Universidad de las Ciencias Informáticas

Facultad 6

Título: “Sistema para la gestión de muestras del banco de ADN del Centro Nacional de Genética Médica”

Trabajo de Diploma para optar por el título de
Ingeniero Informático

Autor: Freddy Fernández Flaqué

Tutor: Ing. Reynaldo Alvarez Luna

Consultante: DrC. Teresa Collazo Mesa

La Habana, 2011

“Año 53 de la Revolución”

"En la tierra hacen falta personas que trabajen más y critiquen menos, que construyan más y destruyan menos, que prometan menos y resuelvan más, que esperen recibir menos y dar más, que digan mejor ahora que mañana."

Ché

DECLARACIÓN DE AUTORÍA

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

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

Freddy Fernández Flaqué **Ing. Reynaldo Alvarez Luna**

_____ _____

Firma del Autor Firma del Tutor

DATOS DE CONTACTO

Tutor: Ing. Reynaldo Alvarez Luna. Ingeniero en Ciencias Informáticas. UCI 2008.
Departamento de Integración de Soluciones. Centro de Tecnologías de Gestión de Datos. UCI.
Correo de contacto: rluna@uci.cu

Consultante: DrC. Teresa Collazo Mesa
Jefa del Departamento de Biología Molecular. Centro Nacional de Genética Médica, La Habana, Cuba.
Correo de contacto: tcollazo@infomed.sld.cu

AGRADECIMIENTOS

A mis padres por haberme traído al mundo y educarme como lo han hecho, sin duda sin su ejemplo, amor, constante atención y preocupación no hubiese logrado nada. En especial a mi madre que no escatima sacrificios, noches sin dormir, lágrimas cuando son necesarias, sonrisas en los momentos más difíciles, todo para estar siempre a mi lado, para que siga viviendo, para seguir latiendo, porque sabe que es mi corazón. A mi padre porque a pesar de la distancia siempre ha estado cuando lo necesito y por ser una guía en mi carácter, en mis principios y un ejemplo en los valores que todo hombre debería cultivar. A mi hermana Roxana por permitirme sentirme orgulloso de ser su hermano, por sus oportunos consejos en los momentos que lo he necesitado. A Jorge y Carlos Arce, los hermanos varones que no tuve. Gracias por permitirme construir esta bella amistad que se ha transformado en estos cinco años en una hermandad inquebrantable. A mi familia toda, por estar siempre tan pendientes de mi y de cada paso que doy, a mis tías por estar siempre apoyándome como si fuera un hijo, a mi abuelita esperanza por su ternura inagotable, por aguantar todas mis malcriadeces y sobre todo por prepararme la merienda antes de acostarme.

Agradecer a aquellos que se han convertido en mi familia y siempre me han apoyado en todo con su amistad infinita, con su cariño, con sus consejos, a Ofelia, Martha, Raulito, a mi hermano Cuenca, a Alexander y Pedro Pablo, a Idanis y mis hermanos de la FEEM. A Arelys por su amor inagotable, por su paciencia y comprensión, por permitirme compartir momentos inolvidables en la vida y porque fuiste, eres y serás por siempre mi flaquita preciosa.

A los amigos que una vez conocí cuando ingresaba por primera vez a la universidad y se han mantenido firmes junto a mí, a esos que nunca podré olvidar y representan para mí el tesoro más valioso, A Luilly, Gendrys, Omar y Pico. A mi grupo 6101 de primer año, por ser mis primeros compañeros en la universidad, sin duda los mejores porque supimos mantenernos casi en masa hasta el final. A mis compañeros de la recta final, al grupo 6512, sin duda la madurez alcanzada y la unidad nos permitieron vencer todos los obstáculos y siempre estar a la vanguardia. A Alejo por sus enseñanzas y su apoyo incondicional, al Yasel por su alegría contagiosa. A Dariel, al Rodo, el Rojo, la negrita Dayné, Lili, la Vivi, Milena, Yani, Yndi, el Rolo y a todos los que guardo en mi corazón.

A mis profesores de la universidad por todo lo que me han inculcado. En especial a Yanelis Benítez, una persona que se ha convertido en una madre para mí y que no solo me enseñó matemática, sino que me inculcó valores, me brindó su amistad desinteresada e incondicional y me abrió su corazón como sólo una madre sabe hacerlo.

A mi tutor Reynaldo Alvarez Luna por el apoyo en el desarrollo de esta tesis, a la DrC. Teresa Collazo del Centro Nacional de Genética Médica por siempre estar en plena disposición y atendernos en cada momento que necesitamos.

A mis hermanos de batalla de la FEU, por permitirme formarme junto a ellos en esa segunda escuela que constituye la Organización.

A Fidel y a Raúl por ser continuadores de la historia de nuestra Patria y permitirnos vivir y formarnos como verdaderos revolucionarios profesionales en esta universidad de futuro y orgullosos de nuestra Revolución.

A todos los que de una manera u otra han colaborado al desarrollo de este sueño.

DEDICATORIA

Dedico este trabajo de diploma a mi abuelita Iluminada. Aunque no esté físicamente, donde quiera que esté, sé que estaría muy orgullosa de mí, pues gracias a sus atenciones cuando era pequeño, a su cuidado, a su cariño también he podido llegar hasta este importante momento de mi vida.

A mi abuelita Esperanza. Tal vez nunca no he tenido la oportunidad expresa de decirte con todas las fuerzas de mi corazón cuanto te quiero, creo que este es el momento. Aquí está tu nietecito deseándote mucha salud y que sigas con esa fuerza y alegría contagiosa. Gracias abue.

A mi madre Ana Caridad por ser mi inspiración en cada momento, por constituir el argumento más importante de mi vida, por su entrega a mi hermana y a mí. Por siempre luchar a pesar del cansancio. Por su alegría y su forma de ser que nos ha permitido llegar hasta aquí. Gracias mamita. Te amo con todo mi corazón.

A mi padre Fernando Fernández por su carácter y la formación que a pesar de no vivir junto a mí siempre me trasmitió. Porque en mi forma de ser llevo mucho de él y porque siempre espero que esté orgulloso de sus hijos.

A mi hermana Roxana por su reciente título como licenciada en derecho. Porque aunque discutamos siempre la querré con todo mi corazón porque es la hermanita menor que más quiero y la única que tengo.

A mis familiares porque siempre me han apoyado dándome consejos y fuerzas para seguir adelante. Yo soy quien se siente orgulloso de tenerlos.

A mis compañeros de aula del grupo 6101 de la Universidad de las Ciencias Informáticas, por ser mi primer grupo, y constante inspiración en los momentos más difíciles de la vida universitaria.

A Fidel mi ejemplo y líder indiscutible, creador de esta universidad que hoy me permite graduarme como profesional. A Raúl, por enseñarnos que los valores y los principios nunca deben perderse, que sólo el trabajo constante, el control y el ejemplo podrán permitirnos seguir adelante.

A la Revolución cubana, esa de la que soy un orgulloso y fiel hijo. Esa que defenderé con mi sangre si es necesario hasta mis últimas consecuencias.

RESUMEN

Los avances en la informática y la computación ha permitido a la biología molecular en las últimas décadas el procesamiento de un gran volúmen de datos. Lo anteriormente planteado, unido a la necesidad de disponer de muestras biológicas de calidad en excelentes condiciones y fácilmente disponibles, hace necesaria la creación de los denominados biobancos de muestras vinculadas a diagnósticos de enfermedades de origen genético, constituyendo estos, una eficaz herramienta al servicio de la comunidad científica.

Los biobancos acogen muchos tipos de materiales e información biológica, desde ADN, tejidos, muestras de biopsias entre otros, de ahí la cada vez más frecuente necesidad del completo control sobre los datos que se gestionan en los mismos. La presente investigación tiene como objetivo la implementación de un sistema para la gestión de muestras del Banco Nacional de ADN del Centro Nacional de Genética Médica. Este centro es el encargado del estudio de las muestras de ADN para el diagnóstico de enfermedades genéticas en el país. El sistema propuesto, primero de su tipo en Cuba, es un sistema multiplataforma desarrollado con el uso de tecnologías libres. El lenguaje de programación empleado es PHP 5.2.5 y como framework de desarrollo se utiliza symfony en su versión 1.4.10.

La solución propuesta es de gran significación para este centro ya que dota a los investigadores de una útil herramienta de apoyo para la gestión de la información. Esto permite tener un mejor control de las muestras de ADN que se registran en el banco pudiendo además integrar resultados y lograr mayor seguridad en la información gestionada.

PALABRAS CLAVE

ADN, biobancos, gestión de muestras, PHP, symfony

TABLA DE CONTENIDOS

INTRODUCCIÓN

Desde tiempos remotos, el hombre ha tratado de entender las causas de las enfermedades que lo aquejan y cómo éstas en ocasiones se presentan en una misma familia siendo trasmitidas de generación en generación. Muchas veces la explicación ha sido basándose en creencias y otras veces en conocimientos científicos que ha ido incorporando a partir de sus investigaciones.

Como resultado de estos estudios surge la "genética humana, ciencia que estudia la variación entre los seres humanos basada en las diferencias en el material hereditario y con ella la genética médica que se ocupa de la aplicación de estos principios a la práctica médica y del estudio del papel de los genes en el origen de los rasgos humanos y de las enfermedades." (1)

En julio de 2001 y hasta junio de 2003, a propuesta del Comandante en Jefe Fidel Castro Ruz, se realiza un estudio de las personas con discapacidades mayores y como parte de las estrategias diseñadas a partir de los resultados de esa investigación, comienza una nueva y revolucionaria etapa en el desarrollo de la genética médica en el país y su extensión a todas las áreas de salud.

Como muestra de este fehaciente esfuerzo está la inauguración de un nuevo Centro Nacional de Genética Médica. Este centro tendría la misión de desarrollar integradamente la especialidad coordinando la formación de recursos humanos, la actividad asistencial e investigativa en la red nacional de centros de genética. Actualmente esta red de centros está integrada por áreas de salud, centros municipales y centros provinciales que garantizan la cobertura y el acceso a estos servicios en todos los municipios y provincias del país incrementando las posibilidades diagnósticas para estas enfermedades.

"El Centro Nacional de Genética Médica (CNGM) tiene dentro de sus funciones principales las investigaciones básicas y aplicadas en el campo de la genética médica, la inmunología, la bioquímica y otras disciplinas afines dirigidas a la obtención de nuevos conocimientos, evaluación y desarrollo de nuevas tecnologías, productos y procedimientos de trabajo, con el fin de mejorar los niveles de salud del pueblo cubano y disminuir el impacto de las enfermedades con implicación genética en el cuadro de la morbimortalidad[1] del país y realizar aportes al desarrollo de esta rama de la ciencia, teniendo en cuenta las potencialidades que se derivan de su integración" (2). Dentro de los departamentos de este centro se encuentra el departamento de Asistencia Médica donde se ubica el laboratorio de biología molecular encargado de los estudios de laboratorio para el diagnóstico de enfermedades genéticas.

[1]**Morbimortalidad:** Mortalidad por causa de una enfermedad.

Los estudios de laboratorio para el diagnóstico de enfermedades genéticas se realizan gracias a la información obtenida de la muestra de ADN[2] de cada paciente. “El ADN es responsable de contener toda la información genética de un individuo o ser vivo y los datos genéticos que serán hereditarios de generación en generación.” (3). De ahí la importancia del cuidado y delicado manejo de estas muestras que deben tener una calidad asociada no solo a las condiciones de recogida y conservación, sino también a su trazabilidad y a la precisión y fiabilidad de los datos clínicos asociados.

Para garantizar las condiciones de estas muestras, éstas deben ser protegidas de forma especial y en ocasiones concentradas durante años. Es por ello que el laboratorio de biología molecular tiene bajo su responsabilidad el banco de muestras de ADN con fines diagnósticos asegurando su adecuada protección de acuerdo con la resolución[3] ministerial Nro. 219 del 2007.

Un banco de ADN es un establecimiento que acoge muestras biológicas asociadas con información clínica. Las funciones de un banco pueden variar en dependencia de los objetivos específicos para el que haya sido creado y las características de la institución de salud que lo coordine.

Entre las principales funciones del banco de ADN definidas por el CNGM se pueden citar:

- La obtención, preparación y almacenamiento de muestras biológicas y datos asociados a las mismas.
- La distribución de muestras de ADN a grupos de investigación.
- La custodia de muestras de ADN que han sido enviada a terceros países para estudios científicos.

Como parte del proceso de gestión del banco de ADN del CNGM las muestras son registradas y clasificadas según la enfermedad o caso de estudio, para ello el técnico conforma para cada enfermedad un registro individual donde inserta datos relacionados con el paciente, la muestra de ADN estudiada y los resultados del estudio. Los registros son creados en hojas de cálculo de Microsoft Excel o en bases de datos de Microsoft Access, lo que dificulta a los investigadores relacionar entre sí toda la información gestionada en las distintas bases de datos del banco. El tener los registros por separado obliga al técnico a crear una nueva base de datos por cada nueva enfermedad genética

[2] **ADN:** Ácido desoxirribonucleico.

[3] **Resolución disponible en**: http://www.sld.cu/galerias/pdf/sitios/genetica/resolucion_219.pdf

estudiada en una muestra, consumiendo así más recursos y siendo más documentos por separados para analizar.

Por otra parte cuando se recibe una nueva muestra en el banco esta debe ser almacenada en un frízer[4] con características específicas para su conservación durante la investigación o para su custodia. Cada vez es mayor el volumen de información que debe ser gestionada por los genetistas y con él, el número de personal y tiempo empleado en la búsqueda de una muestra en específico debido a que en cada frízer se almacenan cientos de ellas que pueden estar guardadas durante varios años. Todo esto provoca atrasos en los estudios y el consecuente arribo tardío a conclusiones que pueden ser de vital importancia para investigaciones genéticas en el país.

Por la situación anteriormente descrita y la importancia que reviste para el CNGM el proceso de gestión de muestras en el banco nacional de ADN se arriba al siguiente **problema de la investigación**: Las deficiencias en el proceso de gestión de muestras del banco de ADN en el Centro Nacional de Genética Médica dificultan la realización de estudios para el diagnóstico de enfermedades genéticas. Enmarcado en el **objeto de estudio**: El proceso de gestión de las muestras en biobancos y delimitado en el **campo de acción**: la automatización del proceso de gestión de muestras del banco de ADN del Centro Nacional de Genética Médica.

Para darle cumplimiento al problema planteado se define como **objetivo general de la investigación**: Desarrollar un sistema para la gestión de las muestras del banco de ADN del Centro Nacional de Genética Médica. El mismo está desglosado en los siguientes **objetivos específicos**:

- Realizar el análisis del proceso de gestión de muestras en el banco de ADN del CNGM.
- Diseñar el sistema a partir del análisis realizado para la gestión de muestras del banco de ADN del CNGM.
- Implementar el sistema diseñado.

Como **tareas de la investigación** se identifican:

- Análisis e identificación de las funcionalidades principales de los sistemas existentes en el mundo para la gestión de muestras de bancos de ADN.
- Definición de requisitos funcionales y no funcionales del sistema.
- Selección de las herramientas a utilizar para la implementación del sistema.

[4] **Frízer:** Congelador para el almacenamiento de las muestras. La temperatura puede variar entre -20, -80 ó -196 grados.

- Definición de la arquitectura del sistema.
- Realización de los diagramas de clases del diseño del sistema.
- Realización del diagrama de despliegue.
- Implementación de los componentes del sistema.

Al finalizar esta investigación se espera obtener un sistema que permita la gestión de las muestras en el banco de ADN del Centro Nacional de Genética Médica. Este sistema ayudará al Laboratorio de Biología Molecular del CNGM a tener un mejor control de las muestras de ADN que se registran en el banco pudiendo además integrar resultados y lograr mayor seguridad en la información gestionada.

Para garantizar estos resultados el presente trabajo está estructurado en 4 capítulos que abordan los elementos fundamentales de la investigación organizados de la siguiente manera:

Capítulo 1: Fundamentos Teóricos.

El capítulo comprende un breve estudio de algunas de las aplicaciones existentes en el mundo para la gestión de biobancos. Se aborda además acerca de la metodología, herramientas y tecnologías propuestas para desarrollar la solución de la presente investigación.

Capítulo 2: Características del Sistema.

En este capítulo se describen como se gestionan las muestras del banco de ADN de CNGM, teniéndose una descripción de los procesos de negocio y un análisis crítico de las causas que originan el problema planteado. Se especifican los requerimientos funcionales y no funcionales de la aplicación propuesta para dar solución al problema de la investigación y se definen y describen los casos de uso del sistema.

Capítulo 3: Análisis y Diseño del Sistema.

En el capítulo se describe y fundamenta la arquitectura propuesta. Se exponen los patrones utilizados en el diseño del sistema. Además, se realiza un diseño donde se obtienen los diagramas de clases del así como otros diagramas y modelos que servirán de base para la implementación del sistema propuesto.

Capítulo 4: Implementación y Pruebas.

El capítulo presenta los diagramas de despliegue y componentes elaborados. Se argumenta sobre la seguridad realizada a la aplicación. Además se presentan las principales interfaces del sistema para la gestión de las muestras del banco de ADN del CNGM.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

Introducción

"Los avances en la informática y la computación ha permitido a la biología molecular en las últimas décadas el procesamiento de un gran volumen de datos, lo que unido a la necesidad de disponer de muestras biológicas de calidad en excelentes condiciones y fácilmente disponibles, hace necesaria la creación de los denominados biobancos, que constituyen una herramienta muy eficaz al servicio de la comunidad científica". (4)

Por otra parte, el almacenamiento masivo de muestras biológicas y la cada vez más frecuente necesidad de un completo control sobre las mismas plantea cuestiones técnicas complejas que afectan desde la propia recolección de la muestra, su transporte, su identificación, la trazabilidad, su conservación a diferentes temperaturas, la recuperación de la muestra guardada y el tratamiento informático de los datos, entre otros importantes aspectos.

En el presente capítulo se tratarán aspectos relacionados con las características de los biobancos y los sistemas existentes en el mundo para la gestión de las muestras en los mismos y se abordará sobre las herramientas y metodología a utilizar para la implementación del sistema propuesto en la investigación.

1.1 Biobancos, herramientas para la investigación biomédica

"Los bancos de muestras o biobancos son instituciones privadas o institucionales que se destinan a almacenar de forma adecuada cualquier tipo de muestra biológica por largos períodos de tiempo. El empleo de biobancos hace factible el desarrollo de investigaciones y constituye una invaluable fuente de información para determinar ciertas enfermedades y para encontrar respuestas a otras". (5)

En este sentido y para potenciar diversos tipos de investigaciones se han constituido los biobancos de muestras vinculadas a diagnósticos desde el que se puedan desarrollar ulteriores investigaciones de algunos procesos tan frecuentes en la población como son las enfermedades infecciosas, de origen genético, neurodegenerativas, el cáncer, entre otros.

"Los Biobancos ofrecen la posibilidad de enlazar las investigaciones de laboratorio con aplicaciones clínicas acelerando el desarrollo de la medicina personalizada." (6)

1.1.1 Tipos de biobancos

Los biobancos varían en tamaño y finalidad y se pueden clasificar siguiendo diferentes criterios: localización (bancos internacionales, nacionales o regionales), tipología de muestras gestionadas

(tejidos, tumores, células madre, ADN), u objetivo específico del biobanco (poblacionales, de enfermedades específicas).

También se pueden diferenciar según la estructura del biobanco, por un lado los mononodales, que están integrados por un único nodo donde se gestionan las muestras y las distribuye directamente a los investigadores, y por otro, los multinodales o distribuidos. Éstos funcionan en forma de red, donde las muestras del biobanco se ubican en diferentes centros de investigación u hospitales y existe una coordinación central que se encarga de mantener las relaciones con los investigadores potenciales usuarios de las muestras.

Es por ello que se hace necesario definir bien el tipo de biobanco en función de los objetivos y características de cada institución. Evidentemente los biobancos multinodales presentan mayor complejidad en la gestión de la información y requieren mayores controles y sistemas que garanticen la calidad, la trazabilidad y la protección de datos.

En el caso de Cuba, el banco nacional de ADN clasifica según su estructura en el tipo de banco mononodal. Cuenta con un único nodo que se encarga de concentrar las muestras y distribuirlas entre los investigadores en el mismo centro.

1.2 Sistemas de gestión de muestras en los biobancos

Originariamente la gestión de los datos en los biobancos se hacía de manera más rudimentaria y poco rigurosa y profesional, pero actualmente, gracias a los avances de las tecnologías de la información, la gestión de toda esta información se puede realizar de manera segura y eficaz.

Un software de gestión de biobancos es un sistema de gestión de información. “Un sistema de gestión de información se puede definir como un conjunto de funciones o componentes interrelacionados que forman un todo, es decir, obtiene, procesa, almacena y distribuye información para apoyar la toma de decisiones y el control en una organización. Igualmente apoya la coordinación, análisis de problemas, visualización de aspectos complejos entre otros.” (7)

1.2.1 Razones para la utilización de un sistema de gestión en los biobancos

Como se ha podido apreciar las actividades de los biobancos requieren sistemas de gestión de información, ya que en todos ellos, la información asociada a las muestras también es muy importante. Existen múltiples e importantes razones que hacen necesario el disponer de un software de gestión de muestras cuando se pone en marcha un biobanco, y todas ellas se pueden clasificar desde un punto de vista operacional o desde un punto de vista legal.

"Entre las principales razones para la utilización de sistemas de gestión de biobancos en Europa, principalmente en España se encuentran:

Operacional:

- El almacenamiento masivo de muestras biológicas requiere un sistema de gestión que permita registrar un gran número de datos.
- La implicación de diferentes agentes (comité ético, comité científico, coordinador y técnicos del biobanco) en el trabajo diario del biobanco requiere un sistema de gestión de datos que permita la conexión de diferentes usuarios simultáneos en diferentes localizaciones.
- La implicación de diferentes agentes hace necesario que sea necesario tener un sistema de gestión de datos estandarizado.
- La implementación del sistema de calidad requiere un sistema de control de modificación de datos y un registro de acceso de datos, disponible por el software de gestión de biobancos.

Legal:

- Asegurar la trazabilidad en los procesos.
- Guardar los datos genéticos durante los 5 años posteriores a la obtención de los datos.
- Anonimizar las muestras cuando el donante lo requiera.
- Permitir el acceso a los datos personales de los pacientes sólo al personal autorizado.
- Cuando se gestionan datos personales, se deben considerar las normas descritas por la Agencia de Protección de Datos (en caso de que un biobanco no gestione datos personales capaces de identificar a una persona individual, no se aplican las normas de la LOPD[5]). Si el biobanco gestiona datos clínicos y genómicos, estos tipos de datos se consideran datos personales, por lo que deberán aplicar las normas de la LOPD". (4)

Los biobancos son sistemas muy complejos, por lo que el software de gestión debe satisfacer los requerimientos de los diferentes usuarios que trabajan en el biobanco, debe ayudar a cumplir con los requerimientos legales y operacionales y debe ser de fácil manejo y amigable para asegurar una correcta utilización.

[5] **LOPD:** Ley Orgánica de Protección de Datos disponible: http://www.leyprotecciondatoslopd.com/

1.2.2 Ventajas y desventajas del uso de sistemas para la gestión de biobancos

"El software de gestión de biobancos tiene ventajas y desventajas reconocidas internacionalmente por las principales empresas dedicadas al desarrollo de aplicaciones informáticas y productos para la biomedicina que hay que tener en cuenta a la hora de ponerlo en marcha.

Ventajas

- Centraliza la información.
- Evita confusiones y registros duplicados.
- Estandariza y automatiza los procesos.
- Proporciona la gestión de calidad y de control.
- Analiza y explota la información almacenada.
- Genera estadísticas e informes.
- Proporciona garantía de confidencialidad de datos (nivel de seguridad 3 ó alto)[6].
- Asegura la trazabilidad de las muestras.
- Sirve para estandarizar los procedimientos de trabajo.

Desventajas

- Es necesario que los usuarios aprendan a utilizar la aplicación.
- Es necesario que los flujos de trabajo del sistema estén muy bien definidos.
- La entrada de datos debe estar regulada." (4)

Como se evidencia las ventajas en el uso de sistemas para la gestión de biobancos tienen una positiva implicación en los procesos que se realizan en el mismo, por lo que se apoya la necesidad del uso de estos sistemas para un mejor funcionamiento de los biobancos.

1.3 Sistemas para la Gestión de Biobancos

En el mundo se aprecia un notable desarrollo en el empleo de sistemas para la gestión de biobancos. Las aplicaciones informáticas desarrolladas son soluciones integrales que permiten gestionar grandes

[6] **Nivel 3 de Seguridad, disponible en:** http://www.seguridadinformatica.es/profiles/blogs/niveles-de-seguridad-segun

biobancos multinodales y al mismo tiempo se adaptan fácilmente para gestionar biobancos más pequeños de tipo mononodales. Los principales resultados se encuentran en Europa y sus soluciones no solo han repercutido en esta región, debido a que muchos de estos sistemas son usados por instituciones en América incluso en los Estados Unidos.

Entre los sistemas más representativos en la gestión de biobancos en el mundo se pueden citar:

Aplicación Bio-e-Bank[7]:

El software Bio-e-bank está desarrollado de forma escalable para biobancos de todos los tamaños y complejidades. Mantiene una licencia privativa y se comercializa a través de Vitro, empresa con gran prestigio en Europa como distribuidor y proveedor de soluciones en el campo de la salud y la información biomédica.

"La aplicación se basa en los tres elementos fundamentales para el funcionamiento de un biobanco y el cumplimiento de las regulaciones de investigación biomédica en Europa: la seguridad de datos, la trazabilidad de muestras y la integridad de procesos". (8)

"Bio-e-Bank no solo permite la comunicación con robots y equipos de almacenamiento automatizado sino que también puede conectarse con otros sistemas informáticos como por ejemplo los Sistemas de Gestión de Laboratorios (LIMS), Información Hospitalaria (HIS) y por supuesto otras unidades de biobanco.

Este sistema garantiza además la recogida de información en cualquier momento relacionada con los donantes, las muestras, los procesos y los resultados. Esta información se recoge en forma de cuestionarios que pueden ser configurables, específicos o la medida y que permiten además captura automática de información. La aplicación facilita la comunicación entre diferentes centros y el envió de muestras hacia centros coordinadores y actualmente ya está siendo utilizada en numerosos biobancos de referencia nacional en España, como por ejemplo el Nodo del Banco Nacional de ADN, el Nodo de Enfermedades Oncológicas y el Nodo de Enfermedades de Metabolismo entre otros." (8)

[7] Disponible en: http://www.bio-e-bank.com

Aplicación NorayBank[8]:

"El software de gestión NorayBank perteneciente a la empresa española Noray *Bioinformatics*, permite gestionar de forma sencilla todo tipo de biobancos: ADN, tejidos, células y microorganismos.

Las especificaciones funcionales se pueden dividir en tres partes en función del tipo de actividades que se gestionen: actividades rutinarias, de coordinación o de explotación de información". (9)

1. **Gestión de actividades rutinarias:**

- **Gestión de donantes:** Información asociada a datos personales del paciente. Codificación del paciente con código unívoco. Módulo preparado para gestión de información de nivel 3 de seguridad.
- **Gestión de muestras:** Recepción, introducción de datos asociados a la muestra y codificación de las mismas.
- **Preparación y alicuotado de las muestras:** Identificación y cadena de custodia de toda alícuota[9] generada de cada muestra.
- **Generación de etiquetas:** Impresión de códigos de barras que incluyan la identificación de la muestra.
- **Protocolos:** Asignación de protocolos estandarizados a cada muestra.
- **Almacenamiento de muestras:** Asignación de localización específica para cada muestra en los lugares de almacenamiento del biobanco.
- **Controles de calidad:** Registro de controles de calidad y bioseguridad realizados sobre cada muestra.

2. **Actividades de coordinación:**

- **Comunicación con investigadores:** Facilita la comunicación entre el biobanco y los investigadores para solicitar muestras o servicios.

[8] Disponible en: http://www.cultek.com/index.asp?p=productos-destacados-detalles&id_pdt=153

[9] **Alícuota:** Es el volumen o cantidad de masa que se va a emplear en una prueba de plataforma o de laboratorio.

- **Gestión de proyectos:** Control de los proyectos a los que se asignan las muestras: validación, autorización comité ético e investigadores autorizados.
- **Comités ético y científico:** Gestión de los proyectos, solicitudes de muestras y protocolos.
- **Comunicación entre nodos:** Facilita la comunicación entre el nodo de coordinación y los nodos asociados al biobanco.

3. **Explotación de la información:**

- **Informes y estadísticas:** Creación y visualización de informes y estadísticas bajo distintos parámetros
- **Control de modificaciones:** Registro de la información relativa a cualquier modificación realizada que ayude a cumplir con el sistema de calidad establecido.
- **Gestión de usuarios:** Asignación de niveles y permisos de acceso a los distintos usuarios autorizados del sistema.

"NorayBank ofrece a sus clientes soportes y desarrollos personalizados que resuelvan sus necesidades específicas por lo que cada vez son más las entidades que apuestan por este tipo de soluciones para sus biobancos, como ya lo han hecho el Banco de Líneas Celulares de Andalucía, el Centro de Investigación Biomédica en Red de Enfermedades Raras (CIBERER), el Centro de Investigación Biomédica en Red de Enfermedades Respiratorias (CIBERES) o el Banco de ADN del País Vasco, quienes han confiado en NorayBio para esta tarea." (9)

Aplicación TechBioBank[10]:

"Es una herramienta para la gestión de biobancos en el ámbito de la web 2.0 con características que la hacen exclusiva a nivel internacional. Aplicación que garantiza acceso web seguro y gestión colaborativa, además almacena conocimiento a través de las tareas ordinarias.

La funcionalidad y gestión de los datos está dirigida por protocolos y el sistema de información basado en objetos. Todo objeto cambia de estado por la ejecución de un protocolo y todo dato está asociado a un objeto y, en general, a un estado del mismo". (10)

[10]Disponible en: http://infozara.4java.ca/WebInfozara/productosTechBioBankAlcance.do?enlaceMenu=productos&enlaceSubmenu=techBioBank

"Entre las principales funcionalidades que le dan un nivel funcional a la aplicación se pueden encontrar:

- Recogida de muestras
- Separación de muestras
- Envío de muestras
- Anonimización[11], desanonimización
- Registro consentimiento informado
- Anulación de consentimiento informado
- Recepción de muestras
- Alta de alícuotas
- Separación de alícuotas
- Preparación de envío, envío de alícuotas, recepción de alícuotas
- Almacenamiento definitivo de alícuotas"(10)

TechBioBank está construido bajo arquitecturas que facilitan y simplifican su adaptación a las necesidades particulares de los que lo usan. Se adapta a los cambios naturales que se produzcan en el entorno y su interoperabilidad con los sistemas y dispositivos del laboratorio.

1.3.1 Valoración de los sistemas estudiados

Los sistemas estudiados anteriormente tienen dentro de su concepción permitir adaptabilidad a los diferentes tipos de biobancos existentes, por lo que podría ser un elemento a tener en cuenta para el desarrollo de la investigación propuesta, sin embargo, no garantizan un importante elemento a defender como lo constituye la independencia tecnológica en las condiciones en que vive el mundo de hoy, cada vez más monopolizado. A pesar de que estas herramientas permiten compartir conocimientos y relaciones entre diferentes centros en función de colaborar con las investigaciones, no dejan de ser herramientas propietarias y con un costo para su adquisición que el país no está en condiciones de sufragar.

Es importante mencionar que si bien Cuba ha apostado por el desarrollo de la genética médica, esta ciencia continúa siendo aún un campo de lujo de países desarrollados. Esto implica que no se disponga de todo el equipamiento necesario y de última tecnología que precisan muchos de estos sistemas para la integración de todas sus funcionalidades. Todo esto fundamenta la idea de que la

[11]**Anonimización:** Eliminar cualquier dato que pueda comprometer la identidad de un paciente. Esta puede ser solicitada por el mismo paciente.

solución que se desarrolle debe además de cumplir con las características de los estándares internacionales, adaptarse concretamente a las condiciones especiales y recursos disponibles en el país.

1.4 Sistemas para la gestión de biobancos en Cuba

En Cuba no se tienen experiencias en el uso de sistemas para la gestión de biobancos, a pesar del desarrollo que han venido teniendo las aplicaciones y soluciones informáticas en el campo de la salud y la gestión de la información biomédica. No obstante hoy se evidencian excelentes relaciones entre las universidades cubanas y las instituciones de salud en las que juegan un papel protagónico algunos centros del Polo Científico como el Centro Nacional de Genética y el Centro de Ingeniería Genética y Biotecnología.

El desarrollo de la presente investigación supone un aporte más de esta joven universidad al Sistema Informático para Genética Médica, además de constituir la primera experiencia en Cuba en el desarrollo de un sistema para la gestión de biobancos adaptado a las características de estos en el país.

1.5 Metodología de desarrollo de software

Desarrollar un buen software no es tarea nada fácil y para ello se hace necesario contar con un grupo de procedimientos, etapas y actividades que apoyen a los desarrolladores y garanticen el éxito del producto. Para lograr esto se emplean las metodologías de desarrollo de software.

“Una metodología de desarrollo de software es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un nuevo software”. (11)

Las metodologías se pueden agrupar en dos grandes grupos, las tradicionales y las ágiles. Las metodologías ágiles dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas, en estas el cliente llega a formar parte del equipo de trabajo; mientras que las metodologías tradicionales se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán.

Para el sistema propuesto es necesaria una metodología caracterizada por un fuerte control de los procesos, una amplia documentación y donde no necesariamente el cliente participe en el desarrollo junto al equipo de trabajo. Es por tal motivo que se decide utilizar una de las metodologías robustas

más conocidas y documentadas, definiéndose al Proceso Unificado de Desarrollo (RUP) como la metodología a emplear para la realización del sistema.

"RUP, es un proceso de ingeniería de software planteado por Kruchten (1996) cuyo objetivo es producir software de alta calidad, es decir, que cumpla con los requerimientos de los usuarios dentro de una planificación y presupuesto establecido. Cubre el ciclo de vida y desarrollo de software". (11)

El Proceso Unificado en su ciclo de vida se caracteriza por estar: dirigido por casos de uso, centrado en la arquitectura y ser iterativo e incremental. Impone además una disciplina de trabajo sobre el proceso de desarrollo del software. Tiene como objetivo expreso asegurar la producción de un software de alta calidad que satisfaga los requerimientos de los usuarios finales además de poder ser adaptada y extendida para satisfacer las necesidades de la organización que la adopte.

RUP consta de cuatro fases o etapas: inicio, elaboración, construcción y transición. En ellas intervienen diversos roles dentro del equipo de trabajo los cuales son los encargados de generar los artefactos en cada una de estas etapas.

En el desarrollo del sistema propuesto los principales roles de esta metodología que estarán presentes serán:

Rol Analista del Sistema: Se encargará del levantamiento de requisitos y realizará los diagramas correspondientes al flujo de trabajo análisis y diseño.

Rol Desarrollador: Implementará los componentes del sistema**.**

1.6 Lenguajes, tecnologías y herramientas de soporte al desarrollo

Las tecnologías y herramientas para el desarrollo del sistema se han definido teniendo en cuenta las pautas establecidas por el equipo de arquitectura del proyecto para las soluciones implementadas para el CNGM. A continuación se argumentan algunas de las características de estas herramientas, lenguajes y tecnologías así como las facilidades que brindan al desarrollo de la aplicación.

Visual Paradigm for UML Enterprise Edition 3.4

Se empleó como herramienta case ya que la misma soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Este software

utiliza como lenguaje de modelado UML lo que ayudará a una más rápida construcción de aplicaciones de calidad.

“UML es un lenguaje estándar para especificar, visualizar, construir y documentar todos los artefactos de un sistema de software”. (12)

“Esta herramienta está desarrollada por Visual Paradigm Internacional una de las principales compañías de herramientas CASE. Su mayor éxito consiste en la capacidad de ejecutarse sobre diferentes sistemas operativos lo que le confiere la característica de ser multiplataforma. Es muy fácil de usar y presenta un ambiente gráfico agradable para el usuario”. (13)

A pesar de tener licencia privativa la universidad garantiza su activación optando por esta robusta herramienta para el desarrollo de aplicaciones.

DB Designer 4.0.5.6

Es una herramienta libre para el diseño visual de bases de datos que integra: diseño, modelado, creación y mantenimiento de estas en un único entorno.

Con este software es posible crear bases de datos MySQL y de otros sistemas gestores de bases de datos libres como PostgreSQL pero permite además trabajar con algunos sistemas comerciales como Oracle y SQLServer.

Sistema Gestor de Bases de Datos

Se empleó MySQL en su versión 5.0.45. MySQL como Sistema Gestor de Bases de Datos (SGBD). El mismo es el SGBD establecido para el desarrollo de aplicaciones web para el sistema de salud cubano. Por otra parte en los últimos años ha tenido un crecimiento vertiginoso. Es una tecnología útil para la creación de bases de datos.

“Este SGBD es un servidor de base de datos muy rápido y sencillo de usar, multiusuario, multiplataforma (Windows, Linux. Mac OS, entre otros) y robusto escrito en C/C++; desarrollado, distribuido y con soporte por parte de MySQL AB y puede ser adquirido bajo dos licencias, el usuario puede adquirir una versión Open Source con términos de GNU General Public License ó puede comprar la versión comercial estándar bajo licencia MySQL AB.” (14)

Framework Symfony 1.4.10

“Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y

la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación". (15)

Este framework fue desarrollado en PHP 5 y ha sido probado en proyectos de gran envergadura, siendo compatible con casi todos los gestores de base de datos como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar en varios sistemas operativos. Incluye a Propel como mapeador Objeto-Relacional y Creole[12] como capa de abstracción de la base de datos. Además puede ser fácilmente configurable con otros plugins probados, tales como el *sfGuardPlugins* para facilitar el proceso de autenticación.

Este framework es hoy uno de los más utilizados en el mundo y en la universidad permitiendo simplificar el desarrollo de las aplicaciones al presentar elementos que permiten garantizar entre otros elementos la seguridad. Su licencia es totalmente gratuita y cuenta con una amplia documentación y ejemplos prácticos en varios idiomas que hacen más práctico su uso.

"Otra de las principales características de Symfony es que 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. Es posible realizar cambios (en caliente) de la configuración (sin necesidad de reiniciar el servidor)". (16)

Servidor de Aplicaciones Apache 2.0

"Apache HTTP[13] Server es un servidor HTTP de código abierto escrito en C y desarrollado como uno de los proyectos de *Apache Software Foundation* que funciona sobre la mayoría de los sistemas operativos incluyendo UNIX, MS-Windows, Macintosh y NetWare. Está basado en software libre y es distribuido bajo la licencia Apache de esta misma empresa. Es un servidor bastante seguro, eficiente y extensible, que proporciona servicios de HTTP en sincronización con los actuales estándares de este protocolo. Apache ha sido el más popular servidor web en Internet desde abril de 1996". (17)

Apache dispone de varios módulos que no están incluidos en el núcleo (core) del servidor, algunos de estos módulos permiten que Apache brinde las funcionalidades básicas para un servidor web y otras lo

[12] **Creole, disponible en:** http://www.propelorm.org/wiki/Introduction

[13] **HTTP:** Protocolo de Transferencia de Hipertexto. Es el protocolo usado en cada transacción de la World Wide Web.

convierten en mucho más que un servidor web básico ya que optimiza el rendimiento y la rapidez del código. Este servidor es utilizado en el desarrollo de softwares para el sistema de salud cubano.

Java Script:

"Esta tecnología del lado del cliente es un lenguaje interpretado que permite incluir macros en páginas web. Estas macros se ejecutan en el ordenador del visitante de nuestras páginas, y no en el servidor (algo muy interesante, porque los servidores web suelen estar sobrecargados, mientras que los PC's de los usuarios no suelen estarlo)". (18)

Java Script proporciona los medios para:

- Controlar las ventanas del navegador y el contenido que muestran.
- Programar páginas dinámicas simples.
- Evitar depender del servidor web para cálculos sencillos.
- Comprobar los datos que el usuario introduce en un formulario antes de enviarlos.

JQuery 1.6.1

Se utilizó como framework de presentación para lograr apariencias más agradables en la interfaz de usuario, además de facilitar el trabajo con el lenguaje Java Script. JQuery es de código abierto y cuenta con diversos complementos y amplia documentación para su desarrollo. Uno de los complementos empleados también en el sistema fue JQuery UI.

"JQuery UI es un complemento que permite implementar componentes diversos para generar interfaces de usuario en páginas web, además de otras funcionalidades básicas para crear aplicaciones web enriquecidas. Como su propio nombre indica, está basado en el popular framework Java Script". (19)

PHP 5.2.5

"PHP (acrónimo de "PHP: *Hyper Text Preprocessor*") es un lenguaje de código abierto interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor.

Este lenguaje puede ser utilizado en cualquiera de los principales sistemas operativos del mercado. PHP soporta la mayoría de servidores web de hoy en día". (20)

El principal objetivo de PHP 5 ha sido mejorar los mecanismos de Programación Orientada a Objetos (POO) para solucionar las carencias de las anteriores versiones. Esto ha permitido conseguir que PHP sea un lenguaje apto para muchos tipos de aplicaciones y entornos.

IDE NetBeans 6.9

"El IDE NetBeans es un entorno integrado de desarrollo *Open Source* escrito íntegramente en Java utilizando la plataforma NetBeans.

Es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación como es el caso de PHP. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso." (21)

NetBeans funciona en sistemas operativos compatibles con la máquina virtual Java (Windows XP, Vista, Windows 7, Ubuntu 9.10, Solaris, Mac OS X 10.5 ó superior).

Conclusiones parciales

Luego del estudio del presente capítulo se expusieron las principales características de los biobancos en el mundo y se identificó como banco mononodal al banco nacional de ADN del CNGM. Se clasificó como biobanco mononodal al Banco de ADN del CNGM a partir de las características vistas para este tipo de institución. Fue definida RUP como metodología de desarrollo y UML como lenguaje de modelado. Además fueron seleccionadas las siguientes herramientas para el desarrollo de la aplicación: PHP 5.2.5 como lenguaje del lado del servidor y Java Script como lenguaje del lado del cliente, como framework de desarrollo a Symfony 1.4.10 y JQuery 1.6.1 como framework de presentación. El sistema gestor de base de datos seleccionado fue MySQL en su versión 5.0.45.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Introducción

Un proceso de desarrollo de software tiene como propósito la producción eficaz y eficiente de un producto de software que reúna los requisitos del cliente. Es por ello, que para lograr un mejor entendimiento de los características del presente sistema informático en el capítulo se muestra parte del resultado del proceso de desarrollo del software donde se exponen los objetivos estratégicos de la organización; la modelación del negocio para lo cual se identifican y describen los actores, trabajadores y casos de uso del negocio, se realizan los diagramas de actividades correspondientes a cada caso de uso del negocio, y el diagrama del modelo de objetos. Se plantean los requisitos funcionales y no funcionales de la aplicación a desarrollar y se modela la misma en términos de casos de uso de sistema.

2.1 Objetivos de la Organización

Dentro de los departamentos del CNGM se encuentra el departamento de Asistencia Médica donde se ubica el Laboratorio de Biología Molecular encargado de los estudios de laboratorio para el diagnóstico de enfermedades genéticas. Los estudios de laboratorio para el diagnóstico de enfermedades genéticas se realizan gracias a la información obtenida de la muestra de ADN de cada paciente, las cuales son almacenadas en el Banco de ADN perteneciente a dicho departamento. El Banco de ADN del CNGM, es un biobanco de tipo mononodal que como bien su nombre lo indica trabaja a partir del estudio de la información genética obtenida de las muestras de ADN de los pacientes o donantes. El principal rol de los genetistas en la organización es el estudio de estas muestras para obtener información y arribar a conclusiones sobre cómo estas enfermedades genéticas se comportan en una determinada población.

El Banco de ADN cuenta con diferentes especialistas de genética que se encargan de la gestión de las muestras y de registrar los resultados que se obtienen del estudio de las mismas. Son los encargados de mantener la confidencialidad de la información gestionada y del cuidado y almacenamiento de las condiciones de las muestras para posteriores estudios.

La gestión de las muestras del Banco de ADN en el CNGM tiene como fin, ayudar a mejorar la calidad de vida de la población y prevenir enfermedades genéticas. Esto posibilita tener información para realizar estudios estadísticos que permitan elaborar un programa cada vez más eficiente, orientado hacia la prevención y control de las enfermedades crónicas. Toda la información recogida en el banco no debe ser eliminada del sistema con el objetivo de emplearlas en posteriores estudios.

2.2 Modelo actual del negocio

En todo proceso de desarrollo de software un primer paso lo constituye precisamente comprender todos los procesos que se realizan manualmente. De ahí que sea útil la creación de modelos que organicen y presenten los detalles importantes de la problemática real vinculados con el sistema informático a construir. Estos modelos deben cumplir una serie de propiedades, entre ellas la de ser coherentes y relacionados para una mejor comprensión de los mismos.

La descripción del negocio propuesto en detalle tendrá entre sus actividades principales la identificación de los procesos del entorno organizacional, delimitación del modelo de casos de uso, la especificación de los casos de uso, la identificación de trabajadores y entidades que ejecutan las realizaciones de los casos de uso y detallar la definición de las entidades y las responsabilidades de los trabajadores.

2.2.1 Flujo Actual de los Procesos

Los médicos solicitan y envían al Banco de ADN una determinada muestra o población para su estudio. En el banco los genetistas reciben las muestras codificándolas y archivando datos asociados a las mismas como la fecha de extracción y entrada al banco, la enfermedad por la que se estudia, así como otros datos relacionados con el paciente. Posteriormente estas muestras son almacenadas en un frízer con características específicas para garantizar la conservación de las mismas. Los genetistas agrupan las muestras por tipo de estudio y estas son custodiadas siempre bajo el consentimiento del paciente donante.

Luego de diferentes exámenes de laboratorio con las muestras de ADN, los genetistas deben entregar el ó los resultados al Centro de Asistencia Médica el cual a su vez los entrega al médico que solicitó el estudio.

El grado de seguridad del resultado y los posibles diagnósticos dependerán en gran medida de los datos obtenidos a partir del estudio de la información genética contenida en las muestras. Los genetistas deben acceder de forma continua tanto a las muestras físicas almacenadas como a la información que se va registrando como resultado de los exámenes de laboratorio a cada muestra, proceso que se torna hoy un poco engorroso por la cantidad de documentos a analizar.

Luego de la entrega al Centro de Asistencia Médica de los resultados, estas muestras quedan registradas en el Banco de ADN para posteriores estudios y análisis en poblaciones mayores. De igual manera la muestra de las muestras de ADN que son enviadas a terceros países para estudios son

custodiadas en el banco registrándose los mismos datos que las que tienen fines de diagnóstico médico.

2.2.2 Procesos que son objetos de Automatización

Gestión de datos de las muestras:

- **Automatización de los datos de las muestras de ADN:** Este proceso consiste en una base de datos que tendrá toda la información de las muestras. Los datos sobre la fecha de extracción y entrada así como los datos personales del paciente y los resultados asociados a la misma se guardarán de modo que al ser consultada la muestra se pueda estudiar con todos los detalles. Cada muestra será codificada automáticamente y podrá ser visualizada o eliminada y se podrán modificar sus datos, por si se detecta algún error en la información. Se permite conocer la ubicación física en el Banco de ADN y el estado de la misma.
- **Registro de Resultados:** Consiste en el registro del resultado del estudio de cada muestra.
- **Búsqueda de Muestras:** Este proceso automatizado es más rápido que en el negocio actual y restringe la búsqueda, posibilitando que se pueda buscar por varios criterios de búsqueda por ejemplo: tipo de registro, nombre del paciente ó fecha de entrada de la muestra al banco.

2.2.3 Actores del Negocio

A continuación se muestra para un mejor entendimiento del propósito del negocio quién envía o recibe algún beneficio de la gestión de las muestras de ADN en el banco.

“El término actor significa el papel que desempeña alguien o algo al interactuar con el negocio”. (22)

Actores del Negocio	Justificación
Médico	Son los individuos que solicitan el estudio de una muestra de ADN y se benefician al recibir el resultado luego del proceso de gestión de la muestra en el Banco de ADN.

Tabla 2.1. Descripción de los actores del negocio.

2.2.4 Trabajadores del Negocio

“Aquellas personas que ejecutan dentro del negocio los diferentes procesos, son llamados trabajadores del negocio.” (22)

Trabajadores del Negocio	Justificación
Técnico	Es el encargado de recibir las muestras y registrar sus datos.
Genetista	Es el que realiza los análisis de laboratorio a las muestras y registra y entrega los resultados de los estudios.
Departamento de Asistencia Médica	Es quien recibe el resultado del estudio de las muestras para entregarlo al médico.

Tabla 2. 2 Descripción de los trabajadores del negocio.

2.2.5 Diagrama de casos de uso del negocio

Figura 2. 1 Diagrama de Casos de Uso del Negocio

2.2.6 Modelo de Objetos del Negocio

"El diagrama de clases del modelo de objeto, es un artefacto que se construye para describir el modelo de objetos del negocio, muestra la participación de los trabajadores y entidades del negocio y la relación entre ellos". (22)

A continuación se muestra en la figura 2.2 el diagrama de objetos del negocio donde se relacionan los trabajadores y las entidades que tienen lugar en el mismo.

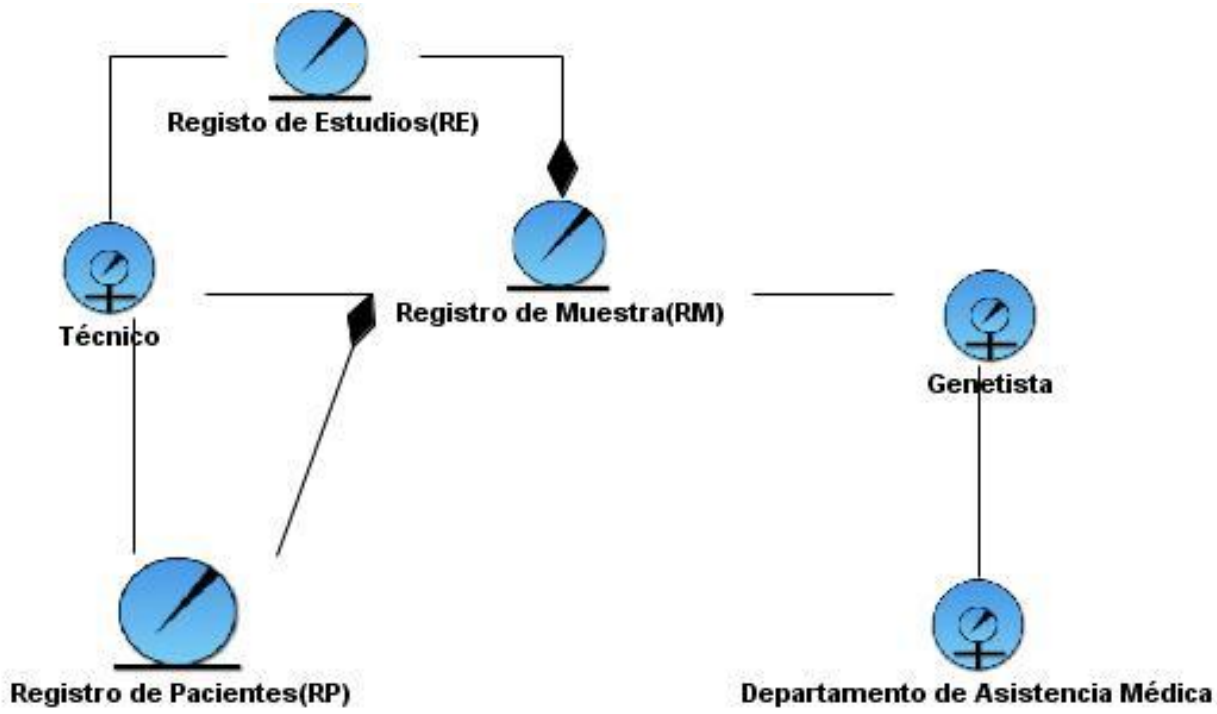

Figura. 2.2 Diagrama de objetos del negocio.

2.3 Especificación de los Requisitos

El sistema propuesto será desarrollado en un ambiente web con una agradable interfaz permitiendo insertar los datos asociados a una muestra recibida en el Banco de ADN del Centro Nacional de Genética Médica. Los datos se almacenarán de forma organizada en el registro de cada muestra. El sistema facilitará la actualización de los datos, así como la información de la ubicación física de cada muestra en el Banco de ADN.

El flujo de trabajo de requerimientos ayuda a establecer y mantener el acuerdo con los clientes ó los interesados en la aplicación. Este flujo proporciona a los desarrolladores del sistema una mejor comprensión de los requisitos, define las fronteras del software, establece una base para planificar el contenido técnico de las iteraciones, costo y tiempo de desarrollo del sistema. Define además una interfaz para el usuario enfocado en las necesidades y metas planteadas.

Requisitos Funcionales

“Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir.” (22)

Para el desarrollo de este sistema se han definido los siguientes requisitos funcionales:

R1. Validar usuario y contraseña de los usuarios del sistema.
R2. Registrar muestra.
R3. Modificar muestra.
R4. Eliminar muestra.
R5. Buscar muestra según criterios de búsqueda especificados.
R6. Visualizar ubicación de las muestras
R7. Registrar resultados.
R8. Modificar resultados.
R9. Visualizar resultados.
R10. Generar reporte.
R10.1 Exportar el resultado del reporte generado.
R10.2 Imprimir el resultado del reporte generado.
R11. Registrar un nuevo registro de estudio.
R12. Buscar registros existentes.
R13. Visualizar registros existentes.
R14. Eliminar registros.
R15. Crear usuarios.
R16. Visualizar usuarios existentes.
R17. Asignar permisos a los usuarios.
R18. Eliminar un usuario.
R19. Insertar Almacén.
R20. Modificar Almacén.
R21. Visualizar Almacenes.
R22. Eliminar Almacén.
R23. Insertar Nivel.
R24. Modificar Nivel.
R25. Visualizar Niveles.
R26. Eliminar Nivel.
R27. Insertar Casilla.
R28. Modificar Casilla.
R29. Visualizar Casilla.
R30. Eliminar Casilla.

Requisitos no Funcionales

"Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Estas propiedades se ven como las características que hacen al producto atractivo, usable, rápido ó confiable". (22)

R1. Apariencia o interfaz externa.

La aplicación presentará una interfaz agradable a la vista del usuario y que al mismo tiempo denota seriedad. Mantendrá un diseño acorde a la organización del contenido disponible permitiendo brindar las principales funcionalidades para la gestión de muestras del Banco de ADN del Centro Nacional de Genética Médica en Cuba. Se seleccionará el verde como color predominante y se incorporarán sólo las imágenes necesarias para la comprensión de términos médicos.

R2. Usabilidad.

Se debe garantizar un acceso fácil y rápido a los usuarios. El sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de una computadora y de un ambiente web en sentido general.

R3. Seguridad.

Se debe garantizar que la información solo pueda ser vista por los usuarios con el nivel de acceso autorizado para ello; permitiendo que las funcionalidades del sistema se muestren de acuerdo al nivel de usuario que esté activo. Deberá garantizar confidencialidad de los datos.

R4. Políticos – Culturales.

En la aplicación se deberá utilizar idioma español. Contará con logotipos e imágenes que se encuentren en correspondencia con el carácter científico y profesional del tema. Los cambios que se requieran realizar deben ser previamente consultados con la dirección del CNGM y el proyecto.

R5. Legales.

El sistema se desplegará en las PC's del Centro Nacional de Genética Médica como herramienta para la gestión de las muestras del banco de ADN y como apoyo para el estudio de enfermedades genéticas. Se usarán principalmente herramientas de software libre.

R6. Software.

La máquina servidor deberá disponer del servidor web Apache versión 2.0 o superior. Además se debe constar con el sistema gestor de base de datos MySQL 5.0.45. El lenguaje de programación debe ser PHP versión 5.2.5.

Las máquinas clientes podrán contar con sistema operativo Windows XP o superior o cualquier distribución de Linux. Las máquinas clientes para acceder al sistema deben hacerlo a través del navegador web Firefox, Internet Explorer 6.0 ó superior. Dichas máquinas deben tener instalado el Adobe Acrobat Reader u otro programa que soportes el formato PDF para la impresión.

R7. Hardware.

La máquina servidor deberá tener como mínimo 80 GB de capacidad de disco duro, microprocesador superior a 3.00 GHz, 512 MB mínimo de memoria RAM.

R8. Restricciones en el diseño y la implementación.

La lógica de presentación constituirá una capa independiente de la lógica de negocio, centrando su función en la interfaz de usuario y validaciones de los datos de entrada. Se utilizará como lenguaje de programación PHP 5.2.5, como Gestor de Base de Datos MySQL 5.0.45, como framework de desarrollo Symfony 1.4.10 y como IDE de desarrollo NetBeans 6.9, la Suite Visual Paradigm 3.4 Enterprise Edition como herramienta CASE para el modelado de los artefactos.

R9. Confiabilidad.

El sistema debe ser administrado solamente por una persona capacitada y será usado por médicos registrados, por tanto, la información que fluirá será la que se gestiona en el banco de ADN del CNGM. Se validará el registro de los datos del sistema disminuyendo el error humano y así la información será lo más confiable posible.

R10. Requisitos para la documentación de usuarios y ayuda del sistema.

Se realizará un manual de usuario para ayudar a los usuarios a trabajar con el sistema.

2.4 Definición de los Casos de Uso del Sistema

2.4.1 Actores del Sistema

"Cada trabajador del negocio que tiene actividades a automatizar es un candidato a actor del sistema. Si algún actor del negocio va a interactuar con el sistema, entonces también será un actor del sistema". (22)

Actor	Descripción
Genetista	Puede gestionar toda la información relacionada con las muestras de ADN registradas en el sistema y los resultados del estudio de las mismas. Puede generar reportes, imprimirlos y exportarlos.
Técnico	Gestiona la información de las muestras de ADN en el banco. Registra la entrada de las muestras al Banco y de la información básica asociada a las mismas.
Administrador del sistema	Es el encargado de administrar los usuarios existentes en el sistema y de autorizar los permisos a los usuarios.
Usuario	Una vez autenticado puede cambiar su rol y así su acceso a otras funcionalidades.

Tabla 2.3. Trabajadores del Sistema

2.4.2 Diagrama del Sistema

Diagrama de casos de uso del sistema

"Los diagramas de casos de uso son importantes para visualizar, especificar y documentar el comportamiento de un elemento. Estos diagramas facilitan que los sistemas, subsistemas y clases sean abordables y comprensibles, al presentar una vista externa de cómo pueden utilizarse estos elementos en un contexto dado". (12)

El sistema cuenta con 10 casos de uso, de ellos 4 son arquitectónicamente significativos.

Para mejor comprensión del diagrama de caso de uso del sistema dirigirse a las figuras 2 y 3 de la página de anexos donde se muestra la vista de los casos de uso arquitectónicamente significativos así como el diagrama íntegro de caso de uso del sistema de la aplicación respectivamente.

En la figura 2.3 se muestra el diagrama con la relación entre los actores del sistema:

Diagrama de relación entre actores del sistema

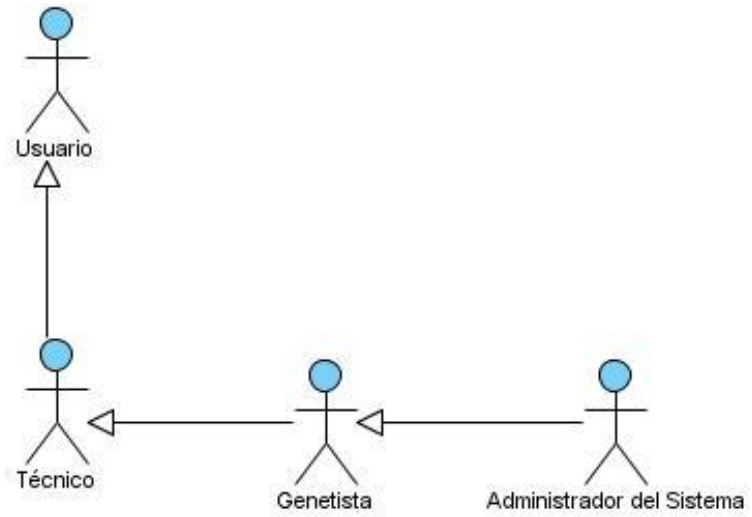

Figura. 2.3 Diagrama de relación entre actores.

2.4.3 Descripciones de los casos de uso del sistema arquitectónicamente significativos

Descripción del caso de uso "gestionar muestras"

Nombre del Caso de Uso	Gestionar Muestras
Actores	Técnico (inicia)
Propósito	Insertar, actualizar, buscar o eliminar muestras
Resumen	El CU inicia cuando el técnico desea insertar, actualizar, buscar o eliminar una muestra. Luego de que el técnico indique los datos de la muestra a insertar, actualizar, buscar o eliminar entonces el sistema registra, actualiza o elimina la muestra correspondientemente.
Referencias	R2, R3, R4, R5
Precondiciones	El técnico está autenticado en el sistema.
Poscondiciones	Se actualiza el listado de muestras.
Prioridad	Crítico

Curso Normal de los Eventos	
Acciones del Actor	**Respuesta del sistema.**
1. El técnico indica la opción Registro de Muestras.	2. El sistema visualiza las muestras existentes brindando la posibilidad de insertar, eliminar, buscar o editar una muestra.
3. El técnico realiza una de las siguientes operaciones: • Insertar una nueva muestra. • Eliminar una muestra existente. • Buscar una muestra. • Actualizar una muestra existente.	4. El sistema en dependencia de la operación indicada por el técnico realiza lo siguiente: • Si el técnico desea insertar una nueva muestra entonces se ejecuta la sección "Insertar muestra". • Si el técnico desea eliminar una muestra existente entonces se ejecuta la sección "Eliminar muestra". • Si el técnico desea buscar una muestra entonces se ejecuta la sección "Buscar muestra". • Si el técnico desea actualizar una muestra entonces se ejecuta la sección "Actualizar muestra".
Sección "Insertar muestras".	
Acciones del Actor	**Respuesta del Sistema**
1. El técnico ingresa los siguientes datos de la muestra: • Código Externo • Procedencia • Folio • Tipo de Muestra • Estado de la Muestra • Fecha de Extracción • Fecha de Entrada • Paciente • Registro de Estudio • Casilla	2. El sistema verifica que todos los campos necesarios estén llenos. 3. El sistema verifica que los datos introducidos sean correctos. 4. El sistema asigna un código a la muestra. 5. El sistema muestra un mensaje indicando que la muestra ha sido registrada satisfactoriamente. 6. El sistema visualiza el listado actualizado de las muestras.
Flujo alternativo Sección "Insertar muestras".	
Acciones del Actor	**Respuesta del Sistema**
	3.1 Si el técnico dejó algún campo necesario vacío

	entonces el sistema emite un mensaje de alerta indicando que los campos están vacíos.
	4.2 Si el técnico introdujo valores incorrectos entonces el sistema emite un mensaje de alerta indicando el error.
Prototipo Sección "Insertar muestra"	
Sección "Eliminar muestras".	
Acciones del Actor	**Respuesta del Sistema**
1. El técnico indica del listado mostrado la muestra que desea eliminar.	2. El sistema muestra un mensaje de alerta indicando si desea eliminar la muestra.
3. El técnico confirma afirmativamente.	4. El sistema elimina la muestra indicada por el técnico, vacía la ubicación en el almacén donde se encuentra y elimina los resultados asociados a la misma.
	5. El sistema muestra un mensaje informando que la muestra ha sido eliminada satisfactoriamente.
	6. El sistema muestra el listado actualizado de las muestras.
Flujo alternativo Sección "Eliminar muestras".	
Acciones del Actor	**Respuesta del Sistema**
3.1. Si el técnico indica que no desea eliminar la muestra entonces sale de la sección.	
Prototipo Sección "Eliminar muestra"	
Sección "Actualizar muestras".	
Acciones del Actor	**Respuesta del Sistema**
1. El técnico indica la muestra que desea actualizar.	2. El sistema da la posibilidad de actualizar los campos correspondientes a la muestra indicada. • Código Externo • Procedencia • Folio • Tipo de Muestra • Estado de la Muestra • Fecha de Extracción • Fecha de Entrada

	• Paciente • Registro de Estudio • Casilla
3. El técnico modifica los campos.	4. El sistema verifica que los campos necesarios estén llenos.
	5. El sistema verifica que los datos introducidos sean correctos.
	6. El sistema actualiza los datos y la ubicación de la muestra y lanza un mensaje informando que los cambios se han registrado satisfactoriamente.
	7. El sistema muestra el listado actualizado de las muestras.
Flujo alternativo Sección "Actualizar muestras".	
Acciones del Actor	**Respuesta del Sistema**
	5.1. Si el técnico dejó algún campo necesario vacío entonces el sistema emite un mensaje de alerta indicando que existen campos necesarios vacíos.
	6.1 Si el técnico indicó un valor incorrecto entonces el sistema emite un mensaje de alerta indicando el error.
Prototipo Sección "Actualizar muestras"	
Sección "Buscar muestras".	
Acciones del Actor	**Respuesta del Sistema**
	1. El sistema muestra la posibilidad de introducir los parámetros para la búsqueda de una muestra. • Registro • Código Externo • Procedencia • Tipo de Muestra • Estado de la Muestra • Casilla • Paciente
2. El técnico introduce el valor en los campos por los que desea realizar la búsqueda.	3. El sistema verifica que los datos introducidos sean correctos.

	4. El sistema visualiza las muestras que coinciden con los parámetros señalados.
Flujo alternativo Sección "Buscar muestras".	
Acciones del Actor	**Respuesta del Sistema**
	4.1. Si el técnico indicó un valor incorrecto, el sistema emite un mensaje de alerta indicando el error.
	4.2. Si el técnico no indicó ningún valor para la búsqueda se muestra un listado con las muestras existentes. No se especifica ningún criterio de búsqueda.
	4.3. Si el sistema no encuentra ninguna muestra que coincida con los parámetros señalados muestra un mensaje indicando que no existen muestras con esas características.
Prototipo Sección "Buscar muestras"	

Tabla 2.4 Descripción del CUS "Gestionar Muestra"

Descripción del caso de uso "gestionar almacén"

Nombre del Caso de Uso	Gestionar Almacén
Actores	Administrador del Sistema (inicia)
Propósito	Insertar, actualizar, buscar o eliminar un almacén.
Resumen	El CU inicia cuando el administrador desea insertar, actualizar, buscar, eliminar o visualizar información de un almacén y luego se realiza alguna de estas acciones actualizándose el listado de almacenes.
Referencias	R19, R20, R21, R22
Precondiciones	El administrador está autenticado en el sistema.
Poscondiciones	Se actualiza el listado de almacenes.
Prioridad	Crítico
Curso Normal de los Eventos	
Acciones del Actor	**Respuesta del sistema.**
1. El administrador indica la opción Configuración Avanzada.	2. El sistema visualiza los almacenes existentes brindando la posibilidad de insertar, eliminar, buscar, editar ó visualizar la información de un almacén.

3. El administrador realiza una de las siguientes operaciones: • Insertar un almacén. • Eliminar un almacén existente. • Buscar un almacén. • Actualizar un almacén existente. • Visualizar información de un almacén.	4. El sistema en dependencia de la operación indicada por el administrador realiza lo siguiente: • Si el administrador desea insertar un nuevo almacén entonces se ejecuta la sección “Insertar Almacén”. • Si el administrador desea eliminar un almacén existente entonces se ejecuta la sección “Eliminar Almacén”. • Si el administrador desea buscar un almacén entonces se ejecuta la sección “Buscar Almacén”. • Si el administrador desea actualizar los datos de un almacén entonces se ejecuta la sección “Actualizar Almacén”. • Si el administrador desea visualizar la información referente a un almacén entonces se ejecuta la sección “Información”
Sección “Insertar almacén”.	
Acciones del Actor	**Respuesta del Sistema**
1. El administrador ingresa los siguientes datos del almacén: • No. Almacén • Cantidad de Niveles • Temperatura • País de Origen	2. El sistema verifica que todos los campos necesarios estén llenos.
	3. El sistema verifica que los datos introducidos sean correctos.
	4. El sistema muestra un mensaje indicando que el almacén ha sido registrado satisfactoriamente.
	5. El sistema visualiza el listado actualizado de los almacenes.
Flujo alternativo Sección “Insertar Almacén”.	
Acciones del Actor	**Respuesta del Sistema**
	a. Si el administrador dejó algún campo necesario vacío entonces el sistema emite un mensaje de alerta indicando que los campos están vacíos.
	4.1 Si el administrador introdujo valores incorrectos entonces el sistema emite un mensaje de alerta indicando el error.
Prototipo Sección “Insertar almacén”	

Sección "Eliminar almacén".	
Acciones del Actor	**Respuesta del Sistema**
1. El administrador indica del listado mostrado el almacén que desea eliminar.	2. El sistema muestra un mensaje de alerta indicando si desea eliminar el almacén.
3. El administrador indica que desea eliminar el almacén.	4. El sistema verifica que no existan muestras o niveles en el Almacén.
	5. El sistema elimina el almacén indicado.
	6. El sistema muestra un mensaje informando que el almacén ha sido eliminado satisfactoriamente.
	7. El sistema muestra el listado actualizado.
Flujo alternativo Sección "Eliminar almacén".	
Acciones del Actor	**Respuesta del Sistema**
3.2. Si el técnico indica que no desea eliminar la muestra entonces sale de la sección.	
	5.1 El sistema muestra un mensaje indicando que no se puede eliminar el almacén porque existen muestras o niveles en el mismo.
Prototipo Sección "Eliminar almacén"	
Sección "Actualizar almacén".	
Acciones del Actor	**Respuesta del Sistema**
1. El administrador indica el almacén que desea actualizar.	2. El sistema posibilita actualizar los campos correspondientes al almacén. • No. Almacén • Cantidad de Niveles • Temperatura • País de Origen
3. El administrador modifica los campos.	4. El sistema verifica que los campos necesarios estén llenos.
	5. El sistema verifica que los datos introducidos sean correctos.
	6. El sistema actualiza los datos e informa un mensaje informando que los cambios se han registrado.
	7. El sistema muestra el listado actualizado.

Flujo alternativo Sección "Actualizar almacén".	
Acciones del Actor	**Respuesta del Sistema**
	5.2. Si el administrador dejó algún campo necesario vacío entonces el sistema emite un mensaje de alerta indicando que existen campos necesarios vacíos.
	6.2 Si el técnico indicó un valor incorrecto entonces el sistema emite un mensaje de alerta indicando el error.
Prototipo Sección "Actualizar almacén"	
Sección "Buscar almacén".	
Acciones del Actor	**Respuesta del Sistema**
	1. El sistema muestra la posibilidad de introducir los parámetros para la búsqueda de una muestra. • No. Almacén • Temperatura • País de Origen
2. El administrador introduce el valor en los campos por los que desea realizar la búsqueda.	3. El sistema verifica que los datos introducidos sean correctos.
	4. El sistema visualiza los almacenes que coinciden con los parámetros señalados.
Flujo alternativo Sección "Buscar almacén".	
Acciones del Actor	**Respuesta del Sistema**
	4.1 Si el técnico indicó un valor incorrecto entonces el sistema emite un mensaje de alerta indicando el error.
	4.2 Si el sistema no encuentra ningún almacén que coincida con los parámetros señalados muestra un mensaje indicando que no existen almacenes con esas características.
Prototipo Sección "Buscar almacén"	

Sección "Visualizar Información".	
Acciones del Actor	**Respuesta del Sistema**
1. El administrador indica del listado mostrado el almacén del cual desea ver la información.	2. El sistema muestra la información asociada al almacén.

Tabla 2.5 Descripción del CUS "Gestionar Almacén"

Descripción de casos de uso "Administrar Resultados"

Nombre del Caso de Uso	Administrar Resultados
Actores	Genetista
Propósito	Actualizar o visualizar un resultado asociado a una muestra existente.
Resumen	El CU inicia cuando el genetista desea actualizar o visualizar un resultado. El sistema muestra un interfaz con las muestras existentes dando la posibilidad de actualizar o visualizar el resultado del estudio a una muestra seleccionada.
Referencias	R7, R8, R9
Precondiciones	El genetista está autenticado en el sistema y selecciona una muestra.
Poscondiciones	Se actualiza o visualiza un resultado asociado a una muestra existente.
Prioridad	Crítico

Curso Normal de los Eventos	
Acciones del Actor	**Respuesta del sistema.**
	1. El sistema brinda la posibilidad de actualizar o visualizar el resultado.
2. El genetista ejecuta una de las siguientes operaciones: • Resultado • Información	3. El sistema en dependencia de la operación indicada por el genetista realiza lo siguiente: • Si el genetista desea actualizar el resultado entonces se ejecuta la sección "Actualizar Resultado". • Si el genetista desea visualizar el resultado entonces se ejecuta la sección "Visualizar Resultado".

Sección "Actualizar Resultado".	
Acciones del Actor	**Respuesta del Sistema**
	1. El sistema muestra una nueva interfaz con los campos a llenar del resultado a modificar.
2. El genetista modifica los campos del resultado.	3. El sistema verifica que los datos introducidos sean correctos.
	4. El sistema muestra un mensaje indicando que el resultado ha sido modificado satisfactoriamente.
	5. El sistema muestra los datos de la muestra.
Flujo alternativo Sección "Actualizar Resultado".	
Acciones del Actor	**Respuesta del Sistema**
	4.1 Si el genetista introdujo valores incorrectos entonces el sistema emite un mensaje de alerta indicando el error.
Prototipo Sección "Actualizar Resultado"	
Sección "Visualizar Resultado".	
Acciones del Actor	**Respuesta del Sistema**
	1. El sistema visualiza una nueva interfaz con los datos de la muestra y los resultados asociados a la misma. Dando nuevamente la posibilidad al usuario de editar los resultados de la muestra.
Prototipo Sección "Visualizar Resultado"	

Tabla 2.6 Descripción del CUS "Administrar Resultados"

Descripción de casos de uso "Autenticar Usuario".

Nombre del Caso de Uso	Autenticar Usuario
Actores	Genetista, Técnico, Administrador del Sistema
Propósito	Permitir que sólo los médicos registrados en el sistema puedan gestionar la información relacionada a la gestión de las muestras de ADN en el sistema.
Resumen	El genetista, administrador ó técnico introduce su usuario y contraseña. El sistema verifica que es usuario autorizado y en caso positivo brinda acceso a las funcionalidades según sus privilegios.
Referencias	R1

Precondiciones	Un administrador, genetista ó técnico solicita autenticarse.
Poscondiciones	Se le brinda acceso o no a las funcionalidades según los privilegios del usuario.
Prioridad	Crítico
Curso Normal de los Eventos	
Acciones del Actor	**Respuesta del sistema.**
1. El genetista indica datos para acceder al sistema: • Nombre de usuario • Contraseña	2. El sistema verifica si los datos introducidos son válidos.
	3. El sistema brinda acceso a las funcionalidades según los privilegios del médico en cuestión.
Flujo Alterno	
Acciones del Actor	**Respuesta del sistema.**
	3.1 El sistema indica un mensaje de error y no brinda acceso a las funcionalidades del sistema.
Prototipo	

Tabla 2.7 Descripción del CUS "Autenticar Usuario".

Conclusiones parciales

En este capítulo se enunciaron los objetivos estratégicos de la organización y el flujo actual del negocio. Se determinaron los casos de uso y se representaron los diagramas de CU del Negocio y Sistema. Se definieron los actores y trabajadores del negocio, de los cuales se realizaron los diagramas necesarios para la correcta modelación del negocio. Se mostraron las funcionalidades del sistema con el diagrama de casos de uso del sistema el cual marca el amplio alcance del mismo, en perspectivas del análisis y diseño de la aplicación.

Fueron definidos 10 requisitos no funcionales y 30 requisitos funcionales agrupados en 10 Casos de Usos de ellos 4 arquitectónicamente significativos los cuales fueron descritos.

CAPÍTULO 3: DISEÑO DEL SISTEMA

Introducción

En el desarrollo de toda aplicación el diseño constituye un importante paso ya que en él se realiza un refinamiento del análisis y se tienen en cuenta los requisitos no funcionales, o sea el cómo cumple el sistema los objetivos definidos. El resultado más significativo de este flujo de trabajo es el modelo de diseño. En el presente capítulo se muestran los diagramas de clases del diseño asociados a los casos de uso crítico descritos en el capítulo anterior, los restantes diagramas se incluyen en la carpeta de archivos del proyecto. Igualmente se refleja la arquitectura empleada y algunos ejemplos de los patrones de diseño utilizados.

3.1 Definición de la Arquitectura

“Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software.

En otras palabras, los patrones de diseño, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares.

Existen varios patrones de diseño popularmente conocidos, los cuales se clasifican como se muestra a continuación:

- **Patrones Creacionales:** Inicialización y configuración de objetos.
- **Patrones Estructurales:** Separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes.
- **Patrones de Comportamiento:** Más que describir objetos o clases, describen la comunicación entre ellos. “ (23)

El desarrollo de la presente aplicación informática utiliza el framework Symfony, el cual está basado en el clásico patrón Modelo Vista Controlador (MVC), que está formado por tres niveles:

- El modelo, que representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- La vista, quien transforma el modelo en una página web que permite al usuario interactuar con ella.

- El controlador, encargado de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

Estos niveles estarán regidos por patrones de diseño, los cuales se explican a continuación:

En el Modelo:

El modelo se puede dividir en la capa de acceso a los datos y en la capa de abstracción de la base de datos, para lograr que las funciones que acceden a los datos no utilicen sentencias ni consultas que dependan de una base de datos, sino que utilicen otras funciones; para realizar las consultas, Symfony emplea los patrones de diseño *Active Record* y *Active Table*:

"Active Record: Representa de forma Orientada a Objetos los datos de una Base de Datos Relacional, modelo conocido también como ORM o "*Object-Relational Mapping*", definiendo interfaces sencillas para acceder y manipular esos datos.

Active Table: El Active Table es un objeto que puede persistir otros objetos en medios no volátiles, como bases de datos. La función de una clase que implementa este patrón es de comunicarse con la base de datos y regresarnos objetos como los tenemos definidos para nuestra aplicación." (24)

De esta manera si se hiciera necesario cambiar el sistema gestor de bases de datos, solamente se precisaría actualizar la capa de abstracción de la base de datos.

En la Vista:

La vista también puede aprovechar la separación de código, las páginas web suelen contener elementos que se muestran de forma idéntica a lo largo de toda la aplicación, por este motivo en Symfony la vista se separa en un layout y en una plantilla. Normalmente, el layout es global en toda la aplicación o al menos en un grupo de páginas, el contenido de la plantilla se integra en el layout, o si se mira desde el otro punto de vista, el layout decora la plantilla. Para realizar este comportamiento Symfony utiliza el patrón de diseño llamado **"***Decorator***"**.

En el Controlador:

El controlador es el componente que recibe los requerimientos, los envía a los elementos encargados de procesar la lógica y luego los envía nuevamente a la vista (esta vez incluyendo los datos obtenidos) a través del patrón de diseño "*Front Controller*".

El controlador normalmente se divide en un controlador frontal, que es único para cada aplicación, y las acciones, que incluyen el código específico del controlador de cada página. Una de las principales ventajas de utilizar un controlador frontal es que ofrece un punto de entrada único para toda la aplicación. Así, en caso de que sea necesario impedir el acceso a la aplicación, solamente es necesario editar el script correspondiente al controlador frontal, evitando de esta forma código duplicado y manejo no centralizado de vistas.

A continuación se muestra como se aplican los diferentes patrones de diseño en la aplicación, tanto los patrones GOF que se clasifican en 3 grandes categorías basadas en su propósito: creacionales, estructurales y de comportamiento, como los patrones GRASP que describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones.

Estos constituyen un apoyo para entender el diseño y aplica el razonamiento para el diseño de una forma sistemática, racional y aplicable.

3.1.1 Empleo de patrones GRASP en la aplicación

Se debe conocer la arquitectura del framework para una mejor comprensión del comportamiento que seguirán las clases contenidas en él y su estructura, la arquitectura brinda una vista panorámica del diseño de este sistema. Son muchos los patrones que se utilizan en la implementación con Symfony, a continuación se mencionan algunos ejemplos de los evidenciados, ubicándolos en las capas de Modelo y Control que plantea el patrón arquitectónico MVC.

Patrón Experto:

En el modelo de la arquitectura Symfony existen dos tipos de clases que son fundamentales:

- Las encargadas de la abstracción de datos. (realizan todas las operaciones con la BD).
- Las de acceso a datos. (interactúan con las clases de abstracción de datos y devuelven los objetos necesarios por los controladores).

Por cada tabla se generan 4 tablas: Clase, ClasePeer, BaseClase y BaseClasePeer (ejemplo de clases: Muestra, MuestraPeer, BaseMuestra, BaseMuestraPeer). Las clases a las que el framework añade el sufijo “Peer” trabajan directamente con la base de datos y por lo tanto son las encargadas de la abstracción utilizando Propel como ORM, en ellas se encuentran los atributos necesarios para este proceso, de ahí la necesidad de que implementen la responsabilidad de efectuar las operaciones con la base de datos, aplicando de esta manera el patrón experto.

Patrón Creador:

Con Symfony se pueden crear objetos de varias formas, mayormente mediante métodos estáticos, para ver un ejemplo de esto se debe analizar cuando una clase "*action*" desea crear una instancia de una clase modelo, en el ejemplo el "*action*" obtiene el identificador del objeto a crear para visualizar los datos del mismo, para esto necesita entonces, crear una instancia de éste. A continuación se muestra un ejemplo del empleo de este patrón en la aplicación en la acción "Ejecutar Resultado" del módulo Muestra.

```
36  public function executeResultados(sfWebRequest $request) {
37      $this->Muestra = MuestraPeer::retrieveByPK($request->getParameter('id'));
38      $this->formResultado = new ResultadoForm();
39
40      if ($this->getRequest()->getMethod() == sfRequest::POST) {
41          $id = $request->getParameter('id');
42          $estado = $request->getParameter('estado_muestra');
43          $result_tipo = $request->getParameter('tipo_resultado_id');
44          $informe_result = $request->getParameter('resultado');
45
46          $muestra = MuestraPeer::retrieveByPK($id);
47
48          $muestra->setEstadoMuestra($estado);
49          $muestra->setTipoResultadoId($result_tipo);
50          $muestra->setResultado($informe_result);
51          $muestra->save();
52
```

Figura 3.1 Ejemplo del empleo del patrón Creador.

Patrón Bajo Acoplamiento:

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

Alta Cohesión:

Symfony permite la organización del trabajo en cuanto a la estructura del proyecto, esto proporciona crear y trabajar con clases con una alta cohesión durante el desarrollo de la aplicación. Un ejemplo de esto se evidencia en la clase "muestraActions**"**, la cual está formada por diferentes funcionalidades para gestionar las muestras de ADN que se encuentran estrechamente relacionadas, teniendo un sentido común y un propósito único, siendo las mismas las encargadas de controlar las acciones de las plantillas correspondientes a las diferentes funcionalidades del módulo.

En la figura 3.2 se muestra un ejemplo de cómo en una misma clase se agrupan todas las funcionalidades de un mismo módulo logrando una alta cohesión con las páginas de las plantillas correspondientes.

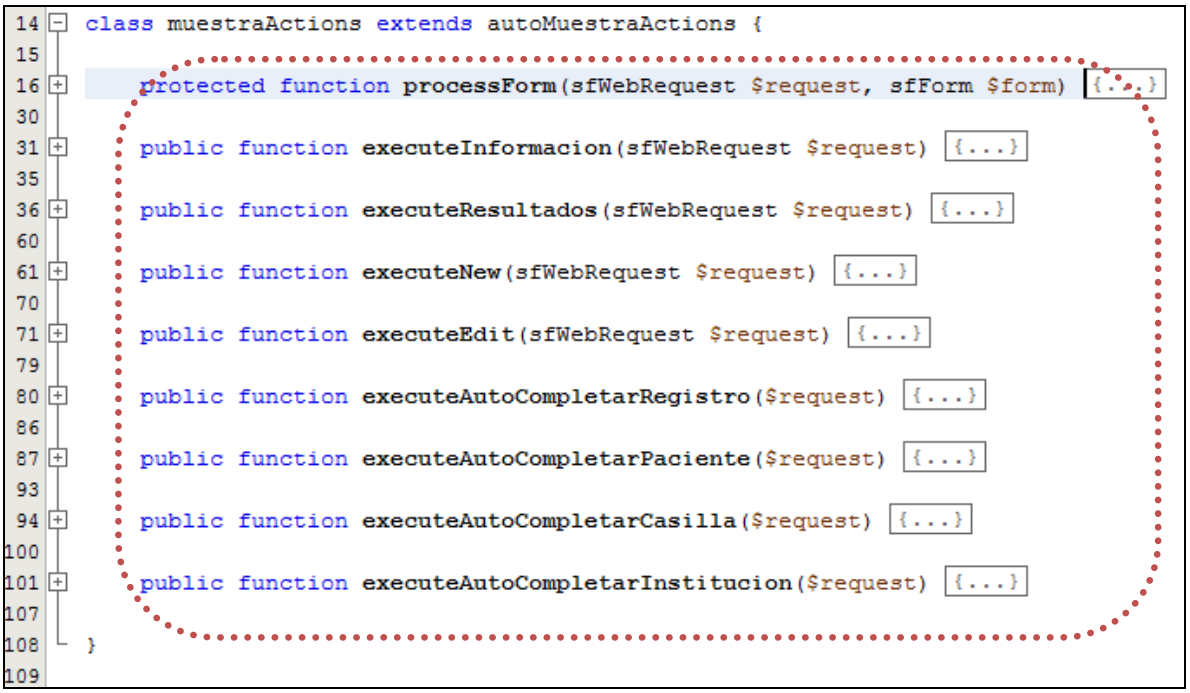

Figura 3.2 clase muestraActions. Ejemplo del empleo del patrón Alta Cohesión.

Patrón Controlador:

El controlador se encarga de asignar la responsabilidad de controlar el flujo de eventos del sistema a clases específicas con las que mantiene un modelo de alta cohesión, esto facilita la centralización de actividades (validaciones, seguridad.). Un ejemplo del patrón controlador se puede ver desde la clase "*sfFrontController*", "*sfWebFrontController*", "*sfContext*", los "*actions*", y el "*index.php*" del ambiente.

Symfony implementa el patrón "*Front Controller*" (Controlador frontal), en el cual existen varias clases controladoras que forman un flujo para atender las peticiones del usuario y de otras clases. La arquitectura del framework (MVC) ayuda desde el principio, existiendo una capa específicamente para los controladores, que son el núcleo del mismo, el puesto de mando.

3.1.2 Empleo de patrones GOF en la aplicación

Patrón Prototype:

La aplicación de este patrón se manifiesta en todas las clases del modelo generadas por el framework, específicamente en el método "*copy*" auxiliándose del método "*copyInto*" el cual permite que estas clases se puedan clonar. Estos métodos se encuentran en todas las clases "BaseClase" del modelo.

A continuación se muestra un ejemplo del empleo del Patrón Prototype en el módulo Paciente.

```
public function copyInto($copyObj, $deepCopy = false)
{
        $copyObj->setProvinciaId($this->provincia_id);
        $copyObj->setMunicipioId($this->municipio_id);
        $copyObj->setNombre($this->nombre);
        $copyObj->setPrimerApellido($this->primer_apellido);
        $copyObj->setSegundoApellido($this->segundo_apellido);
        $copyObj->setSexo($this->sexo);
        $copyObj->setEdad($this->edad);
        $copyObj->setPais($this->pais);

        if ($deepCopy) {
                // important: temporarily setNew(false) because this
                // the getter/setter methods for fkey referrer objec
                $copyObj->setNew(false);

                foreach ($this->getMuestras() as $relObj) {
                        if ($relObj !== $this) {  // ensure that we
                                $copyObj->addMuestra($relObj->copy($
                        }
                }
        } // if ($deepCopy)
        $copyObj->setNew(true);
        $copyObj->setId(NULL); // this is a auto-increment column, s
}
```

Figura 3.3 Método CopyInto de la clase BasePaciente. Empleo del patrón Prototype.

Patrón Decorador:

En este método de la clase abstracta "*sfView*" padre de todas las vistas, tiene cada una un decorador para permitir añadir funcionalidades a las vistas dinámicamente. El archivo llamado "*layout.php*" contiene el layout de la página, este archivo se le denomina plantilla global, la cual almacena el código HTML que es común a todas las páginas de la aplicación, el contenido de la plantilla se integra en el Layout.

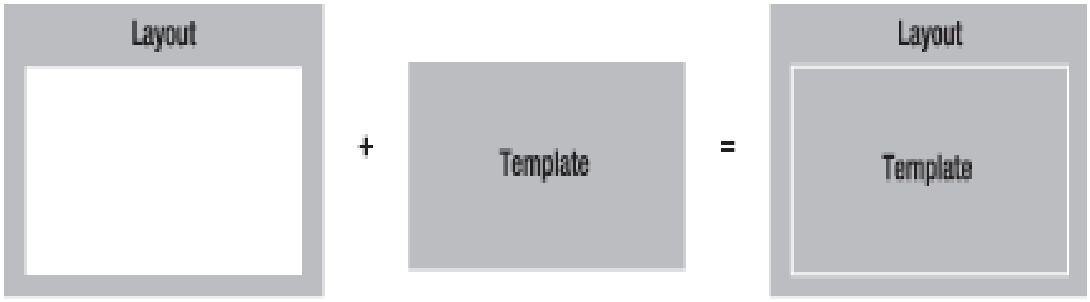

Figura. 3.4 Plantilla decorada con un Layout

Patrón Singleton:

La clase "*sfRouting*" es una de es una de las que se encuentran en la capa Controlador del framework, esta clase es muy utilizada porque es la encargada de enrutar todas las peticiones que se hagan a la aplicación. El ejemplo tiene solamente un punto de creación, el cual es estático lo que permite entonces la aplicación perfecta del patrón.

```
public function initialize(sfEventDispatcher $dispatcher, sfCache $cache = null, $options =
{
  $this->dispatcher = $dispatcher;

  $options['debug'] = isset($options['debug']) ? (boolean) $options['debug'] : false;

  // disable caching when in debug mode
  $this->cache = $options['debug'] ? null : $cache;

  $this->setDefaultParameter('module', isset($options['default_module']) ? $options['default
  $this->setDefaultParameter('action', isset($options['default_action']) ? $options['default

  if (!isset($options['logging']))
  {
    $options['logging'] = false;
  }

  if (!isset($options['context']))
  {
    $options['context'] = array();
  }

  $this->options = $options;

  $this->dispatcher->connect('user.change_culture', array($this, 'listenToChangeCultureEvent
  $this->dispatcher->connect('request.filter_parameters', array($this, 'filterParametersEven

  $this->loadConfiguration();
}
```

Figura. 3.5 Método initialize() de la Clase sfRouting

Patrón Command:

Se evidencia en la clase "*sfFrontWebController*", esta clase es la encargada de determinar cual módulo y acción debe responder a las solicitudes de los usuarios.

```
class sfFrontWebController extends sfWebController
{
    public function dispatch()
  {
    try
    {
      sfFilter::$filterCalled = array();
      // determine our module and action
      $request    = $this->context->getRequest();
      $moduleName = $request->getParameter('module');
      $actionName = $request->getParameter('action');

      if (empty($moduleName) || empty($actionName))
      {
        throw new sfError404Exception(sprintf('Empty module and/or action after
      }

      // make the first request
      $this->forward($moduleName, $actionName);
    }
    catch (sfException $e)
    {
      $e->printStackTrace();
    }
    catch (Exception $e)
    {
      sfException::createFromException($e)->printStackTrace();
    }
  }
}
```

Figura. 3.6 Muestra de la clase sfFrontWebController

Patrón Front Controller:

La aplicación de este patrón se manifiesta en la capa de controladores, donde existe una estructura de clases bien definida, una de sus principales funciones es definir un solo punto de entrada para las peticiones de los usuarios, permitiendo un mejor control del flujo de eventos del sistema. La clase "*sfController*" se encarga de decodificar la petición y transferirla a la acción correspondiente.

3.2 Vista Lógica

"La Vista Lógica describe el diseño más importante de las clases y su organización en paquetes y subsistemas, y la organización de éstos en capas. Es representada por uno o varios diagramas de

clases que son un subconjunto del modelo de diseño. Esta vista muestra cómo la funcionalidad es diseñada en el interior del sistema, en términos de la estructura estática y comportamiento dinámico del sistema". (25)

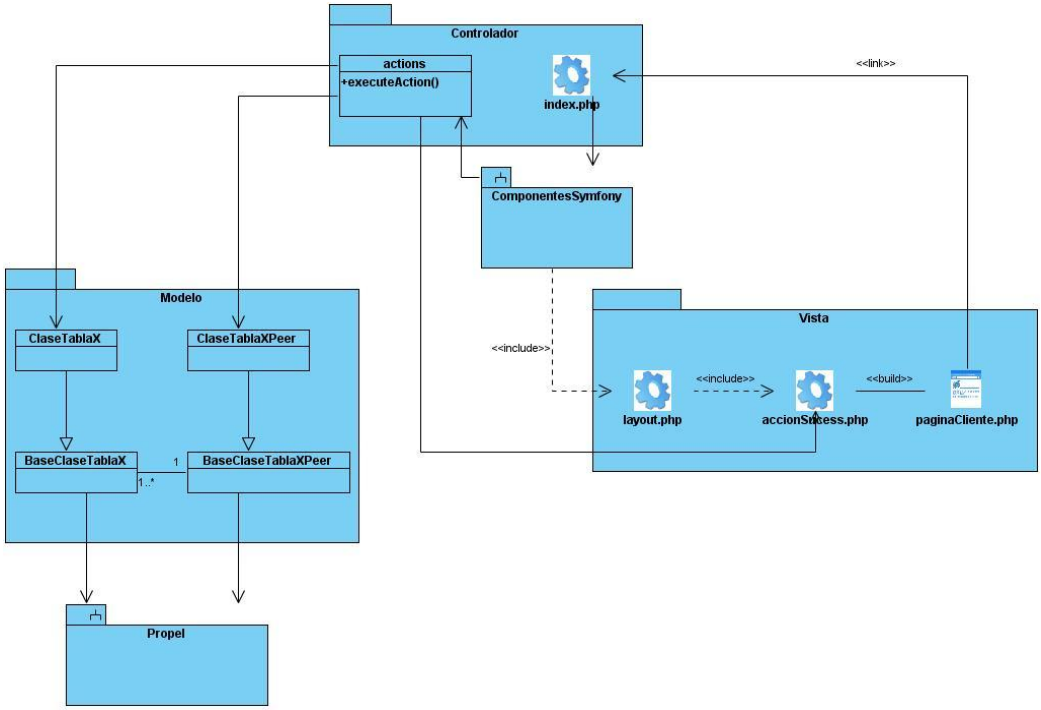

Figura 3.7 Diagrama de Vista Lógica

3.2.1 Diagramas de clases del diseño

"Un diagrama de clases del diseño representa las clases del diseño y sus objetos, así como los subsistemas del diseño". (25)

A continuación se describen cada uno de los paquetes y subsistemas que conforman cada uno de los diagramas de diseño del sistema.

Paquete controlador: Una parte importante de su trabajo es común a todos los controladores de la aplicación. Entre las tareas comunes se encuentran el manejo de las peticiones del usuario, la seguridad, cargar la configuración de la aplicación y otras tareas similares. Por este motivo, el controlador se ha dividido en un controlador frontal "*index.php*", que se encarga de realizar las tareas comunes y las acciones *"actions.php*", que incluyen el código específico del controlador de cada página. Habrá un "*actions.php*" por cada uno de los módulos pero el controlador frontal será el mismo para todo el sistema.

Paquete vista: Las páginas web suelen contener elementos que se muestran de forma idéntica a lo largo de toda la aplicación: cabeceras de la página, el layout genérico, el pie de página y la navegación global. En la mayor parte de las veces sólo cambia el interior de la página. Por este motivo, la vista se separa en un layout y en una plantilla (el nombre de las platillas está compuesto por el nombre de la acción que la origina seguido por sufijo SUCCESS). El layout será global en toda la aplicación o a la mayoría de las páginas. La plantilla sólo se encarga de visualizar las variables definidas en el paquete del controlador.

Paquete del modelo: Solo contiene las clases encargadas del acceso a los datos almacenados en el gestor de base de datos, las cuales utilizan el ORM (*Object Relational Model*) Propel para el acceso a los mismos. En función de lograr una mejor comprensión de los diagramas de clases de diseño se hace necesario las siguientes aclaraciones:

Las clases “BaseClase” y “BaseClasePeer”, tienen una relación de asociación sin navegabilidad porque ambas pueden crear instancias una de la otra. La multiplicidad expresa que una instancia de la clase “BaseClasePeer” puede crear 1 o varias instancias de la clase “BaseClase” cuando se ejecuta**.**

Los estereotipos de “*Server Page*” se utilizan para las páginas web que tienen lógica de negocio para diferenciarlas de las clases “*claseActions*” que no son páginas web sino clases con extensión .php. La relación de dependencia estereotipada <<*include*>> entre algunas clases está justificada precisamente por los elementos propios de la programación en PHP, una clase incluye el código de la otra.

Subsistema Propel: Su función es gestionar el acceso a la base de datos y gestionar el modelo del sistema. Implica que el acceso y la modificación de los datos almacenados en la base de datos se realicen mediante objetos, nunca de forma explícita, permitiendo un alto nivel de abstracción y fácil portabilidad. Propel tiene incluido tareas para generar automáticamente las sentencias SQL necesarias para crear las tablas de la base de datos. La librería Propel se encarga de esta generación automática, ya que crea el esqueleto o estructura básica de las clases y genera automáticamente el código necesario.

Subsistema Componente Symfony: Representa todas las clases del framework Symfony que serán utilizadas durante el funcionamiento del sistema. Dígase validadores de formularios, helpers de objetos y formularios, plantillas, componentes de seguridad, entre otros.

En la figura 3.3 se muestra el diagrama de cases del diseño correspondiente al caso de uso "**Gestionar Registro**". Para el estudio de los diagramas de clase del diseño del sistema remitirse la carpeta del expediente de proyecto.

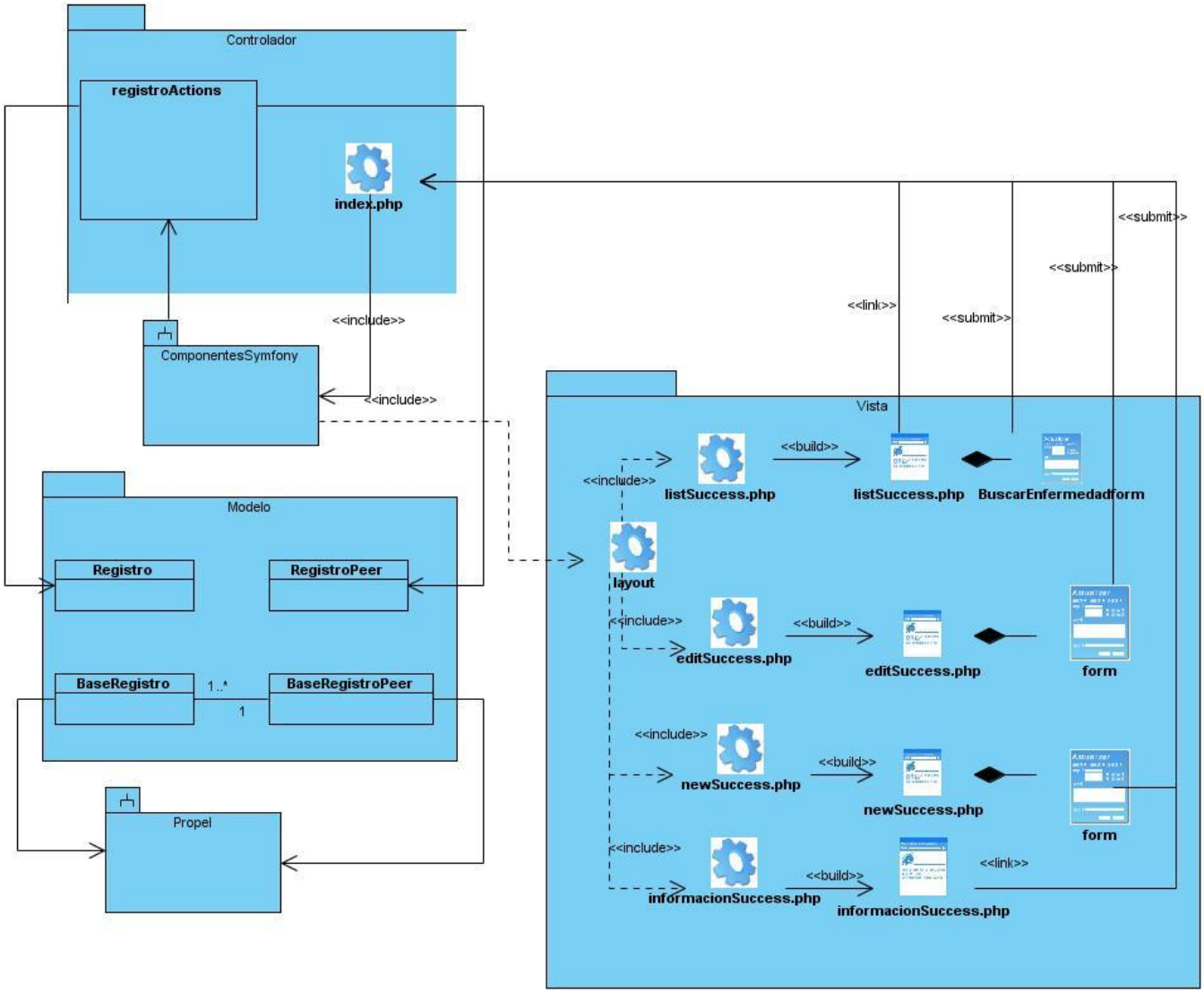

Figura 3.3. Diagrama de Clase del Diseño CU "Gestionar Registro"

3.2.2 Diagrama de Clases Persistentes

"La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Las clases persistentes referencian directamente las entidades lógicas y sus atributos". (11)

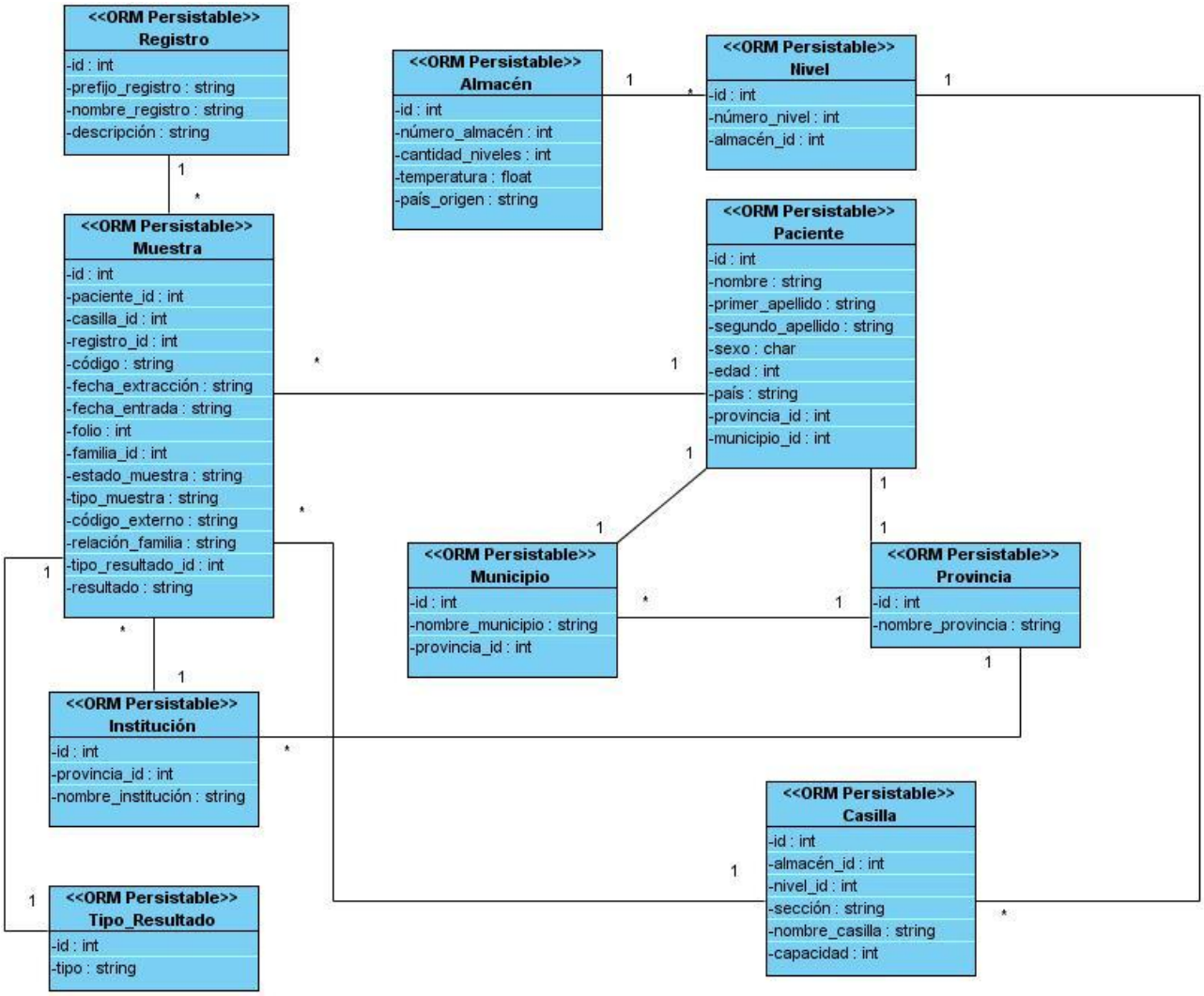

Figura 3.4. Diagrama de Clases Persistentes

3.2.3 Modelo de Datos

"El modelo de datos describe la representación lógica y física de la persistencia de los datos utilizados por la aplicación". (25)

El modelo de datos estará compuesto por las entidades que pasarán a ser las tablas de la base de datos que será utilizada por el sistema. Las entidades de color naranja se corresponden con las clases persistentes definidas anteriormente, las de color verde son las que fueron agregadas por el plugin *sfGuardPlugin* para gestionar la seguridad del sistema.

Para visualizar el modelo de datos remitirse a la figura #4 de la página de anexos.

3.3 Vista de Despliegue

"La vista de despliegue describe varios nodos físicos para las configuraciones más típicas de las plataformas y la asignación de las tareas de la vista de procesos a los nodos físicos. Describe la situación de los componentes en una posible implantación del sistema de acuerdo a los requisitos iníciales". (25)

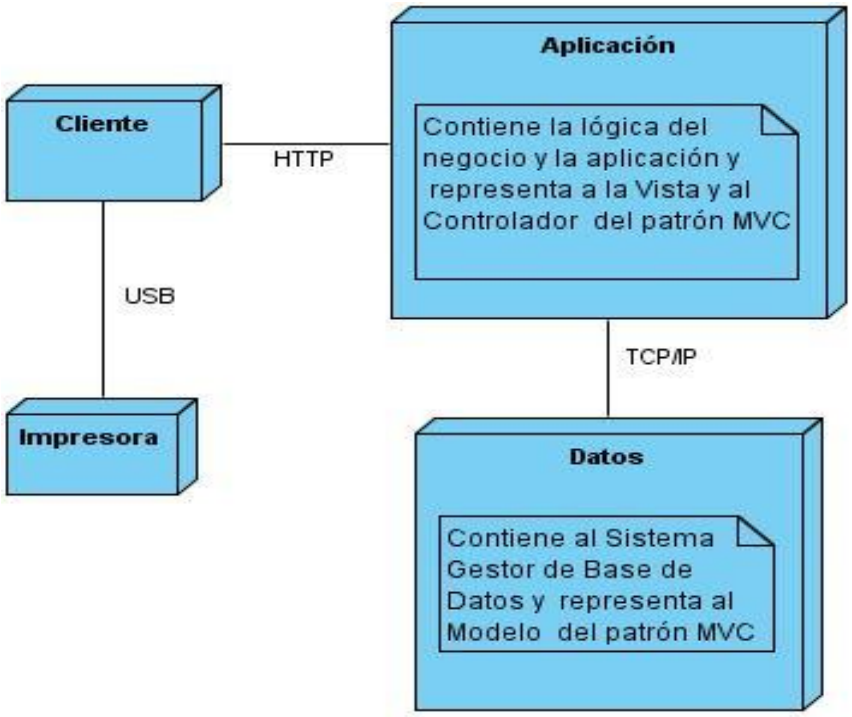

Figura 3.9. Diagrama de Despliegue

3.3.1 Descripción de los nodos físicos

"Un nodo es un objeto físico que representa un recurso informático, este recurso generalmente dispone de datos persistentes y capacidad de proceso. Las conexiones entre nodos muestran las líneas de comunicación con las que el sistema tendrá que interactuar". (25)

En el nodo **Cliente** se encontrará el sistema operativo Windows o Linux y los navegadores web Mozilla Firefox o Internet Explorer mediante los cuales los clientes tendrán acceso al sistema.

En el nodo **Aplicación** estarán agrupados los archivos a través de los cuales el usuario logra acceder al sistema, además se encuentra contenida toda la información específica de cada registro, sus clases; así como almacena además la configuración general del proyecto, las clases y librerías externas, todo el código común de las aplicaciones del proyecto y los plugins de instalación de la aplicación.

En el nodo **Datos** son almacenados los datos de la aplicación, se guarda el modelo de objetos del sistema. El nodo **Impresora** es un dispositivo externo conectado al nodo cliente usado para la impresión de documentos.

Conclusiones parciales

Luego del análisis de las características del sistema se llevó a cabo el refinamiento del diseño, pudiéndose tener una visión más clara del cumplimiento de los objetivos del sistema. Se logran representar los diagramas de clase del diseño de los diferentes Casos de Uso y se define y emplea como patrón arquitectónico el Patrón Modelo Vista Controlador.

Se ejemplificó el uso de algunos de los patrones de diseño y se obtuvo el diagrama de clases persistentes así como el modelo de datos de la aplicación. Por último se logró la representación de los nodos físicos de la aplicación a partir de la obtención del diagrama de despliegue.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBAS

Introducción

Si importante es el flujo de análisis y diseño en el desarrollo de cualquier aplicación no menos importante es la implementación y las pruebas realizadas para validar la solución. En el presente capítulo se presentan los diagramas de componentes elaborados a partir de los diagramas de clases de diseño y se dan elementos sobre las pruebas aplicadas a la aplicación.

4.1 Vista de Implementación

"La vista de implementación contiene la organización de los módulos en términos de paquetes y capas, pueden incluirse también la trazabilidad de la vista lógica. Esta vista toma en cuenta los requerimientos que facilitan la programación, los niveles de reutilización y las limitaciones impuestas por el entorno de desarrollo. Es representada por un diagrama de componentes o especificaciones de paquetes que son básicamente un subconjunto del modelo de despliegue. Para modelarla se dispone de dos elementos, los paquetes que representan una partición física del sistema y los componentes que representan la organización de los módulos de código fuente". (25)

Symfony establece una estructura de carpetas para sus proyectos y almacena los archivos del proyecto en una estructura estandarizada de tipo árbol. Algunos subdirectorios agrupan componentes propios de Symfony, que son clases que implementan el núcleo del framework, y por tanto no se modifican en la construcción del sistema. Dentro de un proyecto, las operaciones se agrupan de forma lógica en aplicaciones. Cada aplicación está formada por uno o más módulos.

Descripción de cada uno de los elementos que conforman la vista de implementación:

- **Apps:** Contiene un directorio por cada aplicación del proyecto.
- **Lib:** Almacena las clases y librerías externas, Se suele guardar todo el código común a todas las aplicaciones del proyecto. El subdirectorio *model* guarda el modelo de objetos del proyecto.
- **Config:** Almacena la configuración general del proyecto.
- **Web:** Contiene los únicos archivos accesibles desde Internet. Los únicos archivos accesibles desde Internet son los que se encuentran en este directorio.
- **Plugins:** Almacena los plugins instalados en la aplicación.

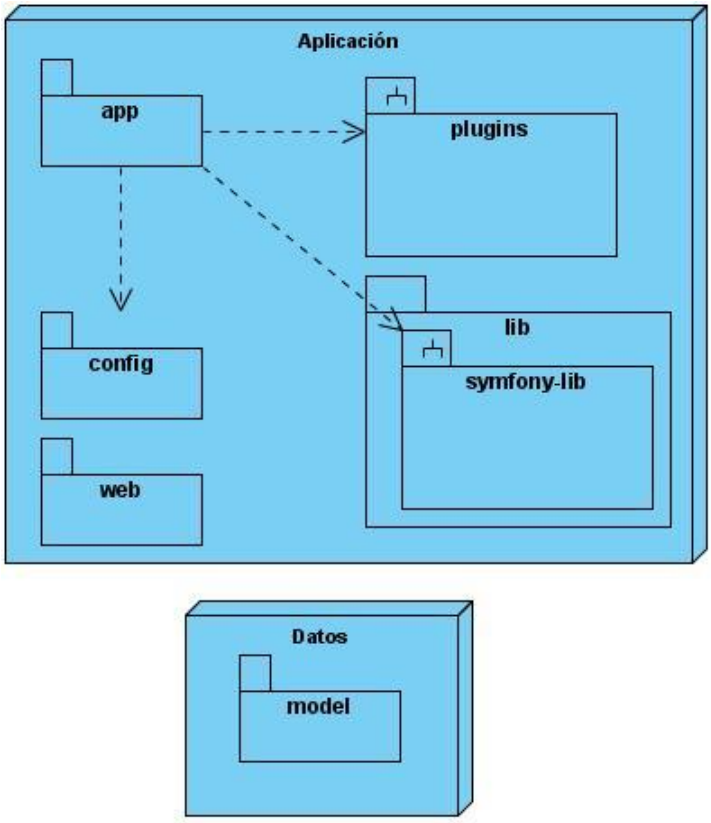

Figura 4.1. Vista de Implementación.

4.2 Diagrama de Componentes

"Los diagramas de componentes representan cómo un sistema es dividido en componentes y muestra las dependencias entre estos. Los componentes físicos incluyen archivos, cabeceras, librerías compartidas, módulos, ejecutables, o paquetes. Pueden ser usados para modelar y documentar cualquier arquitectura de sistema. Son utilizados para modelar la vista estática de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. Uno de los usos principales es que puede servir para ver qué componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema". (26)

A continuación se muestra el diagrama de componentes correspondiente al caso de uso "Gestionar Muestra" y la descripción de los principales elementos que lo conforman. Para visualizar el resto de los diagramas de componentes remitirse a la carpeta de expediente de proyecto.

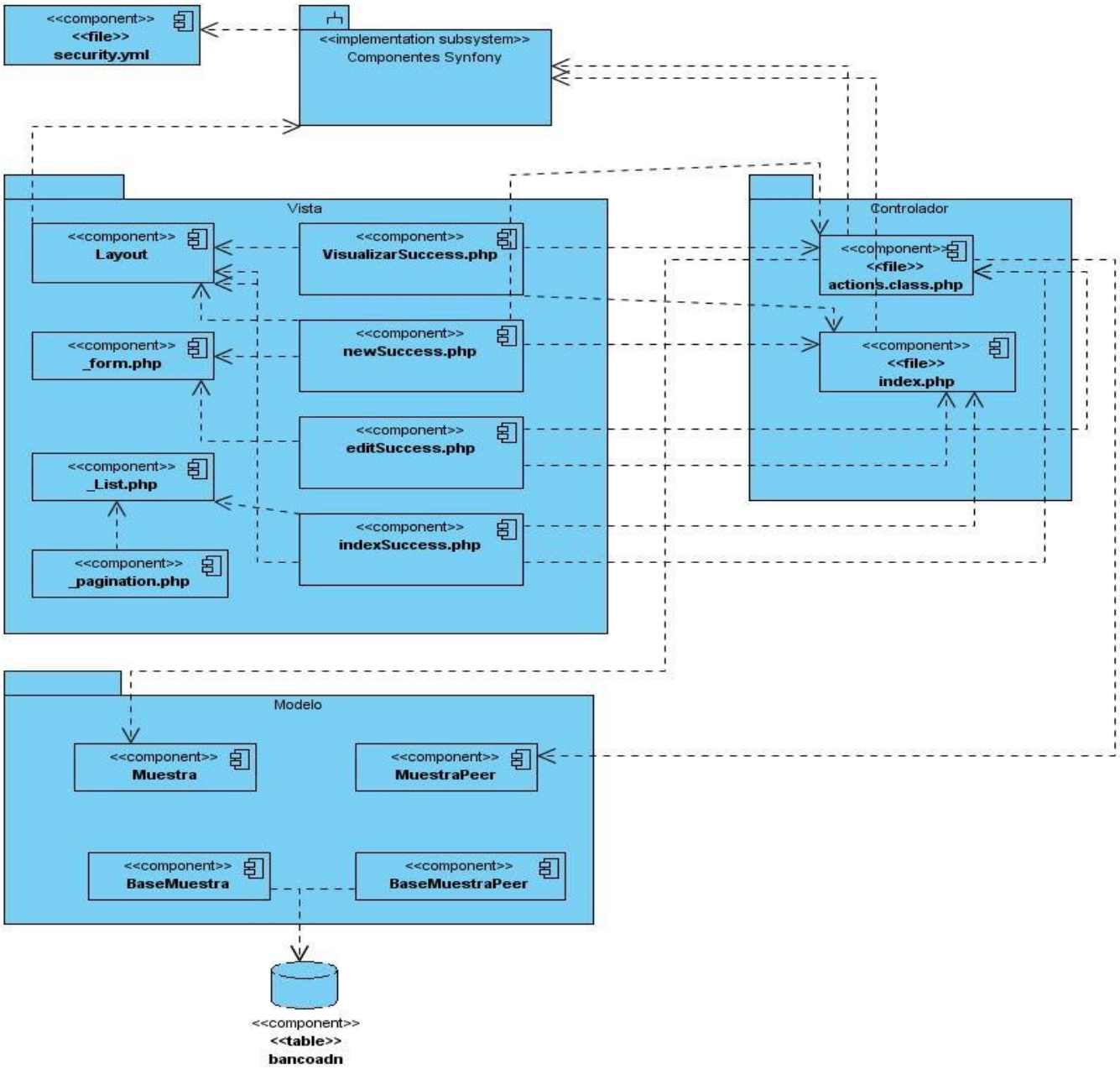

Figura 4.2. Diagrama de Componente CU "Gestionar Muestra"

Paquete Vista: Paquete que agrupa todos los componentes relacionados con la vista de la aplicación tales como:

- **Componente layout.php:** Componente que implementa la clase layout del diseño. Contiene los elementos que se muestran de forma idéntica a lo largo de toda la aplicación.

- **Componentes (Success):** Componentes que implementan el código correspondiente a cada plantilla que utiliza el módulo.

Paquete Controlador: Paquete que agrupa todos los componentes "*actions.class.php*" de cada uno de los módulos y el componente index.php, estos son implementados por las clases "*Actions*" del diseño. Estos componentes incluyen el código específico del controlador para cada página del módulo.

Componente security.yml: Archivo de configuración que permite restringir el acceso a determinadas acciones del módulo.

Paquete Modelo: Se especifican las clases generadas por el subsistema Propel que es el ORM que utiliza Symfony el cual proporciona persistencia para los objetos y un servicio de consultas. Propel a su vez utiliza el componente Creole como sistema de abstracción de la base de datos, sistema similar a los PDO (*PHP Data Object*) y proporciona una interfaz entre el código PHP y el código SQL de la base de datos, permitiendo cambiar fácilmente de sistema gestor de base de datos.

4.3 Seguridad de la Aplicación

La aplicación contará con un mecanismo de seguridad a nivel local donde para poder acceder a la aplicación los usuarios necesitarán estar registrados y autenticados previamente. Una vez insertado el nombre de usuario y la contraseña, el sistema comprueba si coincide con alguno de los usuarios registrado permitiendo el acceso según los permisos del tipo de usuario.

En Symfony, 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.

Los permisos fueron definidos mediante el uso de credenciales, para cada una de las acciones en cada módulo de la aplicación. Para ello se creó un archivo YAML, *security.yml* en el directorio "*config*" de cada módulo.

La seguridad es implementada a partir del plugin "*sfGuardPlugin*" el cual ofrece un modelo de seguridad basado en usuario, grupo y permisos, además de facilitar varios módulos para la administración de los mismos y otras opciones de gestión de usuarios más avanzadas que las que proporciona por defecto Symfony.

Para el acceso a la aplicación se definieron 3 roles según las funcionalidades que estos podrán realizar. Los roles definidos fueron: administrador del sistema, médico y técnico. Para cada rol fueron asignadas determinadas credenciales que regulan el acceso de los mismos a las diferentes acciones de los módulos.

4.4. Pruebas realizadas para validar la aplicación

Todo desarrollo de sistemas de software implica la realización de una serie de actividades predispuestas a incorporar errores. De ahí la importancia de realizar pruebas que validen la solución y permitan detectar estas incongruencias.

En la aplicación se realizaron pruebas a nivel de desarrollador y dentro de estas las pruebas de caja negra. Estas pruebas fueron realizadas con el objetivo de comprobar la operatividad de las funciones del sistema y que la entrada de datos por parte de los usuarios fuese la adecuada. De igual manera se comprueba que la repuesta del sistema es la esperada para cada funcionalidad.

A continuación se muestran el diseño del caso de prueba correspondiente la CUS “Gestionar Muestra”.

Descripción del Caso de Prueba asociado al CUS “Gestionar Muestras”

Nombre de la Sección	Escenario	Descripción	Flujo donde empieza
SC 1: Insertar Muestra	EC 1.1: El usuario selecciona con un clic la opción “Nuevo” e introduce los datos válidos de la nueva muestra.	El sistema muestra un mensaje informando que se ha creado la muestra y visualiza el listado de muestras actualizado.	Flujo Principal.
	EC 1.2: El usuario selecciona con un clic la opción “Nuevo” e introduce datos inválidos a la nueva muestra.	El sistema lanza un mensaje de error indicando los datos inválidos entrados y dando la posibilidad de rectificarlos.	Flujo Alterno.
	EC 1.3: El usuario selecciona con un clic la opción “Nuevo” dejando campos vacíos.	El sistema muestra un mensaje de error indicando los campos vacíos y dando la posibilidad de crear nuevamente la muestra.	Flujo Alterno.

SC 2: Eliminar Muestra	EC 2.1: El usuario selecciona con un clic la opción “Borrar” sobre una muestra y confirma que desea eliminar la muestra.	El sistema elimina la muestra seleccionada y visualiza el listado de muestras actualizado.	Flujo Principal
	EC 2.2: El usuario selecciona con un clic la opción “Borrar” sobre una muestra y confirma que no desea eliminar la muestra.	El sistema no elimina la muestra seleccionada y visualiza el listado de las muestras.	Flujo Alterno.
SC 3: Actualizar Muestra	EC 3.1: El usuario selecciona con un clic la opción “Editar” sobre una muestra e introduce los nuevos datos válidos de la muestra a modificar.	El sistema actualiza los datos de la muestra y muestra un mensaje informando que se han actualizado correctamente los datos, mostrando posteriormente el listado actualizado.	Flujo Principal.
	EC 3.2: El usuario selecciona con un clic la opción “Editar” sobre una muestra e introduce datos inválidos en la muestra a modificar.	El sistema lanza un mensaje de error indicando los datos inválidos entrados y dando la posibilidad de rectificarlos.	Flujo Alterno.
	EC 3.3: El usuario selecciona con un clic la opción “Editar” sobre una muestra dejando campos vacíos.	El sistema muestra un mensaje de error indicando los campos vacíos y dando la posibilidad de crear nuevamente la muestra.	Flujo Alterno.
SC 4: Buscar una muestra	EC 4.1: El usuario selecciona con un clic la opción “Búsqueda Avanzada” e introduce los parámetros correctos para la búsqueda.	El sistema muestra un listado con las muestras que coinciden con los criterios especificados.	Flujo Principal.
	EC 4.2: El usuario selecciona con un clic la opción “Búsqueda Avanzada” e introduce parámetros inválidos.	El sistema muestra un mensaje de error indicando los campos inválidos.	Flujo Alterno.

	EC 4.3: El usuario selecciona con un clic la opción “Búsqueda Avanzada” e introduce parámetros válidos.	El sistema no visualiza ningún listado y muestra “0 Resultados”.	Flujo Alterno.
	EC 4.4: El usuario selecciona con un clic la opción “Búsqueda Avanzada” y no introduce ningún criterio de búsqueda.	El sistema visualiza un listado con todas las muestras que pertenecen a ese usuario. No se especifican criterios de búsqueda.	Flujo Alterno.

Tabla 4.1Descripción del Caso de Prueba correspondiente al CUS “Gestionar Muestra”.

Luego de la descripción de las secciones y los escenarios se describen las variables correspondientes a este CUS.

Descripción de las variables del CUS “Gestionar Muestras”

No.	Nombre del Campo	Clasificación	Valor Nulo	Descripción
1	Nuevo	Botón	No	Se visualiza formulario para insertar una nueva muestra.
2	Borrar	Botón	No	Se elimina una muestra seleccionada.
3	Editar	Botón	No	Se visualiza formulario para modificar una muestra.
4	Buscar	Botón	No	Se buscan muestras según criterios especificados.
5	Datos	Campo de texto	Si	Se introducen los datos de las muestras.
6	Procedencia	Campo autocompletable	Si	Se selecciona una de las instituciones de la lista.
7	Tipo de Muestra	Lista desplegable	No	Se selecciona un tipo de muestra.
8	Estado de la Muestra	Lista desplegable	No	Se selecciona un estado de la muestra.
9	Paciente	Campo autocompletable	Si	Se selecciona uno de los pacientes de la lista.
10	Registro	Campo autocompletable	Si	Se selecciona uno de los registros de la lista.
11	Casilla	Campo autocompletable	Si	Se selecciona una de las casillas de la lista.

Tabla 4.2 Descripción de las variables del CUS “Gestionar Muestra”.

A continuación se muestran las matrices relacionadas con el diseño, donde se probó y evaluó la validez de los datos asociado al CUS Gestionar Muestra.

Matriz de datos. CUS "Gestionar Muestras". SC 1: Insertar Muestra.

ID EC	Escenario	Variables								Respuesta del sistema	Resultado de la Prueba
		1	5	6	7	8	9	10	11		
1.1	El usuario introduce los datos de la muestra correctamente.	V	V	V	V	V	V	V	V	El sistema muestra un mensaje informando que la muestra se ha insertado correctamente y visualiza el formulario de editar muestra.	Insatisfactoria
1.2	El usuario introduce datos de la muestra inválidos.	V	I	V	V	V	V	V	V	El sistema muestra un mensaje de error, indica los campos incorrectos y permite al usuario volver a introducirlos.	Satisfactoria
1.3	El usuario no introduce datos dejando campos en blanco.	V	N/A	V	V	V	V	V	V	El sistema muestra un mensaje de error, indica los campos requeridos y permite al usuario volver a introducirlos.	Satisfactoria

Tabla 4.3 Matriz de Datos. Sección 1: "Insertar Muestra".

Matriz de datos. CUS "Gestionar Muestras". SC 2: Eliminar Muestra.

ID EC	Escenario	Variables	Respuesta del sistema	Resultado de la Prueba
		2		
2.1	El usuario confirma que desea eliminar la muestra.	V	El sistema muestra un mensaje de confirmación y a la respuesta afirmativa del usuario elimina la muestra y visualiza el listado actualizado.	Satisfactoria.
2.2	El usuario confirma que no desea eliminar la muestra.	V	El sistema muestra un mensaje de confirmación y a la respuesta negativa del usuario no elimina la muestra y visualiza el listado.	Satisfactoria

Tabla 4.4 Matriz de Datos. Sección 2: "Eliminar Muestra".

Matriz de datos. CUS "Gestionar Muestras". SC 3: Editar Muestra.

ID EC	Escenario	Variables								Respuesta del sistema	Resultado de la Prueba
		3	5	6	7	8	9	10	11		
3.1	El usuario introduce los datos de la muestra correctamente.	V	V	V	V	V	V	V	V	El sistema muestra un mensaje informando que la muestra se ha actualizado correctamente y visualiza el listado actualizado.	Satisfactoria
3.2	El usuario introduce datos de la muestra inválidos.	V	I	V	V	V	V	V	V	El sistema muestra un mensaje de error, indica los campos incorrectos y permite al usuario volver a introducirlos.	Satisfactoria
3.3	El usuario no introduce datos dejando campos en blanco.	V	N/A	V	V	V	V	V	V	El sistema muestra un mensaje de error, indica los campos requeridos y permite al usuario volver a introducirlos.	Satisfactoria

Tabla 4.5 Matriz de Datos. Sección 3: "Editar Muestra".

Matriz de datos. CUS "Gestionar Muestras". SC 4: Buscar Muestra.

ID EC	Escenario	Variables								Respuesta del sistema	Resultado de la Prueba
		4	5	6	7	8	9	10	11		
4.1	El usuario introduce los datos de la búsqueda correctamente.	V	V	V	V	V	V	V	V	El sistema visualiza un listado con las muestras que coinciden con los criterios.	Satisfactoria
4.2	El usuario introduce datos de búsqueda inválidos.	V	I	V	V	V	V	V	V	El sistema muestra un mensaje de error, indica los campos incorrectos y permite al usuario volver a introducirlos.	Satisfactoria

4.3	El usuario introduce los datos de la búsqueda correctamente.	V	V	V	V	V	V	V	V	El sistema no encuentra muestras que coinciden con los parámetros, y muestra el mensaje de “0 Resultados”.	Satisfactoria
4.3	El usuario no especifica datos dejando los campos en blanco.	V	N/A	V	V	V	V	V	V	El sistema visualiza un listado con las muestras que coinciden que pertenecen al usuario. No se especifican criterios de búsqueda.	Satisfactoria

Tabla 4.6 Matriz de Datos. Sección 4: “Buscar Muestra”.

La siguiente tabla muestra las no conformidades detectadas durante la aplicación de las pruebas en el CUS “Gestionar Muestra”, y como estas fueron mitigadas.

No.	No conformidad	Aspecto correspondiente	Etapa de Detección	Significa-tiva	No Signifi-cativa	Estado NC	Respuesta
1	En el CUS “Gestionar Muestra” en la SC1, cuando se introducen los datos de la muestra, una vez creada satisfactoriamente no se visualiza el listado actualizado sino la página de editar de dicha muestra creada.	Vista “Insertar Muestra“	Pruebas de funcionalidad.		x	PD 30/05/11 Resuelta 4/06/11	El sistema una vez insertada una nueva muestra visualiza el listado actualizado de las mismas.

Tabla 4.7 Tabla de No Conformidades. Caso Prueba: “Gestionar Muestra”.

Conclusiones parciales

Luego del presente capítulo se obtuvo los diagramas de componentes del sistema. Se diseñó el mapa de navegación para una mejor visualización de la navegabilidad. Se fundamentó sobre la utilización del plugin sfGuardPlugins para la gestión de la seguridad en la aplicación donde se definieron 3 roles principales para la autorización de los usuarios: el rol de Administrador del sistema, el técnico y el médico.

Por último, se realizaron pruebas a nivel de desarrollador, dentro de estas las pruebas de caja negra para comprobar la operatividad de las principales funcionalidades. Fueron diseñados 10 casos de pruebas, donde en 5 de ellos fueron encontrados un total de 7 no conformidades las cuales fueron resueltas satisfactoriamente.

CONCLUSIONES GENERALES

El análisis del proceso de gestión de muestras en biobancos y en particular el Banco de ADN del Centro Nacional de Genética Médica permitió la identificación de 30 requisitos funcionales y 10 no funcionales necesarios para el desarrollo del sistema adaptado a las características propias del centro.

La aplicación de la arquitectura del sistema basada en el patrón MVC y a través de un marco de trabajo robusto como Symfony guió el diseño del sistema. Fueron diseñados además, los diagramas correspondientes a cada fase según la metodología RUP para un mejor entendimiento de los flujos del proceso de gestión así como de los componentes del sistema.

La implementación del sistema permitió disponer de una aplicación que mejora la gestión de las muestras en el Banco Nacional de ADN del CNGM. El mismo constituye una herramienta de apoyo para el estudio de enfermedades genéticas en el país y las pruebas realizadas a nivel de desarrollador validaron la operatividad de sus principales funcionalidades.

RECOMENDACIONES

- Hacer un levantamiento de los tipos de resultados posibles para cada registro de estudio en el banco, permitiendo a los genetistas especificar una mayor cantidad de detalles y disminuyendo la cantidad de información que debe introducir en los resultados de las muestras.
- Valorar la utilización del sistema de autenticación SAAA en la aplicación para el aprovechamiento de la utilización de los servicios del Registro Informatizado de Salud (RIS).

BIBLIOGRAFÍA

1. **Ojeda, Dra. Norma Elena de León.** Genética Médica. [En línea] [Citado el: 12 de Octubre de 2010.] http://www.sld.cu/sitios/genetica/verpost.php?blog=http://articulos.sld.cu/genetica&post_id=391&c=2987&tipo=2&idblog=141&p=1&n=de.
2. Genética. [En línea] 19 de Junio de 2010. [Citado el: 11 de Febrero de 2011.] http://articulos.sld.cu/genetica/archives/195.
3. Definición ABC. [En línea] [Citado el: 12 de Octubre de 2010.] http://www.definicionabc.com/ciencia/adn.php.
4. Cultek S.L.U. [En línea] [Citado el: 20 de Octubre de 2010.] http://www.cultek.com/aplicaciones.asp?p=Aplicacion_Almacenamiento&opc=introduccion. 4.
5. Bioética de los Bancos de muestras en la Pesquisa Neonatal. **Vallejera, Lic. Ciencias Farmacéuticas. MSc. Bioquímica Instituto de Información Científica-Tecnológica. Darlenis Herrera.** 1, Bogotá: s.n., Enero-Junio de 2010, Revista Latinoamericana de Bioética, Vol. 10. http://www.scielo.unal.edu.co/scielo.php?pid=S1657-47022010000100010&script=sci_arttext.
6. MKM Publicaciones. BIOTECH magazine. Soluciones informáticas para la gestión y trazabilidad de los Biobancos. [En línea] [Citado el: 10 de febrero de 2011.] http://www.mkm-pi.com/biotech/soluciones-informaticas-para-la-gestion-y-trazabilidad-de-los-biobancos/.
7. **Ing. Marlene M.M, Ing Gonzalo Q.A.** Sistema de información. [En línea] Publicado (2006). [Citado el: 30 de Noviembre de 2010.] www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r14811.DOC.
8. Bio-e-Bank. [En línea] [Citado el: 21 de Octubre de 2010.] http://www.bio-e-bank.com/.
9. Cultek. NorayBank. [En línea] http://www.cultek.com/index.asp?p=productos-destacados-detalles&id_pdt=153.
10. Infozara. TechBioBank. [En línea] http://infozara.4java.ca/WebInfozara/productosTechBioBankAlcance.do?enlaceMenu=productos&enlaceSubmenu=techBioBank.
11. **Kruchten, Philippe.** El proceso unificado racional: Una introducción (3ro Ed.). 2004. ISBN 0-321-19770-4.

12. **G, Rumbaugh J Jacobson I Booch.** El lenguaje unificado de modelado. Pdfgratis.org. [En línea] [Citado el: 30 de Noviembre de 2010.] http://www.pdfgratis.org/Rumbaugh-J-Jacobson-I-Booch-G-El-lenguaje-unificado-de-modelado/1.
13. **Peña, Ing. Dennys J. Hdez.** SACCEM: SISTEMA AUTOMATIZADO CUBANO PARA EL CONTROL DE EQUIPOS MÉDICOS. [En línea] [Citado el: 24 de febrero de 2011.]
14. **Oracle y afiliados.** Manual de Referencia MySQL 5.0 . [En línea] [Citado el: 5 de junio de 2011.] http://dev.mysql.com/doc/refman/5.0/es/features.html.
15. **Narvaez, Alberto Martín.** Sobre Symfony. [En línea] [Citado el: 10 de abril de 2011.]
16. **Fabien Potencier, François Zaninotto.** Symfony la guia definitiva. Libroweb.es. [En línea] [Citado el: 30 de Noviembre de 2010.] http://www.librosweb.es/symfony_1_2/pdf/symfony_1_2_guia_definitiva_2caras.pdf.
17. **Kabir, Mohammed J.** Apache Server 2 Bible. Estados Unidos: Hungry Minds, Inc., 2002. http://es.scribd.com/doc/24039203/Mohammed-Kabir-La-Biblia-Del-Servidor-Apache-2.
18. **Sánchez, Jorge.** JavaScript. Manual de Referencia. . [En línea] 2003. [Citado el: 1 de Diciembre de 2010.] http://www.jorgesanchez.net/web/javascript.pdf.
19. **Alvarez, Miguel Angel.** Desarrolloweb.com. Primeros pasos con JQuery UI. [En línea] 18 de mayo de 2010. [Citado el: 10 de mayo de 2011.] http://www.desarrolloweb.com/articulos/primeros-paso-jquery-ui.html.
20. **Autores, Colectivo de.** PHP- Manual. [En línea] [Citado el: 30 de Noviembre de 2010.] http://www.php.net/manual/es/index.php.
21. Información tecnológica. **García-Nieto, Sergio.** 4, s.l.: La Serena, 2007, Vol. 18, págs. 85-98. versión On-line ISSN 0718-0764.
22. **Sommerville, Ian.** Ingenieria del Software. 7ma Edición. 2005. pág. 687. ISBN: 8478290745.
23. **Nicolás Tedeschi.** MSDN. Qué es un Patrón de Diseño. [En línea] [Citado el: 03 de junio de 2011.] http://msdn.microsoft.com/es-es/library/bb972240.aspx#mainSection.
24. **Dopaso, Ing. Saidel Pérez.** Diseño e implementación del Sistema de Ayuda Médica para la Atención de las Dislipoproteinemias versión 2.0. La Habana: s.n., 2009. Consultado: 22 de marzo de 2010.

25. **Ivar Jacobson, Grady Booch, James Rumbaugh.** El proceso unificado de desarrollo de software. Vol. 1, Consultado: 30 de noviembre de 2010, Disponible en: http://bibliodoc.uci.cu/pdf/reg00060.pdf.

26. **Ing. Luis Antonio Salazar.** Prolegómenos Sobre el Lenguaje de Modelado Unificado (UML). [En línea] 2003. [Citado el: 30 de Noviembre de 2010.] http://www.willydev.net/InsiteCreation/v1.0/descargas/prev/prolego.pdf.

27. Ayuda extendida del Rational Rose Enterprise. Edition. 2003.

28. Ciberaula. [En línea] [Citado el: 30 de Noviembre de 2010.] http://linux.ciberaula.com/articulo/linux_apache_intro/.

29. **José H. Canós, Patricio Letelier y Mª Carmen Penadés.** Métodologías Ágiles en el Desarrollo de Software. [En línea] [Citado el: 30 de Noviembre de 2010.] www.willydev.net/descargas/prev/TodoAgil.pdf.

30. Lector de código de sobremesa para probetas - Micronic. [En línea] [Citado el: 30 de Noviembre de 2010.] http://www.directindustry.es/prod/micronic/lector-de-codigo-de-sobremesa-para-probetas-37875-263458.html.

31. PHP: Hypertext Preprocessor. ¿Qué se puede hacer con PHP? [Online] Diciembre 3, 2010. [Cited: Diciembre 4, 2010.] http://www.php.net/manual/es/intro-whatcando.php.

32. **Larman, Craig.** UML y Patrones. México: PRENTICE HALL, 1999. Consulado 22 de enero de 2011, Disponible en: http://bibliodoc.uci.cu/pdf/reg00061.pdf. ISBN: 970-17-0261-1.

33. **Potencier, Fabien y Zaninotto, François.** Libros Web. Symfony 1.0, la guía definitiva. [En línea] 24 de Diciembre de 2008. [Citado el: 25 de Noviembre de 2010.] http://librosweb.es/symfony_1_0/.

34. **Salazar, Ing. Luis Antonio.** Prolegómenos Sobre el Lenguaje de Modelado Unificado (UML). 2003. Consultado: 20 de noviembre de 2010, Disponible en: http://www.willydev.net/InsiteCreation/v1.0/descargas/prev/prolego.pdf.

35. **Velázquez Nuñez, Ángel.** Aquitectura de Datos II. [En línea] 2011. http://es.scribd.com/doc/54605120/Monografia-1.

36. **Roberth G. Figueroa, Camilo J. Solís.** METODOLOGÍAS TRADICIONALES VS. METODOLOGÍAS ÁGILES. [En línea] 2008. [Citado el: 16 de Noviembre de 2010.] http://adonisnet.files.wordpress.com/2008/06/articulo-metodologia-de-sw-formato.doc.

REFERENCIAS BIBLIOGRÁFICAS

[1]. **Ojeda, Dra. Norma Elena de León.** Genética Médica. [En línea] [Citado el: 12 de Octubre de 2010.] http://www.sld.cu/sitios/genetica/verpost.php?blog=http://articulos.sld.cu/genetica&post_id=391&c=2987&tipo=2&idblog=141&p=1&n=de.

[2]. Genética. [En línea] 19 de Junio de 2010. [Citado el: 11 de Febrero de 2011.] http://articulos.sld.cu/genetica/archives/195.

[3]. Definición ABC. [En línea] [Citado el: 12 de Octubre de 2010.] http://www.definicionabc.com/ciencia/adn.php.

[4]. Cultek S.L.U. [En línea] [Citado el: 20 de Octubre de 2010.] http://www.cultek.com/aplicaciones.asp?p=Aplicacion_Almacenamiento&opc=introduccion. 4.

[5]. Bioética de los Bancos de muestras en la Pesquisa Neonatal. **Vallejera, Lic. Ciencias Farmacéuticas. MSc. Bioquímica Instituto de Información Científica-Tecnológica. Darlenis Herrera.** 1, Bogotá: s.n., Enero-Junio de 2010, Revista Latinoamericana de Bioética, Vol. 10. http://www.scielo.unal.edu.co/scielo.php?pid=S1657-47022010000100010&script=sci_arttext.

[6]. MKM Publicaciones. BIOTECH magazine. Soluciones informáticas para la gestión y trazabilidad de los Biobancos. [En línea] [Citado el: 10 de febrero de 2011.] http://www.mkm-pi.com/biotech/soluciones-informaticas-para-la-gestion-y-trazabilidad-de-los-biobancos/.

[7]. **Ing. Marlene M.M, Ing. Gonzalo Q.A.** Sistema de información. [En línea] Publicado (2006). [Citado el: 30 de Noviembre de 2010.] www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r14811.DOC.

[8]. Bio-e-Bank. [En línea] [Citado el: 21 de Octubre de 2010.] http://www.bio-e-bank.com/.

[9]. Cultek. NorayBank. [En línea] http://www.cultek.com/index.asp?p=productos-destacados-detalles&id_pdt=153.

[10]. Infozara. TechBioBank. [En línea] http://infozara.4java.ca/WebInfozara/productosTechBioBankAlcance.do?enlaceMenu=productos&enlaceSubmenu=techBioBank.

[11]. **Kruchten, Philippe.** El proceso unificado racional: Una introducción (3ro Ed.). 2004. ISBN 0-321-19770-4.

[12]. **G, Rumbaugh J Jacobson I Booch.** El lenguaje unificado de modelado. Pdfgratis.org. [En línea] [Citado el: 30 de Noviembre de 2010.] http://www.pdfgratis.org/Rumbaugh-J-Jacobson-I-Booch-G-El-lenguaje-unificado-de-modelado/1.

[13]. **Peña, Ing. Dennys J. Hdez.** SACCEM: SISTEMA AUTOMATIZADO CUBANO PARA EL CONTROL DE EQUIPOS MÉDICOS. [En línea] [Citado el: 24 de febrero de 2011.]

[14]. **Oracle y afiliados.** Manual de Referencia MySQL 5.0 . [En línea] [Citado el: 5 de junio de 2011.] http://dev.mysql.com/doc/refman/5.0/es/features.html.

[15]. **Narvaez, Alberto Martín.** Sobre Symfony. [En línea] [Citado el: 10 de abril de 2011.]

[16]. **Fabien Potencier, François Zaninotto.** Symfony la guia definitiva. Libroweb.es. [En línea] [Citado el: 30 de Noviembre de 2010.] http://www.librosweb.es/symfony_1_2/pdf/symfony_1_2_guia_definitiva_2caras.pdf.

[17]. **Kabir, Mohammed J.** Apache Server 2 Bible. Estados Unidos : Hungry Minds, Inc., 2002. http://es.scribd.com/doc/24039203/Mohammed-Kabir-La-Biblia-Del-Servidor-Apache-2.

[18]. **Sánchez, Jorge.** JavaScript. Manual de Referencia. . [En línea] 2003. [Citado el: 1 de Diciembre de 2010.] http://www.jorgesanchez.net/web/javascript.pdf.

[19]. **Alvarez, Miguel Angel.** Desarrolloweb.com. Primeros pasos con JQuery UI. [En línea] 18 de mayo de 2010. [Citado el: 10 de mayo de 2011.] http://www.desarrolloweb.com/articulos/primeros-paso-jquery-ui.html.

[20]. **Autores, Colectivo de.** PHP- Manual. [En línea] [Citado el: 30 de Noviembre de 2010.] http://www.php.net/manual/es/index.php.

[21]. Información tecnológica. **García-Nieto, Sergio.** 4, s.l. : La Serena, 2007, Vol. 18, págs. 85-98. Versión On-line ISSN 0718-0764.

[22]. **Sommerville, Ian.** Ingeniería del Software. 7ma Edición. 2005. pág. 687. ISBN: 8478290745.

[23]. **Nicolás Tedeschi.** MSDN. Qué es un Patrón de Diseño. [En línea] [Citado el: 03 de junio de 2011.] http://msdn.microsoft.com/es-es/library/bb972240.aspx#mainSection.

[24]. **Dopaso, Ing. Saidel Pérez.** Diseño e implementación del Sistema de Ayuda Médica para la Atención de las Dislipoproteinemias versión 2.0. La Habana: s.n, 2009. Consultado: 22 de marzo de 2010.

[25]. **Ivar Jacobson, Grady Booch, James Rumbaugh.** El proceso unificado de desarrollo de software. Vol. 1, Consultado: 30 de noviembre de 2010, Disponible en: http://bibliodoc.uci.cu/pdf/reg00060.pdf.

[26]. **Ing. Luis Antonio Salazar.** Prolegómenos Sobre el Lenguaje de Modelado Unificado (UML). [En línea] 2003. [Citado el: 30 de Noviembre de 2010.] http://www.willydev.net/InsiteCreation/v1.0/descargas/prev/prolego.pdf.

ANEXOS

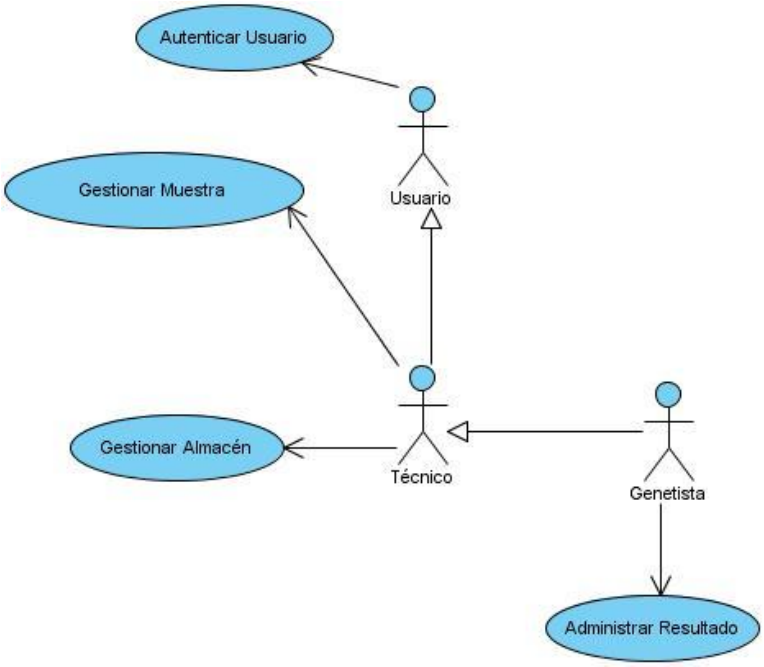

Figura #2. Diagrama de CU arquitectónicamente significativos

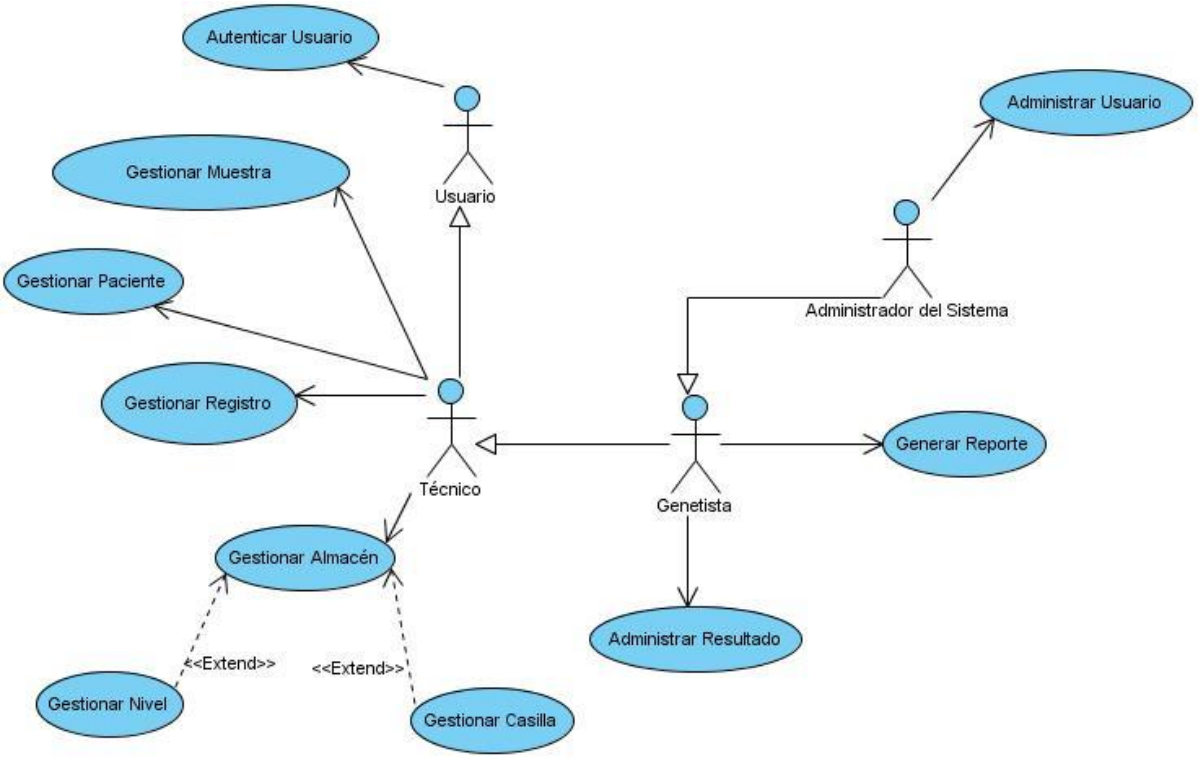

Figura #3. Diagrama de Caso de Uso del Sistema

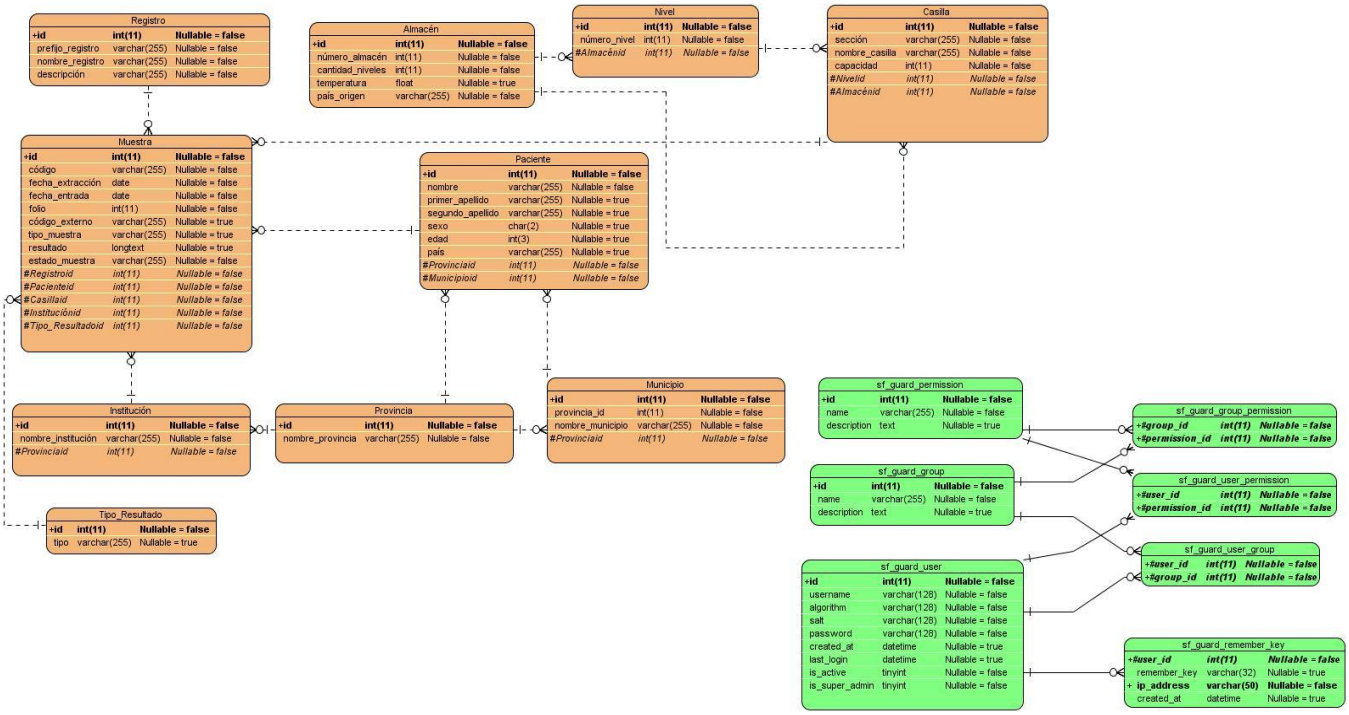

Figura #4. Modelo de Datos

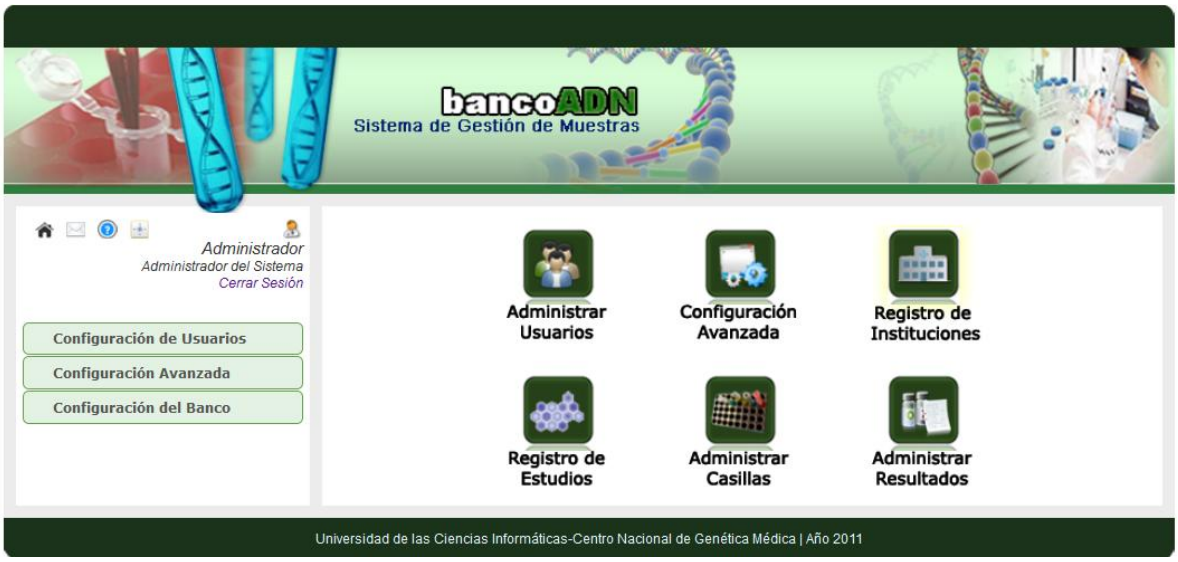

Figura #5. Vista principal del Administrador del Sistema.