UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 2

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

Título: Historia Clínica Portable para el Sistema de Información Hospitalaria del Centro de Informática Médica.

Autores:

Dayana Rivera Muñoz

Yairiel Viñales Lozano

Tutores:

Ing. Luismel Del Valle Román

Ing. Alejandro Catalá Aguirre

La Habana, Junio de 2014

"Año 56 de la Revolución"

Datos de Contacto

Ing. Luismel Del Valle Román

Graduado de Ingeniero en Ciencias Informáticas en el año 2011, actualmente se desempeña como desarrollador principal del proyecto Sistema de Gestión para la Ingeniería Clínica y Electromedicina (SIGICEM), perteneciente al Centro de Informática Médica de la Universidad de las Ciencias Informáticas. Ha sido tutor de varias tesis de pregrado desarrolladas en dicha universidad, obteniendo buenos resultados en cada una de ellas.

Correo electrónico: ldroman@uci.cu

Ing. Alejandro Catalá Aguirre

Graduado en el año 2008 en la Universidad de las Ciencias Informáticas de Ingeniero en Ciencias Informáticas. Durante su trayectoria laboral se ha desempeñado como Administrador de la Configuración, desarrollador en el Departamento de Sistemas Especializados en Salud (SES). Cuenta con 6 años de experiencia, ha tutorado varias tesis de pregrado.

Correo electrónico: acatala@uci.cu

Declaración de Autoría

Declaramos que somos los únicos autores de esta investigación y autorizamos a la Universidad de las Ciencias Informáticas a usarla, con carácter exclusivo. Para que así conste firmamos la presente a los 19 días del mes de Junio del año 2014.

Dayana Rivera Muñoz

_____

Firma del Autor

Yairiel Viñales Lozano

_____

Firma del Autor

Luismel Del Valle Román

_____

Firma del Tutor

Alejandro Catalá Aguirre

_____

Firma del Tutor

Agradecimientos

De Dayana

A las personas que más quiero, mis abuelos, por ser mis ejemplos a seguir y por guiarme hasta aquí.

A mi mamá por sacrificarse por mí todos estos años brindándome siempre su apoyo incondicional y a mi papá por todo el cariño que me ha dado y por su confianza.

A mis tías Yamila y Dayamí por ser las mejores del mundo y por estar siempre pendientes de mí.

A Yada, Ray, Lisbe y Turo, mis primos queridos.

A Any, Jose y Lazarito, mis niños lindos, por hacerme pasar momentos tan divertidos y por su cariño.

A las tatas, por su apoyo y su cariño incondicional.

A Osvaldo y Marlén, por convertirse en mi familia y apoyarme siempre.

A mi familia porque sin su amor y guía no estuviese aquí.

A Jorge (mi Titi) por cuidarme y aguantarme todos estos años.

A Doris y Alexeis por cuidarme y malcriarme todos estos años.

A mis amistades que me han acompañado en las buenas y en las malas y en especial a mi compañero de tesis por su apoyo siempre que lo necesité.

A mis compañeros de la FEU por mostrarme lo que es estar en una universidad y sentirse universitario.

A mis tutores, por su ayuda y su confianza.

A todas aquellas personas que ayudaron a la realización de este trabajo.

Por su dedicación y apoyo.

¡Muchas Gracias!

Agradecimientos

De Yairiel

En estos cinco años de mi carrera muchas personas de una forma u otra me han ayudado y se, que sin su ayuda y apoyo hoy no estuviera haciéndome ingeniero, por lo que no quiero pasar por alto mi agradecimiento.

Quiero agradecer en primer lugar a mi mama, a mi papa, a mi hermano, a mi abuela Nena, a mi abuelo Berto, a mi tío Manoly, a mi tía Maricela, a mi tía Evelin y a toda mi familia en general, por estar siempre que los necesité y ser lo más grande que tengo en la vida.

Quiero agradecer a mis amigos Francis, Asiel, Yunier, Yerandy, Arel, Ernesto y Hendrik que ya son mis hermanos, por el apoyo y los momentos compartidos que son de las mejores cosas que me llevo de la universidad.

Gracias a mi novia Yanet por aguantarme en mis momentos de estrés y malos genios, por estar conmigo en las buenas y las malas, dándome fuerzas para seguir adelante.

Gracias a mi compañera de tesis que se ha esforzado mucho para obtener este resultado y hemos formado un dúo muy compacto que ha vencido todas las dificultades.

Gracias al piquete Wualfa: Alberto, Oscar, Jasiel, France, Chuchi, Maikel y los agregados.

Gracias a Yunier Leiva, Oscar, Aramis, Yasmani y Dainovis por su ayuda en todo momento. Gracias a mis compañeros de grupo, del cuarto y de la uci en general.

Gracias a mis profes, en especial a Tony Rey y gracias a todos los que de una forma u otra han compartido estos inolvidables años de mi vida.

Dedicatoria

De Dayana

Dedico este Trabajo de Diploma a mis abuelos, porque todo lo que he hecho ha sido para que se sientan orgullosos de mí, porque este también es su sueño, por ser el motor que me impulsa y me da fuerza para seguir adelante.

A toda mi familia, por apoyarme y preocuparse tanto por mí.

De Yairiel

Quiero dedicar esta tesis a mi familia, sin ustedes nada de esto hubiera sido posible, los quiero mucho.

En especial a mi tía Maricela que estando cerca o lejos nunca dejó de ayudarme y de preocuparse por mí.

A mis amigos, soy el segundo graduado jajaja.

Resumen

El Sistema de Información Hospitalaria del Centro de Informática Médica gestiona las historias clínicas de los pacientes a través de documentos clínicos generados en los distintos módulos del mismo. La presente investigación tiene como objetivo: desarrollar una aplicación portable para visualizar estos documentos, permitiendo a los pacientes conservar su historia clínica electrónica personalmente si lo desean.

Para el desarrollo de la solución se utilizó como metodología *Rational Unified Process* y como guía para el desarrollo de *software Capability Maturity Model Integration*. El lenguaje de modelado utilizado fue el *Unified Modeling Language* y la herramienta de modelado *Visual Paradigm for UML Enterprise Edition*. C++ como lenguaje de programación, *Qt Creator* como entorno de desarrollo integrado y el *Framework Qt*. Además se implementó el patrón de arquitectura Modelo-Vista-Controlador y los patrones de diseño *General Responsibility Assignment Software Patterns*.

El sistema desarrollado es una aplicación multiplataforma, con las funcionalidades: exportar en formato PDF, imprimir, buscar y listar documentos clínicos, entre otras. Esta aplicación permite visualizar los documentos clínicos asociados a un paciente determinado, desde cualquier estación de trabajo, sin necesidad de conexión al Sistema de Información Hospitalaria del Centro de Informática Médica.

Palabras claves: documentos clínicos, historia clínica electrónica, portable, visualizar.

Tabla de contenidos

Tabla de contenidos

Introducción

La historia está marcada por grandes avances científico-técnicos, resultados de la necesidad del hombre de mejorar sus condiciones de vida, pero sin dudas el surgimiento de la computadora y por consecuencia el de las ciencias informáticas, ha transformado su manera de pensar y de vivir. El desarrollo de estas ciencias ha hecho posible que ocurran grandes adelantos en las Tecnologías de la Información y las Comunicaciones (TIC), propiciando el intercambio de información mediante comunicaciones más rápidas y eficientes. Estas tecnologías han dado paso a la llamada revolución informática, ya que inciden en el progreso de todos los sectores de la sociedad.

En la actualidad, con el creciente desarrollo que experimenta el sector de la salud, se hace necesario informatizar sus procesos. El uso de los sistemas informáticos en la medicina facilita el intercambio entre el paciente y los especialistas de esta rama, mejora las condiciones de trabajo y permite obtener diagnósticos rápidos y precisos de las distintas enfermedades. Son muchas las aplicaciones que se le han dado a la informática en este sector, por citar algunas: el uso de los robots especializados en intervenciones quirúrgicas sustituye en muchas ocasiones la presencia de un cirujano, se pueden realizar estudios de la anatomía de forma tridimensional, además un médico puede diagnosticar a un paciente desde un continente a otro, en tan solo minutos, utilizando las tecnologías de comunicación disponibles.

Cuba también se encuentra inmersa en todo este proceso de desarrollo, fomentando el uso de las TIC en los distintos sectores de la sociedad. El sector de la salud es una prioridad para los centros dedicados a la investigación y el desarrollo del país, dentro de los que se destaca la Universidad de las Ciencias Informáticas (UCI). Este centro tiene como objetivos: formar profesionales altamente calificados en la rama de la informática, producir aplicaciones y brindar servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación, sirviendo de soporte a la industria cubana del *software*.

La infraestructura productiva de la universidad está formada por centros de investigación y desarrollo, dentro de ellos se encuentra el Centro de Informática Médica (CESIM). En él se desarrolla un Sistema de Información Hospitalaria (HIS, por sus siglas en inglés), que tiene como objetivo principal la gestión de los procesos llevados a cabo en un hospital.

El HIS es un sistema integral para la gestión hospitalaria, que tiene como atributo fundamental una Historia Clínica Electrónica (HCE). Esta incluye toda la documentación que se genera en torno a un

paciente, tales como: datos de enfermería, información sobre pruebas, diagnósticos, sistemas de peticiones, resultados de exámenes, citas, entre otros. Aparece totalmente digitalizada, centralizada y almacenada con seguridad, cumpliendo con los principios de la ética médica. Para lograr la utilización, por múltiples instituciones, de una HCE y la interoperabilidad entre los documentos que la componen, es necesario aplicar estándares internacionalmente reconocidos.

El estándar informático para la salud *Health Level Seven* (HL7) es uno de los más aplicados internacionalmente en el desarrollo de HCE. Este estándar cuenta con especificaciones de mensajes, documentos electrónicos y vocabularios controlados para dominios de salud. Específicamente la Arquitectura de Documentos Clínicos (CDA, por sus siglas en inglés) es muy utilizada para dar formato a los documentos que conforman la historia clínica.

En la actualidad el HIS del CESIM maneja la HCE de los pacientes a partir de un conjunto de documentos clínicos que se generan en los distintos módulos del mismo, almacenados como archivos *Extensible Markup Language* (XML), bajo el estándar HL7/CDA. Para acceder a ellos el sistema provee un visor de historias clínicas electrónicas, que además cuenta con la funcionalidad de exportarlos, pero no se tiene un mecanismo que permita visualizar los documentos clínicos asociados a un paciente determinado una vez fuera del HIS del CESIM.

Esta situación implica que los pacientes no puedan conservar su historia clínica personalmente si lo desean, pues aunque tengan los documentos generados por el sistema, no tienen la posibilidad de verlos con la estructura definida por el estándar antes mencionado. El médico que los atiende no puede acceder a sus HCE si no tiene acceso al HIS, lo que dificulta el análisis y la toma de decisiones clínicas. En caso de trasladarse un paciente para ser atendido en un centro donde no se encuentre desplegado el HIS del CESIM, el personal de la nueva entidad no tendrá acceso a su HCE. Por lo que le será imposible conocer con claridad y absoluta certeza sus anteriores enfermedades, resultados de pruebas, diagnósticos y tratamientos.

Analizando esta situación se identifica como **problema a resolver**: ¿Cómo visualizar la historia clínica electrónica de un paciente, sin necesidad de conexión con el Sistema de Información Hospitalaria del Centro de Informática Médica, garantizando la interoperabilidad y la estandarización a través de HL7/CDA?

Introducción

Teniendo en cuenta el problema a resolver se ha definido como **objeto de estudio**: el proceso de visualización de la historia clínica electrónica en los sistemas de información hospitalaria, delimitando como **campo de acción**: la visualización de los documentos clínicos generados por el Sistema de Información Hospitalaria del Centro de Informática Médica.

Con el propósito de solucionar el problema planteado se propone como **objetivo general:** desarrollar un visor portable para las historias clínicas electrónicas extraídas del Sistema de Información Hospitalaria del Centro de Informática Médica.

Para cumplir con el objetivo planteado se proponen las siguientes **tareas de la investigación:**

- Análisis del estado del arte relacionado con la visualización de HCE en los HIS.
- Selección de la metodología, lenguajes y herramientas a utilizar para el desarrollo de la solución.
- Propuesta de una solución informática para resolver la problemática planteada.
- Desarrollo de los artefactos correspondientes a los flujos de trabajo: Modelado del negocio, Requerimientos, Análisis y diseño e Implementación según la metodología a seguir.
- Implementación de las funcionalidades especificadas para el sistema.

Los **métodos y técnicas de la investigación científica** utilizados para la realización de la presente investigación son:

<u>Teóricos</u>

- Histórico–Lógico: se utilizó en la primera fase de la investigación, en la cual se realizó un estudio del estado del arte, para lograr un mejor entendimiento de lo que se debe desarrollar y para estudiar los sistemas nacionales e internacionales. Se hizo énfasis en el estudio de los conceptos, términos y vocabularios propios del objeto de estudio y el campo de acción.
- Analítico-Sintético: se utilizó para analizar la información de las distintas fuentes consultadas y sintetizar los aspectos más importantes para la investigación. Fue de gran utilidad en todo el proceso investigativo, fundamentalmente para la selección de la metodología, lenguajes y herramientas a utilizar para el desarrollo del sistema.

- Modelación: se utilizó para crear abstracciones de la realidad, es decir, representar el proceso de desarrollo y propiciar un mejor entendimiento de la solución a desarrollar. Se evidencia fundamentalmente en la modelación de los diagramas de clases y de interacción.

Empíricos

- Entrevista no estructurada: se utilizó para definir el alcance de la investigación e identificar las funcionalidades que debe tener el sistema. La entrevista realizada no contó con preguntas específicas previamente elaboradas y se realizó en forma de diálogo.

El presente documento estará estructurado como se muestra a continuación:

CAPÍTULO 1. Fundamentación teórica de la investigación:

En este capítulo se describen los conceptos fundamentales abordados en la investigación y se realiza un estudio del estado del arte sobre la visualización de las HCE en los HIS, tanto a nivel internacional como nacional, haciendo un análisis crítico de las soluciones informáticas existentes. Además se presentan características de la metodología, lenguajes y herramientas utilizadas para dar solución al problema planteado.

CAPÍTULO 2. Características del Visor:

En este capítulo se realiza una propuesta de solución a la problemática planteada, se especifica el modelo a seguir para el desarrollo de la aplicación y se realiza el modelado del dominio. Se definen los requisitos funcionales y no funcionales con los que el sistema debe cumplir. A partir de las funcionalidades se obtienen los casos de uso del sistema y se realiza el diagrama de casos de uso. El capítulo finaliza con la descripción textual de los casos de uso, haciendo posible una mejor comprensión del funcionamiento de la aplicación a desarrollar.

CAPÍTULO 3. Diseño del Visor:

En este capítulo se describe del patrón arquitectónico, así como los patrones de diseño utilizados en la solución propuesta. Se realiza un análisis de posibles implementaciones, componentes o módulos ya existentes que pueden ser reutilizados. Se modelan los diagramas de clases del diseño y se especifican los atributos y métodos de las clases a través de una breve descripción de las mismas. También se modelan los diagramas de secuencia para cada caso de uso del sistema.

CAPITULO 4. Implementación del Visor:

Este capítulo comienza con los resultados arrojados por el diseño del sistema. Se modelan los diagramas de componentes correspondientes a cada uno de los casos de uso y se obtiene el modelo de despliegue. Además se especifica el tratamiento de errores, las restricciones de seguridad y los estándares de codificación con los que debe cumplir el sistema. Se realiza la implementación del sistema, logrando de esta manera alcanzar el objetivo general de la investigación y así darle solución al problema planteado.

Capítulo 1: Fundamentación teórica de la investigación

En el presente capítulo se abordan los conceptos fundamentales relacionados con la investigación en curso y se realiza un análisis sobre la visualización de las HCE en algunos sistemas internacionales y nacionales. Se selecciona la metodología, lenguajes y herramientas que serán utilizadas en el desarrollo de la solución.

1.1 Conceptos fundamentales

Sistema de Información Hospitalaria: En una institución médica se genera diariamente un gran volumen de información de todo tipo, tanto administrativa como asistencial. Con el objetivo de gestionar esta información y planificar la actividad hospitalaria surgen los HIS. Estos sistemas permiten registrar toda la información clínica generada en las instituciones. Además garantizan que circule de manera ordenada a través de la red y que la documentación sea almacenada en servidores de gran capacidad, evitando así el deterioro de las mismas. (1)

Historia Clínica Electrónica: Una HCE es un repositorio de información, mantenido electrónicamente con los datos de salud de toda la vida de un paciente, guardados de tal manera que puedan servir para múltiples usuarios del registro médico. Agrega herramientas de manejo de información que provee recordatorios clínicos y alertas enlazados a bases de conocimientos para el soporte a la toma de decisiones. A su vez, brindan opciones de análisis de datos agregados, tanto para el gerenciamiento del cuidado como para la investigación. Actualmente, los profesionales de la salud, registran información de pacientes sanos y enfermos. (2)

Interoperabilidad: Según la *National Alliance for Health Information Technology* (NAHIT), la interoperabilidad es la habilidad de los sistemas de información computarizados y las aplicaciones de *software*, de comunicarse intercambiando datos en forma precisa, efectiva y consistente en tiempo real y de usar esa información intercambiada. (3)

***Health Level Seven* (HL7):** Es el estándar de intercambio electrónico de información clínica, de diferentes tipos, entre sistemas de informática médica independientes, fundado en 1987. Define el formato de las "transacciones" entre diferentes componentes, de forma que dos sistemas completamente independientes puedan comunicarse entre sí, simplificando la integración de información entre sistemas médicos. Debe su nombre a que fue concebido como estándar para la capa 7 (Nivel de aplicaciones), del modelo de

Interconexión de Sistemas Abiertos (OSI, por sus siglas en inglés), donde la unidad de información es el mensaje. Por ello es relativamente independiente del tipo de conexión física y protocolo de comunicación usado; se ocupa exclusivamente del proceso de dar formato a los datos para convertirlos en mensajes que cualquier aplicación que cumpla la norma, pueda entender. (4)

Arquitectura de Documento Clínico: (CDA, por sus siglas en inglés), es un subdominio del HL7, basado en XML, que describe la estructura y semántica de documentos clínicos con el objetivo de facilitar su intercambio en un entorno de interoperabilidad. Fue diseñado para dar prioridad a mejorar el cuidado de los pacientes. Soporta especialmente el intercambio de documentos legibles entre sistemas, permitiendo presentar la información de forma adecuada a usuarios con diferentes requisitos o conocimientos. Promueve la duración, almacenamiento e interpretación de la información más allá de formatos o tecnologías vigentes. (5)

Extensible Markup Language (XML)**:** es un conjunto de reglas para definir etiquetas semánticas que organizan un documento en diferentes partes. XML es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados. (6)

1.2 Análisis de soluciones existentes

Para lograr la fundamentación teórica de la presente investigación, teniendo en cuenta el objeto de estudio de la misma, se realizará el análisis del desarrollo de las HCE tanto a nivel internacional como nacional. A través de la descripción de algunos sistemas informáticos desarrollados para gestionar los procesos hospitalarios, que cuentan con la funcionalidad de visualizar sus HCE, se sentarán las bases para la realización de una propuesta de solución.

1.2.1 Internacionales

En la actualidad, el uso de la HCE es una ley en muchos países, independientemente de las características del sistema nacional de salud que tengan. Se puede citar que en Colombia está vigente la Ley 1438 de 2011, que establece que la Historia Clínica Única Electrónica es de obligatoria aplicación desde el 31 de diciembre del año 2013. También en Perú se promulgó la Ley No 30024 que crea el "Registro Nacional de Historias Clínicas Electrónicas". Otros países como: Estados Unidos, España y México utilizan en gran parte de sus instalaciones de salud la HCE. Esto ha provocado una avalancha de

sistemas informáticos para la creación de historias clínicas digitalizadas. Algunos de estos sistemas se analizan a continuación:

OpenMRS, programado en JAVA sobre PostgreSQL o MySQL, con una licencia muy particular *OpenMRS Public License*. Es modular y escalable, además de trabajar con estándares abiertos e interoperables (HL7, Dicom). La aplicación es configurable basada en formularios y tiene además una serie de ventajas como clientes OpenMRS para dispositivos móviles inteligentes. (7)

OpenEMR, con más de 10 años de existencia, está siendo usado en más de 170 países, funciona en más de 5000 instalaciones y ha sido traducido a casi 20 idiomas, es para muchos el sistema de registros médicos electrónicos más usado del mundo. A las ventajas de ser *software* libre se suma una enorme cantidad de módulos, servicios y componentes, totalmente configurables, interoperables y escalables, que permiten adaptar el sistema a las necesidades de cada clínica, hospital o centro sanitario. OpenEMR funciona sobre PHP y MySQL. (7)

FreeMedForms es un administrador de Registros Médicos Electrónicos (EMR, por sus siglas en inglés). Es multiusuario y se desarrolla desde octubre de 2008 como *software* de código abierto. El objetivo es crear un administrador de EMR donde los documentos de los pacientes estén en formato XML. Con él se puede programar su agenda personal y crear citas para uno o más pacientes, utilizando la tecnología de MySQL. FreeMedForms tiene un módulo de prescripción, varias bases de datos de tratamiento están disponibles hasta el momento: Francia, Canadá, Estados Unidos y Bélgica. Este módulo puede detectar interacciones farmacológicas, así como las interacciones medicamentosas en el paciente. Sus datos están abiertos y verificables, distribuido bajo Licencia Pública General (GPL, por sus siglas en inglés) en su versión 3. (8)

A-medic es un programa informático diseñado para trabajar con ordenadores personales equipados con *Microsoft Windows XP*/Vista/*Windows 7*. A-medic le permite, mediante un modo de operación rápido y sencillo, identificar y clasificar a sus pacientes. Dispone de una estructura en forma de pestañas en donde están organizados los datos de cada uno de ellos. La información contenida en la ficha del paciente consta de datos acerca de la filiación, diagnóstico, antecedentes, curso clínico, etc. (9)

MedFile 5.x permite crear y mantener las HCE de sus pacientes en un formato especial de Base de Datos, asignar turnos (citas) para la consulta con agenda personalizada para cada médico, y emitir prescripciones y órdenes médicas en forma altamente personalizable y configurable. Compatible con

Windows XP, Vista y *Windows 7 – 8*. (10) Además cuenta con la versión portable MedFile Portable 5.x diseñada para ser utilizada en un dispositivo de almacenamiento removible, se instala fácilmente en el dispositivo de memoria y se inicia automáticamente cuando se lo conecta al equipo. Cuenta con todas las funciones de MedFile y es compatible con *Windows XP y Windows* Vista.

Historias Clínicas Online: Es un sistema accesible a través de la *web*, puede conectarse desde cualquier ordenador con conexión a internet, sin tener en cuenta los diferentes sistemas operativos. Ideal para profesionales que dan consulta en diferentes centros y quieren tener el historial clínico de sus pacientes siempre disponible. Puede crear cuantos usuarios quiera para su centro con una única licencia.

mispacientes.es es un servicio *online* para centros médicos, consultorios y clínicas que requieren de un sistema de gestión de HCE. Incluye módulos de prescripción electrónica, evolución del paciente, problemas de salud, exploraciones complementarias, informes, datos biométricos, etc. Permite la gestión integral de la clínica con planificación (agenda) y la gestión económica que contempla, facturación privada, mutuas, honorarios médicos, facturación electrónica, cobros, etc. (11)

Angel 3.0: Es un *software* dirigido a los tres tipos de atención en salud de fácil instalación y actualización. Ha sido concebido con un módulo de codificación por barras, para asignar a cada paciente una tarjeta magnética, con su HCE codificada, la que se imprime en una impresora especial. Además cuenta con un módulo de firma digital que garantiza que solamente el paciente use la tarjeta con sus datos y que solo puedan ser modificados por el personal de salud registrado en el sistema. Para lograr la estandarización necesaria se aplicaron normas internacionales, usando estándares para la arquitectura, la codificación, el formato de mensajes, la seguridad de los datos y la confidencialidad, entre otros. Se utilizó MySQL como gestor o motor de bases de datos.

Después de analizados estos sistemas, se concluye que OpenMRS, OpenEMR y FreeMedForms son una buena alternativa por su escalabilidad e interoperabilidad, además de la gran ventaja que representa el *software* libre en la actualidad. Sin embargo, tienen como limitante que solo son accesibles a través de la *web* y no tienen en cuenta la necesidad de obtener los datos de un paciente desde una computadora sin acceso a internet. Los sistemas A-medic, MedFile 5.x y Angel 3.0 brindan la posibilidad de administrar las HCE de los pacientes y son muy utilizados, pero están limitados a los sistemas operativos propietarios de Microsoft. Al igual que los sistemas Historias Clínicas Online y mispacientes.es, las personas que deseen usar estas aplicaciones, deben pagar licencias costosas y comprar el producto con sus actualizaciones.

En el caso de MedFile 5.x elimina la limitante de la necesidad de conexión a internet con su versión portable, pero esta versión solo es compatible con *Windows XP y Windows* Vista.

1.2.2 Nacionales

Generalmente en los centros de salud cubanos la Historia Clínica (HC) se maneja en formato duro, con los inconvenientes que genera su llenado, almacenamiento y conservación. En muchos de estos centros el paciente es quien se encarga de conservar y almacenar su HC, aumentando el riesgo de deterioro o pérdida de la misma. La mala caligrafía también se presenta como una dificultad para una buena interpretación. Otra de sus limitaciones es que no contribuye de forma activa a la toma de decisiones por lo que se dificulta el análisis con fines científicos.

El **HIS del CESIM** es un sistema desarrollado en la Universidad de las Ciencias Informáticas. Este cuenta con varios módulos que gestionan las diferentes áreas de un hospital. En el módulo de Admisión se realiza la apertura de la HCE de los pacientes, que se encuentra almacenada en un servidor accesible a través de la *web*. El sistema gestiona los documentos CDA versión 2 que se generan en la consulta visitada por el paciente y utiliza mensajes HL7 versión 2.3.1. Brinda la posibilidad de realizar firmas digitales a través de un componente desarrollado en el módulo Visor de HCE. En el año 2011 se desarrolló un Visor Portable para este sistema que visualizaba los CDA generados por el mismo.

Los documentos que visualizaba el Visor Portable no eran válidos pues no cumplían totalmente con el formato del estándar definido. Después de realizar las modificaciones necesarias en los documentos CDA para validarlos, el sistema portable no los visualiza. Este visor necesitaba acceder a la base de datos del HIS para la búsqueda de los pacientes y los documentos asociados a ellos, eliminando parcialmente la portabilidad del mismo, además de violar la seguridad de la información de dichos pacientes.

Con el análisis anterior se evidencia que el HIS del CESIM solo es accesible a través de la *web* y que el visor desarrollado en el año 2011 no visualiza los documentos clínicos bajo el formato CDA. Aunque estos sistemas no resuelven la problemática planteada, tanto el módulo Visor de HCE del HIS del CESIM, como el Visor Portable, pueden servir de guía para la propuesta de solución a desarrollar.

1.3. Metodología de desarrollo

Una metodología de desarrollo de *software* se puede definir como un conjunto de procedimientos, tecnologías y herramientas que permiten generar un grupo de artefactos que documentan y guían el

desarrollo de un sistema. Contienen procesos, actividades y tareas involucradas en el desarrollo, despliegue y mantenimiento de un *software*.

Como metodología de desarrollo se utilizará ***Rational Unified Process*** **(RUP)**, pues esta es la metodología que se utiliza en el Departamento de Gestión Hospitalaria del CESIM donde se desarrolla el HIS, para guiar y documentar el ciclo de vida de sus proyectos. RUP fue la metodología utilizada en el desarrollo del módulo Visor de HCE de este sistema y la documentación que fue generada servirá de guía para el desarrollo de la solución propuesta.

Fases de RUP:

1. Concepción o Inicio: tiene como finalidad definir la visión, los objetivos y el alcance del proyecto, tanto desde el punto de vista funcional como técnico; obteniéndose como uno de los principales resultados una lista de los casos de uso y una lista de los factores de riesgo del proyecto.
2. Elaboración: tiene como principal finalidad completar el análisis de los casos de uso y definir la arquitectura del sistema, además se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.
3. Construcción: está compuesta por un ciclo de varias iteraciones, en las cuales se van incorporando sucesivamente los casos de uso, de acuerdo a los factores de riesgo del proyecto.
4. Transición: se inicia con una versión “beta” del sistema y culmina con el sistema en fase de producción.

Algunas de las mejores prácticas de RUP son:

1. Desarrollo Iterativo: Propone, a partir de una planificación inicial, entrar a un ciclo en las etapas de desarrollo, donde en cada iteración se obtenga un ejecutable del sistema. Este desarrollo permite la mitigación de riesgos, la integración progresiva y la corrección de errores en iteraciones tempranas.

2. Administración de Requerimientos: Aproximación sistemática para la búsqueda, documentación, organización y seguimiento de los cambios en los requerimientos de un sistema.

3. Verificación de la calidad: La verificación y administración de la calidad durante el ciclo de vida del proyecto, es esencial para mantener los objetivos y el tiempo estimado de desarrollo del mismo.

4. Control de cambios: Se administran los flujos, el desarrollo paralelo, la integración y la construcción del *software*. Permite una mejor identificación de los recursos básicos en los riesgos y objetivos del proyecto.

1.3.1 Guía para el desarrollo de *software*

Capability Maturity Model Integration (CMMI, por sus siglas en inglés), es un modelo para la mejora y evaluación de procesos de desarrollo, mantenimiento y operación de sistemas de *software*. Las mejores prácticas se publican en documentos llamados modelos, los cuales contienen el conjunto de prácticas relacionadas que son ejecutadas de forma conjunta para conseguir determinados objetivos. Así es como el modelo CMMI establece una medida del progreso, conforme al avance en niveles de madurez. Cada nivel a su vez cuenta con un número de áreas de procesos que deben lograrse. El hecho de alcanzar estas áreas se detecta mediante la satisfacción o insatisfacción de varias metas claras y cuantificables. (12)

CMMI cuenta con 5 niveles de madurez:

- Nivel 1: Inicial.
- Nivel 2: Administrado.
- Nivel 3: Definido.
- Nivel 4: Cuantitativamente administrado.
- Nivel 5: Optimizado.

Su implementación aumenta la fiabilidad del *software* producido, la visibilidad de los procesos de producción y soporte, la reusabilidad de componentes, y como resultado de la combinación de este tipo de mejoras, disminuye los costes de producción y mantenimiento de las aplicaciones. La UCI fue evaluada por este proceso de mejoras alcanzando el nivel 2 y específicamente el HIS del Departamento de Gestión Hospitalaria del CESIM se desarrolla en este nivel, por lo que será también una guía para la solución informática que se desarrollará para dar respuesta al problema planteado.

El nivel 2 consta de 7 áreas de procesos:

- REQM -Gestión de Requisitos.
- PP -Planificación de Proyectos.

- PMC -Seguimiento y Control de Proyectos.
- SAM -Acuerdos con Proveedores.
- MA -Medición y Análisis.
- PPQA -Aseguramiento de la Calidad de Procesos y Productos.
- CM -Gestión de la Configuración.

1.4. Lenguaje de modelado

Los lenguajes de modelado son muy usados internacionalmente en el desarrollo de proyectos informáticos, fundamentalmente en la ingeniería de requisitos y en el modelado de los procesos durante el desarrollo del *software*. Como lenguaje de modelado se utilizará el Lenguaje de Modelado Unificado, (UML por sus siglas en inglés) en su versión 2.1, ya que ha sido definido por el Departamento de Gestión Hospitalaria para el desarrollo de sus proyectos.

Lenguaje de Modelado Unificado (UML, por sus siglas en inglés): Es un lenguaje para la especificación, visualización, construcción y documentación de los artefactos de un proceso de sistema intensivo. Fue originalmente concebido por la *Corporation Rational Software* y tres de los más prominentes metodologistas en la industria de la tecnología y sistemas de información: Grady Booch, James Rumbaugh e Ivar Jacobson. (13)

UML se ha convertido en la notación estándar para definir la arquitectura de un sistema, materializándolos en modelos. Un modelo es una simplificación de la realidad. Se construyen modelos para comprender mejor el sistema que se está desarrollando. Un sistema es algo "compuesto", una construcción realizada por manos y herramientas siguiendo un propósito. (13)

Las principales funciones de UML son:

- Visualizar: UML permite expresar gráficamente un sistema de forma que otro lo puede entender.
- Especificar: UML permite especificar cuáles son las características de un sistema antes de su construcción.
- Construir: A partir de los modelos especificados se pueden construir los sistemas diseñados.

- Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado, sirviendo para su futura revisión.(14)

1.5 Herramienta CASE

Las herramientas de Ingeniería Asistida por Computadora (*Computer Aided Software Engineering*, CASE por sus siglas en inglés) son un conjunto de herramientas y métodos asociados que proporcionan asistencia automatizada en el proceso de desarrollo del *software* durante su ciclo de vida. (15). En el Departamento de Gestión Hospitalaria se utiliza la herramienta CASE: *Visual Paradigm* para el modelado de sus proyectos. Para el análisis y diseño de la solución se utilizará *Visual Paradigm for UML Enterprise Edition* (VP-UML EE) en su versión 8.0.

Visual Paradigm for UML es una herramienta para el desarrollo de aplicaciones que utiliza como lenguaje de modelado UML. Es ideal para ingenieros de *software*, analistas y arquitectos de sistemas, interesados en la construcción a gran escala. Proporciona confiabilidad y estabilidad en el desarrollo orientado a objetos. (16)

Visual Paradigm for UML Enterprise Edition: Es la edición *top* de la línea de productos, agrega valor en términos de modelado de datos orientado a objetos. Además hace posible la documentación del proyecto, permite el mapeo relacional de objetos para Java, .NET y PHP, reduciendo costos y aumentando su productividad. (16)

Visual Paradigm ofrece:

- Navegación intuitiva entre la escritura del código y su visualización.
- Potente generador de informes en formato PDF/HTML.
- Documentación automática.
- Ambiente visualmente superior de modelado.
- Sofisticado diagramador.
- Sincronización de código fuente en tiempo real.

1.6 Lenguaje de programación

Un lenguaje de programación es un lenguaje artificial que puede ser usado para controlar el comportamiento de una máquina, especialmente una computadora. Está compuesto por un conjunto de reglas sintácticas y semánticas que permiten expresar instrucciones que luego serán interpretadas. (20) A continuación se describen algunos lenguajes de programación que pueden ser utilizados con el propósito de seleccionar uno de ellos según las características que debe tener el sistema.

Java es un lenguaje de programación desarrollado por *Sun Microsystems*. La tecnología *java* contiene dos componentes básicos: el lenguaje *java* y su plataforma, la máquina virtual de *java*. Tiene como característica que ofrece la potencia del diseño orientado a objetos, con una sintaxis fácilmente accesible y un entorno robusto y agradable. Además facilita la utilización de aplicaciones que se pueden incluir directamente en páginas *web*. Es un lenguaje gratuito y muy utilizado a nivel internacional. Para ejecutar cualquier sistema desarrollado con este lenguaje, es necesario tener la máquina virtual de *java* instalada.

Java es un lenguaje interpretado, por lo que se interpretan los ficheros de clases antes de que se ejecuten. Utilizando un lenguaje de programación tradicional como C o C++, el ordenador puede ejecutar directamente el código generado. Debido a la interpretación que se tiene que hacer de los ficheros, los programas escritos en *java* demoran más en ejecutarse que los escritos en C y C++.

C es uno de los lenguajes de programación más usados, funciona a nivel de máquina, por lo que permite entender la interacción entre los programas y la capa de hardware. Permite depurar código y gestionar la memoria. También ofrece una visión panorámica de cómo funciona un ordenador que no ofrecen otros lenguajes. Tiene como desventaja que es rígido y difícil de aprender, por lo que no es recomendable para trabajar en programas que no interactúan con la capa de *hardware*, utilizando unidades de disco o extensiones del sistema operativo.

C no tiene soporte para la programación orientada a objetos ni soporte nativo para la programación multihilo. Tiene como inconveniente que el compilador de C se limita a traducir código sin apenas añadir nada. La gestión de la memoria es un ejemplo clásico: en C el programador ha de reservar y liberar la memoria explícitamente.

C++ surge en 1980 como una extensión del lenguaje C. Es un lenguaje versátil, potente y general que tiene gran aceptación entre los programadores profesionales. Mantiene las ventajas del C en cuanto a riqueza de operadores y expresiones, flexibilidad y concisión. Ha eliminado algunas de las dificultades y limitaciones del C original. C++ es a la vez un lenguaje procedural (orientado a algoritmos) y orientado a objetos. Mantiene una enorme potencia para la programación a bajo nivel, pero se le han añadido elementos que le permiten además un estilo de programación con alto nivel de abstracción.

El lenguaje C++ representa un gran salto cualitativo frente a C al proporcionar nuevas características útiles en diversos contextos. Un ejemplo es la sobrecarga de operadores, que dota al lenguaje de una expresividad notable. La sintaxis de clases y objetos permite manipular convenientemente diversas estructuras de datos y operaciones y el tratamiento de excepciones permite procesar los distintos errores que se puedan producir.

Teniendo en cuenta las características de los lenguajes antes mencionados, se ha decidido utilizar C++, ya que este es un lenguaje rápido y robusto, permite la reutilización de código y usa una sintaxis y chequeo de tipos muy estricto, lo que elimina gran parte de los errores en el código. Permite la eficiencia, uniformidad, comprensión y flexibilidad del código a la hora de trabajar. Existen muchos tutoriales en línea, libros y códigos fuentes abiertos para aprender todo lo necesario sobre este lenguaje. Para su selección se tuvo en cuenta, además de las características antes mencionadas, la necesidad de portabilidad, rapidez y consumo mínimo de recursos que tiene el sistema a desarrollar.

1.7 Entorno de desarrollo

Los Entornos Integrados de Desarrollo, (IDEs por sus siglas en inglés), se pueden definir como una herramienta para editar el código de los lenguajes de programación, compilarlo, depurarlo y construir una interfaz gráfica. Pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. Los entornos que se describen a continuación son muy utilizados en la actualidad y se seleccionará uno de ellos según las características del sistema a desarrollar y del lenguaje seleccionado.

Eclipse es un IDE multiplataforma que permite montar herramientas de desarrollo para cualquier lenguaje, mediante la implementación de los complementos (*plugins*) adecuados. El proyecto *Eclipse* integra tanto el desarrollo del IDE, como de algunos de los complementos más importantes (como el JDT, *plugin* para el

lenguaje *Java*, o el CDT para el lenguaje C/C++). Este IDE está escrito en *java,* por lo que para su utilización se necesita instalar la Máquina Virtual de *Java*. Tiene como desventaja la lentitud tanto para compilar como para depurar.

NetBeans es un IDE de distribución gratuita que proporciona herramientas para el desarrollo de aplicaciones sobre la plataforma *Java.* Es muy utilizado, pues permite desarrollar aplicaciones para varias plataformas y cuenta con gran variedad de *plugins* (componentes de *software* conectables a *NetBeans*) disponibles para incrementar sus funcionalidades. Algunos de estos *plugins* son propietarios y necesitan la compra de una licencia de uso. Este IDE está escrito en *java,* por lo que para su utilización se necesita instalar la Máquina Virtual de *Java*.

Qt Creator automatiza algunas tareas, como la creación de proyectos, proporcionando asistentes que guían paso a paso en el proceso de creación del proyecto, crea los archivos necesarios, y especifica los ajustes en función de las decisiones que toma. Este IDE acelera la escritura de código, ofreciendo resaltado semántico, comprobando la sintaxis del código, proporciona completado de código, refactorización de acciones, y otras características útiles. (17) *Qt Creator* es ideal para los programadores que buscan la simplicidad, facilidad de uso, productividad, extensibilidad y apertura.

Las siguientes herramientas de *Qt* se integran en *Qt Creator*:

- ***Qt Designer*** para el diseño y la creación de interfaces gráficas de usuario (GUI, por sus siglas en inglés). Puede componer y personalizar los diálogos en un editor visual, y los prueba con diferentes estilos y resoluciones. Se puede acceder a *Qt Designer* desde *Qt Creator* en el modo de diseño.
- ***QMake*** para la creación de aplicaciones para diferentes plataformas de destino. Cuando se usa *QMake*, se especifica el sistema de construcción en la modalidad de proyectos.
- ***Qt Linguist*** para la localización de aplicaciones. *Qt Linguist* contiene herramientas para las funciones que suelen participar en aplicaciones de localización.
- ***Qt Asistente*** para la visualización de documentación *Qt.* También se puede ver la documentación en *Qt Creator*. La documentación se muestra automáticamente en el modo de ayuda y se pueden añadir documentos a la lista.

Después de analizadas las características de los entornos de desarrollo *Eclipse, NetBeans y Qt Creator*, se ha decidido utilizar *Qt Creator* en su versión 4.8, ya que es una potente herramienta que se distribuye de forma gratuita y permite desarrollar sistemas multiplataforma. La ventaja principal de *Qt Creator* es el entorno gráfico para crear interfaces de la librería *Qt;* en muchos otros no se tiene la posibilidad de crear el entorno gráficamente, solo mediante código. *Qt Creator* es un IDE para el desarrollo de aplicaciones con las bibliotecas *Qt*, proporcionando herramientas para el cumplimiento de las tareas a lo largo de todo el ciclo de vida del desarrollo de aplicaciones. Tiene como lenguaje nativo a C++ pero soporta otros lenguajes como Pascal, Python, Ruby y PHP.

1.8 Framework

Un *framework* es una colección organizada de clases que constituyen un diseño reutilizable para un dominio específico de *software*. Contiene un conjunto de librerías, componentes de *software* y directrices arquitectónicas que ofrece al desarrollador un *kit* de herramientas completo para construir una aplicación de principio a fin, siempre teniendo en cuenta que es necesario adaptarlo a cada aplicación en particular. (18) El IDE *Qt Creator* está fundamentalmente diseñado para el *framework Qt,* por lo que se utilizará para el desarrollo de la solución propuesta en su versión 5.0.2.

Qt es una biblioteca multiplataforma ampliamente usada para desarrollar aplicaciones con interfaz gráfica de usuario, así como también para el desarrollo de programas sin interfaz gráfica, como herramientas para la línea de comandos y consolas para servidores. Es un *software* libre y de código abierto, desarrollado a través de *Qt Project*. (19)

En este capítulo se abordaron algunos conceptos necesarios para la compresión de la presente investigación. Se realizó un análisis de la situación a nivel nacional e internacional de los HIS que cuentan con HCE. Este evidencia que los sistemas estudiados no son soluciones factibles para el problema planteado, pues son privativos, accesibles solo a través de la *web* o solo pueden ejecutarse en plataformas *Windows*. Se decide realizar una aplicación de escritorio que permita visualizar los documentos exportados en formato CDA por el HIS del CESIM. La selección de la metodología RUP, UML como lenguaje de modelado, VP-UML EE como herramienta CASE, C++ como lenguaje de programación y *Qt Creator* como IDE haciendo uso del *framework Qt,* se realizó teniendo en cuenta las utilizadas en el Departamento de Gestión Hospitalaria del CESIM y las características que el sistema a desarrollar debe tener.

Capítulo 2: Características del visor

En este capítulo se propone una solución para resolver la problemática planteada. Se modela el dominio, con el objetivo de describir los principales conceptos dentro del contexto del sistema y sus relaciones. Se plantean los requisitos funcionales y no funcionales de la aplicación a desarrollar y se identifican los actores del sistema. Por último se realiza el diagrama de casos de uso y la descripción textual de los mismos.

2.1 Propuesta de solución

El Visor de HCE del HIS del CESIM permite visualizar los documentos clínicos que conforman la HCE de un paciente generados por el sistema, pero no se puede visualizar esta HCE fuera de él, lo que implica que los pacientes no puedan conservarla en caso de que deban trasladarse a otro centro o deseen consultar la opinión de un médico que no tenga acceso al HIS.

Dados los inconvenientes de no poder visualizar la HCE de un paciente desde cualquier estación de trabajo, sin tener en cuenta la conexión al HIS, se propone como solución, una aplicación de escritorio que pueda ser ejecutada en varios sistemas operativos *Linux, Mac OS* y *Windows*. Su función principal será visualizar los documentos clínicos de un paciente determinado, extraídos previamente del HIS como archivos XML. Estos archivos se podrán exportar con la extensión .pdf o imprimir directamente si el usuario lo desea. El sistema debe permitir también buscar y listar los documentos clínicos del paciente, además de mostrar sus datos generales.

2.2 Modelo de dominio

Cuando los procesos que se quieren modelar tienen un bajo nivel de estructuración o no se encuentran bien definidos, no se puede realizar un modelo de negocio; en este caso la metodología RUP define que se debe realizar un modelo de dominio. El modelo de dominio es una representación, en forma de diagrama de clases, de los principales conceptos en el área de la aplicación del sistema, representando también las relaciones entre ellos.

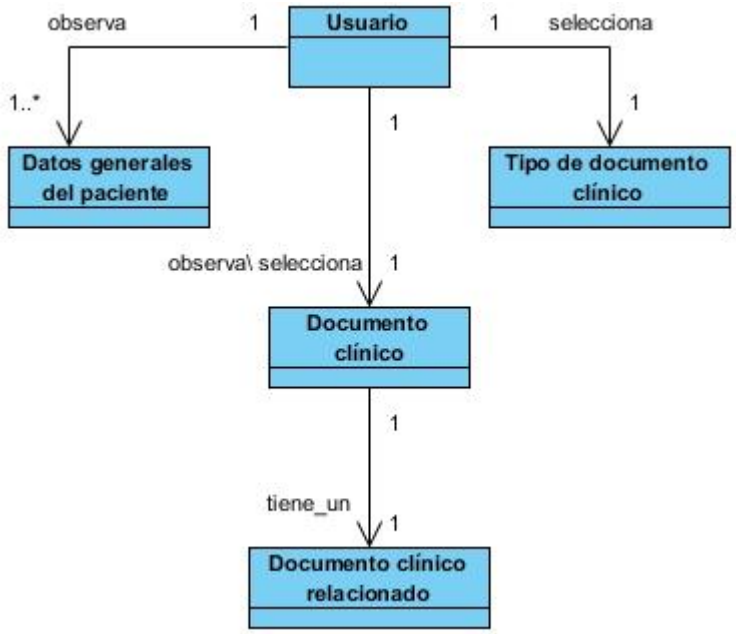


Figura 2.1 Diagrama del modelo de dominio

Conceptos relacionados con el dominio:

Usuario: Representa a la persona que va a interactuar con el sistema, ya sea un médico o un paciente.

Datos generales del paciente: Representa los datos del paciente que se muestran en el sistema al ejecutarse la aplicación.

Documento clínico: Representa los documentos de la historia clínica de un paciente que se visualizan en el sistema.

Tipo de documento clínico: Representa la especialidad médica a la que pertenece un documento clínico.

Documento clínico relacionado: Representa la versión anterior del documento clínico que se visualiza.

2.3 Especificación de los requisitos del software

Los requerimientos de un *software* son las condiciones que debe cumplir el mismo para satisfacer las necesidades de un cliente. Los requisitos deben ser identificados con claridad e igualmente documentados, para ello se realiza un documento de especificación de requisitos, donde quedan plasmadas las funcionalidades y características que el sistema debe tener. Para definir los requerimientos

funcionales y no funcionales del sistema a desarrollar, se realizó una entrevista al jefe del proyecto Sistema de Información Hospitalaria. Sus aspectos fundamentales se muestran en el Anexo 1.

2.3.1 Requisitos funcionales

Son capacidades o condiciones que el sistema debe cumplir. Describen las funcionalidades con las que debe contar el sistema para satisfacer las necesidades del usuario y definen el comportamiento del sistema. Los requisitos funcionales definidos para el sistema son:

RF 1: Mostrar datos generales del paciente.

El sistema debe mostrar los datos generales de un paciente determinado en la interfaz principal.

RF 2: Buscar documentos clínicos.

El sistema debe permitir al usuario realizar una búsqueda de los documentos clínicos que desea visualizar.

RF 2.1: Listar tipos de documentos clínicos.

El sistema debe listar todos los tipos de documentos clínicos que existen.

RF 2.2: Mostrar rango de fechas.

El sistema debe permitir al usuario seleccionar un rango de fechas entre las que se encuentra el documento clínico que desea visualizar.

RF 3: Listar documentos clínicos.

El sistema debe mostrar un listado de los documentos clínicos que tiene un paciente determinado.

RF 4: Mostrar documento clínico.

El sistema debe permitir al usuario visualizar un documento clínico.

RF 5: Mostrar documento clínico relacionado.

El sistema debe permitir al usuario visualizar el documento clínico relacionado al que ha sido seleccionado, en caso de que lo posea.

RF 6: Mostrar menú de secciones del documento clínico.

El sistema debe mostrar un menú desde donde se pueda acceder a las distintas secciones de un documento clínico.

RF 7: Exportar a PDF.

El sistema debe permitir al usuario exportar un documento clínico en formato PDF.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales son características o cualidades que el *software* debe tener para resultar atractivo, usable, rápido y confiable a los usuarios que interactúan con él. Los requisitos no funcionales definidos para el sistema son:

RNF 1: Usabilidad: El sistema está diseñado para que los usuarios aprendan a trabajar con él rápidamente, pues se ha tenido en cuenta el poco conocimiento por parte de los pacientes de los términos médicos e informáticos.

RNF 2: Confiabilidad: El médico del centro donde se atiende el paciente es el encargado de entregarle sus documentos clínicos y el visor portable almacenados en un CD o DVD no reescribible. El sistema solo muestra los documentos clínicos asociados a un paciente lo que garantiza la seguridad de los datos de otros pacientes. Al ocurrir un error se muestra un mensaje indicando dónde se produjo el mismo.

RNF3: Rendimiento: El sistema respetará buenas prácticas de la programación para incrementar el rendimiento en operaciones costosas.

RNF4: Interfaz: El diseño de la interfaz es sencillo, no es necesario tener grandes conocimientos de informática y medicina para interactuar con la aplicación.

RNF5: Soporte: Se generará la documentación suficiente para lograr un mayor entendimiento del sistema y agilizar el proceso de mantenimiento.

RNF6: Portabilidad: El sistema en una aplicación portable, pues se puede ejecutar tanto en sistemas operativos *Windows* como en los que pertenecen al *software* libre.

2.4 Definición de Casos de Uso del Sistema

Los Casos de Uso de Sistema (CUS) posibilitan la comprensión del sistema por parte de los desarrolladores, los usuarios finales del sistema y los expertos del dominio. Describen detalladamente los

procesos, mediante acción – reacción, desde el punto de vista del usuario. A continuación se mencionan los CUS identificados:

CUS 1: Mostrar datos generales del paciente.

CUS 2: Buscar documentos clínicos.

CUS 3: Listar documentos clínicos.

CUS 4: Mostrar documento clínico.

CUS 5: Mostrar documento clínico relacionado.

CUS 6: Mostrar menú de secciones del documento clínico.

CUS 7: Exportar a PDF.

2.4.1 Definición de los actores del sistema

Los actores del sistema son personas (o dispositivos) externos al sistema que interactúan directamente con él. Se identifica como actor del sistema:

Actor	Descripción
Usuario	Es una generalización de las personas que pueden interactuar con el sistema y tiene acceso a todas las funcionalidades del mismo.

Tabla 2.1 Actores del sistema

2.4.2 Diagrama de Casos de Uso del Sistema

Un diagrama de casos de uso se utiliza para representar las relaciones que existen entre los distintos casos de uso y los actores que interactúan con el sistema. A continuación se muestra el diagrama de CUS:

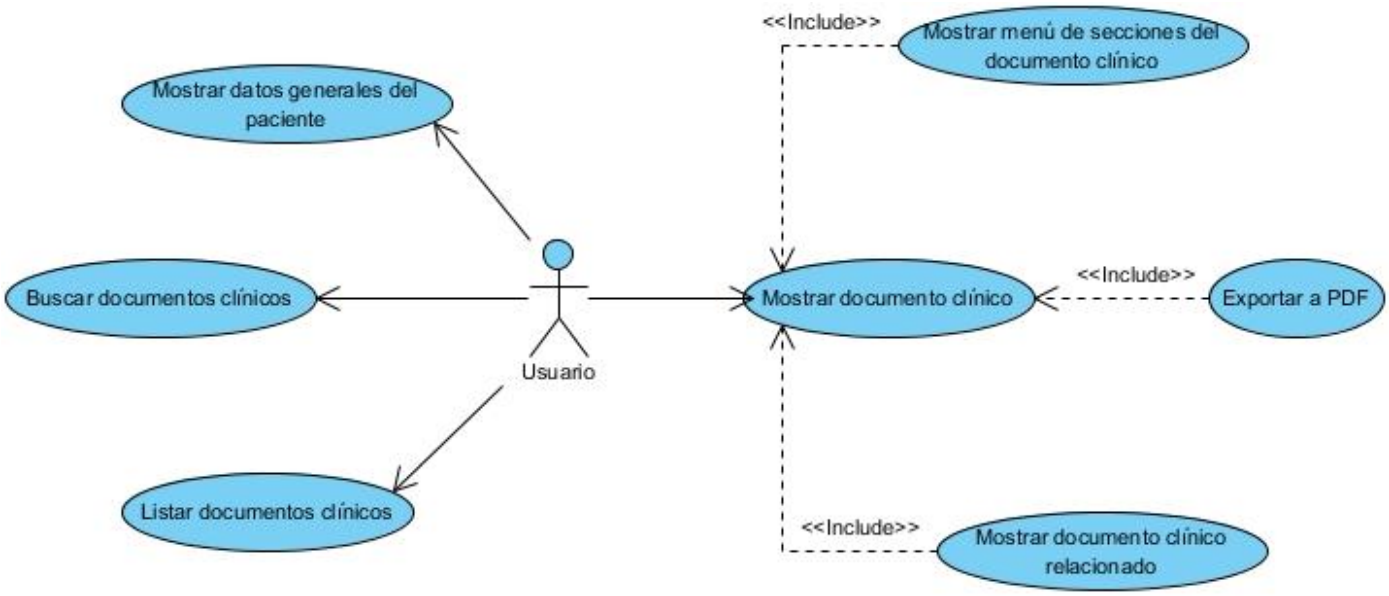


Figura 2.2 Diagrama de Casos de Uso del Sistema

2.4.3 Descripción de los Casos de Uso del Sistema

A continuación se describe el CUS: Buscar documentos clínicos. Los restantes se encuentran en el Anexo 2 de este documento.

CUS: Buscar documentos clínicos.

Objetivo	Buscar los documentos clínicos de un paciente.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el actor luego de haber seleccionado el rango de fechas y el tipo de documento que desea buscar, selecciona la opción “Buscar” y el sistema muestra un listado de documentos que cumplen con los criterios seleccionados en caso de que existan, de lo contrario muestra un mensaje informando que no existen documentos que cumplan con el criterio de búsqueda. El caso de uso termina.

Precondiciones	El sistema debe tener acceso a los documentos clínicos del paciente.
Referencia	RF 2
Prioridad	Alta.
Flujo de eventos	
Flujo Básico	
Actor	**Sistema**
1. Selecciona la opción Buscar después de seleccionar los criterios de búsqueda.	
	2. Muestra un listado con los documentos clínicos del paciente que cumplan con los criterios de búsqueda. Termina el caso de uso.
Flujos Alternos	
1ª El usuario no selecciona criterios de búsqueda.	
Actor	**Sistema**
1ª1. Selecciona la opción Buscar.	
	1ª2. Muestra un listado con todos los documentos clínicos del paciente.
2ª. No existen documentos clínicos que cumplan con los criterios de búsqueda.	
Actor	**Sistema**

	2ª1. Muestra un mensaje indicando que no existen documentos clínicos que cumplan con los criterios de búsqueda especificados. 2ª2. Muestra la opción: ✓ Aceptar.
2ª3. Selecciona la opción Aceptar.	
	2ª4. Cierra el diálogo y vuelve al paso 1.
Postcondiciones	Se listaron los documentos clínicos de un paciente según los criterios de búsqueda seleccionados.

Tabla 2.2 Descripción del CUS Buscar documentos clínicos

Partiendo de un análisis de la situación problemática, se realizó la propuesta de un sistema informático para dar solución al problema planteado. Dada la necesidad de una buena comprensión del contexto, se realizó un diagrama del modelo de dominio, donde se relacionan los principales conceptos del área de aplicación. Se identificaron nueve requisitos funcionales y seis no funcionales que debe cumplir el sistema. A partir de los requerimientos funcionales identificados se definieron siete CUS y se elaboró un diagrama para representar las interacciones entre ellos y con el actor. Además se realizó la descripción de cada uno de ellos, sirviendo de guía para el desarrollo del sistema.

Capítulo 3: Diseño del visor

En el presente capítulo se realizará el diseño de la propuesta de solución, quedarán definidos el patrón arquitectónico a seguir y los patrones a utilizar para el diseño de la misma. Se analizarán posibles implementaciones componentes o módulos ya existentes que puedan ser reutilizados. También se realizarán los diagramas de clases del diseño y los diagramas de interacción. La descripción de cada una de las clases del sistema permitirá un mayor entendimiento y por lo tanto rapidez y calidad en su implementación.

3.1 Arquitectura del sistema

Las técnicas metodológicas desarrolladas con el fin de facilitar la programación se engloban dentro de la llamada Arquitectura de *software* o Arquitectura lógica. Se refiere a un grupo de abstracciones y patrones que brindan un esquema de referencia útil para guiarnos en el desarrollo de *software* dentro de un sistema informático. Así, los programadores, diseñadores, ingenieros y analistas pueden trabajar bajo una línea común que les posibilite la compatibilidad necesaria para lograr el objetivo deseado. (21)

3.1.1 Patrón arquitectónico: Modelo Vista Controlador

Los patrones arquitectónicos definen la estructura general del *software* e indican las relaciones entre los subsistemas y los componentes. Definen las reglas para especificar las relaciones entre los elementos (clases, paquetes, componentes, subsistemas) de la arquitectura, lo que evidencia su importancia para el desarrollo de un software. (22)

El patrón Modelo Vista Controlador (MVC) plantea la separación del problema en tres capas: la capa modelo, que representa la realidad; la capa controlador, que conoce los métodos y atributos del modelo, recibe y realiza lo que el usuario quiere hacer; y la capa vista, que muestra un aspecto del modelo y es utilizada por la capa anterior para interactuar con el usuario. (23)

Descripción de los componentes:

- Modelo: Esta es la representación específica de la información con la cual el sistema opera. Básicamente contiene la lógica de negocio real, el dominio de la aplicación con sus clases, los métodos *get* y *set* y los objetos de acceso a datos. Encapsula los datos y las funcionalidades. El

modelo es independiente de cualquier representación de salida y/o comportamiento de entrada. Se evidencia en las clases entidades Paciente y Documento Clínico.

- Vista: Este presenta el modelo en un formato adecuado para interactuar con el usuario. Responsable de la lógica de presentación y captura de datos del sistema. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador. Se evidencia en la clase interfaz Principal.
- Controlador: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista. Reciben las entradas, como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, etc. Los eventos son traducidos a solicitudes de servicio (*requests*) para el modelo o la vista, traslada las peticiones de la vista al modelo, y según la respuesta, la reenvía o no a la vista. Carga objetos y opera con ellos. (24) Se evidencia en la clase Controladora.

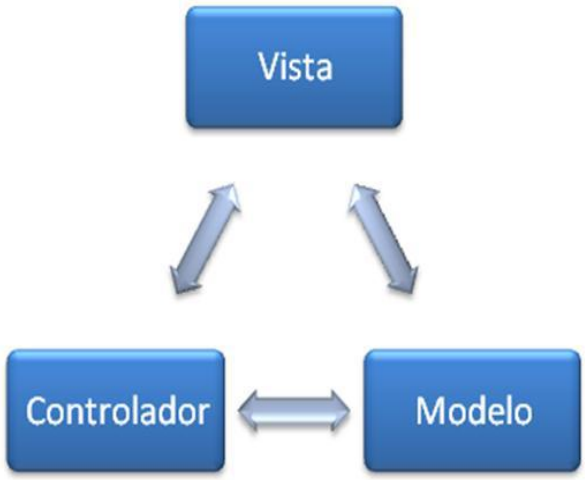


Figura 3.1 Patrón Arquitectónico Modelo Vista Controlador

Se ha decidido utilizar como patrón arquitectónico el MVC ya que este tiene como ventaja que se pueden localizar rápidamente los errores. Además mediante la separación de las capas de presentación, lógica de negocio, y acceso a datos se logra el desarrollo de arquitecturas consistentes y reutilizables. El uso de este patrón trae asociado un grupo de patrones de diseño como son el patrón Bajo acoplamiento y el patrón Alta cohesión. Permite desarrollar aplicaciones más escalables en cuanto a modificación de datos y mantenimiento del sistema.

3.1.2 Patrones de Diseño. Fundamentación

Los patrones de diseño permiten identificar y completar los casos de uso básicos expuestos por el cliente, comprender la arquitectura del sistema, así como su problemática. Brindan la posibilidad de buscar componentes ya desarrollados que cumplan con los requisitos del tipo de aplicación a construir. Fundamentalmente, permiten obtener de una forma sencilla la arquitectura base que se busca durante la fase de diseño arquitectónico. (25)

Los patrones GRASP (*General Responsibility Assignment Software Patterns*) constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objetos. Aplican el razonamiento para el diseño de una forma sistemática, racional y explicable. (26) Permiten construir sistemas reutilizables, ya que proporcionan un estándar para el diseño de todo tipo de aplicaciones informáticas.

Patrones GRASP a utilizar en el desarrollo del sistema:

- Bajo acoplamiento: El acoplamiento mide qué tan fuerte es la dependencia entre las clases. Una clase con un bajo acoplamiento tiene una dependencia mínima de otras clases lo que resulta muy conveniente, pues en caso de producirse una modificación en ella, se tendrá la mínima repercusión posible en el resto de las clases. Este patrón se ve reflejado en las clases Documento Clínico y Paciente utilizado fundamentalmente para potenciar la reutilización.
- Alta cohesión: La cohesión es una medida de cuán enfocadas están las responsabilidades de una clase. Este patrón se evidencia en las clases Documento Clínico y Paciente ya que cada una contiene los atributos y métodos correspondientes a los documentos clínicos y a los pacientes respectivamente. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo excesivo, o sea, que realicen solo las tareas que le son asignadas y no asuman responsabilidades que deberían haber delegado a otros objetos.
- Experto: Se usa más que cualquier otro al asignar responsabilidades. El patrón experto es utilizado en la mayoría de las clases de la aplicación, pues las clases se crean de acuerdo a las funcionalidades que van a tener y a la información que van a manejar, estas clases son expertas en esa información y por lo tanto las más indicadas para manejarla.

- Controlador: Se encarga de asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. La clase Controladora sirve como intermediario entre la clase interfaz Principal y las clases entidades Documento Clínico y Paciente.
- Creador: Se encarga de asignar la responsabilidad relacionada con la creación de objetos. La función de este patrón es encontrar un creador que necesite conectarse al objeto creado en alguna situación, logrando así mayor reutilización. Brinda soporte a un bajo acoplamiento, lo que supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización. Un ejemplo del mismo se aprecia en las clases Controladora y Paciente, donde la clase Controladora es la encargada de la creación del objeto de la clase Paciente.

3.2 Análisis de posibles implementaciones, componentes o módulos ya existentes y que puedan ser reutilizados

En la actualidad la reutilización de código es fundamental para el ahorro de tiempo y esfuerzos en el desarrollo de sistemas informáticos. Los documentos clínicos que se visualizarán en la aplicación son archivos XML generados por el HIS del CESIM. Para su visualización se hizo uso de la plantilla *Extensible Stylesheet Language* (XSL), pues ofrece formato CDA a los documentos clínicos. Durante el estudio del estado del arte, se analizó el Visor de HCE del HIS del CESIM y se decidió tomarlo como guía para la especificación de requisitos y el desarrollo de las funcionalidades del sistema.

3.3 Diagramas de clases de diseño

Por cada caso de uso se realiza un diagrama de clases que muestra las diferentes clases que componen el sistema y cómo se relacionan entre sí. A continuación se muestra el diagrama de clases del diseño realizado para el caso de uso Buscar documentos clínicos. Los demás diagramas de clases realizados se podrán observar en el Anexo 3.

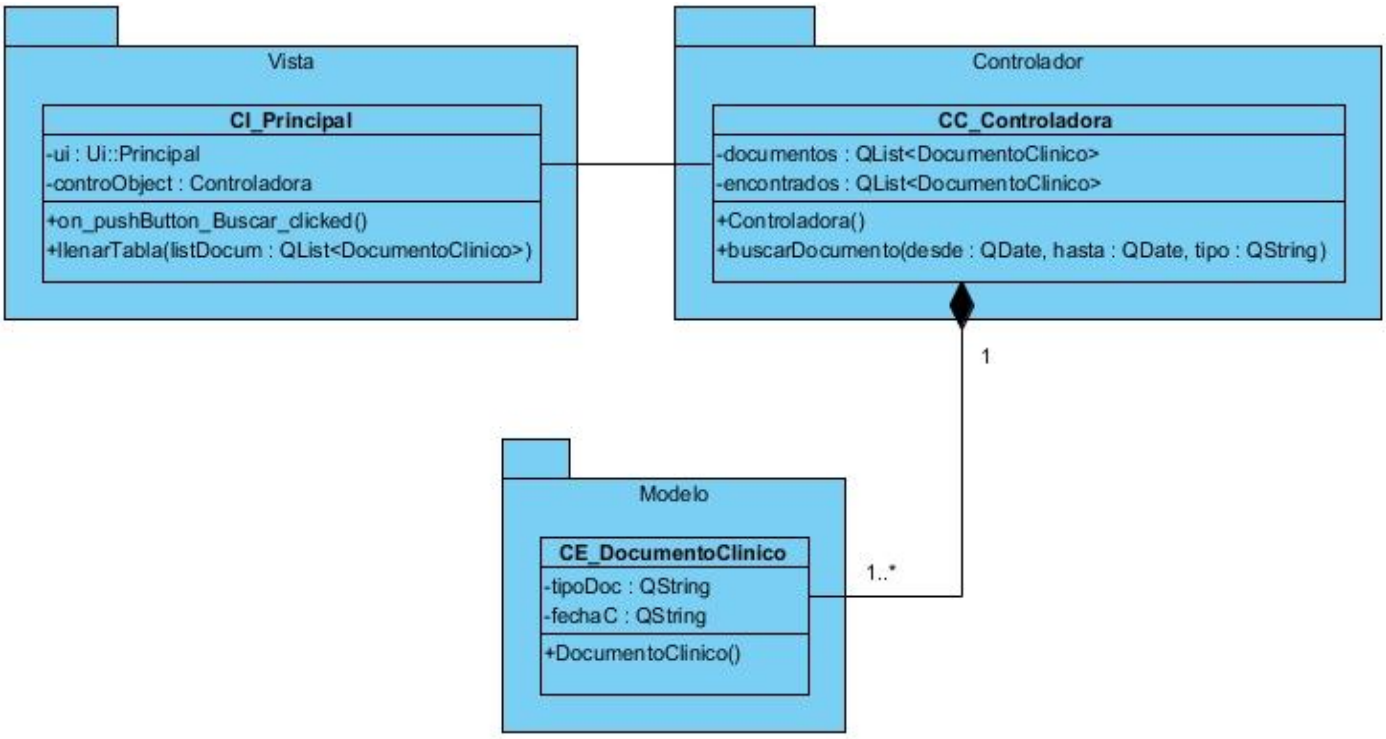


Figura 3.2 Diagrama de clases del diseño Buscar documento clínico

3.4 Descripción de las clases

Nombre: Principal	
Tipo de clase: Interfaz	
Atributo	**Tipo**
*ui	Ui::Principal
controObject	Controladora
Barra	QToolBar
url	Qstring

openUrl	QStringList
Para cada responsabilidad:	
Nombre:	Principal()
Descripción:	Inicializa los atributos de la clase.
Nombre:	on_pushButton_Buscar_clicked()
Descripción:	Muestra un listado de documentos clínicos según los criterios de búsqueda.
Nombre:	llenarTabla(listDocum : QList<DocumentoClinico>)
Descripción:	Muestra un listado de documentos clínicos.
Nombre:	pdf()
Descripción:	Cambia la extensión .xml de los documentos clínicos por la extensión .pdf y los guarda en una ubicación especificada por el usuario.
Nombre:	generarToolBar()
Descripción:	Crea una barra de herramientas.
Nombre:	MostrarDc(url: QString)
Descripción:	Muestra un documento clínico seleccionado por el usuario.
Nombre:	on_pushButton_cancelar_clicked()
Descripción:	Actualiza la tabla donde se listan los documentos clínicos, mostrando todos los documentos del paciente.
Nombre:	on_pushButton_docuRelac_clicked()

Descripción:	Muestra el documento relacionado al que se está visualizando, en caso de que lo posea.
Nombre:	on_tableWidget_cellDoubleClicked(row: int, column: int)
Descripción:	Llama al método “mostrarDc” pasándole la url del documento que se encuentra en la fila de la tabla “row” si el usuario selecciona la columna 5.
Nombre:	on_pushButton_volver_clicked()
Descripción:	Regresa a la primera página de la interfaz.
Nombre:	on_pushButton_anterior_clicked()
Descripción:	Regresa al documento mostrado anteriormente.

Tabla 3.1 Descripción de la clase interfaz Principal

Nombre: Controladora	
Tipo de clase: Controladora	
Atributo	**Tipo**
Documentos	QList<DocumentoClinico>
Encontrados	QList<DocumentoClinico>
Pacient	Paciente
File	QFile
Xml	QXmlStreamReader
Xmls	QStringList

Para cada responsabilidad:	
Nombre:	Controladora()
Descripción:	Inicializa los atributos de la clase.
Nombre:	parserDc()
Descripción:	Parsea todos los documentos clínicos del paciente.
Nombre:	buscarDocumento(desde : QDate, hasta : QDate, tipo : QString)
Descripción:	Busca los documentos clínicos del paciente que cumplan con los criterios de búsqueda entrados por parámetros.
Nombre:	mostrarDatosPaciente()
Descripción:	Parsea todos los datos del paciente
Nombre:	nombre()
Descripción:	Busca en el documento clínico el nombre, los apellidos, el sexo y la fecha de nacimiento del paciente.
Nombre:	cedula()
Descripción:	Busca en el documento clínico la cédula del paciente.
Nombre:	calcEdad()
Descripción:	A partir de la fecha de nacimiento del paciente calcula su edad.
Nombre:	docuMostrar()
Descripción:	Devuelve una lista con los nombres de los documentos que se encuentran

	listados en la tabla.
Nombre:	comparaFecha(x: QDate, fecha: QString)
Descripción:	Compara las fechas pasadas por parámetro y devuelve *true* si la primera fecha es mayor que la segunda.
Nombre:	buscarRelacionado(doc: DocumentoClinico)
Descripción:	A partir de un documento clínico busca su documento relacionado.

Tabla 3.2 Descripción de la clase Controladora

Nombre: DocumentoClinico	
Tipo de clase: Entidad	
Atributo	**Tipo**
tipoDoc	QString
fechaCrea	QString
Version	QString
horaCrea	QString
Firmado	QString
Para cada responsabilidad:	
Nombre:	DocumentoClinico()
Descripción:	Inicializa los atributos de la clase.

Nombre:	effectiveTime(value: QString)
Descripción:	A partir del valor entrado por parámetro obtiene la fecha y la hora de creación del documento clínico.

Tabla 3.3 Descripción de la clase entidad DocumentoClinico

Nombre: Paciente	
Tipo de clase: Entidad	
Atributo	**Tipo**
Nombre	QString
primerApell	QString
segundoApell	QString
Cedula	QString
Sexo	QString
tipoPaciente	QString
aboRh	QString
Edad	int
fechaNaci	QString
Para cada responsabilidad:	
Nombre:	Paciente()

Descripción:	Inicializa los atributos de la clase.
Nombre:	cumple(&value: QString)
Descripción:	A partir del valor pasado por parámetro obtiene la fecha de nacimiento del paciente.

Tabla 3.4 Descripción de la clase entidad Paciente

3.5 Diagramas de interacción: Diagramas de secuencia

Un diagrama de interacción muestra la relación que existe entre un conjunto de objetos en un escenario específico del sistema, incluyendo los mensajes que se pueden enviar entre ellos. Se pueden expresar en diagramas de colaboración y en diagramas de secuencia. Los primeros destacan la organización de los objetos que participan en una interacción y los segundos modelan las interacciones entre las clases de diseño, mediante la transferencia de mensajes entre objetos, teniendo en cuenta la secuencia temporal en la que se originan.

Los diagramas de secuencia se desarrollan tras examinar la descripción de un caso de uso, para determinar qué objetos son necesarios para la implementación del escenario. A continuación se muestra el diagrama de secuencia para el caso de uso Buscar documentos clínicos. Los demás diagramas de secuencia realizados se podrán observar en el Anexo 4.

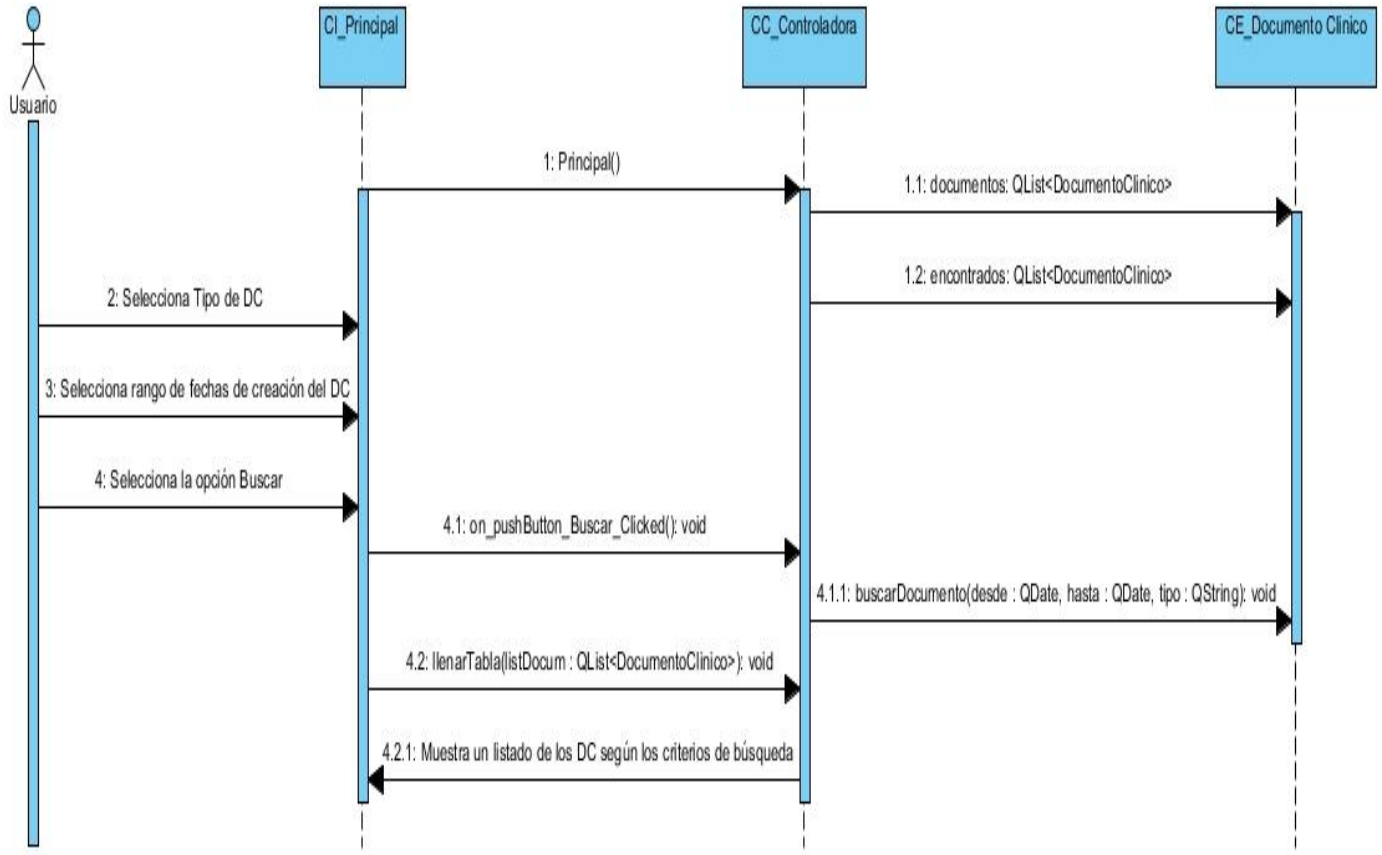


Figura 3.3 Diagrama de Secuencia Buscar documento clínico

Teniendo en cuenta las características que tiene la solución propuesta, se realizó el diseño de la arquitectura definida para el sistema. Se definió como patrón arquitectónico el MVC y los patrones GRASP para el diseño. Mediante los diagramas de clases y de interacción quedaron definidas las clases asociadas a cada caso de uso y las relaciones existentes entre ellas. Además se describieron detalladamente las clases del sistema para una mayor comprensión por parte de los desarrolladores en la fase de implementación.

Capítulo 4: Implementación del visor

En el presente capítulo se muestra la disposición física de los distintos nodos por medio del diagrama de despliegue y se implementan las clases definidas en el capítulo anterior en términos de componentes, siendo parte del diagrama de componentes y dando una visión de cómo quedará construida y distribuida la aplicación. Igualmente se definen los estándares y estilos de codificación a utilizar y se describe el tratamiento de errores a realizar.

4.1 Diagrama de despliegue

Los Diagramas de despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. (27)

Un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación. Un nodo puede contener instancias de componentes de *software*, objetos, procesos. Las instancias de componentes de *software* pueden estar unidas por relaciones de dependencia, posiblemente a interfaces, ya que un componente puede tener más de una interfaz. (27) A continuación se muestra el diagrama de despliegue para la solución propuesta:

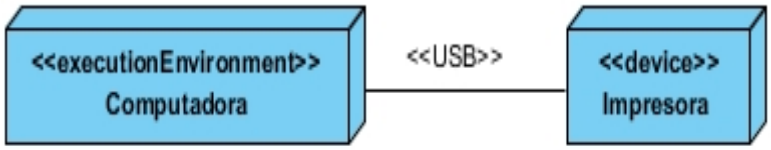


Figura 4.1 Diagrama de despliegue

4.2 Diagrama de componentes

Un diagrama de componentes muestra las dependencias lógicas entre componentes de *software*, que representan una unidad de código fuente, binario o ejecutable, indicando si son útiles en tiempo de compilación, enlace o ejecución. El diagrama se representa como un grafo cuyos nodos están unidos por medio de relaciones de dependencia entre ellos. (28) A continuación se muestra el diagrama de componentes realizado:

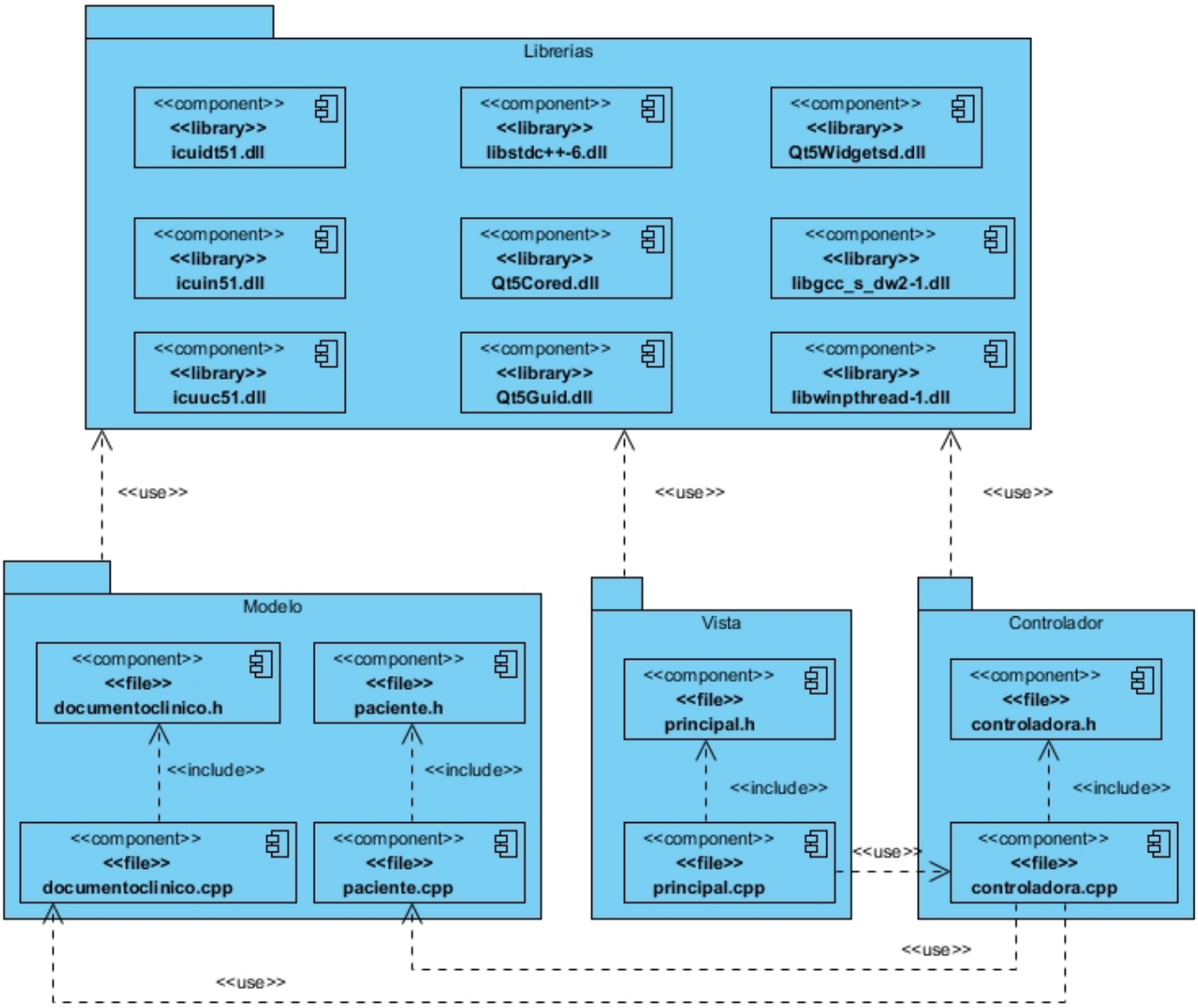


Figura 4.2 Diagrama de componentes

4.3 Interfaces principales

A continuación se muestran las principales interfaces con las que cuenta la aplicación desarrollada:

Figura 4.3 Interfaz Principal (página 1)

En la Figura 4.1 se muestran varias áreas:

A: Barra de título donde se especifica el nombre de la aplicación.

B: Presenta los datos generales del paciente que posee los documentos clínicos.

C: Permite seleccionar los criterios de búsqueda para los documentos clínicos que se desean visualizar.

D: Tabla que muestra el listado de documentos clínicos del paciente.

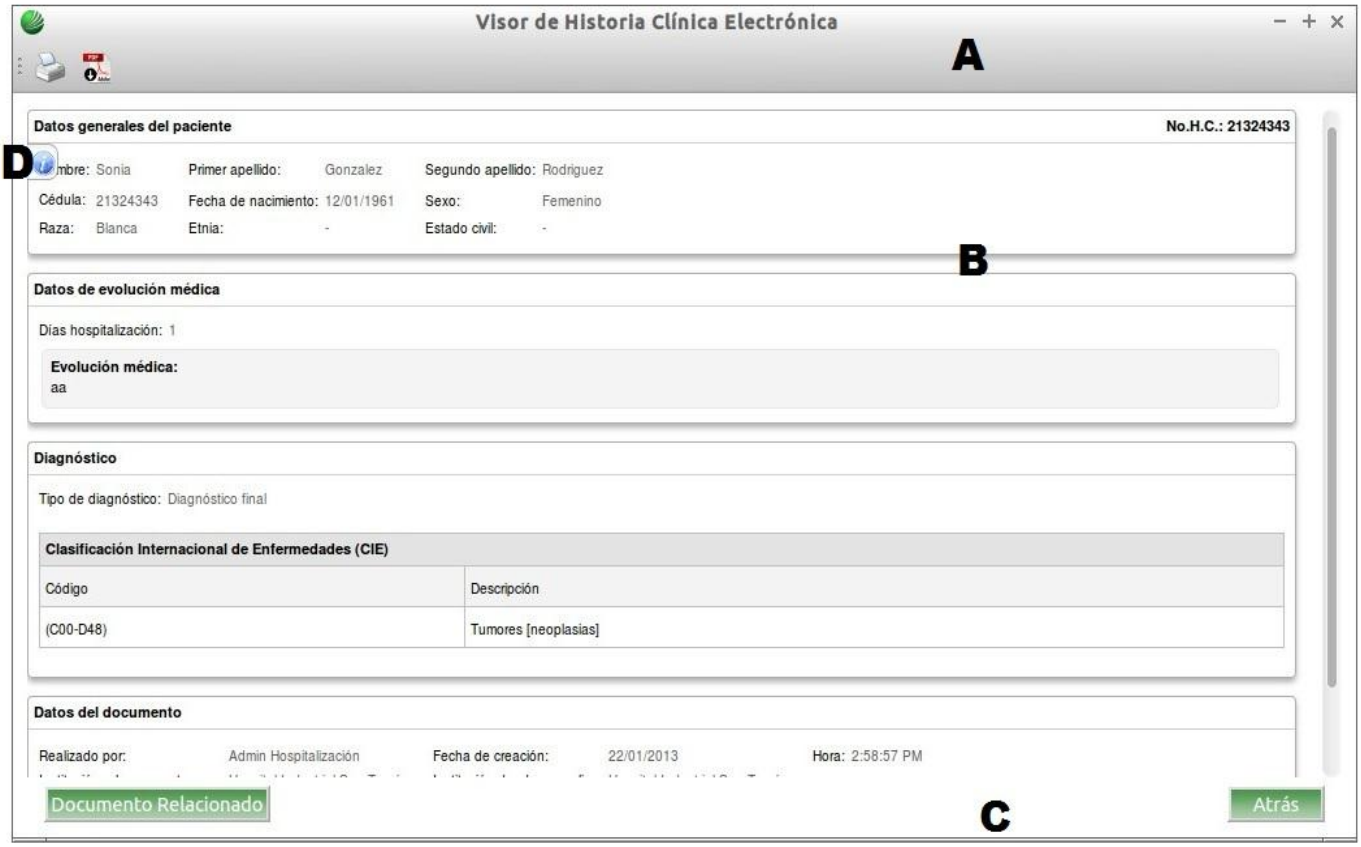


Figura 4.4 Interfaz Principal (página 2)

En la figura 4.2 se muestran varias áreas:

A: Barra de título donde se especifica el nombre de la aplicación y una barra de herramientas con las funcionalidades de Imprimir y Exportar a PDF.

B: Área donde se visualiza el documento clínico seleccionado por el usuario.

C: Área donde se encuentra el botón “Documento Relacionado” para acceder al documento relacionado al que se está visualizando y el botón “Atrás” para volver a la página 1 de la interfaz.

D: Menú de navegación que contiene las secciones del documento clínico que se está visualizando.

4.4 Tratamiento de errores

Para la implementación de un sistema es necesario tener en cuenta los posibles errores que puedan surgir durante su ejecución. El manejo de estos errores le permite al programador crear aplicaciones

tolerantes a fallas, garantizando así la integridad y confiabilidad de los datos. C++ es un lenguaje de programación que soporta el tratamiento de errores, evitando así la vulnerabilidad del sistema.

En el sistema realizado se muestran mensajes de información a través de “QMessageBox”, que son de fácil comprensión para el usuario. Por citar un ejemplo: cuando el usuario introduce criterios de búsqueda para filtrar los documentos clínicos del paciente que desea visualizar y no existen documentos que cumplan con dichos criterios, el sistema muestra el mensaje “No existen documentos clínicos que cumplan con los criterios de búsqueda seleccionados”.

4.4 Seguridad

El sistema realizado tiene como objetivo principal que los pacientes puedan conservar su HCE y la puedan visualizar desde cualquier estación de trabajo. Una vez entregados los documentos clínicos y la aplicación al paciente en un CD o DVD no reescribible, este será responsable por la seguridad de sus datos. Los datos solo serán confiables mientras estén almacenados en el dispositivo entregado al paciente desde el HIS de la institución donde se encuentra registrado. Por esta razón la aplicación desarrollada no tiene en cuenta restricciones de seguridad para la visualización de los documentos clínicos.

4.5 Estrategias de codificación. Estándares y estilos a utilizar

Las convenciones o estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código. (29) Estos estándares facilitan el mantenimiento del código, sirven como punto de referencia para los programadores, mantienen un estilo de programación y ayudan a mejorar el proceso de codificación, haciéndolo, entre otras cosas, más eficiente.

Algunos estándares de codificación para el lenguaje C++ utilizados:

Los identificadores son nombres de variables (arreglos, matrices, apuntadores), funciones, así como cualquier tipo de dato definido por el usuario (estructura, clase). Se regirán por las siguientes normas, además de las definidas por el propio lenguaje:

- Deberán tener un nombre significativo para que por su simple lectura, pueda conocerse su función, sin tener que consultar manuales o hacer demasiados comentarios.

- Para nombres que se usen con frecuencia o para términos largos, se recomienda usar abreviaturas estándares para que tengan una longitud razonable. Si usa abreviaturas deben manejar la misma lógica en todo el programa.
- Evitar identificadores que comiencen con uno o dos caracteres de subrayado para evitar que se confundan con los que el compilador selecciona.
- Para distinguir palabras dentro del nombre deberá emplearse una letra mayúscula o un guión bajo (_).

  Ejemplos:

  DocumentoClinico.

  documento_clinico.
- El nombre de los indicadores de punteros o apuntadores debe comenzar con la letra p.

  Ejemplo: pPaciente.
- En los identificadores de tipos definidos por el usuario la primera letra será mayúscula.

  Ejemplo: class Clase o struct Estructura.

La organización visual del programa se regirá por las siguientes normas:

- No manejar en los programas más de una instrucción por línea.
- Declarar las variables en líneas separadas.
- Añadir comentarios descriptivos junto a cada declaración de variables, si es necesario.
- La estructura para las llaves que definen el cuerpo de una función se muestra en el siguiente ejemplo.

  Ejemplo: void Funcion ( )

  {

  //Instrucciones de la función

  }

- La estructura para las instrucciones del cuerpo de cada estructura de control se muestra en el siguiente ejemplo:

  Ejemplo: for (int x = 0; x < 5; x++)

  {

  //Instrucciones a ejecutar.

  }

- Los operadores unarios (++, --, etc.) deben ponerse junto a sus operandos, sin espacios intermedios.

En este capítulo se revelaron los resultados obtenidos durante la fase de implementación; quedando expuestos el diagrama de despliegue, que brinda los elementos necesarios para un correcto proceso de despliegue de la aplicación informática, y el diagrama de componentes, donde se especifican los componentes (librerías, ejecutables, archivos, etc.) del sistema. Asimismo se muestran las principales interfaces del sistema y se define como se realiza el tratamiento de errores y de la seguridad. Por último se relacionan los estándares de codificación por los que se rige el sistema, permitiendo el desarrollo de una aplicación tolerante a fallas con un código fácilmente entendible y reutilizable.

Beneficios

Como resultado del desarrollo de la investigación se ofrecen los siguientes beneficios:

- La aplicación realizada permite la visualización de los documentos clínicos extraídos del HIS del CESIM desde cualquier estación de trabajo, sin necesidad de conexión al sistema, ya que es una aplicación portable y multiplataforma.
- Los pacientes registrados en el HIS del CESIM podrán visualizar su HCE si lo desean.
- El sistema le permite al personal de la salud, que no tiene acceso al HIS del CESIM, visualizar la HCE de un paciente determinado, influyendo en las decisiones médicas.

Conclusiones

Con el desarrollo de la presente investigación y a partir de los resultados obtenidos se puede arribar a las siguientes conclusiones:

- Los sistemas analizados no ofrecen una solución factible para el problema planteado, lo que evidencia la necesidad de una aplicación portable para visualizar los documentos clínicos que se generan en el HIS del CESIM.
- El estudio y la caracterización de la metodología, lenguajes y herramientas utilizadas facilitaron el desarrollo del sistema.
- El análisis de los procesos que intervienen en la visualización de los documentos clínicos del HIS del CESIM permitió realizar una propuesta de solución para resolver la problemática planteada.
- Se desarrollaron los artefactos correspondientes a cada fase de la metodología RUP permitiendo un mejor entendimiento de los flujos del proceso de visualización de los documentos clínicos.
- Se desarrolló una aplicación de escritorio que permite la visualización de los documentos clínicos generados en el HIS del CESIM, facilitando la utilización de la HCE, tanto por los pacientes, como por el personal de la salud que los atiende.

Recomendaciones

Después de concluida la investigación se ofrece la siguiente recomendación:

- La implementación del visor portable para los sistemas operativos móviles *Android, Windows Phone,* iOS, etc.

Referencias Bibliográficas

1. Hospital Italiano de Buenos Aires. *La Representación del conocimiento médico. Curso Universitario.Sistemas de información en Salud.* 2009.

2. Rodríguez López, Martha y Rodríguez García, Raymundo. *Propuesta de aplicación de los perfiles de integración de IHE entre los sistemas alas PACS–alas RIS–alas HIS.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2010.

3. Indarte, Selene. Interoperabilidad.Capítulo 15. *Manual de salud electrónica para directivos de servicios y sistemas de salud.*

4. Borges Cabrera, Y. y González Díaz, R. "*Arquitectura de un Sistema para la Transmisión de Información Radiológica Inter-Hospitalaria".* Universidad de las *Ciencias Informáticas.*

5. Sánchez Romero, Maikel. *Infraestructura de software para el almacenamiento y consulta de la Historia Clínica Electrónica del sistema alas HIS*. Universidad de las Ciencias Informáticas. La Habana : s.n., 2010.

6. Melián Montalvo, Marlene. Biblioteca Virtual de las Ciencias en Cuba. [En línea] [Citado el: 21 de febrero de 2014.] http://www.bibliociencias.cu/gsdl/collect/eventos/index/assoc/HASH0104/f016d031.dir/doc.pdf.

7. Sánchez, Ramón Ramón. Software Libre y Cooperación. [En línea] noviembre de 2005. [Citado el: 16 de febrero de 2014.] http://ramonramon.org/blog/2013/06/12/gestion-hospitalaria-his-con-software-libre/.

8. Eric Maeker, MD. Free med forms. [En línea] 23 de diciembre de 2012. [Citado el: 18 de febrero de 2014.] http://Association asso.freemedforms.com.

9. Grup de Software, s.l. A-medic . [En línea] 2009. [Citado el: 20 de febrero de 2014.] http://www.a-medic.com/products.htm.

10. Varsur Corporation. MedFile Historias Clínicas Electrónicas en Español. [En línea] 1999. [Citado el: 19 de febrero de 2014.] http://www.medical-soft.com/index.php.

11. Stacks. mispacientes.es. [En línea] 2012. [Citado el: 20 de febrero de 2014.] http://www.mispacientes.es/capac/.

12. Fernández, Natacha González. Ecured.cu. [En línea] [Citado el: 21 de febrero de 2014.] http://www.ecured.cu/index.php/CMMI.

13. Neciosup Quiroz , Dante . UML. [En línea] 10 de marzo de 2008. [Citado el: 21 de febrero de 2014.] http://danequi79.blogspot.com/2008/03/herramientas-de-ayuda-para-la-ejecucion.html.

14. Hernández Orallo, Enrique. Universidad Politécnica de Valencia. *Dpto. de Informática de Sistemas y Computadores.* [En línea] [Citado el: 23 de febrero de 2014.] http://www.disca.upv.es/enheror/pdf/ActaUML.PDF.

15. Meza , Mirna . Herramientas Case . [En línea] 2 de abril de 2011. [Citado el: 25 de febrero de 2014.] http://fds-herramientascase.blogspot.com/.

16. Targetware Informática S.A.C. . Software.com.ar. [En línea] 2007. [Citado el: 21 de febrero de 2014.] http://www.software.com.ar/visual-paradigm-para-uml.html.

17. Gitorious A.S. QtProject. [En línea] 2013. [Citado el: 24 de febrero de 2014.] http://qt-project.org/doc/qt-5/topics-app-development.html.

18. Desarrollo Movil Multiplataforma. [En línea] 26 de agosto de 2012. [Citado el: 25 de febrero de 2014.] http://desarrollomovilmultiplataforma.blogspot.com/2012/08/aspectos-teoricos-framework.html.

19. CódigoQt. [En línea] 24 de marzo de 2013. [Citado el: 25 de febrero de 2014.] http://www.codigoqt.com/index.php/topic,22.0.html.

20. ALEGSA. AlegsaOnLine.com. [En línea] 6 de septiembre de 2006. [Citado el: 25 de febrero de 2014.] http://www.alegsaonline.com/art/11.php.

21. Guglielmetti, Marcos. MASTERMAGAZINE. *Definición de Arquitectura de Software.* [En línea] [Citado el: 28 de marzo de 2014.] http://www.mastermagazine.info/termino/3916.php.

22. Pressman, Roger. *Ingeniería del Software.* Edición: 7ma ed, McGraw-Hill. 958 p. ISBN 970-10-5473-3.

23. Tedeschi, Nicolás. MSDN Library. [En línea] 2012. [Citado el: 28 de marzo de 2014.] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

24. Martínez Jera, Erislán y González Enríquez, Landy. *Desarrollo de la Gestión de Datos de Ensayos Clínicos a partir del sistema OpenClinica.* La Habana : Universidad De las Ciencias Informáticas, 2009.

25. Bernal Kaiser, Armando Moisés y López Escobar, Mireya. UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO. *Facultad de contaduría y administración.* [En línea] septiembre de 2005. [Citado el: 28 de marzo de 2014.] http://fcasua.contad.unam.mx/apuntes/interiores/docs/2005/informatica/1/1166.pdf.

26. Larman, Craig. *UML y Patrones.* Edición: 2da ed, s.l. : Prentice Hall.

27. Marca Huallpara , Hugo Michael y Quisbert Limachi, Nancy Susana. Universidad Salesiana Bolivia. *Diagrama de Despliegue.* [En línea] 6 de diciembre de 2007. [Citado el: 24 de abril de 2014.] http://virtual.usalesiana.edu.bo/web/practica/archiv/.

28. Figueroa, Pablo. University of Alberta. [En línea] [Citado el: 24 de abril de 2014.] http://webdocs.cs.ualberta.ca/~pfiguero/soo/uml/implementacion01.html.

29. DEPARTAMENTO DE ELECTRÓNICA, SISTEMAS E INFORMÁTICA. Coordinacion Docente de Lenguajes de Programación. [En línea] [Citado el: 24 de abril de 2014.] http://progra.iteso.mx/estandares/estandar%20codificacion%20c++/.

Bibliografía

Aires, Hospital Italiano de Buenos. *La Representación del conocimiento médico. Curso Universitario.Sistemas de información en Salud. 2009.*

Albet. [En línea] [Citado el: 10 de diciembre de 2013.] http://www.albet.uci.cu/lineas/salud-y-ciencias/alas-his.

ALEGSA. AlegsaOnLine.com. [En línea] 6 de septiembre de 2006. [Citado el: 25 de febrero de 2014.] http://www.alegsaonline.com/art/11.php.

Alonso Lanza, Ing. José Luis. *La historia clínica electrónica: ideas, experiencias y reflexiones.* La Habana : s.n., 2005, Vol. 13.

Bernal Kaiser, Armando Moisés y López Escobar, Mireya. UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO. *Facultad de contaduría y administración.* [En línea] septiembre de 2005. [Citado el: 28 de marzo de 2014.] http://fcasua.contad.unam.mx/apuntes/interiores/docs/2005/informatica/1/1166.pdf.

Borges Cabrera, Y. y González Díaz, R. "*Arquitectura de un Sistema para la Transmisión de Información Radiológica* Inter-Hospitalaria". Universidad de las Ciencias Informáticas.

Ciberaula Java. *Patrones de Diseño en aplicaciones Web con Java2EE*. [En línea] 2010. [Citado el: 28 de marzo de 2014.] http://java.ciberaula.com/articulo/diseno_patrones_j2ee.

CódigoQt. [En línea] 24 de marzo de 2013. [Citado el: 25 de febrero de 2014.] http://www.codigoqt.com/index.php/topic,22.0.html.

DEPARTAMENTO DE ELECTRÓNICA, SISTEMAS E INFORMÁTICA. Coordinacion Docente de Lenguajes de Programación. [En línea] [Citado el: 24 de abril de 2014.] http://progra.iteso.mx/estandares/estandar%20codificacion%20c++/.

Desarrollo Movil Multiplataforma. [En línea] 26 de agosto de 2012. [Citado el: 25 de febrero de 2014.] http://desarrollomovilmultiplataforma.blogspot.com/2012/08/aspectos-teoricos-framework.html.

Eric Maeker, MD. Free med forms. [En línea] 23 de diciembre de 2012. [Citado el: 18 de febrero de 2014.] http://Association asso.freemedforms.com.

Bibliografía

Fernández y Fernández, Carlos Alberto. El Proceso Unificado Rational para el Desarrollo de Software. Universidad Tecnológica de la Mixteca. [En línea] 26 de octubre de 2000. [Citado el: 24 de marzo de 2014.] http://nuyoo.utm.mx/~caff/doc/El%20Proceso%20Unificado%20Rational.pdf.

Fernández, Natacha González. Ecured.cu. [En línea] [Citado el: 21 de febrero de 2014.] http://www.ecured.cu/index.php/CMMI.

Figueroa, Pablo. University of Alberta. [En línea] [Citado el: 24 de abril de 2014.] http://webdocs.cs.ualberta.ca/~pfiguero/soo/uml/implementacion01.html.

Frontela Cabrera, Ismary y Angulo Díaz, Lorenzo. *Componente Repositorio de Datos Clinicos de la Historia* Clinica Electronica para el Centro de Informatica Medica. La Habana : s.n., 2013.

García de Jalón, Javier, y otros, y otros. *Aprenda C++ como si estuviera en primero.* San Sebastian : Universidad de Navarra, 1998.

Gitorious A.S. QtProject. [En línea] 2013. [Citado el: 24 de febrero de 2014.] http://qt-project.org/doc/qt-5/topics-app-development.html.

Gómez Sánchez, Alberto y Parellada Blanco, Jaime. *Validación de una historia clínica electrónica para pacientes graves.* 2, La Habana : Revista Cubana de Medicina Intensiva y Emergencias, 2007, Vol. 6.

Grup de Software, s.l. A-medic . [En línea] 2009. [Citado el: 20 de febrero de 2014.] http://www.a-medic.com/products.htm.

Guglielmetti, Marcos. MASTERMAGAZINE. *Definición de Arquitectura de Software.* [En línea] [Citado el: 28 de marzo de 2014.] http://www.mastermagazine.info/termino/3916.php.

Hernández Orallo, Enrique. Universidad Politécnica de Valencia. *Dpto. de Informática de Sistemas y Computadores.* [En línea] [Citado el: 23 de febrero de 2014.] http://www.disca.upv.es/enheror/pdf/ActaUML.PDF.

Indarte, Selene. Interoperabilidad.Capítulo 15. *Manual de salud electrónica para directivos de servicios y sistemas de salud.*

Larman, Craig. *UML y Patrones.* Edición: 2da ed, s.l. : Prentice Hall.

Bibliografía

Madrigal Cruz, Rolando Louis. Informática en Salud 2007. [En línea] enero de 2007. [Citado el: 21 de febrero de 2014.] http://www.informatica2007.sld.cu/Members/rolandomc/presentacion-de-un-software-de-hce-para-atencion-primaria-secundaria-terciaria-y-gerencia-de-instalaciones-hospitalarias-1/.

Marañón, G.Á. *Características del lenguaje Java*. [En línea] 2006; [Citado el: 21 de febrero de 2014.] http://www.iec.csic.es/CRIPTONOMICON/java/quesjava.html.

Marca Huallpara , Hugo Michael y Quisbert Limachi, Nancy Susana. Universidad Salesiana Bolivia. *Diagrama de Despliegue.* [En línea] 6 de diciembre de 2007. [Citado el: 24 de abril de 2014.] http://virtual.usalesiana.edu.bo/web/practica/archiv/.

Martínez Jera, Erislán y González Enríquez, Landy. *Desarrollo de la Gestión de Datos de Ensayos Clínicos a partir del sistema OpenClinica.* La Habana : Universidad De las Ciencias Informáticas, 2009

Melián Montalvo, Marlene. Biblioteca Virtual de las Ciencias en Cuba. [En línea] [Citado el: 21 de febrero de 2014.] http://www.bibliociencias.cu/gsdl/collect/eventos/index/assoc/HASH0104/f016d031.dir/doc.pdf.

Meza , Mirna . Herramientas Case . [En línea] 2 de abril de 2011. [Citado el: 25 de febrero de 2014.] http://fds-herramientascase.blogspot.com/.

Monteagudo Peña, José Luis y Hernández Salvador, Carlos. Comunidad Virtual de Anatomía Patológica. *Estándares para la historia clínica electrónica .* [En línea] 2003. [Citado el: 20 de febrero de 2014.] http://www.conganat.org/seis/informes/2003/PDF/CAPITULO7.pdf.

Neciosup Quiroz , Dante . UML. [En línea] 10 de marzo de 2008. [Citado el: 21 de febrero de 2014.] http://danequi79.blogspot.com/2008/03/herramientas-de-ayuda-para-la-ejecucion.html.

Pressman, Roger. *Ingeniería del Software.* Edición: 7ma ed, McGraw-Hill. 958 p. ISBN 970-10-5473-3.

Rodríguez López*, Martha y Rodríguez García, Raymundo.** *Propuesta de aplicación de los perfiles de integración de IHE entre los sistemas alas PACS–alas RIS–alas HIS.** Universidad de las Ciencias Informáticas. La Habana : s.n., 2010.

Sánchez Romero, Maikel. 2010. *Infraestructura de software para el almacenamiento y consulta de la* Historia Clínica Electrónica del sistema alas HIS. Universidad de las Ciencias Informáticas. La Habana: s.n., 2010, Tesis de maestría.

Bibliografía

Sánchez Romero, Maikel. *Infraestructura de software para el almacenamiento y consulta de la Historia Clínica Electrónica del* sistema alas HIS. Universidad de las Ciencias Informáticas. La Habana : s.n., 2010.

Sánchez, Ramón Ramón. Software Libre y Cooperación. [En línea] noviembre de 2005. [Citado el: 16 de febrero de 2014.] http://ramonramon.org/blog/2013/06/12/gestion-hospitalaria-his-con-software-libre/.

Sierra Lombardía, Dra. Virginia M. y Alvarez de Zayas, Dr. Carlos M. *Metodología de la Investigación Científica.*

Stacks. mispacientes.es. [En línea] 2012. [Citado el: 20 de febrero de 2014.] http://www.mispacientes.es/capac/.

Targetware Informática S.A.C. . Software.com.ar. [En línea] 2007. [Citado el: 21 de febrero de 2014.] http://www.software.com.ar/visual-paradigm-para-uml.html.

Tedeschi, Nicolás. MSDN Library. [En línea] 2012. [Citado el: 28 de marzo de 2014.] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

Varsur Corporation. MedFile Historias Clínicas Electrónicas en Español. [En línea] 1999. [Citado el: 19 de febrero de 2014.] http://www.medical-soft.com/index.php.

Varsur Corporation. MedFile Historias Clínicas Electrónicas en Español. [En línea] 1999. [Citado el: 19 de febrero de 2014.] http://www.medical-soft.com/medfile-portable-portatil.php.

Anexo 1

Objetivo de la entrevista: Definir los requisitos funcionales y no funcionales del sistema.

Día de la entrevista: 10 de febrero de 2014

Persona entrevistada: Alain Ramos Medina (Jefe de Proyecto).

Lugar de la entrevista: Laboratorio 207, Docente 6, UCI (Puesto de trabajo).

Modo de realización: Diálogo.

Aspectos a tener en cuenta:

1. Procesos que intervienen en la visualización de los documentos clínicos generados en el HIS del CESIM.
2. Estructura de los documentos clínicos generados en el HIS del CESIM.
3. Funcionalidades con las que cuenta el módulo Visor de HCE.
4. Pautas seguidas en el diseño de las interfaces del módulo Visor de HCE.
5. Seguridad de la información de los pacientes.
6. Características del sistema a desarrollar.
7. Cualidades del sistema a desarrollar.

Anexo 2

CUS: Mostrar datos generales del paciente.

Objetivo	Mostrar los datos generales de un paciente determinado.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el actor ejecuta el sistema. Permite visualizar los datos generales del paciente en la interfaz principal del sistema. Termina el caso de uso.
Precondiciones	El sistema debe tener acceso a los documentos clínicos del paciente.
Referencia	RF 1
Prioridad	Media.
Flujo de eventos	
Actor	**Sistema**
1. Ejecuta el sistema.	
	2. Muestra los datos generales del paciente. Termina el caso de uso.
Postcondiciones	Se mostraron los datos generales de paciente.

CUS: Listar documentos clínicos.

Objetivo	Listar los documentos clínicos de un paciente.

Actores	Usuario.
Resumen	El caso de uso inicia cuando el actor ejecuta el sistema. Se muestra un listado con todos los documentos clínicos asociados a un paciente determinado. Termina el caso de uso.
Precondiciones	El sistema debe tener acceso a los documentos clínicos del paciente.
Referencia	RF 3
Prioridad	Alta.
Flujo de eventos	
Actor	**Sistema**
1. Ejecuta el sistema.	
	2. Muestra el listado de los documentos clínicos asociados a un paciente. Termina el caso de uso.
Postcondiciones	Se listaron los documentos clínicos de un paciente.

CUS: Mostrar documento clínico.

Objetivo	Mostrar un documento clínico seleccionado por el usuario.
Actores	Usuario.

Resumen	El caso de uso inicia cuando el actor selecciona un documento clínico. El sistema muestra el documento. Termina el caso de uso.
Precondiciones	El sistema debe tener acceso a los documentos clínicos del paciente.
Referencia	RF 4
Prioridad	Alta.
Flujo de eventos	
Actor	**Sistema**
1. Selecciona el documento clínico.	
	2. Muestra el documento. Termina el caso de uso.
Postcondiciones	Se mostró el documento clínico seleccionado.

CUS: Mostrar documento clínico relacionado.

Objetivo	Mostrar el documento clínico relacionado a un documento determinado.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el actor selecciona un documento clínico. El sistema muestra el documento y al final del mismo, muestra una sección con el identificador del documento clínico relacionado en caso de que lo tenga. El actor selecciona el documento clínico relacionado y el sistema lo muestra Termina el caso de uso.
Precondiciones	El sistema debe tener acceso a los documentos clínicos del paciente.

Referencia	RF 5
Prioridad	Alta.
Flujo de eventos	
Flujo Básico	
Actor	**Sistema**
1. Selecciona el documento clínico.	
	2. Muestra el documento seleccionado.
	3. Muestra el identificador del documento clínico relacionado en una sección al final del documento.
4. Selecciona la opción “Documento Relacionado”.	
	5. Muestra el documento relacionado. Termina el caso de uso.
Flujos Alternos	
3ª. No existe un documento clínico relacionado.	
Actor	**Sistema**
	3ª1. Muestra el documento seleccionado sin la sección que contiene el identificador para el documento clínico relacionado y vuelve al paso 1.

Postcondiciones	Se mostró el documento clínico relacionado.

CUS: Mostrar menú de secciones del documento clínico.

Objetivo	Mostrar un menú desde donde se pueda acceder a las secciones de un documento clínico.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el actor selecciona la opción Menú de Secciones. El sistema muestra el menú con las diferentes secciones que tiene el documento clínico Termina el caso de uso.
Precondiciones	El usuario debe seleccionar un documento clínico para ser mostrado por el sistema.
Referencia	RF 6
Prioridad	Media.

Flujo de eventos	
Flujo Básico	
Actor	**Sistema**
1. Selecciona la opción que aparece en la parte superior izquierda del documento clínico.	
	2. Muestra un menú con las diferentes secciones que

	tiene el documento clínico. Termina el caso de uso.
Postcondiciones	Se mostró un menú con las diferentes secciones que tiene el documento clínico.

CUS: Exportar a PDF.

Objetivo	Exportar un documento a PDF.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el actor después de seleccionar un documento clínico selecciona la opción Exportar a PDF. El sistema guarda la información almacenada en el documento clínico en un archivo con la extensión .pdf dentro de una ubicación especificada por el actor. Termina el caso de uso.
Precondiciones	El sistema debe tener acceso a los documentos clínicos del paciente.
Referencia	RF 7
Prioridad	Media.

Flujo de eventos	
Actor	**Sistema**
1. Selecciona documento deseado.	
	2. Habilita la opción: ✓ Exportar a PDF.
3. Selecciona la opción Exportar a PDF.	

	4. Abre una ventana que permite al usuario explorar las ubicaciones existentes en su estación de trabajo. Además brinda las opciones: ✓ Guardar. ✓ Cancelar.
5. Selecciona la ubicación deseada, escribe el nombre del archivo que desea exportar y selecciona la opción Guardar.	
	6. Verifica los datos entrados y el espacio disponible.
	7. Guarda el archivo en la ubicación especificada. Termina el caso de uso.
Flujo Alterno	
5.a El usuario selecciona la opción Cancelar.	
Actor	**Sistema**
	5.a.1 Cierra la ventana y vuelve al paso 1.
6.a Existe un error en el nombre del archivo.	
Actor	**Sistema**
	6.a.1 Indica mediante una ventana de diálogo que existe un error al guardar el archivo porque el campo nombre está incorrecto.

	Brinda la opción: ✓ Aceptar.
6.a.2 Selecciona la opción Aceptar.	
	6.a.3 Cierra el diálogo y vuelve al paso 4.
6.b No existe espacio disponible en la ubicación seleccionada.	
Actor	**Sistema**
	6.b.1 Indica mediante una ventana de diálogo que existe un error al guardar el archivo porque no se dispone de espacio en disco para ello. Brinda la opción: ✓ Aceptar.
6.b.2 Selecciona la opción Aceptar.	
	6.b.3 Cierra el diálogo y vuelve al paso 4.
Postcondiciones	Se exportó un archivo con la extensión .pdf.

Anexo 3

Diagrama de clases Mostrar datos generales del paciente.

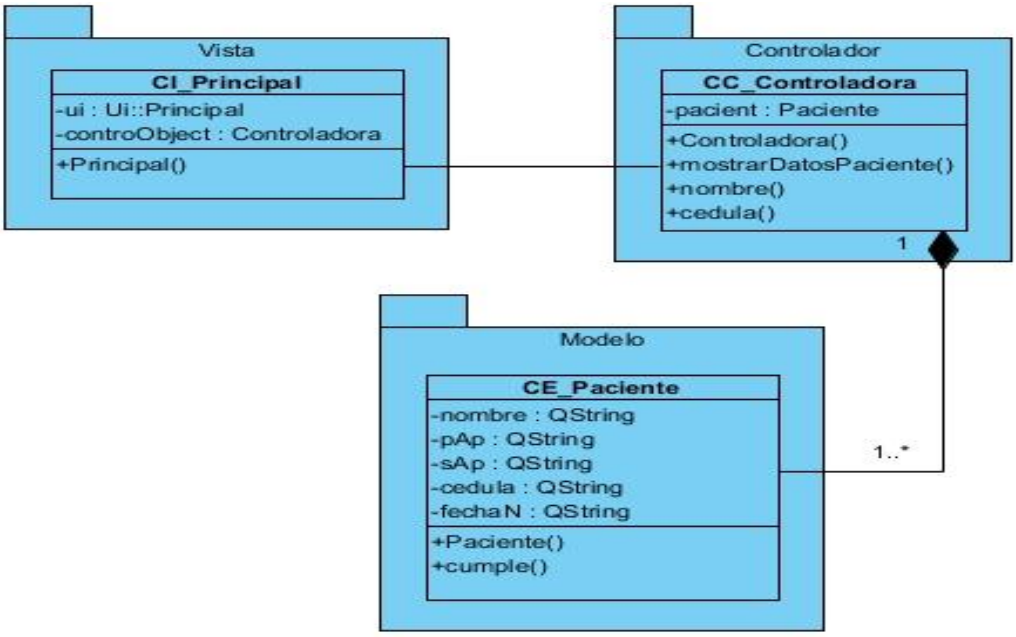


Diagrama de clases Listar documentos clínicos.

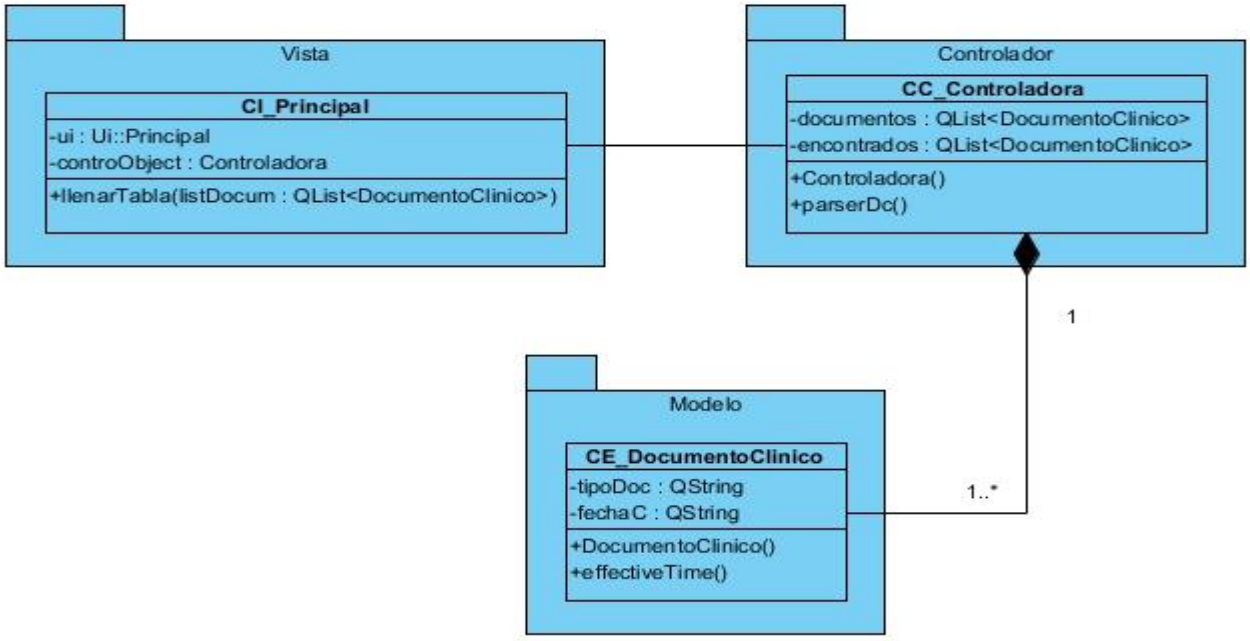


Diagrama de clases Mostrar documento clínico.

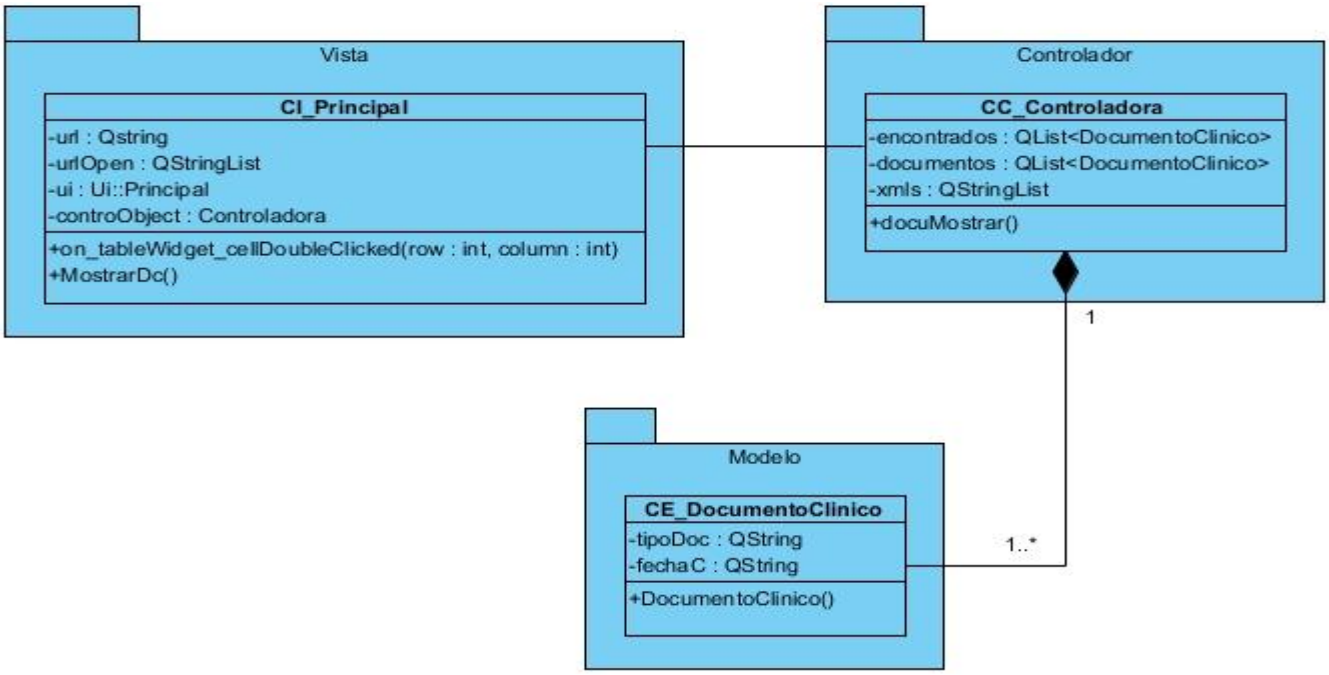


Diagrama de clases Mostrar documento clínico relacionado.

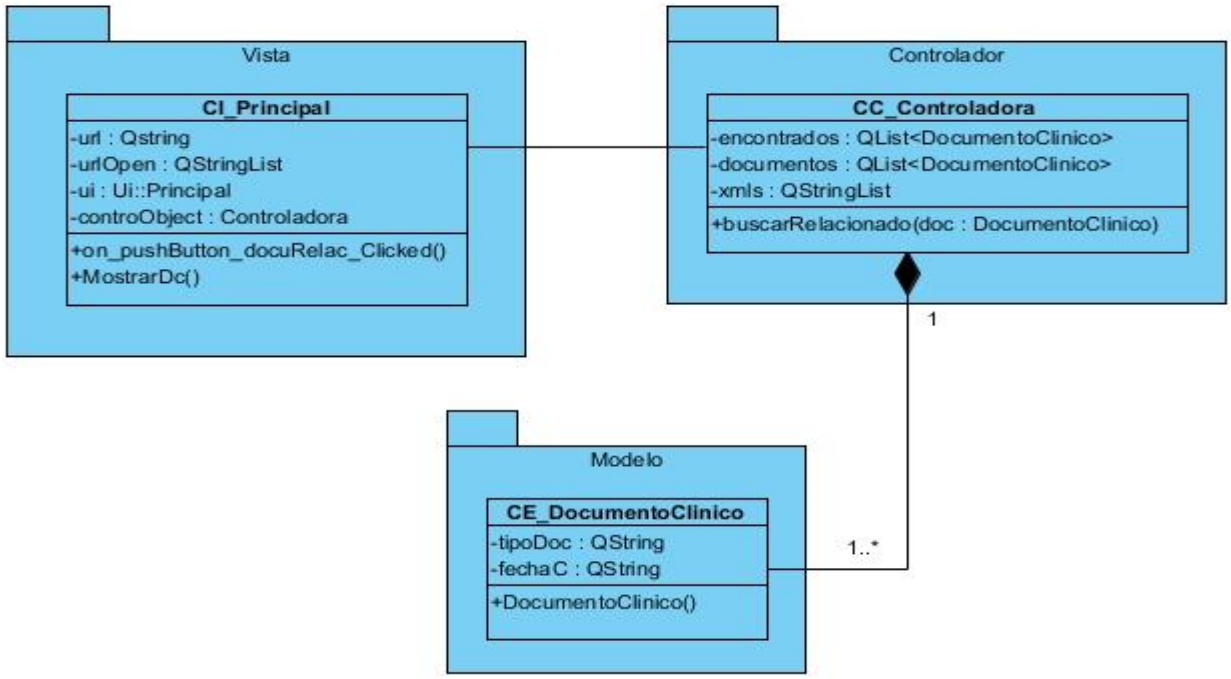


Diagrama de clases Mostrar menú de secciones del documento clínico.

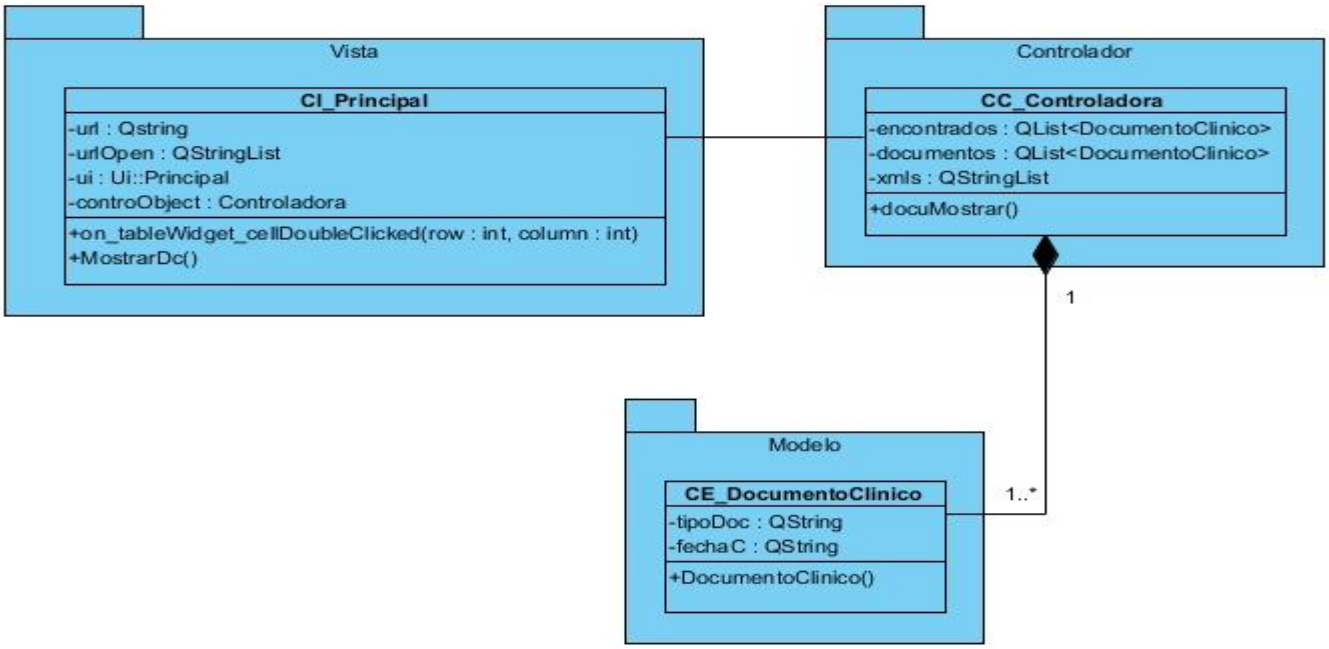
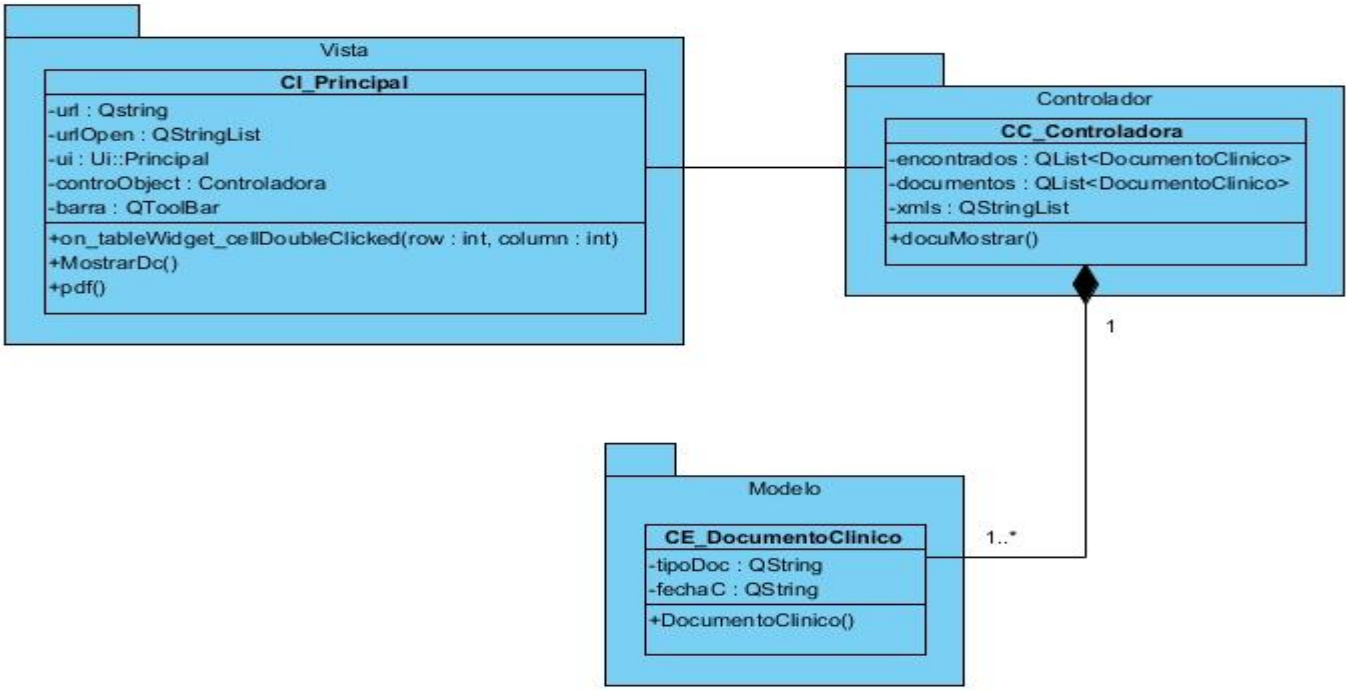


Diagrama de clases Exportar a PDF.



Anexo 4

Diagrama de secuencia Mostrar datos generales del paciente.

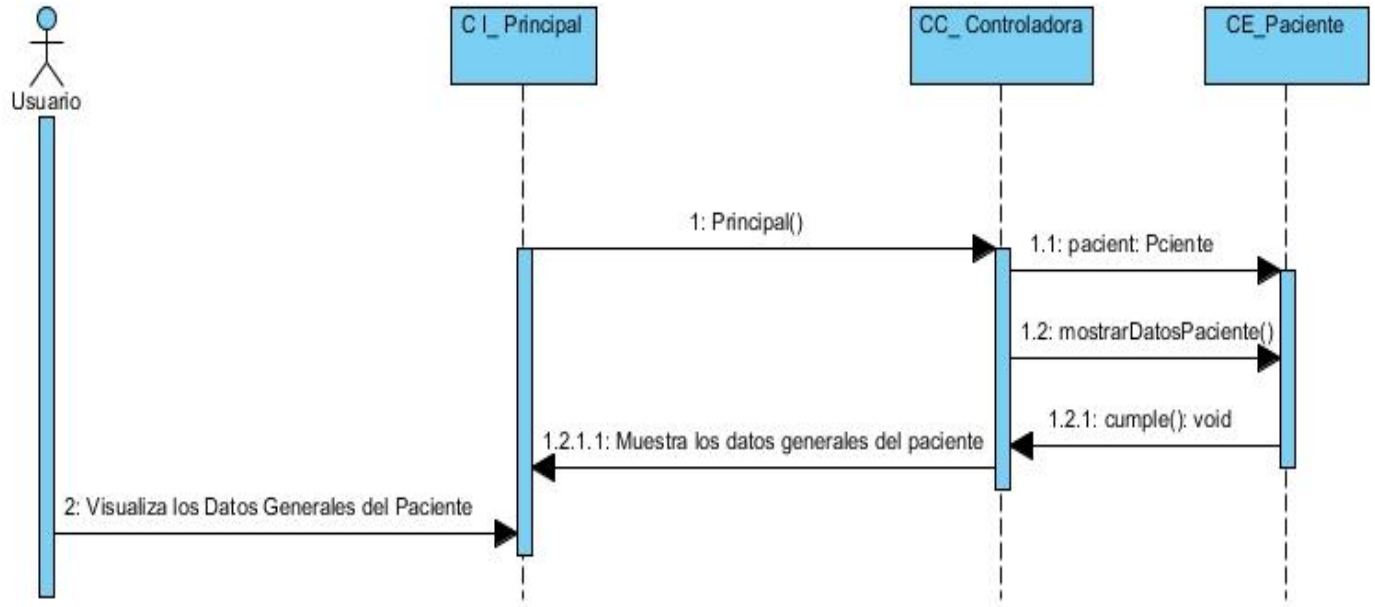


Diagrama de secuencia Listar documentos clínicos.

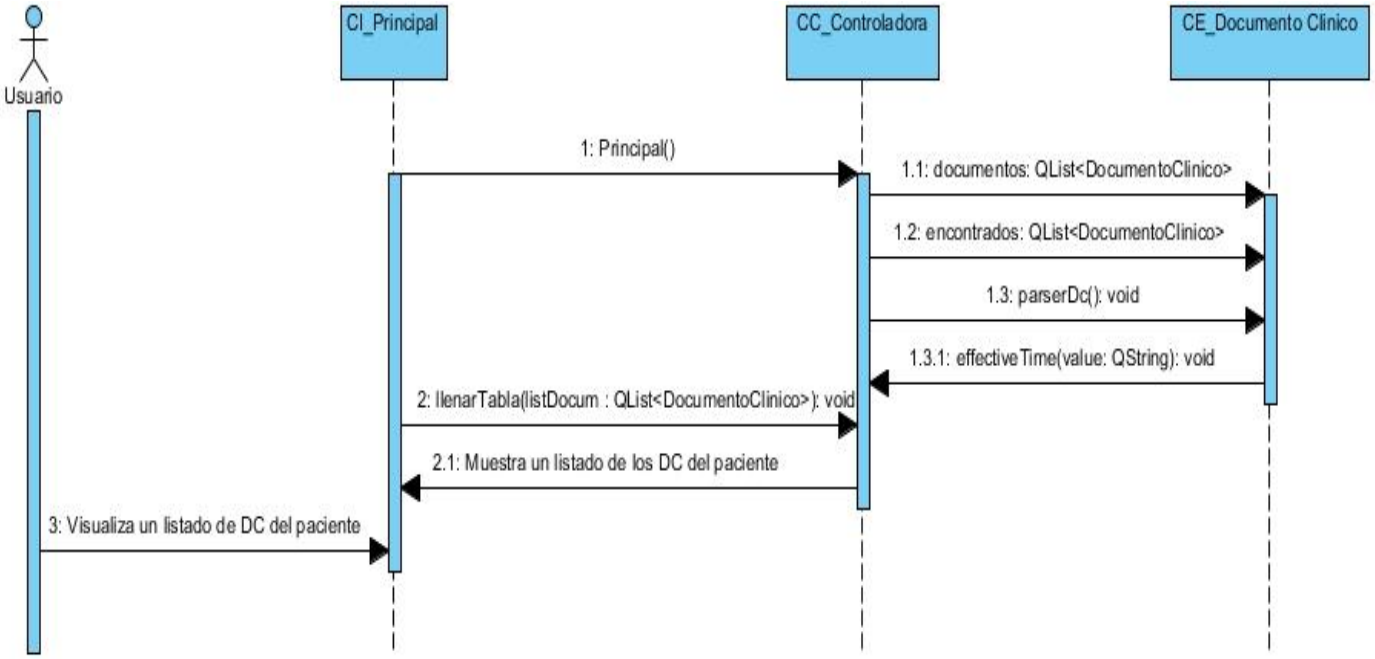


Diagrama de secuencia Mostrar documento clínico.

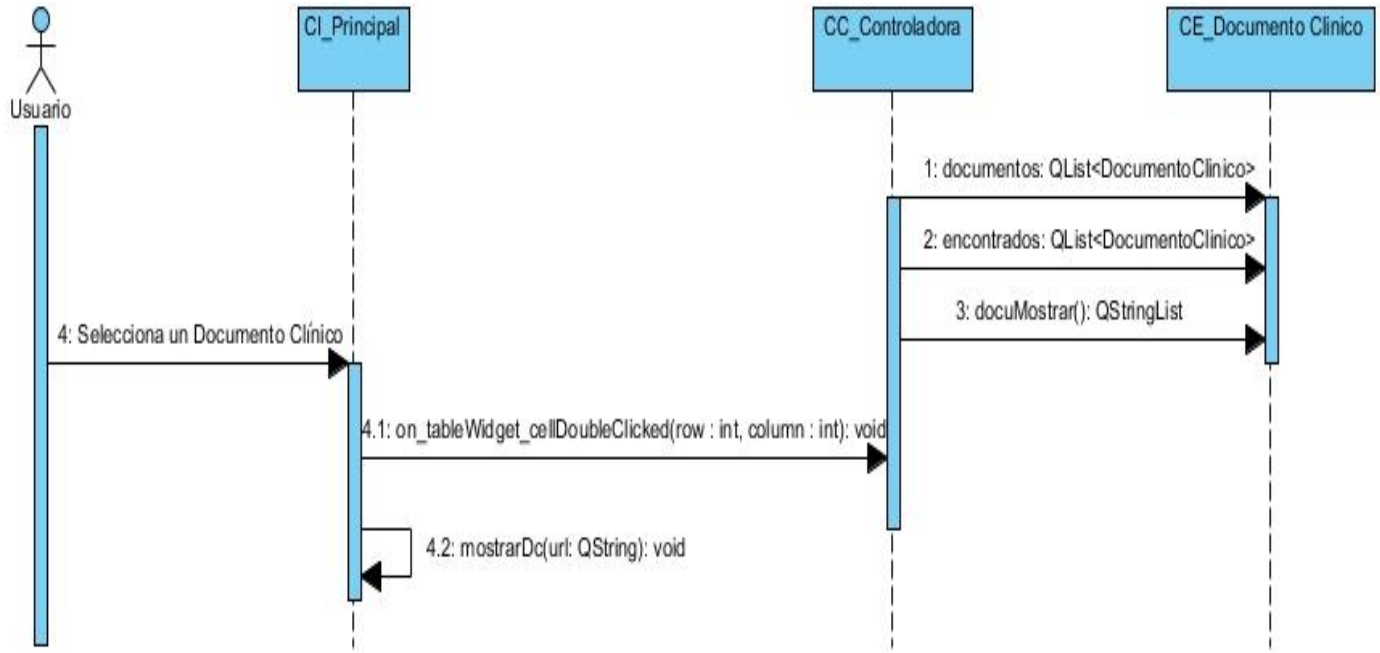


Diagrama de secuencia Mostrar documento clínico relacionado.

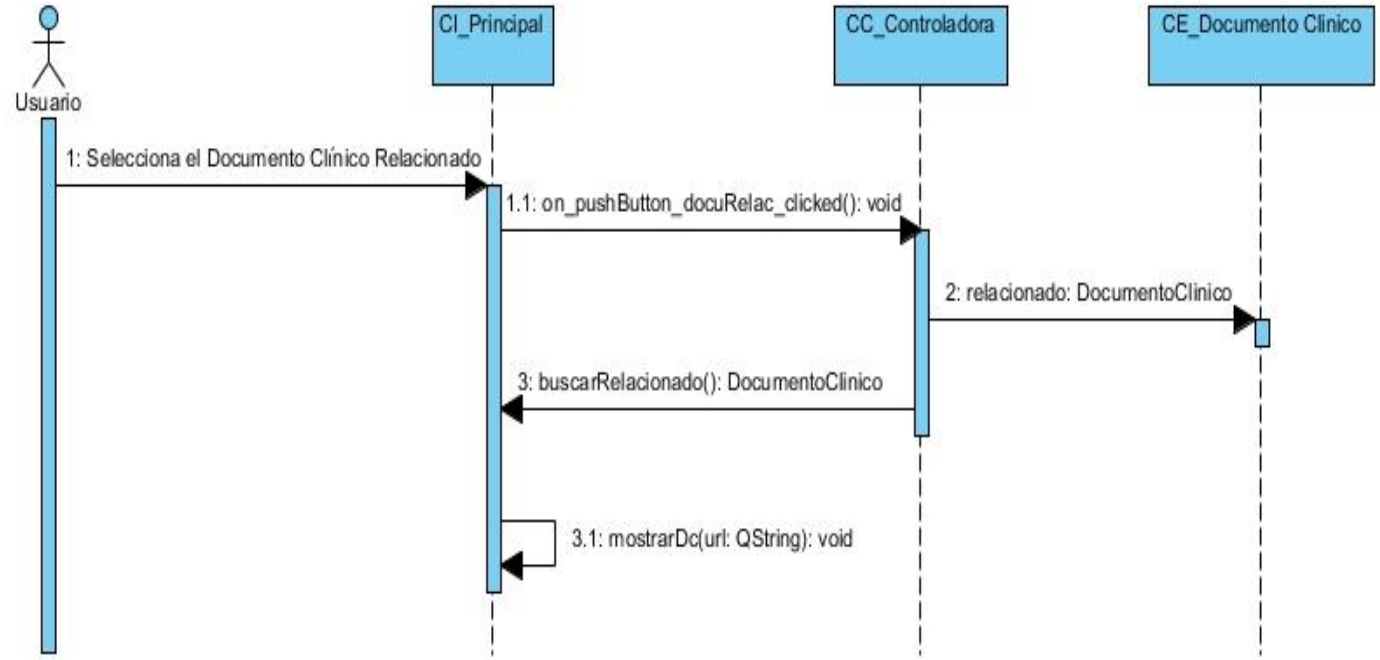


Diagrama de secuencia Mostrar menú de secciones del documento clínico.

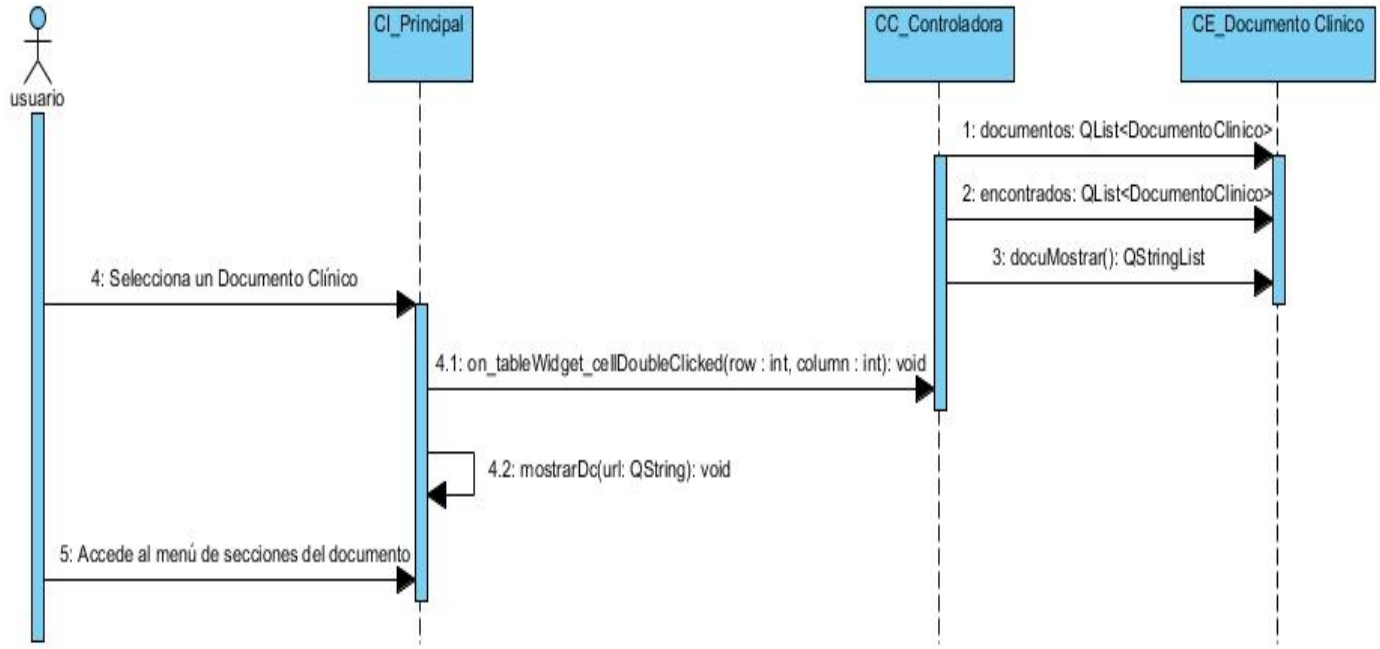
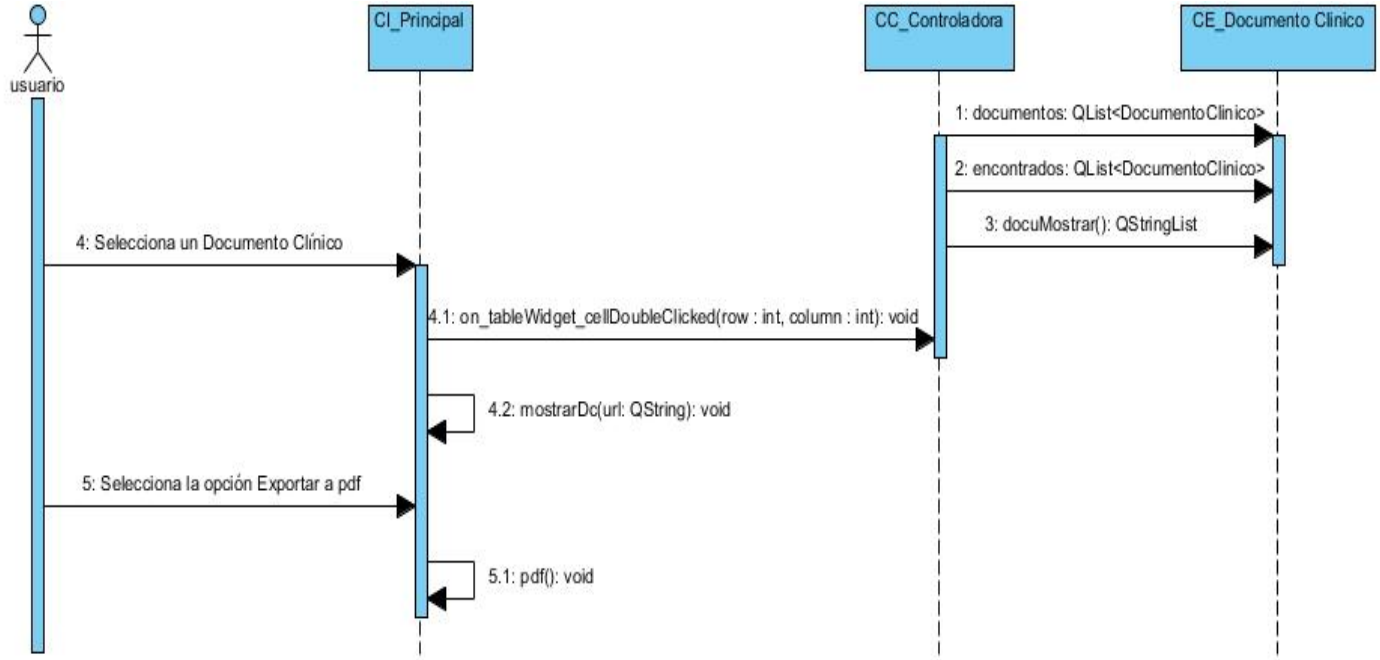


Diagrama de secuencia Exportar a PDF.



Glosario de Términos

Estándar: Es una especificación que regula la realización de ciertos procesos o la fabricación de componentes para garantizar la interoperabilidad de un sistema.

Health Level 7 (HL7): Es el estándar establecido para el intercambio, administración e integración de datos referentes a la atención sanitaria a pacientes así como a la prestación, gestión y evaluación de los servicios sanitarios.

Historia Clínica (HC): Es el único documento válido desde el punto de vista clínico y de ley. Además de los datos clínicos del paciente, su proceso evolutivo, tratamiento y recuperación, la Historia Clínica incluye juicios, documentos, procedimientos e informaciones, documentando fundamentalmente la relación médico-paciente.

Historia Clínica Electrónica (HCE): Es un archivo o conjunto de archivos que posibilita la creación y el almacenamiento digital de los episodios clínicos de un paciente a lo largo de todo el proceso asistencial.

XML (*Extensible Markup Language* o Lenguaje de Marcas Extensibles): Es un metalenguaje extensible de etiquetas que permite definir la gramática de lenguajes específicos. No constituye realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.

Interoperabilidad: Es la capacidad de compartir información, datos y servicios entre los distintos sistemas informáticos.

Tecnologías de la Información y Comunicaciones (TIC): Son las tecnologías que se necesitan para la gestión y transformación de la información, y muy en particular el uso de ordenadores y programas que permiten crear, modificar, almacenar, proteger y recuperar esa información.

Entorno de Desarrollo Integrado (IDE): Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica de usuario.

Framework: Plataforma, entorno, marco de trabajo, es una estructura conceptual y tecnológica de soporte definido, formada por componentes personalizables e intercambiables que componen un diseño reutilizable.

Glosario de Términos

Software: Son las instrucciones que el ordenador necesita para funcionar, no existen físicamente, o lo que es igual, no se pueden ver ni tocar.

Aplicación: Programa informático que cuenta con funcionalidades que permiten al usuario realizar uno o más tipos de tareas.

HIS (Sistema de Información Hospitalaria): Es un conjunto de sistemas de software que trabajan acoplados y permiten la informatización de los distintos servicios de las instituciones de salud.

Arquitectura: Conjunto de decisiones significativas acerca de la organización de un sistema de software. Se interesa no sólo por la estructura y el comportamiento, sino también por las restricciones y compromisos de uso, funcionalidad, funcionamiento, flexibilidad al cambio, reutilización, comprensión, economía y tecnología.

Patrones: Unidades de información nombrada, instructiva e intuitiva que captura la esencia de una familia exitosa de soluciones probadas a un problema recurrente dentro de un cierto contexto. Existen patrones de Diseño, Arquitectura, entre otros.