Universidad de las Ciencias Informáticas

Facultad 7

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

Título: Implementación de los procesos asociados a la evaluación dietética y la configuración del módulo Nutrición del Sistema de Evaluación del Neurodesarrollo en Niños

Autoras: Aismé Martínez Abreu
Ailín Teresa Naranjo Cruz
Tutores: Ing. Nestor Llanes Guerra
Ing. Javier Villares Arias
Asesores: Dr. Pedro Mestre Villavicencio
Msc. Manuel Avelino Ricardo Hidalgo

La Habana, junio de 2011
"Año 53 de la Revolución"

Datos de contacto

Ing. Nestor Llanes Guerra. Graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en el 2007. Instructor. Durante su trabajo como profesor, ha impartido clases de máquinas computadora I y II, y teleinformática I y II, pertenecientes al Departamento Sistemas Digitales de la facultad 7. En la vinculación con la producción, pertenece al Departamento de Sistemas Especializados en Medicina (SEM) del Centro de Informática Médica (CESIM) y específicamente, trabaja en el proyecto Rehabilitación.

Correo electrónico: nllanes@uci.cu

Ing. Javier Villares Arias. Graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en el 2009. Instructor recién graduado en adiestramiento. Durante su trabajo como profesor, ha impartido cursos pertenecientes a práctica profesional en el Departamento Sistemas Especializados en Medicina. Actualmente imparte la asignatura NADSS. En la vinculación con la producción, pertenece al Departamento de Sistemas Especializados en Medicina (SEM) del Centro de Informática Médica (CESIM) y específicamente, trabaja en el desarrollo de los proyectos Estomatología y Prótesis donde se desempeña como desarrollador.

Correo electrónico: jvillares@uci.cu

Resumen

En el departamento Sistemas Especializados en Salud, del centro de informática médica CESIM de la Universidad de las Ciencias Informáticas, se lleva a cabo el Sistema de Evaluación del Neurodesarrollo en Niños (SENDN). Uno de los módulos que conforman este sistema se dedica a la gestión de la información relacionada con la nutrición de los infantes, aspecto importante para el desarrollo de cada individuo puesto que desde edades tempranas es imprescindible alimentarse adecuadamente.

En este trabajo se brinda continuidad al flujo de implementación del módulo nutrición, en el que se realizan diferentes pruebas por parte de un grupo de especialistas, las cuales son muy extensas y realizadas de forma manual, requieren de mucho tiempo e impiden que un mismo día puedan atender a gran cantidad de niños, por lo que el objetivo es implementar los procesos: evaluación dietética y configuración del módulo, a partir del diseño propuesto por los analistas.

Para el desarrollo de la aplicación se emplean como herramientas: PostgreSQL como Sistema Gestor de Base de Datos, el Lenguaje de Programación Java; como Metodología de Desarrollo de Software: el Proceso Unificado de Desarrollo (RUP). Se hace uso del Lenguaje Unificado de Modelado (UML) y Visual Paradigm for UML 2, como herramienta CASE y el Framework JBoss Seam que utiliza el patrón Modelo-Vista-Controlador (MVC).

El desarrollo del módulo de Nutrición del Sistema de Evaluación del Neurodesarrollo en Niños, permitirá a los profesionales de esta especialidad, atender una mayor cantidad de pacientes en el día, con rapidez y calidad.

Palabras clave: Neurodesarrollo, Framework, Software.

Índice

Índice

INTRODUCCIÓN.

“La Informatización de la Sociedad es el proceso de utilización ordenada y masiva de las Tecnologías de la Información y las Comunicaciones (TIC) en la vida cotidiana, para satisfacer las necesidades de todas las esferas de la sociedad. Con este proceso, se logra cada vez más eficacia y eficiencia en todos los procesos y por consiguiente, un aumento en la calidad de vida de los ciudadanos”. [1]

“El Ministerio de Salud Pública (Minsap), ha definido a la informatización como una de sus prioridades, convocando a un grupo de instituciones del Ministerio de Informática y las Comunicaciones (MIC) y de otros Organismos de la Administración Central del Estado (OACE), para definir de conjunto la estrategia a desarrollar”. [2]

La Universidad de las Ciencias Informáticas (UCI), ha puesto en marcha una serie de proyectos productivos apoyando el desarrollo de software para el país y atendiendo a lo que de ella planteaba el Comandante en Jefe Fidel Castro Ruz, *“...Los objetivos de esta institución son altamente estratégicos. Al enemigo le va preocupar diez veces más que cuando era un centro de exploración radio-electrónica, porque este es un centro estratégico del futuro y para el desarrollo del país”.* [3]

En la UCI, específicamente en la Facultad 7 se desarrollan aplicaciones que van encaminadas a la informatización del Sistema Nacional de Salud (SNS), siendo una de ellas, el Sistema de Evaluación del Neurodesarrollo en Niños (SENDN). La misma está vinculado al Hospital Pediátrico Universitario “William Soler”, donde se lleva a cabo el programa “Renacer Contigo”. El mismo evalúa la calidad del neurodesarrollo de los niños de cero a cinco años de edad egresados de las Unidades de Cuidado Intensivo Polivalente y Neonatal (UCIP/UCIN), con el objetivo de lograr una calidad de vida óptima en los afectados. Como se puede apreciar, se realizan un conjunto de evaluaciones por parte de un equipo interdisciplinario compuesto por especialistas en: neurología, psicología, fisiatría, genética, logopedia y nutrición.

Por su parte, los especialistas en nutrición realizan evaluaciones clínicas, mediante exámenes físicos que les permiten conocer el estado del paciente en ese momento, además de evaluaciones bioquímicas mediante la realización de exámenes complementarios, para determinar las condiciones de su organismo.

También realizan evaluaciones antropométricas, mediante mediciones técnicas que expresan cuantitativamente las dimensiones del cuerpo humano y evaluaciones dietéticas, que permiten proporcionar las adecuaciones necesarias al balance energético. Todas estas evaluaciones, permiten determinar el estado nutricional de los pacientes, siendo este, un parámetro imprescindible para su valoración integral, constituyendo además, un indicador de la calidad de vida. Luego de realizadas las evaluaciones, se elabora un resumen, que se archiva en el expediente del paciente. Este resumen refleja los resultados encontrados y la impresión diagnóstica de los médicos. Estos son discutidos entre todos los especialistas para determinar las afectaciones en el neurodesarrollo del paciente y proponer un tratamiento.

Para poder realizar las evaluaciones se hace necesario, consultar tablas matemáticas complejas y de gran tamaño que pueden propiciar errores al registrar la información, además, se dificulta la comparación de diagnósticos emitidos en diferentes consultas, debido a que en el expediente del paciente se van archivando los resultados de cada una, lo cual, a medida que van aumentando las consultas, constituye un gran cúmulo de información. Esto provoca un lento y engorroso procesamiento de la información, además de la posibilidad de introducir errores al registrar la misma, los cuales pueden afectar el diagnóstico y tratamiento del paciente.

En el curso 2009-2010, se comenzó a trabajar en el desarrollo del módulo de nutrición para el Sistema de Evaluación del Neurodesarrollo en Niños, utilizando la metodología de Proceso Unificado de Desarrollo (RUP, por sus siglas en inglés), obteniéndose como resultado del modelamiento del negocio los procesos: realizar evaluación clínica, evaluación bioquímica, evaluación antropométrica y evaluación dietética, así como, la obtención de los resultados de la consulta de nutrición y la confección del resumen de la misma. Luego se realizó el diseño de cada uno de estos procesos, pero solo se implementaron las evaluaciones clínicas, bioquímicas y antropométricas. La solución propuesta no resuelve las dificultades anteriormente mencionadas, ya que constituye una solución parcial de los procesos.

Por lo antes planteado, se identifica como ***problema a resolver*****:** ¿Cómo hacer funcional el diseño de los procesos: realizar evaluación dietética y configurar el módulo Nutrición del Sistema de Evaluación del Neurodesarrollo en Niños, a partir del diseño propuesto por el analista?

Introducción

El **objeto de estudio** se enmarca en el proceso de desarrollo del sistema de Evaluación del Neurodesarrollo en Niños, delimitándose como **campo de acción** los procesos de implementación y prueba del Módulo Nutrición del Sistema de Evaluación del Neurodesarrollo en Niños.

Para la solución del problema, se plantea como ***objetivo general:*** implementar los procesos: evaluación dietética y configuración del módulo Nutrición, pertenecientes al Sistema de Evaluación del Neurodesarrollo en Niños, a partir del diseño propuesto por el analista.

Para dar cumplimiento al objetivo general planteado, se proponen las siguientes ***tareas de investigación***:

1- Realizar un estudio del Proceso de Desarrollo de Software que propone la Metodología Proceso Unificado de Desarrollo (RUP, por sus siglas en inglés).

2- Describir la arquitectura de software propuesta para el desarrollo del Sistema de Evaluación del Neurodesarrollo en Niños.

3- Realizar una valoración crítica del diseño propuesto por el analista para el desarrollo del sistema.

4- Generar los artefactos correspondientes a los Flujos de Trabajo: “Implementación” y “Prueba”.

5- Implementar los procesos de evaluación dietética; realizar resumen de consulta y configuración del módulo Nutrición, pertenecientes al Sistema de Evaluación del Neurodesarrollo en Niños, a partir del diseño propuesto por el analista.

El desarrollo del módulo de Nutrición del Sistema de Evaluación del Neurodesarrollo en Niños proporcionará un grupo de beneficios. Entre ellos se encuentran:

- Mayor rapidez en el diagnóstico médico, traducida en mejor calidad de vida y menos discapacidades que mejoren el entorno familiar.
- Mayor seguridad y confiabilidad de la información médica.
- Disponer de un sistema que tenga como centro de referencia al paciente, dando respuesta a sus necesidades de salud reales y de asistencia médica.
- Facilitar una implementación más rápida del programa médico de atención temprana y evaluación del neurodesarrollo en todos los hospitales pediátricos de Cuba, con la consiguiente mejora en los procesos a los que se refiere.

- El país contará con un sistema informático gratuito y de libre distribución.
- Se podrá atender mayor cantidad de pacientes diariamente.

El documento presenta una estructura por capítulos como se muestra a continuación:

Capítulo 1. Descripción del proceso unificado de desarrollo.

En este capítulo, se enunciarán los principales conceptos que se han tenido en cuenta en la presente investigación; se describirá el proceso unificado de desarrollo de software; se brindará una descripción del sistema de Evaluación del Neurodesarrollo en Niños y del Módulo Nutrición contenido en dicho sistema. Además, se describirán los flujos de trabajo implementación y prueba, las tecnologías, metodología de desarrollo, plataformas, librerías, framework y herramientas propuestas por el Departamento de Gestión Hospitalaria del Centro de Informática Médica para darle solución al problema planteado.

Capítulo 2. Descripción de la arquitectura de software del Sistema de Evaluación del Neurodesarrollo en Niños.

La especificación de los requerimientos no funcionales que apoyarán el funcionamiento del software es uno de los temas abordados en este capítulo, debido a la importancia que tiene para toda aplicación ya que es ahí, donde se define lo que el sistema debe cumplir. Además, se describirá la arquitectura de software del Sistema de Evaluación del Neurodesarrollo en Niños y se determinarán las estrategias de integración. Asimismo, se describirá cómo se garantiza la seguridad de la aplicación y la distribución física del sistema.

Capítulo 3. Descripción y análisis de la solución propuesta.

En el capítulo se realizará una valoración crítica del diseño propuesto por el analista, determinándose las nuevas clases y operaciones necesarias para llevar a cabo la implementación del sistema. Para describir datos, sus relaciones, su significado y sus restricciones de consistencia se plasmará el modelo de datos. El diagrama de componentes muestra las dependencias lógicas entre componentes software, sean éstos componentes fuentes, binarios, o ejecutables.

Capítulo 4. Validación de la solución propuesta.

La validación de la solución propuesta se realizará mediante el diseño de las pruebas de unidades que son las encargadas de probar el sistema. Se describirán los valores utilizados para las pruebas mediante los casos de pruebas, así como la evaluación de la ejecución las pruebas y de los resultados obtenidos.

CAPÍTULO 1: DESCRIPCIÓN DEL PROCESO DE DESARROLLO DE SOFTWARE DEL MÓDULO DE NUTRICIÓN.

En este capítulo se establecen los objetivos siguientes, enunciar los principales conceptos relacionados con la investigación, describir el sistema de Evaluación del Neurodesarrollo en Niños y del Módulo de Nutrición contenido en el mismo, así como, el proceso unificado de desarrollo de software y analizar los flujos de trabajo relacionados con la investigación. Además, caracterizar las tecnologías, plataformas, librerías, framework y herramientas propuestas a utilizar en el desarrollo del sistema.

Marco conceptual.

Nutrición: Ciencia o disciplina que estudia las reacciones del organismo a la ingestión de los alimentos y nutrientes. [4]

Neurodesarrollo: Adquisición de funciones, dependientes del Sistema Nervioso, que implican un incremento de estructuras orgánicas y funcionales a través de un proceso de maduración. [5]

Atención Temprana: Es el conjunto de acciones que tienen como objetivo dar tratamiento lo más pronto posible a niños de cero a seis años con alteraciones del desarrollo de manera transitoria o permanente, a la familia y al entorno, planificadas por un equipo de profesionales de orientación interdisciplinar y transdisciplinar.[6]

Estado Nutricional**:** Se determina mediante la valoración del crecimiento en los niños y los cambios en la masa corporal de los adultos, refleja diversos grados de bienestar, que en sí mismos, son consecuencia de una compleja interacción entre la dieta, factores relacionados con la salud y el entorno físico, social y económico. [7]

Evaluación dietética: “Valoración de las características de la alimentación pasada y actual, factores determinantes y adecuación sobre la base de requerimientos calculados individualmente.”[8]

Metodología: Es el conjunto de métodos por los cuales se regirá una investigación científica. [9]

Capítulo 1

1.1 Descripción del Sistema de Evaluación del Neurodesarrollo en Niños.

“Las Unidades de Cuidados Intensivos Pediátricos (UCIP), surgieron para dar respuesta asistencial eficiente a las urgencias pediátricas, es el servicio del hospital dedicado a la asistencia intensiva integral y continuada al niño críticamente enfermo, independientemente del origen de esta.

En la práctica diaria se encuentra, que el niño sometido a una enfermedad grave es propenso a que se comprometa su calidad de vida, las afecciones del neurodesarrollo son las más frecuentemente halladas, ya que el sistema nervioso central es muy susceptible en los niños en formación. Actualmente ya no basta con brindar una atención especializada al niño grave o con alteraciones del desarrollo, sino lograr una calidad de vida óptima a los afectados, lo que reanudará una población sana en un futuro.” [10]

La pobre presencia de dirigir los esfuerzos no solo al paciente grave, sino a la familia en los servicios de atención al paciente en el país y la necesidad cada vez mayor de mejorar la calidad de vida de nuestra población infantil, ya con tasas de mortalidad comparables con los países desarrollados, motivó a un grupo de especialistas del hospital Pediátrico Universitario “William Soler”, a crear el proyecto “Renacer Contigo”, con el objetivo de evaluar la calidad del neurodesarrollo de los niños de 0-5 años de edad egresados de las Unidades de Terapia Intensiva Polivalente y Neonatal (UCIP/UCIN).

En dicho proyecto trabaja un equipo interdisciplinario compuesto por especialistas en neurología, psicología, fisiatría, neurofisiología, logopedia y nutrición. En el curso 2008-2009 se decidió, comenzar a desarrollar en la Universidad de las Ciencias Informáticas (UCI), el Sistema de Evaluación del Neurodesarrollo en Niños (SENDN), el mismo cuenta con los módulos: Expediente y Turnera, Fisiatría, Genética, Logopedia, Neurología, Nutrición, Psicología, y Evaluación integral y Tratamiento.

El módulo Expediente y Turnera constituye la entrada del paciente al programa de evaluación temprana, en el cual se recogen los datos iniciales de este y se planifican los turnos de las evaluaciones de cada una de las especialidades. En los módulos: Fisiatría, Genética, Logopedia, Neurología, Nutrición y Psicología, se realizan las evaluaciones correspondientes a estas especialidades y se obtienen los resúmenes de cada una de ellas para cada consulta. Para integrar cada uno de estos resultados, se utiliza el módulo

Evaluación integral y tratamiento, en el cual se determinará la evaluación final del paciente y se le indicará un tratamiento.

1.2 Descripción del módulo de Nutrición del Sistema de Evaluación del Neurodesarrollo en Niños.

Como parte del SENDN se encuentra el módulo de *Nutrición*, el cual es muy importante debido a que la nutrición, es vital para el desarrollo de cada individuo, ya que desde edades tempranas, es imprescindible alimentarse correctamente, para contribuir directamente a disminuir el índice de mortalidad en el primer año de vida y garantizar su buen desarrollo durante la infancia. Una alimentación adecuada ayuda a aumentar el nivel de aprendizaje de los niños y evitar futuras complicaciones y enfermedades.

En este módulo los especialistas en Nutrición realizan una serie de evaluaciones, que les permiten determinar el estado nutricional de los pacientes, dicho estado se puede explorar mediante evaluaciones clínicas, que no es más que realizar exámenes físicos que les permiten conocer el estado del paciente en ese momento, además de evaluaciones bioquímicas mediante la realización de exámenes complementarios, para determinar las condiciones de su organismo. También realizan evaluaciones antropométricas, mediante mediciones técnicas que expresan cuantitativamente las dimensiones del cuerpo humano y evaluaciones dietéticas, que permiten proporcionar las adecuaciones necesarias al balance energético.

Todas estas evaluaciones, permiten determinar el estado nutricional de los pacientes, siendo este, un parámetro imprescindible para su valoración integral, constituyendo además, un indicador de la calidad de vida. Luego de realizadas las evaluaciones, se elabora un resumen que se archiva en el expediente del paciente. Este resumen, refleja los resultados encontrados y la impresión diagnóstica de los médicos. Estos son discutidos entre todos los especialistas para determinar las afectaciones en el neurodesarrollo del paciente y proponer un tratamiento.

1.3 Metodología de desarrollo de software a emplear.

La metodología de desarrollo de software a emplear es el Proceso Unificado de Desarrollo de Software (RUP por sus siglas en inglés), definida por el departamento de Gestión Hospitalaria para el desarrollo de

sus aplicaciones, siendo el Sistema de Evaluación del Neurodesarrollo en Niños, al que pertenece el Módulo *Nutrición*, una de ellas.

El Proceso Unificado de Desarrollo de Software, es un proceso genérico que puede ser utilizado para varios tipos de sistemas de software, para diferentes áreas de aplicación, tipos de organizaciones, niveles de competencia y tamaños de proyectos. Provee un enfoque disciplinar en la asignación de tareas y responsabilidades dentro de una organización de desarrollo. Su meta es asegurar la producción de software de alta calidad, que satisfaga las necesidades de los usuarios finales, dentro de un calendario y presupuesto predecible.

Entre sus aspectos clave se encuentran:

- Guiado por casos de uso: Los casos de uso, no son solo una herramienta para especificar los requisitos del sistema, sino que constituyen un elemento integrador y una guía del trabajo, de su diseño, implementación y prueba. Además, estos no solo inician el proceso de desarrollo sino que proporcionan un hilo conductor, permitiendo establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo.

- Centrado en la arquitectura: La arquitectura de un sistema es la organización o estructura de sus partes más relevantes, lo que permite tener una visión común entre todos los involucrados (desarrolladores y usuarios) y una perspectiva clara del sistema completo, necesaria para controlar el desarrollo.

- Iterativo e incremental: Propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Es práctico dividir el trabajo en partes más pequeñas, o mini proyectos. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto.

Esta metodología identifica nueve flujos de trabajo, que van desarrollándose durante las fases de inicio, elaboración, construcción y transición; seis de ingeniería: modelado de negocio, requerimientos, diseño e implementación, prueba, despliegue y tres de apoyo: configuración y cambios, entorno y gestión de proyecto, lo cual se puede apreciar en la Figura 1.

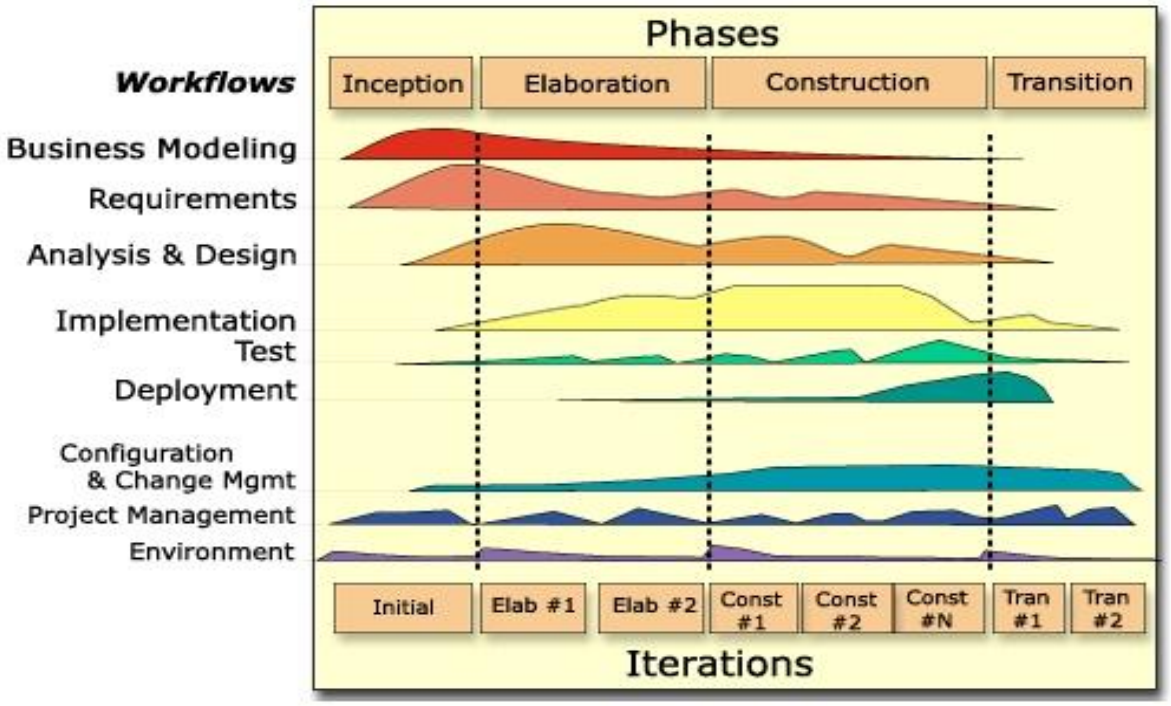


Figura 1 Fases del Proceso Unificado de Desarrollo de Software (RUP).

Las características generales de las fases de RUP son:

Inicio: Se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos. Se define el alcance del proyecto.

Elaboración: Se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos.

Construcción: Se concentra en la elaboración de un producto totalmente operativo y eficiente, se prueba todo en profundidad.

Transición: Se instala el producto al cliente y se entrena a los usuarios. Como consecuencia de esto, suelen surgir nuevos requisitos a ser analizados. [11]

"Las características generales de los flujos de trabajo de RUP son:

- Modelamiento del negocio: describe los procesos de negocio, identificando quienes participan y las actividades que requieren automatización.
- Requisitos: 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 de los componentes en ellos y la estructura de capas de la aplicación.
- Prueba (test): busca los defectos a lo largo del ciclo de vida del software.
- Despliegue (instalación): produce el despliegue del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.), para entregar el software a los usuarios finales.
- Gestión de cambios y configuraciones (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.
- Gestión del proyecto (administración del proyecto): involucra actividades con las que se busca elaborar un producto que satisfaga las necesidades de los clientes.
- Entorno (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." [12]

Para el desarrollo del Módulo Nutrición en el curso 2009-2010, se llevaron a cabo los flujos de trabajo: Modelamiento del negocio, Requisitos, Análisis y Diseño y una primera iteración del flujo de Implementación.

1.4 Análisis de los flujos de trabajo del Proceso Unificado de Desarrollo de Software realizados por el analista.

En el curso 2009-2010, se comenzó a trabajar en el desarrollo del Módulo de Nutrición del Sistema de Evaluación del Neurodesarrollo en Niños, llevándose a cabo por el analista los flujos: Modelamiento del Negocio, Requisitos y Análisis y Diseño.

En el Modelamiento del Negocio, se analizaron los procesos que caracterizan el proceder de los especialistas en nutrición del proyecto Renacer Contigo, del Hospital Pediátrico Universitario William Soler, obteniendo como resultado el modelado y la descripción de los mismos.

Los procesos fundamentales obtenidos fueron:

- Realizar Evaluación Clínica.
- Realizar Evaluación Bioquímica.
- Realizar Evaluación Antropométrica.
- Realizar Evaluación Dietética.
- Realizar Resumen de Nutrición.
- Obtener resultados de consulta.

Luego de modelar los procesos, se realizó un levantamiento de requerimientos, donde se obtuvieron los requisitos funcionales (RF) que debía cumplir el sistema, según las necesidades del cliente y se definieron los casos de uso del sistema.

Los requisitos funcionales obtenidos fueron:

- RF1 Realizar Evaluación Clínica.
- RF2 Visualizar detalles Evaluación Clínica.
- RF3 Modificar Evaluación Clínica.
- RF4 Listar pacientes a realizar Evaluación Bioquímica.

- RF5 Buscar pacientes a realizar Evaluación Bioquímica.
- RF6 Realizar Evaluación Bioquímica.
- RF7 Visualizar detalles Evaluación Bioquímica.
- RF8 Modificar Evaluación Bioquímica.
- RF9 Listar pacientes a atender en Antropométrica.
- RF10 Buscar pacientes a atender en Antropométrica.
- RF11 Realizar Evaluación Antropométrica.
- RF12 Visualizar detalles Evaluación Antropométrica.
- RF13 Modificar Evaluación Antropométrica.
- RF14 Listar Pacientes a realizar Evaluación Dietética.
- RF15 Buscar Pacientes a realizar Evaluación Dietética.
- RF16 Realizar Evaluación Dietética.
- RF17 Visualizar detalles Evaluación Dietética.
- RF18 Modificar Evaluación Dietética.
- RF19 Listar Pacientes Atendidos Nutrición.
- RF20 Buscar Pacientes Atendidos Nutrición.
- RF21 Realizar Resumen Nutrición.
- RF22 Visualizar detalles Resumen Nutrición.
- RF23 Modificar Resumen Nutrición.
- RF24 Buscar Resumen Nutrición.
- RF25 Visualizar Resumen Nutrición.
- RF26 Buscar Resultado Consulta Nutrición.
- RF27 Visualizar Resultado Consulta Nutrición.

- RF28 Gestionar Longitud supina peso talla femenina.
- RF29 Gestionar Peso talla femenino.
- RF30 Gestionar Longitud supina peso talla masculino.
- RF31 Gestionar Peso talla masculina.
- RF32 Gestionar Estatura peso estatura femenino.
- RF33 Gestionar Peso estatura femenino.
- RF34 Gestionar Estatura peso estatura masculino.
- RF35 Gestionar Peso estatura masculino.
- RF36 Gestionar Meses.
- RF37 Gestionar Edad decimal.
- RF40 Gestionar Valor nutritivo.
- RF41 Gestionar Edades.
- RF42 Gestionar Área grasa femenina.
- RF43 Gestionar Área grasa masculina.
- RF44 Gestionar Área músculo femenina.
- RF45 Gestionar Área músculo masculino.
- RF46 Gestionar Circunferencia brazo femenino.
- RF47 Gestionar Circunferencia brazo masculino.
- RF48 Gestionar Circunferencia cefálica femenina.
- RF49 Gestionar Circunferencia cefálica masculino.
- RF50 Gestionar Pliegue tricipital femenino.
- RF51 Gestionar Pliegue tricipital masculino.
- RF52 Gestionar Peso edad femenino.

- RF53 Gestionar Peso edad masculino.
- RF54 Gestionar Talla edad femenino.
- RF55 Gestionar Talla edad masculino.

Luego, se dio paso al flujo análisis y diseño, donde se diseñaron los casos de uso correspondientes a los procesos realizar evaluación clínica, realizar evaluación bioquímica, realizar evaluación antropométrica y realizar resumen; faltando el diseño de los casos de uso del proceso realizar evaluación dietética y del paquete de configuración, los cuales se muestran a continuación:

- CU Crear Evaluación Dietética.
- CU Ver Detalles Evaluación Dietética.
- CU Modificar Evaluación Dietética.
- CU Buscar Resultado Consulta Nutrición.
- CU Ver Resultado Consulta Nutrición.
- CU Gestionar Longitud supina peso talla femenina.
- CU Gestionar Peso talla femenino.
- CU Gestionar Longitud supina peso talla masculino.
- CU Gestionar Peso talla masculina.
- CU Gestionar Estatura peso estatura femenino.
- CU Gestionar Peso estatura femenino.
- CU Gestionar Estatura peso estatura masculino.
- CU Gestionar Peso estatura masculino.
- CU Gestionar Meses.
- CU Gestionar Edad decimal.
- CU Gestionar Valor nutritivo.
- CU Gestionar Edades.

- CU Gestionar Área grasa femenina.
- CU Gestionar Área grasa masculina.
- CU Gestionar Área músculo femenina.
- CU Gestionar Área músculo masculino.
- CU Gestionar Circunferencia brazo femenino.
- CU Gestionar Circunferencia brazo masculino.
- CU Gestionar Circunferencia cefálica femenina.
- CU Gestionar Circunferencia cefálica masculino.
- CU Gestionar Pliegue tricipital femenino.
- CU Gestionar Pliegue tricipital masculino.
- CU Gestionar Peso edad femenino.
- CU Gestionar Peso edad masculino.
- CU Gestionar Talla edad femenino.
- CU Gestionar Talla edad masculino.

1.5 Flujos de trabajos a desarrollar por el implementador.

Los flujos de trabajos de implementación y prueba, son llevados a cabo por el implementador.

Flujo de trabajo de implementación.

El flujo de trabajo de implementación, es más significativo en la fase de construcción del sistema. Se comienza luego del diseño realizado por el analista y se implementa el producto de software, en correspondencia con ese diseño.

En este flujo se describe, cómo los elementos del Modelo de Diseño se implementan en términos de componentes y cómo estos se organizan, de acuerdo a los nodos específicos en el Modelo de Despliegue.

Los objetivos del flujo de Implementación son:

- Definir la organización del código, en términos de los subsistemas de implementación organizados en capas.
- Implementar los elementos de diseño en términos de los elementos de implementación (archivos de origen, binarios, programas ejecutables y otros).
- Probar y desarrollar componentes como unidades.
- Integrar los resultados producidos por los implementadores individuales (o equipos), en un sistema ejecutable.
- Distribuir el sistema, asignando componentes ejecutables a nodos en el Diagrama de Despliegue.

Las principales actividades del flujo de Implementación son:

- Estructurar el Modelo de Implementación: En esta actividad se establece la estructura de los elementos de implementación, basándose en las responsabilidades asignadas a los subsistemas de implementación y su contenido.
- Planificar la integración: Consta de planificar la integración de cada subsistema y del sistema en su conjunto. Consiste en definir el orden en el que se integrarán los elementos contenidos en un subsistema de implementación.
- Implementar componentes: Mediante esta actividad, se completa una parte de la implementación de forma que se puede entregar para la integración.
- Integrar los subsistemas: Consiste en integrar los elementos en un subsistema de implementación y, a continuación, entregar el subsistema de implementación para su integración en el sistema.
- Integrar el sistema: Consiste en integrar las partes de los subsistemas de implementación. [13]

En el curso 2009-2010, se llevó a cabo una primera iteración de este flujo, obteniéndose como resultado de la misma, la implementación de los casos de uso asociados a los procesos: realizar evaluación clínica, realizar evaluación bioquímica, realizar evaluación antropométrica y realizar resumen de nutrición, por lo que en el presente curso (2010-2011) se llevará a cabo una segunda iteración de este flujo, en la cual se implementarán los casos de uso asociados a los procesos: realizar evaluación dietética, obtener resultado de consulta y configurar del sistema.

Flujo de trabajo Prueba.

En el flujo de trabajo Prueba, se verifica el resultado de la implementación probando el sistema entregable.

“El objetivo de la prueba de software es descubrir errores. Para conseguir este objetivo se planifica y se ejecuta una serie de pasos que van revisando todos los elementos del software. En todas las fases del desarrollo del proyecto hay que probar el software que se va construyendo, aunque como el grueso de la programación se realiza en la construcción, es en esa fase en la que se centran los mayores esfuerzos de este flujo. La etapa de prueba es importante ya que en ella se refleja la calidad con que ha sido llevada a cabo la proyección del sistema.” [14]

“Durante la fase de inicio, se puede hacer parte de la planificación inicial de las pruebas cuando se define el ámbito del sistema. Sin embargo, las pruebas se llevan a cabo sobre todo cuando una construcción como resultado de implementación, es sometida a pruebas de integración y de sistema. Esto quiere decir, que la realización de pruebas se centra en las fases de elaboración, cuando se prueba la línea base ejecutable de la arquitectura, y de construcción, cuando el grueso del sistema esta implementado.” [15]

“RUP propone que en cada una de las fases, las pruebas se comporten de la siguiente forma:

- Inicio: el desarrollo del prototipo exploratorio de demostración; no requiere la elaboración de pruebas.
- Elaboración: probar los componentes ejecutables que se han implementado y que deben corresponderse con la arquitectura básica de la aplicación.
- Construcción: desarrollar los casos de prueba y procedimientos de prueba para hacerlos.
- Transición: el producto en su entorno de operación, por lo que es probado por usuarios reales.” [16]

Trabajadores involucrados en el flujo de trabajo de prueba:

- Administrador de Prueba: es el responsable del éxito de la prueba, este rol involucra defensor de prueba y calidad, planificación y administración de recursos, así como, la resolución de problemas que impiden las pruebas.

- Analista de Prueba: es el responsable de identificar y definir las pruebas requeridas, monitorear el progreso de la prueba y el resultado en cada ciclo de prueba evaluando la calidad total experimentada, como un resultado de las actividades de prueba.

- Diseñador de prueba: es el responsable de definir el método de prueba y asegurar su implementación exitosa. El rol incluye la identificación de técnicas apropiadas, herramientas e instrucciones, para implementar las pruebas necesarias y encauzar los recursos correspondientes para estas.

- Probador: es el responsable durante las actividades principales de las pruebas, el cual incluye la conducción de las pruebas necesarias y el registro del resultado de la prueba.[17]

En el presente curso, se llevará a cabo una primera iteración de este flujo, mediante la realización de pruebas unitarias.

1.6 Herramientas y tecnologías a emplear.

1.6.1 Lenguajes.

Lenguaje de Modelado Unificado UML.

“El Lenguaje Unificado de Modelado permite visualizar, especificar, construir y documentar, los artefactos de un sistema que involucra una gran cantidad de software. Brinda la posibilidad de modelar sistemas con tecnología orientada a objetos. Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software, pero no especifica en sí mismo, qué metodología, o proceso utilizar.

Este lenguaje de modelado formal, permite tener un mayor rigor en la especificación, realizar una verificación y validación del modelo desarrollado, automatizar determinados procesos y generar código a partir de los modelos y a la inversa. Esto último, permite que el modelo y el código estén actualizados.” [18]

Capítulo 1

Lenguaje de programación Java.

“Es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios del siglo pasado. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.” [19]

Características:

Las características principales que ofrece Java, respecto a cualquier otro lenguaje de programación son:

Simple: Ofrece toda la funcionalidad de un lenguaje potente, se diseñó para ser parecido a C++ y así, facilitar un rápido y fácil aprendizaje. Mediante el reciclador se permite liberar bloques de (Java) de los lenguajes C y C++ como es el caso de: aritméticas de punteros, registros, definiciones de tipo, macro entre otras.

Orientado a objetos: Trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo.

Distribuido: Construido con extensas capacidades de interconexión TCP/IP, proporciona bibliotecas y herramientas para que los programas puedan ser distribuidos, es decir, que se corran en varias máquinas, interactuando.

Robusto: Realiza verificaciones en busca de problemas, tanto en tiempo de compilación, como en tiempo de ejecución: obliga a la declaración explícita de métodos, reduciendo así las posibilidades de error, maneja la memoria para eliminar las preocupaciones por parte del programador de la liberación o corrupción de memoria, proporciona comprobación de punteros, de límites de array, excepciones, verificación byte-code, entre otros.

Arquitectura neutral: Para establecer Java como parte integral de la red, el compilador Java compila su código a un fichero objeto de formato independiente de la arquitectura de la máquina en que se ejecutará.

Cualquier máquina que tenga el sistema de ejecución run-time, puede ejecutar ese código objeto, sin importar la máquina en que ha sido generado.

Multithreaded: Consiste en un mejor rendimiento interactivo y mejor comportamiento en tiempo real, aunque el comportamiento en tiempo real está limitado a las capacidades del sistema operativo subyacente, aún supera a los entornos de flujo único de programa tanto en facilidad de desarrollo, como en rendimiento.

1.6.2 Tecnologías.

Java Enterprise Edition.

"La tecnología denominada convencionalmente como Java Plataform Enterprise Edition (Java EE, por su sigla en inglés), es el estándar de la industria de desarrollo de software que permite trabajar aplicaciones empresariales seguras, portables, robustas y escalables, basadas en servidores. Java EE provee API's que permiten administrar, entre otros, web services, modelo de componentes, comunicación entre objetos y que hacen posible la implementación de la arquitectura empresarial orientada a servicios (SOA) y aplicaciones web 2.0." [20]

Java Runtime Environment.

Java Runtime Environment (JRE, por su sigla en inglés), se corresponde con un conjunto de utilidades que permite la ejecución de programas Java, sobre todas las plataformas soportadas. La Máquina Virtual Java (JVM, por su sigla en inglés), es una instancia de JRE en tiempo de ejecución, este es el programa que interpreta el código Java y además, por las librerías de clases estándar que implementan las API de Java. Ambas JVM y API, deben ser consistentes entre sí, de ahí que sean distribuidas de modo conjunto.

Un usuario solo necesita el JRE para ejecutar las aplicaciones desarrolladas en lenguaje Java, mientras que para desarrollar nuevas aplicaciones en dicho lenguaje, es necesario un entorno de desarrollo, denominado JDK, que además del JRE incluye, entre otros, un compilador para Java.

Java Persistence API.

Java Persistence API, más conocida por su sigla JPA, es la API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar EJB3. Esta API, busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue el diseño de esta API, es

no perder las ventajas de la orientación a objetos al interactuar con una base de datos, como sí pasaba con EJB2, y permitir usar objetos regulares (conocidos como POJOs).

Hibernate.

"Es una herramienta de Mapeo objeto-relacional para la plataforma Java, que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML), que permiten establecer estas relaciones. Además de ser un entorno de trabajo que tiene como objetivo facilitar la persistencia de objetos Java en bases de datos relacionales y al mismo tiempo, la consulta de estas bases de datos, para obtener objetos." [21]

1.6.3 Framework.

JBoss Seam.

Es un framework Open Source desarrollado por JBoss. Combina a los frameworks EJB3 (Enterprise JavaBeans) y JSF (Java Server Faces). Se puede acceder a cualquier componente EJB desde la capa de presentación refiriéndose a él mediante su nombre de componente Seam, ya que integra la capa de presentación (JSF) con la capa de negocios y persistencia (EJB). Tiene como objetivo simplificar la arquitectura de las aplicaciones, lo que permite integrar tecnologías de forma relativamente transparente y con herramientas de generación de código.

Java Server Faces.

Es un framework que facilita y agiliza el diseño de interfaces de usuario, pues implementa una serie de componentes, estado de los mismos, eventos del lado de servidor. Entre sus ventajas se encuentran:

- El código es muy parecido al HTML estándar. Lo pueden utilizar fácilmente desarrolladores y diseñadores web.
- Se integra dentro de la página JSP y se encarga de la recogida y generación de los valores de los elementos de la página.
- Resuelve validaciones, conversiones, mensajes de error, e internacionalización.

- Permite introducir javascript en la página, para acelerar la respuesta de la interfaz en el cliente (navegador del usuario).
- Es extensible, por lo que se pueden desarrollar nuevos componentes a medida, También se puede modificar el comportamiento del framework mediante APIs que controlan su funcionamiento. [22]

1.6.4 Librerías.

RichFaces 3.2.

RichFaces, es una librería de componentes visuales para JSF que posee un avanzado marco para integrar fácilmente capacidades AJAX, en el desarrollo de aplicaciones de negocios. Permite a los desarrolladores ahorrar tiempo y aprovechar las características de los componentes para crear aplicaciones Web, ricas en interfaz. Proporciona componentes fáciles de utilizar, con etiquetas predefinidas, y brinda capacidades AJAX (Ajax4jsf).

Ajax4jsf.

"Ajax4jsf, es una librería que se integra totalmente en la implementación de JSF usada, y extiende la funcionalidad de sus etiquetas dotándolas con tecnología Ajax de forma limpia y sin añadir código Javascript. Mediante este framework, se puede variar el ciclo de vida de una petición JSF, recargar determinados componentes de la página sin necesidad de recargar por completo, realizar peticiones automáticas al servidor, control de cualquier evento de usuario, etc. Estas características de la librería mencionada, dotan a la aplicación JSF de un contenido mucho más profesional con muy poco esfuerzo." [23]

1.6.5 Servidor de aplicaciones.

JBoss.

"Es un Servidor de Aplicaciones Java EE de Software Libre implementado en Java puro. Al estar basado en Java, JBoss puede ser utilizado en cualquier sistema operativo que lo soporte. Entre sus características más relevantes se encuentran que implementa todo el paquete de servicios de J2EE, es confiable a nivel de empresa, orientado a arquitectura de servicios y presenta flexibilidad consistente.

También proporciona servicios que necesitan la mayoría de las aplicaciones empresariales, tales como: la seguridad, transaccionalidad, persistencia, vigilancia, gestión de recursos y de acceso remoto.

JBoss, es el primer servidor de aplicaciones de código abierto, preparado para la producción y certificado J2EE 1.4, disponible en el mercado, ofreciendo una plataforma de alto rendimiento para aplicaciones de e-business. Combinando una arquitectura orientada a servicios con una licencia de código abierto, JBoss, puede ser descargado, utilizado, incrustado, y distribuido, sin restricciones por la licencia. Por este motivo, es la plataforma más popular de middleware para desarrolladores, vendedores independientes de software y, también, para grandes empresas." [24]

1.6.6 Servidor de base de datos.

PostgreSQL 8.3.

"Es una poderosa herramienta de código abierto para el manejo de bases de datos relacionales, publicada bajo licencia BSD. Puede ser utilizado en diversos sistemas operativos, incluyendo GNU/Linux, UNIX y Windows.

Características de PostgreSQL:

Implementación del estándar SQL92/SQL99:

- Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes, cadenas de bits. También permite la creación de tipos propios como la de los disparadores, e incorpora funciones de diversa índole.
- Incorpora una estructura de datos array.
- Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos, se le incluye entre los gestores objeto- relacionales.
- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.
- Ejecuta procedimientos almacenados en varios lenguajes de programación como Java, Perl, Python, Ruby, C, C + +, y PL / pgSQL y soporta casi toda la sintaxis SQL pues tiene soporte total para llaves extranjeras, joins, vistas.

- Usa una arquitectura proceso-por-usuario cliente/servidor, la cual es similar al método del Apache 1.3.x para manejar procesos. Existe un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.
- Mediante un sistema denominado Acceso concurrente multiversión MVCC por sus siglas en inglés, PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos." [25]

1.6.7 Administración de bases de datos.

PgAdmin.

Es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia Open Source. Está escrita en C++ usando la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda usar en Linux, FreeBSD, Solaris, Mac OS X y Windows. Es capaz de gestionar versiones a partir de la PostgreSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de PostgreSQL como Pervasive Postgres, EnterpriseDB, Mammoth Replicator y SRA PowerGres.

"Está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples, hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados, soporte para el motor de replicación Slony-I y mucho más. La conexión al servidor, puede hacerse mediante conexión TCP/IP o Unix Domain Sockets (en plataformas *nix), y puede encriptarse mediante SSL para mayor seguridad." [26]

1.6.8 Entorno de desarrollo integrado.

Eclipse.

Una herramienta Ambiente de Desarrollo Integrado (IDE por sus siglas en inglés), es un programa compuesto por un conjunto de herramientas para un programador, que se ejecuta a partir de una única interfaz de usuario. Los IDEs pueden ser aplicaciones por si solas, o pueden ser parte de aplicaciones existentes.

Eclipse GANYMADE: "Es un IDE de código abierto extensible. En la actualidad, funciona bien como un IDE Java, e incluye herramientas de desarrollo Java. Se requiere que tenga el entorno de Sun Java Runtime (JRE) instalada. Es multiplataforma y se utiliza para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano", basadas en navegadores. Presenta una arquitectura abierta y basada en plugins, lo que permite expandir las capacidades de la plataforma base, pudiendo ser añadidos automáticamente al entorno de desarrollo.
Eclipse presenta control del editor de código, de compilador y de un potente depurador, lo que permite establecer puntos de interrupción, modificar e inspeccionar valores de variables, e incluso, depurar código que resida en una máquina remota, desde una única interfaz de usuario. Su misión consiste en evitar tareas repetitivas, facilitar la escritura de código correcto, disminuir el tiempo de depuración e incrementar la productividad del desarrollador." [27]

1.6.9 Herramienta para el modelado.

Visual Paradigm 6.4.

Es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software. Ayuda a una más rápida construcción de aplicaciones de calidad y a menor costo. Permite construir todo tipo de diagramas de clases, generar código desde diagramas y generar documentación. Apoya los estándares más altos de las notaciones de Java y de UML. Soporta aplicaciones web y es fácil de instalar y actualizar. Está orientado a la creación de diseños y se usa el paradigma de programación orientada a objetos.

Características de Visual Paradigm:

- Brinda un diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.

- Presenta un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Permite tanto la ingeniería directa como inversa, soportada en varios lenguajes de programación.
- Permite la disponibilidad de múltiples versiones para cada necesidad, capaz de integrase con los principales IDEs.
- Disponibilidad en múltiples plataformas y soporta el análisis y diseño orientado a objetos.
- Generación de Bases de Datos basado en diagramas.

Características:

Es multiplataforma, permite realizar cualquier tipo de diagrama, entiéndase diagrama de estado, de actividades, de despliegue, de actores del negocio y sistema, de flujo de datos, entre otros; tiene soporte para el lenguaje de modelado UML, permite importar y exportar ficheros XML, además de ser robusto y tener gran usabilidad y fiabilidad.

En este capítulo se analizaron algunos conceptos como nutrición, neurodesarrollo, atención temprana, estado nutricional, evaluación dietética y metodología para la compresión del problema planteado; se realizó una breve descripción del programa “Renacer Contigo”, así como, del Sistema de Evaluación del Neurodesarrollo en Niños específicamente en el Módulo Nutrición. Se analizó la metodología de desarrollo de software empleada, la cual permitió guiar todo el proceso de desarrollo en particular los flujos de trabajo implementación y prueba que son los llevados a cabo en este trabajo, así como la descripción de las librerías, plataformas, lenguajes y herramientas propuestas por el departamento de Gestión Hospitalaria. Lo expuesto posibilitó un mayor entendimiento del problema a resolver, para dar continuidad a la propuesta realizada por el analista.

CAPÍTULO 2: DESCRIPCIÓN DE LA ARQUITECTURA DE SOFTWARE DEL SISTEMA DE EVALUACIÓN DEL NEURODESARROLLO EN NIÑOS.

En el capítulo se pretenden alcanzar los objetivos de especificar los requerimientos no funcionales que apoyarán la interfaz y el funcionamiento del software. Se describirá la arquitectura utilizada que es la definida por el departamento de gestión hospitalaria, también las estrategias de integración para conocer cómo se realiza la misma. Además, de conocer cómo se garantizará la seguridad de la aplicación, la distribución física del sistema mediante la vista de despliegue y las estrategias de codificación, estándares y estilos utilizados.

2.1 Requisitos No Funcionales.

Los requisitos no funcionales son propiedades o cualidades, que el producto debe tener. A través de estos, se especifican propiedades del sistema como restricciones de ambiente y desarrollo, rendimiento, dependencias de plataformas y mantenimiento. Debe pensarse en estas propiedades, como las características que hacen al producto atractivo, usable, rápido y confiable.

Usabilidad.

“La usabilidad se refiere a la capacidad de un software de ser comprendido, aprendido, usado y ser atractivo para el usuario, en condiciones específicas de uso.” [28]

El sistema estará diseñado de manera que los usuarios adquieran las habilidades necesarias para explotarlo en un tiempo reducido:

Para alcanzar un nivel Elemental asociado al dominio del sistema y el uso eficiente del mismo, serán necesarios 20 días de preparación, obteniendo la categoría de Usuarios normales.

Para alcanzar un nivel Avanzado asociado al dominio del sistema y el uso eficiente del mismo, serán necesarios 30 días de preparación, obteniendo la categoría de Usuarios avanzados.

La estructura concebida para la organización de la información agiliza el entendimiento del sistema por parte del usuario.

Los usuarios serán capaces de alcanzar sus objetivos con un mínimo esfuerzo y obteniendo los resultados máximos.

Fiabilidad.

“La fiabilidad es la característica de los sistemas informáticos por la que se mide el tiempo de funcionamiento sin fallos.” [29]

En la investigación se propone garantizar la misma de diversas formas:

En los servidores de los hospitales se deberá proporcionar una arquitectura de máxima disponibilidad, tanto de servidores de aplicación como de base de datos. Se plantea el empleo de políticas de respaldo a toda la información, para evitar pérdidas en caso de desastres ajenos al sistema.

El sistema soportará el uso de firmas digitales para la transferencia de información cuya certificación sea imprescindible, para validar el uso de la misma.

Ninguna información que se haya ingresado en el sistema será eliminada físicamente de la base de datos, independientemente de su inexistencia para el sistema. Permitirá la recuperación de la información de la base de datos, a partir de los respaldos, o salvas realizadas.

Eficiencia.

Imponen condiciones a los Requerimientos Funcionales. Por ejemplo, para una acción específica pueden definirse parámetros tales como: velocidad de procesamiento o cálculo, rendimiento, disponibilidad, precisión, tiempo de respuesta, tiempo de recuperación y aprovechamiento de los recursos.

- El Centro de Datos permitirá agregar recursos para aumentar el poder de procesamiento y almacenamiento sin afectar los sistemas, garantizando expansiones motivadas por futuros requerimientos.
- El sistema minimizará el volumen de datos en las peticiones y además, optimizará el uso de recursos críticos como la memoria. Para ello se potenciará como regla, guardar en la memoria caché datos y recursos de alta demanda.

- El sistema respetará buenas prácticas de programación para incrementar el rendimiento en operaciones costosas para la máquina virtual como la creación de objetos. Se deberá usar siempre que sea posible, el patrón Singleton (instancia única), destruir referencias que ya no estén siendo usadas, y optimizar el trabajo con cadenas, entre otras buenas prácticas, que ayudan a mejorar el rendimiento.

Réplica.

Se permitirá realizar réplica de la base de datos de los hospitales con el Centro de Datos. Esta réplica se podrá hacer de forma manual y automatizada, a través de la red.

Restricciones de diseño.

Este tipo de requerimiento, especifica o restringe la codificación o construcción de un sistema; son restricciones que han sido ordenadas y deben ser cumplidas estrictamente.

Ejemplos de ellas son:

- Estándares requeridos.
- Lenguajes de programación a ser usados para la implementación.
- Uso obligatorio de ciertas herramientas de desarrollo.
- Restricciones en la arquitectura o el diseño.
- Bibliotecas de clases.

Soporte.

Abarcan las acciones que se llevarán a cabo una vez que se ha terminado el desarrollo del software, con motivos de asistir a los clientes de este, logrando así su mejora progresiva y evolución en el tiempo. Pueden incluir: pruebas, extensibilidad, adaptabilidad, mantenimiento, configuración, compatibilidad, servicios, instalación.

- Las notificaciones de las deficiencias detectadas en la aplicación desplegada deberán realizarse por escrito.

- Una vez notificada por la entidad médica, la deficiencia detectada en la aplicación desplegada, el equipo de desarrollo deberá solucionarla en un período de 7 días.

La capacitación y entrenamiento del profesional de salud para el uso del sistema, se realizará en un período de 6 meses.

2.2 Descripción de la arquitectura.

Una de las tareas más importantes en el desarrollo de cualquier sistema de información es la definición de su ambiente de desarrollo. La arquitectura de software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos, para la creación de un producto de software.

Una de las tareas importantes de la arquitectura de software es, la elección de patrones a utilizar para el desarrollo de software. Los patrones no son más que una solución de diseño de software a un problema, aceptada como correcta, a la que se ha dado un nombre y que puede ser aplicada en otros contextos. También expresan una organización estructural para un sistema de software, proveen un conjunto de subsistemas predefinidos, e incluyen reglas y lineamientos para conectarlos.

La arquitectura definida para el desarrollo del sistema, está basada en el patrón arquitectónico Modelo Vista Controlador. Este patrón separa el modelado del dominio, la presentación y las acciones basadas en datos ingresados por el usuario en tres clases diferentes:

Modelo: Administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador). Representado por las clases autogeneradas, personalizadas y componentes Hibernate, que permite las operaciones con datos de una base de datos, aprovechando las ventajas de la orientación a objetos.

Vista: Maneja la visualización de la información; compuesta por páginas XHTML, las cuales contienen formularios que mediante controles JSF y RichFaces que recogen, validan y muestran, los datos que el usuario provee y solicita en cada una de las operaciones que realiza.

Controlador: Controla el flujo entre la vista y el modelo (los datos). Integrado por las clases controladoras (beans) que encierran la lógica del negocio del módulo, permiten el manejo de las acciones que el usuario

realiza sobre la vista modificando los datos y las entidades, en el modelo para ser mostrada nuevamente al usuario. Se utiliza Seam como Framework de integración que une las capas, modelo y vista. (Ver la Figura 2).

Figura 2. Visión general de la Arquitectura Modelo Vista Controlador

2.3 Posibles implementaciones de componentes o módulos que puedan ser reutilizados. Estrategias de integración.

El Sistema de Gestión Hospitalaria (alasHIS), cuenta con varios subsistemas, entre los que se encuentra el Sistema de Evaluación del Neurodesarrollo en Niños (SENDN). Nutrición es un módulo perteneciente ha dicho sistema, como se muestra en la Figura 3:

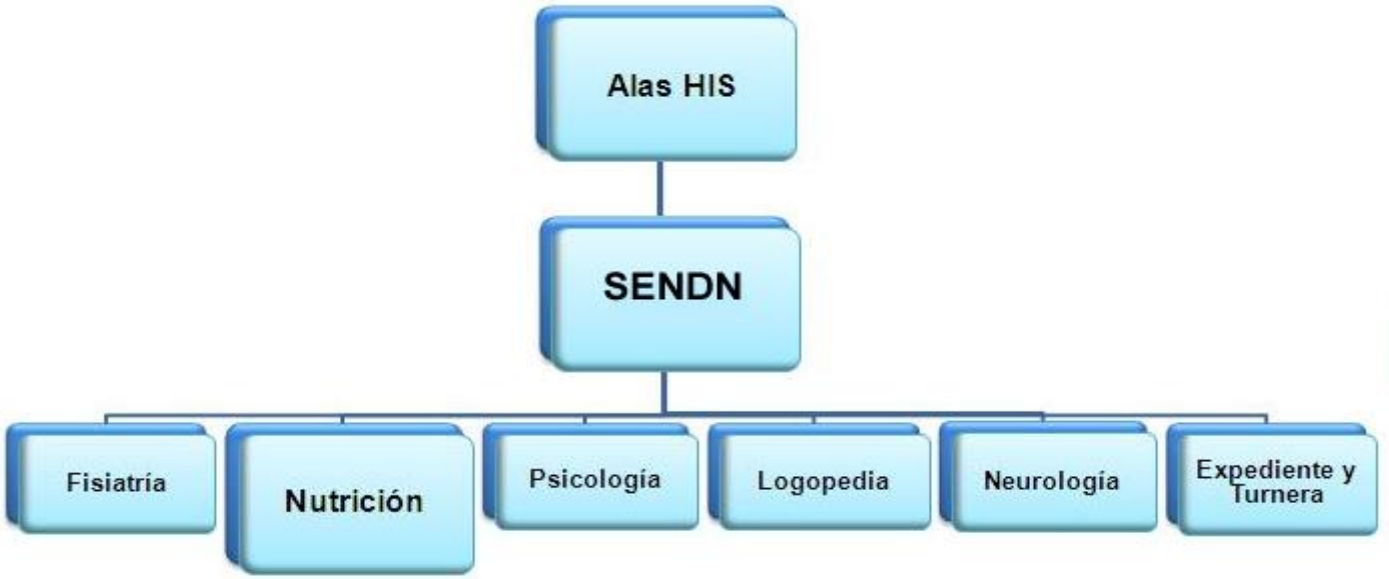


Figura 3. Visión estructural del sistema de Evaluación del Neurodesarrollo en Niños.

Al sistema de Evaluación del Neurodesarrollo en Niños del cual forma parte el Módulo de Nutrición, se le hace imprescindible para llevar a cabo la implementación de sus funcionalidades, la integración con el Módulo Expediente y Turnera. Dicho módulo brinda la posibilidad de obtener un listado con los datos personales del paciente y otro con los que están esperando para ser atendidos. Para obtener dichos listados, se hace necesaria la conexión del Módulo de Expediente y Turnera con el Sistema de Gestión Hospitalaria (HIS), a través de una de las tablas de la base de datos de expediente, ModeExpediente. Esta conexión viene dada con el componente HC_Local del HIS, que gestiona entre otros aspectos, los datos personales del paciente en su tabla Hoja_Frontal con su identificador, que es lo que se relaciona con Expediente.

Es necesario implementar en los controladores del sistema la interacción de los módulos: Nutrición y Expediente, manejando los datos desde el componente ModeExpediente y su identificador, unido a la tabla de Nutrición que manipula esa información que sería en el componente Modn_evaluación, que es el que controla la integración con Expediente.

2.4 Seguridad.

La seguridad es un elemento primordial para toda aplicación, pues en ella se define cómo se va a gestionar el acceso de los usuarios al sistema, así como, los permisos efectivos para cada uno de ellos. Específicamente, el sistema propuesto en lo que se refiere al acceso al sistema, será a nivel de usuarios y contraseñas. Un usuario podrá tener más de un rol en el sistema de acuerdo con las acciones que realiza, mediante el cual, se le otorgan determinados permisos para el acceso a la información y cada vez que el usuario realice una acción sobre el sistema, se registrará una traza que contiene la información gestionada mediante su estancia en el módulo. Estas contraseñas solo pueden ser alteradas por el usuario en cuestión, o por el administrador, en caso excepcional.

A continuación, se describen detalladamente las funcionalidades que ofrece el módulo de Configuración del Sistema de Información Hospitalaria alas HIS, el cual se encargará de garantizar la seguridad en el sistema a desarrollar.

Iniciar/Cerrar sesión de trabajo:

Cuando el usuario necesita acceder al sistema, este solicita: nombre de usuario y contraseña. El usuario introduce los datos solicitados, el sistema verifica que los datos introducidos sean válidos, si es así, el usuario accede al módulo Nutrición. El sistema muestra como opciones del menú, las funcionalidades a las que tiene permiso de acceder el usuario en el módulo, lo que garantiza el acceso de los mismos solo a los niveles establecidos de acuerdo con la función que realizan. El sistema permite: cerrar sesión y salir del módulo.

Registrar trazas:

Cuando el usuario realiza una acción sobre el sistema, que puede ser: inicio o cierre de sesión, acceso al módulo, modificación a un atributo de una entidad, o cualquier otra operación sobre el sistema, este registra una traza en la base de datos.

Administración de seguridad:

El sistema brinda la posibilidad de asignar o denegar permiso a los diferentes roles y usuarios, en los módulos y funcionalidades dentro de estos y también, la eliminación de roles y usuarios de las listas de los que se le negó o permitió algún permiso. Todos estos permisos son registrados por el sistema.

Configuración de funcionalidades:

Los usuarios del sistema pueden adicionar o eliminar las diferentes funcionalidades y categorías de un módulo en específico. En todas las capas de la aplicación se lleva a cabo la seguridad. En este caso toda la autorización, desde la que se realiza a directorios, páginas, controles, opciones del menú, servicios del negocio, está basado en reglas, esto permite que ninguna de estas “reglas del negocio” esté hard-coded en la aplicación y que el cambio de alguna de estas no requiera cambio alguno en el código, solo en la definición de alguna regla en un fichero de configuración. El Seam Security Framework permite todo esto gracias a su integración con el potente motor de reglas JBoss Rules.

2.5 Vista de despliegue

La vista de despliegue muestra la distribución física del sistema y sus conexiones. Cuando se modela la vista de despliegue, normalmente, se utilizarán los diagramas para modelar dicho sistema.

Los usuarios interactúan con la aplicación mediante la PC cliente, la misma debe contar con características como: un microprocesador 2.0 o una versión superior, una memoria RAM mínima de 256 MB, un sistema operativo que puede ser Windows o Linux, además debe tener un navegador web que puede ser internet Explorer 7, Firefox en su versión 2.0 en adelante u Opera 9.0, dicha PC se conecta a través del protocolo http con el servidor de aplicaciones. Dicho servidor Jboss cuenta con características particulares tales como: microprocesador Dual-Core, memoria RAM mínima de 4GB, como sistema operativo Linux, dicho servidor se conecta a través del protocolo TCP/IP al servidor de base de datos, el cual tiene sistema operativo Linux y como gestor de base de datos PostgreSQL en su versión 8.2. Por otra parte, para la impresión de documentos se utiliza una impresora que estará conectada a la PC cliente por el puerto USB. (Ver Figura 4)

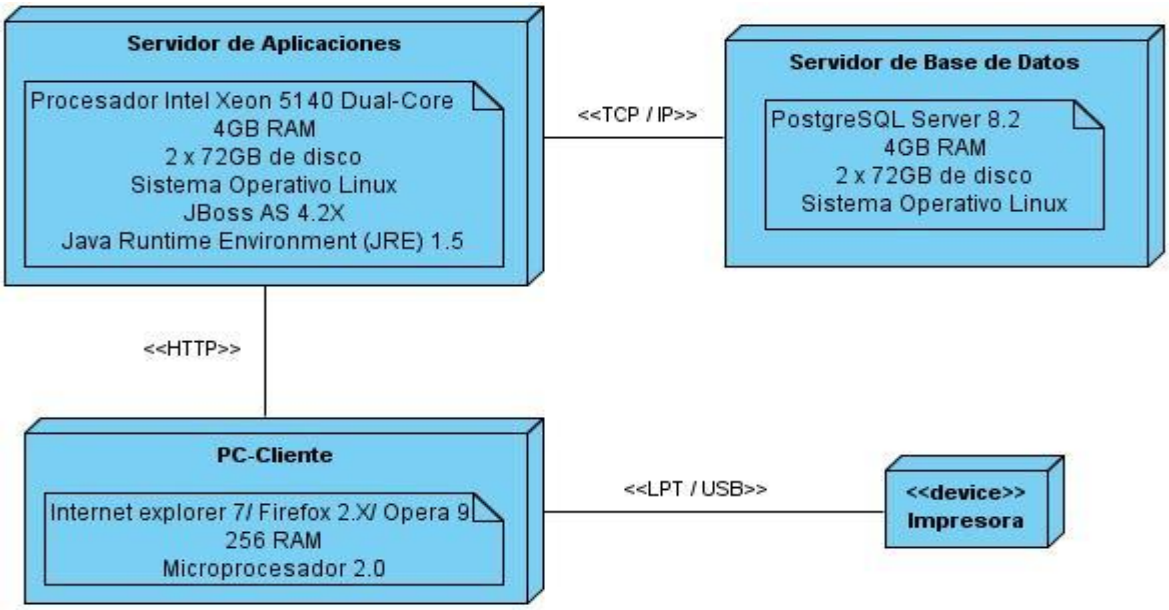


Figura 4. Diagrama Despliegue

2.6 Estrategias de codificación. Estándares y estilos a utilizar.

Los estándares han sido la herramienta base de la interoperabilidad informática. Son los que han permitido definir cómo interactuarán los componentes informáticos. La especificación de un estándar, a su vez, es aquel conjunto de documentos donde se define cómo llevar a cabo un desarrollo de software o hardware.

El Módulo de Nutrición, sigue las pautas de diseño del Sistema de Información Hospitalaria (HIS). Estas pautas permiten lograr una mayor efectividad en el proceso de trabajo al existir una mayor coherencia formal entre los módulos y páginas del sistema, y que estos sean identificados como parte de un todo. Se han pautado una serie de elementos comunes que facilitarán su reconocimiento y el uso que se haga de ellos.

General.

- El género de todos los actores que aparezcan en el prototipo será masculino**.**

- El formato de la fecha es dd/mm/yyyy, en el caso de la hora es hh:mm a.m. /p.m., en el caso de que se requieran ambos campos se manipularán por separado, es decir, se tratarán como dos atributos diferentes. En ambos casos se utilizará para ello el rich: calendar.

- En el título de los formularios se pondrá el nombre de la acción en infinitivo, la capitalización es igual a la de las etiquetas. Ejemplo: crear resumen, realizar evaluación.

- No se podrán utilizar datos pertenecientes a miembros del grupo de desarrollo como juego de datos.

Mensajería.

- El texto del mensaje será el que está identificado en el documento “Pautas de Mensajes del Sistema.doc” y se mostrará alineado a la izquierda.

- Los botones de los mensajes deben cumplir las mismas reglas que la sección de Botones.

- En el caso de usar un componente que permita marcar y desmarcar entre elementos de dos listas se utilizará el list shuttle no poniéndose etiquetas en los íconos de acción (>>, >, <, <<).

- Todos los datos de muestra que se pongan en el prototipo cumplirán con las mismas reglas de capitalización que las etiquetas.

- En los casos en que se requiera seleccionar un rango de fechas éste se pondrá mediante las etiquetas desde: y hasta: asociadas a los calendarios correspondientes. En ambos casos se utilizará para ello el rich: calendar.

El simpletogglepanel asociado a los criterios de búsqueda o de criterios generación de los reportes se titulará “criterios de búsqueda”, apareciendo inicialmente en la página los componentes asociados a los parámetros de búsqueda y los botones correspondientes, una vez que se presione el botón de buscar o generar, entonces se mostrará la tabla con los resultados de la búsqueda.

En las ventanas de Información y Error se mostrará solo un botón de “Aceptar”, centrado en la parte inferior.

En las ventanas de Advertencia se utilizarán dos botones centrados en la parte inferior y contendrán los textos: “Sí” y “No”. El botón “Sí” estará a la izquierda y el “No” a la derecha.

A la izquierda del mensaje aparecerá una imagen que sería el ícono que indica el tipo de mensaje (advertencia (), información () y error ()).

En el título de la ventana aparecerá el tipo de mensaje que se muestra.

Los botones que llevarán estos mensajes son los que se describen en “pautas de mensajes del sistema.doc” en la sección de acciones.

Tablas.

Las tablas aparecerán centradas con los colores definidos en el estilo CCS.

El nombre de la tabla aparecerá en texto alineado a la izquierda y en negrita. Cumpliendo las mismas reglas de capitalización que las etiquetas.

Las tablas que representen listados de pacientes, medicamentos, productos o cualquier otra entidad su nombre comenzarán precedido por el texto Listado de <entidad>.

El nombre de la columna aparecerá alineado a la izquierda y en negro (R0 G0 B0 - #000000). Cumpliendo las mismas reglas de capitalización que las etiquetas. Se le ubicará el título de “Foto” a la columna en la que se muestre la foto del paciente, no así en las columnas donde se ubiquen los íconos de: “Ver” (), “Modificar” (), “Eliminar” (), “Seleccionar” (), Crear (), donde el encabezado de la columna quedará en blanco.

- El contenido de la tabla será alineado a la izquierda excepto en las columnas que se muestren datos numéricos que el contenido se alineará a la derecha.
- Las tablas resultantes de una búsqueda utilizarán columnas diferentes para cada uno de los atributos a mostrar. Los campos de fecha y hora siempre irán en columnas diferentes igualmente sucederá con los campos asociados a los datos del paciente, leyendo los atributos por columnas. En el caso de mostrar el nombre y apellidos de los actores del sistema, se mostrarán todos en una misma columna.

En el caso de los atributos que permitan valores nulos y que no posean valor alguno en la Base de Datos, no se mostrará nada en la celda asociada al mismo en los reportes, es decir, se dejará en blanco.

- Para todos los reportes se debe definir un orden homogéneo para los parámetros de salida, evitando que reportes que muestran información común visualmente los atributos se muestren en un orden diferente, pudiera coincidir con el orden en que se muestran los criterios de búsqueda.

- En el caso de reportes que no lleven parámetros de búsqueda, no se colocará un botón de generar innecesario después de seleccionar el reporte en el Menú, sino que a partir de esta acción, se obtendrá directamente el reporte sin la acción del botón de generar intermedia.

- En el caso de reportes con criterios de búsqueda contendrá los botones de Generar y Cancelar debajo de los criterios de búsqueda, en el caso de que no posea parámetros de búsqueda, los botones serán Imprimir y Salir, en caso de que el reporte sea imprimible y solo Salir en caso de no poseer esta acción.

Botones.

- Los colores están definidos en el estilo CSS.

- El tamaño será el estándar del IDE.

- Aparecerán alineados en la parte inferior a la derecha y el orden será de tal forma, que las acciones positivas al flujo sea de izquierda a derecha.

- El nombre de los botones tiene que cumplir las reglas de las “Pautas de Etiquetado.doc”. Por lo que solo se pueden usar textos que se muestran en este juego de etiquetas, es decir, solo las acciones especificadas en este documento.

- El orden de los botones será de máxima a mínima prioridad de izquierda a derecha, es decir, a la izquierda siempre estará el botón de mayor prioridad.

- La ubicación de la sección de botones será la que se muestra teniendo en cuenta los siguientes casos, las etiquetas no tendrán separación alguna con el componente asociado, la organización de los componentes en las interfaces debe ser, como se observa la búsqueda avanzada va debajo de

la línea de botones y los casos en que los botones van debajo de los componentes estos deben ir alineados a la derecha al último componente. Lo más importante es la separación en píxel dentro de la interfaz, entre el header del panel y la primera línea de componentes 13 píxeles, la primera línea de componentes estará a 8 píxeles del lado izquierdo del panel, entre la primera columna de componentes y la segunda 25 píxeles.

Editores de Texto.

- Tendrán tamaño estándar del IDE.
- Se pondrán alineados a la izquierda.
- Los editores asociados con etiquetas se pondrán debajo de la etiqueta, alineados ambos a la izquierda.

Etiquetas.

- El tamaño y el color están definidos en el estilo CSS.
- Todas las etiquetas estáticas irán en negro **(R0 G0 B0 - #000000)**. La clase de estilo a utilizar es la siguiente:

```
.normalText
        {
    font-family:Verdana !important;
    font-size:10px !important;
  color:#000000 !important;
        }
```

- En caso de mostrar información de solo lectura será mostrada de otro color **(R92 G92 B92 - #5c5c5c)** y se pondrá al lado de la etiqueta que lo identifica la cual estará acompañada de dos puntos. La clase de estilo a utilizar es la siguiente:

```
dataText

        {
```

font-family: Verdana!important;

font-size: 10px!important;

color:#5c5c5c !important; }

- Todas las etiquetas estáticas terminarán con dos puntos (:).
- El texto se escribirá de la forma que sigue la primera letra de la primera palabra en mayúscula, el resto del texto de la etiqueta en minúscula, excepto para aquellos atributos específicos que requieran otra capitalización.

Selectores.

- Los componentes de selección o combobox, visualizarán el texto por defecto: <Seleccione>.
- El texto cumplirá con las mismas reglas de capitalización que las etiquetas Verdana 10px.
- En los selectores donde se permita seleccionar todos los elementos asociados a este, se colocará una opción de Todos o Todas, en dependencia del género del campo asociado.
- El orden de las opciones será: <Seleccione>, Todos o Todas (en caso de ser necesario) y el resto de las opciones en orden alfabético ascendente.

Secciones.

Serán nombradas con un texto que sea afín con la sección el cual no terminará en ningún signo de puntuación.

- El nombre estará alineado a la izquierda, Verdana 12 en negrita, no terminando ni en (.) ni en (:).
- Se utilizarán los paneles.
- El tamaño será a 100% del espacio asignado.
- Los nombres cumplirán con la misma capitalización que las etiquetas.

- Los paneles contenedores tendrán tamaño variable el cual se ajustará en la medida de la cantidad de componentes visuales que contendrá.

Estándares de codificación.

Para definir el estilo de codificación a seguir en la aplicación, se utilizó la notación estándar establecida para las aplicaciones desarrolladas en JAVA (Java Code Conventions), definido en las especificaciones del propio lenguaje.

Nomenclatura.

- Los nombres de cada uno de los elementos del programa deben ser significativos; su nombre debe explicar en lo posible el uso del elemento.
- La mayoría de los elementos se deben nombrar usando sustantivos (posiblemente compuestos), o formas verbales en imperativo.
- La forma de construir los nombres será colocando primero el verbo o el sustantivo, seguido de cada uno de sus complementos con la primera letra en mayúscula.

Archivos fuente.

Cada programa en Java es una colección de uno o más archivos. El programa ejecutable se obtiene compilando estos archivos. En cada archivo especifica su contenido como sigue:

- Los paquetes (instrucción package).
- Los archivos de biblioteca (instrucciones import).
- Un comentario explicando el objetivo del archivo.
- Las clases que defines en ese archivo.

Clases.

Cada clase debe ir precedida por un comentario que explique su objetivo. Es recomendable especificar sus elementos como sigue:

- Estructura de los objetos. Primero las variables y luego las constantes.
- Elementos estáticos.
- Constructores.
- Métodos públicos y privados.
- Métodos estáticos.
- Clases internas.

Dejar una línea en blanco después de cada método. Todos los elementos deben estar precedidos por public, private o protected. Las variables deben ser privadas. Los métodos y las constantes pueden ser privados o públicos, según se requiera.

Tamaño de Líneas.

Las líneas deben ser de un máximo de 99 caracteres. Si es necesario partir la línea, la siguiente línea debe alinearse dejando doble sangría.

Generalidades.

Se empleará el formalismo de precondición, poscondición e invariante. El código debe comentarse utilizando la sintaxis apropiada para uso de javadoc, teniendo en cuenta que para la producción de la documentación, deben incorporarse los tags particulares que no hagan parte del estándar.

En el caso de las poscondiciones, se utilizará la notación prima (x') para referirse al valor de una variable al finalizar el método, y sin prima (x) para su valor antes de iniciarse la ejecución de la rutina. Por ejemplo una rutina que incrementa el valor de x en uno puede especificarse con: @pos x' == x + Los conectores lógicos que se deben emplear son los de Java: &&, ||! Se asume que los atributos que no figuran en la poscondición, no sufren modificaciones. En el caso de funciones analizadoras, éstas no deben afectar atributos, y su poscondición se especifica con el tag @return de javadoc.

Sangría y Ablocamiento.

Todos los archivos fuentes deben seguir el estándar de sangría. Cada entrada corresponde a 4 espacios. No debe usarse el carácter de tabulación, pues es dependiente de la configuración del editor. Todos los constructores que admiten bloques de instrucciones delimitados por {…}, deben usarlos aún, si éstos tienen una sola instrucción. Las funciones deben tener un solo punto de retorno, y el flujo normal de las estructuras de iteración no debe alterarse mediante instrucciones break o continue.

Aserciones.

Se recomienda el uso de aserciones para validación de poscondiciones, invariantes y otros puntos críticos según sea el caso. Para el caso del invariante de la representación, se recomienda la implantación del método invarainteOk, de modo que se pueda invocar desde las aserciones de poscondición de todos los métodos.

Como resultado del análisis realizado en este capítulo, se obtuvo una descripción de la arquitectura propuesta por el departamento de Gestión Hospitalaria, verificándose la eficiencia del patrón Arquitectónico (MVC). Se analizaron los requerimientos no funcionales del sistema, siendo estos de vital importancia para el mismo, además de que permitió hacer un estudio de cómo lograr un sistema con buena seguridad, parámetro imprescindible para todo producto para lograr la confidencialidad, autenticidad e integridad de los datos e información que en él se gestiona. Se elaboró el diagrama de despliegue como propuesta para el establecimiento del sistema, identificando los nodos físicos que lo componen. Por último, se realizó un estudio de los estándares de codificación y estilos de programación a utilizar para un mejor entendimiento y organización del código, definidos previamente por el Departamento de Gestión Hospitalaria.

CAPÍTULO 3: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA.

Los objetivos de las autoras con la elaboración de este capítulo, es realizar una valoración crítica del diseño propuesto por el analista; obtener los diagramas de clases del diseño e interacción; describir las nuevas clases u operaciones necesarias. Exponer el modelo de datos y una evaluación de las técnicas de validación, obtener además, una breve descripción de las tablas, así como la vista de implementación, la cual incluye el diagrama de componentes. Estos, describen los elementos físicos del sistema y sus relaciones, muestran además las opciones de realización incluyendo código fuente, binario y ejecutable.

3.1 Valoración crítica del diseño propuesto por el analista.

El diseño, es el primer paso en la fase de desarrollo de cualquier producto o sistema de ingeniería y su objetivo es producir un modelo o representación, de una entidad que se va a construir posteriormente. En esta etapa, se traducen los requerimientos funcionales y no funcionales en una representación de software.

Los diagramas de clases de diseño, exponen un conjunto de interfaces, colaboraciones y relaciones. Especifican la estructura de clases del sistema y las relaciones entre clases y estructuras de herencia. Se realizan para lograr un mejor entendimiento del sistema.

El diseño del Módulo Nutrición del Sistema de Evaluación del Neurodesarrollo en Niños, solo contempla el diseño de los casos de uso del paquete de pruebas, el cual además, se encontraba incompleto, por lo que se le agregaron nuevos atributos a algunas tablas. En otros casos, se le eliminaron atributos que no tenían sentido que existieran, las relaciones entre algunas de las tablas eran innecesarias y en otras, necesarias para poder llevar a cabo la implementación. Todas estas razones, dieron origen al refinamiento del diseño propuesto por el analista.

Además, se hizo necesario realizar el diseño del paquete configuración. El mismo no se encontraba en el diseño realizado en el curso 2009-2010, para lo cual se generaron los diagramas de clases del diseño y de interacción de este paquete. A continuación se muestran las clases u operaciones, se podrán consultar el resto de las tablas en el Anexo1.

3.2 Descripción de las nuevas clases u operaciones necesarias.

Nombre de la Clase: Valor Nutritivo	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Integer
version	Integer
grupoAlimento	String
energia	Integer
proteina	Integer
grasas	Integer
carbohidratos	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Tabla 1 Clase modelo valor nutritivo

Nombre de la Clase: Valor Nutritivo	
Tipo de la Clase: Controlador	
Nombre:	**insertarValorNutritivo()**
Descripción:	Inserta los datos del valor nutritivo
Nombre:	**modificarValorNutritivo()**
Descripción:	Modifica los datos del valor nutritivo
Nombre:	**eliminarValorNutritivo()**
Descripción:	Elimina los datos del valor nutritivo.

Tabla 1 Clase controladora valor nutritivo

Nombre de la Clase: Valor Nutritivo	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear valor nutritivo

Tabla 1 Clase vista valor nutritivo

Nombre de la Clase: Longitud Supina Peso Talla Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Integer
version	Integer
longitud1	Float
longitud2	Float

eliminado	Boolean
cid	Integer
valor	String
codigo	String

Tabla 2 Clase vista modelo Longitud Supina Peso Talla Femenino

Nombre de la Clase: Longitud Supina Peso Talla Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarLongSupinaPesoTallaFem()**
Descripción:	Inserta los datos de longitud supina peso talla femenino
Nombre:	**modificarLongSupinaPesoTallaFem()**
Descripción:	Modifica los datos de longitud supina peso talla femenino
Nombre:	**eliminarLongSupinaPesoTallaFem()**
Descripción:	Elimina los datos de longitud supina peso talla femenino

Tabla 2 Clase controlador Longitud Supina Peso Talla Femenino

Nombre de la Clase: Longitud Supina Peso Talla Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la longitud supina para el peso y la talla femenino.

Tabla 2 Clase vista Longitud Supina Peso Talla Femenino

Nombre de la Clase: Peso Talla Femenino
Tipo de Clase: Modelo

Atributo	Tipo
Id	Integer
version	Integer
ModnNLongitudSupinaPesoTallaFem	modnNLongitudSupinaPesoTallaFem
peso	Float
percentil	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Tabla 3 Clase modelo Peso Talla Femenino

Nombre de la Clase: Peso Talla Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarModnNPesoTallaFem()**
Descripción:	Inserta los datos de peso talla femenino.
Nombre:	**modificarModnNPesoTallaFem()**
Descripción:	Modifica los datos peso talla femenino
Nombre:	**eliminarPesoTallaFem()**
Descripción:	Elimina los datos peso talla femenino

Tabla 3 Clase controlador Peso Talla Femenino

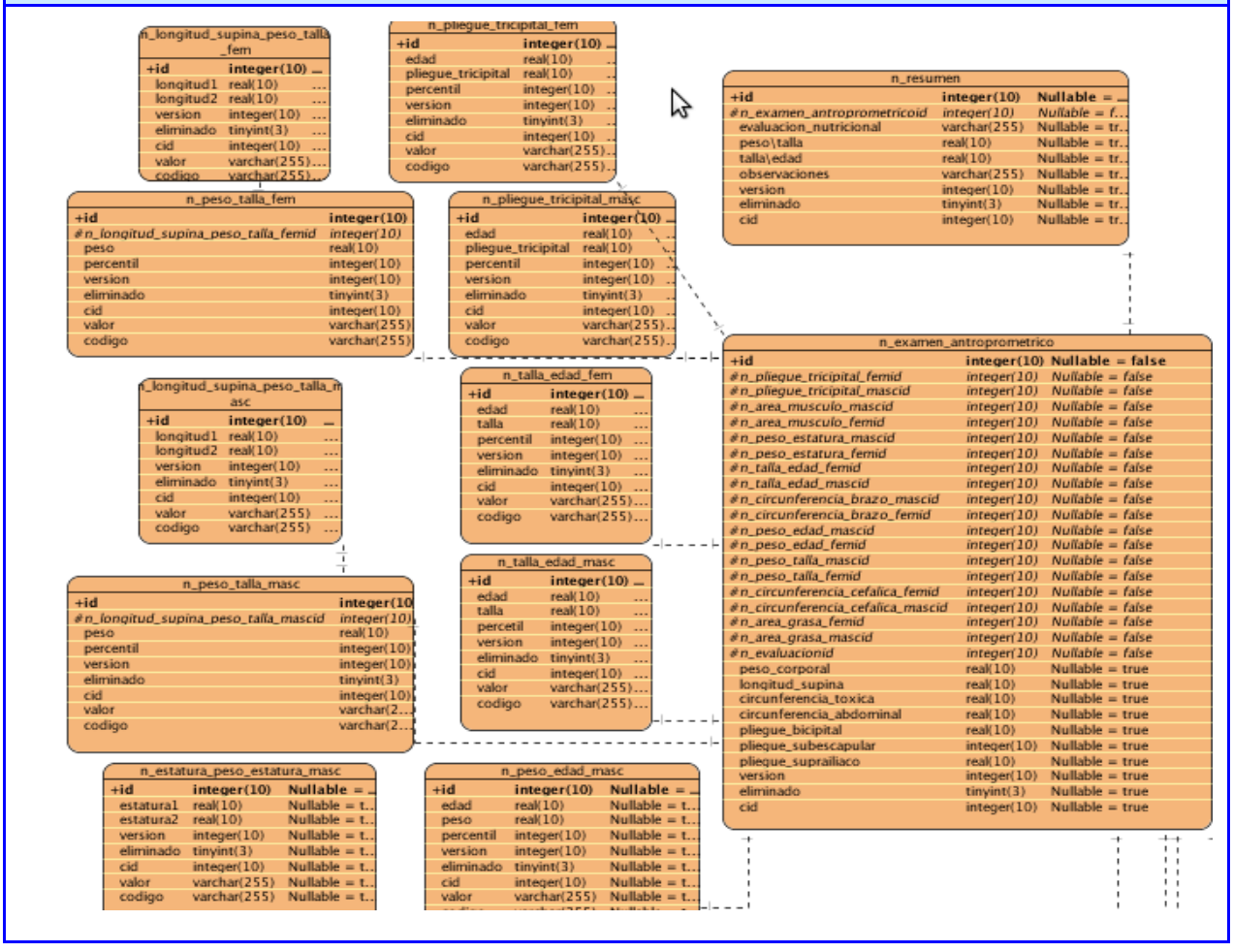
Nombre de la Clase: Peso Talla Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el peso para la talla femenino.

Tabla 3 Clase vista Peso Talla Femenino

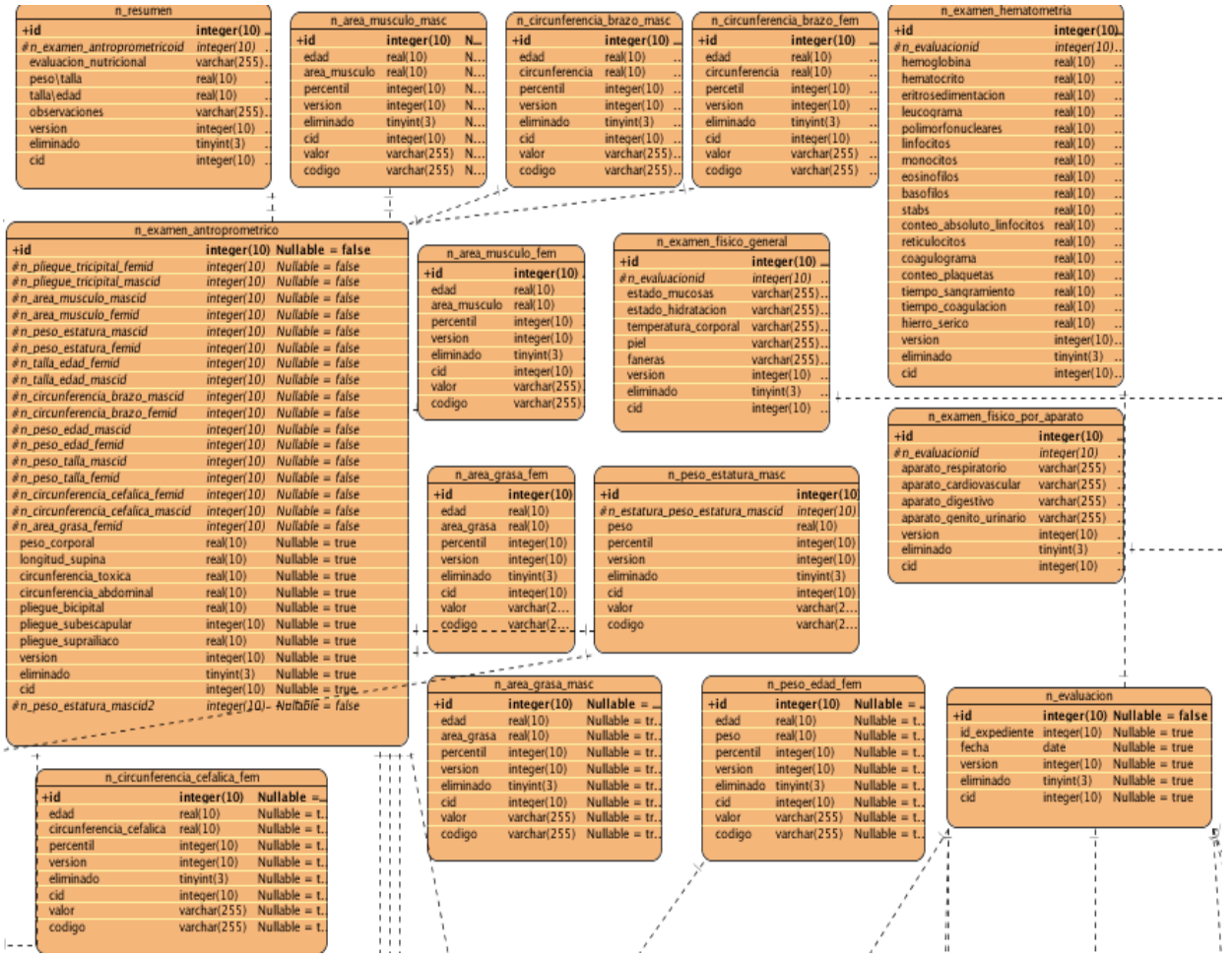
3.3 Modelo de datos.

"El modelo de datos es un lenguaje utilizado para la descripción de la base de datos, permite describir las estructuras de datos (el tipo de los datos que incluye la base de datos y la forma en que se relacionan)." [30] Ver Figura 5.

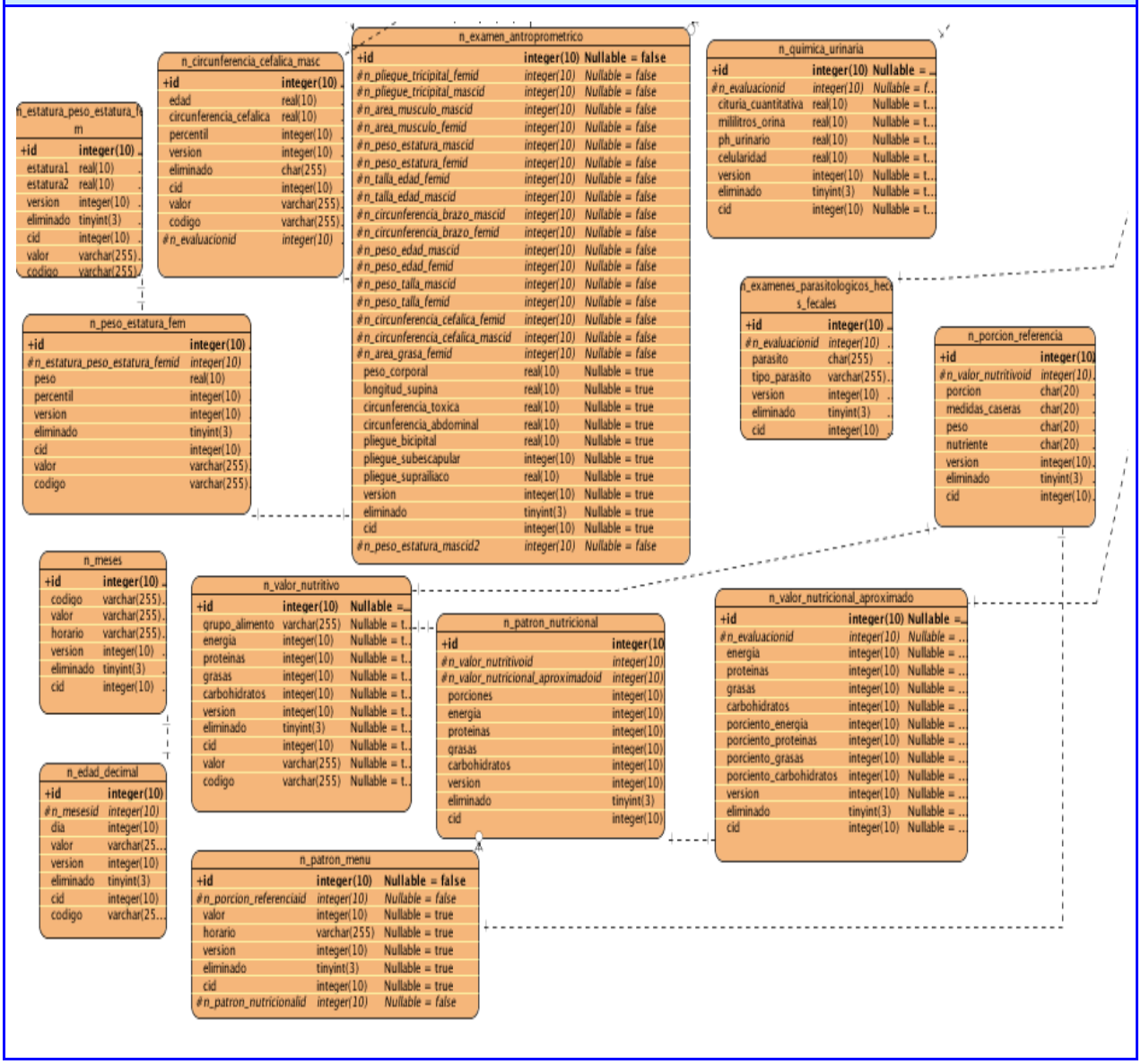
Modelo de datos.



Modelo de datos.



Modelo de datos.



Modelo de datos.

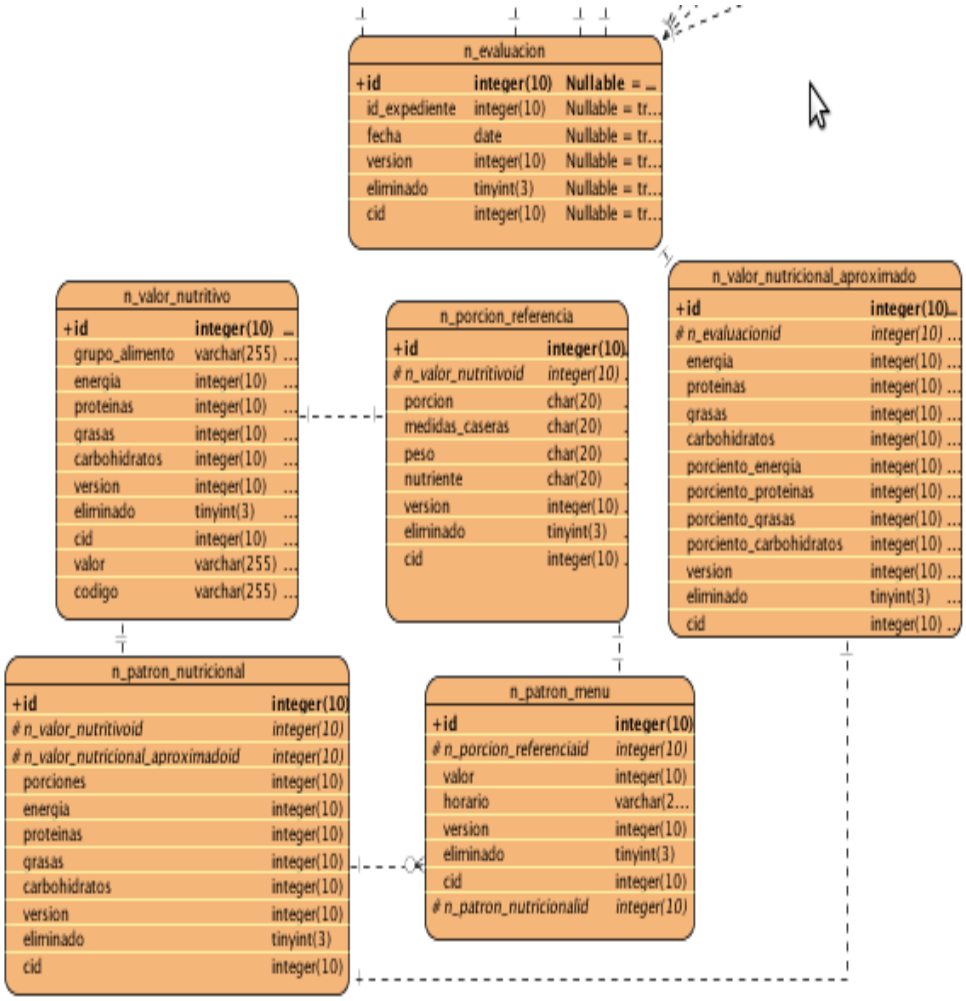


Figura 5. Modelo de datos.

3.4 Descripción de las tablas.

Nombre: n_edad_decimal		
Descripción: Esta tabla permite conocer la edad decimal del paciente.		
Atributo	**Tipo**	**Descripción**

id	Serial	Identificador de la tabla
id_ n_meses	Integer	Identificador de la tabla meses permite seleccionar los valores de meses.
dia	Integer	Atributo para conocer el día.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la edad decimal del paciente ha sido eliminado
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 4 n_edad_decimal

Nombre: n_edades		
Descripción: Esta tabla permite conocer las edades de los pacientes		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
edad	Float	Atributo para conocer la edad.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.

version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la edad ha sido eliminado
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 5 n_edades

Nombre: n_estatura_peso_estatura_fem		
Descripción: Esta tabla permite conocer el rango de estaturas para el sexo femenino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
estatura1	Float	Atributo para conocer la estatura inicial.
estatura2	Float	Atributo para conocer la estatura final.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad.
eliminado	Boolean	Permite verificar si la estatura peso estatura femenino ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 6 n_estatura_peso_estatura_fem

Nombre: n_estatura_peso_estatura_masc		
Descripción: Esta tabla permite conocer el rango de estaturas para el sexo masculino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
estatura1	Float	Atributo para conocer la estatura inicial.
estatura2	Float	Atributo para conocer la estatura final.
código	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
versión	Integer	Atributo para persistir o actualizar la entidad.
eliminado	Boolean	Permite verificar si la estatura peso estatura masculino ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora.

Tabla 7 n_estatura_peso_estatura_masc

Nombre: n_longitud_supina_peso_talla_fem		
Descripción: Esta tabla permite conocer el rango de longitudes del nomenclador para el sexo femenino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
longitud1	Float	Atributo para conocer la longitud inicial.
longitud2	Float	Atributo para conocer la longitud final.

codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad.
eliminado	Boolean	Permite verificar si la longitud supina peso talla femenino ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora.

Tabla 8 n_longitud_supina_peso_talla_fem

Nombre: n_longitud_supina_peso_talla_masc		
Descripción: Esta tabla permite conocer el rango de longitudes del nomenclador para el sexo masculino		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
longitud1	Float	Atributo para conocer la longitud inicial.
longitud2	Float	Atributo para conocer la longitud final.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad.
eliminado	Boolean	Permite verificar si la longitud supina peso talla masculino ha sido eliminado.

cid	Integer	Identificador de modificaciones registradas en la bitácora.

Tabla 9 n_longitud_supina_peso_talla_masc

Nombre: n_meses		
Descripción: Esta tabla permite conocer los meses.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si los meses han sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 10 n_meses

Nombre: n_valor_nutritivo		
Descripción: Esta tabla contiene el valor nutritivo por cada grupo de alimentos que debe consumir el paciente.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
grupo_alimento	Varchar	Atributo para conocer el grupo de alimento que debe consumir el paciente.
energia	Integer	Atributo para conocer la cantidad de energía.

proteina	Integer	Atributo para conocer la cantidad de proteína.
grasa	Integer	Atributo para conocer la cantidad de grasas.
carbohidrato	Integer	Atributo para conocer la cantidad de carbohidratos.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el valor nutritivo ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 11 n_valor_nutritivo

Nombre: n_talla_edad_fem		
Descripción: Esta tabla es para conocer la talla para la edad en el sexo femenino		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
talla	Float	Atributo para la talla.
percentil	Integer	Atributo para conocer el percentil.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.

valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la talla edad femenino ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 12 n_talla_edad_fem

Nombre: n_talla_edad_masc		
Descripción: Esta tabla es para conocer la talla para la edad en el sexo masculino		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
talla	Float	Atributo para la talla.
percentil	Integer	Atributo para conocer el percentil.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la talla edad masculino ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la

		bitácora

Tabla 13 n_talla_edad_masc

Nombre: n_pliegue_tricipital_fem		
Descripción: Esta tabla contiene el pliegue tricipital para el sexo femenino		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
pliegue_tricipital	Float	Atributo para conocer el pliegue tricipital.
percentil	Integer	Atributo para conocer el percentil.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el pliegue tricipital ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 14 n_pliegue_tricipital_fem

Nombre: n_pliegue_tricipital_masc		
Descripción: Esta tabla contiene el pliegue tricipital para el sexo masculino.		
Atributo	**Tipo**	**Descripción**

id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
pliegue_tricipital	Float	Atributo para conocer el pliegue tricipital.
percentil	Integer	Atributo para conocer el percentil.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el pliegue tricipital ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 15 n_pliegue_tricipital_masc

Nombre: n_peso_talla_fem		
Descripción: Esta tabla permite conocer el peso para la talla femenino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id _n_longitud_supina_peso_talla_fem	Integer	Identificador para obtener la longitud supina del peso para la talla femenino.
peso	Float	Atributo para conocer el peso.
percentil	Integer	Atributo para conocer el percentil.

codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el peso talla ha sido eliminado
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 16 n_peso_talla_fem

Nombre: n_peso_talla_masc		
Descripción: Esta tabla permite conocer el peso para la talla masculino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_ n_longitud_supina_peso_talla_masc	Integer	Identificador para obtener la longitud supina del peso para la talla masculino.
peso	Float	Atributo para conocer el peso.
percentil	Integer	Atributo para conocer el percentil.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad

eliminado	Boolean	Permite verificar si el peso talla ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 17 n_peso_talla_masc

Nombre: n_peso_estatura_fem		
Descripción: Esta tabla permite conocer el peso para la estatura femenino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_ n_estatura_peso_estatura_fem	Integer	Identificador para conocer la estatura del peso para la estatura femenino.
peso	Float	Atributo para conocer el peso.
percentil	Integer	
código	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el peso estatura ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 18 n_peso_estatura_fem

Nombre: n_peso_estatura_masc

Descripción: Esta tabla permite conocer el peso para la estatura masculino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_estatura_peso_estatura_masc	Integer	Identificador para conocer la estatura del peso para la estatura masculino.
peso	Float	Atributo para conocer el peso.
percentil	Integer	Atributo para conocer el percentil.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el peso estatura ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 19 n_peso_estatura_masc

Nombre: n_peso_edad_fem		
Descripción: Esta tabla permite conocer la relación peso- edad femenino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.

peso	Float	Atributo para conocer el peso.
percentil	Integer	Atributo para conocer el percentil.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el peso edad ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 20 n_peso_edad_fem

Nombre: n_peso_edad_masc		
Descripción: Esta tabla permite conocer la relación peso- edad masculino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
peso	Float	Atributo para conocer el peso.
percentil	Integer	Atributo para conocer el percentil.
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización del idioma.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.

version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el peso edad ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 21 n_peso_edad_masc

Nombre: n_circunferencia_cefalica_fem		
Descripción: Esta tabla contiene la circunferencia cefálica femenina.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
circunferencia_cefalica	Float	Atributo para la circunferencia cefálica.
percentil	Integer	Atributo para conocer el percentil
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización del idioma.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la circunferencia cefálica ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 22 n_circunferencia_cefalica_fem

Nombre: n_circunferencia_cefalica_mas		
Descripción: Esta tabla contiene la circunferencia cefálica masculina.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
circunferencia_cefalica	Float	Atributo para la circunferencia cefálica.
percentil	Integer	Atributo para conocer el percentil
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización del idioma.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la circunferencia cefálica ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 23 n_circunferencia_cefalica_mas

Nombre: n_circunferencia_brazo_fem		
Descripción: Esta tabla es para conocer la circunferencia del brazo femenino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la

		edad del paciente.
circunferencia	Float	Atributo para la circunferencia cefálica.
percentil	Integer	Atributo para conocer el percentil
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización del idioma.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la circunferencia del brazo ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 24 n_circunferencia_brazo_fem

Nombre: n_circunferencia_brazo_masc		
Descripción: Esta contiene los parámetros para determinar la circunferencia del brazo masculino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
circunferencia	real	Atributo para la circunferencia cefálica.
percentil	Integer	Atributo para conocer el percentil
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización del idioma.

valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la circunferencia del brazo ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 25 n_circunferencia_brazo_masc

Nombre: n_area_musculo_fem		
Descripción: Esta tabla es para determinar el área muscular femenino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
area_musculo	real	Atributo conocer el área músculo.
percentil	Integer	Atributo para conocer el percentil
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización del idioma.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el área músculo ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la

		bitácora

Tabla 26 n_area_musculo_fem

Nombre: n_area_musculo_masc		
Descripción: Esta tabla es para determinar el área muscular masculino.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
area_musculo	real	Atributo conocer el área músculo.
percentil	Integer	Atributo para conocer el percentil
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización del idioma.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el área músculo ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 27 n_area_musculo_masc

Nombre: n_area_grasa_fem		
Descripción: Esta tabla contiene los valores para conocer el área grasa femenina.		
Atributo	**Tipo**	**Descripción**

id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
area_grasa	Real	Atributo conocer el área grasa.
percentil	Integer	Atributo para conocer el percentil
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización del idioma.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el área grasa ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 28 n_area_grasa_fem

Nombre: n_area_grasa_masc		
Descripción: Esta tabla contiene los valores para conocer el área grasa masculina.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_n_edades	Integer	Identificador de la tabla edades para conocer la edad del paciente.
area_grasa	Real	Atributo conocer el área grasa.
percentil	Integer	Atributo para conocer el percentil
codigo	Varchar	Es el código por el cual se reconoce al nomenclador y se utiliza para la internacionalización

		del idioma.
valor	Varchar	Es el valor por el cual se reconoce a cada nomenclador dentro de las tablas.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el área grasa ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 29 n_area_grasa_masc

Nombre: n_patron_menu		
Descripción: Esta tabla es para guardar los parámetros para establecer el patrón del menú.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
id_n_porcion_referencia	Integer	Identificador para conocer la porción de referencia.
Id_n_patron_nutricional	Integer	Identificador para conocer el patrón nutricional.
horario	Varchar	Atributo para conocer el horario.
valor	Integer	Atributo para conocer el valor.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el patrón de menú ha sido eliminado
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 30 n_patron_menu

Nombre: n_patron_nutricional

Descripción: Esta tabla es para determinar el patrón de alimentación del paciente, que no es más que determinar la cantidad de intercambios que debe consumir.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla
Id_ n_valor_nutritivo	Integer	Identificador para conocer el valor nutritivo.
id_n_ valor_nutricional_aproximado	Integer	Identificador para conocer el valor nutricional aproximado.
porciones	Integer	Atributo para conocer las porciones.
energia	Integer	Atributo para conocer la cantidad de energía.
proteina	Integer	Atributo para conocer la cantidad de proteínas.
grasa	Integer	Atributo para conocer la cantidad de grasas.
carbohidrato	Integer	Atributo para conocer la cantidad de carbohidratos.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el patrón nutricional ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 31 n_patron_nutricional

Nombre: n_valor_nutricional_aproximado		
Descripción: Esta tabla es para determinar el valor nutricional aproximado que tiene el paciente.		
Atributo	**Tipo**	**Descripción**
id	Serial	Identificador de la tabla

Nombre: n_valor_nutricional_aproximado		
Id_n_evaluacion	Integer	Identificador para conocer el paciente, al cual se le conformará el patrón de menú.
energia	Integer	Atributo para conocer la cantidad de energía.
proteina	Integer	Atributo para conocer la cantidad de proteínas.
grasa	Integer	Atributo para conocer la cantidad de grasas.
carbohidrato	Integer	Atributo para conocer la cantidad de carbohidrato.
porciento_energia	Integer	Atributo para conocer el porciento de energía que necesita el paciente.
porciento_proteina	Integer	Atributo para conocer el porciento de proteína que necesita el paciente.
porciento_grasas	Integer	Atributo para conocer el porciento de grasas que necesita el paciente.
porciento_carbohidratos	Integer	Atributo para conocer el porciento de carbohidratos que necesita el paciente.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si el valor nutricional aproximado ha sido eliminado.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 32 n_valor_nutricional_aproximado

Nombre: n_porcion_referencia
Descripción: Esta tabla contiene todas las medidas de las porciones de referencias de los nutrientes para cada grupo de alimentos.

Atributo	Tipo	Descripción
id	Serial	Identificador de la tabla
Id_ n_valor_nutritivo	Integer	Identificador que tiene los valores nutricionales para cada grupo de alimento.
porcion	Integer	Atributo para conocer la porción de cada alimento.
medidas_caseras	Varchar	Atributo para conocer las medidas caseras para cada grupo de alimentos.
peso	Varchar	Atributo para conocer el peso.
nutriente	Varchar	Atributo para conocer los nutrientes.
version	Integer	Atributo para persistir o actualizar la entidad
eliminado	Boolean	Permite verificar si la porción de referencia ha sido eliminada.
cid	Integer	Identificador de modificaciones registradas en la bitácora

Tabla 32 n_porcion_referencia

3.5 Diagrama de Componentes

“Un diagrama de componente, representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes, los mismos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables o paquetes.

Los diagramas de componentes prevalecen en el campo de la arquitectura de software, pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema.” [31]

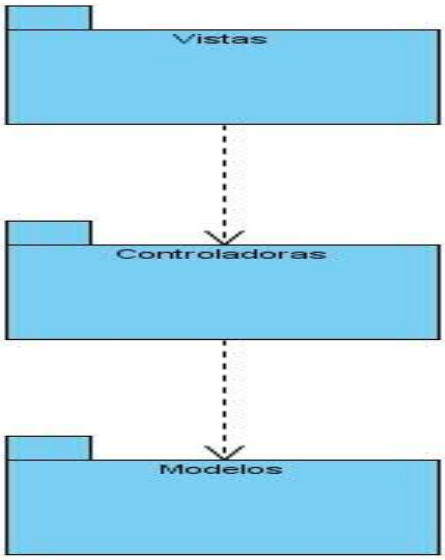


Figura 6 Diagrama de Componente Nutrición (Paquetes).

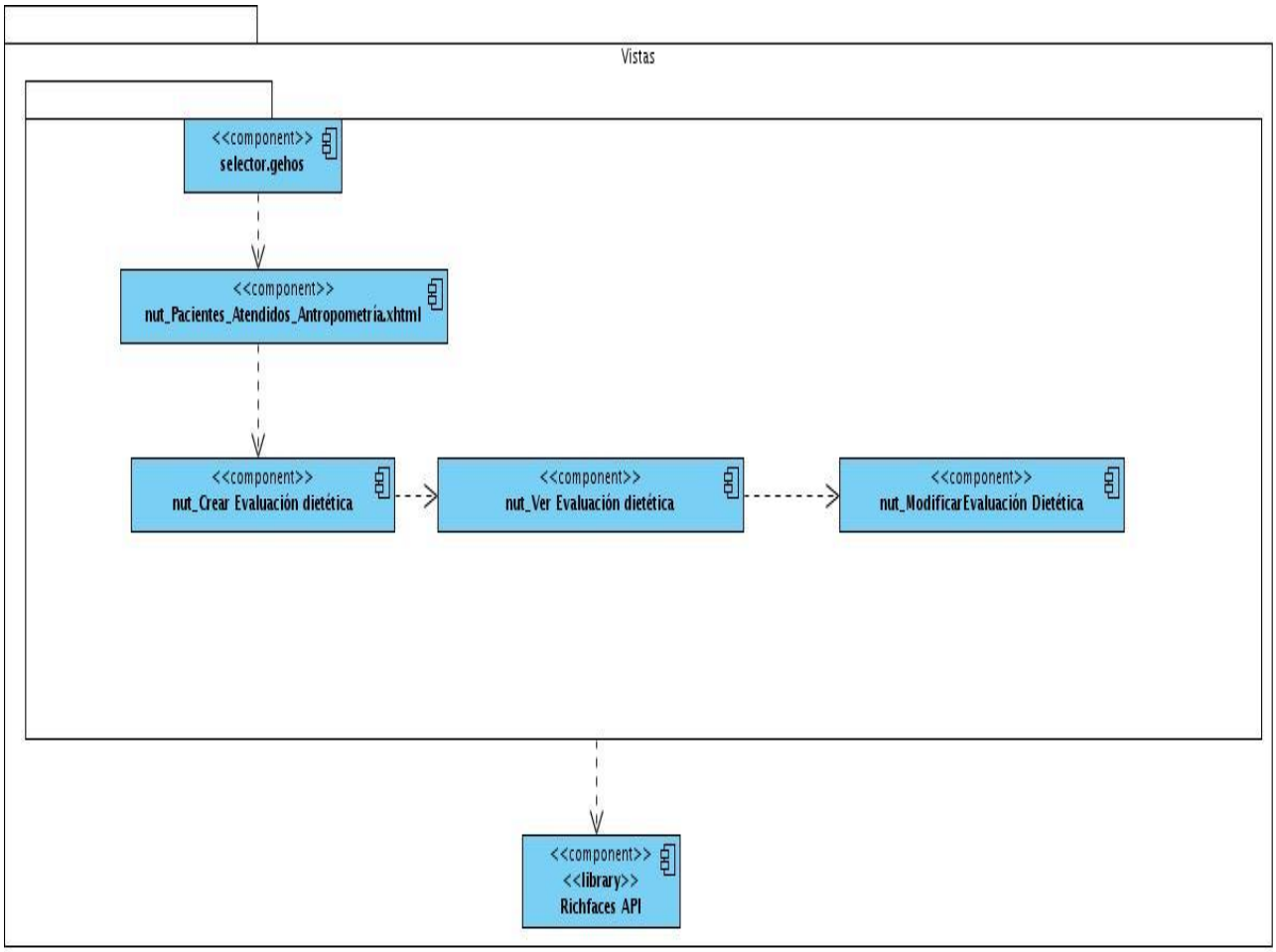


Figura 7 Diagrama de Componente Evaluación Dietética (Paquete Vistas).

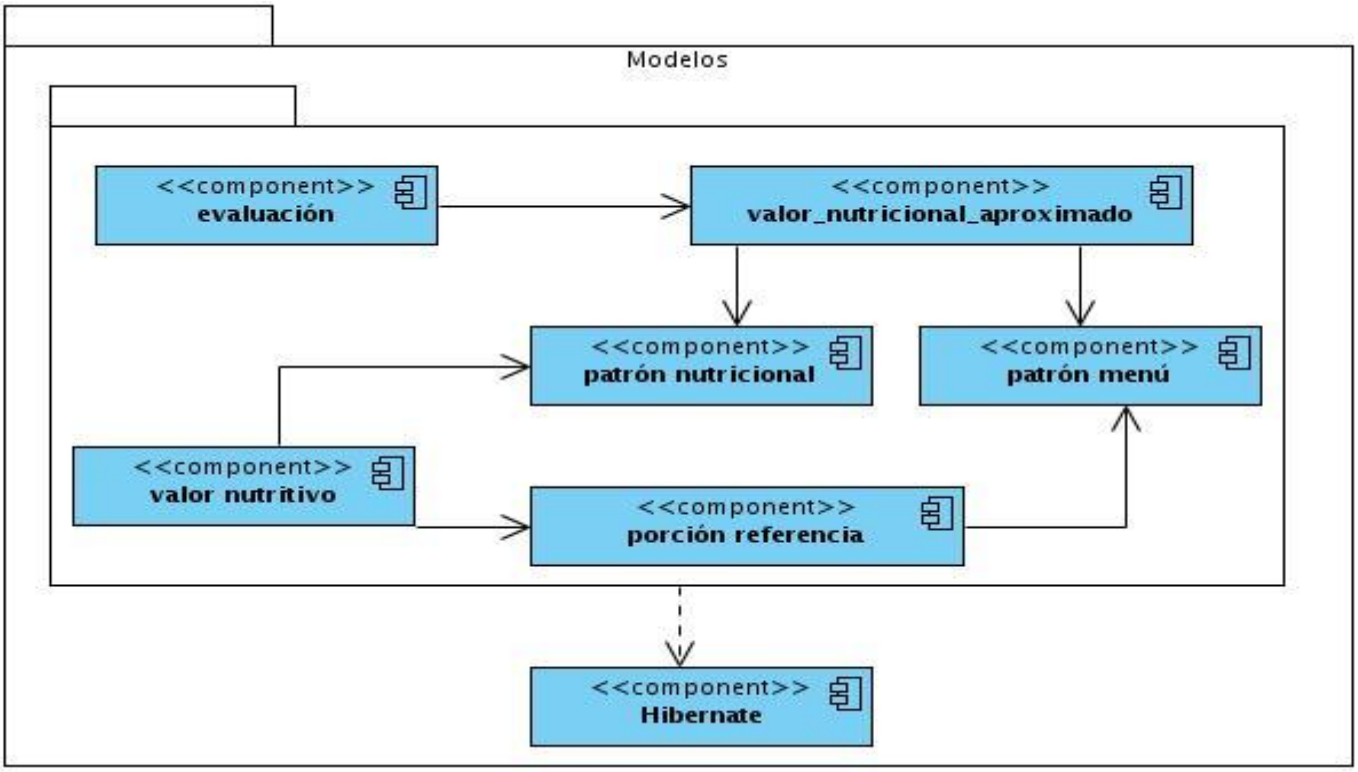


Figura 8 Diagrama de Componente Evaluación Dietética (Paquete Modelo).

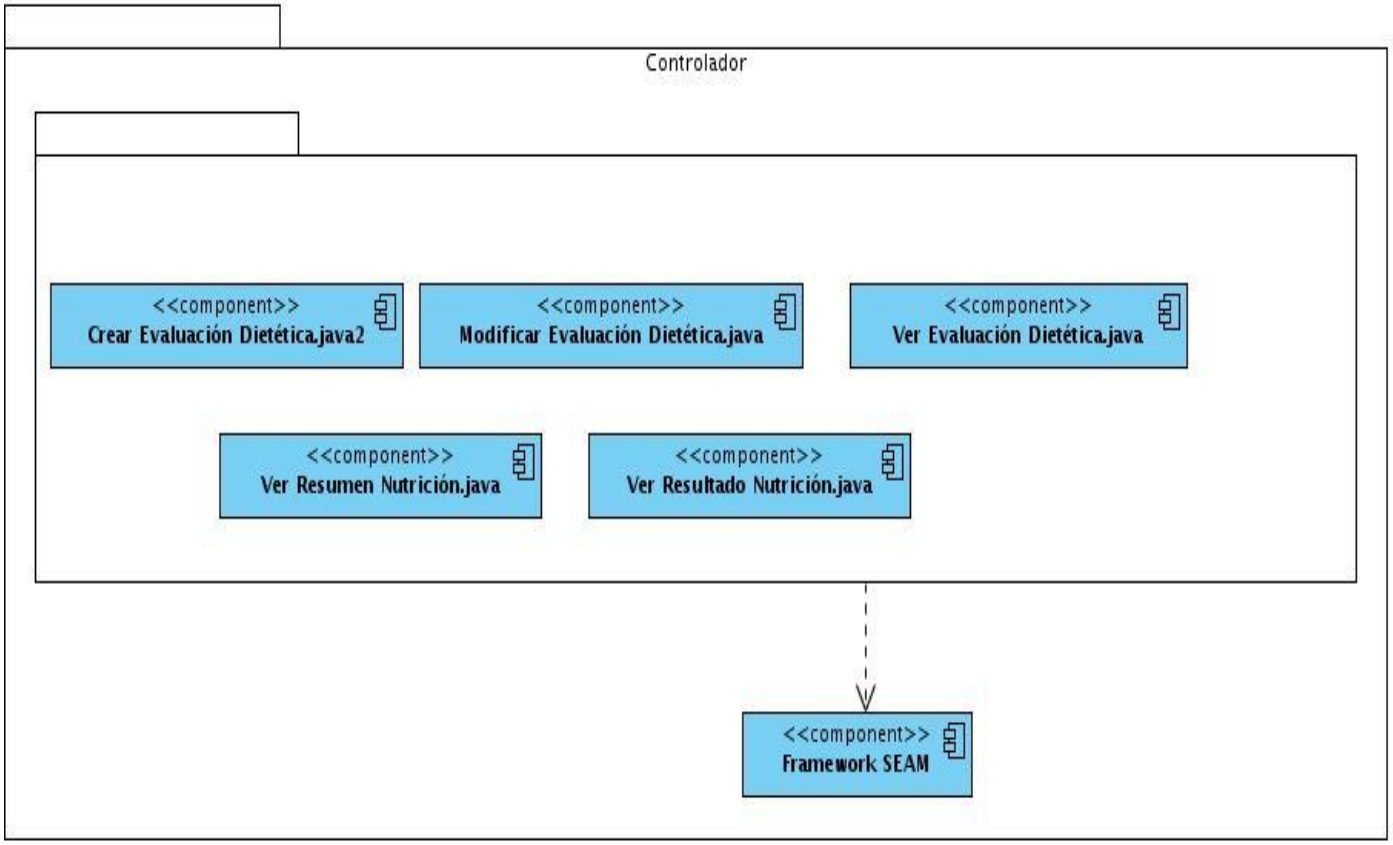


Figura 9 Diagrama de Componente Evaluación Dietética (Paquete Controlador).

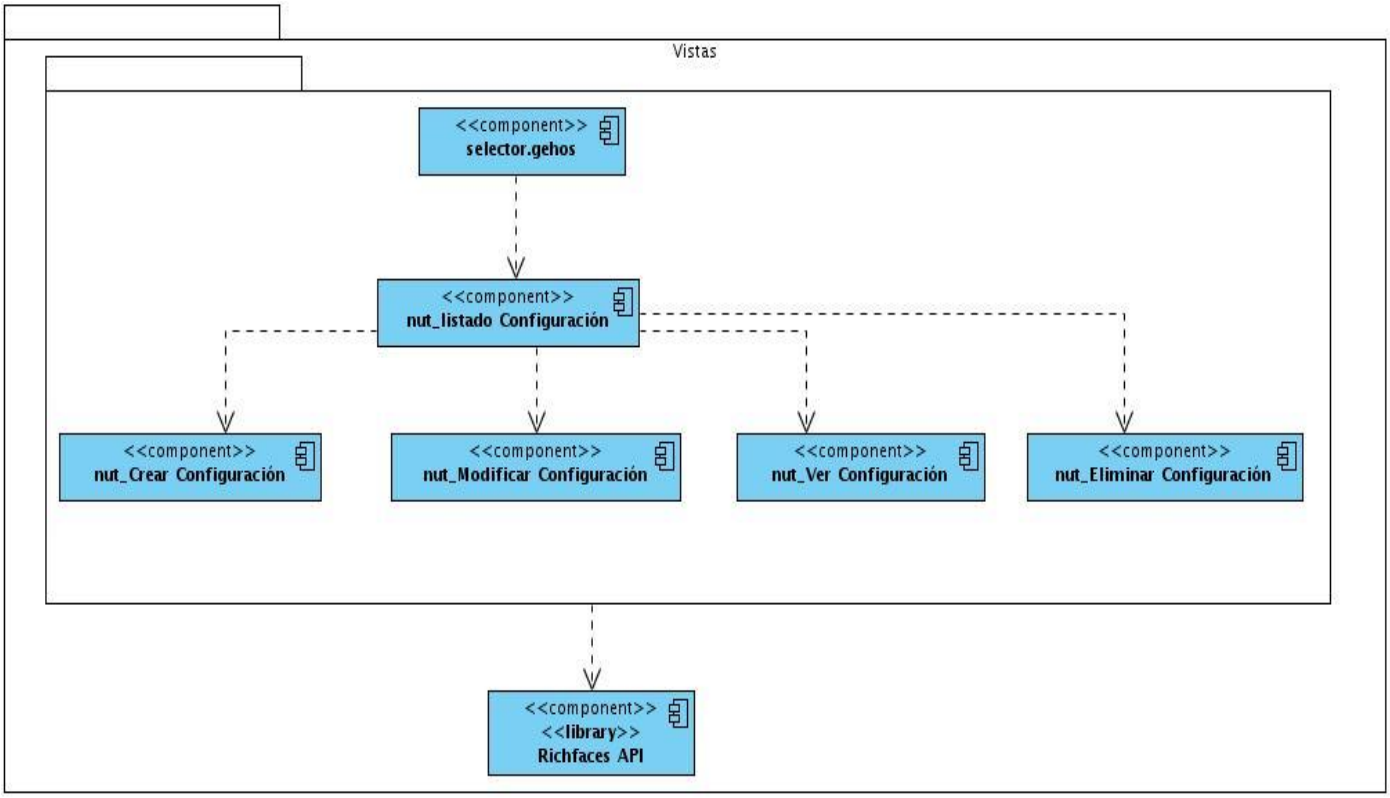


Figura 10 Diagrama de Componente Configuración (Paquete Vistas).

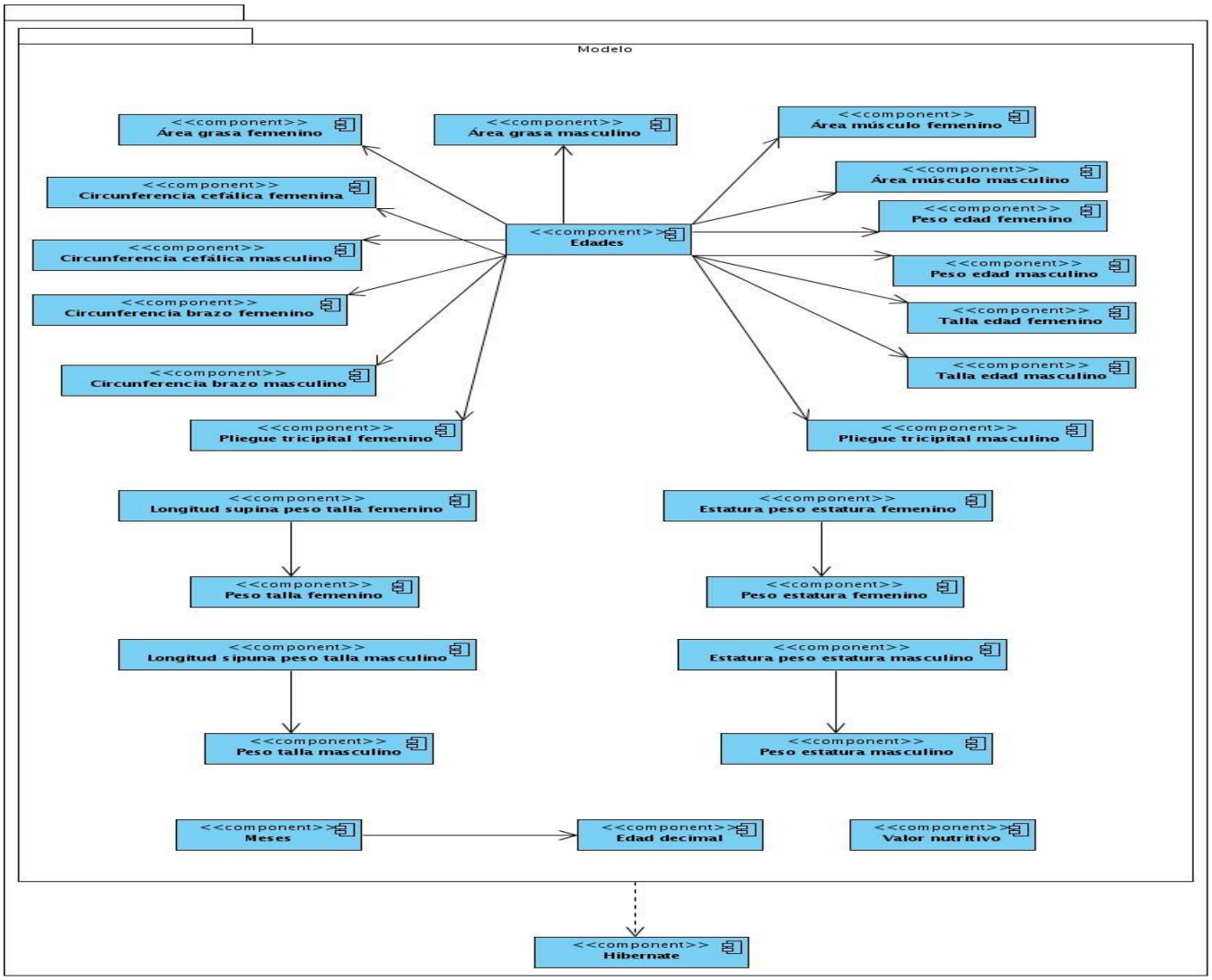


Figura 11 Diagrama de Componente Configuración (Paquete Modelo).

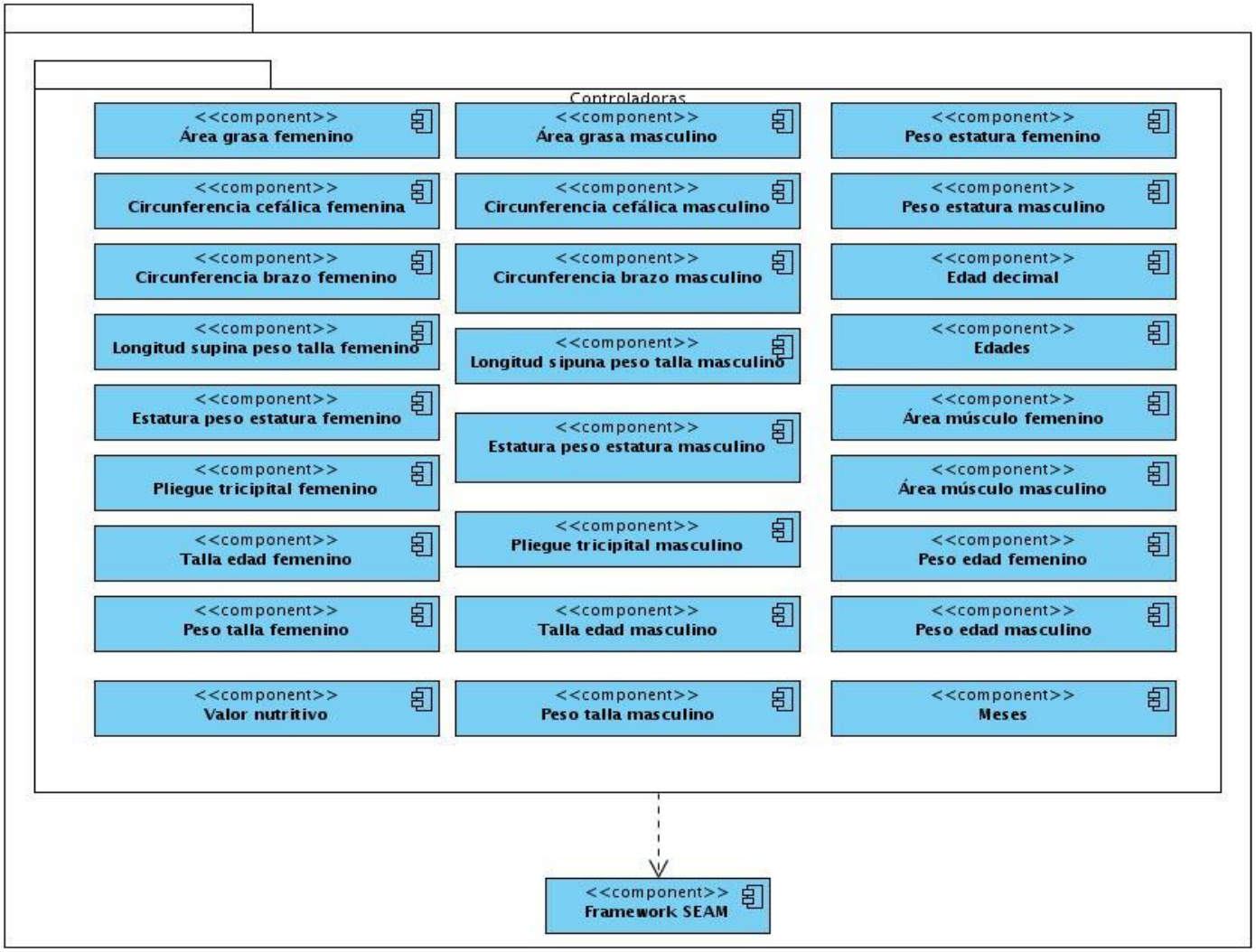


Figura 12 Diagrama de Componente Configuración (Paquete Controlador).

La valoración crítica realizada durante el estudio del diseño propuesto por el analista permitió conocer que el mismo estaba incompleto ya que se pudo definir que faltaban clases y relaciones entre las tablas, se describieron las nuevas clases u operaciones, para poder llevar a cabo la implementación. Además, se generó el modelo de datos, donde se mostró la totalidad de las tablas, así como sus atributos y relaciones, la descripción de las tablas fue de gran ayuda, ya que permitió conocer las peculiaridades de cada una de ellas, también se realizó el diagrama de componentes para conocer la distribución física del sistema.

CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA.

Para la elaboración del capítulo se establecieron los siguientes objetivos: describir las pruebas de caja negra que son las que se les realizan al sistema y las tablas que conforman un caso de prueba. Así como, definir y detallar los casos de pruebas asociados a las funcionalidades implementadas.

El flujo de trabajo Prueba brinda soporte para encontrar, documentar y solucionar defectos en el sistema, por lo debe estar presente en todo el ciclo de vida del desarrollo del sistema para ir refinándolo paulatinamente y no al final del mismo.

4.1 Descripción de las pruebas de caja negra.

Las pruebas de un software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación. La creciente percepción del software como un elemento del sistema y la importancia de los costes asociados a un fallo del mismo, motivan la creación de pruebas minuciosas y bien planificadas.

Cualquier producto de ingeniería puede ser probado conociendo la funcionalidad específica para la cual fue diseñado, se pueden llevar a cabo pruebas que demuestren que cada función es completamente operativa. Este enfoque se denomina: prueba de caja negra.

La prueba de caja negra, se centra principalmente en los requisitos funcionales del software permitiendo obtener un conjunto de condiciones de entrada que ejerciten estos requisitos y se ignora la estructura de control. No son más que las pruebas que se realizan sobre la interfaz del software.

Para desarrollar dicha prueba existen varias técnicas, entre ellas están:

- Técnica de la partición de equivalencia: esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.

- Técnica del análisis de valores límite: esta técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.

- Técnica de grafos de causa-efecto: es una técnica que permite al encargado de la prueba, validar complejos conjuntos de acciones y condiciones.

Se utilizará la técnica de la Partición de equivalencia, que divide el dominio de entrada de un programa en clases de datos, a partir de las cuales deriva los casos de prueba. Un caso de prueba, es un conjunto de entradas de pruebas donde se describen escenarios y se identifican variables. Estas últimas representan a un conjunto de estados válidos o inválidos para las condiciones de entrada para demostrar que las funciones del software son operativas.

4.2 Casos de prueba.

Un caso de prueba específica una forma de probar el sistema, incluyendo la entrada o resultado con lo que se ha de probar y las condiciones bajo las que hay que probarse. Al Módulo Nutrición, se le aplicarán pruebas de “caja negra”, ya que el producto no se liberará, y solo se realizará una iteración de pruebas y no conformidades de las mismas.

4.3 Descripción de los valores utilizados para los test.

Tabla 33 Matriz de datos de la sección Gestionar Nomenclador Valor Nutritivo.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Adicionar Nomenclador Valor Nutritivo	EC 1.1: Adicionar Nomenclador Valor nutritivo exitosamente.	Se agrega un nuevo Nomenclador Valor nutritivo con los datos correspondientes.
	EC1.2: Existen campos incompletos	No se agrega el Nomenclador Valor nutritivo porque existen campos incompletos.
	EC 1.3: Existen campos incorrectos.	No se agrega el Nomenclador Valor nutritivo porque existen campos incorrectos.
SC2: Ver datos del Nomenclador	EC 2.1: Ver datos del Nomenclador Valor nutritivo	Se visualiza los datos del Nomenclador Valor nutritivo

Valor nutritivo	exitosamente.	escogido.
	EC 2.2: Salir del Nomenclador Valor nutritivo exitosamente.	Se visualiza el listado de los nomencladores de Valor nutritivo.
SC3: Modificar Nomenclador Valor nutritivo	EC 3.1: Modificar Nomenclador Valor nutritivo exitosamente.	Se modifica un nuevo Nomenclador Valor nutritivo con los datos correspondientes.
	EC3.2: Existen campos incompletos	No se modifica el Nomenclador Valor nutritivo porque existen campos incompletos.
	EC3.3: Existen campos incorrectos.	No se modifica el Nomenclador Valor nutritivo porque existen campos incorrectos.
SC4: Eliminar Nomenclador Valor nutritivo	EC4.1: Eliminar el Nomenclador Valor nutritivo exitosamente.	Se elimina el Nomenclador Valor nutritivo exitosamente.
	EC4.2:No Eliminar Nomenclador Valor nutritivo.	No se elimina el Nomenclador Valor nutritivo.
SC5: Buscar Nomenclador Valor nutritivo	EC5.1: Busca el Nomenclador Valor nutritivo exitosamente.	Lista el nomenclador Valor nutritivo a buscar, sino encuentra por los parámetros especificados sale un mensaje informando: “No se encontró información que cumpla con los criterios de búsqueda”.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Código	Campo de texto	No	Se admiten números y letras.

2	Valor	Campo de texto	No	Se admiten números y letras.
3	Grasas	Campo de texto	No	Se admiten números.
4	Carbohidratos	Campo de texto	No	Se admiten números.
5	Grupo alimento	Campo de texto	No	Se admiten números y letras.
6	Energía	Campo de texto	No	Se admiten números.
7	Proteína	Campo de texto	No	Se admiten números.

Escenario	Variable 1 Código	Variable 2 Valor	Variable 3 Grasas	Variable 4 Carbohi-dratos	Variable 5 Grupo alimento	Variable 6 Energía	Variable 7 Proteína	Respuesta del Sistema	Resultado de la Prueba	Flujo Central
EC11: Adicionar Nomenclador Valor nutritivo exitosamente.	*V*	*V*	*V*	*V*	*V*	*V*	*V*	*El sistema adiciona un nuevo nomenclador Valor nutritivo.*		1. *Menú Nutrición.* 2. *Submenú Configuración.* 3. *Vínculo Valor nutritivo.* 4. *Vínculo Adicionar nomenclador Valor nutritivo.*
EC1.2: Existen campos incompletos	*I*	*V*	*V*	*V*	*V*	*V*	*V*	*El sistema muestra un carácter especial (asterisco rojo sobre el campo de entrada incompleto).*		5. *Menú Nutrición.* 6. *Submenú Configuración.* 7. *Vínculo Valor nutritivo.* 8. *Vínculo Adicionar nomenclador Valor nutritivo.*
	V	*I*	*V*	*V*	*V*	*V*	*V*			

	V	*V*	*I*	*V*	*V*	*V*	*V*			
	V	*V*	*V*	*I*	*V*	*V*	*V*			
	V	*V*	*V*	*V*	*I*	*V*	*V*			
	V	*V*	*V*	*V*	*V*	*I*	*V*			
	V	*V*	*V*	*V*	*V*	*V*	*I*			
EC1.3: Existen campos incorrectos.	*I*	*V*	*V*	*V*	*V*	*V*	*V*	*El sistema muestra un carácter especial (asterisco rojo sobre el campo de entrada incorrecto).*		9. *Menú Nutrición.* 10. *Submenú Configuración.* 11. *Vínculo Valor nutritivo.* 12. *Vínculo Adicionar nomenclador Valor nutritivo.*
	V	*I*	*V*	*V*	*V*	*V*	*V*			
	V	*V*	*I*	*V*	*V*	*V*	*V*			

	V	*V*	*V*	*I*	*V*	*V*	*V*			
	V	*V*	*V*	*V*	*I*	*V*	*V*			
	V	*V*	*V*	*V*	*V*	*I*	*V*			
	V	*V*	*V*	*V*	*V*	*V*	*I*			

4.4 Evaluación de la ejecución del test y de los resultados obtenidos.

El probador debe verificar el correcto funcionamiento del sistema de acuerdo a las cuantificaciones establecidas en la fase de implementación, detectando y documentando los fallos que presente la misma. El proceso de prueba se lleva a cabo en diferentes iteraciones, cada iteración, muestra la revisión del sistema y la corrección de los fallos detectados. Los resultados obtenidos con la ejecución de las pruebas, al realizar solamente una iteración de este proceso se registraron en los casos de pruebas, las mismas fueron realizadas por las analistas Yeney de la Caridad Sánchez Rodríguez y Rosalia Ación Valiente. (Ver Tabla 34)

Tabla 34 resultados obtenidos con la realización de las pruebas.

No	Funcionalidades	NC	Recomendaciones	Porciento de NC resueltas
1	Crear evaluación dietética	4	1	100%
2	Modificar evaluación dietética	0	0	0
3	Ver evaluación dietética	0	0	0
4	Gestionar valor nutritivo	1	0	100%
5	Gestionar edad decimal	0	0	0
6	Gestionar meses	0	0	0
7	Gestionar longitud supina peso talla femenino	0	0	0

8	Gestionar longitud supina peso talla masculino	0	0	0
9	Gestionar peso talla femenino	2	0	100%
10	Gestionar peso talla masculino	2	0	100%
11	Gestionar estatura peso estatura femenino	0	0	0
12	Gestionar estatura peso estatura masculino	0	0	0
13	Peso estatura femenino	1	0	100%
14	Peso estatura masculino	1	0	100%
15	Buscar Resultado Consulta Nutrición	0	0	0
16	Ver Resultado Consulta Nutrición	0	0	0
17	Gestionar Edades	1	0	100%
18	Gestionar Área grasa femenino	1	0	100%
19	Gestionar Área grasa masculino	1	1	100%
20	Gestionar Área músculo femenino	2	0	100%
21	Gestionar Área músculo masculino	2	0	100%

22	Gestionar Circunferencia brazo femenino	1	0	100%
23	Gestionar Circunferencia brazo masculino	3	0	100%
24	Gestionar Circunferencia cefálica femenino	2	0	100%
25	Gestionar Circunferencia cefálica masculino	0	0	0
26	Gestionar Pliegue tricipital femenino	2	0	100%
27	Gestionar Pliegue tricipital masculino	2	0	100%
28	Gestionar Peso edad femenino	0	0	0
29	Gestionar Peso edad masculino	0	0	0
30	Gestionar Talla edad femenino	2	0	100%
31	Gestionar Talla edad masculino	2	0	100%

La realización de las pruebas de caja negra permitió verificar y revelar la calidad del sistema. Se diseñaron 31 casos de pruebas; los cuales fueron ejecutados en una iteración, detectándose un total de 32 no conformidades, las cuales fueron solucionadas en período de cinco días.

En este capítulo se realizó las pruebas al sistema utilizando el método de caja negra, obteniéndose los casos de prueba como artefactos generados del flujo de trabajo.

CONCLUSIONES

Una vez finalizada procesos asociados a la evaluación dietética, resultados de consulta y la configuración del Módulo Nutrición se arribaron a las siguientes conclusiones:

- El estudio del Proceso Unificado de Desarrollo de Software, posibilitó que se pudiera dar continuidad al diseño propuesto por el analista, comprobándose que el uso de esta metodología es eficiente para guiar el proceso de desarrollo de un software.
- La descripción de la arquitectura propuesta para el desarrollo del sistema, evidenció que la utilización del patrón Modelo Vista Controlador mejora la realización de las aplicaciones, ya que al separar los datos, la interfaz de usuario y la lógica de negocio, posibilita que las modificaciones se realicen solo en nivel requerido, impactando en menor medida en el resto de las capas.
- Al realizar un estudio del diseño propuesto por el analista, se verificó que el mismo se encontraba incompleto, ya que solo contemplaba una parte de los procesos que debían ser informatizados, por lo que se hizo necesario realizar un refinamiento del mismo.
- Se realizó la implementación de los procesos de evaluación dietética; realizar resumen de consulta y configuración, obteniéndose un módulo más adaptable, lo cual permite agilizar las pruebas que se realizan a los niños en la consulta.

RECOMENDACIONES

Con el objetivo de completar y mejorar la solución propuesta, además de ofrecer un mejor servicio a usuarios y pacientes, se recomienda:

- Implementar las salidas del sistema para conocer datos estadísticos, como:
  - Cantidad de pacientes bajo peso.
  - Cantidad de pacientes obesos.
  - Cantidad de pacientes atendidos por grupos de edades.
- Proponer al responsable del Programa Renacer Contigo, desplegar el Sistema de Evaluación del Neurodesarrollo en Niños en todos los hospitales pediátricos de Cuba.

REFERENCIAS BIBLIOGRÁFICAS

1. Ministerio de la Informática y las Comunicaciones. [En línea] [Citado el: 17 de diciembre de 2010.] http://www.mic.gov.cu/sitiomic/servlet/hinfosoc.

2. Delgado Ramos, Ariel y Vidal Ledo, Maria. *Informática en la salud cubana.* 2006.

3. Castro Ruz, Fidel. Reunión del 19/8/2002. Tropa de futuro. Oficina de publicaciones del Consejo de Estado. La Habana 2003. Pág. 25-36.

4. Glosario.Net. [En línea] [Citado el: 12 de diciembre de 2010.] http://salud.glosario.net/alimentacion-nutricion/nutrici%25F3n-2283.html .

5. Medicina de Rehabilitación de Atención Temprana. [En línea] [Citado el: 12 de noviembre de 2010.] http://www.sld.cu/sitios/rehabilitacion-temprana/temas.php?idl=131&idv=16582.

6. La Atención Temprana. [En línea] [Citado el: 12 de noviembre de 2010.] .

7. Scribd.com. [En línea] [Citado el: 12 de diciembre de 2010.] http://www.scribd.com/doc/22630811/ESTIMULACION-TEMPRANA.

8. Alimentación y Salud. [En línea] [Citado el: 14 de diciembre de 2010.]

9. [En línea] [Cited:marzo19,2011.] www.dsic.upv.es/asignaturas/facultad/.../IntroduccionProcesoSW.doc.

10. 1. Definición ABC. [En línea] 9 de diciembre de 2009. [Citado el: 18 de marzo de 2011.] http://www.definicionabc.com/ciencia/metodologia.php .

11. **Mestre Villavicencio, Pedro, et al.** *Proyecto Renacer Contigo.* Habana : s.n., 2005.

12. Scribd.com.[En línea] marzo 2007. [Cited: enero 18, 2011.] http://es.scribd.com/doc/12983228/Fases-en-RUP.

13. IDEM Scribd.com.[En línea] marzo 2007. [Cited: enero 18, 2011.] http://es.scribd.com/doc/12983228/Fases-en-RUP.

14. eva.uci.cu. [En línea] [Citado el: 18 de marzo de 2011.] http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_8/Conferencia_6/Materiales_Basicos/Implementacion.pdf.

15. **Ivar Jacobson, Grady Booch y James Rumbaugh.** El proceso Unificado de Desarrollo de Software. Página 281.

16. IDEM El proceso Unificado de Desarrollo de Software. Página 282.

17. **UCI.** Entorno Visual de Aprendizaje.Pagina 1. [En línea] [Citado el: 12 de enero de 2010.] http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_9/Conferencia_7/Materiales_Basicos/Sobre_la_disciplina_de_Prueba.pdf.

18. **Oré B, Ing. Alexander.** calidadysoftware.com.[En línea] [Cited: enero 18, 2011.] http://www.calidadysoftware.com/testing/pruebas_unitarias1.php .

19. **Mora, Francisco.** Lenguaje Unificado de Modelado. [En línea] [Citado el: 12 de diciembre de 2010.] http://www.dccia.ua.es/dccia/inf/asignaturas/GPS/archivos/Uml.PDF.

20. Soluciones de Alta Calidad de tecnologias de Información. [En línea] [Citado el: 10 de enero de 2011.] www.sacti.com.mx.

21. NeuroNet. [En línea] [Citado el: 10 de enero de 2011.] www.neuronet.cl/downloads/J2EE.pdf.

22. **Almirón, González, Cristóbal.** Adictos al trabajo. [En línea] [Citado el: 12 de enero de 2011.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=IntroduccionJSFJava.

23. **Hernández, Alcantar, Fernando.** GestioPolis. [En línea] 6 de 6 de 2008. [Citado el: 12 de enero de 2011.] http://www.gestiopolis.com/administracion-estrategia/hibernate-para-bases-de-datos-con-xml.htm.

24. **Suárez, Sánchez, Jose Manuel.** Adictos al trabajo. [En línea] 1 de 2 de 2010. [Citado el: 10 de enero de 2011.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro.

25. Scribd.com. [En línea] [Citado el: 12 de diciembre de 2010.] http://www.scribd.com/doc/19026497/JBOSS.

26. Scribd.com. [En línea] [Citado el: 12 de enero de 2011.] http://www.scribd.com/doc/36570462/postgreSQL-investigacion.

27. Ubuntu. [En línea] [Citado el: 10 de diciembre de 2010.] http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.

28. Java. [En línea] [Citado el: 12 de enero de 2011.] http://www.java.com/es/download/help/why_upgrade.xml.

29. Mitecnologico. [En línea] [Citado el: 10 de Enero de 2010.] http://www.mitecnologico.com/Main/LaNormaIsoIec9126.

30. http://definicion.de/modelo-de-datos/.

31. Glosario.net. [En línea] [Citado el: 10 de Febrero de 2011.]

BIBLIOGRAFÍA

1. **Almirón, Cristóbal González.** Adictos al trabajo. [En línea] [Citado el: 12 de enero de 2011.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=IntroduccionJSFJava.

2. Alimentación y Salud. [En línea] [Citado el: 14 de diciembre de 2010.]

3. Ayuda del RUP. Suite del Rational.

4. **Alexander,** P. Aptitudes físicas; características morfológicas y composición corporal. Pruebas estandarizadas en Venezuela / P. Alexander. Caracas: Editorial Depoaction, 1995.

5. **Oré B, Ing. Alexander.** calidadysoftware.com.[En línea] [Cited: enero 18, 2011.] http://www.calidadysoftware.com/testing/pruebas_unitarias1.php .

6. **Booch, G.: Rumbaugh, J. y Jacobson, I.;** "El Lenguaje Unificado de Modelado". 2000. Addison-Wesley. Capítulo 11 Página 281-302 Capítulo 16 Página 381-394 Páginas 339-340, 361-363, 377-378

7. Buen Master.com. [En línea] [Citado el: 2 de Febrero de 2010.] http://www.buenmaster.com/?a=536.

8. **Bunn, John**. Entrenamiento deportivo científico / John. Bunn. México: Editorial Pax México, 1987.

9. **Castro Ruz, Fidel**. Reunión del 19/8/2002. Tropa de futuro. Oficina de publicaciones del Consejo de Estado. La Habana 2003. Pág. 25-36.

10. Carter, J.L. The henth - cárter somatotype method / J.L. Cárter. San Diego StateUniversity : Silla bus service, 1980.

11. **Delgado Ramos, Ariel y Vidal Ledo, Maria**. *Informática en la salud cubana.* 2006.

12. Definición.de. [En línea] [Citado el: 10 de Febrero de 2011.] http://definicion.de/modelo-de-datos/.

13. Dick, Frank. Principio del entrenamiento deportivo / Frank. Dick. Moscú: Editorial Raduga, 1993.

14. **Hernández Alcantar, Fernando.** GestioPolis. [En línea] 6 de 6 de 2008. [Citado el: 12 de enero de 2011.] http://www.gestiopolis.com/administracion-estrategia/hibernate-para-bases-de-datos-con-xml.htm.

15. Forteza de la Rosa, A. Bases metodológicas del entrenamiento / A. Forteza de la Rosa, A. Ransola Rivas. Ciudad de la Habana : Editorial Científico Técnica, 1989.

16. Forteza de la Rosa, A. Alta metodología, carga y estructuración del entrenamiento deportivo / A. Forteza de la Rosa. Ciudad de la Habana: Editorial Pueblo y Educación, 1998.

17. Glosario.Net. [En línea] [Citado el: 12 de diciembre de 2010.] http://salud.glosario.net/alimentacion-nutricion/nutrici%25F3n-2283.html .

18. Glosario.net. [En línea] [Citado el: 10 de Febrero de 2011.] http://tecnologia.glosario.net/terminos-tecnicos-internet/fiabilidad-681.html.

19. **Ivar Jacobson, Grady Booch y James Rumbaugh.** El proceso Unificado de Desarrollo de Software. Página 281.

20. Java. [En línea] [Citado el: 12 de enero de 2011.] http://www.java.com/es/download/help/why_upgrade.xml.

21. **Jacobson, I.; Booch, G. y Rumbaugh, J**.; "El Proceso Unificado de Desarrollo de software". 2000. Addison-Wesley.

22. La Atención Temprana. [En línea] [Citado el: 12 de noviembre de 2010.] http://www.invanep.com/descargas/documentos/atencion/atencion.temprana.mulas.milla.pdf.

23. Ministerio de la Informática y las Comunicaciones. [En línea] [Citado el: 17 de diciembre de 2010.] http://www.mic.gov.cu/sitiomic/servlet/hinfosoc.

24. Medicina de Rehabilitación de Atención Temprana. [En línea] [Citado el: 12 de noviembre de 2010.] http://www.sld.cu/sitios/rehabilitacion-temprana/temas.php?idl=131&idv=16582.

25. **Mestre Villavicencio, Pedro, et al.** *Proyecto Renacer Contigo.* Habana : s.n., 2005.

26. **Martínez Martínez, Alejandro y Raúl**. Guía a Rational Unified Process.http://www.info-ab.uclm.es/asignaturas/42551/trabajosAnteriores/Trabajo-Guia%20RUP.pdf.

27. **Mora, Francisco.** Lenguaje Unificado de Modelado. [En línea] [Citado el: 12 de diciembre de 2010.] http://www.dccia.ua.es/dccia/inf/asignaturas/GPS/archivos/Uml.PDF.

28. Mitecnologico. [En línea] [Citado el: 10 de Enero de 2010.] http://www.mitecnologico.com/Main/LaNormaIsoIec9126.

29. NeuroNet. [En línea] [Citado el: 10 de enero de 2011.] www.neuronet.cl/downloads/J2EE.pdf.

30. **Pressman, Roger;** Ingeniería de software. Un enfoque práctico. 2002. McGraw.Hill/Interamericana de España.

31. Scribd.com. [En línea] [Citado el: 12 de diciembre de 2010.] http://www.scribd.com/doc/22630811/ESTIMULACION-TEMPRANA.

32. Scribd.com. [Online] marzo 2007. [Cited: enero 18, 2011.] http://es.scribd.com/doc/12983228/Fases-en-RUP.

33. Soluciones de Alta Calidad de tecnologias de Información. [En línea] [Citado el: 10 de enero de 2011.] www.sacti.com.mx.

34. **Suárez Sánchez, Jose Manuel.** Adictos al trabajo. [En línea] 1 de 2 de 2010. [Citado el: 10 de enero de 2011.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro.

35. Scribd.com. [En línea] [Citado el: 12 de diciembre de 2010.] http://www.scribd.com/doc/19026497/JBOSS.

36. Scribd.com. [En línea] [Citado el: 12 de enero de 2011.] http://www.scribd.com/doc/36570462/postgreSQL-investigacion.

37. Ubuntu. [En línea] [Citado el: 10 de diciembre de 2010.] http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.

38. **UCI.** Entorno Visual de Aprendizaje.Pagina 1. [En línea] [Citado el: 12 de enero de 2010.] http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_9/Conferencia_7/Materiales_Basicos/Sobre_la_disciplina_de_Prueba.pdf.

39. Modelo de Implementación http://www.info-ab.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

GLOSARIO DE TÉRMINOS

Área músculo: Indicador para evaluar el estado nutricional, ya que constituyen un método rápido, económico y no invasivo para medir la composición corporal.

Área grasa: Son compuestos orgánicos que se componen de carbono, hidrógeno y oxígeno, y son la fuente de energía en los alimentos.

Circunferencia cefálica: Es el perímetro de la cabeza. Hay dos variantes: Circunferencia horizontal de la cabeza (Martín,1928; Martín-Saller, 1959; Pospisil, 1965) y circunferencia de la cabeza (Weiner y Lourie, 1969).

Circunferencia brazo: Es la circunferencia tomada en la mitad del brazo, entre el acromion y olécranon.

Estatura: La estatura es la distancia directa entre vértex y el plano de apoyo del individuo.

Framework: En el desarrollo de software, un framework es una estructura de soporte definida en la cual otro proyecto de software, puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

MVC: *(Modelo Vista Controlador).* Es un patrón de arquitectura de software, compuesto de tres componentes distintos: datos, interfaz de usuario, y lógica del negocio.

ORM *(Object-Relational Mapping):* El mapeo de objetos-relacional, es una técnica de programación utilizada para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos, creando una base de datos orientada a objetos virtuales, por encima de la base de datos relacional.

Proceso: Conjunto de actividades interrelacionadas entre sí, que a partir de una o varias entradas de materiales o información, dan lugar a una o varias salidas, también de materiales, o información con valor añadido.

Pruebas: Es una actividad en la cual un sistema o componente, es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados y registrados y a su vez se realiza una evaluación de algún aspecto del sistema o componente. 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.

Pruebas unitarias: Es el proceso de hacer pruebas sobre los componentes individuales (subprogramas o procedimientos) de un programa. El propósito es encontrar discrepancias entre la especificación de la interfaz del módulo y su comportamiento real.

Patrón de alimentación: es una guía en la cual se le indica al paciente las porciones por cada grupo de alimento puede consumir en un día y tiempo de comida.

Peso: El peso es la acción de la gravedad sobre la masa corporal.

SENDN: Sistema de Evaluación del Neurodesarrollo en Niños.

UCIN: Unidades de Cuidados Intensivos Neonatales.

UCIP: Unidades de Cuidados Intensivos Polivalentes.

ANEXOS

Descripción de las nuevas tablas u operaciones.

Anexo 1

Nombre de la Clase: Longitud Supina Peso Talla Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
longitud1	Float
longitud2	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Longitud Supina Peso Talla Masculino	
Tipo de la Clase: Controlador	
Nombre:	**insertarLongSupinaPesoTallaMasc()**
Descripción:	Inserta los datos longitud supina peso para la talla masculino.
Nombre:	**modificarLongSupinaPesoTallaMasc()**

Descripción:	Modifica los datos de longitud supina peso talla masculino.
Nombre:	**eliminarLongSupinaPesoTallaMasc()**
Descripción:	Elimina los datos.

Nombre de la Clase: Longitud Supina Peso Talla Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para Crear la longitud supina peso para la talla masculino.

Nombre de la Clase: Meses	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
horario	String
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Meses	
Tipo de la Clase: Controlador	
Nombre:	**insertarMeses()**
Descripción:	Inserta los datos de meses.
Nombre:	**modificarMeses()**
Descripción:	Modifica los datos de meses.
Nombre:	**eliminarMeses()**
Descripción:	Elimina los meses.

Nombre de la Clase: Meses	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear meses.

Nombre de la Clase: Edad Decimal	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
ModnNMeses	modnNMeses
dia	Integer

eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Edad Decimal	
Tipo de Clase: Controlador	
Nombre:	**insertarEdadDecimal()**
Descripción:	Inserta los datos de edad decimal.
Nombre:	**modificarEdadDecimal()**
Descripción:	Modifica los datos de edad decimal.
Nombre:	**eliminarEdadDecimal()**
Descripción:	Elimina los datos de edad decimal.

Nombre de la Clase: Edad Decimal	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la edad decimal.

Nombre de la Clase: Estatura Peso Estatura Femenino
Tipo de Clase: Modelo

Atributo	Tipo
Id	Serial
version	Integer
estatura1	Float
estatura2	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Estatura Peso Estatura Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarEstaturaPesoEstaturaFem()**
Descripción:	Inserta los datos de estatura peso estatura femenino.
Nombre:	**modificarEstaturaPesoEstaturaFem()**
Descripción:	Modifica los datos de estatura peso estatura femenino.
Nombre:	**eliminarEstaturaPesoEstaturaFem()**
Descripción:	Elimina los datos de estatura peso estatura femenino.

Nombre de la Clase: Estatura Peso Estatura Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la estatura peso estatura femenino.

Nombre de la Clase: Estatura Peso Estatura Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
estatura1	Float
estatura2	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Estatura Peso Estatura Masculino	
Tipo de la Clase: Controlador	
Nombre:	**insertarEstaturaPesoEstaturaMasc()**

Descripción:	Inserta la estatura peso estatura masculino.
Nombre:	**modificarEstaturaPesoEstaturaMasc()**
Descripción:	Modifica la estatura peso estatura masculino.
Nombre:	**eliminarEstaturaPesoEstaturaMasc()**
Descripción:	Elimina la estatura peso estatura masculino.

Nombre de la Clase: Estatura Peso Estatura Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la estatura peso estatura masculino.

Nombre de la Clase: Peso Estatura Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
ModnNEstaturaPesoEstaturaFem	modnNEstaturaPesoEstaturaFem
peso	Float
percentil	Integer
eliminado	Boolean

cid	Integer
valor	String
codigo	String

Nombre de la Clase: Peso Estatura Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarModnNPesoEstaturaFem()**
Descripción:	Inserta el peso para la estatura femenino.
Nombre:	**modificarPesoEstaturaFem()**
Descripción:	Modifica el peso para la estatura femenino.
Nombre:	**eliminarPesoEstaturaFem()**
Descripción:	Elimina el peso para la estatura femenino.

Nombre de la Clase: Peso Estatura Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el peso estatura femenino.

Nombre de la Clase: Peso Estatura Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**

Id	Serial
version	Integer
ModnNEstaturaPesoEstaturaMasc	modnNEstaturaPesoEstaturaMasc
peso	Float
percentil	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Peso Estatura Masculino	
Tipo de la Clase: Controlador	
Nombre:	**insertarModnNPesoEstaturaMasc()**
Descripción:	Inserta el peso para la estatura masculino.
Nombre:	**modificarPesoEstaturaMasc()**
Descripción:	Modifica el peso para la estatura masculino.
Nombre:	**eliminarPesoEstaturaMasc()**
Descripción:	Elimina el peso para la estatura masculino.

Nombre de la Clase: Peso Estatura Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el peso para la estatura masculino.

Nombre de la Clase: Peso Talla Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
ModnNLongitudSupinaPesoTallaMasc	modnNLongitudSupinaPesoTallaMasc
peso	Float
percentil	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Peso Talla Masculino
Tipo de la Clase: Controlador

Nombre:	**insertarPesoTallaMasc()**
Descripción:	Inserta peso talla masculino.
Nombre:	**modificarPesoTallaMasc()**
Descripción:	Modifica el peso talla masculino.
Nombre:	**eliminarPesoTallaMasc()**
Descripción:	Elimina el peso talla masculino.

Nombre de la Clase: Peso Talla Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el peso talla masculino.

Nombre de la Clase: Edades	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
edad	Float
eliminado	Boolean
cid	Integer

valor	String
codigo	String

Nombre de la Clase: Edades	
Tipo de Clase: Controlador	
Nombre:	**insertarEdades()**
Descripción:	Inserta los datos de la edad del paciente.
Nombre:	**modificarNomencladorEdades()**
Descripción:	Modifica los datos de la edad del paciente.
Nombre:	**eliminarEdades()**
Descripción:	Elimina los datos de la edad del paciente.

Nombre de la Clase: Edades	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear las edades.

Nombre de la Clase: Área Grasa Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**

Id	Serial
version	Integer
area_grasa	Float
percentil	Integer
modn_edades	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Área Grasa Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarAreaGrasaFem()**
Descripción:	Inserta área grasa femenina.
Nombre:	**modificarAreaGrasaFem()**
Descripción:	Modifica los datos de área grasa femenina.
Nombre:	**eliminarAreaGrasaFem()**
Descripción:	Elimina los datos de área grasa femenina.

Nombre de la Clase: Área Grasa Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el área grasa femenina.

Nombre de la Clase: Área Grasa Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
modn_edades	Integer
percentil	Integer
area_grasa	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Área Grasa Masculino	
Tipo de Clase: Controlador	
Nombre:	**insertarAreaGrasaMasc()**
Descripción:	Inserta los datos de área grasa masculina.
Nombre:	**modificarAreaGrasaMasc()**
Descripción:	Modifica los datos de área grasa masculina.
Nombre:	**eliminarAreaGrasaMasc()**
Descripción:	Elimina los datos de área grasa masculina.

Nombre de la Clase: Área Grasa Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el área grasa masculina.

Nombre de la Clase: Área Músculo Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial

version	Integer
modn_edades	Integer
area_musculo	Float
percentil	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Área Músculo Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarAreaMusculoFem()**
Descripción:	Inserta los datos de área músculo femenino.
Nombre:	**modificarAreaMusculoFem()**
Descripción:	Modifica los datos de área músculo femenino.
Nombre:	**eliminarAreaMusculoFem()**
Descripción:	Elimina los datos de área músculo femenino.

Nombre de la Clase: Área Músculo Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el área músculo femenino.

Nombre de la Clase: Área Músculo Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
modn_edades	Integer
area_musculo	Float
percentil	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Área Músculo Masculino	
Tipo de la Clase: Controlador	
Nombre:	**insertarAreaMusculoMasc()**
Descripción:	Inserta los datos de área músculo masculino.
Nombre:	**modificarAreaMusculoMasc()**
Descripción:	Modifica los datos de área músculo masculino.
Nombre:	**eliminarAreaMusculoMasc()**
Descripción:	Elimina los datos de área músculo masculino.

Nombre de la Clase: Área Músculo Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el área músculo masculino.

Nombre de la Clase: Circunferencia cefálica Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer

modn_edades	Integer
percentil	Integer
circunferencia	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Circunferencia cefálica Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarCircunferenciaCefalicaFem()**
Descripción:	Inserta los datos de la circunferencia cefálica femenina.
Nombre:	**modificarCircunferenciaCefalicaFem ()**
Descripción:	Modifica los datos de la circunferencia cefálica femenina.
Nombre:	**eliminarCircunferenciaCefalicaFem ()**
Descripción:	Elimina los datos de la circunferencia cefálica femenina.

Nombre de la Clase: Circunferencia cefálica Femenino
Tipo de Clase: Vista

Descripción:	Se muestran los campos que debe llenar para crear la circunferencia cefálica femenina.

Nombre de la Clase: Circunferencia cefálica Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
modn_edades	Integer
percentil	Integer
circunferencia	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Circunferencia cefálica Masculino	
Tipo de la Clase: Controlador	
Nombre:	**insertarCircunferenciaCefalicaMasc()**
Descripción:	Inserta los datos de la circunferencia cefálica masculina.

Nombre:	**modificarCircunferenciaCefalicaMasc()**
Descripción:	Modifica los datos de la circunferencia cefálica
Nombre:	**eliminarCircunferenciaCefalicaMasc()**
Descripción:	Elimina los datos de la circunferencia cefálica masculina.

Nombre de la Clase: Circunferencia cefálica Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la circunferencia cefálica masculina.

Nombre de la Clase: Circunferencia Brazo Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
modn_edades	Integer
percentil	Integer
circunferencia	Float
eliminado	Boolean
cid	Integer

valor	String
codigo	String

Nombre de la Clase: Circunferencia Brazo Femenino	
Tipo de Clase: Controlador	
Nombre:	**insertarCircunferenciaBrazoFem()**
Descripción:	Inserta los datos de la circunferencia del brazo femenino.
Nombre:	**modificarCircunferenciaBrazoFem()**
Descripción:	Modifica los datos de la circunferencia del brazo femenino.
Nombre:	**eliminarCircunferenciaBrazoFem()**
Descripción:	Elimina los datos de la circunferencia del brazo femenino.

Nombre de la Clase: Circunferencia Brazo Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la circunferencia del brazo femenino.

Nombre de la Clase: Circunferencia Brazo Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**

Id	Serial
version	Integer
modn_edades	Integer
percentil	Integer
circunferencia	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Circunferencia Brazo Masculino	
Tipo de la Clase: Controlador	
Nombre:	**insertarCircunferenciaBrazoMasc()**
Descripción:	Inserta los datos de la circunferencia del brazo masculino.
Nombre:	**modificarCircunferenciaBrazoMasc()**
Descripción:	Modifica los datos de la circunferencia del brazo masculino.
Nombre:	**eliminarCircunferenciaBrazoMasc()**
Descripción:	Elimina los datos de la circunferencia del brazo masculino.

Nombre de la Clase: Circunferencia Brazo Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la circunferencia del brazo masculino.

Nombre de la Clase: Pliegue Tricipital Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
modn_edades	Integer
percentil	Integer
pliegue_tricipital	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Pliegue Tricipital Femenino	
Tipo de Clase: Controlador	
Nombre:	**insertarPliegueTricipitalFem()**
Descripción:	Inserta los datos de pliegue tricipital femenino.
Nombre:	**modificarPliegueTricipitalFem()**
Descripción:	Modifica los datos de pliegue tricipital femenino.
Nombre:	**eliminarPliegueTricipitalFem()**
Descripción:	Elimina los datos de pliegue tricipital femenino.

Nombre de la Clase: Pliegue Tricipital Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear pliegue tricipital femenino.

Nombre de la Clase: Pliegue Tricipital Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial

version	Integer
modn_edades	Integer
percentil	Integer
pliegue_tricipital	Float
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Pliegue Tricipital Masculino	
Tipo de la Clase: Controlador	
Nombre:	**insertarPliegueTricipitalMasc()**
Descripción:	Inserta los datos de pliegue tricipital masculino.
Nombre:	**modificarPliegueTricipitalMasc()**
Descripción:	Modifica los datos de pliegue tricipital masculino.
Nombre:	**eliminarPliegueTricipitalMasc()**
Descripción:	Elimina los datos de pliegue tricipital masculino.

Nombre de la Clase: Pliegue Tricipital Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear pliegue tricipital masculino.

Nombre de la Clase: Peso Edad Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
modn_edades	Integer
peso	Float
percentil	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Peso Edad Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarPesoEdadFem()**
Descripción:	Inserta los datos de peso edad femenina.
Nombre:	**modificarPesoEdadFem()**
Descripción:	Modifica los datos de peso edad femenina.
Nombre:	**eliminarPesoEdadFem()**
Descripción:	Elimina los datos de peso edad femenina.

Nombre de la Clase: Peso Edad Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear el peso edad femenino.

Nombre de la Clase: Peso Edad Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer

modn_edades	Integer
peso	Float
percentil	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Peso Edad Masculino	
Tipo de la Clase: Controlador	
Nombre:	**insertarPesoEdadMasc()**
Descripción:	Inserta los datos de peso edad masculina.
Nombre:	**modificarPesoEdadMasc()**
Descripción:	Modifica los datos de peso edad masculina.
Nombre:	**eliminarPesoEdadMasc()**
Descripción:	Elimina los datos de peso edad masculina.

Nombre de la Clase: Peso Edad Masculino
Tipo de Clase: Vista

Descripción:	Se muestran los campos que debe llenar para crear el peso para la edad masculino.

Nombre de la Clase: Talla Edad Femenino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
modn_edades	Integer
talla	Float
percentil	Integer
eliminado	Boolean
cid	Integer
valor	String
codigo	String

Nombre de la Clase: Talla Edad Femenino	
Tipo de la Clase: Controlador	
Nombre:	**insertarTallaEdadFem()**
Descripción:	Inserta los datos de talla edad femenina.

Nombre:	**modificarTallaEdadFem()**
Descripción:	Modifica los datos de talla edad femenina.
Nombre:	**eliminarTallaEdadFem()**
Descripción:	Elimina los datos de talla edad femenina.

Nombre de la Clase: Talla Edad Femenino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la talla edad femenino.

Nombre de la Clase: Talla Edad Masculino	
Tipo de Clase: Modelo	
Atributo	**Tipo**
Id	Serial
version	Integer
modn_edades	Integer
talla	Float
percentil	Integer
eliminado	Boolean
cid	Integer

valor	String
codigo	String

Nombre de la Clase: Talla Edad Masculino	
Tipo de Clase: Controlador	
Nombre:	**insertarTallaEdadMasc()**
Descripción:	Inserta los datos de talla edad masculina.
Nombre:	**modificarTallaEdadMasc()**
Descripción:	Modifica los datos de talla edad masculina.
Nombre:	**eliminarTallaEdadMasc()**
Descripción:	Elimina los datos de talla edad masculina.

Nombre de la Clase: Talla Edad Masculino	
Tipo de Clase: Vista	
Descripción:	Se muestran los campos que debe llenar para crear la talla edad masculino.