Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA

UNIVERSIDAD AUTÓNOMA METROPOLITANA – IZTAPALAPA

DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA

DESARROLLO DE UNA BIBLIOTECA DE SOFTWARE PARA ALMACENAMIENTO DICOM EN MEDIOS FÍSICOS

Tesis que presenta

Juan Salvador Salgado Blanco

Para obtener el grado de

Maestro en Ciencias en Ingeniería Biomédica

Asesor: M. en C. Alfonso Martínez Martínez

Co-Asesor: Dr. Humberto Cervantes Maceda

Jurado Calificador:

Presidente: Dr. Humberto Cervantes Maceda

Secretario: M. en C. Juan Ramón Jiménez Alanis

Vocal: M. en C. María Guadalupe Elena Ibargüengoitia González

México, D. F. Marzo 2011

El sabio es como lumbre o hacha grande,
y espejo reluciente y pulido de ambas partes,
y buen dechado de los otros, entendido y leído;
también es como un camino para los otros.

Pensamiento Náhuatl rescatado por Fray Bernardino de Sahún
Para el Dr. Juan Ignacio Salgado Gama, un hombre sabio

A mis Asesores, a mis Padres y a mis amadas Esposa e Hija.

RESUMEN

Este documento tiene como propósito dar a conocer el cómo y por qué se desarrolló una biblioteca para almacenamiento en medios físicos DICOM. Para comprender bien el entorno y justificación de este desarrollo, se presenta una introducción a las imágenes digitales, los sistemas para su administración y el estándar utilizado como base para este fin, llamado DICOM (Digital Imaging and Communication in Medicine). Así mismo, se realizó un análisis general de las partes del estándar involucradas en el manejo de imágenes relacionadas con el almacenamiento en medios físicos.

Para conocer con precisión los alcances de este proyecto se realizó una descripción general del estándar DICOM y muy detallada de sus partes relacionadas con el almacenamiento en medios físicos. También se incluyó un análisis del estado del arte para implementaciones DICOM con la finalidad de conocer otros desarrollos y sus alcances.

Para el desarrollo del proyecto se utilizó como marco de referencia un proceso de desarrollo llamado RUP (Rational Unified Process), el cual también se define con detalle. Como metodología específica se aplicó una instancia del proceso procurando cubrir los puntos más relevantes de implementación y administración de forma muy particular y a criterio propio. Para comprender mejor el RUP se realizó un estudio de sus procesos precursores y sus principales características, también se incluyó una justificación en el uso del RUP a nivel más profundo.

Dentro de la instancia del RUP obtenida se definieron sus componentes en dos partes. La primera describe los métodos, técnicas y herramientas que funcionaron como guía para obtención de los resultados correspondientes. La segunda parte, definió la guía administrativa para la planeación, manejo de recursos y el seguimiento del proceso así como la visión y el alcance del mismo, basados en el análisis previo del estándar DICOM para el almacenamiento en medios físicos y de las características requeridas para generar la librería que cumpliera las funciones específicas derivadas del estándar mismo.

Los resultados obtenidos fueron los documentos que sirvieron como guía para el desarrollo, los modelos que sirvieron para el diseño e implementación de la librería de software, la librería como tal y este documento de tesis. Para la presentación de resultados, se siguió la estructura de RUP que corresponde a las fases. Cada resultado que se documentó, cabe mencionar, es el producto de una serie de iteraciones que lo refinan hasta el punto en el cual se presenta. También se documentaron los resultados administrativos en cada una de las fases para guardar la relación con las tareas de planeación y seguimiento.

Luego de la presentación de los resultados se realizó un análisis para la discusión de los mismos con un enfoque principal en la importancia del producto final y de la utilización de procesos de desarrollo.

Posteriormente, se concluyó cuál es la utilidad del producto final, la finalidad en el uso de metodologías para proyectos en general y del impacto de este proyecto a nivel personal y académico. En seguida como parte del trabajo a realizar en el futuro se proponen nuevas líneas de investigación que tienen relación con esta biblioteca y en general con implementación de modelos y sistemas de administración de imágenes.

Se agregaron, además, tres apéndices al final del documento; A, B y C. En ellos se incluye en orden respectivo una guía para obtención de requerimientos siguiendo el estándar de la NASA, el documento Javadoc que sirve como guía para la utilización de la librería desarrollada y el glosario de términos importantes y sus definiciones.

1. Introducción

1.1 Imágenes Digitales

Las necesidades de almacenamiento y manipulación de imágenes médicas surgen a partir de los años 70's como consecuencia del nacimiento de la tomografía computarizada (*CT-Computed Tomography*) como método de diagnóstico basado en imágenes digitales. Desde entonces, se han desarrollado diferentes técnicas en la obtención de imágenes como la medicina nuclear (*NM-Nuclear Medicine*), la resonancia magnética (*MR-Magnetic Resonance*), la radiografía computarizada (*CR-Computed Radiography*) y la angiografía por substracción digital (*DSA-Digital Substraction Angiography*), entre otras. Estas técnicas han contribuido a la generación de diferentes tipos de imágenes médicas digitales para diagnóstico, junto con el consecuente incremento en la producción de las mismas [Martínez, 1999]. Así mismo, han aumentado los requerimientos tecnológicos para el manejo de las imágenes en formato digital.

Un ejemplo del crecimiento se puede ver a través del ejercicio realizado en el Instituto Nacional de Rehabilitación en donde, después de lograr la conversión de analógico a digital de sus equipos de adquisición de imágenes médicas, la demanda de espacio físico creció generando el volumen de información digital mostrada en la Figura 1.1, cuyo impacto se vio reflejado en la necesidad de buscar opciones de administración eficiente para el almacenamiento y manejo de dicha información [Martínez, 2005].

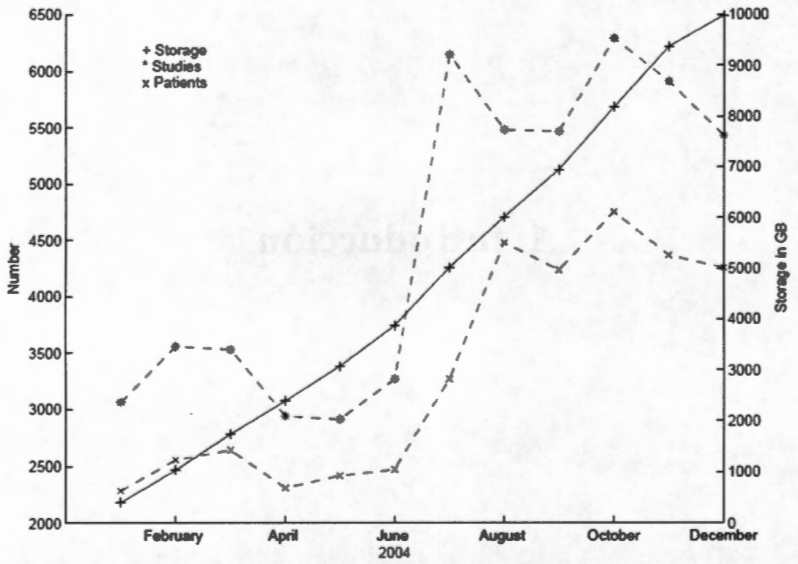

Figura 1.1 – Tendencia de producción considerando modalidades TAC, Rayos X, RM y MN [Martínez, 2005]

1.1.1 Manejo de Imágenes Digitales

La utilización de imágenes médicas digitales sin duda es de gran beneficio para el apoyo en el diagnóstico de los pacientes. Sin embargo, esta solución involucra necesidades o requerimientos para su manejo y administración de tipo tecnológico, ya que involucran desde implementación de estrategias, hasta la utilización de distintos dispositivos con el fin de hacer más eficiente este proceso.

En cualquier hospital en donde se utilicen imágenes médicas en formato digital, se requiere basar el proceso de diagnóstico en el cumplimiento eficiente de necesidades de producción, manejo y almacenamiento. La problemática entonces puede verse reflejada en dos puntos principales. En primer lugar, las necesidades de flujo de información y en segundo lugar las necesidades para almacenamiento, que son posibles de resolverse mediante la definición de un esquema de administración de imágenes adecuado.

1.1.2 Sistemas de Administración de Imágenes

La administración de las imágenes digitales se puede llevar a cabo de manera electrónica mediante la introducción de reglas y políticas para garantizar una mayor eficiencia. Esto es posible mediante herramientas de programación, modelado, estrategias de administración de procesos y metodologías de implementación que permiten generar políticas para la manipulación y almacenamiento de las imágenes médicas digitales, con el fin de administrar toda la información. Es importante no fundamentar esta administración en soluciones que cubran los requerimientos de informática pura ya que, es de importancia vital, tomar en cuenta las necesidades médicas, ingenieriles y de control con enfoque clínico.

Un PACS (siglas en inglés de Picture Archiving and Communication System) es un sistema que se encarga, como su nombre lo indica, de administrar las imágenes médicas almacenándolas y transfiriéndolas entre entidades que lo requieran.

Mediante la administración de imágenes médicas digitales se logra obtener la información de apoyo diagnóstico de manera adecuada en el sitio y momento precisos; en la Figura 1.2 se muestra un diagrama como ejemplo de esta funcionalidad.

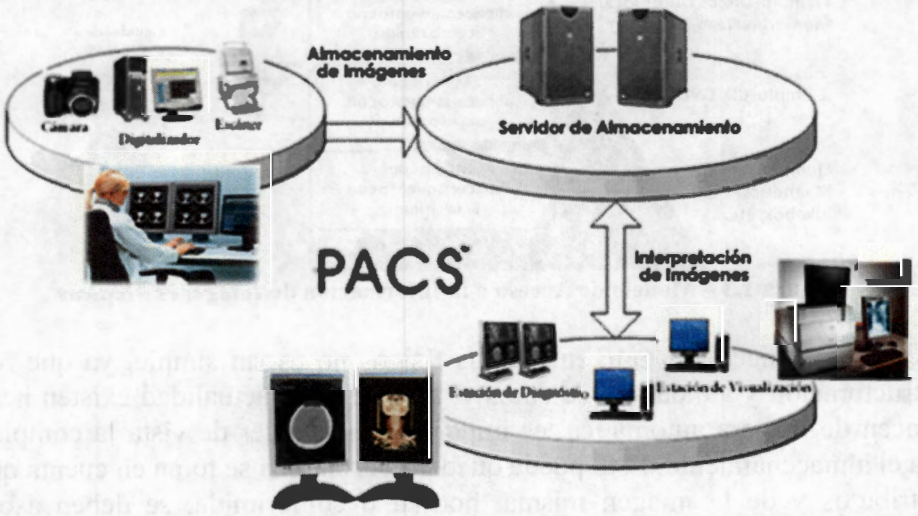

Figura 1.2 – Representación de un sistema para la administración de imágenes médicas (PACS)

La mayoría de hospitales con servicio de imagenología digital requieren tener una estrategia definida que resulte ágil para el acceso, manipulación y almacenamiento de imágenes para garantizar un procedimiento de diagnóstico y/o tratamiento de calidad y eficiente. Por ejemplo, es necesario tener a la mano las imágenes digitales de un paciente

que se encuentra hospitalizado con una fractura para que el médico tratante y los médicos relacionados realicenuna cirugía para inserción de clavos, tengan acceso inmediato a la información para generar un diagnóstico y/o tratamiento más adecuado. Otro ejemplo puede ser la necesidad de acceso a la información de un paciente que se acaba de dar de alta, en donde no es necesario tener los datos e imágenes disponibles de manera inmediata, pero sí de manera relativamente rápida durante un tiempo razonable suponiendo que el paciente puede regresar en cualquier momento. Y un último caso cuando el paciente no regresa con frecuencia, pero que de cualquier manera, se necesita tener la información agregada al expediente.

De aquí la gran importancia de la posibilidad de almacenar la información médica de manera organizada en distintos medios y así optimizar recursos físicos. Por ejemplo, almacenar las imágenes en DVD´s en un arreglo tipo JUKEBOX cada 6 meses, que son dispositivos de velocidad media de acceso y dejar espacio libre en el Disco Duro del servidor de bases de datos, que son dispositivos de acceso rápido para tener las imágenes actuales a la mano, o en cintas magnéticas que son de acceso lento para almacenarlas a largo plazo, o bien en un CD para entregar al paciente y evitar las impresiones con su costo asociado.

Una forma organizada de almacenar garantiza la disponibilidad de la información. Sin embargo, el constante aumento en el consumo del espacio físico en discos duros ha generado una necesidad más, que es la de crear un modelo de acceso a la información desde distintos medios de almacenamiento. En 1992 se propuso un modelo de almacenamiento en medios para imágenes médicas [Allen, 1992], en la Figura 1.3 se muestra dicho modelo en donde la pirámide representa la velocidad de transferencia con respecto al espacio físico utilizado. En la base (que es la parte más ancha) se tiene la mayor cantidad de información, pero la velocidad más lenta y en la punta (la parte más pequeña) se tiene la menor cantidad de información, pero la más alta velocidad de transferencia.

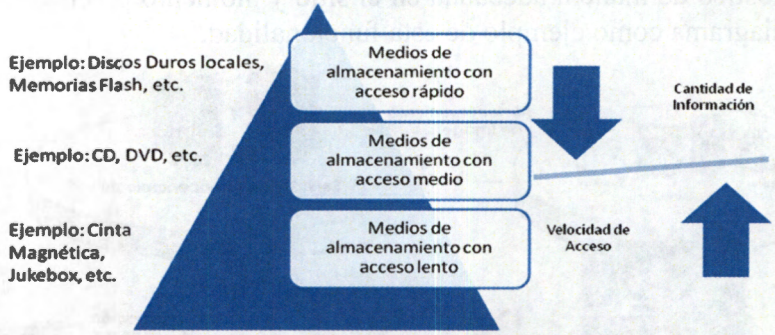

Figura 1.3 – Modelo de Acceso a la Información de Imágenes Médicas

Sin embargo, el almacenamiento en medios físicos no es tan simple, ya que requiere de cierta estructuración y validación de datos y, aunque en la actualidad existen herramientas que lo hacen de manera automática, es importante no perder de vista la complejidad que involucra el almacenamiento. Esto puede quedar más claro si se toma en cuenta que además de los atributos y de la imagen misma, normal o comprimida, se deben asociar datos demográficos y de identificación del paciente, información acerca de las condiciones de adquisición y, en algunos casos, información del examen, serie a la que pertenece la imagen

y orden que guarda en un estudio. La desventaja que se vivió en el inicio de la transformación a imágenes digitales era que los fabricantes de equipos que las generaban, tenían formatos propietarios incompatibles entre marcas e incluso entre modelos de la misma marca. Por estos problemas observados se concluyó, en ese momento, que toda esta información debía estar definitivamente definida bajo un estándar.

1.2 Antecedentes de la Estandarización

A través de los años se ha incrementado la necesidad de comunicar equipos heterogéneos entre sí con el fin de obtener un mejor diagnóstico a partir de la información que brindan los equipos que generan imágenes médicas. Esto hizo surgir la propuesta de crear una convención común de trabajo para todos los equipos que pudiera resultar en un formato común de imagen y diálogos normalizados entre equipos. El tratar de integrar todas estas necesidades de estandarización en un sistema de almacenamiento y transferencia es prácticamente imposible si no es a través de una normalización. Como consecuencia, surgió la necesidad de estandarizar el proceso de manipulación de imágenes médicas digitales y sus datos asociados. Este trabajo se inició en 1983 con la integración de un Comité formado por el Colegio Americano de Radiología (*ACR-American College of Radiology*), en representación de la comunidad de radiólogos y la Asociación Nacional de Manufactura Eléctrica (*NEMA-National Electrical Manufacturers Association*), que agrupa a la industria en el área de radiología, de acuerdo con los procedimientos establecidos por NEMA. Los objetivos iniciales fueron:

- Promover la comunicación entre equipos de imágenes digitales independientemente del fabricante que los produjo, considerando así el problema de compatibilidad.
- Ofrecer mayor flexibilidad a los sistemas de almacenamiento y transferencia de imágenes.
- Facilitar la creación y consulta de imágenes a través de diferentes dispositivos, en diversos lugares locales o remotos, con propósitos de visualización.

Después de tres años de esfuerzo, se dio a conocer la versión ACR/NEMA DICOM (*Digital Imaging and Communications in Medicine*) llamada también DICOM 3.0 [Nema, 2007] En esta versión participaron varias instituciones de la comunidad internacional como la Industria Japonesa de Aparatos de Radiología (JIRA-Japanese Industry Radiology Apparatus) y el Comité Europeo de Normalización (CEN-Comité Européen de Normalisation). Esta versión es considerada como un estándar completo y compatible porque supera las deficiencias de sus predecesores y acepta las especificaciones de las versiones anteriores (ACR-NEMA 1 y 2). En la actualidad el estándar DICOM se compone de 18 partes que definen su estructura y reglas de transferencia, sintaxis y almacenamiento.

1.3 El estándar DICOM

DICOM es el estándar reconocido mundialmente en la actualidad para el intercambio de imágenes médicas, pensado para el manejo, almacenamiento, impresión y transmisión de las mismas. Incluye la definición de formato de imágenes, datos asociados, modelo de información, diccionario de datos y protocolos de comunicación. Los archivos DICOM pueden intercambiarse entre dos entidades que tengan capacidad de recibir imágenes y

datos de pacientes en formato DICOM [Wikipedia.org - DICOM, 2006]. A las entidades que realizan el intercambio de archivos y servicios DICOM se les llama Entidades de Aplicación, por lo que a partir de este momento nos referiremos así a ellas.

DICOM permite la integración de escáneres, servidores de imágenes digitales, estaciones de visualización, impresoras y hardware de red de múltiples proveedores dentro de un sistema de almacenamiento y comunicación de imágenes. Debido a este alcance, un PACS debe ser capaz de funcionar bajo las reglas del estándar DICOM.

Las diferentes máquinas de procesamiento de datos, servidores de imágenes y estaciones de visualización tienen una declaración de conformidad DICOM (conformance statements) que establece claramente los servicios DICOM que soportan. La declaración de conformidad define la función que corresponde a cada entidad de aplicación, pudiendo ser dicha entidad un Tomógrafo, un servidor o una estación de trabajo por ejemplo. Existen dos tipos de roles que pueden adoptar las entidades de aplicación independientemente de su función específica, la de Proveedor de Clases de Servicio o SCP (Service Class Provider) o la de Usuario de Clases de Servicio SCU (Service Class User). Las clases de servicios DICOM son un conjunto de clases que generan o brindan servicios clasificados por utilidad o función. Estos dos tipos de roles establecen el esquema de utilización de las clases de servicio DICOM, ya que definen cuáles soportan, cuáles pueden usar y cuáles pueden ofrecer como servidor o proveedor y a esta combinación de clases de servicio soportadas se le llama Perfil de Aplicación.

Toda la información DICOM está compuesta por SOP´s (Service Object Pair). Estas SOP´s, forman parte de los servicios DICOM, y a través de sus IOD´s (Information Object Definition), describen la estructura, función y atributos que deben tener. Cabe mencionar que existen dos tipos de SOP´s, los compuestos y los normalizados, en donde la diferencia entre ambos es que el segundo normaliza la información para evitar redundancia tal y como se hace en las bases de datos relacionales. En este proyecto, se trabajará con SOP´s compuestos.

A continuación se describen los tipos de clase de servicio y la descripción básica de sus funciones:

- **Store.** El servicio DICOM Store es usado para mandar imágenes u otros objetos persistentes (informes estructurados, etc.) a una estación de trabajo.
- **Storage Commitment.** El servicio DICOM storage commitment es usado para confirmar que una imagen ha sido almacenada permanentemente por un dispositivo. El SCU (modalidad, estación de trabajo, etc.) utiliza la confirmación del SCP (estación de almacenamiento) para asegurarse de que puede borrar la imagen localmente.
- **Query/Retrieve.** Permite a una estación de trabajo hacer búsquedas de imágenes y recuperarlas.
- **Modality Worklist.** Permite a una parte del equipo de imagen (una modalidad) obtener detalles de los pacientes y la planificación de exámenes médicos

electrónicamente, evitando la necesidad de introducir la misma información varias veces (previniendo así los errores que pueden ocurrir al volver a introducirlos).

- **Modality Performed Procedure Step.** Un servicio complementario al Modality Worklist, que permite a la modalidad mandar un informe sobre los exámenes médicos realizados incluyendo datos sobre las imágenes adquiridas, las dosis dispensadas, etc.
- **Printing.** Este servicio es usado para mandar imágenes a una impresora DICOM.

Dentro de las clases de servicio que es necesario conocer, dados los objetivos de este proyecto, se encuentran las que tienen relación con el almacenamiento, particularmente la clase de servicios STORE conformado, a su vez, por las opciones de almacenamiento en medios físicos File-Set-Reader, File-Set-Creator y File-Set-Updater, las cuales permiten leer, crear y actualizar archivos respectivamente.

1.4 Alcance de DICOM en especificaciones de almacenamiento

Para seguir la idea planteada en el modelo de la Figura 1.3, es necesario tener un sistema compatible con múltiples dispositivos, independientemente de la velocidad de acceso y capacidad de almacenamiento, que pueden ser CD´s, DVD´s, discos duros, cintas magnéticas, etc. El estándar DICOM prevé estas necesidades, proponiendo algunos modelos que permiten conectar dispositivos de almacenamiento sin importar su formato, capacidad y/o velocidad de acceso. En las reglas de almacenamiento DICOM se involucran varias partes del estándar, las cuales se pueden ver en los cuadros remarcados en la Figura 1.4. La parte que describe el formato de archivo y reglas de transferencia para el almacenamiento es la 10 del estándar, que conforme se adentra en el detalle de perfiles de aplicación, formatos de medios y perfiles de seguridad, hace referencia a las partes 11,12 y 15 respectivamente.

El estándar cubre todas las necesidades de almacenamiento y lo importante e interesante es que está diseñado para permitir la incorporación de dispositivos que utilicen nuevas tecnologías. No importa realmente si se está almacenando una imagen de TAC (Tomografía Axial Computarizada) o un objeto recientemente definido como una imagen dental, hemodinámica o endoscópica, se puede utilizar el mismo servicio. Se puede comparar esto con un taller que hace muebles, donde los objetos de información serían las piezas del mobiliario y las herramientas (martillo, sierra, cincel, desarmador, etc.) fueran los servicios DICOM. No importa realmente con qué muebles o tipos de madera se esté trabajando, en todos los casos se usan las mismas herramientas.

1.4.1 Partes DICOM involucradas

El almacenamiento en medios y el formato de archivo DICOM corresponde a la parte 10 del estándar DICOM, resaltado en la Figura 1.4, que muestra la interacción entre las partes del estándar. La parte 10 describe cómo almacenar información de imágenes médicas en un medio extraíble adoptando el perfil de aplicación adecuado. Los archivos son organizados bajo una estructura jerárquica y la información de esta se almacena en un archivo llamado DICOMDIR.

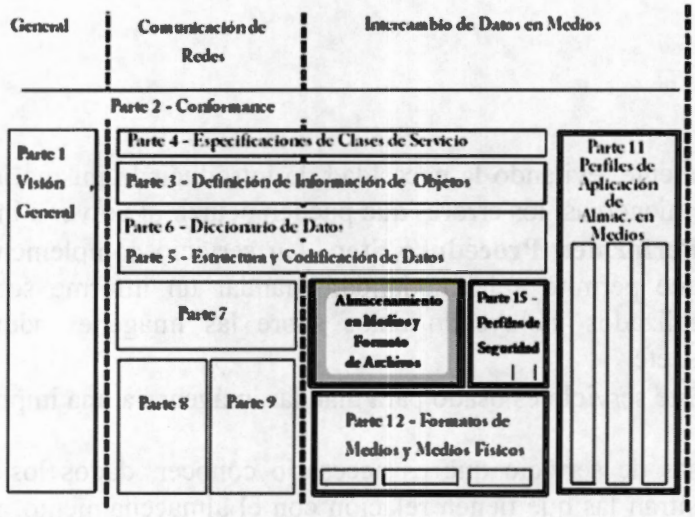

Figura 1.4 - Relación de las partes DICOM, el cuadro con borde gris corresponde a la parte 10 [Nema, 2007]

El modelo de medios de almacenamiento se encuentra integrado en el esquema de capas mostrado en la Figura 1.5. Este modelo se enfoca a aspectos directamente relacionados con el intercambio de datos en medios de almacenamiento y debe estar implementado conforme a las estructuras de datos y a las reglas establecidas en las diferentes capas para lograr interoperabilidad al intercambiar medios físicos de almacenamiento. Los servicios identificados en este modelo son simples límites entre capas funcionales; el modelo de capas general será descrito con detalle en el siguiente capítulo, por ahora la intención es simplemente ubicar la capa correspondiente al almacenamiento en medios físicos.

Figura 1.5 – Diagrama General de Capas DICOM [Nema, 2007]

La Figura 1.5 presenta el modelo general el cual abarca la comunicación en red y el intercambio en medios de almacenamiento. Las entidades de aplicación DICOM, ubicadas en la capa de aplicación de imágenes médicas, pueden transferir información a través de cualquiera de las siguientes vías:

- La capa superior de servicio, la cual provee independencia al soporte de comunicación en red.
- El servicio de archivos DICOM, el cual provee acceso al medio de almacenamiento independientemente de los formatos específicos de medios físicos de almacenamiento y estructuras de archivos.

En caso de querer realizar transferencia de información DICOM a través de la red, la aplicación define la comunicación entre entidades, luego se podrá seleccionar el tipo de almacenamiento a realizarse en medios físicos. Estos servicios y la configuración de archivos relacionada se pueden escoger de manera adecuada mediante las bibliotecas disponibles para el manejo de imágenes y archivos DICOM.

1.4.2 Requerimientos de otras partes de estándar

Este proyecto es parte de un proyecto de amplio alcance que ha considerado la aplicación de procesos de desarrollo de software. El objetivo es sentar las bases para la generación de PACS a partir de bibliotecas que pueden ser reutilizadas según sean las necesidades que se desee cubrir.

Los subsistemas considerados en desarrollo se muestran en la Figura 1.4 y, al igual que el correspondiente al almacenamiento DICOM en medios físicos. En general estas bibliotecas tienen las características para proveer:

- Modelos bien definidos cuya funcionalidad es especificada por las partes de DICOM
- Una manera de manejar reglas de codificación de la información involucrada en objetos compuestos DICOM.
- Independencia con respecto al hardware de los controladores y/o plataformas del sistema operativo (funciones de red, acceso a archivos, acceso a medios, etc.).

Como se puede ver en la Figura 1.4, las partes 3, 4, 5 y 6 del estándar también están involucradas en el almacenamiento en medios físicos en forma indirecta. De esta forma, este proyecto será un complemento de los desarrollos de las partes antes mencionadas.

1.4.3 Propuesta del proyecto utilizando ingeniería de software

Según la definición del IEEE, [Lewis, 1994] "software es la suma total de los programas de computadora, procedimientos, reglas, la documentación asociada y los datos que pertenecen a un sistema de cómputo". Según el mismo autor, "un producto de software es un producto diseñado para un usuario". En este contexto, la Ingeniería de Software (SE del inglés *Software Engineering*) es un "enfoque sistemático del desarrollo, operación, mantenimiento y retiro del software" y, por esta razón, se considera que "la Ingeniería de Software es la rama de la ingeniería que aplica los principios de la ciencia de la computación y las matemáticas para lograr soluciones adecuadas en costo a los problemas

de desarrollo de software", es decir, permite elaborar consistentemente productos correctos, utilizables y dentro del presupuesto establecido [Cota, 1994].

Los procesos en ingeniería de software se definen como un conjunto de etapas parcialmente ordenadas para el logro de ciertos objetivos, en este caso, la obtención de un producto de software de calidad [Jacobson, 1998]. El proceso de desarrollo y mantenimiento de software es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código; el código es probado, documentado y certificado para su uso operativo. Concretamente, define quién debe hacer qué, cuándo debe hacerlo y cómo debe alcanzar un cierto objetivo [Jacobson, 1998].

Para la administración e implementación de este proyecto de desarrollo de software se empleó el Proceso Unificado de Rational (RUP, por sus siglas en inglés) el cual permite tener control del proyecto para propiciar el cumplimiento de objetivos, además de herramientas, modelos y estrategias para garantizar la producción de una aplicación que cubra los requerimientos y que sea de alta calidad. RUP captura muchas de las mejores prácticas en desarrollo de software moderno en una forma que es aplicable a un gran espectro de proyectos y organizaciones. Considera técnicas orientadas a objetos y usa UML (Unified Modeling Language) como la principal notación para los modelos que son construidos durante el desarrollo. También es considerado como un marco de referencia para procesos abiertos que permite a organizaciones que producen software adaptar el proceso a sus necesidades específicas precisas y capturar su proceso específico en componentes. De la misma manera provee una aproximación disciplinada para asignar trabajos y responsabilidades junto con una organización de diseño. La ventaja de utilizar un proceso de desarrollo de software es asegurar la producción de software de alta calidad que conozca las necesidades del usuario final, con una programación predecible y un objetivo claro. En la actualidad existen diferentes filosofías para tratar de hacer más eficientes los procesos de desarrollo, con diferencias en orientación al trabajo individual y trabajo colectivo o basando su garantía de éxito en distintas metodologías.

El proceso de desarrollo y mantenimiento de software requiere de un conjunto de conceptos, una metodología y un lenguaje propio. Estos procesos están estrechamente relacionados con el ciclo de vida del software. De acuerdo al RUP, el ciclo de vida de software comprende cuatro grandes fases: concepción, elaboración, construcción y transición. La concepción define el alcance del proyecto y desarrolla un caso de negocio. La elaboración define un plan del proyecto, especifica las características y fundamenta la arquitectura. La construcción crea el producto y la transición transfiere el producto a los usuarios. La razón principal para seleccionar específicamente el RUP es que es un proceso bastante maduro que desciende del Proceso Unificado [Wikipedia.org - RUP, 2006], con el cual se había estado trabajando y del cual se tiene cierto dominio, por lo que el cambio de una metodología a otra, considerando la administración del proyecto, no representó riesgos significativos al momento de implementación y ejecución del mismo.

1.5 Objetivos del Proyecto

Se definen como objetivos las metas o alcances que se desean cubrir en este proyecto, siendo el alcance principal el cumplimiento del objetivo general.

1.5.1 Objetivo general

El propósito de la parte 10 del estándar DICOM es proveer un marco de referencia para sistemas que permita el intercambio de varios tipos de imágenes médicas e información relacionada en un amplio rango de medios físicos de almacenamiento.

El objetivo principal de este proyecto es implementar las especificaciones de la parte 10 del estándar DICOM, buscando cumplir los siguientes puntos como solución:

1. Un modelo de capas para el almacenamiento de imágenes médicas e información relacionada en medios de almacenamiento.
2. Un formato de archivo DICOM adecuado.
3. Un formato de archivo DICOM confiable, que ofrezca la posibilidad de incorporar seguridad a la información.
4. Un servicio de almacenamiento DICOM que provea independencia del formato del medio y del medio físico.

Al final de este proyecto se generará una biblioteca que cubra al 100% la parte 10 (versión 2004) y sus partes asociadas con las siguientes ventajas:

- Que permita el libre intercambio de medios con una arquitectura de software extensible
- Que dé una solución para soportar el almacenamiento de archivos DICOM de acuerdo con el formato específico de cada medio.
- Que ofrezca facilidades para agregar nuevos dispositivos de almacenamiento.

Además de las especificaciones de la parte 10, el producto a desarrollar debe cubrir requerimientos no funcionales que incluyen extensibilidad, portabilidad y mantenibilidad.

1.5.2 Objetivos específicos

Los objetivos específicos del proyecto incluyen:

- Soporte para Escritura, Lectura y Actualización de grupos de archivos en formato DICOM en plataforma Windows en los siguientes dispositivos:
  - CD
  - DVD
  - USB
  - HD
- Soporte para Escritura, Lectura y Actualización de grupos de archivos en formato DICOM en plataforma Linux en los siguientes dispositivos:
  - CD
  - DVD
  - USB

- HD
- Documentación para apoyo de los usuarios de la biblioteca, quienes la utilizarán para desarrollar aplicaciones compatibles con el formato de archivo DICOM.

1.5.3 Utilización de bibliotecas DICOM

El beneficio que se puede lograr con el cumplimento de los objetivos de este proyecto es que cualquier desarrollador que desee implementar una aplicación DICOM, podrá solucionar la parte de almacenamiento en medios físicos utilizando esta biblioteca de software. Cualquier sistema que utilice esta biblioteca dentro de un conjunto de servicios DICOM tendrá un modo sencillo de operar para el almacenamiento en medios físicos DICOM, en donde sus entidades de aplicación simplemente solicitarán el servicio y la biblioteca misma ofrecerá una solución transparente. La ventaja de desarrollar un conjunto de bibliotecas es dar una mejor funcionalidad a la entidad de aplicación que desee ejecutar algún servicio DICOM dependiendo del rol adoptado (SCU o SCP). En la Figura 1.5 podemos observar la colaboración que existe entre la entidad de aplicación (Aplicación de Imágenes Médicas) y los servicios DICOM.

1.6 Estructura del Documento

En este capítulo se introdujeron los conceptos y definición del estándar DICOM, lo que es un PACS y cuáles son sus funciones y límites. Revisamos también los objetivos y estrategia de este proyecto, así como la ubicación de sus fronteras, para ofrecer una descripción general del desarrollo del proyecto "Desarrollo del Sistema de Almacenamiento DICOM en Medios Físicos" usando un proceso de desarrollo bien definido.
En el capítulo dos se define el estado del arte del estándar DICOM así como las implementaciones existentes y un análisis comparativo breve entre ellos. En el capítulo tres se describe la metodología de desarrollo que fue empleada y la justificación de su uso, así como las herramientas, tareas, funciones y roles que se derivaron de la instancia del proceso (RUP) utilizada. En el capítulo cuatro se lleva a un nivel mayor de detalle los resultados de la propuesta de solución, la visión y los alcances del proyecto. En el capítulo cinco se mostrarán los resultados que fueron generados desde el inicio o concepción según la metodología empleada hasta la finalización o transición según la misma metodología. Por último, el capítulo seis se integra la discusión crítica y las conclusiones correspondientes a la realización del proyecto.

2. El Estándar DICOM

2.1 Introducción

Como se menciona en el capítulo anterior, el estándar DICOM puede ser conceptualizado a través de un modelo de capas (Figura 1.5) en donde cada capa representa una sección importante de las especificaciones que puede, a su vez, involucrar una o más partes [Nema, 2007]. En la Figura 2.1 se muestra una vista detallada de este modelo que sirve como guía para identificar funciones, sus fronteras y las relaciones entre capas.

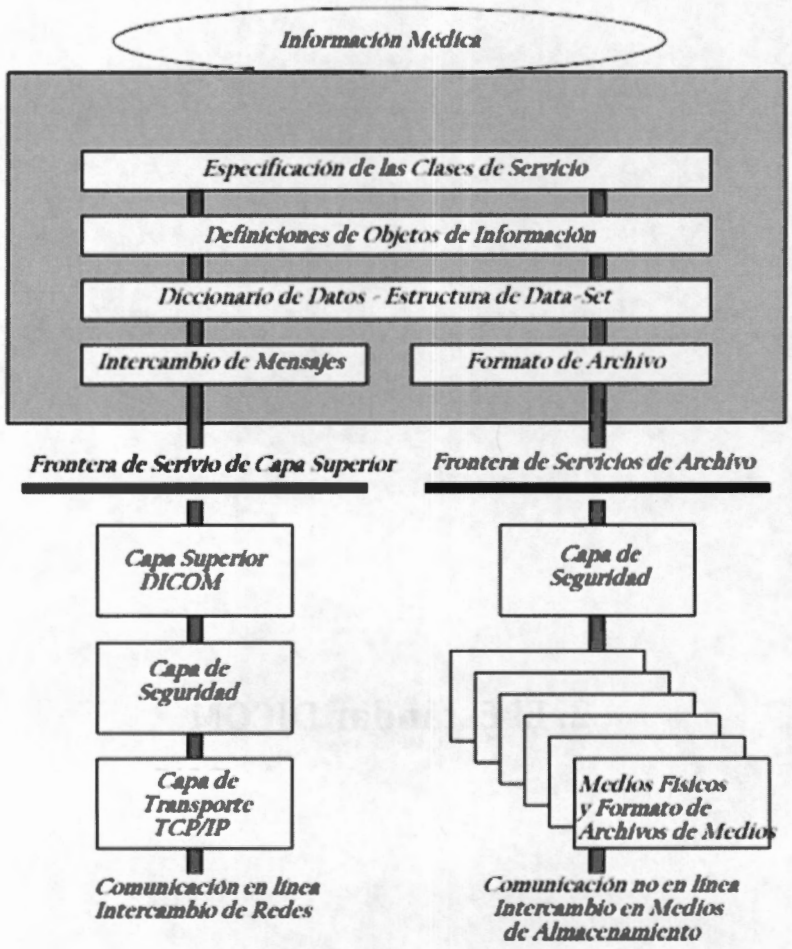

Figura 2.1 – **Modelo de Capas DICOM [Nema, 2007]**

2.1.1 Conformación de aplicaciones en base al estándar

El estándar DICOM incluye la definición de formatos de archivos, protocolos de comunicación, estructuras de datos, diccionario de datos, entidades de aplicación y formatos de medios físicos. Bajo estas especificaciones, la información DICOM puede intercambiarse entre dos o más entidades que tengan capacidad de enviar y recibir imágenes y datos de pacientes.

El estándar se encuentra dividido en 18 partes que especifican con gran detalle cada regla de estructura, actividades, roles y reglas de transferencia de información entre entidades de aplicación. Las partes 9 y 13 fueron eliminadas debido a la incapacidad de adaptarse al

avance tecnológico, es decir, ya no se pueden aplicar a la tecnología actual. El estándar DICOM se organiza de la siguiente manera con su contenido correspondiente:

- Parte 1 - Introducción y visión general
- Parte 2 - Conformación del estándar (Conformance)
- Parte 3 - Definición de objetos de información (IOD´s)
- Parte 4 - Especificación de clases de servicios
- Parte 5 - Estructura de datos y codificación
- Parte 6 - Diccionario de datos
- Parte 7 - Intercambio de mensajes
- Parte 8 - Soporte de comunicación de redes para intercambio de mensajes
- Parte 9 - Retirada
- Parte 10 - Almacenamiento en medios y formato de archivos para intercambio de datos
- Parte 11 - Perfiles de aplicación para almacenamiento en medios
- Parte 12 - Formato de medios y medios físicos para intercambio de datos
- Parte 13 - Retirada
- Parte 14 - Función de despliegue estándar en niveles de gris
- Parte 15 - Perfiles de seguridad
- Parte 16 - Recursos para mapeo de contenido
- Parte 17 - Información explicativa
- Parte 18 - Acceso Web a objetos persistentes DICOM (WADO)

Estas partes representan todas las reglas para implementación y uso del estándar DICOM con el fin de hacerlo independiente de cualquier dispositivo u herramienta, sin importar la marca, modelo, tipo, etc.

Figura 2.1 presenta el modelo general por capas del estándar el cual abarca tanto la parte de redes (en línea) como la parte de intercambio y almacenamiento en medios (fuera de línea). Las aplicaciones deben utilizar como vía de comunicación cualquiera de las siguientes fronteras:

- Frontera de servicios de capa superior (Upper Layer Service), la cual provee independencia del soporte de comunicaciones de red y protocolos, tales como TCP/IP.
- Frontera de servicios de archivo, la cual provee acceso al almacenamiento, independientemente del medio físico y de estructuras de archivos específicos.

Las 18 partes definen el estándar por completo, sin embargo, no todas las entidades de aplicación implementan la totalidad de los servicios DICOM, dado que no los requieren, puesto que tienen una función específica (impresión, almacenamiento, transferencia, etc.) y no todas las entidades de aplicación tendrán interacción con todas las modalidades de imágenes médicas. El conjunto de servicios que implementa cada entidad de aplicación, dependiendo de su función específica, se denomina "conformación DICOM" (DICOM Conformance). El nivel de interoperabilidad entre entidades de aplicación se rige por su conformación DICOM específica [Nema, 2007]:

- Las clases de servicio
- Los objetos de información DICOM
- Protocolos de comunicación
- Perfiles de aplicación para almacenamiento en medios

Como se mencionó en la sección 1.3, los perfiles de aplicación definen las clases de servicios soportadas por una entidad de aplicación en roles SCU o SCP. La Figura 2.2 muestra la relación entre especificaciones usadas en un perfil de aplicación, específico para almacenamiento en medios físicos, y sus partes DICOM. Una entidad de aplicación implementa ciertos servicios DICOM de acuerdo con su función específica, esta implementación puede ser vista como un conjunto que integra las especificaciones de varias partes del estándar, con los requerimientos necesarios. Cabe aclarar que un perfil de aplicación también se puede establecer considerando los servicios de red.

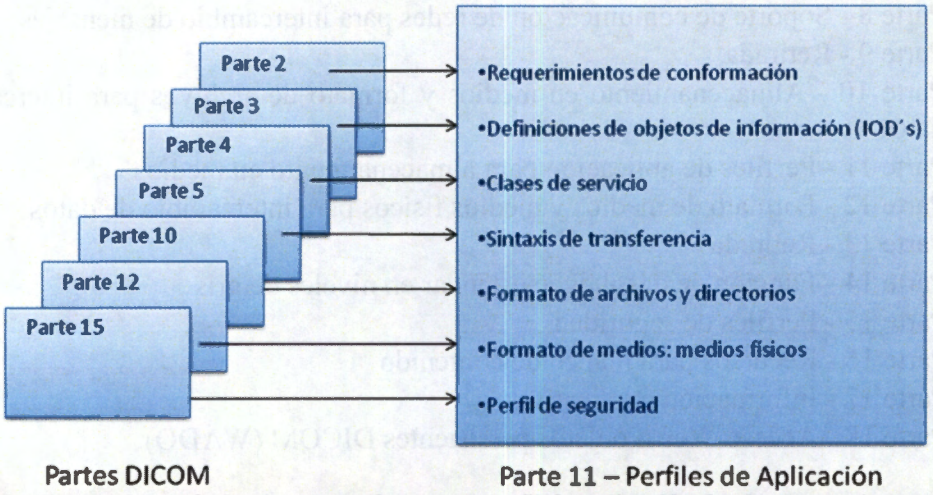

Figura 2.2 – Partes del estándar involucradas en un perfil de aplicación de almacenamiento en medios físicos [Nema, 2007]

Un perfil de aplicación para almacenamiento en medios físicos se organiza en las siguientes partes:

1. El nombre del perfil de aplicación, o la lista de perfiles de aplicación agrupados en la clase de servicio relacionada.
2. La descripción del contexto clínico del perfil de aplicación.
3. La definición de la clase de servicio del medio de almacenamiento con los roles del dispositivo del perfil de aplicación y sus opciones asociadas.
4. La sección informativa que describe los requerimientos operacionales del perfil de aplicación
5. La especificación de las clases SOP y sus IOD´s relacionados soportados y la sintaxis de transferencia que se va a usar, considerando la sintaxis de transferencia como el conjunto de reglas para representar, sin ambigüedades, los elementos de datos a nivel de la capa de aplicación que son independientes de las técnicas de codificación.
6. La selección del formato de medios físicos que se va a usar.

7. Si el módulo de información del directorio (representado a través del DICOMDIR) es utilizado, se requiere de la descripción mínima de un sub-conjunto de las especificaciones del modelo.
8. Otros parámetros que se necesitan especificar para asegurar interoperabilidad en intercambio de medios.
9. Parámetros de seguridad que seleccionan las técnicas criptográficas que se usarán para la seguridad de los perfiles de aplicación de medios de almacenamiento.

En la Figura 2.3 se muestra la relación que existe entre la conformación DICOM, el perfil de aplicación adecuado y las partes DICOM involucradas. Obsérvese que para construir un perfil de aplicación para almacenamiento en medios físicos, se requiere de partes básicas, utilizadas también en servicios de red, como la 3, 4, 5 y 6, que especifican IOD´s, SOP´s, estructuras de datos y diccionario de datos respectivamente.

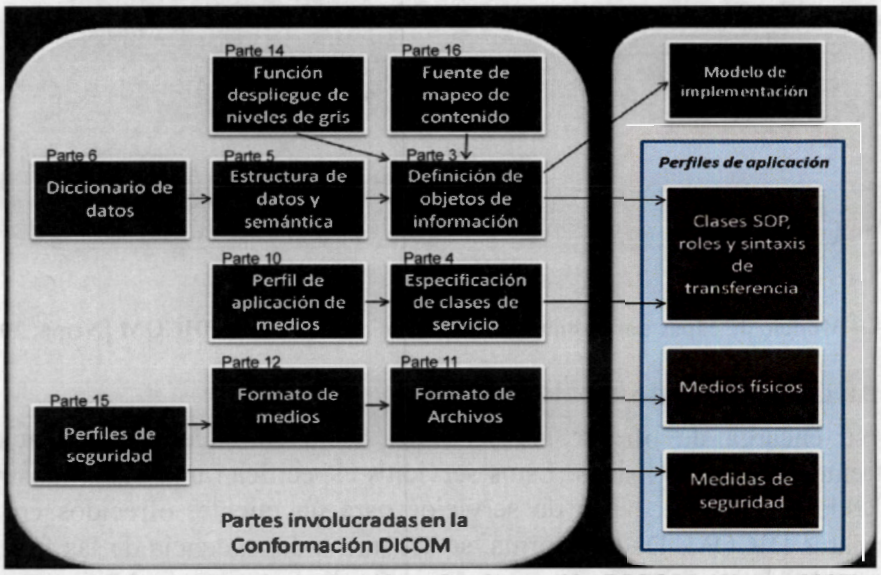

Figura 2.3 – Perfiles de Aplicación y Conformación DICOM [Nema, 2007]

2.1.2 Especificaciones DICOM para el almacenamiento en medios físicos

Con la finalidad de establecer la forma en que una entidad de aplicación pueda utilizar los servicios de almacenamiento, se describe a continuación un modelo de capas de abstracción para almacenamiento en medios físicos:

1) Capa de formato de archivo DICOM - Conformación de la estructura del archivo DICOM
2) Capa de formato de medios - Mapeo de archivos DICOM al formato específico del medio
3) Capa de medios físicos - Definición del medio físico (tipo de partición y espacio físico).

Estas capas se ilustran en la Figura 2.4 y como se puede observar en ella, se encuentran separadas por fronteras que definen interfaces para poder comunicarse entre sí a través de

servicios específicos. Las funciones de dichas capas se describen con detalle a continuación.

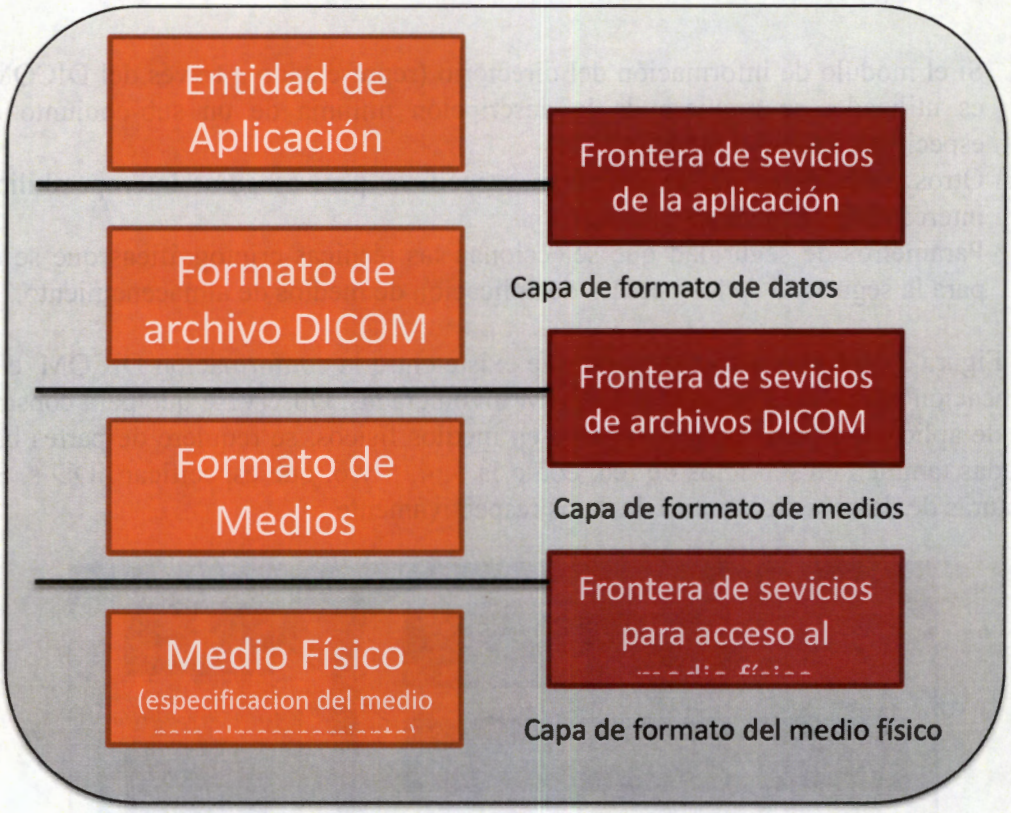

Figura 2.4 – Modelo de capas para almacenamiento en medios físicos DICOM [Nema, 2007, pt. 10]

2.1.2.1 Capa de formato de archivos DICOM

Esta capa se encarga de ofrecer los servicios hacia entidades de aplicación para el almacenamiento de medios físicos. Estos servicios especifican una visión abstracta de los archivos, desde el punto de vista de servicios para un cliente, ofrecidos en la capa de formato de datos DICOM. De esta forma, se asegura independencia de las funciones de la capa de Formato de Archivos con respecto a las de Formato de Medios y de Medios Físicos. Cuando las entidades de aplicación DICOM participan en el intercambio de información, a través del almacenamiento en medios, utilizan los siguientes servicios de la capa de Formato de Archivos:

- M-WRITE, para crear nuevos archivos en un grupo de archivos y la asignación de su ID.
- M-READ, para leer archivos existentes basados en su ID.
- M-DELETE, para borrar archivos existentes basados en su ID.
- M-INQUIRE FILE-SET, para solicitar la cantidad de espacio libre disponible para la creación de nuevos archivos dentro de un grupo de archivos.

- M-INQUIRE FILE, para solicitar la fecha y la hora de creación de archivos (o la última aplicación, en su caso) para cualquier archivo dentro de un grupo de archivos.

Una entidad de aplicación DICOM, como cliente de la capa de Formato de Archivos, puede tomar uno o más de los siguientes tres roles:

1. Creador de grupo de archivos (FSC). Una entidad de aplicación toma este rol a través de operaciones M-WRITE para crear el archivo DICOMDIR y cero o más archivos DICOM.
2. Lector de grupo de archivos (FSR). Una entidad de aplicación toma este rol a través de operaciones M-READ para acceder a uno o más archivos en un grupo de archivos. Un lector de grupo de archivos no puede modificar ninguno de los archivos del grupo de archivos (incluyendo el archivo DICOMDIR).
3. Modificador de grupo de archivos (FSU). Una entidad de aplicación toma este rol a través de operaciones M-WRITE y M-DELETE. Este lee pero no puede modificar el contenido de los archivos en un grupo de archivos excepto el archivo DICOMDIR; éste creará archivos adicionales a través de M-WRITE o borrará archivos existentes a través de M-DELETE.

De acuerdo a estos roles, las posibles tareas a realizar, a través de los servicios de la capa de Formato de Archivos, comprenden una de las siguientes operaciones:

- Creador de grupo de archivos (FSC)
- Lector de grupo de archivos (FSR)
- Creador y lector de grupo de archivos (FSC+FSR)
- Modificador de grupo de archivos (FSU)
- Modificador y creador de grupos de archivos (FSU+FSC)
- Modificador y lector de grupos de archivos (FSU+FSR)
- Modificador, creador y lector de grupo de archivos (FSU+FSC+FSR)

Así mismo, se deben soportar las operaciones de medios definidas en laTabla 2.1.

Tabla 2.1 - Operaciones de medios y roles [Nema, 2007]

Operaciones del rol	ROLES	M-WRITE	M-READ	M-DELETE	M-INQUIRE FILE-SET	M-IQUIRE FILE
FSC		Obligatorio	No requerido	No requerido	Obligatorio	No requerido
FSR		No requerido	Obligatorio	No requerido	No requerido	Obligatorio
FSC+FSR		Obligatorio	Obligatorio	No requerido	Obligatorio	Obligatorio
FSU		Obligatorio	Obligatorio	Obligatorio	Obligatorio	Obligatorio
FSU+FSC		Obligatorio	Obligatorio	Obligatorio	Obligatorio	Obligatorio
FSU+FSR		Obligatorio	Obligatorio	Obligatorio	Obligatorio	Obligatorio
FSU+FSC+FSR		Obligatorio	Obligatorio	Obligatorio	Obligatorio	Obligatorio

2.1.2.1.1 Formato de Datos DICOM

Para dar formato a los datos y transformarlos en archivos DICOM, esta capa incluye los siguientes elementos de especificación:

- Clases SOP de medios de almacenamiento y sus definiciones de objetos de información asociada
- El formato de archivo DICOM
- El formato de archivo seguro DICOM
- La clase SOP del directorio de medios de almacenamiento
- Los perfiles de aplicación de medios de almacenamiento
- Los perfiles de seguridad para medios de almacenamiento
- El formato del directorio (DICOMDIR).

Como se mencionó en la sección 1.3 un IOD es un objeto de información, que contiene a la imagen misma, y las clases SOP contienen a los IOD's. Ejemplos de IOD´s son la información de modalidades de imágenes médicas, información del paciente, información de resultados de diagnóstico, etc.

Las clases SOP y los IOD´s usados por los medios de almacenamiento deben seguir las especificaciones establecidas en las partes 3 y 4 del estándar. Además de la clase SOP relacionada al objeto de información específico, otras clases SOP adaptadas a los medios de almacenamiento pueden usarse para proveer referencias (o directorios) basados en la información médica, facilitando así el acceso a la información clínica de imágenes. Una de ellas sería la clase SOP del directorio de medios de almacenamiento. Todas las clases, incluyendo las de medios físicos, están definidas en la parte 4 del estándar. Las instancias de las clases SOP de medios de almacenamiento están representadas en un archivo llamado DICOMDIR, a través de las especificaciones de la descripción de la organización de los datos (parte 10 del estándar).

Las clases SOP y sus IOD´s asociados se usan para encapsular información específica de imágenes médicas de la entidad de aplicación a la capa de formato de archivos. El uso de IOD´s en conjunto con los servicios de medios de almacenamientos forma un grupo de clases SOP de medios de almacenamiento. Los servicios de medios de almacenamiento (por ejemplo, lectura y escritura) deben ser creados por los servicios de almacenamiento de la capa de Formato de Archivos.

Definición del formato

El formato de Archivo DICOM provee un mecanismo para encapsular, en un archivo, el grupo de datos que representa la instancia SOP relacionada con un IOD DICOM, como producto de la misma capa. Como se muestra en la Figura 2.5, la cadena de bytes del grupo de datos está ubicada dentro del archivo después de la meta-información DICOM. Cada archivo contiene una sola instancia SOP.

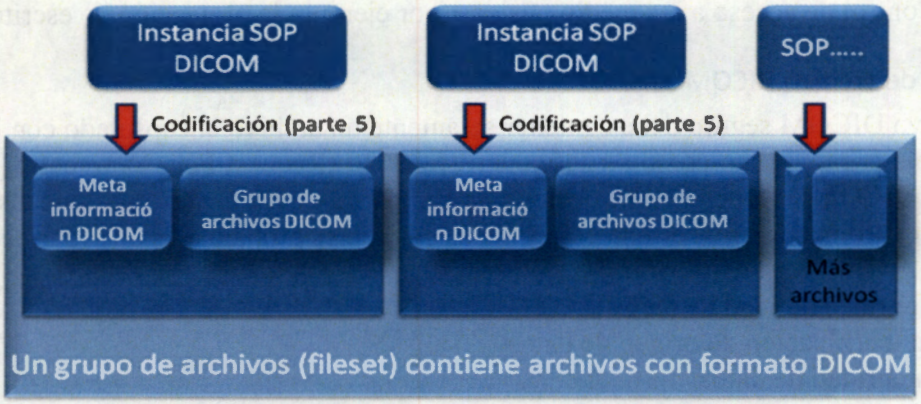

Figura 2.5 – Formato de Archivos DICOM [Nema, 2007]

La meta-información incluye la información de identificación en el grupo de datos encapsulado. Esta cabecera contiene 128 bytes seguida por un prefijo DICOM de 4 bytes, seguido de los meta-elementos mostrados en la Tabla 2.2. Esta cabecera se debe incluir en cualquier archivo DICOM. Todo archivo estructurado de acuerdo con estas reglas puede ser usado por un perfil de aplicación. Cabe señalar que, excepto por el preámbulo de 128 bytes y el prefijo de 4 bytes, los meta-elementos definidos dentro de la meta-información deben ser codificados usando una Sintaxis de Transferencia del valor de representación (VR, por sus siglas en inglés) "Little endian" (UID=1.2.840.10008.1.2.1) como lo define la parte 3.5 del estándar.

Tabla 2.2 – Componentes de un Archivo DICOM [Nema, 2007]

Nombre del atributo	Tag	Tipo	Descripción del atributo
Preámbulo de archivo	Sin campo de longitud	1	Un campo de 128 bytes disponible para perfil de aplicación
Prefijo DICOM	Sin campo de longitud	1	Cuatro bytes que contienen el carácter "DCM"
Versión de la Meta-información	(0002,0001)	1	Este es un campo de dos bytes donde se identifica la versión
Longitud del grupo	(0002,0000)	1	Número de bytes que siguen a los meta-elementos
UID de la clase SOP para almacenamiento en medios	(0002,0002)	1	Identificador de la clase SOP asociada al grupo de archivos

Encapsulado de grupos de datos

Cada archivo debe contener solamente un grupo de datos representando una sola instancia SOP relacionada a una sola clase SOP (y su correspondiente IOD). La Sintaxis de Transferencia usada para codificar el grupo de datos debe ser la que se identifique por la sintaxis de transferencia UID (identificador único DICOM) de la Meta-Información del archivo DICOM.

Soporte del manejo de la información de archivos

El formato del archivo DICOM no incluye información de su manejo con el fin de evitar duplicación con funciones relacionadas con la capa de Formato de Medios. Sin embargo, si es necesario para un perfil de aplicación dado. En este sentido, la siguiente información debe ser entregada por la capa de Formato de Medios:

1. Identificación propietaria del contenido del archivo
2. Estadísticas de acceso al archivo (por ejemplo la fecha y hora de creación)
3. Control de acceso a la aplicación de archivos.
4. Control de acceso a los medios físicos (por ejemplo la protección de escritura)

Formato de archivo DICOM seguro

Un archivo DICOM seguro debe contener solamente un archivo encapsulado con la sintaxis de mensaje criptográfico como se define en RFC2630 (Request For Comments) [pt. 10 del estándar DICOM]. Dependiendo de los algoritmos de encriptación para encapsulado, un archivo seguro puede proveer una o más de las siguientes propiedades de seguridad:

1. Confidencialidad de datos
2. Autentificación de origen de datos
3. Integridad de datos

Grupo de archivos

Los archivos son identificados por un ID que es único dentro del contexto de un grupo de archivos. Un grupo de archivos es una colección de archivos que comparte un espacio común dentro del cual los identificadores (ID´s) de los archivos son únicos. Un ID de un archivo puede contener de uno a ocho componentes. Cada componente es una cadena de uno a ocho caracteres de un subconjunto del repertorio ASCII estándar (específicamente de la sección G0 de ISO 8859). Esta estructura de ID's de archivos, vista como una secuencia de componentes, permite a la capa de Formato de Archivos DICOM organizar una selección de archivos en un modo jerárquico. No hay convenciones definidas para el estándar DICOM en el uso de las estructuras de los componentes ID's y su contenido, excepto para el ID del DICOMDIR. La capa de Formato de Archivos ofrece la capacidad de crear y acceder a más de un archivo dentro del grupo. El espacio de los ID´s de archivos, que corresponde a un grupo de archivos, debe estar asociado a una representación apropiada de la estructura definida en la capa de Formato de Medios. Este mapeo se especifica en la parte 3.12 del estándar DICOM para cada especificación de formato de medio.

Un solo archivo con un ID DICOMDIR debe ser incluido en cada grupo de archivos. Un grupo de archivos debe ser identificado por un UID de grupo de archivo el cual se debe registrar de acuerdo con las reglas de registro de UID especificadas en la parte 3.5 del estándar DICOM. Cuando se agregan o remueven archivos a un grupo, el UID de éste no debe cambiar. Un grupo de archivos debe ser identificado también por un ID, la cual provee una simple referencia legible. Un ID de un grupo de archivos es una cadena de 0 a 16 caracteres de un subconjunto del repertorio ASCII estándar. Un ID debe estar asociado a un identificador apropiado en la capa de Formato de Medios.

Para poder asociar la información de forma adecuada desde el formato de archivo DICOM al formato de medios específicos, es necesario utilizar los servicios de la capa de Formato de Medios.

2.1.2.2 Capa de formato de medios

En esta capa se organiza la información generada en la capa de Formato de Archivos DICOM para presentarla a la capa de Medios Físicos dependiendo del medio físico al que se desee acceder. Esta estructura de datos de archivos y sus estructuras de directorios asociadas se deben organizar de acuerdo al medio como tal, para permitir el acceso y manejo de la información. Por ejemplo, si el medio sobre el cual se va a escribir es un CD, se agrega la cabecera a los archivos DICOM y posteriormente se construye una imagen ISO que finalmente se envía a la capa de Medios Físicos para que ésta concluya la operación.

Cabe mencionar que los servicios para organizar estas estructuras para los formatos específicos que requiere el medio no están especificados en el estándar DICOM. Sin embargo, la estructura de datos se encuentra definida por otros estándares, como el ISO9660 para la estructura de CD/DVD, para los medios específicos soportados por una implementación.

Una vez que se tiene el formato de datos DICOM para el almacenamiento y el formato de medio para el cual se desea realizar el acceso, se requiere utilizar los servicios que ofrece la capa de medios físicos para el acceso al medio físico como tal.

2.1.2.3 Capa de medios físicos

En esta capa se definen las características del medio físico. Dichas características incluyen el factor de forma, dimensión, propiedades de grabado y características mecánicas. Esta capa también define la organización y agrupación de los bits grabados.

A priori, esta capa debe ofrecer información a la capa de Formato de Medios determinar si es posible el acceso al medio específico, dependiendo del espacio disponible, permisos y velocidades soportadas para acceso, etc.

El estándar DICOM tampoco especifica cómo deben implementarse los servicios de ésta capa, ni los detalles de interfaz hacia la capa anterior. Sin embargo, es de utilidad para este proyecto saber que dichos servicios dependerán de la capacidad de comunicación con el medio directamente a través del sistema operativo para la obtención de la información deseada.

2.1.3 Factibilidad de la implementación DICOM para medios físicos.

Luego de haber revisado el estándar DICOM en las partes específicas relacionadas con el almacenamiento en medios físicos, junto con la información y modelos que se plantean, es posible percibir la dimensión del problema a resolver. De esta forma el plantear bibliotecas que implementen las especificaciones del estándar, favorece el desarrollo de PACS estandarizados con funcionalidad para almacenamiento en medios físicos que puedan requerirse en las áreas de radiología de algún hospital, lo cual puede extender los objetivos de este proyecto.

Previo al desarrollo de éste proyecto de tesis, se realizó un análisis de algunos sistemas que han implementado varios servicios DICOM, con la finalidad de conocer sus alcances, en cuanto a cobertura del estándar. En la siguiente sección se presenta el estado del arte en varias de estas implementaciones.

2.2 Estado del arte en implementaciones DICOM

En esta sección se presenta un análisis del estado del arte de las implementaciones DICOM y de sus características, tratando de evaluar y comparar las tecnologías con respecto a la cobertura del estándar.

2.2.1 Implementaciones existentes

Hasta la fecha han existido diversas implementaciones que han reportado una cobertura limitada o parcial de las especificaciones DICOM, haciendo hincapié en que existen implementaciones comerciales que no son abiertas, y por lo tanto, no se incluyen en este análisis [DICOM Int., 2007]. Un punto importante que hay que destacar es que los sistemas y aplicaciones que se utilizan para este análisis son base del funcionamiento de muchos otros. En la siguiente lista se muestran los principales sistemas y aplicaciones que cumplen con los requisitos de ser software libre, código abierto y son una implementación que cubre al menos una parte de las especificaciones DICOM de manera autónoma.

- Pixelmed (http://www.pixelmed.com)
  - Desarrollado en Java, es un conjunto de bibliotecas con una cobertura de la mayor parte de especificaciones DICOM, excepto las partes correspondientes al almacenamiento en medios físicos. Su arquitectura permite la conectividad con otros sistemas o aplicaciones, aunque su configuración es complicada ya que requiere interacción con otras aplicaciones, por ejemplo con JBoss para los servicios Web. Reporta un desarrollo basado en un proceso bien definido.
- Jdicom (http://www.tiani.com/JDicom)
  - Desarrollado en Java, es un conjunto de bibliotecas que ofrece cobertura de visualización para imágenes DICOM. Ofrece soluciones del rol SCU para visualización, por lo que no cubre los servicios relacionados con el rol SCP. Es utilizado por sistemas que sólo requieren de un cliente para visualización. No se reporta ninguna metodología de desarrollo en su sitio oficial.
- jdcm toolkit (http://www.geocities.com/gigiobb/index.html)
  - Desarrollado en Java, es un conjunto de bibliotecas que ofrece cobertura de comunicación en red de las especificaciones DICOM. No tiene cobertura para los servicios Web ni de almacenamiento, aunque su arquitectura permite la conectividad con otros sistemas o aplicaciones. No se reporta ninguna metodología de desarrollo en su sitio oficial.
- cdmedicpacsweb (http://cdmedicpacsweb.sourceforge.net/cdmedic_en.html)
  - Desarrollado en Java, es un conjunto de bibliotecas que ofrece cobertura para servicios Web DICOM. Se utiliza como complemento de jdcm-toolkit para ofrecer una cobertura de las especificaciones DICOM, a excepción del

almacenamiento en medios físicos. No se reporta ninguna metodología de desarrollo en su sitio oficial.

- Osirix (http://homepage.mac.com/rossetantoine/osirix/Index2.html)
  - Está desarrollado en Objective C, y aunque implementa algunas soluciones de Pixelmed, tiene una cobertura amplia sobre el estándar a excepción de los servicios Web. Funciona con base en pluggins, sin embargo, es una aplicación compatible sólo con Mac, aunque si resuelve el almacenamiento en medios físicos.

2.2.2 Cobertura actual del estándar DICOM

Haciendo un análisis comparativo con las implementaciones mencionadas anteriormente, mostrado en la Tabla 2.3, se llega a la conclusión de que en el caso de los tres primeros sistemas a los que se ha hecho referencia y que tienen en común ser código abierto y software libre, se tiene la opción de añadir o modificar el código reutilizando el desarrollo existente, sin embargo, no cubren algunas partes del estándar DICOM, principalmente el almacenamiento en medios físicos.

Tabla 2.3 – Otros sistemas que implementan DICOM

Nombre	Partes no cubiertas	Desventajas
Pixelmed	Partes 10, 11 y 14	No soporta la función de almacenamiento en medios físicos
Jdicom	Partes 10, 11, 14, 15, 16, 17 y 18	No soporta la función de almacenamiento en medios físicos, perfiles de seguridad, mapeo en niveles de grises, reportes estructurados ni WADO (Web Access DICOM Objects)
Jdcm toolkit	Partes 10, 11, 14, 15, 16, 17 y 18	No soporta la función de almacenamiento en medios físicos, perfiles de seguridad, mapeo en niveles de grises, reportes estructurados ni WADO
Cdmedicpacsweb	Partes 10, 11, 14, 15, 16, 17 y 18	No soporta la función de almacenamiento en medios físicos, perfiles de seguridad, mapeo en niveles de grises ni reportes estructurados.
Osirix	Partes 10, 11, 14, 15, 16, 17 y 18	No soporta WADO y funciona sólo en sistemas operativos Mac.

Observando la tabla anterior, podemos darnos cuenta que no es necesario "re-inventar el hilo negro", ya que muchas partes de DICOM ya están implementadas y pueden reutilizarse integrándose en distintos sistemas, sin embargo, habrá que revisar la manera en que están fabricadas y saber si son capaces de soportar cambios o actualizaciones con respecto a las nuevas tecnologías. Este análisis, de hecho, da una justificación plena para desarrollar una biblioteca de software que pueda utilizarse por cualquier aplicación y que funcione como complemento para la cobertura de las especificaciones DICOM, ya que ninguna de las implementaciones mencionadas, a excepción de Osirix, cubre el almacenamiento en medios físicos.

2.2.3 Implementación DICOM en la UAM-I

Desde hace algunos años, en el área de investigación de Procesamiento Digital de Señales e Imágenes Biomédicas (PDSIB) del departamento de Ingeniería Eléctrica, en la UAM-I, se inició el estudio de los sistemas PACS con la intención de aplicarlos en algún hospital del sector salud. Desde entonces con dicho estudio se ha considerado el tener como marco de referencia al estándar DICOM 3.0 el cual se decidió implementar, debido a su aceptación, como norma mundial en el manejo y transferencia de imágenes médicas.

Después de iniciar la revisión de dicho estándar y dimensionar su magnitud y complejidad, se generó un primer trabajo de tesis de maestría con el objetivo de implementar el estándar DICOM. En este trabajo se planteó el desarrollo del protocolo de comunicación DUL (DICOM Upper Layer) [Martínez, 1999] correspondiente a la parte 08 del estándar, teniendo como base los protocolos de Internet (TCP/IP) y considerando como requisito indispensable el diseño de un producto de software de calidad. Para lograrlo, se propuso el uso de un proceso de desarrollo de software bien establecido. A partir de ese momento se planteó como objetivo general el desarrollo de una biblioteca de software para soportar la construcción de sistemas PACS estandarizados a DICOM 3.0 de forma modular. La cobertura de los módulos, hasta la fecha, por parte de proyectos realizados en la UAM-I se ilustran en la Figura 2.6 (partes con bordes grises implementados y codificados, partes con bordes negros diseñados pero sin codificar) y Figura 2.7 (partes con bordes blancos implementados y codificados, partes con bordes negros diseñados pero sin codificar y las partes con sombra gris corresponden al almacenamiento en medios físicos), con el detalle de su implementación lograda (las partes no sombreadas corresponden a las que no han sido cubiertas incluyendo el almacenamiento en medios físicos).

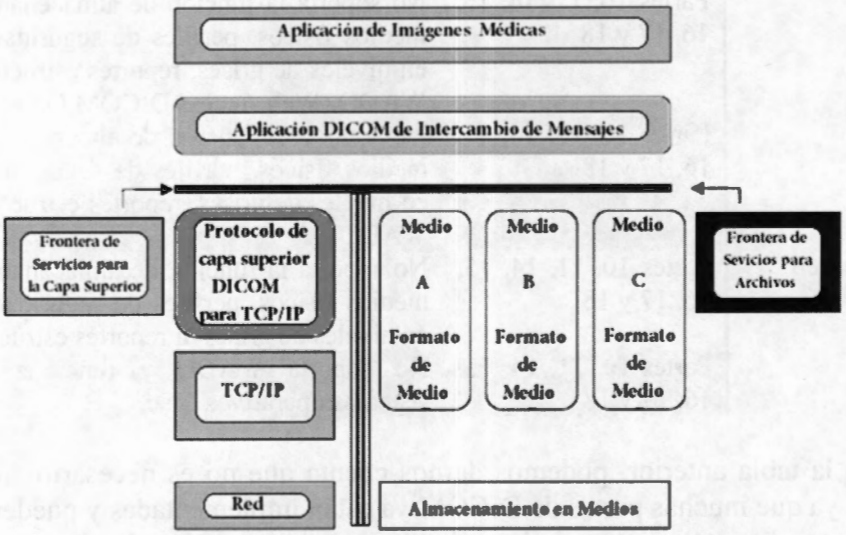

Figura 2.6 – Partes cubiertas en la actualidad por la UAM-I (parte 1) [Nema, 2007]

Posteriormente, se empezó a gestar un segundo trabajo, en donde se propuso el diseño del protocolo DIMSE (DICOM Message Service Element) [Jiménez, 2000] correspondiente a la parte 7 del estándar, a través de un proceso de desarrollo de software. Poco tiempo después se inició un tercer trabajo de tesis de maestría relacionado a la representación del

modelo de información especificado en las partes 3 y 4 del estándar DICOM [Nema, 2007], trabajo que consistió en garantizar una representación adecuada de cada uno de los objetos de información especificados por el estándar, que son básicos para su transferencia por una red de computadoras. Para lograr estándares de calidad adecuados, también se aplicó un proceso de desarrollo de software establecido.

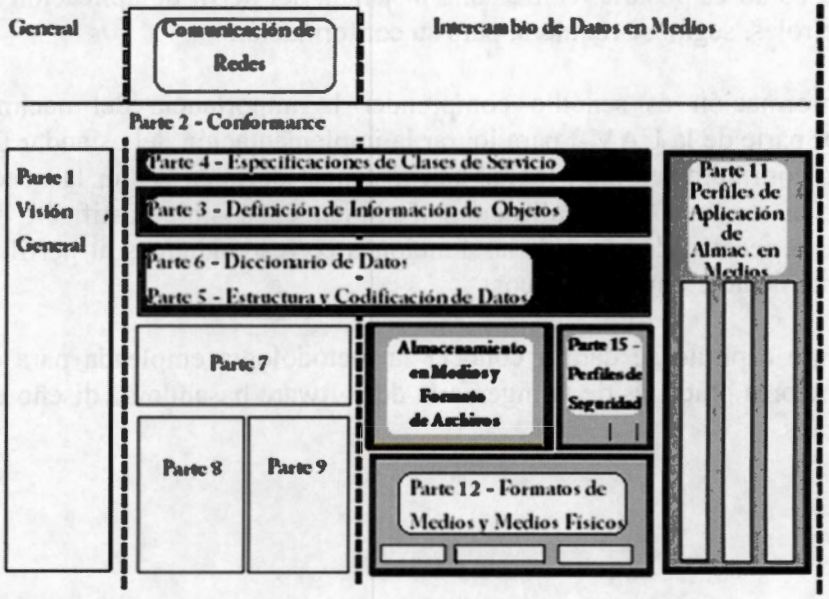

Figura 2.7 – Partes cubiertas en la actualidad por la UAM-I (parte 2) [Nema, 2007]

Adicionalmente, se han realizado trabajos a nivel de proyectos terminales de la licenciatura en ingeniería biomédica [Chavez, 2001] [Muñoz de Cote, 2003]. El primero consistió en cubrir las especificaciones de las partes 5 y 6 del estándar DICOM, en donde se estableció el cumplimiento de las especificaciones para garantizar una codificación apropiada para cada uno de los parámetros de los objetos de información estandarizados; el producto final fue un subsistema codificado en los lenguajes de programación C++ y Java. El segundo consistió en implementar el protocolo de comunicación DUL el cual fue codificado en el lenguaje de programación Java. Varios de estos trabajos han sido presentados en congresos especializados y producido publicaciones en la modalidad de memorias en extenso. La experiencia adquirida se ha estado aplicando en el proyecto "PACS-CNR", propuesto por el Instituto Nacional de Rehabilitación y han generado algunas publicaciones [Martínez, 2005] y la implementación del PACS de este instituto.

El valor que han agregado todos y cada uno los proyectos que tienen la intención de cubrir el estándar en su totalidad son, por una parte, el software que conforma cada biblioteca como tal, la cobertura funcional DICOM y la visión de componentes por biblioteca (que obedecen exactamente a la necesidad de implementación por instancia requerida) y, por otra parte, la estrategia y metodologías para producción de software y manejo de proyectos colaborativos, ya que esto ha generado una proyección más certera en la medición de riesgos, administración de actividades, captura de requerimientos, utilización de herramientas, entre otros mecanismos de control de procesos.

2.3 Sumario

En este capítulo se obtuvo una visión general con más detalle del estándar DICOM, las partes que lo componen y la función de éstas para la comunicación; además, la estructura de datos y el cómo es posible realizar una instancia del perfil de aplicación a través de la definición de roles, según se requiera, para su conformación.

Con esta información es sencillo comprender la importancia del nacimiento de los proyectos por parte de la UAM-I para lograr la implementación del estándar DICOM como componentes configurables para lograr una instancia definida según la necesidad que se desee cubrir de un perfil de aplicación. También es posible justificar, mediante esta información, la necesidad de cubrir las funciones correspondientes al perfil de aplicación para almacenamiento en medios físicos.

En el siguiente capítulo se dará a conocer la metodología empleada para este proyecto usando las mejores prácticas de la ingeniería de software basando su diseño en la creación de componentes.

3. Métodos, técnicas y herramientas de desarrollo

3.1 Introducción

En las dos últimas décadas las propuestas de modelado y posteriormente las herramientas de soporte han sido la clave para el logro de avances significativos en el desarrollo de proyectos de software. El proceso de desarrollo asumido en este contexto lleva asociada una marcada tendencia hacia el control del proceso mediante una rigurosa definición de actividades, artefactos (producto o productos generados por una tarea) y roles. Este esquema para abordar el desarrollo de software ha demostrado, para varios proyectos, ser efectivo en donde por lo general se exige un alto grado de calidad en el proceso [Spinec.org, 2007].

Para entender mejor una metodología, se presentan los siguientes conceptos desde el punto de vista de un proceso de desarrollo de software:

- Proceso: acción o sucesión de acciones continuas regulares que ocurren o se llevan a cabo de una forma definida, y que llevan al cumplimiento de algún resultado; una operación continua o una serie de operaciones.
- Metodología: conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevos proyectos de software. (Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal.)
- Tarea: actividades elementales en que se dividen los procesos.
- Procedimiento: definición de la forma de ejecutar la tarea.
- Método: Procedimiento que se sigue en las ciencias para hallar la verdad y enseñarla.
- Herramienta: para realizar un procedimiento podemos apoyarnos en las herramientas de software que agilizan su aplicación.
- Artefacto o producto: es un producto originado en el flujo de trabajo que se deriva de los ciclos de vida del proceso.

Una metodología puede seguir uno o varios modelos de ciclo de vida para un proyecto. En donde el ciclo de vida indica qué es lo que hay que obtener a lo largo del desarrollo del proyecto pero no cómo hacerlo. La metodología indica cómo hay que obtener los distintos productos parciales y finales. Existen distintos modelos de referencia en los que se describe el proceso de desarrollo de software pero no se define la manera de administrar el proyecto, otros describen dentro del mismo proceso una metodología que integra tareas de administración de proyectos y otros más describen un marco de procesos que tienen un mayor alcance y son más complejos. Un ejemplo es MoProSoft [Oktaba, et al, 2003] que define dentro de su estructura un modelo de 9 procesos.

Una metodología se encuentra contenida en un proceso, sin embargo, lo que define a esta metodología es la instancia que se selecciona para desarrollar una solución específica. Algunas metodologías suelen incluir tareas para la administración dentro del proceso, que es el caso de la metodología empleada en este proyecto. Actualmente son utilizadas varias metodologías dependiendo de las habilidades de los desarrolladores, el tipo de proyecto y el dominio de las herramientas, tales como Proceso Unificado de Rational, (RUP: *Rational Unified Process*) [Wikipedia.org - RUP, 2007], métodos ágiles como podría ser la programación extrema (XP: *eXtreme Programming*) [Extremeprograming.org, 2007],

desarrollo guiado por aspectos (FDD: *Feature Driven Development*) [Nebulon Pyy, 2007], entre otras.

En el caso particular de este proyecto, se utiliza el RUP del cual se genera una instancia como metodología concreta para el desarrollo del sistema de almacenamiento en medios físicos DICOM.

3.1.1 Antecedentes de RUP

En la Figura 3.1 3.1 se muestra la evolución de procesos hasta llegar al RUP.

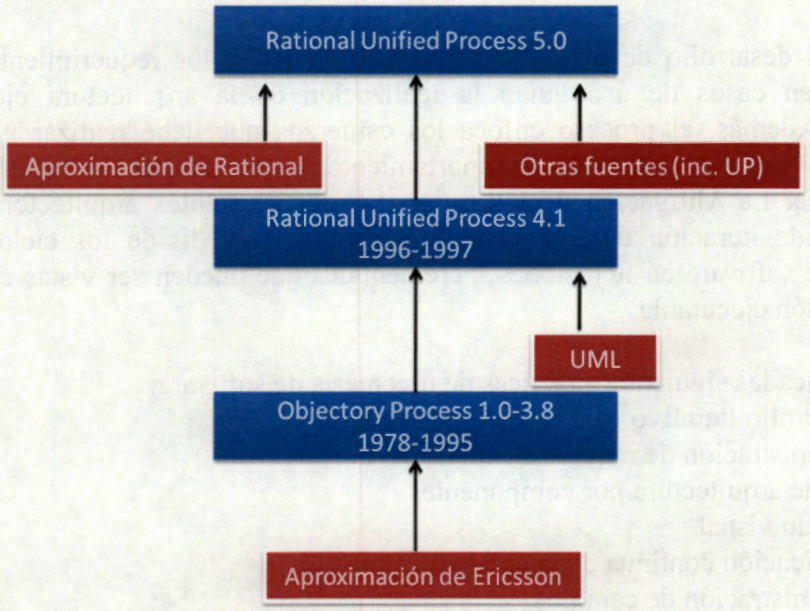

Figura 3.1 - Evolución de RUP [Jacobson, 1999]

El antecedente más importante se ubica en 1967 con la Metodología Ericsson (*Ericsson Approach*) elaborada por Ivar Jacobson, que es una aproximación de desarrollo basada en componentes que introdujo el concepto de Caso de Uso. Entre los años de 1987 a 1995 Jacobson fundó la compañía *Objectory* AB y lanza el proceso de desarrollo *Objectory* (abreviación de *Object Factory*). [UPV, 2006]

Posteriormente en 1995 *Rational Software Corporation* adquiere *Objectory AB* y entre 1995 y 1997 se desarrolla *Rational Objectory Process* (ROP) a partir de *Objectory* 3.8 y del Enfoque Rational (*Rational Approach*) adoptando UML como lenguaje de modelado [UPV, 2006].

Desde entonces y con Grady Booch, Ivar Jacobson y James Rumbaugh a la cabeza, Rational Software desarrolló e incorporó diversos elementos para expandir ROP hacia el RUP, destacándose las siguientes mejoras en el proceso:

- Utilización de la metodología del Proceso Unificado.
- Adaptación de mejoras en la administración de proyectos basadas en el PMBOOK (Project Management Body of Knowledge).
- Adaptación del flujo de trabajo conocido como modelado del negocio.

En 1998 se lanza el RUP con las características mencionadas anteriormente y hasta la fecha se ha mantenido estable y funcional [Project Birdy, 2007]. Sin embargo, con el paso de los años se han hecho extracciones del proceso para ofrecer versiones más ágiles y ligeras basadas en el RUP, por ejemplo el OpenUP [Eclipse, 2007].

3.2 El proceso unificado de Rational

RUP es un proceso de ingeniería de software que describe quién hace qué, cuándo y cómo en un proyecto de desarrollo y mantenimiento de software. Este proceso tiene las características de ser guiado por casos de uso y por riesgos, centrado en arquitectura e iterativo incremental. Define y especifica artefactos como producto de desarrollo de software o como producto de administración del proyecto y roles. Este último es el que desempeña una persona en un determinado momento; una persona puede desempeñar distintos roles a lo largo del proyecto.

A través del desarrollo de un proyecto basado en RUP, los requerimientos funcionales expresados en casos de uso guían la realización de la arquitectura ejecutable de la aplicación. Además, el proceso enfoca los esfuerzos que debe realizar el equipo en la construcción de elementos de comportamiento y estructura desde una perspectiva arquitectónica. La Mitigación de los riesgos de los elementos arquitectónicos guía a la visión de cada iteración durante el ciclo de vida. RUP divide los ciclos de vida del desarrollo de software en iteraciones incrementales que pueden ser vistas como versiones de la aplicación ejecutable.

RUP especifica las siguientes prácticas de ingeniería de software:

1. Desarrollo Iterativo
2. Administración de requerimientos
3. Uso de arquitectura por componentes
4. Modelo visual
5. Verificación continua de la calidad
6. Administración de cambios
7. Modelado de negocio
8. Administración de riesgos
9. Interacción multidisciplinaria
10. Gestión de responsabilidades

RUP, junto con el Lenguaje Unificado de Modelado UML [Amber, 2005], constituyen la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos. RUP [Wikipedia.org - RUP, 2007] incluye al Proceso Unificado [Wikipedia.org - UP, 2006] realizado por "Rational Software" [IBM, 2007]. Estas características le dan a RUP una gran potencialidad.

3.2.1 Estructura del proceso unificado de Rational

El RUP considera 4 fases y varias disciplinas para el desarrollo de un proyecto de software. Este proceso provee un aprovechamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la

producción de un sistema de alta calidad que resuelva realmente las necesidades del usuario final con una programación predecible y alcanzable, a través de elementos muy bien identificados utilizados en cada fase. En la Figura 3.2 se muestran la relación de los elementos básicos de RUP cuyos detalles se presentan en la sección 3.4.

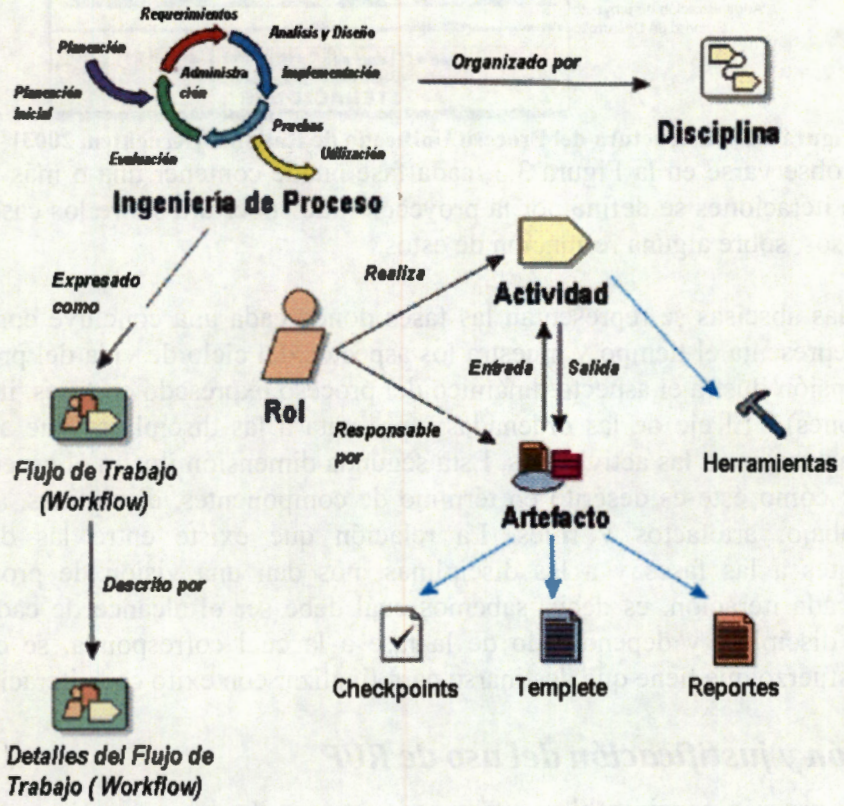

Figura 3.2 – Diagrama de elementos del RUP y sus relaciones [Kruchten, 2003]

La Figura 3.3 ilustra la estructura que contempla el RUP, la cual tiene dos dimensiones:

- El eje horizontal representa el tiempo y muestra los aspectos del ciclo de vida del proceso. Esta dimensión ilustra el aspecto dinámico del proceso y está expresado en términos de fases e iteraciones.
- El eje vertical representa las disciplinas que agrupan a un conjunto de actividades. Esta dimensión enmarca el aspecto estático del proceso y cómo este es descrito en términos de los flujos de trabajo (workflows).

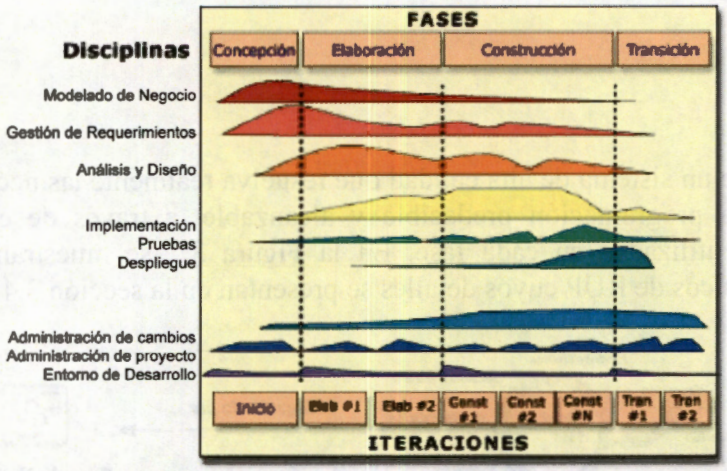

Figura 3.3 – Estructura del Proceso Unificado de Rational [Kruchten, 2003]

Como puede observarse en la Figura 3.3, cada fase puede contener una o más iteraciones. El número de iteraciones se define por la proyección de cobertura sobre los casos de uso y en algunos casos, sobre alguna refinación de estos.

En el eje de las abscisas se representan las fases donde cada una concluye con productos específicos, representa el tiempo y muestra los aspectos del ciclo de vida del proceso. Esta primera dimensión ilustra el aspecto dinámico del proceso expresado en fases, iteraciones e hitos (milestones). El eje de las ordenadas representa a las disciplinas que agrupan por lógica y por naturaleza a las actividades. Esta segunda dimensión ilustra el aspecto estático del proceso y cómo éste es descrito en término de componentes, disciplinas, actividades, flujos de trabajo, artefactos y roles. La relación que existe entre las dimensiones correspondientes a las fases y a las disciplinas, nos dan una visión de profundidad y amplitud de cada iteración, es decir, sabemos cuál debe ser el alcance de cada iteración dentro de su disciplina y dependiendo de la fase a la cual corresponda, se conocerá la cantidad de esfuerzo que tiene que destinarse para finalizar con éxito cada iteración.

3.3 Reflexión y justificación del uso de RUP

El RUP es un proceso considerablemente grande, que va desde una implementación para proyectos pequeños hasta otras para proyectos muy extensos y complejos. En este sentido es necesario seleccionar los elementos relevantes para un proyecto específico, ya que de lo contrario podrían ejecutarse tareas y generar productos innecesarios para la realización del proyecto.

La metodología empleada envuelve una serie de actividades, artefactos y roles configurables dependiendo de la complejidad del desarrollo deseado. De este modo, el administrador del proyecto decide qué elementos del proceso utilizar y cómo configurarlos dependiendo de sus necesidades específicas.

El uso del RUP se debe a dos razones. La primera es por la experiencia adquirida en el uso de éste proceso y en metodologías basadas en su precursor Proceso Unificado, a través del cual se ha buscado mejorar la calidad en desarrollo de software. La segunda es por la facilidad y flexibilidad de adaptación del RUP al desarrollo. Como una breve reflexión para

avalar la facilidad del seguimiento y adaptación del proceso, RUP permite llevar de manera sencilla desde un proyecto pequeño hasta un proyecto de dimensiones considerables, ya que funciona como una guía, tanto para la fabricación del proyecto como para su administración, que genera productos específicos y, de manera intuitiva, es posible obtener una instancia para un proyecto definido.

3.4 Instancia de RUP

La comprensión del conjunto de roles, disciplinas y artefactos que se tienen que considerar en el proceso, conforman un elemento clave para su aplicación en un problema o proyecto en particular. A continuación se define cada uno de estos conjuntos y la particularidad que adquieren (en caso de que se aplique) para ser implementados en este proyecto.

3.4.1 Roles, actividades y responsabilidades

Un Rol es una definición abstracta de un grupo de responsabilidades para la realización de actividades a desempeñar y artefactos a producir. Es importante que sean identificados todos los roles necesarios para este desarrollo. Sin embargo, tenemos que recordar que debido a la naturaleza de este proyecto (tesis de postgrado), sólo una persona está contemplada para el mismo, por lo que tendrá que adoptar todos estos roles y responsabilidades que representan. RUP define un grupo de roles para el desarrollo del proyecto, con sus actividades y responsabilidades correspondientes. El conjunto de roles utilizados en esta instancia se muestra a continuación:

- Analista de requerimientos: se encarga principalmente de investigar y establecer los requerimientos (funcionales y no funcionales).
- Arquitecto de software: se encarga de generar una estructura del software basada en los requerimientos
- Desarrollador de software: se encarga de organizar a la parte de diseño e implementación.
- Administrador de proyecto: organiza a todos los demás roles de acuerdo con la ingeniería de procesos empleada y verifica el desempeño y la evolución del proyecto.
- Evaluador del sistema: se encarga de las pruebas de calidad y funcionamiento del sistema.

3.4.2 Disciplinas

Este proceso está organizado en un conjunto de disciplinas (que se muestran en la Figura 3.4) que definen el flujo de trabajo (workflow) y otros elementos de proceso. Sin embargo, la instancia elegida no incluye todas las disciplinas debido a que no todas son necesarias para el propósito de este proyecto.

Figura 3.4 – Diagrama de Ingeniería de Procesos [Kruchten, 2003]

La Figura 3.4 muestra, además de las disciplinas, la secuencia cíclica de varias de ellas y su relación con las otras que no se plantean como cíclicas, como las de cascada por ejemplo. Prácticamente las disciplinas no cíclicas fueron las que no se emplearon en esta instancia, además de la evaluación la cual tiene dependencia de otros subsistemas.

En la Figura 3.5 se muestra la instancia del proceso, en donde la dimensión de las disciplinas muestra claramente la ausencia de algunas de ellas.

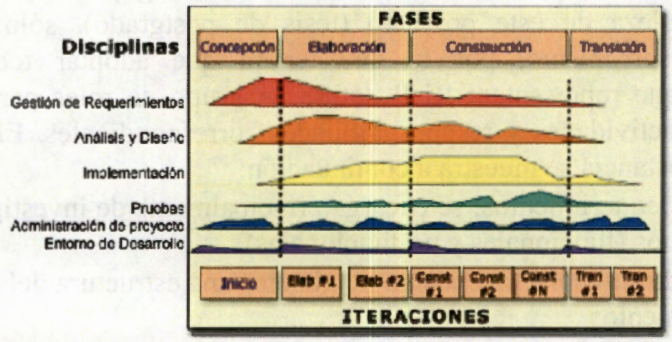

Figura 3.5 - Instancia del proceso con respecto a las disciplinas

La razón por la que se usó un grupo menor de disciplinas es debido a la naturaleza del proyecto, ya que por un lado corresponde a una tesis de maestría y persigue objetivos académicos y, por el otro, fue desarrollado por una sola persona. Las disciplinas que no fueron incluidas en la instancia se describen a continuación con su justificación correspondiente.

- Modelado de negocio.- Esta disciplina es necesaria cuando el cliente describe el sistema y sus funciones. En este caso no fue necesario ya que los requerimientos se establecieron a partir de las especificaciones del estándar DICOM y no de una persona como usuario final.
- Interfaz de usuario.- En el caso de una biblioteca la interfaz de usuario no es gráfica ya que el usuario de los productos generados es un programador.
- Administración de cambios.- La administración de cambios se realizó en forma simple considerando que las especificaciones DICOM no sufren cambios

significativos en el tiempo y los cambios relacionados con los resultados se resolvieron en las reuniones con los asesores y registrándolas en una bitácora.

Las disciplinas incluidas en la instancia se plantean en las siguientes subsecciones.
En particular, las actividades correspondientes a las disciplinas de captura y análisis de requerimientos, análisis y diseño, implementación y pruebas se realizaron a partir de la instancia del proceso unificado de Sun Microsystems [Sun Microsystems, 2002] ya que el RUP determina el flujo de trabajo a través de actividades, pero no ofrece detalle de cómo realizarlas. Esta instancia del proceso unificado se ilustra en la Figura 3.6 3.6 la cual describe el proceso para llegar a un modelo de la solución.

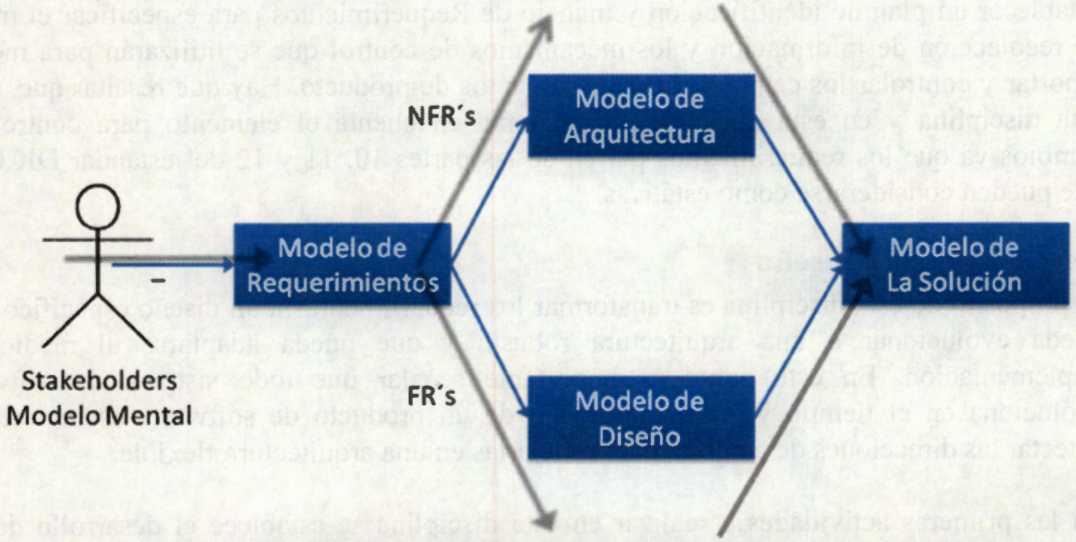

Figura 3.6 – Mapa del proceso propuesto por Sun Microsystems [Sun, 2002]

Los elementos del proceso para llegar a un modelo de la solución se documentan como lo establece el RUP para conservar la convención de tomar esta metodología como marco principal de referencia.

3.4.2.1 Requerimientos

El propósito de esta disciplina es identificar y entender el contexto del problema que se desea resolver. Hacer un análisis del problema envuelve la identificación de puntos clave de interacción entre el sistema y actores externos, definición de fronteras e identificación de impedimentos o riesgos involucrados en el desarrollo.

Los requerimientos de un producto son la descripción de las necesidades para su funcionalidad y sus características sistémicas. La meta principal de los requerimientos es identificar y documentar lo que realmente es necesario en la funcionalidad de un producto.

Las tareas principales de esta disciplina consisten en realizar la captura y especificación de requerimientos. El reto es definir los requerimientos sin ambigüedad, de tal manera que pueda ser cubierto el propósito de esta disciplina con la mayor exactitud posible. Todo proyecto a realizar con el RUP tiene su inicio con la captura de los requerimientos, considerada como el punto de partida para lograr:

- limitar el contexto del problema de acuerdo a sus alcances
- plantear la funcionalidad que se espera del sistema

Existen dos tipos de requerimientos: funcionales (FR´s) y no funcionales (NFR´s). Los FR´s describen las características del sistema en relación a una función u operación de negocio que debe expresarse dentro del mismo. Por ejemplo: "El sistema debe recolectar la siguiente información del cliente – nombre y dirección". Los NFR´s describen las características del sistema que soportan el "cómo" una operación es realizada. Por ejemplo, "El sistema debe soportar 10 usuarios simultáneos en una aplicación Web".

Una vez capturados los requerimientos, se genera el documento de especificación de requerimientos (SRS, por sus siglas en inglés) como producto de la especificación de requerimientos para el desarrollo del proyecto.

En esta disciplina, el trabajo principal lo realiza el rol analista, que es responsable de establecer un plan de identificación y manejo de Requerimientos para especificar el modo de recolección de información y los mecanismos de control que se utilizarán para medir, reportar y controlar los cambios de requerimientos de producto. Hay que resaltar que, para esta disciplina y en esta instancia, no se toma en cuenta el elemento para control de cambios ya que los requerimientos parten de las partes 10, 11 y 12 del estándar DICOM, que pueden considerarse como estáticas.

3.4.2.2 Análisis y Diseño

El propósito de esta disciplina es transformar los requerimientos a un diseño específico que pueda evolucionar a una arquitectura robusta y que pueda adaptarse al medio de implementación. En este punto es importante señalar que todo sistema de software evoluciona en el tiempo y, en el desarrollo de un producto de software, es importante detectar las direcciones de cambio para reflejarlas en una arquitectura flexible.

En las primeras actividades a realizar en esta disciplina se establece el desarrollo de un modelo de casos de uso que incluye la descripción de cada caso de uso con detalle, los cuales se obtienen directamente del documento de especificación de requerimientos. Posteriormente se obtienen los modelos de diseño y arquitectura, los cuales requieren un mayor esfuerzo para su realización dentro de la fase de análisis y diseño. En la Figura 3.6 puede observarse la relación de los modelos de diseño y arquitectura con los FR´s y NFR´s, respectivamente, para plantear un modelo de la solución.

A partir de los FR´s se obtiene el modelo de diseño que, en conjunto con el modelo de arquitectura, conforman el modelo de solución como se observa en la Figura 3.6.

Los productos que se deben generar en esta disciplina son el modelo de casos de uso, el modelo de diseño y el modelo de arquitectura. En ese sentido se utilizaron las plantillas de RUP para documentar y describir la información relacionada con dichos modelos.

3.4.2.2.1 Modelo de casos de uso

Un modelo de casos de uso describe el comportamiento del sistema cuando actores externos interactúan con él para realizar los casos de uso que representa. Está compuesto de diagramas de casos de uso y la documentación que describe a cada caso de uso dentro del diagrama.

Se propone la documentación de los casos de uso en base a los siguientes puntos:

- Nombre del caso de uso
- Propósito
- Curso típico de eventos
- Curso alternativo
- Post-condiciones
- Pre-condiciones
- Requerimientos especiales (no funcionales)
- Relaciones
- Puntos de extensión

3.4.2.2.2 Modelo de diseño

Para la obtención del modelo de diseño se requiere identificar abstracciones clave. Una abstracción clave es una clase u objeto que forma parte del vocabulario del dominio del problema. La determinación de las abstracciones clave para este dominio es un proceso de búsqueda para:

1. Identificar todos los candidatos a abstracciones comenzando por listar los sustantivos contenidos en el SRS.
2. Usar análisis CRC (Class-responsability-collaboration por sus siglas en inglés) para determinar el grupo esencial de las abstracciones clave.

Existen otras formas de encontrar las abstracciones clave, sin embargo, como se mencionó anteriormente, la técnica utilizada fue la sugerida por Sun Microsystems. [Sun, 2002]

La técnica para descubrir abstracciones clave usando análisis CRC consiste en considerar candidatos a abstracciones clave y después:

- Seleccionar una abstracción clave candidato
- Identificar un caso de uso en el cual el candidato es prominente
- Buscar los escenarios de casos de uso y FR´s para determinar las responsabilidades y colaboradores
- Documentar esta abstracción clave
- Actualizar las abstracciones previas con base en nuevos hallazgos

A partir de las abstracciones clave se obtiene un modelo conceptual, que es una especificación del dominio del problema con objetos obtenidos directamente de las abstracciones ya vistas como clases y con sus relaciones y dependencias bien definidas. A este modelo se le denomina modelo de dominio.

El modelo de diseño se obtiene buscando en el modelo de dominio componentes considerados como control, frontera o entidad, validando sus relaciones y comprobando la realización del comportamiento de los casos de uso, así como la interacción de los actores

involucrados. A esta serie de actividades se le llama análisis de robustez. La Figura 3.7 muestra gráficamente este procedimiento.

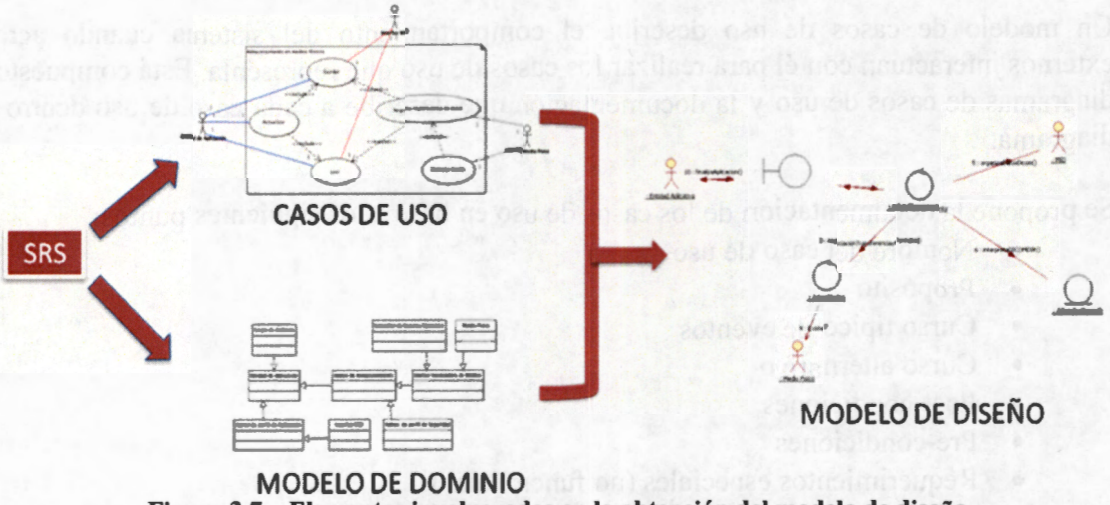

Figura 3.7 – Elementos involucrados en la obtención del modelo de diseño

3.4.2.2.3 Modelo de arquitectura

Es importante definir un tipo de arquitectura y para ello existen distintos modelos identificados por su complejidad, escala y/o distribución. Según [Sun Microsystems, 2002] la arquitectura se refiere a la representación abstracta de los componentes de un sistema y su comportamiento **[LucasianLabs, 2009].**

Este proyecto cubre sólo una parte de la arquitectura correspondiente a un sistema tipo PACS, por lo que tiene que cumplir cinco requisitos fundamentales:

1. Tener una arquitectura adecuada dependiendo de sus requerimientos no funcionales
2. Ser capaz de integrarse a una arquitectura mucho más compleja sin perder sus características no funcionales
3. Definir las limitaciones en la Implementación
4. Brindar un prototipo evolutivo
5. Proporcionar tiempos más exactos para entrega de productos parciales y/o finales.

La arquitectura de un sistema tipo PACS puede ser tan compleja como lo requiera su diseñador. Existen distintos tipos de arquitecturas, algunas de las más conocidas son las siguientes **[Wikipedia, 2009]**:

- Arquitectura de tipo monolítica - El software se estructura en grupos funcionales muy acoplados
- Arquitectura de tipo cliente-servidor – El software reparte su carga de cómputo en dos partes independientes
- Arquitectura de tres capas – Especialización de la arquitectura cliente-servidor donde la carga se divide en tres capas con un reparto claro de funciones, una capa para la presentación (interfaz de usuario), otra para el cálculo (donde se encuentra

modelado el negocio) y otra para el almacenamiento (persistencia). Una capa solamente tiene relación con la siguiente.

- Arquitectura de N-niveles **[LucasianLabs, 2009]** – Especialización de la arquitectura de 3 capas en donde se agrega alguna función por capa dependiendo de la lógica de negocio.

Algunas de las características arquitectónicas que se requieren para construir sistemas tipo PACS son:

- Reutilización de lógica de negocio para diferentes clientes o sistemas.
- Mejorar la escalabilidad.
- Mejorar rendimiento
- Alta escalabilidad
- Mejorar la flexibilidad.
- Independencia de la base de datos

De todas estas posibilidades, primero debemos ubicar al sistema que se obtendrá al finalizar este proyecto dentro de una arquitectura mucho más compleja. Por sus requerimientos no funcionales, podemos identificar a este sistema como un cliente pesado, ya que utilizará los recursos directamente de la máquina en donde funcionará. Sin embargo, forma parte de los servicios DICOM del PACS, por lo que también debe considerarse como parte funcional de este complejo sistema. Podemos entonces ubicar el prototipo generado por este proyecto como un cliente pesado de una arquitectura PACS cualquiera (de dos o más capas) para la función específica de almacenamiento en medios físicos.

Por ser una biblioteca cuya intención es ser portable, podríamos pensar que lo correcto es orientar la arquitectura hacia el tipo monolítico, sin embargo, hay que recordar que el usuario final será el mismo desarrollador que utilice esta biblioteca para la integración de un PACS con características muy específicas, es decir, dependerá del sistema operativo, medios físicos disponibles e interfaz de diccionario de datos independiente. Por esta razón, el sistema también tiene que seguir una arquitectura a la cual dichos usuarios puedan modificar o agregar capacidad de forma sencilla.

Debido a estas razones el tipo de arquitectura que mejor se adapta de acuerdo a los NFR´s es la arquitectura MVC (Por sus siglas en inglés Model/View/Controller), el cual se adapta al diseño en capas para la separación de funciones. Para comprender un poco mejor el modelo MVC adaptado a este trabajo podemos hacer una breve descripción de cada componente.

1. El Modelo es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el Modelo y sus Vistas, y notificar a las Vistas cuando cambia el Modelo.
2. La Vista es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del Modelo y muestra los datos al usuario. Interactúa con el Modelo a través de una referencia al propio Modelo.

3. El Controlador es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo.

En esta disciplina se destaca el trabajo de los roles del analista de requerimientos y arquitecto de software ya que entre ambos generarán la arquitectura del sistema. Ya definida la arquitectura del sistema, la siguiente disciplina es la implementación.

3.4.2.3 Implementación

El propósito principal de esta disciplina es lograr la refinación de los modelos de diseño y arquitectura durante su paso a través de la fase de implementación para poder llegar al código final, como puede observarse en la Figura 3.8. Según [Sun Microsystems, 2002], este es el modelo de la solución. Es importante recalcar que el RUP integra la arquitectura en todas sus disciplinas, mientras que SUN plantea distinta la forma de su obtención. En este caso se documenta la refinación de la arquitectura dentro de la implementación para seguir la metodología de RUP e identificarla como disciplina. Sin embargo, no hay que perder de vista que el contenido de esta sección está conformado por el modelo de la solución, que es el producto más importante que RUP sugiere obtener de esta disciplina.

El modelo de solución integra el modelo de arquitectura visto como esqueleto del sistema y el modelo de diseño como guía funcional del sistema, como puede observarse en la Figura 3.6. El resultado entonces es un diagrama que sirva de guía para que los programadores comiencen a codificar el sistema.

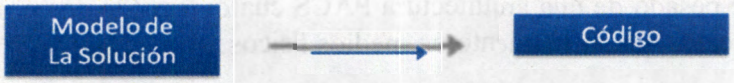

Figura 3.8 – Obtención del código [Sun Microsystems, 2002]

El camino de pasar del modelo de la solución al código, como lo muestra la Figura 3.8, impacta el contenido de la arquitectura. En términos coloquiales podríamos decir que es como "ponerle carne al esqueleto".

La estructuración del modelo de solución resulta en un grupo de subsistemas que se pueden desarrollar relativamente independientes, aclarando que están alineados con la arquitectura ya definida. Un modelo bien organizado puede prevenir los problemas de manejos de configuración y permite que el producto quede listo para agregar funciones extra o actualizaciones.

Es muy importante tener en cuenta que una vez realizados todos estos pasos para el modelo de solución, es muy recomendable aplicar patrones de diseño con los cuales se pretende:

1. Proporcionar catálogos de elementos reusables en el diseño de sistemas de software
2. Evitar la reiteración en búsquedas de soluciones a problemas ya conocidos y solucionados anteriormente
3. Formalizar un vocabulario común entre diseñadores
4. Estandarizar el modo en que se realiza el diseño

5. Facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando el conocimiento ya existente.

El rol de arquitecto de software lleva la responsabilidad de la estructura del modelo de la solución y debe tener experiencia suficiente con las herramientas con las que se esté llevando a cabo la construcción e integración de todos los módulos, definidas en la disciplina de entorno de desarrollo, ya que si no se tiene cuidado, puede obtener resultados inesperados. La construcción e integración de los componentes del sistema deberán llevarse a cabo por el desarrollador de software, el cual tiene que conocer perfectamente el lenguaje que se emplea para codificar el sistema. En esta disciplina se utilizan algunos componentes sugeridos por el RUP, desde las plantillas que describen la arquitectura del sistema, hasta el código generado como producto obtenido.

3.4.2.4 Pruebas

Esta disciplina tiene como propósito verificar que el código generado para el sistema funcione adecuadamente. Para cada Iteración se realizan pruebas parciales en donde el trabajo se enfoca principalmente en:

1. Identificar los objetivos específicos deseados para realizar pruebas
2. Identificar una estrategia de utilización de recursos óptima
3. Definir el alcance apropiado y fronteras para las pruebas
4. Definir como se asesorará y monitoreará el proceso

Basados en la metodología de [Sun Microsystems, 2002], las pruebas se enfocarán sólo en la funcionalidad del sistema. Se sugiere realizar pruebas funcionales y de integración.

Las pruebas de funcionamiento se realizaron introduciendo información sabiendo qué se esperaba como salida para cada caso de uso. Esto se conoce como casos de prueba. Este tipo de prueba se llama de prueba de caja negra ya que son aplicados sobre el sistema en ejecución [Jacobson, 1999] e incluye los siguientes puntos:

- Criterio para datos de entrada
- Criterio para datos de salida
- Criterio para suspensión y autorización de realización de pruebas

Cada caso de prueba está relacionado directamente con un caso de uso para verificar que el sistema cumpla desde el punto de vista funcional con el comportamiento planteado desde el diseño.

Los roles envueltos en la disciplina de pruebas juegan una parte importante en el desarrollo de esta disciplina, el esfuerzo primario se centra en los roles de administrador y analista. Las habilidades más importantes requeridas para esta disciplina incluyen negociación, planeación y estrategias de trabajo.

3.4.2.5 Administración del proyecto

Esta disciplina se encarga de controlar el tiempo de desarrollo, costos, términos y condiciones e integración con sistemas de información externos. Es muy importante no perder de vista que esta disciplina tiene relación con todas las fases, ya que representa la parte dinámica del proceso. Sin embargo, para fines de desarrollo de este proyecto en particular, sólo se encargará de controlar el tiempo de desarrollo basado en los alcances descritos asociados con un rol específico y sus actividades correspondientes, esto nos lleva a centrar la atención en la planeación del proyecto, ilustrado en la Figura 3.9. Es decir, en donde se organizan los roles, actividades, responsabilidades y productos. En la tarea de definir el plan de desarrollo de software hay que tomar varias decisiones, como el tipo de tecnología que se utilizará, la definición de riesgos, herramientas, tiempos, productos etc. La Figura 3.9 muestra un diagrama con el contenido de roles, productos y actividades que integra RUP dentro de la disciplina de administración del proyecto representando el flujo de trabajo o "workflow" de la planeación del proyecto.

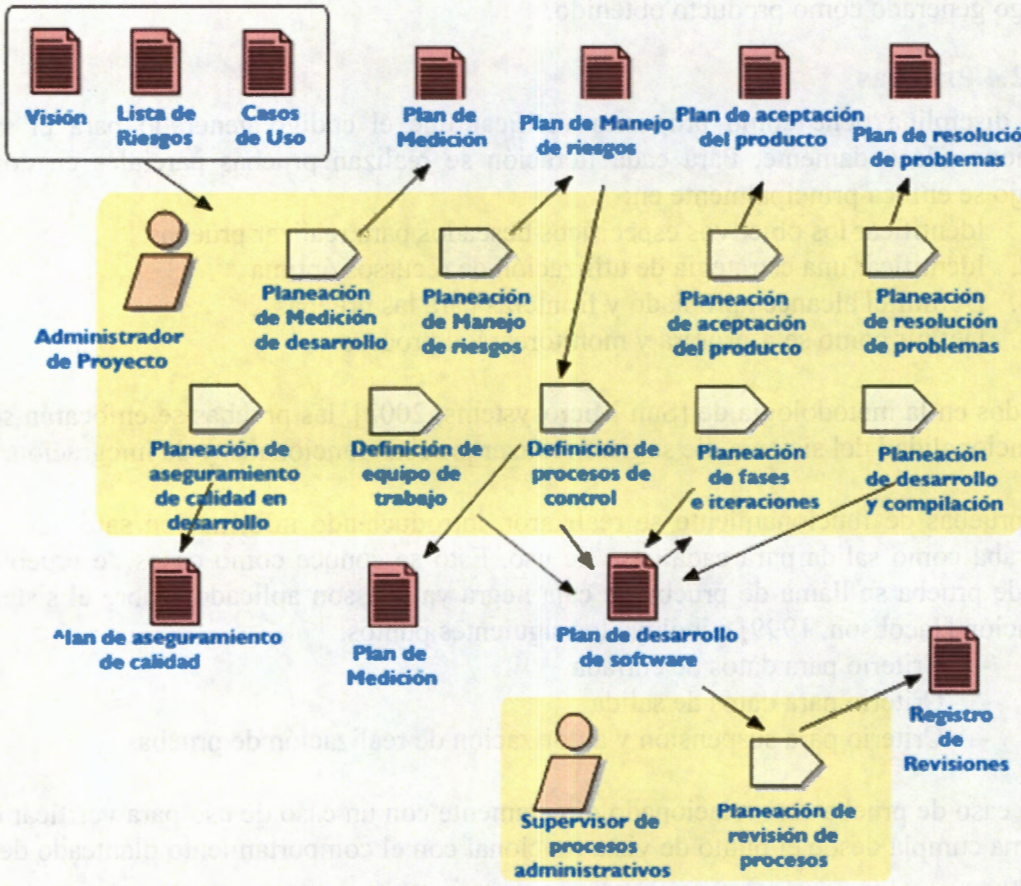

Figura 3.9 – Flujo de trabajo de la planeación del proyecto en base al RUP

La instancia seleccionada para este proyecto no incluye algunos roles, productos y actividades, como ya se ha mencionado. La ejecución de las actividades marcadas en el flujo de trabajo o "workflow" ilustrado en la Figura 3.9 dan como resultado productos específicos que a su vez integran el plan de desarrollo de software. El plan de desarrollo de

software es la base para la administración de este proyecto. En particular, la planeación del proyecto se propone que se realice tomando en cuenta los siguientes puntos de control para generar el plan de desarrollo de software:

1. Organización para el desarrollo. En este punto se define la estructura organizacional del proyecto con sus actividades, roles y responsabilidades basados en los objetivos y en su entorno para la planeación del desarrollo.
2. Plan de fases. Este plan describe los objetivos que tendrán que cubrirse al concluir cada una de las fases después de las iteraciones correspondientes. En este proyecto, dichos objetivos se basan en los logros que marca RUP para cada fase.
3. Plan de iteraciones. Este plan define el número de iteraciones que se realizarán en el proyecto. El proyecto está compuesto por un total de 6 iteraciones las cuales se describirán con detalle en la sección correspondiente dentro del siguiente capítulo.
4. Plan de revisión de la planeación. Es muy importante este plan para llevar un seguimiento de los procesos planteados así como de las modificaciones y cambios sobre los mismos derivados de alguna estrategia de mitigación de riesgos.
5. Plan de manejo de riesgos. Este plan detalla cómo manejar los riesgos asociados al proyecto. Muchas decisiones en un ciclo del proceso son manejados por Riesgos. Con el fin de disminuir las posibilidades de fracaso, es necesario prever anticipadamente los riesgos que enfrenta o pudiera enfrentar en determinado caso el proyecto y generar estrategias para mitigar o solucionar el problema que se pudiera generar.
6. Plan de administración de configuraciones y cambios. Este plan define la manera en que se llevará el control sobre los cambios de versión en software y documentación para poder llevar un mejor seguimiento del desarrollo.

Las actividades que no se tomaron en cuenta dentro de esta instancia son las siguientes:

1. Plan de aseguramiento de calidad.
2. Plan de aceptación del producto
3. Plan de resolución de problemas
4. Plan de medición

Estas actividades no se incluyeron dentro de la instancia ya que se buscó tomar un conjunto de elementos básicos pero elementales y hacer un ejercicio acotado en el uso de metodologías de desarrollo de software. En proyectos futuros podrán ser incorporados los puntos no cubiertos en esta instancia de manera natural y efectiva con la experiencia previa adquirida.

Los productos relacionados con cada actividad incluida en esta instancia se describirán en el siguiente capítulo con detalle, como resultados obtenidos al desarrollar estas actividades. Estos resultados se documentan en el mismo orden en que se listan dentro la sección del plan de desarrollo.

Como puede observarse en la Figura 3.3 el esfuerzo ejercido para la disciplina de administración es relativamente el mismo para todas las fases. Sin embargo, la definición de todos estos puntos clave en la planeación tiene un mayor esfuerzo en la fase de concepción. En el resto de disciplinas se da seguimiento y refinamiento a los planes definidos en esta primera fase.

3.4.2.6 Entorno del desarrollo

Es sumamente importante seleccionar las herramientas adecuadas que se van a utilizar para lograr el cumplimiento de las tareas asignadas a cada rol, de acuerdo con las actividades programadas para el proyecto. En este caso tenemos dos necesidades que cubrir; por un lado se requieren herramientas para la administración del proyecto y, por otro, es necesario contar con herramientas para el desarrollo del mismo. La definición de estas herramientas establece el entorno de desarrollo y se describen a continuación:

1. Herramientas para apoyar la administración del proyecto:
    a. Wiki: es una herramienta de trabajo colaborativo que funciona en línea y que permite a los usuarios crear y editar contenidos de páginas Web a través de un navegador. Esta herramienta permitirá mantener una comunicación eficiente a diario, entre asesores y alumno.
    b. GanttProject: es una herramienta de soporte para la planeación del proyecto estableciendo una organización para manejo de recursos humanos en el tiempo y tareas específicas a través de diagramas de Gantt.
2. Herramientas para apoyar el desarrollo del proyecto:
    a. Eclipse: La plataforma Eclipse se utiliza como ambiente integrado de desarrollo (IDE: Integrated Development Environment) cuya estructura está diseñada en base a "Pluggins", además que soporta más de 80 compiladores para diversas aplicaciones como: php, html, Java™, C++, entre otros.
    b. Plataforma Java 2 (J2SE): Esta plataforma en su versión estándar ofrece un ambiente completo para el desarrollo de aplicaciones en contextos muy diversos.
    c. Lenguaje de modelado unificado (UML: Unified Modelling Language): Se utiliza para modelar la estructura, comportamiento y arquitectura de aplicaciones de software. Existe un "Plug-in" para la plataforma Eclipse que permite modelar las aplicaciones y genera código en Java a partir de dicho modelo. También se utilizó una herramienta para generación de modelos y código en Java llamada StarUML.
    d. Subversión: Ofrece un mecanismo de control de versiones de una aplicación, conservando la sincronización entre versiones durante la edición de código. Existe un "Plug-in" de esta herramienta para la Plataforma Eclipse.

3.4.3 Fases

Como se mencionó en la sección 3.2.1, las fases representan el aspecto dinámico del proceso y, en este sentido, es posible plantear la ejecución del proceso en forma iterativa en donde cada iteración lleva a la generación de una versión del artefacto o producto tal y como se describe en la disciplina de planeación del desarrollo.

Cada artefacto puede tener múltiples formas incluyendo un modelo, elemento de modelo, documento o código. La asociación de las actividades con los artefactos correspondientes que deben generar se define y detalla en las fases del proceso, que se muestran a continuación.

3.4.3.1 Fase 1: Concepción (Inception)

El objetivo principal de la fase de concepción es lograr una concordancia entre las necesidades de los usuarios con los objetivos de los ciclos de vida del proyecto. En la fase de concepción se identifican los requerimientos y los posibles riesgos para evaluar si el proyecto es realizable y si es necesario, establecer planes de contingencia y mitigación. Las actividades esenciales en esta fase son formular la visión y alcance del proyecto y trabajar sobre la generación de una propuesta de arquitectura. En este sentido, el esfuerzo en las disciplinas de requerimientos y análisis y diseño es importante, de acuerdo con la instancia representada en la Figura 3.5. La parte esencial de esta disciplina es la generación de un plan de desarrollo, el cual debe contener las iteraciones necesarias para lograr la cobertura de requerimientos del sistema con base a las necesidades que dichos requerimientos plantean. En este plan se buscará cubrir los logros parciales importantes, a los cuales se les conoce como "milestones", representados a través de los artefactos y/o el avance de los mismos. En la Tabla 3.1 se muestran los artefactos que se generan en esta fase.

Tabla 3.1 – Artefactos que se deben generar para la Fase 1

Artefactos esenciales (en orden de importancia)	Estado
Visión	Los requerimientos, características clave y principales dificultades están documentados.
Lista de riesgos	Los riesgos iniciales identificados y documentados.
Plan de iteraciones	El plan de iteraciones para la siguiente fase está completo y revisado.
Infraestructura de desarrollo	Todas las herramientas que darán soporte al proyecto están seleccionadas.
Glosario	Los términos importantes están definidos.

La fase habrá terminado cuando se hayan cubierto los logros parciales importantes, esto es, que además de haber generado los artefactos mencionados anteriormente, se haya logrado lo siguiente:

- Se han capturado correctamente los requerimientos y que hay una comprensión conjunta de los mismos
- Se han determinado adecuadamente el costo y tiempo estimado, prioridades y riesgos
- Han sido identificados los riesgos y existen planes de mitigación y contingencia para ellos.

3.4.3.2 Fase 2: Elaboración (Elaboration)

La meta de la elaboración es definir la arquitectura del sistema para proveer una base estable para la implementación en la fase de construcción. Las actividades en esta fase

buscarán generar una arquitectura estable y refinar la visión y la información relacionada con ésta, como son los requerimientos por ejemplo. Por esta razón, las actividades están más relacionadas con las disciplinas de análisis y diseño con mayor esfuerzo, implementación con un esfuerzo importante y pruebas con un esfuerzo menor, lo cual toma sentido de nuevo con el diagrama de la Figura 3.5. Los artefactos que deben ser generados y/o refinados en esta fase se listan en la Tabla 3.2 mostrada a continuación.

Tabla 3.2 – Artefactos que se deben generar para la Fase 2

Artefactos esenciales (en orden de importancia)	Estado
Prototipos	Uno o más prototipos arquitectónicos han sido creados para explorar la funcionalidad crítica y los diferentes escenarios o situaciones.
Lista de riesgos	Actualizada y revisada. Nuevos riesgos son probablemente de naturaleza de la arquitectura, principalmente relacionados con el manejo de los requerimientos no funcionales.
Documento de especificaciones técnicas de construcción.	Creado y basado en los casos de uso, y en mecanismos clave
Plan de iteraciones	Plan de iteraciones para la fase de construcción completa y revisada.
Modelo de casos de uso (actores, y casos de uso)	Modelo de casos de uso completado aproximadamente al 80% identificando todos los casos de uso derivados del principal así como su descripción a partir de los requerimientos.
Especificaciones suplementarias	Requerimientos suplementarios capturando los requerimientos no funcionales documentados y revisados.

Esta segunda fase habrá terminado cuando además de haber generado los artefactos mencionados anteriormente, se haya logrado lo siguiente:

- La visión y los requerimientos son estables.
- La arquitectura es estable.
- Las pruebas y evaluación de los prototipos han demostrado que los elementos de mayor riesgo han sido encausados para su resolución.
- Los planes de iteración para la fase de construcción están suficientemente detallados para proceder y sus estimaciones tienen credibilidad.

3.4.3.3 Fase 3: Construcción (Construction)

La meta de la fase de construcción es identificar perfectamente los requerimientos restantes y completar el desarrollo del sistema siguiendo la línea de base arquitectónica. La fase de

construcción es, en cierto sentido, un proceso de manufactura en donde se hace énfasis en los recursos administrativos y operaciones de control para hacer más eficiente el tiempo y generar un sistema de calidad. Por esta razón, en esta fase, existe un mayor número de actividades relacionadas con las disciplinas de implementación y pruebas, mientras que se presentan en proporción muy baja las actividades relacionadas con las disciplinas de requerimientos y de análisis y diseño, como puede observarse en la Figura 3.5. Dichas actividades buscan la generación y/o refinación de los artefactos mostrados en la Tabla 3.3, la cual se muestra a continuación.

Tabla 3.3 – Artefactos que se deben generar para la Fase 3

Artefactos esenciales (en orden de importancia)	Estado
"El sistema"	El sistema entregable listo para la fase Beta (de pruebas).
Plan de iteraciones	Plan de Iteraciones para la fase de transición completa y revisada.

La condición básica para la conclusión de esta fase es que el sistema sea entregado en su versión beta. En esta fase no hay más objetivos parciales que la generación de los artefactos mostrados anteriormente ya que al finalizar, se obtiene la manera en que se realizará la transición con el usuario final y el producto de software en su fase beta.

3.4.3.4 Fase 4: Transición (Transition)

En esta última fase, el sistema está básicamente terminado ya que los objetivos del proyecto fueron cubiertos. Aunque todavía se deben realizar pruebas finales, en caso de necesitar interfaces externas o captura de datos e información específica, las actividades se centran en la integración y pruebas finales, que corresponden a las disciplinas de implementación y pruebas propiamente. Los artefactos que se deben generar en esta fase básicamente son el material de apoyo al usuario final y el sistema mismo completamente funcional y sin errores. En la Tabla 3.4 se muestran los artefactos que se generan en esta fase.

Tabla 3.4 – Artefactos que se deben generar para la Fase 4

Artefactos esenciales (en orden de importancia)	Estado
Material de soporte para utilización de la biblioteca	Materiales para que el usuario aprenda a usar, mostrar, operar y mantener el sistema final.

Al finalizar esta fase se estará concluyendo una versión del sistema y en caso de ser la versión final, se estará concluyendo el proyecto con su información adicional para su posterior soporte y utilización.

3.5 Sumario

En este capítulo se justificó la necesidad de implementar un proceso de desarrollo de software cuando en un proyecto se desea crear alguna aplicación bajo un esquema muy bien definido, se decidió tomar como referencia el proceso unificado de Rational complementado por la metodología de SUN Microsystems, los cuales se consideraron como marco de referencia para definir la instancia que se implementa en este trabajo de desarrollo de software para el almacenamiento en medios físicos DICOM y con ello justificar su uso al garantizar un impacto directo en los resultados obtenidos. Esto permite madurar el proceso de desarrollo utilizado en este tipo de trabajos de tesis de maestría.

3.5.1 Impacto en el uso de RUP

El impacto de la aplicación de RUP en este proyecto se plantea desde tres perspectivas. La primera se encuentra definida por los alcances del mismo, ya que ofrece una solución a las necesidades específicas de un módulo de almacenamiento en medios físicos que puede interactuar con otros sistemas, lo cual lo convierte en una biblioteca reutilizable para el desarrollo de un PACS. La segunda ofrece una organización y un control muy eficiente en la generación de artefactos y documentación tal como se describe en la instancia de RUP, representando experiencias que pueden ser reutilizadas en otros proyectos. La tercera es el impacto en el uso del RUP complementado por la metodología de SUN Microsystems, es la generación de una arquitectura de software que cumple con las necesidades no funcionales, que garantizan eficiencia y flexibilidad.

En el capítulo 4 se plantea la administración del proyecto, solución al problema y la planeación que se utilizó siguiendo la metodología aplicada con actividades organizadas por la fase correspondiente en orden cronológico de manera secuencial y haciendo referencia a los artefactos refinados durante cada iteración.

4. Administración del proyecto: Visión, alcance, plan del proyecto y seguimiento

En este capítulo se muestran los resultados correspondientes a la administración del proyecto que responden a un esfuerzo importante en la fase de concepción (Inception) y tiene una estructura definida por RUP. Los resultados correspondientes a la administración del proyecto se basan en la plantilla del documento de visión y plan de desarrollo para describir en general los alcances y dar una guía para el desarrollo del proyecto. Cabe aclarar que a lo largo de las distintas fases, la planeación presentó cambios debido a riesgos que no fueron bien cuantificados o mitigados. Por esta razón, se muestra el plan de proyecto final, mientras que los detalles que provocaron modificaciones sobre éste se describen en la última sección del capítulo.

4.1 Posicionamiento

En esta sección se hace un planteamiento con un enfoque hacia el desarrollo de la biblioteca vista como producto, en donde se plantea el problema, las características que se desea cubrir y su comparación con otros productos similares existentes.

4.1.1 Planteamiento del problema

Como se describe en la sección 1.5 el problema es que en la actualidad muy pocas aplicaciones PACS integran dentro de su funcionalidad el almacenamiento en medios físicos. Este problema afecta a los programadores que intentan implementar una solución tipo PACS para un proceso muy particular ya que les puede resultar complicado incluir la funcionalidad de almacenamiento en medios físicos.

El impacto que genera no poder almacenar en medios físicos dentro del esquema de un PACS es importante ya que se pierde la capacidad para acceso y consulta inmediata de imágenes médicas. La solución a esta problemática sería la inclusión de esta funcionalidad a través del uso de una biblioteca que sea independiente de la aplicación PACS. La biblioteca será utilizada por el desarrollador como una herramienta que le permitirá incorporar, en forma rápida, el almacenamiento en medios físicos compatible con DICOM.

4.1.2 Posicionamiento del producto

Este proyecto está dirigido a los programadores que deseen integrar el almacenamiento en medios físicos dentro de una aplicación tipo PACS y su viabilidad se basa en la inexistencia de una biblioteca que pueda adaptarse a aplicaciones compatibles a DICOM para almacenamiento en medios físicos. En la sección 2.2 puede observarse con detalle el análisis del estado del arte en donde se muestran algunas de las aplicaciones existentes y sus características.

Este proyecto corresponde al desarrollo de software para almacenamiento DICOM en medios físicos en una biblioteca y será independiente de cualquier aplicación, pero con la posibilidad de integrarse dentro de la misma de manera flexible. La flexibilidad le permitirá al desarrollador elegir el o los medios a utilizar y la plataforma en la cual se ejecutará la aplicación. Por lo tanto, para el producto resultante se plantean los siguientes requerimientos generales:

- que considere el uso de distintos medios físicos
- que sea transportable a través de distintos sistemas operativos

- que soporte especificaciones DICOM posteriores
- que soporte nuevas tecnologías en dispositivos de almacenamiento
- que ofrezca una interfaz adecuada con capacidad de integrarse a aplicaciones de manera flexible y simple.

Las partes del estándar que se requiere cubrir para lograr que el sistema cumpla con los requerimientos generales mencionados se describieron en las secciones 1.4 y 2.1. Cabe aclarar que el diseño de un modelo de almacenamiento para un PACS, como el mostrado en la Figura 1.3, no está especificado por las partes del estándar antes mencionadas. En este sentido, el almacenamiento en medios físicos considerado dentro del estándar especifica el almacenamiento local de objetos de información compuestos para ser utilizados "en línea" por algún usuario.

4.2 Participantes del proyecto (Stakeholders)

4.2.1 Sumario de participantes

Los roles de los desarrolladores participantes se describen con detalle en la sección 4.5.3, dentro de la descripción del plan de proyecto. Se ofrece a continuación una descripción general, acorde al contexto en el cual se plantea el proyecto:

- El Alumno. Este participante es el que se encarga de desarrollar la biblioteca. Sus responsabilidades abarcan desde el análisis de los requerimientos del sistema, hasta su implementación y pruebas de funcionamiento.
- Los Asesores. Estos participantes se encargan de supervisar y dirigir la ejecución correcta del proyecto. Sus responsabilidades son lograr que se cumplan los objetivos y llevar un control sobre la administración del proyecto.
- El Programador o Desarrollador (usuario de la biblioteca). Este participante será el usuario final. Su responsabilidad es integrar la biblioteca dentro de la(s) aplicación(es) tipo PACS a través de la adecuación del sistema como tal.

4.2.2 Entorno de Usuario

El entorno del usuario está vinculado a un proyecto en donde el usuario trata de cubrir una funcionalidad muy específica. En este sentido, esta biblioteca cubriría la funcionalidad correspondiente al almacenamiento en medios físicos. La biblioteca tiene como particularidad la capacidad de soportar nuevos dispositivos y diferentes sistemas operativos, por lo que también puede considerarse una opción complementar la biblioteca para cubrir una funcionalidad mayor.

Para cumplir el objetivo del usuario, la biblioteca debe ser documentada apropiadamente a través de una guía de utilización en donde se describirán todos los detalles de interfaz.

4.3 Descripción general del producto

4.3.1 Perspectiva

Esta biblioteca será utilizada por un desarrollador de aplicaciones PACS para el funcionamiento correspondiente al almacenamiento DICOM en medios físicos. De esta

forma cualquier programador podrá integrarlo a un sistema más complejo de manera transparente y completamente funcional según sean las necesidades de un hospital.

Bajo esta perspectiva, los sistemas PACS podrán desarrollarse a través de bibliotecas y así cubrir las necesidades específicas de cada área de radiología en los hospitales.

4.3.2 Dependencias

Una dependencia importante que tiene este producto es el manejo físico de dispositivos que requiere de manejadores particulares. De esta forma, para tener acceso al CD y DVD, se requiere de manejadores desarrollados para esos fines, considerando además que existen versiones para Windows, Linux y Macintosh.

4.3.3 Necesidades y Características

El requisito más general que se debe considerar es la compatibilidad con el estándar DICOM. En este sentido, algunas características planteadas para el producto a generar tienen que ver con la flexibilidad y extensibilidad. Para garantizar estas características hay que cubrir las necesidades de acoplamiento especificadas para cada medio físico, en la parte 10 del estándar DICOM.

4.3.4 Alternativas y Competencia

Como puede observarse en la sección 2.2 correspondiente al análisis del estado del arte, hay muy pocos sistemas que integran dentro de su funcionalidad el almacenamiento en medios físicos, mientras que los sistemas que resuelven el almacenamiento en medios físicos están muy limitados, ya sea por el sistema operativo o por el número de medios que soportan. Por ejemplo, Osirix cuenta con almacenamiento en medios físicos, pero no puede ser implementado en otra plataforma que no sea con sistema operativo Macintosh. Por otro lado, pixelmed que es el conjunto de bibliotecas más completo de todos por la cobertura de partes DICOM, no cuenta con la funcionalidad para almacenamiento en medios físicos. Por estas razones, la alternativa ofrecida a través de este proyecto es viable. No hay, hasta el momento, un producto desarrollado con la perspectiva planteada.

4.4 Otros requerimientos

Para la realización de este proyecto se requiere de la interacción de varias de las especificaciones DICOM, además de las involucradas en almacenamiento. En este sentido, se requiere de implementaciones de estas partes como son el modelo de información, diccionario de datos y estructuras de datos y codificación. La implementación de estas partes queda fuera del alcance de este proyecto, como se plantea en la siguiente sección.

Una vez concluido este proyecto, se requiere de una documentación adecuada desde dos perspectivas:

1. Un manual técnico que documente los aspectos más importantes de las fases del desarrollo.
2. Un manual de usuario que será utilizado por cualquier desarrollador que desee utilizar el producto.

4.5 Alcance del proyecto

Como se observa en la Tabla 4.1 y como se describió en la sección 1.4.2, las especificaciones DICOM involucradas en este proyecto son las establecidas en las partes 4,10, 11, 12, 14, 15 y 16. Sin embargo, las especificaciones correspondientes a la parte 4 no son consideradas en el desarrollo de este proyecto. Así, la implementación de esas especificaciones y partes asociadas deben implementarse por separado o utilizar alguna ya existente.

Tabla 4.1 – Cobertura del proyecto con relación a las partes del estándar DICOM

Nombre	Partes DICOM	Descripción	Estatus
Biblioteca de Almacenamiento DICOM en medios físicos	Parte 4	Especificación de clases de servicio.	No cubierto (corresponde al modelo de información)
	Parte 10	Almacenamiento en Medios y Formato de Medios para Intercambio de Datos	Cubierto
	Parte 11	Perfiles de Aplicación de Medios de Almacenamiento	Cubierto
	Parte 12	Formatos de Medio y Medios Físicos para Intercambio de Datos	Cubierto
	Parte 14	Función de Despliegue de Escala de Grises	No cubierto (no está dentro del alcance de este proyecto)
	Parte 15	Seguridad y Perfiles de manejo de Sistema	No cubierto (no está dentro del alcance de este proyecto, sin embargo, está contemplado en la arquitectura)
	Parte 16	Recurso de Mapeo de Contenido	No cubierto (no está dentro del alcance de este proyecto, sin embargo, está contemplado en la arquitectura)

4.6 Planeación de proyecto

Estas actividades cubren el contenido de las partes clave de la administración del proyecto. Las siguientes sub-secciones describen los productos generados a partir de las actividades de planeación definidas en la sección 3.4.2.5. Lo que se documenta en las siguientes subsecciones son los productos que se fueron logrando.

4.6.1 Plan de desarrollo de software

Este plan es un artefacto que reúne toda la información requerida para administrar el proyecto. Engloba distintos artefactos desarrollados durante la fase de concepción y es refinado a lo largo del proyecto.

4.6.1.1 Organización para el desarrollo

Los roles son la clave para organizar este proyecto, ya que cada uno tiene una o varias actividades específicas que fueron descritos en la sección 3.4.1 y abarcan actividades muy bien definidas para cada fase. Los roles clave en este proyecto son los siguientes:

- Administrador de proyecto. Este fue un rol compartido entre los asesores con la responsabilidad de sugerir otras opciones y autorizar cambios según los alcances y objetivos planteados.
- Analista de Requerimientos. Este rol fue ejecutado por el alumno, con la responsabilidad de capturar los requerimientos o necesidades del sistema basados en el estándar DICOM así como en la metodología a emplear.
- Arquitecto de Software. Este rol fue ejecutado por el alumno, con la responsabilidad de generar una arquitectura de software consistente con los requerimientos y objetivos del proyecto.
- Desarrollador. Este rol fue ejecutado por el alumno, con la responsabilidad de generar el código guiado por los casos de uso y la arquitectura propuesta.
- Pruebas. Este rol fue ejecutado por el alumno, con la responsabilidad de probar que el código generado tuviera funcionalidad completa y consistente.

En este proyecto, la finalización de cada fase tiene productos muy bien definidos, los cuales serán descritos como hitos. Un hito es un suceso o acontecimiento que sirve como punto de referencia [Becker, 2007] y en la Figura 4.1 pueden identificarse como pequeños rombos y están compuestos por las siguientes partes:

- El sistema o parte de código correspondiente (dependiendo de la fase)
- Modelo de implementación
- Material de guía (documentación)
- Modelo de diseño

4.6.1.2 Plan de fases

Este plan muestra la Estructura de Descomposición del Trabajo (EDT, o WBS por sus siglas en inglés: work breakdown structure), el diagrama de gantt del proyecto y los milestones que se buscan obtener al finalizar cada una de las fases después de las iteraciones correspondientes. En la siguiente sub-sección se describe el plan de iteraciones.

En administración de proyectos, una EDT es una estructura jerárquica y descendente formada por los entregables y las tareas necesarias para completar un proyecto. La EDT sirve como la base para la planeación del proyecto. Todo trabajo que será realizado debe poder rastrear su origen en una o más entradas de la EDT [Wikipedia.org - EDT, 2007].

El resultado de aplicar la técnica de EDT utilizada en este proyecto puede observarse en la Figura 4.1 con la raíz de cada actividad marcada por los títulos principales (en las tablas de la izquierda) y sus actividades secundarias marcadas por los puntos separados por viñetas.

Cabe aclarar que las actividades secundarias definidas para la escritura en CD son las mismas para la escritura, lectura y actualización para CD, DVD, HD y USB. En este diagrama puede observarse la división de actividades por tipo de medio, así como la integración y pruebasdon de se incluye la realización de la tesis.

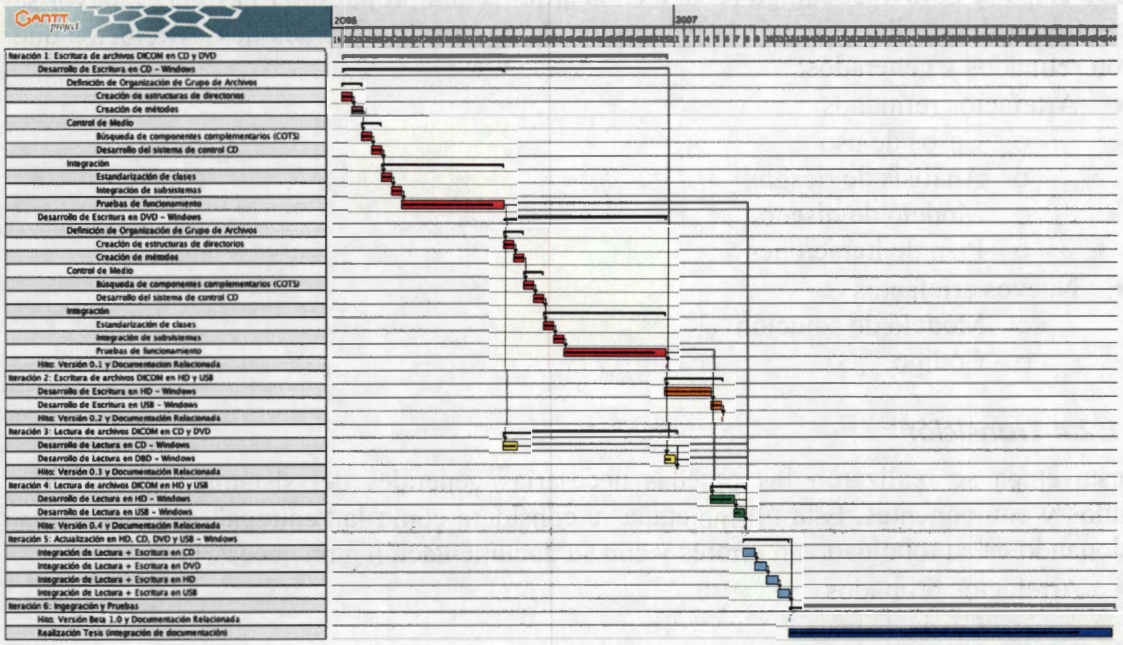

Figura 4.1 - Diagrama de Gantt de las 6 iteraciones

4.6.1.2.1 Concepción

En esta etapa se estudiaron los alcances del proyecto, se analizaron los riesgos de su realización, se realizó una planeación global y se preparó un entorno de soporte. A continuación se muestran los artefactos principales generados en esta etapa.

- Definición de los alcances del proyecto
- Descripción general del producto
- Plan del proyecto
- Glosario con la definición de términos relevantes relacionados con el proyecto
- Documento de especificación de requerimientos (SRS)

4.6.1.2.2 Elaboración

En esta etapa se definió una arquitectura de software basada en los requerimientos y modelos de diseño. Los artefactos o productos que se generaron en la fase anterior sufrieron cambios de mejoras y adecuaciones, a los cuales se les nombró refinaciones, además se muestran a continuación los nuevos artefactos:

- Artefactos refinados
  - Casos de uso
  - Análisis de riesgos
  - Plan de desarrollo de software
- Nuevos artefactos
  - Modelo de diseño

- Modelo de arquitectura
- Plan de iteraciones

4.6.1.2.3 Construcción

En esta etapa se obtuvo el código basado en la arquitectura propuesta. El esfuerzo mayor se concentró en desarrollar y seguir la planeación de desarrollo e iteraciones. Artefactos que fueron refinados y generados:

- Artefactos refinados
  - Casos de uso
  - Análisis de riesgos
  - Modelo de diseño
  - Plan de iteraciones
- Nuevos artefactos
  - Modelo de solución
  - Código

4.6.1.2.4 Transición

En esta etapa se realizaron las pruebas necesarias generales del sistema para integrar módulos y componentes. Esta última etapa se considera como la "entrega" del sistema ya funcionando en su totalidad, sin errores y con su documentación guía necesaria.

- Artefactos refinados
  - Casos de uso
  - Análisis de riesgos
  - Arquitectura del sistema
  - Modelo de diseño
  - Código
- Nuevos artefactos
  - El sistema

4.6.1.3 Plan de Iteraciones

Para este proyecto se definieron 6 iteraciones las cuales, en conjunto, conforman el desarrollo completo. Las iteraciones se definen en relación a las funciones y metas o hitos que se plantean en la visión y alcance, mismas que pueden identificarse en la Figura 4.2 con mayor detalle y con la referencia en el tiempo de cada iteración en un color diferente en la Figura 4.1. Las iteraciones se definieron en la cantidad mencionada debido a la identificación de algunas similitudes, entre los medios CD y DVD entre el USB y disco duro, así como similitudes entre funcionalidad para escritura, lectura y actualización, en donde esta última es una combinación ordenada de las dos primeras. Durante la iteración 1 se enfocó el esfuerzo en lograr el objetivo de realizar escritura de archivos DICOM en los medios físicos correspondientes a CD y DVD. Para la iteración 2 se enfocó el esfuerzo en el mismo fin pero para los medios USB y disco duro. Las iteraciones 3 y 4 centralizaron sus objetivos en dar funcionalidad a la biblioteca de software para la lectura de CD y DVD así como para USB y disco duro respectivamente. La funcionalidad para actualizar la información DICOM en los cuatro medios físicos se fijó como objetivo principal en la iteración número 5 mientras que la integración de la funcionalidad total obtenida de las iteraciones anteriores fue el objetivo principal de la iteración número 6 y última.

Iteración 1: Escritura de archivos DICOM en CD y DVD
Desarrollo de Escritura en CD - Windows
Definición de Organización de Grupo de Archivos
Creación de estructuras de directorios
Creación de métodos
Control de Medio
Búsqueda de componentes complementarios (COTS)
Desarrollo del sistema de control CD
Integración
Estandarización de clases
Integración de subsistemas
Pruebas de funcionamiento
Desarrollo de Escritura en DVD - Windows
Definición de Organización de Grupo de Archivos
Creación de estructuras de directorios
Creación de métodos
Control de Medio
Búsqueda de componentes complementarios (COTS)
Desarrollo del sistema de control CD
Integración
Estandarización de clases
Integración de subsistemas
Pruebas de funcionamiento
Hito: Versión 0.1 y Documentacion Relacionada
Iteración 2: Escritura de archivos DICOM en HD y USB
Desarrollo de Escritura en HD - Windows
Desarrollo de Escritura en USB - Windows
Hito: Versión 0.2 y Documentación Relacionada
Iteración 3: Lectura de archivos DICOM en CD y DVD
Desarrollo de Lectura en CD - Windows
Desarrollo de Lectura en DBD - Windows
Hito: Versión 0.3 y Documentación Relacionada
Iteración 4: Lectura de archivos DICOM en HD y USB
Desarrollo de Lectura en HD - Windows
Desarrollo de Lectura en USB - Windows
Hito: Versión 0.4 y Documentación Relacionada
Iteración 5: Actualización en HD, CD, DVD y USB - Windows
Integración de Lectura + Escritura en CD
Integración de Lectura + Escritura en DVD
Integración de Lectura + Escritura en HD
Integración de Lectura + Escritura en USB
Iteración 6: Ingegración y Pruebas
Hito: Versión Beta 1.0 y Documentación Relacionada
Realización Tesis (integración de documentación)

Figura 4.2 - Estructura de actividades del diagrama de Gantt

La identificación de necesidades, la propuesta de la arquitectura y modelado en casos de uso ayudan a generar un esqueleto del sistema, la construcción e integración de cada función al sistema se realizaron parcialmente para lograr el objetivo planteado en cada una de las iteraciones. Esto significa que para cada iteración se realizó una vuelta completa del proceso pasando por todas las fases y ejecutando todas las disciplinas. En el siguiente

capítulo se muestran los resultados finales obtenidos en la última iteración; sin embargo, es importante mencionar que para llegar a cada uno de esos resultados se tuvieron que hacer 6 refinaciones por cada fase. Cabe mencionar también que, durante cada una de estas fases, se fue generando documentación para integrar la guía de uso del sistema y este documento de tesis.

4.6.1.4 Plan de revisiones al plan de proyecto

Este plan de revisiones consistió en revisiones periódicas semanales. Cada revisión generó un resumen con puntos muy claros para asegurar el seguimiento, cobertura de alcances y dirección del proyecto. El contenido de dichos resúmenes se basó en los siguientes puntos:

1. Temática
2. Actividades
3. Discusión
4. Acuerdos
5. Conclusiones

En los casos en que así era requerido las revisiones también se agregaron a los resúmenes para modificaciones o autorizaciones sobre el plan de desarrollo. Esta información se encuentra dentro del Wiki del servidor UAMISoft en la sección de "resumen de reuniones". La documentación relacionada con este proyecto dentro del Wiki de UAMISoft se encuentra disponible, solicitando acceso al departamento de Informática Médica, en línea en la dirección de Internet http://uamisoft.izt.uam.mx/uamisoft/doku.php

4.6.1.5 Plan de manejo de riesgos

El plan de manejo de riesgos se planteó con base en una plantilla, que contiene los siguientes elementos:

- Categoría: El tipo de riesgo (de planeación, técnico, etc....)
- Estatus: El estatus del riesgo (Activo o inactivo)
- Evento de riesgo: El riesgo en sí mismo
- Motivantes del evento de riesgo: Cosas en el entorno que llevan a pensar que el riesgo puede ocurrir
- Pevt: Probabilidad de que ocurra el evento de riesgo
- Evento de impacto: Descripción del impacto que tendría la ocurrencia del riesgo
- Motivantes del evento de impacto: Cosas en el entorno que llevan a pensar que el impacto sería el que se describe
- Pimp: Probabilidad del impacto
- Plan de prevención: Las acciones que se deberán seguir para prevenir el riesgo en cuestión
- Plan de contingencia: Las acciones que se deberán seguir para corregir el rumbo si el riesgo se convierte en evento.

Existen distintos tipos de riesgos que pueden catalogarse en tecnológicos, de recursos, de habilidades o de requerimientos, en donde éste último se refiere a que no hubo un buen

entendimiento y, consecuentemente se capturan de manera errónea los requerimientos, los de habilidades a que no haya un dominio sobre las herramientas que se usarán durante el desarrollo, los de recursos que no se cuente con equipo o personal para realizar el proyecto y, por último, los tecnológicos que se refieren a la falta de herramientas adecuadas para el desarrollo.

En este proyecto se consideraron los riesgos sugeridos por el RUP. Cabe mencionar, que durante el desarrollo de proyecto el estado de cada riesgo cambió ya sea que haya disminuido o aumentado, incluso pueden aparecer nuevos riesgos. En la administración de este proyecto se consideraron:

- Riesgos Tecnológicos.
  - La comunicación con el Sistema Operativo para grabar archivos en medios físicos implica integrar una función a través de software que se encargue de controlar los medios por lo que en esta etapa se tuvo que considerar invertir un intervalo de tiempo grande. El riesgo es que el manejar archivos a través del sistema operativo resultara más complicado de lo previsto. Este riesgo se documentó como se muestra en la Figura 4.3. El plan de prevención incluyó el generar un análisis detallado de las características y necesidades de Windows y Linux para el manejo de dispositivos de almacenamiento, mientras que el plan de contingencia consistió en generar un tiempo considerable para los detalles para así no salir del plan original.

Identificador de Riesgo	Dueño	Fecha Apertura	Categoría	Estatus
T1 - Tecnológico	Juan Salgado	19-jun-06	Construcción	Inactivo
Evento		**Impacto**	**Pevt**	**Pimp**
La comunicación con el sistema operativo para grabar archivos implica integrar una función a través de software que se encargue de controlar los medios.		El manejo de archivos a través del sistema operativo resulta más complicado de lo previsto	30%	100%
Motivantes de evento	**Plan de prevención**	**Motivantes de impacto**	**Plan de contingencia**	
1.- Java no cuenta con clases para interacción directa con dispositivos 2.- No se conoce la forma exacta en que se solucionará este punto	Hacer un análisis detllado de las caracteristicas y necesidades especificas de Windows y Linux en cuanto al manejo de dispositivos físicos.	No se tiene información de la forma en que se tiene acceso a los dispositivos de almacenamiento.	Se puede asignar un tiempo corto para que se termenien los detalles y así no salirse de la planeación original.	

Figura 4.3 - Riesgos Tecnológicos

  - La comunicación con la Interfaz del Modelo de Información DICOM se simulará enviando datasets ya generados suponiendo que en un futuro sea en realidad el módulo MID el que los envíe. El riesgo es que esta interfaz no esté lista cuando se finalice el desarrollo de esta función y no se puedan realizar pruebas reales, no simuladas. Este riesgo se documentó como se muestra en la Figura 4.4. El plan de prevención consistió en no cargar demasiado las funciones que no corresponden al alcance de este proyecto, específicamente del modelo de información DICOM para no extender el tiempo de desarrollo.

Identificador de Riesgo	Dueño	Fecha Apertura	Categoría	Estatus
T2 - Tecnológico	Juan Salgado	19-jun-06	Construcción	Inactivo
Evento		**Impacto**	**Pevt**	**Pimp**
La comunicación con la interfaz de modelo de información DICOM se simulará enviando datasets suponiendo que en un futuro sea en realidad el módulo MID el que los envíe		La Calendarización se extiende	30%	100%
Motivantes de evento	**Plan de prevención**	**Motivantes de impacto**	**Plan de contingencia**	
No se cuenta con el módulo de MID implementado	No cargar demasiado las funciones que no corresponden al alcance de este proyecto	No es fácil generar un dataset para enviar como simulación	Pasar al siguiente objetivo y dejar este para el final	

Figura 4.4 - Riesgos Tecnológicos

- Riesgos de Recursos
  - El proyecto se realiza por un estudiante de posgrado, el cual tendrá que adoptar los roles especificados por la instancia del RUP. El riesgo es que el estudiante tenga que invertir más tiempo del previsto para conocer las tareas y responsabilidades específicas de cada rol. Este riesgo se documentó como se muestra en la Figura 4.5. El plan de prevención consistió en identificar de manera muy clara los roles requeridos en cada iteración para no incrementar las actividades. El plan de contingencia consistió en hacer ajustes sobre las actividades básicas y prioritarias para no extender el tiempo de desarrollo pero cuidando la calidad del mismo.

Identificador de Riesgo	Dueño	Fecha Apertura	Categoría	Estatus
R1 - Recursos	Juan Salgado	19-jun-06	Construcción	Inactivo
Evento		**Impacto**	**Pevt**	**Pimp**
El proyecto se realiza por un estudiante de posgrado, el cual tendrá que adoptar todos los roles		La calendarización se extiende	30%	100%
Motivantes de evento	**Plan de prevención**	**Motivantes de impacto**	**Plan de contingencia**	
Es un solo alumno asignado al proyecto	No cargar demasiado las actividades que no sean necesarias	No se tiene experiencia previa en el RUP, sólo en el proceso unificado	Hacer ajustes sobre las actividades básicas y prioritarias	

Figura 4.5 - Riesgos de Recursos

- Riesgos de Habilidades
  - En este punto se considera un riesgo mediano el que no se cuente con el conocimiento sobre algunos temas avanzados dentro del lenguaje Java. Este riesgo se documentó como se muestra en la Figura 4.6. Se tuvo que mitigar este riesgo disponiendo tiempo para analizar los requerimientos de clases, haciendo una búsqueda de las clases funcionales para el caso específico y calculando el tiempo en que se tienen que conocer las clases requeridas para el fin deseado. El plan de contingencia consistió en aumentar los cursos avanzados por parte de los asesores sobre java para garantizar efectividad.

Identificador de Riesgo	Dueño	Fecha Apertura	Categoría	Estatus
H1 - Habilidades	Juan Salgado	19-jun-06	Construcción	Inactivo
Evento		**Impacto**	**Pevt**	**Pimp**
El alumno no cuenta con el conocimiento sobre algunos temas avanzados dentro del lenguaje de Java		La calendarización se extiende	30%	100%
Motivantes de evento	**Plan de prevención**	**Motivantes de impacto**	**Plan de contingencia**	
Se conoce el desarrollo orientado a objetos pero no tiene experiencia amplia sobre algunos puntos identificados en el lenguaje Java	Revisar tutoriales similares para ampliar el dominio de Java y planear clases avanzadas con sus asesores	No se tiene información de proyectos similares para tener como referencia	Aumentar los cursos avanzados sobre Java para garantizar efectividad	

Figura 4.6 - Riesgos de Habilidades

- Riesgos de Requerimientos
  - Este riesgo es relativamente bajo ya que los requerimientos se sacan directamente del estándar DICOM, sin embargo, al ser un estándar muy complejo, no debe restarse importancia. Este riesgo se documentó como se muestra en la Figura 4.7. El plan de prevención consistió en realizar una revisión de los requerimientos antes de comenzar cada iteración para asegurarse de que cumpliera con el estándar. El plan de contingencia consistió en invertir el tiempo necesario para la captura de requerimientos en caso de no cubrir lo descrito en el estándar.

Identificador de Riesgo	Dueño	Fecha Apertura	Categoría	Estatus
R1 - Requerimientos	Juan Salgado	19-jun-06	Requerimientos	Inactivo
Evento		**Impacto**	**Pevt**	**Pimp**
Los requerimientos no son identificados en su totalidad		El proyecto sufre modificaciones o se extiende el tiempo de desarrollo	30%	100%
Motivantes de evento	**Plan de prevención**	**Motivantes de impacto**	**Plan de contingencia**	
El estándar es difícil de interpretar	Realizar una revisión a los requerimientos antes de comenzar cada iteración	El método de captura de requerimientos es nuevo para el alumno	Invertir el tiempo que sea necesario para captura de requerimientos	

Figura 4.7 - Riesgos de Requerimientos

4.6.1.6 Plan de administración de configuraciones y cambios

Este plan se basó en un análisis del plan de desarrollo una vez en ejecución, es decir, a lo largo del proyecto de desarrollo. En el caso en que se identificara algún requerimiento para modificar el plan original de desarrollo, el alumno debía presentar la solicitud de autorización con una propuesta para modificar el plan de desarrollo.

Las configuraciones y cambios en los resultados de productos totales o parciales de este proyecto pudieron ser en código, modelos o documentación. Cada uno de estos productos se asocia en convergencia como uno o más hitos, los cuales se encuentran referenciados en la Figura 4.1.

Como se definió en la sección 3.4.2.6, para llevar un seguimiento de la documentación y sus cambios se utilizó un Wiki instalado en el servidor de UAMISoft, en donde se siguió una convención basada en la metodología a manera de bitácora. Los cambios son guardados automáticamente y es posible revisar versiones anteriores.

Para el caso del código y modelos se utilizó Eclipse. Los modelos se generaron en StarUML ya que permite llevar trazabilidad, sin embargo la versión más reciente fue asociada al proyecto de Eclipse en su fase correspondiente. Esta asociación permitió llevar un control de versión sobre el proyecto total a través de SVN o subversión, instalado en el mismo servidor de UAMISoft

4.7 Seguimiento del proyecto

El propósito de esta sección es hacer referencia al seguimiento del plan de proyecto a un nivel más puntual y de manera global. A diferencia del seguimiento de la planeación del proyecto, descrito en la sección 4.1.6.4, cuyo fin es cumplir lo más exacto posible el plan descrito para el desarrollo del proyecto, el seguimiento del proyecto incluye dentro de sus objetivos la supervisión de la metodología y de la revisión de productos y resultados.

Las revisiones correspondientes a los productos y resultados como complemento para la revisión del proyecto entero se diseñaron con base en su función, ligadas directamente con un cada hito definido en el ETD en la Figura 4.1 representados por rombos. La revisión de estos hitos se realizó en los puntos marcados en el calendario buscando cubrir la funcionalidad de cada hito para poder avanzar con el proyecto. Además de revisar la efectividad y cumplimiento de la planeación, se incluyó la revisión de cumplimiento de hitos y metodología dentro del Wiki de UAMISoft siguiendo la estructura descrita en la sección 4.6.1.4.

A través de este seguimiento es posible demostrar cómo evolucionó el plan inicial del proyecto para poder llegar al mostrado en la sección anterior. En primera instancia, se logró presentar un plan de desarrollo basado en los alcances definidos en la sección 1.5. No obstante, en esta etapa no se realizó una valoración correcta sobre los riesgos, razón por la cual los tiempos tuvieron que ser modificados. Cabe aclarar que los cambios que se hicieron sobre la planeación, afectaron únicamente sobre el tiempo, no sobre los alcances, logros o productos que se marcaron como metas u objetivos. Los cambios principales se debieron a que no hubo una buena mitigación sobre los siguientes riesgos:

- Dominio de Java (Riesgo de habilidades). Este problema se derivó de la falta de experiencia y provocó un cambio importante en entrega de algunas partes del código, sobre todo las relacionadas con el acceso a los medios físicos como tal, ya que Java propiamente no tiene la capacidad hasta el momento.
- Recursos (Riesgo de recursos). Este problema se derivó de la falta de tiempo al ser un solo recurso con actividades diversas y provocó un cambio importante en la entrega de resultados planeada inicialmente.
- Interfaz MID (Riesgo tecnológico). Esta biblioteca de la cual depende el almacenamiento en medios físicos se pensaba que pudiera implementarse de manera paralela a este proyecto, por lo que se planearon pruebas al final con esta dependencia que no pudieron ser realizadas y provocaron un cambio muy importante en la disciplina de pruebas.

Habrá que mencionar que se comprobó que el control de riesgos puede evitar prolongación en los periodos de tiempo del desarrollo. En un principio del desarrollo, por falta de

experiencia, no se le dio la importancia requerida y hubo consecuencias. Esto significa que, en varias ocasiones, se tuvo que poner en marcha el plan de prevención y en más de una ocasión el plan de contingencia. Seguido de esta experiencia se tuvo más cuidado con los riesgos y el resultado fue un mejor control sobre la planeación y ejecución del proyecto.

4.8 Sumario

En este capítulo se planteó el problema desde el punto de vista de necesidades de administración del proyecto. Las necesidades de implementación nos permitieron seguir controladamente el desarrollo a partir de la planeación que se ajustó a la realidad en cuanto a posibilidades de cobertura de objetivos y que a la vez obedece las reglas planteadas en el proceso de desarrollo RUP.

Con una descripción de la planeación del proyecto es posible presentar los resultados con un orden relativamente fácil de comprender y, con el conocimiento previo de los resultados esperados es posible hacer comparación y análisis. En el siguiente capítulo se mostrarán los resultados siguiendo el orden de las fases con sus hitos y/o productos así como los riesgos y roles que se tomaron, aclarando que se documentan los productos en su versión final o refinada después de sus 6 iteraciones correspondientes.

5. Resultados: Análisis, Diseño, Implementación y Pruebas.

En este capítulo se documentan los resultados ordenados por fases. Como se menciona en el capítulo anterior, cada fase puede tener varias iteraciones, sin embargo, el resultado mostrado es el logrado luego de una depuración al finalizar cada una de estas.

Para mostrar la correlación del plan de desarrollo de software y los resultados de cada fase, se mostrarán como notas de referencia los puntos de correspondencia con cada una de las iteraciones.

5.1 Fase 1: Concepción

Durante el desarrollo de este proyecto se fueron obteniendo resultados parciales, algunos como productos específicos y otros como parte de otro producto. En esta fase fueron creados y refinados en las fases subsecuentes los productos mencionados en la sección 3.4.3. De estos productos, la definición de los alcances del proyecto se encuentra descrita en la sección 4.5, la descripción general del producto se encuentra documentada en la sección 4.3, el plan del proyecto se encuentra documentado en la sección 4.6 y el glosario con la definición de términos relevantes relacionados con el proyecto se muestran como apéndice al final del documento. La razón de que algunos de estos resultados se encuentren documentados en el capítulo anterior es que corresponden a los resultados relacionados con la planeación y administración del proyecto. El producto que resta por documentar de la lista mostrada en la sección 3.4.3 es el documento SRS, cuya definición puede consultarse en la sección 3.4.2.1. A continuación se describirán los Requerimientos que se obtuvieron para generar el SRS después del análisis realizado sobre el estándar DICOM.

5.1.1 Requerimientos

Acorde a la sección 3.4.2.1, la captura de requerimientos se realizó a partir de las partes 10, 11 y 12 del estándar DICOM. Las siguientes secciones plantean la especificación de los requerimientos siguiendo el estándar (NASA-STD-2100-91) propuesto por la NASA [Wilson, 1991]. En el apéndice A se muestra un resumen de este estándar con más detalle.

5.1.1.1 Requerimientos funcionales (FR´s)

Para definir los requerimientos funcionales, hubo que comprender lo que se requería que realizara el sistema. En este caso, se transformaron a requerimientos las especificaciones para el almacenamiento DICOM en medios físicos (ver sección 2.1.2).

El almacenamiento en medios físicos inicia una vez que se manda la imperativa desde la interfaz de usuario para almacenar en los medios físicos disponibles, también corresponde a esta biblioteca validar los medios disponibles para almacenamiento y establecer un método de escritura y recuperación de datos dependiendo del medio mismo. Cabe recordar que la creación de las SOP´s necesarias no le corresponde a ésta biblioteca, solo la interpretación de dichas SOP´s a través de una interfaz.

Cada perfil de aplicación para almacenamiento en medios físicos puede tomar tres roles:

1. FSC (creación de grupos de archivos),
2. FSR (lector de grupos de archivos) y
3. FSU (actualizador de grupos de archivos).

Los dos primeros roles tienen funciones diferentes y especificas, y el tercer rol incluye funciones de los dos anteriores. La definición de estas actividades basadas en el rol que contiene un conjunto de servicios específicos genera un flujo de trabajo, el cual se esquematiza en la Figura 5.1.

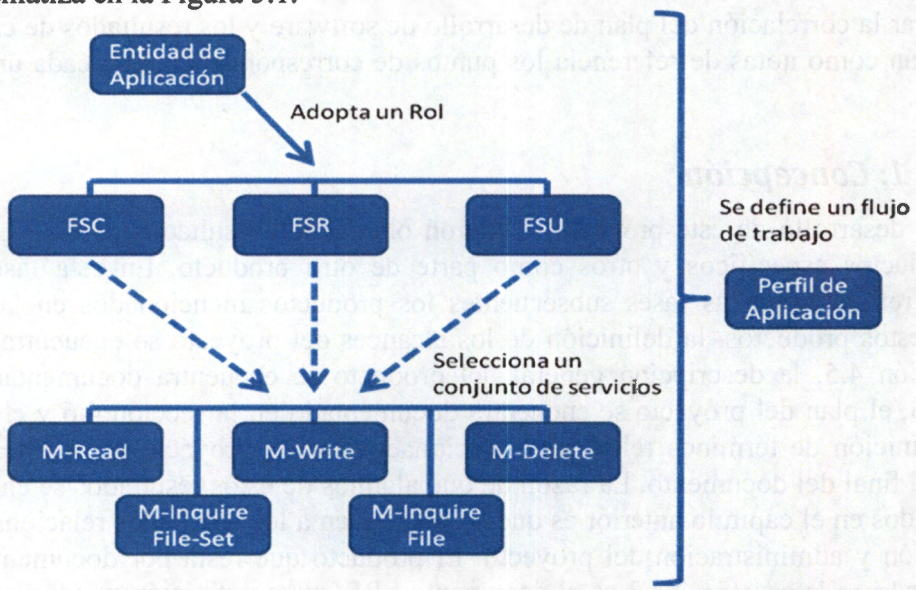

Figura 5.1 – Diagrama del flujo de trabajo para generar un perfil de aplicación

Para ejecutar cualquiera de estos servicios DICOM se requiere que las clases SOP estén disponibles. Esto significa que debe hacer uso del modelo de información DICOM para cubrir las necesidades de los medios para almacenamiento en medios físicos y se traduce a:

1) Petición de servicio por parte de la entidad de aplicación
2) Creación de SOP´s específicas para almacenamiento en medios (interfaz MID)

La especificación de requerimientos realizada con base en el estándar DICOM que obedece a la descripción del proceso descrita anteriormente y detallada en la sección 2.1.2 se muestra a continuación:

La entidad de aplicación **debe** seleccionar un tipo de operación entre FSC, FSR y FSU:

1. Basado en la conformación de un FSC, una implementación de la clase de servicio de almacenamiento en medios:
   1.1. **debe** crear un archivo DICOMDIR que contiene la clase SOP del directorio del medio de almacenamiento para el grupo de archivos creados invocando la operación M-WRITE con sus instancias SOP (que deben conocer los requerimientos de su IOD correspondiente).
   1.2. El sistema **debe** definir una selección de opciones en las diferentes capas del modelo de medios de almacenamiento que sean aplicables a una necesidad específica (Perfil de Aplicación para almacenamiento).
      1.2.1. Se **debe** describir la necesidad que se desea cubrir por el perfil de aplicación (ejemplo, ecocardiografía, angiografía, etc.) y su contenido de aplicación.
      1.2.2. Se **deben** seleccionar los IOD´s y sus SOP´s asociadas en la capa de formato de Datos.

1.2.3. Se **debe** seleccionar una definición de un formato de medios específico.

1.2.3.1. Se **debe** realizar un mapeo de datos DICOM al formato de Medio para cada uno de los medios de almacenamiento, los cuales se describen a continuación:

1.2.3.1.1. Para Disco Duro mediante el acceso a los archivos de sistema de la PC por S.O.

1.2.3.1.1.1. Se **debe** realizar un Mapeo de grupo de archivos del formato DICOM al formato específico del medio físico.

1.2.3.1.1.2. Se **debe** realizar un Mapeo de ID de archivos (nombre de archivos y DICOMDIR) del formato DICOM al formato específico del medio físico.

1.2.3.1.1.3. El perfil de aplicación **debe** proporcionar la Información de manejo de archivos DICOM al medio físico.

1.2.3.1.2. Para CD basado en las reglas de sintaxis de transferencia indicadas en el estándar ISO 9960.

1.2.3.1.2.1. Se **debe** realizar un Mapeo de grupo de archivos del formato DICOM al formato específico del medio físico.

1.2.3.1.2.2. Se **debe** realizar un Mapeo de ID de archivos (nombre de archivos y DICOMDIR) del formato DICOM al formato específico del medio físico.

1.2.3.1.2.3. El perfil de aplicación **debe** proporcionar la Información de manejo de archivos DICOM al medio físico.

1.2.3.1.3. DVD basado en las reglas de sintaxis de transferencia indicadas en los estándares ISO 9960 y UDF (universal disk format).

1.2.3.1.3.1. Se **debe** realizar un Mapeo de grupo de caracteres de configuración para validación de transferencia.

1.2.3.1.3.2. Se **debe** realizar un Mapeo de grupo de archivos del formato DICOM al formato específico del medio físico.

1.2.3.1.3.3. Se **debe** realizar un Mapeo de ID de archivos (nombre de archivos y DICOMDIR) del formato DICOM al formato específico del medio físico.

1.2.3.1.3.4. El perfil de aplicación **debe** proporcionar la Información de manejo de archivos DICOM al medio físico.

1.2.3.1.4. Para medios removibles tipo USB.

1.2.3.1.4.1. Se **debe** realizar un Mapeo de grupo de archivos del formato DICOM al formato específico del medio físico.

1.2.3.1.4.2. Se **debe** realizar un Mapeo de ID de archivos (nombre de archivos y DICOMDIR) del formato DICOM al formato específico del medio físico.

1.2.3.1.4.3. El perfil de aplicación **debe** proporcionar la Información de manejo de archivos DICOM al medio físico.

1.2.3.2. El perfil de aplicación **debe** establecer el Formato de medios asociado con el medio Físico, los medios físicos disponibles y su formato se muestran a continuación:

1.2.3.2.1. Disco Duro en el cual el acceso a los archivos de sistema se realizan mediante S.O. directamente

1.2.3.2.1.1. Se **debe** definir el Formato Lógico (FAT)

1.2.3.2.2. Para CD

1.2.3.2.2.1. Se **debe** definir el Formato Físico dependiendo de la selección entre formato de sector de sesión única y multisesión

1.2.3.2.2.2. Se **debe** definir el Formato Lógico:

1.2.3.2.2.2.1. Se **debe** definir el del Campo identificador de sistema

1.2.3.2.2.2.2. Se **debe** definir el del área de descripción de volumen y de sistema

1.2.3.2.2.3. Se **debe** describir el Medio Físico: Disco CD-R de 120 mm como se describe en la parte II del estándar ISO 9960.

1.2.3.2.3. Para DVD

1.2.3.2.3.1. Se **debe** describir el Formato Físico para que cumpla con una de las siguientes definiciones aplicables:

1.2.3.2.3.1.1. Se **debe** especificar si es DVD para disco grabable (DVD-R)

1.2.3.2.3.1.2. Se **debe** especificar si es DVD para disco de solo lectura (DVD-ROM)

1.2.3.2.3.1.3. Se **debe** especificar si es disco regrabable (DVD-RW)

1.2.3.2.3.2. Se **debe** definir el Formato Lógico conforme a los archivos de sistema de los estándares UDF e ISO 9960.

1.2.3.2.4. Para medios removibles tipo USB

1.2.3.2.4.1. Se **debe** definir el Formato Lógico (FAT16)

1.2.3.2.4.1.1. Se **debe** proveer un conector que cumpla con las especificaciones físicas, eléctricas y de señales para los protocolos de comunicación USB 1.1 o 2.0

1.2.3.2.4.1.2. El dispositivo **debe** funcionar como un dispositivo de almacenamiento masivo, de acuerdo con la clase de almacenamiento masivo USB.

1.2.4. Se **debe** seleccionar una sintaxis de transferencia para el perfil de aplicación.

1.2.5. Se **debe** seleccionar un perfil de seguridad para el perfil de aplicación.

1.3. **debe** realizar operaciones M-WRITE de acuerdo a la especificación de la clase SOP identificada por el UID de la clase SOP en la meta-información del archivo (Meta File Information).

1.4. **debe** soportar la clase SOP del directorio del medio de almacenamiento (almacenado en el archivo DICOMDIR).

1.5. **debe** crear archivos DICOMDIR que contengan la clase SOP del directorio de almacenamiento del medio con registros de directorio haciendo múltiples referencias a un archivo a través de los registros de directorio MRDR.

2. Basado en la conformación de un FSR, una implementación de la clase de servicio de almacenamiento en medios:

2.1. **debe** reconocer un grupo de archivos y su correspondiente DICOMDIR que contiene la clase SOP del directorio de medios de almacenamiento a través de operaciones M-READ para acceder a su contenido (el FSR también debe soportar las operaciones M-INQUIRE FILE y M-INQUIRE FILE-SET).

2.2. **debe** conocer los requerimientos especificados en el punto 1.2 del rol FSC

2.3. **debe** realizar operaciones M-READ de acuerdo con la especificación de la clase SOP identificada por la UID de la clase SOP en la meta-información del archivo (META FILE INFORMACION).
2.4. **debe** leer archivos DICOMDIR que contengan la clase SOP del directorio del medio de almacenamiento con registros de directorio realizando múltiples referencias hacia archivos a través del registro de directorio MRDR.
2.5. **debe** leer archivos DICOMDIR con o sin módulo de información incluyendo los registros de directorio de tipos no soportados por la implementación.

3. Basado en la conformación de un FSU. Una Implementación de la clase de servicio de almacenamiento en medios:
   3.1. **debe** reconocer un grupo de archivos y su correspondiente DICOMDIR que contiene la clase SOP del directorio de medios de almacenamiento a través de operaciones M-READ para acceder a su contenido (el FSU también debe soportar las operaciones M-INQUIRE FILE y M-INQUIRE FILE-SET).
   3.2. **debe** ser capaz de crear uno o más archivos que pertenezcan al grupo de archivos invocando las operaciones M-WRITE con sus instancias SOP (que deben conocer los requerimientos de su IOD correspondiente)
   3.3. **debe** conocer los requerimientos especificados en el punto 1.2 del rol FSC
   3.4. **debe** realizar operaciones de M-READ de acuerdo a las especificaciones de la clase SOP identificadas por su UID en la meta información del archivo (META FILE INFORMATION).
   3.5. **debe** realizar operaciones de M-WRITE de acuerdo a las especificaciones de la clase SOP identificadas por su UID en la meta información del archivo (META FILE INFORMATION).
   3.6. **debe** soportar la clase SOP del directorio del medio de almacenamiento (almacenado en el archivo DICOMDIR).
   3.7. **debe** leer los archivos DICOMDIR que contenga la clase SOP del directorio del medio de almacenamiento con los registros de directorio haciendo múltiples referencias hacia archivos a través del registro de directorio MRDR.
   3.8. **debe** actualizar los archivos DICOMDIR que contienen la clase SOP del directorio del medio de almacenamiento creando registros de directorio, utilizando en registro de directorio MRDR en donde se necesitan múltiples referencias hacia archivos.
   3.9. **debe** leer los archivos DICOMDIR con o sin modulo de información de directorio incluyendo los registros de directorio de tipos no soportados por la implementación.

5.1.1.2 Requerimientos no funcionales (NFR´s)

Este tipo de requerimientos nos ayuda a identificar las cualidades sistémicas de la aplicación y de qué manera se comporta la interfaz del sistema con los actores. Para este sistema, se definen, derivadas de las cualidades sistémicas, las siguientes necesidades no funcionales:

- De manifestación (cualidades notorias para el usuario):
  - El sistema tiene que identificar el hardware y aceptar las diferentes marcas de fabricantes para la escritura de la información.

- El sistema debe identificar y estar preparado para trabajar con cualquier tipo de velocidad de transferencia disponible para el hardware.
- El sistema debe tener una comunicación rápida con el S. O. Para acelerar el tiempo de escritura.
- El sistema es un subsistema de un software que hace uso de sus funciones y al estar conformado por bibliotecas, solo habrá que darlas de alta para que puedan funcionar.

- De comportamiento:
  - El sistema tiene la opción de incrementar el grado de seguridad en la medida que el usuario desee, insertando nuevos algoritmos de compresión y/o encriptación.
  - El sistema debe ser autoadministrable ya que solo requiere el paso de la información de control y los datos que se desea manipular y el resto hacerlo automáticamente.
- Desarrollables
  - El sistema debe tener la capacidad de ser desarrollado, es decir, que no sea demasiado ambicioso para ser creado pero al mismo tiempo dejar abierta la posibilidad para seguir creciendo si así se demanda.
- Evolutivas
  - El sistema debe ser flexible para no quedar obsoleto en un lapso de tiempo considerable, esto se logrará trabajando con el esquema de arquitectura adecuado para incrementar el número de dispositivos y perfiles de aplicación y seguridad.
  - El mantenimiento debe ser sencillo y no afectar el funcionamiento del software general, esto se logrará modificando las bibliotecas y compilando para probar y ejecutar individualmente.

Los esfuerzos para obtener los resultados de esta primera fase se encuentran distribuidos en el primer período de tiempo de cada una de las 5 primeras iteraciones. El esfuerzo en la última iteración no es significativo, ya que sólo fueron refinaciones para integrar la funcionalidad total.

5.2 Fase 2: Elaboración

En esta fase, se centralizó el esfuerzo en identificar los casos de uso a partir de los requerimientos, así como en generar a partir de estos una arquitectura de software adecuada. En la primera iteración se definió la arquitectura base, mientras que en las demás iteraciones ser realizó el trabajo necesario para agregar la funcionalidad correspondiente. Los resultados que tienen que obtenerse en esta fase se listan en la sección 4.6.1.2.2. Los resultados de análisis y diseño se documentan en esta fase, ya que la mayor parte del trabajo se realiza dentro de la misma, como puede observarse en la Figura 3.5.

5.2.1 Requerimientos como casos de uso

El SRS nos ofreció una guía para construir los casos de uso que definirán el comportamiento del sistema. Hay que tomar en cuenta que para este fin se tuvieron que cubrir los FR´s y los NFR´s. Primero se identificaron los actores, relaciones, correspondencias y responsabilidades contenidos en los FR´s, seguido por la separación de

cada caso de uso tomando como guía los NFR´s o cualidades sistémicas. En el apéndice A se encuentra una guía para captura de requerimientos y cómo llegar a los casos de uso. En la siguiente sección se describen con detalle los casos de uso definidos a partir del documento de especificación de requerimientos.

5.2.2 Casos de uso

Auxiliado con las especificaciones del estándar para almacenamiento DICOM en medios físicos descrito en la sección 2.1.2 y tomando como referencia la especificación de requerimientos descritos en la sección 5.1.1.1, se generó como producto el modelo de casos de uso, en el cual se define como primera instancia el diagrama de casos de uso mostrado en la Figura 5.2.

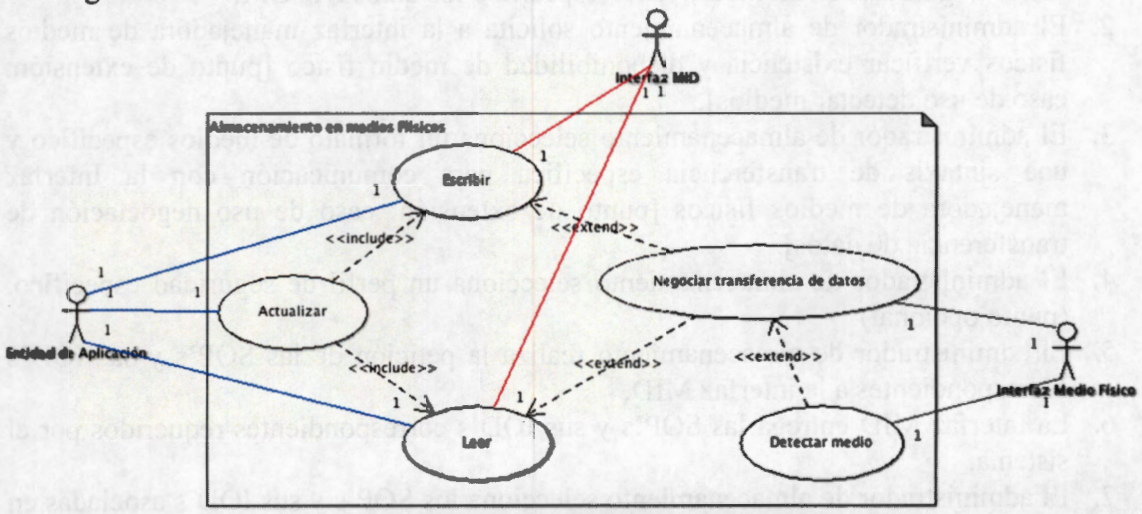

Figura 5.2 – Diagrama de casos de uso principal

A partir de la generación del diagrama de casos de uso principal se identificaron los siguientes actores:

- Entidad de aplicación (usuario).- Representa la petición que el usuario o entidad de aplicación desee ejecutar entre tres opciones, leer, escribir y actualizar.
- Interfaz hacia el medio físico.- Representa el medio físico al cual se va a tener acceso, esta interfaz tiene que validar la comunicación con este medio para que la transferencia siga la sintaxis específica y estándar para cada medio.
- Interfaz hacia el MID.- Representa la interacción con el modelo de información DICOM para la petición de SOP's específicas, esta interfaz tiene que hacer la petición entregando el VR correspondiente y verificar que las clases SOP entregadas sean correctas.

En el diagrama de casos de uso principal, las líneas simples que unen a los actores con los casos de uso, significan una asociación directa entre ambos elementos. Las líneas punteadas que tienen la leyenda de "include" significan que el caso de uso que apunta otro caso de uso está incluido dentro del mismo. En el caso de las líneas punteadas que tienen la leyenda "extend" significa que el caso de uso al que apunta otro caso de uso tiene una extensión de la funcionalidad del mismo.

En las siguientes sub secciones se describen de forma individual los casos de uso que integran al diagrama de casos de uso principal ilustrado en la en la Figura 5.2.

5.2.2.1 Descripción del Caso de uso: Escribir

5.2.2.1.1 Propósito

Escribir o guardar el archivo DICOM en un medio físico específico.

5.2.2.1.2 Curso típico de eventos

1. La entidad de aplicación hace la petición al administrador de almacenamiento para escribir (guardar) en un medio físico específico los datos DICOM.
2. El administrador de almacenamiento solicita a la interfaz manejadora de medios físicos verificar existencia y disponibilidad de medio físico [punto de extensión: caso de uso detectar medios].
3. El administrador de almacenamiento selecciona un formato de medios específico y una sintaxis de transferencia específica para comunicación con la Interfaz manejadora de medios físicos [punto de extensión: caso de uso negociación de transferencia de datos].
4. El administrador de almacenamiento selecciona un perfil de seguridad específico. (punto opcional)
5. El administrador de almacenamiento realiza la petición de las SOP's y sus IOD's correspondientes a la interfaz MID.
6. La interfaz MID entrega las SOP's y sus IOD's correspondientes requeridos por el sistema.
7. El administrador de almacenamiento selecciona los SOP´s y sus IOD's asociadas en la capa de formato de datos
8. El administrador de almacenamiento verifica la petición de escritura validando la existencia del comando escribir y obtiene la información del perfil de aplicación (ejemplo, eco cardiografía, angiografía, etc.) y sus datos para la aplicación (imagen y datos relacionados del paciente, tal como, nombre, dirección, edad, sexo, historia clínica, etc.) a través del manejador de perfiles de aplicación.
9. El administrador de almacenamiento crea el grupo de archivos [puntos de repetición: del punto 8 al punto 11 para crear un archivo a la vez hasta completar la estructura del grupo de archivos necesaria]
10. El administrador de almacenamiento añade al archivo DICOM la Meta-Información que consiste en un archivo cabecera (preámbulo) de 128 bytes, seguido por un prefijo DICOM de 4 bytes, seguido de los Meta-Elementos correspondientes.
11. El administrador de almacenamiento genera el ID del archivo DICOM.
12. El administrador de almacenamiento genera el DICOMDIR que contiene la clase SOP del directorio del medio de almacenamiento que corresponde al archivo dentro del grupo de archivos creados.
13. El administrador de almacenamiento genera el ID del DICOMDIR.
14. El administrador de almacenamiento ejecuta la operación M-WRITE de acuerdo a la especificación de la clase SOP identificada por el UID de la clase SOP en la meta-información del archivo DICOM (Meta File Information) correspondiendo una ejecución por cada archivo dentro del grupo de archivos.

15. El administrador de almacenamiento envía un mensaje a la entidad de aplicación de escritura finalizada (ya sea correcta o incorrecta)

5.2.2.1.3 Cursos alternos

- Línea 2: El administrador de almacenamiento no reconoce el comando
- Línea 2: El medio físico no existe o no está listo (disponible)
- Línea 3: El medio físico no soporta el formato seleccionado
- Línea 3: El administrador de almacenamiento no reconoce la sintaxis solicitada
- Línea 5: Falta información dentro de las SOP's entregadas por la interfaz MID
- Línea 14: El grupo de archivos todavía no contiene los archivos completos (con su ID, DICOMDIR, y DICOMDIR ID) y se vuelve a ejecutar hasta completar el grupo de archivos.

5.2.2.1.4 Post-condiciones

Se requiere la disponibilidad del medio físico para realizar la escritura, la creación del DICOMDIR y su estructura, así como la validación de las clases SOP de acuerdo al diccionario de datos.

5.2.2.1.5 Pre-condiciones

Se requiere la comunicación con la Interfaz MID para la solicitud de las clases SOP correspondientes.

5.2.2.1.6 Requerimientos especiales (no funcionales)

Este caso de uso debe satisfacer los requerimientos desarrollables descritos en la sección 5.1.1.2

5.2.2.1.7 Relaciones

Como puede observarse en la Figura 5.1, este caso de uso se asocia directamente con la interfaz MID y con la entidad de aplicación

5.2.2.1.8 Puntos de extensión

Este caso de uso no tiene puntos de extensión.

5.2.2.2 Descripción del Caso de Uso: Lectura

5.2.2.2.1 Propósito

Leer un archivo DICOM desde un medio físico específico.

5.2.2.2.2 Curso típico de eventos

1. La entidad de aplicación hace la petición al administrador de almacenamiento para leer desde un medio físico especifico los datos DICOM.

2. El administrador de almacenamiento solicita a la interfaz manejadora de medios físicos verificar existencia y disponibilidad de medio físico [punto de extensión: caso de uso detección de medios].
3. El administrador de almacenamiento selecciona un formato de medios específico y una sintaxis de transferencia específica para comunicación con la interfaz manejadora de medios físicos [punto de extensión: caso de uso negociación de transferencia de datos].
4. El administrador de almacenamiento selecciona un perfil de seguridad específico. (punto opcional)
5. El administrador de almacenamiento realiza la petición de las SOP's y sus IOD's correspondientes a la interfaz MID.
6. La interfaz MID entrega las SOP's y sus IOD's correspondientes requeridos por el sistema.
7. El administrador de almacenamiento selecciona los SOP's y sus IOD's asociadas en la capa de formato de datos
8. El administrador de almacenamiento verifica la petición de lectura validando la existencia del comando leer y obtiene la información del perfil de aplicación (ejemplo, ecocardiografía, angiografía, etc.) y sus datos para la aplicación (imagen y datos relacionados del paciente, tal como, nombre, dirección, edad, sexo, historia clínica, etc.) a través del manejador de perfiles de aplicación.
9. El administrador de almacenamiento crea el grupo de archivos [puntos de repetición: del punto 8 al punto 11 para crear un archivo a la vez hasta completar la estructura del grupo de archivos necesaria]
10. El administrador de almacenamiento interpreta la Meta-Información que consiste en un archivo cabecera (preámbulo) de 128 bytes, seguido por un prefijo DICOM de 4 bytes, seguido de los Meta-Elementos correspondientes.
11. El administrador de almacenamiento interpreta el ID del archivo DICOM.
12. El administrador de almacenamiento interpreta el DICOMDIR que contiene la clase SOP del directorio del medio de almacenamiento que corresponde al archivo dentro del grupo de archivos leídos.
13. El administrador de almacenamiento interpreta el ID del DICOMDIR.
14. El administrador de almacenamiento ejecuta la operación M-READ de acuerdo a la especificación de la clase SOP identificada por el UID de la clase SOP en la meta-información del archivo DICOM (Meta File Information) correspondiendo una ejecución por cada archivo dentro del grupo de archivos.
15. El administrador de almacenamiento envía un mensaje a la entidad de aplicación de lectura finalizada (ya sea correcta o incorrecta)

5.2.2.2.3 Cursos alternos

- Línea 2: El administrador de almacenamiento no reconoce el comando
- Línea 2: El medio físico no existe o no está listo (disponible)
- Línea 3: El medio físico no soporta el formato seleccionado
- Línea 3: El administrador de almacenamiento no reconoce la sintaxis solicitada
- Línea 5: Falta información dentro de las SOP's entregadas por la interfaz MID

- Línea 14: El grupo de archivos todavía no contiene los archivos completos (con su ID, DICOMDIR, y DICOMDIR ID) y se vuelve a ejecutar hasta completar el grupo de archivos.

5.2.2.2.4 Post-condiciones

Se requiere la disponibilidad del medio físico para realizar la lectura, la verificación de existencia del DICOMDIR y su estructura, así como la validación de las clases SOP de acuerdo al diccionario de datos.

5.2.2.2.5 Pre-condiciones

Se requiere la comunicación con la Interfaz MID para la solicitud de las clases SOP correspondientes.

5.2.2.2.6 Requerimientos especiales (no funcionales)

Este caso de uso debe satisfacer los requerimientos desarrollables descritos en la sección 5.1.1.2

5.2.2.2.7 Relaciones

Como puede observarse en la Figura 5.1, este caso de uso se asocia directamente con la interfaz MID y con la entidad de aplicación

5.2.2.2.8 Puntos de extensión

Este caso de uso no tiene puntos de extensión.

5.2.2.3 Descripción del Caso de Uso: Actualización

5.2.2.3.1 Propósito

Actualizar la información DICOM en un medio físico específico a través de la combinación de escritura y lectura.

5.2.2.3.2 Curso típico de eventos

Es una ejecución de la secuencia de eventos de lectura, seguido por la escritura en medios físicos, especificados en las secciones 5.2.2.2.2 y 5.2.2.1.2 respectivamente

5.2.2.3.3 Cursos alternos

Se especifican en las secciones 5.2.2.2.3 y 5.2.2.1.3

5.2.2.3.4 Post-condiciones

Se requiere la disponibilidad del medio físico para realizar la actualización de información DICOM.

5.2.2.3.5 Pre-condiciones

Se requiere la comunicación con la Interfaz MID para la solicitud de las clases SOP correspondientes. Se requiere la inclusión de las funciones de casos de uso Lectura y Escritura

5.2.2.3.6 Requerimientos especiales (no funcionales)

Este caso de uso debe satisfacer los requerimientos desarrollables descritos en la sección 5.1.1.2

5.2.2.3.7 Relaciones

Como puede observarse en la Figura 5.1, este caso de uso se asocia directamente con la entidad de aplicación

5.2.2.3.8 Puntos de extensión

Como puede observarse en la Figura 5.1, este caso de uso incluye del caso de uso Negociación de transferencia de datos las funciones de:

- Escritura en medios físicos
- Lectura en medios físicos

5.2.2.4 Descripción del Caso de Uso: Detección de medios

5.2.2.4.1 Propósito

Detectar la existencia y disponibilidad del medio físico en donde se desea ejecutar un comando

5.2.2.4.2 Curso típico de eventos

1. El administrador de almacenamiento solicita a la interfaz manejadora de medios físicos verificar existencia y disponibilidad de medio físico.
2. El manejador de medios físicos solicita al consultor del sistema operativo la información del medio físico específico para verificar su existencia.
3. El manejador de medios físicos toma el control del medio físico y manda mensajes de prueba para verificar que el medio físico no se encuentre realizando otra actividad
4. El manejador de medios físicos envía un mensaje de control al administrador de almacenamiento para indicar que todo está bajo control.

5.2.2.4.3 Cursos alternos

- Línea 2: No existe el medio físico
- Línea 3: El medio no está disponible

5.2.2.4.4 Post-condiciones

Se requiere la extensión de funciones del caso de uso Negociación de transferencia de datos.

5.2.2.4.5 Pre-condiciones

Se requiere la definición del tipo de medio y el tipo de sistema operativo.

5.2.2.4.6 Requerimientos especiales (no funcionales)

Este caso de uso debe satisfacer los requerimientos evolutivos y de manifestación (cualidades notorias para el usuario) descritas en la sección 5.1.1.2

5.2.2.4.7 Relaciones

Como puede observarse en la Figura 5.1, este caso de uso se asocia directamente con la interfaz de medios físicos

5.2.2.4.8 Puntos de extensión

Como puede observarse en la Figura 5.1, este caso de uso extiende al caso de uso Negociación de transferencia de datos las funciones de:

- Definición de protocolo de comunicación para acceso al medio
- Definición de reglas de transferencia para acceso al medio.

5.2.2.5 Descripción del Caso de Uso: Negociación de transferencia de datos

5.2.2.5.1 Propósito

Negociar las reglas de sintaxis de transferencia de datos y el formato específico para almacenar los datos en el medio físico.

5.2.2.5.2 Curso típico de eventos

1. El administrador de almacenamiento se comunica con la interfaz manejadora de medios físicos para solicitar un formato específico.
2. El manejador de medios físicos verifica que el medio físico soporte el formato deseado.
3. El manejador de medios físicos envía un mensaje al sistema Storage para confirmar que el formato deseado es correcto y se puede continuar.
4. El administrador de almacenamiento solicita a la interfaz manejadora de medios físicos las reglas de sintaxis necesarias para poder escribir en el medio físico deseado.
5. El manejador de medios físicos envía un paquete al administrador de almacenamiento con las reglas de sintaxis necesarias.
6. El administrador de almacenamiento realiza un mapeo de la información DICOM ya generada basándose en las reglas de sintaxis recibidas y envía un mensaje al manejador de medios físicos para indicar que ya está listo para iniciar la transferencia.
7. El manejador de medios físicos envía un mensaje al administrador de almacenamiento para indicar que el medio se encuentra listo y en espera de datos.

5.2.2.5.3 Cursos alternos

- Línea 2: El medio físico no soporta el formato deseado
- Línea 4: El manejador de medios físicos no tiene establecidas las reglas de transferencia para el medio físico específico

5.2.2.5.4 Post-condiciones

Se requiere la extensión de funciones de los casos de uso Escritura y Lectura.

5.2.2.5.5 Pre-condiciones

Se requiere la definición de los protocolos de comunicación y formato de medio físico.

5.2.2.5.6 Requerimientos especiales (no funcionales)

Este caso de uso debe satisfacer los requerimientos de comportamiento descritos en la sección 5.1.1.2

5.2.2.5.7 Relaciones

Como puede observarse en la Figura 5.1, este caso de uso se asocia directamente con la interfaz de medios físicos

5.2.2.5.8 Puntos de extensión

Como puede observarse en la Figura 5.1, este caso de uso extiende al caso de uso Negociación de transferencia de datos las funciones de:

- Escritura en medios físicos
- Lectura en medios físicos

5.2.3 Análisis y Diseño

Durante este conjunto de iteraciones se obtuvieron resultados para generar el "esqueleto" de la solución. El modelo de diseño se documenta en la siguiente subsección, luego de documentar los productos generados para las disciplinas de análisis y diseño correspondientes a esta subsección.

5.2.3.1 Abstracciones

Para llegar al modelo de dominio se tuvo que realizar una abstracción de cada clase, sus relaciones y dependencias. A partir del SRS se realizó una selección de candidatos a abstracciones, luego se aplicó la técnica de clasificación utilizando análisis CRC creando así las relaciones o dependencias basadas en los casos de uso. La mayor parte de estas abstracciones se encuentran dentro del glosario documentado en el apéndice C de ésta tesis en conjunto con otros conceptos definidos que tienen relación con este desarrollo. Algunas de estas abstracciones se convierten en entidades a través del modelo de dominio. Estas entidades se clasifican en clases de datos, las demás abstracciones se clasificarán en clases de control y de frontera (o interfaz). La lista entera de abstracciones obtenida fue la siguiente:

- Grupo de Archivos (File-Set)
- SOP's
- IOD's
- Medio Físico
- Interfaz manejadora de medios
- Interfaz MID
- Entidad de aplicación

- Administrador de perfil de seguridad
- Administrador de perfil de aplicación
- Administrador de almacenamiento
- Consultor de sistema operativo
- Datos DICOM
- DICOMDIR
- Meta-Información
- Preámbulo
- Meta-Elementos
- Prefijo DICOM
- ID de archivo DICOM
- ID de archivo DICOMDIR
- Formato de medios
- Formato de sintaxis de transferencia
- Método de encripción
- UID de la clase SOP

5.2.3.2 Modelo de Diseño

Para determinar las responsabilidades y colaboradores fue necesario analizar los escenarios de los FR´s de los casos de uso identificados para cada abstracción. Las responsabilidades de cada abstracción son cualquier operación o especificación, mientras que los colaboradores son otros objetos (generalmente otra abstracción clave) a la cual la abstracción clave está asociada. El diagrama generado es entonces el modelo de dominio y puede observarse en la Figura 5.3.

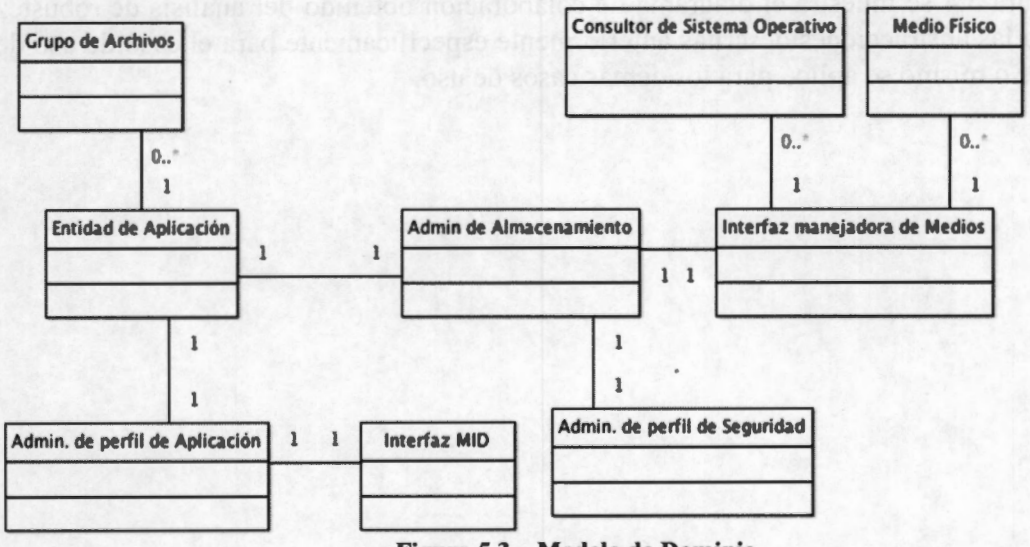

Figura 5.3 – Modelo de Dominio

De la lista de abstracciones, se obtuvieron los candidatos a clases de datos o entidades:

1. Grupo de Archivos (File-Set)
2. SOP's

3. IOD's
4. Medio Físico

Y sus candidatos a atributos fueron los siguientes:

1. Datos DICOM
2. DICOMDIR
3. Meta-Información
4. Preámbulo
5. Meta-Elementos
6. Prefijo DICOM
7. ID de archivo DICOM
8. ID de archivo DICOMDIR
9. Formato de medios
10. Formato de sintaxis de transferencia
11. Método de encriptación
12. UID de la clase SOP

Los candidatos a clases de control son:

1. Administrador de almacenamiento
2. Administrador de perfiles de seguridad
3. Administrador de perfiles de aplicación

Los candidatos a clases de interfaz son:

1. Interfaz manejadora de medios
2. Interfaz MID
3. Entidad de aplicación
4. Consultor de sistema operativo

En la Figura 5.4 se muestra el diagrama de colaboración obtenido del análisis de robustez aplicado a las abstracciones descritas anteriormente específicamente para el caso de uso de escritura. Lo mismo se aplicó para los demás casos de uso.

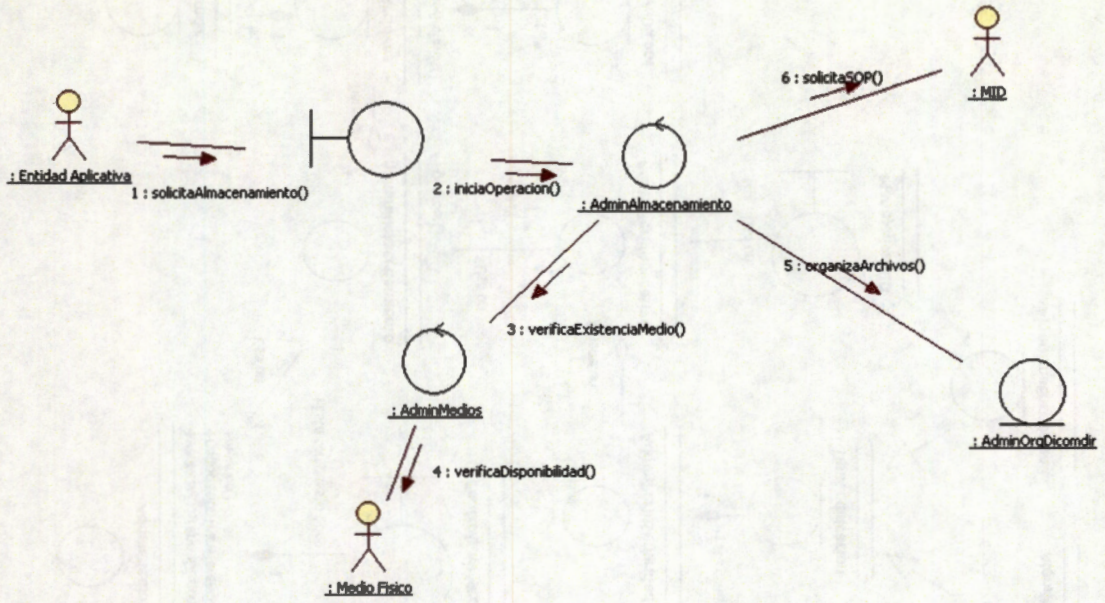

Figura 5.4 – Diagrama de colaboración del caso de uso Escritura

Para identificar conceptos hubo que basarse en los requerimientos, en los casos de uso y en el conocimiento general acerca del dominio del problema. Esto nos generó conocimiento acerca del entorno del sistema.

El diagrama que se generó a partir de este análisis para todos los casos de uso es el diagrama de clases en una primera versión, correspondiente al modelo de análisis, para posteriormente generar el diagrama de clases refinado y final, correspondiente al modelo de diseño. El diagrama de clases en su primera versión, representado en forma icónica para una mejor comprensión de los tipos de clases, se muestra en la Figura 5.5 5.5 con la siguiente correspondencia:

- Clases de datos (elementos de vista):
- Clases de control (elementos de control):
- Clases de interfaz (elementos de modelo):

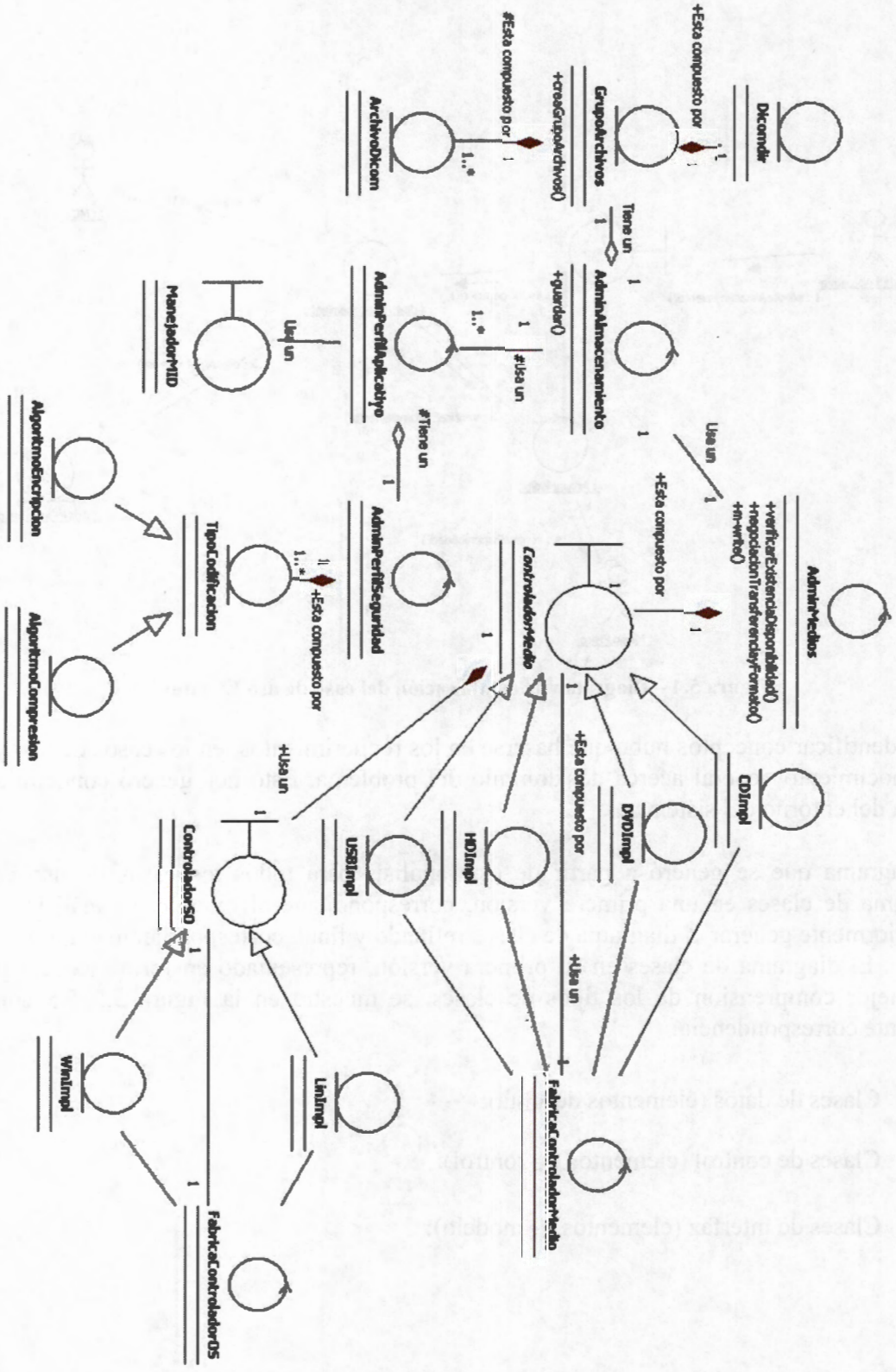

Figura 5.5 – Diagrama de clases (Modelo de diseño final) – representación icónica

5.2.3.3 Arquitectura del sistema (Modelo de Arquitectura)

Para obtener una arquitectura basada en el modelo MVC, la cual se detalló en la sección 3.4.2.2, se tuvo que realizar primero la abstracción del sistema en el mundo real, descrito en la sección 5.2.3.1. Posteriormente en un modelado conceptual, estas abstracciones se convirtieron en objetos con relaciones bien definidas para finalmente poder transformarlos en componentes de software. En la sección 5.2.3.2 se muestra esta transformación de abstracciones a componentes de software y los resultados que se obtuvieron. Cabe mencionar que los resultados de estos tres pasos fueron obtenidos de manera incremental dentro de las 5 primeras iteraciones antes de comenzar la construcción de cada una de ellas.

Para esta etapa es posible clasificar en paquetes y definir qué clases formaran parte de ellos. Los diagramas de clases y componentes ofrecen una vista de la arquitectura del sistema. A continuación se muestra el diagrama de clases refinado en la Figura 5.6.

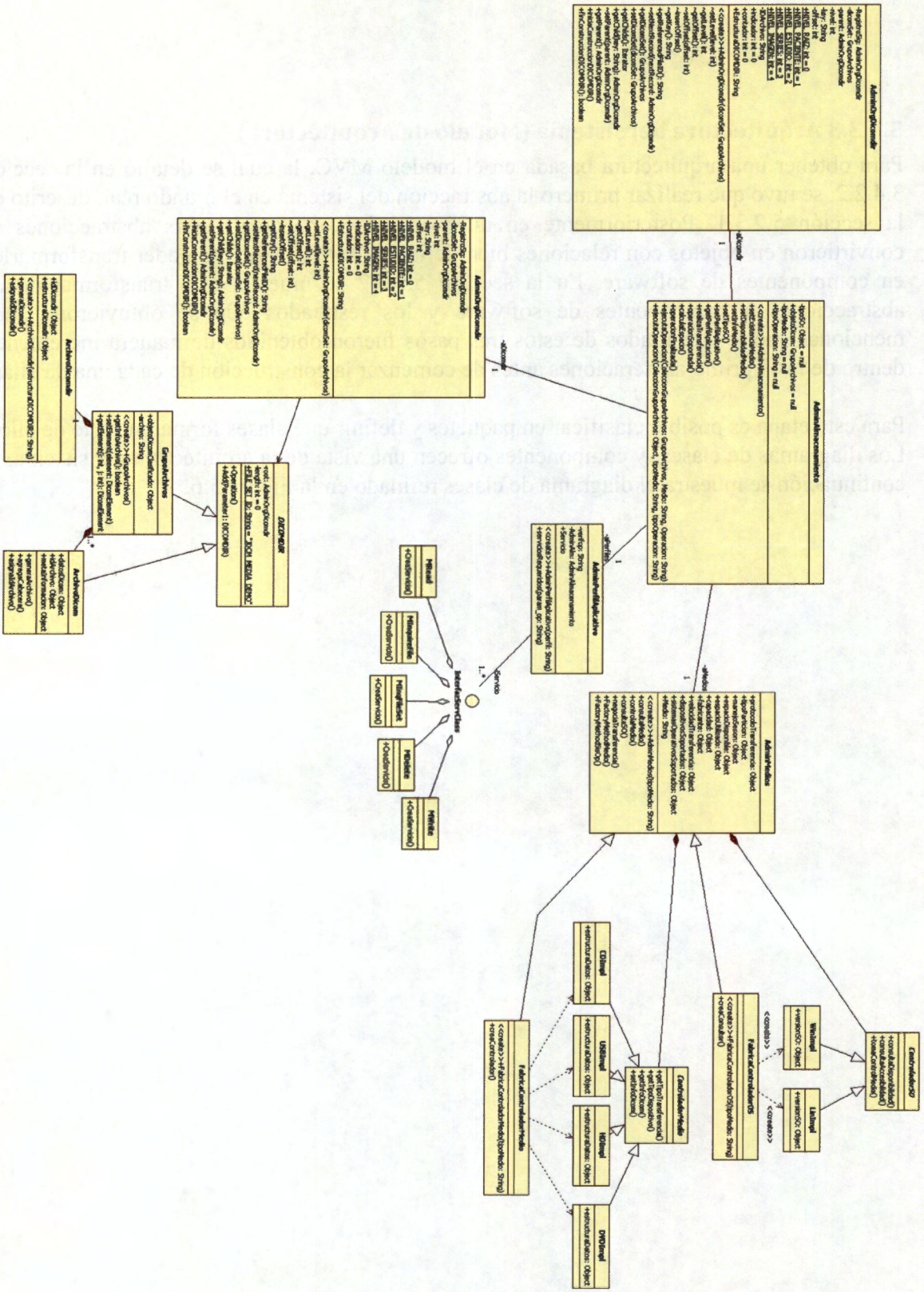

Figura 5.6 – Diagrama de clases refinado

Una vez obtenido el diagrama de clases refinado, las clases que lo componen pueden ser agrupadas en componentes dependiendo de su relación y funcionalidad. En este caso se definió una estructura determinada, la cual se muestra en el diagrama de componentes ilustrado en la Figura 5.7.

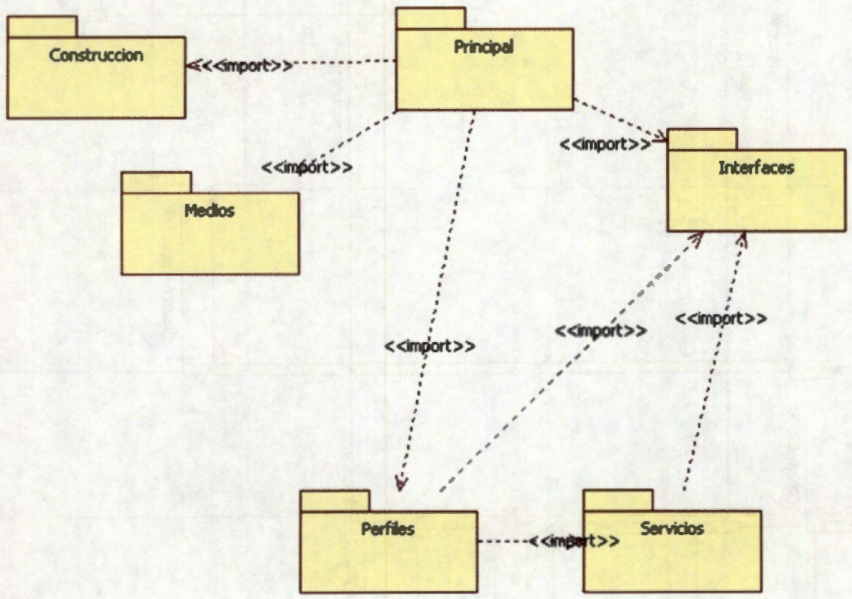

Figura 5.7 – Diagrama de componentes

La obtención de los diagramas de clases y componentes, le dan al desarrollador una base fundamental para dar la funcionalidad al sistema, sin embargo, es necesario en ocasiones, complementar con algunos diagramas el modelo para comprender a un mejor nivel de detalle. En este caso se decidió necesario el apoyo del diagrama de secuencia de eventos para ese fin, el cual se representa en la Figura 5.8.

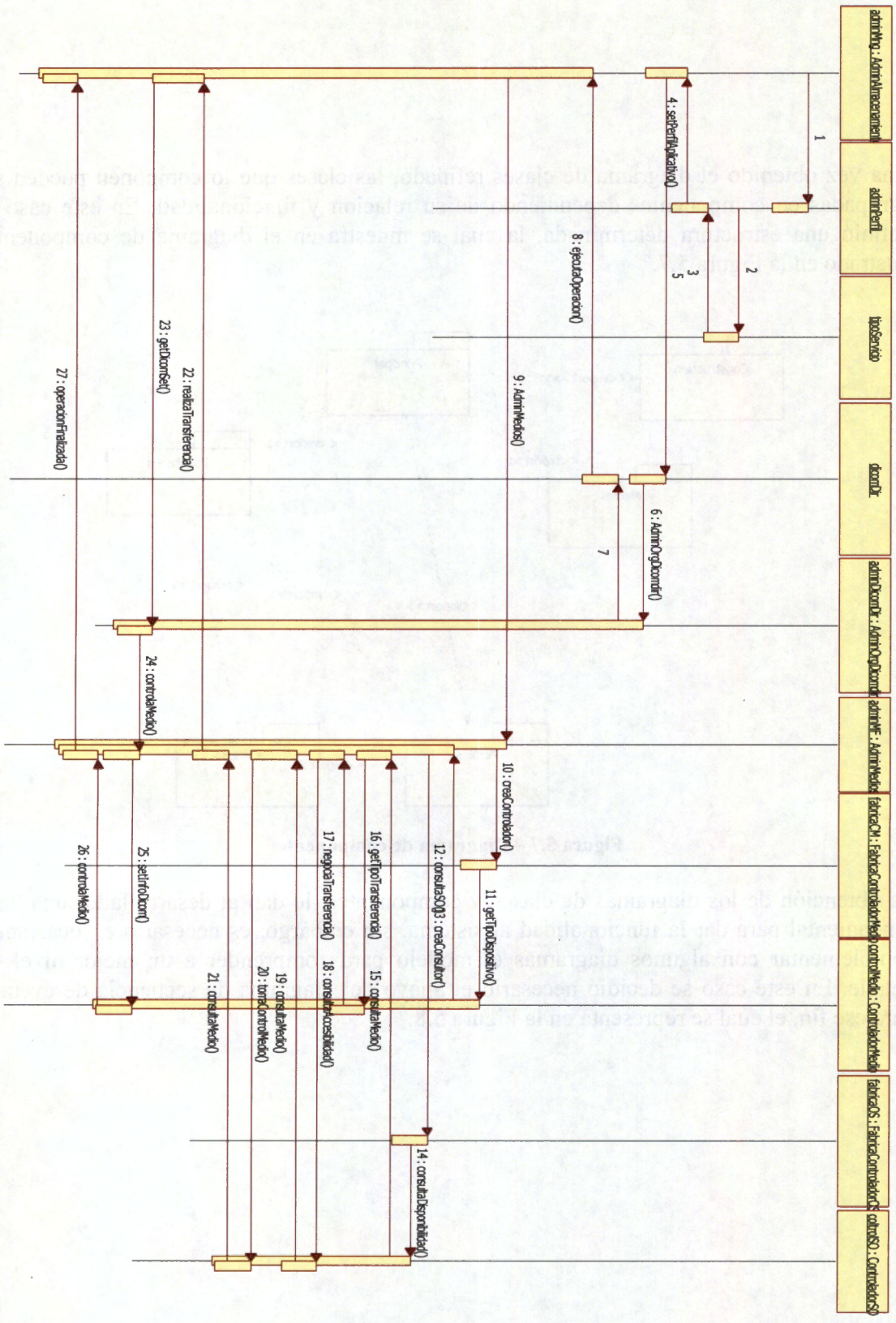

Figura 5.8 – Diagrama de secuencia

El diagrama de secuencia nos ofrece un panorama del flujo de datos para el acceso DICOM a CD, DVD, USB y a disco duro, que es básicamente el mismo, sólo cambia el medio físico al cual se realizará la conexión para lectura/escritura de datos DICOM. Este cambio se traduce en la generación de un objeto distinto a partir de la misma clase y el diagrama de secuencia, por lo tanto, es exactamente el mismo para los cuatro medios y nos da una referencia para cifrar los métodos y poder pasar a la fase de construcción con la arquitectura completamente validada e incluso con un ejecutable de la arquitectura del sistema.

5.2.3.4 Patrones de diseño

En la arquitectura de esta biblioteca de software se identificaron algunas estructuras que requirieron la aplicación de patrones de diseño. Un patrón de diseño es una solución a un problema recurrente de diseño [Copilen, 1994].

Los patrones de diseño utilizados en la arquitectura fueron los siguientes:

- Patrón de Método – Fábrica (FactoryMethod). Consiste en utilizar una clase constructora abstracta con unos cuantos métodos definidos que usan algunos otros métodos abstractos con el fin de que utilizando uno u otro hijo (por herencia) se tenga uno u otro comportamiento [Wikipedia.org – Patrón de diseño, 2007].
- Composite. Sirve para construir objetos complejos a partir de otros más simples y similares entre sí, gracias a la composición recursiva y a una estructura en forma de árbol [Wikipedia.org – Patrón de diseño, 2007].
- Estado (State - no codificado, correspondiente a perfil de seguridad). Se utiliza cuando el comportamiento de un objeto cambia dependiendo del estado del mismo. Por ejemplo, una alarma, que puede tener estados de activado, desactivado o en configuración.

El diagrama de clases refinado ya tiene aplicados los patrones de diseño y es muy evidente su estructura dentro del diagrama mostrado en la Figura 5.6. El patrón Método-Fabrica se aplicó a la generación dinámica de clases que manejan las interfaces con medios físicos y con los sistemas operativos. El patrón Composite se aplicó sobre la estructura que genera el DICOMDIR y organiza los archivos DICOM relacionados. El patrón State se aplicó para la generación de perfiles de seguridad y aunque se tomó en cuenta para el diseño de la arquitectura no fue codificado por que no estuvo dentro de los alcances del proyecto.

5.3 Fase 3: Construcción

Una vez generada la base de la arquitectura del sistema, fue posible comenzar con la programación como tal. En la sección 4.6.1 pueden identificarse las actividades, responsabilidades y roles que corresponden a esta fase. Los productos que se deben generar y refinar en esta fase se listan en la sección 4.6.1.2.3 específicamente.

Para continuar garantizando una realización ordenada del proyecto, se convino generar una estructura basada en el lugar de realización del código. Esto dio como resultado la ruta mx/uam/izt debido a que se desarrolló en México, dentro de la UAM en la unidad Iztapalapa. En la Figura 5.9 puede observarse esta estructura. El diagrama obtenido en la

Figura 5.7, se refleja en el siguiente nivel de esta misma estructura respetando los nombres de cada componente.

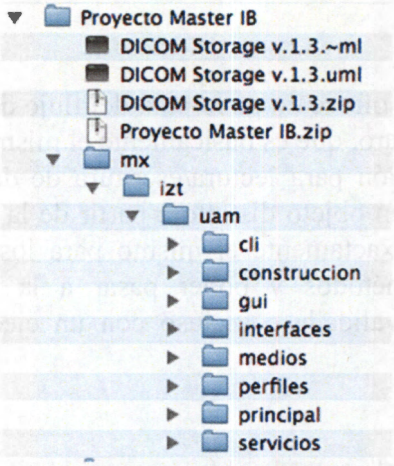

Figura 5.9 - Estructura de código

El desarrollo siguió el orden marcado por el plan de iteraciones, que se encuentra en la sección 4.6.1.3. Sin embargo, para comprender la cobertura de requerimientos, los resultados se documentarán describiendo en primer lugar el desarrollo del módulo a través del cual se manipula la información DICOM para lectura, escritura y actualización, que corresponde a los casos de uso escritura, lectura y actualización descritos en la sección 5.2.2 y que cubre los requerimientos 1.1, 1.3, 1.4, 1.5, 2.1, 2.3, 2.4, 2.5, 3.1, 3.3, 3.4 y 3.5, especificados en la sección 5.1.1.1 y posteriormente el desarrollo de la interacción con los medios físicos, que corresponde a los casos de uso negociación de transferencia de datos y detección de medios descritos en la sección 5.2.2 y que cubre los requerimientos 1.2, 2.2 y 3.2 especificados en la sección 5.1.1.1.

5.3.1 Desarrollo de módulo para manipulación de información DICOM para escritura, lectura y actualización

Para generar la información DICOM necesaria para el almacenamiento en medios físicos, tienen que generarse los objetos a partir de las clases contenidas en 3 de los componentes de la arquitectura. Estos componentes pueden observarse con detalle en la Figura 5.10, en el orden que marca el diagrama de secuencia ilustrado en la Figura 5.8.

Figura 5.10 - Estructura de código – manejo de información DICOM

En esta conjunción, se genera el archivo DICOMDIR, dentro del componente de construcción ilustrado en la Figura 5.10, basado en la información del DATASET obtenido por la interfaz MID (que en este caso tuvo que ser simulada por la falta de implementación de dicho módulo) para luego presentar en forma adecuada el perfil de aplicación, dentro del componente de perfiles y finalmente enviar la información y comandos necesarios como un servicio DICOM en el contexto adecuado, los cuales se encuentran agrupados dentro del componente de servicios igualmente ilustrado en la Figura 5.10.

En las siguientes líneas de código puede observarse como se genera el perfil adecuado bajo el contexto requerido por la entidad de aplicación.

```
import mx.izt.uam.interfaces.*;
import mx.izt.uam.principal.AdminAlmacenamiento;
import mx.izt.uam.servicios.*;

public class AdminPerfilAplicativo {
private String verifop;
private AdminAlmacenamiento adminAlm;
private InterfazServClass iServicio;
    public AdminPerfilAplicativo(String perfil) {
        // TODO Auto-generated constructor stub
        verifop = perfil;

    }
    public InterfazServClass serviciosRequeridos(String param_op) {
        if (param_op.equals("read")){
            iServicio = new MRead();
            }
            if (param_op.equals("delete")){
            iServicio = new MDelete();
            }
            if (param_op.equals("write")){
            iServicio = new MWrite();
            }
            if (param_op.equals("inq_file")){
            iServicio = new MInquireFile();
            }
            if (param_op.equals("inq_file_set")){
            iServicio = new MInqFileSet();
            }
    return iServicio;
    }
}
```

5.3.2 Desarrollo para interacción con los medios físicos CD y DVD

El desarrollo de esta versión, corresponde a la finalización de las iteraciones 1, 2 y parte de la 5 para luego integrarla a la versión final en la iteración 6. En la Figura 5.11 se pueden identificar las clases que corresponden al manejo de medios físicos con el sistema operativo correspondiente.

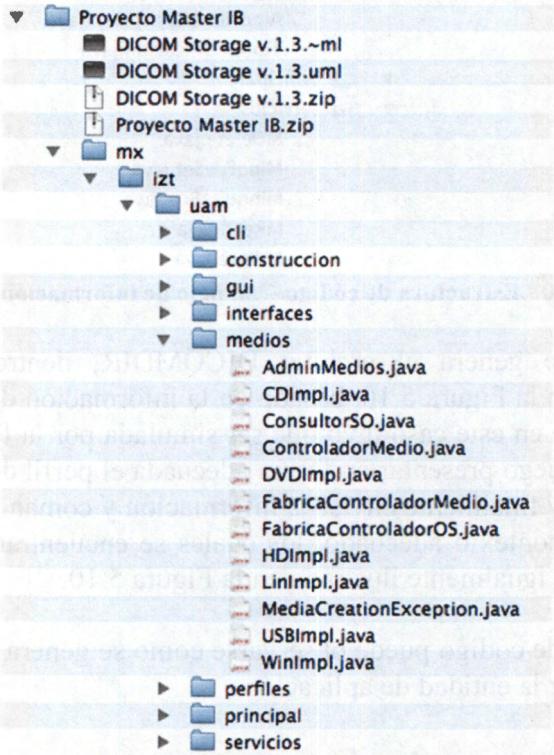

Figura 5.11 - Estructura de código - medios

Para lograr el acceso a los dispositivos de CD y DVD, se optó por utilizar una biblioteca de software ya existente llamada cdrtools, la cual provee herramientas necesarias para acceder

al medio y poder leer, guardar y actualizar la información bajo ciertas reglas y parámetros [Berlios, 1997]. A esta estrategia utilizada, se le conoce como COTS (por sus siglas en inglés "commercial off the shelf") que es la acción de seleccionar componentes de software para un fin específico [Wikipedia.org – COTS, 2008].

La estructura DICOM definida en el modelo es independiente a esta herramienta. Lo que hace la biblioteca cdrtools es tomar el control del CD y DVD para tener acceso (escritura, lectura y/o actualización de datos). A continuación se muestra parte del código de acceso al medio CD a través de esta biblioteca, por medio de la cual se puede controlar la velocidad de escritura, unidad del medio, entre otros parámetros.

```
public String existenciaDispositivo(String Medio) throws Exception {
    String comando1 = "cdrecord dev=1,0,0 -checkdrive";
    String comando2 = "cdrecord dev=0,0,1 -checkdrive";
    String comando3 = "cdrecord dev=1,0,1 -checkdrive";

    SysCommandExecutor ComEx = new SysCommandExecutor(); //ComEx.getCommandOutput();
 ComEx.runCommand(comando1);
 if(datos_resultado != "")
 {
     datos_resultado = "1,0,0";
     //System.out.println(Salida);
 }
 else
 {
     ComEx.runCommand(comando2);
     if(datos_resultado != "")
     {
         datos_resultado = "0,0,1";
         //System.out.println(Salida);
     }
     else{
         ComEx.runCommand(comando3);
         if(datos_resultado != "")
         {
             datos_resultado = "1,0,1";
             //System.out.println(Salida);
         }
         else{System.out.println("No hay ning'n medio disponible");}
    }
 }return datos_resultado;
 TODO Auto-generated method stub
}
public int getGraceTime() {
    return velocidad_escritura;
}
```

5.3.3 Desarrollo para interacción con los medios físicos USB y HD

El desarrollo de esta versión, corresponde a la finalización de las iteraciones 3, 4 y parte de la 5 para luego integrarla a la versión final en la iteración 6. En esta parte de la fase de construcción el acceso a los medios de disco duro y USB fue realmente sencillo, ya que dentro de las clases de java existe un conjunto que tiene exactamente el mismo fin, por lo que al igual que en el caso del acceso a CD y DVD, se incluyó el uso de la biblioteca bajo ciertas reglas y parámetros. A continuación se muestran algunas líneas de código utilizadas para crear y acceder a archivos.

```
//MÉtodo para escritura de archivos.
public void mkfile(File file)throws IOException{
    String archivo = file.getName();
    FileWriter fw = new FileWriter(archivo);
    BufferedWriter bw = new BufferedWriter (fw);


}
//MÉtodo para lectura de archivos.
public void rdfile (File file) throws IOException{
    String archivo = file.getAbsolutePath();
    String linea;
    //CreaciÓn del lector de archivo
    FileReader fr = new FileReader(archivo);
    //Para manejo de archivo
    BufferedReader entArchivo = new BufferedReader(fr);
    linea =entArchivo.readLine();
    while (linea != null)
    {
        System.out.println(linea + "\n");
        linea = entArchivo.readLine(); //se lee una nueva linea
    }
    entArchivo.close();;
}
```

5.4 Fase 4: Transición

El trabajo de esta última fase fue realizado casi en su mayoría por el rol de "tester" o integrador para pruebas. Los resultados son el sistema ya en su fase de entrega, como se menciona en la sección 4.6.1.2.4 , el cual tiene que ser consistente con el modelo de la Figura 5.6.

5.4.1. Integración del sistema

La integración es la unión de toda la funcionalidad en el sistema completo. En la sección 4.6.1.1 puede encontrarse mayor información sobre la planeación de esta fase así como de los roles, actividades y responsabilidades que deben ser cubiertos.

Los casos de pruebas entonces están relacionados directamente con los casos de uso bajo los criterios descritos en la sección 3.4.2.4. Los casos de uso se muestran en la sección 5.2.1. Los casos de prueba están documentados bajo el mismo nombre de su correspondiente caso de uso y se documentan de la siguiente forma.

Caso de prueba:

- Entrada
- Resultados
- Condiciones

En las siguientes subsecciones se muestran los resultados de las pruebas, basados en los casos de pruebas, tomando como referencia los casos de uso ilustrados en la Figura 5.2.

5.4.2 Caso de prueba: Escribir

5.4.2.1 Entradas

Dentro de esta prueba se debe garantizar la cobertura de los requerimientos funcionales descritos en la sección 5.1.1, ya que los datos de entrada deben estar bajo el formato DICOM.

5.4.2.2 Resultado

Una vez ejecutada la prueba para Escritura en medios, se deberán almacenar datos en el medio en estado virgen, es decir, acceso por primera vez.

5.4.2.3 Condiciones

Para generar la información DICOM, es necesario que se encuentre funcionando la interfaz con el MID para validar los datos DICOM generados. Lamentablemente, este complemento no estuvo dentro de los alcances de este proyecto, por lo que se omitió este punto dentro de esta prueba y se trabajó con DATASETS generados de manera manual para integrar el manejo de información DICOM con el acceso a medios físicos.

5.4.3 Caso de prueba: Leer

5.4.3.1 Entradas

Dentro de esta prueba se debe garantizar la cobertura de los requerimientos funcionales descritos en la sección 5.1.1, ya que los datos de entrada deben estar bajo el formato DICOM.

5.4.3.2 Resultado

Una vez ejecutada la prueba para lectura en medios, se deberá poder recuperar datos DICOM e información del medio, tal como fecha de almacenamiento, modelo del medio, formato del medio, etc.

5.4.3.3 Condiciones

Para generar la información DICOM, es necesario que se encuentre funcionando la interfaz con el MID para validar los datos DICOM obtenidos. Lamentablemente, este complemento no estuvo dentro de los alcances de este proyecto, por lo que se omitió este punto dentro de esta prueba y se trabajó con lectura de archivos que no son DICOM.

5.4.4 Caso de prueba: Actualizar

5.4.4.1 Entradas

Dentro de esta prueba se debe garantizar la cobertura de los requerimientos funcionales descritos en la sección 5.1.1, ya que los datos de entrada deben estar bajo el formato DICOM.

5.4.4.2 Resultado

Una vez ejecutada la prueba para lectura en medios, se deberá poder recuperar datos DICOM e información del medio, tal como fecha de almacenamiento, modelo del medio, formato del medio, etc. Luego se deberá ejecutar la prueba para escritura en medios para agregar nuevos datos al medio y borrar el archivo al que se desee sustituir en caso de existir previamente.

5.4.4.3 Condiciones

Para generar la información DICOM, es necesario que se encuentre funcionando la interfaz con el MID para validar los datos DICOM obtenidos. Lamentablemente, este complemento

no estuvo dentro de los alcances de este proyecto, por lo que se omitió este punto dentro de esta prueba y se trabajó con lectura de archivos que no son DICOM.

5.4.5 Caso de prueba: Negociar transferencia de datos

5.4.5.1 Entradas

Dentro de esta prueba se debe garantizar la cobertura de los requerimientos funcionales descritos en la sección 5.1.1, ya que los datos de entrada deben estar bajo el formato DICOM. También se debe garantizar la cobertura de los estándares y protocolos de comunicación que rigen a cada medio físico que se desee implementar. En este caso serán el CD, DVD, USB y HD.

5.4.5.2 Resultado

Una vez verificada la petición de almacenamiento de medios físicos DICOM y validada la información que se manipulará, se tiene que entablar la comunicación con el medio físico previamente detectado (caso de prueba detectar medio) para iniciar la escritura (caso de prueba escritura), lectura (caso de prueba lectura) o actualización (caso de prueba actualización).

5.4.5.3 Condiciones

Para cubrir esta prueba, requieren haber sido finalizados los casos de uso de escritura, lectura, actualización y detección de medios así como sus casos de prueba.

5.4.6 Caso de prueba: Detectar medio

5.4.6.1 Entradas

Dentro de esta prueba se debe garantizar la cobertura de los estándares y protocolos de comunicación que rigen a cada medio físico que se desee implementar. En este caso serán el CD, DVD, USB y HD.

5.4.6.2 Resultado

Una vez ejecutada la prueba para negociar la transferencia, se deberá verificar que el medio al que se desea tener acceso cumpla con los parámetros especificados. Estos pueden ser formato de medio, capacidad de almacenamiento, espacio disponible, espacio utilizado, etc. En el caso de que no se cumplan los parámetros, el programa deberá corregirlo o en el caso de imposibilidad, notificar al usuario para cambiar de opción de medios físicos disponibles.

5.4.6.3 Condiciones

Para acceder al medio físico, se requiere tener control sobre éste. Esto es posible tomando control sobre sistema operativo sobre el cual esté operando.

5.4.7 Documentación de apoyo

Otro punto que se cubrió en esta última fase fue la generación de la guía del sistema para el usuario, que como ya se ha mencionado, sería un programador que intente integrar el almacenamiento DICOM en medios físicos y que es el JavaDoc, que se genera automáticamente por Eclipse y se muestran en el apéndice C de este documento.

5.5 Sumario

En este capítulo se presentaron los resultados correspondientes al desarrollo del proyecto (los que corresponden a la administración se mostraron en el capítulo anterior). La manera en que se muestran sigue el orden sugerido por RUP y en este caso por la instancia definida de éste marco de procesos complementado con la metodología de SUN Microsystems para desarrollo de software, la cual se definió en la sección 3.4.

Algunos modelos pueden no quedar tan claros si no se sigue el orden en que estos fueron generados, es decir, si el lector observa el diagrama de componentes sin ver antes el de clases, no sabrá de donde sale cada uno de los componentes y si ve el de clases no sabrá porque tiene las clases o porqué se llaman así si no observa el diagrama de robustez y así sucesivamente.

La guía de utilización del sistema, como ya se mencionó en la sección 5.4 se incluye como apéndice C en caso de que algún programador desee utilizar esta biblioteca dentro de algún sistema.

En el siguiente capítulo se discute, analiza y concluyen todos los puntos relacionados con el desarrollo de este proyecto, desde la utilización de la metodología, hasta el beneficio de la generación de la biblioteca como tal.

6. Discusión, conclusiones y trabajo futuro

6.1 Discusión

6.1.1 La importancia del producto final

La propuesta general de este proyecto fue la generación de una biblioteca para almacenamiento DICOM en medios físicos y los resultados demuestran que se obtuvo el producto planteado. Estos resultados obtenidos, además, permiten hacer énfasis en la relevancia de este proyecto para la informática médica, específicamente dentro del área de PACS. Recordemos que es una biblioteca de software que es de fácil integración dentro de un PACS, que es independiente de la plataforma en que funcione y que puede aceptar nuevos medios físicos para almacenamiento.

Todas estas características fueron cubiertas, sin embargo, hay que mencionar que de acuerdo a los objetivos planteados no se cubrió el 100% de ellos.

6.1.2.1 Objetivos

Lo importante es el logro de una biblioteca para almacenamiento en medios físicos que, de acuerdo al diseño arquitectónico, se puede extender de acuerdo a las necesidades particulares de un PACS. Sin embargo, la funcionalidad en otros sistemas operativos no fue probada, pero de acuerdo al diseño, solo faltaría crear algunas interfaces y utilizar controladores de medios particulares al sistema operativo para poder extender su uso en otras plataformas. En una primera estimación se puede considerar que se logró el 90% de los objetivos.

6.1.2 El uso de procesos de desarrollo de software

La aplicación de metodologías en el desarrollo de software ayuda a llevar control y dar seguimiento a la planeación. Un buen control y seguimiento de la planeación se logra a través de una adecuada administración de proyectos y mediante la definición de un proceso de desarrollo como modelo o guía para fabricar o implementar el software. Estos dos puntos justificaron el uso de RUP ya que funciona como un marco de referencia para procesos de desarrollo de software e integra las funciones administrativas para dar seguimiento a la planeación.

6.1.2.1 Planeación y seguimiento

La planeación de los objetivos y alcances, no necesariamente puede contemplar técnicas o guías de control y administración de las actividades y tareas para lograrlos. La ventaja que ofrece el RUP dentro de sus tareas administrativas es la capacidad de generar una instancia con base en las necesidades específicas del proyecto. Esto nos permitió en este caso, seleccionar una instancia no muy compleja para poder dar seguimiento a la planeación y por esta razón se omitieron algunos detalles que no dejan de ser importantes dentro de cualquier instancia pero que incrementan significativamente la complejidad para el seguimiento a la planeación, particularmente en este caso ya que se contaba con poca experiencia y con un sólo recurso para el desarrollo. Si se hubiera contado con más recursos humanos para el proyecto por ejemplo, la instancia podría haber abarcado más disciplinas para generar un producto de mejor calidad.

Los resultados obtenidos desde el punto de vista administrativo, mostrados en el capítulo 4, permitieron dar un seguimiento adecuado de la planeación y eventualmente corregir el rumbo midiendo alcances y mitigando riesgos.

Para hacer más evidente el impacto que tuvo la integración de la administración de proyectos dentro del proceso podemos hacernos la siguiente pregunta: ¿Qué hubiera podido pasar si no se generaba un plan de desarrollo y seguimiento al proceso? Una posibilidad es que si el planteamiento de los alcances hubiera sido incorrecto no hubiera habido convergencia de los resultados con los requerimientos del estándar DICOM. En cambio, esta manera de trabajar arrojaría rápidamente alertas de no convergencia en unas cuantas iteraciones ya que es un proceso incremental. Otra posibilidad es que los roles no cubrieran sus objetivos y tareas específicas. Esto daría como resultado un retraso general en el proyecto, pero en este caso, al tratarse de un proceso iterativo incremental, se pueden ejecutar planes de prevención o contingencia para no retrasar la calendarización y resolver los problemas de manera puntual.

6.1.2.2 Definición del proceso de desarrollo

En cuanto a la definición del proceso de desarrollo hay también elementos que merecen ser subrayados ya que permitieron generar un producto final que resuelve la funcionalidad deseada definida por los requerimientos funcionales y no funcionales respectivamente.

La instancia seleccionada a partir del RUP para el desarrollo incluye para cada iteración el análisis, diseño, implementación y pruebas. A través de estas disciplinas fue posible generar una serie de productos que ayudaron a definir una arquitectura basada en los requerimientos funcionales que se deseaban cubrir. Esta forma de guiar el desarrollo permitió que esta misma arquitectura estuviera fundamentada en los requerimientos no funcionales que definieron las propiedades del sistema. La primera propiedad es la de flexibilidad, por ejemplo, si en un futuro el mercado saca a la venta un nuevo tipo de dispositivo de almacenamiento, es posible integrarlo modificando algunas reglas de sintaxis y formato para reconocer el nuevo medio sin afectar el resto del código y la funcionalidad de la biblioteca. Otra característica es la portabilidad ya que también es posible aumentar la integración a aplicaciones de tipo PACS dentro de distintos sistemas operativos de forma muy clara.

El uso de un conjunto de herramientas seleccionadas de forma adecuada también tuvo un impacto importante en el desarrollo de la biblioteca de software ya que se agilizó la generación de productos específicos como documentos, versiones, modelos, etc. Si no se hubieran definido este conjunto de herramientas lo más probable es que se hubieran generado retrasos en la búsqueda de las mismas modificando fechas de revisión. Por el contrario, puede caber la posibilidad de que en un futuro proyecto se definan herramientas más útiles o que signifiquen mejor eficiencia para la obtención de resultados e incluso para auxiliar la planeación y su seguimiento.

Los resultados obtenidos de la parte de desarrollo fueron muy buenos ya que ofrecen una guía muy clara de la manera en que se llega al producto final mediante modelos, código y documentación. También dejan ver de manera muy clara, en conjunto con los resultados administrativos, el curso del desarrollo del proyecto desde su inicio hasta su término con el

fin de que cualquier persona conozca la forma de trabajar y pueda tomarla como referencia o bien pueda continuar el trabajo previo.

6.2 Conclusiones

6.2.1 Utilidad de la biblioteca dentro de una aplicación tipo PACS

El desarrollo de esta biblioteca se plantea como una herramienta muy práctica para los objetivos del desarrollo de aplicaciones tipo PACS basadas en componentes, ya que es posible incluirla bajo un esquema determinado a un sistema de administración de imágenes médicas de manera independiente a los demás servicios DICOM. Esto abre muchas posibilidades hacia el área médica debido a que permite resolver necesidades de manejo de imágenes médicas DICOM, muy particulares, a través de aplicaciones tipo PACS y ofrece muchas ventaja sobre varios sistemas implementados, como la portabilidad, flexibilidad, extensibilidad, etcétera. Este beneficio impacta en proyectos actuales y futuros de desarrollo aplicados a necesidades reales en el área médica para ofrecer beneficios en un mercado complicado y competitivo.

6.2.2 Experiencia en el uso de metodologías

A pesar de que se definió una instancia del RUP, se concluye que esta puede ser refinada para futuros proyectos a desarrollar en el Laboratorio de Investigación de Informática Médica así como por el postulante en su desempeño profesional.

Un aspecto que se debe tomar en consideración para proyectos de desarrollo futuros con una metodología definida es evaluar las habilidades de los recursos humanos para que puedan ejecutar las tareas y actividades específicas relacionadas en su rol de forma eficiente, sobre todo si se trata de proyectos grandes o ambiciosos. En este caso en particular, hubo algunos detalles relacionados con intereses, habilidades y propósitos que tuvieron impacto (no necesariamente positivo) en el desarrollo del proyecto. Por ejemplo, para este proyecto en donde un solo recurso adopto todos los roles, las actividades de planeación y su seguimiento, requerían cierta experiencia y habilidades administrativas para poderlas realizar de manera eficiente, por esta razón se generaron algunos retrasos y modificaciones en la planeación original. Además, también puede volverse un gran problema la falta de dedicación requerida para obtención de resultados cuando no se conoce el impacto de la realización de las actividades correspondientes, ya que puede no cubrirlas con el detalle requerido y afectar los productos y las actividades de otros roles, incluso puede volverse mucho más complicado cuando no se contemplan todos los riesgos o ni si quiera se tiene la habilidad de detectarlos.

Otros detalles que tuvieron un impacto fundamental en los retrasos relacionados con la conclusión de este proyecto, en particular con el documento de tesis, fue la falta de planeación para la organización de la estructura del documento así como la falta e incumplimiento de acuerdos para su revisión, dado que se contó con dos asesores para este fin y se anticipaba que no sería tarea sencilla por cuestiones de coordinación. La sugerencia que puede derivarse de esta experiencia es que se debería llevar tal vez una mejor planeación y un seguimiento más estricto.

6.2.3 Impacto del proyecto en la formación personal académica y profesional

Desde el punto de vista personal este proyecto fue de mucha utilidad para comprender la importancia de la administración del proyecto, incluso aun cuando no necesariamente sea de desarrollo de software. El haber implantado una instancia de un proceso para convertirla en la metodología y luego llevar la administración del mismo generó experiencia y será de mucha ayuda para siguientes proyectos tanto para formación académica como profesional y personal. La culminación de este proyecto y su análisis en retrospectiva permite hacer un resumen de lo que debe ser repetido, mejorado, omitido e incluso agregado para proyectos futuros de propósitos personales. De la misma manera permitió una mejor valoración y descubrimiento de habilidades, deficiencias y necesidades que deberán tomarse en cuenta para usarlas en beneficio de las futuras actividades con el fin de buscar la eficiencia y el éxito.

6.3 Trabajo futuro

6.3.1 Sobre la biblioteca

Para incrementar la utilidad de esta biblioteca de software, es necesario extender su funcionalidad para nuevos medios, extenderla a nuevas plataformas e integrarla en sistemas tipo PACS. Estos tres rumbos pueden generar trabajos posteriores con distinto nivel de complejidad.

6.3.2 Sobre la implementación de modelos

El modelo utilizado para este proyecto, como se mencionó anteriormente, fue sólo una instancia del conjunto de componentes que contiene el RUP, por lo que pudiera pensarse en usar con más o menos componentes este mismo modelo para otros proyectos similares de desarrollo o incluso buscar alguna adaptación para proyectos que no sean propiamente de desarrollo ya que en general todos los proyectos pueden tener requerimientos administrativos de este tipo.

6.3.3 Sobre la aplicación en PACS

Retomando las posibilidades de integración a un PACS mencionadas en el punto 6.3.1 esta solución puede integrarse para probar un sistema con parte de la funcionalidad distribuida para fines específicos dentro de una clínica u hospital. Esta opción podría pensarse más como para un proyecto de desempeño o de integración con otras aplicaciones o bibliotecas para probar la funcionalidad completa del sistema.

Apéndice A - Guía para escribir requerimientos

A.1 Antecedentes

A.1.1 Características de calidad deseables para un SRS (especificación de requerimientos de un sistema)

Las características deseables para especificaciones de requerimientos son:

- Que se encuentre completo
- Que sea consistente
- Que esté correcto
- Que no contenga ambigüedades
- Que sea compacto
- Que sea modificable
- Que sea trazable
- Que sea verificable

A.1.2 Indicadores de Fortaleza y Debilidad

Casi todos los atributos de calidad de los requerimientos del documento son subjetivos, existen aspectos de la documentación que pueden ser medidos y que son indicadores de atributos que se relacionan con la calidad. El tamaño, el cual es una medida primaria para medidas de calidad, y se puede medir directamente. El tamaño del documento de requerimientos puede ser fácilmente medido por número de páginas, o de párrafos, líneas de texto o por el número de declaraciones de especificaciones individuales que contiene. La profundidad de jerarquía que engloban las declaraciones de especificaciones del documento pueden ser medidas usando el esquema de identificación interno del documento. Estas medidas proveen pistas para la organización del documento y su profundidad con detalle. El número de declaraciones de especificaciones identificado como único para cada nivel de estructura de documento también puede ser cuantificado. Estos conteos proveen una indicación de cómo las declaraciones de especificación están organizadas y el nivel de detalle al cual se especifican los requerimientos. También es posible contar la ocurrencia de palabras y frases específicas que indiquen si las declaraciones de especificación son fuertes o débiles.

A.1.3 Categorías de los Indicadores.

Se establecen nueve categorías de requerimientos e indicadores de calidad de declaración de especificaciones basados en un grupo de documentos de requerimientos de la NASA seleccionados por su estructura. Los indicadores individuales se identifican encontrando palabras, frases y estructuras usadas frecuentemente y otros aspectos de los documentos seleccionados que están relacionados a la calidad de los atributos y pueden ser fácilmente identificados. Estos indicadores individuales se agrupan de acuerdo a sus características indicativas. Las categorías resultantes caen en dos clases. Aquellas que se encuentran relacionadas con las declaraciones de las especificaciones individuales y las que están

relacionadas con el documento de requerimientos totales. Las categorías relacionadas con las declaraciones de especificaciones individuales son:

1. Imperativas (debe, no debe, requiere, es aplicable, hará, etc.)
2. Aplazamientos (abajo:, como sigue:, listado:, en particular:, soporta:, etc.)
3. Directivas (figura, tabla, por ejemplo, nota)
4. Opciones (puede, opcionalmente, etc.)
5. Frases Débiles (como mínimo, aplicable, fácil, apropiado, ser capaz de, adecuado, etc.)
6. Las categorías de los indicadores relacionadas al documento de requerimientos totales son:
7. Tamaño (líneas de texto, imperativas, sujetos de declaración de especificación, párrafos)
8. Estructura de texto
9. Especificación de profundidad del documento
10. Legibilidad

A.2 Análisis de Procesos de medición de requerimientos automatizada (ARM)

La herramienta ARM se enfoca en los identificadores de declaración para analizar en nivel de estructura (que tan profundo es) del texto. Si se presenta una imperativa, la línea de texto es considerada como una declaración de especificación. Las palabras inmediatamente después de la imperativa se consideran el sujeto de la especificación. Las declaraciones de especificación se buscan para identificar continuidades y frases débiles.
En el siguiente ejemplo, la herramienta ARM graba una línea de texto en el nivel 3, un requerimiento en el nivel 3 que contiene dos continuidades y dos frases débiles, y tres requerimientos en el nivel 4.
El sistema XYZ debe tener la capacidad de generar reportes mostrando información detallada acerca del calendario de mantenimiento para hardware, software, incluyendo como mínimo:

- Calendario de mantenimiento de rutina
- Calendario de mantenimiento no rutinario
- Calendario de mantenimiento actualizado

A.2.1 Objetivos del SRS

El objetivo del SRS es definir las capacidades que satisfacen una necesidad o problema. Con el fin de hacer esto, la fase de requerimientos de un proyecto usualmente consiste en dos actividades traslapadas. La actividad inicial de la fase de requerimientos es estudiar la empresa estratégicamente profundamente para activar la misión necesaria que se debe definir para un análisis más a fondo. Una vez que el medio y la necesidad del problema son entendidos del todo, son analizados con el fin de postular una solución a la necesidad o problema. La solución se documenta en términos de aquellos aspectos del medio que inciden en la solución y los requerimientos que se deben satisfacer con el fin de llegar a una solución satisfactoria.

A.3 Contenido del Documento

A.3.1 Tópicos de especificaciones de requerimientos.

La IEEE y otros estándares resueltos por organizaciones diferentes han identificado nueve tópicos que se necesitan tomar en cuenta cuando se especifica un sistema o requerimientos de software, estos son:

1. Interfaces
2. Capacidades Funcionales
3. Niveles de Funcionalidad
4. Elementos de Estructuras de Datos
5. Seguridad
6. Fiabilidad
7. Privacidad
8. Calidad
9. Limitaciones

Los primeros 4 tópicos se enfocan en requerimientos ingenieriles asociados con los elementos individuales de la capacidad necesaria. Los cinco restantes se enfocan en requerimientos que engloban todos los elementos de la capacidad necesaria.
Estos 9 tópicos no son sujetos aislados y la información se almacena en varios niveles. Por ejemplo, las funciones, interfaces y los datos están ligados fuertemente. El asunto es como tratar estos tópicos para que las relaciones se establezcan claramente pero sin mucha redundancia.

A.3.2 Propósito del SRS

El propósito general del SRS es proveer un medio para capturar y comunicar a las partes interesadas que es necesario. El significado más significante es servir como un contrato entre el usuario y la capacidad proveedora definiendo que se va a proveer, la manera en la que se va a producir y la tecnología que este incorpora. En adición, el SRS provee una base

para la mayoría de manejos de proyectos y funciones de ingeniería tal como asesoría ingenieril con propósitos de cambios, desarrollo de pruebas de requerimientos, escritura del manual de usuario preliminar, planear las actividades de mantenimiento y soporte e implementar mejoras operacionales. Con el fin de servir a estos propósitos, la SRS debe proveer una definición del futuro operacional y lo medios de soporte en los cuales la capacidad requerida existirá así como la provisión de una prescripción para el sistema que intenta realizar como la solución de la necesidad o problema. Si es necesario el SRS también debe identificar las metodologías y tecnologías que serán usadas durante el ciclo de vida del desarrollo.

A.4 El Problema Estructural

Con el fin de ser efectivo, el SRS debe describir exactamente los medios estratégicos y tácticos y la solución del sistema dentro de estos medios. Las descripciones de estos medios deben dar dirección a los elementos de las situaciones de soporte y operacionales que afecten la solución como se espera para que existan mientras dure el tiempo de vida de la solución operacional del sistema. Es necesario distinguir entre aquellos aspectos de los medios (estratégicos, tácticos, operacionales y de soporte) que son continuaciones de las situaciones existentes y aquellos que se espera que sean provistos en el futuro por el proveedor o por otras fuentes. En adición, el SRS debe dar dirección a los nueve tópicos fundamentales que son relevantes para cada uno de los medios.

El diseño del documento SRS debe estar basado en una aproximación predeterminada para estructurar la información de especificaciones en secciones y párrafos. La información nunca debe ser agrupada arbitrariamente. Esto no solo hace el documento difícil de entender, sino que también lo hace difícil de mantener (actualizar). Las declaraciones que son parte de una simple función son normalmente agrupadas. Las funciones que se conectan en series por relaciones de salida o entrada deben aparecer en el documento en la misma secuencia. Las funciones que comparten entradas y salidas comunes deben tener una dirección dentro de la misma área del documento. Si varios procesos deben ser logrados en el mismo intervalo de tiempo, sus especificaciones deben juntarse por el documento para hacer al mismo más claro. Las funciones que son similares necesitan ser distinguidas unas de otras, pero las similitudes deben ser enfatizadas.

A.4.1 Estructura de Declaración

Las declaraciones de requerimientos pobremente estructuradas resultan en especificaciones confusas que son propensas a interpretaciones incorrectas. Una declaración de ese tipo es la siguiente:

"El sistema XYZ debe proveer una información de varianza comparativa oportuna, ser detallada lo suficiente para que las varianzas individuales importantes no sean despreciadas a causa de que se cancelen entre sí, localizar la fuente de cada varianza e indicar el área de investigación que maximice los beneficios globales".

Esta especificación puede ser entendida más fácilmente si se estructura como sigue:

- 5.1 El sistema XYZ debe proveer la información de varianza comparativa.
    - 5.1.1 La información de varianza comparativa debe ser oportuna

- 5.1.2 La varianza comparativa debe tener el detalle suficiente para:
  - 5.1.2.1 Prevenir las varianzas individuales importantes para no ser canceladas.
  - 5.1.2.2 Localizar la fuente de cada varianza
  - 5.1.2.3 Indicar el área de investigación que maximiza los beneficios globales.

Cada declaración de especificación consiste en cuatro elementos básicos para la estructura: Entidades, Acciones, Eventos y Condiciones. Estos elementos pueden ser usados o modificados por varios casos como:

- Propietario
- Actor
- Objetivo
- Limitación
- Propiedad
- Acción
- Objeto
- Localización

El modelo recomendado para una estructura de especificaciones es: [Localización] [Actor | Propietario] [Acción] [Objetivo | Propiedad] [Limitación]
Por ejemplo: "7.1 Cuando tres o más localizadores de estrellas pierdan la referencia de las estrellas, la nave debe alinearse inmediatamente su eje principal a la línea Tierra-Sol a menos que la tapa de los instrumentos ópticos no se encuentre cerrada".
Localización: "Cuando tres o más localizadores de estrellas pierdan la referencia de las estrellas.
Actor | Propietario: "Nave" Acción: "Alinear"
Objetivo | Propiedad: "Eje Principal"
Limitante: "A menos que la tapa de los instrumentos ópticos no se encuentre cerrada"

A.5 El problema del Lenguaje

Se debe prestar atención al rol de cada palabra y frase cuando se está formulando la declaración de especificación. Las frases y las palabras que no se seleccionan o colocan con atención producen ambigüedades e imprecisiones en las especificaciones. Ejemplos de las especificaciones que contienen una selección pobre de palabras se muestra a continuación:
"El sistema debe ser amigable". Puede ser interpretado subjetivamente y su implementación será difícil para probar objetivamente.

"Los Usuarios que intenten acceder a la base de datos ABC se les deberá recordar por un mensaje de sistema que sea reconocido y cabeceras de página en todos los reportes que los datos sean sensibles y el acceso es limitado por sus privilegios de sistema".

A.6 Recomendaciones

La efectividad de expresar las especificaciones de requerimientos con un lenguaje natural puede ser mejorada mediante métodos relativamente simples. Los directores de proyectos deben asegurarse que los estándares para las especificaciones de requerimientos y el estilo sean establecidos desde el principio y que los participantes del proyecto estén capacitados en el uso de estos estándares. También se recomienda que la naturaleza técnica y el uso intencionado del documento SRS hagan énfasis a un estilo de escritura simple y estructuras simples de sentencia para especificar cada requerimiento. Los documentos técnicos no requieren ser interesantes, sin embargo, es necesario que se comuniquen efectivamente con el lector. Algunas directrices para producir un SRS que sea comunicativo son: - Palabras y Frases - Proveer Ejemplos - Citar Referencias - Usar Tablas y Gráficos - Sumario y Conclusión

Apéndice B - Javadoc

B.1 Introducción

El Javadoc es una herramienta para analizar los comentarios, documentación y declaraciones en un grupo de archivos fuente y produce un conjunto de páginas HTML describiendo las clases, interfaces, constructores, métodos y campos. Este grupo de archivos fue generado por la herramienta StarUML. Esta herramienta genera páginas con la documentación API a partir del código fuente de Java.

El Javadoc generado para este proyecto se encuentra disponible en el Laboratorio de Informática Médica dentro del Departamento de Ingeniería Eléctrica junto con el código fuente de la biblioteca y los ejecutables como tal.

B.2 Contenido del API

Este documento parte de una guía que contiene los paquetes de clases de la biblioteca. Esta guía se muestra en la Figura B.1.

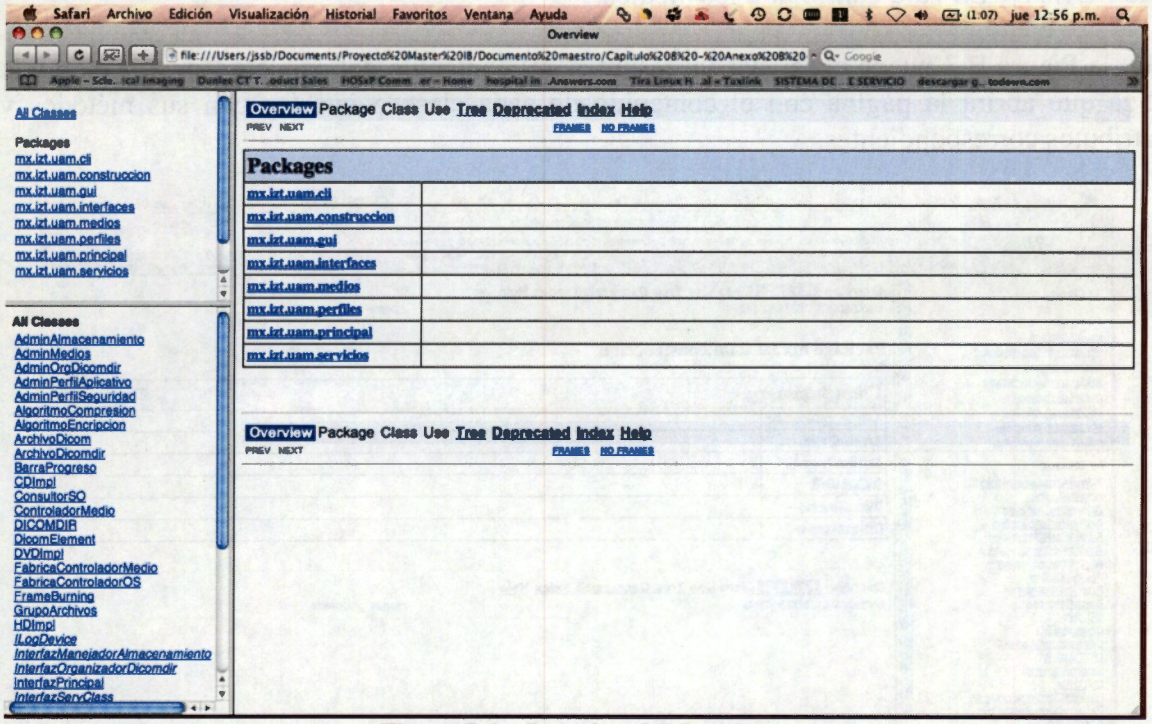

Figura B.1 - Overview del Javadoc

Cada paquete contiene la información de las clases contenidas en el mismo. Dentro de la tabla de paquetes de la guía (overview) mostrada en la Figura B.1 pueden observarse las ligas a cada uno de estos paquetes.

En la Figura B.2 puede observase un resumen del contenido del paquete Cli y la liga que abrirá la página con el contenido de cada clase o interfaz con sus métodos y atributos correspondientes.

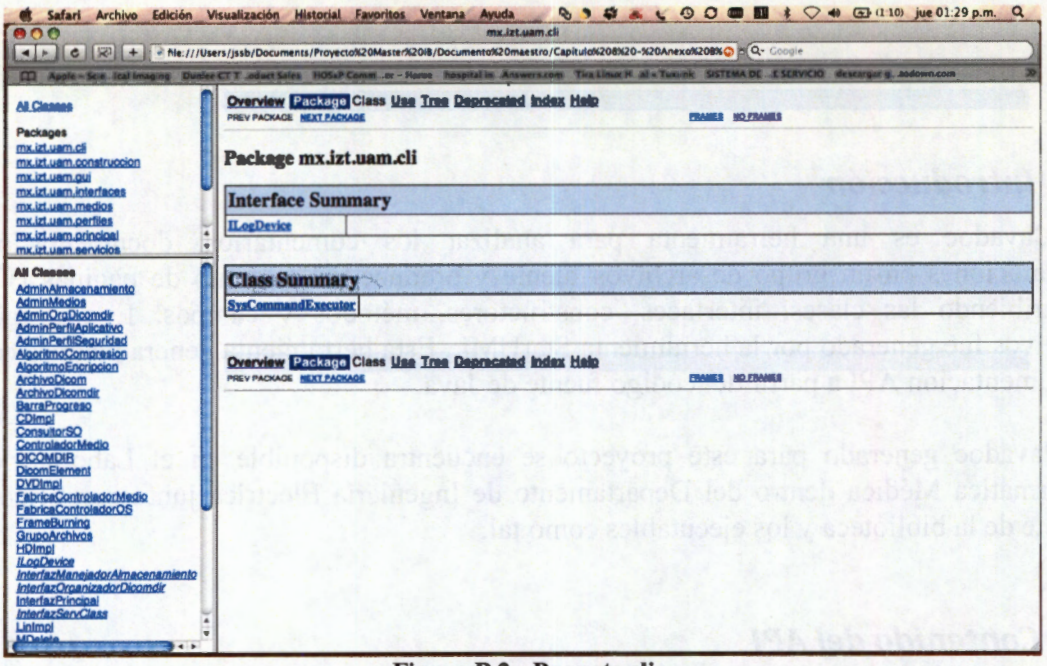

Figura B.2 - Paquete cli

Este conjunto de clases tienen primordialmente la función de interactuar con la biblioteca de CDRTOOLS para controlar a los medios.

En la Figura B.3 puede observase un resumen del contenido del paquete Construcción y la liga que abrirá la página con el contenido de cada clase o interfaz con sus métodos y atributos correspondientes.

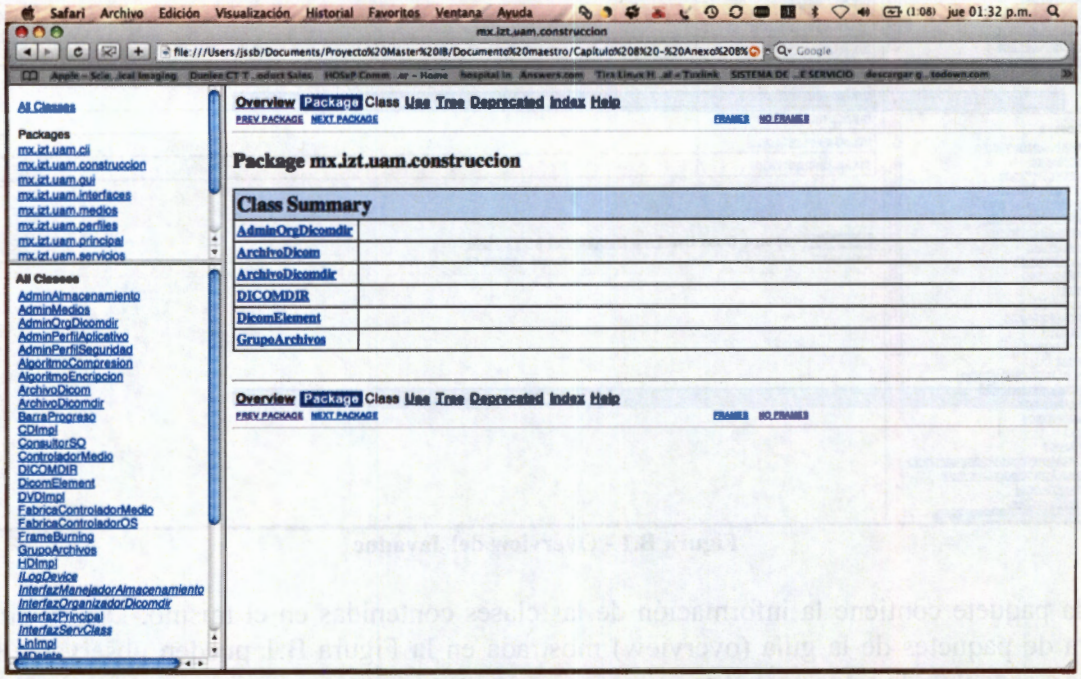

Figura B.3 - Paquete construcción

Este paquete y las clases contenidas en él tienen la función de generar y validar los archivos y el DICOMDIR para almacenamiento en medios basado en el estándar DICOM a través de la interfaz MID.

En la Figura B.4 puede observase un resumen del contenido del paquete Gui y la liga que abrirá la página con el contenido de cada clase o interfaz con sus métodos y atributos correspondientes.

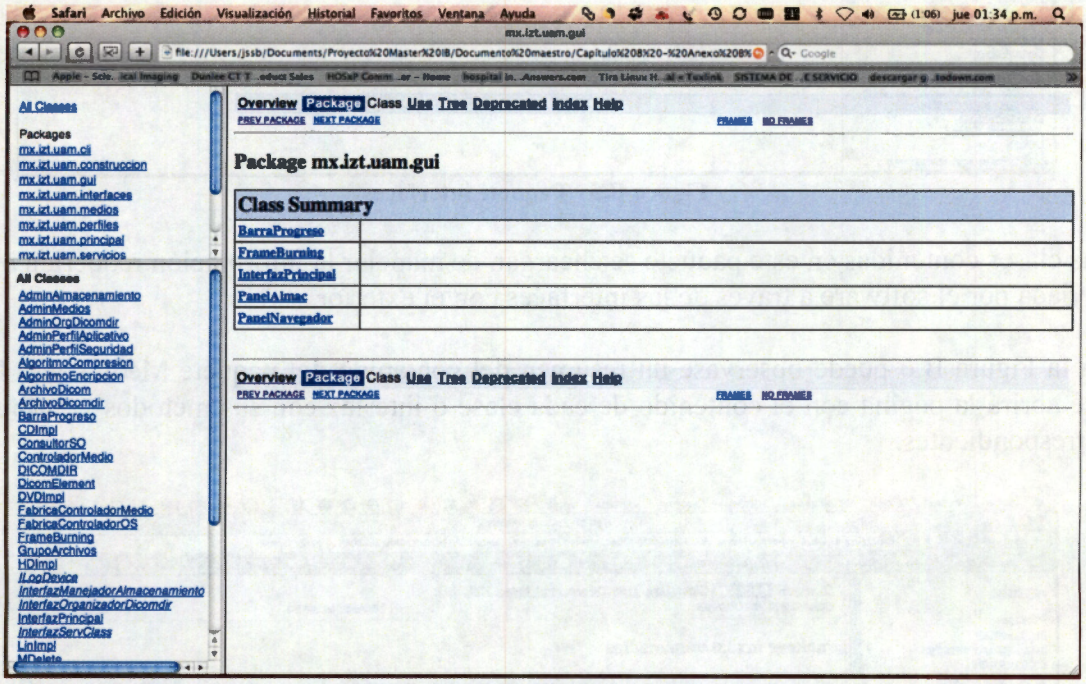

Figura B.4 – Paquete GUI

Este paquete contiene clases que conforman una interfaz para usuario gráfica con el simple fin de probar funciones, no está dentro del alcance del proyecto. Se generó con fines de realización de pruebas generales.

En la Figura B.5 puede observase un resumen del contenido del paquete Interfaces y la liga que abrirá la página con el contenido de cada clase o interfaz con sus métodos y atributos correspondientes.

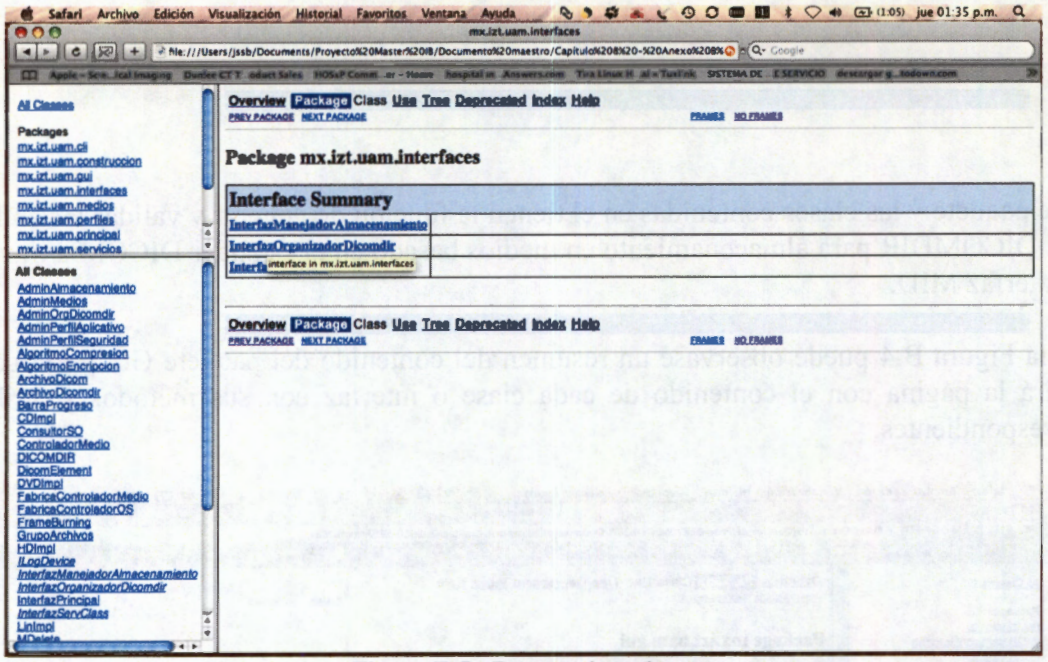

Figura B.5 - Paquete interfaces

Las clases contenidas en este paquete se encargan de manejar la información requerida o enviada por el software a través de las interfaces con el exterior.

En la Figura B.6 puede observase un resumen del contenido del paquete Medios y la liga que abrirá la página con el contenido de cada clase o interfaz con sus métodos y atributos correspondientes.

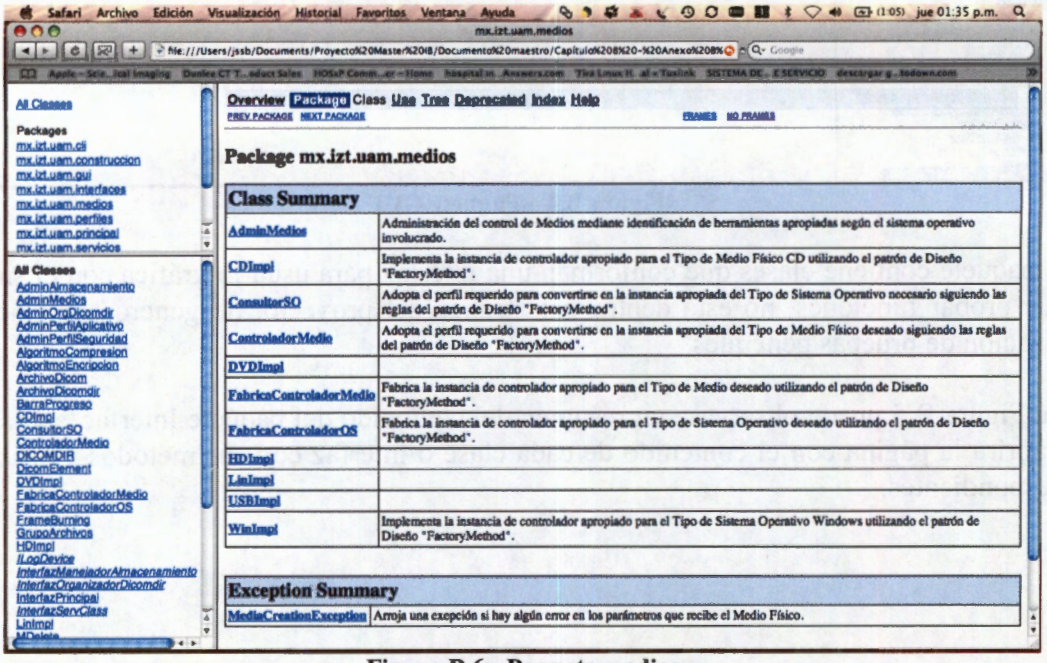

Figura B.6 - Paquete medios

Este paquete contiene clases que tienen como función principal manejar la información necesaria para conexión de los medios ya sea a través del sistema operativo (acceso directo) o a través de la interfaz con la biblioteca CDRTOOLS.

En la Figura B.7 puede observase un resumen del contenido del paquete Perfiles y la liga que abrirá la página con el contenido de cada clase o interfaz con sus métodos y atributos correspondientes.

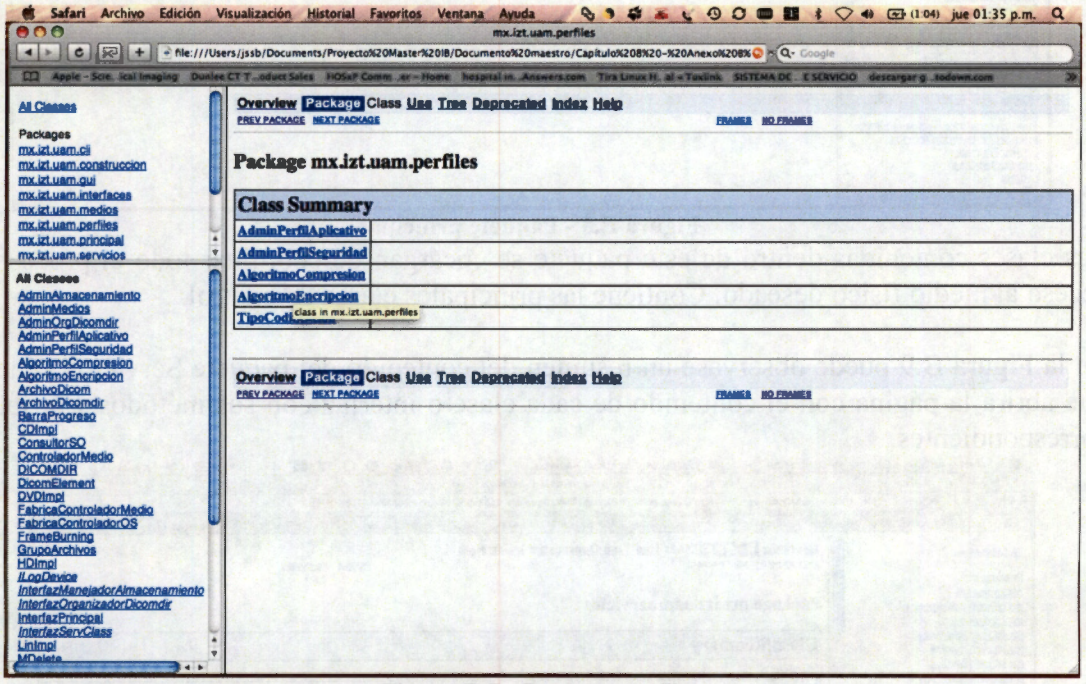

Figura B.7 - Paquete perfiles

Este paquete contiene un conjunto de clases que tienen como función principal generar y validar los perfiles de aplicación con el contenido de acciones para acceso a medios bajo el estándar DICOM.

En la Figura B.8 puede observase un resumen del contenido del paquete Principal y la liga que abrirá la página con el contenido de cada clase o interfaz con sus métodos y atributos correspondientes.

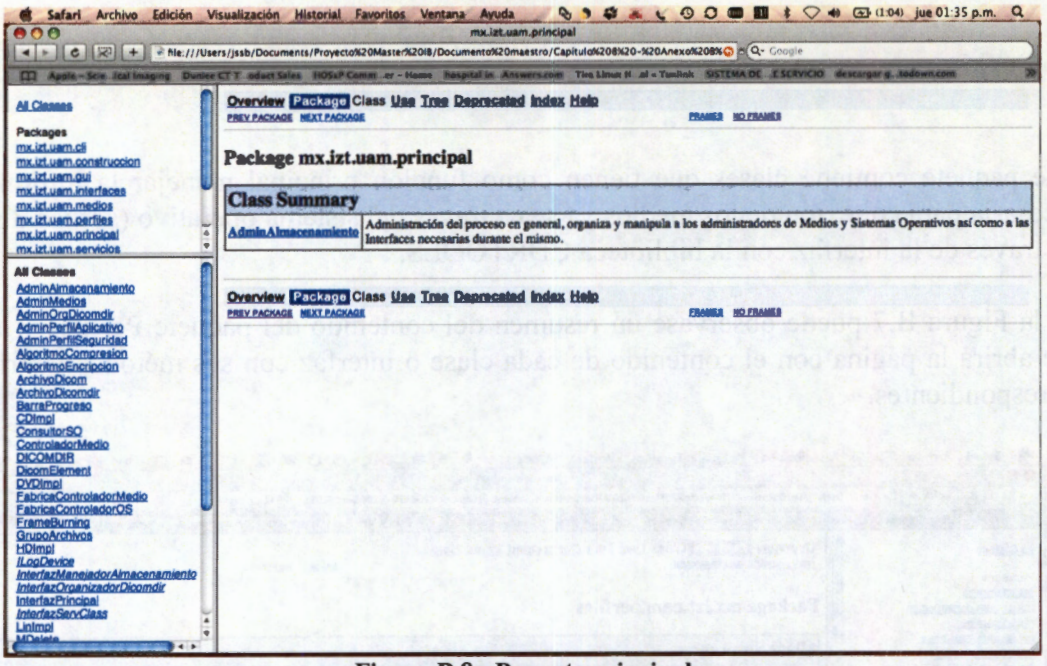

Figura B.8 - Paquete principal

Las clases contenidas dentro de este paquete se encargan de coordinar todo el proceso de acceso al medio físico deseado. Contiene las principales clases de control.

En la Figura B.9 puede observase un resumen del contenido del paquete Servicios y la liga que abrirá la página con el contenido de cada clase o interfaz con sus métodos y atributos correspondientes.

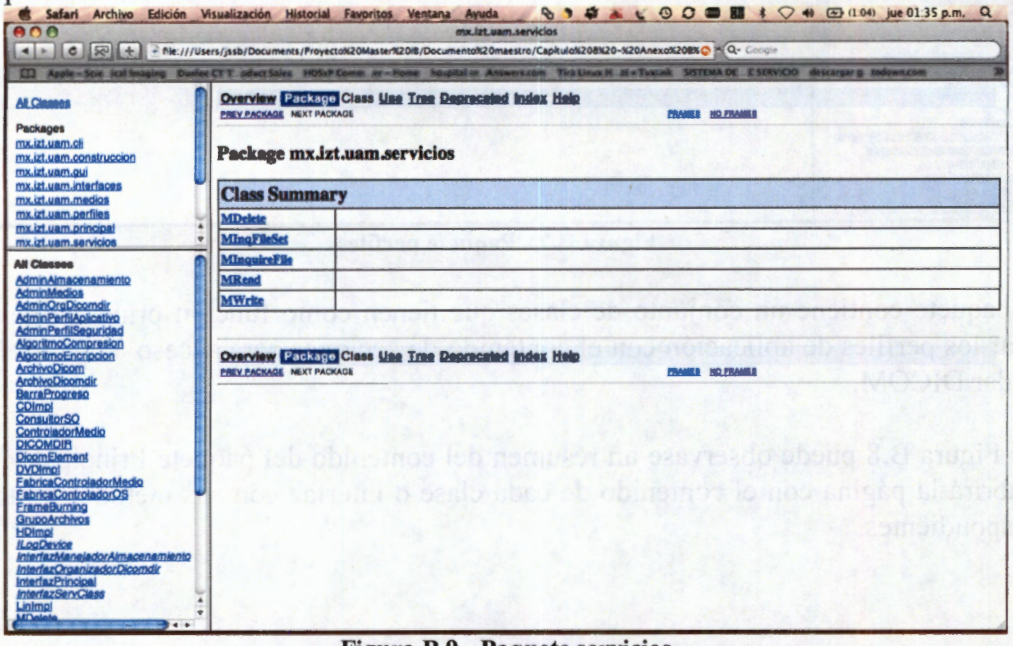

Figura B.9 - Paquete servicios

Dentro de este paquete se encuentran las clases que contienen los servicios DICOM para escritura, actualización y lectura bajo el estándar DICOM.

Apéndice C - Glosario

ACR: Siglas de American College of Radiology.

Actualizador de grupo de archivos: Entidad de aplicación que accede a archivos, crea archivos adicionales o borra archivos existentes en un grupo de archivos. Un actualizador de grupo de archivos realiza las alteraciones apropiadas al archivo DICOMDIR reflejando las adiciones o eliminaciones.

AE: VR de tipo Entidad de Aplicación.

Almacenamiento: Acción y efecto de almacenar.

ANSI: Siglas de American National Standards Institute.

Archivo DICOM: Un archivo es una cadena ordenada de cero o más bites, en donde el primer byte se encuentra en el principio del archivo y el último al final. Los archivos se identifican por un ID de archivo único y pueden ser escritos, leídos y/o borrados. La estructura de archivos DICOM se encuentra definida en la parte 10 del estándar DICOM. Ver ID de archivo.

ARTIM: Requisición de asociación / Rechazo / Resetear Timer (Association Request / Reject / Release Timer).

AS: VR de tipo Age String.

Asociación: Una conexión de red.

Atributo de imagen: Propiedad del Objeto de Información DICOM. Un atributo tiene un nombre y un valor que son independientes de cualquier esquema de codificación.

Big Endian: Forma de ordenar bytes en donde los valores binarios se codifican con el byte más significativo al principio y los bytes restantes en orden descendente. Ver también Little Endian.

Bits Alojados: El elemento (0028, 0100) especifica el número total de bits alojados para el alojamiento de la muestra de pixeles. Ver también Atributos de imagen.

Bits Almacenados: El elemento (0028, 0101) especifica el número real de bits usados para almacenar los pixeles. Los bits almacenados nunca deben ser de mayor tamaño que los bits alojados. Ver también atributos de imagen.

Byte Swapping: Conversión de datos de Little a Big Endian o viceversa.

Called AE Title: La entidad de aplicación del SCP que trata de encontrar un SCU para abrir una asociación.

Calling AE Title: La entidad de aplicación del SCU que trata de abrir una asociación con un SCP.

Cliente: la aplicación que inicia la asociación DICOM con un "servidor".

Cliente-Servidor (dentro de una Red): Término para roles específicos, equivalentes al SCU y SCP en DICOM.

Comando: Una petición de operación o información a través de la red.

Composite Information Object Definition (C-IOD): Definición que representa partes o varias entidades en un modelo de aplicación DICOM. Esta IOD incluye atributos que son inherentes a los objetos del mundo real que el IOD representa.

Conformance Statement (Declaración de conformidad): Documento provisto por los fabricantes de la compatibilidad del equipo con DICOM que identifica el conjunto de estándares que soporta cada producto individualmente.

Contexto de aplicación: Identifica el contexto entero de comunicaciones.

Creador de grupo de archivos: Entidad de aplicación que crea el archivo DICOMDIR y cero o más archivos DICOM.

Data Element (Elemento de datos): La unidad más pequeña de información como se define en la parte 3.6 del estándar. Cada elemento de datos contiene un atributo del objeto en el mundo real.

Data Element Tag: Un número de 32 bits que identifica únicamente a un elemento de datos (data element).

Dataset (grupo de datos): Información que consiste en un grupo estructurado de valores de atributos directa o indirectamente relacionados con la información de objetos. El valor de cada atributo es expresado como elemento de datos (data element).

DA: VR de tipo fecha (date).

DICOM: Acrónimo de Imagen Digital y Comunicaciones en Medicina por sus siglas en inglés "Digital Image and Communications in Medicine".

DICOMDIR: Archivo único y de referencia DICOM con el grupo de archivos que contiene el directorio de clases SOP de almacenamiento en medios.

Diccionario de datos: Registro de elementos de datos DICOM que asignan un único Tag, descripción, características de valor y semántica a cada elemento de dato. Éste se lista en la parte 3.6 del estándar.

DICOM Upper Layer (capa superior): Los protocolos de capa superior se relacionan a la sesión, presentación y parte de la capa de aplicación del modelo de referencia ISO. Estos protocolos proveen un servicio de capa superior.

DIMSE (Message): Es el mensaje que envía un comando DICOM y datos relacionados. Los tipos de mensaje son:

- DIMSE-C - DIMSE Compuesto
- DIMSE-N - DIMSE Normalizado.

DDL (Digital Driving Level): Un valor digital dado como entrada a un sistema de despliegue que produce luminiscencia. El grupo de DDL´s de un sistema de despliegue, son todos los posibles valores que pueden producir los valores de luminiscencia en este sistema. El mapeo de los DDL´s a valores de luminiscencia para un sistema de despliegue produce la curva característica de este sistema. La salida actual para un DDL es específica del sistema de despliegue y no está corregida la función de despliegue de niveles de gris estándar.

DS: VR de tipo Decimal String.

DT: VR de tipo Date Time.

Encriptación: Proceso para convertir la información a un formato más seguro mediante un proceso matemático a un formato encriptado o codificado.

Entidad de Aplicación: Nombre que se le da a un agente DICOM dentro de una red. Consiste en un código de caracteres ASCII y está limitado a una longitud de 16. El Nombre de la entidad de aplicación no es el mismo que el del "host".

Explicit VR: Ver el valor de representación (Value Representation o VR).

FD: VR de 8 bytes de tipo Floating Point Double.

Formato de archivo DICOM: Agrupación de la información DICOM dentro de un conjunto de datos. Los archivos DICOM consisten en una cabecera con campos estandarizados y de forma libre, y un cuerpo con datos de imagen.

Formato de medio: Estructura de datos y políticas asociadas que organizan los bits definidos por el medio físico en las estructuras de archivos y directorios.

Formato de sintaxis de transferencia: Es el formato que se genera para entablar una comunicación entre dos entidades bajo las reglas de sintaxis para presentar la información.

FL: VR de tipo Floating Point Single (IEEE 754:1985) 4 bytes.

Grupo de archivos: Una colección de archivos DICOM (y posiblemente de archivos no DICOM) que comparten un espacio común en donde su id de archivo es único.

Grupo de elemento de comando (0000): Elementos que representan un segmento del comando de un mensaje DIMSE.

ID de archivo DICOM: Identificador de archivo único que contiene el contexto del grupo de archivos al cual pertenece.

ID de archivo DICOMDIR: Identificador de archivo único que contiene el contexto de la ubicación que tiene.

Information Entity (entidad de información): Porción de la información definida por un IOD la cual se relaciona a la clase de objetos del mundo real. Existe una correspondencia uno a uno entre las entidades de información y las entidades del modelo de aplicación DICOM.

Information Object (objeto de información): Conjunto de datos que describen la estructura, función y atributos de cada objeto que contenga información DICOM

Information Object Definition (definición de objeto de información o IOD): Número de módulos de entidad de información relacionados lógicamente agrupados. Por ejemplo un modulo IOD de una imagen CT consiste en el modulo de paciente, modulo de estudio, modulo de imagen, etc.

IS: nombre VR para Integer String

Lector de grupo de archivos: Entidad de aplicación que accede a los archivos, crea archivos adicionales o borra algún archivo existente en un grupo de archivos.

Little Endian: Forma de ordenar bytes en donde los valores binarios se codifican con el byte menos significativo al principio y los bytes restantes en orden ascendente. Ver también Big Endian.

LT: VR de tipo Long Text.

LUT (Look up table): Tabla de colores de la imagen.

Medio físico: medio en con capacidad de almacenar información digital a través de distintos mecanismos, por ejemplo el DVD es un medio óptico para almacenamiento.

Meta elementos de archivo DICOM: Esta información se encuentra encapsulada dentro de la información del grupo de datos (dataset).

Meta información de archivo DICOM: Esta información se encuentra encapsulada dentro de la información del grupo de datos (dataset).

Método de encriptación: Método matemático para codificar la información digital de manera segura

Mensaje: Unidad de datos del protocolo de intercambio de mensajes entre dos aplicaciones DICOM.

Modelo de almacenamiento en medios: Estructura de datos usada en distintas capas para garantizar interoperabilidad a través de intercambio de medios.

Modelo de información DICOM (MID): Un diagrama de entidad-relación que es usado para modelar las relaciones entre las definiciones de objetos de información representando clases de objetos del mundo real definidas por el modelo de aplicación DICOM.

Objeto compuesto: un objeto normalizado. Ver también IOD normalizado

Perfiles de Aplicación: Es un conjunto de reglas del estándar DICOM que describen el contenido y formato para almacenamiento de imágenes médicas en un medio físico transportable. Es específico para cada modalidad de imagen y depende del tamaño de la matriz de la imagen misma y de las capacidades del sistema.

Perfiles de Seguridad: Es un conjunto de reglas del estándar DICOM que describen el contenido y formato para encriptar la información que se desea almacenar en medios físicos transportables.

Preámbulo de archivo DICOM: Esta información se encuentra encapsulada dentro de la información del grupo de datos (dataset).

Print: Servicio DICOM usado para mandar imágenes a una impresora DICOM.

P-Valor: Un valor de dispositivo independiente definido en un espacio lineal de niveles de gris. La salida de la presentación LUT DICOM son los P-Valores, es el valor del píxel después que las transformaciones definidas de niveles de gris se han aplicado. Los P-Valores se introducen a un sistema de despliegue estandarizado.

Query/Retrieve: Servicio que permite a una entidad de aplicación realizar búsquedas de imágenes en un PACS y recuperarlas

Rango de Luminescencia: El espacio de valores de luminiscencia de un sistema de despliegue desde una mínima hasta una máxima luminiscencia.

Servicios DICOM: Conjunto de servicios DICOM, la mayoría de los cuales implica transmisión de datos sobre la red y el formato de archivo.

Servicios de almacenamiento en medios: Servicios que definen un conjunto de operaciones con medios para facilitar el almacenamiento y recuperación de información de instancias SOP DICOM en un medio físico.

Service Object Pair (SOP): Son las clases que definen los servicios DICOM, sus especificaciones se encuentran en las partes 3 y 4 del estándar.

Sistema Operativo: Es un conjunto de aplicaciones que permite la administración de los recursos físicos de una computadora.

Store (Dicom Store): Servicio DICOM usado para mandar imágenes u otros objetos persistentes (informes estructurados, etc.) a un PACS o a una estación de trabajo

Storage Commitment: Servicio DICOM usado para confirmar que una imagen ha sido almacenada permanentemente por un dispositivo. El usuario de la clase de servicio (modalidad, estación de trabajo, etc.) utiliza la confirmación de la clase de servicio proveedor (estación de almacenamiento) para asegurarse de que puede borrar la imagen localmente.

UID de clase SOP: Identificador único que utilizan las SOP del estándar DICOM

Usuario de grupo de archivos: Entidad de aplicación que accede a uno o más archivos dentro del grupo de archivos.

Worklist: Servicio DICOM que permite a una entidad de aplicación leer la "Lista de trabajo" y obtener detalles.

Bibliografía

- **[Allen, 1992].** Allen, F. L.. Exploting database technology in the medical arena. *IEEE Engineering in Medicine and Biology* , (MARZO de 1992) PG. 42-49.
- **[Amber, 2005].** Amber, S. W.. *The Elemenst of UML 2.0 Syle.* Recuperado el 13 de Junio de 2007, de Cambridge University Press: (2005) http://www.ambysoft.com/books/elementsUMLStyle.html
- **[Berlios, 1997].** Berlios. *CD Record.* Recuperado el 13 de Junio de 2007, de The open source mediator: (s.f.).http://cdrecord.berlios.de/old/private/cdrecord.html
- **[Becker, 2007].** Becker, Yvonne. PROZ - The traslators workspace. (20 de Agosto de 2007). Recuperado el 20 de Agosto de 2007, de http://www.proz.com/kudoz/1001906
- **[Bosh, 2004].** Bosh, Mela. Modelo de objetos para la representación y rastreabilidad de documentos digitales. *V Congreso del capítulo español de ISKO, (International Society for Knowledge Organization) y IV Coloquio internacional de ciencias de la documentación.* (Marzo de 2004).
- **[Chavez, 2001].** Chavez, A. N. *Codificación de los objetos de información del estándar DICOM.* México D. F.: Universidad Autónoma Metropolitana, Iztapalapa. (2001).
- **[Copilen, 1994].** Copilen, J.. Paterns/Software design common question and answers in proceedings of object oriented. *New York SIGSPublications, Expo New York* , (JUNIO de 1994), pg. 39-42.
- **[Cota, 1994].** Cota, A. *"Ingeniería de Software". Soluciones Avanzadas.*
- D´Alessandro, M. (1988). Computers in radiology: "The totally digital radiology department of the future". *Sigbio newsletter , VOL. 10* (NO. 4), (1994)., pg. 2-6.
- **[DICOM Int., 2007]** DICOM Introduction and free software. Recuperado el 12 de Octubre del 2007 de http://www.sph.sc.edu/comd/rorden/dicom.html
- **[Eclipse, 2007].** Eclipse Process Framework Project (EPF), *Eclipse.Org.* Recuperado el 16 de Octubre de 2007 de http://www.eclipse.org/epf/
- **[Extremeprograming.org, 2007].** Extremeprograming.org. *Extremeprograming.org.* Recuperado el 13 de Junio de 2007, de http://www.extremeprogramming.org/
- **[Hinojosa, 2007].** Hinojosa, M. A. *Gestiópolis.* Recuperado el 13 de Junio de 2007, de http://www.gestiopolis.com/recursos/documentos/fulldocs/ger/diaggantaleja.htm
- **[IBM, 2007].** International Business Machines Corporation. Wikipedia. Recuperado el 31 de mayo del 2007 de http://es.wikipedia.org/wiki/Computing_Scale_Corporation#Historia
- **[Jacobson, 1998].** Jacobson, I.. *"Applying UML in The Unified Process".* Obtenido de Presentación Rational Software: (1998). http://www.rational.com/uml
- **[Jacobson, 1999].** The Unified Software Development Process, Ivr Jacobson, Grady Booch, James Rumbaugh, Racional Software Corp. Addison-Wesley, prefacio xx y xxi. Año 1999.
- **[Jiménez, 2000].** Jiménez, A. J.. *Protocolo de servicio de intercambio de mensajes DICOM.* México D. F.: Universidad Autónoma Metropolitana, Iztapalapa. (2000)

- **[Kruchten, 2003].** Kruchten, P. *The rational unified process: an introduction* (THIRD EDITION ed.). (B. JACOBSON, Ed.) USA: Addison/Wesley. (2003).
- **[Lewis, 1994].** Lewis, G. *"What is software engineering?".* USA. (1994).
- **[Lucasian Labs, 2009]**. Lucasian Labs, Lucasian Development. Recuperado el 16 de Marzo de 2009, e http://www.lucasian.com/lucasian_development.htm
- **[Martínez, 1999].** Martínez, A. M. *Desarrollo de protocolo de capa superior del estándar DICOM utilizando metodología orientada a objetos.* México, D. F.: Universidad Nacional Autónoma de México. (1999).
- **[Martínez, 2007].** Martínez, M. A., & Cervantes, M. H. *Wiki UAMISoft.* Recuperado el 13 de Junio de 2007, de http://ixil.izt.uam.mx/~almm/uamisoft/doku.php
- **[Martínez, 2005].** Martínez, M. A., Nuñez, G. M., Jiménez, A. J., Muñoz de Cote, J. E., Chávez, A. N., Delgado, E. R., y otros. A Strategy for PACS Development using introductory team software process. (SPIE, Ed.) *Electrical Engineering Department, Universidad Autónoma Metropolitana - Iztapalapa* , (2005). *V.1* (5748-3), p.1 de 10.
- **[Muñoz de Cote, 2003].** Muñoz de Cote, E. J. *Implementación del protocolo de capa superior DICOM para el desarrollo de un PACS.* México D. F.: Universidad Autónoma Metropolitana Iztapalapa. (2003).
- **[Nebulon Pyy, 2007].** Nebulon Pyy, L. *I. T. Solutions that make a difference.* Recuperado el 12 de Junio de 2007, de http://www.nebulon.com/fdd/
- **[Nema, 2007].** Nema, S. *Digital imaging and communications in medicin (DICOM).* National electrical manufacturers association. (2007).
- **[Oktaba, 2003]** Oktaba, H. et Al., "Moprosoft: Modelo de Procesos para la Industria de Software. (Mayo 2003). Recuperado el 12 de Octubre del 2007 de www.lania.mx/biblioteca/manuales/
- **[Salinas, 2007].** Salinas, C. P. *Tutorial UML.* Obtenido de Universidad de Chile, facultad de ciencias físicas y matemáticas: http://www.dcc.uchile.cl/~psalinas/uml/ (13 de Junio de 2007).
- **[Proyect Birdy, 2007].** Project Birdy.com. Recuperado el 16 de Octubre de 2007. http://www.projectbirdy.com/?t=rational%20unified%20process&gclid=COLEodD9lI8CFRtzhgodMyQEHQ
- **[Software Engineering, 2007].** Software Engineering, I. *Camegie mellon.* Recuperado el 17 de Enero de 2007, de http://www.sei.cmu.edu/cmm/
- **[Spinec.org, 2007].** Spinec.org. (1 de Enero de 2007). *Spinec.org.* Recuperado el 13 de Junio de 2007, de http://www.spinec.org/?m=200604
- Sun, M. (s.f.). *Sun Hispanoamérica.* Recuperado el 13 de Junio de 2007, de http://mx.sun.com/
- **[UPV, 2006]**. Universidad Politécnica de Valencia. Recuperado el 12 de Octubre del 2007, de https://pid.dsic.upv.es
- **[UCM, 2007].** UCM.. *UCM.* Recuperado el 12 de Junio de 2007, de http://www.ucm.es/info/mageeq/glosario.htm
- **[Wikipedia.org – COTS, 2008].** Wikipedia.org. Commercial off-the-shelf. Recuperado el 17 de Marzo del 2008, de http://en.wikipedia.org/wiki/Commercial_off-the-shelf

- **[Wikipedia.org - DICOM, 2006].** Wikipedia.org. *DICOM.* Recuperado el 29 de Diciembre de 2006, de http://es.wikipedia.org/wiki/dicom
- **[Wikipedia.org – Patrón de diseño, 2007].** Wikipedia.org. *Patrones de diseño.* Recuperado el 17 de Marzo de 2008, de http://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o
- **[Wikipedia.org - EDT, 2007].** Wikipedia.org. *Estructura de descomposición del trabajo.* Recuperado el 23 de Noviembre de 2007, de http://es.wikipedia.org/wiki/Work_Breakdown_Structure
- **[Wikipedia.org - RUP, 2006].** Wikipedia.org. *Proceso unificado.* Recuperado el 13 de Junio de 2007, de http://es.wikipedia.org/wiki/Proceso_Unificado
- **[Wikipedia.org - UP, 2006].** Wikipedia.org. *Rational unified process.* Recuperado el 12 de Junio de 2007, de http://es.wikipedia.org/wiki/Rational_Unified_Process
- **[Wilson, 1991].** NASA-STD-2100-91, NASA Software Documentation Standard, NASA Headquarters Software Engineering Program. Recuperado el 12 de Octubre del 2007 de http://satc.gsfc.nasa.gov/support/STC_APR97/write/writert.htm