Universidad de las Ciencias Informáticas

Facultad 6

Título: alasEPIGEN v2.0: "Aplicación informática para el análisis estadístico en estudios de Epidemiología Genética"

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias de la Informática

Autores:

Yudiel La Rosa González.

Lianet Pupo Santana.

Tutores:

Ing. Yosuán Crespo García.

Ing. Dioleisys Fontela González.

Consultante:

Dr. C. Roberto Lardoeyt Ferrer.

Ciudad de la Habana, 8 de junio del 2010

"Año 52 de la Revolución"

“Sólo triunfa en el mundo quien se levanta y busca las circunstancias y las crea si no las encuentra”.

Bernard Shaw

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 _____.

Yudiel La Rosa González

_____

Firma del autor

Lianet Pupo Santana

_____

Firma del autor

Ing. Yosuán Crespo García

_____

Firma del tutor

Ing. Dioleisys Fontela González

_____

Firma del tutor

DATOS DE CONTACTO

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

Ing. Dioleisys Fontela González
Universidad de las Ciencias Informáticas, Habana, Cuba
Email: dfontela@uci.cu

Consultante:
Dr. C. Roberto Lardoeyt Ferrer
Centro Nacional de Genética Médica, Habana, Cuba
Email: lardgen@infomed.sld.cu

AGRADECIMIENTOS

Agradecemos a nuestro comandante en jefe Fidel Castro y a la Revolución Cubana por darnos la oportunidad de estudiar en esta universidad de excelencia. A nuestros familiares que siempre están ahí cuando los necesitamos. Al Dr. C Roberto por toda su ayuda, a los tutores Yosuán y Dioleisys que tanto nos apoyaron. A nuestros amigos por toda su comprensión en los momentos difíciles. A los compañeros del proyecto y a todos los que nos ayudaron en nuestra formación.

DEDICATORIA

De Lianet

Dedico mi tesis a la mujer tan maravillosa que tengo por madre, a mi excepcional padre, a mi esforzada hermanita, a mi segunda madre Fifi y a mi compañero, amigo y amor de mi vida Yudiel. A mi abuela Rosa y a mis tías Ruth y Odalis por ser mis amigas y consentirme muchas veces. A mis amigas de la universidad: Elezky, Surama y Karelia.

De Yudiel

Dedico este trabajo a las dos personas que les debo mi vida; mi madre y mi padre que tanto se sacrificaron todos estos años para que pudiera estudiar y ser la persona que soy, a mi sobrina Karla por siempre apoyar mis ideas al igual que mi abuela Julia, a mi hermana por ser la persona que es. Del mismo modo quiero dedicárselo a la chica que siempre estuvo a mi lado en la realización de este trabajo, mi novia y compañera de tesis Lianet. A la FEU por las clases de amor, sacrificio y entrega que no se dan en la escuela.

RESUMEN

En el Centro Nacional de Genética Médica, los estudios sobre análisis estadísticos de Genética Poblacional se realizan utilizando diferentes software estadísticos, los cuales no cubren muchas de las funcionalidades demandadas por los especialistas y en muchos casos son herramientas propietarias. Al mismo tiempo, estas herramientas no son pertinentes para la realización de estos estudios y no arrojan un resultado completo sobre los mismos, teniendo los especialistas que recurrir a cálculos manuales de fórmulas matemáticas complejas para completar la investigación iniciada. En el año 2009 el Centro Nacional de Genética Médica conjuntamente con la Universidad de las Ciencias Informáticas desarrollaron la aplicación alasEPIGEN que realiza dos estudios; el estudio de Epidemiología Genética y el estudio de Epidemiología Tradicional. Además de estos dos estudios, los genetistas realizan el estudio de Genética Poblacional, el cual es de gran importancia para sus investigaciones, por lo que es necesario incluir este estudio en la aplicación informática alasEPIGEN para obtener una herramienta más potente que realice y centralice los análisis estadísticos. Para los genetistas es necesario salvar y cargar los resultados de los estudios que realizan para utilizarlos en futuras investigaciones, pero que estos no sean accedidos por otras personas, manteniendo la integridad y confidencialidad de los datos.

Esta nueva versión de la aplicación informática alasEPIGEN permitirá la realización de estudios de Genética Poblacional, realizando diferentes cálculos estadísticos. Se podrá salvar y cargar el resultado de un estudio realizado además de poder imprimir el resultado del mismo.

PALABRAS CLAVE

Análisis estadísticos, Centro Nacional de Genética Médica, Epidemiología Genética, Genética Poblacional, estudios epidemiológicos.

TABLA DE CONTENIDOS

INTRODUCCIÓN

La Epidemiología Genética es una disciplina relativamente nueva, que se ha desarrollado en los últimos años. Estudia la interacción entre los factores genéticos y ambientales que dan origen a las enfermedades del ser humano valiéndose de marcadores genéticos desarrollados a través de la biología molecular. Sin embargo, pasaron casi 100 años antes que los epidemiólogos interesados en la genética y los genetistas interesados en la epidemiología pudieran desarrollar los primeros métodos analíticos para identificar los factores ambientales y genéticos involucrados en los procesos patológicos.

Las estrategias de investigación en la Epidemiología Genética pueden ser de dos tipos: descriptivas y analíticas. Las descriptivas, tanto en el nivel poblacional como en el familiar, se basan en el estudio del tiempo, el lugar y la persona. Las analíticas por el contrario, tienen como objetivo identificar la función de factores genéticos en la historia natural de las enfermedades, tanto en poblaciones como en familias.

Con los avances tecnológicos y el conocimiento biológico que subyace en la acción de los genes, se puede decir que la Epidemiología Genética es una disciplina que combina el método epidemiológico con el genético para estudiar la variación genética en poblaciones humanas y su relación con los cambios fenotípicos normales y patológicos. [1]

Tradicionalmente, los epidemiólogos se han ocupado de la relación entre la aparición de enfermedades y el ambiente; a su vez los investigadores en el campo de la genética se han ocupado de evaluar los efectos de la estructura de la población y de las fuerzas de la selección sobre la frecuencia de los rasgos genéticos. Los estudios en este campo en el mundo han aumentado en estos últimos años, ya que se ha podido disponer de numerosa información genética gracias al material biológico perteneciente a estudios de Epidemiología Tradicional que estudia la relación entre el ambiente y la incidencia de determinada enfermedad, reconociendo la importancia del huésped y su constitución genética. [1]

Un estudio importante dentro de la Epidemiología Genética es el de Genética Poblacional, que se ocupa de predecir las consecuencias que entrañan la estructura de la población y los fenómenos de selección y mutación para los fenotipos constitucionales y las enfermedades. Finalmente, la Epidemiología Genética estudia la manera en que los factores de riesgo presentes en el medio ambiente interactúan con la constitución genética de una población determinada. [1]

En Cuba la principal fortaleza de la genética es la genética comunitaria, no solo por la población que posee y por la accesibilidad de ésta a los servicios médicos, sino por la organización de un sistema nacional de salud que va desde el nivel primario hasta el terciario. Dado el drástico descenso en el coste de "*genotipado*", los estudios de Epidemiología Genética suelen disponer de una gran cantidad de

información y para realizar el análisis estadístico de estos es preciso utilizar un software que integre la mayor cantidad de funcionalidades y que permita realizar dichos análisis de forma rápida y factible. [2]

En el Centro Nacional de Genética Médica, los estudios sobre análisis estadísticos de Genética Poblacional se realizan utilizando diferentes software estadísticos, los cuales no cubren muchas de las funcionalidades demandadas por los especialistas y en muchos casos son herramientas propietarias, requiriendo el pago de licencias en cada una de las instituciones donde se vayan a utilizar, lo que constituye un gasto considerable para el país. Al mismo tiempo, estas herramientas no son pertinentes para la realización de estos estudios y no arrojan un resultado completo sobre los mismos, teniendo los especialistas que recurrir a cálculos manuales de fórmulas matemáticas complejas para completar la investigación iniciada.

En el año 2009 el Centro Nacional de Genética Médica conjuntamente con la Universidad de las Ciencias Informáticas desarrollaron la aplicación alasEPIGEN que realiza los estudios de Epidemiología Genética y Epidemiología Tradicional. Además de estos, los genetistas realizan el estudio de Genética Poblacional, el cual es de gran importancia para sus investigaciones, por lo que es necesario incluir el mismo en la aplicación informática alasEPIGEN para obtener una herramienta más potente que realice y centralice los análisis estadísticos. Para los genetistas es necesario salvar y cargar los estudios que realizan para utilizarlos en futuras investigaciones, pero que estos no sean accedidos por otras personas, manteniendo la integridad y confidencialidad de los datos.

Por todo lo antes expuesto se plantea como **problema científico**: ¿Cómo contribuir a la realización de análisis estadísticos sobre estudios de Genética Poblacional en Cuba?

Este problema se enmarca en el **objeto de estudio**: Proceso de gestión de información de estudios de Epidemiología Genética.

El **campo de acción** abarcado es: Proceso de gestión de información de análisis estadísticos en estudios de Genética Poblacional en Cuba.

El objetivo general de la investigación: Desarrollar en la aplicación informática alasEPIGEN el estudio de Genética Poblacional.

Los objetivos específicos de la investigación:

- Identificar las nuevas funcionalidades de la aplicación informática.
- Diseñar las nuevas funcionalidades de la aplicación informática.
- Implementar las nuevas funcionalidades de la aplicación informática.

Las tareas de la investigación:

- Análisis de software existentes que posibilitan realizar análisis estadísticos en el campo de la Epidemiología Genética.
- Estudio de la aplicación informática alasEPIGEN v1.0.
- Identificación de los nuevos requerimientos de la aplicación informática.
- Diseño de las nuevas funcionalidades de la aplicación informática.
- Implementación de las nuevas funcionalidades de la aplicación informática.
- Implementación de la seguridad de la aplicación informática.

Estructura del documento

Capítulo 1. Fundamentación teórica: En este capítulo se realiza una breve descripción del proceso de gestión de la información de la salud en Cuba. Se explican brevemente algunos software que ayudan a realizar cálculos estadísticos vinculados a estudios de Genética Poblacional. Además, se describen las tecnologías, las herramientas, los roles a desempeñar, los artefactos generados, así como los patrones de diseño y el patrón arquitectónico a utilizar.

Capítulo 2. Características del sistema: En este capítulo se caracteriza el negocio, identificándose los actores y casos de uso del mismo. Se describen los procesos que son objeto de automatización. Se definen los requisitos funcionales y no funcionales que tendrá la aplicación. Se identifican los actores y casos de uso del sistema a desarrollar, y se brinda una breve descripción de estos casos de uso. Todo lo anteriormente expuesto es en respuesta al problema presentado por los genetistas a la hora de realizar estudios de Genética Poblacional.

Capítulo 3. Diseño del sistema: En este capítulo se describe el diseño del sistema, evidenciándose el uso de patrones de diseño aplicados para el desarrollo del mismo. Se modelan las clases del diseño y los diagramas de interacción en correspondencia con la realización de los casos de uso descritos en el capítulo anterior. Se definirá el diagrama de despliegue de la aplicación.

Capítulo 4. Implementación y prueba del sistema: En este capítulo se representan los diagramas de componentes y el diagrama de despliegue del sistema. Además se brinda una descripción de los principales métodos implementados, se muestran imágenes de la interfaz de la aplicación. Se hace referencia a la validación de la aplicación y se incluye el modelo de prueba con la descripción de los casos de prueba basados en casos de uso.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se realiza una breve descripción del proceso de gestión de la información de la salud en Cuba. Se explican brevemente algunos software que ayudan a realizar cálculos estadísticos vinculados a estudios de Genética Poblacional. Además, se describen las tecnologías, las herramientas, los roles a desempeñar, los artefactos generados, así como los patrones de diseño y el patrón arquitectónico a utilizar.

1.1. Proceso de gestión de información de la salud en Cuba

La informatización del proceso de gestión de información del Sistema Nacional de Salud (SNS) está sostenida en estrategias y políticas trazadas por la dirección del país y el Ministerio de Salud Pública, siendo ésta una tarea de vital importancia para nuestro país. Con el propósito de avanzar en esta dirección, se ha hecho imprescindible el equilibrio entre el conocimiento científico y las tecnologías de la informática y las comunicaciones (TICs). Por lo que se hace necesaria la realización de nuevas aplicaciones informáticas dirigidas a elevar la calidad en la gestión de la información en el campo de la salud, conociendo los recursos científicos y tecnológicos con los que se cuenta.

1.1.1. Estudio de Genética Poblacional

La Genética Poblacional se ha convertido en un arma poderosa de la Epidemiología Genética por las importantes aplicaciones que tiene para la Genética. La Genética Poblacional describe la variación genética, que no es más que la distribución de los genes en las poblaciones; permite modelar los genes y los fenotipos de las mismas, lo que ha permitido estudiar la evolución genética. También permite estudiar la introducción de una nueva variación, patrones de reproducción diferencial de genotipos, predicción de efectos sobre composición genética permitiendo estudiar los mecanismos que garantizan la variabilidad genética como son la emigración, los efectos climatológicos, entre otros.

Se pueden realizar estudios de asociación alélica, para demostrar el equilibrio génico de alelos de interés en los controles de las poblaciones. Posibilita el asesoramiento genético de entidades genéticas a partir del conocimiento de la frecuencia de genes de interés clínico. Permite emitir y estimar riesgos teniendo en cuenta la incidencia de una enfermedad en una población. Todas estas aplicaciones permiten un mayor control de las enfermedades en las poblaciones y de esta forma poder tomar medidas en beneficio de las personas que conviven en estas.

1.2. Software que permiten realizar cálculos estadísticos

Como parte de la investigación se analizaron varios software que se utilizan para realizar análisis estadísticos en busca de posibles soluciones que pudieran ayudar en la solución del problema planteado. A continuación se relacionan un grupo de características de cada uno de estos software.

IBM SPSS Statistics (anteriormente SPSS Statistics): es un programa informático de análisis estadístico muy usado en las ciencias sociales y las empresas de investigación de mercado. Está estructurado por un módulo base y otros módulos adicionales que proveen nuevas funcionalidades al software. Permite integrar otros productos de la familia de SPSS para enriquecer sus funcionalidades. Es un software multiplataforma, disponible en su versión 18 para las plataformas Windows, Mac y Linux. De manera general permite el acceso y la gestión de datos desde Excel, base de datos (Oracle, SQL Server, IBM AIX), permite exportar datos a XML, HTML o en formato PDF. Posibilita definir propiedades de variables, realizar estadística descriptiva, inferencia estadística, prueba de hipótesis entre otras. [3]

Este software presenta muchas funcionalidades desde el punto de vista estadístico pero presenta las siguientes privaciones:

- Es un software propietario de la empresa SPSS, de IBM, de licencia altamente costosa, por lo que su utilización significaría un gasto sustancial para la economía del país.
- Tiene soporte técnico sólo para clientes registrados con licencia para la versión más actual del software y la que no sea la versión más actual, mientras dicha versión no tenga más de un año, lo que provoca un costo adicional en la actualización y mantenimiento del software.
- Desde el punto de vista de la Genética Poblacional, no contiene todas las funcionalidades o requisitos necesarios para realizar un estudio particular de este tipo.

STATISTICA: es un paquete completo de análisis estadístico usado en minería de datos, investigaciones y en el ámbito empresarial. Tiene una amplia selección de procedimientos estadísticos y gráficos, además de módulos especializados; los cuales están disponibles como parte de un paquete en conjunto. El programa calcula prácticamente todos los datos estadísticos descriptivos incluyendo medianas, medias, modas y límites de confianza para la media. Dispone de un conjunto de pruebas para el ajuste de distribuciones normales a los datos, aunque también es posible trabajar con otras distribuciones. Permite calcular todas las medidas normales de asociación, incluyendo coeficientes de

incertidumbre, de Pearson, de Spearman, de Kendall, entre otros. El entorno del sistema es totalmente personalizable, pudiéndose modificar según las necesidades del usuario. Permite la gestión de datos desde bases de datos SQL. [4]

Este software presenta muchas funcionalidades desde el punto de vista gráfico y estadístico pero presenta las siguientes privaciones:

- Es un software propietario de la empresa StatSoft, de licencia altamente costosa, por lo que su utilización significaría un gasto sustancial para la economía del país.
- Tiene soporte técnico solo para usuarios de la última versión del programa.

Arlequín: es un software de análisis estadístico que proporciona al usuario medio en la genética de poblaciones un conjunto de métodos básicos y pruebas estadísticas, con el fin de extraerle información sobre las características genéticas y demográficas de una colección de muestras de población. Algunos de los métodos que desarrolla son índices estándar para la diversidad genética, diversidad molecular y prueba de equilibrio de Hardy-Weinberg. Es capaz de manejar los datos genéticos de formas diferentes, y tratar de llevar a cabo el mismo tipo de análisis independientemente del formato de los datos. Puede manejar varios tipos de datos moleculares, convencionales y también maneja datos presentados en forma de frecuencia genotípica o haplotípica, así como la posibilidad de tratamiento codominante o datos recesivos. Los tipos de datos básicos son: las secuencias de ADN y datos de frecuencia de alelos. Es un software multiplataforma, disponible en su versión 3.1 para las plataformas Windows, Mac y Linux. Está disponible de forma gratuita. [5]

Este software presenta muchas funcionalidades estadísticas para estudios de Genética Poblacional pero presenta las siguientes privaciones:

- No presenta ningún tipo de soporte para sus usuarios.
- La interfaz gráfica no es gentil con el usuario.
- Desde el punto de vista de la Genética Poblacional, a pesar de que contiene muchas de las funcionalidades o requisitos necesarios para realizar un estudio particular de esta ciencia no incluye todos los necesarios.

Epidat: es un software de libre distribución para el análisis epidemiológico y estadístico de datos tabulados. Aunque no permite la gestión de bases de datos, integra procedimientos que no se encuentran

habitualmente en los paquetes estadísticos para epidemiología, por lo que constituye un buen complemento a estas aplicaciones. Permite el estudio de pruebas diagnósticas tanto de pruebas simples como de pruebas múltiples en serie o en paralelo con la opción de trabajar con los datos tabulados o con los valores predictivos, realización de pruebas de hipótesis, cálculo de probabilidades, métodos de selección centrándose en el muestreo aleatorio simple, sistemático y estratificado con afijación proporcional. Permite importar datos desde archivos en formato dBase, Excel o Access. Está compuesto en su versión 3.1 por 12 módulos adaptados a su propia entrada de datos. Opera en la plataforma Windows y el diseño de su entorno general se adapta a los estándares de Windows con barras de menús para facilitar el trabajo de los usuarios. Epidat 4.0 se está programando en java lo que le permitirá funcionar en las plataformas Windows, Linux y Mac. [6]

Se evidenció que el desarrollo de software estadísticos a nivel mundial se ha incrementado estos últimos años debido a la necesidad de resolver problemas de muchísimas índoles. La mayoría de estos sistemas informáticos tienen limitantes que impiden su utilización para el desarrollo de estudios de Genética Poblacional. Teniendo en cuenta la necesidad de utilizar un software pertinente para este tipo de estudio en el Sistema Nacional de Salud se ha decidido desarrollar la segunda versión del software de análisis estadístico alasEPIGEN que incluya el estudio de Genética Poblacional, para que ésta sea distribuida por todos los Centros Municipales de Genética Médica.

1.2.1. Estudio de factibilidad económica

Por concepto de sustitución de importaciones, el país con la introducción del software alasEPIGEN en cada uno de los 184 Centros Municipales de Genética Médica, se ahorra USD 312 616.00 por la no adquisición de la licencia del software SPSS v18.0 la cual cuesta USD 1 699.00. A la par del SPSS también se utiliza el Minitab v15.0 cuya licencia cuesta USD 1 195.00 por lo que al adquirir una para cada centro municipal se invertiría USD 219 880.00. También se utiliza el Statistica cuyo módulo base tiene un costo por cada licencia de USD 995.00, por lo que al adquirir una para cada centro municipal se invertiría USD 183 080.00, para un total entre los tres software de USD 715 576.00.

1.3. Metodologías de desarrollo de software

El desarrollo de software es riesgoso y muy difícil de controlar, pero si no se aplica una metodología que guíe el desarrollo del mismo lo que se obtendrá al final es insatisfacción por parte de los clientes y por ende del equipo de desarrollo. Una metodología de desarrollo de software es un proceso en el que se define "quién" está haciendo "qué", "cómo" y "cuándo". [7]

No existe un concepto único para definir las metodologías de desarrollo del software. Especialistas en este tema opinan que las metodologías de desarrollo del software son como un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. Otro concepto importante según Jacobson es que "Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software. Una metodología puede seguir uno o varios modelos de ciclo de vida, es decir, el ciclo de vida indica que es lo que hay que obtener a lo largo del desarrollo del proyecto pero no como hacerlo". [7]

1.3.1. Proceso Unificado de Rational

El Proceso Unificado de Rational más conocido como RUP es un proceso de desarrollo de software. Un proceso de desarrollo de software es un conjunto de actividades necesarias para transformar los requisitos del usuario en un producto (software). Ver figura 1.

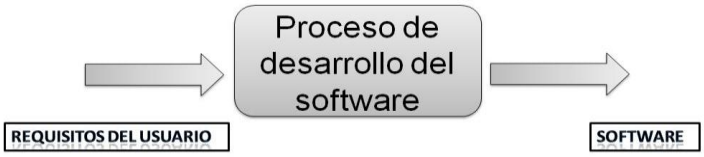

Figura 1 Proceso de desarrollo del software.

RUP es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto. [7]

RUP presenta tres características importantes las cuales se mencionan a continuación:

- Dirigido por casos de uso: Los sistemas de software surgen para dar servicios a sus usuarios, están compuestos por casos de uso. Los casos de uso son una técnica para capturar requisitos de importancia para el usuario. Los casos de uso guían el diseño, la implementación y las pruebas, es

decir todo el proceso de desarrollo del software. Los casos de uso guían la arquitectura del sistema y la arquitectura del sistema influye en la selección de los casos de uso. [7]

- Centrado en la arquitectura: La arquitectura de software es lo más parecido a la arquitectura de edificaciones porque las dos dan un resultado final. La arquitectura de software involucra los aspectos estáticos y dinámicos más significativos del sistema, está relacionada con la toma de decisiones que indican cómo tiene que ser construido el sistema y ayuda a determinar en qué orden. Además la definición de la arquitectura debe tomar en consideración elementos de calidad del sistema, rendimiento, reutilización y capacidad de evolución por lo que debe ser flexible durante todo el proceso de desarrollo. [7]
- Iterativo: El desarrollo de un producto de software supone un gran esfuerzo que puede durar varios meses o hasta años. Por lo antes expuesto se recomienda dividir el trabajo en partes más pequeñas o miniproyectos. Cada miniproyecto es una iteración que va en incremento. Las iteraciones hacen referencia a pasos en el flujo de trabajo, y los incrementos al crecimiento del producto. Para que sea esto efectivo las iteraciones deben estar controladas, esto quiere decir, que deben ser seleccionadas y ejecutarse de forma planificada. [7]

RUP presenta cuatro fases y nueve flujos de trabajo de los cuales seis son de ingeniería y tres de apoyo. [7]

Las fases son:

- Inicio: En esta fase se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.
- Elaboración: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido.
- Construcción: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtiene 1 o varios “release” del producto que han pasado las pruebas. Se ponen estos “release” a consideración de un subconjunto de usuarios.

- Transición: El “release” ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

Los flujos de trabajo son:

- Modelado del negocio: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
- Requerimientos: Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
- Análisis y diseño: Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.
- Implementación: Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán, la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
- Prueba (Testeo): Busca los defectos a lo largo del ciclo de vida.
- Instalación: Produce el “release” del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.
- Administración del proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
- Administración de configuración y cambios: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.
- Ambiente: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

En la siguiente figura se muestra a RUP en sus dos dimensiones.

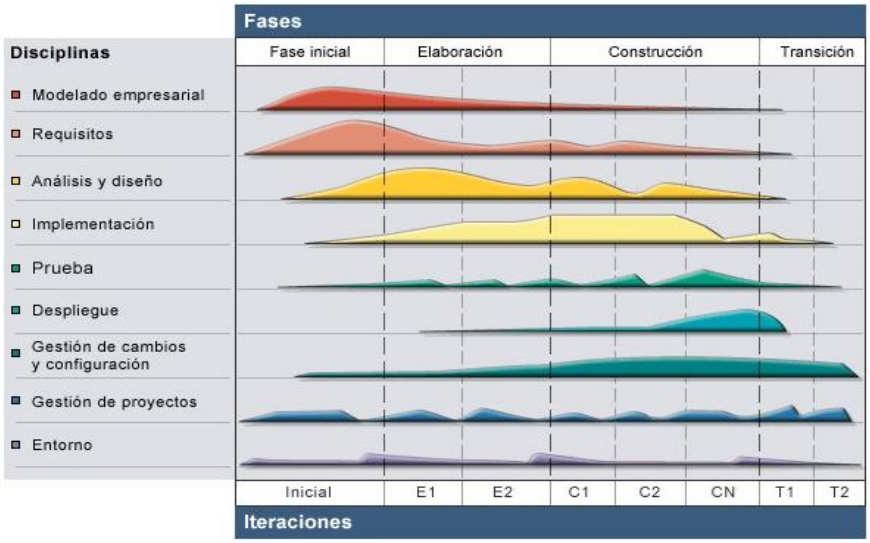

Figura 2 RUP en dos dimensiones.

Se utilizará como metodología de desarrollo del software RUP para llevar a cabo el desarrollo de este trabajo porque es uno de los procesos más generales de los existentes, ya que está pensado para adaptarse a cualquier proyecto. Aunque RUP está pensado para equipos de proyectos grandes tolera a equipos de desarrollo pequeños. Esta metodología crea como base los casos de uso los cuales describen los requerimientos de la aplicación desde el punto de vista del usuario. Además define en cada momento del ciclo de vida del proyecto, qué artefactos, con qué nivel de detalle, y por cuál rol, se deben crear. Con RUP se presentarán al cliente los artefactos al final de una fase y se valorarán las pre condiciones para la siguiente (definición de riesgos, aceptación del plan de iteración, prototipos, entre otros.) Es una guía para utilizar de manera efectiva UML (Lenguaje Unificado de Modelado) y le proporciona a cada miembro del equipo fácil acceso a una base de conocimientos con guías, plantillas y herramientas para todas las actividades críticas de desarrollo. Lo más importante de RUP es que es una metodología muy organizativa y el objetivo de cada actividad es la creación de los artefactos.

1.4. Lenguaje Unificado de Modelado (UML)

UML es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. [8]

En estos tiempos se considera a UML como el lenguaje estándar en el análisis y diseño de sistemas de software. A través de este lenguaje es posible establecer los requerimientos y estructuras necesarias para plasmar un software antes del proceso de implementación. UML no es un método de desarrollo, por lo que no sirve para determinar qué hacer o cómo diseñar el software, sino que simplemente ayuda a visualizar el diseño y a hacerlo más accesible para los desarrolladores. Su utilización es independiente del lenguaje de programación a utilizar para el desarrollo del software y de las características que éste pueda presentar. Incluye semántica y notación, pero es utilizable con diversas herramientas y lenguajes de implementación.

UML es un lenguaje de modelado de propósito general que pueden usar todos los modeladores. No tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática. Incorpora buenas prácticas de diseño, tales como la encapsular, separar temas, y capturar la intención del modelo construido. Aborda los problemas actuales del desarrollo de software, tales como gran tamaño, distribución, concurrencia, patrones, y desarrollo en equipo. [8]

Para una mejor comprensión de UML se puede dividir en varias vistas. Las vistas son un subconjunto de UML que modela construcciones que representan un aspecto de un sistema. En un nivel superior las vistas se pueden dividir en tres áreas: clasificación estructural, comportamiento dinámico y gestión del modelo.

1.5. Roles y artefactos

Un rol define el comportamiento y responsabilidades de un individuo, o un conjunto de individuos que trabajan juntos como un equipo, en el contexto de una organización de ingeniería de software. Rol es una definición abstracta de un conjunto de responsabilidades para las actividades a realizar y los artefactos a producir. [9]

Una persona puede desempeñar diferentes roles, así como un mismo rol puede ser representado por varias personas. Los roles no son individuos, sino que describen cómo los individuos deben comportarse y las responsabilidades de un individuo. Los roles proporcionan información adicional sobre las funciones que realiza un trabajador.

RUP presenta nueve flujos de trabajo, donde diferentes roles realizan un grupo de actividades que dan como resultado artefactos. Los roles y artefactos se relacionan a continuación por cada flujo de trabajo: [9]

Modelado del negocio

Analista de procesos del negocio: es responsable de definir la arquitectura del negocio, los casos de uso y los actores del negocio, así cómo interactúan.

Los artefactos más importantes que genera este rol son:

- Reglas del negocio: es una declaración de política o condiciones que deben cumplirse.
- Modelo de casos de uso del negocio: es un modelo que describe los procesos de negocio y sus relaciones con los participantes externos, como clientes y socios.

Diseñador del negocio: detalla la especificación de una parte de la organización.

Los artefactos más importantes que genera este rol son:

- Actor del negocio: representa un papel, en relación con la empresa, por alguien o algo en el ambiente de negocios.
- Caso de uso del negocio: define un conjunto de instancias de caso de uso en el que cada instancia es una secuencia de acciones que realiza una empresa que produce un resultado observable de valor para un actor particular del negocio.
- Entidad del negocio: representa una parte significativa y persistente de la información que es manipulada por los actores y los trabajadores del negocio.
- Trabajador del negocio: es una abstracción de una persona o un software que representa un papel desempeñado dentro de las realizaciones de caso de uso.

Requerimientos

Analista del sistema: es el responsable de investigar, planear, coordinar y recomendar opciones de software y sistemas para cumplir los requerimientos de una empresa de negocios.

Los artefactos más importantes que genera este rol son:

- Modelo de caso de uso del sistema: es un modelo de funciones previstas del sistema y su entorno, y sirve como un contrato entre el cliente y los desarrolladores. El modelo de casos de uso se utiliza como un insumo esencial para las actividades de análisis, diseño y prueba.
- Glosario de términos: define los términos importantes utilizados por el proyecto.
- Documento Visión: define la vista de los interesados del producto a desarrollar, se especifica en términos de los actores principales, las necesidades y las características. Contiene un resumen de las necesidades básicas, proporciona la base contractual de los requisitos técnicos más detallados.

Especificador de requerimientos: especifica los detalles de una o más partes de las funcionalidades del sistema mediante la descripción de uno o varios aspectos de los requisitos.

Los artefactos más importantes que genera este rol son:

- Casos de uso del sistema (CUS): define un conjunto de instancias de caso de uso, donde cada instancia es una secuencia de acciones que lleva a cabo un sistema y que produce un resultado observable de valor a un actor en particular.
- Requerimientos del software: es la condición o capacidad de que un sistema se ajuste a las necesidades del cliente.
- Especificación de los requisitos del software: captura los requisitos de software para el sistema completo o una parte de ese sistema.

Análisis y Diseño

Diseñador: es responsable de diseñar una parte del sistema, dentro de los límites de los requisitos, la arquitectura y el proceso de desarrollo para el proyecto.

Los artefactos más importantes que genera este rol son:

- Clases del diseño: son similares a una clase en la implementación del sistema; tiene propiedades, comportamientos y relaciones con otras clases.
- Realización de los casos de uso: describe cómo un caso de uso particular, se realiza en el modelo de diseño, en términos de colaboración de objetos.

Diseñador de interfaz de usuario: coordina el diseño de la interfaz de usuario. Los diseñadores de interfaz de usuario también están involucrados en la recopilación de los requisitos de facilidad de uso y prototipo candidato del diseño de interfaz de usuario para satisfacer esos requisitos.

El artefacto más importante que genera este rol es:

- Prototipo de interfaz de usuario: es un ejemplo de la interfaz de usuario que se construye con el fin de explorar y/o validar el diseño de la interfaz de usuario.

Diseñador de pruebas: es responsable de definir el enfoque de la prueba y garantizar su aplicación exitosa. La función consiste en identificar las técnicas apropiadas, herramientas y directrices para aplicar las pruebas necesarias, y para dar orientación sobre los requisitos correspondientes para la prueba de esfuerzo.

El artefacto que genera este rol es:

- Diseño de prueba: descripción de los elementos de prueba estructural y las realizaciones de los casos de prueba.

Implementación

Implementador: es responsable del desarrollo y pruebas de componentes, de conformidad con las normas aprobadas del proyecto, para su integración en grandes subsistemas.

Los artefactos más importantes que genera este rol son:

- Elementos de implementación: son las partes físicas que componen una aplicación, incluyendo los archivos y directorios. Estos incluyen los archivos de código de software (fuente, binario o ejecutable), archivos de datos, archivos de documentación y archivos de ayuda en línea.
- Subsistema de implementación: es un conjunto de elementos de implementación. Subsistemas de implementación es la estructura del modelo de implementación dividiéndolo en partes más pequeñas que pueden ser integradas y probadas por separado.
- Elementos de prueba: da cuenta de la prueba de un comportamiento específico que soporta el software.

Pruebas

Analista de pruebas: es responsable de identificar y definir las pruebas requeridas, de darle seguimiento a los avances y resultados de análisis detallados en cada ciclo de prueba y evaluación de la calidad global experimentada como resultado de actividades de prueba.

El artefacto más importante que genera este rol es:

- Caso de prueba: la especificación (generalmente formal) de un conjunto de entradas de prueba, las condiciones de ejecución y los resultados esperados, identificados con el propósito de hacer una evaluación de algún aspecto particular de un elemento de prueba objetivo.

1.6. Herramientas para la modelación

Las Herramientas CASE (***C**omputer **A**ided **S**oftware **E**ngineering, Ingeniería de Software Asistida por Computadoras*) son aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del

software en tareas como el proceso de realizar un diseño del proyecto, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación y detección de errores. [10]

1.6.1. Visual Paradigm

El Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una rápida construcción de aplicaciones con calidad y a un menor costo. Permite crear todos los tipos de diagramas de clases, ingeniería inversa, generar código desde diagramas y generar documentación. A continuación se mencionan algunas de sus características:[11]

- Soporte de UML versión 2.1.
- Diagramas de Procesos de Negocio.
- Modelado colaborativo con CVS y Subversion.
- Soporta múltiples usuarios trabajando sobre el mismo proyecto a la vez.
- Ingeniería inversa.
- Generación de código (Modelo a código, diagrama a código).
- Editor de Detalles de Casos de Uso (Entorno todo en uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso).
- Distribución automática de diagramas (Reorganización de las figuras y conectores de los diagramas UML).
- Es un software multiplataforma.

Se decidió utilizar el Visual Paradigm (Community Edition) en su versión 3.4 para el modelado de nuestro sistema por todas las características antes mencionadas, además de ser una herramienta sencilla y de fácil manejo por los usuarios. Por otra parte una de las grandes ventajas de Visual Paradigm es que utiliza la notación estándar en la arquitectura de software (UML), la cual permite a los arquitectos de software y desarrolladores visualizar el sistema completo utilizando un lenguaje común, asimismo los diseñadores pueden modelar sus componentes e interfaces en forma individual y luego unirlos con otros

componentes del proyecto. Una de las características más importantes que tiene el Visual Paradigm es que su distribución Community Edition es gratuita.

1.7. Herramientas de desarrollo

1.7.1. Lenguaje de programación Java

Java, el lenguaje de programación creado por la empresa Sun Microsystems, se ha consolidado firmemente como uno de los más utilizados en la actualidad y ha demostrado ser un lenguaje muy efectivo en programación general. Los programadores resultan ser mucho más productivos en Java que en cualquier otro lenguaje. Examinando la arquitectura Java se puede decir que sus principales características son: [12]

- Orientado a objetos: Java fomenta los diseño que conlleven a componentes reutilizables, extensibles y sostenibles. Estos componentes son lo bastante flexibles como para controlar los cambios que se puedan producir con el tiempo. Soporta las características propias de la programación orientada a objetos: clase, objeto, herencia, encapsulamiento y polimorfismo.
- Interpretado: Los programas de Java en lugar de ser compilados en ejecutables nativos, su código es interpretado en una Máquina Virtual de Java (MVJ) y de este modo pueden ejecutarse sin tener que volver a compilarlos.
- Robusto: Java es un lenguaje basado en tipos lo que evita las diferencias implícitas entre tipos, hay referencias en lugar de punteros por lo que no se puede hacer referencia a un puntero en memoria corrompiendo accidentalmente la memoria.
- Seguro: Garantiza la seguridad del código que se está ejecutando y evita que el código no seguro realice operaciones seguras.
- De arquitectura neutral: Si una empresa desarrolla un nuevo sistema operativo o un hardware completamente nuevo, no tiene que empezar desde cero sin ningún software. Con tan solo agregar la MVJ en la plataforma recién diseñada se puede ejecutar todos los programas de Java existentes.

Muchos expertos opinan que Java es el lenguaje ideal para aprender y desarrollar la informática moderna, porque incorpora todos estos conceptos de un modo estándar, mucho más sencillo y claro que con las citadas extensiones de otros lenguajes. Se distingue de otros lenguajes, en que es una plataforma

completa de desarrollo, consta de un gran conjunto de componentes que se pueden reutilizar y mecanismos para extenderlos, facilitando el trabajo a los desarrolladores. Java es Open Source (Software libre o Código abierto).

1.7.2. Entorno de desarrollo

Los entornos de desarrollo integrado (IDE del inglés Integrated Development Environment) son programas informáticos que incluyen un conjunto de herramientas de programación que facilitan un grupo de tareas a los programadores. Las herramientas más importantes de un IDE son: un buen editor de código, administrador de proyectos y archivos e integración con sistemas controladores de versiones o repositorios.[12] Para el desarrollo de aplicaciones usando como lenguaje de programación Java se encuentran principalmente el NetBeans y el Eclipse.

Eclipse IDE: Integra herramientas de desarrollo, con una arquitectura abierta y basada en plug-ins. La arquitectura de plug-ins permite integrar diversos lenguajes sobre un mismo IDE e introducir otras aplicaciones complementarias. Tiene un editor visual con sintaxis coloreada, permite el control de versiones con CVS o Subeclipse, y posibilita las pruebas unitarias con Junit. Una característica importante es su desarrollo basado en "*plug-in*"; la mínima unidad de la plataforma que puede ser desarrollada por separado y que le aporta una nueva funcionalidad. Es un software multiplataforma inicialmente desarrollado por IBM, y actualmente gestionado por la Fundación Eclipse con el apoyo de IBM.

NetBeans IDE: En su núcleo, es una herramienta de desarrollo Java, escrita puramente sobre la base de la tecnología Java, de modo que puede ejecutarse en cualquier ambiente que ejecute Java. Es un producto de código abierto, con todos los beneficios del software disponibles gratuitamente. Cada módulo dentro del IDE tiene una función bien definida, funciones tales como el soporte de Java, edición, o soporte para el sistema de control de versiones. Los desarrolladores se sienten atraídos por NetBeans IDE, gracias a su soporte a Java Enterprise Edition, su facilidad de uso, su cumplimiento de regulaciones, sus perfiles de rendimiento, además de su flexibilidad entre plataformas. NetBeans Enterprise Pack soporta el desarrollo de Aplicaciones empresariales, incluyendo herramientas de desarrollo visuales de SOA, herramientas de esquemas XML, orientación a servicios web y modelado UML. Con el configurador NetBeans Swing GUI, tanto los usuarios finales como los desarrolladores obtienen mayor facilidad de uso. Swing simplifica drásticamente la creación de interfaces gráficas de usuario, además de permitir a los

desarrolladores manejar diferentes guías de estilo en diversas plataformas. Fue creado y es respaldado hoy por Sun Microsystems. [13]

Se seleccionó para el desarrollo de la aplicación el Netbeans IDE en su versión 6.8 principalmente porque permite el control de versiones con CVS y Subversion, es el primer IDE que da completamente soporte a JDK 6.0, tiene un potente editor de texto y permite modelado UML.

1.7.3. Sistema de control de versiones

Cuando se hace frente a un proyecto de software, uno de los problemas que aparecen es el de controlar el crecimiento y evolución del producto. Se desea mantener copias y salvas para salvaguardar una historia del producto. Los sistemas de control de versiones buscan resolver este problema y también agregan funcionalidades adicionales que facilitan estas tareas a los desarrolladores.

El control de versiones es el arte del manejo de los cambios en la información. Ha sido durante mucho tiempo una herramienta crítica para los programadores, quienes normalmente empleaban su tiempo haciendo pequeños cambios en el software y después deshaciendo esos cambios al día siguiente. Pero la utilidad del software de control de versiones se extiende más allá de los límites del mundo del desarrollo de software. Allá donde pueda encontrarse a gente usando ordenadores para manejar información que cambia a menudo, hay un hueco para el control de versiones. La misión principal de un sistema de control de versiones es permitir la edición colaborativa y la compartición de los datos. [14]

Para el desarrollo de la aplicación se utilizará el sistema de control de versiones Subversión (SVN).

¿Qué es Subversion (SVN)?

SVN es un sistema de control de versiones libre y de código fuente abierto. Maneja ficheros y directorios a través del tiempo y tiene un árbol de ficheros en un repositorio central. El repositorio es como un servidor de ficheros ordinario, excepto porque recuerda todos los cambios hechos a sus ficheros y directorios. Esto le permite recuperar versiones antiguas de sus datos, o examinar el historial de cambios de los mismos. En este aspecto, mucha gente piensa en los sistemas de versiones como en una especie de “máquina del tiempo”. [14]

SVN puede acceder al repositorio a través de redes, por lo que personas que se encuentren en distintos ordenadores pueden modificar y administrar el mismo conjunto de datos sin que estos tengan que pasar por un único conducto, lo que permite progresar más rápidamente. Es un sistema general que puede ser usado para administrar cualquier conjunto de ficheros. Aunque fue desarrollado inicialmente por CollabNet que todavía financia una gran parte del trabajo, cualquier persona es libre de descargar, modificar, y redistribuir SVN como desee; no se requiere ningún permiso de CollabNet o de cualquier otra persona. Está disponible para las plataformas Windows, Linux/Unix, Mac y Solaris. SVN es la mejora del diseño de CVS.

Sus principales características son:[14]

- Versiones de directorios: CVS solamente lleva el historial de ficheros individuales, pero SVN implementa un sistema de ficheros de versiones virtuales que sigue los cambios sobre árboles de directorios completos a través del tiempo. Ambos, ficheros y directorios, se encuentran bajo el control de versiones.
- Verdadero historial de versiones: Dado que CVS está limitado a versiones de ficheros, operaciones como copiar y renombrar; las cuales pueden ocurrir sobre ficheros, no son soportadas por CVS. En el CVS no se puede reemplazar un fichero con algo nuevo que lleve el mismo nombre sin que el nuevo elemento herede el historial del fichero antiguo; que quizás sea completamente distinto al anterior. Con SVN, usted puede añadir, borrar, copiar, y renombrar ficheros y directorios. Cada fichero nuevo añadido comienza con un historial nuevo, limpio y completamente suyo.
- Envíos atómicos: Una colección cualquiera de modificaciones o bien entra por completo al repositorio, o bien no lo hace en absoluto, permitiéndole a los desarrolladores construir y enviar los cambios como fragmentos lógicos e impide que ocurran problemas cuando sólo una parte de los cambios enviados lo hace con éxito.
- Versiones de metadatos: Cada fichero y directorio tiene un conjunto de propiedades (claves y valores) asociado a él. Usted puede crear y almacenar cualquier par arbitrario de clave/valor que desee. Las propiedades son versionadas a través del tiempo, al igual que el contenido de los ficheros.
- Elección de las capas de red: SVN tiene una noción abstracta del acceso al repositorio, facilitando a las personas implementar nuevos mecanismos de red. Puede conectarse al servidor HTTP Apache como un módulo de extensión. Esto proporciona a SVN una gran ventaja en estabilidad e

interoperabilidad, y acceso instantáneo a las características existentes que ofrece este servidor (autenticación, autorización, compresión de la conexión, etcétera). También tiene disponible un servidor de SVN independiente, y más ligero.

- Manipulación consistente de datos: SVN expresa las diferencias del fichero usando un algoritmo de diferenciación binario, que funciona idénticamente con ficheros de texto y ficheros binarios. Ambos tipos de ficheros son almacenados e igualmente comprimidos en el repositorio, y las diferencias son transmitidas en ambas direcciones a través de la red.
- Ramificación y etiquetado eficientes: El coste de ramificación y etiquetado no necesita ser proporcional al tamaño del proyecto. SVN crea ramas y etiquetas simplemente copiando el proyecto, usando un mecanismo similar al enlace duro. De este modo estas operaciones toman solamente una cantidad de tiempo pequeña y constante.
- Extensible: SVN no tiene un equipaje histórico; está implementado como una colección de bibliotecas compartidas en C con APIs bien definidas. Esto lo hace extremadamente fácil de mantener y reutilizable por otras aplicaciones y lenguajes.

Se utilizará la versión de SVN 1.6. Para la implementación del proyecto se utilizará el soporte a SVN integrado en el IDE NetBeans que permite trabajar con CVS o SVN (en este caso) automáticamente.

1.8. Patrones de casos de uso, diseño y arquitectura

Un patrón no es más que una pareja de problema-solución con un nombre, que estandariza buenos principios y sugerencias relacionadas con la asignación de responsabilidades.

Los patrones siguen el siguiente formato:

- Nombre
- Solución
- Problema
- Explicación
- Ejemplo de utilización

1.8.1. Patrones de casos de uso

La utilización de casos de uso ha evolucionado en un conjunto de patrones que permiten con mayor precisión reflejar los requisitos reales, haciendo más fácil el trabajo con los sistemas, y mucho más simple su mantenimiento. Se presentan a modo de herramientas que permiten resolver los problemas que se les planteen a los desarrolladores de una forma ágil y sistemática. Estos patrones se enfocan hacia el diseño y las técnicas utilizadas en modelos de alta calidad, y no en cómo modelar usos específicos. Utilizando estos patrones, arquitectos, analistas e ingenieros pueden lograr mejores resultados de forma más rápida.

Concordancia: Adición

La sub secuencia común de casos de uso, extiende los casos de uso compartiendo la sub secuencia de acciones. Los otros casos de uso modelan el flujo que será expandido con la sub secuencia. Este patrón es preferible usarlo cuando otros casos de uso se encuentran propiamente completos, o sea, que no requieren de una sub secuencia común de acciones para modelar los usos completos del sistema. [15]

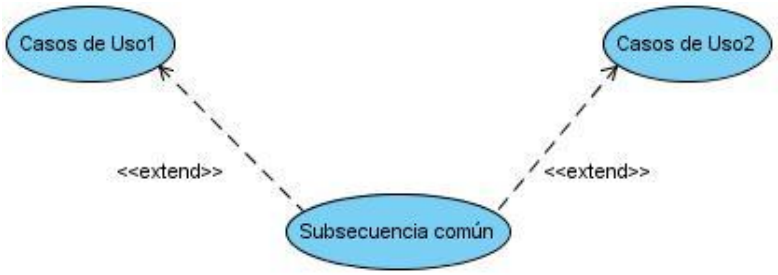

Figura 3 Ejemplo de patrón de concordancia: Adición.

Concordancia: Especialización

La especialización es otro patrón de concordancia que contiene casos de uso del mismo tipo. En este caso, estos son modelados como una especialización de casos de uso de tipo de uso común. Todas las acciones en estos casos de uso son heredadas por los casos de uso hijos, donde otras acciones serán adicionadas o acciones heredadas que serán especializadas. Este patrón es aplicable cuando la utilización de los casos de uso que han sido modelados son del mismo tipo, y este tipo debe hacerse visible en el modelo. [15]

Figura 4 Ejemplo de patrón de concordancia: Especialización.

1.8.2. Patrones de diseño

Un patrón de diseño es:

- una solución estándar para un problema común de programación.
- una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios.
- un proyecto o estructura de implementación que logra una finalidad determinada.
- un lenguaje de programación de alto nivel.
- una manera más práctica de describir ciertos aspectos de la organización de un programa.
- conexiones entre componentes de programas.
- la forma de un diagrama de objeto o de un modelo de objeto. [16]

Las principales características de un patrón de diseño son:[16]

- Son soluciones concretas (proponen soluciones a problemas concretos, no son teorías genéricas).
- Son soluciones técnicas (indican resoluciones técnicas basadas en Programación Orientada a Objetos (POO), en ocasiones tienen más utilidad con algunos lenguajes de programación y en otras son aplicables a cualquier lenguaje).
- Se utilizan en situaciones frecuentes (ya que se basan en la experiencia acumulada de resolver problemas reiterativos).
- Favorecen la reutilización de código (ayudan a construir software basado en la reutilización (a construir clases reutilizables), los propios patrones se reutilizan cada vez que se vuelven a aplicar).
- El uso de un patrón no se refleja en el código (al aplicar un patrón, el código resultante no tiene por qué delatar el patrón o patrones que lo inspiró, no obstante, últimamente hay múltiples esfuerzos enfocados a la construcción de herramientas de desarrollo basados en los patrones y

frecuentemente se incluye en los nombres de las clases el nombre del patrón en que se basan, facilitando así la comunicación entre desarrolladores).

- Es difícil reutilizar la implementación de un patrón (al aplicar un patrón aparecen clases concretas que solucionan un problema concreto y que no será aplicable a otros problemas que requieran el mismo patrón).

Patrones de asignación de responsabilidades

Patrones GRASP

Los patrones GRASP son parejas de problema-solución con un nombre, que codifican buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades. Describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. [16]

Patrón Alta Cohesión: Mantiene la complejidad dentro de límites manejables, es decir asigna una responsabilidad de modo que la cohesión siga siendo alta. La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. [16]

Beneficios:

1. Mejoran la claridad y facilidad con que se entiende el diseño.
2. Se simplifica el mantenimiento y las mejoras de funcionalidad.
3. A menudo se genera un bajo acoplamiento.
4. Soporta mayor capacidad de reutilización.

Patrón Creador: Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que permita conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. Lo que define este patrón es que una instancia de un objeto la tiene que crear el objeto que tiene la información para ello. ¿Qué significa esto?, pues que si un objeto A utiliza específicamente otro B, o si B forma parte de A, o si A almacena o contiene B, o si simplemente A tiene la información necesaria para crear B, entonces A es el perfecto creador de B. [16]

Patrón Controlador: Es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Asigna las responsabilidades de capturar los eventos del sistema a las clases. [16]

De acuerdo con el patrón Controlador, se dispone de las siguientes opciones:[16]

1. El "sistema" global (controlador de fachada).
2. La empresa u organización global (controlador de fachada).
3. Algo en el mundo real que es activo (por ejemplo, el papel de una persona) y que pueda participar en la tarea (controlador de tareas).
4. Un manejador artificial de todos los eventos del sistema de un caso de uso, generalmente denominados "Manejador<NombreCasodeUso>" (controlador de casos de uso).
5. En la decisión de cuál de las cuatro clases es el controlador más apropiado influyen también otros factores como la cohesión y el acoplamiento.

Patrón Bajo Acoplamiento: Es la idea de tener las clases lo menos ligadas entre sí. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Acoplamiento bajo significa que una clase no depende de muchas clases. [16]

Beneficios:

1. No se afectan por cambios de otros componentes.
2. Fáciles de entender por separado.
3. Fáciles de reutilizar.

Patrones de diseño GOF

Los patrones GOF constituyen patrones de diseño surgidos en 1995 por Erich Gamma, Richard Helm, Ralph Jonson y John Vissidess, promueven una expansión de la programación orientada a objetos y se pueden clasificar según su propósito en patrones de creación (para creación de instancias), estructurales

(relaciones entre clases, combinación y formación de estructuras mayores) y de comportamiento (interacción y cooperación entre clases). [16]
Algunos de estos patrones son:

Patrones de comportamiento

Patrón Instancia Única: Garantiza que una clase sólo tenga una instancia, y proporciona un punto de acceso global a ella.[16]

Patrones estructurales

Patrón Fachada: Proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que hace que el subsistema sea más fácil de usar.[16]

1.8.3. Patrones de arquitectura

Los patrones arquitectónicos no son más que un patrón de alto nivel que fija la arquitectura global de una aplicación. Se definen como una descripción de un problema particular y recurrente de diseño, que aparece en contextos de diseño específico, y presenta un esquema genérico demostrado con éxito para su solución. Los patrones arquitectónicos proveen un conjunto de subsistemas predefinidos, especifican sus responsabilidades e incluyen reglas y pautas para la organización de las relaciones entre ellos. [16]

Dentro de los diversos patrones de arquitectura existentes, cabe destacar los más significativos: el patrón de arquitectura en capas y el patrón Modelo Vista Controlador (MVC).

Patrón de arquitectura en capas

El patrón de arquitectura en capas define cómo organizar el modelo de diseño a través de capas, que puedan estar físicamente distribuidas, lo cual quiere decir que los componentes de una capa sólo pueden hacer referencia a componentes en capas inmediatamente inferiores. Éste simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. A continuación se mencionan las capas de la arquitectura en tres capas: [2]

- Capa de presentación, contiene todas las interfaces y sus controles visuales junto con sus eventos.

- Capa de negocio (lógica del dominio), aquí se define todo el código que especifican las reglas de negocio. Surge de los procesos que se han encontrado en el análisis.
- Capa de acceso a datos, contiene el código que permite acceder a las fuentes de datos. Esencialmente trata sobre cuatro operaciones básicas, llamadas Create, Retrieve, Update and Delete (CRUD), que se realizan sobre cualquier fuente de datos (normalmente alguna base de datos relacional).

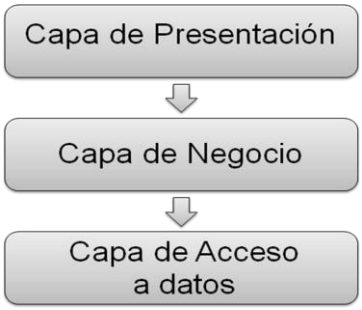

Figura 5 Patrón de Arquitectura: Tres Capas.

1.9. Conclusiones del capítulo

En este capítulo se han descrito las herramientas y el lenguaje a utilizar, así como la metodología de desarrollo de software para llevar a cabo la aplicación. Además se describieron los patrones de diseño de manera general y el patrón de arquitectura como estructura fundamental del software.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

En este capítulo se caracteriza el negocio, identificándose los actores y casos de uso del mismo. Se describen los procesos que son objeto de automatización. Se definen los requisitos funcionales y no funcionales que tendrá la aplicación. Se identifican los actores y casos de uso del sistema a desarrollar, y se brinda una breve descripción de estos casos de uso. Todo lo anteriormente expuesto es en respuesta al problema presentado por los genetistas a la hora de realizar estudios de Genética Poblacional.

2.1. Objeto de estudio

2.1.1. Objetivos estratégicos de la organización

La Red Nacional de Genética está integrada por 184 centros ubicados en todos los municipios y provincias del país, coordinados por el Centro Nacional de Genética Médica. Las investigaciones las realizan los especialistas ubicados en el área de Atención Primaria en los Policlínicos y Centros Municipales de Genética, los especialistas del área de Atención Secundaria en los Centros Provinciales de Genética y los especialistas ubicados en el Centro Nacional de Genética Médica. El objetivo fundamental es desarrollar proyectos de investigación e innovación, evaluar e introducir nuevas tecnologías para el diagnóstico, tratamiento y asesoramiento en relación con las enfermedades genéticas, todo esto encaminado a mejorar la calidad de vida de la población. [17]

2.1.2. Flujo actual de los procesos

En el CNGM el Consejo Científico es quien aprueba el inicio de una investigación epidemiológica basándose en que el instrumento de recolección de datos para la realización del estudio está previamente elaborado con las variables para la obtención de la información, posteriormente el genetista inicia la investigación acudiendo a cada unos de los pacientes aprobados para que formen parte de la misma llenando con sus datos el instrumento. Se realizan estudios de Genética Poblacional sobre los datos recogidos y se obtienen los resultados que son informados al Consejo Científico por el genetista.

2.1.3. Análisis crítico de la ejecución de los procesos

Los genetistas autorizados a realizar estudios de Genética Poblacional están presentando limitaciones en cuanto a la realización de los mismos, porque al realizar diferentes cálculos estadísticos deben hacerlo utilizando diversos software estadísticos que no les brindan todas las funcionalidades necesarias; teniendo

en ocasiones que recurrir a cálculos manuales. Esto conlleva a que la solución que se obtiene pueda no ser exacta y que para estudios posteriores no se cuente con los casos ya analizados por pérdida de información.

2.2. Objeto de automatización

La nueva versión de la aplicación informática alasEPIGEN permitirá la realización de estudios de Genética Poblacional realizando diferentes análisis estadísticos, dando la opción de escoger el tipo de estudio que se desea realizar, insertando los datos necesarios para ello y obteniendo resultados de los cálculos aplicados al mismo, así como, se podrán graficar los valores obtenidos en las tablas de contingencia correspondientes a cada estudio específico. Se podrá salvar y cargar el resultado de un estudio realizado además de poder imprimirlo.

2.3. Modelo de Negocio

2.3.1. Actores del negocio

Actor	Descripción
Consejo Científico	Indica inicio de la investigación y recibe resultados de la misma.

Tabla 1 Actores del negocio.

2.3.2. Trabajadores del negocio

Trabajador	Descripción
Genetista	Especialista en Genética encargado de realizar estudios de Genética Poblacional.

Tabla 2 Trabajadores del negocio.

2.3.3. Diagrama de casos de uso del negocio

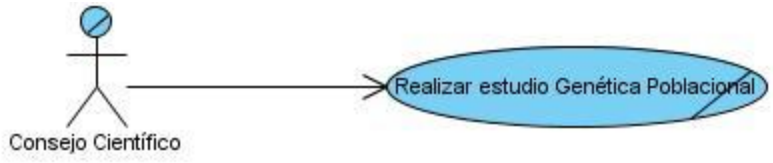

Figura 6 Diagrama de caso de uso del negocio.

2.3.4. Descripción textual del caso de uso del negocio

Caso de uso del negocio:	**Realizar estudio de Genética Poblacional**
Actores:	Consejo Científico(Inicia)
Trabajadores:	Genetista
Resumen:	El caso de uso se inicia cuando el Consejo Científico indica el inicio de la investigación. El Genetista realiza los estudios correspondientes, comunica los resultados al Consejo Científico terminando así el caso de uso.
Pre condiciones:	Se debe haber elaborado el instrumento de recolección de datos.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Negocio**
1. El Consejo Científico indica inicio de la investigación.	2. El genetista aplica instrumento de recolección de datos.
	3. El genetista realiza los estudios de Genética Poblacional.
	4. El genetista obtiene los resultados del estudio de Genética Poblacional.
	5. El genetista comunica al Consejo Científico los resultados.
6. El Consejo Científico recibe resultados sobre el estudio de Genética Poblacional.	
Pos condiciones	Queda creado el estudio de Genética Poblacional.

Tabla 3 Descripción textual del caso de uso del negocio.

2.3.5. Diagrama de actividades

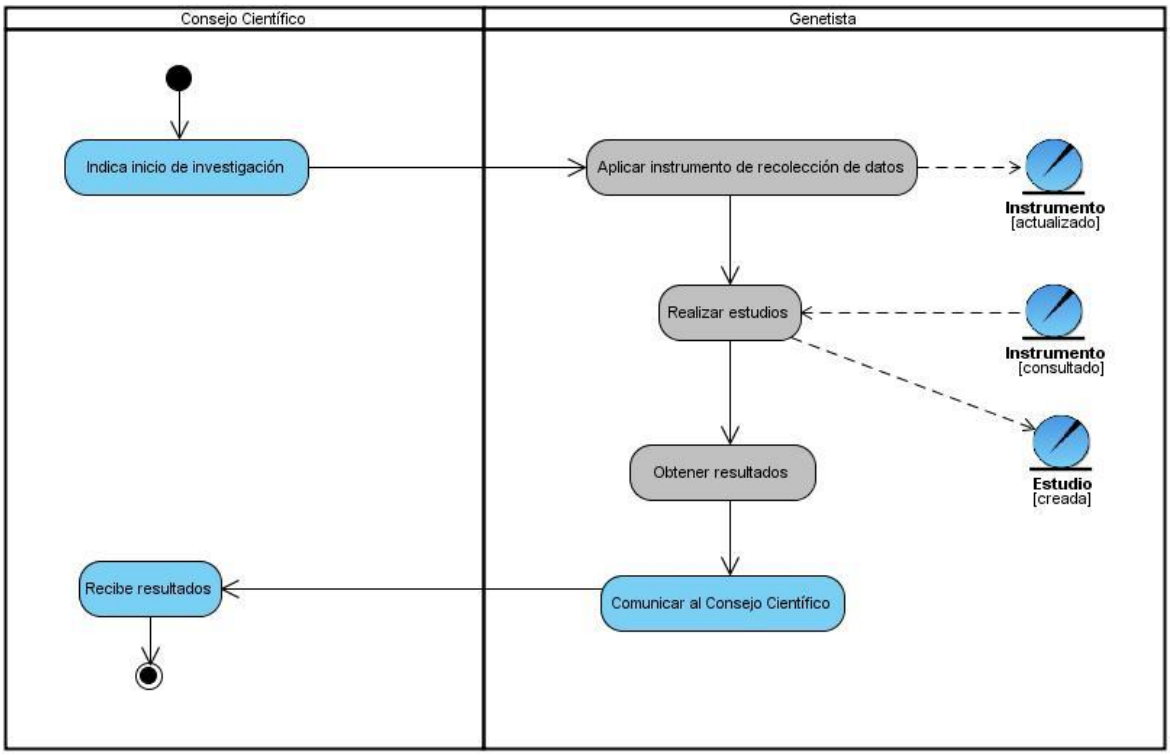

Figura 7 Diagrama de actividades.

2.3.6. Modelo de objetos del negocio

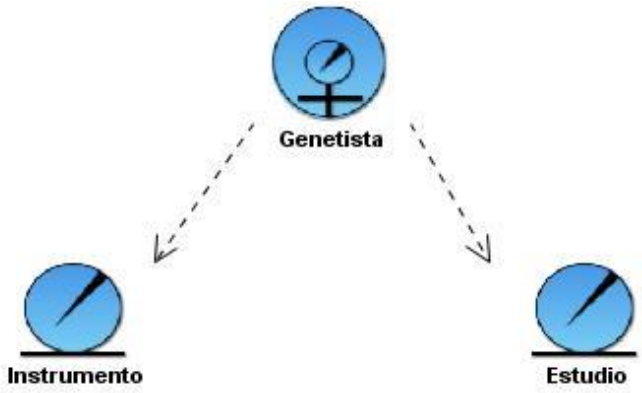

Figura 8 Modelo de objetos del negocio.

2.3.7. Reglas del negocio

Las reglas del negocio de la aplicación alasEPIGEN en su versión primera se mantienen porque las nuevas reglas no las contradicen. A continuación se mencionan las reglas del negocio que se adicionan para la segunda versión de la aplicación alasEPIGEN:

1. Solo el Consejo Científico puede indicar el inicio de una investigación.
2. El instrumento es una planilla que recoge los datos necesarios para realizar un estudio de Genética Poblacional.
3. La probabilidad nunca debe ser cero, se tiene que especificar el último valor en caso de que sea muy pequeña.
4. La cantidad de variables cuando se desea realizar un estudio de Genética Poblacional a partir de una plantilla de datos no puede ser mayor que 30.
5. El valor de la incidencia de la enfermedad en la población en análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva tiene que ser mayor que 0 y menor que 1.
6. El valor de la frecuencia de la enfermedad en varones en análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X tiene que ser mayor que 0 y menor que 1.
7. El tamaño de la población tiene que ser mayor que 0.
8. El equilibrio de Hardy Weinberg se realiza hasta cuatro alelos.
9. Para cargar un estudio hay que entrar la misma contraseña con la que fue guardado.

2.4. Especificación de los requerimientos de la aplicación informática

2.4.1. Requerimientos funcionales

Los requerimientos funcionales son condiciones o capacidades que deben ser alcanzadas o poseídas por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Con ellos se pretende determinar de manera clara y concisa lo que debe hacer el sistema siguiendo un enfoque funcional. [7]

Los requerimientos funcionales agrupados por casos de uso son los siguientes:

1. Crear estudio de Genética Poblacional.

1.1. Crear estudio de Genética Poblacional.

2. Crear análisis de dos alelos con relación.
   2.1. Insertar datos de análisis de alelos con relación.
   2.2. Obtener resultados de análisis de alelos con relación.
   2.3. Imprimir estudio.
3. Crear análisis de tres alelos con relación de codominancia.
   3.1. Insertar datos de análisis de tres alelos con relación de codominancia.
   3.2. Obtener resultados de análisis de tres alelos con relación de codominancia.
   3.3. Imprimir estudio.
4. Crear análisis de cuatro alelos con relación de codominancia.
   4.1. Insertar datos de análisis de cuatro alelos con relación de codominancia.
   4.2. Obtener resultados de análisis de cuatro alelos con relación de codominancia.
   4.3. Imprimir estudio.
5. Crear análisis de cinco o más alelos con relación de codominancia.
   5.1. Insertar datos de análisis de cinco o más alelos con relación de codominancia.
   5.2. Obtener resultados de análisis de cinco o más alelos con relación de codominancia.
   5.3. Imprimir estudio.
6. Crear análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva.
   6.1. Insertar datos de análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva.
   6.2. Obtener resultados de análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva.
   6.3. Imprimir estudio.
7. Crear análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X.
   7.1. Insertar datos de análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X.
   7.2. Obtener resultados de análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X.
   7.3. Imprimir estudio.

8. Crear análisis de alelos ligados al cromosoma X.
    8.1. Insertar datos de análisis de alelos ligados al cromosoma X.
    8.2. Obtener resultados de análisis de alelos ligados al cromosoma X.
    8.3. Imprimir estudio.
9. Guardar resultado del estudio.
    9.1. Guardar resultado del estudio.
10. Cargar resultado del estudio.
    10.1. Cargar resultado del estudio.

2.4.2. Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Normalmente están vinculados a requerimientos funcionales, es decir una vez que se conozca lo que el sistema debe hacer se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser.[7] Teniendo en cuenta que la aplicación es de escritorio se definieron los siguientes requisitos no funcionales:

1. Apariencia o interfaz externa

La interfaz externa debe estar diseñada para verse en cualquier resolución de pantalla. El acceso a la aplicación será sencillo y amigable, el color predominante en la misma debe ser el azul.

2. Facilidad de uso

La aplicación informática debe garantizar un acceso fácil y rápido, contando con un menú que satisfaga las necesidades de los usuarios. Éste podrá ser usado por genetistas que posean conocimientos avanzados en el dominio de la especialidad de la Epidemiología Genética, de la Bioestadística y de la Bioinformática.

3. Rendimiento

Los tiempos de respuestas deben ser generalmente rápidos al igual que la velocidad de procesamiento de la información ya que es una aplicación de escritorio y no posee ningún tipo de intercambio de información a través de la red.

4. Soporte

Se debe asegurar el soporte para los usuarios, de manera que se puedan satisfacer sus necesidades a partir de mejoras, una vez puesta en marcha la aplicación. Para ello se creará un manual de usuarios y una guía de instalación.

5. Software

Se requiere para el funcionamiento de la aplicación en los sistemas operativos (Windows, Linux, Mac); disponer de la Máquina Virtual de Java versión 1.3 o superior.

6. Hardware

Para el desarrollo y ejecución de la aplicación se necesitará:

Procesador de tipo Pentium III a 450 megahercios (MHz) o superior, 256 Megabytes (MB) de memoria RAM o superior y 30 MB de capacidad en el disco duro o superior.

7. Requisitos Legales

Las herramientas y las tecnologías en que estará basada la aplicación informática deberán cumplir con las licencias de software libre.

2.5. Definición de los casos de uso del sistema

2.5.1. Actores del sistema

Actor	Descripción
Genetista	Especialista en Genética que interactúa con el sistema, es el encargado de realizar los estudios y cálculos estadísticos sobre epidemiología.

Tabla 4 Actores del sistema.

2.5.2. Paquetes del sistema

Para la modelación de los casos de uso del sistema (CUS) se decidió dividirlos en paquetes de acuerdo al siguiente criterio de empaquetamiento:

Los CUS requeridos para dar soporte a un determinado proceso de negocio. El paquete Genética Poblacional responde a este criterio. Los casos de uso Guardar estudio y Cargar estudio son extendidos de los casos de uso que se encuentran dentro de estos paquetes. De manera que quedaron integrados a los 10 casos de uso del sistema los 18 requisitos funcionales identificados, teniendo en cuenta los patrones de caso de uso: Concordancia (Adición) y Concordancia (Especialización).

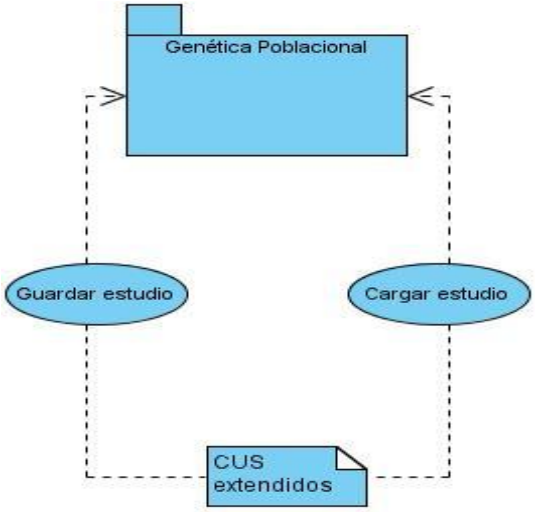

Figura 9 Diagrama de paquetes del sistema.

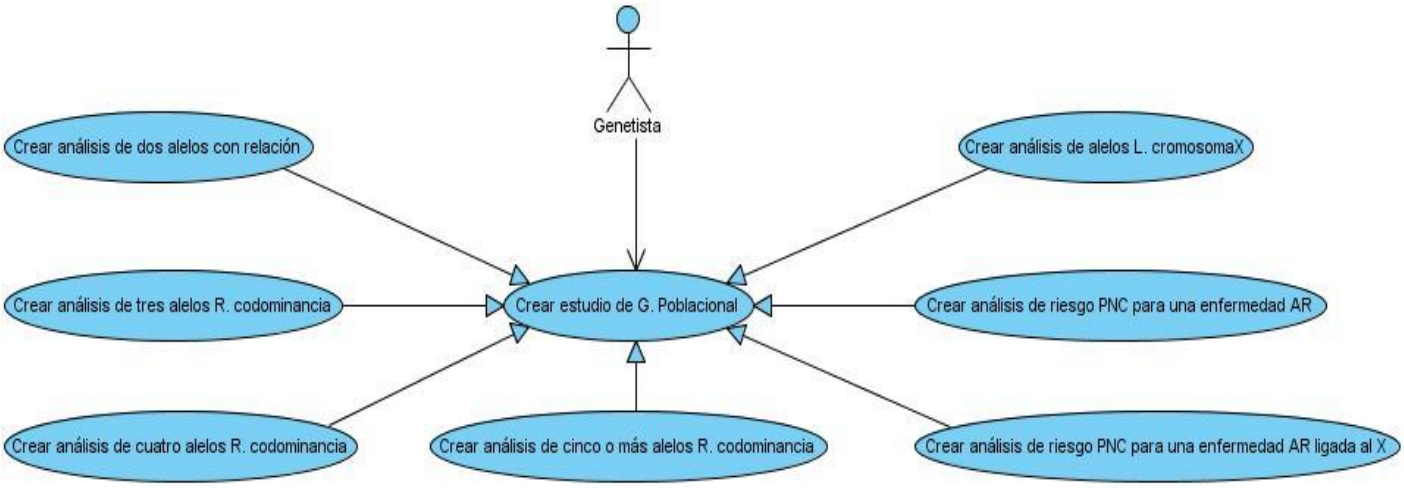

Figura 10 Diagrama de casos de uso del sistema.

2.5.3. Descripción de los casos de uso del sistema

Caso de Uso 1:	**Crear estudio de Genética Poblacional**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *Crear estudio*. El sistema muestra la interfaz correspondiente. Se introducen los datos. El caso de uso finaliza cuando se selecciona algún estudio específico de Genética Poblacional.
Referencias	**RF 1.1**
Pos condiciones	Queda creado un estudio específico de Genética Poblacional.

Tabla 5 Breve descripción del caso de uso del sistema: Crear estudio de Genética Poblacional.

Caso de Uso 2:	**Crear análisis de dos alelos con relación**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *Análisis de dos alelos con relación* en el menú. El sistema muestra la interfaz correspondiente. Se introducen los datos necesarios para la creación del estudio y se obtienen los resultados del mismo. El caso de uso finaliza cuando se obtienen los resultados del estudio.
Referencias	**RF 2.1, RF 2.2, RF 2.3 CUS 1**
Pos condiciones	Queda creado un análisis de tres alelos con relación de codominancia.

Tabla 6 Breve descripción del caso de uso del sistema: Crear análisis de dos alelos con relación.

Caso de Uso 3:	**Crear análisis de tres alelos con relación de codominancia**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *Análisis de tres alelos con relación* en el menú. El sistema muestra la interfaz correspondiente. Se introducen los datos necesarios para la creación del estudio y se obtienen los resultados del mismo. El caso de uso finaliza cuando se obtienen los resultados del estudio.
Referencias	**RF 3.1, RF 3.2, RF 3.3, CUS 1**

Pos condiciones	Queda creado un análisis de tres alelos con relación de codominancia.

Tabla 7 Breve descripción del caso de uso del sistema: Crear análisis de tres alelos con relación de codominancia.

Caso de Uso 4:	**Crear análisis de cuatro alelos con relación de codominancia**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *Análisis de cuatro alelos con relación* en el menú. El sistema muestra la interfaz correspondiente. Se introducen los datos necesarios para la creación del estudio y se obtienen los resultados del mismo. El caso de uso finaliza cuando se obtienen los resultados del estudio.
Referencias	**RF 4.1, RF 4.2, RF 4.3, CUS 1**
Pos condiciones	Queda creado un análisis de tres alelos con relación de codominancia.

Tabla 8 Breve descripción del caso de uso del sistema: Crear análisis de cuatro alelos con relación de codominancia.

Caso de Uso 5:	**Crear análisis de cinco o más alelos con relación de codominancia**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *Análisis de tres alelos con relación* en el menú. El sistema muestra la interfaz correspondiente. Se introducen los datos necesarios para la creación del estudio y se obtienen los resultados del mismo. El caso de uso finaliza cuando se obtienen los resultados del estudio.
Referencias	**RF 5.1, RF 5.2, RF 5.3, CUS 1**
Pos condiciones	Queda creado un análisis de cinco o más alelos con relación de codominancia.

Tabla 9 Breve descripción del caso de uso del sistema: Crear análisis de cinco o más alelos con relación de codominancia.

Caso de Uso 6:	**Crear análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *Análisis de*

	riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva en el menú. El sistema muestra la interfaz correspondiente. Se introducen los datos necesarios para la creación del estudio y se obtienen los resultados del mismo. El caso de uso finaliza cuando se obtienen los resultados del estudio.
Referencias	**RF 6.1, RF 6.2, RF 6.3, CUS 1**
Pos condiciones	Queda creado un análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva.

Tabla 10 Breve descripción del caso de uso del sistema: Análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva.

Caso de Uso 7:	**Crear análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *Análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X* en el menú. El sistema muestra la interfaz correspondiente. Se introducen los datos necesarios para la creación del estudio y se obtienen los resultados del mismo. El caso de uso finaliza cuando se obtienen los resultados del estudio.
Referencias	**RF 7.1, RF 7.2, RF 7.3, CUS 1**
Pos condiciones	Queda creado un análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X.

Tabla 11 Breve descripción del caso de uso del sistema: Crear análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X.

Caso de Uso 8:	**Crear análisis de alelos ligados al cromosoma X**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *Análisis de alelos ligados al cromosoma X a partir de datos pre analizados* en el menú.

	El sistema muestra la interfaz correspondiente. Se introducen los datos necesarios para la creación del estudio y se obtienen los resultados del mismo. El caso de uso finaliza cuando se obtienen los resultados del estudio.
Referencias	**RF 8.1, RF 8.2, RF 8.3, CUS 1**
Pos condiciones	Queda creado un análisis de alelos ligados al cromosoma X.

Tabla 12 Breve descripción del caso de uso del sistema: Crear análisis de alelos ligados al cromosoma X.

Caso de Uso 9:	**Guardar resultado del estudio**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *salvar estudio*. El caso de uso finaliza cuando se salva el estudio.
Pre condiciones:	Solo se puede salvar un estudio cuando se insertan datos o cuando se obtienen resultados.
Referencias	**RF 9.1, CUS 2, CUS 3, CUS 4, CUS 5, CUS 6, CUS 7, CUS 8**
Pos condiciones	Se salva el estudio realizado.

Tabla 13 Breve descripción del caso de uso del sistema: Guardar estudio.

Caso de Uso 10:	**Cargar resultado del estudio**
Actores:	**Genetista(inicia)**
Resumen:	El caso de uso se inicia cuando el genetista selecciona la opción *cargar estudio*. El caso de uso finaliza cuando se carga el estudio.
Pre condiciones:	Solo se puede cargar un estudio que exista.
Referencias	**RF 10.1, CUS 1**
Pos condiciones	Se carga el estudio realizado.

Tabla 14 Breve descripción del caso de uso del sistema: Cargar estudio.

Para consultar la descripción detallada de los casos de uso del sistema remitirse al expediente de proyecto, específicamente la plantilla Modelo del sistema v2.0.

2.6. Conclusiones del capítulo

En este capítulo quedaron identificados los actores y trabajadores del negocio, se redactaron las reglas del negocio, se realizó el diagrama de actividades y el modelo de objetos. Además se hizo una descripción detallada del caso de uso del negocio. Se definieron los requisitos funcionales y no funcionales que presenta la aplicación informática, se agruparon los requisitos por casos de uso y se elaboró el diagrama de casos de uso del sistema con el actor que se definió para ello. También se describieron los casos de uso del sistema.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

En este capítulo se describe el diseño del sistema, evidenciándose el uso de patrones de diseño aplicados para el desarrollo del mismo. Se modelan las clases del diseño y los diagramas de interacción en correspondencia con la realización de los casos de uso descritos en el capítulo anterior. Se definirá el diagrama de despliegue de la aplicación.

3.1. Aplicación de patrones de diseño

Un patrón de diseño constituye un esquema para refinar subsistemas o componentes. Es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. Un patrón de diseño identifica: clases, instancias, roles, colaboraciones y la distribución de responsabilidades, además que ayuda a construir clases y a estructurar sistemas de clases. Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Con la aplicación de los patrones de diseño se reutilizan las experiencias de otros desarrolladores ya que estos patrones están basados en la recopilación del conocimiento de los expertos en desarrollo de software.

3.1.1. Patrones GRASP aplicados

Experto: este patrón tiene como propósito asignar una responsabilidad al experto en información: la clase que tiene la información necesaria para cumplirla. En la siguiente figura se muestra como la clase Estudio_Gen_Poblacional que es la que contiene la información sobre el estudio de genética poblacional tiene asignada la responsabilidad de realizar todo el estudio a través del método analisisAlelos().

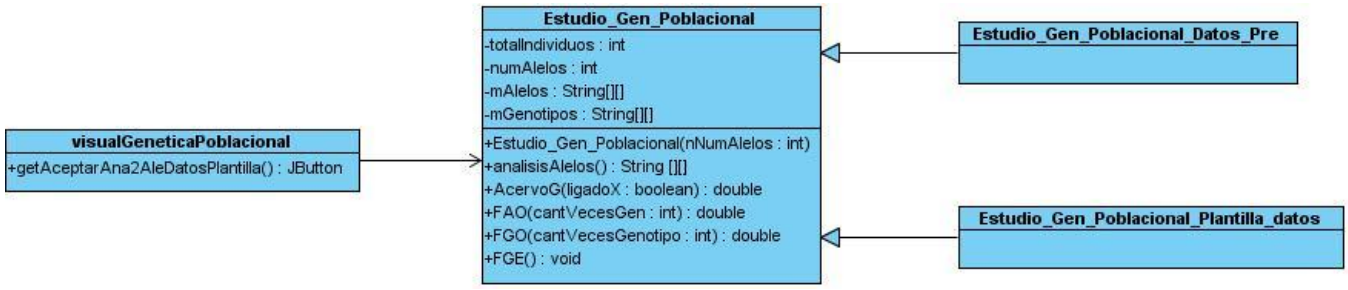

Figura 11 Patrón GRASP Experto.

Creador: éste guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que permita conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. En la figura se muestra que la clase visualGeneticaPoblacional es la encargada de crear instancias de la clase Estudio_Gen_Poblacional. A continuación se muestra la creación de un objeto de una de las clases hijas de la clase Estudio_Gen_Poblacional.

```
Estudio_Gen_Poblacional estudio=new Estudio_Gen_Poblacional_Datos_Pre(numAlelos);
lgenotipos=estudio.getGenerarGenotipos(arrAlelos,2);
```

Figura 12 Fragmentos de la inicialización de un objeto.

Bajo Acoplamiento: éste guía la asignación de responsabilidades de forma tal que las clases se comuniquen con el menor número de clases que sea posible, de manera que de producirse una modificación en algunas de ellas se tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización, y disminuyendo la dependencia entre las mismas. En la figura 14 se evidencia este patrón.

Alta Cohesión: éste mantiene la complejidad dentro de límites manejables y garantiza que las clases con responsabilidades estrechamente relacionadas no realicen un trabajo enorme. En la siguiente figura se evidencia con claridad el uso de este patrón.

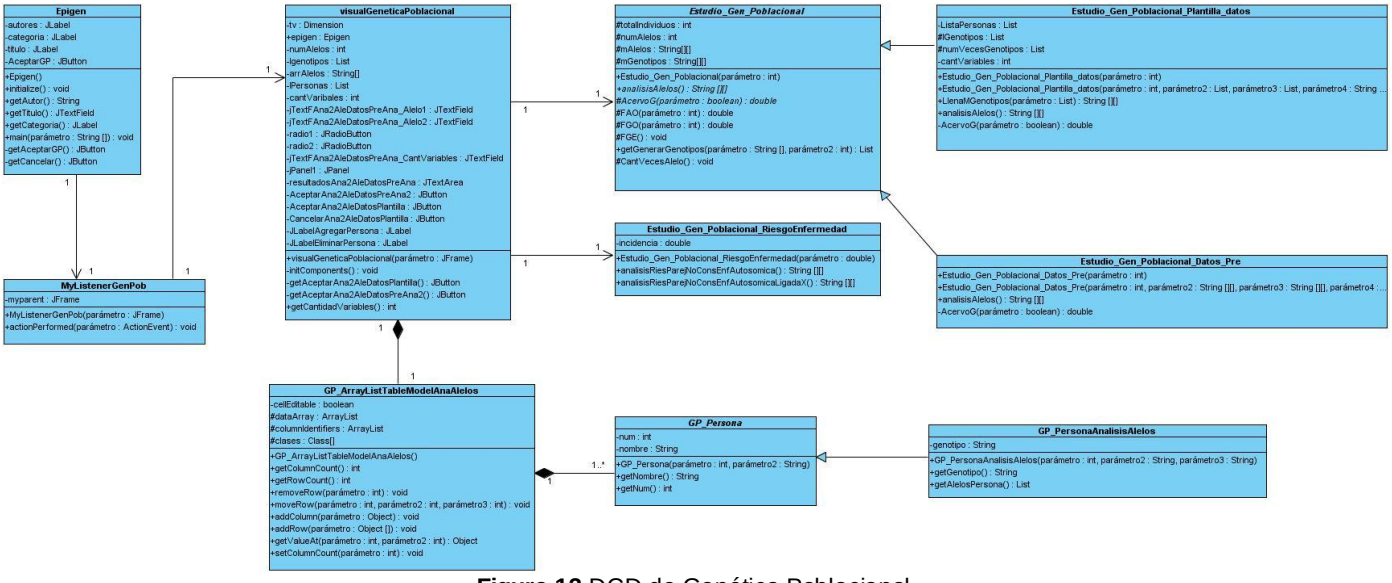

Figura 13 DCD de Genética Poblacional.

Polimorfismo: cuando varía el tipo (clase) de alternativas o comportamientos relacionados, asignar la responsabilidad del comportamiento mediante operaciones polimórficas a los tipos en que varía el comportamiento. [16] En la figura 15 se observa claramente el uso de este patrón.

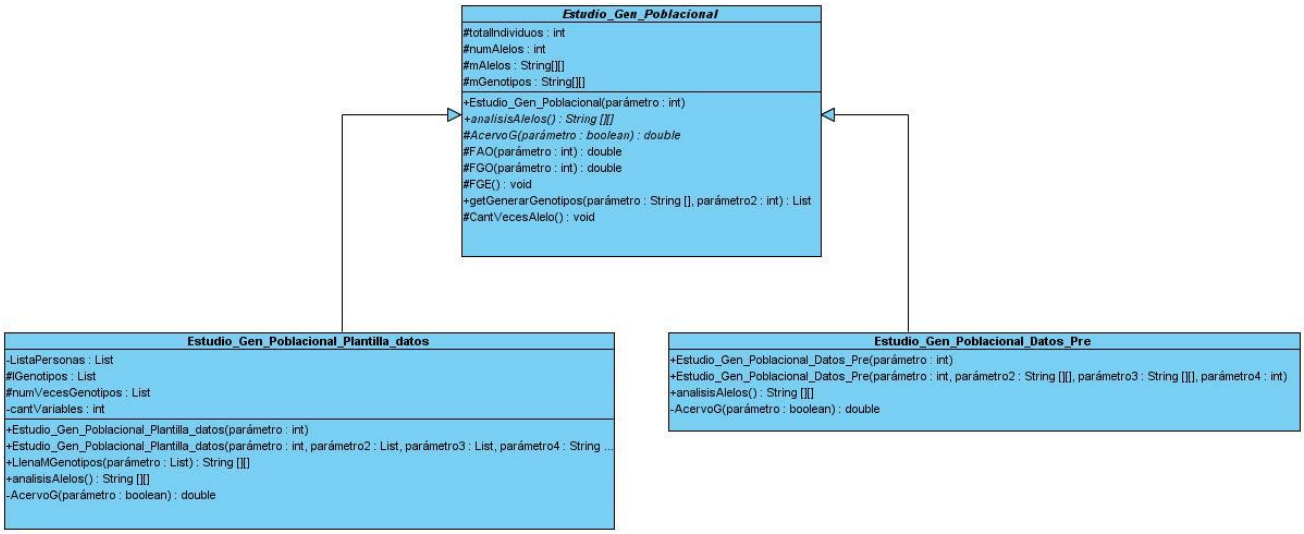

Figura 14 Patrón GRASP Polimorfismo.

3.1.2. Patrones GOF aplicados

Facade (Fachada): este patrón define una sola clase que unifique las interfaces y le asigne la responsabilidad de colaborar con el subsistema. En la figura se muestra que el usuario para poder acceder al estudio de Genética Poblacional primero accede a la clase Epigen que está se encarga del acceso a las demás interfaces como son Genética Poblacional, Epidemiología Genética o Genética Poblacional.

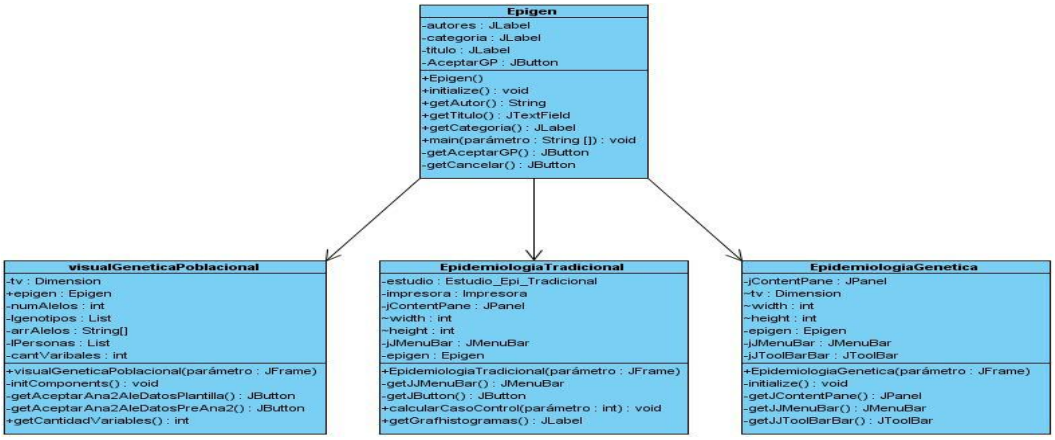

Figura 15 Patrón GOF Fachada.

3.2. Patrón de Arquitectura en Capas

Conociendo que la arquitectura es la base para cualquier aplicación se decidió utilizar el patrón de arquitectura en capas. Según las características que tiene la aplicación se definieron dos capas solamente, la capa de presentación y la capa de negocio. No se identificó para la aplicación la capa de acceso a datos porque no se requiere la persistencia de datos en una base de datos.

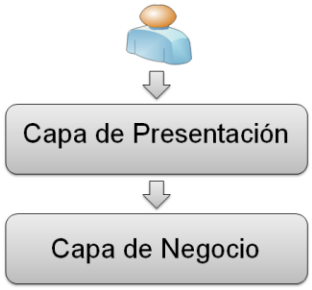

Figura 16 Arquitectura en capas.

Capa de presentación: es la que ve el usuario (también se le denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información en un mínimo de procesos (realiza un filtrado previo para comprobar que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio. También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario.

Capa de negocio: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados.

3.3. Diagramas de clases del diseño

Un diagrama de clases del diseño (DCD) representa una abstracción de una o varias clases en la implementación del sistema, dependiendo del lenguaje de programación. Las clases definen los objetos, con los cuales se implementan los casos de uso. Las particularidades del lenguaje de programación influyen en el diseño de las mismas. Como la aplicación alasEPIGEN es una aplicación de escritorio y no usa ninguna base de datos, no se necesitarán de estereotipos para el diseño. A continuación se muestran los diagramas de clases de los casos de uso dos, seis y ocho.

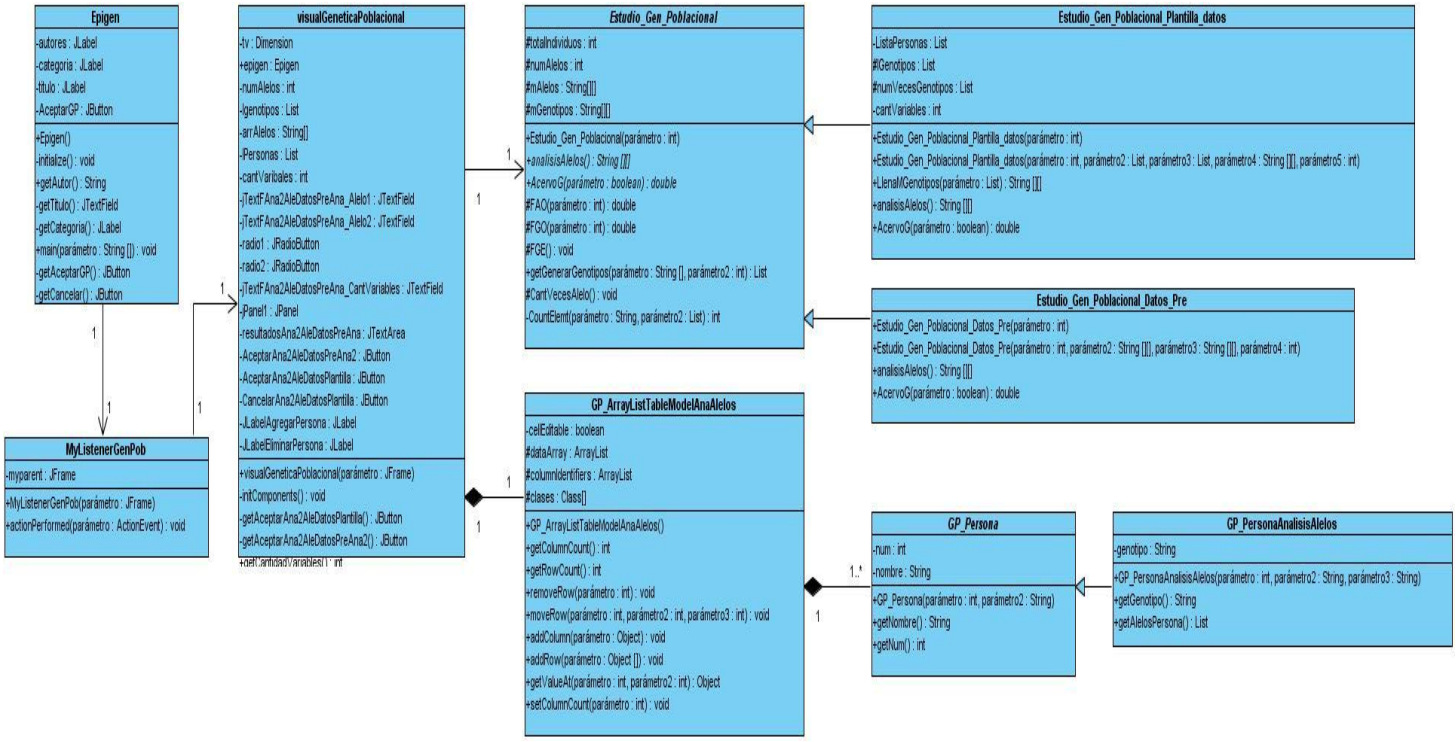

Figura 17 DCD CU 2 Crear análisis de dos alelos con relación.

Para consultar los restantes diagramas de clases del diseño remitirse al expediente de proyecto, específicamente a la plantilla Modelo de diseño v2.0.

3.4. Descripción de las clases del diseño

3.4.1. Clase presente en la capa de Negocio

Nombre: Estudio_Gen_Poblacional	
Tipo de clase: Clase	
Responsabilidades: Es la responsable de crear las instancias de Estudio de Genética Poblacional.	
Nombre:	**Estudio_Gen_Poblacional(int nNumAlelos)**
Descripción:	Es el método que construye la clase.
Nombre:	**analisisAlelos()**
Descripción:	Realiza el análisis de alelos.
Nombre:	**AcervoG(boolean ligadoX)**

Descripción:	Calcula el acervo genético para ligado al X.
Nombre:	**FAO(int cantVecesGen)**
Descripción:	Calcula la frecuencia génica o alélica observada a partir de una plantilla de datos.
Nombre:	**FGO(int cantVecesGenotipo)**
Descripción:	Calcula la Frecuencia Genotípica observada a partir de datos pre analizados.
Nombre:	**FGE()**
Descripción:	Calcula la Frecuencia Genotípica esperada.
Nombre:	**getGenerarGenotipos(String[] arrAlelos, int tipo)**
Descripción:	Genera los genotipos a partir de alelos.
Nombre:	**CantVecesAlelo()**
Descripción:	Cantidad de veces en que está presente un alelo en los genotipos de todas las personas para los datos pre analizados.
Nombre:	**CountElemt(String element, List<String> lista)**
Descripción:	Cuenta la cantidad de veces que está un elemento string en una lista string.

Tabla 15 Descripción de la clase Estudio_Gen_Poblacional.

Para consultar las restantes descripciones de las clases del diseño remitirse al expediente de proyecto, específicamente a la plantilla Modelo de diseño v2.0.

3.5. Diagramas de secuencias

Un diagrama de interacción muestra una interacción que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos.

Un diagrama de secuencia (DS) no es más que un diagrama de interacción que destaca la ordenación temporal de los mensajes. Los mismos están compuestos por objetos, línea de vida y foco de control. La línea de vida representa la existencia de un objeto a lo largo de un período de tiempo. El foco de control representa el período de tiempo durante el cual un objeto ejecuta una acción. A continuación se muestran varios diagramas de secuencia que se realizaron por cada sección de las descripciones de los casos de uso.

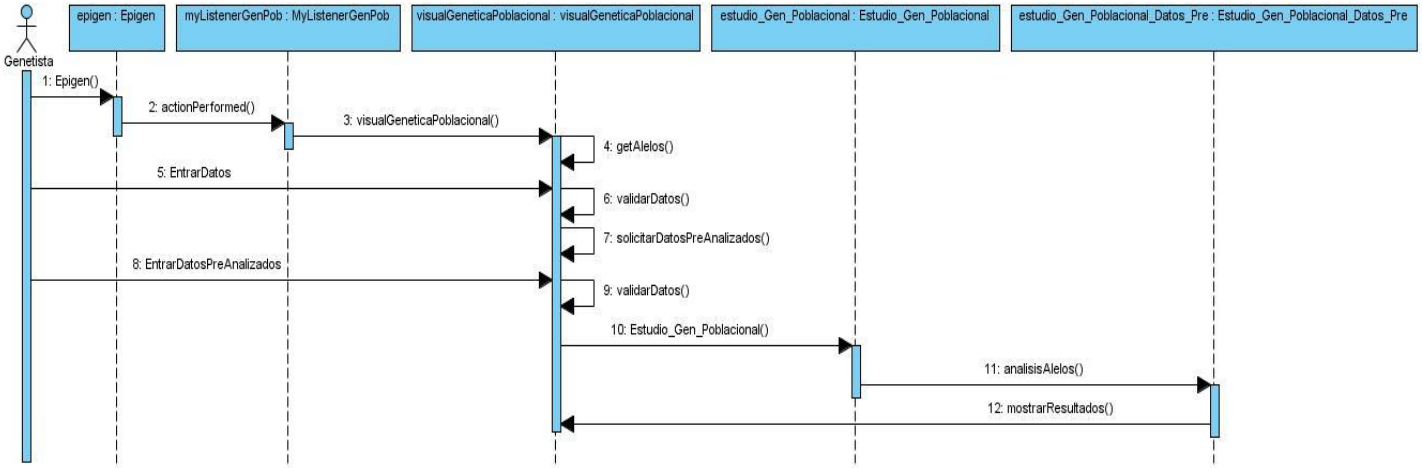

Figura 18 DS CU 2 Crear análisis de dos alelos con relación a partir de datos pre analizados.

Para consultar los restantes diagramas de secuencia remitirse al expediente de proyecto, específicamente a la plantilla Modelo de diseño v2.0.

3.6. Modelo de despliegue

El Modelo de Despliegue provee un modelo detallado de la forma en la que los componentes se desplegarán a lo largo de la infraestructura del sistema. Detalla las capacidades de red, las especificaciones del servidor, los requisitos de hardware y otras informaciones relacionadas al despliegue del sistema propuesto. Esta vista representa el mapeo de componentes de software ejecutables con los nodos de procesamiento (hardware), tiene en cuenta los siguientes requerimientos: disponibilidad del sistema, rendimiento y escalabilidad.

Debido a que la aplicación es de escritorio ya que los genetistas necesitan utilizarla en cualquier estación de trabajo sin importar el lugar donde se encuentren, no debe estar conectada a la red y solo intervienen como nodo principal la computadora cliente donde esté instalada la aplicación y una impresora que se utilizará solo en el caso que se desee imprimir un reporte de algún estudio epidemiológico.

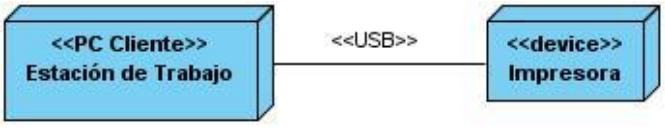

Figura 19 Diagrama de despliegue de alasEPIGEN v2.0.

3.7. Conclusiones del capítulo

En este capítulo se dieron a conocer los principales elementos del diseño para el desarrollo de la aplicación alasEPIGEN v2.0. Se explicaron y ejemplificaron los patrones de arquitectura y diseño utilizados, quedaron representados los diagramas de clases y secuencia correspondientes para cada caso de uso del sistema. Para una mejor comprensión de las clases del diseño se describieron las funciones principales de cada clase que son las responsables de modelar la forma en que fue implementada la aplicación. Se definió el diagrama de despliegue de la aplicación.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

En este capítulo se representan los diagramas de componentes y el diagrama de despliegue del sistema. Además se brinda una descripción de los principales métodos implementados, se muestran imágenes de la interfaz de la aplicación. Se hace referencia a la validación de la aplicación y se incluye el modelo de prueba con la descripción de los casos de prueba basados en casos de uso.

4.1. Diagrama de Componentes

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

Para poder entender los diagramas de componentes que se muestran a continuación, primero debe conocerse cuales son las clases contenidas en los componentes. A continuación se relaciona cada componente con las clases que representa:

Epigen.java: este componente solamente contiene la clase Epigen.

visualGeneticaPoblacional.java: este componente solamente contiene la clase visualGeneticaPoblacional.

GP_ArrayListTableModelAnaAlelos.java: este componente solamente contiene la clase GP_ArrayListTableModelAnaAlelos.

GP_Persona.java: este componente contiene las clases GP_Persona y GP_PersonaAnalisisAlelos.

Estudio_Gen_Poblacional.java: este componente contiene las clases Estudio_Gen_Poblacional, Estudio_Gen_Poblacional_Plantilla_datos y Estudio_Gen_Poblacional_Datos_Pre.

Estudio_Gen_Poblacional_RiesgoEnfermedad.java: este componente solamente contiene la clase Estudio_Gen_Poblacional_RiesgoEnfermedad.

El subsistema Presentación recoge todas las clases visuales del módulo de Genética Poblacional.

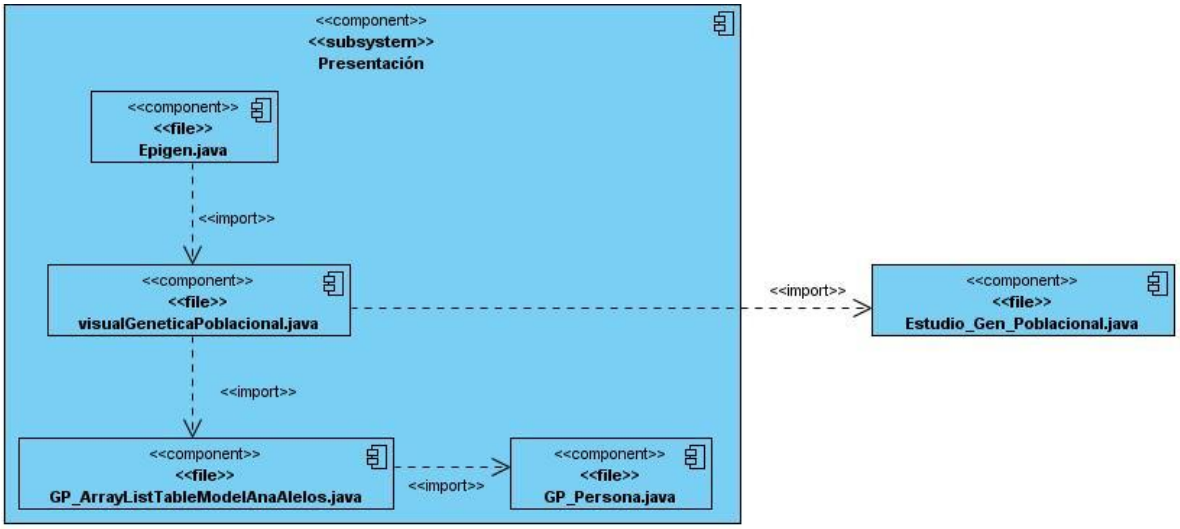

Figura 20 DC CU 2 Crear análisis de dos alelos con relación.

4.1.1. Seguridad de la aplicación

Para lograr la seguridad en la aplicación se utilizó la librería Extensión Criptográfica de Java (JCE). Las bibliotecas de JCE proporcionan apoyo para las tareas de cifrar y descifrar datos. Los resultados de los estudios son cifrados a la hora de guardarlos y descifrados cuando se cargan.

4.2. Estándar de Codificación

Un estándar de codificación describe las convenciones que los desarrolladores deben seguir para crear un código fuente descifrable en un sistema. Éste debe ser escrito de la misma forma por todos los desarrolladores, permitiendo mayor efectividad y organización en el trabajo. Permite además obtener un código homogéneo, mejorar la lectura del software permitiendo entender el código nuevo mucho más rápido y más a fondo. El estándar de codificación utilizado en EPIGEN es básicamente el definido en el proyecto MEDIGEN con algunas modificaciones referentes a una aplicación de escritorio. Dicho estándar se define a continuación:

- Los nombres de las clases deben ser sustantivos, cuando son compuestos tendrán la primera letra de cada palabra que lo forma en mayúsculas. Intentar mantener los nombres de las clases lo más simples y descriptivos posibles.

- En los comentarios de las clases debe aparecer el autor de ésta y el objetivo de la misma.
- Los nombres de las variables y métodos tienen que ser lo más descriptivos posibles, procurando que sean palabras en minúsculas con significado claro. Si realmente necesita más de una palabra, póngalas juntas, poniendo la inicial de cada palabra en mayúscula siempre que no sea la primera, pero procure mantenerlas tan breves como sea posible. [18]
- Inicializar las variables locales donde se declaran. La única razón para no inicializar una variable donde se declara es si el valor inicial depende de algunos cálculos que deben ocurrir.
- Utilice nombres en plural para arreglos, listas o matrices de objetos. [18]
- Todos los métodos y clases deben tener comentarios de documentación y comentarios de implementación. Los comentarios deben ser añadidos de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones o variables.
- Usar paréntesis en expresiones que implican distintos operadores para evitar problemas con el orden de precedencia de los operadores. Incluso si parece claro el orden de precedencia de los operadores, podría no ser así para otros, no se debe asumir que otros programadores conozcan el orden de precedencia.
- Las líneas en blanco mejoran la facilidad de lectura separando secciones de código que están lógicamente relacionadas por lo que se debe usar siempre una línea en blanco entre métodos.

4.3. Fragmento de código fuente

```
/**
 * Cantidad de veces en que esta presente un alelo en
 * los genotipos de todas las personas.
 * Para los datos pre analizados
*/
protected void CantVecesAlelo(){
    for(int i = 0; i < mGenotipos.length; i++){
      if(Integer.parseInt(mGenotipos[i][1])>0){
        List<String> tempAlelos=new LinkedList<String>();
        char[] arrC=mGenotipos[i][0].toCharArray();
        String aleloTemp="";
        for(int j=0;j<arrC.length;j++){
            if(arrC[j]=='.'){
                tempAlelos.add(aleloTemp);
                aleloTemp="";
            }
            else
                aleloTemp+=arrC[j];
        }
        for (int j = 0; j < mAlelos.length; j++) {
           mAlelos[j][1]=Integer.toString(Integer.parseInt(mAlelos[j][1])
           +CountElemt(mAlelos[j][0],tempAlelos)*
           Integer.parseInt(mGenotipos[i][1]));
        }
      }
    }
}
```

Figura 21 Fragmento de código fuente.

4.4. Interfaces de la aplicación

Esta es la interfaz principal de la aplicación en la misma se escoge el tipo de estudio a realizar que puede ser Epidemiología Genética, Epidemiología Tradicional o Genética Poblacional.

Figura 22 Interfaz Principal de la aplicación.

Esta interfaz es la principal del estudio de Genética Poblacional en el mismo se encuentra el menú para acceder a los distintos análisis de este estudio que se deseen realizar.

Figura 23 Interfaz principal del estudio de Genética Poblacional.

Esta interfaz es para el análisis de dos alelos con relación a partir de una plantilla de datos. Esta interfaz es la misma que se utiliza para los casos de tres y cuatro alelos.

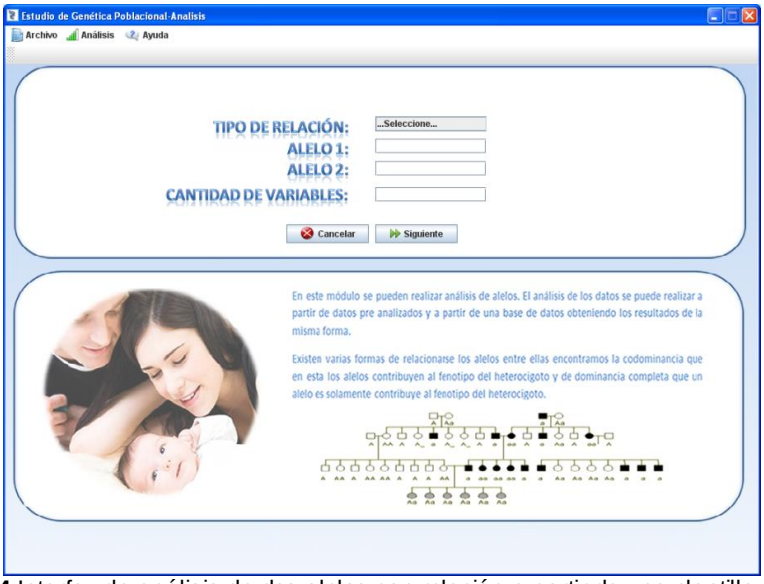

Figura 24 Interfaz de análisis de dos alelos con relación a partir de una plantilla de datos.

Esta interfaz es para el análisis de dos alelos con relación a partir de datos pre analizados. Esta interfaz es la misma que se utiliza para los casos de tres y cuatro alelos.

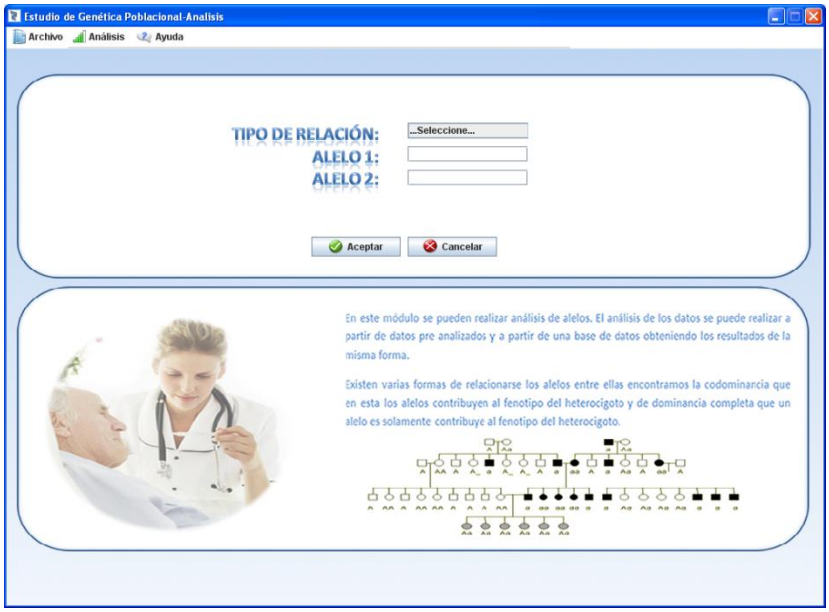

Figura 25 Interfaz de análisis de dos alelos con relación a partir de datos pre analizados.

Esta interfaz es para el análisis de cinco o más alelos con relación de codominancia a partir de una plantilla de datos.

Figura 26 Interfaz de análisis de dos alelos con relación a partir de datos pre analizados.

Esta interfaz es para el análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva. En el caso del análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X solamente cambia el nombre de la variable a entrar que en ese caso sería frecuencia de la enfermedad en varones.

Figura 27 Interfaz de análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva.

Esta interfaz es para el análisis de alelos ligados al cromosoma X a partir de datos pre analizados. En el caso del análisis de alelos ligados al cromosoma X a partir de una plantilla de datos solamente se le agrega la cantidad de variables.

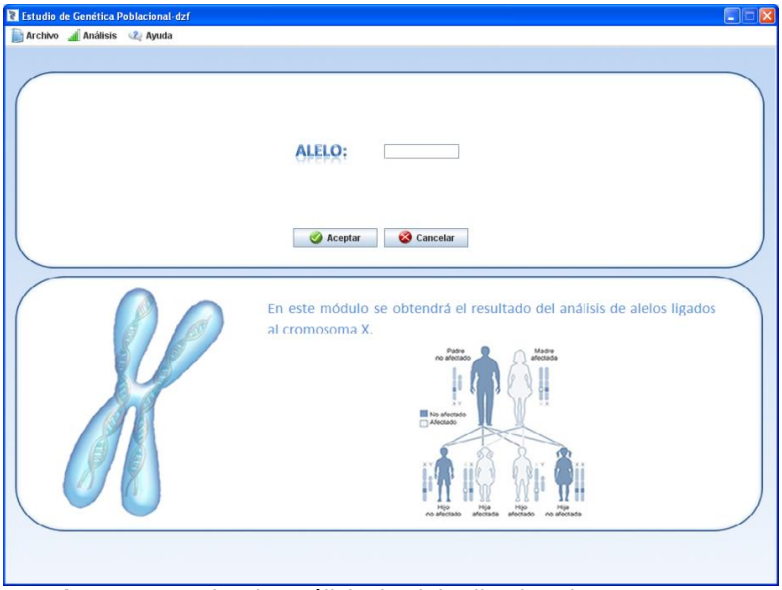

Figura 28 Interfaz de análisis de alelos ligados al cromosoma X.

4.5. Pruebas

Las pruebas son una actividad en la cual un sistema o componente es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente.[19]

La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación.

Para poder realizar pruebas a un software se necesita tener claro conceptos como: niveles de prueba, tipos de prueba y los métodos de prueba. Las pruebas se aplican para diferentes tipos de objetivos, en diferentes niveles de trabajo o escenarios. A continuación se mencionan distintos niveles de prueba:[19]

- **Prueba de desarrollador:** Es la prueba diseñada e implementada por el equipo de desarrollo.
- **Prueba independiente:** Es la prueba que es diseñada e implementada por alguien independiente del grupo de desarrolladores.
- **Prueba de Unidad:** Es la prueba enfocada a los elementos probables más pequeños del software.

- **Prueba de Integración:** Es ejecutada para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso.
- **Prueba de sistema:** Son las pruebas que se hacen cuando el software está funcionando como un todo.
- **Prueba de aceptación:** Prueba de aceptación del usuario es la prueba final antes del despliegue del sistema.

Cada tipo de prueba tiene un objetivo específico y una técnica que lo soporte. A continuación se muestra una tabla con los tipos de pruebas basados en el atributo de calidad Funcionalidad.

Dimensión de Calidad **Riesgo de calidad**	**Tipos de Prueba**
Funcionalidad	Función: Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso. Seguridad: Asegurar que los datos o el sistema solamente es accedido por los actores deseados. Volumen: Enfocada en verificar las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la BD.

Tabla 16 Tipos de pruebas basadas en dimensiones de calidad.[19]

Existen dos métodos de pruebas fundamentales: el método de caja blanca y el método de caja negra.[19]

- Método de caja blanca: Se comprueban los caminos lógicos del software proponiendo casos de prueba que examinen que están correctas todas las condiciones y/o bucles para determinar si el estado real coincide con el esperado o afirmado.
- Método de caja negra: se refiere a las pruebas que se llevan a cabo sobre la interfaz del software.

4.5.1. Diseño de casos de prueba y registro de no conformidades

Un caso de prueba se diseña según las funcionalidades descritas en los casos de usos. Este diseño se elabora previamente a realizar las pruebas funcionales a la aplicación. Se parte de la descripción de los casos de uso del sistema, como apoyo para las revisiones. Cada planilla de caso de prueba recoge la especificación de un caso de uso, dividido en secciones y escenarios, detallando las funcionalidades descritas en él y escribiendo cada variable que recoge el caso de uso. Existen casos de pruebas para los diferentes métodos: Caja Negra y Caja Blanca. En el proceso de pruebas se hace uso de los casos de prueba de Caja Negra por ser éste el método empleado en la liberación del producto software, de ahí el motivo por el que se diseña un caso de prueba por caso de uso del sistema.

Partiendo de la descripción de los casos de uso del sistema, como apoyo para las revisiones, se diseñó un caso de prueba asociado a cada caso de uso. Para detallar el caso de uso se utiliza una tabla, donde se desglosa esta funcionalidad en secciones y a su vez éstas en escenarios, para hacer más fructífera la ejecución de las pruebas. Esta tabla contiene los campos:

- Nombre de la sección: Se especifica el nombre de la sección [SC 1: Nombre de la sección].
- Escenarios de la sección: Se especifican los escenarios de cada sección [EC 1.1: Nombre del Escenario].
- Descripción de la funcionalidad: Se describe brevemente la funcionalidad del escenario.

El siguiente es un ejemplo donde se detalla el caso de uso Crear análisis de dos alelos con relación.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Crear análisis de dos alelos con relación de dominancia completa a partir de una plantilla de datos	EC 1.1: El genetista inserta los datos.	En este escenario el genetista inserta los datos.
	EC 1.2: El genetista inserta los datos incorrectamente.	Este escenario sigue la misma funcionalidad que el anterior verificando que todos los datos estén correctamente insertados.
	EC 1.3: El genetista no inserta los datos necesarios.	Este escenario sigue la misma funcionalidad que el primero pero verifica que todos los datos estén insertados.
	EC 1.4: El genetista no inserta variables.	Este escenario verifica que el genetista inserte variables dentro del estudio.

Tabla 17 CP Crear análisis de dos alelos con relación.

A partir de esta descripción se detallan las variables que se encuentran en las interfaces asociadas al caso de uso que se le diseñó el caso de prueba. En la siguiente tabla se muestra un ejemplo de la descripción de variables del caso de uso Crear análisis de dos alelos con relación.

No	Nombre del campo	Clasificación	Valor nulo	Descripción
Dominancia completa a partir de una plantilla de datos				
1	Tipo de relación	Lista de selección	No	Se introduce el tipo de relación entre alelos que tiene que ser dominancia completa.
2	Alelo 1	Texto	No	Se introduce el nombre del alelo 1.
3	Alelo 2	Texto	No	Se introduce el nombre del alelo 2.
4	Cantidad de variables	Entero	No	Se introduce la cantidad de variables.
5	Alelo dominante	Checkbox	No	Se tiene que escoger el alelo dominante.

Tabla 18 Descripción de variables.

Los resultados de las pruebas que no fueron satisfactorios pasaron a ser no conformidades y se emitieron en el registro de defectos y dificultades detectados que se encuentra en la parte final de cada diseño de caso de prueba. En la siguiente tabla se muestran algunas de las no conformidades encontradas tras la revisión del software.

Elemento	No	No conformidad	Ubicación de la NC	Etapa de detección	Significativa	No Significativa	Recomendación	Estado NC	Respuesta del Equipo Desarrollo
Aplicación	1	En la inserción de alelos este campo acepta símbolos del teclado por ejemplo(+, -, etc)	Crear análisis de dos alelos con relación de dominancia completa a partir de una plantilla de datos. Campos: alelo 1 y alelo 2.	Pruebas de funcionalidad y confiabilidad	X			28 de abril de 2010.	
Aplicación	2	Cuando se cancela un estudio y se desea realizar otro del mismo tipo la aplicación muestra en agregar personas las que antes se habían insertado y las nuevas variables no las muestra.	Crear análisis de dos alelos con relación de dominancia completa a partir de una plantilla de datos. Campos: alelo 1 y alelo 2.	Pruebas de funcionalidad y confiabilidad	X			29 de abril de 2010.	
Aplicación	3	Cuando se cancela un estudio y se desea realizar otro del mismo tipo la aplicación no permite eliminar las personas insertadas y tampoco agrega a otra persona.	Crear análisis de dos alelos con relación de dominancia completa a partir de una plantilla de datos. Campos: alelo 1 y alelo 2.	Pruebas de funcionalidad y confiabilidad	X			29 de abril de 2010.	

Figura 29 Fragmento de no conformidades encontradas a la aplicación.

4.6. Conclusiones del capítulo.

Con la realización del presente capítulo se describió como fue implementada la aplicación en términos de componentes de cada uno de los casos de uso del sistema, los cuales están basados en los casos de uso realizados durante el diseño. Al mismo tiempo se brindó una breve descripción de los principales métodos implementados. Se definió el estándar de codificación utilizado en la implementación. Además se realizaron pruebas funcionales al software tanto a nivel de desarrollador, como a nivel independiente y como a nivel de unidad.

CONCLUSIONES

El desarrollo de la aplicación alasEPIGEN v2.0 constituirá un importante aporte al desarrollo de la genética en Cuba ya que le permitirá a los especialistas tener un mayor control de las enfermedades en las poblaciones y de esta forma poder tomar medidas en beneficio de las personas que las conforman.

Los resultados de los estudios brindados por dicha aplicación constituirán una valiosa fuente de información para la investigación y la toma de decisiones en función del beneficio de cada uno de los implicados en estos estudios de Genética Poblacional.

RECOMENDACIONES

Como parte del proceso investigativo que se llevó a cabo, han surgido algunas ideas que sería recomendable tener en cuenta para el mejoramiento del sistema.

Se recomienda agregar el estudio de consanguinidad que se encarga de analizar el apareamiento de individuos que están relacionados entre sí más estrechamente que el promedio de la población, es decir, el acoplamiento de individuos que tienen uno o más antepasados en común.

Al no existir en Cuba un software que agrupe todas las funcionalidades que contiene éste dentro de la rama de la Epidemiología Genética, sería conveniente su despliegue por todos los centros de genética del país.

REFERENCIAS BIBLIOGRÁFICAS

[1]. *La Epidemiología Genética: disciplina científica en expansión.* **WYSZYNSKI, Diego F.** 3, s.l. : Pan Am, 1998, Rev Panam Salud Publica, Vol. Public Health, págs. 26, 27.

[2]. **FONTELA GONZÁLEZ, Dioleisys y GUERRA MACHADO, Leinys.** *EPIGEN: "Aplicación informática para el análisis estadístico en estudios de Epidemiología Genética".* Ciudad Habana : s.n., 2009.

[3]. **Company, IBM.** IBM SPSS Statistics. [En línea] 2010. [Citado el: 7 de 1 de 2010.] http://www.spss.com/es/statistics/.

[4]. **StatSoft.** Statistica. [En línea] 2010. [Citado el: 8 de 1 de 2010.] http://www.statsoft.com/.

[5]. Arlequín 3.11. [En línea] 2007. [Citado el: 11 de 1 de 20010.] http://cmpg.unibe.ch/software/arlequin3/.

[6]. **PÉREZ, M. I. SANTIAGO, BARBEITO, G. Naveira y 4, Equipo de Epidat.** *EPIDAT 4.0: UNA HERRAMIENTA DE APOYO PARA LA ENSEÑANZA DE LA ESTADÍSTICA.* 2009.

[7]. **JACOBSON, Ivar, BOOCH, Grady y RUMBAUGH, James.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Educación, S.A, 2000. 84-7829-036-2.

[8]. **RUMBAUGH, James, JACOBSON, Ivar y BOOCH, Grady.** *El Lenguaje Unificado de Modelado.* Madrid : Pearson Educación, 2000.

[9]. *Ayuda extendida del Rational Rose Enterprise.* 2003.

[10]. Informática. [En línea] [Citado el: 26 de 1 de 2010.]

[11]. Sitio oficial de Visual Paradigm. [En línea] [Citado el: 27 de 1 de 2010.] http://www.visual-paradigm.com/.

[12]. **ZUKOWSKI, John.** *Programación Java 2.* La Habana : Félix Varela, 2007.

[13]. **Corporación Oracle y afiliados.** Sitio oficial del Netbeans. [En línea] [Citado el: 15 de 3 de 2010.] http://netbeans.org/.

[14]. **COLLINS SUSSMAN, Ben, FITZPATRICK, Brian W. y PILATO, C. Michael.** *Version Control with Subversion For Subversion 1.6.* California : s.n., 2009.

[15]. **ÖVERGAARD, Gunnar y PALMKVIST, Karin.** *Use Cases: Patterns and Blueprints.* s.l. : Addison Wesley, 2004.

[16]. **LARMAN, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México : PRENTICE HALL, 1999. 970-17-0261-1.

[17]. CNGM Sitio web Genética Médica. [En línea] [Citado el: 5 de 2 de 2010.] http://www.sld.cu/sitios/genetica/.

[18]. **SÁNCHEZ PERODÍN, Yusdenis**. Línea base de la arquitectura del Sistema de Información de Genética Médica. Universidad de Ciencias Informáticas. Ciudad Habana, 2008.

[19]. Conferencia # 5 Pruebas. Ingeniería del Software II. [En línea] [Citado el: 20/4/10] http://teleformacion.uci.cu

BIBLIOGRAFÍA

Pressman, Roger S. *Ingienería del Softawre Un enfoque práctico.* La Habana : Félix Varela, 2007. Vol. 2.

Meyer, Bertrand. *Construcción de software orientado a objetos.* La Habana : Félix Varela, 2006.

Zukowski, John. *Programación Java 2 .* La Habana : Félix Varela, 2007.

La Epidemiología Genética: disciplina científica en expansión. **Wyszynski, Diego F.** 3, s.l. : Pan Am, 1998, Rev Panam Salud Publica, Vol. Public Health, pp. 26, 27.

Fontela González, Dioleisys and Guerra Machado, Leinys. *EPIGEN: "Aplicación informática para el análisis estadístico en estudios de Epidemiología Genética".* Ciudad Habana : s.n., 2009.

Company, IBM. IBM SPSS Statistics. [Online] 2010. [Cited: 1 7, 2010.] http://www.spss.com/es/statistics/.

StatSoft. Statistica. [Online] 2010. [Cited: 1 8, 2010.] http://www.statsoft.com/.

Arlequín 3.11. [Online] 2007. [Cited: 1 11, 20010.] http://cmpg.unibe.ch/software/arlequin3/.

Pérez, M. I. Santiago, Barbeito, G. Naveira and 4, Equipo de Epidat. *EPIDAT 4.0: UNA HERRAMIENTA DE APOYO PARA LA ENSEÑANZA DE LA ESTADÍSTICA.* 2009.

Jacobson, Ivar, Booch, Grady and Rumbaugh, James. *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Educación, S.A, 2000. 84-7829-036-2.

Ayuda extendida del Rational Rose Enterprise. 2003.

Rumbaugh, James, Jacobson, Ivar and Booch, Grady. *El Lenguaje Unificado de Modelado.* Madrid : Pearson Educación, 2000.

Informática. [Online] [Cited: 1 26, 2010.] http://informatica.gonzalonazareno.org/file.php/8/case.pdf.

Sitio oficial de Visual Paradigm. [Online] [Cited: 1 27, 2010.] http://www.visual-paradigm.com/.

Collins Sussman, Ben, Fitzpatrick, Brian W. and Pilato, C. Michael. *Version Control with Subversion For Subversion 1.6.* California : s.n., 2009.

ÖVERGAARD, Gunnar and PALMKVIST, Karin. *Use Cases: Patterns and Blueprints.* s.l. : Addison Wesley, 2004.

Larman, Craig. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México : PRENTICE HALL, 1999. 970-17-0261-1.

CNGM Sitio web Genética Médica. [Online] [Cited: 2 5, 2010.] http://www.sld.cu/sitios/genetica/.

Sánchez Perodín, Yusdenis. Línea base de la arquitectura del Sistema de Información de Genética Médica. Universidad de Ciencias Informáticas. Ciudad Habana, 2008.

Corporación Oracle y afiliados. Sitio oficial del Netbeans. [En línea] [Citado el: 15 de 3 de 2010.] http://netbeans.org/.

ANEXOS

Anexo 1. Diagramas de clases del diseño.

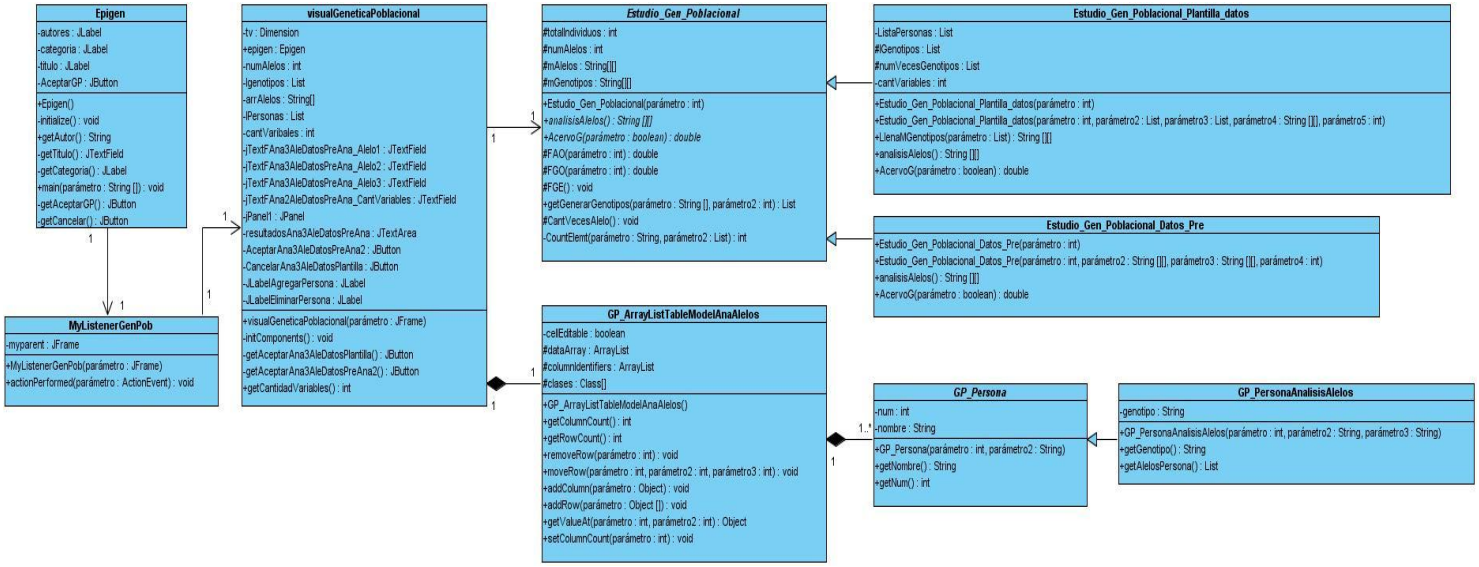

Figura 30 DCD CU 3 Crear análisis de tres alelos con relación de codominancia.

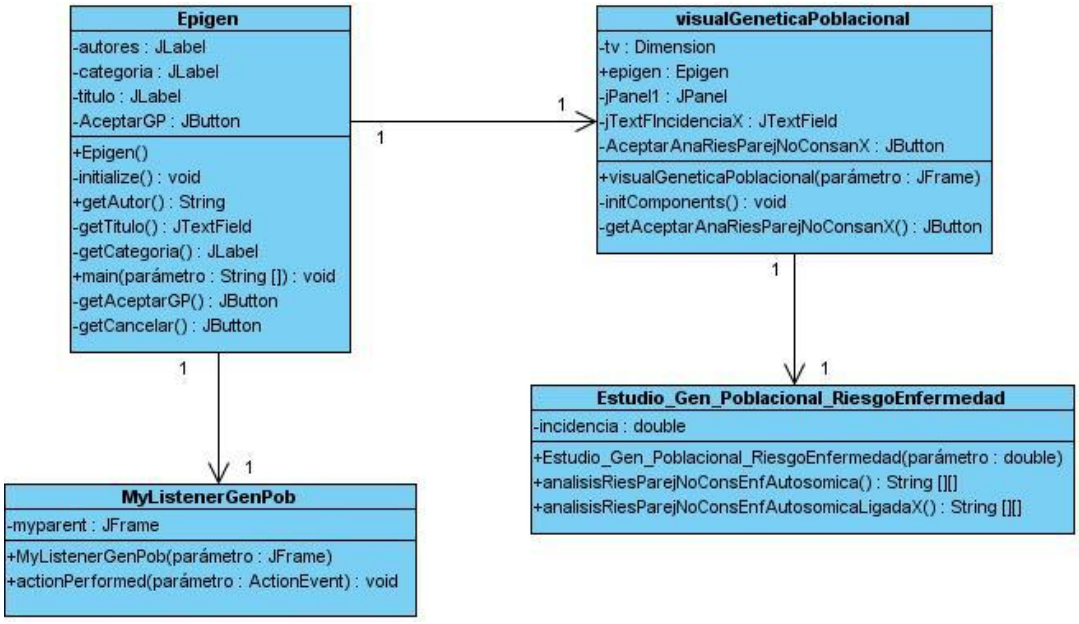

Figura 31 DCD CU 7 Crear análisis de riesgo de una pareja no consanguínea para una enfermedad autosómica recesiva ligada al X

Anexo 2. Diagramas de secuencia.

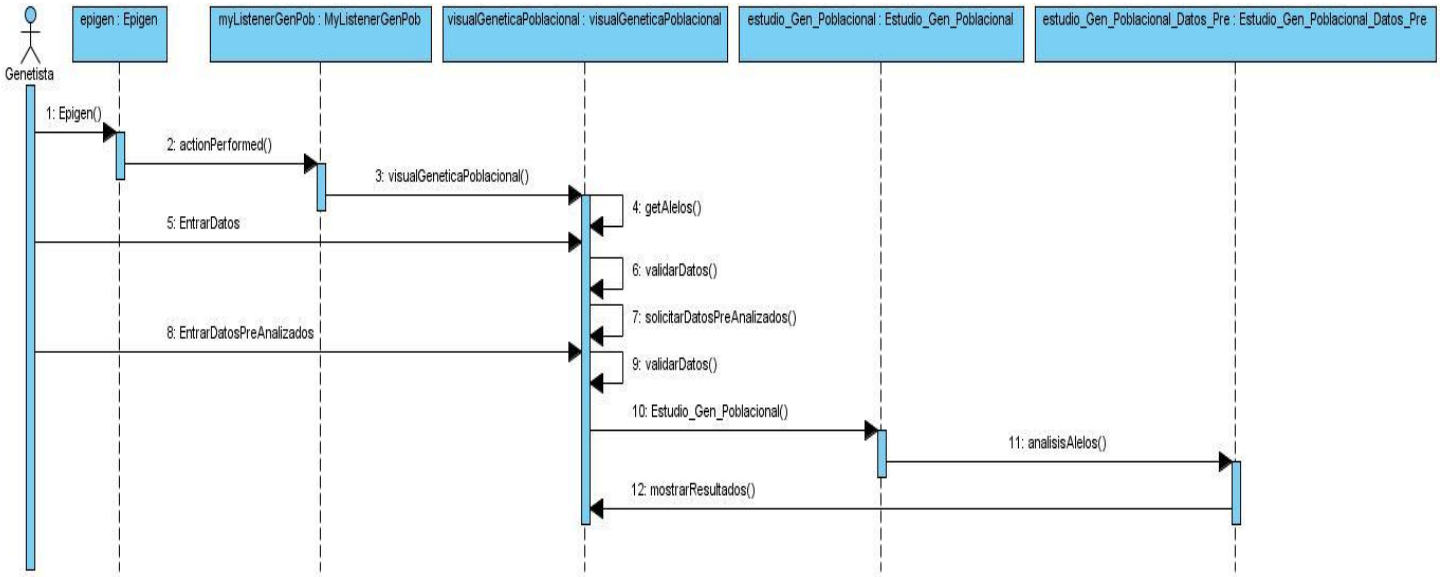

Figura 32 DS CU 3 Crear análisis de tres alelos con relación de codominancia a partir de datos pre analizados.

Anexo 3. Diagramas de componentes.

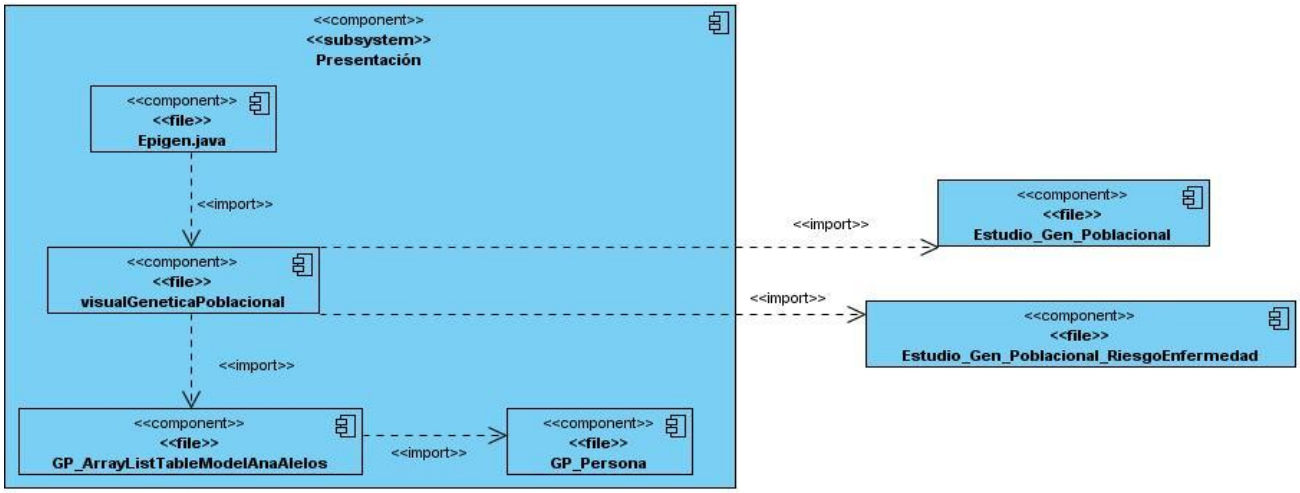

Figura 33 DS CU 1 Crear estudio de Genética Poblacional.

GLOSARIO

1. **Afijación proporcional:** Pretende que cada estrato tenga la misma proporción en la muestra que en la población. Por ejemplo, si en la muestra E1 la proporción de mujeres es de un 40%, en la población también tiene que ser la misma proporción.
2. **Alelo:** Es cada una de las formas alternativas que puede tener un gen que se diferencian en su secuencia y que se puede manifestar en modificaciones concretas de la función de ese gen.
3. **Alelo dominante:** Es aquel que enmascara la presencia del otro alelo diferente para el mismo carácter.
4. **Alelo recesivo:** Es aquel que solo se manifiesta cuando el individuo es raza pura para un carácter.
5. **API:** Interfaz de programación de aplicaciones.
6. **CASE:** Ingeniería de Software Asistida por Computadoras.
7. **Casos:** Son las personas de una población que están enfermos.
8. **Codominancia:** Es el proceso por el cual un individuo manifiesta dos características genéticas.
9. **Controles:** Son las personas de una población que no están enfermos.
10. **CNGM:** Centro Nacional de Genética Médica.
11. **CVS:** Sistema de Control de Versiones.
12. **Dominancia completa:** Es el proceso por el cual un individuo manifiesta una característica genética.
13. **Fenotipos:** Es la expresión del genotipo en un determinado ambiente.
14. **Frecuencia fenotípica:** Es la proporción o porcentaje de individuos de cada fenotipo que están presentes en la población.
15. **Frecuencia genotípica:** Es la proporción o porcentaje de individuos de cada genotipo que están presentes en la población.
16. **Gen:** Es un segmento corto de ADN, que le dice al cuerpo cómo producir una proteína específica. Hay aproximadamente 30.000 genes en cada célula del cuerpo humano y la combinación de todos los genes constituye el material hereditario para el cuerpo humano y sus funciones.
17. **Genotipos:** Información hereditaria de un organismo transmitida por la reproducción sexual de ambos progenitores.
18. **GUI:** Interfaz Gráfica de Usuario.
19. **HTTP:** Protocolo de Transferencia de Hipertexto.
20. **IDE:** Entono de Desarrollo Integrado.

21. **Junit:** Es un conjunto de bibliotecas creadas por Erich Gamma y Kent Beck que son utilizadas en programación para hacer pruebas unitarias de aplicaciones Java.
22. **MVC:** Modelo-Vista-Control.
23. **MVJ:** Máquina Virtual de Java.
24. **ODBC:** Microsoft Open Data Base Connectivity.
25. **RUP:** Proceso Unificado del Rational.
26. **SNS:** Sistema Nacional de Salud.
27. **SQL:** Lenguaje Estructurado de Consulta.
28. **SVN:** Subversion.
29. **TICs:** Tecnologías de la informática y las comunicaciones.
30. **UML:** Lenguaje Unificado de Modelado.
31. **Variabilidad genética:** Es una medida de la tendencia de los genotipos de una población a diferenciarse.
32. **XML:** Lenguaje de Marcado Extensible.