Access Protocol*), SNMP (del inglés *Simple Network Management Protocol*), NNTP (del inglés *Network News Transport Protocol*), POP3 (del inglés *Post Office Protocol*), HTTP (del inglés *Hypertext Transfer Protocol*), y otros. También se pueden crear *sockets* puros e interactuar usando cualquier otro protocolo. PHP soporta WDDX (del inglés *Web Distributed Data eXchange*), para el intercambio de

datos entre lenguajes de programación en *web*. Puede utilizar objetos Java de forma transparente como objetos de PHP. (10)

PHP es código abierto, lo cual significa que el usuario no depende de una compañía específica para arreglar lo que no funcione, no es necesario pagar actualizaciones anuales para tener una versión que funcione. (11)

1.3.2 MySQL 5.0

Para añadir, acceder, y procesar los datos almacenados en una base de datos, se necesita un sistema de gestión de base de datos como *MySQL Server*. MySQL, el sistema de gestión de bases de datos *SQL Open Source* más popular, lo desarrolla, distribuye y soporta MySQL AB. El *software* MySQL® proporciona un servidor de base de datos SQL muy rápido, *multi-threaded*, multi usuario y robusto. El servidor MySQL está diseñado para entornos de producción críticos, con alta carga de trabajo, así como para integrarse en *software* para ser distribuido.

MySQL tiene una doble licencia. Los usuarios pueden elegir entre usar el *software* MySQL como un producto *Open Source* bajo los términos de una Licencia Pública General de GNU (GNU GPL por sus siglas en inglés) o pueden adquirir una licencia comercial estándar de MySQL AB.

Panorámica del sistema de gestión de base de datos MySQL

- Es un sistema de gestión de bases de datos relacionales.
- El uso del servidor MySQL es libre de costo para toda la comunidad.
- Su rendimiento es considerable, la documentación es abundante por lo cual se convierte en una herramienta relativamente fácil de usar.
- Funciona en entornos cliente/servidor o incrustados.
- Una gran cantidad de *software* de contribuciones está disponible para MySQL.

Principales características de MySQL:

- Es multiplataforma.
- Puede mezclar tablas de distintas bases de datos en la misma consulta.
- Un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite la verificación basada en el *host*. Las contraseñas son seguras porque todo el tráfico de contraseñas está cifrado cuando se conecta con un servidor.

- Soporte a grandes bases de datos.
- Los clientes pueden conectar con el servidor MySQL usando *sockets* TCP/IP (Protocolo de control de transmisión/Protocolo de Internet) en cualquier plataforma.
- El servidor puede proporcionar mensajes de error a los clientes en muchos idiomas.
- MySQL server tiene soporte para comandos SQL para chequear, optimizar, y reparar tablas. (12)

1.3.3 Apache 2.0

Los servidores *web* se usan para servir páginas *web* solicitadas por equipos cliente. Los clientes normalmente solicitan y muestran páginas *web* mediante el uso de navegadores *web* como Firefox, Opera o Mozilla. Apache es un servidor *web* HTTP de código abierto, para plataformas Unix, Microsoft Windows, Macintosh y otras. Tiene amplia aceptación en la red, es el servidor *web* más usado en sistemas Linux.

Los servidores *web* Apache a menudo se utilizan en combinación con el motor de bases de datos MySQL, el lenguaje de *scripting* PHP, y otros lenguajes de *scripting* populares como Python y Perl. Esta configuración se denomina LAMP (Linux, Apache, MySQL y Perl/Python/PHP) y conforma una potente y robusta plataforma para el desarrollo y distribución de aplicaciones basadas en la *web*.

Es usado principalmente para el envío de páginas *web* estáticas y dinámicas en la *World Wide Web*. Muchas aplicaciones *web* están diseñadas para ser tratadas con Apache como servidor. Además es usado en muchas otras tareas donde el contenido necesita ser puesto a disposición en una forma segura y confiable. La licencia de *software* bajo la cual el *software* de la fundación Apache es distribuido es una parte distintiva de la historia de *Apache HTTP Server* y de la comunidad de código abierto y permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original. (13)

1.3.4 Visual Paradigm 6.0

Existen diferentes tipos de herramientas CASE (del inglés *Computer Aided Software Engineering*), dentro de las cuales se encuentra Visual Paradigm, la cual es una herramienta que es aplicada a lo largo de todo el ciclo de vida de un *software*. En estas son incluidas actividades como la gestión de proyectos y la estimación. También ofrecen un conjunto completo de herramientas de los equipos de desarrollo de *software* necesario para la captura de requisitos, *software* de planificación, la planificación de controles, el modelado de clases y de datos, entre otras.

Características de Visual Paradigm:

- Es una herramienta CASE que soporta las últimas versiones de UML y la notación y modelado de procesos de negocios.
- En adición al soporte de Modelado UML esta herramienta provee el modelado de procesos de negocios, además de un generador de mapeo de objetos-relacionales para los lenguajes de programación Java, .NET y PHP.
- Se integra con las siguientes herramientas Java: Eclipse/IBM WebSphere, JBuilder, NetBeans IDE, Oracle JDeveloper, BEA Weblogic.
- Está disponible en varias ediciones, cada una destinada según las necesidades del proceso de negocio: *Enterprise, Professional, Community, Standard, Modeler y Personal.*

Ventajas de Visual Paradigm

- Navegación intuitiva entre código y el modelo.
- Poderoso generador de documentación y reportes UML, PDF/HTML/MS, *Word*.
- Demanda en tiempo real, modelo incremental de viaje redondo y sincronización de código fuente.
- Superior entorno de modelado visual.
- Soporte completo de notaciones UML.
- Diagramas de diseño automático sofisticado.
- Análisis de texto y soporte de tarjeta CRC (clase, responsabilidad y colaboración).
- Proporciona soporte a varios lenguajes en generación de código e ingeniería inversa a través de plataformas java. (14)

1.3.5 Symfony 1.0.22

Un *framework* simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un *framework* facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

Symfony es un completo *framework* diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones *web*. 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*.

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

Symfony está desarrollado completamente con PHP 5. Es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. (15)

1.3.6 Netbeans 7.1

NetBeans es un Entorno de Desarrollo Integrado (IDE por sus siglas en inglés). Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. Está basado en *software* de Netbeans.org, bajo la Licencia de Distribución y Desarrollo Común y la Licencia Pública General de GNU versión 2.

Netbeans hace que el acceso al Subversion y otros repositorios sea más fácil. La integración fácil de usar entre los sistemas de control de versiones y seguimiento de incidencias le permite confirmar los cambios de código, y actualizar la situación del tema correspondiente en un solo paso. Crear nuevos temas y editar los problemas existentes, las cuestiones de ordenar, filtrar y guardar las consultas, crear parches y añadir archivos adjuntos - todo desde dentro del IDE.

NetBeans IDE 7.1 es la última versión estable del entorno de desarrollo integrado que ofrece características para ayudar en la construcción y mantenimiento de aplicaciones *web* y móviles. Entre los cambios implementados en NetBeans IDE 7.1 destaca soporte para JavaFX 2.0, soporte para WebLogic 12c, soporte para CSS3, mejoras en el constructor de interfaces *Swing*, mejoras en el *debug* de PHP, entre otras. (16)

1.3.7 Subversion 1.4.5

Se utilizará el sistema de control de versiones Subversion (SVN) 1.4.5. El sistema de control de versiones es un *software* que administra el acceso a un conjunto de ficheros y mantiene un historial de

cambios realizados. El control de versiones es útil para guardar cualquier documento que se cambie con frecuencia, como una novela, o el código fuente de un programa.

Normalmente consiste en una copia maestra en un repositorio central, y un programa cliente con el que cada usuario sincroniza su copia local lo que permite compartir los cambios sobre un mismo conjunto de ficheros. Además, el repositorio guarda registro de los cambios realizados por cada usuario, y permite volver a un estado anterior en caso de necesidad.

Es necesario el uso de este sistema de control de versiones dada las ventajas que brinda el mismo:

- Actualización de ficheros modificados.
- Copias de seguridad centralizadas.
- Historial de cambios.
- Brinda acceso remoto.
- Provee seguridad al sistema. (9)

1.3.8 OpenLayers 2.10

OpenLayers es una librería Javascript de uso libre para acceder, manipular y mostrar mapas en páginas *web*. Proporciona un API (Interfaz de Programación de Aplicaciones) que permite la creación de clientes *web* para acceder y manipular información geográfica proveniente de muy variadas fuentes, por ejemplo *Web Map Services*, *Web Feature Services*, Mapas comerciales, información genérica vectorial, etc. Esta librería permite incorporar mapas dinámicos en las páginas *web*. Los mapas se pueden dotar de diversos controles con capacidades de *zoom*, medida de distancias y muchas otras herramientas.

OpenLayers es un proyecto del *Open Source Geospatial Foundation* (OSGeo). Es una librería en puro Javascript, de uso totalmente libre bajo licencia BSD (del inglés *Berkeley Software Distribution*). (17)

1.3.9 Servicio Web

Un servicio web es un programa informático que permite la comunicación y el intercambio de datos entre aplicaciones y sistemas heterogéneos en entornos distribuidos. Los servicios web son por ende un conjunto de funcionalidad expuesta en una intranet o a través de Internet, por y para aplicaciones y computadoras sin la intervención humana.

El concepto ha sido perfilado en varios trabajos del comité *Web Service Activity* perteneciente al W3C (*World Wide Web Consortium*), particularmente con la propuesta del protocolo SOAP (del inglés

Simple Object Access Protocol). Ha sido utilizado desde su concepción para automatizar el intercambio empresarial. Las especificaciones están dictadas por los estándares SOAP y WSDL (del inglés *Web Services Description Language*) y tienen el objetivo de solucionar los problemas de integración heredados de las tecnologías anteriores y lograr su interoperabilidad.

Los servicios web implementan su lógica mediante la utilización de estándares. Para el transporte suele utilizarse TCP/IP, URI/URN/URL, MIME (del inglés *Multipurpose Internet Mail Extensions*), HTTP/SMTP, SSL/TLS, entre otros. Para el contenido suele utilizarse XML (del inglés eXtensible Markup Language) y SOAP. La utilización de estándares permite que cualquier tecnología que utilice esos estándares pueda hacer uso de estos servicios web, facilitando así la interoperabilidad de las aplicaciones.

El sistema consumirá un servicio web desarrollado por el centro de desarrollo GEYSED, usando los estándares SOAP y WSDL. Este servicio brindará las imágenes necesarias para construir los mapas geográficos.

Conclusiones del capítulo

En el desarrollo del capítulo se muestra la necesidad existente de desarrollar un módulo que lleve a cabo la visualización geográfica de la información gestionada en el sistema alasMEDIGEN, producto a que el mismo no cuenta con este tipo de servicio. Los Sistemas de Información Geográfica estudiados no se adaptan a las necesidades a satisfacer para el Centro Nacional de Genética Médica. Siguiendo las pautas establecidas en el sistema alasMEDIGEN se usa como metodología de desarrollo RUP, apoyado en el lenguaje de modelado UML. Las herramientas y tecnologías: PHP 5.0 como lenguaje de programación, MySQL 5.0 como sistema gestor de base de datos, Apache 2.0 como servidor de aplicaciones, Visual Paradigm 6.0 como herramienta CASE. Además de la utilización del framework Symfony 1.0.22, Subversion 1.4.5 como sistema de control de versiones, Netbeans 7.1 como entorno de desarrollo y la librería OpenLayer’s para la gestión de mapas.

CAPÍTULO 2: ANÁLISIS Y DISEÑO

En este capítulo se aborda todo lo referido al análisis y diseño del módulo a desarrollar. Se hace necesaria la definición de conceptos y sus relaciones mostrado en el modelo de dominio. Se describen los requisitos funcionales y no funcionales, así como los casos de usos del sistema. Se define la arquitectura a utilizar, obteniéndose los diagramas de clases del diseño, además de su descripción.

2.1 Modelo de Dominio

Un Modelo del dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan los principales conceptos. Muchos de los objetos o clases pueden obtenerse de una especificación de requisitos.

La modelación del dominio tiene como objetivo fundamental la comprensión y descripción de las clases más importantes en el sistema. (1)

2.1.1 Descripción de las clases del Modelo de Dominio

En el sistema a desarrollar ha sido difícil identificar las especificidades de los procesos del negocio, así como las fronteras y los actores del mismo, razón por la cual se hace necesario la confección de un modelo de dominio, para el cual se identificaron las siguientes entidades y conceptos:

Tabla 1. Descripción de las clases del dominio

Clases	Descripción
Genetista	Especialista que interactúa con el sistema solicitando información a través de un reporte.
alasMEDIGEN	Sistema Informático para la Red Nacional de Genética Médica. Este sistema contiene diferentes módulos para gestionar la información.
Módulo	Porción de la aplicación. Se encarga de generar el reporte solicitado por el genetista.
Reporte	Información solicitada por el genetista, generada por los diferentes módulos.
Criterio	Regla introducida por el genetista para obtener la información solicitada mediante un reporte.
Mapa	Imagen en la cual el genetista representa manualmente los reportes para visualizarlos geográficamente.

2.1.2 Diagrama de Clases del Modelo de Dominio

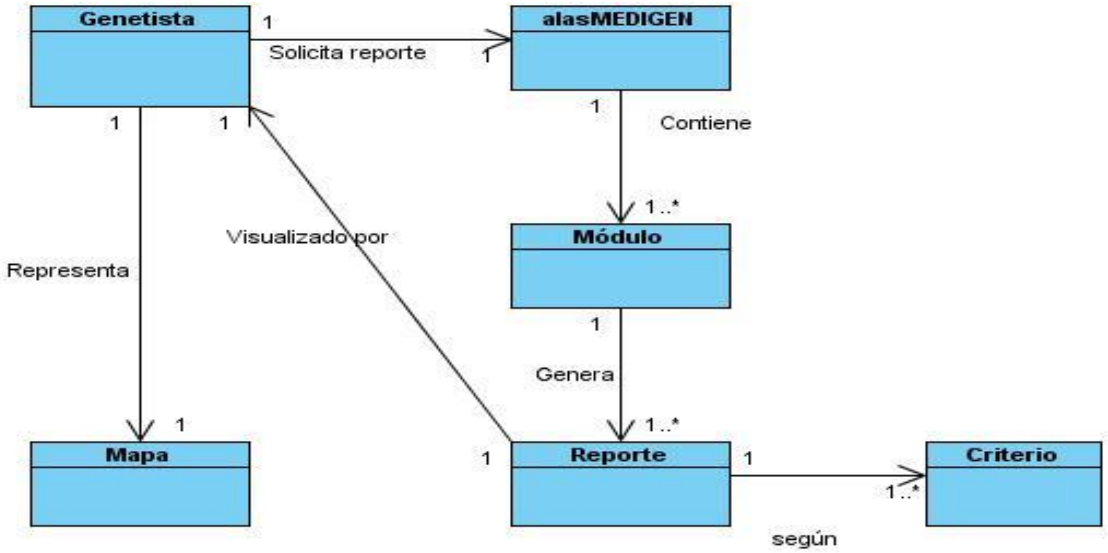


Fig 2. Modelo de Dominio

2.2 Levantamiento de requisitos

En el flujo de trabajo de requerimientos se realizó un levantamiento de seis requisitos funcionales y dieciocho requisitos no funcionales.

2.2.1 Requerimientos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Normalmente están vinculados a requerimientos funcionales, es decir una vez que se conozca lo que el sistema debe hacer se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser. (18)

Requerimiento de Usabilidad

RNF 1. La aplicación informática debe contar con una interfaz de usuario tan familiar como sea posible, 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* preferiblemente, para ahorrar tiempo de entrenamiento por el usuario.

Requerimiento de Rendimiento

RNF 2. Los tiempos de respuestas deben ser generalmente rápidos (3000 ms) al igual que la

velocidad de procesamiento de la información.

Requerimiento de Seguridad

RNF 3. El sistema debe tener un mecanismo propio para gestionar la seguridad a través de niveles de acceso a la información. Los permisos al ejecutar cualquier acción deben estar de acuerdo con el nivel jerárquico de acceso que presente el usuario en cada módulo, el cual es definido por los administradores del sistema.

Requerimiento de Disponibilidad

RNF 4. El sistema debe estar disponible 100% o muy cercano a esta disponibilidad durante las 24 horas del día los siete días de la semana.

Es necesario que el servicio *web* de GEYSED esté disponible durante ese período, con el menor tiempo posible de recuperación ante fallos.

Requerimiento de Apariencia o interfaz externa

RNF 5. Se deben utilizar imágenes y colores identificados con el sistema alasMEDIGEN.

FNR 6. El degradado de los colores en los mapas irá de una gama más clara hasta una gama más oscura en dependencia de la información generada.

RNF 7. La interfaz externa debe estar diseñada para verse en cualquier resolución, pero se recomienda 1024x768 px.

Requerimiento de Software

- Servidor

RNF 8. Sistema Operativo Linux.

RNF 9. Servidor de Aplicaciones Apache 2.0.

RNF 10. Servidor de bases de datos MySQL 5.0 o versiones superiores.

RNF 11. Servidor para realizar la representación geoespacial.

- Cliente

RNF 12. Los usuarios del sistema deberán contar con un navegador Internet Explorer 5.5 o Mozilla Firefox 4.0 o superior, para poder acceder a las opciones que brinda el sistema.

Requerimiento de Hardware

Para el desarrollo y ejecución de la aplicación se necesitará:

Para los servidores:

RNF 13. Microprocesador Pentium IV a 3.0 GHz o superior.

RNF 14. Memoria RAM de 1 GB o superior.

RNF 15. 2GB de espacio en disco duro.

Para las estaciones de trabajo del cliente:

RNF 16. Microprocesador Pentium a 512 MHz o superior.

RNF 17. Memoria RAM de 256 Mb o superior.

RNF 18. 1Gb de espacio en disco duro.

Requerimientos Legales

RNF 18. Las herramientas y las tecnologías en que estará basada la aplicación informática, deberán cumplir con las Licencias Públicas Generales (GNU) de *software* libre.

2.2.2 Requerimientos funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Estos no alteran la funcionalidad del producto, es decir que se mantienen invariables sin importarle con que propiedades o cualidades se relacionen. (18)

Con ellos, se pretende determinar de manera clara y concisa lo que debe hacer el sistema siguiendo un enfoque funcional. Todas las ideas que los clientes, usuarios y miembros del equipo de proyecto tengan acerca de lo que debe hacer el sistema, deben ser analizadas como candidatas a requisitos. Los requisitos funcionales del módulo son:

RF1 Visualizar geográficamente (VG) la prevalencia de pacientes con malformaciones congénitas al nacimiento por regiones geográficas.

Visualizar a nivel nacional o provincial la incidencia de una determinada malformación congénita. El usuario genetista selecciona el nivel y la malformación congénita a visualizar, esta debe ser mostrada en un mapa geográfico a partir del nivel escogido, a través de escala de colores, según la cantidad de casos existentes por región.

RF2 VG la cantidad de pacientes con discapacidad intelectual por regiones geográficas.

Visualizar a nivel nacional o provincial la cantidad de discapacitados mentales. El usuario genetista selecciona el nivel a visualizar, este debe ser mostrado en un mapa geográfico, a través de escala de colores, según la cantidad de casos existentes por región.

RF3 VG la cantidad de pacientes por tipo de discapacidad física a nivel provincial.

Visualizar a nivel provincial la cantidad de pacientes que presentan una discapacidad física determinada. El usuario o genetista selecciona la región y la discapacidad a visualizar, esta debe ser mostrada en un mapa geográfico, a través de escala de colores, según la cantidad de casos existentes.

RF4 VG la cantidad de pacientes con discapacidad física con vinculación laboral a nivel provincial.

Visualizar a nivel provincial la cantidad de discapacitados con vinculación laboral. El usuario o genetista selecciona la región a visualizar, esta debe ser mostrada en un mapa geográfico, a través de escala de colores, según la cantidad de casos existentes.

RF5 VG las enfermedades genéticas según rango de edades por regiones geográficas.

Visualizar a nivel provincial o municipal la existencia de enfermedades genéticas existentes en el país a partir de un rango de edades. El usuario o genetista selecciona la región a visualizar, esta debe ser mostrada en un mapa geográfico, a través de escala de colores, diferenciando cada rango de edad.

RF6 VG la cantidad de casos de Teleconsulta Genética atendidos por regiones geográficas.

Visualizar a nivel provincial o municipal la cantidad de casos de Teleconsulta Genética atendidos. El usuario o genetista selecciona la región a visualizar, esta debe ser mostrada en un mapa geográfico, a través de escala de colores.

2.2.3 Diagrama de casos de usos del sistema

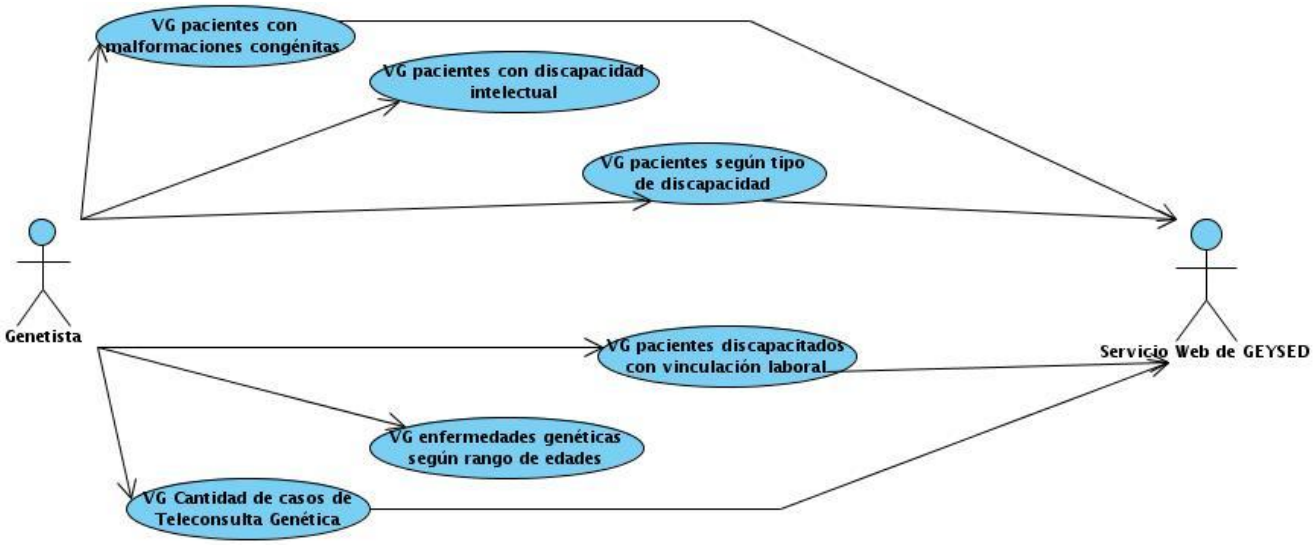


Fig 3. Casos de Uso del Sistema

2.2.4 Descripción textual de casos de usos del sistema

Tabla 2. Descripción del CU VG pacientes con malformaciones congénitas

	VG pacientes con malformaciones congénitas.
Actores	Genetista (Inicia el caso de uso), Servicio Web de GEYSED (SWG).
Resumen	El caso de uso se inicia cuando el genetista desea visualizar la prevalencia de una determinada malformación congénita al nacimiento tanto a nivel nacional como provincial, visualizando esta información en un mapa geográfico a través de una escala de colores según la cantidad de casos por región.
Precondiciones	El usuario debe estar autenticado.
Referencias	RF1
Prioridad	Crítica
Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del sistema**

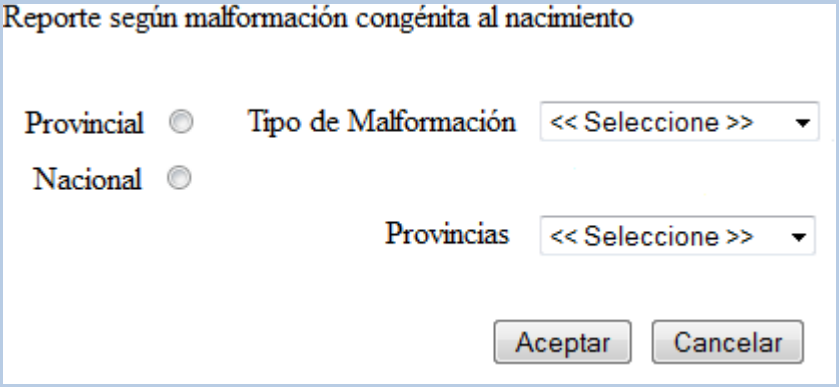
1- El genetista selecciona la opción Malformaciones Congénitas.	2- El sistema le brinda una interfaz que muestra las distintas malformaciones congénitas y la región geográfica a representar.
3- El genetista selecciona la malformación congénita y la región geográfica de la cual desea ver la prevalencia.	4- El sistema verifica que no se haya quedado ningún campo vacío.
	5- El sistema en dependencia de la región geográfica seleccionada muestra una interfaz con un mapa geográfico consumiendo el SWG, representando a través de escala de colores la cantidad de casos existentes.
Prototipo de Interfaz	
	
Flujos Alternos	
Acción del Actor	**Respuesta del sistema**
	4.1- Se emite un mensaje para que se llenen los campos vacíos.

Tabla 3. Descripción del CU VG pacientes con discapacidad intelectual

	VG pacientes con discapacidad intelectual.
Actores	Genetista (Inicia el caso de uso), Servicio Web de GEYSED (SWG).

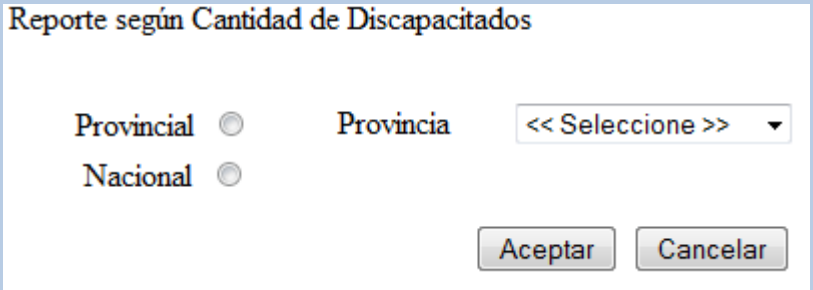
Resumen	El caso de uso se inicia cuando el genetista desea visualizar a nivel nacional o provincial la cantidad de discapacitados mentales, debe ser visualizado en un mapa geográfico, a través de escala de colores, según la cantidad de casos existentes por región.
Precondiciones	El usuario debe estar autenticado.
Referencias	RF2
Prioridad	Crítica

Flujo Normal de Eventos

Sección “”

Acción del Actor	**Respuesta del sistema**
1- El genetista selecciona la opción Cantidad de Discapacitados	2- El sistema le brinda una interfaz que muestra la región geográfica a representar.
3- El genetista selecciona la región geográfica de la que desea observar la cantidad de discapacitados mentales existentes.	4- El sistema verifica que no se haya quedado ningún campo vacío. 5- El sistema en dependencia de la región geográfica seleccionada muestra una interfaz con un mapa geográfico consumiendo el SWG, representando a través de escala de colores la cantidad de casos de discapacitados mentales existentes.

Prototipo de Interfaz



Flujos Alternos	
Acción del Actor	**Respuesta del sistema**
	4.1- Se emite un mensaje para que se llenen los campos vacíos.

Tabla 4. Descripción del CU VG pacientes según tipo de discapacidad

	VG pacientes según tipo de discapacidad.	
Actores	Genetista (Inicia el caso de uso), Servicio Web de GEYSED (SWG).	
Resumen	El caso de uso se inicia cuando el genetista desea visualizar a nivel provincial a través de escala de colores, según la cantidad de casos existentes, la cantidad de pacientes que presentan una discapacidad específica.	
Precondiciones	El usuario debe estar autenticado.	
Referencias	RF3	
Prioridad	Crítica	
Flujo Normal de Eventos		
Sección “”		
Acción del Actor		**Respuesta del sistema**
1- El genetista selecciona la opción Tipo de Discapacidad.		2- El sistema le brinda una interfaz que muestra el tipo de discapacidad y la región geográfica a representar.
3- El genetista selecciona el tipo de discapacidad y la región para observar la cantidad de pacientes con ese tipo de discapacidad.		4- El sistema verifica que no se haya quedado ningún campo vacío.
		5- El sistema en dependencia de la región

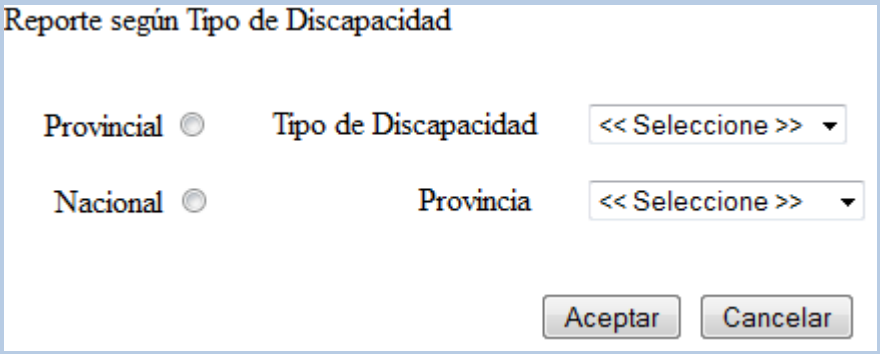
	geográfica seleccionada muestra una interfaz con un mapa geográfico consumiendo el SWG, a través de escala de colores donde se representa la cantidad de pacientes que presentan este tipo de discapacidad.
Prototipo de Interfaz	
	
Flujos Alternos	
Acción del Actor	**Respuesta del sistema**
	4.1- Se emite un mensaje para que se llenen los campos vacíos.

Tabla 5. Descripción de CU VG pacientes discapacitados con vinculación laboral

	VG pacientes discapacitados con vinculación laboral.
Actores	Genetista (Inicia el caso de uso), Servicio Web de GEYSED (SWG).
Resumen	El caso de uso se inicia cuando el genetista desea visualizar a nivel provincial la cantidad de discapacitados con vinculación laboral existentes.
Precondiciones	El usuario debe estar autenticado.
Referencias	RF4
Prioridad	Crítica
Flujo Normal de Eventos	

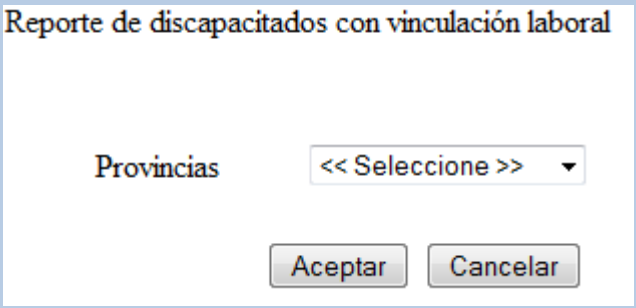
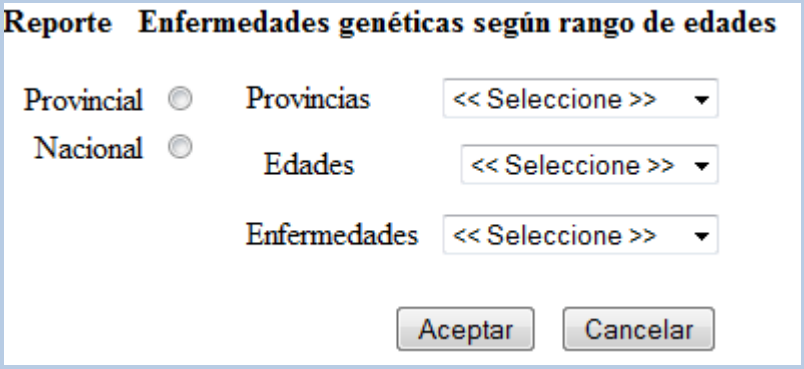
Sección “”	
Acción del Actor	**Respuesta del sistema**
1- El genetista selecciona la opción Vinculación Laboral.	2- El sistema le brinda una interfaz que muestra la región geográfica a representar.
3- El genetista selecciona la región de la que desea conocer la cantidad de discapacitados con vinculación laboral existentes.	4- El sistema verifica que no se haya quedado ningún campo vacío.
	5- El sistema en dependencia de la región geográfica seleccionada muestra una interfaz con un mapa geográfico consumiendo el SWG, a través de escala de colores donde se representa la cantidad de discapacitados con vinculación laboral existentes.
Prototipo de Interfaz	
	
Flujos Alternos	
Acción del Actor	**Respuesta del sistema**
	4.1- Se emite un mensaje para que se llenen los campos vacíos.

Tabla 6. Descripción del CU VG enfermedades genéticas según rango de edades

	VG enfermedades genéticas según rango de edades.
Actores	Genetista (Inicia el caso de uso), Servicio Web de GEYSED (SWG).

Resumen	El caso de uso se inicia cuando el genetista desea visualizar tanto a nivel provincial como nacional según un rango de edades seleccionado y la enfermedad genética, la cantidad de pacientes que presentan este tipo de enfermedad.
Precondiciones	El usuario debe estar autenticado.
Referencias	RF5
Prioridad	Crítica

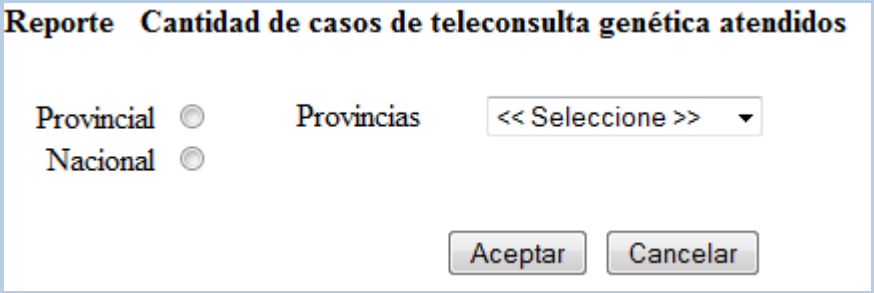
Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del sistema**
1- El genetista selecciona la opción Enfermedades Genéticas.	2- El sistema le brinda una interfaz que muestra la región geográfica a representar, el rango de edades y la enfermedad a seleccionar.
3- El genetista selecciona la región, la enfermedad genética y el rango de edades para conocer la cantidad de pacientes que presentan dicha enfermedad en ese rango de edades.	4- El sistema verifica que no se haya quedado ningún campo vacío.
	5- El sistema en dependencia de la región geográfica seleccionada muestra una interfaz con un mapa geográfico consumiendo el SWG, a través de escala de colores donde se representa la existencia de pacientes con la enfermedad genética ene ese rango de edades.
Prototipo de Interfaz	



Flujos Alternos	
Acción del Actor	**Respuesta del sistema**
	4.1- Se emite un mensaje para que se llenen los campos vacíos.

Tabla 7. Descripción del CU VG Cantidad de casos de Teleconsulta Genética

	VG Cantidad de casos de Teleconsulta Genética
Actores	Genetista (Inicia el caso de uso), Servicio Web de GEYSED (SWG).
Resumen	El caso de uso se inicia cuando el genetista desea visualizar tanto con todas las provincias a nivel nacional como con todos los municipios a nivel provincial la cantidad de casos de teleconsulta genética atendidos.
Precondiciones	El usuario debe estar autenticado.
Referencias	RF6
Prioridad	Crítica
Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del sistema**
1- El genetista selecciona la opción Casos de Teleconsulta Genética.	2- El sistema le brinda una interfaz que muestra la región geográfica a representar.

3- El genetista selecciona la región de la que desea la cantidad de casos de teleconsultas genéticas atendidos.	4- El sistema verifica que no se haya quedado ningún campo vacío.
	5- El sistema en dependencia de la región geográfica seleccionada muestra una interfaz con un mapa geográfico consumiendo el SWG, a través de escala de colores donde se representa la cantidad de casos de teleconsulta genética atendidos.
Prototipo de Interfaz	
	
Flujos Alternos	
Acción del Actor	**Respuesta del sistema**
	4.1- Se emite un mensaje para que se llenen los campos vacíos.

2.3 Patrones de diseño

Los patrones de diseño constituyen soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Son soluciones basadas en la experiencia y que se ha demostrado que funcionan. (19)

2.3.1 Patrones GRASP

Los patrones GRASP (del inglés *General Responsibility Assignment Software Patterns*) describen los principios fundamentales de la asignación de responsabilidades a objetos. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable. (20)

Los patrones básicos GRASP se refieren a cuestiones y aspectos fundamentales del diseño, ellos son:

Experto: Este patrón es utilizado en el modelo, puesto que Propel es la librería externa que utiliza Symfony para realizar su ORM (*Object-Relational Mapping*), o mapeo de objetos a bases de datos. El mismo garantiza acceder a la base de datos como si fuera orientada a objetos a partir de su estructura relacional. El *framework* Symfony divide el modelo en dos capas, la primera de acceso a datos y la segunda de abstracción de datos donde las clases generadas contienen todos los atributos y funcionalidades únicas para acceder a los datos correspondientes y modificarlos, es decir que cada una de estas clases es la encargada de brindar las funcionalidades de acceso a sus datos ya que son ellas mismas las que poseen dicha información.

Creador: Este patrón es el encargado de crear las instancias de los objetos para acceder a las entidades que brindan la información necesaria para realizar cualquier acción. En la clase vgActions encontramos las funciones definidas para el módulo de visualización geográfica. En las acciones se crean los objetos de las clases que representan las entidades, evidenciando de este modo que la clase vgActions es "creador" de dichas entidades. Un ejemplo de su uso es en la función PrevalenciaMalfCongNacimiento() donde se crean instancias de diferentes clases como: EgMcMalformacionesCongen, McCodigoMalformacionPorPaciente, entre otras con el objetivo de manipular determinada información que poseen estas entidades, convirtiendo a vgActions en creadora de dichas entidades.

Controlador: Este patrón es muy importante, ya observamos otros patrones capaces de crear instancias de otras clases, capaces de brindar determinada información que poseen solo ellos. Pero para lograr la unidad entre todas las partes y que el sistema funcione como un todo es necesario este patrón controlador que se encargue de unir todas las partes. Todas las peticiones al sistema son manejadas por un controlador frontal que se divide en varios componentes encargados de la seguridad, validaciones, configuración y enrutamiento, de igual modo se definen las acciones a ejecutar por el sistema en la clase vgActions, parte fundamental de dicho controlador, encargada de realizar todas las acciones del sistema. (9)

2.3.2 Patrones GoF

Los patrones GoF (del inglés *Gang of Four*) se dividen en tres categorías:

Patrones de creación: Muestran la guía de cómo crear objetos cuando su creación requiere tomar decisiones. Estas decisiones normalmente serán resueltas dinámicamente decidiendo qué clases instanciar o sobre qué objetos un objeto delegará responsabilidades.

Patrones estructurales: Describen la forma en que diferentes tipos de objetos pueden ser organizados para trabajar unos con otros.

Patrones de comportamiento: Se utilizan para organizar, manejar y combinar comportamientos.

En el desarrollo de este trabajo de diploma se utiliza el *framework* Symfony el cual hace uso de una serie de patrones GoF.

En la categoría de patrones estructurales, Symfony utiliza el patrón *Decorator* (Envoltorio): Añade funcionalidad a una clase, dinámicamente. Por ejemplo un archivo layout_mapa.php, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas del módulo, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el layout_mapa, o si se mira desde el otro punto de vista, el layout_mapa decora la plantilla. Este comportamiento es una implementación de este patrón de diseño.

2.4 Patrones de arquitectura

Los patrones de arquitectura ayudan a especificar la estructura fundamental de una aplicación. Expresan el esquema fundamental de organización para sistemas de *software*. Proveen un conjunto de subsistemas predefinidos; especifican sus responsabilidades e incluyen reglas y guías para organizar las relaciones entre ellos. Cada patrón de arquitectura ayuda a conseguir una propiedad específica en el sistema global, por ejemplo, la adaptabilidad de la interfaz de usuario. Dentro de los patrones de arquitectura se puede encontrar el patrón Modelo Vista Controlador (MVC) y el patrón modelo de tres capas.

En el desarrollo de este trabajo de diploma se utiliza el *framework* Symfony, el cual está basado en el patrón MVC, que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones *Web*, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el modelo es el Sistema de Gestión de Base de Datos y el controlador representa la Lógica de negocio, que está formado por tres niveles:

- El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- La vista transforma el modelo en una página *Web* que permite al usuario interactuar con ella.
- El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. El controlador se encarga de aislar al modelo y la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de

comandos, email, etc.). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes, por ejemplo, del tipo de gestor de bases de datos utilizado por la aplicación. (9)

2.5 Vista lógica de la arquitectura

La vista lógica describe el sistema mostrando las clases más importantes y su organización en paquetes y subsistemas. Esta vista permite observar cómo está diseñada la funcionalidad en el interior del sistema.

Symfony define un diseño, conceptos, componentes y herramientas de administración que se reutilizarán en el desarrollo del sistema. Establece una estructura del proyecto en aplicaciones, y cada aplicación está compuesta por módulos que representa a un grupo de páginas con un propósito relacionado. Symfony implementa el patrón MVC seleccionado para el sistema, por cuanto la estructura organizacional del SIGM se representa a través de tres elementos fundamentales: el controlador, el modelo y la vista. En la siguiente figura se muestra la vista lógica del módulo incluido al sistema.

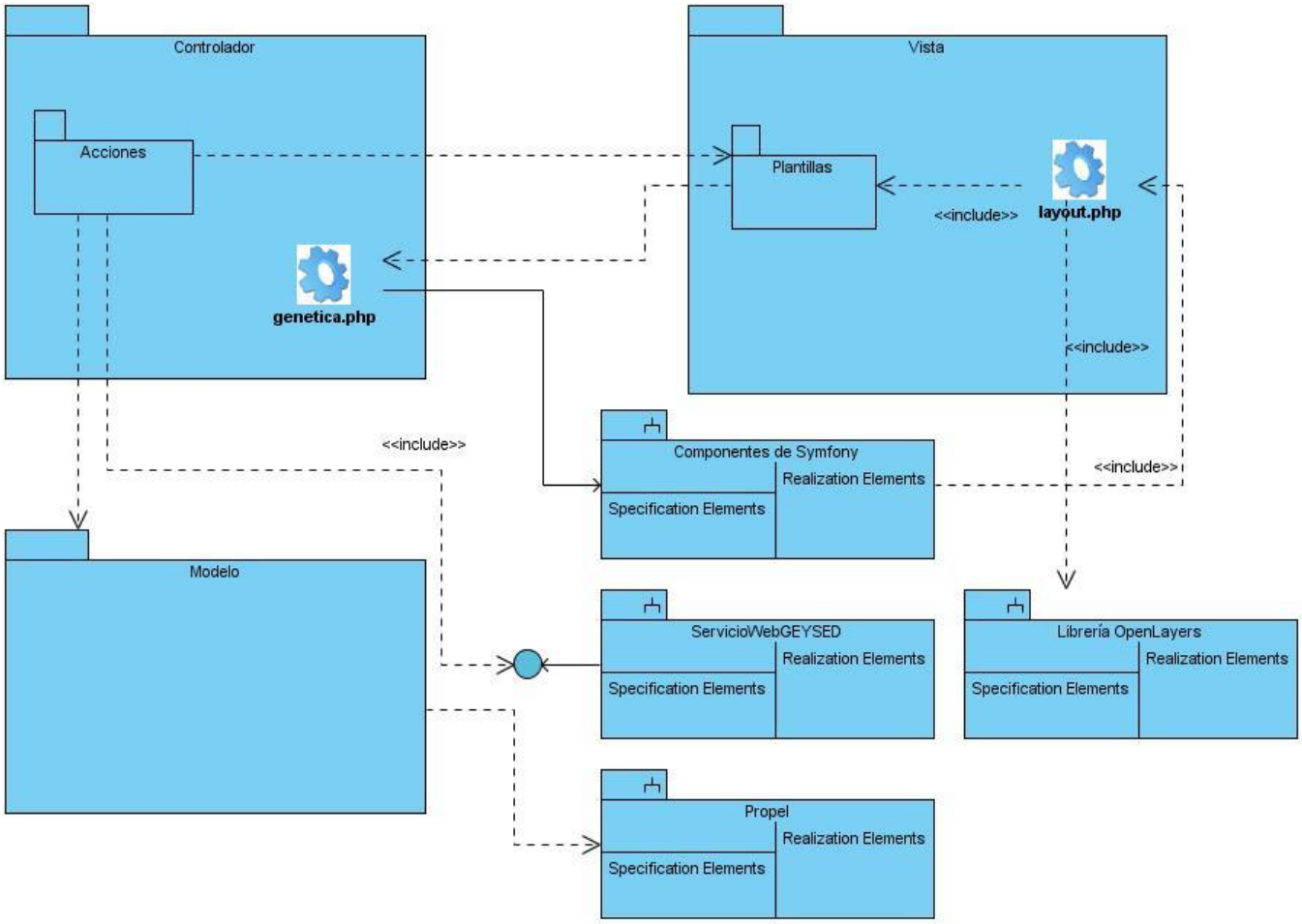


Fig 4. Vista lógica de la arquitectura

En la figura anterior se muestran las peticiones hechas por los usuarios al sistema, las mismas son manejadas por el controlador frontal, que es el único punto de entrada de la aplicación. Cuando el controlador frontal recibe una petición, inicializa las clases y constantes del núcleo del *framework*, luego carga la configuración, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la URL escrita, ejecuta la acción y produce la vista, mostrando finalmente la respuesta.

Las acciones contienen la lógica de la aplicación, verifican la integridad de las peticiones, utilizan el modelo y definen variables para la vista. El valor retornado por la acción determina como será producida la vista para especificar la plantilla que se utiliza al mostrar el resultado de la acción. El contenido de la plantilla se incluye en el layout, visualizando finalmente el resultado de la petición al usuario.

El modelo es el encargado de obtener los datos a través del acceso directo a la base de datos del sistema. Las clases del modelo obtienen los datos de la base de datos utilizando el ORM Propel que viene integrado con el *framework*.

Cuando el controlador procesa todos los datos, consume el servicio que brinda GEYSED, el cual genera todas las imágenes que finalmente se muestran en la vista que se construirá para mostrar los mapas tematizados resultantes.

2.6 Diagramas de Clases del Diseño

Un diagrama de clases del diseño representa las clases del diseño y sus objetos, así como los subsistemas del diseño.

A continuación se exponen los diagramas de clases del diseño de los Casos de Uso VG pacientes con malformaciones congénitas y VG pacientes discapacitados con vinculación laboral (para visualizar el resto de los diagramas puede acudir al expediente de proyecto).

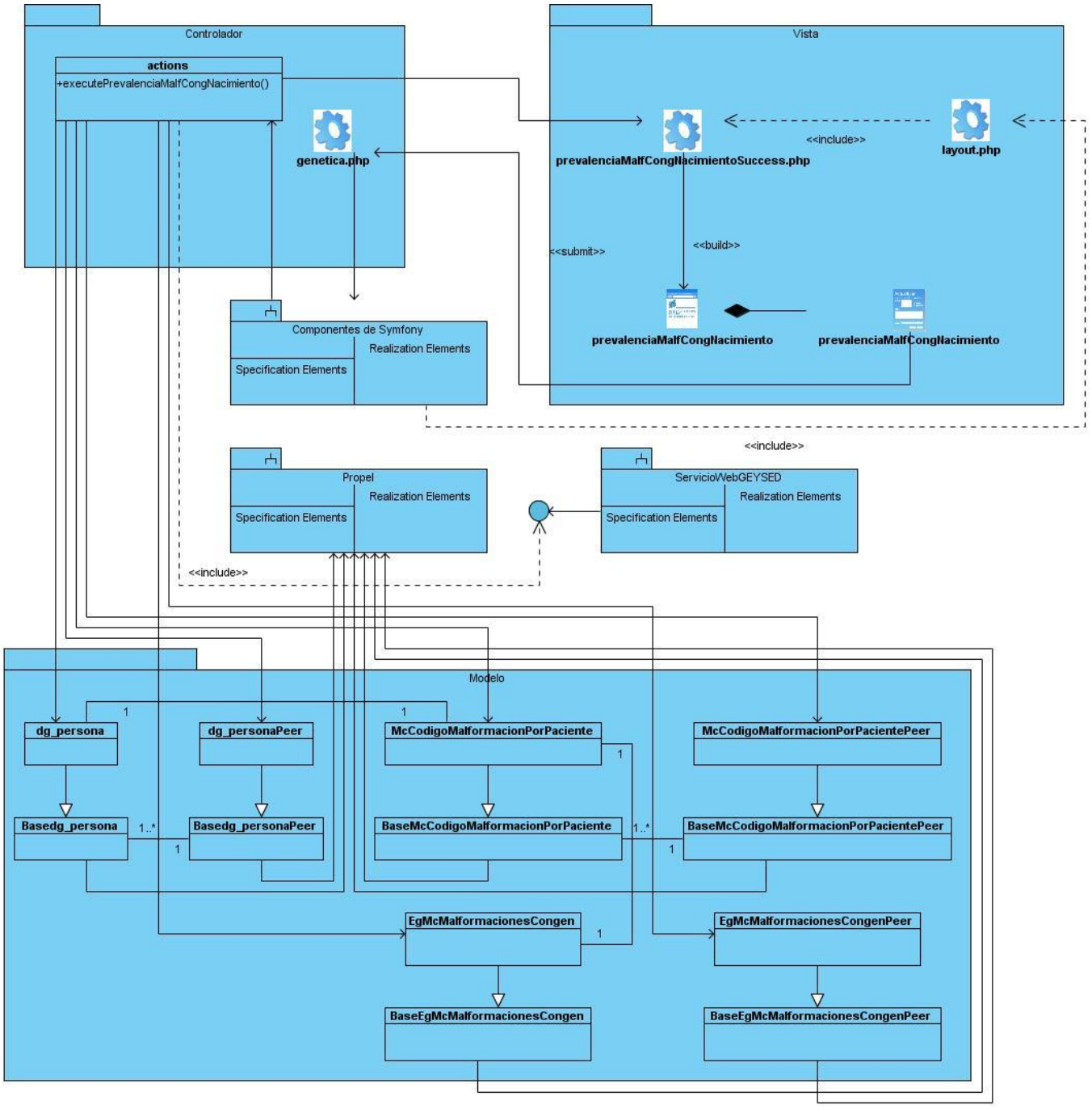


Fig 5. Diagrama de Clases del Diseño. CU VG pacientes con malformaciones congénitas

Cuando el usuario accede al sistema alasMEDIGEN, al módulo de visualización geográfica y selecciona la opción Prevalencia de Malformaciones Congénitas, se crea la página prevalenciaMalfCongNacimientoSucces.php la cual muestra la vista prevalenciaMalfCongNacimiento que tiene un formulario con el mismo nombre donde el usuario introduce los criterios de búsqueda,

como el nivel (Provincal o Nacional), el tipo de malformación y en dependencia del nivel selecciona la provincia o no. Luego se envía una petición *submit* al controlador frontal el cual a través de los componentes de Symfony delega la responsabilidad al controlador del módulo *actions*, el cual accede al modelo que es el encargado del acceso a los datos almacenados en la base de datos, las clases del modelo acceden a los datos de la base de datos utilizando el ORM (*Object-Relational Mapping*) Propel que viene integrado en Symfony.

Cuando el *actions* procesa todos los datos, consume el servicio *web* que brinda GEYSED, generando todas las imágenes con ayuda de la librería OpenLayers, que finalmente se muestran en la vista que se construirá para mostrar los mapas.

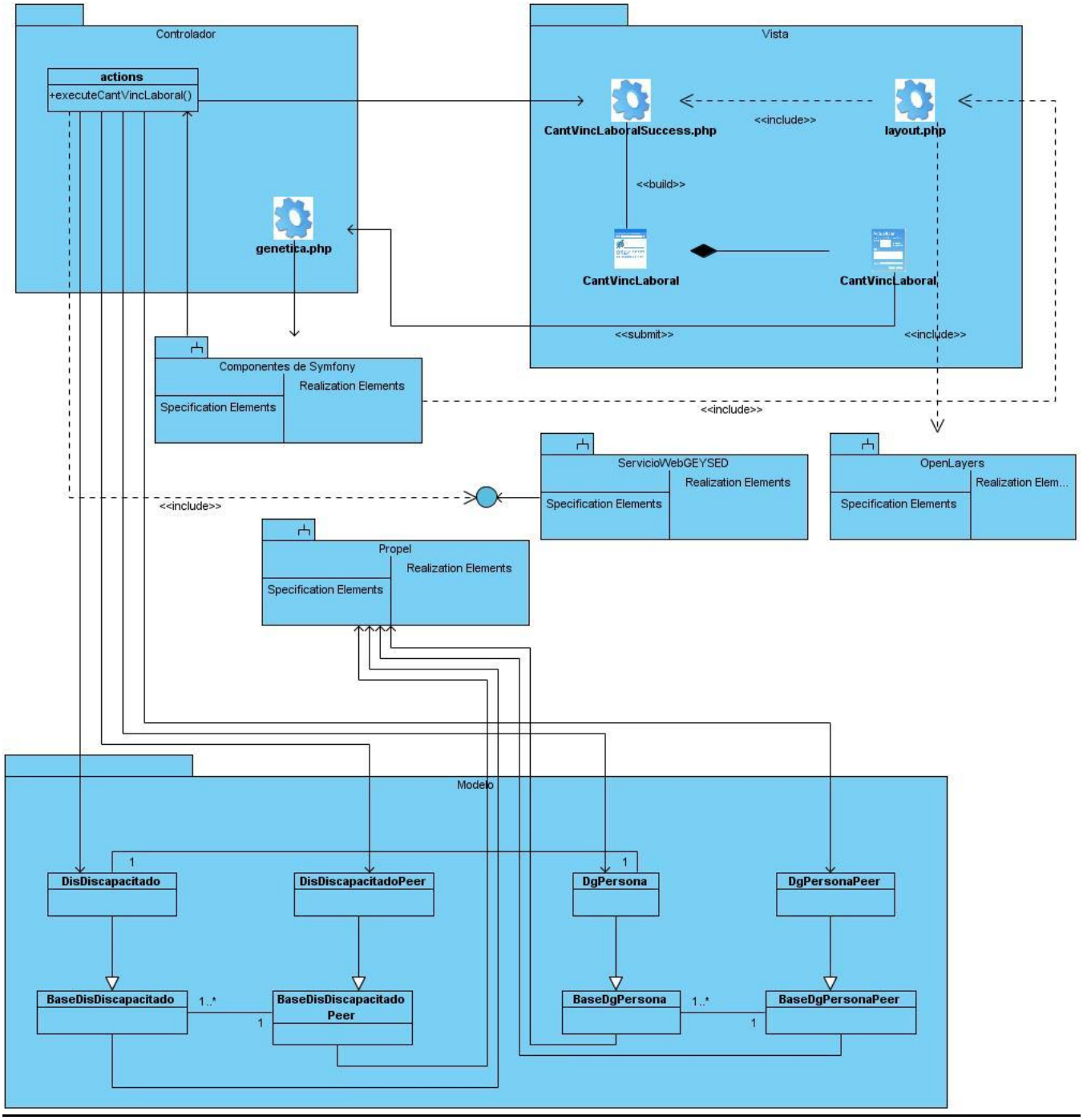


Fig 6.Diagrama de Clases del Diseño. CU VG pacientes discapacitados con vinculación laboral

Cuando el usuario accede al sistema alasMEDIGEN, al módulo de visualización geográfica y selecciona la opción Vinculación Laboral, se crea la página catVincLaboralSucces.php la cual muestra la vista catVincLaboral que tiene un formulario con el mismo nombre donde el usuario introduce los criterios de búsqueda, como este reporte se visualiza a nivel provincial solo le da la posibilidad de

seleccionar la provincia. Luego se envía una petición *submit* al controlador frontal el cual a través de los componentes de Symfony delega la responsabilidad al controlador del módulo *actions*, el cual accede al modelo que es el encargado del acceso a los datos almacenados en la base de datos, las clases del modelo acceden a los datos de la base de datos utilizando el ORM Propel que viene integrado en Symfony.

Cuando el *actions* procesa todos los datos, consume el servicio *web* que brinda GEYSED, generando todas las imágenes con ayuda de la librería OpenLayers, que finalmente se muestran en la vista que se construirá para mostrar los mapas.

2.7 Diagrama de Despliegue

El diseño también obtiene como resultado un modelo de despliegue, que describe todas las configuraciones de red sobre las cuáles debería implantarse el sistema. El modelo de despliegue incluye:

- Nodos, sus características y sus conexiones.
- Una correspondencia inicial de clases activas sobre los nodos.
- La vista arquitectónica del modelo de despliegue, que incluye a sus elementos relevantes para la arquitectura. (21)

En el siguiente diagrama se representan los diferentes nodos en los que se desplegará el sistema. Para el mismo se necesitan tres nodos para desplegar respectivamente el servidor de aplicaciones, el de base de datos y el del servicio de GEYSED. Además se necesitan clientes que interactúen con la aplicación, el diagrama además muestra el protocolo de conexión entre ellos.

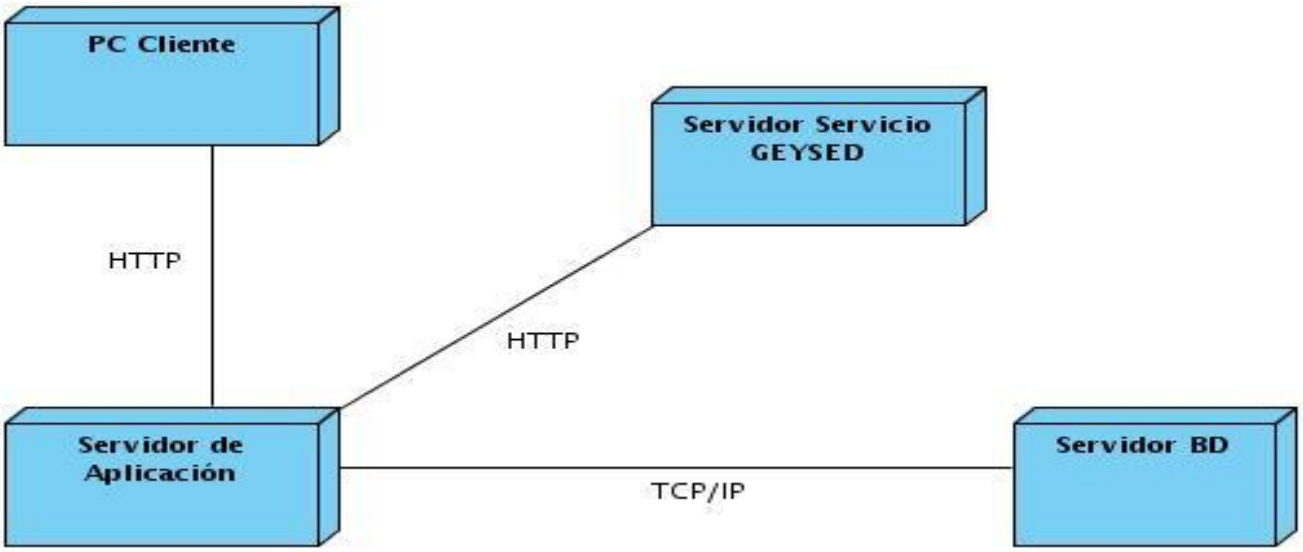


Fig 7. Diagrama de Despliegue

2.8 Seguridad

El sistema brinda la posibilidad de restringir la ejecución de una acción a usuarios con ciertos privilegios ya que necesitan estar autenticados antes de acceder a alguna característica o a partes de la aplicación.

Añadir esta seguridad a una aplicación requiere dos pasos: declarar los requerimientos de seguridad para cada acción y autenticar a los usuarios con privilegios para que puedan acceder estas acciones seguras.

Antes de ser ejecutada, cada acción pasa por un filtro especial que verifica si el usuario actual tiene privilegios de acceder a la misma. Los privilegios están compuestos por dos partes:

- Las acciones seguras requieren que los usuarios estén autenticados.
- Las credenciales son privilegios de seguridad agrupados bajo un nombre y que permiten organizar la seguridad en grupos.

Para restringir el acceso a una acción se crea y se edita un archivo de configuración llamado “security.yml” donde se especifican los requerimientos de seguridad que los usuarios deberán satisfacer para cada acción o para todas.

Lo que sucede cuando un usuario trata de acceder una acción restringida depende de sus credenciales.

Por ejemplo:

- Si el usuario está autenticado y tiene las credenciales apropiadas, entonces la acción se ejecuta.
- Si el usuario no está autenticado, es redireccionado a la acción de login.
- Si el usuario está autenticado, pero no posee las credenciales apropiadas (es decir, que no están definidas en el archivo security.yml), será redirigido a la acción segura por defecto.

2.9 Pautas del Diseño

Las pautas de diseño que se utilizarán serán las definidas para alasMEDIGEN. Los formularios que sean creados deben estar centrados y tener bordes con valores 0. La primera fila del formulario debe contener el nombre de la operación que se pretende realizar con el formulario. El estilo a utilizar para dicha fila es " hOperation".
Por ejemplo si la finalidad del formulario fuera visualizar la cantidad de discapacitados con vinculación laboral, el encabezado sería el siguiente:

```
<td class="hOperation"> Cantidad de discapacitados con vinculación laboral </td>
```

Cuando se le solicite datos al usuario se hará a través de una tabla de dos columnas. En la columna izquierda se ubicará el nombre del dato solicitado alineado a la derecha y en la columna de la derecha el componente adecuado para seleccionar el dato alineado a la izquierda. El estilo a utilizar en la columna izquierda es “nombrecampo”, el de los componentes de la columna derecha es “entradaplana”.

Por ejemplo, se le solicita al usuario seleccionar la provincia.

```
<td  align="right" class="nombrecampo">Provincias</td>
        <td  align="left" class="nombrecampo" >
          <select name="id_prov" id="id_prov" class="entradaplana" >
            <option value=""> &lt;&lt;  Seleccione  &gt;&gt;</option>
```

```
<?php foreach ($provincias as $prov): ?>
<option value="<?php echo $prov->getIdProvincia() ?>">
<?php echo  utf8_decode( $prov->getDescripcion() ) ?></option>
<?php endforeach; ?>
</select>
</td>
```

Los botones para efectuar operaciones sobre el formulario se ubicaran en la parte inferior derecha del formulario y utilizarán el estilo “sbttn”.
Ejemplo de un formulario que los botones de las operaciones son ''Aceptar '' y ''Cancelar ''.

```
<tr>
<td >
<?php echo submit_tag('Aceptar','class=sbttn')?>
<?php echo button_to('Cancelar ', ' ' , 'class="sbttn" type="submit"')?>
</td>
</tr>
```

2.8 Prototipo de Interfaz

Para el SIGM se elaboró un prototipo de interfaz, planificando un diseño orientado a las necesidades de sus futuros usuarios y al ambiente en el que será desplegado. Se ha tenido en cuenta el orden de los elementos que aparecen en el instrumento, acorde al orden de prioridad de los elementos que lo componen.

Descripción de los elementos que componen las interfaces del prototipo diseñado para el SIGM.

Fig 8. Prototipo de Interfaz

2.9 Tratamiento de errores

El módulo contará con el tratamiento de errores, pues los datos que van a ser introducidos por el usuario serán validados en el lado del servidor mediante archivos .yml vinculados a los formularios de las interfaces. Una vez que es detectado un error (datos sin introducir) se lanzará un mensaje de error, indicándole cuál es el dato que falta y dirigirá al usuario al campo en el cuál está el error, el usuario podrá seguir introduciendo los datos en el formulario.

Conclusiones del capítulo

Este capítulo brinda una visión del módulo a desarrollar. Se obtuvo un modelo de dominio por la poca definición de procesos del negocio que existe. Se definieron los requerimientos funcionales y no funcionales, así como la descripción de los casos de uso del sistema. La aplicación de patrones en el diseño permite agilizar y hacer más eficiente el proceso de codificación. Se realizó la distribución física del módulo a través de un modelo de despliegue, que describe todas las configuraciones sobre las cuales debe implementarse el mismo. Los diagramas de diseño y despliegue generados se consideran

la entrada principal para el Flujo de Trabajo de Implementación. Además se explicó la forma de tratar los errores dentro del módulo y la seguridad del mismo.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS

En este capítulo se aborda sobre la implementación del módulo. Se realiza la representación de los diagramas de componentes del módulo. Se brinda una descripción de los principales métodos implementados así como se muestran imágenes de la interfaz del módulo. Además se incluyen las principales pruebas a realizar.

3.1 Modelo de implementación.

El modelo de implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes se pueden encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos. (18)

3.1.1 Diagramas de componentes.

Un diagrama de componentes representa la separación de un sistema de *software* en componentes físicos (por ejemplo archivos, cabeceras, módulos, paquetes, etc.) y muestra las dependencias entre estos componentes. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema. (21)

A continuación se exponen los diagramas de componentes de los Casos de Uso VG pacientes con malformaciones congénitas y VG pacientes discapacitados con vinculación laboral. (Para visualizar el resto de los diagramas acudir al expediente de proyecto)

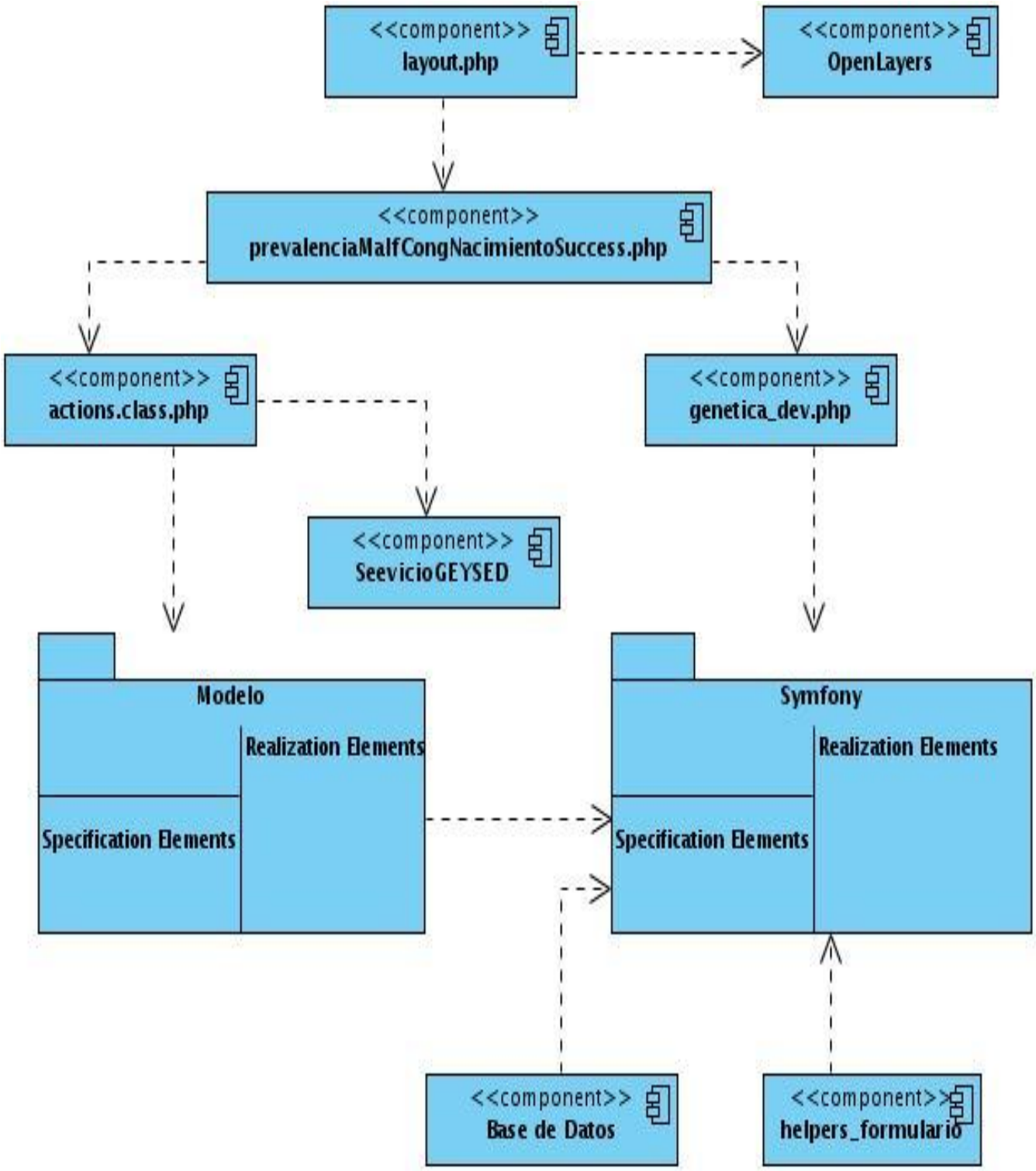


Fig 9.Diagrama de Componente. CU VG pacientes con malformaciones congénitas

Cuando el usuario accede al sistema alasMEDIGEN, al módulo de visualización geográfica y selecciona la opción Prevalencia de Malformaciones Congénitas, se crea la página

prevalenciaMalfCongNacimientoSucces.php cuyo contenido se incluye en el layout. Las peticiones son atendidas por el controlador frontal genetica.php que a través de los componentes de Symfony delega la responsabilidad al controlador del módulo *actions*, el cual accede al modelo que es el encargado del acceso a los datos almacenados en la base de datos, las clases del modelo acceden a los datos de la base de datos utilizando el ORM Propel que viene integrado en Symfony.

Cuando el *actions* procesa todos los datos, consume el servicio *web* que brinda GEYSED, generando todas las imágenes con ayuda de la librería OpenLayers, que finalmente se muestran en la vista que se construirá con la ayuda de los helpers de formulario de symfony para mostrar los mapas.

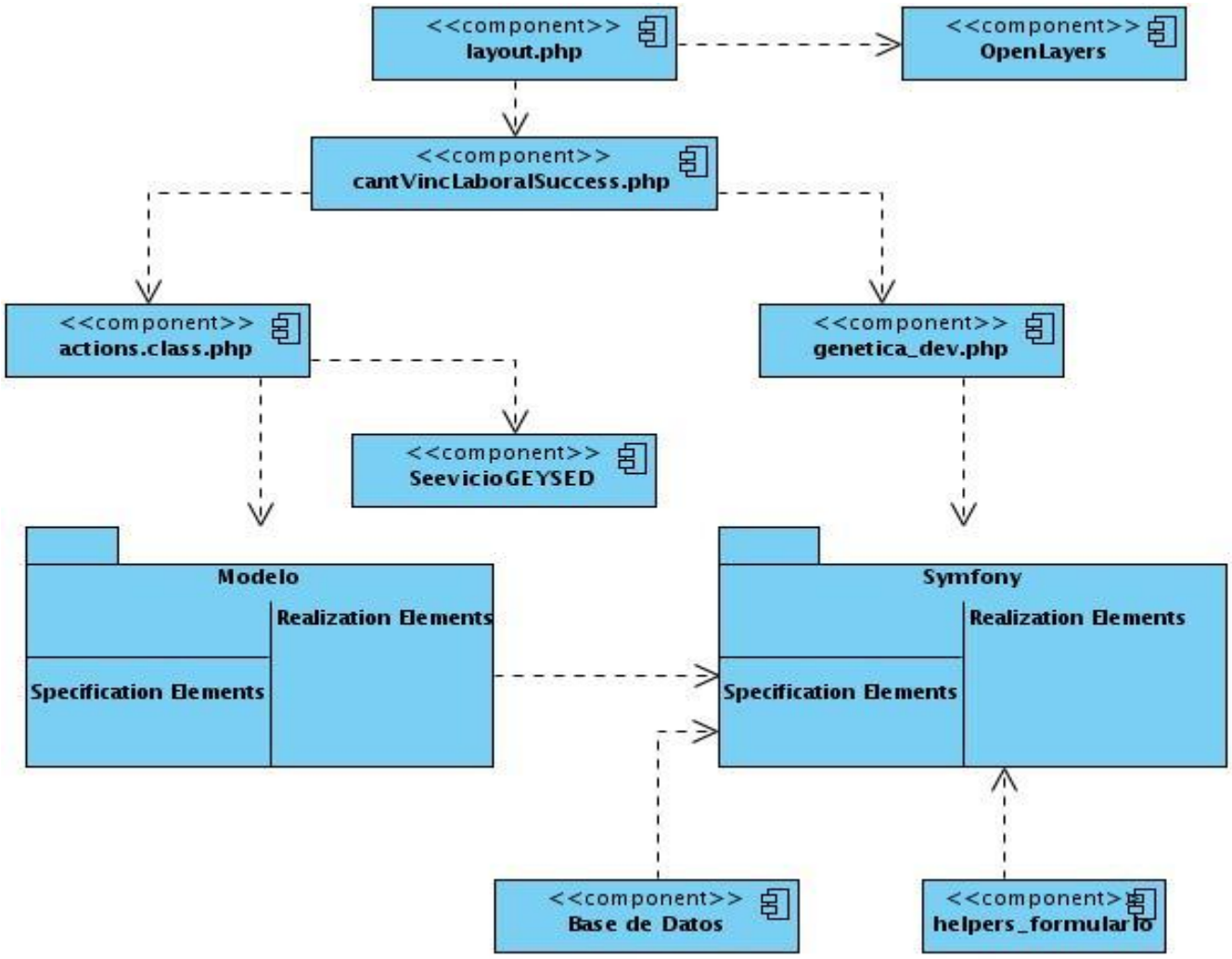


Fig 10.Diagrama de Componente. CU VG pacientes discapacitados con vinculación laboral

Cuando el usuario accede al sistema alasMEDIGEN, al módulo de visualización geográfica y

selecciona la opción Vinculación Laboral, se crea la página cantVincLaboralSucces.php cuyo contenido se incluye en el layout. Las peticiones son atendidas por el controlador frontal genetica.php que a través de los componentes de Symfony delega la responsabilidad al controlador del módulo *actions*, el cual accede al modelo que es el encargado del acceso a los datos almacenados en la base de datos, las clases del modelo acceden a los datos de la base de datos utilizando el ORM Propel que viene integrado en Symfony.

Cuando el *actions* procesa todos los datos, consume el servicio *web* que brinda GEYSED, generando todas las imágenes con ayuda de la librería OpenLayers, que finalmente se muestran en la vista que se construirá con la ayuda de los helpers de formulario de symfony para mostrar los mapas.

3.2 Estilos de codificación definidos para alasMEDIGEN

Con el objetivo de hacer que el código fuente sea más legible y así propiciar mayor comprensión de la programación realizada, es necesario definir estilos o estándares de codificación para el desarrollo de cualquier sistema.

A continuación se definen los siguientes estilos para la codificación dentro de la implementación de los componentes:

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

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

El sangrado del texto debe ser siempre de 4 espacios. No utilizar los tabulador (todos los editores no interpretan el tab de la misma manera).

Los nombres de las variables y funciones tienen que ser siempre fáciles de leer, procurando que sean palabras en minúsculas con significado claro análogo a la información que almacenan o la acción que realizan. Si realmente se necesita más de una palabra, deben ponerse juntas, poniendo la inicial de cada palabra en mayúscula siempre que no sea la primera, procurando mantenerlas tan breves como sea posible. Utilizar nombres en plural para arreglos o matrices de objetos.

Ejemplos:

Etiquetas php:

```
<?php echo label_for('label1','Provincial','class="nombrecampo"') ?>

<?php echo radiobutton_tag('nivel','Nacional',false) ?>
```

Variables:

$nivel;

$indice;

$vinculados;

$municipios;

Los bloques de código siempre deben estar encerrados por llaves (incluso si solo constan de una línea).

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

Todas las funciones y clases deben tener comentarios. **Los comentarios** deben ser añadidos de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones o variables.

En los comentarios de las funciones debe aparecer el autor de la función, el objetivo de la misma, y una descripción de cada uno de los parámetros que se le pasen.
Ejemplo:

```
/* El objetivo de esta función es.....
*nombre del autor
*parámetro1 es el id de la provincia
*parámetro2 es el token del usuario
*/
public function getVinculados($provincia,$token)
{
/* bloque de instrucciones.....
}
```

3.3 Código fuente de las principales clases

Se presenta el código fuente de las principales clases del Módulo de Visualización Geográfica. La clase vgActions contiene todas las funcionalidades necesarias para manipular la información gestionada en cada una de los templates donde se muestran los diferentes reportes.

```
class vgActions extends sfActions {

    /**
     * Executes index action
     *
     */
    public function executeIndex() {...}

    public function executePrevalenciaMalfCongNacimiento() {...}

    public function executeCantDiscapacitados() {...}

    public function executeMayorCantDiscapacitados() {...}

    public function executeCantTipoDiscapacidad() {...}

    public function executeCantVincLaboral() {...}

    public function executeCasosAtendTeleGeneticas() {...}

    public function executeEnferGeneticasRangoEdades() {...}

}
```

A continuación se muestra un fragmento de código del método CantVincLaboral con una breve explicación del mismo.

```
public function executeCantVincLaboral() {
    $token = $this->getUser()->getToken();
    $this->provincias = proxyUbicacion::getAllProvincias($token);
    if ($this->getRequest()->getMethod() == sfWebRequest::POST) {
        $provincia = $this->getRequestParameter('id_prov');
        $vinculados = array();
        $this->rangos = array();
        $values = array();
        $vinculados = DisDiscapacitadoPeer::getVinculados($provincia, $token);
        $j = 0;

        $keys = array_keys($vinculados);
        foreach($keys as $key)
        {
           if (!in_array($vinculados[$key], $this->rangos))
           {
               $this->rangos[$j] = $vinculados[$key];
               $values[$j] = array();
               $values[$j][] = $key;
               $j++;
           }
           else
           {
               for($k = 0; $k < $j; $k++)
               {
                   if($this->rangos[$k] == $vinculados[$key])
                     $values[$k][] = $key;
               }
           }
        }
```

```
$wsdl = 'http://10.54.12.17/genesig/htdocs/cartoserver.wsdl.php?mapId=geoweb.geoweb&'.time();
$soapCliente = new SoapClient($wsdl, array());
$map = new MapRequest();
$colores_general = sfConfig::get('app_colores_list');
for($i = 0; $i < count($this->rangos); $i++)
$colores[] = '#'.$colores_general[$i];
$this->colores=$colores;
$map->mapaTematicoRequest = new MapaTematicoRequest('Municipios', $colores, $values, $this->rangos, $provincia);
$soapCliente->__setCookie('PHPSESSID', 'sadasdasdasdasd');
$respuesta = $soapCliente->getMap($map);
```

```
class DisDiscapacitadoPeer extends BaseDisDiscapacitadoPeer
{
    public static function getVinculados($provincia,$token = null)
    {
        $vinculados=array ();
        $municipios = proxyUbicacion::getAllMunicipios($token, $provincia);
        foreach ($municipios as $municipio)
            {
            $vinculados[$municipio->getIdMunicipio()]=self::getCantMunicipio($municipio->getIdMunicipio());
            }
            return $vinculados;
    }
    public static function getCantMunicipio($mun_id)
     {
        $e = new Criteria();
        $e->addJoin(DisDiscapacitadoPeer::ID_PERSONA,DgPersonaPeer::ID_PERSONA);
        $e->add(DisDiscapacitadoPeer::VINCLABORALANTES,"si");
        $e->add(DgPersonaPeer::ID_MUNNACE, $mun_id);
        return DisDiscapacitadoPeer::doCount($e);
     }
}
```

El método CantVincLaboral() permite obtener la cantidad de discapacitados existentes en cada uno de los municipios de una provincia determinada. Se hace una llamada al método getVinculados() de la clase DisDiscapacitadoPeer en el cual se obtienen todos los municipios de la provincia especificada y con la llamada al método getCantMunicipio() se obtiene la cantidad de discapacitados de cada municipio de esa provincia. Una vez obtenidos los resultados del reporte, se agrupan en rangos según la cantidad de discapacitados, posteriormente se le asigna a cada uno de esos rangos un color para identificarlo en la leyenda, a continuación, se crea una instancia de MapaTematicoRequest, especificándole el nivel del mapa, la gama de colores, los valores y rangos, finalmente se le envía esta

instancia como parámetro al servicio *web* de GEYSED para que construya el mapa que se visualizará en caso que el proceso de generación sea exitoso

3.4 Librería usada en los reportes

Una forma de mostrar información es mediante el uso de mapas geográficos con el fin de propiciar una interfaz más amigable para el usuario. En el pasado, la creación de mapas interactivos se reservaba para las grandes empresas o expertos con una gran cantidad de dinero. Hoy en día, con las herramientas adecuadas, cualquier persona puede crear fácilmente un mapa *web* con poco o ningún conocimiento de la geografía, la cartografía, o la programación. Los mapas *web* se espera que sean rápidos, precisos y fáciles de usar. Puesto que están en línea, se espera que sea accesible desde cualquier lugar en casi cualquier plataforma. Hay sólo unas pocas herramientas que cumplen con todas estas expectativas. OpenLayers es una de estas herramientas. Es gratuito, de código abierto, y muy poderoso. Proporciona a los desarrolladores principiantes y profesionales experimentados de SIG, con una biblioteca robusta, una fácil, rápida, moderna e interactiva creación de aplicaciones de mapeo *web*.

Open Layers

OpenLayers es una biblioteca JavaScript para la fabricación de mapas *web* interactivos, se pueden ver en casi cualquier navegador *web*. Dado que es una biblioteca del lado del cliente, no requiere *software* especial del lado del servidor o la configuración, se puede usar sin descargar nada. Se ha vuelto muy popular entre muchos desarrolladores apasionados y una buena comunidad de ayuda.

Obtener imágenes de mapa

OpenLayers reside en el lado del cliente. Una de las principales tareas que el cliente lleva a cabo es conseguir imágenes de mapa de un servidor de mapas. En esencia, el cliente tiene que pedir al servidor de mapas lo que desee ver. Cada vez que se navegue o se haga *zoom* en torno al mapa, el cliente tiene que hacer nuevas peticiones al servidor, porque está tratando de visualizar algo diferente. OpenLayers se encarga de realizar todo esto por el cliente y las llamadas que están sucediendo vía asíncrona JavaScript (AJAX) al servidor de mapa. Básicamente, envía peticiones de imágenes de mapa a un servidor de mapas, cada vez que interactúa con el mapa, a continuación, Open Layers devuelve las piezas de imágenes de mapas todas juntas lo cual parece un mapa grande, sin fisuras.

3.5 Interfaz de la Aplicación

Fig 11.Interfaz de entrada al sistema

Fig 12.Interfaz Principal

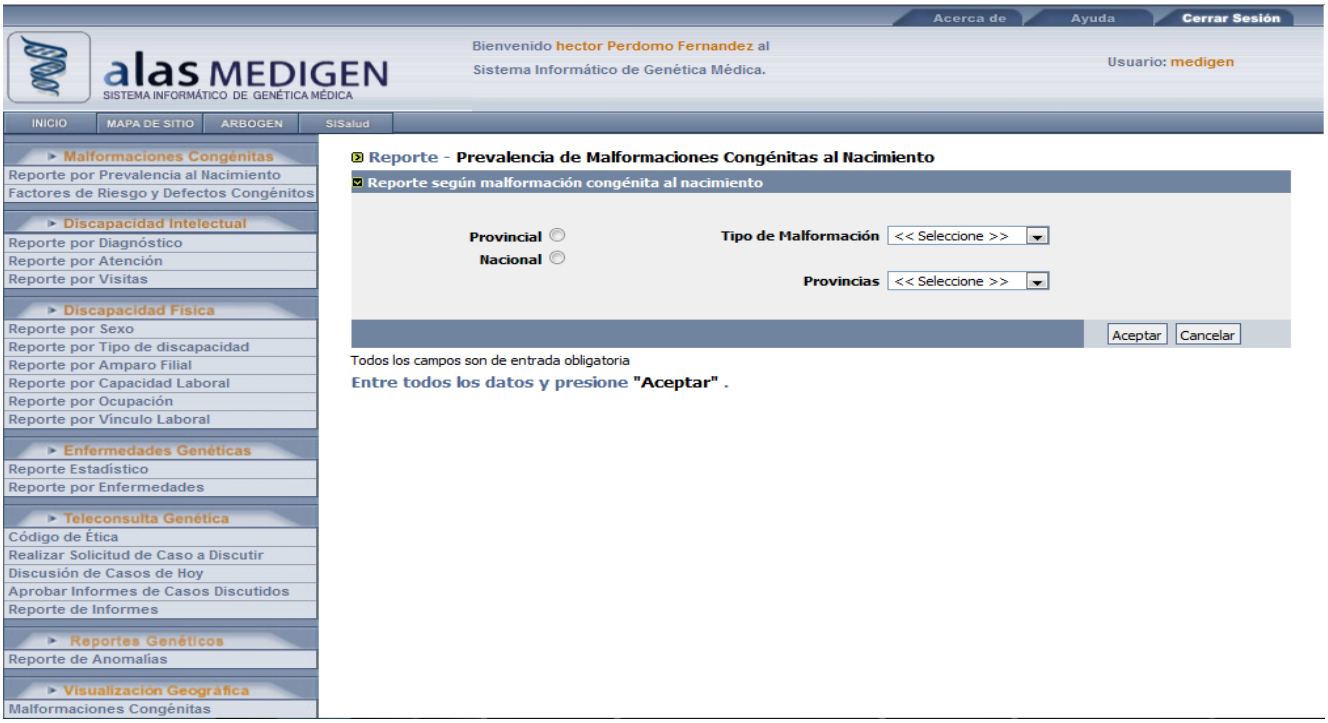


Fig 13.Interfaz del Reporte Prevalencia de Malformaciones Congénitas

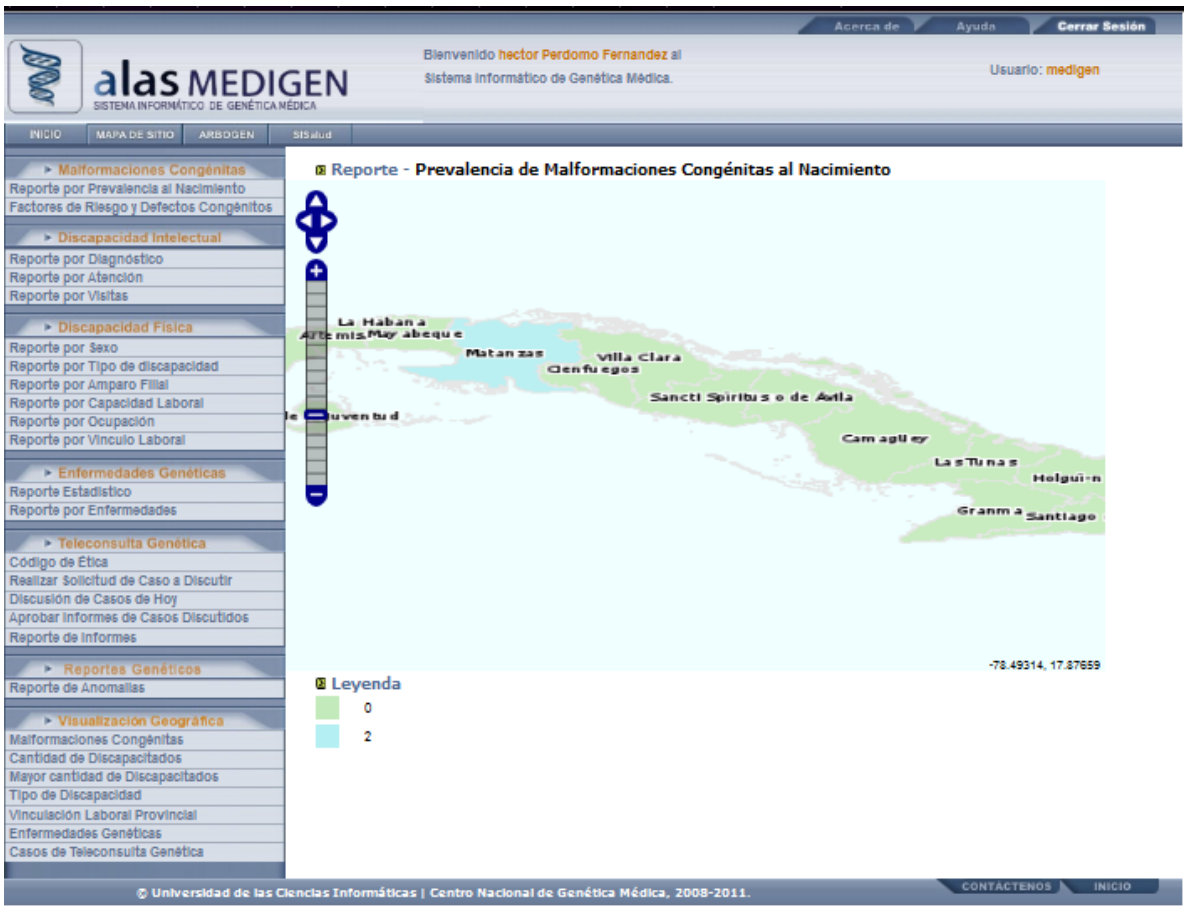


Fig 14.Interfaz resultado del Reporte Prevalencia de Malformaciones Congénitas

3.7 Validación a nivel del desarrollador

Con el propósito de lograr mayor calidad y eficiencia, el desarrollador debe validar el sistema con el objetivo de comprobar que cumple con las funcionalidades específicas. Durante la implementación del módulo de visualización geográfica, se realizaron validaciones tratando de encontrar fallos en la aplicación para corregirlos y así lograr que las entradas se acepten de forma adecuada, se produzca una salida correcta y que la integridad de la información se mantenga. Para ello se explotaron al máximo las entradas de datos con el objetivo de encontrar vulnerabilidades en el sistema, además se verificó sistemáticamente que cada una de las funcionalidades necesarias para el cumplimiento de los requerimientos del sistema hicieran lo esperado, para lograr de ese modo satisfacer las necesidades del cliente.

Las principales validaciones utilizadas estuvieron encaminadas a evitar que el usuario fuera a dejar campos sin seleccionar, y así propiciar que la entrada de información al sistema fuera correcta. Estas validaciones se realizaron utilizando archivos “yml”.

Dejar de introducir datos de entrada obligatoria es una manera de verificar la consistencia de la aplicación que se está desarrollando. A continuación se muestra un ejemplo de los mensajes que aparecen cuando no se introducen datos que son de entrada obligatoria:

Fig 15. Interfaz con mensajes asociados a la entrada de datos obligatorios

3.8 Pruebas

Los casos de prueba representan los datos que se utilizarán como entrada para ejecutar el *software* a probar. Más concretamente los casos de prueba determinan un conjunto de entradas, condiciones de ejecución y resultados esperados para un objetivo particular. Como veremos posteriormente, cada técnica de pruebas proporciona criterios distintos para generar estos casos o datos de prueba. Por lo tanto, durante la tarea de generación de estos casos, se han de confeccionar los distintos casos de prueba según la técnica o técnicas identificadas previamente. La generación de cada caso de prueba debe ir acompañada del resultado que ha de producir el *software* al ejecutar dicho caso. (22)

Existen varios tipos de prueba que se le pueden realizar a un sistema, después de un estudio realizado se decidió realizar las pruebas a nivel de sistema utilizando el método de caja negra.

Método de Caja Negra.

El método de caja negra permite definir las entradas al sistema y los resultados esperados de estas entradas, se realizan con el objetivo de asegurar que las funcionalidades del sistema cumplen con lo que se espera de ellas. Este método permite identificar claramente las entradas y salidas y estudiar las relaciones que existen entre ellas, permitiendo así maximizar la eficiencia de los sistemas sin tener que introducirnos en los procesos complejos que se encuentran en la Caja Negra.

3.8.1 Casos de prueba

Un caso de prueba permite detallar la forma en que se va a probar el sistema, incluyendo los datos de entrada con las que se realizará la prueba correspondiente, las condiciones de ejecución y resultados obtenidos. (21)

Deben verificar:

- Si el producto satisface los requerimientos del usuario, tal y como se describe en las especificación de los requerimientos.
- Si el producto se comporta como se desea, tal y como se describe en las especificaciones funcionales del diseño.

A continuación se exponen los casos de prueba de los Casos de Uso VG pacientes con malformaciones congénitas y VG pacientes discapacitados con vinculación laboral. (Para visualizar el resto de los casos de pruebas acudir al expediente de proyecto)

Tabla 8. Caso de Prueba CU VG prevalencia de pacientes con malformaciones congénitas

Descripción general
El caso de uso inicia cuando el actor accede a la opción Enfermedades Genéticas, el sistema brinda la posibilidad de seleccionar criterios de búsqueda como un rango de edades, la enfermedad genética y la región geográfica a representar, el actor introduce los datos que considera como criterios para realizar una representación, el sistema en dependencia de la región geográfica seleccionada, la enfermedad genética y el rango de edades, muestra una interfaz con un mapa geográfico representando a través de escala de colores la cantidad de pacientes con esa enfermedad genética en la región especificada que se encuentren en ese rango de edades, el caso de uso termina.
Condiciones de ejecución
Que se haya seleccionado la región, la enfermedad genética y el rango de edades a representar.
SC Pacientes con malformaciones congénitas

		Variables				
Escenario	**Descripción**	**nivel**	**Tipo_malf**	**id_prov**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Representar pacientes con malformaciones congénitas en el sistema con datos correctos.	Permite representar por región seleccionada la cantidad de pacientes con cierta malformación congénita.	V	V	V	En dependencia de los criterios seleccionados para representar, el sistema muestra un mapa geográfico que a través de escala de colores muestra la cantidad de casos existentes.	Visualización Geográfica/ Malformaciones Congénitas/ Después de seleccionar los criterios de representación, dar clic en la parte inferior derecha en el botón Aceptar.
EC 1.2Representar pacientes con malformaciones congénitas en el sistema con datos incorrectos.		I(/*)	V	V	En caso que no seleccione correctamente los criterios, se muestran los campos vacíos con el siguiente mensaje: “Campo Obligatorio”.	
		V	I(/*)	V		
		V	V	I(/*)		
		V	V	V		

Descripción de las variables				
No	**Nombre de campo**	**Clasificación**	**Valor Nulo**	**Descripción**
1	nivel	Radio	No	Escoger dato
2	Tipo_malf	Lista desplegable	No	Escoger dato
3	id_prov	Lista desplegable	Si	Escoger dato

Tabla 9. Caso de Prueba CU VG pacientes con vinculación laboral

Descripción general

El caso de uso inicia cuando el actor accede a la opción Vinculación Laboral, el sistema brinda la posibilidad de seleccionar criterios de búsqueda como la región geográfica a representar, el actor introduce los datos que considera como criterios para realizar una representación, el sistema en dependencia de la región geográfica seleccionada, muestra una interfaz con un mapa geográfico representando a través de escala de colores la cantidad de pacientes con vinculación laboral existentes en la región especificada, el caso de uso termina.
Condiciones de ejecución
Que se haya seleccionado la región a representar.

SC Pacientes discapacitados con vinculación laboral

		Variables		
Escenario	**Descripción**	**id_prov**	**Respuesta del sistema**	**Flujo central**
C 1.1 Representar pacientes con vinculación laboral en el sistema con datos correctos.	Permite representar por región seleccionada la cantidad de pacientes con vinculación laboral.	V	En dependencia de los criterios seleccionados para representar, el sistema muestra un mapa geográfico que a través de escala de colores muestra la cantidad de pacientes con vinculación laboral.	Visualización Geográfica/ Vinculación Laboral/ Después de seleccionar los criterios de representación, dar clic en la parte inferior derecha en el botón Aceptar.
EC1.2 Representar pacientes con vinculación laboral en el sistema con datos incorrectos.		I(/*)	En caso que no seleccione correctamente los criterios, se muestran los campos vacíos con el siguiente mensaje: “Campo Obligatorio”	

Descripción de las variables				
No	**Nombre de campo**	**Clasificación**	**Valor Nulo**	**Descripción**
1	id_prov	Lista desplegable	No	Escoger dato

Conclusiones del capítulo

En este capítulo se representaron los diagramas de componentes de cada uno de los casos de usos, los cuales están basados en las definiciones realizadas durante el diseño. Además se brindó una breve descripción de los principales métodos implementados para obtener un mejor entendimiento de los mismos. Se mostraron algunas de las interfaces del módulo y se expusieron algunos ejemplos de las

validaciones realizadas a nivel de desarrollador, las cuales permitieron detectar y solucionar los errores identificados. La prueba realizada demostró que el módulo responde acertadamente a la interacción con el usuario.

CONCLUSIONES

Con el desarrollo del módulo de visualización geográfica de la información que se gestiona en alasMEDIGEN se ha arribado a las siguientes conclusiones:

Las herramientas, metodología y tecnologías utilizadas, brindaron el sostén necesario para lograr un producto con los requerimientos deseados.

El análisis y diseño del módulo, proporcionaron una correcta implementación del mismo.

Con la implementación del módulo se obtuvo un valor añadido para el sistema alasMEDIGEN, conforme con los requerimientos propuestos, lo cual facilitará la mejor realización de los estudios genéticos en el territorio nacional, relacionados con el origen de las enfermedades genéticas en las diferentes regiones del país. Esto permitirá un mejor análisis de los pacientes, según los riesgos, ya sean de origen genético o ambiental, a partir de la ubicación geográfica de los casos genéticos.

RECOMENDACIONES

A partir de los resultados o beneficios que proporciona este trabajo de diploma, se proponen las siguientes recomendaciones:

Construcción de una API con su respectiva documentación para el consumo de los servicios de GEYSED para la visualización de los mapas en alasMEDIGEN.

Hacer una optimización al código JavaScript utilizado para la visualización de los mapas en la *web* con el fin de mejorar el rendimiento.

Desarrollar otra versión de la aplicación donde se analice la información hasta niveles más específicos, como municipal.

REFERENCIAS BIBLIOGRÁFICAS

1. *Genética y enfermedad. Concepto de genética médica.* **Guillén Navarro, E., Ballesta Martínez, M. J. y López González, V.** Murcia. España : s.n., 2011.

2. *SIG-MES: Análisis del sistema para la representación geográfica de los Centros Universitarios del Ministerio de Educación Superior Cubana.* **Ávila Bárzaga, Olaidis.** La Habana : Facultad 6. Universidad de las Ciencias Informáticas, 2011.

3. *Integración de capacidades de visualización geográfica en el software de gestión de proyectos LEADER.* **Latre, M. Á., y otros, y otros.** 72, Universidad de Zaragoza. España : s.n., 2001.

4. eptisa. *eptisa.* [En línea] 2011. [Citado el: 29 de octubre de 2011.] http://www.eptisa.com/es/experiencia/sistema-de-informacion-geografica-decorreos/mercados/.

5. WORDPRESS. [En línea] 9 de Junio de 2011. [Citado el: 1 de noviembre de 2011.] http://www.turismoynegocios.wordpress.com/tag/sigtur.

6. MappingInteractivo. *MappingInteractivo.* [En línea] 2011. [Citado el: 28 de octubre de 2011.] http://www.mappinginteractivo.com/plantilla-ante.asp?id_articulo=1548.

7. **Pantoja Saldivar, Yoenis.** Portal Gladiadores. *Portal Gladiadores.* [En línea] 18 de Noviembre de 2011. [Citado el: 18 de Noviembre de 2011.] http://http://facultad6.uci.cu/index.php?option=com_k2&view=item&id=471:sig-rutas-v10&Itemid=82.

8. *APLICACIÓN DE LA METODOLOGÍA RUP PARA EL DESARROLLO RÁPIDO DE APLICACIONES BASADO EN EL ESTÁNDAR J2EE.* **Rueda Chacón , Julio César.** Guatemala : Universidad de San Carlos de Guatemala, 2006.

9. *Sistema Informático para la Red Nacional de Genética Médica (SIGM): Registro Cubano de Retraso Mental versión 2.0.* **Crespo García, Yosuán y Sterling Velásquez, Otto Leosdán.** Ciudad de la Habana : Facultad 6. Universidad de las Ciencias Informáticas, 2007.

10. PHP. [En línea] [Citado el: 6 de Diciembre de 2011.] http://www.php.net/manual/es/intro-whatcando.php.

11. PHP. [En línea] 2001. [Citado el: 6 de Diciembre de 2011.] http://www.php.net/.

12. MySQL. [En línea] 20 de 12 de 2011. [Citado el: 8 de Enero de 2012.] http://dev.mysql.com/doc/mysql/en.

13. Ubuntu. [En línea] [Citado el: 6 de 12 de 2011.] http://doc.ubuntu-es.org/HTTPD_Servidor_web_Apache2.

14. Visual Paradigm. [En línea] [Citado el: 7 de Diciembre de 2011.] http://www.visual-paradigm.com/support/documents/vpumluserguide.jsp.

15. **Zaninotto, François y Potencier, Fabien.** *Symfony la guía definitiva.*

16. NetBeans. [En línea] Oracle, 2012. [Citado el: 8 de Diciembre de 2011.] http://netbeans.org/features/ide/index.html#.

17. **Higuera, Santiago.** OpenLayers. *OpenLayers.* [En línea] 7 de Junio de 2010. [Citado el: 4 de Noviembre de 2011.] http://www.openlayers.ingemoral.es/manualOpenLayers.html.

18. Parte II. [aut. libro] Ian Sommerville. *Ingeniería del Software. Séptima Edición.* Madrid : Pearson Educación S.A., 2005.

19. **Gracia, Joaquín.** Patrones de Diseño. [En línea] 27 de Mayo de 2005. [Citado el: 11 de Febrero de 2012.] http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php..

20. El mundo informático. *El mundo informático.* [En línea] 8 de Mayo de 2007. [Citado el: 12 de Noviembre de 2011.] http://jorgesaavedra.wordpress.com/category/patrones-grasp/.

21. Parte II. [aut. libro] Ivar Jacobson, Grady Booch y James Rumbaugh. *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Educación S.A., 2000.

22. Ingeniería del software un enfoque práctico. [En línea] [Citado el: 8 de Mayo de 2012.] http://es.scribd.com/doc/7978336/Ingenieria-de-Software-Un-Enfoque-Practico-Pressman-5th-Ed..

23. Pruebas de Software. [En línea] [Citado el: 8 de Mayo de 2012.] http://gemini.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node55.html..

BIBLIOGRAFÍA

1. *Genética y enfermedad. Concepto de genética médica.* **Guillén Navarro, E., Ballesta Martínez, M. J. y López González, V.** Murcia. España : s.n., 2011.

2. *SIG-MES: Análisis del sistema para la representación geográfica de los Centros Universitarios del Ministerio de Educación Superior Cubana.* **Ávila Bárzaga, Olaidis.** La Habana : Facultad 6. Universidad de las Ciencias Informáticas, 2011.

3. *Integración de capacidades de visualización geográfica en el software de gestión de proyectos LEADER.* **Latre, M. Á., y otros, y otros.** 72, Universidad de Zaragoza. España : s.n., 2001.

4. eptisa. *eptisa.* [En línea] 2011. [Citado el: 29 de octubre de 2011.] http://www.eptisa.com/es/experiencia/sistema-de-informacion-geografica-decorreos/mercados/.

5. WORDPRESS. [En línea] 9 de Junio de 2011. [Citado el: 1 de noviembre de 2011.] http://www.turismoynegocios.wordpress.com/tag/sigtur.

6. MappingInteractivo. *MappingInteractivo.* [En línea] 2011. [Citado el: 28 de octubre de 2011.] http://www.mappinginteractivo.com/plantilla-ante.asp?id_articulo=1548.

7. **Pantoja Saldivar, Yoenis.** Portal Gladiadores. *Portal Gladiadores.* [En línea] 18 de Noviembre de 2011. [Citado el: 18 de Noviembre de 2011.] http://http://facultad6.uci.cu/index.php?option=com_k2&view=item&id=471:sig-rutas-v10&Itemid=82.

8. *APLICACIÓN DE LA METODOLOGÍA RUP PARA EL DESARROLLO RÁPIDO DE APLICACIONES BASADO EN EL ESTÁNDAR J2EE.* **Rueda Chacón , Julio César.** Guatemala : Universidad de San Carlos de Guatemala, 2006.

9. *Sistema Informático para la Red Nacional de Genética Médica (SIGM): Registro Cubano de Retraso Mental versión 2.0.* **Crespo García, Yosuán y Sterling Velásquez, Otto Leosdán.** Ciudad de la Habana : Facultad 6. Universidad de las Ciencias Informáticas, 2007.

10. PHP. [En línea] [Citado el: 6 de Diciembre de 2011.] http://www.php.net/manual/es/intro-whatcando.php.

11. PHP. [En línea] 2001. [Citado el: 6 de Diciembre de 2011.] http://www.php.net/.

12. MySQL. [En línea] 20 de 12 de 2011. [Citado el: 8 de Enero de 2012.] http://dev.mysql.com/doc/mysql/en.

13. Ubuntu. [En línea] [Citado el: 6 de 12 de 2011.] http://doc.ubuntu-es.org/HTTPD_Servidor_web_Apache2.

14. Visual Paradigm. [En línea] [Citado el: 7 de Diciembre de 2011.] http://www.visual-paradigm.com/support/documents/vpumluserguide.jsp.

15. **Zaninotto, François y Potencier, Fabien.** *Symfony la guía definitiva.*

16. NetBeans. [En línea] Oracle, 2012. [Citado el: 8 de Diciembre de 2011.] http://netbeans.org/features/ide/index.html#.

17. **Higuera, Santiago.** OpenLayers. *OpenLayers.* [En línea] 7 de Junio de 2010. [Citado el: 4 de Noviembre de 2011.] http://www.openlayers.ingemoral.es/manualOpenLayers.html.

18. Parte II. [aut. libro] Ian Sommerville. *Ingeniería del Software. Séptima Edición.* Madrid : Pearson Educación S.A., 2005.

19. **Gracia, Joaquín.** Patrones de Diseño. [En línea] 27 de Mayo de 2005. [Citado el: 11 de Febrero de 2012.] http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php..

20. El mundo informático. *El mundo informático.* [En línea] 8 de Mayo de 2007. [Citado el: 12 de Noviembre de 2011.] http://jorgesaavedra.wordpress.com/category/patrones-grasp/.

21. Parte II. [aut. libro] Ivar Jacobson, Grady Booch y James Rumbaugh. *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Educación S.A., 2000.

22. Ingeniería del software un enfoque práctico. [En línea] [Citado el: 8 de Mayo de 2012.] http://es.scribd.com/doc/7978336/Ingenieria-de-Software-Un-Enfoque-Practico-Pressman-5th-Ed..

23. Pruebas de Software. [En línea] [Citado el: 8 de Mayo de 2012.] http://gemini.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node55.html..

24. OGC. [En línea] 2011. http://www.opengeospatial.org/standards/wms.

25. **Hazzard, Erik.** *OpenLayers 2.10 Beginner's Guide.* [PDF] Birmingham, UK : Packt Publishing Ltd., 2011.

26. *DT-Active Open Layers User's Manual for Imaging Boards.* [PDF] Marlboro, MA : Data Translation, Inc., 2000.

27. *Rational Unified Process.* [PDF] Lexington, MA : Rational Software, 1998.

28. **Molpeceres, Alberto.** *Procesos de desarrollo: RUP, XP, FDD.* 2002.

29. **Visconti, Marcello y Astudillo, Hernán .** *Fundamentos de Ingeniería de Software.* [PDF] Valparaíso : Universidad Técnica Federico Santa María, 2005.

30. **La Rosa González, Yudiel , y otros, y otros.** *alasMEDIGEN 1.2: SISTEMA INFORMÁTICO DE GENÉTICA MÉDICA.* La Habana : s.n., 2010.

31. *GESTIÓN DEL CONOCIMIENTO Y GESTIÓN DE LA INFORMACIÓN.* **Bustelo Ruesta, Carlota y Amarilla Iglesias, Raquel .** 34, Andalucía : INFORAREA S.L., 2001.

32. *MySQL 5.0 Reference Manual.* s.l. : Oracle and/or its affiliates, 2011.

GLOSARIO DE TÉRMINOS

A

ADN: Ácido desoxirribonucleico. Material genético de casi todos los organismos vivos que controla la herencia y se localiza en el núcleo de las células.

Arquitectura: La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema. Una Arquitectura de Software, también denominada *Arquitectura lógica*, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco.

Artefacto: Resultado de trabajo parcial o final que es producido y usado durante un proyecto. Los artefactos son usados para capturar y llevar la información del proyecto.

B

Base de datos: Conjunto de registros cuantitativos y/o cualitativos interrelacionados que se almacenan con objeto de satisfacer las necesidades del proceso de información en una organización.

C

Consulta SQL: Consulta o búsqueda que se realiza en una base de datos.

H

Hardware: Componentes físicos de un ordenador o de una red, en contraposición con los programas o elementos lógicos que los hacen funcionar.

Host: Es usado en informática para referirse a las computadoras conectadas a una red.

HTML: HyperText Markup Language. Lenguaje de Marcas de Hypertexto. Lenguaje para elaborar páginas Web actualmente se encuentra en su versión 4. Fue desarrollado en el CERN. Gracias a él ves esta página.

I

Interfaz: Zona de contacto, conexión entre dos componentes de "hardware", entre dos aplicaciones o entre un usuario y una aplicación.

J

JavaScript: Lenguaje de programación para WWW desarrollado por Netscape. Pertenece a la familia Java pero se diferencia de este último en que los programas están incorporados en el fichero HTML. Es un script de Java que es interpretado por la aplicacion cliente, normalmente un navegador (Browser).

L

LEADER: es una de las cuatro iniciativas financiadas por los Fondos Estructurales de la UE y está diseñada para ayudar a los agentes del mundo rural a considerar el potencial a largo plazo de su región.

M

Módulo: porción de un programa de computadora.
Morbimortalidad: Mortalidad por causa de una enfermedad.

P

Plataforma GeneSIG: Plataforma creada en el Centro de Desarrollo de Geoinformática y Señales Digitales (GEySED) de la UCI tiene como principal objetivo realizar la representación geoespacial de la información asociada a negocios específicos, permitiendo además realizar análisis sobre dicha información.
Plataforma: Sistema que sirve como base para hacer funcionar determinados módulos de hardware o de software con los que es compatible.
Prevalencia: Número de casos clínicos o de portadores existentes en determinado momento en una comunidad. Se expresa en tasas.

S

Software: Programas o elementos lógicos que hacen funcionar un ordenador o una red, o que se ejecutan en ellos, en contraposición con los componentes físicos del ordenador o la red.

Socket: Designa un concepto abstracto por el cual dos programas (posiblemente situados en computadoras distintas) pueden intercambiar cualquier flujo de datos, generalmente de manera fiable y ordenada.

U

URL: Acrónimo de Universal Resource Locator (Localizador Universal de Recursos /Identificador Universal de Recursos). Sistema unificado de identificación de recursos en la red. Es el modo estándar de proporcionar la dirección de cualquier recurso en Internet.