Universidad de las Ciencias Informáticas

Facultad 7

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

Desarrollo del Módulo Farmacia Extrahospitalaria del Sistema de Información Hospitalaria alas HIS

Autores:
Gisel Acuña González
Carlos Manuel Álvarez López

Tutores:
MSc. Karel Gómez Velázquez
Ing. Isnel Leyva Herbella

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

Agradecimientos

De Gisel:

Si tuviera que nombrar a todas las personas que de una forma u otra han demostrado estar conmigo en estos cinco largos y duraderos años de mi carrera, creo no tener espacio suficiente para ubicarlas. Voy a comenzar haciendo mención a mis abuelos, por haberme transmitido parte de su conocimiento y servirme hoy sus experiencias de preparación para la vida.

A mis padres, por la energía que me han brindado, por ser sencillamente mi ejemplo, mi motor impulsor, inculcándome principalmente el amor hacia lo que se realice. Gracias por estar ahí.

A mi hermano, que a pesar de su corta edad para comprenderlo todo, ha querido igual que alcance resultados satisfactorios en lo que he estado desarrollando.

A Blanca Linda, mi perra más bella, por ser mi relajante y causarme diversión en mis momentos de estrés.

A mis tíos, primos que estuvieron atentos, preocupados en cada paso que di en este centro y contentos siempre por los logros obtenidos.

Quiero agradecer a Roberto, un ser especial, una persona muy respetuosa, dotado de una gran capacidad de aprendizaje, talento incomparable, a él y a su familia por haberme atendido de la forma más correcta en el momento en que lo necesité.

Gracias a los tutores por su participación en conjunto, por el chequeo constante del trabajo, por sus criterios expresados.

Agradezco a los 53 años de Revolución por hacer posible que a través del conocimiento expresado en libros, en materiales televisivos y a la tecnología en desarrollo, permita a los jóvenes estudiar para en un futuro convertirnos en personas útiles.

A las muchachitas del apartamento por mantenerse atentas y desearme siempre muy buena suerte.

A Greicel, cariñosamente Tita, gracias por su amistad desinteresada, por brindarme su total confianza y por haber creído en mí.

A las amistades del grupo, que siendo duda o verificación de un contenido no dudaron en ayudar y contestar. Una vez más se puede afirmar que en la fuerza está la unión, doy gracias por en este último curso habernos dado recíprocamente ese apoyo incondicional.

A Jorge, una persona que ha sido muy importante para mí, que ha tenido que ver muy de cerca con mi evolución en esta escuela, gracias por tu entrega desde el primer curso y en este trabajo final haberme dado tu apoyo.

A Abelardo que desde la asignatura de MIC, donde se repartieron los posibles temas de tesis, ha venido compartiendo su criterio conmigo, por su cariño, comprensión, y por haber sabido brindarme su conocimiento con paciencia, gracias.

Por último agradecer a mi compañero de tesis, que por ocupar su mayor tiempo de producción y libre en el desarrollo de la aplicación, con paciencia y esmero, hoy se obtiene un sistema robusto y flexible.

De Carlos:

A mi familia, sin su confianza llegar hasta aquí hubiese sido una tarea imposible.

A mis tutores, gracias por su apoyo y por tener una mano siempre tendida en los momentos más difíciles.

A mis amigos, que son parte de este trabajo.

A todos aquellos que dieron su aporte y supieron en los momentos buenos y malos mostrarme el camino, que no siempre fue el más fácil, pero fue el que más disfrutamos.

Dedicatoria

De Gisel:

Quiero dedicar este trabajo a todas aquellas personas que contribuyeron a su desarrollo, personas que me guiaron, que me ayudaron a reflexionar, que compartieron conmigo momentos de alegría y de depresión: A mis padres, mi fuente de luz, amor y progreso, sin ellos tan fácil no lo hubiese logrado; a mi tía Gertrudis; por inculcarme al igual que mi madre, esa fuerza de voluntad para lograr un propósito, por enseñarme a levantar si en un momento me he caído; a dos compañeros amigos míos que hoy si se lo hubiesen permitido estarían exponiendo su trabajo junto conmigo; y en especial a dos seres queridos; a mi abuelo, símbolo de preparación intransigente y a mi hermano para el cual pretendo ser un ejemplo.

De Carlos:

A todos aquellos que me demostraron que un ingeniero no es una copia, es original y se atreve a cambiar una realidad, no importa el tiempo o el espacio, y todo es posible mientras crea que es así.

RESUMEN

Actualmente en el área de la Farmacia Extrahospitalaria se realiza todo el trabajo en forma manual y se maneja gran cantidad de información por el personal que allí labora. El objetivo del presente trabajo es desarrollar el Módulo Farmacia Extrahospitalaria del Sistema de Información Hospitalaria alas HIS, para facilitar la gestión de información en esta área de las instituciones hospitalarias.

El desarrollo de este sistema se encuentra guiado por el Proceso Unificado de Desarrollo (RUP), se basa en tecnologías libres, multiplataforma y sobre una arquitectura en capas. Utiliza la Notación para el Modelado de Procesos de Negocio (BPMN) y como lenguaje de programación, Java. Se emplea el patrón arquitectónico Modelo-Vista-Controlador (MVC) que facilita la creación de una aplicación robusta y flexible.

Como resultado se pretende que el sistema permita un control estricto de los medicamentos sumistrados a los pacientes de Consulta Externa. Además la generación de reportes que muestren información relacionada con la salida de los medicamentos por concepto de venta. El sistema se rige por la Clasificación Anatómica, Terapéutica y Química de medicamentos, más conocida como “clasificación ATC” (Anatomical Therapeutic Chemical) avalada por la Organización Mundial de la Salud (OMS). Lo cual pondrá a disposición del personal médico una herramienta de gestión clínica y administrativa que responde a sus necesidades.

Palabras Claves: *clasificación ATC, Consulta Externa, Farmacia Extrahospitalaria, reportes.*

ÍNDICE

ÍNDICE DE TABLAS

INTRODUCCIÓN

Las tecnologías de la información son disciplinas que rápidamente han pasado del plano estrictamente científico al mundo cotidiano. Cada vez y con mayor frecuencia el hombre incorpora a su vida diaria una serie de instrumentos de naturaleza electrónica que tienen como referencia obligada a la recopilación, al procesamiento o uso de datos. El más conocido de estos equipos es la computadora. Puede ser considerada como la representación física de la modernidad, la eficiencia y el cambio tecnológico. [1]

La Informática ha evolucionado en paralelo con las crecientes necesidades del hombre, es la ciencia que se encarga del tratamiento automático de la información. Este tratamiento automático ha propiciado y facilitado la manipulación de grandes volúmenes de datos además de la ejecución rápida de cálculos complejos. [2] Ha sido reconocida a nivel mundial por su finalidad, métodos, herramientas y su relación en las áreas de las ciencias médica, por ejemplo, Epidemiología, Banco de Sangre, Hospitalización entre otras.

El acelerado desarrollo de esta rama ha conducido a un mejor accionar dentro del campo de la salud. Se encuentra orientada a garantizar la efectividad clínica permitiendo obtener un resultado satisfactorio en la atención al paciente y en la gestión de los procesos asistenciales. Estos son centrados en el usuario, implican a los profesionales de los hospitales y los centros de salud, equipos multidisciplinarios, evidencia científica y un Sistema de Información integrado que diseñe, recoja y evalúe indicadores conjuntos en Atención Primaria y Atención Especializada. [3]

La Informática Médica consiste en la aplicación de la Informática y las Comunicaciones en el sector de la medicina. Es una especialidad de apoyo a la medicina clínica y académica, y a la administración de salud. Puede traer mejoras importantes en la asignación de recursos, en la seguridad de los pacientes y en la educación médica. La Asociación Americana Informática Médica la define como: “La Informática Médica se preocupa de entender y promover el uso, organización, análisis y administración eficiente de la información en el cuidado de la salud…”. [4]

Por tal motivo resulta ser un campo multidisciplinario que acoge a todos los profesionales de áreas como la Biomedicina, diversas ingenierías como la Informática, Telecomunicaciones, Electrónica y Recursos

Financieros. Los instrumentos informáticos de la salud incluyen además de los ordenadores, guías de práctica clínica, terminología médica formal y de sistemas de información y comunicación.

En la década de los setenta, en el campo de la medicina, se perfilaron los primeros sistemas de información médica que posteriormente, dieron lugar a los Sistemas de Información Hospitalario (HIS, por sus siglas en inglés). [5] Los HIS son aquellos que se encuentran orientados a satisfacer las necesidades de generación de información, para almacenar, procesar, recopilar datos médicos de atención al paciente y administrativos de cualquier institución hospitalaria. [6]

En la actualidad los HIS deben presentar funcionalidades como: generación de recetas, órdenes o formularios de trabajo, llevar el registro del paciente con expediente digital, control de laboratorios y quirófanos desde una sala de mando, generación de estadísticas y reportes, acceso a bibliotecas y base de datos, entre otras de categoría relevante. Implican una base de datos a disposición de usuarios autorizados en el lugar y en el momento en que estos datos sean requeridos, en un formato adecuado a sus necesidades específicas. [6]

El desarrollo de los HIS posibilita a los profesionales de la salud encontrar un recurso idóneo, amigable y flexible que acelere la obtención de la información en la institución hospitalaria, permite lograr así la optimización de recursos humanos y materiales. Estos sistemas generan reportes e informes, en dependencia del área o servicio para el cual se requiera, lo cual da lugar a la retroalimentación de la calidad de la atención médica.

Una institución hospitalaria se encuentra conformada por diversas áreas que responden a las necesidades del sistema sanitario. De los servicios de apoyo, el servicio de Farmacia es posiblemente uno de los de creación más reciente. Es el área en el cual se comprende que el medicamento sea el recurso natural del farmacéutico para integrarse, de forma responsable, en el equipo asistencial y cuidado directo del paciente. Esta se especializa en la Farmacia Intrahospitalaria y la Farmacia Extrahospitalaria. La Farmacia Intrahospitalaria es la encargada de administrar y proveer medicamentos a los pacientes que se encuentran hospitalizados y en emergencia y la Farmacia Extrahospitalaria es la encargada de atender los pacientes provenientes de Consulta Externa.

Todos los datos necesarios del medicamento son registrados en las tarjetas de estantería que existen en la Farmacia. Se almacena toda la información con respecto a la tramitación de los medicamentos que hayan sido entregados, vencidos, solicitados o cualquier otra gestión que se necesite realizar. Esto trae como desventaja que dificulte la utilización y la manipulación en general de los datos. Al ser archivada de forma manual la información, no tiene seguridad debido a que el deterioro por antigüedad o negligencia, puede conllevar a que sea descifrable o a la pérdida de la misma. Por tal motivo se adoptan medidas de preservación creando copias de seguridad que pueden no implicar una desactualización en los datos duplicados, pero sí un gasto excesivo de recursos y tiempo de los funcionarios encargados de controlar las salidas y entradas de los medicamentos.

Para tener control sobre el consumo de cada medicamento en la Farmacia por período de tiempo se realizan los cálculos de stock. Estos cálculos se efectúan con el objetivo de mantener un estricto registro de los valores máximos y mínimos de consumo para conocer el comportamiento de estos y poder dar un tratamiento diferenciado, o sea, mantener los diferentes lotes de cada producto separados e identificados. Todo este proceso se realiza de forma manual con el riesgo de que existan errores, generándose información incorrecta.

Actualmente las indicaciones médicas al paciente se realizan a través de una receta en formato de papel. Al portar el paciente este documento puede conllevar al deterioro o pérdida del mismo. En ocasiones resulta común que su información no sea legible debido a la mala caligrafía y a la rapidez con que son elaboradas las recetas médicas. El área de Farmacia Extrahospitalaria no presenta hoy comunicación con el resto de las áreas del hospital, por lo que se dificulta determinar el origen exacto de la receta médica y que no exista seguridad de su procedencia. Además de imposibilitar establecer consultas bidireccionales y poder detectar casos de abusos de medicamentos o adicciones. Se necesita mantener un control estricto de los precios y los fondos que son ingresados por la venta de los medicamentos prescritos al paciente.

El Centro de Informática Médica (CESIM) de la Universidad de las Ciencias Informáticas desarrolla el Sistema de Información Hospitalaria alas HIS, con el propósito de obtener un sistema centralizado de información y lograr elevar la calidad de la atención al paciente. Está compuesto por diferentes módulos que interconectan las diferentes áreas de una institución hospitalaria como Admisión, Consulta Externa, Hospitalización, Emergencia, Epidemiología, Banco de Sangre, además de Farmacia, este módulo cuenta

con el servicio de apoyo Farmacia Intrahospitalaria. Este sistema no comprende los procesos que se desarrollan en el área de la Farmacia Extrahospitalaria.

Por la problemática anteriormente expuesta se determina como ***Problema a resolver*** ¿Cómo facilitar la gestión de la Información relacionada con los procesos del área de Farmacia Extrahospitalaria de las instituciones hospitalarias?

Por consiguiente, el ***Objeto de estudio*** lo constituye el Proceso de gestión de la información del área de Farmacia de las instituciones hospitalarias. El ***Campo de acción*** se enmarca en el Proceso de gestión de la información del área de Farmacia Extrahospitalaria de las instituciones hospitalarias.

Para solucionar los problemas mencionados, se determina como ***Objetivo general***: Desarrollar el Módulo Farmacia Extrahospitalaria del Sistema de Información Hospitalaria alas HIS, que facilite la gestión de la información en esta área de las instituciones hospitalarias.

Se identifican las siguientes ***Tareas a desarrollar*** con el propósito de dar cumplimiento al objetivo general anteriormente propuesto:

1. Describir los procesos del negocio relacionados con el área de Farmacia Extrahospitalaria de las instituciones hospitalarias.
2. Evaluar las ventajas y desventajas de sistemas de informática médica que cuenten con funcionalidades para la gestión de los procesos en el área de Farmacia Extrahospitalaria de las instituciones hospitalarias.
3. Aplicar la metodología, plataforma, tecnologías, librerías, herramientas y pautas definidas por el Departamento de Sistemas de Gestión Hospitalaria para el desarrollo de sus aplicaciones.
4. Obtener mediante el Proceso Unificado de Desarrollo, los artefactos correspondientes a los flujos de trabajo de “Implementación” y “Pruebas”.
5. Implementar los procesos de negocio correspondientes al área de la Farmacia Extrahospitalaria.
6. Desarrollar el proceso de pruebas para obtener la correspondiente Acta de Liberación por el Grupo de Calidad interno del Departamento de Sistemas de Gestión Hospitalaria.

El desarrollo del Módulo Farmacia Extrahospitalaria del Sistema de Información Hospitalaria alas HIS proporcionará una serie de beneficios entre ellos pueden ser mencionados los siguientes:

1. Organización de los diferentes procesos que se desarrollan en el área de Farmacia Extrahospitalaria, con el objetivo de lograr una homogeneidad de los mismos en cualquier institución hospitalaria, así como el uso de catálogos de medicamentos estandarizados y aprobados por la OMS.
2. Mantenimiento de un control de la existencia y la dispensación de los medicamentos en el área de la Farmacia que garantice un correcto funcionamiento y evite errores en la entrega de los mismos que minimicen los riesgos para el paciente.
3. Optimización del aprovechamiento de los recursos farmacéuticos lo cual posibilita hacer rentable el consumo de bienes materiales en el área.
4. Mantener un control estricto de los fondos ingresados por concepto de venta de los medicamentos indicados al paciente.
5. Garantizar la autenticidad y fidelidad de los datos por el personal que en esta área opera para elaborar el tratamiento de atención al paciente.
6. Mejora de las condiciones de trabajo del personal en aras de garantizar la agilidad de los procesos y calidad del trabajo.

El documento presentado se encuentra estructurado en cuatro capítulos:

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA: Se realiza un estudio profundo sobre los Sistemas de Información Hospitalaria existentes que gestionen los procesos del área de Farmacia Extrahospitalaria. Se describen además las tendencias, tecnologías, herramientas y metodología, utilizadas para la implementación de los procesos de negocio identificados en la presente investigación.

CAPÍTULO 2: DESCRIPCIÓN DE LA ARQUITECTURA: Se especifican los requisitos no funcionales, se fundamenta y describe la arquitectura propuesta, además de realizarse un análisis de la implementación de la aplicación.

CAPÍTULO 3: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA: Se detalla mediante el esbozo de diagramas el diseño del sistema. Se muestran los artefactos generados, se describen las clases con sus atributos y operaciones necesarias encargadas de manejar el negocio del sistema y se presenta la propuesta de solución para lograr una gestión eficiente de los procesos hospitalarios asociados al área.

CAPÍTULO 4: MODELO DE PRUEBA: Se exponen los conceptos de los métodos existentes para realizar la evaluación del sistema en la etapa de pruebas, describiéndose el utilizado, método de caja negra y se muestran los casos de prueba.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este primer capítulo se abordan los conceptos más relevantes que sirven de base para el entendimiento del desarrollo de un HIS que contenga el módulo de Farmacia Extrahospitalaria. Se describen los Sistemas de Información Hospitalaria existentes que gestionen los procesos relacionados a esta área de la Farmacia, que faciliten desde el momento de su creación funcionalidades que hoy se quieren optimizar y nuevas que se pretenden analizar para su correcta implementación en dependencia de las necesidades del sistema en desarrollo. Se detallan las herramientas, tecnologías y la metodología utilizada con el propósito de desarrollar una aplicación, robusta y flexible.

1.1 Sistemas de salud pública

Los sistemas de salud crecieron explosivamente con el desarrollo del conocimiento científico y con la revolución industrial. Consisten en todas las personas, recursos disponibles y acciones cuyo propósito primario es el de mejorar, promover, recuperar y mantener la salud. En la teoría clásica de Organización de la Salud Pública, se ha justificado como una necesidad para la racionalización de los recursos a partir que la mayoría de las necesidades de la población se satisfacen en el nivel de menor complejidad.

A raíz de esto se han definido tres niveles, de los cuales la **Atención Primaria de Salud (APS)**, según su definición, da solución al 80 % o más de los problemas de salud de la población, con las especialidades básicas, mediante la ejecución de los programas aprobados; el segundo nivel, la **Atención Médica Secundaria,** tiene como objetivo principal tratar al hombre enfermo, presta en estos momentos servicios de diagnóstico, ofrece tratamiento curativo y de rehabilitación, además de continuar trabajando en la prevención de la salud; este nivel lo integran las diversas redes hospitalarias del sistema de salud. Por último, el tercer nivel, la **Atención Médica Terciaria**, lo componen las unidades de atención especializada o terciaria, que combinan las actuales unidades hospitalarias con los recursos tecnológicos y terapéuticos más innovadores, brindan servicios de muy alta complejidad, relacionados con secuelas o aumento de las complicaciones de determinadas dolencias**.** [7]

1.2 Descripción de los procesos que se desarrollan en el campo de acción

La Farmacia Extrahospitalaria puede ser parte de las áreas que conforman una institución hospitalaria o puede encontrarse fuera de ella, en la comunidad. Presenta como objetivo, atender a los pacientes no hospitalizados que reciben las prescripciones médicas directamente con el médico de la consulta. Es la encargada de realizar en la práctica la preparación, presentación y dispensación de medicamentos así como el desarrollo de la venta de productos farmacéuticos.

Para la informatización de esta área se identificaron los principales procesos del negocio, como el primero y más esencial: Despachar medicamento **(Ver Anexo 1)**, consiste en la llegada del paciente con la indicación médica (receta), el dependiente verifica este documento, comprobando que la fecha sea válida, que esté presente la firma del médico que realizó la prescripción y que se encuentre en existencia el medicamento solicitado, de no cumplirse algunos de estos parámetros es devuelta la receta al paciente. Durante la venta se crea un nuevo documento, el vale, el cual recoge información acerca de los medicamentos a despachar, como el nombre, la cantidad solicitada y el precio correspondiente.

Cada farmacia debe tener interna un área específica para mantener un control estricto de la entrada y salida de los medicamentos al área de despacho, esta área se llama Almacén. Solicitar medicamento fue otro proceso identificado **(Ver Anexo 2)**, cuando entra el primer turno de la mañana se realiza un conteo de los medicamentos para verificar la cantidad disponible tanto en el almacén como en el área de despacho, de no haber se realiza semanal o si urge, una solicitud al Almacén Central.

Por último se encuentra el proceso Liquidación de caja **(Ver Anexo 3)**, donde se extrae el dinero recaudado durante ese turno por concepto de venta. Al finalizar se le hace la entrega al Banco, donde este semanalmente emite un comprobante.

1.3 Clasificación Anatómica, Terapéutica y Química (ATC)

Por la necesidad de establecer un sistema de clasificación internacional que pudiera utilizarse en los estudios sobre uso de medicamentos, se dan los primeros pasos en 1969 con aplicaciones que dieran solución a esta problemática. Ya en la década de los setenta, investigadores noruegos junto a demás expertos modificaron y ampliaron el antiguo sistema de clasificación anatómica (AC System, por sus siglas

en inglés) de productos farmacéuticos, concibieron un sistema de Clasificación Anatómica, Terapéutica y Química, más conocido como “clasificación ATC” (Anatomical Therapeutic Chemical). [8]

Al mismo tiempo de la creación del sistema de clasificación ATC, surgió la necesidad de formular una unidad de técnica de medida, Defined Daily Dose (DDD). Esta unidad de medida no refleja la dosis diaria que se recomienda o prescribe al paciente. Al contrario, la dosis que se asigna a un paciente suele diferir de la DDD, ya que se basa en factores individuales como el peso, la edad. Años más tarde, fue aprobada por la OMS. Hoy es utilizada la clasificación ATC/DDD en varios centros colaboradores de la OMS que participan en actividades de vigilancia farmacéutica. [8]

La clasificación ATC se encuentra estructurada en 5 niveles:

- **Primer nivel (anatómico):** Órgano o sistema sobre el que actúa el fármaco.
- **Segundo nivel:** Subgrupo terapéutico.
- **Tercer nivel:** Subgrupo terapéutico o farmacológico.
- **Cuarto nivel:** Subgrupo terapéutico, farmacológico o químico.
- **Quinto nivel:** Nombre del principio activo (monofármaco) o de la asociación medicamentosa.

1.4 Sistemas automatizados existentes vinculados al campo de acción

A continuación se mencionan algunos de los principales sistemas informáticos que en el mundo se encuentran vinculados a la gestión farmacéutica en general, con el objetivo de encontrar soluciones factibles a los problemas planteados de esta área. Se describen las ventajas y desventajas de cada uno, así como las tecnologías y herramientas utilizadas para su desarrollo y posterior funcionamiento.

FARHOS (Gestión de Farmacias Hospitalarias)

Es un sistema de información para la gestión integral de la farmacia hospitalaria. Entre las principales funcionalidades que brinda están: la gestión de compras y almacén, la prescripción asistida desde planta, la distribución de medicamentos, la dispensación individualizada tanto a pacientes ingresados como ambulatorios y conexiones a otros sistemas. Está desarrollada con el modelo Cliente-Servidor. Como cliente utiliza cualquier máquina con el sistema operativo Windows 95, 98, NT, W2000. El servidor puede

estar sobre sistema operativo Windows o Unix. Como sistema gestor de base de datos posee Interbase 6, y como herramientas de desarrollo cliente MS-Access 2000. [9]

TiCares-Dis

TiCares-Dis es un sistema de farmacia hospitalaria desarrollado por la empresa Telvent, que permite mantener un control de la distribución de los fármacos a las unidades asistenciales en la base de las prescripciones activas de los pacientes. Contempla distintos mecanismos de distribución de los fármacos: unidosis o multidosis intermedia u optimizada. Permite la consulta de tratamientos activos y del historial de tratamientos así como la gestión de devoluciones de fármacos. Está integrado con el Sistema de Información Hospitalaria (HIS), TiCares, desarrollado por la misma empresa. [10]

HIS CNT Pacientes

La empresa CNT Sistemas de Información, establecida en Colombia con 25 años de experiencia, especializada en realizar soluciones integradas en el área de la salud. Presenta como su principal producto el HIS CNT Pacientes, el cual está diseñado para integrar todo el ciclo de atención del paciente frente a la prestación de servicios médicos, de diagnósticos y terapéuticos.

HIS CNT Pacientes, presenta el módulo Farmacia-Almacén. Dicho módulo permite optimizar la gestión de inventarios, esto permite controlar la administración de depósitos de medicamentos. Provee funcionalidades para el registro de cotizaciones, órdenes de compra, entradas a almacén y demás movimientos. Posibilita crear diferentes niveles de Sub Stock o dependencias, así como el respectivo procesamiento de los pedidos de requisición y despachos a los diferentes Sub Stocks definidos. El módulo cuenta con funcionalidades para administrar el stock del depósito, de esta forma permite ajustes e informando qué ítems se encuentran por debajo del nivel mínimo del punto de reposición. [11]

SIGHO

SIGHO es un Sistema de Información para la Gerencia Hospitalaria, basado en la Norma Oficial Mexicana NOM-168-SSA1-1998 referente al resguardo y uso del expediente clínico electrónico para facilitar las

actividades de gerencia dentro del hospital. Se apoya de estándares internacionales para el diagnóstico de enfermedades y realización de procedimientos tales como el CIE-10 y CIE9MC.

Permite realizar registros individuales alrededor de la Historia Clínica Electrónica, en cada uno de los módulos relacionados con la atención al paciente. Cuenta con 14 módulos dentro de los cuales se encuentra el módulo de Farmacia-Almacén. Este módulo permite administrar y tener un buen control de los medicamentos de la farmacia, entregados a pacientes atendidos en la unidad médica. [12]

Este sistema es una aplicación de escritorio poco portable. No utiliza los estándares internacionales de codificación de medicamentos (ATC) y sólo realiza despacho de medicamentos a los stocks y recetas médicas sin crear un perfil farmacoterapéutico, es utilizado el sistema de dispensación por dosis unitaria. [11] No brinda la posibilidad de tener en cuenta a pacientes remitidos desde Consulta Externa, lo cual se considera como una desventaja.

INDRA

Una nueva solución en lo que respecta a gestión administrativa para el sector de la salud es la receta médica electrónica; sistema desarrollado por la empresa Indra que conecta a los centros de salud con las farmacias para que éstas puedan acceder a la prescripción realizada por el médico, permitir el intercambio de mensajes entre farmaceuta y médico, así como facilitar la consulta sobre los medicamentos entregados a cada paciente. La receta electrónica proporciona control sobre los medicamentos entregados, lo que permite verificar el estado del inventario y facilitar el proceso de aprovisionamiento.

La receta electrónica ha sido implementada en las comunidades de Madrid, Valencia y Andalucía, con diferentes módulos de aplicación: uno de identificación, para la acreditación y autorización de usuarios, médicos y farmacéuticos mediante el uso de la tarjeta sanitaria y de certificados digitales para la firma electrónica; otro, para la prescripción de medicamentos por parte del médico en el centro de salud, y uno de dispensación.

Esta solución busca reducir enormemente el tiempo de registro y gestión de las recetas, además de evitar gran número de visitas a los centros de salud para renovar la indicación médica de pacientes que requieren medicación permanente. Para esto se realiza una prescripción global de una sola vez, lo que

permite mejorar la calidad del servicio, aumentar la disponibilidad de consulta de los médicos, facilitar el seguimiento de las prescripciones y disponer de información clínica, farmacológica y farma-económica completa y actualizada. [13]

FARMATOOLS

El sistema es una solución orientada a la gestión del medicamento en el entorno hospitalario, provee de herramientas clínicas de alto nivel para controlar del uso del medicamento y facilitar la gestión de almacenes, prescripciones, fabricaciones de mezclas, citostáticos y otros circuitos necesarios en los servicios de farmacia hospitalaria.

FARMATOOLS está compuesto por cuatro grandes módulos: Módulo Unidosis, Pacientes ambulantes (Citostáticos), Pacientes Externos y Gestión Económica.

Es una herramienta útil para optimizar la gestión de compras y de almacenes farmacéuticos; mejorar la prescripción, vigilancia y la gestión de alertas; controlar el uso de medicamentos estupefacientes; realizar estadísticas para el control del gasto farmacéutico. Las tecnologías incluidas en el software permiten una rápida adaptación a las necesidades del cliente y una integración nativa con otras aplicaciones clínicas y dispensadores automatizados. [14]

Los sistemas de información hospitalarios encargados de llevar la gestión farmacéutica como uno de sus servicios anteriormente descritos, presentan como característica en común que son software propietario. Todos excepto el TiCares-Dis, son aplicaciones de escritorio, haciéndoles carecer de multifuncionalidades con otros sistemas operativos. Una más de sus características es que no utilizan estándares para la codificación de medicamentos.

La nueva solución debe abarcar las funcionalidades de la mayoría de estos sistemas existentes más las que lo pueden hacer atractivo, eficiente y óptimo. Debe encontrarse desarrollado sobre tecnologías libres, ser una aplicación Web, ventaja que presenta el INDRA, lo que posibilita gran capacidad de adaptación dependiendo de las necesidades que presente independientemente del sistema operativo usado. Debe estar integrado al sistema alas HIS, comunicándose directamente con los Módulos de Consulta Externa y Almacén, de forma que permite recibir las indicaciones médicas prescriptas a pacientes externos, al

mismo tiempo llevar un control constante del consumo y existencia de cada uno de los medicamentos. La aplicación se debe regir por la clasificación ATC.

1.5 Tendencias y tecnologías actuales a considerar

El determinar con exactitud las herramientas a utilizar para llevar el desarrollo de una aplicación resulta de carácter significativo ya que permite trabajar de forma organizada, obtener una solución satisfactoria del problema a resolver. Se presentan a continuación las herramientas y tecnologías analizadas para la realización del sistema informático.

1.5.1 Patrón Modelo-Vista-Controlador (MVC)

Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón de llamada y retorno MVC, se ve frecuentemente en aplicaciones Web, donde la vista es la página HTML (HyperText Markup Language) y el código que provee de datos dinámicos a la página. El modelo lo constituyen los datos con los que trabaja la aplicación y la lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista. El estilo fue descrito por primera vez en 1979 por Trygve Reenskaug, cuando trabajaba en Smalltalk en laboratorios de investigación de Xerox. [15]

Modelo: Esta es la representación específica de la información con la cual el sistema opera. El modelo se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas. El sistema también puede operar con más datos no relativos a la presentación, haciendo uso integrado de otras lógicas de negocio y de datos afines con el sistema modelado.

Vista: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

Controlador: Este responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y, probablemente, a la vista.

1.5.2 Tecnologías utilizadas en el proceso de desarrollo

Se exponen las tecnologías agrupadas según la capa en que estén presentes, utilizadas en el proceso de desarrollo.

Capa de Presentación

XHTML

Combina el eXtensible Markup Language (XML, por sus siglas en inglés) y el HyperText Markup Language (HTML, por sus siglas en inglés) en un mismo formato. Permite extender etiquetas propietarias. Es el leguaje de marcas por defecto que utiliza JSF. [16]

Java Server Faces (JSF)

Es un framework orientado a la construcción de interfaces gráficas de usuario (GUI, por sus siglas en inglés), este marco separa el comportamiento de la presentación, teniendo su propio server como controlador, esto pone de manifiesto el uso del patrón de diseño Modelo-Vista-Controlador, lo que hace más simple su utilización. [17]

Facelets

Es un marco de trabajo orientado a JSF para el trabajo con plantillas, permite separar los componentes de interfaces de usuario en diferentes archivos. Una de las ventajas que se explotan en el sistema de este framework es que no depende de un contenedor Web. [18]

Ajax4JSF

Es una librería de código abierto que se integra totalmente a la arquitectura de JSF. Mediante este framework se puede alargar el ciclo de vida de JSF, recargar componentes sin la necesidad de hacerlo en la página completa además de hacer peticiones asíncronas al servidor. [19]

RichFaces

Es una biblioteca de componentes JSF y un avanzado framework para la integración de Ajax con facilidades en la capacidad de desarrollo de aplicaciones de negocio. Incluye un fuerte apoyo para la skinnability (cambio de apariencia) de aplicaciones JSF y aprovecha al máximo los beneficios de este incluyendo la validación y conversión de instalaciones, junto con la gestión estática y dinámica los recursos. [20]

Seam UI

Serie de controles JSF altamente integrables con JBoss Seam. Adicionan varias mejoras a JSF, desde validación, expresiones Extended EL, integración de la navegación en la interfaz de usuario basada en procesos del negocio. [21]

Capa de Negocio

JBoss Seam

Es un software de código abierto que asocia diferentes tecnologías y estándares Java adicionando algunas funcionalidades no contempladas en frameworks anteriores. Automatiza muchas de las tareas comunes y hace un uso extensivo de las anotaciones para poder reducir la cantidad de código XML que es necesario escribir, reduce de manera importante la cantidad total de codificación necesaria. Permite la creación de complejas aplicaciones Web. [22]

Integra además, el concepto de workspaces (espacios de trabajo) permite que el usuario tenga en varios tabs o ventanas del navegador actividades del negocio con contextos completamente aislados. Seam integra transparentemente la administración de procesos del negocio vía jBoss jBPM, hace más fácil implementar y optimizar complejas colaboraciones e interacciones con el usuario.

Drools

Drools (o JBoss Rules) es un Sistema de Gestión de Reglas de Negocio (BRMS, por las siglas en inglés de Business Rule Management System) con un motor de reglas basado en inferencia de encadenamiento

hacia adelante (forward chaining), conocido como sistema de reglas de producción, usando una implementación avanzada del algoritmo Rete.

Es software libre distribuido según los términos de la licencia Apache. Drools soporta el estándar JSR-94 para su motor de reglas de negocio y framework de empresa para construcción, mantenimiento y refuerzo de políticas de empresa en una organización, aplicación o servicio. Drools usa JCR (JackRabbit) para gestionar el repositorio de reglas, y el estándar Java Authentication and Authorization Service (JAAS, por sus siglas en inglés) para la autorización y autenticación. [23]

Capa de Datos

Java Persistence API (JPA)

Java Persistence API, más conocida por sus siglas 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. [24]

Enterprise JavaBeans (EJB3)

La especificación de Enterprise JavaBeans, tiene la intención de proporcionar una forma estándar para implementar el código "negocio" el back-end se encuentran típicamente en las aplicaciones de empresa (en lugar de código de interfaz de 'front-end "). Dicho código se encuentra con frecuencia para hacer frente a los mismos tipos de problemas, y se encontró que las soluciones a estos problemas son a menudo varias veces re-implementado por los programadores. Enterprise JavaBeans están preparadas para soportar los problemas comunes, tales como la persistencia, la integridad transaccional, y la seguridad de una manera estándar. [25]

Hibernate

Es una herramienta de Mapeo Objeto Relacional (ORM) y un generador de sentencias SQL (Structured Query Language). Permite diseñar objetos persistentes que podrán incluir polimorfismo, relaciones, colecciones, y un gran número de tipos de datos. Brinda la posibilidad de generar bases de datos en cualquiera de los entornos soportados: Oracle, DB2, MySql y PostgreSQL.

Permite expresar consultas en una extensión de SQL (HQL), así como en SQL nativo o utilizando criterios orientado a objetos. Puede ser usado para desarrollar y distribuir aplicaciones de forma gratuita. Hibernate implementa las especificaciones de EJB3 y sigue el estándar JPA. [26]

1.6 Tendencias y metodologías horizontales

PosgtreSQL

PostgreSQL es un Sistema de Gestión de Bases de Datos (DBMS) objeto relacional de código abierto, que se destaca por su robustez, escalabilidad y cumplimiento de los estándares SQL. Este cuenta con diversas versiones para sistemas operativos tales como Linux, Windows, Mac OS X, Solaris, BSD, Tru64 y otros. Entre sus principales características se encuentran:

Extensible: El código fuente está disponible para todos sin costo. Si un equipo necesita extender o personalizar PostgreSQL de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales. Esto es complementado por la comunidad de profesionales y entusiastas de PostgreSQL alrededor del mundo que también extienden PostgreSQL todos los días.

Multiplataforma: PostgreSQL está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y una versión nativa de Windows está actualmente en estado beta de pruebas. Diseñado para ambientes de alto volumen: PostgreSQL usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mejor respuesta en ambientes de grandes volúmenes. Los principales proveedores de sistemas de bases de datos comerciales usan también esta tecnología, por las mismas razones. [27]

Java

Java es un lenguaje de Programación Orientado a Objetos (POO), desarrollado por Sun Microsystems a principios de los años 90. 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. La programación en el mismo permite el desarrollo de aplicaciones bajo el esquema cliente servidor, lo que lo hace capaz de ejecutar tareas simultáneamente, distribuyendo el trabajo a realizar. [28]

Entre sus principales características se encuentran:

1. Realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores, lo antes posible, en el ciclo de desarrollo.

2. 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 puede ejecutar ese código objeto, sin importar en modo alguno la máquina en que ha sido generado. Actualmente existen sistemas de ejecución para Windows, Linux y Mac entre otros.

3. Las aplicaciones de Java resultan extremadamente seguras, ya que no acceden a zonas delicadas de memoria o de sistema, con lo cual evitan la interacción de ciertos virus. Java no posee una semántica específica para modificar la pila de programa, la memoria libre o utilizar objetos y métodos de un programa sin los privilegios del núcleo del sistema operativo.

4. Los programas Java pueden ser aplicaciones de escritorio, se ejecutan sin necesidad de un navegador.

   Applets: Se pueden descargar de internet y se observan en un navegador.

   JavaBeans: Componentes software Java, que se puedan incorporar gráficamente a otros componentes.

JavaScript: Conjunto del lenguaje Java que puede codificarse directamente sobre cualquier documento HTML.

Servlets: Módulos que permiten sustituir o utilizar el lenguaje Java en lugar de programas CGI (Common Gateway Interface) a la hora de dotar de interactividad a las páginas Web.

Java Enterprise Edition 5 (JEE5)

Es la plataforma que provee Sun Microsystem para dar soporte y desarrollar software para las empresas. Esta edición, dio un gran paso hacia la excelencia, pues incorporó nuevas tecnologías de punta y que le incorporan a la plataforma gran robustez y simplicidad a la hora de trabajar. [29]

Entre sus características se encuentran:

1. Es un estándar del sector para desarrollar aplicaciones Java portátiles, robustas, escalables y seguras para el servidor.
2. Proporciona API de servicios Web, modelos de componente, gestión y comunicación que lo convierten en el estándar del sector para implementar aplicaciones SOA (Arquitectura Orientada a Servicios).
3. Define una infraestructura para aplicaciones que distintas organizaciones pueden implementar de forma independiente, pero se comporta de forma coherente en todas ellas.

Java Virtual Machine

La Máquina Virtual Java es el núcleo del lenguaje de programación Java. De hecho, es imposible ejecutar un programa Java sin ejecutar alguna implantación de la misma. En ella se encuentra el motor que en realidad ejecuta el programa Java y es la clave de muchas de las características principales de Java, como la portabilidad, la eficiencia y la seguridad. Se encarga de interpretar todo el código java y convertirlo al lenguaje nativo del sistema operativo en uso. [30]

JBoss Application Server

JBoss Application Server es un servidor de aplicaciones que contiene subdirectorios que tienen los scripts de inicio del servidor, jars, grupos de configuración de éste y directorios de trabajo. Necesita conocer la distribución para poder ubicar las jars para compilar código, actualizar configuraciones, implementar código, entre otras funciones. [31]

Proceso Unificado de Desarrollo (RUP)

RUP es una de las metodologías de desarrollo de software más utilizada en el mundo para el análisis, implementación y documentación de sistemas orientados a objetos. Durante todo el ciclo de desarrollo, se administran los requisitos, con el objetivo de mantener la trazabilidad de los requerimientos funcionales, siendo así el proceso de desarrollo guiado por casos de uso. Permite la reutilización de estos requerimientos para otros sistemas o funcionalidades del propio software. RUP es centrado en la arquitectura, por lo que es fundamental identificar los casos de uso de mayor prioridad e importancia basándose en este criterio. [32]

Dentro de sus disciplinas gestiona el control de cambios, que permite mantener al equipo trabajando en los mismos artefactos, en cualquier momento del desarrollo del producto. En su modelación RUP define como sus principales elementos a los trabajadores, sus responsabilidades, los artefactos y los flujos de actividades. Los trabajadores son los propietarios de elementos o artefactos y se encargan de realizar las actividades, las cuales describen cómo una tarea es realizada por un trabajador y a su vez, manipulan elementos. RUP realiza varias iteraciones en número variable en las que se hacen mayor o menor hincapié en las distintas actividades por lo que se han definido en nueve flujos o disciplinas. [32]

Lenguaje Unificado de Modelado (UML)

UML es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. Este se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software como es el caso de RUP, pero no especifica en sí mismo qué metodología o proceso usar. UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. [33]

1.7 Herramientas

Eclipse

Eclipse es un entorno de desarrollo integrado (IDE), portable y multiplataforma. Diseñado por la empresa IBM y actualmente sujeto a la Fundación Eclipse. Brinda una plataforma universal para integrar herramientas de desarrollo, con una arquitectura abierta, basada en plugins que posibilita integrar múltiples lenguajes como Java, C++, PHP, Perl sobre un mismo entorno de desarrollo. Brinda nuevas funcionalidades en dependencia de las necesidades o lo que se desee obtener. Permite además la vinculación con la herramienta de modelado Visual Paradigm mediante el Enterprise Edition, lo que proporciona un mejor entendimiento entre analistas y desarrolladores. [34]

Visual Paradigm 6.4

Visual Paradigm es una herramienta CASE (Computer Aided Software Engineering, en su traducción Ingeniería de Software Asistida por Computación) que da soporte al modelado visual con UML y con la Notación para la Modelación de Procesos de Negocio (BPMN, por sus siglas en inglés). Esta permite recorrer el ciclo de vida del desarrollo de software, desde el modelado de negocio hasta el despliegue del producto, permite además generar diagramas de clases. [35]

Se integra a la Plataforma Java, la cual funciona en sistemas operativos Windows, Linux y Mac OS X. La Integración es vista con ambientes inteligentes de desarrollo (SDE) para Eclipse, NetBeans, Oracle JDeveloper, JBuilder, entre otros.

JBoss Tools

JBoss Tools es una colección de plugins que se le añaden a Eclipse, los mismos brindan una serie de funcionalidades, además de poder ser añadidos a diversos servidores de aplicación. Entre los plugins se puede hacer mención a: [36]

- **Hibernate tools:** Sirve de apoyo para la utilización de los componentes del framework Hibernate, así como para el mapeo de archivos y anotaciones.

- **Seam tools:** Incluye soporte para la vinculación de los componentes del framework Seam.
- **RichFaces VE:** Es un editor visual que incluye soporte para las librerías JSF incluyendo JBoss RichFaces.
- **JBoss AS tools:** Contiene funciones para el despliegue eficiente de cualquier tipo de proyecto en el IDE.
- **JBPM tools:** Permite la edición del flujo de trabajo de JBPM, además de funcionar como motor de procesos del mismo.

Se puede concluir con el estudio de los Sistemas de Información Hospitalaria asociados al campo de acción, no cumplen con todos los requisitos identificados que requiere el área de Farmacia Extrahospitalaria. Por lo que se evidenció la necesidad de desarrollar funcionalidades en la que los funcionarios de esta área despacharan medicamentos a pacientes que no se encuentren hospitalizados ni en emergencia. La aplicación se debe encontrar debidamente documentada para una versión futura y desarrollada con herramientas y tecnologías no privativas. Se justificó el uso de los patrones de diseño y el empleo de las tendencias y metodologías propuestas para la obtención de las funcionalidades descritas.

CAPÍTULO 2: DESCRIPCIÓN DE LA ARQUITECTURA

En el presente capítulo se hace referencia a las características que hacen al producto atractivo, usable, rápido o confiable, las cuales corresponden a la especificación de requerimientos no funcionales. Se realiza un examen de la arquitectura del sistema a desarrollar y el análisis de posibles implementaciones, componentes o módulos ya existentes y que puedan ser rehusados; así como identificar las diferentes estrategias de integración, de codificación y se argumenta sobre la seguridad que deber presentar la creación del software.

2.1 Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. En muchos casos son fundamentales para el éxito del mismo. Normalmente están vinculados a los requerimientos funcionales, es decir, una vez se conozca lo que el sistema debe hacer se puede determinar cómo debe comportarse, qué cualidades debe tener o cuán rápido o grande debe ser. [37]

Los requerimientos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se tiene conocimiento que cumple con todas las funcionalidades requeridas, estas propiedades pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

Existen diversas categorías para clasificar a los requerimientos no funcionales, entre las más usadas en el software en desarrollo, se encuentran:

Requerimiento de Hardware:

- **Estaciones de trabajo**

En la solución se incluyen estaciones de trabajo para las consultas del Sistema de Información Hospitalaria alas HIS. Estas necesitan capacidad de hardware que permita un sistema operativo, el cual soporte un navegador actualizado y que siga los estándares, se recomienda sea, IE7, Firefox 2 o

versiones superiores. Se escogieron por ello, estaciones de trabajo de 256 Mb de RAM y un microprocesador de 2.0 Hz y con sistema operativo Linux.

- **Servidores**

La solución estará conformada, fundamentalmente, por servidores de alta capacidad de procesamiento y redundancia, que permitan garantizar movilidad y residencia de la información y las aplicaciones bajo esquemas seguros y confiables.

Se dispone de *Servidores de Base de datos: 1 DL380 G5, Procesador Intel® Xeon® 5140 Dual - Core 4GB de memoria y 2x72GB de disco y sistema operativo Linux.*

Servidores de Aplicaciones: 2 DL380 G5, Procesador Intel® Xeon® 5140 Dual - Core 4GB de memoria y 2x72GB de disco y sistema operativo Linux.

Y con *Servidores de Intercambio: 1 DL380 G5, Procesador Intel® Xeon® 5140 Dual - Core 2 GB de memoria y 2x72GB de disco y sistema operativo Linux.*

Requerimientos de Software:

La aplicación debe poder ser desplegada en sistemas operativos Windows, Unix y Linux, utilizando la plataforma Java (Java Virtual Machine, JBoss AS y PostgreSQL). Los usuarios deberán disponer de un navegador Web, estos pueden ser IE 7, Opera 9, Google Chrome 11 y Firefox 3 o versiones superiores de estos.

Usabilidad:

El sistema estará diseñado de manera que los usuarios adquieran las habilidades necesarias para explotarlo, en un tiempo reducido, de forma tal que los usuarios normales puedan utilizarlo en 30 días y los usuarios más avanzados en 20 días.

La usabilidad se refiere a la capacidad de un software de ser comprendido, usado y ser atractivo para el usuario, en condiciones específicas de uso; pero no depende sólo del producto, sino también del usuario,

de la facilidad con que este interactúe con la herramienta, en menos pasos o más naturales a su formación específica.

Rendimiento:

El sistema minimizará el volumen de datos en las peticiones y además optimizará el uso de recursos críticos como la memoria y a la vez respetará buenas prácticas de programación para incrementar el rendimiento en operaciones costosas para la máquina virtual como la creación de objetos.

Soporte:

Los requerimientos de soporte abarcan todas las acciones a tomar una vez que se ha terminado el desarrollo del software con motivos de asistir a los clientes de este así como lograr su mejoramiento progresivo y evolución en el tiempo.

Se permitirá la creación de usuarios, otorgamiento de privilegios y roles así como asignación de perfiles.

Se permitirá realizar copias de seguridad de la base de datos hacia otro dispositivo de almacenamiento externo, además de recuperar la base de datos a partir de los respaldos realizados y el chequeo de las operaciones y acceso de los usuarios al sistema. Así como establecer parámetros de configuración del sistema y actualización de nomencladores.

Portabilidad:

La aplicación podrá ser desplegada sobre los Sistemas Operativos *Linux o Windows*.

Seguridad:

La seguridad es con pocas palabras garantizar la confidencialidad, integridad y disponibilidad del sistema. Es prácticamente imposible mantener un sistema seguro al 100 % en todas las circunstancias. La seguridad es un aspecto que debe vigilarse constantemente, pues pueden ocurrir grandes riesgos que resulten significativos para el sistema.

Se mantendrá seguridad y control a nivel de usuario, con el objetivo de garantizar el acceso de los mismos sólo a los niveles establecidos de acuerdo a la función que realizan. Las contraseñas podrán cambiarse sólo por el propio usuario o por el administrador del sistema. Se registrarán todas las acciones que se realizan, llevando el control de las actividades de cada usuario en todo momento. El sistema proporcionará un registro de actividades (log) de cada usuario en el sistema. Ninguna información que se haya ingresado en el sistema será eliminada físicamente de la base de datos. Y por último, el sistema permitirá la recuperación de la información de la base de datos a partir de los respaldos o salvas realizadas.

Restricciones en el diseño y la implementación:

La Capa de presentación contendrá todas las vistas y la lógica de la presentación. El flujo Web se manejará de forma declarativa. La capa del negocio mantendrá el estado de las conversaciones y procesos del negocio que concurrentemente pueden ser ejecutados por cada usuario, mientras que por su lado la capa de acceso a datos comprende las entidades y los objetos de acceso a datos correspondientes a las mismas; el acceso a datos está basado en el estándar JPA y particularmente en la implementación del motor de persistencia de Hibernate.

Interfaz:

- **Interfaces de usuario**

Las ventanas del sistema contendrán los datos legibles y bien estructurados, además de permitir la correcta interpretación de la información. La interfaz dispondrá con teclas de función y menús desplegables que faciliten y aceleren su utilización. La entrada de datos incorrecta será detectada claramente e informada al usuario.

2.2 Descripción de la arquitectura

La arquitectura de software se centra en la idea de reducir la complejidad a través de abstracciones y patrones que permiten el desarrollo del software. El origen de esta disciplina como concepto fue identificado en el trabajo de investigación de Edsger Dijkstra en 1968 y David Parnas a principios de los

años 70. En su forma más simple, la arquitectura es la estructura jerárquica de los componentes del programa, la forma en que los componentes interactúan y la estructura de datos que van a utilizar éstos.

Para la implementación de este sistema se utiliza la arquitectura basada en el patrón Modelo Vista Controlador. Convierte una aplicación en un paquete modular y de desarrollo rápido. La modularidad y el diseño independiente permiten a los diseñadores y desarrolladores hacer cambios en alguna parte de la aplicación sin afectar a los demás. Es muy usado para la creación de aplicaciones web porque separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres áreas esenciales: el modelo, es el Sistema de Gestor de Base de Datos, la vista, muestra la información al usuario y el controlador contiene las reglas de gestión de eventos.

La capa de presentación permite la navegabilidad del sistema ya que dispone de páginas XHTML, esta se integra fácilmente con el framework Seam, brinda la posibilidad de generar vistas, conversión y validación de campos, desarrollada con JSF, usando RichFaces como librería de componentes. La utilización de estos componentes nutre el diseño de la interfaz de usuario y se obtiene una mayor naturalidad en la interacción con el sistema.

La capa de negocio implementa procesos de negocio identificados durante el análisis funcional mediante clases controladoras encargadas del manejo y validación de los datos capturados en la capa de presentación, además de brindar la posibilidad de especificar el contexto, puede ser conversacional, evento, página, a través de las anotaciones que Seam ofrece. En esta capa se identifican las reglas del negocio, llamadas Drools, las cuales logran un mayor dinamismo.

La capa de acceso a datos, mediante el uso de los componentes Hibernate, se encarga de modificar, eliminar y persistir la información existente en la base de datos. Como ventaja fundamental de la utilización de estos componentes se logra la abstracción del programador con respecto al gestor de base de datos, mediante clases mapeadas y consultas en lenguaje orientado a objetos.

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

La práctica de reutilización de código fuente trae ventajas para toda aplicación que se encuentre en desarrollo y permite principalmente reducir tiempo. La forma más eficiente de la reutilización de código es la creación de componentes reutilizables para evitar la duplicidad del mismo.

Consulta Externa mantiene estrecha relación con el módulo Farmacia Extrahospitalaria, debido a que el mismo se encarga de atender a los pacientes que no se encuentran hospitalizados ni en emergencia, prescribe los medicamentos indicados y envía esta lista a la Farmacia al Módulo Farmacia Extrahospitalaria, en este se efectúa el proceso de Despachar medicamento a pacientes.

Durante el proceso de despacho de productos, se crean los vales correspondientes a los medicamentos vendidos, estos vales pasan al área de Facturación con el objetivo de contabilizar de forma óptima las ganancias de la farmacia por el concepto de ventas. Para desarrollar las funcionalidades relacionadas a este proceso se utilizaron componentes entre los que se encuentra: EntityManager; la clase IActiveModule que facilita la información sobre el módulo y la entidad, la cual es manejada por el usuario que trabaja en el sistema; JaspertReport para la generación de reportes; ListadoControler y GenericData con el propósito de brindar los datos del usuario autenticado en el sistema junto a otras funcionalidades.

El Módulo Almacén se relaciona de forma directa a la Farmacia Extrahospitalaria. Después de realizar cada venta es necesario efectuar una rebaja de los medicamentos disponibles y satisfacer las órdenes de reabastecimiento de las farmacias asociadas.

2.4 Seguridad

En aras de garantizar una correcta protección de los datos de los usuarios asociados a la Farmacia Extrahospitalaria, así como los medicamentos dispensados en la misma, se propone un control de acceso a nivel de usuario y contraseña, con el propósito de obtener el principio de mínimo privilegio, lo que asegura el acceso de cada usuario a los lugares de la aplicación asignados a su uso. A través de una

bitácora de sucesos del sistema, se llevará un registro de las acciones realizadas por el usuario en todo momento.

Todo tipo de autorización está basada en las reglas del negocio y las mismas no se encuentran incrustadas en el código de la aplicación, con el objetivo de lograr una independencia total gracias a la integración existente entre Seam Security Framework y el potente motor de reglas JBoss Rules.

2.5 Vista de Despliegue

La vista de despliegue define la arquitectura física del sistema por medio de nodos interconectados. Estos nodos son elementos hardware sobre los cuales pueden ejecutarse los elementos software.

El modelado de la vista de despliegue puede implicar modelar la topología del hardware sobre el que se ejecuta el sistema. Aunque UML no es un lenguaje de especificación hardware de propósito general, se ha diseñado para modelar muchos de los aspectos del hardware de un sistema, a un nivel suficiente para que un ingeniero de software, pueda especificar la plataforma sobre la que se ejecutará el sistema. La figura a continuación representa el modelo de despliegue correspondiente al módulo Farmacia Extrahospitalaria.

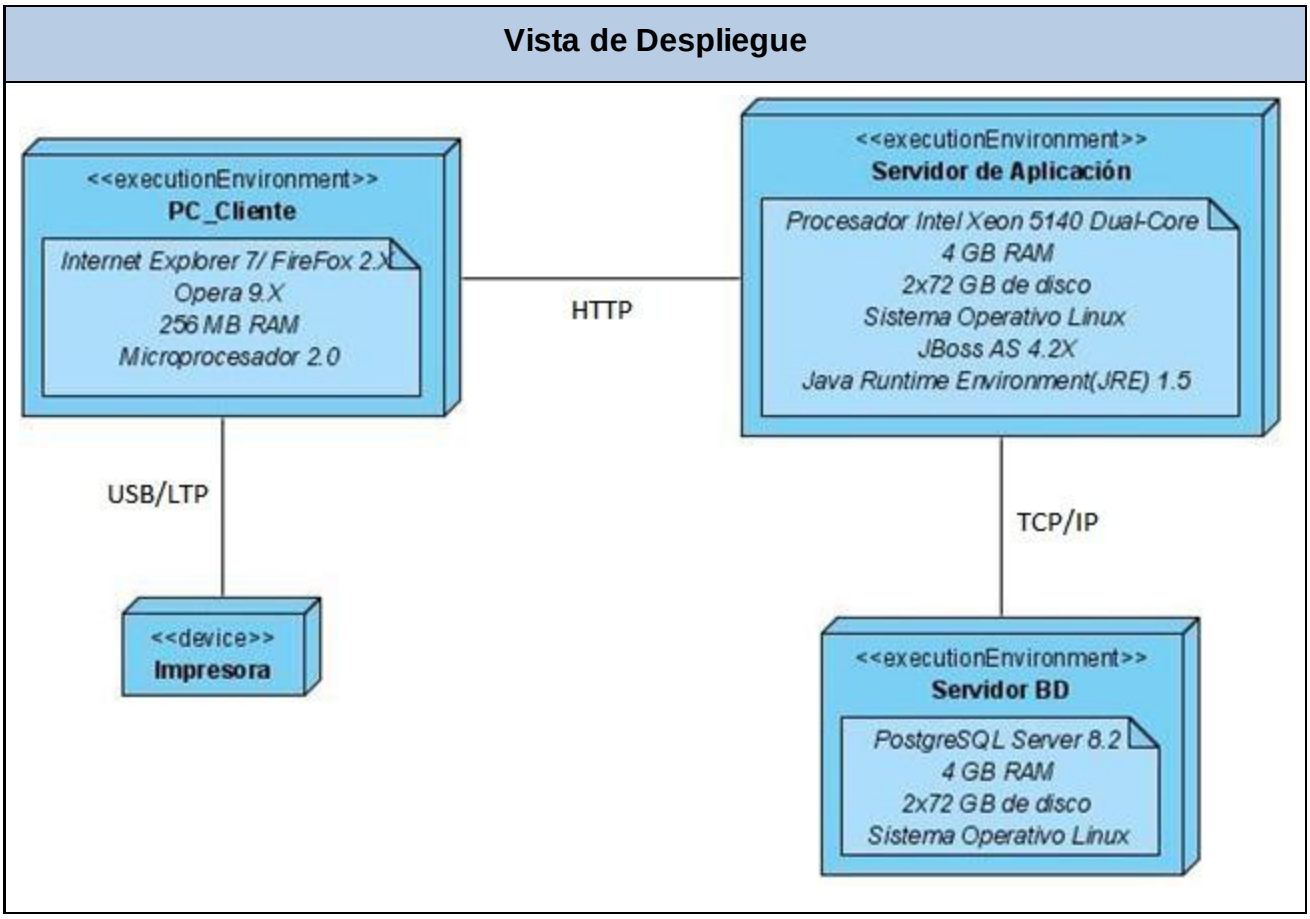


Tabla 2.1. Diagrama de Despliegue

2.6 Estrategias de codificación. Estándares y estilos a utilizar

Un estándar de codificación es un conjunto de directrices, normas y reglamentos sobre la forma de escribir código. Por lo general, un estándar de codificación incluye pautas sobre cómo nombrar variables, la forma de guión, el código, cómo poner entre paréntesis y palabras clave. La idea es ser constante en la programación de modo que, en caso de que varias personas se encuentren trabajando en el mismo código, posibilite entender lo que otros han hecho. La legibilidad del código es sólo uno de los aspectos de mantenimiento del programa.

Identación

1. Inicio y fin de bloque

   Se recomienda dejar dos espacios en blanco desde la instrucción anterior para el inicio y fin de bloque {}. Lo mismo sucede para el caso de las instrucciones if, else, for, while, do while, switch, foreach.

2. Aspectos generales

   El identado debe ser de dos espacios por bloque de código. No se debe usar el tabulador; ya que este puede variar según la computadora o la configuración de dicha tecla. Los inicios ({) y cierre (}) de ámbito deber estar alineados debajo de la declaración a la que pertenecen y deben evitarse si hay sólo una instrucción. Nunca colocar { en la línea de un código cualquiera, esto requiere una línea propia.

Variables y constantes

3. Apariencia de variables

   Las variables tendrán un prefijo para el tipo de datos en minúscula. El nombre que se le da a las variables debe comenzar con la primera letra en minúscula, la cual identificará el tipo de datos al que se refiere. En caso de que sea un nombre compuesto se empleará notación CamellCasing**. Ejemplo: sNombrePaciente.

4. Apariencia de constantes

   Las constantes deben declararse con todas sus letras mayúsculas.

5. Aspectos generales

   El nombre empleado para las variables y constantes, debe permitir que con sólo leerlo se conozca el propósito de la misma.

Comentarios, separadores, líneas, espacios en blanco y márgenes.

6. Ubicación de comentarios.

   Al inicio de cada clase o función y al final de cada bloque de código.

   Se recomienda comentar al inicio de la clase o función de forma que se especifique el objetivo de la misma así como los parámetros que usa (declarar tipos de dato, y objetivo del parámetro) entre otras cosas.

7. Líneas en blanco

   Se emplean antes y después de métodos, clases y estructuras.

   Se recomienda dejar una línea en blanco antes y después de la declaración de una clase o de una estructura y de la implementación de una función.

8. Espacios en blanco

   Entre operadores lógicos y aritméticos.

   Se recomienda usar espacios en blanco entre estos operadores para lograr una mayor legibilidad en el código. Ejemplo: medicamento = nommedicamento

9. Aspectos generales

   Sobre el comentario:

   Se debe evitar comentar cada línea de código. Cuando el comentario se aplica a un grupo de instrucciones debe estar seguido de una línea en blanco. En caso de que se necesite comentar una sola instrucción se suprime la línea en blanco o se escribe a continuación de la instrucción.

Sobre los espacios en blanco:

No se debe usar espacio en blanco. Después del corchete abierto y antes del cerrado de un arreglo. Luego del paréntesis abierto y antes del cerrado. Antes de un punto y coma.

Clases y Objetos

10. Apariencia de clases y objetos

    Primera letra en mayúscula. Los nombres de las clases deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing*. Ejemplo: MiClase(). Para el caso de las instancias se comenzará con un prefijo que identificará el tipo de dato, este se escribirá en minúscula.

11. Apariencia de atributos

    Primera letra en minúscula. El nombre que se le da a los atributos de las clases debe comenzar con la primera letra en minúscula, la cual estará en correspondencia al tipo de dato al que se refiere, en caso de que sea un nombre compuesto se empleará notación CamellCasing*.

12. Apariencia de las funciones

    Primera letra en mayúscula. Para nombrar las funciones se debe tratar de utilizar verbos que denoten la acción que hace la función. Se empleará notación CamellCasing*. Ejemplo: function buscarUnidad(). Si son funciones que obtienen un dato se emplea el prefijo get y si fijan algún valor se emplea el prefijo set.

13. Declaración de parámetro en funciones

    Agrupados por tipos. Poner los string 1 numéricos 2, además, agrupar según valores por defecto. Los parámetros que se le pasan a las funciones se recomienda sean declarados de forma tal que estén agrupados por el tipo de dato que contienen.

14. Aspectos generales

    Sobre las clases, los objetos, los atributos y las funciones. El nombre empleado para las clases, objetos, atributos y funciones debe permitir que con sólo leerlo se conozca el propósito de los mismos.

Bases de Datos, Tablas, esquemas y Campos

15. Apariencia de la base de datos

    Las 2 primeras letras representan el tipo. Los nombres de las Bases de Datos deben comenzar con el prefijo bd a continuación underscoard y luego el nombre completamente en minúscula, en caso de que sea un nombre compuesto se empleará notación. Los nombres serán cortos y descriptivos.

16. Apariencia de las vistas

    Las 2 primeras letras representan el tipo. Todas las letras en minúscula. El nombre a emplear para las vistas deben comenzar con el prefijo vt seguido de underscoard y el nombre debe escribirse con todas las letras en minúscula para evitar problemas con el Case Sensitive del gestor.

    Ejemplo:

    create view vt_finanzas".

17. Apariencia de las tablas

    Las 2 primeras letras representan el tipo. Todas las letras en minúscula. El nombre a emplear para las tablas debe comenzar con el prefijo tb seguido de underscoard y luego debe escribirse todas las letras en minúscula, en caso de que sea un nombre compuesto se utilizara underscoard para separarlo.

18. Tablas que representen Relaciones

    Las 2 primeras letras representan el tipo. Todas las letras en minúscula. El nombre a emplear para estas tablas de relación debe comenzar con el prefijo tr seguido de underscoard y el nombre de será la concatenación del nombre de las dos tablas que la generaron separados por uderscoard todo en minúscula.

19. Tablas que representen nomencladores

    Las 2 primeras letras representan el tipo. Todas las letras en minúscula. El nombre a emplear para estas tablas de relación debe comenzar con el prefijo tn seguido de underscoard. El nombre de será corto y descriptivo todo en minúscula.

20. Apariencia de los procedimientos almacenados

    Las 2 primeras letras representan el tipo. Todas las letras en minúscula. En los procedimientos el nombre debe comenzar con el prefijo pa, underscoard y luego todas las letras en minúscula en caso de que sea un nombre compuesto se utilizara underscoard para separarlo.

21. Apariencia de los campos

    Todas las letras en minúscula. El nombre a emplear para los campos debe escribirse con todas las letras en minúscula para evitar problemas con el Case Sensitive del gestor.

22. Nombre de los campos

    En caso de identificadores. Todos los campos identificadores van a comenzar con el identificador id seguido de underscoard y posteriormente el nombre del campo.

23. Sentencias SQL

    Todas las letras en mayúscula. Las palabras correspondientes a las sentencias SQL y sus parámetros deben ir en mayúsculas.

24. Aspectos generales

    Sobre las bases de datos, vistas, tablas atributos y procedimientos. El nombre empleado para las Bases de Datos, las vistas, las tablas, los campos y los procedimientos almacenados, deben permitir que con sólo leerlos se conozca el propósito de los mismos.

Controles

25. Apariencia de los controles

    Los controles tendrán un prefijo para el tipo de datos en minúscula. El nombre que se le da a los controles deben comenzar con las primeras letras en minúscula, las cuales identificaran el tipo de datos al que se refiere, en caso de que sea un nombre compuesto se empleará notación CamellCasing**.

En el presente capítulo se llevó a cabo la descripción de unos de los aspectos fundamentales en la construcción de un software su arquitectura. Además se especificaron los requerimientos no funcionales que hacen atractivo y usable al sistema. Además se realizó el análisis de posibles implementaciones y componentes reutilizables con vista a facilitar la implementación de las funcionalidades en un menor tiempo de desarrollo.

CAPITULO 3: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA

En este capítulo se realiza una valoración detallada del diseño, argumentándolo con los diagramas de clases del diseño y de interacción, como el de secuencia, que son una primera vista para el modelo de implementación. Se presenta el modelo de datos, el diagrama de paquetes, el diagrama de componentes con el objetivo de mostrar la estructura física de la aplicación propuesta. Se describen las clases u operaciones necesarias y las entidades relacionadas en cada diagrama representado.

3.1 Valoración crítica del diseño propuesto por el analista

A través de las actividades contempladas en el análisis se representa una vista interna del sistema en la que, usando el lenguaje de los desarrolladores se refina los requisitos y se estructura en base a clases y paquetes. Este proceso se continúa haciendo en el diseño con el propósito de obtener los objetos que interactúan para cumplir los requisitos funcionales y no funcionales descritos. La actividad del diseño se refiere al establecimiento de las estructuras de datos, la arquitectura general del software, representaciones de interfaz y los componentes.

Por lo general, durante su elaboración se usan patrones. Los patrones de diseño son aquellos que expresan esquemas para definir relaciones entre clases y objetos con las que construir sistemas software. Describen una estructura de diseño que resuelve un problema en caso de presentarse, lo que puede tener un impacto en la forma en la que se aplica y utiliza el patrón, esto le permite al diseñador decidir si es necesario o no emplearlo en el desarrollo actual. [38]

Para contribuir a la definición del diseño de la solución propuesta se tuvo en cuenta una serie de patrones, entre ellos se encuentran los *Patrones de Software para la Asignación General de Responsabilidad* (*GRASP, por sus siglas en inglés)*. Se utilizaron con el objetivo de asignarle a cada clase o componente las responsabilidades a cumplir, atendiendo a la información que este brinde o su comportamiento, así como la creación de instancias a otras clases. Dentro de este grupo de patrones se pueden mencionar: *Experto, Creador, Controlador.*

Otros de los patrones utilizados es el de *Bajo acoplamiento y Alta cohesión*, permitiendo la colaboración entre las clases sin afectar la reutilización de los mismos. La creación de páginas controladoras constituye un ejemplo más de su aplicación, como ventaja tiene, posibilitar realizar las operaciones del sistema, las cuales reflejan los procesos del negocio y resulta ser factible para ser manejadas en capas como la de interfaz o presentación.

Se usó además el patrón de creación *Abstract Factory*, este permite el acceso a la base de datos a través de EntityManager. Resulta por otro lado ser la interfaz principal de JPA con el objetivo de lograr la persistencia de las aplicaciones, es responsable de realizar operaciones como inserción, lectura, modificación y eliminación (CRUD, por sus siglas en inglés) sobre un conjunto de objetos persistentes.

Hibernate utiliza el patrón *ActiveRecord,* donde una fila de cualquier tabla de la base de datos se convierte en un objeto, de manera que se asocian filas únicas de la base de datos, con objetos del lenguaje de programación usado. Por él son implementados varios patrones que facilitan el mapeo, entre los que se encuentra *Identity Field*, el cual registra un campo identificador en un objeto manteniendo la identidad entre un objeto en memoria y una fila de una tabla en una base de datos.

Forein Key Mapping es otro patrón de comportamiento, permite establecer una asociación entre objetos para referencias de llave foránea en tablas de la base de datos, para el mapeo de relaciones de uno a mucho, *Association Table Mapping*, el cual posibilita el mapeo de relaciones de muchos a muchos. Un patrón de relevante importancia que no se debe dejar de mencionar implementado por Hibernate es *Query Object*. Este es encargado de interpretar la estructura de objetos y traducirlos a consultas SQL (Structured Query Lenguaje), de forma que facilite así el trabajo a los desarrolladores.

Elaborar los Diagramas de Diseño implica una mejor comprensión para iniciar la implementación, los mismos exponen un conjunto de interfaces, colaboraciones y sus relaciones. El Modelo de Diseño es un modelo de objetos que describe la realización de los casos de uso y es usado como una entrada inicial en las actividades de implementación y prueba. Es abarcador, compuesto por artefactos que engloban todas las clases del diseño, subsistemas, paquetes, colaboraciones, y las relaciones entre ellos.

Al igual los Diagramas de Interacción son considerados artefactos que gráficamente muestran las relaciones entre los objetos, incluyendo los mensajes que se pueden enviar entre ellos. Se utilizan no sólo

para modelar los aspectos dinámicos de los sistemas sino también para construir sistemas ejecutables por medio de ingeniería directa e inversa. Pueden utilizarse para visualizar, especificar, construir y documentar la dinámica de una sociedad particular de objetos, o se pueden utilizar para modelar un flujo de control particular de un caso de uso.

Este tipo de diagrama se encuentra dividido por los Diagramas de Colaboración y de Secuencia. Un Diagrama de Colaboración destaca la organización estructural de los objetos que envían y reciben mensajes. Los Diagramas de Secuencia se diferencian en dos características de los Diagramas de Colaboración, en primer lugar, la línea de la vida, la cual es la línea discontinua vertical que representa la existencia de un objeto a lo largo de un período de tiempo y en segundo lugar está el foco de control. El foco de control es un rectángulo delgado y estrecho que representa el período de tiempo durante el cual un objeto ejecuta una acción, bien sea directamente o a través de un procedimiento subordinado.

Para elaborar el diseño se definió una estructura de paquetes que permite dividir el sistema en fragmentos manejables para su implementación. Se emplea además el criterio de empaquetamiento por procesos y por clases, siguiendo la estructura de procesos definidos en el sistema. Los paquetes que hacen referencia a procesos se componen por subcarpetas que responden a las realizaciones de los casos de uso y los cuales contienen un diagrama de clases y los respectivos diagramas de secuencia.

Se crea un paquete *Repositorio de clases* que contiene tres subpaquetes: el de las *Vistas*, el de las *Sesiones* y el paquete de las *Entidades*.

El paquete de las *Vistas* contiene los contenidos Web referentes a las páginas clientes y los formularios que la componen.

En el de *Sesiones* se agrupan las clases controladoras autogeneradas por el entorno de desarrollo y las clases controladoras personalizadas.

El subpaquete de las *Entidades* contiene las *Entidades Autogeneradas* y las *Personalizadas*. Las *Entidades Autogeneradas* son obtenidas desde la base de datos mediante el ORM Hibernate (permite generar entidades desde la base de datos). Las *Entidades Personalizadas* son las modificadas o entidades que heredan de las autogeneradas y cambiadas para lograr un mejor funcionamiento.

A continuación se muestra el Diagrama de Paquetes del Diseño de la aplicación propuesta:

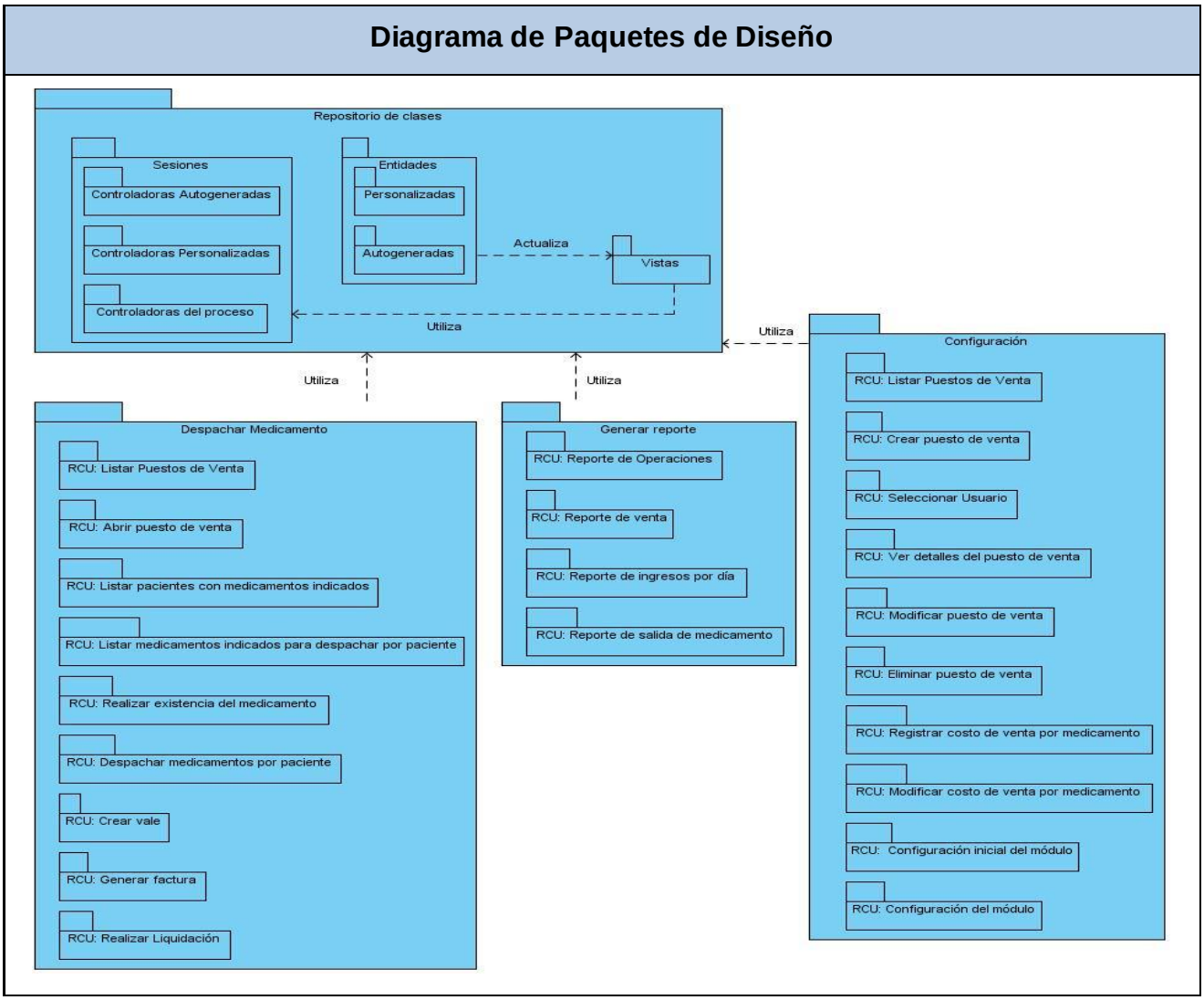


Tabla 3.1. Diagrama de Paquetes

Los Diagramas de Clases del Diseño se realizan mediante estereotipos Web que son una representación gráfica de los componentes a los cuales se hace referencia, se emplean para ello relaciones entre páginas y clases. En general muestran el flujo como una página servidora que construye una página cliente la cual a su vez contiene un formulario, el cual puede actualizar directamente a las entidades o enviar las

peticiones a la página servidora, estas invocan métodos y responsabilidades en las clases controladoras, las cuales a su vez pueden modificar las entidades.

A continuación se muestran algunas de las realizaciones de los casos de usos del sistema propuesto, relacionado con el proceso *Despachar medicamentos por paciente*.

DCD_ Despachar medicamentos por paciente

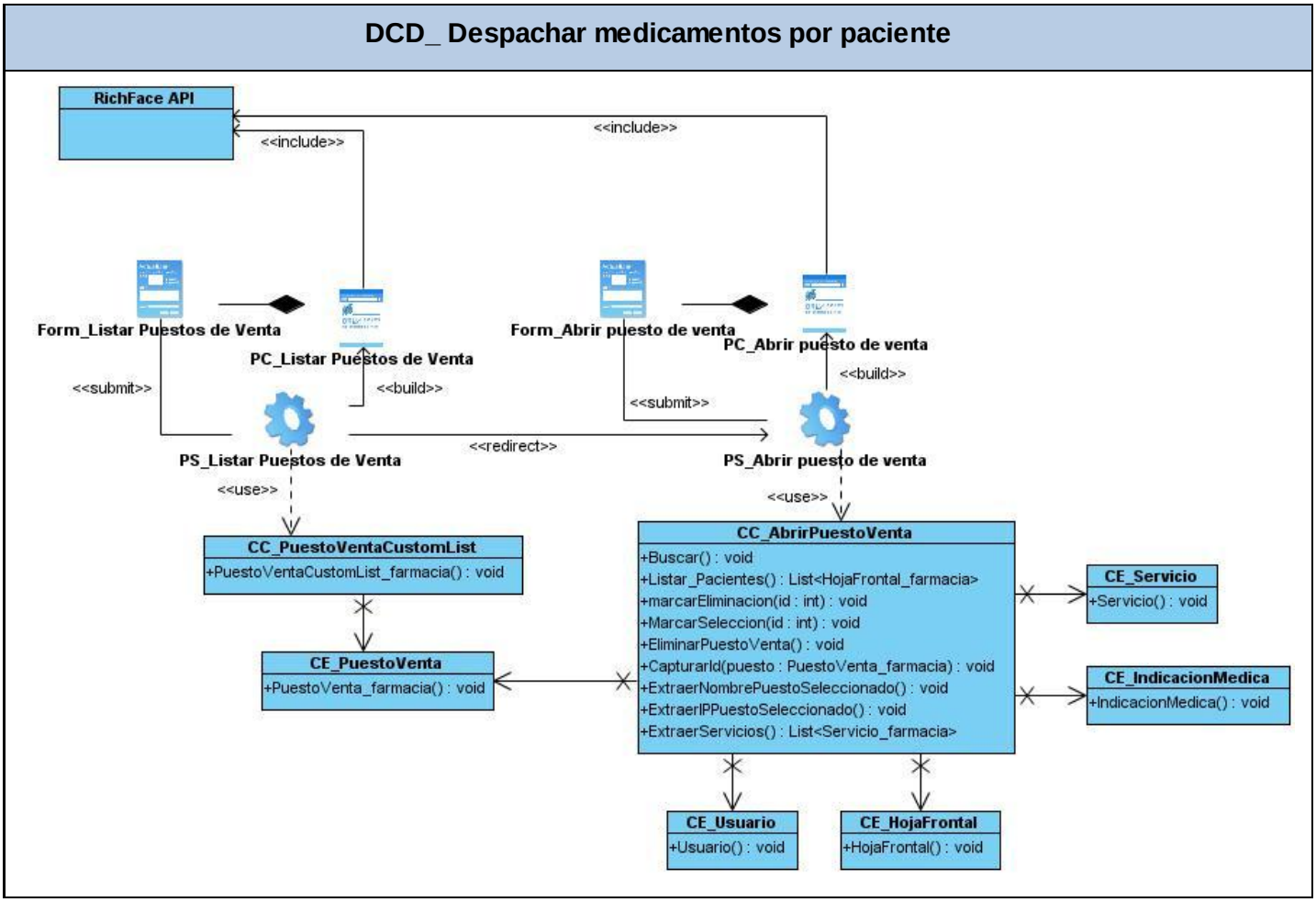


Tabla 3.2. DCD_ Despachar medicamentos por paciente

DCD_ Despachar medicamentos por paciente

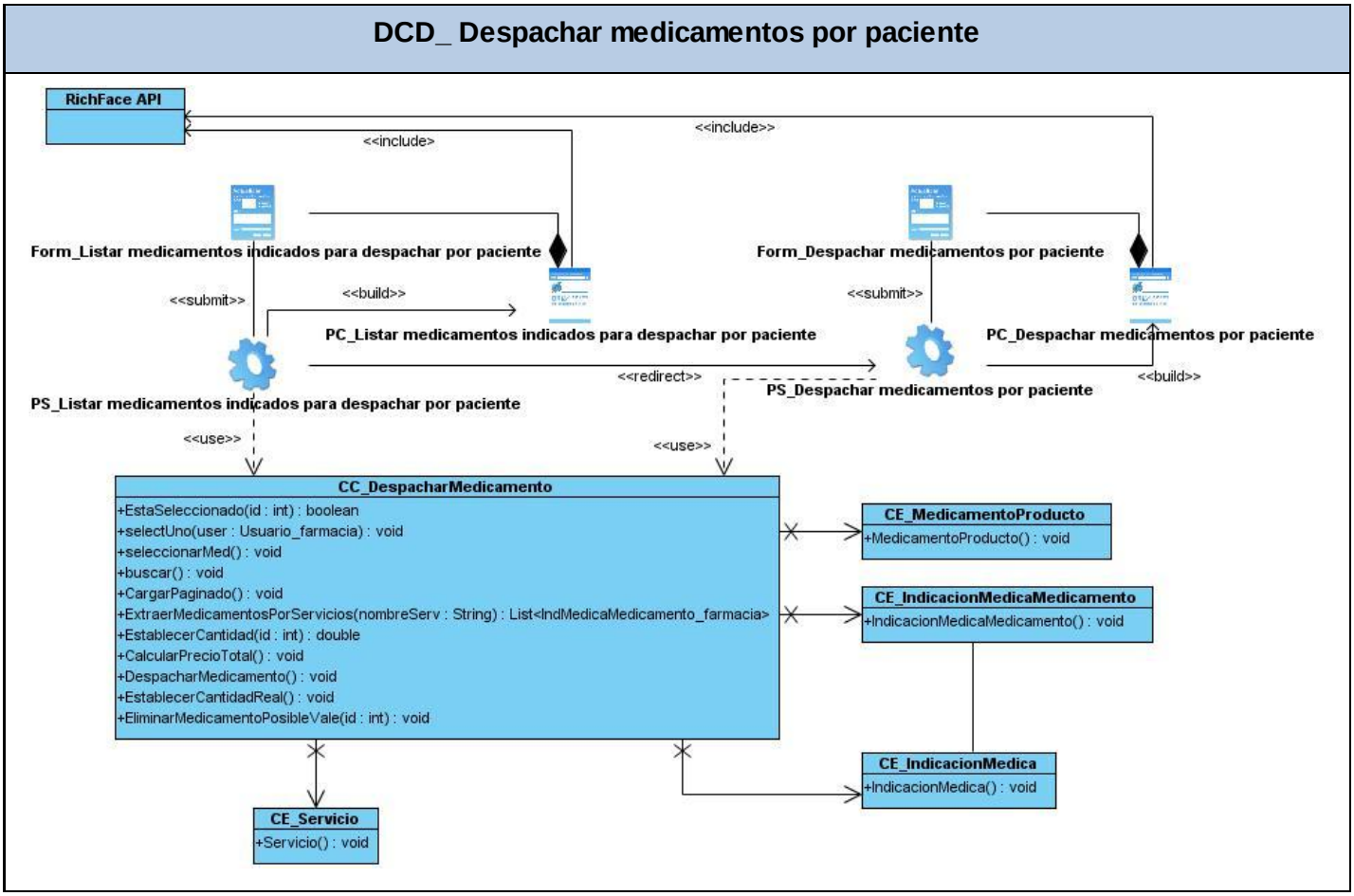


Tabla 3.3. DCD_ Despachar medicamentos por paciente

DCD_ Despachar medicamentos por paciente

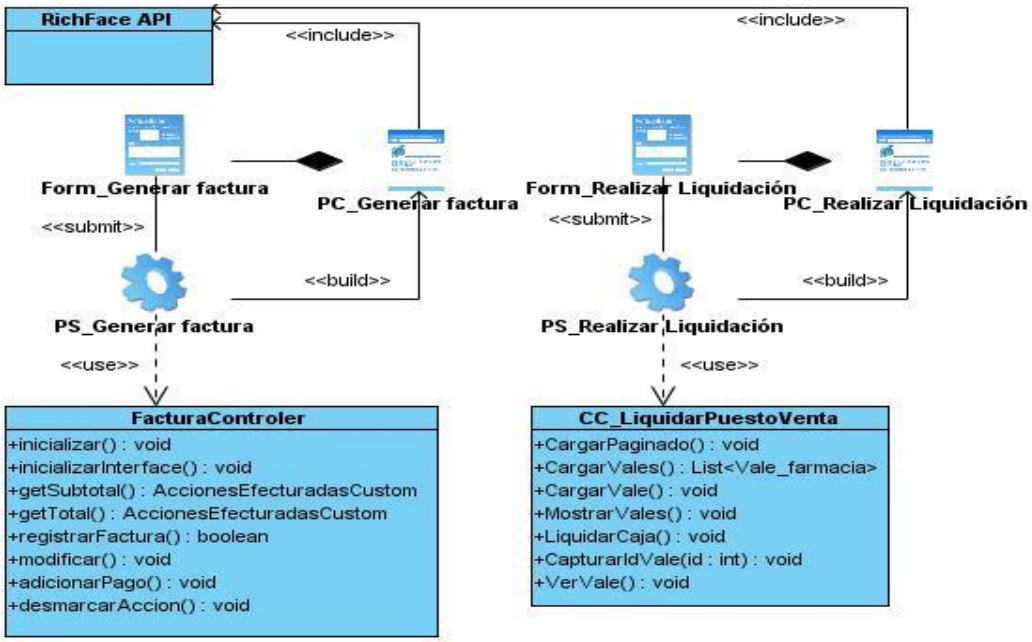


Tabla 3.4. DCD_ Despachar medicamentos por paciente

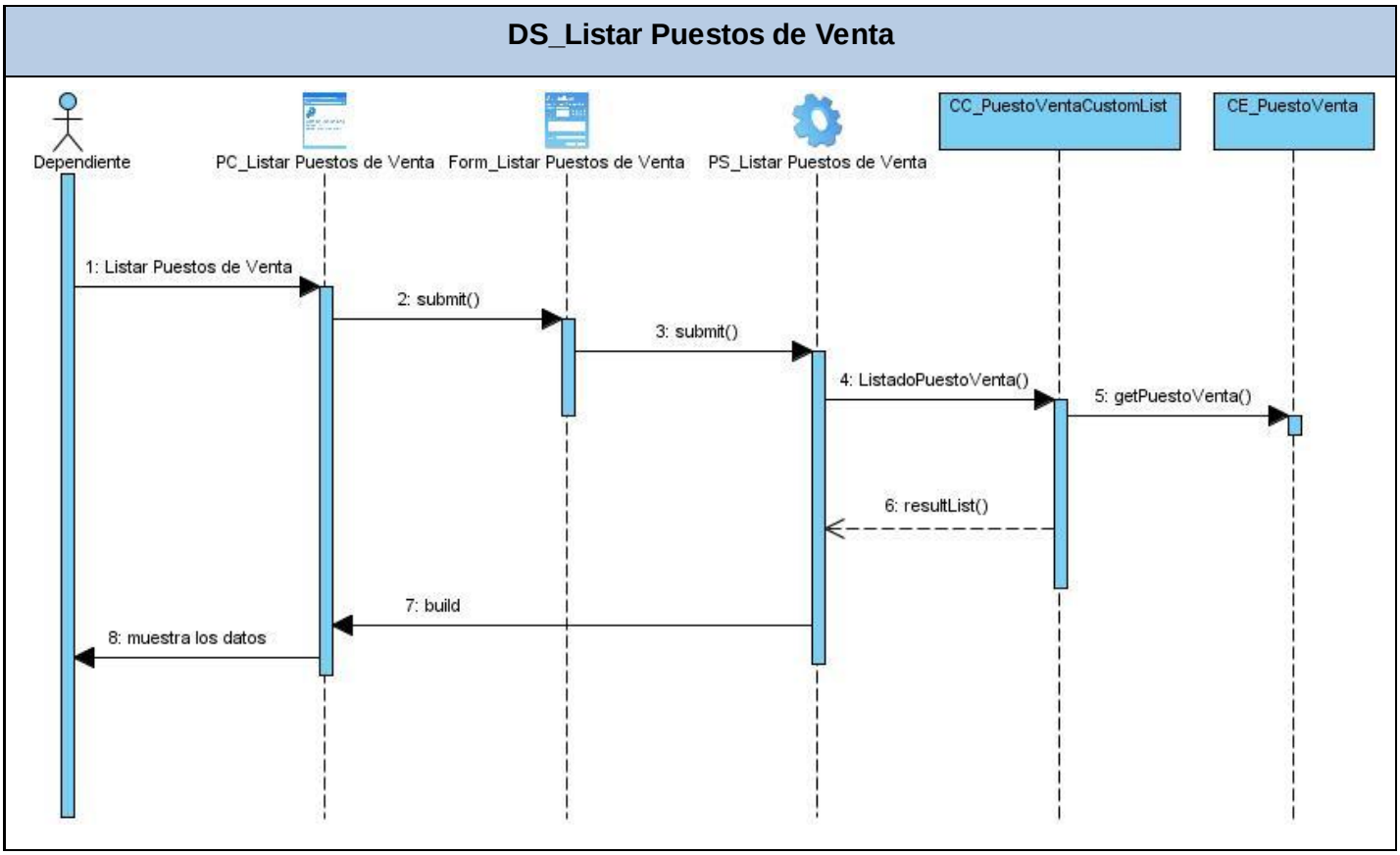


Tabla 3.5. DS_ Listar Puesto de Venta

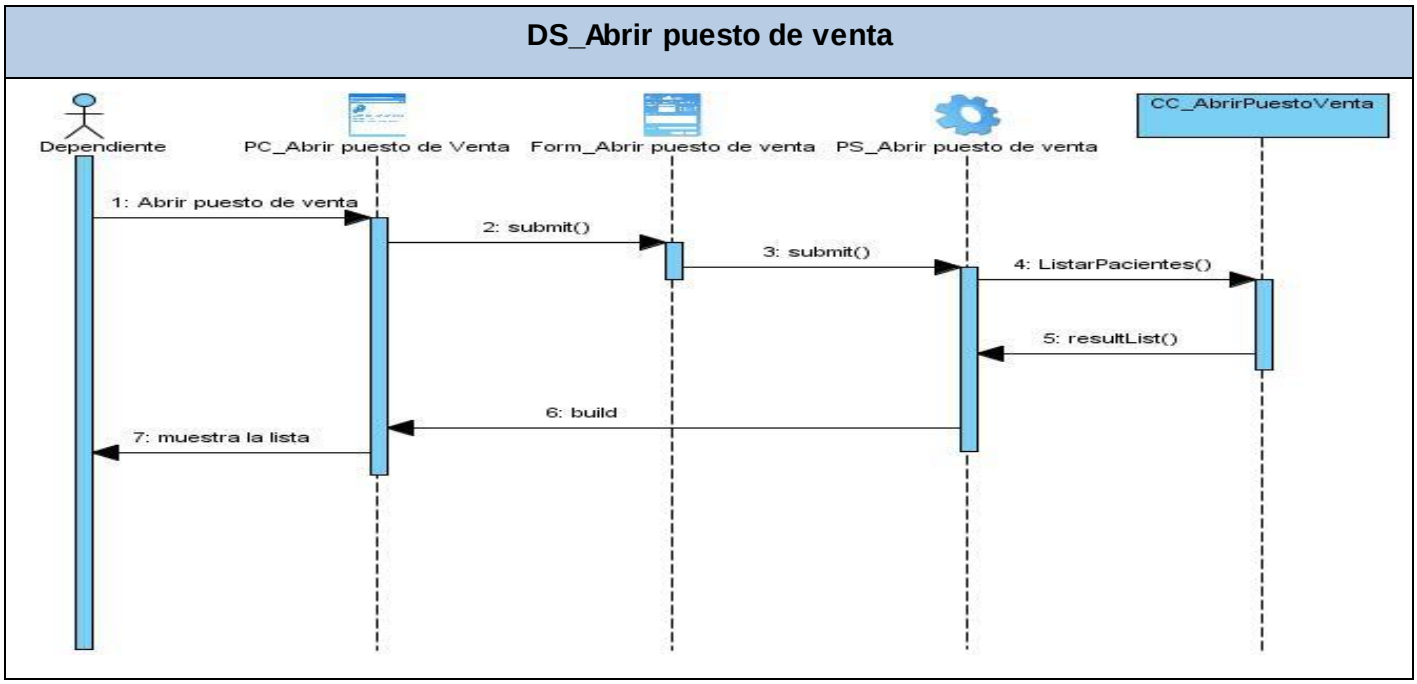


Tabla 3.6. DS_ Abrir puesto de venta

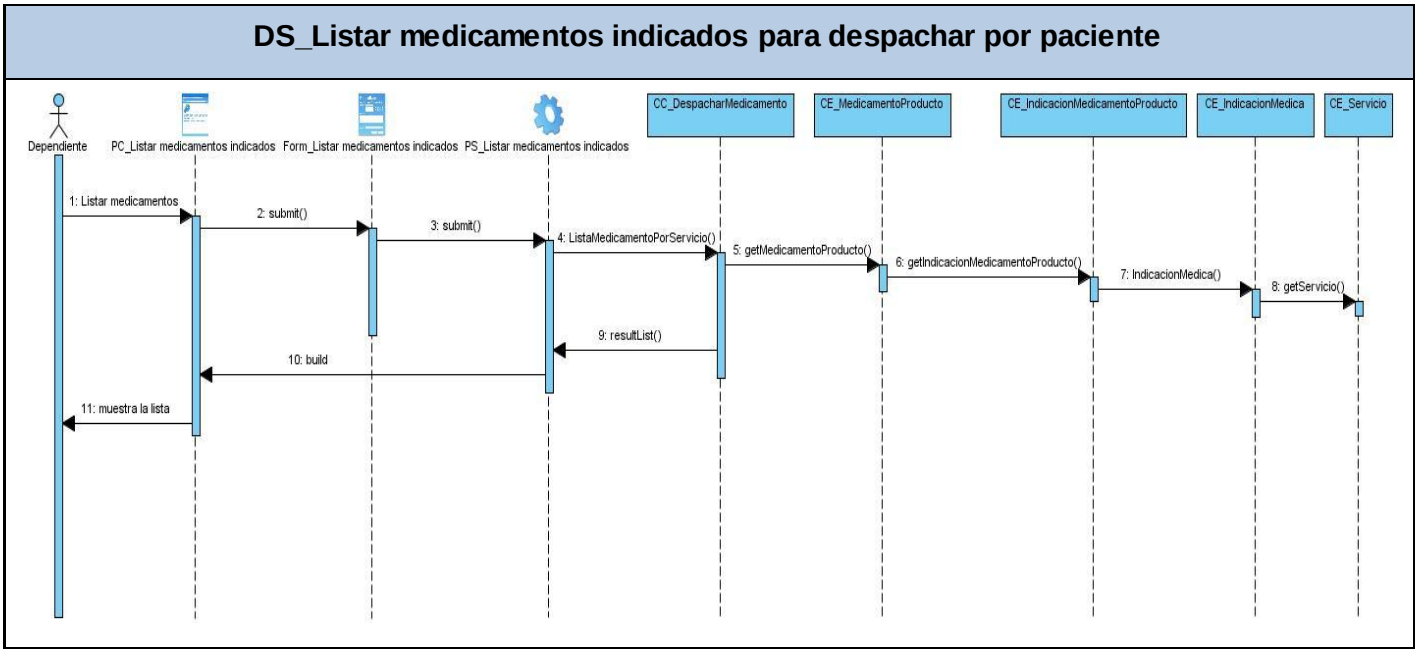


Tabla 3.7. DS_ Listar medicamentos indicados para despachar por paciente

DS_Despachar medicamentos por paciente

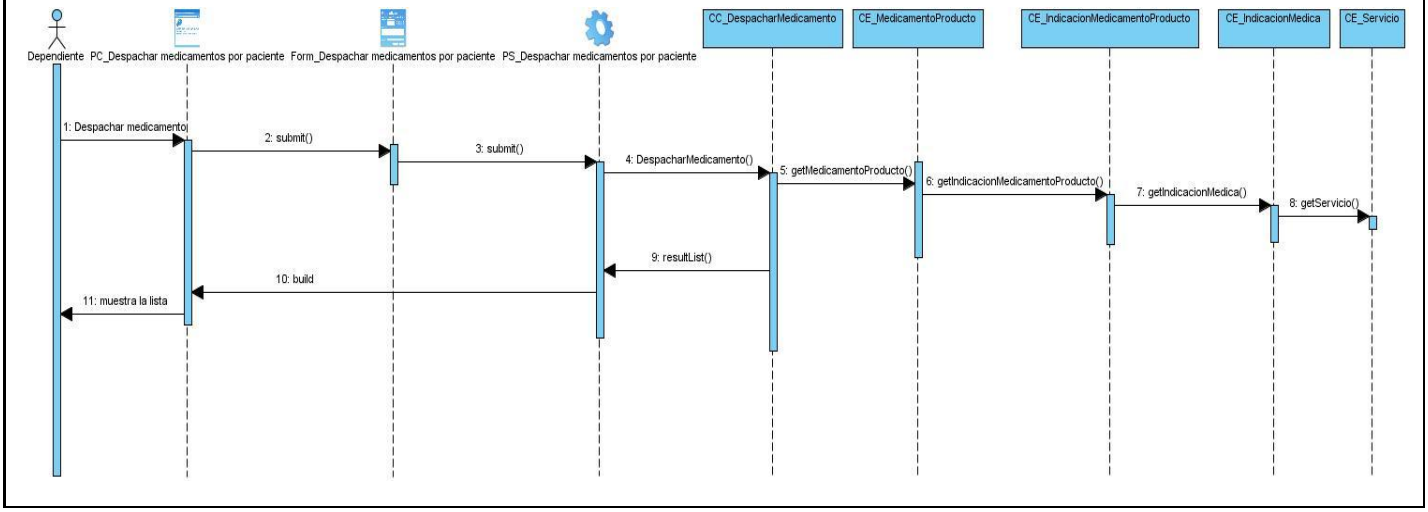


Tabla 3.8. DS_ Despachar medicamentos por paciente

DS_Generar factura

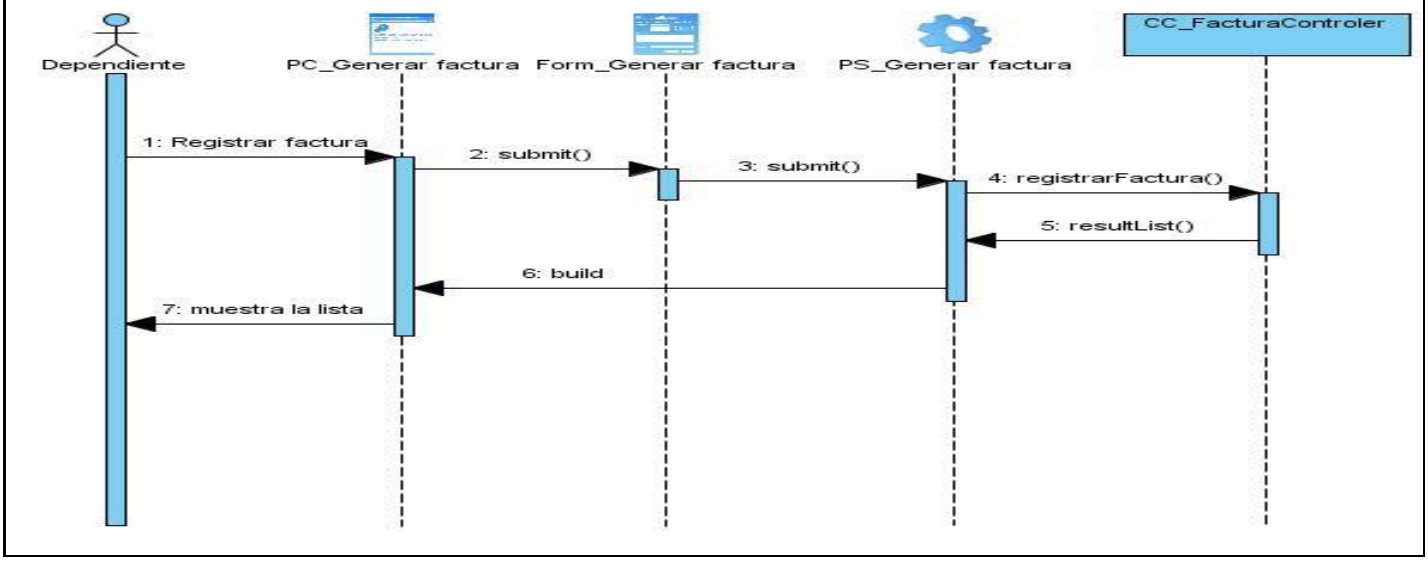


Tabla 3.9. DS_ Generar factura

DS_Realizar Liquidación

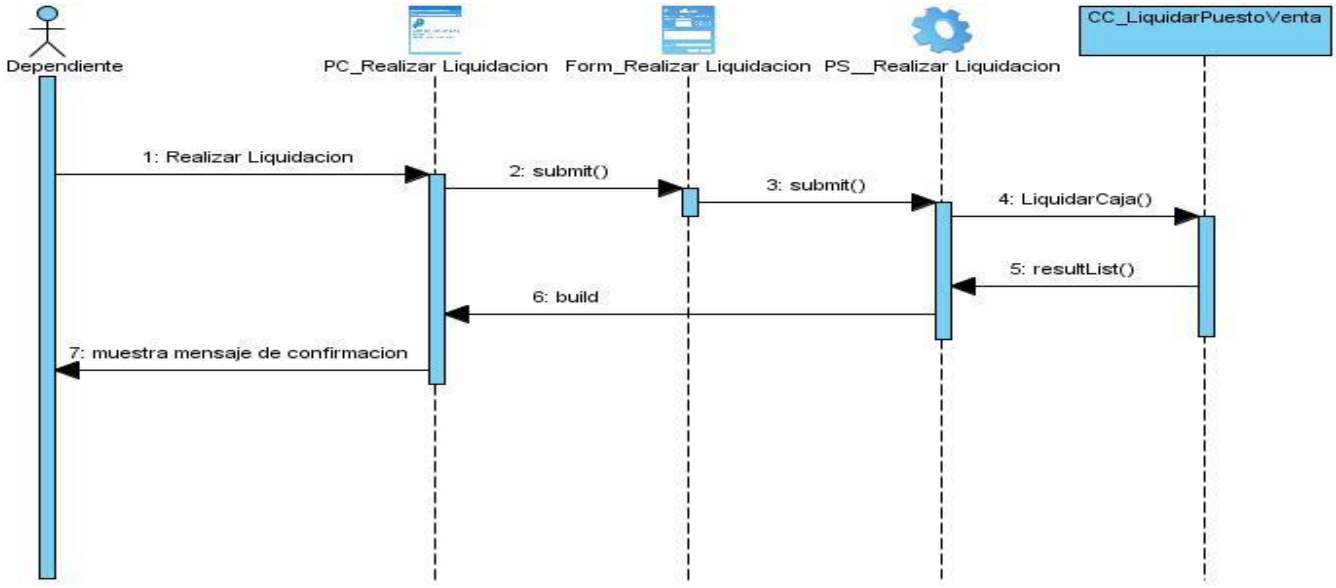


Tabla 3.10. DS_ Realizar Liquidación

3.2 Descripción de las clases u operaciones necesarias

Nombre: PuestoVentaCustomList	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	PuestoVentaCustomList_farmacia() : void
Descripción:	Inicializa los datos del puesto de venta

Tabla 3.11. Descripción de la clase PuestoVentaCustomList

Nombre: AbrirPuestoVenta	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Buscar() : void
Descripción:	Busca las indicaciones médicas por los criterios seleccionados.

Nombre:	ListarPacientes() : Listar<HojaFrontal_farmacia>
Descripción:	Lista los pacientes que presentan indicaciones médicas.

Tabla 3.12. Descripción de la clase AbrirPuestoVenta

Nombre: DespacharMedicamento	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Buscar() : void
Descripción:	Busca los medicamentos indicados por los criterios seleccionados.
Nombre:	ExtraerMedicamentosPorServicios(nombreServ : String) : List<IndMedicaMedicamento_farmacia>
Descripción:	Extrae los medicamentos indicados teniendo en cuenta el servicio seleccionado.
Nombre:	EstablecerCantidad(id : Integer) : Double
Descripción:	Establece la cantidad a despachar por receta.
Nombre:	CalcularPrecioTotal() : void
Descripción:	Calcula el precio total de los medicamentos a despachar.
Nombre:	DespacharMedicamento() : void
Descripción:	Despacha los medicamentos seleccionados.
Nombre:	EstablecerCantidadReal() : void
Descripción:	Establece la cantidad real de medicamentos a despachar al paciente.
Nombre:	EliminarMedicamentoPosibleVale(id : Integer) : void
Descripción:	Elimina los medicamentos seleccionados de una lista.

Tabla 3.13. Descripción de la clase DespacharMedicamento

Nombre: FacturaControler	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	inicializar() : void
Descripción:	Carga los datos de los medicamentos despachados para realizar la factura.
Nombre:	inicializarInterface() : void
Descripción:	Visualiza los datos de la Factura.
Nombre:	registrarFactura() : Boolean
Descripción:	Registra factura realizada.
Nombre:	adicionarPago() : void
Descripción:	Adiciona un tipo de pago a la factura, ya sea efectivo o a crédito.

Tabla 3.14. Descripción de la clase FacturaControler

Nombre: LiquidarPuestoVenta	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	CargarVales() : List<Vale_farmacia>
Descripción:	Cargar los vales creados por la venta de los medicamentos en ese puesto de venta.
Nombre:	LiquidarCaja() : void
Descripción:	Realiza la liquidación del puesto de venta.

Tabla 3.15. Descripción de la clase LiquidarPuestoVenta

3.3 Modelo de Datos

El Modelo de Datos describe la representación lógica y física de la información persistente manejada por el sistema. Puede ser inicialmente creado a través de la ingeniería inversa de un almacenamiento de datos persistentes que ya exista o puede ser inicialmente creado a partir de un conjunto de clases del

diseño persistentes en el modelo de diseño. Es usado para definir el mapeo entre las clases del diseño y las estructuras de datos. [39]

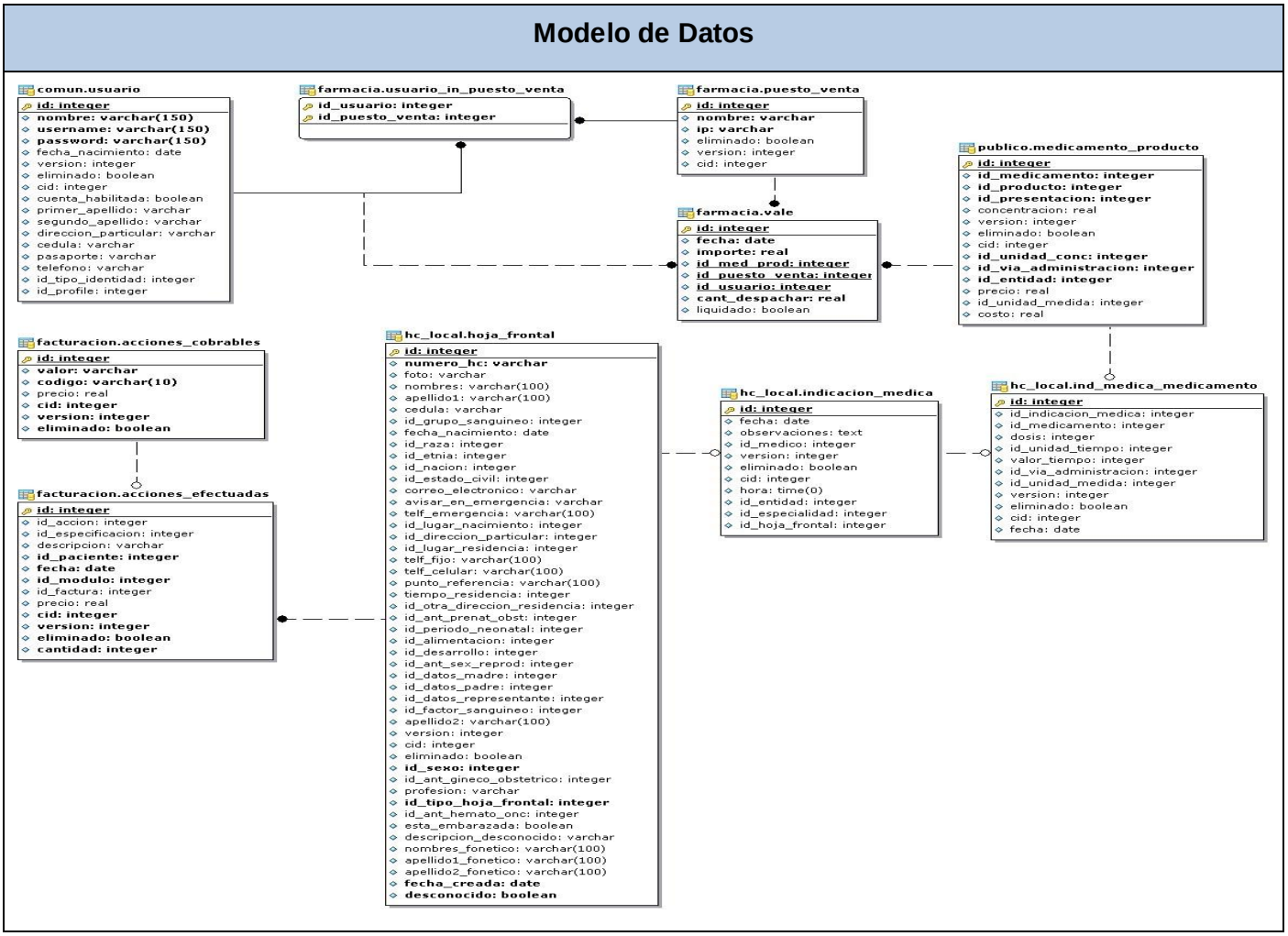


Tabla 3.16. Diagrama Entidad Relacional

3.4 Breve valoración de las Técnicas de validación

Durante el desarrollo de la aplicación el tratamiento de errores fue uno de los aspectos más importantes a medir. La validación de la información garantizó la corrección y precisión de todos los valores introducidos en la aplicación, además de lograr elevar la calidad de la misma. Mediante el uso de clases validadoras se

consolidó la integridad y la fidelidad de los datos. Estas clases se encargan a través de métodos específicos que los datos se correspondan a las adecuaciones del negocio. Otra vía de validación utilizada fue el uso del Framework JSF, el cual permite la manipulación y gestión de estas validaciones en las vistas, muestra así mensajes globales de errores generales como mensajes en errores particulares.

Integridad

La integridad en una base de datos se refiere a la corrección y exactitud de la información almacenada, además de permitir conservar la seguridad, la cual facilita el acceso a múltiples usuarios en tiempos paralelos. En los sistemas actuales se realiza la verificación de la integridad mediante procedimientos escritos por los usuarios. [40]

Una vez definida la estructura de datos del modelo relacional, es decir, el modelo conceptual, resulta necesario establecer reglas de integridad, donde los datos almacenados en dicha estructura deben cumplir para garantizar que son válidos. Al definir cada atributo sobre un dominio se impone una restricción sobre el conjunto de valores permitidos para cada atributo. Este tipo de restricciones recibe el nombre de restricciones de dominio.

Existen dos reglas de integridad generales asociadas con el modelo relacional que se deben cumplir en toda base de datos: *la Integridad de Entidad y la Integridad Referencial*. Las restricciones de entidades aseguran la integridad de las entidades que son modeladas por el sistema. Esta regla explica que la existencia de una clave principal es una restricción de calidad que impone la regla “cada entidad debe estar identificada de forma única”, además de no estar permitido que algún componente de la llave primaria acepte valores nulos. [40]

La regla de la Integridad Referencial se emplea en las llaves foráneas. Si en una relación existe una llave foránea, entonces sus valores deben coincidir con los valores de la llave primaria a la que hace referencia, o deben ser completamente nulos.

La Integridad Referencial verifica además que se cumplan las siguientes reglas:

- No se podrá introducir un valor en la tabla relacionada si antes no ha sido introducida en la tabla principal.
- No se pueden eliminar registros de la tabla principal si existen registros coincidentes en la tabla relacionada.
- No se puede cambiar un valor de la clave primaria en la tabla principal si el registro tiene registros relacionados.

3.4.1 Descripción de las tablas

Nombre: usuario			
Descripción: Contiene los datos de los usuarios creados en el sistema.			
Atributo	**Tipo**	**Nulo**	**Descripción**
id **(PK)**	Integer	No	Identificador
nombre	Varchar(25)	Sí	Nombre del usuario
username	Varchar(25)	Sí	Nombre usuario en el sistema
password	Varchar(25)	Sí	Contraseña del usuario
fecha_nacimiento	Date	Sí	Fecha de nacimiento (dd/mm/aaaa)
version	Integer	No	Versión
eliminado	Boolean	Sí	Si fue eliminado o no
cid	Integer	Sí	Identificador de la bitácora de sucesos
cuenta_limitada	Boolean	Sí	Si la cuenta es limitada o no
primer_apellido	Varchar(25)	Sí	Primer apellido
segundo_apellido	Varchar(25)	Sí	Segundo apellido
direccion_particular	Varchar(25)	Sí	Dirección particular
cedula	Varchar(25)	Sí	Carné de identidad
pasaporte	Varchar(25)	Sí	Número de pasaporte

telefono	Varchar(25)	Sí	Número de teléfono
id_tipo_identidad **(FK)**	Integer	No	Identificador que específica el tipo de identidad
id_profile **(FK)**	Integer	No	Identificador del perfil

Tabla 3.17. Descripción de la tabla usuario

Nombre: usuario_in_puesto_venta			
Descripción: Contiene los datos de la relación de usuario y puesto_venta.			
Atributo	**Tipo**	**Nulo**	**Descripción**
id_usuario **(PK/FK)**	Integer	No	Identificador del usuario
id_puesto_venta **(PK/FK)**	Integer	No	Identificador del puesto de venta

Tabla 3.18. Descripción de la tabla usuario_in_puesto_venta

Nombre: acciones_cobrables			
Descripción: Contiene los datos de las acciones cobrables efectuadas.			
Atributo	**Tipo**	**Nulo**	**Descripción**
id **(PK)**	Integer	No	Identificador
valor	Varchar(25)	Sí	Nombre de la acción cobrable
codigo	Varchar(25)	Sí	Código de la acción cobrable
precio	Real	Sí	Precio de la acción cobrable
cid	Integer	Sí	Identificador de la bitácora de sucesos
version	Integer	Sí	Versión
eliminado	Boolean	Sí	Si fue eliminado o no

Tabla 3.19. Descripción de la tabla acciones_cobrables

Nombre: acciones_efectuadas			
Descripción: Contiene los datos de las acciones efectuadas.			
Atributo	**Tipo**	**Nulo**	**Descripción**

id **(PK)**	Integer	No	Identificador
id_accion **(FK)**	Integer	No	Identificador de la acción
id_especificacion **(FK)**	Integer	No	Identificador de la especificación
descripcion	Varchar(25)	Sí	Descripción de las acciones efectuadas
id_paciente **(FK)**	Integer	No	Identificador del paciente
fecha	Date	Sí	Fecha de la creación de la acción (dd/mm/aaaa)
id_modulo **(FK)**	Integer	No	Identificador del módulo
id_factura **(FK)**	Integer	No	Identificador de la factura
precio	Real	Sí	Precio de la acción efectuada
cid	Integer	Sí	Identificador de la bitácora de sucesos
version	Integer	Sí	Versión
eliminado	Boolean	Sí	Si fue eliminado o no
cantidad	Integer	Sí	Cantidad de acciones efectuadas

Tabla 3.20. Descripción de la tabla acciones_efectuadas

Nombre: puesto_venta			
Descripción: Contiene los datos de la entidad puesto_venta.			
Atributo	**Tipo**	**Nulo**	**Descripción**
id **(PK)**	Integer	No	Identificador
nombre	Varchar(25)	Sí	Nombre del puesto de venta
ip	Varchar(25)	Sí	IP del puesto de venta
eliminado	Boolean	Sí	Si fue eliminado o no
version	Integer	Sí	Versión
cid	Integer	Sí	Identificador de la bitácora de sucesos

Tabla 3.21. Descripción de la tabla puesto_venta

Nombre: vale			
Descripción: Contiene los datos de la entidad vale.			
Atributo	**Tipo**	**Nulo**	**Descripción**
Id **(PK)**	Integer	No	Identificador
fecha	Date	Sí	Fecha de creación (dd/mm/aaaa)
importe	Real	Sí	Importe
id_med_prod **(FK)**	Integer	No	Identificador del MedicamentoProducto
id_puesto_venta **(FK)**	Integer	No	Identificador del puesto de venta
id_usuario **(FK)**	Integer	No	Identificador del usuario
cant_despachar	Real	Sí	Cantidad de medicamento en vale
liquidado	Boolean	Sí	Si fue liquidado o no

Tabla 3.22. Descripción de la tabla vale

Nombre: indicacion_medica			
Descripción: Contiene los datos de la entidad indicacion_medica.			
Atributo	**Tipo**	**Nulo**	**Descripción**
id **(PK)**	Integer	No	Identificador
fecha	Date	Sí	Fecha de la indicación (dd/mm/aaaa)
observaciones	Varchar(255)	Sí	Observaciones realizadas al paciente
id_medico **(FK)**	Integer	No	Identificador del médico
version	Integer	Sí	Versión
eliminado	Boolean	Sí	Si fue eliminado o no
cid	Integer	Sí	Identificador de la bitácora de sucesos
hora	Time	Sí	Hora de la indicación médica realizada (hh:mm:ss)

id_entidad **(FK)**	Integer	No	Identificador de la entidad
id_especialidad **(FK)**	Integer	No	Identificador de la especialidad
id_hoja_frontal **(FK)**	Integer	No	Identificador de la hoja frontal

Tabla 3.23. Descripción de la tabla indicacion_venta

3.5 Vista Implementación. (Diagrama de Componentes)

El Modelo de Implementación consiste en obtener una visión general de lo que tiene que ser implementado, y una estructura para cada iteración con los componentes y subsistemas a implementar durante esa iteración, así como el testeo que se ha de realizar sobre ellos. Esta vista presenta como objetivo determinar la organización del código, planificar las integraciones de sistemas necesarias en cada iteración e implementar las clases y subsistemas definidos durante el diseño. [37]

A continuación se presenta el Diagrama de Paquetes de Componentes implementado con tres componentes esenciales: Modelo, Vista y Controlador, donde se describe de forma detallada los componentes que usan, sean éstos de código fuente, librerías, binarios o ejecutables, realizado en el Lenguaje Unificado de Modelado UML.

Diagrama de Componentes

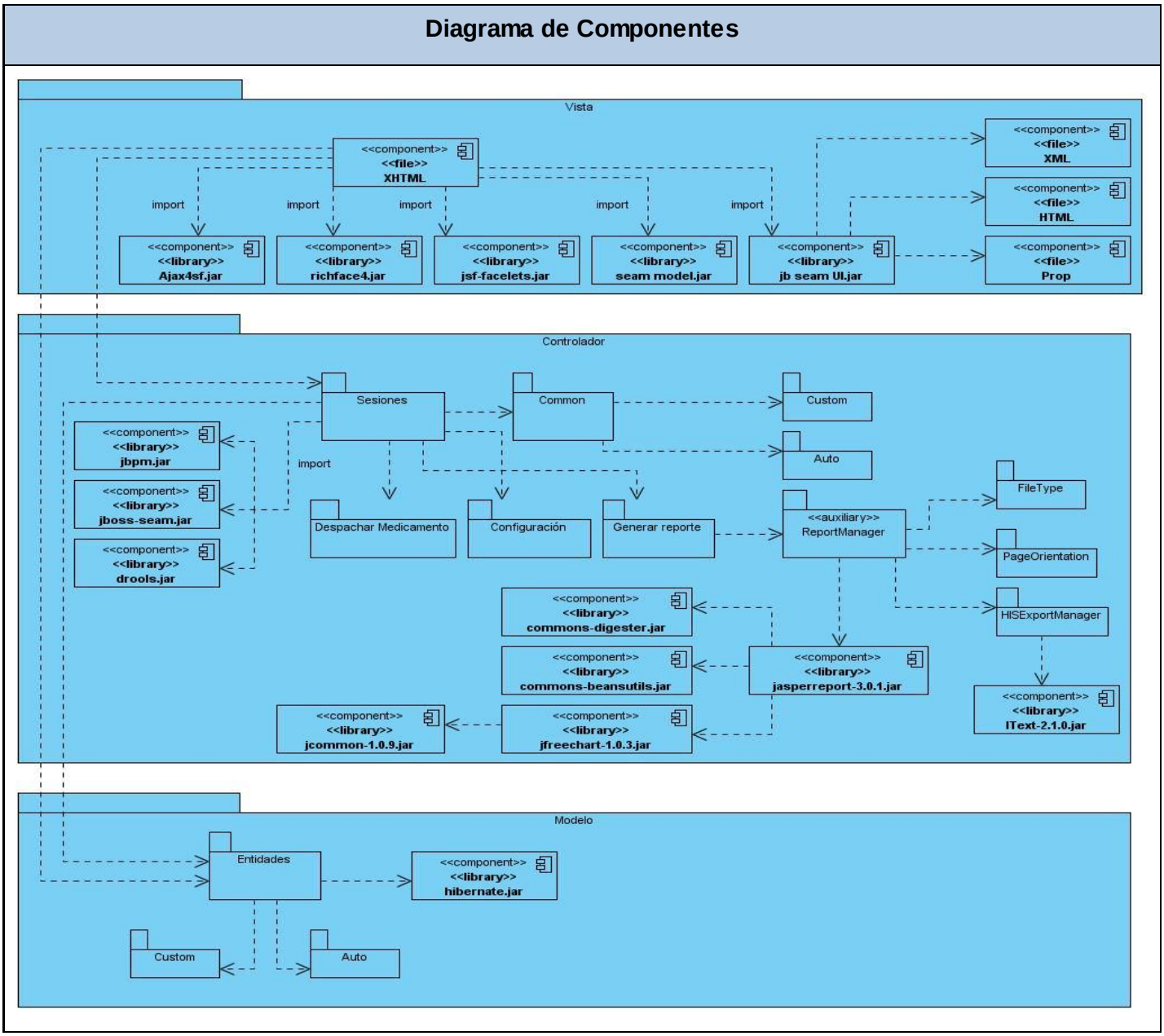


Tabla 3.24. Diagrama de Componentes

3.6 Descripción de la propuesta de solución

En este punto se hace mención a las funcionalidades implementadas y se describe uno de los procesos identificados previamente en el negocio. El sistema presenta funcionalidades como la generación de reportes, los cuales recogen información sobre los ingresos y las salidas de medicamentos del área de despacho de la Farmacia por concepto de venta. Cuenta además con un listar puestos de ventas, el cual brinda la posibilidad de realizar su modificación una vez creado este, eliminarlo y liquidar el mismo. El sistema permite registrar el costo de venta del medicamento con la opción de poder modificarlo.

En el proceso *Despachar medicamentos por paciente* se presenta una primera interfaz que lista los puestos de ventas creados (**Ver Anexo 4**). Una vez accedido a uno de ellos, se muestra el listado de los pacientes con indicaciones médicas, el sistema permite buscarlos dado criterios y seleccionarlos (**Ver Anexo 5**). A través de esta opción se procede a despachar los medicamentos por el servicio en que se atendió el paciente, donde se muestran por el código, la descripción, el precio y la cantidad indicada. El dependiente selecciona el o los medicamentos a despachar (**Ver Anexo 6**) y se realiza la facturación de dicha venta.

En esta interfaz (**Ver Anexo 7**) se presenta un listado de todos los medicamentos disponibles a despachar, se muestra además la cantidad real a ser despachada y se le facilita al dependiente la opción de ver el importe de los medicamentos. En la siguiente vista (**Ver Anexo 8**) se procede a crear la factura, el sistema permite hasta dos veces escoger la forma de pago, debido a que se puede realizar en efectivo o crédito. Se introduce el monto y el sistema muestra los detalles de la factura creada por la venta de medicamentos (**Ver Anexo 9**).

Como resultado del estudio realizado en este capítulo, correspondiente al flujo de diseño, se identificaron las clases fundamentales que deben ser definidas para que el sistema funcione satisfactoriamente. Asimismo fueron especificados los atributos y métodos que deben tener las clases para brindarle al desarrollador una idea clara de lo que se debe implementar. Se elaboraron los Diagramas de Clases y los Diagramas de Secuencia correspondientes a cada escenario. Además se obtuvo la realización de la vista de despliegue y se detalló uno de los procesos fundamentales para este sistema: *Despachar medicamentos por paciente.*

CAPITULO 4: MODELO DE PRUEBA

En este último capítulo del trabajo se realizará una breve descripción de los métodos de prueba que se investigaron y el escogido para modelar los casos de pruebas, como lo es para este caso el procedimiento de caja negra. Se expondrán los diferentes casos de pruebas realizados a los casos de uso descritos anteriormente en el flujo de Análisis y Diseño.

4.1 Métodos de prueba

Se realizó en este material la investigación de dos métodos existentes para comprobar la calidad del producto obtenido tras su etapa de implementación. Son utilizados con el objetivo de encontrar diferentes tipos de errores cada uno. Estos procedimientos corresponden a los nombres del método de caja blanca y el método de caja negra.

Resulta de carácter significativo valorar ciertos principios básicos que guían las pruebas del software antes de la aplicación de métodos para el diseño de casos de prueba efectivos: La prueba puede ser usada para mostrar la presencia de errores, pero nunca de su ausencia. Una parte necesaria de un caso de prueba es la definición del resultado esperado. Los casos de prueba tienen que ser escritos no solo para condiciones de entradas válidas y esperadas sino también para condiciones no válidas e inesperadas, son ejemplos de estos principios. [37]

El método de Caja Blanca se basa en la verificación de los detalles procedimentales. Se comprueban los caminos lógicos del software proponiéndose realizar casos de prueba que examinen que están correctas todas las condiciones para determinar si el estado real coincide con el esperado o afirmado. Esto genera gran cantidad de caminos posibles por lo que hay que dedicar esfuerzos a la determinación de las condiciones de prueba que se van a verificar. [37]

Las pruebas de Caja Negra se llevan sobre la interfaz de la aplicación, centrándose en los requisitos funcionales y ejercitándolos. Es decir, el propósito de los casos de prueba es demostrar que las funciones del sistema son operativas, que la entrada se acepta de forma adecuada, además de producirse un resultado correcto, y que la integridad de la información externa se mantiene. Este método es el más

utilizado ya que permite comenzar a realizar los casos de pruebas más temprano durante el proceso de desarrollo del software y en las clases de datos se pueden conocer los valores que deben ser ingresados con el objetivo de verificar los resultados.

El método de la Caja negra presenta tres técnicas clásicas para su realización:

Técnica de Partición de Equivalencia: Es una de las técnicas más efectivas y la escogida para realizar los casos de prueba en esta etapa, ya que permite dividir el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del sistema, posibilita examinar los valores válidos e inválidos de las entradas existentes en la aplicación y detecta de forma inmediata una clase de errores que requerirían la ejecución de muchos casos antes de descubrir el error genérico. [37]

Técnica del Análisis de Valores Límites: Esta técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables. [37]

Técnica de Grafos de Causa-Efecto: Esta técnica permite al encargado de la prueba validar complejos de acciones y condiciones. [37]

Un Caso de Prueba se define como un conjunto de condiciones que posibilitan la entrada de datos con el objetivo de obtener ciertos resultados. Estos resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente. Al realizar pruebas al producto obtenido garantizas la calidad del sistema y representa una revisión final de las especificaciones del diseño y de la codificación.

4.2 Descripción de los casos de prueba

Caso de uso: Seleccionar Usuario

Escenarios del Seleccionar Usuario	Descripción de la funcionalidad	Flujo Central
EC 1: Seleccionar Usuario	Seleccionar un usuario satisfactoriamente.	Muestra la interfaz para seleccionar el usuario. Se introducen los datos correspondientes a la búsqueda

		simple. Se selecciona la opción Buscar. Muestra un listado de usuarios que cumplen con los criterios de búsqueda. Se selecciona el usuario deseado.
EC 2: No se encuentra información	No se encuentra información que cumpla con los criterios de búsqueda.	Muestra la interfaz para seleccionar el usuario. Se introducen los datos correspondientes. Se selecciona la opción Buscar. Muestra el mensaje de información “No se encontró información que cumpla con los criterios de búsqueda.”
EC 3: Cancelar operación	Cancelar la opción de Seleccionar Usuario.	Muestra la interfaz para seleccionar el usuario. Se introducen los datos correspondientes. Se selecciona la opción Cancelar. Se regresa a la vista anterior.
EC 4: Ordenar	Permite ordenar el resultado ascendente o descendentemente por un atributo.	Muestra la interfaz para seleccionar el usuario. Se introducen los datos correspondientes. Se selecciona la opción Buscar. Muestra un listado de usuarios que cumplen con los criterios de búsqueda. Se ordenan los campos del listado por atributos.

SC 1: Seleccionar Usuario

Id del escenario	EC 1	EC 2	EC 3	EC 4
Escenario	Seleccionar Usuario	No se encuentra información	Cancelar operación	Ordenar
Variable 1 (Nombre)	V	I	V	NA
Variable 2 (Primer apellido)	V	V	V	NA
Variable 3 (Segundo apellido)	V	I	V	NA
Variable 4 (Desde)	V	I	I	
Variable 5 (Hasta)	V	V	I	
Botón 1 (Buscar)	NA	NA		
Botón 2 (Cancelar)			NA	
Botón 3 (Seleccionar)	NA			
Respuesta del Sistema	Muestra: Foto Nombre Primer apellido Segundo apellido	Muestra el mensaje de información: “No se encontró información que cumpla con los criterios de búsqueda.”	Regresa a la vista anterior.	Se muestran los atributos del listado ordenados.
Resultado de la Prueba				

Caso de uso: Listar pacientes con medicamentos indicados

Escenarios del Listar pacientes con medicamentos indicados	Descripción de la funcionalidad	Flujo Central
EC 1: Listar pacientes con medicamentos indicados	Mostrar el listado de los pacientes con medicamentos indicados satisfactoriamente.	Muestra un listado de los pacientes con indicaciones médicas y los criterios de búsqueda.
EC 2: Existen datos incorrectos	Existen datos incorrectos.	Muestra el mensaje de información “Existen datos incorrectos.”
EC 3: No se encuentra información	No se encuentra información que cumpla con los criterios de búsqueda.	Muestra el mensaje de información “No se encontró información que cumpla con los criterios de búsqueda.”
EC 4: Ordenar	Permite ordenar el resultado ascendente o descendentemente por un atributo.	Muestra un listado de los pacientes con indicaciones médicas. Se ordenan los campos del listado por atributos.
EC 5: Realizar búsqueda	Buscar un paciente con indicaciones médicas dados criterios.	Muestra la interfaz Listar pacientes con indicaciones médicas. Se introducen los datos correspondientes a la búsqueda simple. Se selecciona la opción Buscar. Muestra un listado de los pacientes con indicaciones médicas que cumplen con los criterios de búsqueda.
EC 6: Cancelar operación	Cancelar la opción de Buscar pacientes con indicaciones médicas.	Muestra la interfaz Listar pacientes con indicaciones médicas. Se introducen los datos correspondientes a la búsqueda simple. Se selecciona la opción Cancelar.

		Se regresa a la vista anterior.
EC 7: Seleccionar paciente	Seleccionar un paciente satisfactoriamente.	Muestra un listado de las indicaciones médicas y los criterios de búsqueda. Se selecciona la opción Seleccionar. Ver DCP **Listar medicamentos indicados para despachar por paciente.**

SC 1: Listar pacientes con medicamentos indicados

Id del escenario	EC 1	EC 2	EC 3	EC 4	EC 5	EC 6	EC 7
Escenario	Listar pacientes con medicamentos indicados	Existen datos incorrectos	No se encuentra información	Ordenar	Realizar búsqueda	Cancelar operación	Seleccionar paciente
Variable 1 (Nombre)	V	I	V	NA	V		
Variable 2 (Primer apellido)	V	V					
Variable 3 (Segundo apellido)	V	V	V				
Variable 4 (Servicios)	V	I	I				
Botón 1 (Buscar)					NA		
Botón 2 (Seleccionar)							NA
Botón 3 (Cancelar)						NA	
Respuesta del Sistema	Muestra: • Foto • Nombre	Muestra el mensaje de información: "Existen	Muestra el mensaje de información: "No se	Se muestran los atributos	Muestra: • Foto • Nombre	Regresa a la vista anterior.	Muestra la interfaz para

	• Primer apellido • Segundo apellido	datos incorrectos.”	encontró información que cumpla con los criterios de búsqueda.”	del listado ordenados por el nombre.	• Primer apellido • Segundo apellido		listar medicamentos indicados para despachar por paciente.
Resultado de la Prueba							

Caso de uso: Listar medicamentos indicados para despachar por paciente

Escenarios del Listar medicamentos indicados para despachar por paciente	**Descripción de la funcionalidad**	**Flujo Central**
EC 1: Listar medicamentos indicados para despachar por paciente	Mostrar el listado de las indicaciones médicas para despachar por paciente satisfactoriamente.	Muestra un listado de las indicaciones médicas para despachar por paciente y los criterios de búsqueda.
EC 2: Existen datos incorrectos	Existen datos incorrectos.	Muestra el mensaje de información “Existen datos incorrectos.”
EC 3: No se encuentra información	No se encuentra información que cumpla con los criterios de búsqueda.	Muestra el mensaje de información “No se encontró información que cumpla con los criterios de búsqueda.”
EC 4: Ordenar	Permite ordenar el resultado ascendente o descendentemente por un atributo.	Muestra un listado de las indicaciones médicas para despachar por paciente. Se ordenan los campos del listado por atributos.
EC 5: Realizar búsqueda	Buscar las indicaciones médicas para despachar por paciente dados criterios.	Muestra la interfaz Listado de medicamentos por servicio Se introducen los datos correspondientes a la búsqueda simple. Se selecciona la opción Buscar.

		Muestra un listado de las indicaciones médicas para despachar por paciente que cumplen con los criterios de búsqueda.
EC 6: Cancelar operación	Cancelar la opción de Buscar indicaciones médicas para despachar por paciente.	Muestra la interfaz Listado de medicamentos por servicio. Se introducen los datos correspondientes a la búsqueda simple. Se selecciona la opción Cancelar. Se regresa a la vista anterior.
EC 7: Seleccionar medicamentos	Seleccionar los medicamentos satisfactoriamente.	Muestra un listado de las indicaciones médicas y los criterios de búsqueda. Se selecciona la opción Seleccionar. Ver DCP **Despachar medicamentos por paciente.**

SC 1: Listar medicamentos indicados para despachar por paciente

Id del escenario	EC 1	EC 2	EC 3	EC 4	EC 5	EC 6	EC 7
Escenario	Listar medicamentos indicados para despachar por paciente	Existen datos incorrectos	No se encuentra información	Ordenar	Realizar búsqueda	Cancelar operación	Seleccionar medicamentos
Variable 1 (Descripción)	V	I	V	NA	V		
Variable 2 (Desde)	V	V					
Variable 3 (Hasta)	V	V	V				
Variable 4 (Servicios)	V	I	I				

(Combo)							
Variable 5 (Código)							
Variable 6 (Nombre)							
Variable 7 (Precio) **(Numérico)**							
Variable 8 (Cantidad) **(Numérico)**							
Botón 1 (Buscar)					NA		
Botón 2 (Cancelar)						NA	
Botón 3 (Seleccionar) **(Checkbox)**							NA
Respuesta del Sistema	Muestra: • Código • Nombre • Precio • Cantidad	Muestra el mensaje de información: "Existen datos incorrectos. "	Muestra el mensaje de información: "No se encontró información que cumpla con los criterios de búsqueda."	Se muestran los atributos del listado ordenados por el nombre.	Muestra: • Código • Nombre • Precio • Cantidad	Regresa a la vista anterior.	Muestra la interfaz para listar medicamentos indicados para despachar por paciente.
Resultado de la Prueba							

Caso de uso: Despachar medicamentos por paciente

Escenarios del Despachar medicamentos por paciente	Descripción de la funcionalidad	Flujo Central
EC 1: Despachar medicamentos	Despachar medicamentos por	Muestra el listado de los

por paciente.	paciente satisfactoriamente.	medicamentos seleccionados para despachar. Se introducen los datos correspondientes. Se selecciona la opción Aceptar. Se despachan los medicamentos.
EC 2: Ordenar	Permite ordenar el resultado ascendente o descendentemente por un atributo.	Muestra un listado de los medicamentos seleccionados para despachar. Se ordenan los campos del listado por atributos.
EC 3: Cancelar operación	Cancelar la opción Despachar medicamentos por paciente.	Muestra el listado de los medicamentos seleccionados para despachar. Se introducen los datos correspondientes. Se selecciona la opción Cancelar. Se regresa a la vista anterior.
EC 4: Eliminar	Permite eliminar los medicamentos seleccionados.	Muestra el listado de los medicamentos seleccionados para despachar. Se introducen los datos correspondientes. Se selecciona la opción Eliminar.

SC 1: Despachar medicamentos por paciente

Id del escenario	EC 1	EC 2	EC 3	EC 4
Escenario	Despachar medicamentos por paciente	Ordenar	Cancelar operación	Eliminar
Variable 1 (Código)	V	NA		
Variable 2 (Nombre)	V	NA		
Variable 3 (Precio) **(Numérico)**	V	NA		

Variable 4 (Cantidad real) **(Numérico)**	V			
Botón 1 (Aceptar)	NA			NA
Botón 2 (Cancelar)			NA	
Respuesta del Sistema	Se despachan los medicamentos seleccionados.	Muestra los atributos del listado ordenados.	Regresa a la vista anterior.	Es eliminado los medicamentos seleccionados.
Resultado de la Prueba				

Se emplearon 2 iteraciones para desarrollar el proceso de liberación del Módulo Farmacia Extrahospitalaria. Se obtuvo un total de 54 no conformidades, correspondientes al aspecto de funcionalidad se detectaron 8, las cuales de ellas todas se corrigieron; de interface de usuario se detectaron 13, todas al igual se corrigieron y en relación a las pautas de diseño, se detectaron 33 y 3 sin resolver, debido a que las mismas se definieron que no procedían.

A continuación se muestra en una tabla con el resumen las no conformidades detectadas durante la etapa de pruebas realizada al producto obtenido:

Artefacto	Estado final
Farmacia Extrahospitalaria	
Primera Iteración	43 NC detectadas, 2 NC sin resolver, representa el 0,8 % del total.
Segunda Iteración	11 NC detectadas, 1 NC sin resolver, representa el 1,2 % del total.

Tabla 4.1 Resumen de no conformidades por iteración.

Elemento	Cantidad
Funcionalidad	8 NC detectadas, 0 NC sin resolver, 0 % que representa del total.
Interface de usuario	13 NC detectadas, 0 NC sin resolver, 0 % que representa del total.
Pautas de diseño	33 NC detectadas, 3 NC sin resolver, 1 % que representa del total.

Tabla 4.2 Resumen de no conformidades por categoría.

En este último capítulo se plasmó con detalles el método escogido para realizar las pruebas al Módulo de Farmacia Extrahospitalaria, planteando el principal objetivo que presentan las mismas. Fue utilizada la Técnica de Partición de equivalencia para la realización de los casos de pruebas. Las no conformidades obtenidas por el revisor en los diferentes aspectos por las que son medidas en la universidad fueron respondidas correctamente y en tiempo, por lo que el proceso se realizó con buen desempeño.

CONCLUSIONES

Con el desarrollo del Módulo Farmacia Extrahospitalaria del Sistema de Información Hospitalaria alas HIS, se concluye lo siguiente:

1. La descripción de los procesos de negocio permitió el entendimiento e identificación de los requerimientos funcionales del sistema para su correcta implementación.
2. El análisis de los sistemas relacionados con el campo de acción, evidenció que los mismos no cumplen con todos los requerimientos funcionales deseados para el sistema alas HIS.
3. Las tecnologías, herramientas y patrones utilizados, permitieron la construcción de un sistema robusto y flexible.
4. La utilización de las pautas de diseño aplicadas permitieron lograr una uniformidad y homogeneidad en las interfaces visuales obtenidas.
5. Los procesos implementados permiten el fortalecimiento funcional del sistema alas HIS, que los gestionará mediante el Módulo de Farmacia.
6. Con la ejecución de las pruebas realizadas al Módulo Farmacia Extrahospitalaria se obtuvo una mejor calidad de las funcionalidades implementadas.

RECOMENDACIONES

Para una versión posterior se recomienda incluir funcionalidades para:

1. Dispensar medicamentos elaborados a través de fórmulas oficinales y magistrales.
2. Permitir la comunicación con sistemas externos, que implementen los mecanismos y especificaciones para la gestión de Recetas Médicas Electrónicas.
3. Leer indicaciones médicas desde una receta en formato de papel que contenga el código de barras de cada medicamento a despachar.
4. Gestionar el despacho de productos para los cuales no se requiera receta médica.
5. Despachar los medicamentos contenidos en las indicaciones médicas de los pacientes egresados desde el área de Hospitalización.

REFERENCIAS BIBLIOGRÁFICAS

[1] **Medina Salgado, César. Profesor e investigador del Departamento de Administración de la UAM-A.** La computadora como representante de la tecnología de la información. [Online] http://www.azc.uam.mx/publicaciones/enlinea2/num2/2-1.htm.

[2] **JOSEMI .** Tu web de Informática y Tecnología. [Online] http://www.configurarequipos.com/doc344.html.

[3[Online] http://www.csalud.junta-andalucia.es/principal/documentos.asp?pagina=Procesos_asistenciales.

[4] **Herskovic, Jorge. Documentos Gastr Latinoam 2005; Vol 16.** *Gastroenterología, Internet y algo más … ¿Qué es la Informática Médica?* Documentos Gastr Latinoam 2005; Vol 16.

[5] **García Nogueira, Keila.** Procesos Básicos Epidemiológicos para un Sistema de Gestión Hospitalario. Universidad de las Ciencias Informáticas. Ciudad de la Habana. 2009. 10p.

[6] **2006.** Sistemas de Información Hospitalario. Su importancia para el desarrollo de los Servicios de Salud y el control de la gestión. [Online] diciembre 10, 2006. http://www.Intramed.net/contenidover.asp?contenidoID=44061.

[7] **Vera Martin, Harold. 2009.** Niveles de Atención en Salud. [Online] noviembre 29, 2009.

[8] **Saladrigas, María Verónica.** *El sistema de clasificación ATC de sustancias farmacéuticas para uso humano.* [Online] http://www.medtrad.org/pana.htm.

[9] **Farhos.** Gestión de farmacias hospitalarias. [Online]: 2005.http://www.visual-limes.com/es/index.html?sec=pro&subsec=farhos&ap=quees.

[10] **Telvent.** Farmacia Hospitalaria. [Online] Televent, 2008. http://www.telvent.com/sites/telvent/es/soluciones/administracionespublicas/salud/soluciones/sistemasinformaciondepartamental/farmacia.html.

[11] **CNT.** Hospital Information System Edition. [Online] CNT PACIENTES, 2009. http://www.cnt.com.co/pagina/index.asp?id=178.

[12] **Paúl Céspedes, Jesús Armando.** Manuales SIGHO. [Online] 2007. http://sigho.saludtlax.gob.mx/wp-content/uploads/2007/08/farmacia-210.pdf.

[13] **Acacio Zavala, Ricardo. 2010.** Decisión TIC. [Online] septiembre 5, 2010. http://dtic.com.mx/index.php/opinion/1371-salud-20-hacia-un-servicio-de-salud-agil-y-eficiente; http://www.consumer.es/web/es/tecnologia/Internet/2005/11/02/146606.php/.

[14] FARMATOOLS. Máxima innovación para la seguridad del paciente. España : s.n., 2010.

[15] **León, Jeimy.** *MIS PRIMEROS PASOS POR EL MUNDO DE LA ARQUITECTURA DEL SOFTWARE.* **Octubre, 2010.** 19, Octubre, 2010.

[16] **Pérez, J. E.** (s.f.). *Librosweb.es.Introducción a XHTML*. Recuperado el 24 de junio de 2011, de [Online] http://www.librosweb.es/xhtml/.

[17] **García, A. P.** (n.d.). *desarrolloweb.com*. Retrieved junio 24, 2011, from Introducción a JSF (Java Server Faces). Primer artículo de un pequeño manual sobre esta tecnología.: http://www.desarrolloweb.com/articulos/2380.php/.

[18] **Desarrollo en Web.** Blog sobre desarrollo de aplicaciones Web en Java, Python y JavaScript. (2008, diciembre 17). [Online] http://blogs.antartec.com/desarrolloweb/2008/12/facelets-y-jsf-uso-de-templates/

[19] **JBoss Community.** JBoss Ajax4jsf Introduction. [Online] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.

[20] **Suárez, J. M.** (2010, 02 01). *Adictos al trabajo.* Retrieved from Introducción a RichFaces.: [Online] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro.

[21] **Seam UI.** refcardz.dzone.com. [Online] http://refcardz.dzone.com/refcardz/seam-ui.

[22] [Online] http://seamframework.org/.

[23] [Online] http://www.jboss.org/drools.

[24] [Online] www.oracle.com/technetwork/articles/.../jpa.

[25] [Online] www.oracle.com/technetwork/java/javaee/ejb.

[26] **Hibernate.** JBoss Community. [Online] http://www.hibernate.org/.

[27] PostgreSQL Marzo 2008. [Online]:http://www.freedownloadscenter.com/es/Programacion/Base_de_Datos_y_Redes/PostgreSQL_Maestro.html.

[28] **ARNOLD, KEN; GOSLING, JAMES; HOLMES, DAVID.** *El lenguaje de Programación Java.* Addison Wesley. 2009. [Online]: http://www.lenguajes-de-programacion.com/programacion-java.shtml

[29] **Java Platform, Enterprise Edition (JEE) documentation.** Oracle. [Online] http://www.oracle.com/technetwork/java/javaee/documentation/index.html.

[30] ***OsmosisLatina*.** (n.d.). Retrieved junio 2, 2011. [Online]http://www.osmosislatina.com/java/basico.htm

[31] ***OsmosisLatina*.** (n.d.). Retrieved junio 2, 2011. [Online]: http://www.osmosislatina.com/jboss/basico.htm

[32] **GOMEZ, JUAN.** Fundamentos de la metodología RUP. Septiembre 2007. [Online]: http://www.scribd.com/doc/297224/RUP

[33] **Tutorial UML.** [Online]: http://www.dcc.uchile.cl/~psalinas/uml/introduccion.html

[34] **Eclipse.** wiki.eclipse.org. [Online] http://wiki.eclipse.org/FAQ_What_is_Eclipse%3F.

[35] **Visual Paradigm for UML.** visual-paradigm.com. [Online]: http://www.visual-paradigm.com/product/vpuml/.

[36] **JBoss tools.** Febrero 2009 [Online]: http://es.wikipedia.org/wiki/JBoss_Seam

[37] **Pressman, Roger. 2005.** *Un Enfoque Práctico de la Ingeniería de Software.* 2005.

[38] **SAAVEDRA, JORGE.** PATRONES GRASP (Patrones de Software para la asignación General de Responsabilidad). Parte 2. Mayo 2007. [Online]: http://jorgesaavedra.wordpress.com/category/patrones-grasp/

[39] **GONZÁLEZ, RAÚL.** Modelo de datos .Mayo 2009 [Online]: http://mundogeek.net/archivos/2004/08/26/modelo-de-datos/

[40] **Juan. 2009.** *Integridad Informática. Enfocada a las bases de datos.* España, 12 de Septiembre de 2009. http://juanin.bligoo.com/content/view/606209/Integridad-en-las-Bases-de-Datos.html

BIBLIOGRAFIA

[1] **Medina Salgado, César. Profesor e investigador del Departamento de Administración de la UAM-A.** La computadora como representante de la tecnología de la información. [Online] http://www.azc.uam.mx/publicaciones/enlinea2/num2/2-1.htm.

[2] **JOSEMI .** Tu web de Informática y Tecnología. [Online] http://www.configurarequipos.com/doc344.html.

[3[Online] http://www.csalud.junta-andalucia.es/principal/documentos.asp?pagina=Procesos_asistenciales.

[4] **Herskovic, Jorge. Documentos Gastr Latinoam 2005; Vol 16.** *Gastroenterología, Internet y algo más … ¿Qué es la Informática Médica?* Documentos Gastr Latinoam 2005; Vol 16.

[5] **García Nogueira, Keila.** Procesos Básicos Epidemiológicos para un Sistema de Gestión Hospitalario. Universidad de las Ciencias Informáticas. Ciudad de la Habana. 2009. 10p.

[6] **2006.** Sistemas de Información Hospitalario. Su importancia para el desarrollo de los Servicios de Salud y el control de la gestión. [Online] diciembre 10, 2006. http://www.Intramed.net/contenidover.asp?contenidoID=44061.

[7] **Vera Martin, Harold. 2009.** Niveles de Atención en Salud. [Online] noviembre 29, 2009.

[8] **Saladrigas, María Verónica.** *El sistema de clasificación ATC de sustancias farmacéuticas para uso humano.* [Online] http://www.medtrad.org/pana.htm.

[9] **Farhos.** Gestión de farmacias hospitalarias. [Online]: 2005.http://www.visual-limes.com/es/index.html?sec=pro&subsec=farhos&ap=quees.

[10] **Telvent.** Farmacia Hospitalaria. [Online] Televent, 2008. http://www.telvent.com/sites/telvent/es/soluciones/administracionespublicas/salud/soluciones/sistemasinformaciondepartamental/farmacia.html.

[11] **CNT.** Hospital Information System Edition. [Online] CNT PACIENTES, 2009. http://www.cnt.com.co/pagina/index.asp?id=178.

[12] **Paúl Céspedes, Jesús Armando.** Manuales SIGHO. [Online] 2007. http://sigho.saludtlax.gob.mx/wp-content/uploads/2007/08/farmacia-210.pdf.

[13] **Acacio Zavala, Ricardo. 2010.** Decisión TIC. [Online] septiembre 5, 2010. http://dtic.com.mx/index.php/opinion/1371-salud-20-hacia-un-servicio-de-salud-agil-y-eficiente; http://www.consumer.es/web/es/tecnologia/Internet/2005/11/02/146606.php/.

[14] FARMATOOLS. Máxima innovación para la seguridad del paciente. España : s.n., 2010.

[15] **León, Jeimy.** *MIS PRIMEROS PASOS POR EL MUNDO DE LA ARQUITECTURA DEL SOFTWARE.* **Octubre, 2010.** 19, Octubre, 2010.

[16] **Pérez, J. E.** (s.f.). *Librosweb.es.Introducción a XHTML*. Recuperado el 24 de junio de 2011, de [Online] http://www.librosweb.es/xhtml/.

[17] **García, A. P.** (n.d.). *desarrolloweb.com*. Retrieved junio 24, 2011, from Introducción a JSF (Java Server Faces). Primer artículo de un pequeño manual sobre esta tecnología.: http://www.desarrolloweb.com/articulos/2380.php/.

[18] **Desarrollo en Web.** Blog sobre desarrollo de aplicaciones Web en Java, Python y JavaScript. (2008, diciembre 17). [Online] http://blogs.antartec.com/desarrolloweb/2008/12/facelets-y-jsf-uso-de-templates/

[19] **JBoss Community.** JBoss Ajax4jsf Introduction. [Online] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.

[20] **Suárez, J. M.** (2010, 02 01). *Adictos al trabajo.* Retrieved from Introducción a RichFaces.: [Online] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro.

[21] **Seam UI.** refcardz.dzone.com. [Online] http://refcardz.dzone.com/refcardz/seam-ui.

[22] [Online] http://seamframework.org/.

[23] [Online] http://www.jboss.org/drools.

[24] [Online] www.oracle.com/technetwork/articles/.../jpa.

[25] [Online] www.oracle.com/technetwork/java/javaee/ejb.

[26] **Hibernate.** JBoss Community. [Online] http://www.hibernate.org/.

[27] PostgreSQL Marzo 2008. [Online]:http://www.freedownloadscenter.com/es/Programacion/Base_de_Datos_y_Redes/PostgreSQL_Maestro.html.

[28] **ARNOLD, KEN; GOSLING, JAMES; HOLMES, DAVID.** *El lenguaje de Programación Java.* Addison Wesley. 2009. [Online]: http://www.lenguajes-de-programacion.com/programacion-java.shtml

[29] **Java Platform, Enterprise Edition (JEE) documentation.** Oracle. [Online] http://www.oracle.com/technetwork/java/javaee/documentation/index.html.

[30] ***OsmosisLatina*.** (n.d.). Retrieved junio 2, 2011. [Online]http://www.osmosislatina.com/java/basico.htm

[31] ***OsmosisLatina*.** (n.d.). Retrieved junio 2, 2011. [Online]: http://www.osmosislatina.com/jboss/basico.htm

[32] **GOMEZ, JUAN.** Fundamentos de la metodología RUP. Septiembre 2007. [Online]: http://www.scribd.com/doc/297224/RUP

[33] **Tutorial UML.** [Online]: http://www.dcc.uchile.cl/~psalinas/uml/introduccion.html

[34] **Eclipse.** wiki.eclipse.org. [Online] http://wiki.eclipse.org/FAQ_What_is_Eclipse%3F.

[35] **Visual Paradigm for UML.** visual-paradigm.com. [Online]: http://www.visual-paradigm.com/product/vpuml/.

[36] **JBoss tools.** Febrero 2009 [Online]: http://es.wikipedia.org/wiki/JBoss_Seam

[37] **Pressman, Roger. 2005.** *Un Enfoque Práctico de la Ingeniería de Software.* 2005.

[38] **SAAVEDRA, JORGE.** PATRONES GRASP (Patrones de Software para la asignación General de Responsabilidad). Parte 2. Mayo 2007. [Online]: http://jorgesaavedra.wordpress.com/category/patrones-grasp/

[39] **GONZÁLEZ, RAÚL.** Modelo de datos .Mayo 2009 [Online]: http://mundogeek.net/archivos/2004/08/26/modelo-de-datos/

[40] **Juan. 2009.** *Integridad Informática. Enfocada a las bases de datos.* España, 12 de Septiembre de 2009. http://juanin.bligoo.com/content/view/606209/Integridad-en-las-Bases-de-Datos.html

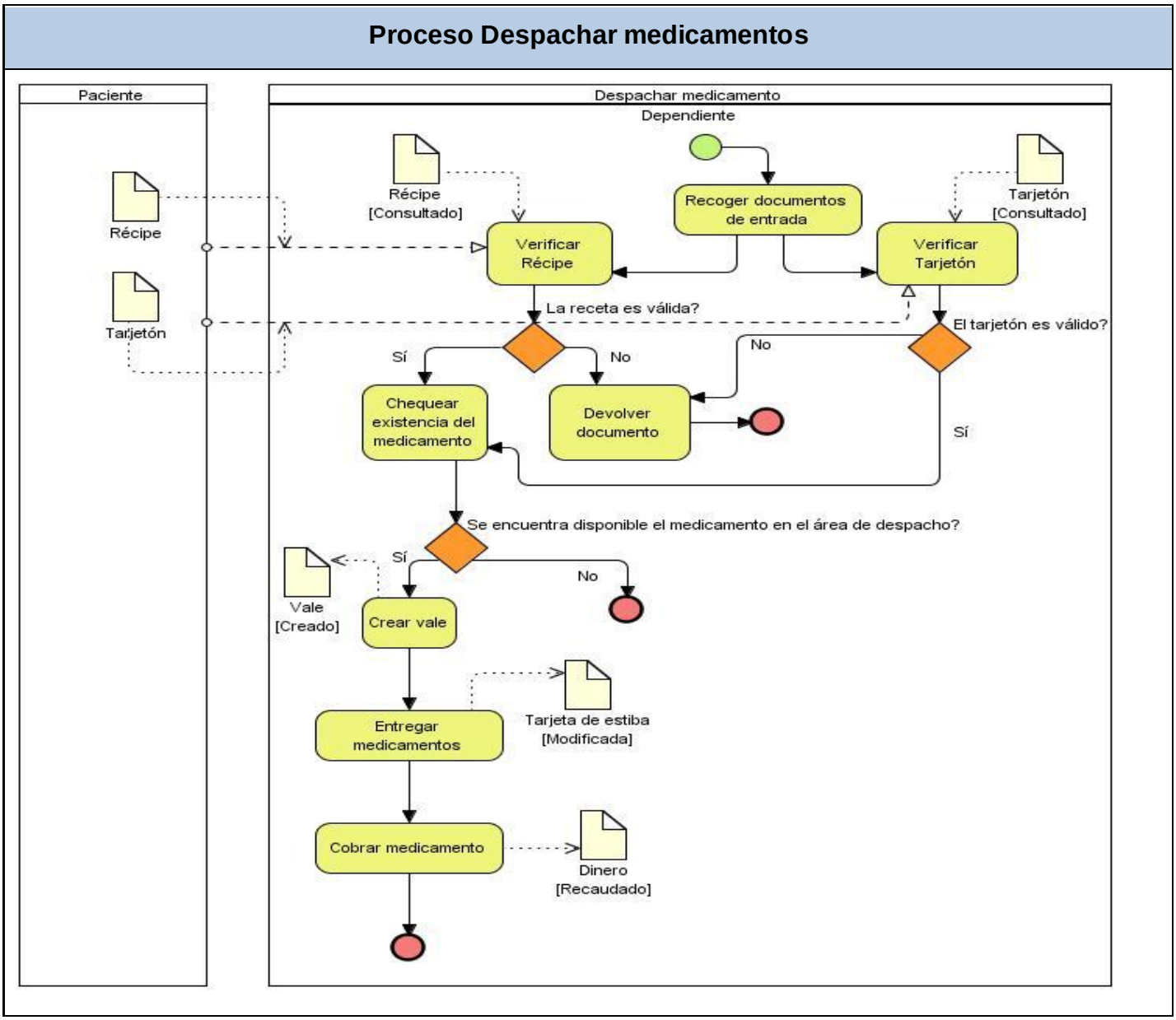
Torres, Jiménez. *ATENCIÓN FARMACEÚTICA EN UN HOSPITAL GENERAL. Capítulo 1.* V.

Ingeniería de Software. Material de apoyo. Conferencia Diseño. (EVA). http://eva.uci.cu/mod/resource/view.php?id=35381

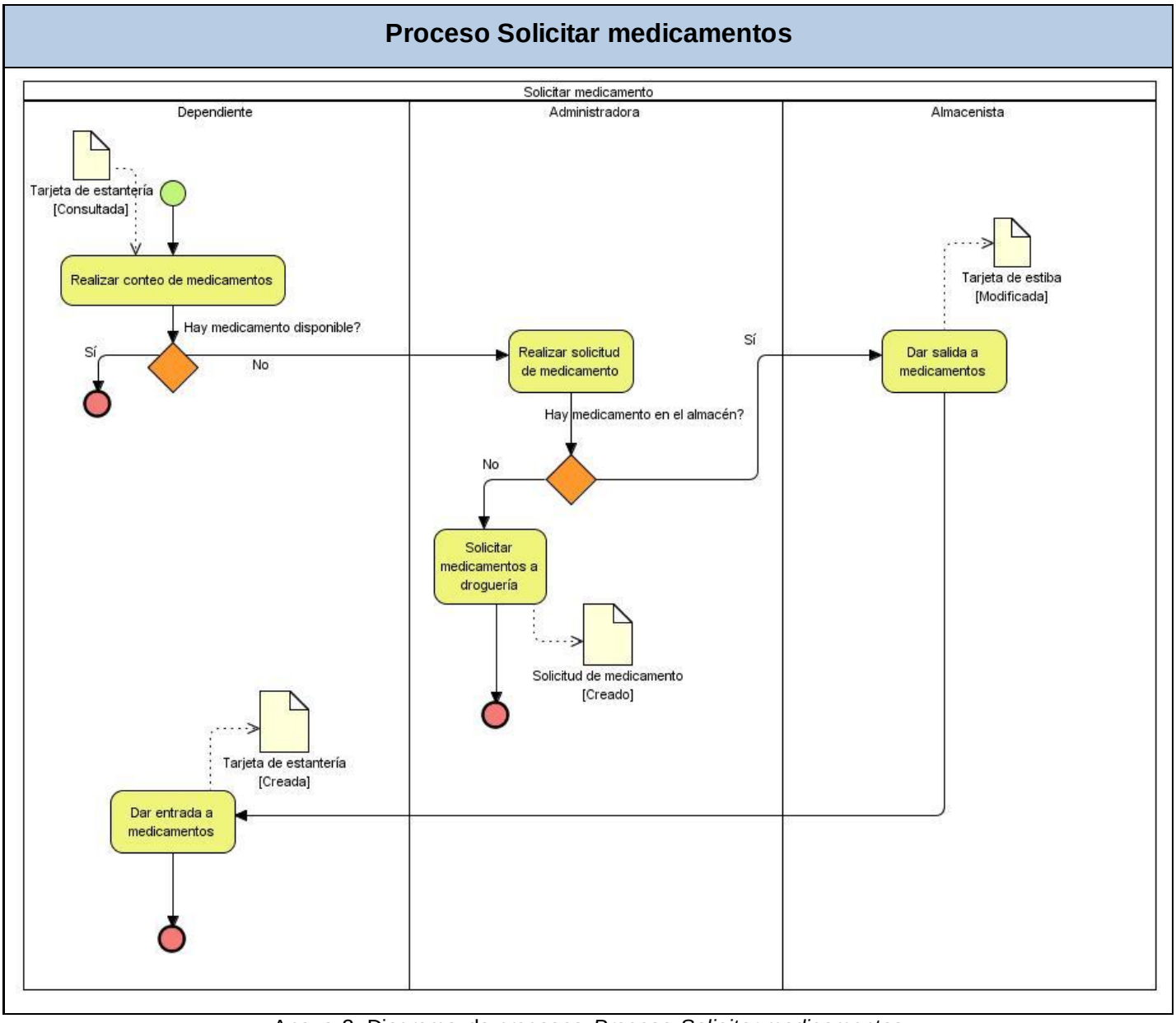
Sanchez Santos, Leonardo, et al. *Introducción a la Medicina General Integral.* La Habana, Cuba : Ciencias Médicas, 2001.

Velázquez Carralero, Alejandro Mario. *IH-SW-DR-091 ALAS-HIS_Documento de Arquitectura del Sistema.* 2008.

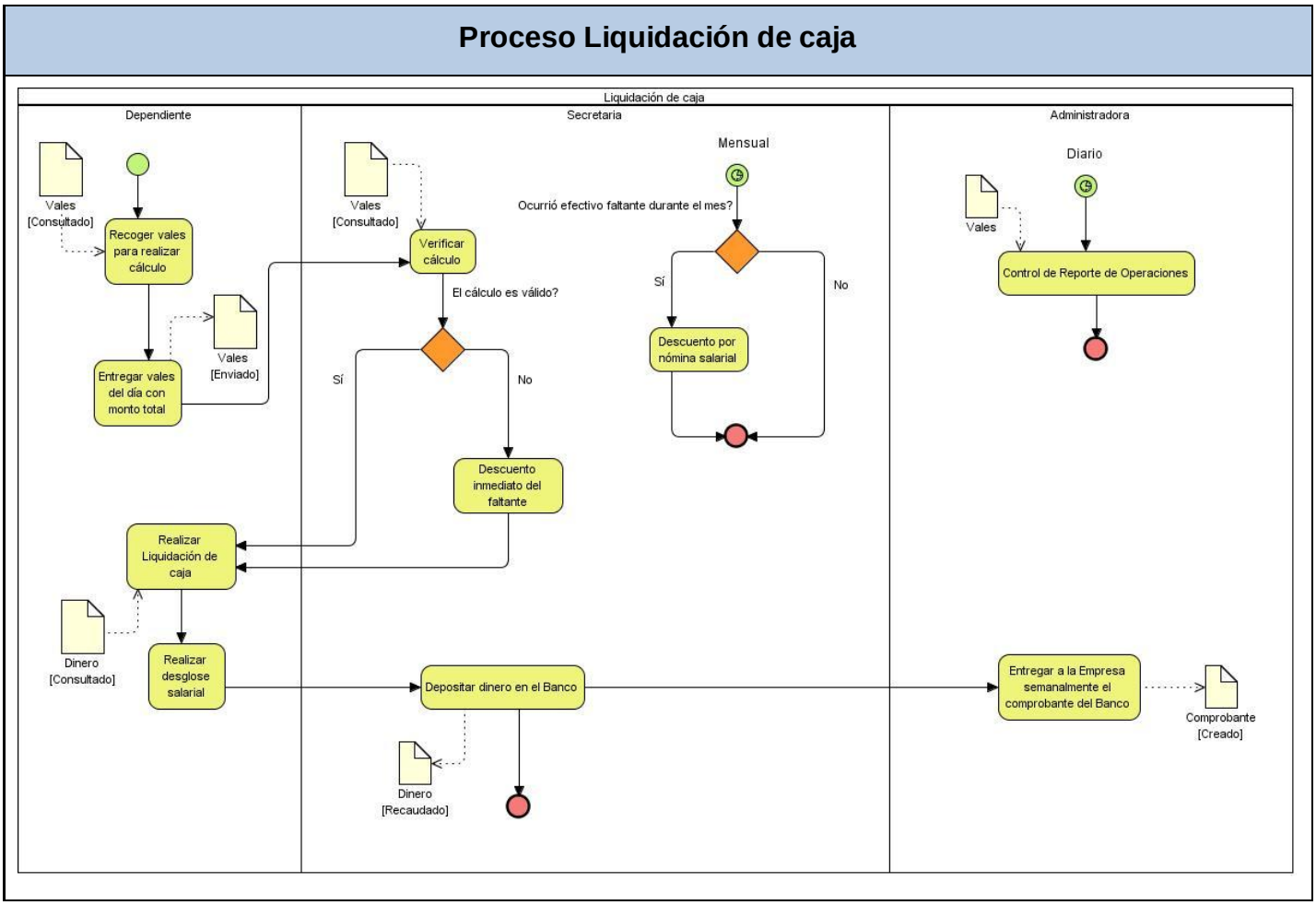
ANEXOS



Anexo 1: Diagrama de procesos *Proceso Despachar medicamentos*

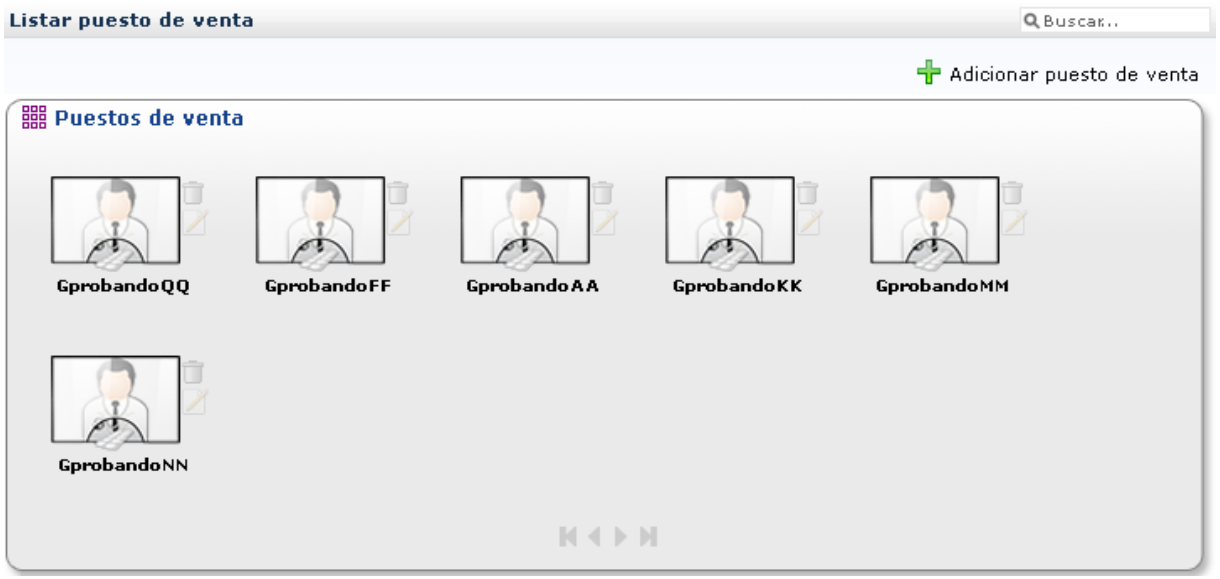


Anexo 2: Diagrama de procesos *Proceso Solicitar medicamentos*



Anexo 3: Diagrama de procesos *Proceso Liquidación de caja*

Interfaz Listar puestos de ventas



Anexo 4: Proceso Despachar medicamentos por paciente

Interfaz Listar pacientes con indicaciones médicas

Anexo 5: Proceso Despachar medicamentos a paciente

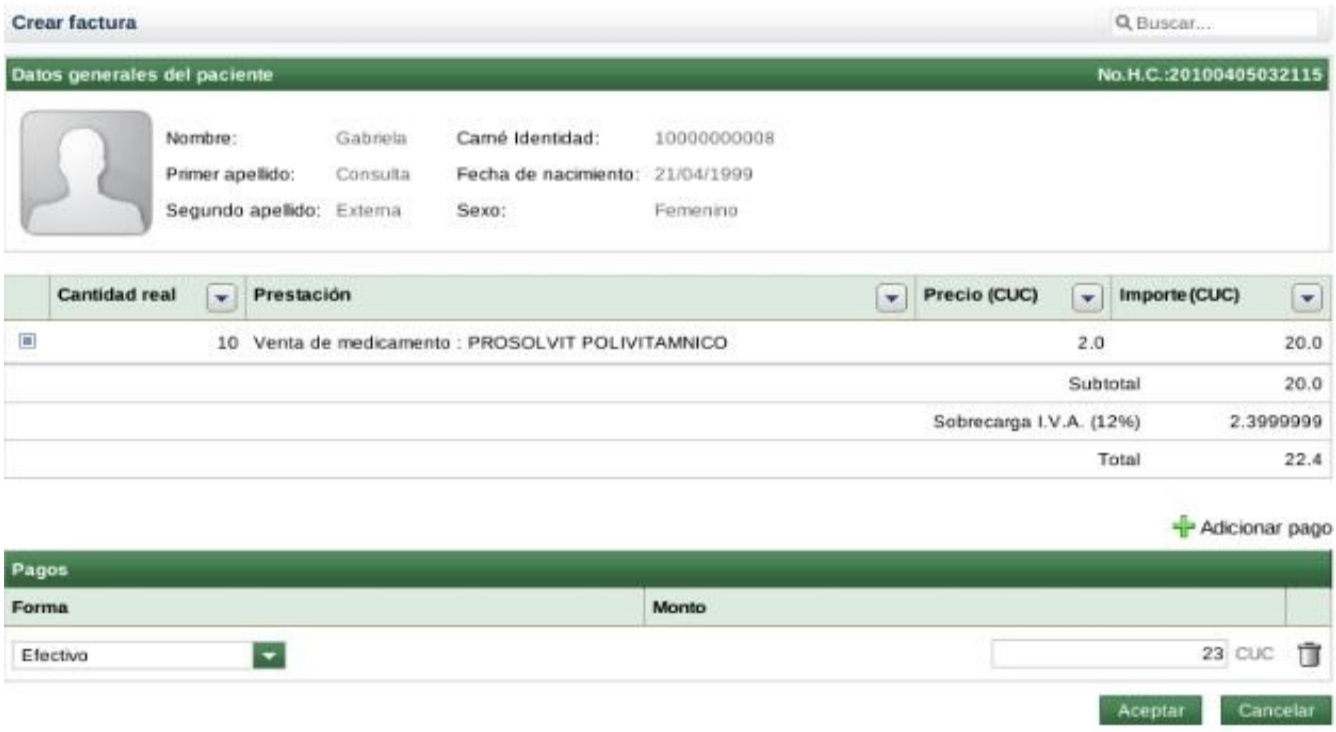
Interfaz Despachar medicamentos por servicio

Anexo 6: Proceso Despachar medicamentos por paciente

Interfaz Facturar ventas de medicamentos

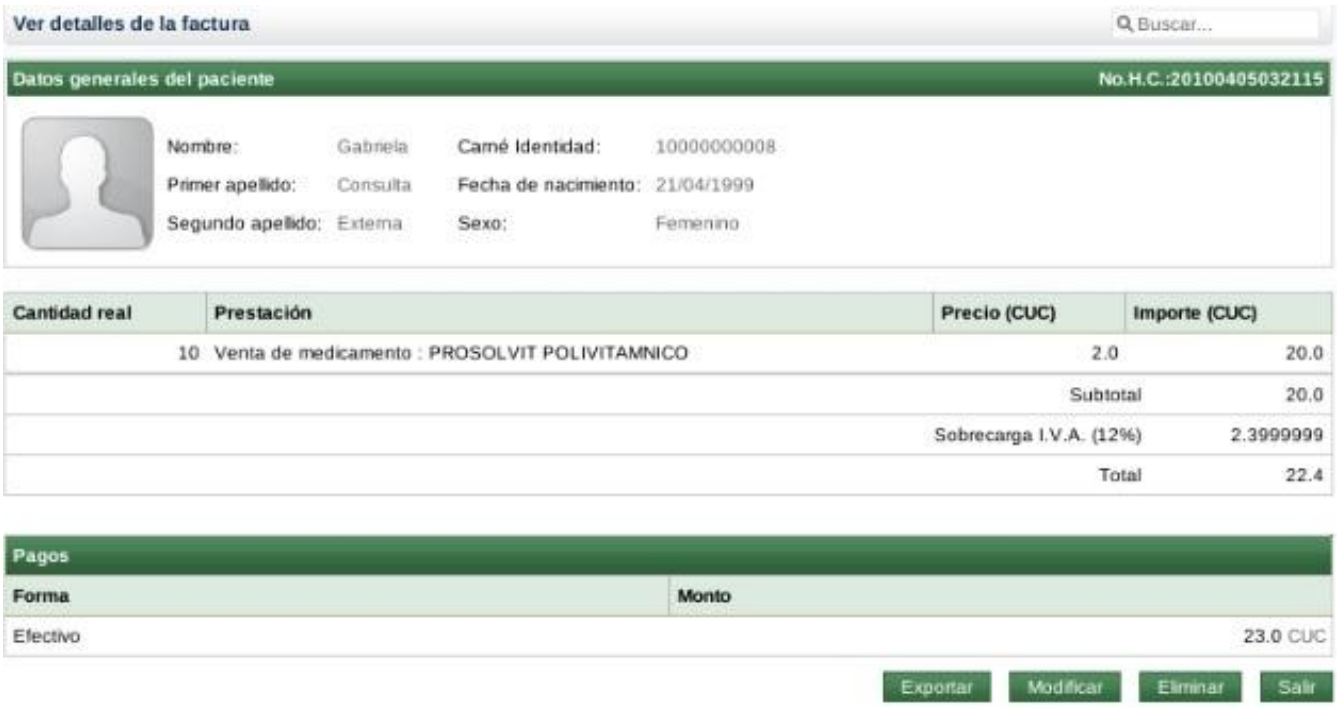
Anexo 7: Proceso Despachar medicamentos por paciente

Interfaz Crear factura



Anexo 8: Proceso Despachar medicamentos por paciente

Interfaz Ver detalles de la factura



Anexo 8: Proceso Despachar medicamentos por paciente