REPÚBLICA DE CUBA
MINISTERIO DE EDUCACIÓN SUPERIOR
UNIVERSIDAD DE GRANMA
DEPARTAMENTO DE INFORMÁTICA

UNIVERSIDAD TÉCNICA DE COTOPAXI
UNIDAD ACADÉMICA DE CIENCIAS DE LA INGENIERÍA Y APLICADAS

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN INFORMÁTICA

Sistema de Gestión de Historias Clínicas e Ingresos del Centro Hospitalario del Cantón La Maná, República del Ecuador.

AUTORES: Guillermo Ruiz Rendón
Danny Pincay Vinces

TUTORES: Ing. DanierMarante Jacas
Ing. DannerMarante Jacas

GRANMA, CUBA
ENERO, 2013

Declaración de auditoría

Declaramos que somos los únicos autores del trabajo de diploma titulado: “Sistema de Gestión de Historias Clínicas e Ingresos del Centro Hospitalario del Cantón La Maná, República del Ecuador”, y que el mismo pertenece a la Facultad de Ciencias Técnicas para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmamos la presente a los 16 días del mes deEnero del **2013.**

_____	_____
Firma del Autor	Firma del Autor
Guillermo Leonardo Ruiz Rendón	Danny Vicente Pincay Vinces

Opinión del usuario del trabajo de diploma.

El trabajo de diploma titulado: **"**Sistema de Gestión de Historias clínicas e ingresos del Centro Hospitalario del Cantón La Maná, República del Ecuador**",** fue realizado para la facultad de Ciencias Técnicas de la Universidad de Granma. Esta entidad considera que en correspondencia con los objetivos trazados el trabajo de diploma le satisface.

☒ **Totalmente**

☐ **Parcialmente en un ___ %**

Los resultados de este trabajo de diploma le reportan a la entidad los beneficios siguientes:

La aplicación Web presentada en esta investigación es de vital importancia pues

Y para que así conste se firma el presente a los **___** días del mes de ___del año **2013**.

_____ _____

Nombre del representante de la Entidad Cargo

_____

Firma y Cuño

Índice.

Índice de tablas.

Introducción.

Desde los inicios de la humanidad el hombre ha destinado gran parte de su tiempo a mejorar su salud e incrementar sus años de vida.

En el transcurso de los años, se determinó que era necesario almacenar el historial médico de los individuos, para mejorar la calidad de los servicios médicos y lograr una mayor efectividad en la lucha contra las enfermedades.

En Ecuador el gobierno financia una serie de centros hospitalarios en los cuales son atendidos los ciudadanos. En dichos centros se registra la historia clínica de los pacientes donde se almacena la información relacionada con las enfermedades que ha padecido, los antecedentes familiares con enfermedades congénitas, los padecimientos, los datos de los ingresos y las intervenciones quirúrgicas. Además de una serie de datos personales.

Esta información se maneja de forma manual, haciendo el proceso lento, en ocasiones generando errores humanos al momento del ingreso de las historias clínicas de los pacientes debido a la enorme cantidad de historias almacenadas y al volumen de información manejada para uso médico.

Por lo antes expuesto se plantea el siguiente **problema científico**:

¿Cómo mejorar el proceso de Historia Clínica e Ingreso de los pacientes en El Centro Hospitalario del Cantón La Maná?

Se define como **objeto** de la investigación:

El proceso de manejo de la información relacionada con la historia clínica e ingreso de los pacientes, en el Centro Hospitalario del Cantón La Maná.

Se puntualiza como **campo de acción:**

La gestión de información relacionada con el ingreso, historia clínica de los pacientes, en el Centro Hospitalario del Cantón La Maná.

Se estableció como **objetivo general:**

Desarrollar un sistema de gestión de historias clínicas e ingresos para mejorar el trabajo realizado en el Centro Hospitalario del Cantón La Maná.

Para lograr el cumplimiento de este objetivo general se han trazado como **objetivos específicos** los siguientes:

- Sintetizar los procesos llevados a cabo en el Centro Hospitalario del Cantón La Maná.
- Implementar una aplicación informática que mejore la gestión de los procesos de historia clínica e ingresos de los pacientes en el Centro Hospitalario.

Como **idea a defender** se considera que el desarrollo de un sistema informático para la gestión de las historias clínicas e ingresos de los pacientes en el Centro Hospitalario del Cantón La Maná, mejorará los servicios brindados por la institución.

Las **tareas** para dar cumplimiento al objetivo de la presente investigación se relacionan a continuación:

1. Analizar el estado del arte referente a la presente investigación.
2. Revisar detalladamente la documentación y procesos referentes a la gestión de historias clínicas e ingresos de los pacientes.
3. Determinar las herramientas para el desarrollo del sistema propuesto.
4. Desarrollar los artefactos de ingeniería de software para el sistema de gestión de historias clínicas e ingresos para el centro hospitalario del Cantón La Maná.
5. Implementar el sistema propuesto.
6. Diseño y aplicación de las pruebas propuestas por la metodología.

Para realizar las tareas se emplearon los siguientes **métodos**:

Método empírico:

- Revisión de documentos: para la recopilación de la información y la observación, lo que permitió conocer con claridad los datos que son de interés, para así poderlos procesar de una forma correcta.

Método teórico:

- Análisis y síntesis: para la recopilación y el procesamiento de la información y arribar a las conclusiones de la investigación, la obtención de conocimiento y resumir la información a procesar.

El presente documento está estructurado por dos capítulos, los cuales se describen a continuación:

Capítulo 1.

Este capítulo contiene la información del estudio del estado del arte de la presente investigación. Contiene además una investigación detallada sobre los lenguajes de programación, los sistemas de gestión de bases de datos, las metodologías de software y los framework utilizados en el mundo para el desarrollo de aplicaciones web, terminando con una propuesta para dar solución a la problemática expuesta.

Capítulo 2.

Este capítulo contiene la información referente a las etapas de desarrollo de la aplicación, contiene muestras de los principales artefactos de la metodología de software utilizada, además de la documentación referente al diseño de la base de datos, de lógica de negocio y presentación, terminando con los casos de pruebas utilizados para validar la calidad de la aplicación.

Capítulo 1: Fundamentación Teórica.

1.1 Introducción.

En el presente capítulo se exponen los fundamentos teóricos aplicados para el desarrollo del trabajo, abordando los Sistemas de Gestión, así como los conceptos principales relacionados con la temática. Se brinda la descripción de algunos sistemas en el ámbito internacional y nacional que se pueden utilizar como referencia a la hora de elaborar esta solución. Además de exponer las características de las herramientas, metodologías y tecnologías escogidas para la solución del problema, así como argumentar la elección de las mismas.

1.2 Conceptos.

Aplicaciones web.

Es un sistema informático que los usuarios utilizan accediendo a un servidor web a través de Internet o de una Intranet. Las aplicaciones web son populares debido a la practicidad del navegador web como cliente ligero. La facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software en miles de potenciales de clientes, es otra razón de popularidad (Reyes, 2010).

1.3 Sistemas similares.

En el mundo se han desarrollado una serie de aplicaciones destinadas para controlar los datos manejados en el control de los pacientes.

Sistema Hospitales Es un sistema desarrollado por la Universidad de Ciencias Informáticas para los C.D.I. y demás hospitales de Venezuela. Dicha aplicación se implementó utilizando P.H.P. como lenguaje de programación y MySQL como gestor de bases de datos. El sistema no es libre de pago, además no satisface las necesidades del cliente debido a que la información manejada en Ecuador dista de la manejada en los centros médicos de Venezuela.

1.4 Sistemas de gestión de bases de datos.

Se trata de un conjunto de programas no visibles al usuario final, que se encargan de la privacidad, la integridad, la seguridad de los datos y la interacción con el sistema operativo. Proporciona una interfaz entre los datos, los programas que los manejan y los usuarios finales. Cualquier operación (inserción, consulta o eliminación) que el usuario hace sobre la base de datos está controlada por el gestor. El gestor almacena una descripción de datos en lo que se llaman diccionario de datos, así como los usuarios permitidos y los permisos de cada uno de ellos. Tiene que haber un usuario administrador encargado de centralizar todas estas tareas (G. A. B. Díaz & Aguilar, 2011).

1.4.1 MySQL.

MySQL es un sistema de administración relacional de bases de datos. Una base de datos relacional archiva datos en tablas separadas en vez de colocar todos los datos en un gran archivo. Esto permite velocidad y flexibilidad. Las tablas están conectadas por relaciones definidas que hacen posible combinar datos de diferentes tablas sobre pedido. En aplicaciones Web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones (Zamora, 2009).

MySQL es el sistema gestor de bases de datos de código abierto más utilizado del mercado. Las empresas que lo utilizan consiguen recortar el gasto dedicado a este tipo de software en un 90%. Por regla general no cuesta dinero utilizarlo, siempre y cuando se sigan unas reglas muy sencillas. Este gestor se caracteriza por su facilidad de uso, velocidad y flexibilidad para funcionar en diferentes sistemas operativos.

1.4.2 PostgreSQL.

Es un sistema de gestión de bases de datos distribuido bajo la licencia GPL. Se encuentra en el segundo nivel del estándar del SQL99 revisión 3. Este sistema de gestión es considerado como un sistema objeto-relacional debido a que

implementa funciones objetuales como lista, herencia etc. El PortgreSQL es un gestor que funciona perfectamente con grandes volúmenes de datos. Cuenta con un lenguaje de programación pgplsql el cual tiene implementada una serie de funcionalidades que facilitan el manejo de los datos.

Las principales ventajas del PostgreSQL son.

- Distribuido bajo licencia GPL.
- Se encuentra en el segundo nivel del estándar SQL99.
- Buen funcionamiento ante grandes volúmenes de datos.
- Una cooperativa comunidad de desarrollo.
- Tiene implementado clúster y vistas materializadas.
- Multiplataforma

Desventajas.

- Comparado con gestores más pequeños consume muchos recursos.

1.5 LENGUAJES DE PROGRAMACIÓN.

1.5.1 Java.

Java se ha consolidado firmemente como el lenguaje de programación más utilizado en la actualidad por las aplicaciones *network-aware*de detención de redes y ha demostrado ser un lenguaje muy efectivo en programación general. Su gran popularidad se debe especialmente a que proporciona un ambiente de programación segura, transferible y de gran expresividad que además soporta la distribución de software de una manera invisible y sin interrupciones a través de la red (Zamora, 2009).

1.5.2JavaScript.

Es un lenguaje de programación interpretado, es decir, no requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos. Tiene la ventaja de ser incorporado en cualquier página web. Su principal característica es ser un lenguaje independiente de la plataforma (Día & Rodríguez, 2010).

JavaScript es un lenguaje de programación utilizado para crear pequeños programas encargados de realizar acciones dentro del ámbito de una página web. Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Su uso se basa, fundamentalmente, en la creación de efectos especiales en las páginas y la definición de interactividades con el usuario (G. A. B. Díaz & Aguilar, 2011).

1.5.3 Lenguaje de Marcado de Hipertexto Extensible (XHTML).

Es una versión más estricta y limpia de HTML, que nace con el objetivo de remplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML. XHTML extiende HTML 4.0, combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos(Día & Rodríguez, 2010).

1.5.4 Extensible MarkupLanguage (XML).

No es sólo un lenguaje, es una forma de especificar lenguajes, de ahí el término de extensible Es un lenguaje de etiquetas no predefinidas previamente, es decir, el programador es el que las crea en cada caso. El XML ahorra tiempos de desarrollo y proporciona ventajas, dotando a webs y a aplicaciones de una forma realmente potente de guardar la información. Se ha convertido en un formato universal para el intercambio de información estructurada entre diferentes

plataformas. En la actualidad permite la compatibilidad entre sistemas para compartir la información(Día & Rodríguez, 2010).

1.6 Entorno de Desarrollo Integrado.

1.6.1 Eclipse.

Para el desarrollo del *software* se utilizará el Eclipse, que es un entorno integrado (IDE) para desarrollo de aplicaciones con Java. Está soportado por IBM, es un proyecto *open source*, multiplataforma para desarrollar lo que el proyecto llama Aplicaciones de Cliente Enriquecido. Se está convirtiendo en el estándar de facto de los entornos de desarrollo para Java. Y es que Eclipse no es tan sólo un I.D.E. (Entorno de Desarrollo Integrado), se trata de un marco de trabajo modular ampliable mediante complementos (*plugins*). De hecho, existen complementos que permite usar Eclipse para programar en PHP, Perl, Python, C/C++, etc.

1.6.2 NetBeans

NetBeans fue desarrollado inicialmente por *SunMycrosystems*, y es un software libre y gratuito, de manera que puedes utilizarlo gratuitamente sin ningún costo. Permite programar aplicaciones principalmente en Java, posteriormente se complementó con paquetes adicionales del software que permiten programar en lenguajes como Ruby, C/C++ o bien PHP, pero su fuerte de programación se puede decir que es Java, y se puede programar en sus diferentes ediciones como la SE, ME o bien EE. Algo muy importante de NetBeans es que funciona en diversos sistemas operativos, tal como lo es *Windows*, Mac, Linux o Solaris, de manera que es muy compatible y el programador no tendrá problemas para instalarlo.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de Java escritas para interactuara con las APIs de NetBeans y un archivo especial (manifestfile) que lo identifica como módulo. Este IDE es muy flexible y basado en la calidad (Serrano & Minta, 2011).

1.7 Metodologías de desarrollo de software.

1.7.1 Metodologías ágiles.

El término ágil aplicado al desarrollo del software, surge en febrero del 2001, tras una reunión celebrada en Utah (EE.UU.), en la cual participan 17 expertos de la industria del software. Entre los que se encontraban algunos creadores o impulsores de metodologías de software. Su objetivo principal era permitir a los equipos desarrollar software rápidamente y respondiendo a los cambios de que puedan surgir a lo largo del proyecto. La reunión dio a luz la organización *The*AgileAlliance**(Agilealliance, 2011)**, sin ánimo de lucro, dedicada a promover los conceptos relacionados con el desarrollo ágil de software y cuya base y punto de partida fue el Manifiesto Ágil**(Agilemanifesto, 2011),** que resume la filosofía Ágil. Las metodologías ágiles han tomado tanta fuerza y es tal su impacto que actualmente existen 4 conferencias de alto nivel**(Universe, 2011)**.

Programación Extrema (Extreme Programing).

La metodología ágil XP está centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo (Extremeprogramming, 2011). XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios (XProgramming, 2011). XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (Canos & Letelier).

XP es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado. Surgió como respuesta y posible solución a los problemas derivados del cambio en los requerimientos, se plantea como una metodología a emplear en proyectos de riesgo y aumenta la productividad (Castro., Guerrero, & 2009).

1.8 FRAMEWORK.

En la actualidad, con la existencias de nuevas metodologías y herramientas en el ámbito de la Ingeniería de Software, el desarrollo de aplicaciones Web ha tomado un cauce distinto a lo que era años atrás, la demora de semanas con el desarrollo y las herramientas tradicionales para prototipos, ha sido reducido hoy en día con la aparición de los *FrameWork*. En los epígrafes siguientes se mencionaran algunos de ellos.

1.8.1 Java Server Faces.

La tecnología *Java Server Faces* es un marco de desarrollo de los componentes de la interfaz de usuario, válido para todas aquellas aplicaciones web basadas en la tecnología Java.

Java Server Faces (JSF) es un estándar de Java hacia la construcción de interfaces de usuario para aplicaciones web que simplifican el desarrollo de aplicaciones web del lado del cliente, JSF está basado en la tecnología Java EE. En el 2009 se dio a conocer la nueva versión JSF 2.0, que contiene algunas características y/o mejoras con respecto a las versiones anteriores (JSF 1.0, JSF 1.1 y JSF 1.2) como son: mejoras en la navegación, navegación condicional, inspección en tiempo de ejecución en las reglas de navegación; control de excepciones, permite fácilmente la creación de una página de error que utiliza componentes JSF; mejoras en la expresión del lenguaje, compatibilidad con métodos arbitrarios incluyendo el paso de parámetros; validación, es una nueva especificación Java desarrollada para la validación de beans(May, Gomez, & otros, 2011).

Por otra parte JSF es una tecnología para aplicaciones web que simplifica el desarrollo de interfaces de usuarios en aplicaciones Java EE. JSF usa JavaServer Pages (JSP) como la tecnología que permite hacer el despliegue de las páginas, pero también se puede acomodar a otras tecnologías (Serrano & Minta, 2011).

El objetivo de la tecnología JavaServer Faces es desarrollar aplicaciones web de forma parecida a como se construyen aplicaciones locales con Java Swing, AWT (AbstractWindowToolkit), SWT (Standard WidgetToolkit) o cualquier otra API similar. Esta pretende facilitar la construcción de estas aplicaciones proporcionando un entorno de trabajo (*FrameWork*) vía web que gestiona las acciones producidas por el usuario en su página HTML y las traduce a eventos que son enviados al servidor con el objetivo de regenerar la página original y reflejar los cambios pertinentes provocados por dichas acciones (Tordesillas, 2010).

1.8.2 PrimeFace

Es un marco de código abierto que añade a las aplicaciones capacidad de Ajax en JSF, sin recurrir a *JavaScript*. PrimeFaces aprovecha el F*rameWork Java Server Faces*, incluyendo su ciclo de vida, la validación, los medios de conversión y la gestión de los recursos estáticos y dinámicos. Los componentes de PrimeFace con soporte Ajax y aspecto altamente personalizable pueden ser fácilmente incorporados a aplicaciones JSF.

Ajax

Significa acrónimo para JavaScript asíncrono y XML, está basada en arquitectura web del lado del cliente, es multiplataforma, para el desarrollo de aplicaciones Web. Estas aplicaciones se ejecutan en el cliente, que sería, en el navegador de los usuarios, mientras se mantiene la comunicación asíncrona con el servidor en segundo plano es decir esta constantemente en modo *background,* intercambiando datos con el servidor sin que el usuario lo note, ya que la página no cambia su estado completo sino ciertas partes. Esto se debe a que el motor de Ajax está diseñado en JavaScript, pero está conformado por otras tecnologías como: html, xml, css, y XML HTTP. AJAX permite simular a la perfección el comportamiento de aplicaciones de escritorio en la web (Castro., et al., 2009).

1.8.3 Spring.

Spring es, como lo definen sus autores, un FrameWork ligero para construir aplicaciones empresariales. Aunque se encuentra dividido en distintos módulos, cada uno de los cuales se encarga de partes diferentes de la aplicación, no deja de ser un muy extenso, por lo que no se recomendarían usarlo en el desarrollo de pequeñas o medianas aplicaciones; pero en grandes o realmente grandes puede se ahorra mucho trabajo ya que puede coordinar todas las partes de la aplicación. Esta separación en módulos permite usar solo las partes que se necesitan, sin tener la carga de los que no se usará (Alex, 2010).

Está diseñado para no ser intrusivo, esto significa que no es necesario que la aplicación extienda o implemente alguna clase o interface del mismo (si no se quiere), por lo que el código de lógica quedará libre y completamente reutilizable para otro proyecto, o por si se debe quitar de una aplicación que ya lo esté usando.

Spring está dividido en alrededor de 20 módulos (ver **Fig1.1**) y colocados en los siguientes grupos:

- Contenedor Central (*CoreContainer*).
- Acceso a Datos / Integración.
- WEB.
- AOP (Programación Orientada a Aspectos).
- Instrumentación.
- Pruebas.

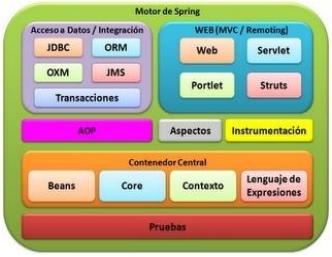

Fig1.1 Motor de Spring

1.8.4 Hibernate.

Trabajar con software orientado a objetos y bases de datos relacionales, puede ser embarazoso y demandar mucho tiempo, en los entornos corporativos actuales. Hibernate es una herramienta de mapeo objeto/relacional para ambientes Java. El término mapeo objeto/relacional (ORM por sus siglas en inglés) se refiere a esta técnica de mapear la representación de los datos desde un modelo de objetos hacia un modelo de datos relacional, con un esquema de base de datos basado en SQL.

Está no sólo se hace cargo del mapeo de clases Java a las tablas de una base de datos (y de los tipos Java a los tipos de la base de datos), sino que también provee utilidades para consulta y captura de datos, y puede reducir considerablemente el tiempo que, de otra manera, habría que invertir con el manejo manual de datos mediante SQL y JDBC.

La meta de Hibernate es aliviar al programador del 95% de las tareas más comunes relacionadas con persistencia. Probablemente, ella no sea la mejor solución para aplicaciones data-céntricas que tengan casi toda su lógica de negocios en procedimientos almacenados (*storedprocedures*) en la base de datos; es más útil con modelos orientados a objetos cuya lógica de negocio reside en la capa intermedia. Sin embargo, puede ayudarlo a encapsular o eliminar código SQL que sea específico de un proveedor de BD, y ayudará en la tarea usual de traducir desde una representación tabular a un gráfico de objetos (G. Díaz, 2009).

Hibernate es también definido como un entorno de trabajo que tiene como objetivo facilitar la persistencia de objetos Java en bases de datos relacionales y al mismo tiempo la consulta de estas bases de datos para obtener objetos.

1.8.5 Facelets.

Es un *FrameWork* simplificado de presentación, en donde es posible diseñar de forma libre una página web y luego asociarle los componentes JSF específicos. Aporta mayor libertad al diseñador y mejora los informes de errores que tiene JSF.

Permite que JSP (Java Server Pages) y JSF (Java Server Faces) puedan funcionar conjuntamente en una misma aplicación web. Estos no se complementan naturalmente. JSP procesa los elementos de la página de arriba a abajo, mientras que JSF dicta su propio *re-rendering* (ya que su ciclo de vida está dividido en fases marcadas). Facelets llena este vacío entre JSP y JSF, siendo una tecnología centrada en crear árboles de componentes y estar relacionado con el complejo ciclo de vida JSF (Día & Rodríguez, 2010).

Las principales ventajas de Facelets son:

- ✓ Construcción de interfaces basadas en plantillas.
- ✓ Rápida creación de componentes por composición.
- ✓ Fácil creación de funciones y librerías de componentes.
- ✓ Facelets provee un proceso de compilación más rápido que JSP.
- ✓ Provee *templating,* lo cual implica reutilización de código, simplificación de desarrollo y facilidad en el mantenimiento de grandes aplicaciones.

1.9 Servidor de Aplicaciones.

1.9.1 Apache Tomcat.

Es un servidor de aplicaciones para Java. Es muy reconocido por ser de los primeros servidores de aplicación empresarial gratuito y *open source*. Al estar basado en Java, puede ser utilizado en cualquier sistema operativo que lo soporte, ofreciendo una plataforma de alto rendimiento para aplicaciones java, aplicaciones Web y Portales.

El Servidor Apache HTTP es un servidor Web de tecnología Código Abierto (*Open Source*) sólido y para uso comercial desarrollado por la *Apache Software Foundation*(***Foundation*, 2011)**.Este servidor ofrece un grupo de ventajas, por ejemplo, es:

- ✓ Modular.
- ✓ Open source.
- ✓ Multi-plataforma.
- ✓ Extensible.
- ✓ Popular (fácil conseguir ayuda/soporte).
- ✓ Gratuito.
- ✓ Tiene capacidad para servir páginas tanto de contenido estático, como de contenido dinámico (Castro., et al., 2009).

1.9.2 GeoServer.

El proyecto GeoServer en una aplicación Java que integra la versión 1.0 de WFS 1.1.1 de WMS y 1.0 de WCS para poder publicar información, bien directamente, permitiendo su manipulación, bien en forma de imágenes o mapas. Geoserver tiene en cuenta cuatro metas principales en el ámbito de desarrollo, ordenadas por importancia:

- Cumplimiento de los estándares: El proyecto GeoServer intenta promover la estandarización, y soportar tantos estándares como sea posible, para permitir a todos compartir su información Geoespacial rápidamente y de una forma interoperable, disminuyendo así las barreras entre proveedores de información geográfica.
- Soporte para diferentes formatos información: Para hacer un producto útil, GeoServer intenta traducir todos los formatos de datos de información geográfica en uno solo. Sin embargo, el soporte para varios formatos de datos es una de las prioridades. Actualmente soporta eficientemente almacenamiento en los formatos Shapefile, PostGIS, DB2, Oracle, ArcSDE y, además ofrece servicio para soportes en prueba como MySQL, *Vector ProductFormat Library* (VPF), *Web Feature Server* (WFS) y MapInfo. Y en

cuanto a los formatos de salida que Geoserver puede generar como respuesta a peticiones WFS-T y WMS se encuentran, entre otros:

- JPEG: *JointPhotographicExpertsGroup*, algoritmo de compresión de imágenes con pérdida de información.
- PNG: *Portable Network Graphics*, algoritmo de compresión de imágenes sin pérdida de información.
- SVG: *Scaleable Vector Graphics*, lenguaje para describir gráficos vectoriales bidimensionales.
- GML: *GeographyMarkupLanguage*.
- PDF: *Portable DocumentFormat*, formato de almacenamiento de documentos, desarrollado por la empresa *Adobe Systems*.
- Shapefiles: formato propietario abierto de datos espaciales desarrollado por la compañía ESRI, originalmente creado para su producto ArcView GIS, pero actualmente se ha convertido en formato estándar de facto por la importancia que los productos ESRI tienen en el mercado SIG. Un Shapefile es un formato vectorial de almacenamiento digital donde se guarda la localización de los elementos geográficos y los atributos aso ciados a ellos. El formato carece de capacidad para almacenar información topológica.
- KML/KMZ: *KeyholeMarkupLanguage*, lenguaje de marcado basado en XML para representar datos geográficos en tres dimensiones, desarrollado para ser manejado con *Google Earth*.

- Fácil de usar: Fácil de instalar, configurar y ejecutar para organizaciones con pocos recursos técnicos. Orientado para organizaciones con experiencia técnica mínima.

- Eficiencia: El procesado de información geográfica normalmente requiere muchas cargas computacionales y de ancho de banda, GeoServer intenta minimizar ambas.

Desde el punto de vista del usuario, GeoServer es una herramienta necesaria para mostrar mapas en las páginas web, donde el usuario puede hacer zoom, cambiar la vista y hacer operaciones soportadas por las especificaciones WMS y WFS de OGC (Barbeito, 2007).

1.10 Propuesta de solución.

Para solucionar la problemática propuesta se plantea desarrollar una aplicación usando como metodología de desarrollo de software XP, como lenguaje de programación Java, con jsf como framework para la capa de presentación, primeface para agregar ajax, spring para capa media e hibernate para el acceso a datos. Para la capa de datos se usará PostgreSQL.

Conclusiones

Teniendo en cuenta la necesidad de desarrollar una aplicación se analizaron las soluciones similares tanto nacionales e internacionales. Además estudio del estado del arte referente a la problemática propuesta. Analizando las posibilidades de software, herramientas y tecnologías existentes en el mercado. Determinando una propuesta de solución para solucionar las necesidades del cliente.

Capítulo 2: Descripción y Construcción dela Solución Propuesta.

2.1 Introducción.

Después de haberse realizado un estado del arte sobre el objeto de la presente investigación y de haber seleccionado las herramientas y las metodologías a utilizar, se está en condiciones de realizar una propuesta de solución del sistema, basándose en la metodología ágil XP. Este capítulo está enmarcado en desarrollar las fases definidas en la metodología escogida, así como los artefactos que se generan en dichas fases.

2.2 Personas relacionadas con el sistema.

Se define como persona relacionada con el sistema toda aquella que de una manera u otra interactúa con este, y obtiene un resultado de uno o varios procesos que se ejecutan en el mismo. Además de aquellas que se encuentran involucradas en dichos procesos, que participan en ellos pero no obtienen ningún resultado de valor(Castro., et al., 2009). A continuación en la **Tabla 1** se muestran las personas relacionadas con el sistema de la investigación.

Tabla 1. Persona Relacionada con el Sistema

Persona Relacionada con el sistema	Justificación
Doctor	**Médico que trabaja en el hospital, tiene completo acceso a las historias de los pacientes, y realizarle modificaciones a estas.**
Enfermera	**Enfermera que trabaja en el hospital, puede acceder a las historias de los pacientes, y a las**

	salas a las que ella trabaja.
Administrador	**Tiene acceso a todas las funcionalidades del sistema. Con la posibilidad de modificar la estructura de los datos y del centro hospitalario.**
Recepcionista	**Es la persona que se encarga de ingresar a los pacientes en el sistema.**
Recursos Humanos	**Es la persona encargada de ingresar un doctor o enfermera en el sistema.**

2.3 FASE DE EXPLORACIÓN.

La exploración es la primera fase de la metodología de Programación Extrema. En esta los clientes plantean a grandes rasgos las historias de usuario necesarias para lograr el funcionamiento del producto. Además, se persigue en esta fase lograr la familiarización entre los miembros del equipo de trabajo y las herramientas, tecnologías y prácticas.

Historias de usuario.

Las Historias de Usuario (H.U.) son el medio mediante el cual se logra una especificación de los requisitos que conformarán el sistema. Estas son generadas por el cliente, contando con alguna ayuda del desarrollador en caso de ser necesario. El nivel de detalle de las H.U. debe de ser el mínimo posible que permita hacerse una idea de cuánto costará realizar la implementación del sistema. En los **Anexos 1, 2** se muestran H.U. pertenecientes al sistema.

2.4 Fase de Planificación.

La planificación es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario, y, asociadas a éstas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entregas, o Release Plan (Joskowicz, 2008).

Estimación de Esfuerzo por Historias de Usuario.

En esta fase el cliente establece la prioridad de cada historia de usuario, y los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. A continuación en la **Tabla 2** se observa la planificación del sistema.

Tabla 2. Resumen H.U.

No	Nombre H.U.	Prioridad	Riesgo	Esfuerzo	Iteración
1	Autenticar usuario.	Alta	Media	1	1
2	Administrar usuario.	Alta	Media	2	1
3	Gestionar doctor.	Baja	Media	2	1
4	Gestionar enfermera	Baja	Media	2	1
5	Gestionar sala	Baja	Baja	1	1
6	Gestionar cantón	Baja	Baja	1	1
7	Gestionar provincia	Baja	Baja	1	1
8	Gestionar parroquia	Baja	Baja	1	1
9	Gestionar paciente	Alta	Alta	3	1
10	Gestionar medicamento	Baja	Baja	1	1
11	Gestionar consulta	Media	Media	2	1

12	Gestionar ingreso	Media	Media	1	1
13	Gestionar alta medica	Media	Media	1	1
14	Gestionar historia clínica	Alta	Alta	2	2
15	Gestionar especialidad médica	Baja	Baja	2	2
16	Listar salas	Baja	Baja	1	2
17	Listar pacientes por sala	Media	Media	1	2
18	Listar pacientes por consulta	Baja	Baja	1	2
19	Reservar consulta	Alta	Media	2	2
20	Listar doctor	Baja	Baja	1	2
21	Listar enfermeras	Baja	Baja	1	2
22	Listar camas por sala	Baja	Baja	1	2
23	Buscar paciente	Alta	Alta	2	2
24	Listar parroquias	Baja	Baja	1	2
25	Listar cantón	Baja	Baja	1	2
26	Listar provincia	Baja	Baja	1	2
27	Gestionar Padecimiento	Media	Baja	1	2
28	Listar padecimientos-paciente	Media	Baja	1	2
29	Ingresar paciente	Alta	Media	2	2

2.5 Plan de Iteraciones.

Luego de que las historias de usuarios fueron descritas e identificadas, así como estimado el esfuerzo que cada una de ellas, se procede a implementar el sistema. Este plan muestra el orden en que serán implementadas las historias de usuarios y en qué iteración se desarrollará, así como el tiempo que demorará dicha implementación y la fecha para las liberaciones del producto. En la tabla 3 se muestran las iteraciones de las que se precisa para la elaboración del proyecto:

Tabla 3. Plan de Duración de Iteraciones.

Iteración	Descripción de la iteración	Orden de las H.U. a implementar	Duración Total
1	En esta iteración se implementarán las historias de usuarios de mayor prioridad. Al finalizar dicho proceso se contará con las funcionalidades descritas en las historias de usuarios de la 1 a la 15.	**1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15**	**4 semanas.**
2	En esta iteración se tiene como objetivo la implementación de las restantes historia de usuarios, con su finalización se lograra una versión final de la aplicación.	**16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30**	**4 semanas.**

2.6 Plan de entrega.

En este plan se realiza el cronograma de entregas que establece qué historias de usuario serán agrupadas para conformar una entrega, y el orden de las mismas. Además se acoplan las funcionalidades referentes a un mismo tema en módulos, esto permite un mayor entendimiento en la fase de implementación quedando de la siguiente manera:

Tabla 4. Plan de Duración de la Entrega.

Iteración	Iteración 1	Iteración 2
Entrega	Final 1ra Iteración 2da semana diciembre(2012)	Final 2da Iteración 1ra semana de enero(2013)

2.7 Primera iteración.

En esta iteración se espera lograr una versión funcional de la aplicación para mostrársela al cliente. Se ha planificado la programación de las historias de usuario básicas que en su ejecución plantean un flujo de trabajo, con el objetivo de lograr una aplicación con una lógica de negocio útil.

Tabla 5. Historia 1, Autenticar usuario.

Historia de Usuario	
Número: 1	**Usuario**: Doctor, Enfermera, Administrador, Recepcionista, Recursos Humanos.
Nombre de Historia: Autenticar usuario.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	**Riesgo en Desarrollo:** Media (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada:** 1
Programador Responsable: Guillermo Ruiz Rendón,Danny Pincay Vinces	
Descripción: Inicia cuando el usuario accede a la aplicación, se le brinda la posibilidad de que introduzca sus datos (usuario y contraseña) con el objetivo de verificar si el mismo está registrado y si él es quien dice ser, para poder asignarle los permisos según su rol.	
Observaciones:	

Tabla 6. Historia 2, Gestionar usuario.

Historia de Usuario	
Número: 2	**Usuario**: Administrador
Nombre de Historia: Administrar usuario**.**	
Prioridad en Negocio: Media (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 1
Programador Responsable: Guillermo Ruiz Rendón,Danny Pincay Vinces	
Descripción: Inicia cuando el jefe del departamento necesita crear o modificar un usuario existente en el sistema.	
Observaciones:	

Tabla 7. Historia 3,Gestionar doctor.

Historia de Usuario	
Número: 3	**Usuario**: Recursos Humanos.
Nombre de Historia: Gestionar doctor.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo:**Media (Alta, Media, Baja)
Puntos Estimados: 2	**IteraciónAsignada**: 1
Programador Responsable: Guillermo Ruiz Rendón,Danny Pincay Vinces	
Descripción: Inicia cuando el de Recursos Humanos introduce un nuevo doctor al sistema.	
Observaciones:	

Tabla 8. Historia 4,Gestionar enfermera.

Historia de Usuario	
Número: 4	**Usuario**: Recursos Humanos.
Nombre de Historia: Gestionar enfermera.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Media (Alta, Media, Baja)
Puntos Estimados: 2	**Iteración Asignada:**1
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando Recursos Humanos introduce una nueva enfermera al sistema.	
Observaciones:	

El resto de las historias correspondiente a esta iteración se encuentran en el Anexo 1.

A continuación se representan un conjunto de tareas correspondiente a las historias de usuario de la primera iteración. El resto de las tareas correspondientes a la primera iteración se encuentran en el Anexo 2.

Tabla 9. Tarea 1, Validar usuario y contraseña.

Tareas	
Número de Tarea : 1	**Numero de Historia:**1
Nombre de Tarea: Validar usuario y contraseña.	
Tipo de Tarea: Desarrollo	**Puntos Estimados**: 0.2
Fecha de Inicio: 24/9/12	**Fecha de Fin**: 24/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Cuando la persona entre su usuario y contraseña se verificará que los datos sean correctos, además se comprobará que no quede ningún campo por llenar en blanco.	

Tabla 10. Tarea 2, Insertar usuario.

Tareas	
Número de Tarea : 2	**Numero de Historia**: 2
Nombre de Tarea: Insertarusuario**.**	
Tipo de Tarea: Desarrollo	**Puntos Estimados:**0.2
Fecha de Inicio: 24/9/12	**Fecha de Fin**: 24/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Es necesario que se introduzcan los datos correspondientes al nombre, usuario y cargo.	

Tabla 11. Tarea 3, Eliminar usuario.

Tareas	
Número de Tarea : 3	**Numero de Historia:**2
Nombre de Tarea: Eliminarusuario**.**	
Tipo de Tarea: Desarrollo	**Puntos Estimados:**0.2
Fecha de Inicio: 26**/**9/12	**Fecha de Fin:**26/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona el usuario que se desea eliminar y se desactiva del sistema.	

Tabla 12. Tarea 4, Modificar usuario.

Tareas	
Número de Tarea : 4	**Numero de Historia:**2
Nombre de Tarea: Modificarusuario**.**	
Tipo de Tarea: Desarrollo	**Puntos Estimados:**0.2
Fecha de Inicio: 28/9/12	**Fecha de Fin**: 28/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona el usuario que se necesita modificar y se cargan todos los valores actuales en un formulario, permitiéndole al jefe de departamento modificar los valores.	

Tabla 13.Tarea 5, Insertar Doctor .

Tareas	
Número de Tarea : 5	**Numero de Historia:**3
Nombre de Tarea: InsertarDoctor	
Tipo de Tarea: Desarrollo	**Puntos Estimados:**0.2
Fecha de Inicio: 29/9/12	**Fecha de Fin:**29/9/12
Programador Responsable: Julio Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se introducen en un formulario la cedula, nombres, apellidos, sexo, calle, localidad, número, provincia, cantón, parroquia, referencia, especialidad, cargo, teléfono y se acciona el botón salvar.	

Tabla 14. Tarea 6, Eliminar Doctor.

Tareas	
Número de Tarea : 6	**Numero de Historia:**3
Nombre de Tarea: EliminarDoctor**.**	
Tipo de Tarea: Desarrollo	**Puntos Estimados:**0.2
Fecha de Inicio: 26/4/12	**Fecha de Fin:**26/4/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona el doctor que se desea eliminar y se elimina.	

2.8. Segunda Iteración.

En esta iteración se espera lograr un producto terminado con todas las funcionalidades que desea el cliente. Para esta parte se han dejado las historias de mayor complejidad y riesgo en el negocio.

A continuación se expondrá una selección de algunas de las historias de usuarios implementadas en esta fase, el resto de las historias se encuentran en el anexo 3.

Tabla 15. Historia 11, Visualizar solicitud.

Historia de Usuario	
Número: 11	**Usuario**: Administrador.
Nombre de Historia: Gestionar consulta.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 1
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando administrador introduce los datos de una nueva consulta en el sistema.	

Observaciones:

Tabla 16. Historia 12,Gestionar ingreso

Historia de Usuario	
Número: 12	**Usuario**: Doctor.
Nombre de Historia: Gestionar ingreso.	
Prioridad en Negocio: Media (Alta, Media, Baja)	**Riesgo en Desarrollo**: Media (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada:**1
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor determina la necesidad de ingresar a un paciente.	
Observaciones:	

Tabla 17. Historia 13, Gestionar alta médica.

Historia de Usuario	
Número: 13	**Usuario**: Doctor.
Nombre de Historia: Gestionar alta médica.	
Prioridad en Negocio: Media (Alta, Media, Baja)	**Riesgo en Desarrollo**: Media (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 1
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor determina que un paciente ingresado esta de alta.	
Observaciones:	

Tabla 18. Historia 14, Gestionar historia clínica.

Historia de Usuario	
Número: 14	**Usuario**: Doctor, Enfermera.
Nombre de Historia: Gestionar historia clínica.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	**Riesgo en Desarrollo:**Alta (Alta, Media, Baja)
Puntos Estimados: 2	**Iteración Asignada:** 1
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor o la enfermera necesita agregar contenido a la historia clínica de un paciente.	
Observaciones:	

Tabla 19. Historia 15, Gestionar especialidad médica.

Historia de Usuario	
Número: 15	**Usuario**: Administrador.
Nombre de Historia: Gestionar especialidad médica.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 1
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el administrador necesita gestionar las especialidades médicas.	
Observaciones:	

Tabla 20. Historia 16, Listar salas.

Historia de Usuario	
Número: 16	**Usuario**: Doctor, Enfermera, Administrador.
Nombre de Historia: Listar salas.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el administrador, doctor, enfermera, necesitan listar las salas del área de salud.	
Observaciones:	

Tabla 21. Historia 17, Listar pacientes por sala.

Historia de Usuario	
Número: 17	**Usuario**: Doctor, Enfermera.
Nombre de Historia: Listar pacientes por sala.	
Prioridad en Negocio: Media (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor o la enfermera necesita listar los pacientes de una sala.	
Observaciones:	

Tabla 22. Historia 18, Crear Historia.

Historia de Usuario	
Número: 18	**Usuario**: Doctor, Enfermera.
Nombre de Historia: Crear Historia	
Prioridad en Negocio: Alta (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor o la enfermera gestionan los datos de la historia clínica de un paciente determinado.	
Observaciones:	

Tabla 23. Historia 19, Ingresar paciente

Historia de Usuario	
Número: 19	**Usuario**: Doctor.
Nombre de Historia: Ingresar paciente	
Prioridad en Negocio: Alto (Alta, Media, Baja)	**Riesgo en Desarrollo**: Media (Alta, Media, Baja)
Puntos Estimados: 2	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor ingresa a un paciente.	
Observaciones:	

Tabla 24. Historia 20,Listar doctor.

Historia de Usuario	
Número: 20	**Usuario**: Recursos humanos
Nombre de Historia: Listar doctor.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
Programador Responsable:	
Descripción: Inicia cuando el de recursos humanos necesita visualizar los doctores.	
Observaciones:	

Tabla 25. Historia 21, Listar enfermeras.

Historia de Usuario	
Número: 21	**Usuario**: Recursos humanos
Nombre de Historia: Listar enfermeras.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
ProgramadorResponsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el de recursos humanos necesita visualizar las enfermeras del sistema.	
Observaciones:	

Tabla 26.Historia 22,Listar camas por sala

Historia de Usuario	
Número: 22	**Usuario**: Doctor, enfermera, recepcionista
Nombre de Historia: Listar camas por sala.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados:1	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor, enfermera, o recepcionista visualiza las camas por sala.	
Observaciones:	

Tabla 27. Historia 23, Buscar paciente.

Historia de Usuario	
Número: 23	**Usuario**: Doctor, enfermera, recepcionista
Nombre de Historia: Buscar paciente.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	**Riesgo en Desarrollo:**Alta (Alta, Media, Baja)
Puntos Estimados: 2	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor, enfermera, o recepcionista inicia una búsqueda del paciente en el sistema.	
Observaciones:	

Tabla 28. Historia 24, Listar parroquias.

Historia de Usuario	
Número: 24	**Usuario**: Doctor, enfermera, recepcionista, administrador
Nombre de Historia: Listar parroquias.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor, enfermera, recepcionista o administrador lista las parroquias del sistema.	
Observaciones:	

Tabla 29. Historia 25, Listar cantón.

Historia de Usuario	
Número: 25	**Usuario**: Doctor, enfermera, recepcionista, administrador
Nombre de Historia: Listar cantón.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor, enfermera, recepcionista o administrador lista los cantones del sistema.	
Observaciones:	

Tabla 30. Historia 26, Listar provincia.

Historia de Usuario	
Número: 26	**Usuario**: Doctor, enfermera, recepcionista, administrador
Nombre de Historia: Listar provincia.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor, enfermera, recepcionista o administrador lista las provincias del sistema.	
Observaciones:	

Tabla 31. Historia 28, Gestionar Padecimiento.

Historia de Usuario	
Número: 28	**Usuario**: Doctor, enfermera, administrador
Nombre de Historia: Gestionar Padecimiento.	
Prioridad en Negocio: Media (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 1	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor, enfermera, o administrador gestiona un padecimiento en el sistema.	
Observaciones:	

Tabla 32. Historia 29, Altas.

Historia de Usuario	
Número: 29	**Usuario**: Doctor.
Nombre de Historia: Altas.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	**Riesgo en Desarrollo**: Baja (Alta, Media, Baja)
Puntos Estimados: 2	**Iteración Asignada**: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor determina que un paciente determinado se encuentra de alta médica.	
Observaciones:	

2.9 PRODUCCIÓN.

Si bien al final de cada iteración se entregan módulos funcionales y sin errores, puede ser deseable por parte del cliente no poner el sistema en producción hasta tanto no se tenga la funcionalidad completa. En esta fase no se realizan más desarrollos funcionales, pero pueden ser necesarias tareas de ajuste (fine tuning)(Joskowicz, 2008).

Diseño.

La metodología XP hace especial énfasis en los diseños simples y claros. Los conceptos más importantes de diseño en esta metodología son simplicidad, soluciones, re codificación y metáforas(Joskowicz, 2008).

El sistema fue diseñado con el fin de facilitar el almacenamiento, procesamiento y representación de información referente a los recursos materiales de la Universidad de Granma. Se diseñó un sistema basado en la arquitectura n capas. Para el cual se utilizó java como lenguaje de programación, jsf como framework de

presentación, Spring como framework de capa media e Hibernate para acceso a datos.

El diseño se realizó de una forma dinámica logrando una apariencia agradable para el usuario. Se usaron los colores azul y blancos para personalizar las vistas, ya que son los colores que dan un ambiente profesional y refrescan la vista. Los menuces de usuarios se organizaron en grupos según sus prioridades logrando mejor accesibilidad y compenetración con el sistema. El formato de diseño de las páginas se realizó utilizando los estándares jsf, usando una página principal como escritorio de trabajo que cuentan con un header, menú izquierdo y un área central de despliegue de información que permiten gran velocidad en la carga y recarga de las páginas. El diseño de la base de datos se realizó utilizando Erwind, la misma cuenta con 23 tablas y está normalizada hasta la cuarta forma normal.

La programación se realizó usando la filosofía orientada a objeto, para el cual se determinó un dominio que contiene la lógica de negocio. Un paquete de daos para controlar el acceso a datos. Un paquete Facade para comunicación con las capas superiores, un paquete beans para controlar la lógica de presentación, un paquete reporte para las clases que exportan a .pdf y dao que controla la seguridad con Acegi.

Codificación.

El estándar de código utilizado es el estándar java.

2.10 Pruebas de Software.

Las pruebas en la metodología XP se consideran como parte indispensable de la misma, no debe existir ninguna característica en el sistema que no haya sido probado, los programadores realizan pruebas para chequear el correcto funcionamiento del sistema y los clientes pruebas funcionales. El resultado, un programa más seguro, que conforme pasa el tiempo es capaz de aceptar nuevos cambios(Solís., 2003).

Los programadores prueban constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones.

XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida y es diseñadas por el cliente final.

Pruebas de aceptación

Las pruebas de aceptación son creadas en base a las historias de usuarios, en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada. Las pruebas de aceptación son consideradas como pruebas de caja negra (*Black box systemtests*). Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. Asimismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es grupal, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todo el equipo esté al tanto de esta información(Joskowicz, 2008). A continuación se muestran algunas de las pruebas aplicadas al sistema.

Pruebas para la primera iteración.

Tabla 33. Prueba 1 HU_1.

Caso de prueba de aceptación	
Código: HU_1_P1	**Historia de usuario**: Autenticar usuario.
Nombre: Autenticación correcta de usuario**.**	
Descripción: Prueba para la funcionalidad de autenticación de usuario**.**	
Condiciones de ejecución: Los datos del usuario deben de ser válidos.	
Entrada\ Pasos de ejecución: Se introduce el usuario danier y la contraseña dmarantej.	
Resultado esperado: La autenticación correcta del usuario.	
Evaluación de prueba: Prueba satisfactoria.	

Tabla 34. Prueba 2 HU_2.

Caso de prueba de aceptación	
Código: HU_2_P1	**Historia de usuario**: Administrar usuario.
Nombre: Gestión correcta de usuario.	
Descripción: Prueba para la funcionalidad de insertar usuario**.**	
Condiciones de ejecución: Los datos del usuario deben de ser válidos.	
Entrada\ Pasos de ejecución: Se introduce el usuario danner y la contraseña dmarante y el rol administrador.	
Resultado esperado: La creación correcta de un usuario**.**	
Evaluación de prueba: Prueba satisfactoria.	

Tabla 35. Prueba 3 HU_3.

Caso de prueba de aceptación	
Código: HU_3_P1	**Historia de usuario**: Gestionar parroquia.
Nombre: Gestión correcta de una parroquia.	
Descripción: Prueba para la funcionalidad de insertar una parroquia.	
Condiciones de ejecución: Los datos de la parroquia deben de ser validos.	
Entrada\ Pasos de ejecución: Se introduce el nombre El Carmen, se selecciona la Provincia de Cotopaxi y el Cantón La Maná.	
Resultado esperado: La inserción correcta de una parroquia	
Evaluación de prueba: Prueba satisfactoria**.**	

Pruebas de la segunda iteración.

Tabla 36. Prueba 11 HU_11.

Caso de prueba de aceptación	
Código: HU_11_P1	**Historia de usuario**: Visualizar solicitud.
Nombre: Visualización de solicitudes.	
Descripción: Pruebapara la funcionalidad de filtrar solicitudes.	
Condiciones de ejecución: Los el número de inventario puede ser cualquier valor.	
Entrada\ Pasos de ejecución: Se muestran todas las solicitudes del sistema, se escribe el valor 001 en el número de inventario, se muestran solo las solicitudes hechas a los equipos que tienen en su número de inventario la cadena 001.	
Resultado esperado: Lista correcta de solicitudes.	
Evaluación de prueba: Prueba satisfactoria.	

Tabla 37. Prueba 12 HU_12.

Caso de prueba de aceptación	
Código: HU_12_P1	**Historia de usuario:**Asignar Solicitud.
Nombre: Asignación de Solicitud.	
Descripción: Prueba para la funcionalidad de asignar solicitudes.	
Condiciones de ejecución: Los datos deben de ser correctos.	
Entrada\ Pasos de ejecución: Se muestran todas las solicitudes que se encuentran en estado no asignadas, se selecciona la solicitud de tipo rotura al equipo 00089 asignándosela al técnico danier.	
Resultado esperado: Solicitud asignada.	
Evaluación de prueba: Prueba satisfactoria.	

2.11 MANTENIMIENTO.

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.

Luego de la finalización de cada iteración se realizaron mantenimientos correctivos y adaptivos según correspondía en cada caso, perfeccionando las funcionalidades del sistema y dando respuestas a inconformidades del cliente.

2.12 MUERTE DEL PROYECTO.

Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto

también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.

Se incluyeron todas las historias de usuarios que el cliente reporto, dándole respuestas y solución a sus requerimientos, por lo que se procedió a dar muerte al proyecto generando la documentación del sistema y no realizando cambios en su arquitectura.

Conclusiones.

- Se logró sintetizar los procesos manejados en el Centro Hospitalario del Cantón La Maná relacionados con el manejo de Historias clínicas e ingresos de los pacientes. Permitiendo el desarrollo de una aplicación para informatizar el trabajo realizado.
- Con la realización de los artefactos relacionados con las fases de exploración y planificación se permitió obtener una mejor comprensión del sistema. Tras la fase de implementación y prueba el sistema quedó terminado.
- Las pruebas realizadas arrojaron algunos errores que fueron corregidos durante la implementación, estos permitieron aumentar la calidad final del sistema. El sistema final cultivó las ventajas de Framework de Java y quedó listo para entrar en funcionamiento.

Recomendaciones.

Terminada la aplicación Man@Medic, desarrollada con el fin de controlar los ingresos e historias clínicas del Centro Hospitalario del Cantón La Maná de la República de Ecuador se recomienda:

- La puesta en práctica del mismo para facilitar el trabajo realizado por personal médico.
- Determinar posibles módulos para enriquecer el sistema.

REFERENCIAS BIBLIOGRÁFICAS:

Agilealliance (2011). Agilealliance.

Agilemanifesto (2011). Agilemanifesto.

Alex (2010). Spring3 - Parte1: Introducción from http: //www.javatutoriales.com/2010/09/spring-parte-1-introduccion.html

Barbeito, M. L. (2007). *Aplicación web para la gestión de contenidos geolocalizados.* Caruña.

Canos, J. H., & Letelier, P. *Métodologías Ágiles en el Desarrollo de Software*. Valencia.

Castro., B. P., Guerrero, A. A. Á., & (2009). *"Sistema para la gestión de imágenes libres para los proyectos productivos de la Universidad de las ciencias Informaticas"* Universidad de las ciencias Informáticas, Ciudad de La Habana.

Día, Y. M., & Rodríguez, D. M. (2010). *Módulo Medios de Diagnóstico del Subsistema Web del Sistema Integral para la Atención Primaria* Universidad de las Ciencias Informáticas Ciudad de la Habana.

Díaz, G. (Ed.). (2009). *HIBERNATE - Persistencia relacional para Java Idiomático*.

Earthwatch (2011). Earthwatch, from www.**earthwatch**.org

Extremeprogramming (2011). Extremeprogramming.

Foundation, A. S. (2011). Apache Tomcat

Gómez-Rubio, V., López-Quílez, A., & Verdejo, F. (2003). AEGIS: Herramienta de análisis epidemiológico en un sistema de información geográfica.

Irarragorri, I. J. C. C. (2001). **I**mplementación para el manejo integral del municipio del mariel.

.

Joskowicz, I. J. (2008). *Reglas y Prácticas en eXtreme Programming*

Loor, J. M. (2010). Primefaces.

May, F. P., Gomez, M. A., & otros (2011). Desarrollo de Aplicaciones Web con JPA,EJB,JSF y Primefaces.

Serrano, P. A. V., & Minta, M. A. G. (2011). *Análisis Comparativo de Tecnologías de Aplicaciones Web en el Entorno JSF Y ADF.* Escuela Superior Politécnica de Chimborazo, Ecuador.

Solís., M. C. (2003). apolosoftware, from http: //www.apolosoftware.com/

Spaceimaging (2011). Spaceimaging.

Tordesillas, R. T. (2010). Estudio de FrameWorks para la creación de interfaces gráficas. .

Ley_1224 (1969).

Úbeda, M. Á. (2009). *Herramienta Web genérica de publicación de cartografía municipal.* Unpublished Maestria, Universidad de Coruña, Coruña.

Unidas, N. (2000). *Manual de sistemas de información geográfica y cartografía digital.* Nueva York.

Universe, X. A. (2011). XP Agile Universe.

Resolución 21 (2010).

Vidal, X. H., Barbeito, G. N., & Pérez, M. I. S. (2008). *Epidat 4.0 Análisis Epidemiológico de datos.*

Vidal, X. H., Pérez, M. I. S., Fernández, E. V., &otros, y. (2007). Ayuda General Epidaf 3.1

Villacrés, C. A. B. (2008). *Servidores Webmapping para base de datos espaciales .Aplicativo: Sistema de Información Geográfica (Sig) prototipo del cuerpo de bombero Riobamba.* Escuela Superior Politécnica del Chinborazo, Riobamba Ecuador.

XProgramming (2011). XProgramming.

Zamora, D. N. V. (2009). *Sistema de planificación y consultas del Horario Docente de la Facultad de Informática de la Universidad de Granma. .* Universidad de Granma Bayamo.

Anexo 1. Historias de Usuario.

Historia 4 Autenticar usuario.

Historia de Usuario	
Número: 1	Usuario: Doctor, Enfermera, Administrador, Recepcionista, Recursos Humanos.
Nombre de Historia: Autenticar usuario.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	Riesgo en Desarrollo: Media (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable: Guille Danny	
Descripción: Inicia cuando el usuario accede a la aplicación, se le brinda la posibilidad de que introduzca sus datos (usuario y contraseña) con el objetivo de verificar si el mismo está registrado y si él es quien dice ser, para poder asignarle los permisos según su rol.	
Observaciones:	

Historia 5 Gestionar usuario.

Historia de Usuario	
Número: 2	Usuario: Administrador
Nombre de Historia: Administrar usuario.	
Prioridad en Negocio: Media (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el jefe del departamento necesita crear o modificar un	

usuario existente en el sistema.
Observaciones:

Historia 6 Gestionar doctor.

Historia de Usuario	
Número: 3	Usuario: Recursos Humanos.
Nombre de Historia: Gestionar doctor.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Media (Alta, Media, Baja)
Puntos Estimados: 2	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el de Recursos Humanos introduce un nuevo doctor al sistema.	
Observaciones:	

Historia 4 Gestionar enfermera.

Historia de Usuario	
Número: 4	Usuario: Recursos Humanos.
Nombre de Historia: Gestionar enfermera.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Media (Alta, Media, Baja)
Puntos Estimados: 2	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando Recursos Humanos introduce una nueva enfermera al sistema.	
Observaciones:	

Historia 5 Gestionar sala.

Historia de Usuario	
Número: 5	Usuario: Recursos Humanos.
Nombre de Historia: Gestionar sala.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando Recursos Humanos introduce una nueva enfermera al sistema.	
Observaciones:	

Historia 6 Gestionar cantón

Historia de Usuario	
Número: 6	Usuario: Administrador.
Nombre de Historia: Gestionar cantón.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el administrador introduce un nuevo cantón en el sistema.	
Observaciones:	

Historia 7 Gestionar provincia

Historia de Usuario	
Número: 7	Usuario: Administrador.
Nombre de Historia: Gestionar provincia.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el administrador introduce una nueva provincia en el sistema.	
Observaciones:	

Historia 8 Gestionar parroquia

Historia de Usuario	
Número: 8	Usuario: Administrador
Nombre de Historia: Gestionar parroquia.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el administrador introduce una nueva parroquia en el sistema	
Observaciones:	

Historia 9 Gestionar paciente

Historia de Usuario	
Número: 9	Usuario: Doctor, Recepcionista
Nombre de Historia: Gestionar paciente.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	Riesgo en Desarrollo: Alta (Alta, Media, Baja)
Puntos Estimados: 3	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el doctor o la recepcionista introduce los datos de un nuevo paciente en el sistema.	
Observaciones:	

Historia 10 Gestionar medicamento

Historia de Usuario	
Número: 10	Usuario: Doctor, Administrador
Nombre de Historia: Gestionar medicamento.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el doctor o el administrador introduce los datos de un nuevo medicamento en el sistema.	
Observaciones:	

Historia 11 Gestionar Consulta

Historia de Usuario	
Número: 11	Usuario: Administrador.
Nombre de Historia: Gestionar consulta.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando administrador introduce los datos de una nueva consulta en el sistema.	
Observaciones:	

Historia 12 Gestionar ingreso

Historia de Usuario	
Número: 12	Usuario: Doctor.
Nombre de Historia: Gestionar ingreso.	
Prioridad en Negocio: Media (Alta, Media, Baja)	Riesgo en Desarrollo: Media (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el doctordetermina la necesidad de ingresar a un paciente.	
Observaciones:	

Historia 13 Gestionar alta médica.

Historia de Usuario	
Número: 13	Usuario: Doctor.
Nombre de Historia: Gestionar alta médica.	
Prioridad en Negocio: Media (Alta, Media, Baja)	Riesgo en Desarrollo: Media (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 1
Programador Responsable:	
Descripción: Inicia cuando el doctor determina que un paciente ingresado esta de alta.	
Observaciones:	

Historia 14 Gestionar historia clínica.

Historia de Usuario	
Número: 14	Usuario: Doctor, Enfermera.
Nombre de Historia: Gestionar historia clínica.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	Riesgo en Desarrollo: Alta (Alta, Media, Baja)
Puntos Estimados: 2	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor o la enfermera necesita agregar contenido a la historia clínica de un paciente.	
Observaciones:	

Historia 15 Gestionar especialidad médica.

Historia de Usuario	
Número: 15	Usuario: Administrador.
Nombre de Historia: Gestionar especialidad médica.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el administrador necesita gestionar las especialidades medicas.	
Observaciones:	

Historia 16 Listar salas.

Historia de Usuario	
Número: 16	Usuario: Doctor, Enfermera, Administrador.
Nombre de Historia: Listar salas.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el administrador, doctor, enfermera, necesitan listar las salas del área de salud.	
Observaciones:	

Historia 17 Listar pacientes por sala.

Historia de Usuario	
Número: 17	Usuario: Doctor, Enfermera.
Nombre de Historia: Listar pacientes por sala.	
Prioridad en Negocio: Media (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor o la enfermera necesita listar los pacientes de una sala.	
Observaciones:	

Historia 18 Crear Historia.

Historia de Usuario	
Número: 18	Usuario: Doctor, Enfermera.
Nombre de Historia: Crear Historia	
Prioridad en Negocio: Alta (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor o la enfermera gestionan los datos de la historia clínica de un paciente determinado.	
Observaciones:	

Historia 19. Ingresar paciente

Historia de Usuario

Número: 19	Usuario: Doctor.
Nombre de Historia: Ingresar paciente	
Prioridad en Negocio: Alto (Alta, Media, Baja)	Riesgo en Desarrollo: Media (Alta, Media, Baja)
Puntos Estimados: 2	Iteración Asignada: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor ingresa a un paciente.	
Observaciones:	

Historia 20 Listar doctor.

Historia de Usuario	
Número: 20	Usuario: Recursos humanos
Nombre de Historia: Listar doctor.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el de recursos humanos necesita visualizar los doctores.	
Observaciones:	

Historia 21 Listar enfermeras.

Historia de Usuario	
Número: 21	Usuario: Recursos humanos
Nombre de Historia: Listar enfermeras.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el de recursos humanos necesita visualizar las enfermeras del sistema.	
Observaciones:	

Historia 22 Listar camas por sala

Historia de Usuario	
Número: 22	Usuario: Doctor, enfermera, recepcionista
Nombre de Historia: Listar camas por sala.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor, enfermera, o recepcionista visualiza las camas por sala.	
Observaciones:	

Historia 23 Buscar paciente.

Historia de Usuario	
Número: 23	Usuario: Doctor, enfermera, recepcionista
Nombre de Historia: Buscar paciente.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	Riesgo en Desarrollo: Alta (Alta, Media, Baja)
Puntos Estimados: 2	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor, enfermera, o recepcionista inicia una búsqueda del paciente en el sistema.	
Observaciones:	

Historia 24 Listar parroquias.

Historia de Usuario	
Número: 24	Usuario: Doctor, enfermera, recepcionista, administrador
Nombre de Historia: Listar parroquias.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor, enfermera, recepcionista o administrador lista las parroquias del sistema.	
Observaciones:	

Historia 25 Listar Cantón.

Historia de Usuario	
Número: 25	Usuario: Doctor, enfermera, recepcionista, administrador
Nombre de Historia: Listar cantón.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor, enfermera, recepcionista o administrador lista los cantones del sistema.	
Observaciones:	

Historia 26 Listar provincia.

Historia de Usuario	
Número: 26	Usuario: Doctor, enfermera, recepcionista, administrador
Nombre de Historia: Listar provincia.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor, enfermera, recepcionista o administrador lista las provincias del sistema.	
Observaciones:	

Historia 27 Listar provincia.

Historia de Usuario	
Número: 27	Usuario: Doctor, enfermera, recepcionista, administrador
Nombre de Historia: Listar provincia.	
Prioridad en Negocio: Baja (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor, enfermera, recepcionista o administrador lista las provincias del sistema.	
Observaciones:	

Historia 28 Gestionar Padecimiento.

Historia de Usuario	
Número: 28	Usuario: Doctor, enfermera, administrador
Nombre de Historia: Gestionar Padecimiento.	
Prioridad en Negocio: Media (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 1	Iteración Asignada: 2
Programador Responsable:	
Descripción: Inicia cuando el doctor, enfermera, o administrador gestiona un padecimiento en el sistema.	
Observaciones:	

Historia 29 Altas.

Historia de Usuario	
Número: 29	Usuario: Doctor.
Nombre de Historia: Altas.	
Prioridad en Negocio: Alta (Alta, Media, Baja)	Riesgo en Desarrollo: Baja (Alta, Media, Baja)
Puntos Estimados: 2	Iteración Asignada: 2
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Inicia cuando el doctor determina que un paciente determinado se encuentra de alta médica.	
Observaciones:	

Anexo 2. Tareas de las historias de usuario.

Tabla 2.1: Validar usuario y contraseña.

Tareas	
Número de Tarea : 1	Numero de Historia: 1
Nombre de Tarea: Validar usuario y contraseña.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 24/9/12	Fecha de Fin: 24/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Cuando la persona entre su usuario y contraseña se verificará que los datos sean correctos, además se comprobará que no quede ningún campo por llenar en blanco.	

Tabla 2.2: Insertar usuario.

Tareas	
Número de Tarea : 2	Numero de Historia: 2
Nombre de Tarea: Insertar usuario.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 24/9/12	Fecha de Fin: 24/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Es necesario que se introduzcan los datos correspondientes al nombre, usuario y cargo.	

Tabla 2.3 Eliminar usuario.

Tareas	
Número de Tarea : 3	Numero de Historia: 2
Nombre de Tarea: Eliminar usuario.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 26/9/12	Fecha de Fin: 26/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona el usuario que se desea eliminar y se desactiva del sistema.	

Tabla 2.4 Modificar usuario.

Tareas	
Número de Tarea : 4	Numero de Historia: 2
Nombre de Tarea: Modificar usuario.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 28/9/12	Fecha de Fin: 28/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona el usuario que se necesita modificar y se cargan todos los valores actuales en un formulario, permitiéndole al jefe de departamento modificar los valores.	

Tabla 2.5 Insertar área.

Tareas	
Número de Tarea : 5	Numero de Historia: 3
Nombre de Tarea: Insertar Doctor	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 29/9/12	Fecha de Fin: 29/9/12
Programador Responsable: Julio Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se introducen en un formulario la cedula, nombres, apellidos, sexo, calle, localidad, número, provincia, cantón, parroquia, referencia, especialidad, cargo, teléfono y se acciona el botón salvar.	

Tabla 2.6 Eliminar área

Tareas	
Número de Tarea : 6	Numero de Historia: 3
Nombre de Tarea: Eliminar Doctor.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 29/9/12	Fecha de Fin: 29/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona el doctor que se desea eliminar y se elimina.	

Tabla 2.7 Actualizar Doctor.

Tareas	
Número de Tarea : 7	Numero de Historia: 3
Nombre de Tarea: Actualizar Doctor.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 29/9/12	Fecha de Fin: 30/9/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selecciona el doctor que se desea modificar, se cargan los valores en un formulario permitiendo modificar los valores.	

Tabla 2.8 Insertar enfermera

Tareas	
Número de Tarea : 8	Numero de Historia: 4
Nombre de Tarea: Insertar enfermera.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 1/10/12	Fecha de Fin: 1/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se introducen en un formulario la cedula, nombres, apellidos, sexo, calle, localidad, número, provincia, cantón, parroquia, referencia, especialidad, cargo, teléfono y se acciona el botón salvar.	

Tabla 2.9 Eliminar enfermera.

Tareas	
Número de Tarea : 9	Numero de Historia: 4
Nombre de Tarea: Eliminar conservación.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 2/10/12	Fecha de Fin: 3/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selecciona la enfermera y se elimina.	

Tabla 2.10. Modificar enfermera.

Tareas	
Número de Tarea : 10	Numero de Historia: 4
Nombre de Tarea: Modificar conservación.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 3/10/12	Fecha de Fin: 3/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selecciona la enfermeray se cargan los valores en un formulario y se permiten modificar los valores.	

Tabla 2.11. Insertar sala.

Tareas	
Número de Tarea : 11	Numero de Historia: 5
Nombre de Tarea: Insertar sala.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 4/10/12	Fecha de Fin: 4/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se introducen en nombre, especialidad, jefe de sala y capacidad.	

Tabla 2.12. Modificar sala.

Tareas	
Número de Tarea : 12	Numero de Historia: 5
Nombre de Tarea: Modificar sala.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 4/10/12	Fecha de Fin: 5/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona una sala, se cargan todos los datos en un formulario permitiendo modificar los valores.	

Tabla 2.13. Eliminar sala.

Tareas	
Número de Tarea : 13	Numero de Historia: 5
Nombre de Tarea: Eliminar sala.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 5/10/12	Fecha de Fin: 5/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selecciona una sala y se elimina del sistema.	

Tabla 2.14. Adicionar Cantón.

Tareas	
Número de Tarea : 14	Numero de Historia: 6
Nombre de Tarea: Adicionar cantón.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 8/10/12	Fecha de Fin: 8/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se introduce el nombre, y la provincia.	

Tabla 2.15. Modificar Cantón.

Tareas	
Número de Tarea : 15	Numero de Historia: 6
Nombre de Tarea: Modificar Cantón.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 8/10/12	Fecha de Fin: 8/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selección el Cantónque se desea modificar, luego se cargan los valores actuales en un formulario brindando la opción de modificar los valores.	

Tabla 2.16. Eliminar Cantón.

Tareas	
Número de Tarea : 16	Numero de Historia: 6
Nombre de Tarea: Eliminar Cantón.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 8/10/12	Fecha de Fin: 8/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selecciona el Cantóny se brinda la posibilidad de eliminarlo.	

Tabla 2.17. Insertar provincia.

Tareas	
Número de Tarea : 17	Numero de Historia: 7
Nombre de Tarea: Insertar provincia	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 7/10/12	Fecha de Fin: 7/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se introducen en un formulario el nombre de la provincia.	

Tabla 2.18. Modificar provincia.

Tareas	
Número de Tarea : 18	Numero de Historia: 7
Nombre de Tarea: Modificar provincia	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 7/4/12	Fecha de Fin: 7/4/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona la provincia que se desea modificar, se cargan los valores en un formulario permitiendo modificar los valores.	

Tabla 2.19. Eliminar provincia.

Tareas	
Número de Tarea : 19	Numero de Historia: 7
Nombre de Tarea: Eliminar provincia.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 7/10/12	Fecha de Fin: 7/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selecciona la provincia y se le brinda la posibilidad de eliminarla.	

Tabla 2.20. Insertar parroquia

Tareas	
Número de Tarea : 20	Numero de Historia: 8
Nombre de Tarea: Insertar parroquia.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 9/10/12	Fecha de Fin: 9/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se introducen en un formulario los siguientes datos: el nombre, la provincia y el Cantón.	

Tabla 2.21. Modificar parroquia.

Tareas	
Número de Tarea : 21	Numero de Historia: 8
Nombre de Tarea: Modificar Parroquia.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 9/10/12	Fecha de Fin: 9/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selecciona la parroquia que se desea modificar, se cargan los datos en un formulario y se permiten modificar los valores.	

Tabla 2.22. Eliminar Parroquia.

Tareas	
Número de Tarea : 22	Numero de Historia: 8
Nombre de Tarea: Eliminar Parroquia.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 9/10/12	Fecha de Fin: 10/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se selecciona la parroquia que se desea eliminar.	

Tabla 2.23. Insertar Paciente

Tareas	
Número de Tarea : 23	Numero de Historia: 10
Nombre de Tarea: Insertar Paciente.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 11/10/12	Fecha de Fin: 11/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Se introducen en un formulario la cedula, nombres, apellidos, sexo, calle, localidad, número, provincia, cantón, parroquia, referencia, seguro médico y se acciona el botón salvar.	

Tabla 2.24. Modificar Paciente.

Tareas	
Número de Tarea : 24	Numero de Historia: 10
Nombre de Tarea: Modificar paciente.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 12/10/12	Fecha de Fin: 12/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Se selecciona el paciente que se desea modificar y se cargan los datos en un formulario, permitiendo modificación de los mismos.	

Tabla 2.24. Eliminar paciente.

Tareas	
Número de Tarea : 24	Numero de Historia: 10
Nombre de Tarea: Eliminar paciente.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 12/10/12	Fecha de Fin: 12/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Se selecciona el paciente y se elimina del sistema.	

Tabla 2.25. Insertarmedicamento.

Tareas	
Número de Tarea : 25	Numero de Historia: 11
Nombre de Tarea: Insertar Medicamento.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 13/10/12	Fecha de Fin: 13/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces	
Descripción: Se introduce el nombre del medicamento y se está liberado o no.	

Tabla 2.26. Eliminar medicamento.

Tareas	
Número de Tarea : 26	Numero de Historia: 11
Nombre de Tarea: Eliminar medicamento.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 13/12/12	Fecha de Fin: 13/12/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Se selecciona el medicamento y se brinda la posibilidad de eliminarla.	

Tabla 2.27. Modificar medicamento

Tareas	
Número de Tarea : 27	Numero de Historia: 11
Nombre de Tarea: Modificar medicamento.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 14/10/12	Fecha de Fin: 14/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Permite al usuario modificar los datos de un medicamento.	

Tabla 2.28. Crear intervención quirúrgica.

Tareas	
Número de Tarea : 28	Numero de Historia: 12
Nombre de Tarea: Crear intervención quirúrgica.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 15/10/12	Fecha de Fin: 15/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Permite al usuario crear una intervención quirúrgica.	

Tabla 2.29. Buscar contraindicaciones.

Tareas	
Número de Tarea : 29	Numero de Historia: 13
Nombre de Tarea: Buscar contraindicaciones.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 16/10/12	Fecha de Fin: 16/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Permite al usuario visualizar las contraindicaciones de un paciente.	

Tabla 2.30. Insertar enfermedad.

Tareas	
Número de Tarea : 30	Numero de Historia: 14
Nombre de Tarea: Insertar enfermedad.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 17/10/12	Fecha de Fin: 17/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Permite al usuario insertar una nueva enfermedad al sistema.	

Tabla 2.31. Modificar enfermedad.

Tareas	
Número de Tarea : 31	Numero de Historia: 14
Nombre de Tarea: Modificar enfermedad.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 18/10/12	Fecha de Fin: 18/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Permite al usuario modificar los datos de una enfermedad.	

Tabla 2.32. Eliminar enfermedad.

Tareas	
Número de Tarea : 32	Numero de Historia: 14
Nombre de Tarea: Eliminar enfermedad.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 18/10/12	Fecha de Fin: 18/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Permite al usuario eliminar del sistema a una enfermedad seleccionada.	

Tabla 2.33. Insertar sala

Tareas	
Número de Tarea : 33	Numero de Historia: 15
Nombre de Tarea: Insertar sala.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 19/10/12	Fecha de Fin: 19/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Permite al usuario crear una nueva sala en el sistema, especificando la cantidad de camas, la enfermera jefa y la especialidad.	

Tabla 2.34. Modificar sala.

Tareas	
Número de Tarea : 34	Numero de Historia: 15
Nombre de Tarea: Modificar sala.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 20/10/12	Fecha de Fin: 20/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces**.**	
Descripción: Permite al usuario modificar una sala existente.	

Tabla 2.35. Eliminar sala.

Tareas	
Número de Tarea : 35	Numero de Historia: 15
Nombre de Tarea: Eliminar sala.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 21/10/12	Fecha de Fin: 21/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Permite al usuario eliminar una sala existente.	

Tabla 2.36. Gestionar alta médica.

Tareas	
Número de Tarea : 36	Numero de Historia: 16
Nombre de Tarea: Gestionar alta médica.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 22/10/12	Fecha de Fin: 22/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Permite al doctor darle el alta a un paciente ingresado.	

Tabla 2.37. Filtrar paciente.

Tareas	
Número de Tarea : 37	Numero de Historia: 17
Nombre de Tarea: Filtrar paciente.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 23/10/12	Fecha de Fin: 23/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Permite al usuario buscar una paciente según los criterios de búsquedas.	

Tabla 2.38. Filtrar salas.

Tareas	
Número de Tarea : 38	Numero de Historia: 18
Nombre de Tarea: Filtrar salas.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 23/10/12	Fecha de Fin: 23/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Permite al usuario visualizar las salas buscándolas por diferentes criterios de búsquedas.	

Tabla 2.39. Listar capacidades.

Tareas	
Número de Tarea : 39	Numero de Historia: 19
Nombre de Tarea: Listar capacidades.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 29/10/12	Fecha de Fin: 29/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Permite al usuario listar las camas disponibles de una sala.	

Tabla 2.40. Reporte de pacientes por fechas.

Tareas	
Número de Tarea : 40	Numero de Historia: 20
Nombre de Tarea: Reporte de pacientes por fecha.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0.2
Fecha de Inicio: 30/10/12	Fecha de Fin: 30/10/12
Programador Responsable: Guillermo Ruiz Rendón, Danny Pincay Vinces.	
Descripción: Permite al usuario visualizar los pacientes porfechas.	