

Universidad de las Ciencias Informáticas
Facultad 6



Título: “Sistema Informático para la Red Nacional de Genética Médica: Registro Cubano de Malformaciones Congénitas versión 2.0”.

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

Autoras: Leidy González Bernal.
Danayi Daniela Hernández Laguna.

Tutores: Ing. Irina Collazo Ávila.
Ing. Lien Costales Leiva.

Cotutor: Ing. Alden Hernández Gómez

Ciudad de la Habana, Cuba
Junio 2008

*Es de importancia para quien desee alcanzar una certeza en su investigación,
el saber dudar a tiempo.
Aristóteles*

DECLARACIÓN DE AUTORÍA

Declaramos ser autoras del presente trabajo 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 _____.

Leidy González Bernal

Danayi Daniela Hernández Laguna

Ing. Irina Collazo Ávila

Firma del Autor

Firma del Autor

Firma del Tutor

Ing. Lien Costales Leiva

Ing. Alden Hernández Gómez

Firma del Tutor

Firma del Cotutor

Agradecimientos

El agradecer a quienes han intervenido en la realización de este trabajo, el cual simboliza la culminación de nuestros estudios y muchos años de sacrificio no resulta para nada sencillo. Sin embargo si existen personas que merecen tratamiento especial y que no podemos dejar de mencionar, entre esas personas están Maikel, Franklin, Jommy y el profesor Yusdenis, que más que una ayuda en un momento determinado siempre se preocuparon por nuestras dudas, realmente sin su ayuda y amistad sobre todas las cosas nos hubiese resultado un poco mas difícil obtener nuestra meta. Agradecemos además a nuestro líder de proyecto Alfonso Claro por ser tan persistente en su trabajo y saber guiarnos siempre dentro del proyecto por el camino correcto. Agradecemos a nuestros tutores por el apoyo brindado, sobre todo a Irina y Lien por sabernos calmar en los momentos de desespero y confiar en nosotras, a Alden por su paciencia y ayuda en todo momento y por preocuparse tanto por nuestros resultados. Agradecemos a Alejandro por su ayuda en el desarrollo del proyecto, al igual que a Ramses por sus clases magistrales de Symfony, a todas las personas del proyecto que nunca dijeron un no ante cualquier duda y que siempre nos animaban en momentos difíciles. A todos nuestros compañeros de estudio que nos apoyaron durante estos 5 años de carrera. En fin a todas aquellas personas que de una forma u otra nos ayudaron a lograr nuestro sueño, de todo corazón muchas gracias.

Dedicatoria

De Leidy:

A mi mami por su amor, apoyo, confianza y sobre todo su dedicación incondicional, por ser tan especial y quererme tanto.

A mi papi por su cariño y preocupación a pesar de estar tan lejos de mi, se que ahora más que nunca le hubiese gustado ser partícipe de este momento.

A mis abuelas que adoro y que siempre han sabido ocupar el lugar de mis segundas madres, por su amor y preocupación sobre todo en estos últimos tiempos.

A mi abuelo Luis que aunque ya no está físicamente se que se sentiría orgulloso de mis resultados.

A Yanosky por todo su amor, comprensión y ayuda durante todos estos años de relación.

A Martica por siempre estar tan pendiente de mis resultados y quererme como la hija hembra que no tiene.

A mis amigos de siempre que más que amigos han sido como hermanos durante estos años de carrera, a Nelvis, Ana, Marta, Maiquel, Daniela y Rosana, se que los extrañaré muchísimo.

De Daniela:

A mi padre que fue siempre mi guía aunque ya no este conmigo se que esta muy orgulloso de su hijita.

A mi madre por su cariño, su apoyo y por quererme tanto.

A mis tías Puchita, Juana y tata que siempre me cuidaron y son mi segunda madre.

A Alden por su amor, comprensión y ayuda durante todos estos años.

A mis amigos de la universidad que siempre los tendré presente y los extrañare muchísimo en especial a Aida, Nelvis, Marta, Anita, Leidy, Maiqelin.

Resumen

Las malformaciones congénitas constituyen un grupo de enfermedades de tratamiento y rehabilitación no siempre exitoso, muchas de ellas de evolución crónica y con secuelas que representan una desventaja social con un alto costo para la familia y el estado, de difícil prevención y de alta mortalidad, por lo que su estudio ha constituido un hecho de interés durante años para los especialistas en genética. Para lograr un mejor resultado en las investigaciones científicas sobre las anomalías congénitas en el país, el Centro Nacional de Genética Médica (CNGM) en conjunto con la Facultad de Bioinformática de la Universidad de las Ciencias Informáticas desarrollaron la primera versión de una aplicación para la gestión de la información del Registro Cubano de Malformaciones Congénitas. Dicha aplicación no satisface todas las necesidades identificadas por el cliente y no fue desarrollada sobre la arquitectura de software propuesta por el Ministerio de Salud Pública (MINSAP) para sus productos informáticos, lo que provoca que los genetistas no puedan hacer uso de la aplicación para realizar sus investigaciones científicas.

Para lograr una gestión eficiente de la información referente a las malformaciones congénitas se crea la versión 2.0 del Registro Cubano de Malformaciones Congénita. Este nuevo producto dará solución a los problemas presentados en la primera versión y se integrará al Sistema Informático para la Red Nacional de Genética Médica (SIGM) con el objetivo de centralizar toda la información existente en Cuba sobre los trastornos genéticos.

Específicamente el objetivo del presente trabajo es desarrollar una aplicación capaz de brindar toda la información referente a los recién nacidos con defectos congénitos, y que dicha información sea accesible desde cualquier parte del país a través de su publicación en la Red de Salud de Cuba (INFOMED), permitiendo que los genetistas puedan realizar sus investigaciones científicas y dar un paso de avance en la informatización del Sistema Nacional de Salud (SNS).

Palabras claves: malformaciones congénitas.

Índice

INTRODUCCIÓN	9
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA	13
INTRODUCCIÓN	13
1.1. ESTUDIO DEL ESTADO DEL ARTE.....	13
1.2. REGISTROS Y ESTUDIOS DE MALFORMACIONES CONGÉNITAS EN EL ÁMBITO NACIONAL Y EXTRANJERO	14
1.3. DESARROLLO DEL SISTEMA INFORMÁTICO PARA LA RED NACIONAL DE GENÉTICA MÉDICA: RECUMAC VERSIÓN 2.0.....	18
1.4 FUNDAMENTACIÓN DE LAS TECNOLOGÍAS Y HERRAMIENTAS A UTILIZAR	19
1.4.1 Metodología de desarrollo del software: RUP.....	20
1.4.2 Lenguaje Unificado de Modelado: UML	21
1.4.3 Herramienta Case: Visual Paradigm 3.1	22
1.4.4 Servidor de base de datos: MySQL 5.0	23
1.4.5 Servidor Web: Apache 2.0.....	23
1.4.6 Sistema de control de versiones: Subversion 1.4	24
1.4.7 Entorno de desarrollo: Eclipse.....	25
1.4.9 Framework para el desarrollo: Symfony 1.0.....	26
1.4.10 Lenguaje de programación del lado del servidor: PHP 5.....	27
1.5 ROLES Y ARTEFACTOS	28
1.5.1 Rol Diseñador	28
1.5.2 Rol Implementador	29
1.6 PATRONES DE DISEÑO Y PATRONES DE ARQUITECTURA.....	29
1.6.1 Patrones de Arquitectura.....	30
1.6.2 Patrones Básicos de GRASP.....	31
1.6.3 Tipos de patrones GoF.....	31
CONCLUSIONES	32
CAPÍTULO 2: DISEÑO DEL SISTEMA	33
INTRODUCCIÓN	33
2.1 CARACTERÍSTICAS DEL SISTEMA	33
2.1.1 Requisitos funcionales del sistema	34
2.1.2 Requisitos no funcionales del sistema	34
2.2 PAUTAS DEL DISEÑO.....	35
2.3 ¿CÓMO SE PONE DE MANIFIESTO EL USO DEL PATRÓN MVC QUE IMPLEMENTA SYMFONY?	36
2.4 APLICACIÓN DE LOS PATRONES GRASP EN SYMFONY.....	38
2.5 APLICACIÓN DE LOS PATRONES GOF EN SYMFONY	39
2.6 MODELO DE DISEÑO	41
2.6.1 Descripción de las Clases del diseño.....	42
2.6.2 Diagramas de clases del diseño.....	49
2.6.3 Diagramas de Interacción (Secuencia)	55
2.7 MODELO DE DESPLIEGUE	60
2.8 SEGURIDAD	61
2.9 VALIDACIÓN	63

CONCLUSIONES	66
CAPÍTULO 3: IMPLEMENTACIÓN	67
INTRODUCCIÓN	67
3.1 MODELO DE IMPLEMENTACIÓN	67
3.1.1 <i>Diagramas de componentes</i>	67
3.2 ESTÁNDAR DE CODIFICACIÓN	71
3.3 CÓDIGO FUENTE DE LOS PRINCIPALES MÉTODOS Y SU DESCRIPCIÓN	73
3.4 VALIDACIÓN A NIVEL DE DESARROLLADOR.....	78
3.5 INTERFACES DE LA APLICACIÓN	81
CONCLUSIONES	84
CONCLUSIONES GENERALES	85
RECOMENDACIONES	86
REFERENCIAS BIBLIOGRÁFICAS	87
BIBLIOGRAFÍA	89
ANEXOS	92
ANEXO # 1 INSTRUMENTO ACTUAL	92
ANEXO # 2 FORMULAS USADAS PARA LOS REPORTES DE LA APLICACIÓN.....	93
ANEXO # 3 DESCRIPCIÓN DE LOS CU DEL SISTEMA.....	94
ANEXO # 4 PAQUETE DE MODELO.....	99
GLOSARIO DE TÉRMINOS	100

Índice de Figura

Figura 1 Diagrama de caso de uso del sistema.....	33
Figura 2 Planilla decorada con un Layout.....	41
Figura 3 Diagrama de clase del diseño CU insertar datos complementarios del paciente.....	50
Figura 4 Diagrama de clase del diseño CU modificar datos de los pacientes.....	51
Figura 5 Diagrama de clase del diseño CU codificar malformación.....	52
Figura 6 Diagrama de clase del diseño CU generar reportes estadísticos.....	53
Figura 7 Diagrama de clase del diseño CU buscar paciente a codificar.....	54
Figura 8 Diagrama de secuencia CU Codificar Malformación.....	56
Figura 9 Diagrama se secuencia CU Insertar Datos Complementarios del Paciente (escenario agregar malformado).....	57
Figura 10 Diagrama se secuencia CU Insertar Datos Complementarios del Paciente (escenario seleccionar paciente).....	58
Figura 11 Diagrama de secuencia CU Generar Reporte Estadístico (Reporte Factor de riesgo y Defecto Congénito).....	59
Figura 12 Diagrama de Despliegue.....	60
Figura 13 Diagrama de componente CU insertar datos complementarios del paciente.....	68
Figura 14 Diagrama de componente CU modificar Datos del paciente.....	69
Figura 15 Diagrama de componente CU Generar Reportes Estadísticos.....	70
Figura 16 Diagrama de componente CU Buscar Paciente a Codificar.....	70
Figura 17 Diagrama de componente CU Codificar Malformación.....	71
Figura 18 Mensajes de validación.....	80
Figura 19 Mensajes de validación.....	80
Figura 20 Interfaz correspondiente al CU Insertar Datos Complementarios del Paciente.....	81
Figura 21 Interfaz correspondiente al CU Modificar Datos del Paciente.....	82
Figura 22 Interfaz correspondiente al CU Generar Reporte Estadísticos (Reporte Factor de Riesgo y Defecto Congénito).....	82
Figura 23 Interfaz correspondiente al CU Generar Reporte Estadístico (Reporte Prevalencia al Nacimiento).....	83
Figura 24 Interfaz correspondiente al CU Codificar Malformación.....	83
Figura 25 Interfaz correspondiente al CU Buscar Paciente a Codificar.....	84

Introducción

Las malformaciones congénitas son la principal causa de morbilidad y mortalidad neonatal en casi todo el mundo; constituyendo un problema grave aún no resuelto. Su repercusión tanto social como en la esfera familiar es inmensa y también lo es desde el punto de vista económico. Sin embargo, se conoce como los programas de la Revolución Cubana, y en especial los programas de genética, han influenciado favorablemente en esta situación. No ocurre así en muchos países del mundo, donde las malformaciones continúan ocupando el segundo lugar dentro de la mortalidad infantil.[1]

Cuba en los últimos 49 años no ha estado exenta de esto, motivo por el cual ha desarrollado un grupo de programas sociales para mejorar el nivel de vida de las personas con defectos congénitos, entre los que se encuentran la educación y la salud. La salud pública cubana constituye un proyecto social en sí mismo que garantiza la equidad, accesibilidad y seguridad que requieren todos los ciudadanos. Este sector social ha desarrollado importantes proyectos para el bienestar de la sociedad cubana, entre los que se encuentra el Registro Cubano de Malformaciones Congénitas (RECUMAC).

El gobierno revolucionario cubano ha conferido siempre gran importancia al sector de la salud y ha apostado por el desarrollo paulatino de este. En aras de perfeccionar y lograr mayor eficiencia en la prestación de servicios a la sociedad, ha encaminado sus pasos a la informatización de estos. La informática en sus diferentes manifestaciones, tiene asegurado un papel protagónico en el futuro de la humanidad. La informatización del SNS está dada por el conjunto de métodos, técnicas, procedimientos y actividades gerenciales dirigidas al manejo de la información en salud, la cual comprende la información sobre el estado de salud de la población cubana.[2]

Para la contribución al desarrollo de la salud cubana el MINSAP ha definido la informatización como una de sus prioridades y ha convocado a un grupo de instituciones del Ministerio de Informática y las Comunicaciones (MIC) para el desarrollo de las mismas, como son: La Empresa Nacional de Software (DESOFT), el Poder de Conexión Máxima (PCMAX), la Universidad de las Ciencias Informáticas (UCI), INFOMED, el Centro de Desarrollo Informático para la Salud (CEDISAP) y las Direcciones Nacionales del Ministerio de Salud Pública.[3]

La UCI como parte del grupo de apoyo a la informatización del país desarrolla productos para el bienestar de la sociedad entre los que se encuentran, software para la educación, la economía, el

transporte, la salud entre otros. Esta institución a pesar de ser muy joven aún, juega un papel importante en el desarrollo informático de la sociedad cubana. Este centro de estudios ha desarrollado productos informáticos para disímiles instituciones de la salud en el país, entre los que se encuentran el Centro de Ingeniería Genética y Biotecnología (CIGB), Centro de Control Estatal de Equipos Médicos (CCEEM), Centro Nacional de Investigaciones Científicas (CNIC) y el Centro Nacional de Genética Medica (CNGM) entre otros.

Uno de los centros más jóvenes y prestigiosos dentro del sector de la salud pública cubana en estos momentos es el CNGM único de su tipo en el país, creado el 5 de agosto de 2003 como parte de la batalla de ideas y gran colaborador de la Organización Mundial de la Salud (OMS) en Cuba. Su objetivo fundamental es llevar a cabo acciones asistenciales, docentes y de investigación en el campo de los problemas de salud de carácter genético, encaminados a elevar la calidad de vida y el bienestar del pueblo cubano.

El CNGM lleva a cabo disímiles proyectos entre los que se encuentra el RECUMAC que es el encargado de registrar y controlar los casos de malformaciones congénitas existentes en Cuba. En dicho registro la gestión de la información referente a los malformados se lleva a cabo de forma manual, las planillas, también llamadas instrumentos, se archivan en papel y el envío de los informes al CNGM se realiza a través del correo postal. Para lograr mejorar la gestión de toda la información referente a las anomalías congénitas se realizó la versión 1.0 del RECUMAC en la UCI.

Dicha versión no fue desarrollada sobre la arquitectura de software establecida por el MINSAP para el desarrollo de sus aplicaciones informáticas. Además no cumple con todos los requisitos planteados por los especialistas del CNGM vinculados al RECUMAC, no responde a todas las necesidades para las que se concibió puesto que no ofrece todos los reportes necesarios para que los genetistas realicen sus investigaciones científicas.

Para el desarrollo de sistemas informáticos que puedan ser utilizados en todo el país a través de la Red de Salud de Cuba, el MINSAP ha definido un grupo de premisas y requisitos que incorporan los últimos adelantos en el área de las tecnologías de la información y las comunicaciones y que garantizan la plataforma de integración de las aplicaciones, la compatibilidad y sostenibilidad de los productos a desarrollar, tales como: empleo de tecnologías basadas en Internet (Extensible Markup

Language (XML), Web Services) y software libre (Personal Home Page (PHP),MySQL, Linux), documentación de todo el proceso productivo, requisitos de seguridad del software, independencia de la base de datos, desarrollo en multiplataforma y empleo de estándares internacionales para los productos relacionados con la salud.[4]

La aplicación de estas tecnologías propuestas por el MINSAP posibilitará el desarrollo y publicación de un producto informático que permita gestionar toda la información referente al RECUMAC lo que le conferirá mayor confiabilidad a los datos debido a que en ocasiones la información recopilada en los hospitales de todo el país no llega al CNGM porque se pierde en el envío hacia dicho centro, a este producto informático se podrá acceder desde cualquier localidad del país.

Como la versión existente del RECUMAC no satisface todas las necesidades identificadas por el cliente, además no se puede publicar en la Red de Salud de Cuba y el registro de los nacimientos con malformaciones congénitas no se puede llevar a cabo de forma automatizada en el país, lo que provoca retraso en las investigaciones de los genetistas, surge el siguiente **problema científico**: ¿Cómo obtener un producto funcional a partir de los requerimientos identificados para el Registro Cubano de Malformaciones Congénitas integrado a la red de salud?.

Teniendo como **objeto de estudio** el proceso de gestión de información de la salud, y como **campo de acción** derivado del mismo, el proceso de desarrollo de software para la gestión de la información de las malformaciones congénitas del Sistema Informático para la Red Nacional de Genética Médica.

Para dar solución a la problemática planteada se ha trazado como **objetivo general**: Desarrollar el diseño y la implementación del Registro Cubano de Malformaciones Congénitas (RECUMAC) 2.0 del Sistema Informático para la Red Nacional de Genética Médica.

Entre los **objetivos específicos** que se proponen se encuentran:

- Diseñar las clases del RECUMAC 2.0.
- Implementar las clases del RECUMAC 2.0.

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

1. Análisis de la versión existente del RECUMAC.
2. Análisis de la arquitectura de software establecida para el desarrollo de sistemas informáticos para el MINSAP.
3. Diseño del RECUMAC 2.0.
4. Implementación de los componentes del RECUMAC 2.0.
5. Validación de la aplicación a nivel de desarrollador.

El documento está estructurado de la siguiente manera:

Capítulo 1: Fundamentación Teórica.

En este capítulo se realiza la fundamentación teórica donde se incluye el estado del arte de la investigación, a nivel nacional e internacional principalmente después del surgimiento de la versión 1.0 del RECUMAC. Se aborda y profundiza en el estudio de las herramientas y metodologías definidas por el MINSAP para el desarrollo de sistemas informáticos que serán publicados en INFOMED. Además se estudian y exponen los roles que se desarrollarán producto de las necesidades actuales del proyecto, así como el estudio y selección del patrón de arquitectura y los patrones de diseño que se usarán en el desarrollo del sistema.

Capítulo 2: Diseño del sistema.

En este capítulo se diseña el sistema, permitiendo mostrar los detalles descriptivos de la solución propuesta. Se explican los elementos asociados a la seguridad y validación incorporados a Symfony. Se muestra a través del diagrama de despliegue la distribución de los componentes físicos necesarios para la implantación del sistema, así como otros elementos significativos del diseño.

Capítulo 3: Implementación del sistema.

En este capítulo se realiza la implementación del sistema, donde los elementos del diseño se convierten en elementos de implementación en términos de componentes. Se especifican los estándares de codificación que se aplican en el desarrollo de la implementación del sistema informático, así como la descripción y el código fuente de los principales métodos.

Capítulo 1: Fundamentación Teórica

Introducción

En este capítulo se hace un análisis de los diferentes tipos de registros de malformaciones congénitas existentes tanto en el ámbito nacional como extranjero. Se hace un análisis de los antecedentes al RECUMAC versión 2.0 donde se expone el estado actual de la versión 1.0 del RECUMAC. Se muestran tanto los patrones de diseño, como los de arquitectura y los roles que se ponen en práctica para el desarrollo del presente trabajo de diploma. También se describe la arquitectura propuesta para el SIGM, la cual define las tecnologías, metodología de trabajo y herramientas informáticas a utilizar.

1.1. Estudio del Estado del Arte

Una de las preocupaciones principales de los padres ante la llegada de un bebé es que nazca sano. Para ello, es importante tomar algunas precauciones antes y durante los meses de gestación. Los defectos congénitos afectan a la formación del embrión o feto. Existen algunos como los morfológicos, que afectan la forma física del bebé, los funcionales, que inciden en el comportamiento de distintos órganos del cuerpo y los estructurales, que se deben a alteraciones de los genes o de las células.

Durante la década del 70 se efectuaron varias investigaciones para analizar los diferentes tipos de malformaciones y su prevalencia en cada región. Posteriormente, estos estudios se constituyeron en entidades de vigilancia epidemiológicas y como organizaciones activas en la búsqueda de factores de riesgo y causalidad de dichas malformaciones y con ello se crean los registros de malformaciones congénitas. [5]

Las **malformaciones congénitas** son defectos estructurales de la morfogénesis presente al nacimiento. Puede comprometer a un órgano o sistema o varios al mismo tiempo. Las causas que pueden provocar las anomalías congénitas son genéticas, ambientales y multifactoriales.[6]

1.2. Registros y estudios de malformaciones congénitas en el ámbito nacional y extranjero

Los registros de malformaciones congénitas son creados para el control, prevención y vigilancia epidemiológica de trastornos genéticos. Muchos países cuentan ya con sus propios registros lo que les permite hacer un estudio más profundo sobre las anomalías congénitas y poder disminuir la morbimortalidad infantil¹, ya que este fenómeno genético en el mundo constituye una de las principales causas de mortalidad infantil.

En el mundo existen varios programas de vigilancia epidemiológica de las malformaciones, en Europa funciona **European Registration of Congenital Anomalies and Twins (EUROCAT)** y **Estudio Colaborativo Español de Malformaciones Congénitas (ECEMC)** quienes reportan una prevalencia de 4,7 y 1,6 por 1,000 respectivamente.

EUROCAT iniciada en 1979 basada en los registros de las vigilancias epidemiológicas de las anomalías congénitas, tiene como objetivos proporcionar información epidemiológica sobre las anomalías congénitas en Europa, facilitar la alerta temprana de los nuevos riesgos teratogénicos², evaluar la eficiencia de la prevención primaria y el impacto de la evolución de diagnóstico prenatal, proporcionar una red de colaboración y una infraestructura para las investigaciones relacionadas con las causas, la prevención de anomalías congénitas, el tratamiento y cuidado de los niños afectados.[7]

El Estudio Colaborativo Español de Malformaciones Congénitas es un programa de investigación clínico-epidemiológico y multidisciplinario, de ámbito nacional que controla las malformaciones congénitas, es multicéntrico permitiendo que médicos de todo el país colaboren agrupando sus casos para poder ejecutar la investigación de las causas y una excelente atención a sus pacientes. Cuenta con una experiencia de 30 años. Ha examinado más de 31.000 recién nacidos que presentaban anomalías congénitas identificables al nacimiento. Esto implica que tiene una gran experiencia en el diagnóstico de síndromes malformativos raros hasta incluso ha descrito algunos síndromes nuevos, en la transmisión de la información sobre esos casos y sobre factores de riesgo, así como en el establecimiento de posibles medidas de prevención primaria.

¹ Se define como todos los niños que fallecen antes de cumplir el año de edad.

² Genera malformaciones en la formación del feto.

En Latinoamérica existen dos grandes registros: el **Registro y Vigilancia Epidemiológica de Malformaciones Congénitas Externas (RYVEMCE)** en México y el **Estudio Colaborativo Latinoamericano de Malformaciones Congénitas (ECLAMC)** que cubre la mayor parte de Sudamérica.

Aún se desconocen las causas por las que se originan más de la mitad de las malformaciones congénitas. El objetivo principal del ECLAMC es la prevención primaria de los defectos congénitos mediante la vigilancia epidemiológica y la investigación científica. También lleva a cabo 3 actividades principales como:

1. Vigilancia epidemiológica trimestral de la frecuencia de defectos congénitos y evaluación de la ocurrencia inusual de un defecto congénito específico en un lugar y tiempo determinado, a través de una metodología sistematizada denominada Rumor.
2. Estudio de factores genéticos y ambientales involucrados en la causalidad de defectos congénitos, mediante el desarrollo de trabajos de investigación en el marco de la epidemiología genética.
3. Difusión de normas de prevención primaria de defectos congénitos, en libros y publicaciones de divulgación general.

Desde su creación en 1967, el ECLAMC ha estudiado, sistemáticamente, más de tres millones de nacimientos, ocurridos en 150 series consecutivas, correspondientes a 150 hospitales participantes, distribuidos en 40 ciudades de 12 países latinoamericanos.

Primero en su tipo en América latina y uno de los pioneros en el mundo, el programa ECLAMC fue modelo en la creación de programas semejantes en México, España e Italia, así como también de sistemas nacionales de registro y vigilancia epidemiológica de malformaciones en Cuba y Costa Rica. Estos continúan funcionando bajo el asesoramiento del ECLAMC. De este modo, el programa ECLAMC ha creado una metodología y tecnología adecuadas para la investigación, registro y vigilancia de los defectos congénitos en países subdesarrollados. En 1989, el ECLAMC fue designado por la Organización Mundial de la Salud, centro colaborador para la prevención de las malformaciones congénitas. [8]

El RYVEMCE ha encaminado sus pasos a poner un nuevo grano de arena en la batalla para tratar de prevenir los defectos congénitos de los recién nacidos, ha reportado una tasa de prevalencia al nacimiento de 1 por 1,000. Al igual que los demás registros tiene como objetivo la prevención primaria de los defectos congénitos y la investigación científico, así como estudiar los diferentes aspectos de las malformaciones congénitas en los recién nacidos vivos y muertos.

Algunos países latinoamericanos también poseen registros de anomalías congénitas como el caso de Nicaragua con el **Registro Nicaragüense de Malformaciones Congénitas (RENIMAC)** que se diseñó desde 1992. En Chile existe el **Registro Regional de Malformaciones Congénitas (RRMC)**, este registro surge en la región de Maule debido a que esta región es predominantemente agro-industrial, con uso abundante de productos tóxicos como plaguicidas y con indicadores de alta prevalencia de alcoholismo, principal teratógeno conocido y la percepción por parte del equipo de salud local que nacen numerosos malformados. Usando la metodología del ECLAMC, se estableció el **Programa de Vigilancia Epidemiológica de Malformaciones Congénitas en Colombia (VIDEMCO)** que durante 4 años en once hospitales se recolectaron datos para estudios poblacionales.

Con los importantes avances en el establecimiento de las bases científicas de la patología de las malformaciones congénitas, se crean organizaciones internacionales que se encargan del estudio y registro de los defectos al nacimiento, estas organizaciones son: el **Registro Internacional para el Monitoreo de Malformaciones Congénitas Externas (Clearinghouse)**, formada por más de 20 países y el **International Clearinghouse of Birth Defects Monitoring System (ICBDMS)** que representa a 39 programas de monitoreo de las malformaciones congénitas de todo el mundo.

Con la creación de los registros de malformaciones congénitas se logra conservar gran cantidad de información referente a las anomalías congénitas y en la creación de políticas de seguridad en la salud pública en el mundo. Los registros anteriormente expuestos de una forma u otra solo se centran en el estudio de malformaciones específicas, se acotan a poblaciones concretas con características determinadas y no se tiene información de que estén informatizados, se infiere que lo estén. Se conoce solo que en el ECLAMC la modalidad caso control se llenan de forma manual y los demás se utilizan Microsoft Excel versión 2000 para el almacenamiento de los formularios.

En el ECLAMC actualmente participan 4 maternidades dando una cobertura de registro en recién nacidos malformados de 22% en el territorio nacional y de 44% en los nacidos en el Departamento de Montevideo, lo que provoca en ocasiones un déficit de información y a una fluctuación de la frecuencia de algunas malformaciones que no representan la realidad local, sino mas bien que son artificiales, producto de dicha situación.

El SsdwLab5 nuevo software para detectar que el futuro bebé padezca Síndrome de Down dentro de los tres primeros meses se embarazo. Desarrollado por un grupo de ginecólogos de Gerona, es un software que no es todavía 100% fiable, la nueva herramienta que a diferencia de las convencionales, se trata de un sistema que no presenta riesgo alguno y más al no ser invasivo. Detecta además del Síndrome de Down, el síndrome de Edwards y el Síndrome de Patau. El funcionamiento del software se basa en un modelo matemático estadístico que toma como referencia datos de diversos ámbitos. El SsdwLab5, ya ha comenzado a introducirse en varios hospitales, como el Clínico San Carlos de Madrid, el Clínico de Barcelona y el Carlos Haya de Málaga, entre otros. Además, la multinacional suiza Roche ha adquirido los derechos para comercializarlo en el exterior.[9]

Actualmente en Cuba los defectos congénitos constituyen la segunda causa de muerte en niños menores de un año de edad, con una prevalencia al nacimiento de 1,7 por 10 000 nacimientos, por lo cual el MINSAP le ha conferido un lugar prioritario en los programas médico-sociales del país; es por ello que Cuba cuenta con un sistema de vigilancia de malformaciones congénitas muy efectivo, para ello fue necesario introducir un registro permanente de malformaciones congénitas, denominado **Registro Cubano de Malformaciones Congénitas (RECUMAC)**, en Cuba también se encuentra el **Registro Cubano Prenatal de Malformaciones Congénitas (RECUPREMAC)**, en el cual se registra toda la información de malformaciones congénitas detectadas por medio del ultrasonido, alfafeto proteína o citogenética³, antes del nacimiento.

Desde al año 1985 se dispone de un Registro de Malformaciones Congénitas (RECUMAC) que posee la información necesaria para establecer la frecuencia de base de los defectos congénitos, efectuando la caracterización patogénica de recién nacidos (RN) con malformaciones múltiples y su monitoreo como el medio más sensible de detección de agentes teratogénicos, en un corto período.

³ Rama de la genética que estudia los cromosomas, su estructura y su herencia.

El RECUMAC es el encargado de ofrecer una atención clínica y vigilancia epidemiológica de las anomalías congénitas en Cuba. El estudio de las malformaciones congénitas permite la introducción en todo el país del diagnóstico prenatal masivo de un grupo de malformaciones congénitas, propiciando una atención materno-infantil de mayor calidad lo que repercute positivamente en la morbimortalidad perinatal⁴. Dicho registro ha analizado 520 578 recién nacidos en 36 hospitales ginecobstétricos de 13 provincias de Cuba en un período de 12 años, en el cual la prevalencia de malformaciones congénitas al nacimiento es de 179,8 por 10 000 nacimientos. [10]

1.3. Desarrollo del Sistema Informático para la Red Nacional de Genética Médica: RECUMAC versión 2.0

Con el objetivo de contribuir a la gestión adecuada de la información en los registros cubanos de trastornos genéticos que radican en el CNGM, se crearon una serie de proyectos informáticos vinculados a la UCI que respondían a las necesidades de los genetistas vinculados a cada registro. Estos proyectos no fueron desarrollados sobre la arquitectura de software establecida por el MINSAP para el desarrollo de sus aplicaciones informáticas lo que provocó que no pudieran ser publicados en INFOMED y por lo tanto no se pudiera acceder a cada proyecto desde cualquier localidad del país. Además estos proyectos fueron desarrollados sin tener en cuenta la información común que estos manejan para los casos en que el paciente posea más de un trastorno genético, lo que provocaba la descentralización de la información.

Por todo lo planteado anteriormente surge el Sistema Informático para la Red Nacional de Genética Médica (SIGM) con el objetivo de contribuir a la informatización e integración de una serie de registros que se controlan a nivel nacional en el CNGM además de ayudar a mejorar la comunicación entre genetistas de todo el país. Dentro de este sistema serán establecidos una serie de proyectos entre los que se encuentran: Registro Cubano de Gemelo, Registro Cubano de Historia Clínicas, Registro Cubano de Enfermedades Genéticas, Registro Cubano de Retraso Mental, Registro Cubano de Discapacitados, Teleconsulta y Registro Cubano de Malformaciones Congénitas. Este sistema será desarrollado utilizando la arquitectura de software establecida por el MINSAP para el desarrollo de sus productos informáticos y todos los registros también tendrán que dar cumplimiento a la utilización de esta arquitectura debido a que el sistema será publicado en INFOMED. Este sistema va a contribuir en

⁴ Lo que está alrededor del nacimiento, como el embarazo, el parto y la lactancia.

gran medida a mejorar toda la gestión de la información existente en el CNGM además de ser un paso de avance en la informatización del Sistema Nacional de Salud (SNS).

La realización del Registro Cubano de Malformaciones Congénitas versión 2.0 es el tema que se trata en el presente trabajo de diploma, como antecedente tendrá la versión 1.0 del RECUMAC, este sistema ya existente, además de no ser desarrollado sobre la arquitectura de software propuesta por el MINSAP para sus productos informáticos, no dio cumplimiento a todos sus requisitos funcionales, de 6 requisitos identificados solo se implementaron 4 de ellos, los 2 requerimientos que quedaron sin cubrir le imprimen un valor significativo al proyecto pues se relacionan con la obtención de reportes de “Prevalencia al nacimiento” y “Factor de riesgo y defecto congénitos”, reportes que brindarían información estadística de vital importancia para el estudio de los genetistas vinculados al RECUMAC. Por el estado actual de la versión 1.0 del RECUMAC surge dentro del SIGM el proyecto Registro Cubano de Malformaciones Congénitas versión 2.0.

1.4 Fundamentación de las tecnologías y herramientas a utilizar

INFOMED es la red del sistema de salud cubana, ella permite la accesibilidad a toda la información relacionada con las ciencias de la salud y especialmente dar acceso a la información producida en Cuba es este campo. Para la incorporación de un sistema informático a INFOMED debe ser desarrollado sobre la arquitectura de software establecida por el MINSAP.

El RECUMAC en su versión 2.0 que forma parte del Sistema Informático de la Red Nacional de Genética Médica y este es un sistema integral se definió por la tesis de arquitectura utilizar las siguientes metodologías y herramientas para su desarrollo

1.4.1 Metodología de desarrollo del software: RUP

El Proceso Unificado de Desarrollo (RUP) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP utiliza UML para definir los modelos de software y puede definirse como un modelo que es dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental. Se divide en 4 fases: Inicio, Elaboración, Construcción y transición.

- Inicio: El objetivo en esta etapa es determinar la visión del proyecto.
- Elaboración: En esta etapa el objetivo es determinar la arquitectura óptima.
- Construcción: En esta etapa el objetivo es llegar a obtener la capacidad operacional inicial.
- Transición: El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

En RUP se han agrupado las actividades en grupos lógicos definiéndose 9 flujos de trabajo principales. Los 6 primeros son considerados como flujos de ingeniería y los tres últimos de apoyo.

Un Proceso de Desarrollo de Software es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto, tiene la misión de transformar los requerimientos del usuario en un producto de software. Un proceso define “quién” está haciendo “qué”, “cuándo” y “cómo” para alcanzar un determinado objetivo.

Características del Proceso Unificado de Desarrollo

Los verdaderos aspectos definitorios del Proceso Unificado, y que lo convierten en único, se resumen en tres fases claves – dirigido por casos de uso, centrado en la arquitectura, e iterativo e incremental.

Dirigido por los casos de uso:

Teniendo en cuenta que la razón de ser de un sistema es brindar servicios a los usuarios, RUP define como caso de uso el conjunto de acciones que debe realizar un sistema para dar un resultado de valor a un determinado usuario y los utiliza tanto para especificar los requisitos funcionales del sistema, como para guiar todos los demás pasos de su desarrollo, dígame diseño, implementación y prueba.

Centrado en la arquitectura:

La arquitectura es una vista del diseño completo con las características más importantes, dejando a un lado los detalles. Esta no solo incluye las necesidades de los usuarios e inversionistas, sino también otros aspectos técnicos como el hardware, sistema operativo, sistema de gestión de base de datos y protocolos de red con los que debe coexistir el sistema. En otras palabras, la arquitectura representa la forma del sistema, la cual va madurando en su interacción con los casos de uso hasta llegar a un equilibrio entre funcionalidad y características técnicas.

Iterativo e incremental:

La alta complejidad de los sistemas actuales hace que sea factible dividir el proceso de desarrollo en varios mini-proyectos. Cada uno de estos mini-proyecto se les denomina iteración y pueden o no representar un incremento en el grado de terminación del producto completo. En cada iteración los desarrolladores seleccionan un grupo de casos de uso, los cuales se diseñan, implementan y prueban. La planificación de iteraciones hace que se reduzcan los riesgos de los costos de un solo incremento, no sacar al mercado un producto en el tiempo previsto.

El RUP divide el proceso de desarrollo en ciclos, obteniéndose un producto funcional al concluir cada uno. A su vez, cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante.[11]

1.4.2 Lenguaje Unificado de Modelado: UML

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software.

UML es un lenguaje expresivo, claro y uniforme, 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.

UML tiene como características:

- Tecnología orientada a objetos.
- Viabilidad en la corrección de errores.

- Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, etc.).
- Cubre las cuestiones relacionadas con el tamaño propio de los sistemas complejos y críticos.
- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas.
- Existe un equilibrio entre expresividad y simplicidad, pues no es difícil de aprender ni de utilizar.

UML es independiente del proceso, aunque para utilizarlo óptimamente se debe usar en un proceso que sea dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental. Es importante resaltar que UML es un lenguaje para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir.

UML es desde finales de 1997, un lenguaje de modelado orientado a objetos estándar, de acuerdo con el Object Management Group, siendo utilizado diariamente por grandes organizaciones como: Microsoft, Oracle y Rational. [12]

1.4.3 Herramienta Case: Visual Paradigm 3.1

Visual Paradigm es una poderosa herramienta para visualizar y diseñar elementos de software, para ello utiliza el lenguaje UML. Ofrece una gama de facilidades para el modelado de aplicaciones, genera la documentación del proyecto automáticamente en varios formatos como Web o Pdf. Además presenta otras características como:

- Genera código en PHP, Java, C++, y otros. Permite incorporar los dibujos del Visio en cualquier diagrama de UML.
- No solo realiza diagramas de UML normales, sino también posibilita modelar hardware, dominio-específico, el software, conectando una red de computadoras los componentes usando sus iconos, más allá de las anotaciones de UML normales.

- Es multiplataforma, está disponible en las plataformas Windows y Linux.
- Está orientada a la creación de diseños usando el paradigma de programación orientado a objetos. [13]

1.4.4 Servidor de base de datos: MySQL 5.0

MySQL es un sistema de gestión de base de datos relacional, multihilo mediante hilos del kernel y multiusuario. Opera en una arquitectura cliente/servidor, de tal manera que el servidor sólo tiene que enviarle una cadena de caracteres y esperar la devolución de los datos. Está muy ligado a PHP y tiene Múltiples motores de almacenamiento (MyISAM, Merge, InnoDB, BDB, Memory/heap, MySQL Cluster, Federated, Archive, CSV, Blackhole y Example en 5.x).

MySQL 5.0 añade nuevas características que lo hacen más atractivo, algunas de estas son:

- En MySQL 5.0, los clientes y servidores Windows se pueden conectar usando memoria compartida.
- Un amplio subconjunto de ANSI SQL 99, y varias extensiones.
- Soporte a multiplataforma.
- Triggers.
- Vistas actualizables.
- Soporte a VARCHAR.
- Soporte X/Open XA de transacciones distribuidas; transacción en dos fases como parte de esto, utilizando el motor InnoDB de Oracle.
- Motores de almacenamiento independientes (MyIsam para lecturas rápidas, InnoDB para transacciones e integridad referencial).
- Transacciones con los motores de almacenamiento InnoDB, BDB Y Cluster; puntos de recuperación (savepoints) con InnoDB. [14]

1.4.5 Servidor Web: Apache 2.0

Un servidor web se encarga de mantenerse a la espera de peticiones HTTP llevadas a cabo por un cliente HTTP que se conoce como navegador. Apache es un software libre servidor Web de código abierto, presenta entre otras características mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido. Apache está diseñado para ser un servidor Web potente y

flexible que pueda funcionar en la más amplia variedad de plataformas y entornos. En el desarrollo del presente trabajo de diploma se utiliza la versión 2.0 de Apache, algunas características de esta versión son:

- Nuevo sistema de configuración y compilación.
- Soporte Multiprotocolo.
- Soporte mejorado para las plataformas que no son tipo Unix.
- Mensajes de error en diferentes idiomas.
- Configuración simplificada. [15]

1.4.6 Sistema de control de versiones: Subversion 1.4

Un sistema de control de versiones es un sistema de gestión de archivos y directorios, cuya principal característica es que mantiene la historia de los cambios y modificaciones que se han realizado sobre ellos a lo largo del tiempo. De esta forma, el sistema es capaz de “recordar” las versiones antiguas de los datos, lo que nos permite examinar el histórico de cambios o recuperar versiones anteriores de un fichero, incluso aunque haya sido borrado.

El sistema de control de versiones definido por la arquitectura del Sistema Informático de Genética Médica es subversion y algunas de sus características son:

- Mantiene versiones no sólo de archivos, sino también de directorios
- También se mantienen versiones de los metadatos asociados a los directorios.
- Además de los cambios en el contenido de los documentos, se mantiene la historia de todas las operaciones de cada elemento, incluyendo la copia, cambio de directorio o de nombre.
- Atomicidad de las actualizaciones. Una lista de cambios constituye una única transacción o actualización del repositorio. Esta característica minimiza el riesgo de que aparezcan inconsistencias entre distintas partes del repositorio.
- Posibilidad de elegir el protocolo de red. Además de un protocolo propio (svn), puede trabajar sobre http (o https) mediante las extensiones WebDAV. WebDAV (más conocido como DAV) es un protocolo que amplía las posibilidades del HTTP/1.1 añadiendo nuevos métodos y cabeceras. La capacidad de funcionar con un protocolo tan universal como el http simplifica la

implantación (cualquier infraestructura de red actual soporta dicho protocolo) y universaliza las posibilidades de acceso (si se quiere, puede utilizarse a través de Internet).

- Soporte tanto de ficheros de texto como de binarios.
- Mejor uso del ancho de banda, ya que en las transacciones se transmiten sólo las diferencias y no los archivos completos.
- Mayor eficiencia en la creación de ramas y etiquetas que en CVS. [16]

1.4.7 Entorno de desarrollo: Eclipse

Eclipse es una plataforma universal para integrar herramientas de desarrollo, con una arquitectura abierta y basada en plug-ins. Además, Eclipse da soporte a todo tipo de proyectos que abarcan desde el ciclo de vida del desarrollo de aplicaciones, incluyendo soporte para modelado. Es soportado por los principales sistemas operativos como Linux, Windows, Solaris 8, Mac OSX-Mac.

El Eclipse es un IDE para todo tipo de aplicaciones, inicialmente desarrollado por IBM, y actualmente gestionado por la Fundación Eclipse. La característica clave de Eclipse es la extensibilidad. Eclipse es una gran estructura formada por un núcleo y muchos plugins que van conformando la funcionalidad final. La forma en que los plugins interactúan es mediante interfaces o puntos de extensión así, los nuevos aportes se integran sin dificultad ni conflictos. Eclipse permite el desarrollo de aplicaciones utilizando Subversion como sistema de control de versiones y presenta un potente editor de código.

Otras de las características del IDE eclipse son multiplataforma, soporte para distintas arquitecturas, resaltado de sintaxis, auto completado, tabulador de un bloque de código seleccionado, asistentes (wizards) para la creación, exportación e importación de proyectos y para generar plantillas de códigos (templates). [17]

Para el diseño de las páginas Web se utilizó como interfaz de desarrollo WEB Quanta Plus que entre sus principales características se puede mencionar que:

Quanta es un editor web desarrollado para el escritorio KDE. Soporta una multitud de lenguajes como HTML, Javascript, CSS, PHP, SQL, XML, ColdFusion, Perl, DTML, Zope o C++. Entre sus principales características se encuentran:

- Coloreado de sintaxis para todos los lenguajes soportados.

- Cliente FTO integrado para colgar las páginas de una forma fácil y rápida.
- Cajas de diálogos contextuales, para elegir las etiquetas HTML que se quieren utilizar.
- Excelente navegador de directorios integrado, para acceder a los ficheros fácilmente.
- Completo panel de previsualización en el resultado final.
- Validador HTML integrado en la propia aplicación.
- Soporte de extensiones/plugins para añadir funcionalidades extra.[18]

1.4.9 Framework para el desarrollo: Symfony 1.0

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, un framework facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

Características de Symfony:

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

2. Symfony puede ser completamente personalizado para cumplir con los requisitos de las empresas que disponen de sus propias políticas y reglas para la gestión de proyectos y la programación de aplicaciones.
3. Por defecto incorpora varios entornos de desarrollo diferentes e incluye varias herramientas que permiten automatizar las tareas más comunes de la ingeniería del software.
4. Symfony proporcionaba calidad de su código fuente y gran cantidad de documentación disponible, dos ventajas muy importantes sobre otros frameworks disponibles.

Además la capa de internacionalización que incluye Symfony permite la traducción de los datos y de la interfaz, así como la adaptación local de los contenidos y la capa de presentación utiliza plantillas y layouts que pueden ser creados por diseñadores HTML sin ningún tipo de conocimiento del framework. Los helpers incluidos permiten minimizar el código utilizado en la presentación, ya que encapsulan grandes bloques de código en llamadas simples a funciones. Sus formularios incluyen validación automatizada y relleno automático de datos (repopulation), lo que asegura la obtención de datos correctos y mejora la experiencia de usuario. La autenticación y la gestión de credenciales simplifican la creación de secciones restringidas y la gestión de la seguridad de usuario. [19]

1.4.10 Lenguaje de programación del lado del servidor: PHP 5

PHP (acrónimo de PHP: Hypertext Preprocessor), es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de las páginas dinámicas, utiliza el motor Zend Engine (o Zend Engine 2). Lo mejor de usar PHP es que es simple para el principiante, pero a su vez, ofrece muchas características avanzadas para los programadores profesionales. Es un lenguaje libre, por lo que se presenta como una alternativa de fácil acceso para todos.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite; lo cual permite la creación de aplicaciones web muy robustas.

La versión 5.0 de PHP además de tener las características expuestas anteriormente también:

- Incorpora el Zend Engine 2.
- Reescritura del soporte de XML.
- Incorporación de XimpleXML para el fácil acceso y manejo de documentos XML.
- Nueva extensión SOAP para una fácil creación de servicios web.
- Nueva extensión MySQL para explotar las características del MySQL 4.1.
- Soporte para el motor de bases de datos SQLite y muchas más.
- Nueva extensión MySQL denominada MySQLi. [20]

1.5 Roles y artefactos

Un rol es una definición abstracta de un conjunto de actividades realizadas y de artefactos obtenidos. Los roles son realizados típicamente por un individuo, o un conjunto de individuos, trabajando juntos en equipo.

Se hará uso la metodología de desarrollo RUP, la cual define grupos de roles, agrupados por participación en actividades relacionadas. Estos grupos son: Analistas, Desarrolladores, Gestores, Apoyo, Especialistas en Prueba y otros roles.[11]

En el desarrollo del presente trabajo de diploma de acuerdo a las necesidades actuales del proyecto se desarrollarán los roles de, diseñador e implementador, pertenecientes al grupo de Desarrolladores definido por el Proceso Unificado de Desarrollo (RUP).

1.5.1 Rol Diseñador

En la metodología RUP el diseñador es el responsable de diseñar una parte del sistema cumpliendo con las restricciones de los requerimientos, arquitectura y proceso de desarrollo del proyecto, identifica y define las responsabilidades, operaciones, atributos y relaciones de los elementos de diseño. Debe asegurarse que el diseño es consistente con la arquitectura del software y que está detallado al punto que se puede proceder con la implementación.

Los artefactos realizados por el diseñador que se obtendrán en el presente trabajo son:

- **Realización de casos de uso del diseño:** Es una colaboración en el modelo de diseño que describe como se realiza un caso de uso específico, y como se ejecuta en términos de casos

de uso del diseño. Una realización de caso de uso del diseño proporciona una traza directa a una realización de caso de uso del análisis en el modelo de análisis.

- **Clases del diseño:** Una clase es una descripción de un conjunto de objetos que comparten las mismas responsabilidades, las relaciones, las operaciones, atributos, y la semántica.
- **Paquetes de diseño:** Es una colección de clases, relaciones, realizaciones de casos de uso, diagramas y otros paquetes que estén de alguna forma relacionados. Es usado para estructurar el modelo de diseño dividiéndolo en partes más pequeñas.
- **Subsistema de diseño:** Es una parte del sistema que encapsula el comportamiento, incluye interfaces y paquetes. [21]

1.5.2 Rol Implementador

En la metodología RUP el rol implementador es responsable de desarrollar y de probar componentes de acuerdo con los estándares adoptados del proyecto para la integración en subsistemas más grandes. Cuando los componentes de prueba, tales como drivers o partes se deben crear para apoyar la prueba, el implementador es también responsable de desarrollar y de probar los componentes de prueba y los subsistemas correspondientes.

El artefacto que realiza el rol implementador que se obtendrá en el presente trabajo de diploma es:

- **Elementos de implementación:** Los elementos de implementación son la parte física de la implementación, incluyen los archivos y directorios. Incluyen ficheros de código (fuentes, binarios o ejecutables), de datos y de documentación como son ficheros de ayuda online.[21]

1.6 Patrones de Diseño y Patrones de Arquitectura

Un patrón es un modelo posible a seguir para realizar algo. Los patrones surgen de la experiencia de seres humanos al tratar de lograr ciertos objetivos, estos capturan la experiencia existente y probada para promover buenas prácticas.

Según Christopher Alexander⁵ “Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, para describir después el núcleo de la solución a ese problema, de tal manera que

⁵ Christopher Alexander: Destacado Arquitecto reconocido internacionalmente por sus numerosos aportes en la Teoría de Patrones.

esa solución pueda ser usada más de un millón de veces sin hacerlo siquiera dos veces de la misma forma”

Dentro de la rama de la informática los patrones se clasifican según su escala en:

- Patrones de Arquitectura.
- Patrones de Diseño.

1.6.1 Patrones de Arquitectura

Los patrones de arquitectura 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. Los patrones de arquitectura ayudan a especificar la estructura fundamental de una aplicación. Cada actividad de desarrollo es gobernada por esta estructura; por ejemplo, el diseño detallado de los subsistemas, la comunicación y colaboración entre diferentes partes del sistema. 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 encuentran el patrón Modelo Vista Controlador (MVC) y el patrón modelo de tres capas.

Para el desarrollo de este trabajo de diploma como fue anteriormente especificado se hará uso del framework Symfony, el cual está basado en el patrón MVC, 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.

El patrón de 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 a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email, y otros.). El modelo se encarga de la abstracción de la lógica relacionada con los

datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación. [22]

1.6.2 Patrones Básicos de GRASP

Los patrones GRASP son parejas de problema solución con un nombre, que codifican buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades.

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

Experto: Este se encarga de asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

Creador: Este patrón se encarga de asignarle a la clase B la responsabilidad de crear una instancia de clase A. B es un creador de los objetos A.

Alta Cohesión: Este patrón se encarga de asignar una responsabilidad de modo que la cohesión siga siendo alta.

Bajo Acoplamiento: Este patrón se encarga de asignar una responsabilidad para mantener bajo acoplamiento. Las clases deben comunicarse con un número pequeño de clases tanto como sea posible.

Controlador: Asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. [23]

1.6.3 Tipos de patrones GoF

Los patrones GoF se clasifican en 3 categorías basadas en su propósito: creacionales, estructurales y de comportamiento. Durante el desarrollo de este trabajo de diploma se hará uso del framework Symfony, este framework utiliza una serie de patrones GoF como son:

Patrón de creación:

Singleton (Instancia única): Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.

Patrones estructurales:

Decorator (Envoltorio): Responde a la necesidad de añadir dinámicamente funcionalidad a un objeto.

Command: Encapsula peticiones en forma de objetos permitiendo así parametrizar los clientes utilizando distintas peticiones, encolar las peticiones y ofrecer la posibilidad de deshacer las operaciones. Permite solicitar operaciones sin tener que saber como o quien lleva a cabo esas operaciones.[24]

Otros Patrones

Registry: Es un medio simple y eficiente de compartir datos y objetos en la aplicación sin tener que preocuparse de mantener numerosos parámetros o hacer uso de variables globales.

Conclusiones

Después de lo analizado y teniendo en cuenta la existencia de la versión 1.0 del RECUMAC, se arribó a la conclusión de que dicha versión no cumple con todas las necesidades que exige el CNGM para sus proyectos informáticos y que los intentos de informatización del RECUMAC en Cuba, han sido primarios y sin arribar a resultados significativos, por lo que se reafirma la necesidad de desarrollar una solución informática para una mejor gestión de la información de los recién nacidos con anomalías congénitas y que pueda ser usada por todos los genetistas del país. Para lograr un satisfactorio desarrollo del proyecto se establecieron las metodologías, técnicas y herramientas a utilizar.

Capítulo 2: Diseño del sistema

Introducción

En el presente capítulo se modelan los artefactos correspondientes al flujo de trabajo análisis y diseño profundizando en el flujo de diseño. Se presenta el modelo de diseño, las pautas del diseño, así como la vista lógica de la arquitectura, son expuestos los diagramas de clases del diseño web, como también las descripciones de las clases y el modelo de despliegue.

2.1 Características del sistema

Para la realización del diseño se deben conocer las características principales del sistema especificando las diferentes acciones que el sistema debe permitir y los actores que intervienen en las mismas. Las relaciones que se establecen entre estos dos elementos se reflejan en el siguiente diagrama de casos de uso del sistema.

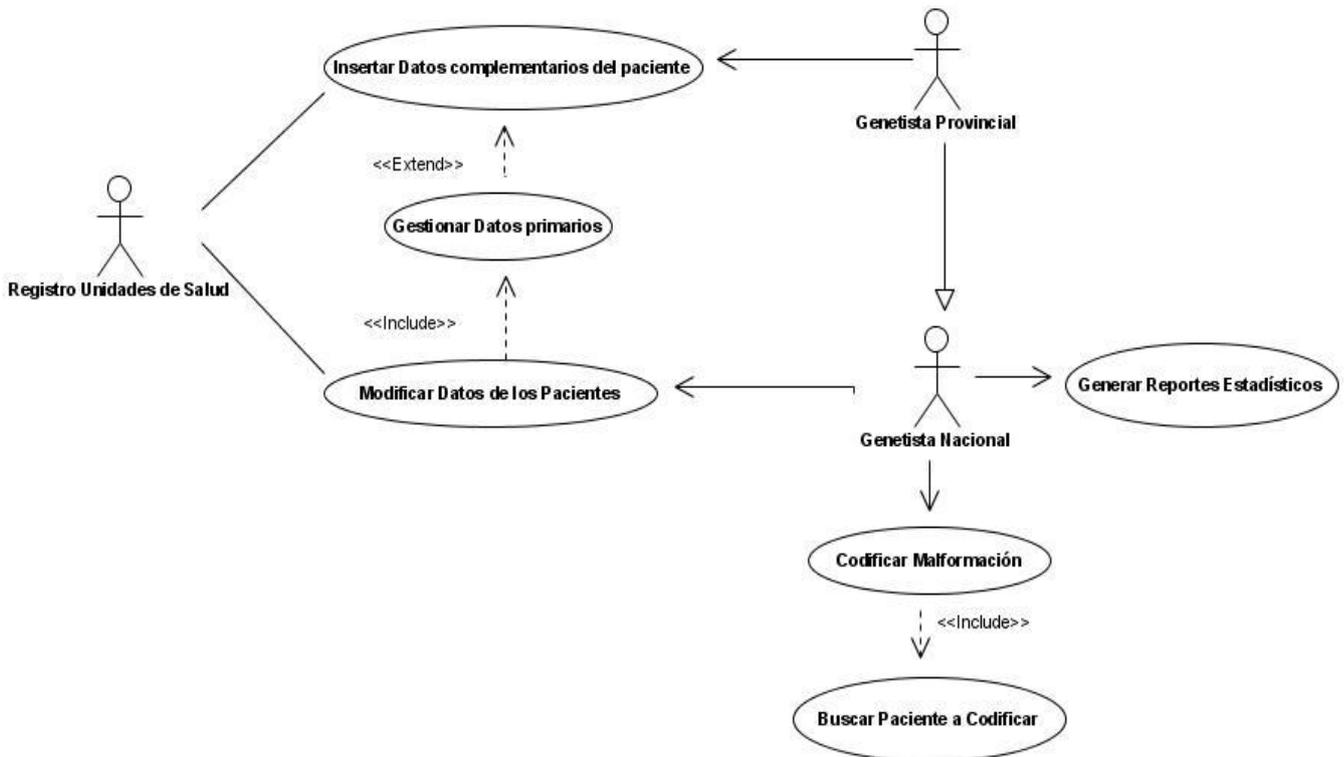


Figura 1 Diagrama de caso de uso del sistema

2.1.1 Requisitos funcionales del sistema

- R 1. Modificar datos del paciente.
- R 2. Insertar datos complementarios del paciente.
- R 3. Generar reportes estadísticos.
- R 4. Codificar malformación.
- R 5. Buscar paciente a codificar.

2.1.2 Requisitos no funcionales del sistema

- **Apariencia o interfaz externa:** Se deben utilizar imágenes y colores identificados con el negocio del sistema. La interfaz externa debe estar diseñada para verse en cualquier resolución igual o superior a 1024x768.
- **Usabilidad:** La aplicación informática debe garantizar un acceso fácil y rápido, contando con un menú que satisfaga las necesidades de los usuarios. Este podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de una computadora y del ambiente web.
- **Rendimiento:** Los tiempos de respuestas deben ser generalmente rápidos al igual que la velocidad de procesamiento de la información.
- **Soporte:** Se debe asegurar el soporte para los usuarios de manera que se puedan satisfacer sus necesidades a partir de mejoras, una vez puesta en marcha la aplicación. Para ello se crearán una serie de manuales de usuarios y videos tutoriales, y se mantendrá la asistencia a los usuarios.
- **Seguridad:** 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.
- **Software:** Se requiere para el funcionamiento del sistema disponer de un servidor que cuente con Sistema Operativo Linux, Apache 2.0 y MySQL 5.0 o versiones superiores. Los usuarios del sistema deberán contar con un navegador Internet Explorer 5.5 o Mozilla Firefox 2.0 o superior, para poder acceder a las opciones que brinda el sistema.

➤ Hardware

Para el desarrollo y ejecución de la aplicación se necesitará:

Para el servidor de aplicación:

- Microprocesador Pentium IV a 3.0 GHz
- 1 GB de RAM

Para el Cliente:

- Microprocesador Pentium a 233 MHz (Recomendado: Pentium a 500MHz o superior)
- 64 MB de RAM (Recomendado: 128 MB RAM o superior)
- 52 MB de espacio de disco duro. (Recomendado: 120MB para la instalación completa del Internet Explorer 5.5)
- Conexión al servidor a través de MODEM o tarjeta de red.

- **Disponibilidad:** El sistema debe ser capaz de funcionar por si solo en caso de que los servicios de los diferentes componentes de SISalud no estén disponibles. Se debe garantizar el funcionamiento de la aplicación durante las 24 horas del día y los siete días de la semana, con el menor tiempo posible de recuperación de fallos. Se deben crear copias de respaldo periódicas que puedan restaurar el sistema en caso de fallo crítico o pérdida total de la información.
- **Requisitos Legales:** Las herramientas y las tecnologías en que estará basada la aplicación informática deberán cumplir con las licencias de software libre.
- **Persistencia:** La información debe almacenarse en bases de datos con carácter permanente con el objetivo de poder realizar análisis de la misma con el transcurso de los años.

2.2 Pautas del diseño

Para lograr una agradable apariencia y facilitar el uso del software se definieron pautas de diseño, a continuación se enumeran algunas de ellas, para más información consultar el documento de las pautas del diseño definidos por el grupo de arquitectura del proyecto. [25]

1. Los formularios que sean creados deben estar centrados, con un ancho de un 90 % y con bordes de valor 0.
2. La primera fila del formulario debe contener el nombre de la operación que se pretende realizar con el formulario.

3. Cuando se le solicita datos al usuario se hará a través de una tabla de dos columnas. En la columna izquierda se ubicará el nombre del dato solicitado al usuario, alineado a la derecha y en la columna de la derecha el componente adecuado para la recogida de la información.
4. Los botones para efectuar operaciones sobre el formulario se ubicarán en la parte inferior derecha del formulario. Se posicionarán de izquierda a derecha teniendo en cuenta el peso de la operación que representan.
5. Para representar campos que deben ser de entrada obligatoria se colocará un asterisco en el lado derecho del componente.
6. Los mensajes de error ocurridos durante la validación del formulario se mostrarán en la parte superior del campo validado en el cual ocurrió el error y con un color rojo.

2.3 ¿Cómo se pone de manifiesto el uso del patrón MVC que Implementa Symfony?

El uso de un framework que utiliza MVC obliga a dividir y organizar el código de acuerdo a las convenciones establecidas por el framework. El código de la presentación se guarda en la vista, el código de manipulación de datos se guarda en el modelo y la lógica de procesamiento de las peticiones constituye el controlador. Aplicar el patrón MVC a una aplicación resulta bastante útil además de restrictivo.

La implementación que realiza Symfony del patrón de arquitectura MVC incluye varias clases como son:

- *sfController* es la clase del controlador. Se encarga de decodificar la petición y transferirla a la acción correspondiente.
- *sfRequest* almacena todos los elementos que forman la petición (parámetros, cookies, cabeceras)
- *sfResponse* contiene las cabeceras de la respuesta y los contenidos. El contenido de este objeto se transforma en la respuesta HTML que se envía al usuario.
- El *singleton* de contexto (que se obtiene mediante `sfContext::getInstance()`) almacena una referencia a todos los objetos que forman el núcleo de Symfony y puede ser accedido desde cualquier punto de la aplicación.

Symfony toma lo mejor del patrón MVC y lo implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo.

En la capa de la Vista se encuentran los elementos cuyas responsabilidades están asociadas a la presentación de los datos, aquí se ubica el Layout, que contiene la información visual común a todas las páginas. Se encuentra también el código asociado a las plantillas la cual se encarga de visualizar la información que el controlador devuelve y además contiene la lógica de la vista cuya responsabilidad es asegurar la correcta integración del Layout y la Plantilla.

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

En la capa de Modelo se localiza el código referente al acceso a datos, dichas clases se generan de forma automática en dependencia de la estructura de la base de datos, la librería Propel se encarga de esta generación automática y crea la estructura básica de las clases y genera el código necesario. En Symfony el acceso y la modificación de los datos almacenados en la base de datos se realiza mediante objetos. En la generación automática se crean 4 archivos por cada tabla de la Base de Datos, a continuación se explica de manera resumida el porque se procede de esta manera.

En primer lugar puede ser necesario que a medida que el proyecto se esté desarrollando, se añadan tablas o columnas. Además, cada vez que se realizan modificaciones en el esquema de la base de datos se deben regenerar las clases del modelo de objetos. Si se añaden los métodos personalizados en las clases que se generan, se borrarían cada vez que se vuelven a generar esas clases.

Las clases con nombre Base son las que se generan directamente a partir del esquema. Nunca se deberían modificar esas clases, porque cada vez que se genera el modelo, se borran todas las clases. Por otra parte, las clases de objetos propias heredan de las clases con nombre Base. Estas clases no se modifican cuando se genera el modelo, por tanto estas son las clases en las que se añaden los métodos propios.

Ejemplo, para la tabla mc_planilla se generan las siguientes 4 clases.

Nombre de la clase	Descripción
BaseMcPlanillaPeer	Esta clase se genera automáticamente a partir del esquema, no se debe modificar porque cada vez que se genera el modelo se vuelven a crear. Es una clase con todos sus métodos abstractos, cuya responsabilidad es interactuar directamente con la tabla que ella representa en la base de datos.
McPlanillaPeer	Esta clase Hereda de BaseMcPlanillaPeer y se utiliza para añadir las funcionalidades propias no contenidas en la clase base. Se utiliza también para interactuar directamente con la tabla que representa en la base de datos.
BaseMcPlanilla	Clase base que no se debe modificar puesto que se genera automáticamente cada vez que se regenera el modelo, se dice que es una clase objeto, y representa un registro de la tabla en cuestión.
Mcplanilla	Es una clase de acceso a datos que hereda de la clase BaseMcPlanilla, se utiliza para añadir funcionalidades propias puesto que no se modifica cada vez que se regenera el modelo.

2.4 Aplicación de los Patrones GRASP en Symfony

Al usar un framework de PHP para desarrollar, se tiene que adoptar la metodología que este impone, una de las principales ventajas a señalar en el uso de los framework es que están basados en patrones de diseño, se dice incluso que los framework son implementaciones de patrones de diseño

conocidos, esta es una de las características que hace posible la gran usabilidad que tienen los frameworks, casi siempre independiente del tipo de aplicación Web que se desea desarrollar.

Symfony no queda exento de esto, y está concebido de tal manera que obliga al programador a aplicar diferentes patrones de diseño. En las clases del modelo por ejemplo se encuentran las denominadas *Peer* que contienen la información necesaria de la tabla que representan y se utilizan para realizar todas las operaciones de gestión y consulta sobre los datos almacenados en la base de datos, aquí se pone de manifiesto el uso del patrón de diseño, Experto.

En el modelo también se encuentran las clases que implementan la lógica de negocio y de acceso a datos, estas clases no tienen asociaciones con otras de la vista o el controlador por lo que la dependencia en este caso es baja, poniéndose de manifiesto el patrón Bajo Acoplamiento. Varias clases, cada una con un propósito específico, solamente con las responsabilidades que le corresponden, y cada responsabilidad no desempeñada por ninguna otra se observa en la capa de Control, en la que intervienen varias clases controladoras que colaboran entre sí, realizando muchas operaciones desde que se recibe la petición hasta que se determina a qué acción le corresponde actuar, observándose una alta cohesión pues cada clase que interviene en esta secuencia de actividades tiene un único propósito y no se encuentra sobrecargada ni ejecuta responsabilidades que no le conciernen.

Son muchos los patrones que se aplican en la implementación con Symfony, se han mencionado solo algunos, ubicándolos en las capas de Modelo y Control que plantea el patrón arquitectónico MVC.

2.5 Aplicación de los Patrones GoF en Symfony

Patrón Singleton

Clase `sfRouting` – método `getInstance`

```
public static function getInstance()
{
    if (!isset(self::$instance))
    {
        self::$instance = new sfRouting();
    }
    return self::$instance;
}
```

La clase `sfRouting` es una de las que utiliza el controlador frontal (`sfWebFrontController`), es muy utilizada porque es la encargada de enrutar todas las peticiones que se hagan a la aplicación. La clase `sfRouting` define otros métodos muy útiles para la gestión manual de las rutas: `ClearRoutes ()`, `hasRoutes ()`, `getRoutesByName ()`.

Patrón Command

El patrón Command se evidencia en la clase `sfRouting`, pero esta clase puede ser activada o desactivada en dependencia de las preferencias del administrador de la aplicación donde se aplique el framework, por defecto `sfRouting` está desactivada y la clase que actúa en correspondencia es la `sfWebFrontController` y el método `dispatch()` es el que se encarga de determinar cual módulo y acción usar por la petición de parámetros especificados por el usuario.

```
/**
 * Dispatches a request.
 *
 * This will determine which module and action to use by request
 * parameters specified by the user.
 */
public function dispatch()
{
    try
    {
        . . . . .

        // reinitialize filters (needed for unit and functional tests)
        sfFilter::$filterCalled = array();

        // determine our module and action
        $request = $this->getContext()->getRequest();
        $moduleName = $request->getParameter('module');
        $actionName = $request->getParameter('action');

        // make the first request
        $this->forward($moduleName, $actionName);
    }
    . . . . .
}
```

Patrón Decorador

El uso de este patrón se pone de manifiesto en la relación que se establece entre el layout o plantilla global y las diferentes plantillas que forman parte de la vista. El archivo llamado `layout.php` que

contiene el Layout de la página, almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el Layout, o si se mira desde otro punto de vista, el Layout decora la plantilla. Este comportamiento es una implementación del patrón de diseño llamado "Decorator". [22]

La siguiente figura ilustra lo anteriormente descrito.

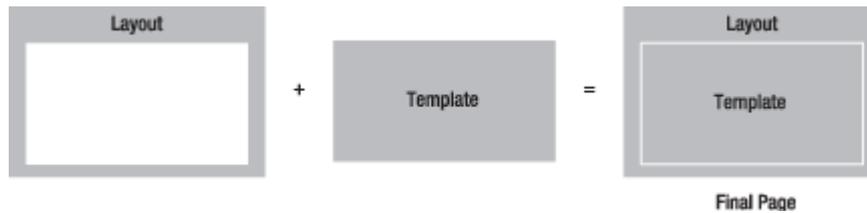


Figura 2 Planilla decorada con un Layout

Patrón Registry

El patrón Registry, pese a su simplicidad, es un patrón sumamente útil para el desarrollo en la programación orientada a objeto. En resumidas palabras, el Patrón Registry, es un medio simple y eficiente de compartir datos y objetos en la aplicación sin tener que preocuparse de mantener numerosos parámetros o hacer uso de variables globales. La aplicación de este patrón se ve en la clase sfConfig la cual es la encargada de almacenar todas las variables de uso global en la aplicación, decir también que esta clase aplica el patrón Singleton. [22]

2.6 Modelo de diseño

En el diseño se modela el sistema y es el centro de atención al final de la fase de elaboración y el comienzo de las iteraciones de construcción. Esto contribuye a una arquitectura estable y sólida.

Propósitos del flujo de trabajo diseño:

- Adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia y tecnologías de interfaz de usuario.
- Crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.

- Descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo.
- Capturar las interfaces entre los subsistemas antes en el ciclo de vida del software, lo cual es muy útil cuando se crean interfaces como elementos de sincronización entre diferentes equipos de desarrollo. [26]

2.6.1 Descripción de las Clases del diseño

Nombre de la Clase: mcActions.class
Nombre del método: executeAgregarmf()
Descripción: Este método se encarga de mostrar los datos primarios del paciente seleccionado y se insertan los demás datos del paciente, insertándose el paciente como malformado.
Nombre del método: validateAgregarmf()
Descripción: Permite las validaciones que no son posibles de realizar en los archivos "yml", haciéndose directamente contra la BD.
Nombre del método: handleErrorAgregarmf()
Descripción: Este método se ejecuta cuando el archivo agregarmf.yml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.
Nombre del método: executeAgregarcs()
Descripción: Este método se encarga de insertar los pacientes primarios y secundarios de los pacientes controles simples de los malformados.
Nombre del método: validateAgregarcs()
Descripción: Permite las validaciones que no son posibles de realizar en los archivos .yml, haciéndose directamente contra la BD.
Nombre del método: handleErrorAgregarcs()
Descripción: Este método se ejecuta cuando el archivo agregarcs.yml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.
Nombre del método: executeAgregarcg()
Descripción: Este método se encarga de mostrar los datos primarios de un paciente gemelo control de un malformado simple, además de insertarlo como control gemelo.
Nombre del método: validateAgregarcg()

Descripción: Permite las validaciones que no son posibles de realizar en los archivos .yaml, haciéndose directamente contra la BD.
Nombre del método: handleErrorAgregarcg()
Descripción: Este método se ejecuta cuando el archivo agregarcg.yaml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.
Nombre del método: executeAgregarcgh()
Descripción: Este método se encarga de insertar los datos de un paciente control que es gemelo con un paciente malformado.
Nombre del método: handleErrorAgregarcgh()
Descripción: Este método se ejecuta cuando el archivo agregarcgh.yaml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.
Nombre del método: executeBuscar_modificar()
Descripción: Este método se encarga de buscar los pacientes a modificar, mostrando los resultados de la búsqueda.
Nombre del método: executeCodificarmf()
Descripción: Este método se encarga de mostrar la descripción de la malformación del paciente y además se codifica el paciente de acuerdo a dicha descripción.
Nombre del método: executeInterrupción()
Descripción: Este método se encarga de insertar los datos de los pacientes que son interrupción, almacenando principalmente los datos de la madre de este paciente.
Nombre del método: handleErrorInterrupcion()
Descripción: Este método se ejecuta cuando el archivo interrupción.yaml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.
Nombre del método: executeModificarmf()
Descripción: Este método se encarga de mostrar los datos del paciente malformado seleccionado y además almacena los cambios que se realicen en la plantilla correspondiente a este execute.
Nombre del método: validateModificarmf()
Descripción: Permite las validaciones que no son posibles de realizar en los archivos .yaml, haciéndose directamente contra la BD.
Nombre del método: handleErrorModificarmf()
Descripción: Este método se ejecuta cuando el archivo modificarmf.yaml o la función validate

<p>correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.</p>
<p>Nombre del método: executeMalfxClasf()</p>
<p>Descripción: Este método se encarga de cargar dada una descripción de las malformaciones la clasificación de las mismas.</p>
<p>Nombre del método : executeClasificación()</p>
<p>Descripción: Este método se encarga de devolver las descripciones de los tipos de las malformaciones simples.</p>
<p>Nombre del método: executeCodigoMal()</p>
<p>Descripción: Este método se encarga de devolver los códigos de acuerdo a la malformación seleccionada por el genetista.</p>
<p>Nombre del método: executeModificarcs()</p>
<p>Descripción: Este método se encarga de mostrar los datos del paciente control seleccionado y además almacena los cambios que se realicen en la plantilla correspondiente a este execute.</p>
<p>Nombre del método: handleErrorModificarcs()</p>
<p>Descripción: Este método se ejecuta cuando el archivo modificarcs.yml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.</p>
<p>Nombre del método: executeModificarint()</p>
<p>Descripción: Este método se encarga de mostrar los datos del paciente que es clasificado como interrupción y además almacena los cambios que se realicen en la plantilla correspondiente a este execute.</p>
<p>Nombre del método: validateModificarint()</p>
<p>Descripción: Permite las validaciones que no son posibles de realizar en los archivos .yml, haciéndose directamente contra la BD.</p>
<p>Nombre del método: handleErrorModificarint()</p>
<p>Descripción: Este método se ejecuta cuando el archivo modificarint.yml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.</p>
<p>Nombre del método: executeBuscarMC()</p>
<p>Descripción: Este método se encarga de mostrar los criterios de búsqueda para buscar y seleccionar los pacientes que se le van a insertar los datos complementarios dentro del módulo de RECUMAC.</p>
<p>Nombre del método: executeBuscar_modificar()</p>
<p>Descripción: Este método se encarga de mostrar los criterios de búsqueda para buscar y</p>

seleccionar los pacientes que ya existen dentro del módulo de RECUMAC para modificar sus datos.
Nombre del método: <code>executeCodificarmf()</code>
Descripción: Este método se encarga de mostrar los datos necesarios para codificar la malformación del paciente además de guardar las malformaciones que se le asignen a un paciente.
Nombre del método: <code>validateCodificarmf()</code>
Descripción: Permite las validaciones que no son posibles de realizar en los archivos .yml, haciéndose directamente contra la BD.
Nombre del método: <code>handleErrorCodificarmf()</code>
Descripción: Este método se ejecuta cuando el archivo <code>codificarmf.yml</code> o la función <code>validate</code> correspondiente al <code>execute</code> lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.
Nombre del método: <code>DatosPaginaCodificar()</code>
Descripción: Este método es el encargado de capturar a la persona seleccionada por el genetista para poder acceder a los atributos de esa persona, es llamado dentro del <code>executeCodificarmf</code> .
Nombre del método: <code>Codificar_malformación()</code>
Descripción: Este método es el encargado de codificar a los pacientes malformados, o sea es aquí donde se le asigna a un paciente todas sus malformaciones.
Nombre del método: <code>insertarDatosCS()</code>
Descripción: Este método se encarga de insertar todos los datos complementarios de los pacientes que son controles.
Nombre del método: <code>executeGetSelected()</code>
Descripción: Este método se encarga de gestionar todo lo referente al código automático del instrumento, en este caso de la página <code>agregarmf</code> .
Nombre del método: <code>insertarDatosCG()</code>
Descripción: Este método se encarga de insertar todos los datos complementarios de los pacientes que son gemelos controles.
Nombre del método: <code>insertarInterrupción()</code>
Descripción: Este método se encarga de insertar todos los datos de los pacientes que son interrupciones.
Nombre del método: <code>insertarMalformado()</code>
Descripción: Este método se encarga de insertar todos los datos de los pacientes malformados.
Nombre del método: <code>insertarHermanoGemelo()</code>
Descripción: Este método se encarga de insertar todos los datos de los pacientes que son hermanos gemelos de pacientes malformados.

Nombre del método: executeBuscarp()
Descripción: Este método se encarga de mostrar los criterios de búsqueda para buscar los pacientes malformados que aún no están codificados.
Nombre del método: executeReporteFRyDC()
Descripción: Este método se encarga de mostrar el valor del reporte de prevalencia al nacimiento y defectos congénitos.
Nombre del método: validateReporteFRyDC()
Descripción: Permite las validaciones que no son posibles de realizar en los archivos .yml, haciéndose directamente contra la BD.
Nombre del método: handleErrorReporteFRyDC()
Descripción: Este método se ejecuta cuando el archivo reporteFRyDC.yml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.
Nombre del método: FCyDC()
Descripción: Este método se encarga de calcular el reporte factores de riesgo y defectos congénitos.
Nombre del método: executeReporteprevalencia()
Descripción: Este método se encarga de calcular el reporte de prevalencia al nacimiento.
Nombre del método: handleErrorReporteprevalencia()
Descripción: Este método se ejecuta cuando el archivo reporteprevalencia.yml o la función validate correspondiente al execute lanzan un error, el resultado es que no se envíe el formulario y se muestren los mensajes de errores en la página. Se encarga de mantener la estructura de la página con su información.
Nombre del método: Prevalencia()
Descripción: Este método es el encargado de calcular la prevalencia al nacimiento.
Nombre del método: Area_Salud_Mun()
Descripción: Este método es el encargado de devolver un arreglo con todas las áreas de salud y hospitales pertenecientes a un municipio determinado.
Nombre del método: executeBuscarPlanilla()
Descripción: Este método es el encargado de visualizar los resultados de la búsqueda del buscar pacientes malformados sin codificar.
Nombre del método: BuscarPlanillaBD()
Descripción: Este método es el encargado de realizar las consultas sobre la base de datos y devolver los resultados de acuerdo a los criterios de búsqueda introducidos.
Nombre del método: updateModificar()
Descripción: Este método es el encargado de cargar todos los datos de los pacientes

malformados en el modificar de este tipo de paciente.
Nombre del método: cargarControlSimple()
Descripción: Este método es el encargado de cargar todos los datos de los pacientes controles en el modificar de este tipo de paciente.
Nombre del método: cargarInterrupcion()
Descripción: Este método es el encargado de cargar todos los datos de los pacientes interrupciones en el modificar de este tipo de paciente.
Nombre del método: TablaFA(\$id)
Descripción: Este método es un método auxiliar para verificar si la persona malformada esta en la tabla DgFactores_ambientales, es usado en el reporte de FCy DC.
Nombre del método: TablaHistEmb(\$id)
Descripción : Este método es un método auxiliar para verificar si la persona malformada esta en la tabla DgHist_embarazo, es usado en el reporte de FCy DC.
Nombre del método: TablaDgRadiacion(\$id)
Descripción: Este método es un método auxiliar para verificar si la persona malformada esta en la tabla DgRadiacion, es usado en el reporte de FCy DC.
Nombre del método: TablaDgAaborto(\$id)
Descripción: Este método es un método auxiliar para verificar si la persona malformada esta en la tabla DgAaborto, es usado en el reporte de FCy DC.
Nombre del método: TablaDgEnf_maternas(\$id)
Descripción: Este método es un método auxiliar para verificar si la persona malformada esta en la tabla DgEnf_mat, es usado en el reporte de FCy DC.
Nombre del método: TablaDgfactores_gen(\$id)
Descripción: Este método es un método auxiliar para verificar si la persona malformada esta en la tabla TablaDgfactores_genéticos, es usado en el reporte de FC y DC.
Nombre del método: executeClasif()
Descripción: Este método es el encargado de devolver todas las malformaciones simples de la tabla nomencladora McTipoSindrome.
Nombre del método: executeSimple_ultimo()
Descripción: Este método es el encargado de devolver todas las malformaciones simples de la tabla nomencladora McTipoMonogenica.
Nombre del método: executeRedireccionar_modificar()
Descripción: Este método es el encargado de verificar el tipo de paciente seleccionado y mandar a ejecutar la página correspondiente.
Nombre del método: executeUnidad()
Descripción: Este método es el encargado de devolver las unidades de salud que sean hospitales dado un municipio seleccionado por el genetista.

Nombre del método: <code>executeAreas()</code>
Descripción: Este método es el encargado de devolver las unidades de salud que sean áreas de salud dado un municipio seleccionado por el genetista.
Nombre del método: <code>executeCi()</code>
Descripción: Este método es el encargado de devolver el carnet de identidad de un gemelo que ya existe.
Nombre del método: <code>Agregarcs_c()</code>
Descripción: Este método es el encargado de cargar los datos primarios de un control gemelo.
Nombre del método: <code>Agregarcg_c()</code>
Descripción: Este método es el encargado de cargar los datos primarios de un control gemelo que es gemelo con un malformado.
Nombre del método: <code>CargarDatosAgregarmf()</code>
Descripción: Este método es el encargado de cargar los datos primarios de un paciente malformado.

2.6.2 Diagramas de clases del diseño

Los diagramas de clases son ampliamente utilizados en el modelado de sistemas orientados a objetos y se emplean para representar las relaciones que se establecen entre las clases. Los diagramas de clases del diseño se utilizan para modelar la vista de diseño estática de un sistema, la vista dinámica se representa a través de los diagramas de interacción.

Para modelar las clases del diseño del sistema utilizando Symfony, es necesario delimitar bien lo que es del framework de lo que es propio del caso de uso. Se explicará de manera general el significado de los principales elementos presentes en los diagramas para lograr una mejor comprensión de los mismos, todos los diagramas se modelaron siguiendo estos significados, las diferencias que existen entre ellos se debe a las características independientes de cada caso de uso.

Los componentes de Symfony se encapsulan en un subsistema que contiene todos los elementos del framework que actúan o intervienen en la realización de los casos de uso, pero al ser propios de Symfony, la manera en que trabajan es transparente al programador, por una cuestión de no complejizar tanto el diagrama de clases del diseño se decidió modelarlos como un subsistema con el nombre de componentes de Symfony.

Otro elemento fundamental de Symfony es el componente que realiza el mapeo objeto-relacional, el cual proporciona la persistencia de los objetos y un servicio de consulta, este componente se denomina Propel y se representa también como un subsistema.

Para lograr mayor organización, los diagramas están organizados de tal manera que cada elemento del diseño se ubica según su naturaleza en el nivel correspondiente que define el patrón MVC.

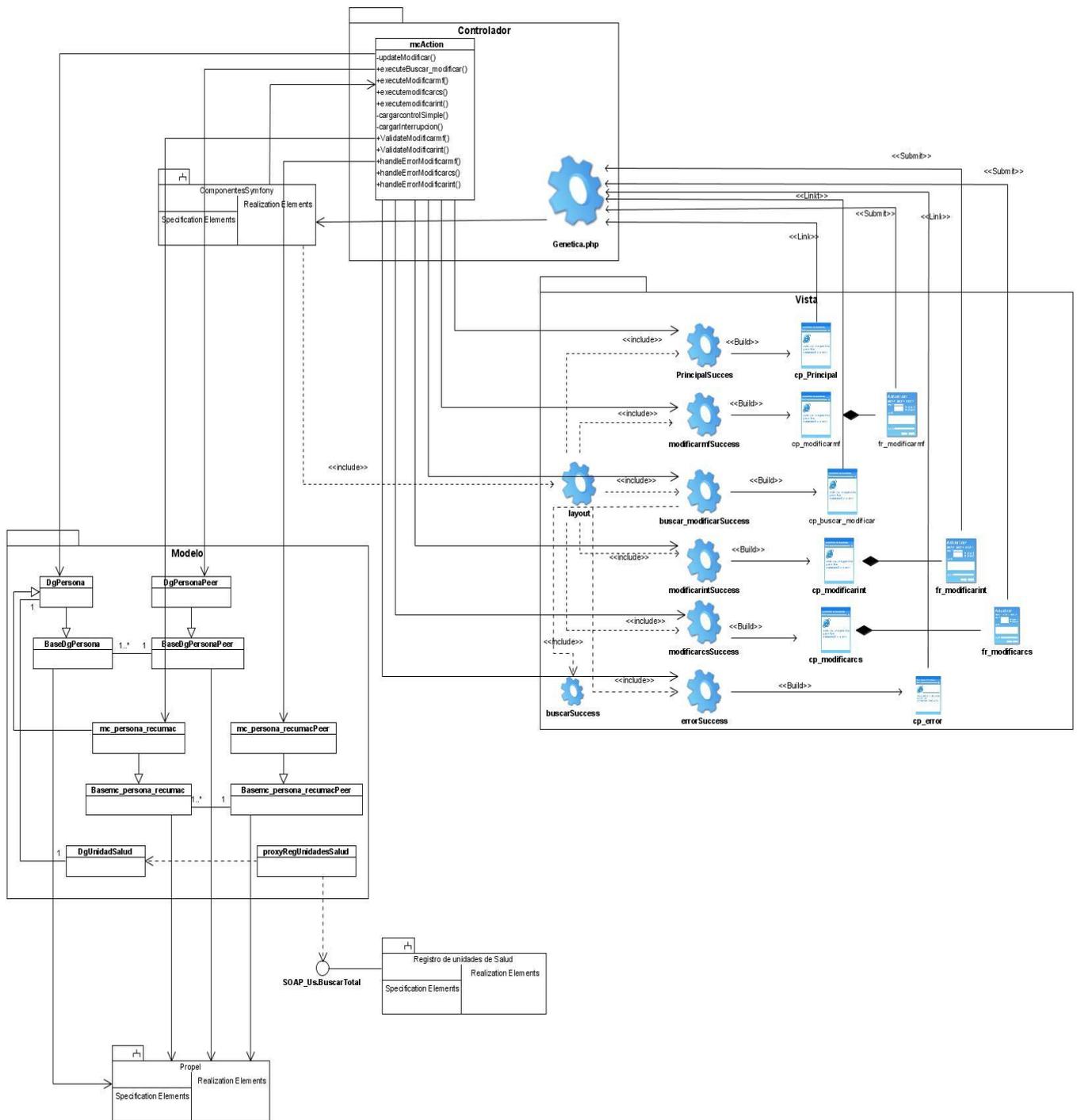


Figura 4 Diagrama de clase del diseño CU modificar datos de los pacientes

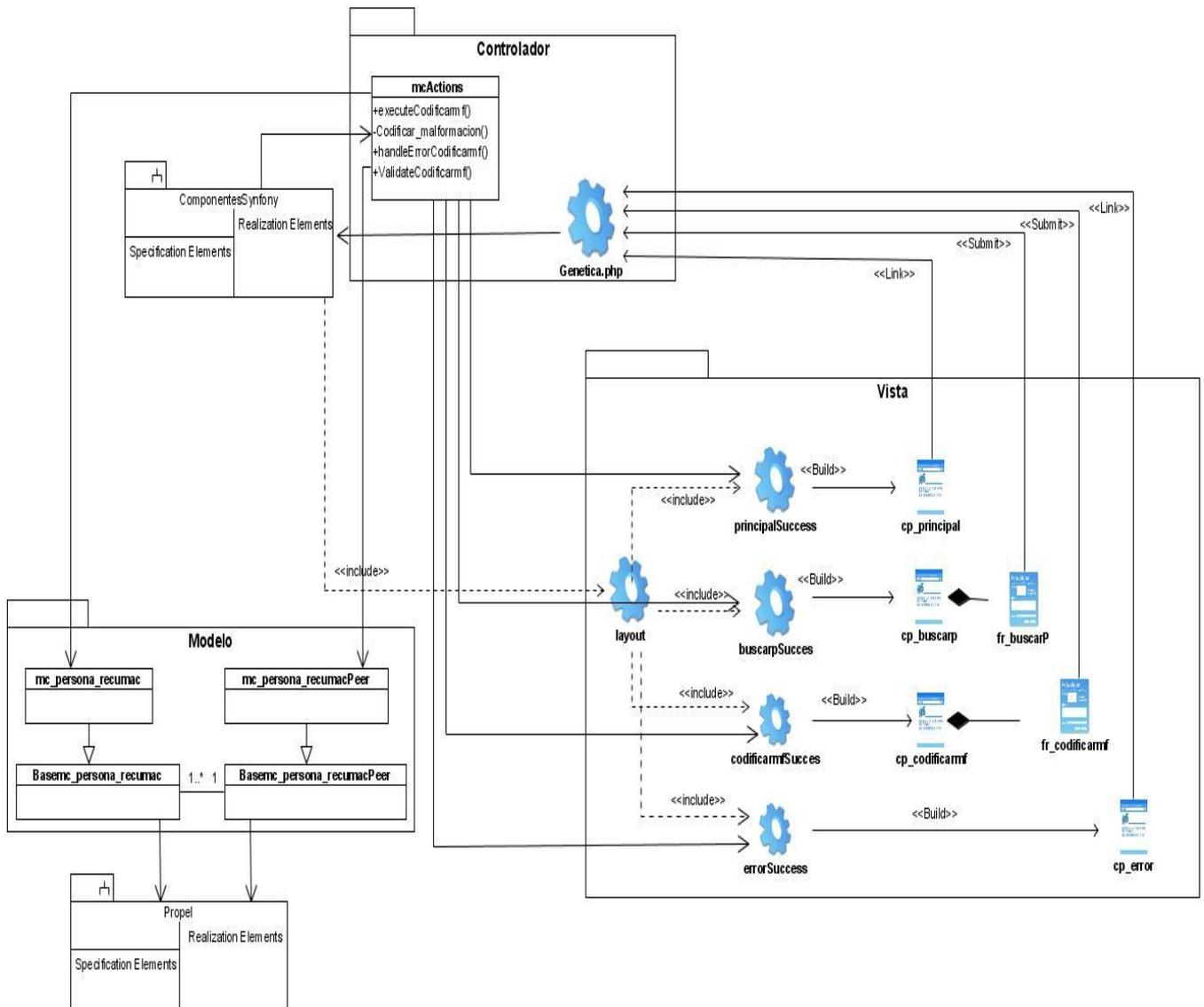


Figura 5 Diagrama de clase del diseño CU codificar malformación

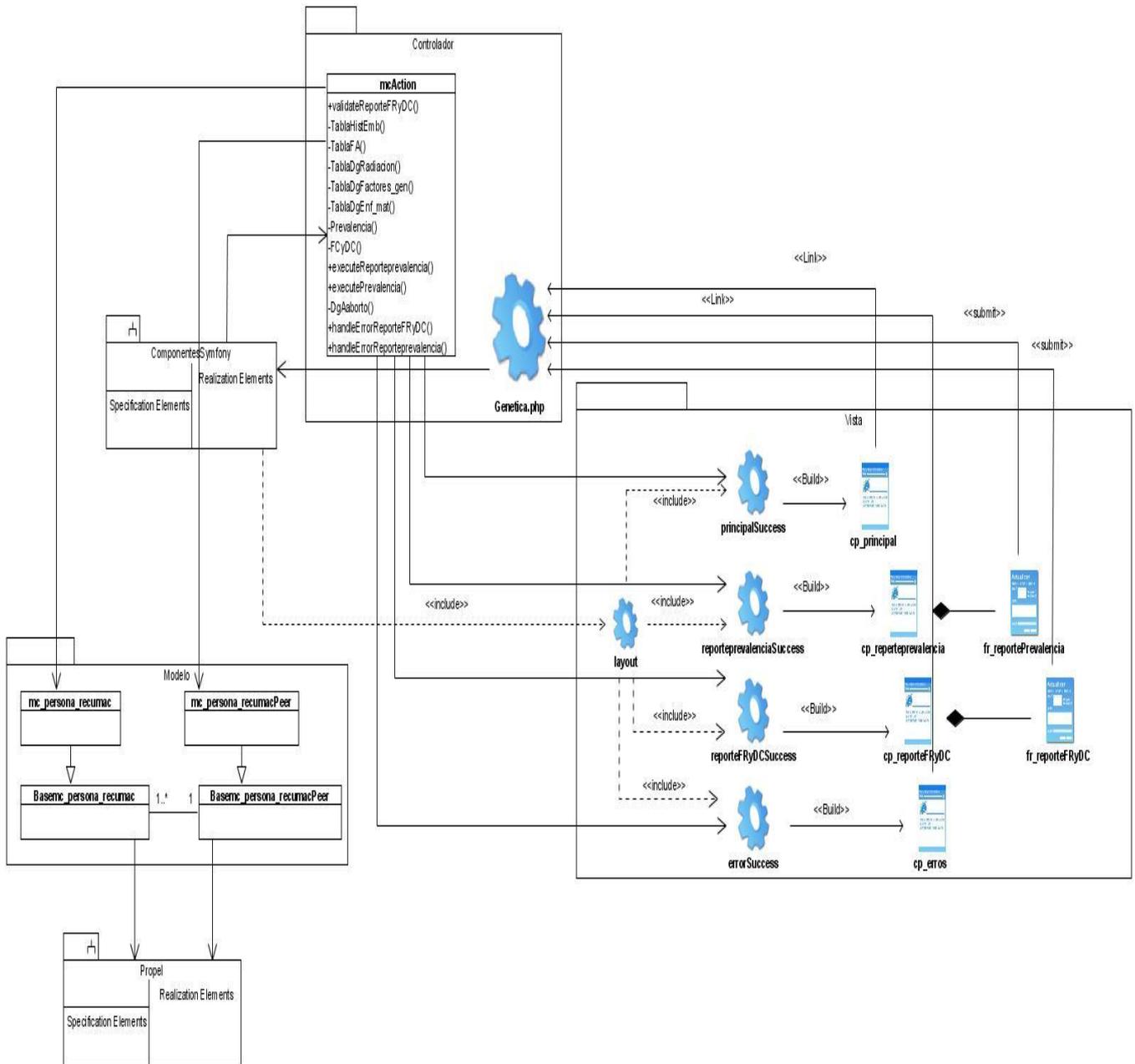


Figura 6 Diagrama de clase del diseño CU generar reportes estadísticos

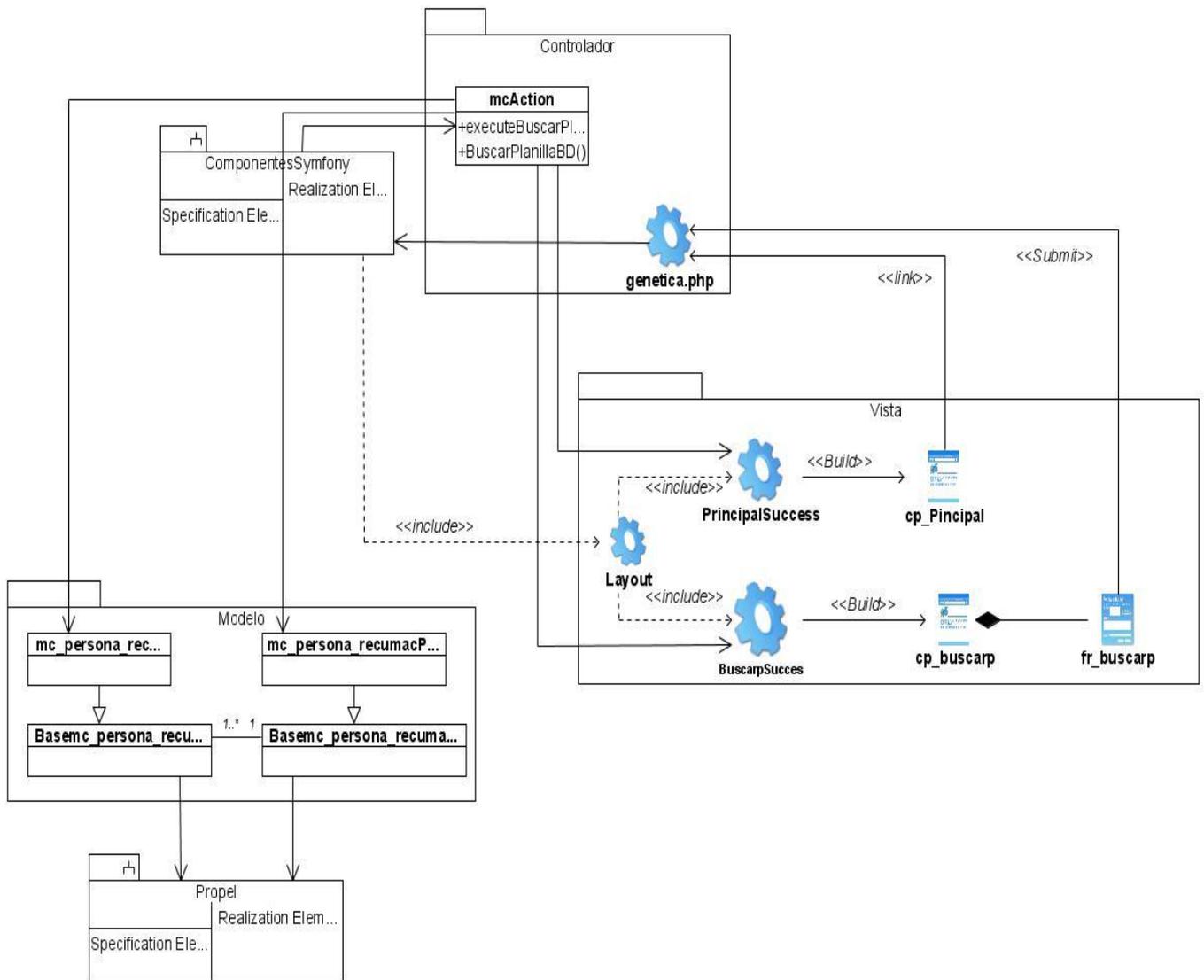


Figura 7 Diagrama de clase del diseño CU buscar paciente a codificar

2.6.3 Diagramas de Interacción (Secuencia)

Los diagramas de interacción se utilizan para modelar la vista dinámica del diseño, en ellos se muestran las relaciones que se establecen entre los objetos y los diferentes mensajes que se pueden enviar entre ellos.

Realizar los diagramas de secuencia con Symfony es una tarea compleja, puesto que modelar la secuencia de mensajes que se establecen entre los objetos que forman parte de los componentes de Symfony es engorroso. Por tal motivo se decidió representar solamente la información específica al caso de uso, para lograr una mejor comprensión.

A continuación se muestran algunos de los escenarios de los diagramas de secuencia, para más información consultar el expediente de proyecto.

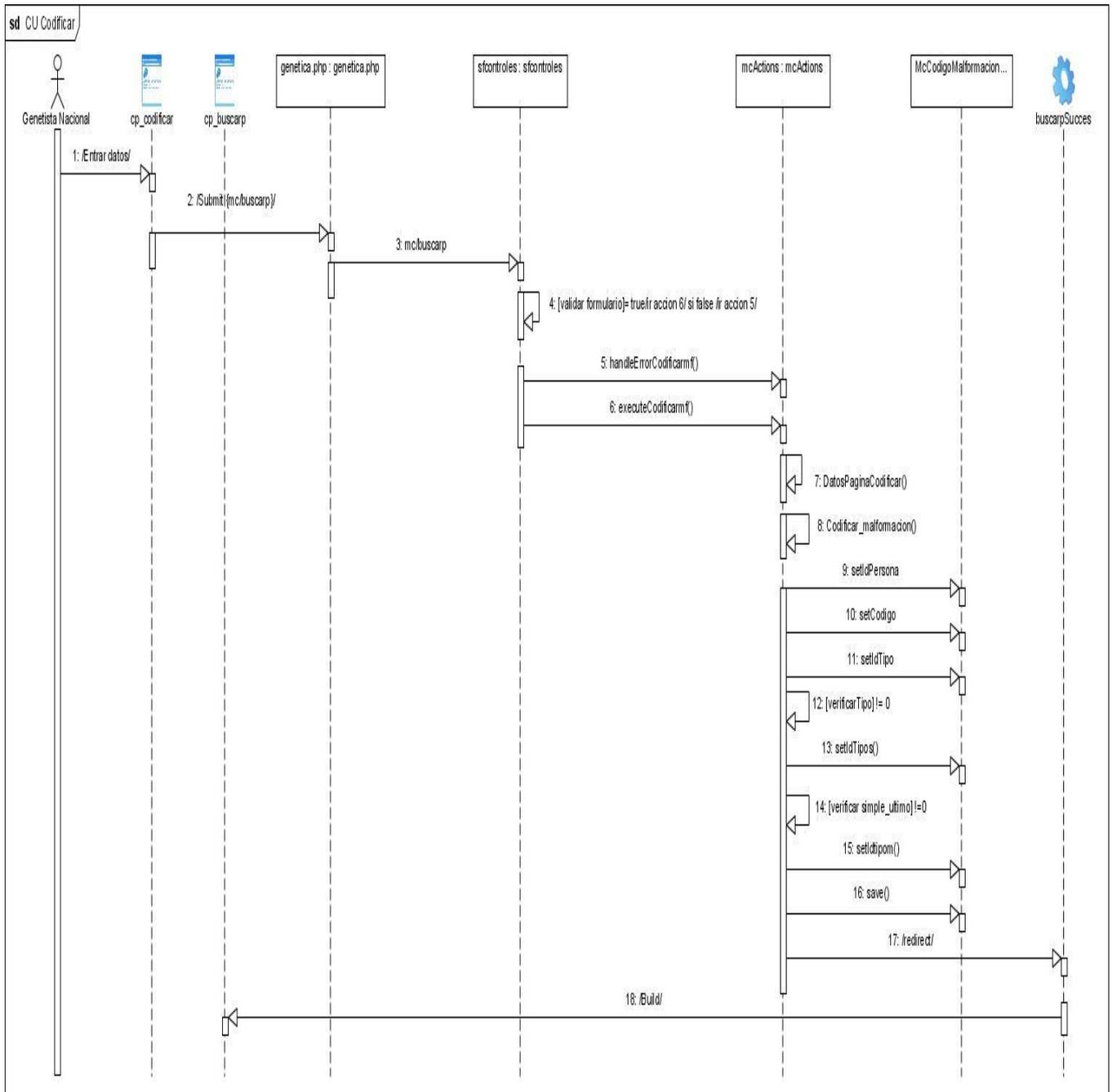


Figura 8 Diagrama de secuencia CU Codificar Malformación.

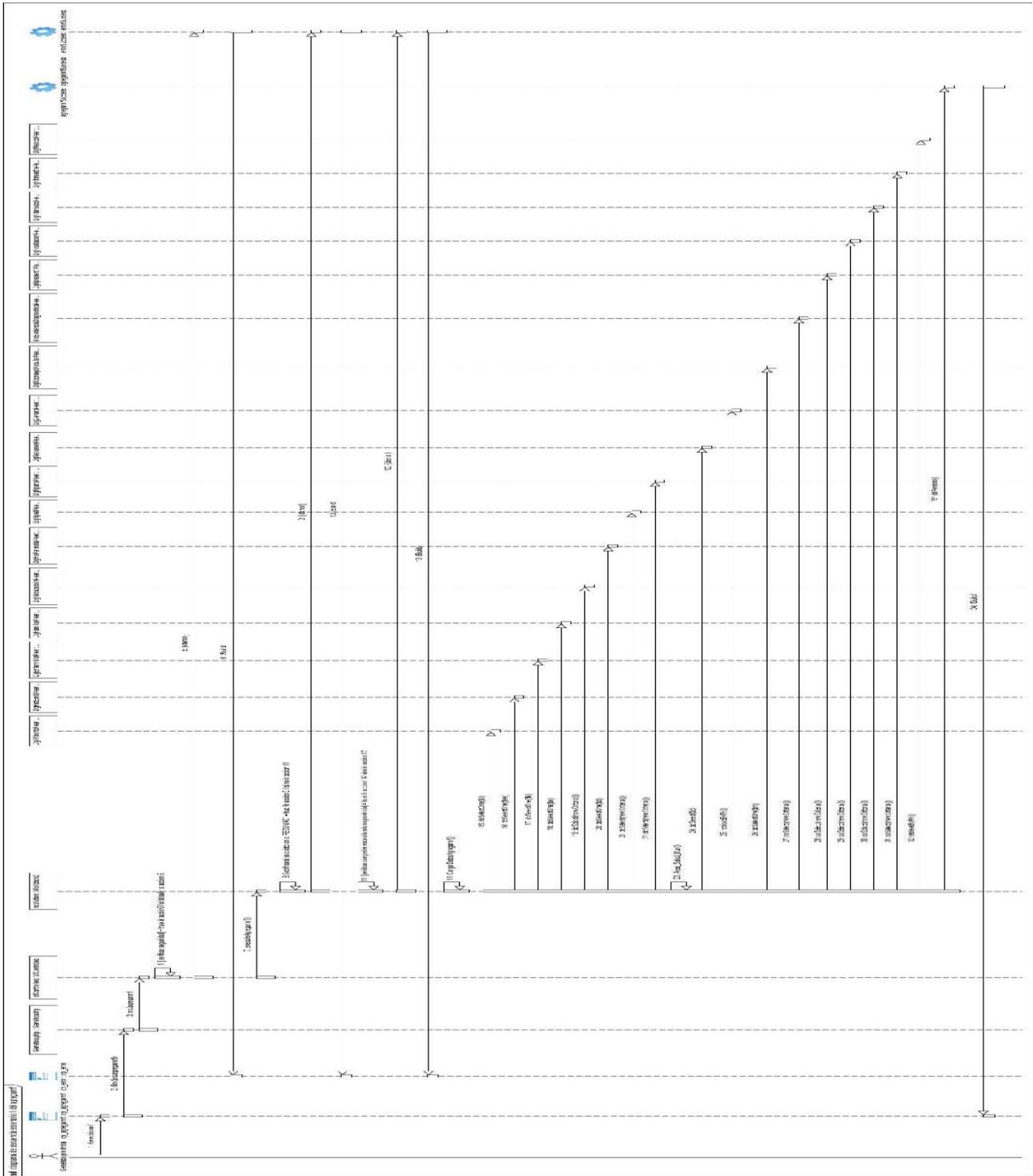


Figura 10 Diagrama de secuencia CU Insertar Datos Complementarios del Paciente (escenario seleccionar paciente).

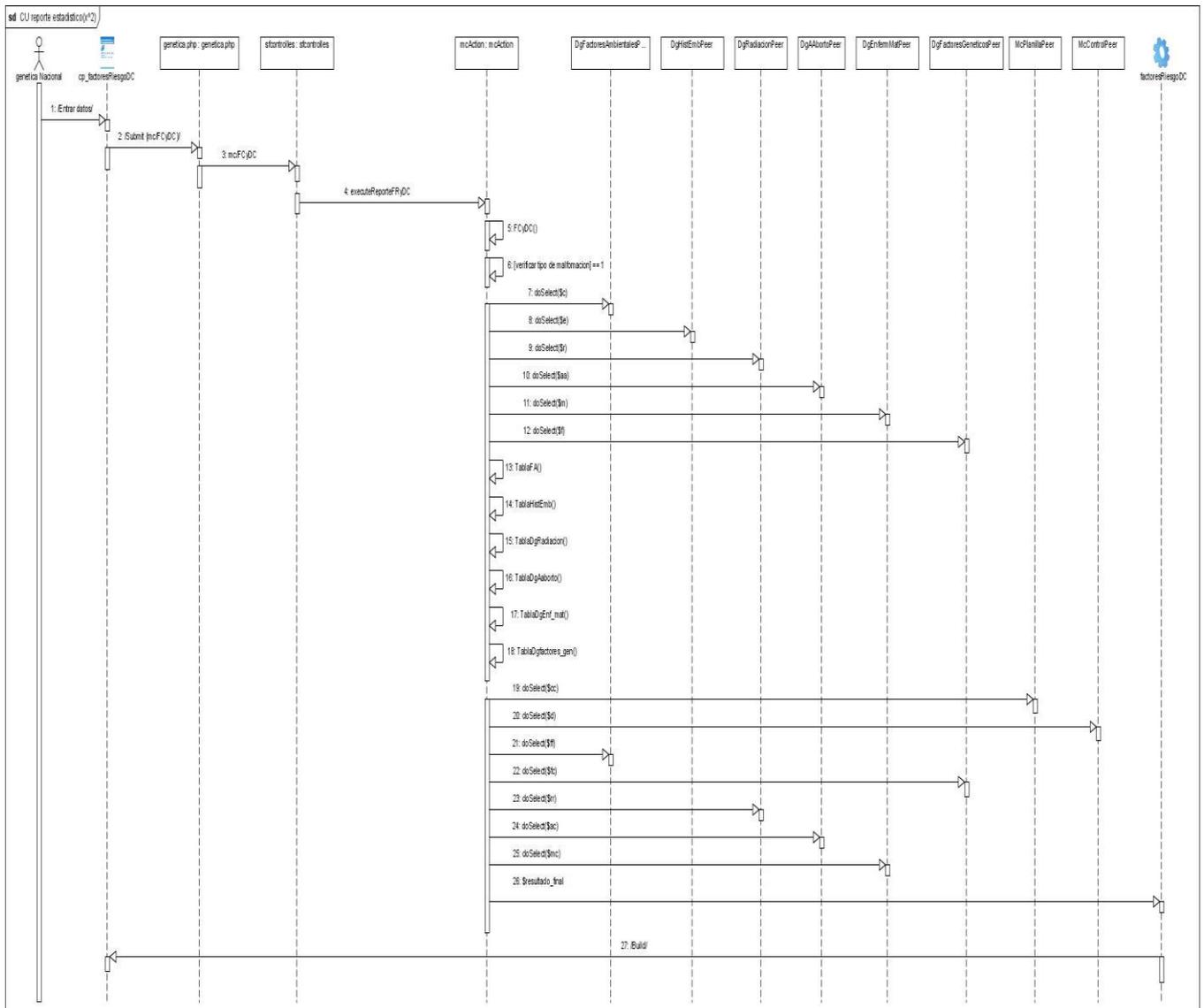


Figura 11 Diagrama de secuencia CU Generar Reporte Estadístico (Reporte Factor de riesgo y Defecto Congénito).

2.7 Modelo de despliegue

El modelo de despliegue muestra la configuración de los nodos de procesamiento en tiempo de ejecución, los links de comunicación entre ellos, y las instancias de los componentes y objetos que residen en ellos. Consiste en:

- **Nodos:** Elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos
- **Dispositivos:** Nodos estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela
- **Conectores:** Expresa el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo.[27]

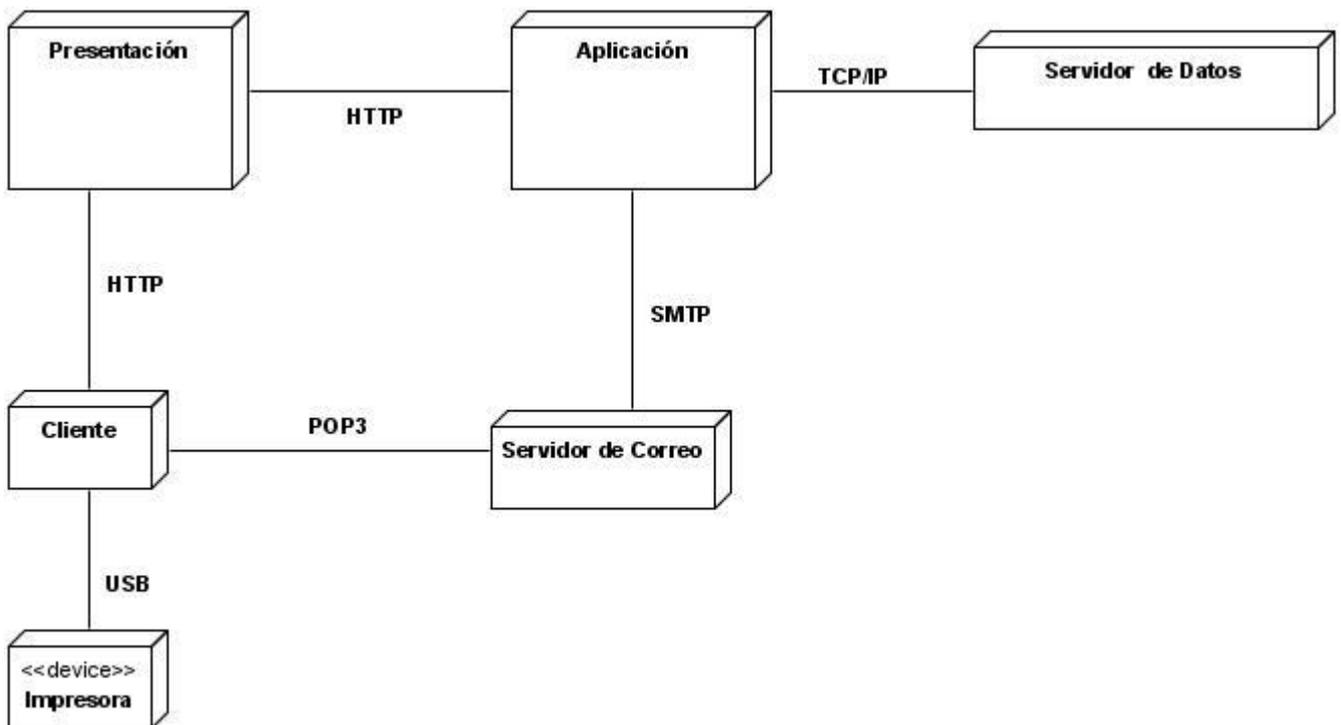


Figura 12 Diagrama de Despliegue

En el diagrama de despliegue que se muestra en la figura 12 es el correspondiente al SIGM, los nodos que se encuentran dentro de círculos son los necesarios para el despliegue del módulo de RECUMAC.

2.8 Seguridad

En Symfony la seguridad se garantiza a través de un filtro especial que verifica si el usuario actual tiene los privilegios necesarios para ejecutar la acción a la que se intenta acceder. En este framework, 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.[22]

Para restringir el acceso a una acción en el módulo del RECUMAC se edita un archivo de configuración YAML llamado "security.yml" en el directorio config del módulo malformaciones congénitas (MC). En este archivo, se pueden especificar los requerimientos de seguridad que los usuarios deberán tener para ejecutar una o todas las acciones. En el caso del RECUMAC las acciones requieren que los usuarios estén autenticados en el sistema para poder acceder a ellas. Los roles utilizados son los definidos a nivel de aplicación por el módulo de administración del SIGM. Para el caso particular del RECUMAC solamente son usados los grupos de nacMcEditor, nacMcRevisor, provMcEditor y provMcRevisor, estos son los roles que intervienen en el RECUMAC, las operaciones son restringidas por grupos, lo que significa que un usuario que pertenezca a otro grupo no podrá ejecutar ninguna acción dentro del módulo de malformaciones congénitas.

A continuación se muestra el contenido del archivo security.yml del módulo RECUMAC, donde se establecen los permisos para cada acción del módulo.

```
agregarmf:
  credentials: provMCEditor

buscarMC:
  credentials: provMCEditor

interrupcion:
```

credentials: provMCEditor

buscar_modificar:

credentials: [[provMCEditor, nacMcEditor, nacMcRevisor]]

buscarp:

credentials: [[provMCEditor, nacMcEditor]]

codificarmf:

credentials: [[provMCEditor, nacMcEditor]]

reporteprevalencia:

credentials: [[provMCEditor, nacMCEditor]]

reporteFRyDC:

credentials: [[provMCEditor, nacMCEditor]]

agregarcs:

credentials: provMCEditor

agregarcg:

credentials: provMCEditor

agregarcgh:

credentials: provMCEditor

modificarmf:

credentials: [[provMCEditor, nacMcEditor, nacMcRevisor]]

modificarcs:

credentials: [[provMCEditor, nacMcEditor, nacMcRevisor]]

modificarint:

credentials: [[provMCEditor, nacMcEditor, nacMcRevisor]]

all:

is_secure:on

2.9 Validación

La validación de los datos de la acción, que normalmente se corresponden con los parámetros de la petición es una tarea repetitiva y tediosa. Symfony incluye un sistema de validación, utilizando métodos de la clase acción.

Cuando un usuario realiza una petición a una acción, Symfony siempre busca primero un método llamado `validateNombredeLaAccion ()`. Si lo encuentra, Symfony ejecuta ese método. El valor de retorno de esta validación determina el siguiente método que se ejecuta: si devuelve `true`, entonces se ejecuta el método `executeNombredeLaAccion ()`; si el resultado es `false` entonces se ejecuta `handleErrorNombredeLaAccion ()`. En el caso de que `handleNombredeLaAccion ()` no exista, Symfony busca un método genérico llamado `handleError ()`. Si tampoco existe, simplemente devuelve el valor `sfView::ERROR` para producir la plantilla `NombredeLaAccionError.php`. La clave para un correcto funcionamiento de la validación es respetar la convención de nombres para los métodos de la acción, que se muestra a continuación:

- `validateNombreAccion` es el método de validación, que devuelve `true` o `false`. Se trata del primer método buscado cuando se solicita la acción `NombreAccion`. Si no existe, la acción se ejecuta directamente.
- `handleErrorNombreAccion` es el método llamado cuando el método de validación falla. Si no existe, entonces se muestra la plantilla `Error`.
- `executeNombreAccion` es el método de la acción. Debe existir para todas las acciones.

Se puede utilizar este mecanismo para implementar la validación de los formularios (esto es, controlar los valores introducidos por el usuario en un formulario antes de procesarlo).[22]

Para la validación con ficheros `YML` Symfony incluye un conjunto de clases tales como `sfStringValidator`, `sfMailValidator`, `sfNumberValidator`, `sfURLValidator`, `sfFileValidator` entre otras; y además la posibilidad de implementación de nuevas clases en los casos que las existentes no resuelvan el problema en cuestión. Específicamente en este caso se utilizó la clase `geneticaDateValidator`, que fue implementada por otro módulo del SIGM, esta clase extienden de la clase del Symfony `sfValidator`. Esta clase es utilizada con el objetivo de validar en el caso de uso de

generar reportes, específicamente en el reporte de prevalencia al nacimiento es usada para validar que el período de tiempo seleccionado por el usuario sea válido.

Los formularios utilizados en el módulo de RECUMAC son validados con archivos YML vinculados a las acciones, y en ocasiones los archivos YML no son suficientes para alguna validación, por lo que se implementa el método `executeValidateNombredeLaAccion ()` esta validación se estaría haciendo del lado del servidor, también se validan algunos datos en el lado del cliente con el lenguaje Javascript. Para mostrar los mensajes de error al usuario se utiliza un helpers especializado en esto que muestra de una forma muy amigable al usuario los mensajes de notificación de error sobre las acciones incorrectas que se produzcan en la entrada de datos, de esta forma se validan todos los campos necesarios en cada uno de los formularios.

A continuación se muestra el archivo YML perteneciente a la página reporte prevalencia:

```
methods:
  post:
    - "fecha_inicio"
    - "fecha_fin"
    - "total_nac"
  fillin:
    enabled: true
  names:
    total_nac:
      required: Yes
      required_msg: Entre un valor
      validators: nac
    fecha_inicio:
      required: Yes
      required_msg: Seleccione una fecha
      validators: fechaInicio
    fecha_fin:
      required: Yes
```

```
required_msg: Seleccione una fecha
validators: fechaFin
fechaInicio:
  class:      geneticaDateValidator
  param:
    date_error: El campo fecha no es correcto
    compare:    fecha_fin
    operator:   <
    compare_error: La fecha de inicio debe ser anterior a la fecha de fin
fechaFin:
  class:      geneticaDateValidator
  param:
    date_error: El campo fecha no es correcto
    compare:    fecha_inicio
    operator:   >
    compare_error: La fecha de inicio debe ser anterior a la fecha de fin
nac:
  class:      sfNumberValidator
  param:
    min:      1
    min_error: Minimo valor 1
```

En este archivo YML se ponen en práctica algunos de los validadores antes mencionados, en los demás archivos YML se ponen de manifiesto otros de los validadores mencionados.

Conclusiones

Con los diagramas de clases de diseño realizados y descritos en este capítulo se garantiza que las clases tengan el comportamiento que requieren las realizaciones de los casos de uso. Se concluye además que, la modelación de los aspectos dinámicos del sistema a través de los diagramas de secuencia posibilitan detallar la interacción que existe entre cada uno de los elementos que intervienen en la realización de los diferentes casos de uso involucrados en la realización de este trabajo. Estos dos elementos contribuyen considerablemente a lograr una implementación del sistema sin ambigüedad.

Capítulo 3: Implementación

Introducción

En el presente capítulo se representa mediante el artefacto modelo de componentes que se obtiene en el flujo de trabajo de implementación, la organización y las dependencias lógicas que existen entre los ficheros que contienen el código fuente. También se explica los estándares de codificación utilizados, los principales métodos desarrollados y la información visual de todas las pantallas de la aplicación.

3.1 Modelo de implementación

El modelo de implementación representa la composición física de la implementación en términos de subsistemas de implementación, y elementos de implementación (directorios y archivos, incluyendo código fuente, datos y archivos ejecutables).

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes software, sean estos componentes de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo.[28]

3.1.1 Diagramas de componentes

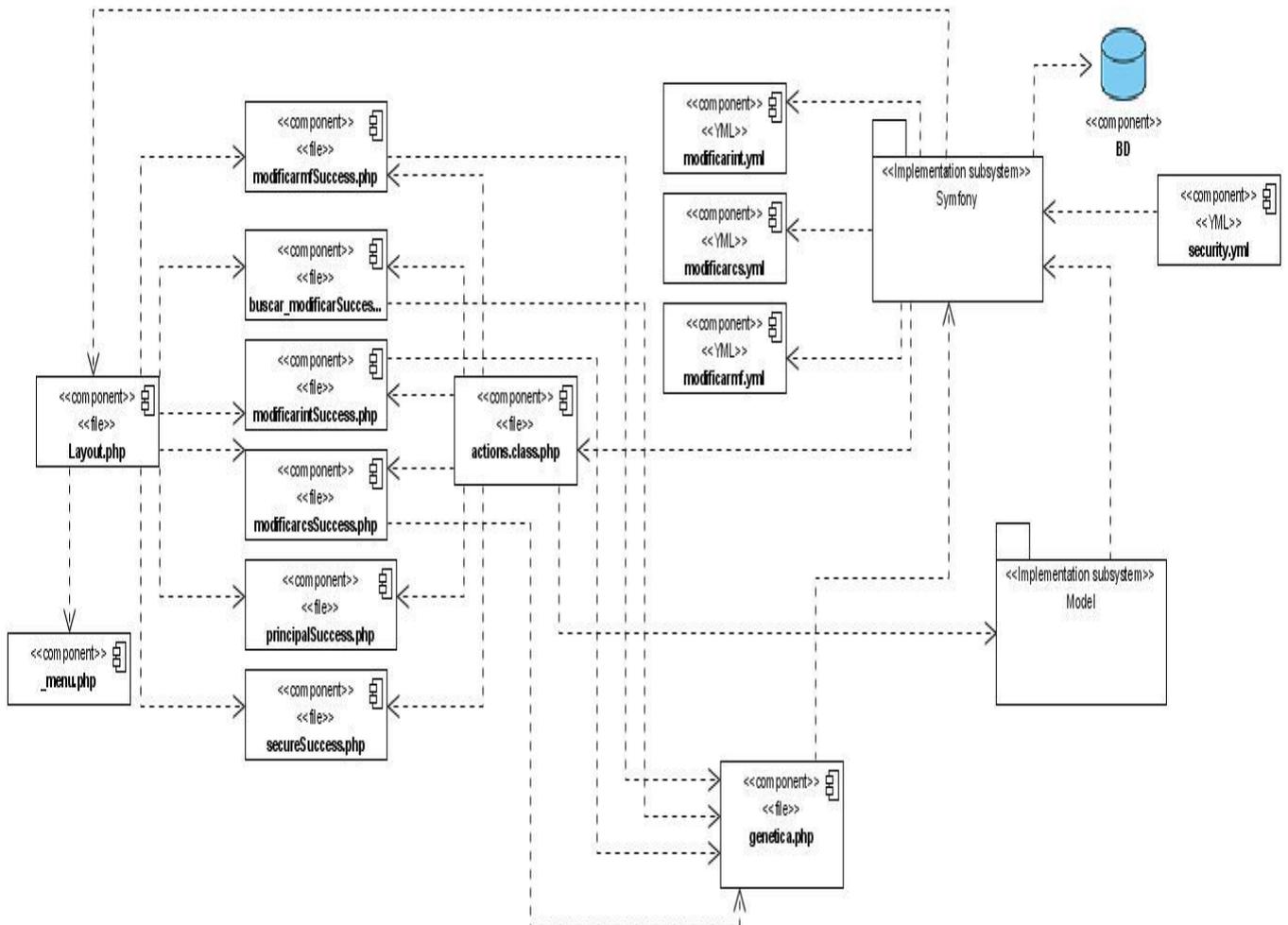


Figura 14 Diagrama de componente CU modificar Datos del paciente

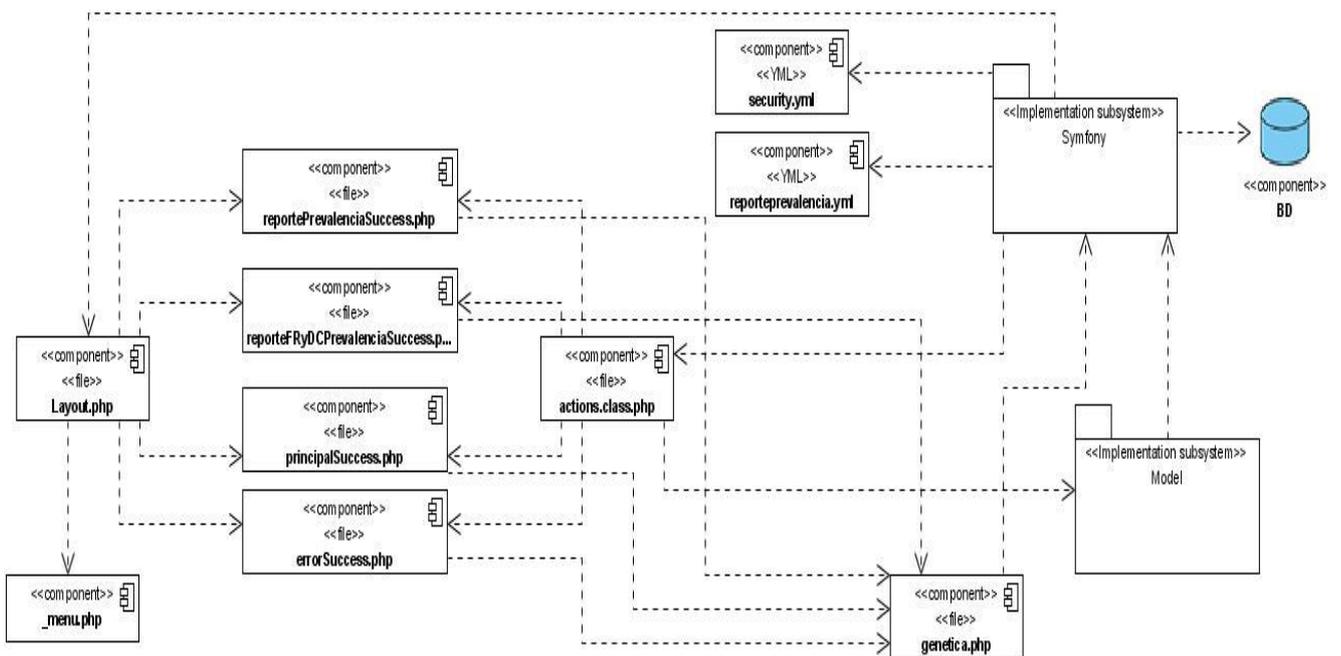


Figura 15 Diagrama de componente CU Generar Reportes Estadísticos

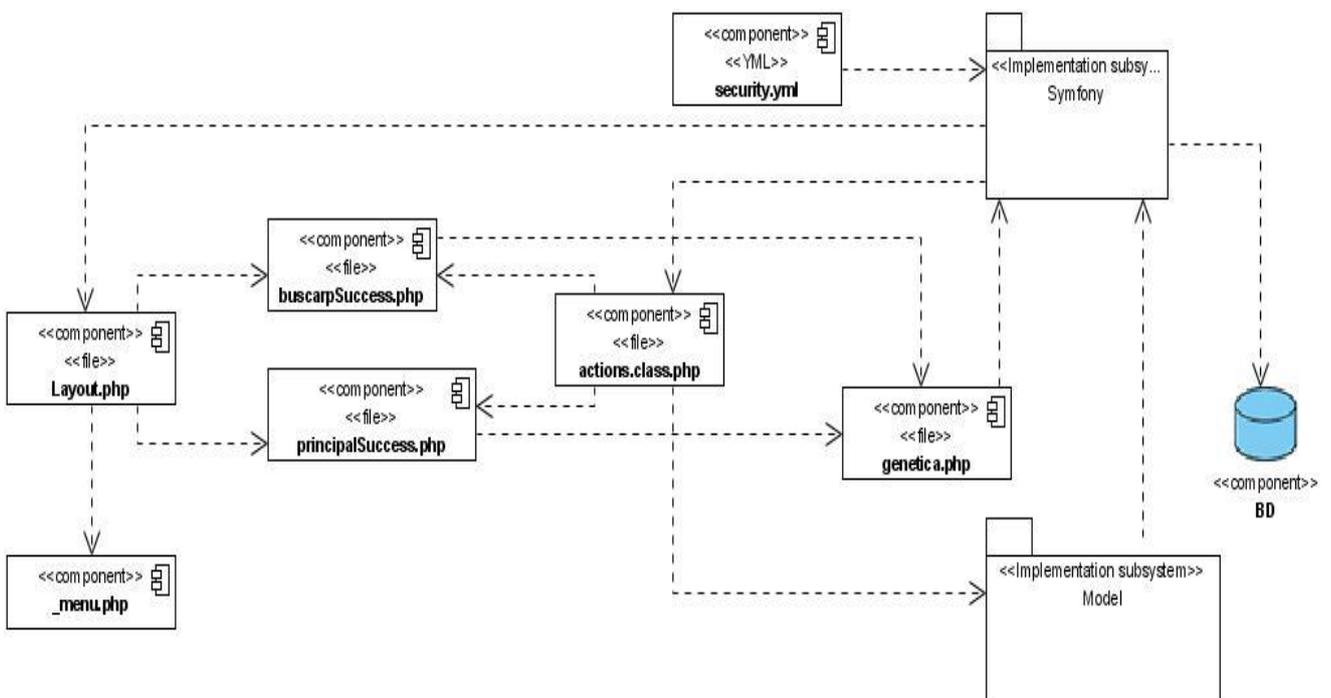


Figura 16 Diagrama de componente CU Buscar Paciente a Codificar

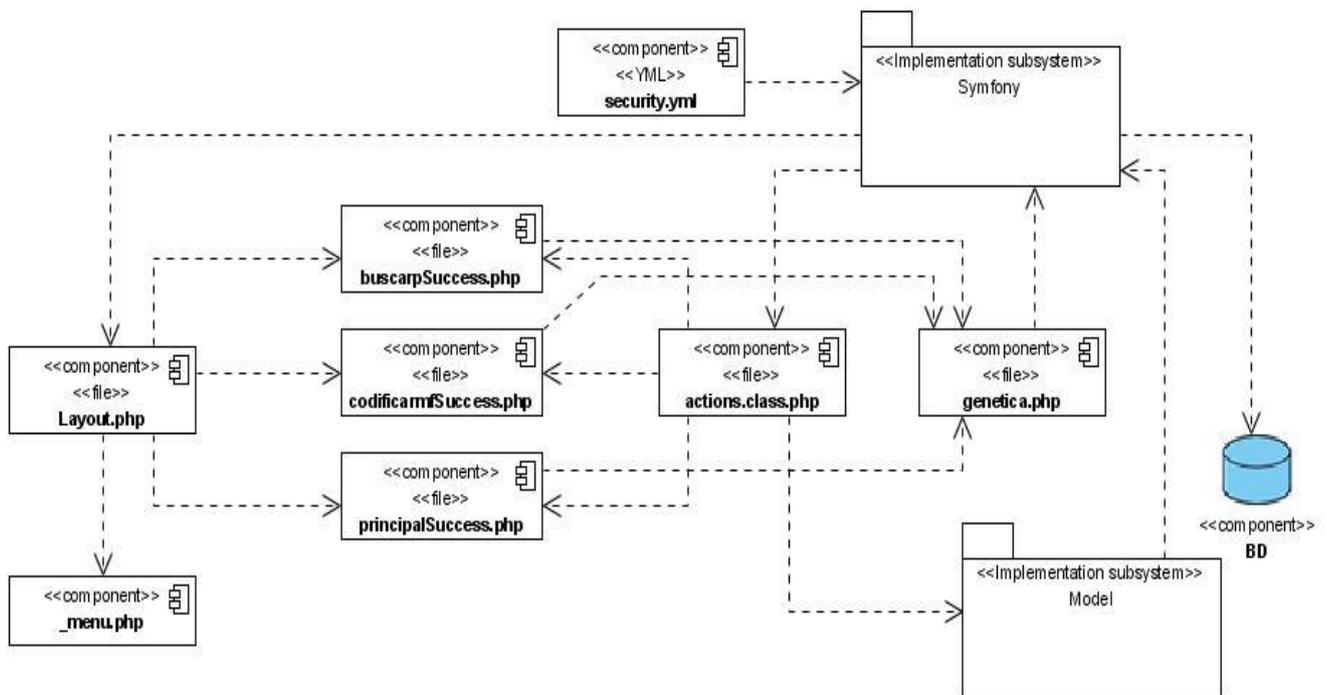


Figura 17 Diagrama de componente CU Codificar Malformación

3.2 Estándar de codificación

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. La manera de escribir código es individual y depende totalmente de cada programador, lo que origina problemas de mantenimiento y entendimiento del código por terceras personas.

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez.

Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento. Además, si se aplica de forma continuada un estándar de codificación bien definido, se

utilizan técnicas de programación apropiadas, y, posteriormente, se efectúan revisiones del código de rutinas, caben muchas posibilidades de que un proyecto de software se convierta en un sistema de software fácil de comprender y de mantener.

Utilizando un buen estilo de codificación el código constará de algunas cualidades. Ellas son:

- **Extensibilidad:** La facilidad con que se adapta el software a cambios de especificación. Un buen estilo de código fomenta programas que no sólo resuelven el problema, sino que también reflejan claramente la relación problema/solución. Esto tiene como efecto que muchos cambios simples en el problema reflejen de forma obvia los cambios a hacer en el programa.
- **Verificabilidad:** La facilidad con que pueden comprobarse propiedades de un sistema. Si el estilo de código hace obvia la estructura del programa, eso ayuda a verificar que el comportamiento sea el esperado.
- **Reparabilidad:** La posibilidad de corregir errores sin demasiado esfuerzo.
- **Capacidad de evolución:** La capacidad de adaptarse a nuevas necesidades.
- **Comprensibilidad:** La facilidad con que el programa puede ser comprendido.

Estilo de codificación para el SIGM

1. Todas las etiquetas php deben ser completas (`<?php ?>`)... no reducidas (`<? ?>`).
2. Todas las variables deberían ser inicializadas o, al menos, comprobada su existencia utilizando `isset ()` antes de ser utilizadas.
3. La sangría del texto debe ser siempre de 4 espacios. No utilices los tabulador (todos los editores no interpretan el tab de la misma manera).
4. Los nombres de las variables y funciones tienen que ser siempre fáciles de leer, procurando que sean palabras en minúsculas con significado claro. Si realmente necesita más de una palabra, póngalas juntas, poniendo la inicial de cada palabra en mayúscula siempre que no sea la primera, pero procure mantenerlas tan breves como sea posible. Utilice nombres en plural para arreglos o matrices de objetos. Ejemplos:
`$edad`
`$fechaNacimiento`
`$personas`
5. Los bloques de código siempre deben estar encerrados por llaves (incluso si solo constan de una línea).

6. Las cadenas tienen que ser definidas utilizando comillas simples siempre que sea posible, para obtener un mejor rendimiento.
7. Todas las funciones y clases deben estar comentadas. Los comentarios deben ser añadidos de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones o variables.
8. 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:

```
/** Esta función tiene como objetivo
 * @author Comentario sobre el nombre del autor
 * @param $edad edad de la persona
 * @param $sexo sexo de la persona
 */
private function MostrarDatos ($edad, $sexo)
{
    .....
}
```

3.3 Código fuente de los principales métodos y su descripción

En este epígrafe se realizará una breve descripción de algunos de los métodos de mayor peso en el desarrollo del módulo RECUMAC, debido a que los métodos de insertar, modificar y buscar son métodos de registro en la base de datos y obtención de ella, no dejando de ser importantes por esto, pero debido a que el instrumento a informatizar consta con una gran cantidad de campos es un poco difícil explicar detalladamente aquí todo su funcionamiento. Symfony con su ORM (Mapeo de objeto a bases de datos) Propel genera la mayor parte del código de acceso a datos proporcionando una abstracción al punto que permite que los implementadores no tengan que utilizar ni una sola consulta a SQL.

Se hará mención por tanto, a los métodos que por su particularidad sobresalen entre los demás, y entre ellos se encuentran los métodos que responden al caso de uso Codificar malformación que se encuentran en la clase mcActions.

```
/**Esta función tiene como objetivo codificar los pacientes malformados
 * @author Leidy Gonzalez, Daniela Hernández
 */
public function executeCodificarmf()
{
    if ($this->getRequest()->getMethod() != sfRequest::POST)
    {
        //llamada al método que carga la descripción del paciente a codificar
        $this->DatosPaginaCodificar();
    }
    else
    {
        // llamada al método que codifica la malformacion
        $this->Codificar_malformacion();
        $this->redirect('mc/buscarp');
    }
}
```

Este método es el execute correspondiente a la plantilla codificarmfSuccess, dentro de él se hace una llamada al método DatosPaginaCodificar que es el encargado de cargar los datos del paciente, específicamente la descripción de la malformación que es el dato necesario para que el genetista pueda codificar la malformación de acuerdo a la descripción, además se llama al método Codificar_malformación que es el encargado de asignarle a cada paciente los códigos de las malformaciones seleccionados por el usuario.

```
/**Este método privado tiene como objetivo cargar los datos necesarios en el template codificarmf
 * @author Leidy Gonzalez , Daniela Hernández
```

```

*/
private function DatosPáginaCodificar()
{
    //Consulta para cargar la descripcion de un paciente

    $this->id_planilla = $this->getRequestParameter('id_planilla');
    $c= new Criteria();
    $c->add(McPersonaRecumacPeer::ID_PLANILLA,$this->id_planilla);
    $persona = McPersonaRecumacPeer::doSelectOne($c);
    $this->per=$persona->getIdPersona();
    $c= new Criteria();
    $c->add(McPersonaRecumacPeer::ID_PLANILLA,$this->id_planilla);
    $persona = McPersonaRecumacPeer::doSelectOne($c);
    $this->per=$persona->getIdPersona();
    $c = new Criteria();
    $c->add(McPacienteMalformadoPeer::ID_PERSONA,$persona-
>getIdPersona());
    $pac_mal= McPacienteMalformadoPeer::doSelectOne($c);

    $this->descripcion = strip_tags($pac_mal->getDescripcion());
    //FIN de la consulta
    //Consulta para mostrar las descripciones de las malformaciones agrupadas
    $c = new Criteria();
    $c-
>addAscendingOrderByColumn(EgMcMalformacionesCongenPeer::DESCRIPCION);
    $c->addGroupByColumn(EgMcMalformacionesCongenPeer::DESCRIPCION);//
    $this->opcion =EgMcMalformacionesCongenPeer::doSelect($c);
    //Consulta para mostrar los codigos dado un tipo de malformacion
    $this->cod=EgMcMalformacionesCongenPeer::doSelect(new Criteria());
    //Consulta del nomenclador mc_clasificacion_malformacion
    $this->result = McClasificacionMalformacionPeer::doSelect(new Criteria());

```

```
        //Consulta para que me devuelva todos los tipos de malformaciones de la tabla
        mc_simple_compuesta
        $this->simple = McSimpleCompuestaPeer::doSelect(new Criterias());
    }
```

El método privado DatosPáginaCodificar() es invocado en el executeCodificarmf() para cargar en la plantilla correspondiente a dicho execute los datos necesarios para proceder a codificar la malformación, dígame por ejemplo cargar la descripción de la malformación, insertada en el caso de uso insertar datos complementarios del paciente, este campo se carga en la plantilla llamando a la variable descripción debido a que en esta clase (mcActions) la variable es global y por lo tanto se puede llamar en la plantilla sin ninguna dificultad. Las demás variables globales declaradas en el método son utilizadas en la plantilla para cargar todas las clasificaciones de los tipos de malformaciones, entiéndase tipo de malformación, descripción de la misma, entre otros.

```
/** Esta funcion privada es para codificar a los pacientes
 * @author Leidy Gonzalez , Daniela Hernández
 */
private function Codificar_malformación()
{
    $codigos= $this->getRequestParameter('codigos_malf');
    for($i=0;$i< count($codigos);$i++)
    {
        $codificar = new McCodigoMalformacionPorPaciente();
        $codificar->setIdPersona($this->getRequestParameter('id_persona'));
        $codificar->setCodigo($codigos[$i]);
        $codificar->setIdTipo($this->getRequestParameter('clasif'));
        $tipo=$this->getRequestParameter('tipo');
        if($prueba!=0)
        {
            $codificar->setIdTipos($tipo)!=0?$tipo:null;
        }
    }
}
```

```
        $ultimo_valor = $this->getRequestParameter('simple_ultimo');
        if($ultimo_valor !='0')
        {
            $codificar->setIldtipom($this->getRequestParameter('simple_ultimo')!=0?$this->getRequestParameter('simple_ultimo'):null);
        }
        $codificar->save();
    }
}
```

El método privado Codificar_malformación es el que se llama en el execute Codificarmf para responder a la necesidad de codificar al paciente que presenta un defecto congénito, este método es un clásico insertar donde se insertan en la tabla Mc_código_malformación_por_paciente todos los códigos de las malformaciones pertenecientes al paciente en cuestión, seleccionadas previamente por el usuario.

```
/** Esta página es para calcular el reporte de Prevalencia al nacimiento
 * @author Leidy Gonzalez y Daniela Hernández
 */
public function executeReporteprevalencia()
{
    //Consulta para mostrar los codigos dado un tipo de malformacion
    $this->cod=EgMcMalformacionesCongenPeer::doSelect(new Criteria());
    $this->valor = "";

    if($this->getRequest()->getMethod()== sfRequest::POST)
    {
        $this->valor = $this->Prevalencia();
    }
}
```

```
        return sfView::SUCCESS;
    }
```

Este método es el execute correspondiente a la plantilla `reportePrevalenciaSuccess` que es la encargada de mostrar el valor del reporte así como introducir los datos necesarios para el mismo. Dentro de este método se llama al método `Prevalencia` que es el que calcula la fórmula matemática que da respuesta a este reporte, el valor de este método se guarda en una variable global para que posteriormente sea llamada en la plantilla y así mostrar el valor numérico del reporte.

3.4 Validación a nivel de desarrollador

Los lenguajes de programación interpretado fueron diseñados para ser ejecutados por medio de un interprete, en contraste con los lenguajes de compilación, el lenguaje de programación utilizado en el presente trabajo de diploma es PHP 5.0 con el framework orientado a objeto Symfony y precisamente PHP 5.0 es un lenguaje interpretado, por lo que es muy difícil poder debuggear cualquier aplicación web que es desarrollada con este lenguaje de programación.

Debido a que los desarrolladores no pueden debuggear el código de sus clases para ir realizando pruebas, es muy recomendable utilizar la función “echo” de PHP para que muestre en el navegador los mensajes pertinentes y así el desarrollador puede ir probando los métodos que va implementando. Ejemplo del uso de la función “echo” para mostrar el resultado del reporte de factores de riesgo y defectos congénitos.

```
/* * Esta pagina es para calcular el reporte de factores de riesgo y defectos congénitos
 * @author Leidy González, Daniela Hernández
 */
public function executeReporteFRyDC()
{
    //Consulta para mostrar los códigos dado un tipo de malformación
    $this->cod=EgMcMalformacionesCongenPeer::doSelect(new Criteria());
    $this->valor ="";
    if($this->getRequest()->getMethod()== sfRequest::POST){
```

```
        echo $this->valor = $this->FCyDC();  
    return sfView::SUCCESS;  
}
```

El resultado del “echo” en la función anterior se muestra en la siguiente figura.

1.625

Warning: Cannot modify header information - headers already sent by (output started at E:\Intalador_Wamp\wamp\Proyecto\apps\genetica\modules\mc\actions\actions.class.php:3937) in E:\Intalador_Wamp\wamp\symfony\lib\response\sfWebResponse.class.php on line 264

Warning: Cannot modify header information - headers already sent by (output started at E:\Intalador_Wamp\wamp\Proyecto\apps\genetica\modules\mc\actions\actions.class.php:3937) in E:\Intalador_Wamp\wamp\symfony\lib\response\sfWebResponse.class.php on line 274

The screenshot displays the SIGMédica web application interface. At the top, there is a navigation bar with links for 'Acerca de', 'Ayuda', and 'Cerrar Sesión'. The main header area includes the SIGMédica logo (a blue DNA helix) and the text 'SISTEMA INFORMÁTICO DE GENÉTICA MÉDICA'. A welcome message reads 'Bienvenido nacional al Sistema Informático de la Red Nacional de Genética Médica'. User information is shown as 'Usuario: lbernal', 'Derechos: Administrador', and 'Nivel: Nacional'. Below the header is a menu with 'INICIO', 'MAPA DE SITIO', 'ARBOGEN', 'SISalud', and 'Módulos SIGM'. The main content area is titled 'Malformaciones - Factores de Riesgo y Defectos Congénitos'. It features a sidebar with 'Datos Primarios' and 'Historia Clínica Genética' sections. The main form area is titled 'Datos del reporte de Factores de Riesgo y Defectos Congénitos' and contains two dropdown menus: 'Tipos de malformación' (set to '<< Todas las malformaciones >>') and 'Factores de Riesgo' (set to '<<Seleccione>>'). A 'Calcular X^2 de MH' button is located below the second dropdown. A red asterisk warning at the bottom states: '(*) Los campos señalados son de entrada obligatoria'.

Otra forma de verificar la consistencia de los datos de la aplicación es mediante la inserción de datos erróneos para comprobar cuán riguroso es el sistema con el tema de la consistencia de datos. Esto podría considerarse como un paso de avance para el flujo de trabajo de prueba.

Codificación de la malformación

↓ Campo obligatorio ↓

Clasificación << Seleccione >> ↓*

↓ Campo obligatorio ↓

Malformación << Seleccione >> ↓*

Código

↓ Campo obligatorio ↓

Clasificación << Seleccione >> ↓*

<< Seleccione >> ↓

↓ Campo obligatorio ↓

<< Seleccione >> ↓*

<< Seleccione >> ↓

(*) Los campos señalados son de entrada obligatoria

Aceptar Cancelar

Figura 18 Mensajes de validación

Datos de la madre

Nombre Rosa

1er Apellido ↓ Solo letras ↓ Gomez3

2do Apellido ↓ Solo letras ↓ P455

Color de la Piel <<Seleccione>> ↓

Edad ↓ Solo numeros ↓ ertytyt

HC

Tipo de parto <<Seleccione>> ↓

Ocupación <<Seleccione>> ↓

Presente <<Seleccione>> ↓

Provincia Pinar del Río

Municipio Los Palacios

Consejo Popular Los Palacios Norte ↓*

Hospital ↓ Campo obligatorio ↓ <<Seleccione>> ↓*

A.Salud <<Seleccione>> ↓

F.U.M

Exámenes <<Seleccione>> ↓ Resultado ----- ↓

Exámenes <<Seleccione>> ↓ Resultado ----- ↓

Exámenes <<Seleccione>> ↓ Resultado ----- ↓

Figura 19 Mensajes de validación

3.5 Interfaces de la aplicación

Con la ayuda de la pautas de diseño y encaminando los pasos a lograr interfaces agradables, sencillas y atractivas a la vista del cliente, se obtiene las siguientes interfaces de la aplicación.

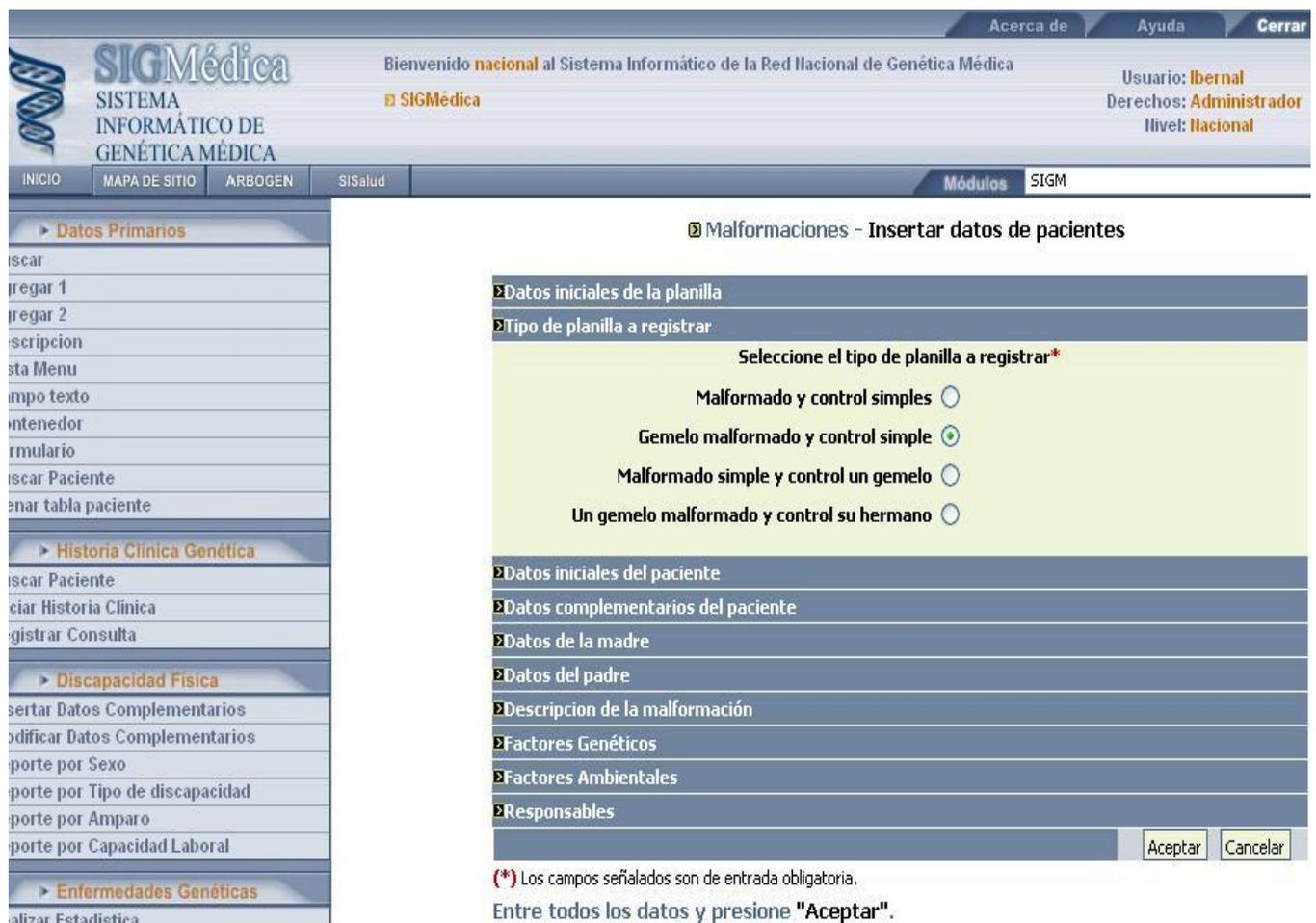


Figura 20 Interfaz correspondiente al CU Insertar Datos Complementarios del Paciente

The screenshot shows the SIGMédica web application interface. The header includes the logo, the text 'SISTEMA INFORMÁTICO DE GENÉTICA MÉDICA', and a welcome message for user 'nacional'. The user's role is 'Administrador' and the level is 'Nacional'. The main menu on the left lists various options under 'Datos Primarios', 'Historia Clínica Genética', and 'Discapacidad Física'. The main content area is titled 'Malformaciones - Modificar datos de pacientes' and contains several sections:

- Datos iniciales de la planilla:** Sistema de monitorización, No.IDENT: 9001640.
- Datos iniciales del paciente:** Sexo: Femenino, Nombre: Mili, Peso: 58 g, Long.Corp: 1.54 cm, Fecha de Nacimiento: 12/06/1990.
- Datos complementarios del paciente:** Monovitelinos (radio button), Bivitelinos (radio button checked), Circ.Cefalica: 0 cm, Circ.Torac: cm, Evidencia Diagnóstica: Radiológica, Nacimiento Multiple: Si (radio button), No (radio button checked), Nacimiento Simple: Si (radio button checked), No (radio button), Estado al nacer: Vivo (radio button checked), Muerto (radio button), Estado al alta: Vivo (radio button checked), Muerto (radio button).
- Datos de la madre:** (Section header only).

Figura 21 Interfaz correspondiente al CU Modificar Datos del Paciente

The screenshot shows the SIGMédica web application interface for 'Factores de Riesgo y Defectos Congénitos'. The header and main menu are identical to Figure 21. The main content area is titled 'Malformaciones - Factores de Riesgo y Defectos Congénitos' and contains a section 'Datos del reporte de Factores de Riesgo y Defectos Congénitos' with the following fields:

- Tipos de malformación: << Todas las malformaciones >>
- Factores de Riesgo: Oligoamnios *
- Calcular X² de MH: 1.625

 A red asterisk note at the bottom states: '(*) Los campos señalados son de entrada obligatoria.'

Figura 22 Interfaz correspondiente al CU Generar Reporte Estadísticos (Reporte Factor de Riesgo y Defecto Congénito)

SIGMédica
SISTEMA INFORMÁTICO DE GENÉTICA MÉDICA

Bienvenido **nacional** al Sistema Informático de la Red Nacional de Genética Médica
SIGMédica

Usuario: **Ibernal**
Derechos: **Administrador**
Nivel: **Nacional**

INICIO MAPA DE SITIO ARBOGEN SISalud Módulos SIGM

Datos Primarios

- Buscar
- Agregar 1
- Agregar 2
- Descripción
- Lista Menu
- Campo texto
- Contenedor
- Formulario
- Buscar Paciente
- Llenar tabla paciente

Historia Clínica Genética

- Buscar Paciente
- Iniciar Historia Clínica
- Registrar Consulta

Discapacidad Física

- Insertar Datos Complementarios
- Modificar Datos Complementarios
- Reporte por Sexo
- Reporte por Tipo de discapacidad
- Reporte por Amparo
- Reporte por Capacidad Laboral

Malformaciones - Prevalencia al Nacimiento

Datos de la prevalencia al nacimiento

Datos del Reporte

Fecha de inicio: 19/06/1997 *

Fecha de fin: 19/06/2008 *

Total de nacimientos: 2 *

Tipos de malformación: << Todas las Malformaciones >>

Calcular prevalencia: **6000**

(**) Los campos señalados son de entrada obligatoria.
Entre todos los datos y presione "Calcular Prevalencia".

Figura 23 Interfaz correspondiente al CU Generar Reporte Estadístico (Reporte Prevalencia al Nacimiento)

SIGMédica
SISTEMA INFORMÁTICO DE GENÉTICA MÉDICA

Bienvenido **nacional** al Sistema Informático de la Red Nacional de Genética Médica
SIGMédica

Usuario: **Ibernal**
Derechos: **Administrador**
Nivel: **Nacional**

INICIO MAPA DE SITIO ARBOGEN SISalud Módulos SIGM

Datos Primarios

- Buscar
- Agregar 1
- Agregar 2
- Descripción
- Lista Menu
- Campo texto
- Contenedor
- Formulario
- Buscar Paciente
- Llenar tabla paciente

Historia Clínica Genética

- Buscar Paciente
- Iniciar Historia Clínica
- Registrar Consulta

Discapacidad Física

- Insertar Datos Complementarios
- Modificar Datos Complementarios
- Reporte por Sexo
- Reporte por Tipo de discapacidad
- Reporte por Amparo
- Reporte por Capacidad Laboral

Malformaciones - Codificar Malformación

Descripción de la malformación

Descripción de la malformación
ojos, chinos, oído deforme, cabeza grande

Codificación de la malformación

Clasificación: << Seleccione >> *

Malformación: << Seleccione >> *

Código:

Clasificación: << Seleccione >> * << Seleccione >> *

<< Seleccione >> << Seleccione >>

(**) Los campos señalados son de entrada obligatoria. Aceptar Cancelar

Figura 24 Interfaz correspondiente al CU Codificar Malformación.

The screenshot shows the SIGMédica web application interface. The header includes the logo, the text 'SISTEMA INFORMÁTICO DE GENÉTICA MÉDICA', and a welcome message: 'Bienvenido nacional al Sistema Informático de la Red Nacional de Genética Médica'. User information is displayed as 'Usuario: lbernal', 'Derechos: Administrador', and 'Nivel: Nacional'. The navigation menu on the left includes 'Datos Primarios', 'Historia Clínica Genética', and 'Discapacidad Física'. The main content area is titled 'Malformaciones - Buscar Pacientes a Codificar' and contains a search form with fields for 'Nombre del Paciente' and 'Código de la planilla', a 'Buscar' button, and a table of search results.

Resultados de Búsqueda	
Código de la Planilla	Nombre del Paciente
8601720	Luisa H G
0701719	
8601211	mirelys benitez perez
0801231	mairene c c
9001640	Millier Edg

Figura 25 Interfaz correspondiente al CU Buscar Paciente a Codificar.

Conclusiones

La representación física de cada uno de los elementos que se muestran en el diagrama de componentes generado como artefacto en el modelo de implementación permite una mejor comprensión y gestión de los mismos, así como la explicación de los estándares de codificación hace posible un mejor entendimiento por parte de terceras personas del código de la aplicación, algo muy importante a tener en cuenta para la realización de cambios o mejoras que en un futuro puedan surgir. Con la visualización de las pantallas de la aplicación se logra suministrar más información sobre la aplicación informática que se ha desarrollado. Se explican además algunos de los principales métodos obtenidos durante el desarrollo de la aplicación, todo con el fin de lograr una mayor comprensión de este trabajo.

Conclusiones Generales

Durante la realización de este trabajo se identificó los aspectos positivos y las principales deficiencias que debían ser corregidas en el presente trabajo. La elaboración del diseño con la aplicación de los patrones permitió mejorar el proceso de codificación.

El producto funcional obtenido a partir de la implementación de las clases del diseño permitió integrar el módulo del RECUMAC al SIGM, dando solución al problema científico , poniendo al alcance de los genetistas una valiosa herramienta que contribuirá a lograr una correcta gestión de toda la información que resulta de interés para los especialistas en la Genética de todo el país, una vez que se publique en la red de salud de Cuba.

Recomendaciones

Teniendo en cuenta la investigación del trabajo se recomienda:

- Realizar las pruebas exploratorias, así como las pruebas de aceptación.

- Añadir nuevas funcionalidades al sistema que le permita a los especialistas del RECUMAC aumentar la calidad de sus investigaciones científicas.

Referencias Bibliográficas

1. **Rodriguez Perdomo, Yuleisy Y. L.** (2007). <http://bvs.sld.cu>. Retrieved enero 12, 2008, from <http://bvs.sld.cu>:
http://bvs.sld.cu/revistas/mciego/vol13_supl2_07/articulos/a5_v13_supl207.htm
2. **Delgado Ramos, Ariel y Vidal Ledo, María.** Informática en la salud pública cubana. *Rev Cubana Salud Pública* . [En línea] 2006. [Citado el: 13 de noviembre de 2008.]
http://bvs.sld.cu/revistas/spu/vol32_3_06/spu15306.htm..
3. **Blanco, Castilla Roberto Carlos.** *Sistema Informatizado para la Gestión del Registro Cubano de Malformaciones Congénitas.* 2007.
4. **Thaureaux, Dr. Denis Derivet, Hernández, Ing. Mirna Cabrera and Ramos, Dr. Ariel Delgado.** *Revista Cubana de Informatica. Revista Cubana de Informatica.* [Online] 2007. [Cited: Febrero 12, 2008.]
http://www.cecarn.sld.cu/pages/rcim/revista_14/articulos_hm/aplicacion.htm#t.
ISSN:1684- 1859.
5. **Báez, Vargas Dr. Francisco Mauricio y Suárez, Heredia Dr. Alvaro.** 2005.
6. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* [ed.] Felix Varela. 2004.
7. <http://www.eurocat.ulster.ac.uk/pubdata/> . <http://www.eurocat.ulster.ac.uk>. [Online] [Cited: abril 12, 2008.] <http://www.eurocat.ulster.ac.uk/pubdata/> .
8. **Desconocido.** http://www.cemic.edu.ar/investigacion/inv_unidadesyprog_eclamc.asp.
<http://www.cemic.edu.ar/>.
9. (n.d.). Retrieved abril 22, 2008, from <http://www.mysql.com/>:
http://www.mysql.com/common/pages/download_access_denied.html
10. **Dra. Francisca Alonso Lotti, Dra Alina Ferreiro.** *Revista Cubana de Pediatría. Revista Cubana de Pediatría.* [Online] abril- junio 1998. [Cited: febrero 15, 2008.]
http://scielo.sld.cu/scielo.php?pid=S0034-75311998000200002&script=sci_arttext. ISSN 0034-7531.
11. *Introducción a la ingeniería de software. Autores, Colectivo de.* UCI : s.n., 2007-2008. **Cesar, Tardaguila.** [En línea] 6 de Julio de 2007. [Citado el: 10 de Enero de 2008.] <http://www.designation.net/es/archivos/003798.php>.
12. **Zamitiz, Roman Ing. Carlos Alberto.** *El lenguaje Unificado de Modelado(UML).* 2006.
13. **Hernández Jose Alberto.** *VersionCero.* [En línea] 2005. [Citado el: 14 de diciembre de 2007.]
<http://www.versioncero.com/noticia/210/visual-paradigm-for-uml>. **Hernández Jose Alberto.**

14. <http://dev.mysql.com>. <http://dev.mysql.com>. [Online] junio 03, 2007.
<http://dev.mysql.com/doc/refman/5.0/es/introduction.html>.
15. (n.d.). Retrieved febrero 5, 2008, from <http://httpd.apache.org>:
http://httpd.apache.org/docs/2.0/es/new_features_2_0.html
16. **Cabezas Granado, Luis Miguel (ANAYA MULTIMEDIA)**. *PHP 5*. España : s.n., 2004. ISBN: 8441517851..13. **García, Luis**. Sistema de control de versiones: SUBVERSION . *Sistema de control de versiones: SUBVERSION* . [Online] educación y observatorio tecnológico, enero 17, 2008.[Cited:enero25,2008.]
<http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=548>.
17. **Sanz, Laura Bermejo and Monreal, Enrique Gómez**. Eclipse como IDE. *Eclipse como IDE*. [Online] [Cited: abril 12, 2008.]
<http://kybele.escet.urjc.es/documentos/HC/Exposiciones/EclipseIDE.pdf>.
18. **Desconocido**. <http://indira-informatica.blogspot.com/2007/04/editores-web-quanta-plus.htm>
19. **Zaninotto, Fabien Potencier y François**. [En línea] 20 de diciembre de 2007. [Citado el: 12 de Enero de 2008.] <http://www.librosweb.es/symfony/>. www.bebesymas.com. www.bebesymas.com. [En línea] 06 de 12 de 2007. [Citado el: 07 de 05 de 2008.]
<http://www.bebesymas.com/2007/12/05-software-para-detectar-el-sindrome-de-down-en-el-primer-trimestre-del-embarazo>.
20. **Cabezas Granado, Luis Miguel (ANAYA MULTIMEDIA)**. *PHP 5*. España : s.n., 2004. ISBN: 8441517851..
21. uci, C. d. (2007-2008). FT Análisis y Diseño. Modelo de Diseño. Habana : Ciudad Habana .
22. **Fabien Potencier, François Zaninotto**. *Symfony la guía definitiva*. s.l. : Apress, 2008. ISBN-13.
23. Patrones de diseño. **UCI, Colectivo de**. 2007-2008.
24. Diseño Orientado a Objetos Patrones *GoF*. **Soriano, Ing. Fernando**. 2008 : s.n., ----.
25. Alfonso Claro : <http://10.7.9.6:5901/default.aspx>
26. Conferencia 1 Continuación del FT Análisis de Diseño. De ISW2.[En línea] 2006-2007. UCI. La Habana, Cuba.
27. Conferencia 2 IS2 Flujo de Implementación. [En línea] 2005-2006. Universidad de las Ciencias Informáticas. La Habana, Cuba..

Bibliografía

1. (n.d.). Retrieved abril 22, 2008, from <http://www.mysql.com/>:
http://www.mysql.com/common/pages/download_access_denied.html
2. (n.d.). Retrieved febrero 5, 2008, from <http://httpd.apache.org/>:
http://httpd.apache.org/docs/2.0/es/new_features_2_0.html
3. Alfonso Claro : <http://10.7.9.6:5901/default.aspx>
4. **Alonso Lotti, Dra. Francisca, Dra Alina Ferreiro.** Revista Cubana de Pediatría. *Revista Cubana de Pediatría.* [Online] abril- junio 1998. [Cited: febrero 15, 2008.]
http://scielo.sld.cu/scielo.php?pid=S0034-75311998000200002&script=sci_arttext. ISSN 0034-7531.
5. **Báez, Vargas Dr. Francisco Mauricio y Suárez, Heredia Dr. Alvaro.** 2005.
6. **Baltaxe Erik, Ignacio Zarante.** Prevalencia de malformaciones cardíacas congénitas en 44,985 nacimientos en Colombia. Colombia : s.n., 2006.
7. **Blanco, Castilla Roberto Carlos.** *Sistema Informatizado para la Gestión del Registro Cubano de Malformaciones Congénitas.* 2007.
8. **Cabezas Granado, Luis Miguel (ANAYA MULTIMEDIA).** *PHP 5.* españa : s.n., 2004. ISBN: 8441517851..
9. **Cabezas Granado, Luis Miguel (ANAYA MULTIMEDIA).** *PHP 5.* españa : s.n., 2004. ISBN: 8441517851..13. **García, Luis.** Sistema de control de versiones: SUBVERSION . *Sistema de control de versiones: SUBVERSION* . [Online] educacion y observatorio tecnologico, enero 17, 2008.[Cited:enero25,2008.]
<http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=548>.
10. **Cabezas Granado, Luis Miguel (ANAYA MULTIMEDIA).** *PHP 5.* españa : s.n., 2004. ISBN: 8441517851..
11. *Comportamiento clínico-epidemiológico de los defectos congénitos en la Ciudad de La Habana.* **Lic. María Emilia Ferrero Oteiza, Dra. María Teresa Pérez Mateo, Dr. Roberto Álvarez Fumero y Dra. Lidia Rodríguez Peña.** n.1, Ciudad de la Habana : s.n., 2005, Vol. v.77 . ISSN 0034-7531 .
12. Conferencia 1 Continuación del FT Análisis de Diseño. De ISW2.[En línea] 2006-2007. UCI. La Habana, Cuba. Conferencia 2 IS2 Flujo de Implementación. [En línea] 2005-2006. Universidad de las Ciencias Informáticas. La Habana, Cuba..
13. **corporativo, Autor.** Dos subvenciones del Gobierno regional en el ámbito de Salud Pública. *Dos subvenciones del Gobierno regional en el ámbito de Salud Pública.* [Online] 05 18, 2006. [Cited: enero 4, 2008.]
http://www.sanidad.jcyl.es/sanidad/cm/institucion/tkContent?pgseed=1175642750376&idContent=127679&locale=es_ES&textOnly=false.

14. **Delgado Ramos, Ariel y Vidal Ledo, María.** Informática en la salud pública cubana. *Rev Cubana Salud Pública* . [En línea] 2006. [Citado el: 13 de noviembre de 2008.]
http://bvs.sld.cu/revistas/spu/vol32_3_06/spu15306.htm..
15. **Desconocido.** <http://indira-informatica.blogspot.com/2007/04/editores-web-quanta-plus.htm>
16. **Desconocido.** http://www.cemic.edu.ar/investigacion/inv_unidadesyprog_eclamc.asp.
<http://www.cemic.edu.ar/>.
17. Diseño Orientado a Objetos Patrones GoF. **Soriano, Ing. Fernando.** 2008 : s.n., -----.
18. **Dra. Francisca Alonso Lotti, Dra Alina Ferreiro.** Revista Cubana de Pediatría. *Revista Cubana de Pediatría*. [Online] abril- junio 1998. [Cited: febrero 15, 2008.]
http://scielo.sld.cu/scielo.php?pid=S0034-75311998000200002&script=sci_arttext. ISSN 0034-7531.
19. *elbebe.com.* *elbebe.com*. [En línea] 19 de enero de 2008. [Citado el: 6 de febrero de 2008.]
<http://www.elbebe.com/index.php/es/salud/malformaciones-congenitas>.
20. **Eugenia, Dra. Maria, Ramírez, Dr. Rodrigo y Herrera, Dr. Julio Nazer.** *Malformacione congénitas Diagnostico y manejo neonatal*. s.l. : Editorial Universitaria .
21. **Fabien Potencier, François Zaninotto.** *Symfony la guía definitiva*. s.l. : Apress, 2008. ISBN-13.
22. **García, Luis.** Sistema de control de versiones: SUBVERSION . *Sistema de control de versiones: SUBVERSION* . [Online] educacion y observatorio tecnologico, enero 17, 2008. [Cited: enero 25, 2008.]
<http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=548>.
23. **Hernández Jose Alberto.** VersionCero. [En línea] 2005. [Citado el: 14 de diciembre de 2007.]
<http://www.versioncero.com/noticia/210/visual-paradigm-for-um>!. **Hernández Jose Alberto.**
24. <http://dev.mysql.com>. <http://dev.mysql.com>. [Online] junio 03, 2007.
<http://dev.mysql.com/doc/refman/5.0/es/introduction.html>.
http://www.cecam.sld.cu/pages/rcim/revista_14/articulos_hm/aplicacion.htm#t.
25. <http://www.eurocat.ulster.ac.uk/pubdata/> . <http://www.eurocat.ulster.ac.uk>. [Online] [Cited: abril 12, 2008.] <http://www.eurocat.ulster.ac.uk/pubdata/> .
26. Introducción a la ingeniería de software. **Autores, Colectivo de.** UCI : s.n., 2007-2008. **Cesar, Tardaguila.** [En línea] 6 de Julio de 2007. [Citado el: 10 de Enero de 2008.] <http://www.design-nation.net/es/archivos/003798.php>.
ISSN:1684- 1859.
27. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software*. [ed.] Felix Varela. 2004.
28. *Iarman, C.* (2004). *UML y Patrones* . Habana: Felix Varela.
29. *Medicina Fetal y Neonatología.* **Galána, Riaño, et al.** Número 04, Madrid : s.n., 200, Vol.Volumen 52 .

30. Patrones de diseño. **UCI, Colectivo de.** 2007-2008.
31. **Potencier Fabien, François Zaninotto.** *Symfony la guía definitiva.* s.l. : Apress, 2008. ISBN-13.
32. **Rodríguez Perdomo, Yuleisy Y. L.** (2007). <http://bvs.sld.cu>. Retrieved enero 12, 2008, from http://bvs.sld.cu: http://bvs.sld.cu/revistas/mciego/vol13_supl2_07/articulos/a5_v13_supl207.htm
33. **RUMBAUGH JAMES, GRADY BOOCH , JACOBSON IVAR.** *El lenguaje unificado de modelado.manual de referencia.* s.l. : Pearson Prentice Hall, 2003. ISBN: 8478290370.
34. **Sanz, Laura Bermejo and Monreal, Enrique Gómez.** Eclipse como IDE. *Eclipse como IDE.* [Online] [Cited: abril 12, 2008.] <http://kybele.escet.urjc.es/documentos/HC/Exposiciones/EclipseIDE.pdf>.
35. **Sanz, Laura Bermejo and Monreal, Enrique Gómez.** Eclipse como IDE. *Eclipse como IDE.* [Online] [Cited: abril 12, 2008.] <http://kybele.escet.urjc.es/documentos/HC/Exposiciones/EclipseIDE.pdf>.
36. **Solís, Dra. Lila Umaña.** Centro de Registro de Enfermedades Congénitas. *Centro de Registro de Enfermedades Congénitas.* [Online] 2005. [Cited: marzo 6, 2008.] http://www.inciensa.sa.cr/contenido/como_lo_hacemos/CNR/CREC/crec.htm.
37. **Suárez, Dr. Álvaro D. Heredia.** Malformaciones congénitas diagnosticadas por ultrasonido. *Malformaciones congénitas diagnosticadas por ultrasonido.* [Online] julio 4, 2005. [Cited: marzo 1, 2008.] <http://www.ilustrados.com/publicaciones/EEkpuAkEuukZxbMXHz.php>.
38. **Tapia, Dra. M.Aurora Canessa and Abarza, Rosa Gajardo.** Registro Regional de Malformaciones congénitas (RRMC). *Registro Regional de Malformaciones congénitas (RRMC).* [Online] [Cited: enero 7, 2008.] http://www.prematuros.cl/weboctubre05/malformaule/malformaciones_SSM.htm.
39. **Thaureaux, Dr. Denis Derivet, Hernández, Ing. Mirna Cabrera and Ramos, Dr. Ariel Delgado.** Revista Cubana de Informatica. *Revista Cubana de Informatica.* [Online] 2007. [Cited: Febrero 12, 2008.] http://www.cecama.sld.cu/pages/rcim/revista_14/articulos_hm/aplicacion.htm#t. ISSN:1684-1859.
40. **Thaureaux, Dr. Denis Derivet, Hernández, Ing. Mirna Cabrera and Ramos, Dr. Ariel Delgado.** Revista Cubana de Informatica. *Revista Cubana de Informatica.* [Online] 2007. [Cited: Febrero 12, 2008.]
41. uci, C. d. (2007-2008). FT Análisis y Diseño. Modelo de Diseño. Habana : Ciudad Habana .
42. **Zamitiz, Roman Ing.Carlos Alberto.** *El lenguaje Unificado de Modelado(UML).* 2006.
43. **Zaninotto, Fabien Potencier y François.** [En línea] 20 de diciembre de 2007. [Citado el: 12 de Enero de 2008.] <http://www.librosweb.es/symfony/>. www.bebesymas.com. www.bebesymas.com. [En línea] 06 de 12 de 2007. [Citado el: 07 de 05 de 2008.] <http://www.bebesymas.com/2007/12/05-software-para-detectar-el-sindrome-de-down-en-el-primer-trimestre-del-embarazo>.

Anexos

Anexo # 1 Instrumento actual

MINISTERIO DE SALUD PUBLICA				REGISTRO CUBANO DE MALFORMACIONES CONGENITAS			
SISTEMA DE MONITORIZACION				No. IDENT. _____			
1-MALFORMADO		2-CONTROL		1-SIGUIENTE		2-NO SIGUIENTE	
RECIENTE NACIDO:				MADRE:			
HOSP. _____ FECHA NAC. ____/____/____				H.C.: _____ NOMBRE: _____			
NOMBRE: _____				DIRECCION: _____			
SEXO: 1-MASCULINO 2-FEMENINO 3-INTERSEXO				PROV.: _____ MUNIC.: _____ A. SALUD: _____			
NACIMIENTO: 1-SIMPLE 2-MULTIPLE No. ORDEN: _____				F.U.M. _____			
GEMELOS MONOVITELINOS _____ BIVITELINOS _____				T. PARTO: 1-EUTOC. 2-INSTR. 3-CESAREA 4-N.E.			
ESTADO AL NACER: 1-VIVO 2-MUERTO				PRESENT. 1-CEFAL 2-PODAL 3-TRANSV. 4-PELV. 5-N.E.			
ESTADO AL ALTA: 1-VIVO 2-MUERTO				A.F.P. _____ 1-SI 2-NO RESULT: 1-POS. 2-NEG.			
PESO: _____ g LONG. CORP.: _____ cm				ULTRASONIDO 1-SI 2-NO RESULT: 1-POS. 2-NEG.			
CIRC. CEFALICA: _____ cm CIRC. TORAC: _____ cm				CITOGENETICA 1-SI 2-NO RESULT: 1-POS. 2-NEG.			
EVIDENCIA DIAGNOSTICA: 1-CLIN. 2-RADIOL 3-QUIRURG.				INTERRUPCION POR D.P.N.: 1-SI 2-NO			
4-ANAT. PAT. 5-OTRA _____							
1er. APELLIDO		2do. APELLIDO		EDAD		COLOR DE LA PIEL	
MADRE						1-B. 2-N. 3-M. 4-O	
PADRE						1-B. 2-N. 3-M. 4-O	
DESCRIPCION DE LA MALFORMACION.							
FACTORES GENETICOS				FACTORES AMBIENTALES			
HERMANDAD				ANTI-CONCEPTIVO ORAL 1-SI 2-NO			
				TRIMESTRE 1 2 3			
PROD. (V,M,A)				FUMA: 1-SI 2-NO No. Cigarros día _____			
SEXO (M, F, In)				RAYOS X: Examen: _____ No. exp. _____			
				AM. ABORTO: CAUSA _____			
FECHA NAC.				AM. PARTO PRET.: CAUSA _____			
DIA				ANESTESIA: CAUSA _____			
MES				ALCOHOL: dosis prom/D _____			
AÑO				HIPERTERMIA: CAUSA _____			
EST. DO ACT. (V,M)				DROGAS _____			
No. ABORTOS PROVOCADOS: _____				ENF. AGUDAS: _____			
CONSANGUIN. PARENTAL: 1-SI 2-NO CUAL? _____				ENF. CRONICAS _____			
MALFORM. EN LA FAMILIA: 1-SI 2-NO				OLIGOAMNIOS: 1-SI 2-NO POLIHIDRAMNIOS: 1-SI 2-NO			
				MEDICAMENTOS TRIMESTRE: 1 2 3 Dosis/D Tiempo/tto			
CONFECCIONADO POR:				RESP. DEL REGISTRO:			

Anexo # 2 Formulas Usadas para los reportes de la aplicación

Reporte de Prevalencia al nacimiento = Total de nacidos con determinado defecto congénito /Total de nacidos en un hospital determinado*10000.

Reporte de Factores de riesgo y defectos congénitos = $((a*d - b*c) / (No*N1*Mo*M1))^{(N-1)}$.

Donde:

- a: Cantidad de pacientes malformados con un X factor de riesgo durante el embarazo de su madre.
- b: Cantidad de pacientes controles de a que sus madres tuvieron durante el embarazo un X factor de riesgo.
- c: Cantidad de pacientes malformados que sus madres no tuvieron ningún factor de riesgo durante el embarazo.
- d: Cantidad de pacientes controles de c que sus madres no tuvieron ningún factor de riesgo durante el embarazo.

No: Suma de a+b

N1: Suma de c+d

Mo: Suma de a+c

N: Suma de No+N1+Mo+M1

Anexo # 3 Descripción de los CU del sistema

Caso de Uso: Modificar Datos del Paciente.	
Propósito: Modificar datos de los pacientes pertenecientes al RECUMAC.	
Actores: Genetista Nacional(Inicia)	
Resumen: El caso de uso inicia cuando el genetista que puede ser nacional o provincial decide modificar los datos de un paciente, para ello primeramente debe buscar los datos del paciente a modificar en el sistema para luego realizarle los cambios deseados, finalizando el caso de uso con el almacenamiento de los datos en el sistema.	
Referencias: R1	
Precondiciones: Los usuarios deben estar registrados con los permisos pertinentes para poder acceder a esta parte del sistema y deben realizar las búsquedas de los pacientes.	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El genetista decide modificar datos del paciente, pinchando en el menú la opción de Modificar Datos del Paciente.	2. El sistema muestra la ventana de búsqueda de los datos de los pacientes. Ver caso de uso " Gestionar datos primarios" del modulo RCHCL en el flujo normal de los eventos
3. El genetista selecciona la opción "seleccionar" para ver los datos correspondientes a este paciente.	4. El sistema consulta el tipo de paciente seleccionado, que puede ser malformado, control o interrupción y redirecciona a la página correspondiente, mostrando sus datos.
5. El genetista realiza los cambios deseadas en la planilla del paciente.	6. Se verifica que los campos requeridos estén llenos y los demás estén con valores válidos.
	7. El sistema almacena los datos modificados y finaliza así el caso de uso.
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
Acción 5:	El genetista no introduce los datos a modificar correctamente por lo que el sistema muestra los mensajes de errores pertinentes, permitiendo que este los corrija.
Poscondiciones:	Se registran los datos modificados del paciente.

Caso de Uso: Insertar datos complementarios del paciente.	
Propósito: Insertar los datos complementarios de un paciente.	
Actores: Genetista Provincial (Inicia).	
Resumen: El caso de uso inicia cuando el genetista provincial decide insertar los datos complementarios de un paciente, para ello primeramente debe buscar por diferentes criterios los datos del paciente a insertar en el sistema, para luego insertarle los datos complementarios, finalizando el caso de uso con el almacenamiento de los datos en el sistema.	
Referencias: R2	
Precondiciones: Los usuarios deben estar registrados con los permisos pertinentes para poder acceder a esta parte del sistema y deben realizar las búsquedas de los pacientes.	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El genetista decide insertar datos complementarios del paciente (malformado y control simple, gemelos malformados, gemelo malformado y control simple, malformado y control un gemelo, gemelo malformado y control su hermano).	2. El sistema muestra la ventana de búsqueda de los datos de los pacientes. Ver caso de uso “Gestionar Datos Primarios” del modulo RCHCL en el flujo normal de los eventos
3. El genetista selecciona el paciente al cual se le van a insertar los datos complementarios.	4. El sistema muestra las diferentes clasificaciones para la inserción de los datos (malformado y control simple, gemelos malformados, gemelo malformado y control simple, malformado y control un gemelo, gemelo malformado y control su hermano)
5. El genetista selecciona el criterio deseado.	6. El sistema muestra los datos primarios ya introducidos anteriormente y los datos complementarios a llenar.
7. El genetista introduce los datos complementarios.	8. El sistema verifica que los campos requeridos estén llenos y los demás estén con valores válidos.
	9. El sistema almacena los datos insertados y finaliza así el caso de uso.
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
1. El genetista selecciona la opción insertar datos complementarios caso de interrupción.	2. El sistema muestra la ventana de búsqueda de los datos de los pacientes. Ver caso de uso “Gestionar Datos Primarios” del modulo RCHCL en el flujo alternativo de los eventos
Acción 4:	4. El sistema muestra los datos primarios ya introducidos anteriormente y los datos complementarios a llenar.
Acción 7:	El genetista no introduce los datos a insertar

	correctamente por lo que el sistema muestra los mensajes de errores pertinentes, permitiendo que este los corrija.
Poscondiciones:	Se registran los datos del paciente.

Caso de Uso: Generar reportes estadísticos.	
Propósito: Generar los reportes de Prevalencia al Nacimiento y Factores de Riesgo y Defectos Congénitos.	
Actores: Genetista Nacional(Inicia)	
Resumen: El caso de uso inicia cuando el genetista que puede ser provincial o nacional decide generar los reportes estadísticos del RECUMAC para ello debe seleccionar el tipo de reporte deseado, insertar los datos necesarios para este, y muestra el resultado al usuario del reporte finalizando así el caso de uso.	
Referencias: R3	
Precondiciones: Los usuarios deben estar registrados con los permisos pertinentes para poder acceder a esta parte del sistema y deben realizar las búsquedas de los pacientes.	

Flujo Normal de Eventos

Acción del Actor	Respuesta del Sistema
1. El genetista selecciona el reporte a obtener.	2. El sistema muestra la página del reporte correspondiente solicitando al usuario los datos necesarios para el mismo.
3. El genetista introduce los campos solicitados.	4. Se verifica que los campos requeridos estén llenos y los demás estén con valores válidos.
	5. El sistema muestra el resultado del reporte.

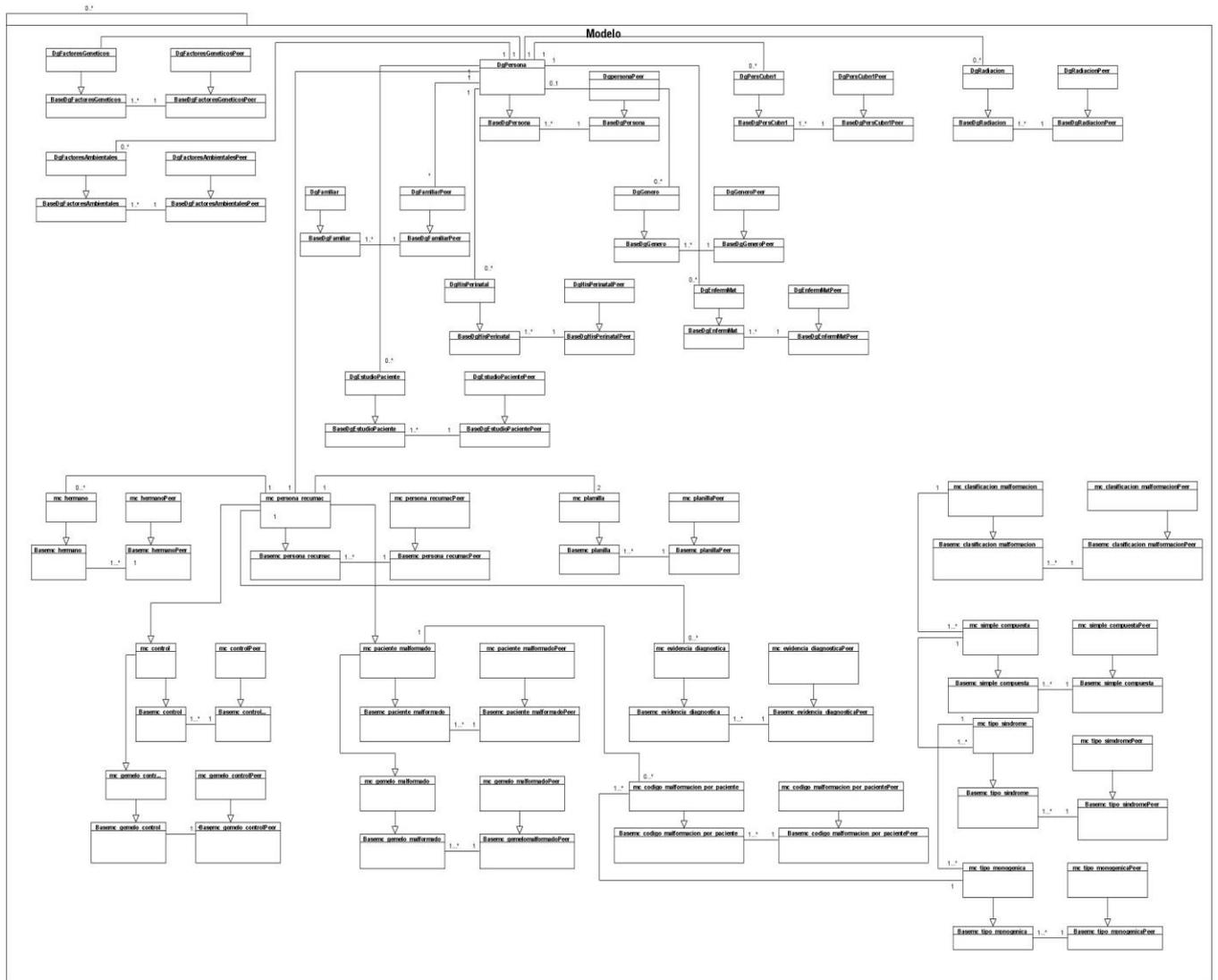
Flujos Alternos

Acción del Actor	Respuesta del Sistema
Acción 3:	El genetista no introduce los datos necesarios para efectuarse el reporte por lo que el sistema muestra los mensajes de errores pertinentes, permitiendo que este los corrija.
Poscondiciones:	-----

Caso de Uso: Codificar malformación.	
Propósito: Codificar los pacientes malformados e interrupciones según la descripción de la malformación que presenta el paciente.	
Actores: Genetista Nacional(Inicia)	
Resumen: El caso de uso inicia cuando se decide codificar los pacientes malformados e interrupción según el tipo de malformación, para ello inicialmente debe realizar la búsqueda de los pacientes no codificados, finalizando el caso de uso con la codificación de la malformación del paciente seleccionado.	
Referencias: R4	
Precondiciones: Los usuarios deben estar registrados con los permisos pertinentes para poder acceder a esta parte del sistema y deben realizar las búsquedas de los pacientes.	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El genetista decide codificar los pacientes no codificados.	2. El sistema muestra la interfaz de búsqueda de los pacientes malformados sin codificar. Ver descripción del caso de uso "Buscar paciente sin codificar"
	3. El sistema muestra los resultados de la búsqueda según criterios establecidos.
4. El genetista selecciona el paciente al cual se le desea codificar la malformación.	5. El sistema muestra la descripción de la malformación del paciente seleccionado.
6. El genetista realiza la codificación de la malformación.	7. Se verifica que los campos obligatorios estén llenos.
	8. El sistema guarda la información sobre el tipo de malformación del paciente, finalizando así el caso de uso.
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
Acción 3:	Los criterios de búsqueda introducidos por el genetista no muestran ningún resultado, por lo que el sistema muestra un mensaje que no hay resultados en la búsqueda finalizando así el caso de uso.
Acción 7:	El genetista introduce incorrectamente los datos necesarios para codificar la malformación del paciente por lo que el sistema muestra los mensajes de errores pertinentes, permitiendo que este los corrija.
Poscondiciones:	Se registran los datos de la codificación de la malformación.

Caso de Uso: Buscar paciente sin codificar (Incluido)	
Propósito: Buscar todos los pacientes malformados sin codificar.	
Actores: Genetista Nacional(Inicia)	
Resumen: El caso de uso inicia cuando el Genetista Nacional decide buscar los pacientes malformados para realizar la codificación de la malformación.	
Referencias: R4	
Precondiciones: Los usuarios deben estar registrados con los permisos pertinentes para poder acceder a esta parte del sistema y deben realizar las búsquedas de los pacientes.	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El Genetista accede a la opción buscar pacientes sin codificar.	2. El sistema muestra una interfaz de búsqueda por diferentes criterios.
3. El Genetista introduce los datos correspondientes para la búsqueda.	4. El sistema muestra una interfaz con los resultados de la búsqueda, finalizando así el caso de uso.
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
Acción 3:	Los criterios de búsqueda introducidos por el genetista no muestran ningún resultado, por lo que el sistema muestra un mensaje que no hay resultados en la búsqueda finalizando así el caso de uso.
Poscondiciones:	-----

Anexo # 4 Paquete de Modelo



Glosario de términos

Agente Teratogénicos: Cualquier principio o sustancia con la capacidad de producir malformaciones en un feto en desarrollo. Pueden ser: físicos, químicos o biológicos.

Alfafeto Proteína (AFP): Es una proteína que normalmente sólo se produce en el feto durante su desarrollo. Cuando aparece en adultos, puede servir como un marcador tumoral. Si aparecen niveles elevados de AFP en el líquido amniótico puede ser una indicación de un defecto en el desarrollo del bebé. Normalmente se ofrece consejo genético si una prueba rutinaria de AFP resulta positiva.

Apache: Es un software (libre) servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etcétera), Windows y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.

Citogenética: Rama de la genética que estudia los cromosomas, su estructura y su herencia.

Control: Niño sano del mismo sexo nacido en el mismo hospital después de un malformado.

Embrión o feto: Ser vivo en las primeras etapas de su desarrollo, desde la fecundación hasta que el organismo adquiere las características morfológicas de la especie.

Gen: es el conjunto de una secuencia determinada de nucleótidos de uno de los lados de la escalera del cromosoma referenciado.

Http: (HyperText Transmission Protocol) Protocolo para transferir archivos o documentos hipertexto a través de la red. Se basa en una arquitectura cliente/servidor.

Java Script: Es un lenguaje interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas Web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C. Al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de herencia, es más bien un lenguaje basado en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad.

Malformación: Alteración o deformidad de nacimiento en alguna parte del organismo.

Morbimortalidad perinatal e infantil: Se define como todos los niños que fallecen antes de cumplir el año de edad.

Morbimortalidad: Número de personas afectadas y/o fallecidas por una enfermedad en un periodo de tiempo.

Morfogénesis: Incluye la forma de los tejidos, de los órganos y de los organismos completos y las posiciones de varios tipos de células especializadas

Morfológicos: El estudio de la forma de un organismo o sistema.

Mortalidad: Tasa de muertes producidas en una población durante un tiempo dado.

MySQL: Es un sistema de gestión de base de datos, multihilo y multiusuario, software libre en un esquema de licenciamiento dual.

Oligoamnios: Disminución del volumen de líquido amniótico por debajo de 100 ml.

Perinatal: Problemas que pueden suceder durante el momento de dar a luz / Que pertenece u ocurre durante el período anterior al parto o nacimiento, o es simultáneo o posterior a él, es decir, desde las 28ª. Semana de gestación hasta los primeros los siete días después del parto.

PHP: Es un lenguaje de programación usado generalmente para la creación de contenido para sitios web.

Polihidramnios: Hidropesía del amnios, membrana que envuelve el feto. A veces se llama simplemente hidramnios. Es un exceso de líquido en la cavidad amniótica.

Reporte: Informe detallado sobre alguna información, o sobre el estado de la información.

Software: Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

SIGM: Sistema Informático para la Red Nacional de Genética Médica.

Vigilancia Epidemiológica: Es un proceso sistemático, ordenado y planificado de observación y medición de ciertas variables para luego describir, analizar, evaluar e interpretar tales observaciones y mediciones con propósitos definidos. Es un proceso lógico y práctico de evaluación permanente sobre la situación de salud de un grupo humano, que permite utilizar la información para tomar decisiones de intervención a nivel individual y colectivo, con el fin de disminuir los riesgos de enfermar y morir.

Web Services: Un servicio web (en inglés Web service) es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet.