Universidad de las Ciencias Informáticas

Facultad 1

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Componente de lectura e interpretación de información neurofisiológica para el sistema XAVIA HIS

Autor: Eliedan Cala Padrón

Tutor(es): Dr.C. Arturo Orellana García

Ing. Ronal Cárdenas Cruz

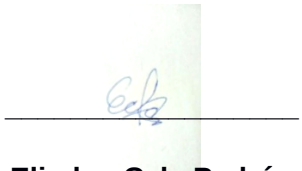
La Habana, 2020

“Año 62 de la Revolución”

Declaración de autoría

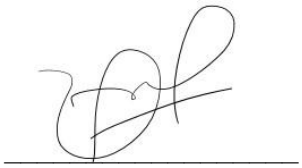
Declaro ser el autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los 24 días del mes de septiembre del año 2020.



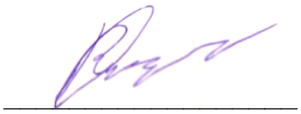
Eliedan Cala Padrón

Autor



Dr.C. Arturo Orellana García

Tutor



Ing. Ronal Cárdenas Cruz

Tutor

Dr.C. Arturo Orellana García: aorellana@uci.cu graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas en el año 2012. Se desempeña como líder del Grupo de Investigación de Minería de procesos y Asesor de Capacitación, Desarrollo e Investigación del Centro de Soluciones de Informática Médica. Ha liderado proyectos I+D+i de desarrollo de componentes de *software* a partir de minería de procesos para el análisis de procesos de negocio del entorno hospitalario. Investiga la Ingeniería de comportamiento, la medicina de precisión y el procesamiento de imágenes médicas. Tutora varias tesis de grado, maestrías y doctorados enfocados al análisis de procesos de negocio, la informática médica y otras áreas del conocimiento. Obtuvo el grado de Máster en Informática Aplicada en 2015 desarrollando una herramienta informática basada en técnicas de minería de procesos para identificar problemas en la ejecución de procesos de negocio. Doctor en Ciencias Técnicas desde 2016 presentando un modelo computacional para la detección de variabilidad en procesos de negocio del entorno sanitario aplicando minería de procesos.

Ing. Ronal Cárdenas Cruz: rccruz@uci.cu

Eliedan Cala Padrón: ecala@estudiantes.uci.cu

Dedicatoria

Comenzaré con alguien que hoy no está, pero definitivamente ayudo a formar mi persona inculcándome infinidad de valores. Esta tesis va dedicada a ti abuelo Nano.

El presente trabajo investigativo lo dedico principalmente a Dios, por ser el inspirador y darle fuerza para continuar en este proceso de obtener uno de los anhelos más deseados.

A mis padres Elieser Cala y Danaisy Padrón, por su amor, trabajo y sacrificio en todos estos años, gracias a ustedes he logrado llegar hasta aquí y convertirme en lo q soy. Es un orgullo y un privilegio el ser su hijo, son los mejores padres.

A mi hermano Eduardo Miguel Cala y en especial a mi hermana Daniser Cala por estar siempre presente, acompañándome y por su apoyo incondicional, que me ha brindado a lo largo de esta etapa de mi vida.

A toda mi familia porque con sus oraciones, consejos y palabras d aliento hicieron de mí una mejor persona y de una u otra forma me acompañan en todos mis sueños y metas. En especial a mi tía María del Carmen Cala, que no sólo en mi proceso universitario sino en todo momento ha estado presente como una madre más regalándome los mejores consejos y brindándome todo su apoyo.

Finalmente, y no menos importante a mi novia que siempre estuvo a mi lado desde el inicio de la carrera y más aún en este periodo de elaboración de tesis animándome y brindándome toda su ayuda, comprensión y amor.

Resumen

El Centro de Informática Médica de la Universidad de las Ciencias Informáticas desarrolla el Sistema de Información Hospitalaria XAVIA HIS, el cual permite facilitar la gestión de la información en las distintas áreas de una institución hospitalaria. A su vez, está compuesto por varios módulos, siendo el módulo Medios Diagnósticos el encargado de gestionar la información relacionada a la especialidad de Neurofisiología. Sin embargo, en esta especialidad se dificulta la gestión e interpretación de los resultados de estudios generados por sus equipos médicos. El objetivo de la presente investigación estuvo enfocado a desarrollar un componente de lectura e interpretación de información neurofisiológica para el sistema XAVIA HIS.

En el presente trabajo se utilizó como metodología de desarrollo AUP en su variante UCI, JBoss Developer Studio como entorno de desarrollo integrado, Java v1.6 como lenguaje de programación orientado a objeto, JBoss v4.2.2 como servidor de aplicaciones, PostgreSQL v9.4 como sistema gestor de bases de datos, pgAdmin III v1.10.5 como herramienta para la administración de bases de datos y Visual Paradigm v8.4 como herramienta CASE, el cual soporta UML v2.1 como lenguaje unificado de modelado y BPMN v.2.3 como notación para modelar los procesos del negocio.

Como resultado se obtuvo un componente para la lectura e interpretación de estudios de neurofisiología desde el sistema, además permite mejorar la gestión de la información del paciente en el sistema XAVIA HIS, integrando la información obtenida en la historia clínica digital, lo cual propicia el incremento en la calidad del servicio que se brinda en la especialidad de Neurofisiología.

Palabras clave: Medios Diagnósticos, Neurofisiología, sistema de información hospitalaria.

Indice

Introducción

El desarrollo tecnológico propicia una constante evolución de los procesos, los servicios y la informatización de los diferentes sectores de la sociedad. Las Tecnologías de la Información y las Comunicaciones (TIC) se define como una gama amplia de servicios, aplicaciones, y tecnologías, que utilizan diversos tipos de equipos y de programas informáticos. A menudo estos se transmiten a través de las redes de telecomunicaciones y cuyo principal fundamento radica en fortalecer e impulsar el desarrollo económico y social de un país. (Avella Martínez, et al., 2013)

La Salud es uno de los sectores priorizados de la sociedad. La introducción de tecnologías en este contexto constituye un objetivo de la ciencia moderna, debido a que se enfrentan continuamente a distintos cambios como la estructura demográfica o el aumento de enfermedades crónicas. Esto provoca un ascenso en la asistencia de pacientes y un gasto económico considerable en insumos y medicamentos. A la vez, se dificultad la disponibilidad de profesionales cualificados y gestión de los recursos de manera eficiente (Guanyabens y Calvet, 2015). Las instituciones sanitarias adquieren constantemente nuevas tecnologías que propician mejoras sustanciales en el trabajo y la atención al paciente. (Racoveanu,1995)

Una de las especialidades médicas que requiere el uso de las tecnologías informáticas es la Neurofisiología. La Neurofisiología Clínica es una especialidad médica que se encarga de la exploración del sistema nervioso central (cerebro, medula espinal) y periférico (músculos, nervios, órganos de los sentidos) a partir de diferentes técnicas y tecnología puntera con equipos de alto nivel. (Cardinali, 1991)

Varias especialidades médicas son demandantes de exploraciones neurofisiológicas lo que le confiere a esta en particular el carácter de ser un servicio central. (Cardinali, 1991). La Neurología, Psiquiatría, Medicina Interna, Oncología utilizan técnicas como electroencefalografía (Ramos-Argüelles, 2009), electromiografía (Villa Moreno, 2008), polisomnografía (Urquijo, 2001) y magnetoencefalografía (Maestú, C 1999).

Para la aplicación de estas técnicas se han desarrollado múltiples equipos médicos que ayudan a detectar problemas relacionados con esta especialidad y a entender mejor el funcionamiento del cerebro humano. Ejemplo de estos son el sistema de EEG digital MEDICID 4 (Cuba), Sistema de Monitoreo prolongado Video EEG Stellate (Canadá), Sistema para medición de potenciales evocados Neuropack Four mini (Nihon Kohden Corporation, Japan), Equipo de Potenciales Evocados y Electromiografía Neuropack Sigma (Nihon Kohden), Estimulador magnético: MagPro

(Dantec). Al concluir los estudios, estos equipos médicos proporcionan resultados, los cuales se muestran en varios formatos tales como: .txt, .pdf, .db, imágenes, ficheros con gráficos, audios o videos, etc.

El Centro de Neurociencias de Cuba (CNEURO) es una institución cubana dedicada a la investigación producción y comercialización de tecnologías avanzadas para el diagnóstico y tratamiento de enfermedades del cerebro. Esta institución incrementa y diversifica a partir de su labor investigativa, un grupo de tecnologías y su introducción en Programas Nacionales de Salud en la isla. Como parte del encadenamiento productivo universidad-empresa se crean proyectos de colaboración entre CNEURO y la Universidad de las Ciencias Informáticas (UCI).

La UCI cuenta en su estructura con centros de desarrollo de software tales como: el Centro de Software Libre (CESOL), Centro de Ideoinformática (CIDI) y el Centro de Informática Médica (CESIM), este último tiene como objetivo la informatización del sector sanitario. CESIM desarrolla proyectos como el Sistema para el manejo de datos de ensayos clínicos XAVIA SIDEC, Plataforma para la gestión de la información imagenológica XAVIA PACS-RIS y el Sistema de Gestión Hospitalaria XAVIA HIS.

El sistema XAVIA HIS se centra en la informatización de los procesos hospitalarios, el cual permite la gestión clínica y administrativa de los procesos médicos. Está compuesto por diferentes módulos que interconectan las diferentes áreas de una institución hospitalaria como son Admisión, Emergencia, Epidemiología, Banco de Sangre, Farmacia, Consulta Externa, Medios Diagnósticos, y otros. Además, gestiona la información de disímiles especialidades como Neurología, Ginecología, Psicología y Psiquiatría, sin embargo, existen especialidades médicas que no tienen implementadas todas sus funcionalidades, tal es el caso de Neurofisiología.

El proceso de atención al paciente, especialmente la gestión de estudios neurofisiológicos, se basa en dos actividades: la solicitud de evaluación y la generación de los informes de impresión diagnóstica.

En investigaciones previas de CESIM se desarrollaron funcionalidades para la informatización de las solicitudes de evaluación de estudios neurofisiológicos. Una vez que la gestión de las solicitudes se realiza desde el sistema XAVIA HIS el proceso queda incompleto al no culminar con la captura de la información generada en los estudios. Al emitirse de forma manual la impresión diagnóstica de estudios neurofisiológicos trae consigo ciertas consecuencias negativas en la toma de decisiones, debido a que la falta de una estandarización en los procedimientos médicos provoca ilegibilidad de lo escrito, por lo que en ocasiones la descripción del

diagnóstico no es la adecuada o falta información esencial en la Historia Clínica para que el especialista pueda realizar o evaluar el estudio indicado. Esto trae consigo, que los especialistas, al no tener información para levantar indicaciones o contraindicaciones sobre una determinada enfermedad, no cuenten con una base que apoye la emisión de diagnósticos y el seguimiento del paciente.

Además, la acumulación de documentación física induce a que el proceso sea más lento, donde la entidad solicitante lleva físicamente el estudio para realizarse y luego se devuelve de igual forma, lo cual atenta el tiempo de emisión de los resultados y en el tratamiento inmediato al paciente.

La situación descrita anteriormente trae consigo las siguientes limitantes:

- El sistema XAVIA HIS no propicia la generación de informes de emisión diagnóstica de estudios neurofisiológicos.
- La ausencia de la información clínica asociada a los informes neurofisiológicos impide el seguimiento oportuno de la condición de salud del paciente en estos estudios.
- La imposibilidad de recopilar la información adecuada de los estudios neurofisiológicos afecta la permanencia de los datos en una Historia Clínica Digital única, pues esta se realiza de forma manual y persisten los datos en dos formatos diferentes.
- Los especialistas no cuentan con información que apoye la emisión de diagnósticos.
- El Sistema XAVIA HIS no posee un mecanismo para la lectura e interpretación de los archivos que generan los equipos médicos para estudios neurofisiológicos que se encuentran en CNEURO.

Por todo lo anterior se plantea como problema de la investigación: ¿Cómo obtener información neurofisiológica de interés contenida en equipos EEG MEDICID 4 desde el sistema de información hospitalaria XAVIA HIS?

El **objeto de estudio** de la investigación se enmarca en el proceso de gestión de la información en la especialidad de Neurofisiología.

El **campo de acción** se centra en la lectura, interpretación y generación de informes de impresión diagnóstica de estudios neurofisiológicos.

Para dar solución al problema planteado se define como **objetivo general**: Desarrollar un componente de lectura e interpretación de información neurofisiológica para el

sistema XAVIA HIS.

Para dar cumplimiento al objetivo general se definen las siguientes **tareas de la investigación:**

1. Sistematización de los referentes teóricos y metodológicos de la investigación relacionado con la obtención de información neurofisiológica y los elementos fundamentales del objeto de estudio.
2. Desarrollo un componente para la lectura e interpretación de estudios neurofisiológicos.
3. Integración del componente de lectura e interpretación de estudios neurofisiológicos al sistema XAVIA HIS.
4. Validación del producto obtenido a partir de las pruebas de *software*.

- **Métodos teóricos (Sampieri, 2018):**

✓ Método analítico-sintético: en la presente investigación se utilizó este método para el análisis de la teoría y extracción de los principales conceptos a incluir en el marco teórico como: Sistema de Información, Sistema de Información Hospitalaria, Proceso de gestión de información en la especialidad de Neurofisiología, entre otros y luego sintetizarlos en la propuesta de solución.

✓ Método histórico-lógico: se utiliza este método para investigar acerca de otras soluciones similares como el Sistema Experto en Neurofisiología, el Sistema de evaluación del neurodesarrollo en niños, entre otros. Además, se analizó la evolución de los sistemas de información en salud y cómo se han modernizado para lograr la informatización de la gestión de la información generada en la especialidad de Neurofisiología.

✓ Método inductivo-deductivo: se utilizó en la aplicación de casos de pruebas al sistema, obteniendo conclusiones a partir de las respuestas proporcionadas por este.

✓ Modelación: se empleó como recurso auxiliar en la búsqueda teórica para caracterizar y representar mediante diagramas el campo de acción de la presente investigación. (Diaz, 2005)

- **Métodos empíricos:**

✓ Observación: se utilizó como método para validar la propuesta de solución.

- **Técnica de recopilación de datos (Sampieri, 2010)**

- ✓ Entrevista: posibilitó la recolección de información a través de conversaciones planificadas con especialistas de Neurofisiología.

La presente investigación está compuesta por tres capítulos, quedando estructurados de la siguiente manera:

Capítulo 1: Fundamentación teórica

Este capítulo contiene el marco conceptual en el que se muestran los principales elementos teóricos de la investigación. Se describen las diferentes herramientas y metodologías de desarrollo definidas para el desarrollo del sistema. Se hace referencia a los lenguajes de programación y tecnologías que apoyen el desarrollo de la solución. Además, se realiza un análisis de cómo se realiza el proceso de atención al paciente en la especialidad de Neurofisiología.

Capítulo 2: Análisis y diseño

En este capítulo se exponen las etapas del proceso de desarrollo de *software* que permiten describir la propuesta de solución, tales como: el modelado del negocio con descripción de proceso de negocio (DPN) y el modelado del sistema con descripción de requisitos de procesos (DRP). Se desarrollan los artefactos ingenieriles que responden a dichas fases y se presenta el patrón arquitectónico utilizado.

Capítulo 3: Implementación y pruebas

Este capítulo hace énfasis en la implementación de las clases y subsistemas de la solución propuesta. Se presenta el modelo de datos y se describen los atributos comunes entre las entidades. Además, se realiza un estudio de los mecanismos para el tratamiento de errores y las pruebas pertinentes para la validación de dicha solución.

Posteriormente se presentan las Conclusiones, se emiten las Recomendaciones, se listan las Referencias bibliográficas y se incluyen los Anexos, que proveen mayor información de la investigación realizada.

Capítulo 1: Fundamentación teórica de la investigación

El presente capítulo tiene como objetivo abordar los diferentes elementos que brindan la base teórica y conceptual para la investigación. Se realiza un análisis del proceso de atención al paciente en las consultas de Neurofisiología, en específico la obtención de ficheros generados en los estudios. Además, describe la metodología de desarrollo, las herramientas, tecnologías y lenguajes de programación a utilizar.

1.1. Conceptos asociados a la investigación

Resulta necesario y de importancia conocer los conceptos relacionados a la solicitud de estudios neurofisiológicos, para facilitar la comprensión del objeto de estudio de la investigación y conocer cómo funciona la especialidad de Neurofisiología.

Un Sistema de Información se puede definir como el conjunto de personas, normas, procesos, procedimientos, datos y recursos tecnológicos que funcionan articuladamente y que buscan facilitar y apoyar el desempeño de los usuarios para el cumplimiento de objetivos y metas previstas para el adecuado funcionamiento, desarrollo y crecimiento de la organización (Atienza, 2015).

1.1.1. Sistemas de Información Hospitalaria (HIS)

Debido al aumento de los servicios médicos en la sociedad, la informatización del proceso de tramitar la información aflora como la solución más óptima para la demora en la gestión de un cúmulo de información que surge de la atención a los pacientes. La creación de sistemas de información sanitaria responde a la necesidad de establecer un sistema de evaluación del estado de salud de la población y de las actividades de promoción y prevención y de asistencia sanitaria (Clínica, 2017).

En la actualidad existen múltiples sistemas informáticos empleados para modernizar los diferentes sectores de la sociedad como los Sistemas de Información Hospitalaria (HIS). Un HIS está orientado a satisfacer las necesidades de generación de información, para almacenar, procesar y reinterpretar datos médico-administrativos de la cualquier institución hospitalaria. Permitiendo la optimización de los recursos humanos y materiales y apoyando en las actividades en los niveles operativos, tácticos y estratégicos dentro de un hospital (Cerritos, y otros, 2017).

1.2. Proceso de gestión de información en la especialidad de Neurofisiología

La especialidad de Neurofisiología se encarga del estudio, la valoración y la modificación funcional del sistema nervioso (central y periférico), y de los órganos de los sentidos y musculares, tanto en condiciones normales como patológicas (Almárcegui Lafita, y otros, 2015).

Uno de los objetivos principales es almacenar la información relacionada a los tipos de estudios como:

- Estudio de Conducción Nerviosa (ECN): mide la velocidad y el grado de la actividad eléctrica en un nervio para determinar si está funcionando normalmente.

- Potencial Evocado Auditivo de Tallo Cerebral (Audiológico - Neurológico): permite diagnosticar diversas patologías o disfunciones del aparato auditivo y las vías nerviosas.

- Potencial Evocado Visual (PEV): permite diagnosticar problemas en los nervios ópticos aplicando un estímulo luminoso en los ojos.

- Potencial Evocado Somato Sensorial (miembros inferiores - superiores): valora la función en las vías nerviosas de la sensibilidad somática, estimulando un nervio periférico (en manos o pies) y recogiendo la respuesta en la corteza cerebral.

- EEG y Estudio de Sueño: diagnostica los trastornos del sueño, apoyados en el electroencefalograma.

1.2.1. Proceso de Atender paciente en la especialidad de Neurofisiología

Este proceso se lleva a cabo en el módulo Medios Diagnósticos del sistema XAVIA HIS, donde se involucran el especialista, la Estación de Registro (ER), la Estación de Evaluación (EE) y el paciente. A continuación, se explica cómo se atiende al paciente en la especialidad de Neurofisiología.

El paciente acude a la consulta de las especialidades de Neurocirugía o Neurología en donde, en dependencia de los síntomas que se presente, el doctor realiza una impresión diagnóstica. A partir de esta acción, el especialista mediante una solicitud realizada de forma manual le indica al paciente que debe hacerse el/los estudio(s) neurofisiológico(s).

A partir de lo mencionado anteriormente, el paciente acude a la ER con la solicitud, donde una vez realizado el/los estudio(s) se efectúa una nueva solicitud para el servicio de evaluación de los resultados de los mismos. A continuación, se muestran los datos comunes que son requeridos para la solicitud de evaluación de cada estudio.

- Nombre o código del Centro solicitante.
- Código del paciente.
- Edad.
- Fecha de realización.
- Medicación previa.
- Datos clínicos del paciente.
- Impresión diagnóstica.
- Observaciones.

En caso de ser una solicitud de evaluación del estudio Potencial Evocado Somato Sensorial (PESS) de miembros inferiores, se le añade:

- Estatura.
- Distancia tomada desde el sitio de estimulación hasta la espina ilíaca anterosuperior.

Si es una solicitud de evaluación del estudio Potencial Evocado Somato Sensorial (PESS) de miembros superiores, se recoge:

- Estatura.

- Distancia tomada desde el sitio de estimulación hasta la espina ilíaca anterosuperior.

- Distancia tomada desde el sitio de estimulación hasta el punto de Erb.

Mientras que, si es una solicitud de evaluación del estudio EEG y estudio de sueño, se selecciona:

- Condiciones en que se realizó el estudio: vigilia, sueño, sedación.

Una vez realizada la solicitud de evaluación de los estudios, pasa a la EE donde se redacta el informe final de un determinado estudio neurofisiológico, que a su vez se divide en 5 subinformes que constituyen cada uno de los estudios realizados. Posteriormente estos serán recibidos en la ER para su posterior revisión e impresión diagnóstica.

1.3. Sistemas informáticos existentes vinculados al campo de acción

Como parte de la investigación se analizaron varios sistemas informáticos que brindan un conjunto de funcionalidades relacionadas con la especialidad de Neurofisiología con el objetivo de analizar el proceso de obtención de la información de interés para esta investigación, específicamente en el desarrollo de una solicitud de evaluación de estudios neurofisiológicos. Como parte de los resultados se mencionan algunos de estos:

Lectura de Estudios Electrofisiológicos (LEES)

El Hospital General de Santiago de Cuba, dispone de un sistema informatizado que permite el análisis y la entrega de los resultados de los estudios neurofisiológicos, denominado Lectura de Estudios Electrofisiológicos (LEES). Este sistema está compuesto por dos módulos LEESID para la recepción y el almacenamiento de los parámetros de los estudios neurofisiológicos, y LEESOR para el análisis y la entrega de los resultados de los potenciales evocados de los pacientes. Además, garantiza un aumento en la calidad de los servicios que brinda la institución, debido a que gestiona la base de datos de los diferentes potenciales evocados y manipula toda la información necesaria para la entrada de datos, la confección del informe y la salida de los resultados.

EQUIPO NEURONIC E8.5

Electroencefalógrafo digital de 36, 64 y 128 canales, destinados al diagnóstico en las siguientes especialidades:
-Neurofisiología Clínica
-Neurología
-Psiquiatría
-Enfermedades Neurológicas y Psiquiátricas
-Psicofisiología
-Polisomnografía
-Monitoreo trans-operatorio

Prestaciones Generales.

Está constituido por un Bloque de Control que se comunica a una microcomputadora (IBM compatible), mediante una interfaz serie de alta velocidad (USB), permitiendo ejecutar diferentes softwares de registro y análisis, a través de los cuales el operador controla el desarrollo del experimento en curso (software de registro) y realiza el posterior análisis de los datos obtenidos (software de Aplicación Médica). Al Bloque de Control se pueden conectar Bloques de Amplificadores Bioeléctricos (de 36, 64 y 128 canales) así como los estimuladores Fótico y Psicofisiológico.

Neuronic EMG

Es un sistema diseñado para explorar la función e integridad del aparato neuromuscular a través de pruebas electroneuromiográficas, las cuales constituyen un valioso instrumento para la detección de las enfermedades que aquejan a este sistema. Está formado por los módulos:
-Electromiografía
-Estudios de Conducción Nerviosa
-Onda F
-Reflejo H
-Estimulación Nerviosa Repetitiva
-Reflejo de Parpadeo
-Administrador de Estudios

Neuronic Psicofisiología

Está diseñado siguiendo la lógica del trabajo experimental en Psicofisiología. Presenta los siguientes módulos:
-Registro de Psicofisiología

-Edición y Análisis de Pes

-Disminución de ruido de fondo

-Localización de los generadores de los componentes Endógenos

-Análisis estadísticos de los PRE

1.3.1. Valoración de los sistemas analizados

Por todo lo anteriormente expuesto se muestra a continuación una comparación teniendo en cuenta los siguientes criterios:

- Servicio de Neurofisiología: permite comprobar si en los sistemas mencionados se incluye durante el proceso de atención al paciente, la indicación de una solicitud de estudios neurofisiológicos.

- Solicitud de generación de informes de emisión diagnóstica de estudios neurofisiológicos: permite identificar si en los sistemas mencionados, se realiza una solicitud de los informes de los resultados de los estudios neurofisiológicos indicados.

- Anonimización de los datos del paciente: permite analizar los sistemas que cumplen con la condición de ocultar los datos personales del paciente, con el objetivo de que su identidad sea invisible para el especialista de Neurofisiología que evalúa los resultados de los estudios realizados.

- Tipo de licencia de *software*: permite comprobar de los sistemas mencionados si cumplen con la política de independencia tecnológica.

Tabla 1. Comparación teniendo en cuenta los criterios de selección. Fuente: Elaboración propia.

Características	**Lectura de Estudios Electrofisiológicos (LEES)**	**EQUIPO NEURONIC E8.5**	**Neuronic EMG**	**Neuronic Psicofisiología**
Funcionalidades	Este sistema está compuesto por dos módulos LEESID para la recepción y el almacenamiento de los parámetros de los estudios neurofisiológicos, y LEESOR para el análisis	Incluye los siguientes tipos de estudios neurofisiológicos: -Neurofisiología Clínica -Neurología -Psiquiatría -Enfermedades	Incluye los siguien tipos de estud neurofisiológicos: -Electromiografía -Estudios de Conducción Nerviosa -Onda F -Reflejo H	Incluye los siguien tipos de estud neurofisiológicos: -Registro de Psicofisiología -Edición y Análisis de Pes -Disminución de ruido de fondo

	y la entrega de los resultados de los potenciales evocados de los pacientes.	Neurológicas y Psiquiátricas -Psicofisiología -Polisomnografía -Monitoreo trans-operatorio	-Estimulación Nerviosa Repetitiva -Reflejo de Parpadeo -Administrador de Estudios	-Localización de los generadores de los componentes Endógenos -Análisis estadísticos de los PRE
Obtención de información de interés para ser transmitida a una historia clínica digital	No lo realiza	Gestión parcial debido a que durante todo el proceso de atención al paciente no se hace mención de una solicitud de generación de informes de emisión diagnóstica de estudios neurofisiológicos, es decir, se menciona la solicitud del estudio, se describe el proceso de realización del estudio en los equipos correspondientes y automáticamente la evaluación de los resultados con las observaciones emitidas por el especialista de Neurofisiología.		
Anonimización de los datos del paciente	No	No	No	No
Tipo de licencia de *software*	*Sowtware privativo*	*Software* libre	*Software* libre	*Software* libre

Teniendo en cuenta lo anterior se obtienen las siguientes conclusiones:

A pesar del desarrollo tecnológico que existe hoy en día se cuenta con muy pocos sistemas que gestionan funcionalidades relacionadas con la especialidad de Neurofisiología, algunos de estos no gestionan la información de todos los estudios neurofisiológicos, asumen en todo momento que las instituciones cuentan con todos los recursos necesarios para generar los informes de impresión diagnostica y la evaluación automática de sus resultados, aspecto que resulta complejo cuando en la institución no hay especialistas que realicen este último servicio, además, no garantizan la anonimización de los datos generales del paciente, siendo esta una funcionalidad esencial debido a que en algunos casos el paciente solicita ocultar su identidad durante este proceso.

En los sistemas identificados, así como en una revisión de la literatura no se identificaron funcionalidades que propicien la lectura e interpretación de los archivos que generan los equipos médicos de Neurofisiología.

1.4. Metodologías de desarrollo de software

El desarrollo de *software*, es uno de los sectores tecnológicos más competitivos, sin embargo, ha tenido una evolución constante en lo que se refiere a las metodologías con el objetivo de mejorar, optimizar procesos y ofrecer una mejor calidad. Son un enfoque estructurado para el desarrollo de *software* que incluyen modelos de sistemas, notaciones, reglas, sugerencias de diseño y guías de procesos (Modelos Y Metodologías Para El Desarrollo De Software, 2018). Estas metodologías se clasifican en dos grandes grupos: tradicionales y ágiles.

Las metodologías tradicionales se caracterizan por:

- Documentación exhaustiva de todo el proyecto.
- Los costos son altos al implementar un cambio.
- No es una buena solución para entornos volátiles.
- El equipo de desarrollo debe ser grande.

Mientras las metodologías ágiles se definen por:

- Buena solución para proyectos a corto plazo.
- Facilidad de respuesta a cambios repentinos en el desarrollo.
- Equipos de desarrollo pequeños.
- Ciclos de desarrollo cortos.

1.4.1. Metodología AUP-UCI

Al no existir una metodología de *software* universal, ya que toda metodología debe ser adaptada a las características de cada proyecto, se decide hacer una variación de la metodología Proceso Unificado Ágil (*AUP*, por sus siglas en inglés) de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI. De las cuatro fases que propone AUP: Inicio, Elaboración, Construcción y Transición, se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes tres fases de AUP en una sola que es Ejecución y se agrega la fase de Cierre (Sánchez, 2014).

AUP propone siete disciplinas: Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyecto y Entorno, por lo que se decide para el ciclo de vida de los proyectos de la UCI tener 7 disciplinas también (Sánchez, 2014). A partir de que la disciplina de Modelado de negocio propone tres variantes a utilizar en los proyectos: casos de uso del negocio, descripción de proceso de negocio y modelo

conceptual, y existen tres formas de encapsular los requisitos: casos de uso del sistema, historias de usuario y descripción de requisitos por proceso, surgen cuatro escenarios para modelar el sistema en los proyectos, quedando de la siguiente forma:

- Escenario 1: proyectos que modelen el negocio con casos de uso del negocio solo pueden modelar el sistema con casos de uso del sistema.
- Escenario 2: proyectos que modelen el negocio con modelo conceptual solo pueden modelar el sistema con casos de uso del sistema.
- Escenario 3: proyectos que modelen el negocio con descripción de proceso de negocio solo pueden modelar el sistema con descripción de requisitos por proceso.
- Escenario 4: proyectos que no modelen negocio solo pueden modelar el sistema con HU.

En la presente investigación se utilizará AUP – UCI como metodología de desarrollo de *software* en su escenario 3.

1.5. Ambiente de desarrollo

Las herramientas son un punto importante en la elaboración de una aplicación, son los programas que se reutilizan para automatizar las actividades definidas en el proceso de desarrollo de *software*; permiten crear y darle soporte al mismo, muchas veces haciendo el trabajo más factible y sencillo. En este epígrafe se especifican las tecnologías definidas por el proyecto Desarrollo del Sistema de Información Hospitalaria del CESIM y utilizadas para el desarrollo de la propuesta de solución. Se abordan las herramientas, marcos de trabajo y lenguajes de programación utilizados, estos son:

1.5.1. Herramientas de desarrollo

Se describen las herramientas utilizadas para el desarrollo de la solicitud de evaluación de estudios neurofisiológicos del sistema XAVIA HIS.

Herramienta de integración de desarrollo: JBoss Developer Studio v8.1

JBoss Developer Studio es un Entorno de Desarrollo Integrado (*IDE*, por sus siglas en inglés) certificado y basado en Eclipse para desarrollar, probar e implementar aplicaciones web avanzadas, aplicaciones web móviles, aplicaciones empresariales transaccionales, entre otras (RedHat, 2018).

Contiene un amplio conjunto de herramientas y soporte para varios modelos y marcos de programación, entre los que se incluyen (RedHat, 2018):

- Java™ Enterprise Edition 6.
- JavaServer Faces (JSF).
- Enterprise JavaBeans (EJB).
- HTML5.

Herramienta de modelado: Visual Paradigm v8.0

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite crear todos los tipos de diagramas de clases, ingeniería inversa, generar código desde diagramas y generar documentación (Paradigm, 2017).

Se caracteriza por:

- Generación de código para Java y exportación como HTML.
- Fácil de instalar y actualizar.
- Compatibilidad entre ediciones.
- Soporte de UML versión 2.1.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.

Sistema Gestor de Base de Datos: PostgreSQL v.9.4.1

Es un sistema de base de datos relacional de objetos de código abierto. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema (Postgresql, 2017).

A continuación, se presentan algunas de las características más importantes y soportadas por PostgreSQL (Ecured, 2018):

- Soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes, cadenas de bits, entre otros.
- Incorpora una estructura de datos array.
- Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, entre otro.
- Incluye herencia entre tablas.

PgAdmin v3.0

PgAdmin es una herramienta de código abierto para la administración de bases de datos PostgreSQL. Se desarrolla bajo la licencia PostgreSQL Licence a partir de la versión v1.10 (PgAdmin, 2018).

Incluye (V-espino, 2018):

- Interfaz administrativa gráfica.
- Herramienta de consulta SQL.
- Editor de código procedural.
- Agente de planificación SQL.

1.5.2. Tecnologías

Java Server Faces (JSF) v1.2

La tecnología JSF (*Java Server Faces*, por sus siglas en inglés) constituye un marco de trabajo para la creación de interfaces de usuario del lado del servidor, dirigido a aplicaciones web basadas en tecnología Java. Aporta mayor libertad al diseñador y mejora los informes de errores que tiene JSF. (Sánchez, 2018). El objetivo del desarrollo de aplicaciones mediante JSF, es construir aplicaciones web que se parezcan a las aplicaciones de escritorio (Facelets, 2018).

RichFaces v.3.3.0

RichFaces es una librería de código abierto que añade los componentes de Ajax en aplicaciones existentes de JSF sin recurrir a JavaScript. Aprovecha la librería de Java Server Faces para incluir validaciones, instalaciones de conversión y la gestión de los recursos estáticos y dinámicos. De esta forma permite al desarrollador ahorrar tiempo y aprovechar las características de los componentes para crear aplicaciones web, con mejor apariencia visual (Álvarez, 2013).

Ajax4jsf

Ajax es una librería de código abierto que se integra totalmente en la arquitectura de JSF y extiende la funcionalidad de sus etiquetas dotándolas con tecnología Ajax de forma limpia y sin añadir código JavaScript. Mediante este *framework* se puede variar el ciclo de vida de una petición JSF, recargar determinados componentes de la página sin necesidad de recargarla por completo, realizar peticiones al servidor automáticas, control de cualquier evento de usuario, entre otros. Esta librería permite dotar a la

aplicación JSF de contenido mucho más profesional con muy poco esfuerzo (Ramos, 2017).

Facelets

Facelets es un lenguaje de declaración de páginas potente pero ligero que se utiliza para crear vistas de JavaServer Faces utilizando plantillas de estilo HTML y para construir árboles de componentes. Las características de Facelets incluyen lo siguiente (Oracle, 2017):

- Uso de XHTML para la creación de páginas web.
- Soporte para el lenguaje de expresión (EL).
- Plantillas para componentes y páginas.

Las ventajas de Facelets para proyectos de desarrollo a gran escala incluyen lo siguiente (Oracle, 2017):

- Soporte para la reutilización de código a través de plantillas y componentes compuestos.
- Tiempo de compilación más rápido.
- Resuelve validaciones, conversiones, mensajes de error e internacionalización.

JBoss Seam v.2.1.1

JBoss Seam es un *framework* que integra y unifica los distintos *standars* de la plataforma Java EE 5.0, pudiendo trabajar con todos ellos siguiendo el mismo modelo de programación.

Ha sido diseñado intentado simplificar al máximo el desarrollo de aplicaciones, basando el diseño en *Plain Old Java Objects* (POJO's) con anotaciones. Estos componentes se usan desde la capa de persistencia hasta la de presentación, poniendo todas las capas en comunicación directa. El núcleo principal de Seam está formado por las especificaciones Enterprise JavaBeans 3 (EJB3) y JavaServer Faces (JSF) (Neodoo, 2018).

Java Persistence API v.2.1

Java Persistence API (*JPA*, por sus siglas en inglés) proporciona un modelo de persistencia basado en Plain Old Java Objects (POJO's) para mapear bases de datos relacionales en Java. El Java Persistence API fue desarrollado por el grupo de

expertos de EJB 3.0, aunque su uso no se limita a los componentes *software* EJB. También puede utilizarse directamente en aplicaciones web y aplicaciones clientes; incluso fuera de la plataforma Java EE, por ejemplo, en aplicaciones Java SE. En su definición, se han combinado ideas y conceptos de las principales librerías de persistencia como Hibernate, Toplink y JDO, y de las versiones anteriores de EJB (Andalucía, 2015).

Enterprise JavaBeans v.3.0

Una de las metas de la arquitectura EJB es la de poder escribir de manera fácil aplicaciones de negocio orientadas a objetos y distribuidas, basadas en el lenguaje de programación JAVA. El propósito de EJB 3 es el de proveer el soporte de la arquitectura de EJB y al mismo tiempo reducir la complejidad para el desarrollo de aplicaciones empresariales (Herrera, 2014).

Hibernate v.3.2.5

Hibernate es una herramienta de mapeo objeto-relacional para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones. Hibernate es *software* libre, distribuido bajo los términos de la licencia GNU LGPL (Componente web para el análisis de información clínica usando la técnica de Minería de Datos por agrupamiento, 2013).

Java Enterprise Edition (JEE) v6:

Java Platform Enterprise Edition (*JavaEE*, por sus siglas en inglés) es una plataforma de programación para desarrollar y ejecutar *software* de aplicaciones en lenguaje de programación Java con arquitectura de N niveles distribuida (Franky, 2016). Permite el manejo de diversos detalles mediante una programación simple y al no ser privativa, el sistema que se desarrolle usando Java puede ser comercializado en el mundo entero (Java Enterprise Edition, 2018).

JBoss Server v4.2.2

JBoss Server es un servidor de aplicaciones Java y actualmente es el más utilizado. Cientos de profesionales y desarrolladores de código abierto han contribuido a su creación y desarrollo. Provee servicios extendidos de almacenamiento de datos en memoria y de manera persistente. Permite la integración de todas las tecnologías y

herramientas utilizadas por Seam. Es actualizado e integrado constantemente con lo último del estado del arte de las aplicaciones web (Soldano, y otros, 2017).

1.5.3. Lenguajes

Se describen a continuación los lenguajes utilizados para el desarrollo de la propuesta de solución.

Lenguaje de programación: Java versión v1.6

Java es un lenguaje de programación Java diseñado por la compañía Sun Microsystems Inc, con el propósito de crear un lenguaje que pudiera funcionar en sistemas de ordenadores heterogéneos y que fuera independiente de la plataforma en la que se vaya a ejecutar (Garro, 2018).

Algunas de sus principales características se enuncian a continuación (Wikilibros, 2018):

- Lenguaje totalmente orientado a Objetos.
- Disponibilidad de un amplio conjunto de bibliotecas.
- Portable.
- Alto rendimiento.
- Interpretado y compilado a la vez.

XHTML v.1.0

Lenguaje extensible de marcación hipertexto (*XHTML*, por sus siglas en inglés) es similar a HTML, pero con algunas diferencias que lo hacen más robusto y aconsejable para la modelación de páginas web. Este lenguaje suprime todas las etiquetas y atributos que sirven para definir el aspecto y sólo se dejan las etiquetas que sirven para definir el significado de cada elemento de la página. (Alvarez, 2011)

JavaScript v.1.8

Javascript es un lenguaje de programación que surgió con el objetivo inicial de programar ciertos comportamientos sobre las páginas web, respondiendo a la interacción del usuario y la realización de automatismos sencillos. Es el motor de las aplicaciones más conocidas en el ámbito de Internet: Google, Facebook, Twitter, Outlook (Alvarez, 2011).

XML v.1.0

XML es un lenguaje de marcado similar a HTML. Sus siglas significan *Extensible Markup Language* (Lenguaje de Marcado Extensible) y es una especificación de W3C como lenguaje de marcado de propósito general. El propósito principal del lenguaje es compartir datos a través de diferentes sistemas, como Internet. Hay muchos lenguajes basados en XML como: XHTML, MathML, SVG, XUL, XBL, RSS, y RDF. También puedes crear uno propio. (Muñoz, 2019)

CSS v.3.0

Hojas de estilo en cascada (CSS, por sus siglas en inglés) es un lenguaje que define la apariencia de un documento escrito en un lenguaje de marcado. Así, a los elementos de la página web creados con HTML se les dará la apariencia que se desee utilizando CSS: colores, espacios entre elementos, tipos de letra, separando de esta forma la estructura de la presentación. Esta separación entre la estructura y la presentación es muy importante, ya que permite que sólo cambiando los CSS se modifique completamente el aspecto de una página web. (Alvarez, 2011)

Notación para el Proceso de Modelado de Negocio (BPMN) v2.3

Business Process Model and Notation (*BPMN*, por sus siglas en inglés), en español Modelo y Notación de Procesos de Negocio, es una notación gráfica estandarizada que permite el modelado de procesos de negocio, en un formato de flujo de trabajo. Su objetivo es respaldar el modelado de procesos de negocios al proporcionar una notación estándar que sea comprensible para los usuarios de negocios, pero que represente una semántica de procesos complejos para los usuarios técnicos (Group, 2018).

Lenguaje Unificado de Modelado: UML v2.1

UML es un lenguaje de modelado orientado a objetos que permite representar gráficamente los elementos estáticos y dinámicos de una aplicación de *software* (Vidal, y otros, 2014). Es una herramienta usada por analistas funcionales y analistas-programadores (Krall, 2006-2029).

Su objetivo es visualizar, especificar, construir y documentar los artefactos que se crean durante el proceso de desarrollo. Involucra todo el ciclo de vida del proyecto y está pensado para varios lenguajes y plataformas tales como ASP, PHP, entre otros (Larman, 2016).

1.6. Conclusiones del capítulo

Los sistemas para la gestión de la información neurofisiológica identificados en el estado del arte propiciaron conocer las características del proceso de atención al paciente en la especialidad de Neurofisiología.

Los ficheros de estudios de neurofisiología analizados permitieron identificar la información relevante para constituir el informe de resultados de la especialidad.

Las herramientas, tecnologías y lenguajes de programación definidas para el Sistema de Información Hospitalaria XAVIA HIS son adoptadas para desarrollar la solución propuesta con el objetivo de lograr una integración entre ambas soluciones.

Capítulo 2: Análisis y diseño de la propuesta de solución

En el presente capítulo se realiza la descripción de los procesos del negocio, desarrollándose el modelo de negocio y definiendo los actores y trabajadores. Se especifican los requisitos funcionales y no funcionales, mediante el diagrama de casos de uso del sistema, las relaciones entre los actores y casos de uso del sistema, así como las descripciones textuales de cada uno de ellos.

1.7. Modelo conceptual

El modelo conceptual tiene como propósito organizar y representar el conocimiento de un área o campo específico asociado a un sistema de gestión o de información. Está orientado a identificar y describir la información del dominio de un problema. Es un artefacto útil que permite obtener conocimiento acerca de cómo se desenvuelve el problema en el contexto real. (Fowler y Scott, 1999)

En la Figura 1 se muestra el modelo conceptual, detallando cada uno de los conceptos asociados al entorno del problema y las relaciones entre ellos.

Toda la información referente al modelo conceptual de la solución forma parte de Medios diagnósticos y se encuentra en el expediente del proyecto Desarrollo de XAVIA HIS en el documento CESIM_PRODUCTO_Modelo_conceptual_MD. CESIM_PRODUCTO_Modelo_conceptual_MD, 2020

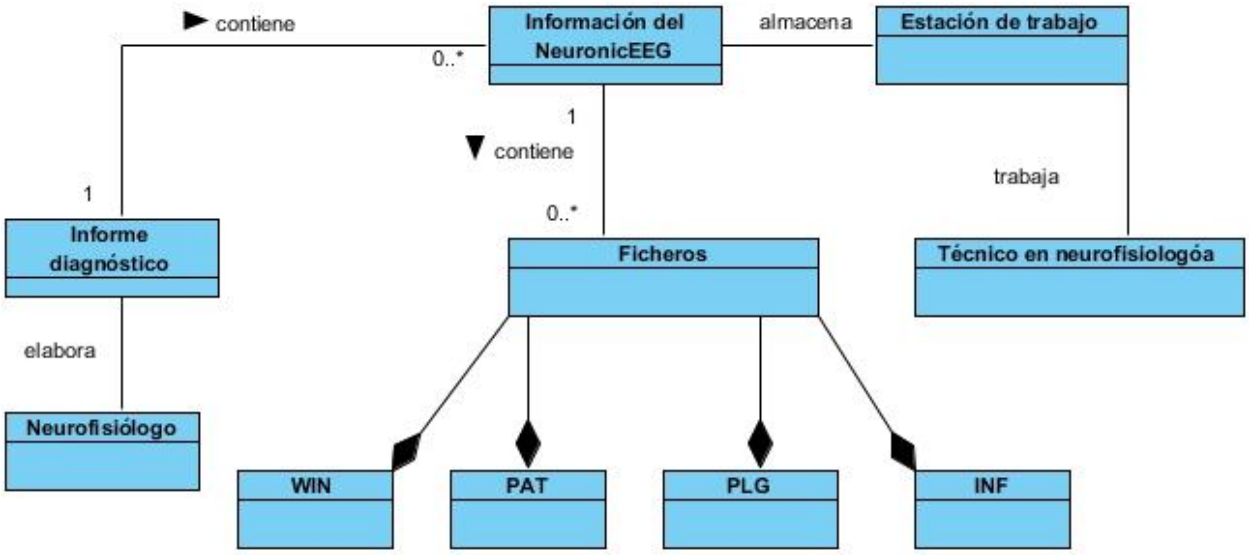


Figura 1. Modelo conceptual Medios diagnósticos.

2.1. Modelo de negocio

El modelo del negocio describe cada proceso del negocio, especificando sus datos, actividades, roles y las reglas del mismo. Facilita un entendimiento entre clientes y desarrolladores y se enfoca en comprender los problemas actuales de la organización e identifica mejoras potenciales.

2.1.1. Identificación de roles del entorno del negocio

Una vez identificados los procesos de negocio, es preciso encontrar los involucrados en su realización. A continuación, se muestran los roles que se identificaron en la solución propuesta:

Tabla 2. Actores del negocio. Fuente: Elaboración propia.

Actores del negocio	Descripción
Paciente	Personas egresadas a una institución hospitalaria.
Estación de Evaluación (EE)	Se encarga de evaluar los resultados de los estudios neurofisiológicos y redactar el informe
Estación de Registro (ER)	Se encarga de realizar una solicitud de evaluación del estudio neurofisiológico.
Especialista en Neurología/Neurocirugía	Se encargado de realizar una impresión diagnóstica y la solicitud de estudios neurofisiológicos.

2.1.2. Diagrama del proceso del negocio

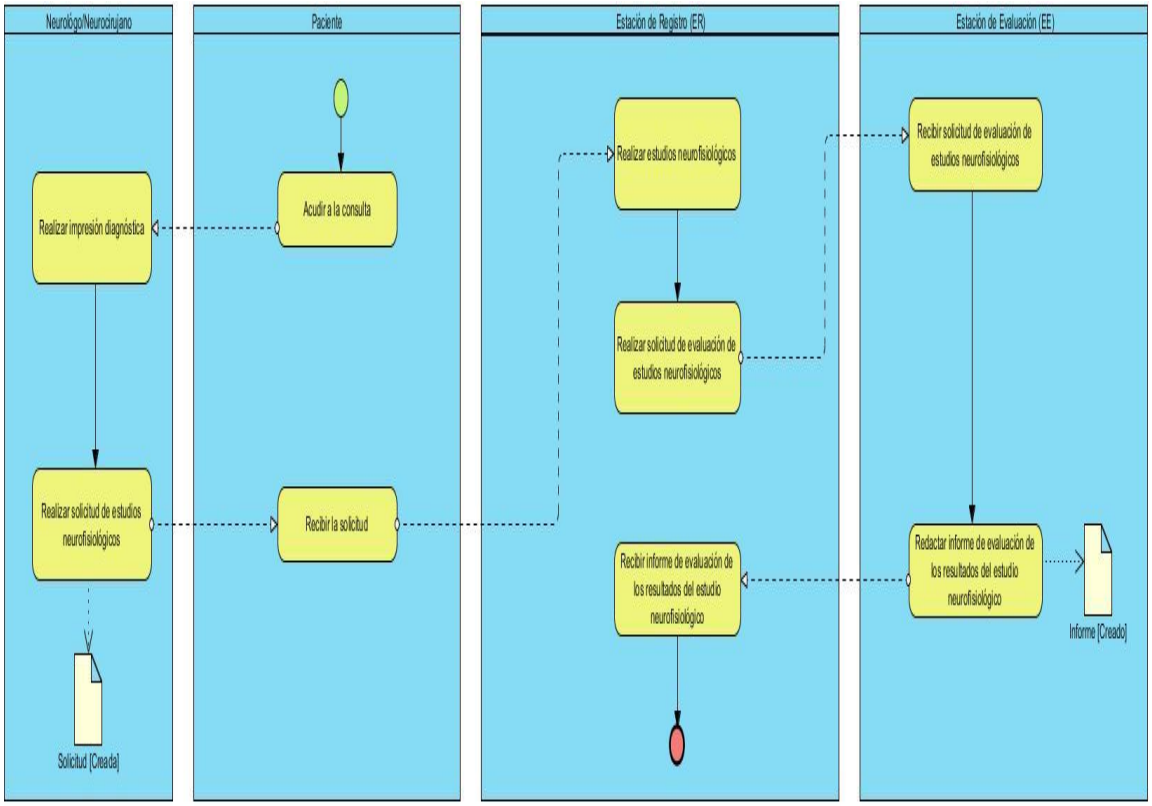


Figura 2. Diagrama del proceso Atender paciente en Neurofisiología. Fuente: Elaboración propia.

En el diagrama anterior se representa el flujo completo del proceso de Atender paciente en la especialidad de Neurofisiología para una mejor comprensión del negocio. Especificar que en la actual investigación se incide en la obtención de información contenida en los ficheros de estudios neurofisiológicos.

2.2. Requisitos de software

Los requerimientos para un sistema son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos requerimientos reflejan las necesidades que poseen los clientes ante un problema actual (Sommerville, 2005).

2.2.1 Requisitos funcionales. Descripción de requisitos por procesos

Los requerimientos funcionales de un sistema describen lo que el sistema debe hacer. Estos requerimientos dependen del tipo de *software* que se desarrolle, en los posibles usuarios del *software* y del enfoque general tomado por la organización al redactar requerimientos (Sommerville, 2005).

A continuación, se muestran los requisitos funcionales definidos y sus descripciones:

Tabla 3. Descripción de requisitos funcionales. Fuente: Elaboración propia.

RF	Nombre	Descripción	Prioridad	Complejidad
1	Capturar fichero.	Permite capturar el fichero sobre el cual se desea trabajar.	Alta	Alta
2	Leer fichero.	Permite leer el contenido del fichero.	Media	Media
3	Interpretar fichero.	Permite entender la información del fichero.	Alta	Alta
4	Capturar datos dentro del fichero.	Permite la captura de los datos significativos después de la interpretación de la información.	Alta	Alta
6	Transformar el formato del fichero.	Convertir al estándar de mensajería HL7-CDA para q pueda ser enviado al Sistema XAVIA-HIS.	Alta	Alta
6	enviar datos	Permite enviar los datos estandarizados en HL7-CDA.	Alta	Alta

2.2.2 Especificación de requisitos por procesos

Tabla 4. Descripción del requisito Capturar Fichero. Fuente: Elaboración propia.

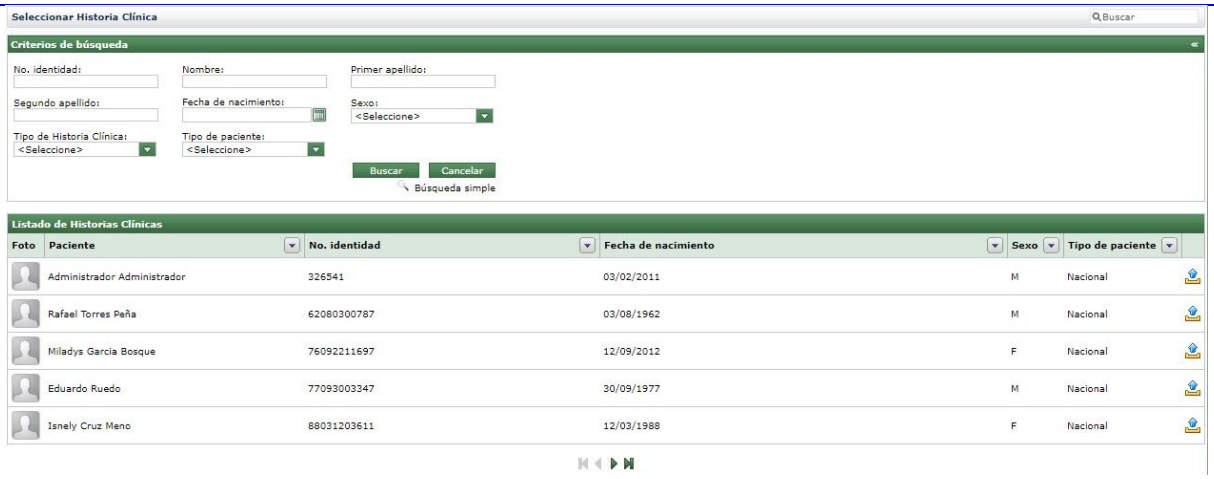
RF 1. Capturar fichero

Descripción textual	El requisito inicia cuando el actor accede al módulo Medios Diagnósticos y el sistema brinda la posibilidad de obtener el informe de evaluación de estudios neurofisiológicos de un paciente.
Actores	Estación de Registro
Precondiciones	Se debe seleccionar la historia clínica de un paciente.
Flujo de eventos	
Flujo básico Capturar fichero.	
1.	El requisito inicia cuando el actor accede a la opción Obtener informe de evaluación de estudios.
2.	El sistema brinda la posibilidad de: • Seleccionar historia clínica del paciente. Se ejecuta el requisito, ver requisito **Elementos Comunes::Seleccionar Historia Clínica**. • Seleccionar el tipo de estudio neurofisiológico indicado: - Estudio de Conducción Nerviosa - Potencial Evocado Auditivo de Tallo Cerebral Audiológico. - Potencial Evocado Auditivo de Tallo Cerebral Neurológico. - Potencial Evocado Visual. - Potencial Evocado Somato Sensorial Inferior. - Potencial Evocado Somato Sensorial Superior. - EGG y Estudio de Sueño Y permite: • Aceptar. • Cancelar. Ver **Flujo alternativo 1:** “Cancelar operación”
3.	El actor Selecciona el informe deseado.
4.	El sistema obtiene los datos contenidos en los ficheros generados por el equipo médico MEDICID 4
5.	El actor selecciona la opción Aceptar.
6.	El sistema valida y envía los datos. Si hay datos incompletos. **Ver Flujo Alternativo 9:** “Existen datos incompletos.” Si hay datos incorrectos. **Ver Flujo Alternativo 10:** “Existen datos incorrectos.”
7.	El sistema adiciona los datos de la solicitud de evaluación de estudio neurofisiológico y muestra los datos del mismo. Se ejecuta el requisito, ver requisito **Ver detalles de solicitud de evaluación de estudio neurofisiológico.**
8.	El requisito termina.
Secciones	
Sección 1 <Nombre de la pestaña 1 en la interfaz>	
1.	N/A

Flujos alternativos	
Flujo alternativo 1 “Cancelar operación”	
1.	El actor selecciona la opción Cancelar operación.
2.	El sistema regresa a la interfaz de Seleccionar historia clínica.
3.	Regresa al paso 7 del Flujo básico.
Flujo alternativo 2 “Estudio de Conducción Nerviosa”	
1.	El sistema brinda la posibilidad de introducir los datos: • Nombre de centro • Medicación previa • Impresión diagnóstica • Observaciones
2.	El actor introduce los datos requeridos.
3.	El actor selecciona la opción Aceptar.
4.	Regresa al paso 6 del **Flujo básico**.
Flujo alternativo 3 “Potencial Evocado Auditivo de Tallo Cerebral Audiológico”	
1.	El sistema brinda la posibilidad de introducir los datos: • Nombre de centro • Medicación previa • Impresión diagnóstica • Observaciones
2.	El actor introduce los datos.
3.	El actor selecciona la opción Aceptar.
4.	Regresa al paso 6 del **Flujo básico**.
Flujo alternativo 4 “Potencial Evocado Auditivo de Tallo Cerebral Neurológico”	
1.	El sistema brinda la posibilidad de introducir los datos: • Nombre de centro • Medicación previa • Impresión diagnóstica • Observaciones
2.	El actor introduce los datos.
3.	El actor selecciona la opción Aceptar.
4.	Regresa al paso 6 del **Flujo básico**.
Flujo alternativo 5 “Potencial Evocado Visual”	
1.	El sistema brinda la posibilidad de introducir los datos: • Nombre de centro • Medicación previa • Impresión diagnóstica

	• Observaciones
2.	El actor introduce los datos.
3.	El actor selecciona la opción Aceptar.
4.	Regresa al paso 6 del **Flujo básico**.
Flujo alternativo 6 "Potencial Evocado Somato Sensorial Inferior"	
1.	El sistema brinda la posibilidad de introducir los datos: • Nombre de centro • Medicación previa • Impresión diagnóstica • Observaciones • Estatura • Distancia tomada desde el sitio de estimulación hasta la espina ilíaca anterosuperior
2.	El actor introduce los datos.
3.	El actor selecciona la opción Aceptar.
4.	Regresa al paso 6 del **Flujo básico**.
Flujo alternativo 7 "Potencial Evocado Somato Sensorial Superior"	
1.	El sistema brinda la posibilidad de introducir los datos: • Nombre de centro • Medicación previa • Impresión diagnóstica • Observaciones • Estatura • Distancia tomada desde el sitio de estimulación hasta la espina ilíaca anterosuperior • Distancia tomada desde el sitio de estimulación hasta el punto de Erb
2.	El actor introduce los datos.
3.	El actor selecciona la opción Aceptar.
4.	Regresa al paso 6 del **Flujo básico**.
Flujo alternativo 8 "EGG y Estudio de sueño"	
1.	El sistema brinda la posibilidad de introducir los datos: • Nombre de centro • Medicación previa • Impresión diagnóstica • Observaciones • Condiciones en que se realizó el estudio: ○ Vigilia ○ Sueño ○ Sedación

2.	El actor introduce los datos.		
3.	El actor selecciona la opción Aceptar.		
4.	Regresa al paso 6 del **Flujo básico**.		
Flujo alternativo 9 “Existen datos incompletos”			
1.	El sistema muestra un indicador (asterisco rojo) sobre los campos incompletos.		
2.	Regresa al paso 3 del Flujo básico.		
Flujo alternativo 10 “Existen datos incorrectos”			
1.	El sistema muestra un indicador (asterisco rojo) sobre los campos incorrectos con un mensaje en dependencia del error cometido.		
2.	Regresa al paso 3 del **Flujo básico**.		
Pos-condiciones			
1.	Se creó la solicitud de evaluación de un estudio neurofisiológico.		
Validaciones			
1.	N/A		
Conceptos	**N/A**	N/A	
	N/A	N/A	
Restricciones del sistema	Si la primera vez que se entra al sistema, no hay información para mostrar solo se verá el botón adicionar solicitud de evaluación de estudios neurofisiológicos, una vez que se adicione el tipo de solicitud entonces se podrá modificar o eliminar solicitud de obtener el informe de evaluación correspondiente al estudio seleccionado.		
Dependencias	**Obligatoria**	N/A	
	Opcional	Ver detalles de solicitud de evaluación de estudio neurofisiológico.	
Requisitos especiale	N/A		
Asuntos pendientes	N/A		
Prototipo elemental de interfaz gráfica del Crear solicitud de evaluación de estudio neurofisiológico			



Formatos de entrada/salida
N/A
Entradas
N/A
Salidas
N/A

2.2.3 Requisitos no funcionales

Los requerimientos no funcionales, son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. (Sommerville, 2005).

Los requisitos no funcionales van a ser afiliados a los requisitos definidos para el proyecto XAVIA HIS en la documentación del producto(Centro de Informática Médica, 2018).

2.3. Descripción de la arquitectura

La arquitectura de *software*, de acuerdo con la definición que brinda la IEEE Std 1471-2000, es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución (Camarena et al., 2016).

2.3.1. Patrón arquitectónico Modelo Vista Controlador (Cambiar foto)

La arquitectura de la solución propuesta está basada en el patrón arquitectónico Modelo Vista Controlador (MVC). Este patrón garantiza la reducción del esfuerzo de programación, la cual es necesaria en la implementación de sistemas múltiples. (Romero Fernández & González Díaz, 2014). Permite separar cada una de las capas

del sistema: el modelo, donde se encuentran los datos y las reglas del negocio; la vista, que muestra la información del modelo al usuario; y el controlador, que gestiona las entradas del usuario.

A partir de la descripción realizada de las tecnologías en el anterior acápite, se evidencia la utilización del MVC de la siguiente manera:

- En la capa de modelo se maneja Hibernate como herramienta de mapeo objeto relacional, que es la implementación de EJB 3.0 y JPA.
- La capa de la vista está desarrollada con JSF, compuesta por páginas XHTML. Se emplea componentes Seam de interfaz de usuario y Facelets como motor de plantillas. Además, se utiliza las librerías Ajax4JSF y RichFaces.
- La capa controladora está constituida por clases controladoras que definen la lógica del negocio del módulo. Se utiliza Seam como *framework* de integración.

Figura 3. Interrelación entre los elementos del patrón MVC.

2.4. Modelo del diseño

El modelo del diseño describe la realización física de los casos de uso centrándose en los requisitos funcionales y no funcionales, permitiendo descomponer los trabajos de implementación en partes más adaptables que puedan ser llevadas a cabo por diferentes equipos de desarrollo (Gamma, Helm, & Vlissides, 2013).

2.4.1. Utilización de los patrones de diseño

Los patrones de diseño son unas técnicas para resolver problemas comunes en el desarrollo del *software* y otros ámbitos referentes al diseño de interacción o interfaces. Cada patrón describe un problema que ocurre en nuestro entorno y también, el núcleo de la solución al problema, de forma que puede utilizarse un varias veces (Tabares, 2014).

Patrones GRASP

Los Patrones Generales de *Software* para Asignar Responsabilidades (GRASP, por sus siglas en inglés), son parejas de problema y solución, que codifican buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades. Describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones (ApunteRUP.doc, 2017). A continuación, se mencionan los patrones GRASP utilizados en la investigación:

2.4.2. Diagrama de paquetes

Para la confección del modelo del diseño, se plantea una estructura de paquetes que sea manejable para la implementación. Cada uno de estos paquetes está compuesto por diversos subpaquetes que a su vez contienen los diagramas de clases del diseño. Todas las clases están agrupadas en el paquete Repositorio de clases. En Sesiones se encuentran todas las clases controladoras agrupadas en paquetes, donde un paquete tiene las controladoras autogeneradas, otro las personalizaciones que se hacen sobre las controladoras autogeneradas y uno para las controladoras propias del proceso. El paquete Entidades contiene a su vez otros paquetes con las entidades autogeneradas y personalizadas. Todas las vistas están contenidas en el paquete Vistas. Estos paquetes se relacionan entre ellos ya que las vistas consultan y actualizan las entidades e invocan a las controladoras y estas modifican las entidades.

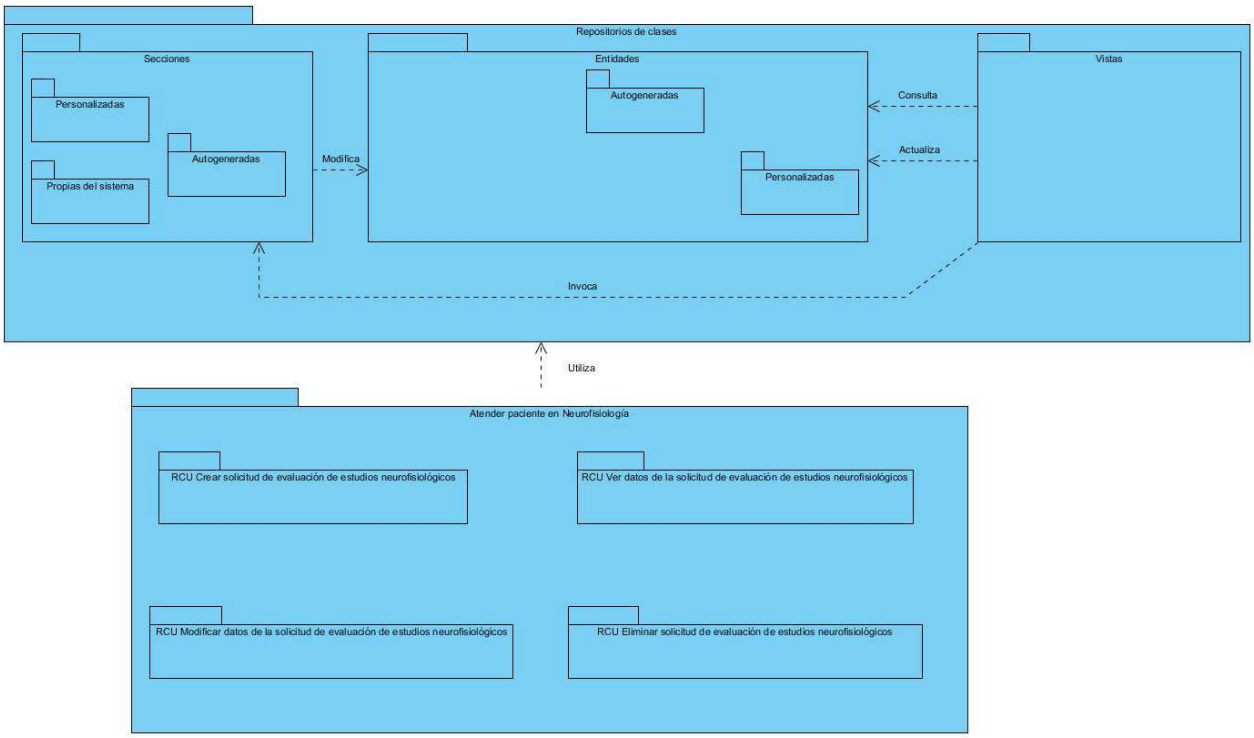


Figura .4. Interrelación entre los elementos del patrón MVC.

2.5. Diagrama de clases del diseño

Los diseños de clases permiten obtener de forma estática la representación de los requisitos que se lleva a cabo a través de las clases del sistema y sus relaciones. Su principal utilidad radica en mostrar a través de sus atributos y métodos la estructura de las clases que después serían traducidas al lenguaje de programación Java.

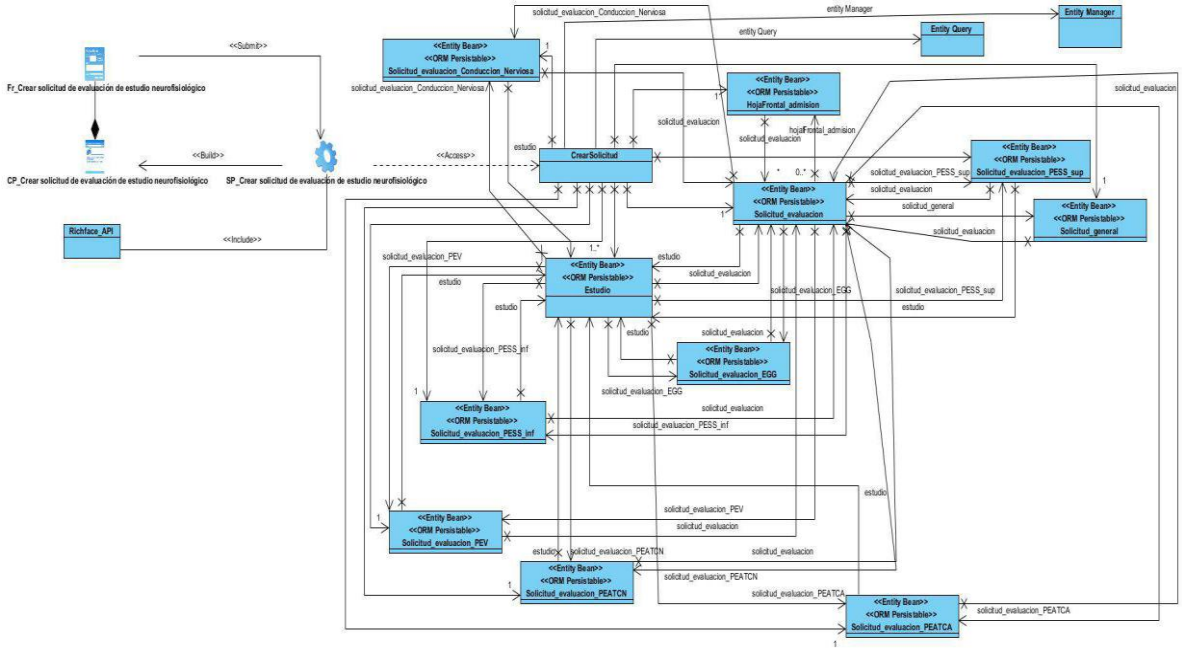


Figura 5. Diagrama de clases del diseño: Generar informe diagnóstico de estudio neurofisiológico. Fuente: Elaboración propia.

2.6. Modelo de datos

Un modelo de datos constituye una definición lógica y abstracta de los objetos y operadores que en conjunto constituyen la máquina abstracta con la que interactúan los usuarios. Se obtiene a partir del diagrama de clases persistentes y su forma se expresa mediante un diagrama de UML.

2.6.1. Modelo de datos de la propuesta de solución

A continuación, se presenta el modelo de datos correspondiente al componente de lectura e interpretación de información neurofisiológica:

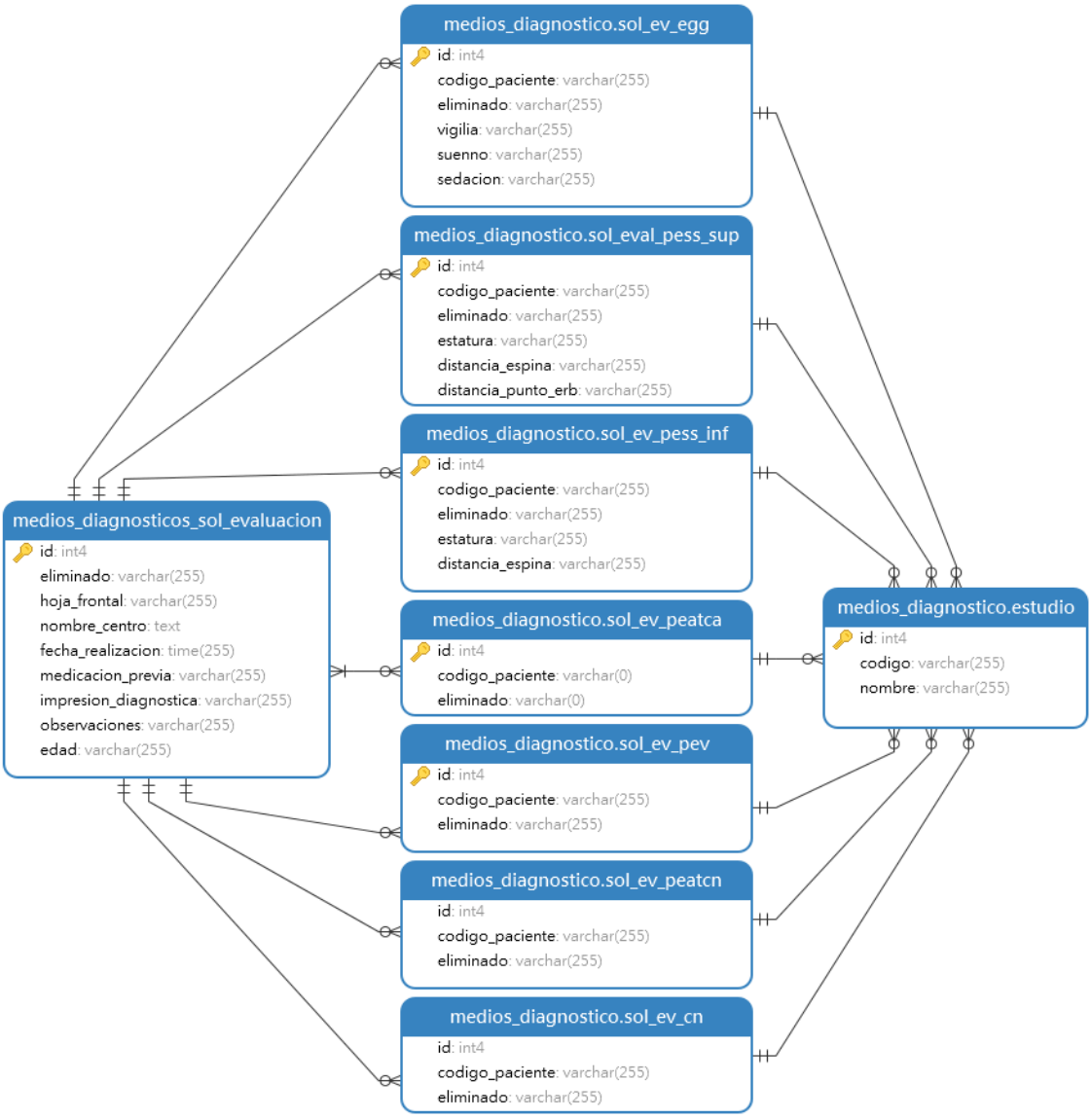


Figura 6. Modelo de datos de la solicitud de evaluación de estudios neurofisiológicos. Fuente: Elaboración propia.

En la siguiente tabla se describen los atributos que son comunes en todas las entidades de la solicitud de los estudios:

Tabla 5. Atributos comunes de las entidades de la solicitud de evaluación de estudios neurofisiológicos. Fuente: Elaboración propia.

Atributo	Tipo	Descripción
identificador	bigint	Id necesario en cada entidad para las referencias en las relaciones entre tablas. Es la llave principal en todas las entidades.
eliminado	varchar	Permite la eliminación lógica con que cuenta el sistema, cuando está en verdadero indica que la entidad está eliminada.
código_paciente	varchar	Permite identificar al paciente.
id_estudio	bigint	Id de la entidad medios_diagnostico. estudio.

Tabla 6. Atributos de la entidad medios_diagnostico. estudio. Fuente: Elaboración propia.

Medios_diagnostico.estudio		
Atributo	**Tipo**	**Descripción**
identificador	bigint	Id de la entidad medios_diagnostico. estudio.
código	varchar	Permite identificar al estudio.
nombre	varchar	Nombre del estudio.

Tabla 7. Atributos de la entidad medios_diagnostico. solicitud_evaluación. Fuente: Elaboración propia.

Medios_diagnostico.solicitud_evaluación		
Atributo	**Tipo**	**Descripción**
identificador	bigint	Id de la entidad medios_diagnostico. solicitud_evaluación.
código	varchar	Permite identificar al estudio.
nombre_centro	varchar	Nombre del centro solicitante.

eliminado	varchar	Permite la eliminación lógica con que cuenta el sistema, cuando está en verdadero indica que la entidad está eliminada.
hoja_frontal	varchar	Permite obtener los datos de la hoja frontal como los datos personales del paciente.
fecha_realización	varchar	Fecha de realización de la solicitud.
medicación_previa	varchar	Medicación previa emitida por el médico.
impresión_diagnostica	varchar	Impresión diagnóstica emitida por el médico.
observaciones	varchar	Observación emitida por el médico.
edad	varchar	Edad del paciente.
id_solic_ev_pess_inf	int	Id de la entidad medios_diagnostico. solic_ev_pess_inf.
id_solic_ev_pess_sup	int	Id de la entidad medios_diagnostico. solic_ev_pess_sup.
id_solic_ev_pev	Int	Id de la entidad medios_diagnostico. solic_ev_pev.
id_solic_ev_egg	int	Id de la entidad medios_diagnostico. solic_ev_egg.
id_solic_ev_peatca	int	Id de la entidad medios_diagnostico. solic_ev_peatca.
id_solic_ev_peatcn	int	Id de la entidad medios_diagnostico. solic_ev_peatcn.
id_solic_ev_cn	int	Id de la entidad medios_diagnostico. solic_ev_cn.

2.7. Propuesta del Sistema

Como resultado de la investigación desarrollada y con el objetivo de dar solución a la problemática existente se propone el desarrollo de un componente para la lectura e interpretación de estudios de la especialidad de Neurofisiología para el módulo Medios Diagnósticos. Esta componente tiene como objetivo lograr una estandarización mediante la especificación de los datos que son requeridos para que las instituciones hospitalarias puedan obtener un informe de evaluación organizado y con la

información necesaria para una impresión diagnóstica. A su vez permite mejorar la gestión de la información en el sistema XAVIA HIS mediante la Historia Clínica Electrónica del paciente, en la cual se encontrarán anexados los resultados de los estudios previos, propiciando el incremento en la calidad del servicio que se brinda y en la disponibilidad de la información asociada al paciente en la especialidad de Neurofisiología. Además, garantiza la anonimización del paciente, con el fin de garantizar la confidencialidad de sus datos generales.

2.7.1. Implementación del componente de lectura e interpretación de ficheros de estudios de neurofisiología

El modelo de implementación está compuesto por un conjunto de componentes y subsistemas que forman la parte física de la implementación de un sistema. Este describe como se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje de programación utilizado.

2.7.2. Diagrama de componentes

Los diagramas de componentes son utilizados para mostrar los elementos de *software* y la relación existente entre ellos en un sistema. Además, se utilizan para mostrar las dependencias de compilación de los ficheros de código, relaciones de derivación entre ficheros de código fuente y ficheros que son resultados de la compilación.

DC 1. Crear solicitud de evaluación de estudio neurofisiológico

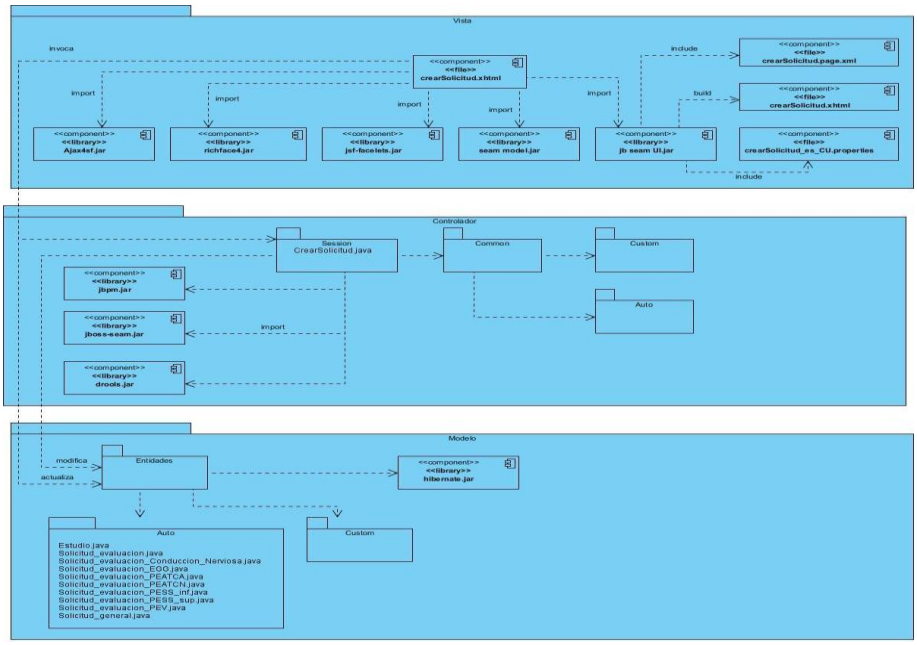


Fig.7. Diagrama de componente de lectura e interpretación de estudios neurofisiológicos. Fuente: Elaboración propia.

2.8. Conclusiones del capítulo

La definición de los requisitos funcionales y no funcionales que permitieron identificar las funcionalidades de la solución propuesta a desarrollar. A raíz de esto, se determinaron 6 requisitos funcionales con sus respectivas descripciones y se adoptaron los requisitos no funcionales del sistema XAVIA HIS.

La adopción del patrón arquitectónico el MVC propició un diseño de la propuesta acorde con las características del sistema XAVIA HIS.

Capítulo 3: Desarrollo y pruebas de la propuesta de solución

En el presente capítulo se realiza un estudio de los mecanismos para el tratamiento de errores y se presenta el modelo de datos y se describen los atributos comunes entre las entidades del modelo de datos. La seguridad informática es abordada con la finalidad de prevenir acciones que puedan afectar la integridad, disponibilidad y confidencialidad de la información contenida y se valida la solución propuesta a través de una estrategia de pruebas de software.

3.1. Arquitectura de software

Para el desarrollo de la solución y teniendo en cuenta la tecnología propuesta, se define como parte de la línea base de la arquitectura la implementación del patrón de diseño de arquitectura de software Modelo-Vista-Controlador. Este patrón arquitectónico permite la separación de los datos de una aplicación, la interfaz de usuario y la lógica de control, en tres componentes distintos: el modelo, que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia; la vista, que compone la información que se envía al cliente y los mecanismos interacción con éste; y el controlador, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno (ALICANTE, 2014).

El patrón MVC es un paradigma que divide las partes que conforman una aplicación en el Modelo, las Vistas y los Controladores, permitiendo la implementación por separado de cada elemento, garantizando así la actualización y mantenimiento del software de forma sencilla y en un reducido espacio de tiempo.

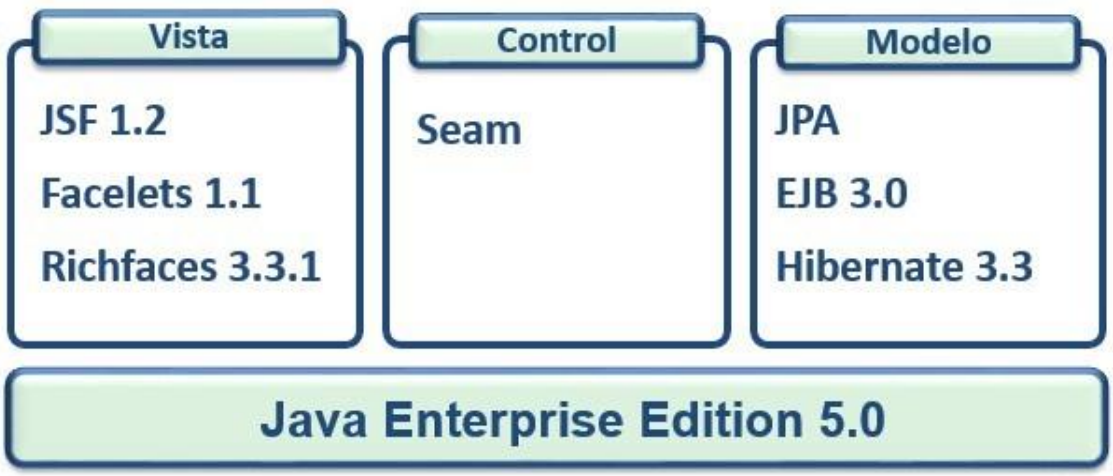


Fig. 6. Representación de las tecnologías en el Patrón MVC.

3.2. Estándares de codificación

La adopción de estándares de estilo y codificación son de vital importancia para asegurar la calidad del software. El uso de los mismos tiene ventajas tales como:

- Asegurar la legibilidad del código entre distintos programadores, facilitando el *debugging* del mismo.
- Proveer una guía para el encargado de mantenimiento/actualización del sistema, con código claro y bien documentado.
- Facilitar la portabilidad entre plataformas y aplicaciones.

Las convenciones de código o estándares de codificación son importantes para los programadores por las siguientes razones:

- El 80% del coste del código de un programa va a su mantenimiento.
- Casi ningún software es mantenido toda su vida por el autor original.
- Las convenciones de código mejoran la lectura del software lo que permite entender código nuevo de manera más óptima y rápida.
- Si distribuyes tu código fuente como un producto, necesitas asegurarte de que está bien hecho y presentado como cualquier otro producto.

A continuación, se presentan algunos de los estándares de codificación definidos para el proyecto Desarrollo del Sistema de Información Hospitalaria del CESIM y adoptados para el desarrollo del componente de lectura e interpretación de estudios neurofisiológicos:

- Se debe utilizar como idioma el español, las palabras no se acentuarán.
- Las líneas en blanco mejoran la facilidad de lectura separando secciones de código que están lógicamente relacionadas. Se deben usar siempre dos líneas en blanco en las siguientes circunstancias:
    - ✓ Entre las secciones de un fichero fuente.
    - ✓ Entre las definiciones de clases e interfaces.
- Se debe usar siempre una línea en blanco en las siguientes circunstancias:
    - ✓ Entre métodos.
    - ✓ Entre las variables locales de un método y su primera sentencia.

- ✓ Antes de un comentario de bloque o de un comentario de una línea.
- ✓ Entre las distintas secciones lógicas de un método para facilitar la lectura.

- Se debe dar un espacio en blanco en la siguiente situación:
  - ✓ Entre una palabra clave del lenguaje y un paréntesis.
- Respecto a las normas de inicialización, declaración y colocación de variables, constantes, clases y métodos:
  - ✓ Todas las instancias y variables de clases o métodos empezarán con minúscula. Las palabras internas que lo forman, si son compuestas, empiezan con su primera letra en mayúsculas. Los nombres de variables no deben empezar con los caracteres subguión "_" o signo de peso "$", aunque ambos están permitidos por el lenguaje.
  - ✓ Los nombres de las clases deben ser sustantivos, cuando son compuestos tendrán la primera letra de cada palabra que lo forma en mayúscula. Mantener los nombres de las clases simples y descriptivas. Usar palabras completas, evitar acrónimos y abreviaturas.
- Respecto a la identación y longitud de la línea:
  - ✓ Se deben emplear cuatro espacios como unidad de identación. Los tabuladores deben ser exactamente cada 8 espacios.
  - ✓ Evitar las líneas de más de 80 caracteres, ya que no son manejadas bien por muchas terminales y herramientas.

3.3. Tratamientos de errores

Durante el tiempo de ejecución de un sistema pueden fracasar diferentes rutinas, es a esto a lo que comúnmente se le llama excepción. No tienen por qué ser errores. Las excepciones son el mecanismo recomendado para tratar los errores que se produzcan durante la ejecución de las aplicaciones. Cuando ocurre un error dentro de un método Java, automáticamente se crea un objeto Exception el cual es tratado en el sistema. Este objeto contiene información sobre la excepción, incluyendo su tipo y el estado del programa (Prieto Saez, 2016).

El uso de diferentes tecnologías y la integración que existe entre ellas, permiten capturar y controlar posibles situaciones desde diferentes puntos de la aplicación. En las páginas clientes se cuenta con un conjunto de componentes denominados

validadores, que permiten establecer tipos de datos y formatos, controlando el envío de información correcta, al servidor.

En el sistema XAVIA HIS se propone el tratamiento de excepciones principalmente en las regiones críticas de código, es decir, donde los datos son insertados o modificados en la base de datos, así como en el proceso de validación. El control de la navegación, en caso de ocurrir una excepción que implique una redirección, se maneja mediante los pages.xml, estos se encargan de capturar globalmente las excepciones y ejecutar las instrucciones determinadas. Para controlar el resto de las excepciones se utiliza el componente FacesMessages del marco de trabajo Seam, el cual brinda un potente conjunto de excepciones predefinidas. Este se encarga de mostrar los mensajes que se manejan a través del objeto facesMessages inyectado en las clases controladoras tratando los mensajes por tipo (error, alerta y notificación).

3.4. Seguridad informática

La seguridad informática es el conjunto de métodos y herramientas destinados a proteger los bienes informáticos en una institución. El término seguridad informática está estrechamente relacionado con 3 aspectos fundamentales de cualquier sistema de información.

- ✓ Confidencialidad: la información o los activos informáticos son accedidos solo por las personas autorizadas.
- ✓ Integridad: los activos o la información solo pueden ser modificados por las personas autorizadas y de la forma autorizada.
- ✓ Disponibilidad: los activos informáticos son accedidos por las personas autorizadas en el momento requerido.

Al desarrollar una aplicación informática, la seguridad es un tema muy importante. En el sistema XAVIA HIS esta cuestión adquiere mayor relevancia pues se gestiona información relacionada con los procesos de atención al paciente, por lo cual es de gran interés que la misma esté bien protegida. Posteriormente se presentan un conjunto de acciones llevadas a cabo para asegurar la seguridad de la información:

- ✓ Partiendo del principio de mínimo privilegio y con la finalidad de asegurar la confidencialidad de la información contenida, en el sistema se dan los permisos de acuerdo con la función que ocupa el usuario en el mismo. Ello permite solo tener acceso a las secciones, páginas, directorios, opciones del menú y servicios que respondan directamente a su rol. Para acceder al sistema se cuenta con un módulo de autenticación donde se deberá introducir un usuario y una contraseña.

- ✓ Las informaciones médicas relacionadas con los pacientes que vayan a ser intercambiadas con otras instituciones de salud, emplearan formato HL7-CDA (Documentos de Arquitectura Clínica). Este formato permite definir permisos de visualización, estableciendo la capacidad de que la información que contiene el documento sea vista solo por quiénes tienen privilegios suficientes para verla. El grado de confidencialidad lo establece de forma general en el encabezado, pero también a nivel de sección, de forma que puede haber secciones con información más sensible que tengan un nivel de confidencialidad mayor que el resto del documento.

- ✓ Durante la confección de los CDA se emplea el mecanismo de seguridad firma electrónica por parte de los especialistas que certifican las informaciones médicas relacionadas con los pacientes garantizando su autoría y no adulteración.

3.5. Pruebas de software

Las pruebas constituyen un elemento de vital importancia en el desarrollo de software debido a que garantizan la obtención de una aplicación con las características requeridas e identificadas en la fase de requisitos. Este proceso es constante y no concluye hasta entregada la aplicación al cliente final.

La estrategia de pruebas pudiera verse como un proceso en espiral, al igual que el proceso de desarrollo de software, comenzando desde la ingeniería del sistema, el levantamiento de requisitos, el diseño de los requerimientos y la implementación de los mismos. Al mismo nivel de cada una de estas etapas de desarrollo de software se realizan las pruebas del sistema, de validación, de integración y de unidad (Sommerville, 2005). En la figura se muestra la representación descrita anteriormente:

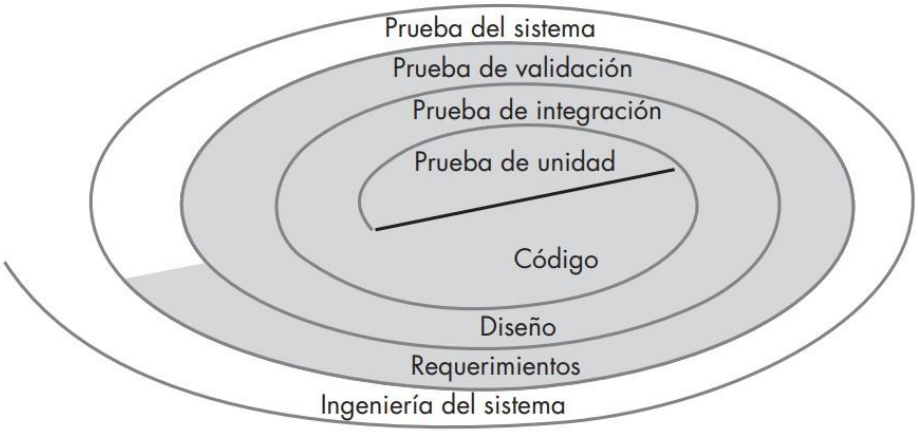


Fig. 7. Estrategia de pruebas.

Para lograr verificar y validar la propuesta de solución se diseñó una estrategia de prueba basada en los niveles de integración, sistema y validación o aceptación. Por cada uno de estos niveles de prueba se define el tipo de prueba que se realiza (funcional o no funcional), el método de prueba (caja blanca o caja negra), la técnica asociado al método escogido, así como la generación de los casos de prueba y los resultados obtenidos luego de ser aplicados en el nivel correspondiente.

3.5.1. Tipos de prueba

Funcionales: Se basan en funciones y prestaciones (descritas en documentos tales como la especificación de requisitos) y su interoperabilidad con sistemas específicos, y pueden llevarse a cabo en todos los niveles de prueba. Las técnicas basadas en la especificación sirven para obtener condiciones de prueba y casos de prueba a partir de la funcionalidad de un software o sistema. Las pruebas funcionales tienen en cuenta el comportamiento externo del software (pruebas de caja negra). (Muller, et al., 2010)

Regresión: Son la prueba reiterada de un programa ya probado, después de haber sido modificado, con vistas a localizar defectos surgidos o no descubiertos como resultado del cambio o de los cambios Estos defectos pueden estar en el software objeto de las pruebas, o en cualquier otro componente de software asociado o no asociado. Se realizan cuando el software, o su entorno, sufren modificaciones. Pueden realizarse en todos los niveles de prueba e incluyen pruebas funcionales, no funcionales y estructurales. (SEI, 2010)

3.5.2. Métodos de prueba

Caja negra. Técnica de partición equivalente

Las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software. Permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra intenta encontrar errores de las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación. (Pressman, 2010)

De las técnicas de prueba de caja negra se utiliza la técnica de partición equivalente la cual divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se dirige a la

definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.

3.5.3. Resultado al aplicar la técnica de partición equivalente.

Para llevar a cabo el desarrollo de las pruebas de software se generaron los casos de prueba referentes a cada uno de los requisitos de software. El caso de prueba del requisito Capturar fichero con 3 escenarios a probar, mientras que el requisito Transformar datos del fichero cuenta con 4 escenarios. Al aplicar las pruebas se obtuvieron los siguientes resultados:

Tabla 8. Resultado de aplicar la prueba de caja negra al RF1 Capturar Fichero. Fuente (elaboración propia)

raciones	NC detectadas	Correspondencia	terfaz	Funcionalidad	NC resueltas
1era					
2da					
3ra					

3.5.4. Pruebas de regresión

Las pruebas de regresión se deben llevar a cabo cada vez que se hace un cambio en el sistema, tanto para corregir un error como para realizar una mejora. No es suficiente probar solo los componentes modificados o añadidos, o las funciones que en ellos se realizan, sino que también es necesario controlar que las modificaciones no produzcan efectos negativos sobre el mismo u otros componentes. (Sommerville 2011)

Normalmente, este tipo de pruebas implica la repetición de las pruebas que ya se han realizado previamente, con el fin de asegurar que no se introducen errores que puedan comprometer el funcionamiento de otros componentes que no han sido modificados y confirmar que el sistema funciona correctamente una vez realizados los cambios. Las pruebas de regresión pueden incluir (Pressman 2010):

- ✓ La repetición de los casos de pruebas que se han realizado anteriormente y están directamente relacionados con la parte del sistema modificada.
- ✓ La revisión de los procedimientos manuales preparados antes del cambio, para asegurar que permanecen correctamente.

- ✓ La obtención impresa del diccionario de datos de forma que se compruebe que los elementos de datos que han sufrido algún cambio son correctos.

Las pruebas de regresión fueron aplicadas en el desarrollo de la hoja de consulta de Geriatría en cada una de las iteraciones realizadas como parte de la estrategia de pruebas de caja negra.

3.5.5. Pruebas de aceptación

Cuando se construye software a medida para un cliente, se llevan a cabo una serie de pruebas de aceptación para permitir que el cliente valide todos los requisitos. Las realiza el usuario final en lugar del responsable del desarrollo del sistema, una prueba de aceptación puede ir desde un informal “paso de prueba” hasta la ejecución sistemática de una serie de pruebas bien planificadas. De hecho, la prueba de aceptación puede tener lugar a lo largo de semanas o meses, descubriendo así errores acumulados que pueden ir degradando el sistema. (Pressman, 2010)

La mayoría de los desarrolladores de productos de software llevan a cabo un proceso denominado prueba alfa y beta para descubrir errores que parezca que solo el usuario final puede descubrir. La prueba alfa se lleva a cabo, por un cliente, en el lugar de desarrollo. Se usa el software de forma natural con el desarrollador como observador del usuario y registrando los errores y los problemas de uso. Las pruebas alfa se llevan a cabo en un entorno controlado. (Pressman,2010)

A la propuesta de solución se le fueron aplicadas pruebas de aceptación de tipo alfa, pues el cliente, en este caso el proyecto, junto al desarrollador realizaron pruebas guiadas por casos de prueba mediante el método de caja negra, utilizando la técnica de partición equivalente.

Conclusiones generales

Con el desarrollo del componente de lectura e interpretación de ficheros de neurofisiología se cumple con el objetivo de la investigación, por lo que se concluye que:

- El análisis de los sistemas permitió definir las características que debe poseer la propuesta de solución, haciendo posible estructurar la información a recoger en la solucion.
- El análisis de las herramientas brindó la posibilidad de desarrollar la propuesta de solución bajo las mismas tecnologías en las que está soportada el sistema XAVIA HIS, facilitando así su posterior integración.
- El proceso de desarrollo de software, guiado por la metodología AUP UCI, hizo posible contar con artefactos ingenieriles acorde a los documentados en el expediente de proyecto del sistema XAVIA HIS.

Recomendaciones

Para futuras investigaciones se recomiendan las siguientes acciones:

- Desarrollar componentes para capturar la información de otros equipos médicos de estudios de neurofisiología.

Referencias bibliográficas

1. Almárcegui Lafita, C. y Sáenz de Cabezón, A. 2015. Guía o itinerario formativo tipo Neurofisiología clínica. s.l.: Servicio de Neurofisiología Clínica Hospital Universitario Miguel Servet, 2015.

2. Álvarez Lorenzo, A., y otros. SLD237 Desarrollo de la especialidad psicología del módulo Consulta Externa del sistema ALAS-HIS. La Habana: s.n., 2013.

3. Andrés, Fernández. 2014. TIC y salud: promesas y desafíos para la inclusión social. [En línea] 2014. [Citado el: 16 de mayo de 2020.] http://www.eclac.cl/cgibin/getprod.asp?xml=/socinfo/noticias/paginas/9/40689/P40689.xml&xsl=/socinfo/tpl/p18f.xsl&base=/socinfo/tpl/top-bottom.xsl 2011.

4. Atienza, Oscar Alfredo. 2015. Historia clinica informática única. Una herramienta en la mejora de procesos en salud pública. Ciudad de Córdoba: s.n., 2015.

5. Avella Martínez, Laura Yaneth y Parra Ruiz, Paola Patricia. 2013. Tecnologías de la información y la Comunicación (TICS) en el sector de la salud. Bogotá: s.n., 2013.

6. Baum, Analía y Campos, Fernando. 2014. La importancia de los CDA -Documentos de Arquitectura Clínica-. [aut. libro] Baum Analía y Campos Fernando. Estándares para el intercambio de documentos clínicos. Buenos Aires: s.n., 2014.

7. Bradanovic. Conceptos Básicos de Seguridad Informática. [En línea]. 2009. [Citado el: 7 de abril de 2020]. Disponible en: http://www.bradanovic.cl/pcasual/ayuda3.html

8. Bascón Pantoja, Ernesto. 2011. El patrón de diseño Modelo-Vista-Controlador(MVC) y su implementación en Java Swing. 2011.

9. Bass, L, Clements, P y Kazman, R. 2003. Software Architecture in Practice. s.l. : Addison Wesley, 2003.

10. Belloch O.C. 2016. Las Tecnologías de la Información y Comunicación (T.I.C.). Valencia: Unidad de Tecnología Educativa. Universidad de Valencia.

11. Camarena Sagredo, J. G., Trueba Espinosa, A., & López García, M. D. L. (2016). Redalyc. Automatización de la codificación del patrón modelo vista controlador (MVC) en proyectos orientados a la Web. Ciencia Ergo Sum, 19(3), 239-250.

12. Cardinali, D. P. (1991). Manual de neurofisiología. Ediciones Díaz de Santos.

13. Center, Nextech Education. 2018. Nextech Education Center. [En línea] 2018. [Citado el: 19 de noviembre de 2019.] https://nextech.pe/que-es-bpmn-y-para-que-sirve/.

14. Centro de Informática Médica, 2018. Especificación de requisitos de software. 5 February 2018.

15. Cerritos, A., Fernández, F. J. y Gatica, F. 2017. Sistema de Información Hospitalaria. 2017.

16. Convención Internacional de Salud. [En línea] 2018. [Citado el: 9 de noviembre de 2019]. http://www.convencionsalud2018.sld.cu/index.php/connvencionsalud/2018/paper/viewFile/115/1022.

17. Ecured. [En línea] Ecured, 2018. [Citado el: 1 de noviembre de 2019.] https://www.ecured.cu/Metodolog%C3%ADas_Tradicionales.

18. Ecured. 2018.. [En línea] 2018. [Citado el: 18 de noviembre de 2019.] https://www.ecured.cu/PostGreSQL.

19. Facelets. (2018). [En línea] 2018. [Citado el: 26 de mayo de 2020]. Disponible en: http://facelets.java.net/

20. Franky, C. 2016. Java EE 5 (sucesor de J2EE). [En línea] 2016. [Citado el: 25 de noviembre de 2019.] http://www.acis.org.co/fileadmin/Conferencias/ConfConsueloFranky_Abr19.pdf

21. Garro, Arkaitz. 2018. [En línea] 2018. [Citado el: 16 de noviembre de 2019.] https://www.arkaitzgarro.com/java/capitulo-1.html.

22. Gamma, E., Helm, R., & Vlissides, J. a. (2013). Design Patterns. Obtenido de http://siul02.si.ehu.es/~alfredo/iso/06Patrones.pdf.

23. Group, Object Management. 2018. Object Management Group. [En línea] Object Management Group, 2018. [Citado el: 21 de noviembre de 2019.] https://www.omg.org/bpmn/.

24. Guanyabens i Calvet, J. (2015). Las TIC y la salud. Madrid: Universitat Oberta de Catalunya.

25. Herrera, Cristhian. Adictos al trabajo. [En línea] 17 de octubre de 2014. [Citado el: 9 de febrero de 2020.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=EJB3vsSpring#1.2.1.Enterprise%20JavaBeans|outline.

26. Java Enterprise Edition. (2018). [En línea] 2018. [Citado el: 16 de enero de 2020]. Disponible en: http://www.oracle.com/technetwork/java/javaee/tech/javaee6technologies-1955512.html.

27. Krall, César. 2006-2029. Qué es y para qué sirve UML, el Lenguaje Unificado de Modelado. Aprende a programar. 2006-2029.

28. Larman, C. (2016). UML y patrones: introducción al análisis y diseño orientado a objetos Madrid: Prentice-Hall, 2016.

29. Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). [En línea]. 2016. [Citado el: 3 de noviembre de 2019]. http://www.cyta.com.ar/ta0502/b_v5n2a1.htm

30. Guerrero Martínez, Juan F. Universitat de Valencia, Open Course Ware. [En línea] 2015. [Citado el: 2 de noviembre de 2019.] http://ocw.uv.es/ingenieria-y-arquitectura/1-5/ib_material/IB_T11_OCW.pdf.

31. Modelos Y Metodologías Para El Desarrollo De Software. [En línea]. 2018. [Citado el: 2 de noviembre de 2019]. http://www.eumed.net/tesis doctorales/2014/jlcv/software.htm.

32. Muñoz, Israel. 2019. MDN web docs. [En línea] 13 de enero de 2019. [Citado el: 21 de enero de 2020.] https://developer.mozilla.org/es/docs/Introducci%C3%B3n_a_XML.

33. Neodoo. 2018. Neodoo. [En línea] 4 de febrero de 2018. [Citado el: 23 de noviembre de 2019.] https://blog.neodoo.es/2018/02/04/que-es-jboss-seam/.

34. Oracle. 2017. Java Enterprise Edition. [En línea] Oracle, 2017. [Citado el: 10 de diciembre de 2019.] https://javaee.github.io/tutorial/jsf-facelets001.html#GIJTU.

35. Paradigm, Visual. Visual Paradigm. [En línea] 2017. [Citado el: 23 de octubre de 2019.] http://www.visual-paradigm.com/aboutus/newsreleases/vpuml80.jsp.

36. Peña, Ayala A. 2016. Ingeniería de Software: Una Guía para Crear Sistemas de Información. México: s.n., 2016.

37. PgAdmin. (2018). [En línea] 2018. [Citado el: 17 de noviembre de 2018]. Disponible en: https://www.pgadmin.org/

38. PostgreSQL. [En línea] 2017. [Citado el: 13 de noviembre de 2018]. Disponible en: https://www.postgresql.org/

39. Ramos-Argüelles, F., et al. Técnicas básicas de electroencefalografía: principios y aplicaciones clínicas. En Anales del sistema sanitario de Navarra. Gobierno de Navarra. Departamento de Salud, 2009. p. 69-82.

40. Roberth G. Figueroa, C. y J. Solís. 2016. Metodologías Tradicionales vs. Metodologías Ágiles. 2016.

41. Sánchez, T. 2014. Metodología de desarrollo para la Actividad productiva de la UCI. Habana: s.n.

42. La neurofisiología clínica en la práctica médica. Shkurovich Bialik, Paul. 2016. 2, Ciudad de México: Anales Médicos, 2016, Vol. 61.

43. La neurofisiología clínica: pasado, presente y futuro. Morales, Gonzalo. 2016. 3, Pamplona: Sistema Sanitario Navar, 2016, Vol. 32.

44. Pressman, Roger S. 2010. Ingeniería de software: Un enfoque práctico. México: Mc Graw Hill, 2010.

45. Maestú, C., et al. Magnetoencefalografía: una nueva técnica de diagnóstico funcional en neurociencia. Rev Neurol, 1999, vol. 28, no 11, p. 1077-1090.

46. Urquijo, JI Aguirregomoscorta, et al. Polisomnografía de siesta: ¿Es suficiente para iniciar tratamiento con CPAP? Archivos de Bronconeumología, 2001, vol. 37, no 8, p. 302-306.

47. Racoveanu, Ni. T.; Staehr , K. (1995). Tecnología para el mejoramiento continuo de la calidad de la atención sanitaria. Foro mundial de la salud 1995; 16 (2): 158-165, 1995.

48. Villa, A., Gutiérrez, E., Pérez, J. C. 2008. Consideraciones para el análisis de la marcha humana. Técnicas de videogrametría, electromiografía y dinamometría. Revista ingeniería biomédica, vol. 2, no 3, p. 16-26.

No presenta anexos