

**Universidad de las Ciencias Informáticas**  
**Facultad 7**



**Alas PACS Burner:**  
**Servidor de Quemado DICOM3.0 para Clínicas**  
**Imagenológicas**

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

**Autores:**

- Alex Miranda Guevara
- Leandro Taset Blanco

**Tutores:**

- Ing. Lázaro González Rodríguez
- Ing. Yanoksy Durañona Yero

Ciudad de la Habana, Junio 2008

“Año 50 de la Revolución”

## DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los 23 días del mes de Junio del año 2008.

---

Alex Miranda Guevara  
Autor

---

Leandro Taset Blanco  
Autor

---

Ing. Lázaro González Rodríguez  
Tutor

---

Ing. Yanoksy Durañona Yero  
Tutor

TUTOR: Ing. Lázaro González Rodríguez ([lgonzalezr@uci.cu](mailto:lgonzalezr@uci.cu))

Profesor graduado de Ingeniero en Ciencias Informáticas. Imparte la asignatura de Práctica Profesional en la facultad 7 como adiestrado. Es jefe del área temática de software médico imagenológico en el Grupo de Procesamiento de Imágenes y Señales (GPI).

TUTOR: Ing. Yanoksy Durañona Yero ([yduranona@uci.cu](mailto:yduranona@uci.cu))

Profesor graduado de Ingeniería en Ciencias Informáticas. Ha impartido las asignaturas Teleinformática I y Teleinformática 2 en el colectivo de profesores de la facultad 7. Actualmente se desempeña como profesor y Jefe de Sub-Proyecto dentro de GPI en la Universidad de las Ciencias Informáticas (UCI).

## DEDICATORIA

### **Alex:**

A la memoria de mi abuelo...

A mi máspreciado tesoro, mi madre...

A mi padre y hermanos...

A mi familia en general y a Ana, que ya es parte de ella.

### **Leandro:**

A mi madre...

A mi padre...

A mis hermanos...

A Mónica, mi novia...

Y a toda mi familia.

### RESUMEN

Con la modernización y creación de centros especializados para el diagnóstico médico a través de imágenes digitales, las técnicas de diagnóstico han revolucionado los modos de persistencia de las imágenes médicas, incrementándose el uso de medios ópticos para el almacenamiento de las mismas.

**Alas PACS Burner** es una solución de software diseñada con el objetivo de facilitar la organización y el almacenamiento de las imágenes médicas en CD y en DVD. El software se ha elaborado según lo estipulado en el estándar DICOM<sup>1</sup> 3.0 para el almacenamiento y transmisión de las imágenes médicas. Ha sido desarrollado usando C# como lenguaje de programación sobre el NET Framework<sup>2</sup> 2.0.

**Alas PACS Burner** hace posible la organización de las imágenes en paquetes, bajo las restricciones de diferentes métodos de empaquetado. Permitiendo la entrega de estudios a los pacientes y el proceso de archivamiento a largo plazo de las imágenes médicas. Además, presenta la posibilidad de ser utilizado, tanto independientemente, como formando parte de un PACS<sup>3</sup>. Con su aplicación en los hospitales se logra un aumento en la eficiencia y la calidad del servicio.

### PALABRAS CLAVE

DICOM, DICOMDIR, DICOM-Directory, empaquetador, File-Set Creator, quemador, radiología

---

<sup>1</sup> Digital Imaging and Communications in Medicine.

<sup>2</sup> Marco de trabajo. Conjunto de librerías para el desarrollo de aplicaciones de software.

<sup>3</sup> Picture Archiving and Communication System.

# Alas PACS Burner

---

INTRODUCCIÓN .....	1
CAPÍTULO I FUNDAMENTACIÓN TEÓRICA .....	5
I.1 Conceptos y estándares .....	5
I.2 Herramientas y tecnologías de desarrollo .....	14
I.3 Estado del Arte .....	18
CAPÍTULO II CARACTERÍSTICAS DEL SISTEMA .....	21
II.2 Problema científico y situación problemática .....	21
II.3 Objeto de automatización .....	23
II.4 Información que se maneja .....	23
II.5 Propuesta de sistema .....	24
II.6 Modelo de negocio .....	25
II.7 Diagramas de actividades de los Casos de Uso del Negocio .....	27
II.8 Especificación de los requisitos de software .....	29
II.9 Definición de los casos de uso .....	34
II.10 Casos de uso por ciclo .....	40
CAPÍTULO III ANÁLISIS Y DISEÑO DEL SISTEMA .....	44
III.1 Análisis y diseño del caso de uso: Crear DicomDir .....	44
III.2 Análisis y diseño del caso de uso: Empaquetar ficheros .....	46
III.3 Análisis y diseño del caso de uso: Quemar Disco .....	49
CAPÍTULO IV IMPLEMENTACIÓN DEL SISTEMA .....	52
IV.2 Componentes del Software .....	52
IV.3 Despliegue del Sistema .....	54
CONCLUSIONES .....	55
RECOMENDACIONES .....	56
BIBLIOGRAFÍA	58
ANEXOS	60
Anexo 1 Diagrama de requisitos: Organizar las imágenes .....	60
Anexo 2 Casos de Uso expandidos .....	61

# Alas PACS Burner

---

Anexo 3 Diagramas de Clases de Diseño .....	66
Anexo 4 Descripción de Clases de Diseño .....	69
GLOSARIO	96

## INTRODUCCIÓN

Los humanos en su andar por la historia siempre han caminado hacia un mayor desarrollo, tanto social como tecnológico. El siglo XX fue un escenario perfecto para el surgimiento de una nueva y revolucionaria ciencia: la informática. Con sus diferentes aplicaciones, revolucionó y revoluciona el desenvolvimiento de la sociedad. “Vivimos rodeados de datos, totalmente invadidos por la información. Desde la década de los 80, se ha producido una proliferación de artilugios electrónicos hasta entonces desconocidos”. (Sáez 2007)

“La infraestructura que mantiene al mundo es básicamente la de la información, el flujo de datos en movimientos” (Sáez 2007). Una esfera social que se ha beneficiado en gran medida con el desarrollo de la informática es la salud.

En la década del 70, debido a las investigaciones de Godfrey Hounsfields<sup>4</sup>, aparecen en el ámbito médico las tomografías axiales computarizadas, surgiendo así la radiografía digital. Posteriormente surgen otras modalidades de diagnóstico médico, tales como: las resonancias magnéticas, la medicina nuclear, etc. Debido a las ventajas que ofrecen las imágenes digitales, todas estas modalidades y otras que se incorporaron, se sustentaron en una base digital. Con la digitalización de gran parte de las modalidades de diagnóstico médico, se incrementó el uso de las computadoras, medios de almacenamiento y las redes informáticas en la medicina.

Muchas compañías dedicaron su producción sobre la línea de los equipos de diagnóstico digital, por lo que, tanto la calidad, como la cantidad de dispositivos aumentó. Surge así la necesidad de un estándar para la transmisión, estructuración y organización de la información entre los dispositivos de los diferentes fabricantes (específicamente de los relacionados con imágenes médicas). En el año 1983 la Asociación de Radiólogos Americanos (ACR<sup>5</sup>) y la Asociación Nacional de Empresas Eléctricas (NEMA<sup>6</sup>), ambas de Estados Unidos, forman en conjunto un comité para el desarrollo de un estándar. Resultado de la labor de este comité, ve la luz en el año 1985 el estándar **DICOM 1.0**.(National Electrical Manufacturers Association 2007)

---

<sup>4</sup> Electrónico británico que colaboró en el desarrollo de una de las más importantes invenciones médicas del siglo XX: la tomografía axial computarizada (TAC).

<sup>5</sup> American College of Radiology (ACR)

<sup>6</sup> National Electrical Manufacturers Association (NEMA)



## Alas PACS Burner

---

Para el año 1997, Cuba dio los primeros pasos para la informatización de algunos servicios, entre ellos la salud, pero estos fueron incipientes. Con el arribo del nuevo milenio surge una nueva forma de lucha en la sociedad cubana: la Batalla de Ideas. Una de las premisas de la Batalla de Ideas es lograr construir una “sociedad del conocimiento”. La informatización tenía que ser punta de lanza fundamental, por lo que bajo el calor de la Batalla de Ideas surge la Universidad de la Ciencias Informáticas (UCI). En los años siguientes el proceso de informatización se incrementó, beneficiándose al sector de la salud. Como parte de la informatización se adquirieron equipos médicos de última tecnología. Los equipos adquiridos, sobre todo los relacionados con las técnicas imagenológicas, presentan precios casi prohibitivos. Además presentan el inconveniente de que unas pocas empresas controlan el desarrollo de soluciones informáticas para su mejor aprovechamiento.

Como consecuencia de la modernización de clínicas imagenológicas y hospitales se presentan algunas dificultades en el aspecto del almacenamiento. Una de ellas es que estos equipos generan una gran cantidad de imágenes digitales. Por su importancia, éstas deben ser almacenadas en las instituciones un mínimo de cinco años. Lo que conlleva al agotamiento de la capacidad de almacenamiento en los servidores de imágenes.

Por otra parte, los pacientes prefieren llevarse sus estudios, lo que se ve limitado por el encarecimiento de los medios de impresión de esas imágenes, y surge la necesidad de hallar una alternativa para solucionar esta situación. En la práctica para paliar esta dificultad se quemaron discos ópticos (**CD/DVD**) en las **wizards**. Produciéndose con ello retrasos en los diagnósticos y una pérdida de tiempo de trabajo para el técnico encargado del equipo.

A lo anterior se puede agregar, que no todas las instituciones hospitalarias cuentan con redes informáticas óptimas para la transmisión de imágenes DICOM, o simplemente no existe ninguna. Todo ello, entorpece la colaboración imagenológica entre los hospitales.

Ante las problemáticas enunciadas, se plantea como problema científico: ¿Cómo facilitar el almacenamiento e intercambio de imágenes médicas utilizando medios externos de almacenamiento?

## Objeto de estudio y campo de acción

Con el presente trabajo se pretende abordar los procesos de organización y almacenamiento de imágenes médicas. El campo de acción se centra en la organización y almacenamiento de imágenes médicas en dispositivos externos para instituciones hospitalarias (específicamente en CD/DVD).

## Objetivos

- El **objetivo general** que se proponen los autores es desarrollar un software capaz de organizar y almacenar imágenes médicas en dispositivos externos (CD/DVD).

Además se proponen los siguientes objetivos específicos:

- Lograr que los discos cumplan con el estándar **DICOM 3.0**, para que sean compatibles con otros sistemas.
- Permitir que el sistema sea configurable, según las necesidades de la institución.
- Lograr un seguimiento controlado durante todo el proceso de organización y almacenamiento.
- Permitir interactuar con otros componentes de un PACS.

## Tareas desarrolladas para cumplir los objetivos

Para el cumplimiento de los objetivos propuestos se realizaron una serie de tareas, dentro de las cuales se encuentran:

- Analizar el estado del arte para aplicaciones de este tipo a nivel internacional y nacional.
- Aplicar los patrones de desarrollo de aplicaciones que se suelen usar en estas condiciones.
- Analizar el estándar **DICOM 3.0**.
- Analizar el **ISO 9660**.
- Analizar el **Universal Disk Format**.
- Implementar la aplicación en cuatro módulos:
  - Módulo de Obtención de Imágenes Médicas.
  - Módulo de Empaquetamiento.
  - Módulo de Almacenamiento.
  - Módulo de Presentación (Interfaz Gráfica de Usuario).

Este trabajo propone como solución a las dificultades expresadas realizar un software capaz de organizar y almacenar las imágenes en discos ópticos. Con la integración de la aplicación a los **PACS**

existentes en los hospitales se obtendría un grupo de beneficios. Uno de ellos es que representaría un ahorro considerable de tiempo en las consultas, el cual se podría destinar a la atención de más público en las instituciones de salud, ganando de esta forma en calidad y eficiencia. Además, se evitarían posibles salidas de funcionamiento o pérdida de información en los servidores. Por otro lado, tanto los pacientes como las instituciones podrían conservar sus estudios en dispositivos baratos y de calidad. Es válido hacer mención al ahorro que se le facilitaría la economía del país, por conceptos de importación y eficiencia laboral.

### **Estructuración del contenido**

El presente trabajo ha sido estructurado en capítulos, para separar la información en grupos conceptuales semejantes. A través de ellos se plasma todo lo necesario para la comprensión del software, quedando los mismos distribuidos de la siguiente forma:

En el Capítulo I se plasma lo relacionado con la fundamentación teórica en la que se sustenta este trabajo. En el mismo se hace un estudio de algunos de los sistemas similares que existen, tanto a nivel nacional como a nivel internacional. Además, se exponen conceptos relacionados con los estándares que rigen esta área de la informática y se justifica la elección de las tecnologías y las herramientas seleccionadas para la realización del trabajo.

El Capítulo II refleja con todo detalle el problema científico, la situación problemática, el objeto a automatizar y la información que se maneja. Se expone también la propuesta del sistema. Para ello se parte desde el análisis de los procesos de negocio, se pasa por la exposición de los requisitos y se llega a la definición de los casos de uso del sistema. En este capítulo se define la distribución final de los casos de uso para su realización en los distintos ciclos de desarrollo.

Por su parte el Capítulo III aborda el tema del análisis y el diseño del software en cuestión, esto basado en el uso de la metodología de desarrollo **RUP**. En este capítulo se muestran los diagramas de realización, de clases de análisis y los de clases de diseño, que ilustran el comportamiento estático del sistema.

En el Capítulo IV queda registrado lo relacionado con la implementación del sistema, para ello se incluyen los diagramas de componentes y de despliegue. Con lo contenido en este capítulo el lector puede ver la distribución física que el sistema presentará en su instalación final.

## CAPÍTULO I FUNDAMENTACIÓN TEÓRICA

### Introducción

En este capítulo se caracterizan los **PACS**, para poder ubicar el software construido dentro de los mismos. Además se describe el estándar **DICOM** porque en él descansan las restricciones fundamentales del software elaborado. Dentro de la explicación de los temas ligados al estándar se enumeran algunos conceptos fundamentales para entender el basamento teórico de este trabajo.

En un punto más avanzado del capítulo se hace referencia a las herramientas y tecnologías utilizadas para la realización del producto. Por constituir estas el medio necesario para dar cumplimiento al objetivo principal del trabajo. Para concluir el capítulo se realiza un análisis crítico de algunos sistemas similares que se utilizan en el ámbito internacional y nacional. Así como un pequeño apunte sobre los pasos dados en la universidad<sup>7</sup> en esta materia.

### I.1 Conceptos y estándares

Para la realización de este trabajo los autores se rigieron por una serie de conceptos y estándares. En las secciones siguientes se describe los principales, aquellos que se hacen imprescindibles para el entendimiento del mismo.

#### I.1.1 Sistema de Almacenamiento y Comunicación de Imágenes (PACS)

En la actualidad en los hospitales se extiende el uso de las imágenes, los archivos y los diagnósticos digitalizados. Esta información ha revolucionado los procesos hospitalarios, los que han tenido que incluir nuevos procesos y flujos de información. Por lo general estos nuevos procesos y flujos están respaldados por un PACS.

Pero, ¿qué es un PACS? Un PACS es un sistema para la gestión, transmisión y exhibición de imágenes médicas (Huang 1999). Otra definición es la de computadoras y redes dedicadas al almacenamiento,

---

<sup>7</sup> Universidad de las Ciencias Informáticas.

obtención, distribución y visualización de imágenes (Stryker 2007). Estas definiciones implican el hecho de que estos sistemas son caros, complejos y están dominados por unas pocas compañías.

Los PACS están compuestos por varios componentes según (Rincón y Rodríguez s.f.), estos son:

- **Sistemas de Adquisición:** Son los equipos encargados de la adquisición de las imágenes. Estos se separan en dos modalidades fundamentales. Por un lado están los que presentan una interfaz de acceso digital tales como TC, RM y por el otro los que precisan de una digitalización posterior (Ultrasonidos, placas, etc.).
- **Redes de Comunicación:** Redes LAN<sup>8</sup> o WAN<sup>9</sup> y los dispositivos que permiten su funcionamiento. Las redes de comunicación son fundamentales para el funcionamiento de un PACS. Por el gran flujo de datos que involucran deben de poseer una red de alta velocidad.
- **Manejadores de Bases de Datos:** Para el buen funcionamiento de un PACS el diseño e implantación de un sistema de bases de datos es fundamental. Se debe tener en cuenta una estrategia para el almacenamiento de la información. En las horas siguientes a la adquisición de una imagen, esta se consultará con más frecuencia. Luego la probabilidad de consulta disminuirá paulatinamente. Es por esto que el almacenamiento a corto plazo debe hacerse en sistemas locales.
- **Estaciones de Trabajo:** Las estaciones de trabajo o de visualización son elementos que presentan la información visual a los médicos. Por lo general son computadoras equipadas con múltiples monitores de alta resolución y usadas solamente para este fin. Aquí se incluyen las **wizards** o estaciones de apoyo que acompañen a un equipo de adquisición.
- **Sistemas de Almacenamiento:** En general las imágenes recientemente adquiridas se consultan con mucha frecuencia en los minutos siguientes a su adquisición y su frecuencia de consulta disminuye con el tiempo. El almacenamiento a corto plazo tiene las siguientes características: decenas de GB<sup>10</sup>, transferencia de alrededor de 50 imágenes por minuto, 1-15 días de almacenamiento. El almacenamiento a largo plazo debe cumplir con los siguientes requisitos: capacidad de varios Terabytes<sup>11</sup>, capacidad de almacenar la información durante dos años o más, empleo de cinta e imágenes comprimidas para almacenamiento a plazos mayores.

---

<sup>8</sup> Local Area Network: Red de Área Local.

<sup>9</sup> Wide Area Network: Red de Comunicación Extendida.

<sup>10</sup> Giga Byte: Mil millones de bytes. Unidad de medida de la capacidad de almacenamiento de una computadora.

<sup>11</sup> Aproximadamente equivalente a mil Giga Bytes.

- **Almacenamiento en línea:** Contienen la información de los estudios recién generados o aquellos que el radiólogo necesita para realizar un diagnóstico o una revisión. Su vida dentro del dispositivo es del orden de 15 días, luego de este período se incorporan al archivo histórico.
- **Almacenamiento histórico:** No posibilitan un acceso inmediato, sino que necesitan un período de algunas horas para recuperar el estudio. Esta tarea puede requerir intervención humana o ser un proceso automático. Un estudio dentro del archivo histórico tiene una duración del orden de miles de días.

### I.1.2 Estándar DICOM

En las clínicas imagenológicas, es muy común encontrarse con equipos de varios fabricantes para las diferentes modalidades de imágenes que se generan. Tratar de integrar todos ellos en un sistema que los manipule es prácticamente imposible. En base a esto surgió la necesidad de estandarizar el manejo y transmisión de imágenes médicas digitales. Este trabajo se inició en 1983, con la integración de un comité formado por el "American College of Radiology" (ACR), representando a la comunidad de radiólogos y la "National Electrical Manufacturers Association" (NEMA), representando a la industria en el área de radiología, de acuerdo a los procedimientos establecidos por NEMA. Entre los objetivos de este comité se encontraban:

1. Promover la comunicación de imágenes digitales independientemente del fabricante que las produjo.
2. Ofrecer mayor flexibilidad a los sistemas de almacenamiento y comunicación de imágenes.
3. Facilitar la creación y consulta a sistemas de diagnóstico por diferentes dispositivos y en diversos lugares locales o remotos.

Los primeros resultados en los trabajos de estandarización fueron publicados en 1985, ACR-NEMA Versión 1.0. En la misma se detectaron varios errores y el comité encargado (ACR/NEMA) autorizó a los

grupos de trabajo involucrados a la realización de dos revisiones en octubre de 1986 y en enero de 1988, que produjeron una segunda versión, ACR-NEMA Versión 2.0, en 1988.

En el tiempo que se dio a conocer la segunda versión, surgió la demanda de una interfaz entre dispositivos involucrados en la generación y manejo de las imágenes y las redes de cómputo, sin embargo, el estándar no ofrecía ningún soporte de comunicación en red. La respuesta a estas demandas implicaba grandes cambios a lo ya establecido, considerando como restricción principal el mantener la compatibilidad con las versiones anteriores, lo cual fue un gran reto para los grupos de trabajo.

De esta forma, a partir de 1988 se comenzó a trabajar en una tercera versión, en donde el proceso de diseño sufrió un cambio radical adoptando modelos para simular el mundo real, modelos de capas o pila para comunicación entre sistemas heterogéneos utilizando protocolos de comunicación en red y el modelo de cómputo Cliente/Servidor para establecer asociaciones entre dispositivos compatibles, a través de envío de mensajes.

Después de tres años de esfuerzo, se dio a conocer la versión ACR/NEMA DICOM llamada también DICOM 3.0, en la que participaron varias instituciones de la comunidad internacional como JIRA<sup>12</sup> y CEN<sup>13</sup>. Esta versión es considerada como un estándar completo, compatible con las versiones anteriores.

El estándar DICOM 3.0 cuenta con varias partes, las cuales a pesar de estar relacionadas son documentos independientes. Al estándar se le han agregado partes según las necesidades que han surgido. En el pasado año presentó 18 partes de las cuales dos han sido retiradas.(NEMA 2007)

El contenido abordado en este trabajo tiene sus principales fundamentos en tres partes del estándar. Estas partes son:

- **PS 3.10: Media Storage and File Format for Data Interchange:** Especifica un modelo general para el almacenamiento de imágenes médicas en dispositivos removibles. El propósito de esta parte es proveer un marco de trabajo que permita el intercambio de varios tipos de imágenes médicas en dispositivos de almacenamiento físico.
- **PS 3.11: Media Storage Application Profiles:** Especifica subconjuntos del estándar para los cuales una aplicación puede declarar conformidad. Tal declaración de conformidad se aplica al

---

<sup>12</sup> Japanese Industry Radiology Apparatus

<sup>13</sup> Comité Européen de Normalisation

intercambio interoperable de imágenes médicas en medios de almacenamiento para usos clínicos específicos.

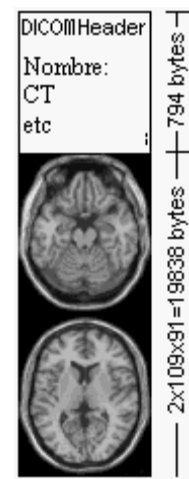
- **PS 3.12: Storage Functions and Media Formats for Data Interchange:** Esta parte del estándar facilita el intercambio de información entre aplicaciones en un entorno médico. Para ello describe la relación entre el modelo de almacenamiento y cada uno de los dispositivos físicos permitidos por el estándar.

### I.1.3 DICOM File

Una de las interrogantes que se planteó el comité para la creación del estándar fue la necesidad de que todos los ficheros que contenían las imágenes cumplieran con una estructura similar. Además, la información inherente a las distintas fuentes de las imágenes se debería poder obtener de forma implícita. En los ficheros normales de imágenes esto no es posible.

El fichero DICOM (DICOM file) contiene dos partes: el encabezado y los datos. En la parte del encabezado se encuentra la información acerca del paciente, la institución hospitalaria en la que se produjo la imagen, el tipo de estudio, las dimensiones de la imagen, etc. Mientras en la parte de los datos se encuentra la información de las imágenes en las tres dimensiones.

La **Figura 1** muestra un ejemplo hipotético de un fichero DICOM. En el mismo se puede observar las dos partes del fichero, el encabezado con 794 bytes y los datos con 19838 bytes. En la parte superior se ve la información de interés. Esta información en las aplicaciones que cumplen el rol de **FSC**<sup>14</sup> o **FSU**<sup>15</sup> es muy importante ya que acceden a ella y la modifican o la consultan. Para la realización del software de este trabajo se selecciona de esta información la relevante y se pone en un fichero denominado **DicomDir**. En la misma *figura 1* se puede observar en la parte inferior una representación de los datos, en este caso son imágenes. En un fichero real estas imágenes están codificadas según las dimensiones que posean.



**Figura 1** El fichero DICOM está dividido en dos partes.

<sup>14</sup> File Set Creator, son aplicaciones con la capacidad de organizar las imágenes y crear un DICOMDirectory.

<sup>15</sup> File Set Updater modifica algún fichero del File Set y actualiza el DICOMDirectory.



Los ficheros DICOM contienen en el encabezado un campo ID que los identifica en el contexto en que se encuentran.

#### I.1.4 File-Set

Es una colección de archivos DICOM (y posiblemente no DICOM) que comparten un espacio de nombres donde los identificadores de los archivos (**File ID**) son únicos. En el mismo se debe de encontrar un fichero cuyo identificador es DICOMDIR.

Como norma para la identificación del File-Set se le asignará un identificador, este identificador por lo general será una etiqueta que posea un significado y pueda leerse. Además debe de cumplir con una independencia con respecto al contenido del File-Set. De esta forma, si se le añade o elimina un fichero al File-Set el identificador de este no debe variar.

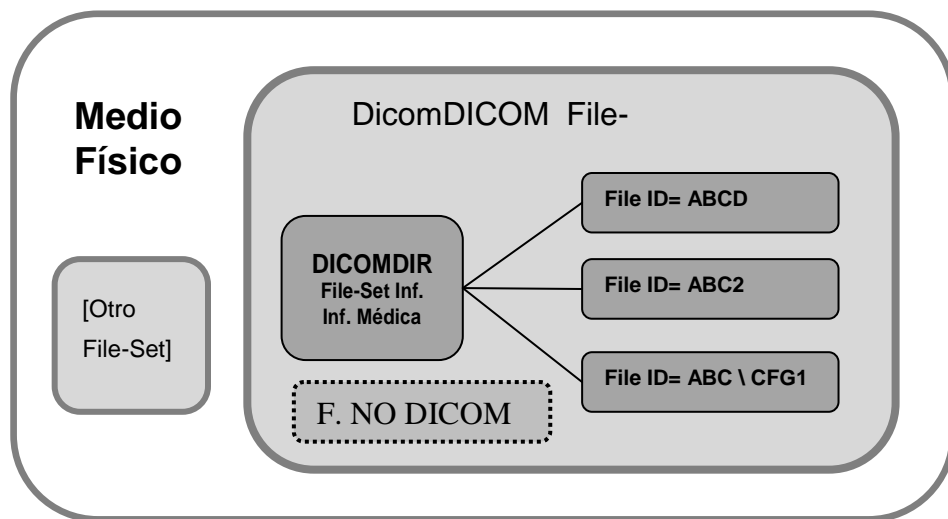


Figura 2 Estructura del FileSet.

En la **Figura 2** se ilustra la estructura de un File-Set. En la estructura representada se puede ver como un File-Set contiene varios ficheros. Uno de estos es el DicomDir, el cual guarda la información correspondiente al File-Set y la información inherente a las imágenes médicas. Obsérvese como aquellos que son DICOM-Compatibles se referencian en el

fichero denominado DicomDir, mientras que los no compatibles, no se referencian.

Además, observar que en un mismo medio físico (dígase CD, DVD, etc.), siempre que esté estandarizado, se puede almacenar más de un File-Set.

### I.1.5 Fichero DicomDir

El DicomDir es un fichero DICOM especial que contiene representada la organización de la información en el File-Set. Su organización está basada en una jerarquía de entidades. Su contenido parte de una entidad padre, la cual puede contener una o varias entidades anidadas. En un directorio cada entidad posee una referencia al fichero DICOM que la contiene, esto es posible a través de las componentes del **FileID**. (NEMA 2007, 966)

Cabe señalar que el identificador de este fichero es único y es precisamente DICOMDIR.

En la **Figura 3** se presenta una representación hipotética de la relación entre los ficheros DICOM y el “Directorio DICOM” (en el contexto de este trabajo referenciado como DicomDir), así como la organización de las distintas entidades del directorio dentro del mismo. De forma general los archivos DICOM o Imágenes DICOM que están en el File-Set son representados en el gráfico por los fotogramas en la parte inferior y el DicomDir por el recuadro de la parte superior.

Este ejemplo muestra cómo la información de los ficheros almacenados y referenciados por el DicomDir, es agrupada en entidades del directorio. Fijarse como la información de los estudios (Estudio 1 y Estudio 2) forman parte de la entidad relativa al Paciente A.

A pesar de que la estructura física del DicomDir es simple, ya que la información se almacena en forma secuencial, las entidades están referenciadas de una manera muy robusta. En la gráfica se usan las saetas para representar las referencias entre entidades. Como es el caso del Estudio 1 que referencia a la entidad de directorio que contiene la información de las series de este estudio (**Entidades Estudio 1** en la **Figura 3**). Por otro lado las referencias de las imágenes (**Entidades Serie 1** en la **Figura 3**), representadas por líneas truncadas, referencian a los objetos dentro de los ficheros DICOM. Las referencias internas (saetas) en la práctica se realizan mediante distintos métodos, la mayoría atendiendo al tamaño de las entidades. Mientras que las referencias hacia los ficheros se realiza mediante los identificadores de los ficheros (**FileID**).

Otro punto a resaltar es que algunas entidades no presentan referencia a los ficheros (**Entidades Series 1** en la **Figura 3**), mientras que otras no las presentan hacia las entidades de los niveles inferiores como el caso de las imágenes.

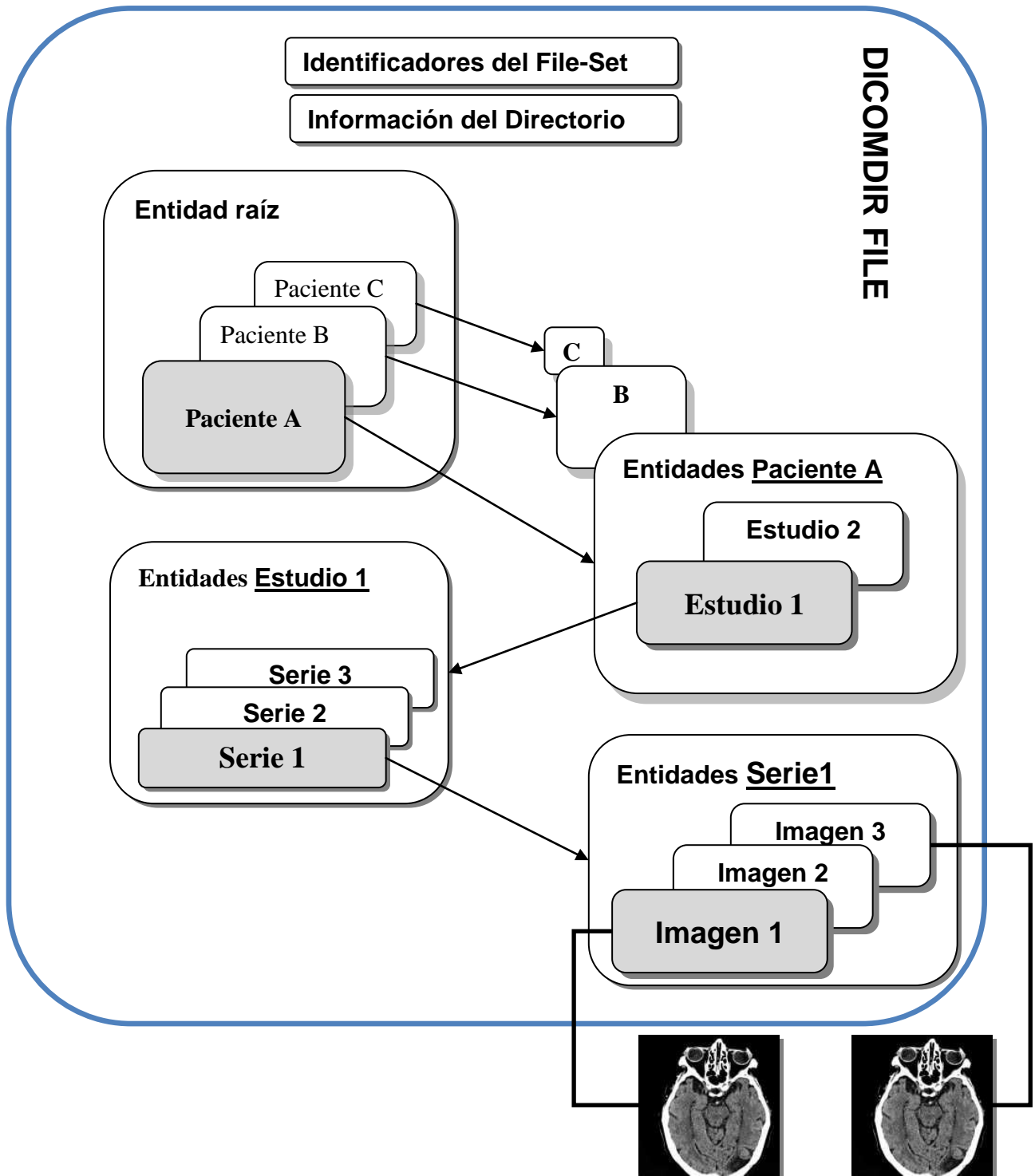


Figura 3 Ejemplo de la organización del Directorio (DicomDir) y su contenido.

### I.1.6 File-Set Reader

Es un rol que se establece para las aplicaciones que deben de ser capaces de leer un File-Set que les ha sido transferido. Las aplicaciones que se consideren File-Set Reader deben de ser capaces de usar las sintaxis de transferencia especificadas. (NEMA 2007, 20)

En el caso del software propuesto se deben leer archivos desde un File-Set existente. Para ello se apoya en el fichero DicomDir.

### I.1.7 File Set Creator

Una aplicación cuyas entidades actúan como File-Set Creator es capaz de generar un File-Set. La aplicación debe ser capaz de crear un fichero DicomDir con los principales datos de los ficheros que están en el File-Set. Este tipo de aplicaciones debe permitir, si un estudio no cabe en un CD/DVD, separarlo y colocarlo en otro. (NEMA 2007, 21)

### I.1.8 Estándar de quemado ISO 9660

El estándar **ISO 9660** es una norma publicada inicialmente en 1986 por la **ISO**<sup>16</sup>, que especifica el formato para el almacenamiento de archivos en los soportes de tipo disco compacto. El estándar **ISO 9660** define un sistema de archivos para CD-ROM. Su propósito es que tales medios sean legibles por diferentes sistemas operativos, de diferentes proveedores y en diferentes plataformas, por ejemplo, Microsoft Windows, Mac OS y UNIX.

El estándar en cuestión especifica la estructura de los ficheros y volúmenes en los discos compactos usados por los usuarios para el intercambio de información. Entre las cuestiones que regula se encuentran: la información y organización de los volúmenes<sup>17</sup> de información, la forma de almacenar los flujos de información, los atributos y la organización de los ficheros, entre otros. (ECMA 1987)

Para el almacenamiento de imágenes médicas se define como el estándar de quemado el ISO 9660. Algunas de las aplicaciones son: el identificador del **File-Set** se ubica como etiqueta del CD, la

---

<sup>16</sup> International Standards Organization. Organización que desarrolla estándares internacionales.

<sup>17</sup> Se refiere al disco compacto.

profundidad de los subdirectorios no debe de exceder de 8 niveles, los nombres de los archivos por su parte contendrán hasta 8 caracteres alfanuméricos. (NEMA 2007, 22-24)

### **I.1.9 Estándar de quemado UDF**

Universal Disk Format (UDF) es un sistema de archivos como el estándar ISO 9660 propiedad de Adaptec que utiliza las grabadoras de CD/DVD como un dispositivo de almacenamiento lógico. Este formato permite leer, escribir o modificar los archivos contenidos en discos, CD/DVD y re-escribibles (RW), del mismo modo que se hace en el disco duro, memorias USB o diskettes. Utiliza la tecnología de grabación por paquetes (Packet Printing) soportado por grabadoras CD-RW, DVD-RAM/RW.(ECMA 1998)

Hasta el momento se han liberado varias revisiones de este estándar. Para el almacenamiento en media de imágenes médicas, la creación de DVD se podrá realizar en cualquiera de las revisiones siguientes: 1.02, 1.5, 2.0, 2.01.

## **I.2 Herramientas y tecnologías de desarrollo**

La creación de todo producto, sea tangible o intangible, requiere de un conocimiento adquirido y aplicado. La aplicación de un conocimiento está sujeta a un medio, en este caso las herramientas. La tecnología es la aplicación de los avances de la ciencia y la técnica, es por ello que cumple un papel importante en la confección del software.

En los siguientes tópicos se hará referencia a las principales tecnologías, lenguajes y herramientas que se emplearon para la confección del software y su documentación.

### **I.2.1 Visual C# .Net 2.0**

C# es un lenguaje de programación de uso general diseñado por Microsoft para la plataforma .Net. Presenta seguridad de tipos y es orientado a objetos, por lo que es útil a la hora de crear aplicaciones robustas. Visual C# ofrece a los desarrolladores herramientas eficaces centradas en el código, en unión

de compatibilidad de lenguajes para crear aplicaciones tanto web como clientes completas y conectadas en .NET Framework.

El lenguaje C# toma la mejores características de varios lenguajes como Visual Basic, Java, C++ y otros. Esto lo hace robusto, pero sin influir complejidad del mismo sino todo lo contrario.

Otra de las ventajas que presenta es su capacidad para la interacción con código no administrado. Esta interoperabilidad se realiza de forma sencilla abstrayéndose de las técnicas complejas.

### **1.2.2 Microsoft Visual Studio 2008**

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE<sup>18</sup>) para sistemas Windows. Soporta varios lenguajes de programación, entre ellos C#.

Visual Studio 2008 permite incorporar características del nuevo Windows Presentation Foundation sin dificultad tanto en los formularios de Windows existentes como en los nuevos. Ahora es posible actualizar el estilo visual de las aplicaciones al de Windows Vista debido a las mejoras en Microsoft Foundation Class Library (MFC).

Este entorno mejora la interoperabilidad entre código nativo y código manejado por .NET. Esta integración más profunda simplifica el trabajo de diseño y codificación.

Otra de las características del mismo, es la capacidad de crear soluciones adaptadas para funcionar con las diferentes versiones de .Net Framework (2.0, 3.0 y 3.5).

### **1.2.3 MyDicom.NET SDK**

MyDicom.Net es un kit de desarrollo de software (SDK por sus siglas en ingles<sup>19</sup>), el mismo implementa el estándar DICOM 3.0. El uso de este SDK es bastante simple y permite la creación de productos médicos robustos.

Con el uso de MyDicom.Net se puede acceder a los ficheros DICOM-compatibles de forma sencilla y rentable en cuanto a rendimiento. Lo cual facilita el trabajo de extracción de la información. Además de implementar clases que le dan soporte a la creación de directorios DICOM (DICOMDirectory).

---

<sup>18</sup> Integrated Development Environment.

<sup>19</sup> Software Development Kit.

Entre las características que sobresalen de MyDicom.Net es que está implementado en C#, que es el lenguaje seleccionado para la confección del trabajo. Esto en adición a las prestaciones que facilita lo pone en ventaja sobre otros kits de desarrollo.

La migración del software a plataforma Unix es uno de los puntos que se le ha agregado a este kit. Esto se hace posible por el uso de la alternativa .Net MONO<sup>20</sup>.

#### **1.2.4 Lenguaje de modelado UML**

Un lenguaje proporciona un vocabulario y las reglas para combinar palabras de ese vocabulario con el objetivo de posibilitar la comunicación (Booch, Rumbaugh and Jacobson 1999). Un lenguaje de modelado es aquel cuyo vocabulario y reglas se centran en la representación conceptual y física de un sistema. Por lo que se podría concluir que el **Unified Modeling Lenguaje (UML)** por sus siglas en inglés) es un lenguaje estándar para los planos del software.

UML puede utilizarse para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad líneas de códigos. El mismo es apropiado para modelar desde sistemas de información en empresas hasta aplicaciones basadas en la Web, e incluso para sistemas en tiempo real muy exigentes(Flower 1999). Es un lenguaje que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas, todo esto logrado por los años de acumulación de experiencias en el desarrollo de software.

Cabe señalar que el UML es solamente un lenguaje y por tanto una parte de un método de desarrollo de software(Larman 1999). Esta independencia lo hace factible a usarlo con diferentes metodologías, para el desarrollo de este trabajo se aprovecha esta propiedad para juntarlo al Proceso Unificado de Rational.

#### **1.2.5 Enterprise Architect 7.0**

Según (Kendall 1997) la ingeniería de sistemas asistida por ordenadores es la aplicación de la tecnología informática a las actividades, las técnicas y las metodologías propias de desarrollo, su objetivo es acelerar

---

<sup>20</sup> Alternativa de .Net en software libre.

el proceso para el que han sido diseñadas. En el caso de los CASE, su misión es automatizar u apoyar una o varias fases del ciclo de vida del desarrollo del sistema.

El Enterprise Architect 7.0 es una herramienta de tipo de Ingeniería de Software Asistida por Ordenador más conocidas por herramientas CASE<sup>21</sup>. Esta herramienta ayuda en todos los aspectos del ciclo de vida de desarrollo del software. Se le usa en tareas como: el proceso de realizar un diseño del proyecto, en la implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

Con el uso adecuado del mismo se puede mejorar la productividad en el desarrollo y mantenimiento del software. Además de aumentar la calidad del software sin que esto conlleve a un aumento en el tiempo o coste de desarrollo del software. En un proyecto algo importante es el tiempo, de aquí se deriva el uso de este CASE ya que automatiza partes del desarrollo de software que consumen un tiempo prolongado: la generación automática de la documentación y parte del código son dos de las principales características. La herramienta en cuestión facilita y es compatible con la aplicación del Proceso Unificado de Desarrollo de Software. Esta metodología, que ha sido seleccionada para guiar el desarrollo del producto, cuenta con varias fases de desarrollo. El Enterprise Architect soporta todas las fases y personaliza los artefactos para cada una de ellas.

## **I.2.6 RUP**

El Proceso Unificado de Rational (RUP), es la metodología de desarrollo escogida para la modelación del software en cuestión. Esta elección está fundamentada en una serie de primacías que presenta RUP, sobre todo en lo correspondiente a la organización de la información. Este proceso en unión al Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizadas para la realización de proyectos de software.

El RUP no es un sistema con pasos firmemente establecidos, sino que es un conjunto de metodologías adaptables al contexto y necesidades de cada organización. Esta es una de las propiedades que el equipo de desarrollo pudo explotar. El proceso en cuestión presenta como principales características las siguientes:

---

<sup>21</sup> Computer Aided Software Engineering.



- **Iterativo e incremental:** A medida que avanza el proceso de desarrollo se producen versiones incrementales, las cuales se acercan cada vez más al producto terminado.
- **Guiado por los casos de uso:** Los casos de uso son los que indican cómo debe actuar el sistema con el usuario final o con otro sistema para conseguir su objetivo.
- **Centrado en la arquitectura:** Los modelos son proyecciones del análisis y del diseño, lo cual constituye la arquitectura del producto a desarrollar.

### I.3 Estado del Arte

En el mercado en la actualidad se encuentran gran cantidad de servidores de quemado y componentes de PACS que realizan la función de quemado. Por lo general los servidores de quemado son robots con un software agregado, lo cual aumenta considerablemente su precio.

#### I.3.1 Ámbito Internacional

##### **MDS 200 Medical Disc Systems**

Solución de hardware con software incluido de la compañía Hewlett Packard. Soporta el estándar DICOM 3.0 para el intercambio de imágenes en una red. Incorpora un visor DICOM en los discos para recrear las experiencias de un PACS. Imprime etiquetas personalizadas en los discos automáticamente usando campos de los metadatos de las imágenes.

##### **Sorna Corporation**

La oferta de productos de Sorna incluye la familia eXpedo de sistemas de grabación de CD/DVD. Estos productos son robots de quemado que permiten el etiquetado de los discos. Las soluciones son compatibles con el estándar DICOM 3.0.

### **DigiNet DICOM Server**

Servidor de imágenes DICOM, compatible con el estándar, que permite el almacenamiento de las imágenes en CD o DVD.

#### **I.3.2           Ámbito Nacional**

En la comunicación e intercambio de imágenes médicas en la nación cubana con soluciones propias ya venían alboreando desde finales del siglo pasado. Esto se puede corroborar con las soluciones presentadas en los distintos foros de Ciencia y Técnica Nacionales.(Mesias 1998) Si bien al principio no se contemplaba el almacenamiento como algo primordial, con el avance de las tecnologías imperantes en el país, se comenzó a prestar bastante atención a esta área.

Un software pionero en el área de imágenes y la radiología fue el IMAGIS. Desarrollado en el Centro de Biofísica de Santiago De Cuba, con el objetivo de aplicarse a la telemedicina y a la transmisión de imágenes médicas multimodales en el Sistema Nacional de Salud.

El IMAGIS cuenta con tres módulos: de transmisión, de visualización y de almacenamiento. El módulo de almacenamiento define las acciones encargadas para realizar almacenamiento de distintas formas. Permite almacenamiento en CD.

En las primeras versiones el almacenamiento en CD se realizaba exportando las imágenes médicas y los ficheros del visor hacia una carpeta, luego el usuario tenía que encargarse de quemar el contenido de esta carpeta. Solamente se gestionaba una carpeta. En versiones posteriores ya se podían organizar las imágenes en varios volúmenes, que se guardaban en carpetas previamente seleccionadas. Si bien ya se permitía una determinada pluralidad de volúmenes, el almacenamiento y la organización se debe de seguir realizando de forma manual. Las versiones actuales queman en discos pero la organización sigue siendo manual.

#### **I.3.3           Ámbito Universitario**

En la Universidad de las Ciencias Informáticas se han dado los primeros pasos en el quemado en discos de imágenes médicas. En el producto de **Alas PACS Viewer**, que es un visor de imágenes DICOM, posee una funcionalidad de exportar imágenes médicas. Esto sólo se realiza para Discos Compactos, y con el

estudio que esté activo en el visor. Otro producto que posee una funcionalidad similar es la Bandeja de Casos, del propio Alas PACS.

### **Conclusiones**

En este capítulo se caracterizaron los **PACS** y se describió el estándar **DICOM**.

Se explicó la importancia del DICOMDIR en las aplicaciones que asumen los roles de FSC y FSR. En un punto más avanzado del capítulo se hizo referencia a las herramientas y tecnologías utilizadas para la realización del producto. Concluyendo el capítulo se realizó un análisis crítico de algunos sistemas similares que se utilizan en el ámbito internacional y nacional, tales como: MDS 200 Medical Disc Systems, las soluciones de Sorna Corporation y DigiNet DICOM Server. Y se apuntó sobre los pasos dados en la UCI en esta materia.

## CAPÍTULO II CARACTERÍSTICAS DEL SISTEMA

### Introducción

En este capítulo se presenta formalmente el problema científico, la situación problemática, el objeto de automatización, punto en el cual se describen los procesos de negocio que serán automatizados, así como una descripción de los documentos que se procesan y la información que se manipula. También se hace una descripción general del sistema que propone este trabajo de diploma y una comparación con algunos de los sistemas existentes. Además se expone el modelo de negocio, la especificación de requerimientos funcionales y no funcionales del sistema y finalmente se muestran los diagramas de casos de uso.

### II.2 Problema científico y situación problemática

Las instituciones hospitalarias cubanas tienen buena reputación en cuanto a la calidad de sus servicios, siendo esto último uno de sus objetivos fundamentales. Como parte del aseguramiento de esta calidad el país prioriza la importación de tecnologías y técnicas de punta. Una de las áreas que en los últimos años se ha beneficiado con esta política es la del diagnóstico a través de imágenes.

Las técnicas imagenológicas se encuentran presentes en casi todas las áreas de la medicina, de ahí su importancia. Dichas técnicas conjuntamente con la utilización de imágenes digitales, ha dado lugar a que surjan nuevos flujos de trabajo en las clínicas imagenológicas. En el proceso para realización de estudios médicos, se pueden identificar varias fases: el registro de los datos del paciente, la obtención de las imágenes, el diagnóstico del radiólogo y la entrega de los resultados al interesado. En unión a los anteriores está que es opcional: la entrega de las imágenes a los pacientes.

La entrega de imágenes a los pacientes se ha convertido en algo casi obligatorio en la actualidad. Para los pacientes esto representa un beneficio, por el hecho de poder conservar sus estudios para futuras consultas o simplemente para obtener segundas opiniones de otros especialistas. Estas entregas se realizan en soporte externo, principalmente en CD y en menos ocasiones en DVD. En la actualidad el

quemado de los discos se realiza en las **estaciones de diagnóstico**<sup>22</sup> y en mayor medida en las **computadoras asistentes (wizard)**.

El almacenamiento en dispositivos externos forma parte de los flujos de trabajo de una clínica imagenológica. Para la puesta en práctica de este tipo de almacenamiento existen disímiles razones, algunas necesarias para el mejoramiento de la calidad del servicio y otras con fines científico-educativos.

La obtención de imágenes con fines científico-educativos, se lleva a cabo en aquellas instituciones que lo permita su código de ética. Las imágenes son seleccionadas, anonimizadas y organizadas por los interesados. Luego son almacenadas en algún soporte externo, principalmente en CD y DVD. Esto se realiza en **estaciones de diagnósticos** o en la **wizard**.

El desarrollo tecnológico alcanzado ha permitido la creación de **redes de imágenes** en las instituciones. Esto posibilita que las imágenes luego de almacenadas en un **servidor de imágenes**, sean consultadas por los especialistas desde distintas computadoras. En una clínica imagenológica se realizan una determinada cantidad de estudios diarios lo que provoca que la capacidad de almacenamiento en los **servidores de imágenes** disminuya. Proporcional a la disminución de la capacidad es la disminución de la frecuencia de consulta de las imágenes. La frecuencia de consulta en las primeras horas por lo general es elevada y disminuye en función del tiempo. Pero estas imágenes a pesar de ser poco consultadas no se pueden eliminar debido a que todo hospital implementa como política tenerlas almacenadas por un período de tiempo. Para resolver esto se realiza un almacenamiento pasivo o sea en medios externos, dispositivos menos costosos.

Todos los procesos antes mencionados se realizan con métodos que resuelven los problemas pero no de la manera más eficiente. El quemado de **CD** y **DVD** es una actividad que usualmente toma unos cuantos minutos. En el caso del quemado de los estudios para la entrega a los pacientes, esto se realiza en la mayoría de las veces en la estación de apoyo o **wizard**. Lo anterior provoca que tanto el radiólogo como el equipo<sup>23</sup> de diagnóstico pierdan tiempo de trabajo; el radiólogo invierte su tiempo en el quemado del disco y en la espera de la finalización del mismo y el equipo en todo ese tiempo espera por otro paciente. Cabe resaltar que las consultas radiográficas son una de las más solicitadas, por consiguiente las largas colas de pacientes son frecuentes.

---

<sup>22</sup> Computadora con fines de diagnostico, presenta instalado un software capaz de abrir y realizar transformaciones a las imágenes.

<sup>23</sup> Tomógrafo RM,etc

La selección y quemado de las imágenes con fines científico-educativos se realiza en las estaciones de diagnóstico. Lo cual afecta al servicio de consulta si se realiza en el mismo horario que las consultas.

El almacenamiento pasivo de los estudios en medios externos por las instituciones hospitalarias pasa de una posibilidad a una necesidad. El volumen de información que se manipula es grande. ¿Qué sucede con esto? Simplemente que en la actualidad en las instituciones que se realiza, las imágenes son organizadas de forma manual. La organización manual presenta algunos inconvenientes: el tiempo invertido en ella es considerable, existe una amplia posibilidad de cometer errores y requiere que el realizador posea conocimientos del funcionamiento del **servidor de imágenes**.

De forma general la situación problemática que se puede identificar es la tardanza e ineficiencia en los procesos de organización y almacenamiento en las clínicas imagenológicas. Para dar pie a una solución cabría preguntarse: ¿Cómo facilitar el almacenamiento e intercambio de imágenes médicas utilizando como soportes CD y DVD? Apuntando que esta solución debe de cumplir con lo especificado en el estándar DICOM 3.0, para “hablar” el mismo lenguaje de otras aplicaciones médicas.

### **II.3 Objeto de automatización**

Para mejorar el servicio de las instituciones hospitalarias, que presenten un área de imagenología, es necesario incrementar la eficiencia de los mismos. Existen algunos servicios que presentan partes en las que se puede mejorar su gestión; uno de ellos es el almacenamiento en media.

Para los autores, la automatización de los procesos de organización y almacenamiento en media es el objetivo primordial del presente trabajo. Todo esto debe de ser posible bajo lo estipulado por el estándar DICOM 3.0, para lograr una mayor compatibilidad con otros sistemas.

### **II.4 Información que se maneja**

La información manejada son las imágenes médicas. Dichas imágenes son por lo general imágenes digitales que poseen información adicional o metadatos. La información adicional es tan variada como útil, se pueden encontrar cosas tan generales como el tipo de institución donde se realiza el estudio, y otras tan específicas como la numeración del equipo donde se realiza el estudio.

## II.5 Propuesta de sistema

Basado en el análisis del objeto de estudio y en la definición del objeto de automatización, los autores han concebido implementar un sistema para gestionar la organización y el almacenamiento de imágenes médicas en medios externos. El sistema que se propone consiste en una aplicación de escritorio.

La aplicación se conectará a un servidor DICOM-Compatible y obtendrá de él las imágenes que se quemarán en CD y DVD. Además de permitir obtener las imágenes de forma local. Luego de obtenidas las imágenes se organizarán, de forma automática preferentemente, en paquetes. Estos paquetes el usuario podrá decidir si desea quemarlos inmediatamente o almacenarlos en formas de **imágenes de datos**<sup>24</sup>

El sistema propuesto podrá interactuar con un **PACS** siempre que este cumpla con lo establecido en el estándar **DICOM 3.0**.

En el marco internacional existen diferentes soluciones que poseen muy buenas prestaciones, pero la adopción de las mismas presenta ciertas dificultades. La aplicación de un PACS así como los componentes secundarios de los mismos es un proceso costoso.(Bryan 1999) Las grandes soluciones internacionales que existen por lo general son soluciones integradas de hardware y software. Esta unión obliga a la dependencia hacia determinada tecnología. Además de presentar un costo elevado inherente a sus características, agravado por una serie de parámetros subjetivos a los que se ve expuesta Cuba.

Las soluciones nacionales, que propiamente no son de almacenamiento, no resuelven un porcentaje elevado del dominio de problemas relacionados con la organización y almacenamiento de imágenes médicas. Las soluciones locales presentan como dificultad la organización manual de la información, problema que se pretende paliar en la aplicación con métodos automáticos para la organización.

---

<sup>24</sup> Se refiere a la organización de varios ficheros en el mismo formato en que se quemarían.

**II.6 Modelo de negocio**

**II.6.1 Actores del Negocio**

Los actores del negocio son aquellas personas o sistemas que obtienen un beneficio como resultado de uno o varios procesos del negocio. En la tabla siguiente se muestran los actores del negocio en cuestión:


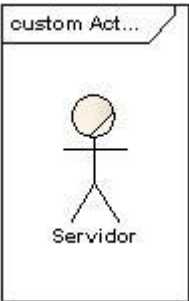
Actor	Justificación
	<p>El actor paciente es la persona que se beneficia con el proceso de realizarse un estudio. El mismo llega a una clínica con el objetivo de realizarse un estudio. Generalmente desea conservar ese estudio en un CD o DVD.</p>
	<p>Este actor es un servidor de imágenes médicas, DICOM compatible, se beneficia con un proceso del negocio. Este beneficio consiste en un aumento de la capacidad de almacenamiento debido a la liberación de espacio. El mismo es capaz de presentar alertas cuando esté agotando su capacidad de almacenamiento.</p>

Tabla 1 Actores del negocio.

**II.6.2 Trabajadores del negocio**

En un negocio los trabajadores son aquellas personas o sistemas que están involucrados en uno o más procesos del negocio, pero no obtienen ningún resultado de valor. El trabajador del negocio colabora con otros trabajadores, es notificado con eventos de negocio y manipula entidades de negocio para realizar sus responsabilidades. En la tabla siguiente se pueden encontrar los trabajadores identificados:






Trabajador	Justificación
 Radiólogo	Ente principal en la realización del estudio. Su trabajo consiste en analizar las imágenes que se obtienen del equipo para emitir un diagnóstico.
 Tecnólogo	Técnico que colabora con el proceso de realizar un estudio. Le toma los datos al paciente, lo prepara y guía para la realización del estudio. Es el encargado de verificar que el equipo esté listo para realizar el estudio.
 Administrador de Servidor	Es el encargado de administrar y trabajar en el servidor de imágenes DICOM. Velará por correcto funcionamiento del servidor.

Tabla 2 Trabajadores del negocio.

### II.6.3 Diagrama de Casos de uso del negocio

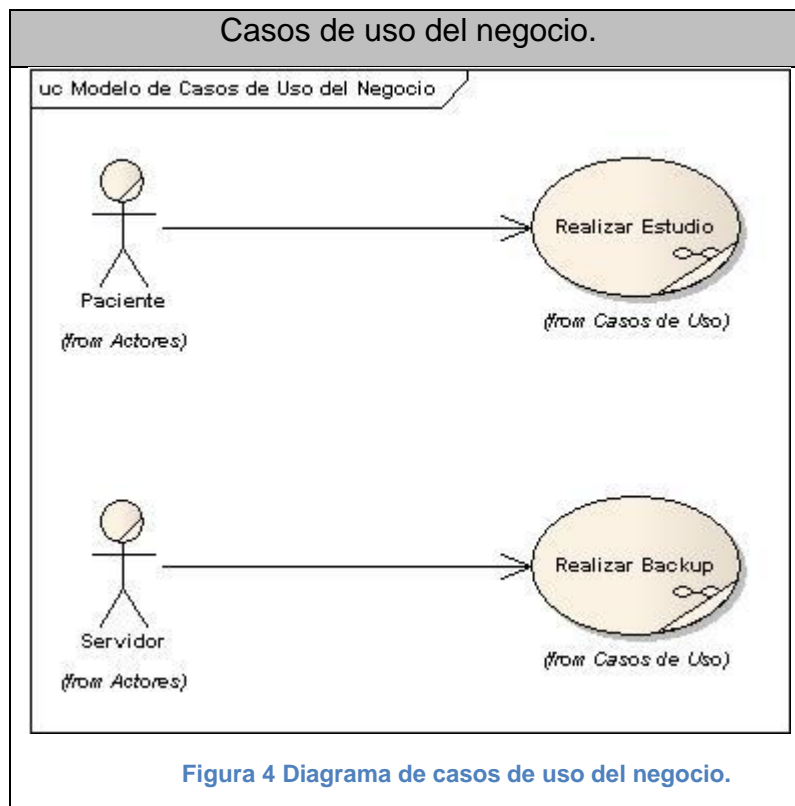


Figura 4 Diagrama de casos de uso del negocio.

II.7 Diagramas de actividades de los Casos de Uso del Negocio

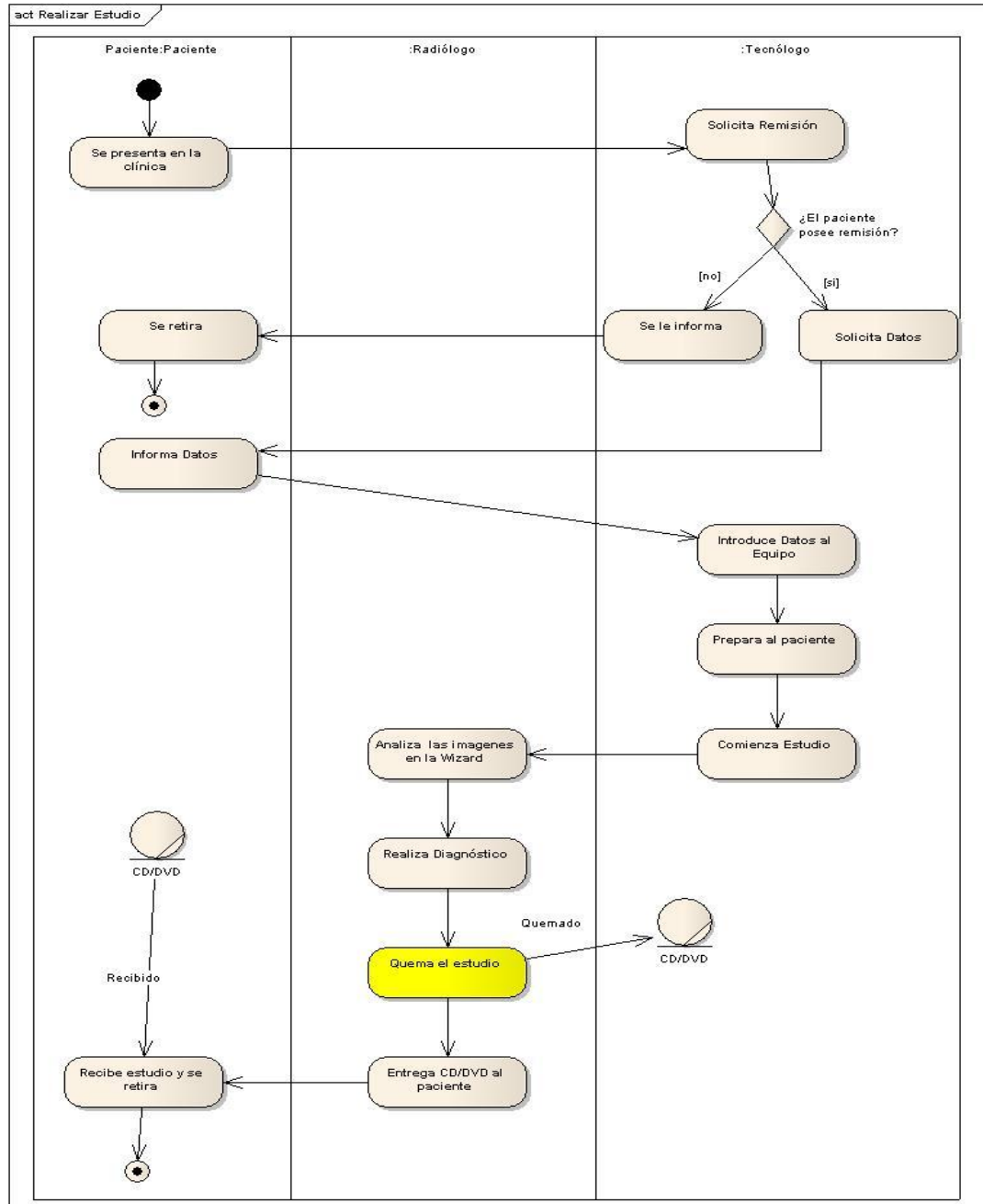


Figura 5 Diagrama de Actividades CU Realizar Estudio

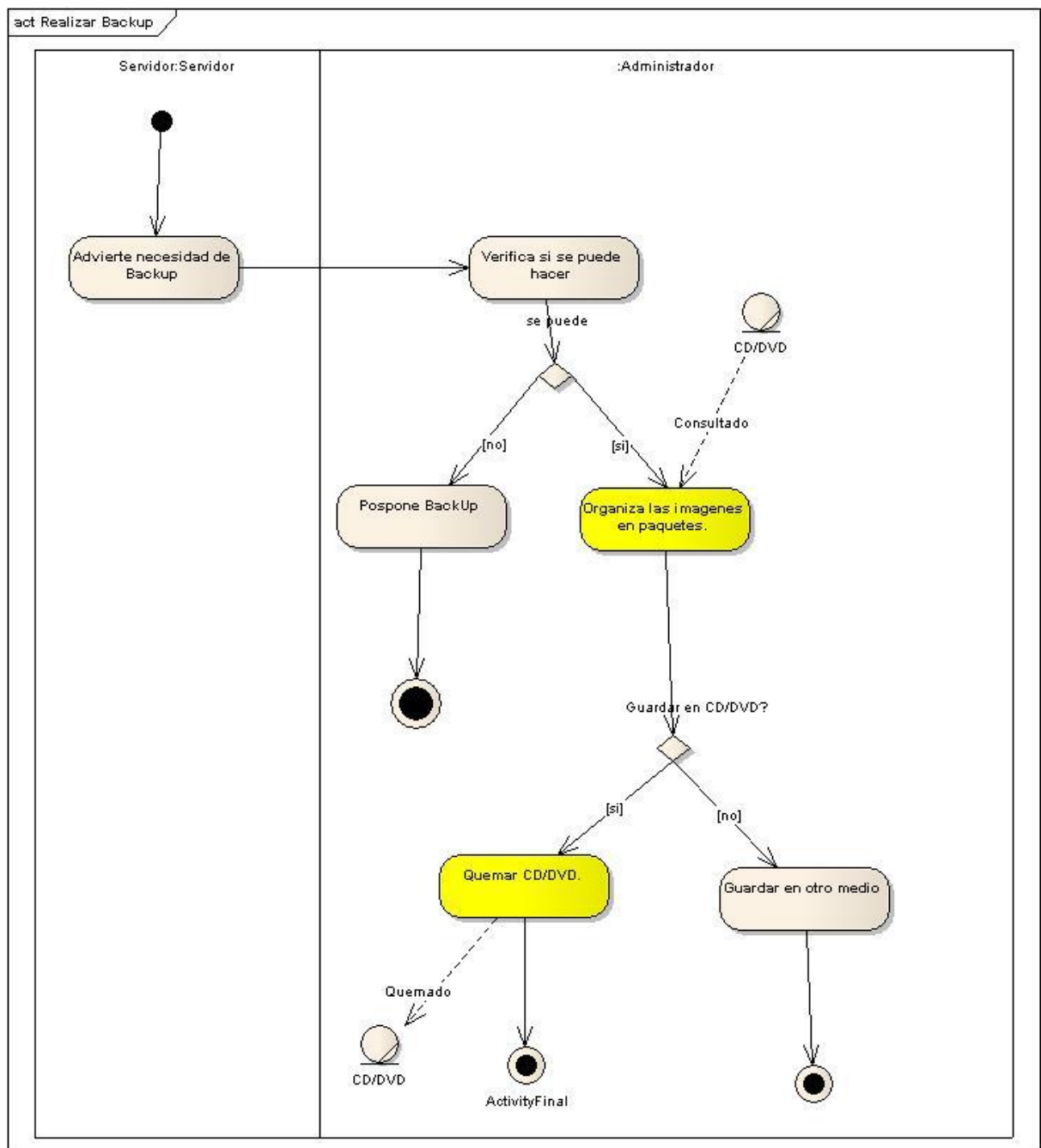
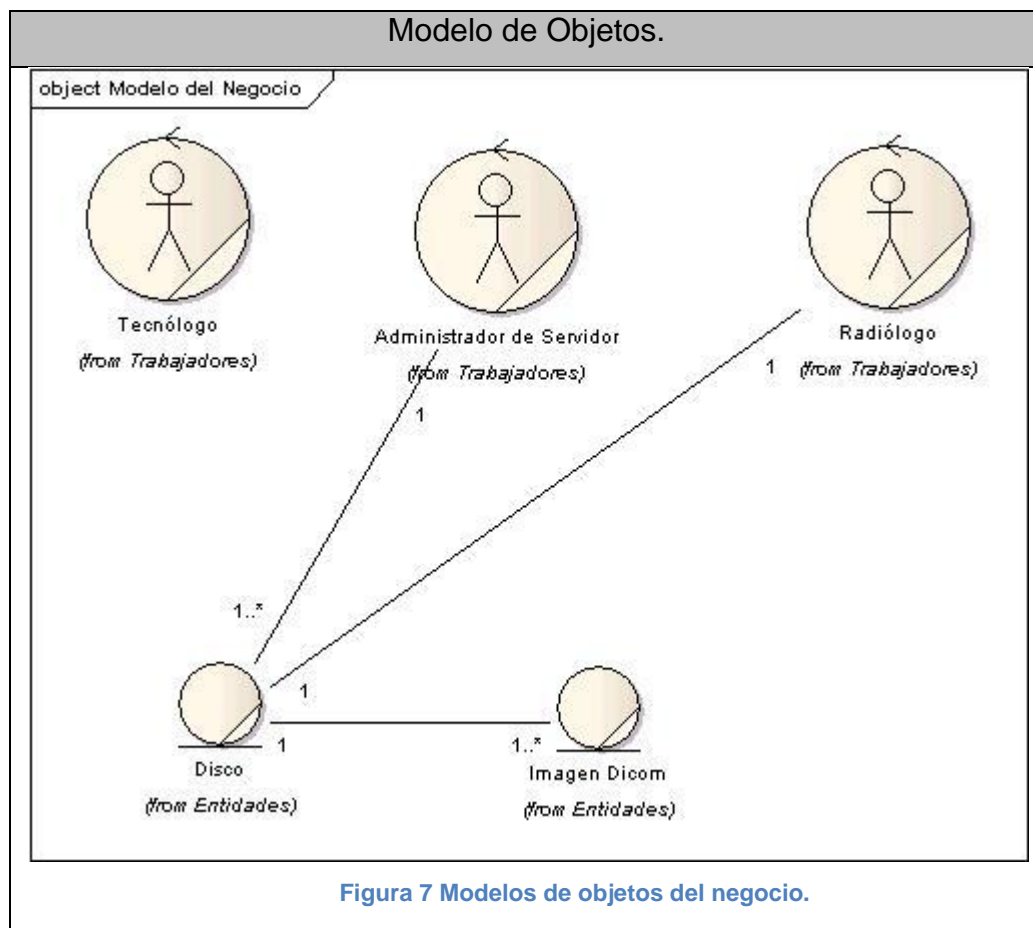


Figura 6 Diagrama Actividades CU Realizar Backup

### II.7.1 Diagrama de Clases del Modelo de Objetos

El modelo de objetos del negocio expresa las relaciones que existen entre las clases del negocio. Aunque en el negocio que se presenta se podrán encontrar otros objetos, solamente se han puesto los referentes a la situación problemática.



### II.8 Especificación de los requisitos de software

El Servidor de Quemado (Alas PACS Burner) está pensado para ser un software que colabore dentro de un **PACS** ya que se relaciona con un servidor **DICOM** compatible. Este servidor es el encargado de proveerle las imágenes a Alas PACS Burner, ya sea para realizar **backup** o para la entrega de resultados.

Esto no quiere decir que este sujeto a esta restricción, por lo que podría trabajar de forma independiente. El sistema se puede integrar al Alas PACS e interactuar con los elementos que conforman al mismo.

### II.8.1 Requisitos funcionales

Los requerimientos funcionales se agrupan atendiendo a las funcionalidades que generarán. Los cuatros grupos son los siguientes:

- **Obtención de DICOM:** Son los que especifican la recuperación de imágenes del servidor DICOM compatible.
- **Empaquetado de las imágenes:** Todos los referentes a la organización de las imágenes en paquetes. Para mejor comprensión de la relación entre los mismos revisar el **Anexo 1 Diagrama de requisitos: Organizar las imágenes** (pág. 60)
- **Quemado de las imágenes:** Estos agrupan las características referentes al proceso de quemado de las imágenes en los dispositivos, específicamente en CD/DVD.
- **Configuración:** Elementos que se requieren para la configuración del software.

En las páginas siguientes se expondrán los requerimientos que componen cada uno de los grupos antes mencionados.

#### Obtención de DICOM

Requerimiento.	Descripción.
REQ001 -Buscar datos de las imágenes almacenadas en un servidor.	Se quiere que dado algunos descriptores se busque en el servidor DICOM-compatible y se muestren los principales datos. Esto se debe realizar con un C-FIND de acuerdo a lo estipulado en el DICOM 3.0
REQ003 - Obtener imágenes a partir del nivel de pacientes en un servidor DICOM compatible	La capacidad del software para que a partir de criterios de selección obtenga imágenes del servidor. Estas imágenes deben ser a partir del nivel de paciente.
REQ004 - Obtener imágenes a partir del nivel de estudio en un servidor DICOM compatible	Al igual que el anterior es la capacidad del software para que a partir de criterios de selección obtenga imágenes del servidor. Pero en este punto deben de

	obtenerse las imágenes a partir del nivel de estudio.
REQ010 - Incluir ficheros DICOM desde discos locales	El sistema debe ser capaz de encontrar, empaquetar y quemar en CD/DVD imágenes DICOM almacenadas en los medios de almacenamiento de la propia máquina.
REQ019 - Cargar FileSet desde DicomDir	El sistema debe ser capaz de leer un DicomDir e incluir en el paquete las imágenes referenciadas en el FileSet.

Tabla 3 RF del paquete de Obtención de DICOM.

### Empaquetado de las imágenes

Requerimiento.	Descripción.
REQ008 - Indexar File-Set con un DICOMDirectory	<p>Consiste en que se debe resumir los datos fundamentales de las imágenes médicas que se empaqueten y guardarlas en un fichero.</p> <p>Una vez que están definidos los archivos que se van a poner en un paquete, estos tiene que indexarse con un DICOMDirectory. Se tiene que seguir lo estipulado por el estándar DICOM 3.0.</p>
REQ011 - Construir paquetes optimizando espacio	Los paquetes deben ser organizados de manera tal que se aproveche de la forma más eficiente posible el espacio de almacenamiento disponible en los CD/DVDs.
REQ012 - Construir paquetes por afinidad de paciente	Un modo de empaquetado debe garantizar que las imágenes pertenecientes a un mismo paciente se empaqueten en la menor cantidad posible de volúmenes.
REQ013 - Construir paquete de imágenes de un solo paciente	Se requiere que el sistema brinde la posibilidad de crear paquetes de imágenes pertenecientes a un

	solo paciente. Estas serán generalmente las correspondientes al estudio que se realizó recientemente.
REQ017 - Generar identificador de paquete	El sistema debe generar un identificador único para cada paquete de imágenes DICOM.
REQ020 Empaquetar Imágenes	El sistema debe crear paquetes que aglomeren a las imágenes que el usuario seleccionó.

Tabla 4 RF del paquete de Empaquetado de las imágenes.

### Quemado de las imágenes

Requerimiento.	Descripción.
REQ005 - Quemar en CD	Requiere la condicionalidad del sistema de quemar en CD.
REQ007 - Quemar en DVD	Necesidad de que el software soporte quemar en DVD. Para ello los paquetes deben de estar acorde a las condiciones del DVD.
REQ009 - Realizar imagen de los datos	Especifica que se deben crear imágenes de los datos antes de quemarlos.
REQ018 - Etiquetar CD/DVD	Poner al CD/DVD como etiqueta de software (Volume Label) el identificador único del paquete que contiene.

Tabla 5 RF del paquete Quemado de las imágenes.

### Configuración

Requerimiento.	Descripción.
REQ014 - Configurar el origen de las imágenes	Se debe poder especificar y seleccionar la fuente de las imágenes médicas. Esta puede ser un servidor DICOM compatible o de forma local. En caso de ser local debe poder darse la capacidad de leer un Dicomdir.

REQ015 - Configurar las grabadoras a usar	Si el equipo tiene varias grabadoras (Quemadores), se le especifican las que se usarán en el proceso de quemado.
REQ016 - Configurar los tamaños de los paquetes	Especifica de qué tamaño serían los paquetes que se generarían. Debe de haber dos predeterminados, uno para CD y otro para DVD. Pero el usuario puede definir otros.

Tabla 6 RF del paquete Configuración.

## II.8.2 Requisitos no funcionales

- Usabilidad:  
El sistema debe ser sencillo de operar. Todo el proceso debe de ser transparente para el usuario, por lo que este no debe poseer grandes conocimientos de la informática para trabajar con el software. Por otro lado en el sistema se podrán configurar las características referentes al quemado y al empaquetado.
- Rendimiento:  
Para mejorar el rendimiento el sistema debe de ser capaz de quemar en varios quemadores simultáneamente. Además de realizar algunas tareas de forma simultáneas tales como: empaquetar y quemar, obtener imágenes y quemar, entre otras.
- Interfaz gráfica:  
El software debe tener una interfaz atractiva y amigable, afín a los sistemas de quemado más comunes. Debe de tenerse en cuenta las experiencias de usuarios.
- Legales:  
El sistema y toda la documentación generada con el mismo pertenecen al Grupo de Procesamiento de Imágenes y Señales de la Universidad de las Ciencias Informáticas.
- Portabilidad:  
El módulo de empaquetado debe de ser compatible con MONO, para una posible migración a Linux.



## II.9 Definición de los casos de uso

Un caso de uso es, en esencia una interacción típica entre un usuario y un sistema de cómputo (Flower 1999). El mismo capta alguna funcionalidad visible para el usuario y logra un objetivo discreto para el mismo. Los casos de uso pueden considerarse como un documento narrativo que describe la secuencia de eventos de un actor que utiliza un sistema para completar un proceso (Jacobson 1992).

Durante el levantamiento de requisitos para el presente trabajo se identificaron los actores y casos de uso que guiarían el desarrollo del producto final. En esta sección se enumerarán y describirán cada uno de ellos.

### II.9.1 Actores del Sistema

Los actores del sistema se definen como un conjunto coherente de roles que un usuario del sistema puede desempeñar cuando interactúa con el mismo. Un actor puede ser un individuo o un sistema externo. En el caso del software en cuestión, solo aparece un actor.


Actor	Descripción
 Usuario	El único actor que tendrá el sistema. Es el encargado de explotar todas las funcionalidades del sistema.

Tabla 7 Actores del Sistema.

### II.9.2 Casos de Uso Alas PACS Burner

Los casos de uso se obtienen de las necesidades requeridas por los usuarios finales, por lo que son un buen faro para guiarse en el desarrollo del software. A continuación se listan los que se identificaron en la ejecución de este trabajo.

<b>CU01</b>	Configurar Servidores
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Consiste en establecer el o los servidores y probar sus conexiones. Desde estos servidores se obtendrán las imágenes que se usarán en el proceso de organización y almacenamiento.
<b>Referencia:</b>	REQ014

<b>CU02</b>	Configurar Empaquetado
<b>Actores:</b>	Usuario
<b>Resumen:</b>	El usuario podrá cambiar algunas opciones de empaquetado, como el tamaño de los mismos.
<b>Referencia:</b>	REQ016

<b>CU03</b>	Configurar Grabadoras
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Permitirá configurar las grabadoras o quemadores que se van a usar y verificar los que se han detectado.
<b>Referencia:</b>	REQ015

<b>CU04</b>	Crear DicomDir
<b>Actores:</b>	
<b>Resumen:</b>	Abrirá las imágenes y tomará del metadato de la misma los campos relevantes. Creando con esto un DICOMDirectory que se incluirá en el paquete que se cree.
<b>Referencia:</b>	REQ008, CU05

<b>CU05</b>	Empaquetar
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Este caso de uso realizará la selección y organización de las imágenes

	que formarán parte de los paquetes. Esto se debe de regir por el tipo de empaquetado que el usuario haya definido.
<b>Referencia:</b>	REQ011, REQ012, REQ013, REQ017, REQ020, CU04, CU06

<b>CU06</b>	Seleccionar Modo Empaquetado
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Brindará la funcionalidad de que el usuario seleccione el tipo de empaquetado que desee para la organización. Los tipos de empaquetado son: por afinidad de pacientes, por optimización de espacio y de un solo paciente (el estudio realizado).
<b>Referencia:</b>	CU05

<b>CU07</b>	Buscar imágenes DICOM
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Busca en un servidor DICOM-Compatible las imágenes que cumplan con los descriptores propiciados por el usuario. Esta operación se ejecuta mediante el uso de un C-FIND de acuerdo a lo especificado en el estándar DICOM 3.0.
<b>Referencia:</b>	REQ001

<b>CU08</b>	Leer DicomDir
<b>Actores:</b>	
<b>Resumen:</b>	Carga toda la información que existe en un directorio DICOM y la comprueba.
<b>Referencia:</b>	REQ019, CU11

<b>CU09</b>	Obtener Imágenes DICOM Locales
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Es el encargado de proporcionar los medios para que el usuario seleccione las imágenes que se empaquetarán de forma local. El

	usuario seleccionará las ubicaciones de las mismas en los discos duros locales.
<b>Referencia:</b>	REQ010

<b>CU10</b>	Obtener Imágenes DICOM desde Servidor.
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Luego de tener localizadas las imágenes que se encuentran en el servidor, se procede a copiarlas hacia la máquina cliente. Esta operación se realiza con la implementación de un C-STORE.
<b>Referencia:</b>	REQ003, REQ004

<b>CU11</b>	Obtener desde un File-Set.
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Obtiene las imágenes desde un File-Set existente. Es muy parecido a obtenerlas de forma local pero ahora se realiza cargando la información que se encuentra en el DicomDirectory que acompaña al File-Set.
<b>Referencia:</b>	REQ019, CU08

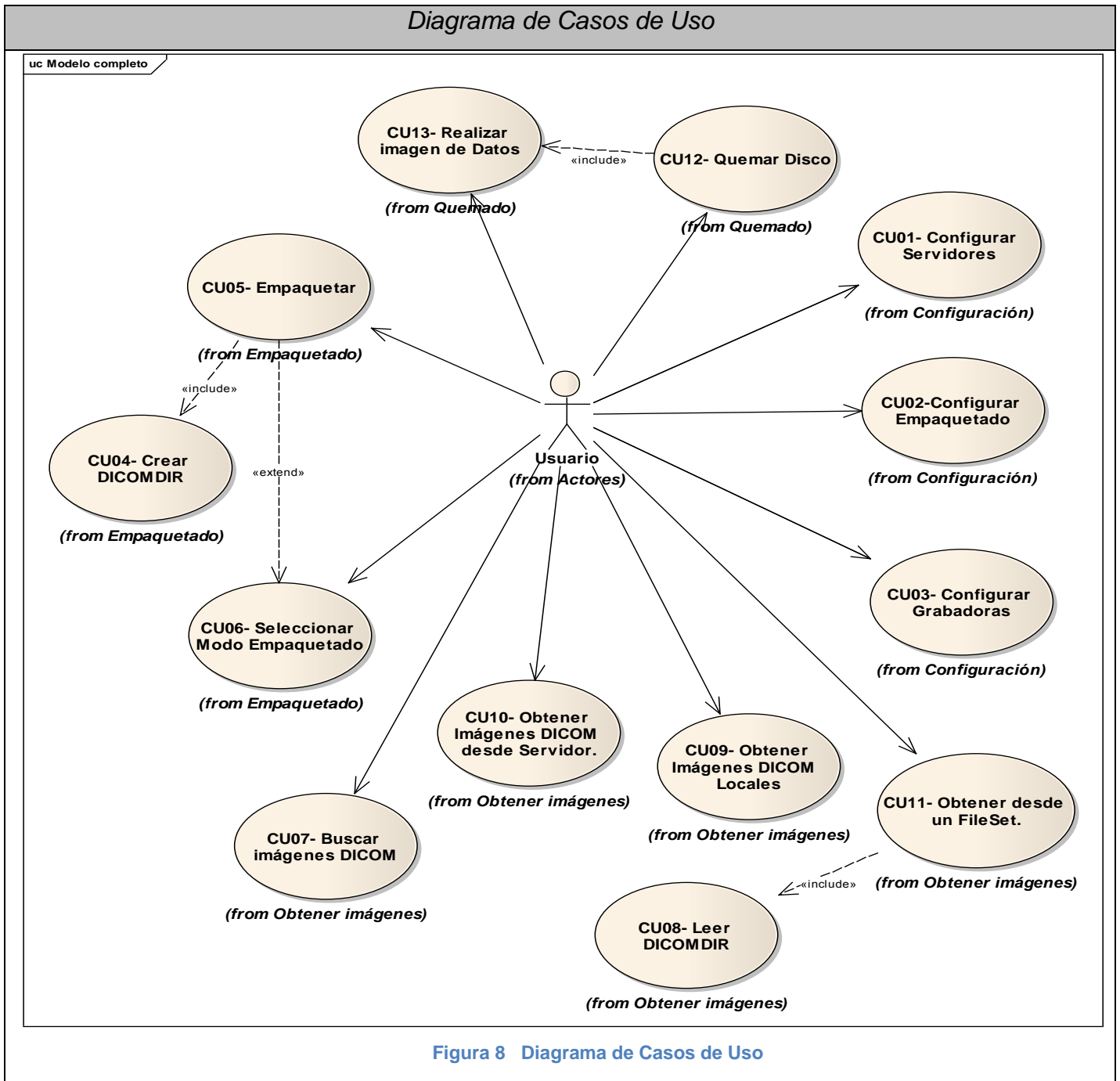
<b>CU12</b>	Quemar Disco
<b>Actores:</b>	Usuario
<b>Resumen:</b>	El proceso de almacenamiento, de los paquetes creados, en medios de almacenamiento. En este proceso se utiliza como medio de almacenamiento el CD y el DVD.
<b>Referencia:</b>	REQ005, REQ007, REQ018, CU13

<b>CU13</b>	Realizar imagen de Datos
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Consiste en realizar una composición de datos similar a la que contendrá el disco. Esta luego se quemará en disco o podrá salvarse para el disco duro.
<b>Referencia:</b>	REQ009, CU12

### II.9.3 Diagrama de Casos de Uso

El diagrama de casos de uso es un artefacto importante dentro del UML, ya que permite modelar el comportamiento de un sistema, un subsistema o una clase (Booch, Rumbaugh and Jacobson 1999), según sea el caso. Este diagrama explica gráficamente un conjunto de casos de uso, el actor y las relaciones que existen entre el actor y los casos de uso.

Esta sección tiene como objetivo ofrecer un diagrama contextual que permita reconocer rápidamente el actor externo del sistema y la forma básica en que este lo utiliza.



## II.10 Casos de uso por ciclo

Para una mejor distribución del trabajo y el aseguramiento de las funcionalidades indispensables según su necesidad, se dividió la realización e implementación de los distintos casos de uso. Se distribuyeron los casos de uso entre tres ciclos de desarrollo:

### II.10.1 Primer ciclo de desarrollo

Cód.	Nombre de caso de uso	Paquete	Justificación de la selección.
CU04	Crear DicomDir	Empaquetado	Para la culminación de un paquete es necesario indexarlo para que otros sistemas lo puedan utilizar. Si no está indexado no es estándar.
CU05	Empaquetar	Empaquetado	Este caso de uso constituye el corazón de la aplicación. Todo el que se remita a usar la aplicación tendrá como necesidad primaria, agrupar los ficheros DICOM-Compatibles. Con la finalización de este caso de uso se podrá quemar el paquete en un disco.
CU12	Quemar Disco	Quemado	Para que la aplicación preste un buen servicio es necesario quemar en un CD o DVD las imágenes. Por ello es una prioridad que para el primer ciclo tener la capacidad de quemar discos.

Tabla 8 Casos de uso para el primer ciclo de desarrollo.

### II.10.2 Segundo ciclo de desarrollo

Cód.	Nombre de caso de uso	Paquete	Justificación de la selección.
CU05	Empaquetar	Empaquetado	Se le agregan nuevas funcionalidades, nuevos tipos de empaquetados. Una de ellas es el empaquetamiento por afinidad de pacientes.
CU12	Quemar Disco	Quemado	La realización e implementación de este caso de uso se extiende al segundo ciclo ya que es necesario incorporar la funcionalidad de quemar varios discos al mismo tiempo. Esto siempre que se pueda, es decir si la máquina lo soporta.
CU07	Buscar imágenes DICOM	Empaquetado	Para poder obtener las imágenes desde un servidor DICOM-compatible es necesario conocer acerca de la existencia de las mismas, así como su información. Es por eso que este caso de uso se selecciona para este ciclo.
CU09	Obtener Imágenes DICOM Locales	Empaquetado	El caso básico para la obtención de las imágenes es que las mismas se encuentren en la maquina que hospeda la aplicación.
CU10	Obtener Imágenes DICOM desde Servidor.	Empaquetado	Luego de poder obtener las imágenes de forma local se hace imprescindible la obtención de las mismas de forma remota.
CU13	Realizar imagen de Datos	Quemado	Como medio alternativo a quemado de discos se puede dar la creación de imágenes de datos. Esto no cambia radicalmente la funcionalidad de la



			aplicación es por ello que se asigna a este ciclo de desarrollo.
--	--	--	--

Tabla 9 Casos de uso para el segundo ciclo de desarrollo.

### II.10.3 Tercer ciclo de desarrollo

Cód.	Nombre de caso de uso	Paquete	Justificación de la selección.
CU01	Configurar Servidores	Configuración	Estos casos de uso son los encargados de modificar los descriptores de la configuración del sistema. Su realización e implementación aunque importante, no le dan demasiado valor a la aplicación.
CU02	Configurar Empaquetado		
CU03	Configurar Grabadoras		
CU06	Seleccionar Modo Empaquetado	Empaquetado	Funcionalidades que sirven para darle valor agregado a la aplicación.
CU08	Leer DicomDir	Obtener imágenes	
CU11	Obtener desde un FileSet.		Aunque no por ello dejan de ser importantes para los usuarios de la aplicación.

Tabla 10 Casos de uso para el tercer ciclo de desarrollo.

### II.10.4 Casos de uso expandidos

Los casos de uso expandidos muestran más detalles del problema a analizar que los de alto nivel, suelen ser útiles para alcanzar un conocimiento más profundo de los procesos y requerimientos. (Larman 1999) Las descripciones textuales de los principales casos de uso se podrán encontrar en el **Anexo 2 Casos de Uso expandidos** (pág. 61).

**Conclusiones**

En este capítulo se presentó formalmente el problema científico, la situación problemática y el objeto de automatización. Se explicó la necesidad de desarrollar una aplicación que resuelva el problema planteado; se hizo una descripción general del sistema que propone este trabajo de diploma y una comparación con algunos de los sistemas existentes. Además, se expuso el modelo de negocio, la especificación de requerimientos funcionales y no funcionales del sistema y finalmente se mostraron los diagramas de casos de uso.

**CAPÍTULO III ANÁLISIS Y DISEÑO DEL SISTEMA****Introducción**

En esta fase, se analizan los requisitos que se describen en la captura de requisitos, refinándolos y estructurándolos. El objetivo es conseguir una comprensión más precisa de los requisitos. Esto no quiere decir que el trabajo se repita, ya que en el mismo se usa un lenguaje más técnico, un lenguaje de programadores, para describir los resultados.

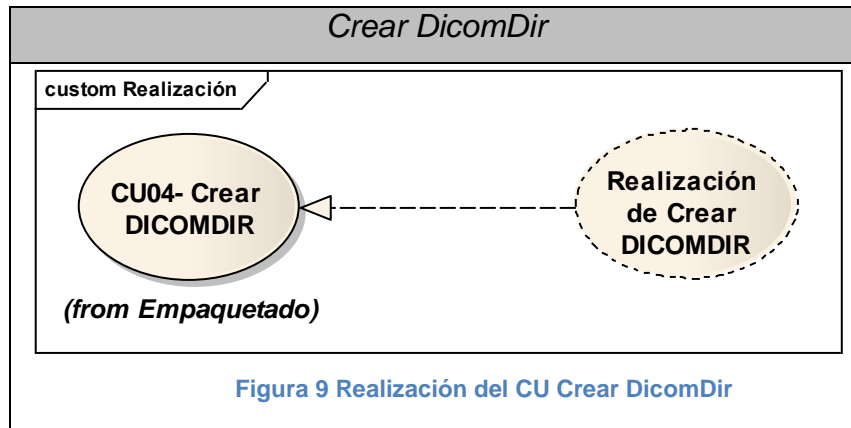
Para la realización del análisis de Alas PACS Burner el refinamiento de los requisitos se llevó a cabo razonando aspectos internos sobre el sistema. Esto permitió resolver situaciones relativas a interferencias y conflictos que pudieron existir entre los casos de uso.

En este capítulo se presentan las clases de análisis y las clases de diseño que intervienen en la realización de los casos de uso. Además de resaltarse la dinámica que existe entre las distintas clases a través de diagramas de interacción. Cabe señalar que únicamente se han tenido en cuenta aquellos casos de uso que presentan mayor relevancia para el software en cuestión.

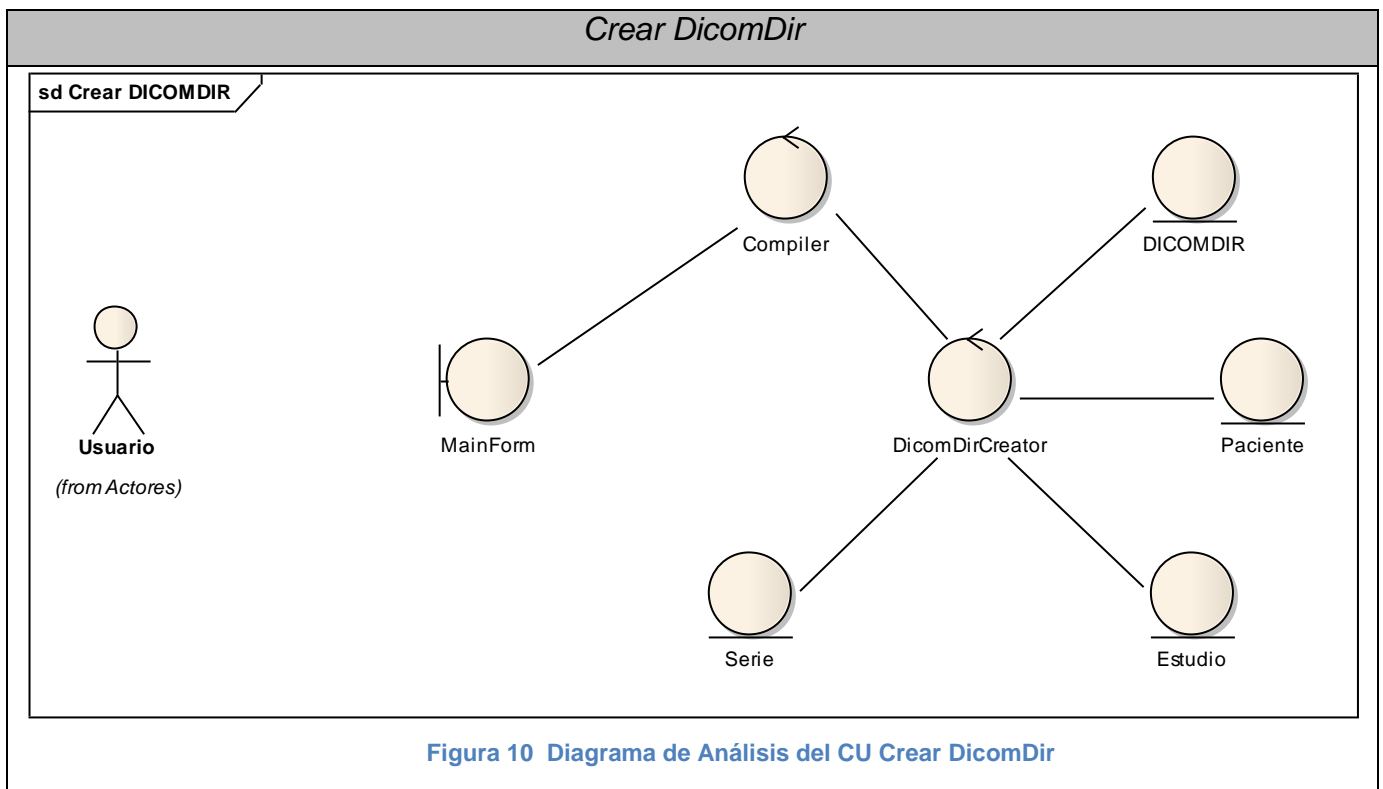
El capítulo se ha organizado en diferentes secciones, haciendo corresponder cada una con un caso de uso en específico. En cada sección se expondrá el análisis y el diseño correspondiente al caso de uso que la misma trate.

**III.1 Análisis y diseño del caso de uso: Crear DicomDir**

En esta sección se presentan las clases de análisis y clases de diseño del caso de uso: Crear DicomDir. La responsabilidad de este caso de uso es que a partir de un grupo de imágenes construya un fichero DicomDir.



III.1.1 Diagrama de clases de análisis



III.1.2 Diagrama de clases de Diseño

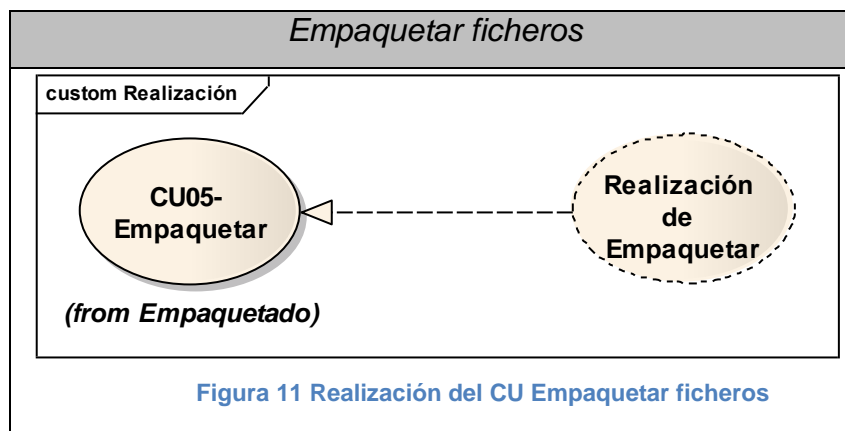
El diagrama de clases de diseño se podrá encontrar en el **Anexo 3 Diagramas de Clases de Diseño**(pág.66)

### III.1.3 Descripción de las clases de diseño

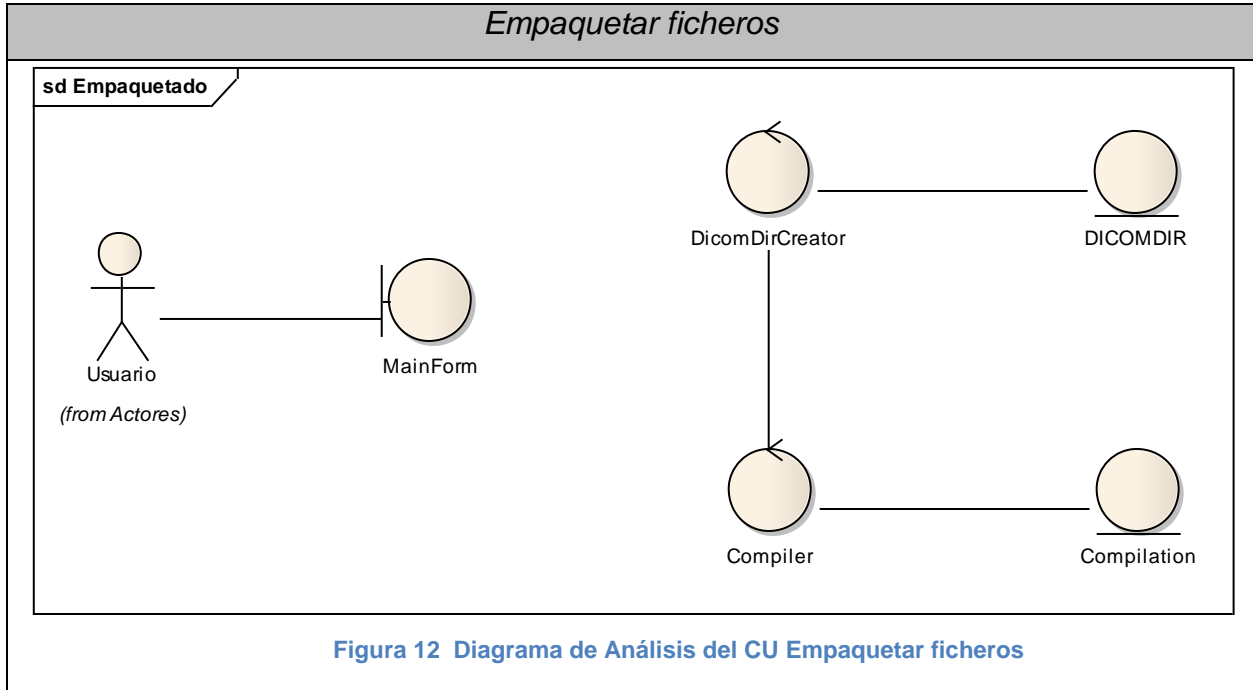
Para consultar la descripción de alguna de las clases de diseño diríjase a **Descripción de clases del CU Crear DicomDir** En el **Anexo 4 Descripción de Clases de Diseño** (pág. 69)

### III.2 Análisis y diseño del caso de uso: Empaquetar ficheros

En esta sección se presentan las clases de análisis, clases de diseño y el diagrama de secuencia del caso de uso Empaquetar ficheros. El mismo se encarga de la selección y organización de las imágenes que conformarán los paquetes.



**III.2.1 Diagrama de clases de análisis**



**III.2.2 Diagrama de clases de Diseño**

Para observar el diagrama de las clases de diseño remítase a **Anexo 3 Diagramas de Clases de Diseño**(pág. 66)

III.2.3 Diagrama de interacción

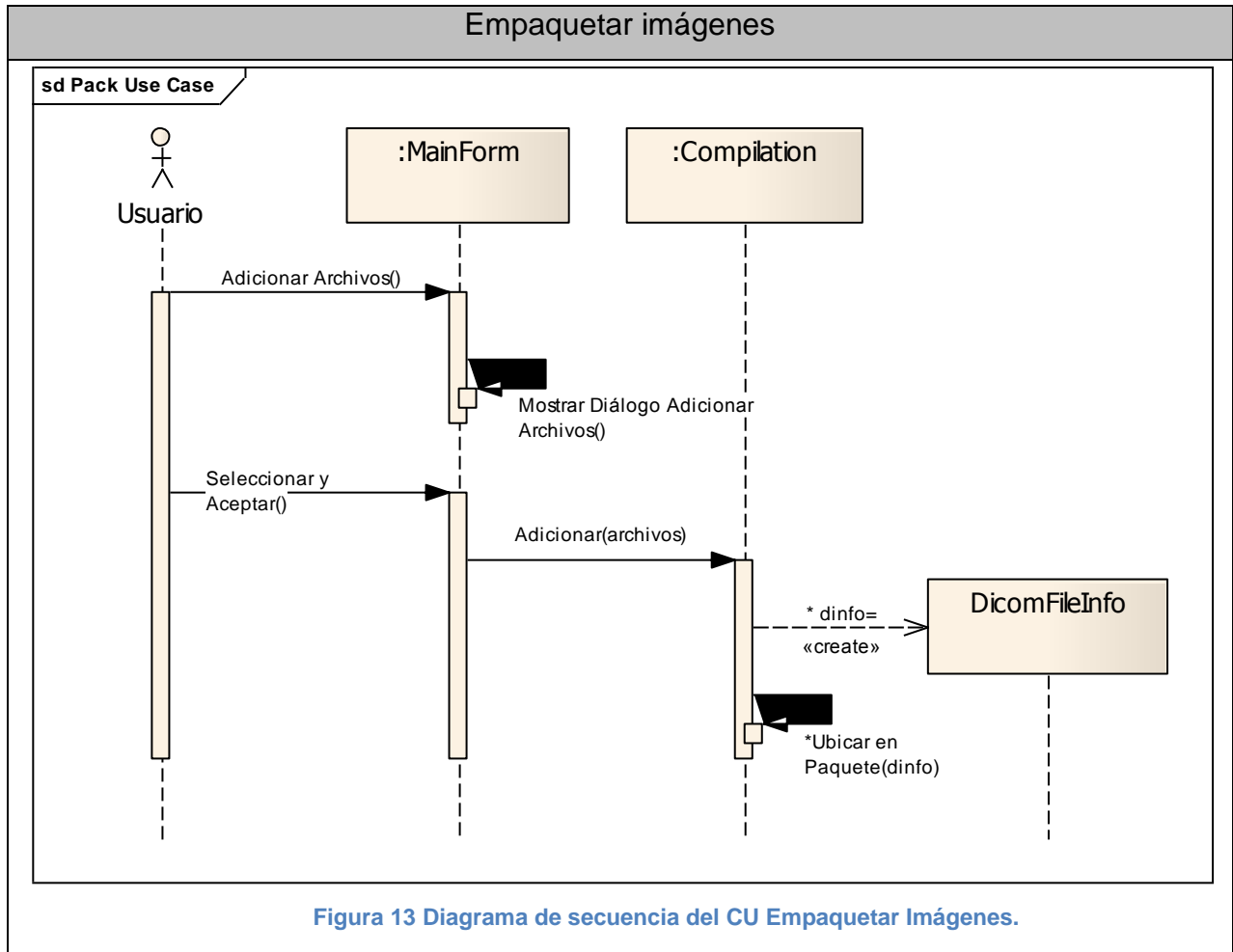


Figura 13 Diagrama de secuencia del CU Empaquetar Imágenes.

III.2.4 Descripción de las clases de diseño

Para consultar la descripción de alguna de las clases de diseño diríjase a **Descripción de clases del CU Empaquetar Imágenes** en el **Anexo 4 Descripción de Clases de Diseño** (pág. 74)

### III.3 Análisis y diseño del caso de uso: Quemar Disco

En esta sección se presentan las clases de análisis, clases de diseño y el diagrama de secuencia del caso de uso Quemar Disco (Figura 14). Este caso de uso es el encargado del proceso del almacenamiento de los paquetes creados en CD o DVD.

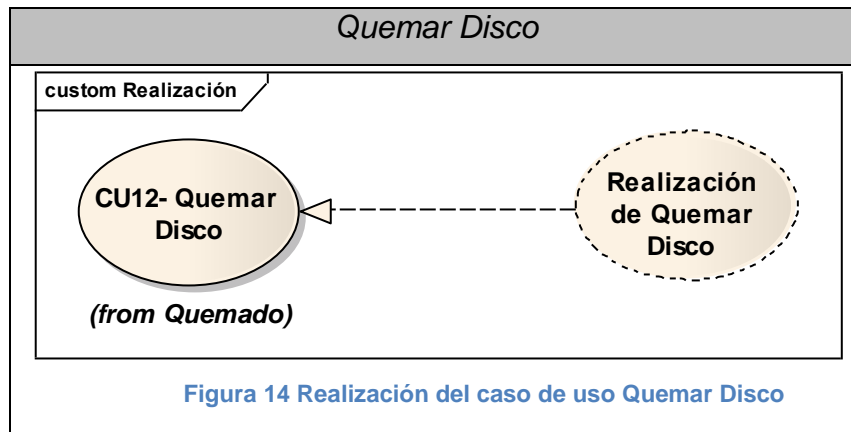


Figura 14 Realización del caso de uso Quemar Disco

### Diagrama de clases de análisis

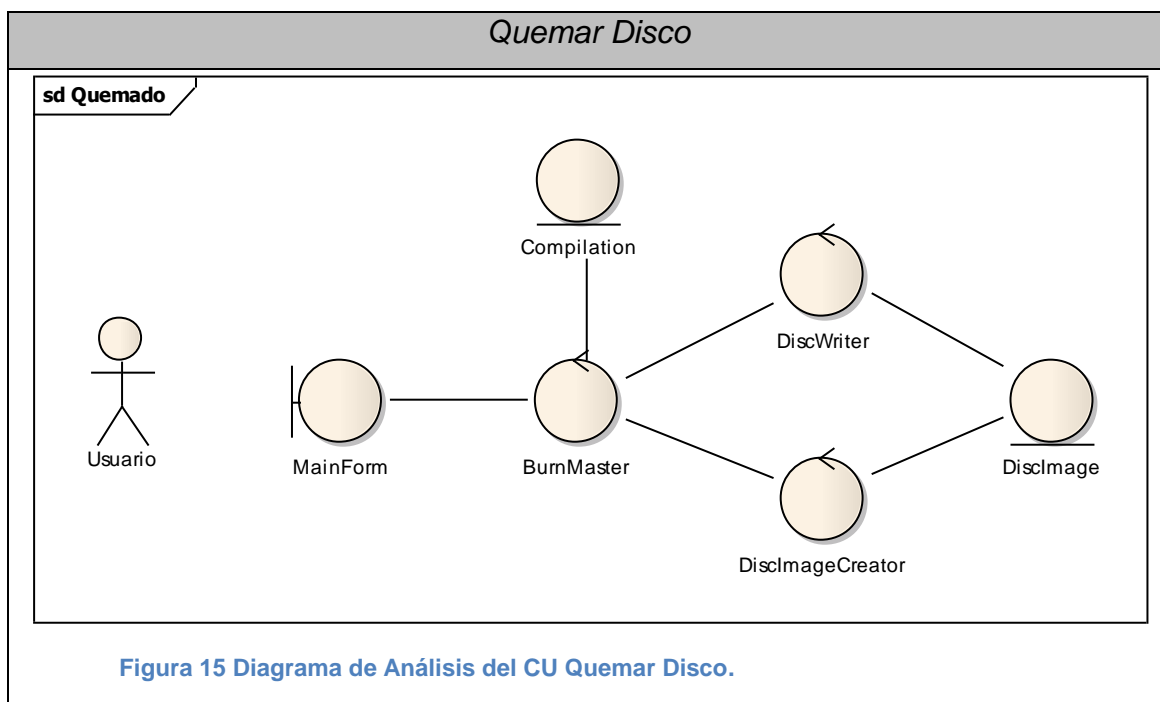


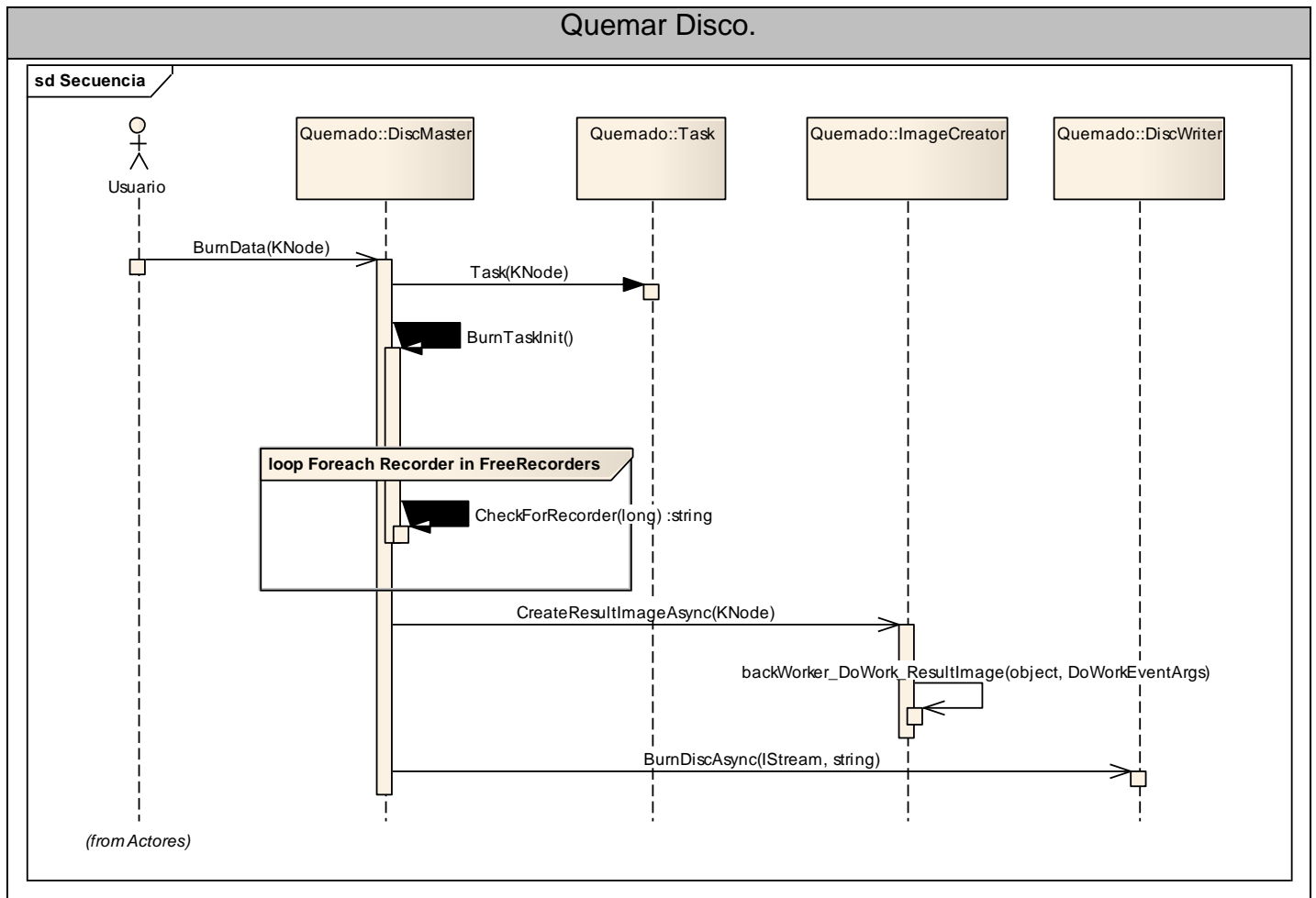
Figura 15 Diagrama de Análisis del CU Quemar Disco.



### III.3.1 Diagrama de clases de Diseño

Para observar el diagrama de las clases de diseño remítase al **Anexo 3 Diagramas de Clases de Diseño**(pág. 66)

### III.3.2 Diagrama de interacción



### III.3.3 Descripción de las clases de diseño

Para consultar la descripción de alguna de las clases de diseño diríjase a **Descripción de clases del CU Quemar Disco** en el **Anexo 4 Descripción de Clases de Diseño** (pág. 85)

### Conclusiones

Hasta aquí se analizaron los requisitos descritos en la captura de los mismos, refinándolos y estructurándolos.

En este capítulo se presentaron las clases de análisis y las clases de diseño que intervienen en la realización de los casos de uso. Además de resaltarse la dinámica que existe entre las distintas clases a través de diagramas de interacción.

## CAPÍTULO IV IMPLEMENTACIÓN DEL SISTEMA

En este capítulo quedan expuestos los componentes de Alas PACS Burner, además de plantear las relaciones que existen entre los mismos. Más adelante se podrá encontrar la disposición física del sistema en los distintos nodos que para ello se han definido.

### IV.2 Componentes del Software

En la ciencia como en casi todos los procesos de la vida se aplica la máxima de la programación algorítmica de “divide y vencerás”. Ante una situación compleja o extensa se pueden separar problemas o procesos que compartan algunas características en común, ya sea de comportamiento, escenario o funcionalidades, en otros más sencillos o más delimitados. Una vez hecho esto la tarea de resolver esto es relativamente menos compleja.

En la informática, la programación orientada a objetos (POO) con su abstracción del mundo permite modelar en términos de códigos algunos de los problemas antes mencionados. Es por ello que para la construcción del software referenciado se decidió separar las funcionalidades que compartían interfaces y comportamientos, procediendo a agruparlas en componentes. Entendiéndose por componente como los elementos físicos del sistema. Las clases y otros elementos de bajo nivel se unen para formar componentes de alto nivel.

El siguiente diagrama de componentes muestra los principales componentes construidos, así como las relaciones entre ellos. En el mismo se pueden identificar dos componentes fundamentales: el UCI.Alas.Packaging y el UCI.Alas.Burner, los que soportan el peso de la aplicación. Estos últimos son los encargados del empaquetado y el almacenamiento en media respectivamente. Los demás componentes son principalmente de presentación y de soporte, importantes para el funcionamiento del software como un todo.

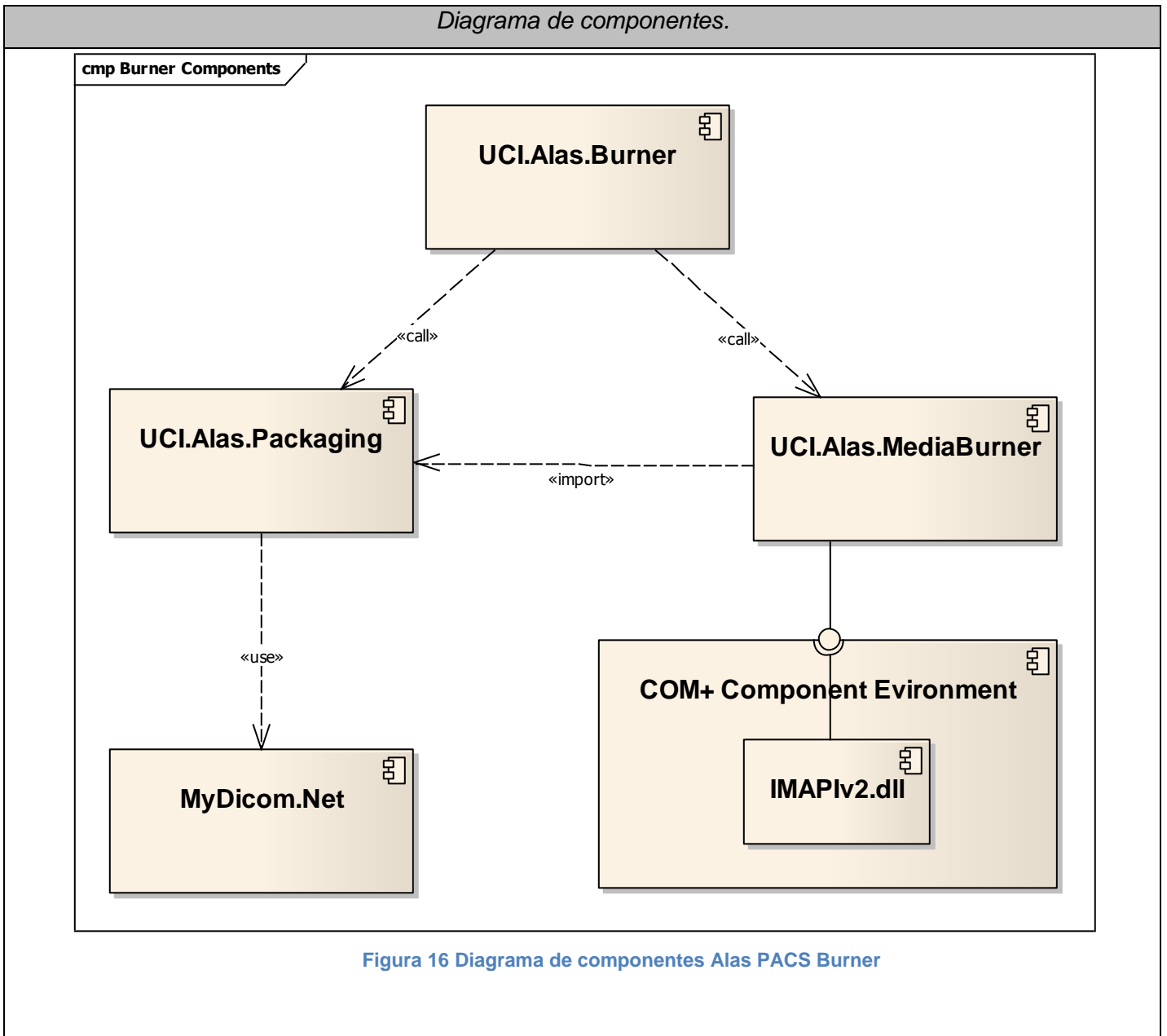
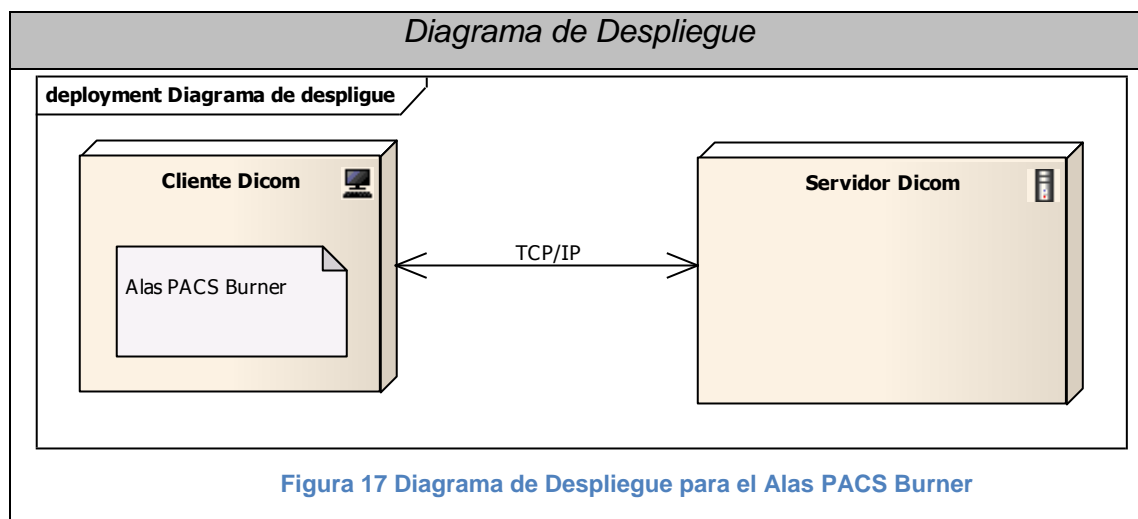


Figura 16 Diagrama de componentes Alas PACS Burner

### IV.3 Despliegue del Sistema

Como vehículo para el cumplimiento a los objetivos de este trabajo está el software elaborado. Esta aplicación debe de mantener una ubicación físico-espacial para su correcto funcionamiento. Una interpretación de esta ubicación real se realiza a través de un diagrama de despliegue.

Los diagramas de despliegues muestran la disposición física de los distintos nodos que entran a jugar un papel en la composición de un sistema y el reparto de los programas ejecutables sobre estos.



En este capítulo se realizaron los diagramas de componentes y de despliegue. Los mismos exponen el resultado de la implementación del software en cuestión. Estos diagramas ilustraron la arquitectura del sistema así como la interacción entre los distintos componentes. Esto ayuda a formar una idea virtual de la ubicación final del software.

### CONCLUSIONES

En la realización del trabajo de diploma se cumplió con el objetivo y las tareas propuestas. Para ello se explicaron las problemáticas existentes en las instituciones hospitalarias cubanas, con respecto al almacenamiento de las imágenes médicas en dispositivos externos. Se analizaron las tendencias actuales de los sistemas encargados de este tipo de almacenamiento en el ámbito nacional e internacional.

Como resultado de este trabajo se obtuvo un software que brinda las funcionalidades correspondientes a la organización y almacenamiento de las imágenes médicas en CD/DVD: **Alas PACS Burner**. Este software se ha logrado integrar con el sistema **Alas PACS**, desarrollado por el Grupo de Procesamiento de Imágenes y Señales. No imposibilitándole esto su funcionamiento de forma independiente.

**Alas PCAS Burner** fue implementado con el lenguaje de programación C# 2.0. Se utilizaron algunas herramientas complementarias y se pusieron en práctica distintos patrones de la programación orientada a objetos.

Los discos generados presentan una estructura acorde a lo planteado en el estándar DICOM 3.0. La presencia de un fichero DicomDir les permite ser accesibles por otras aplicaciones DICOM.

### RECOMENDACIONES

En **Alas PACS Burner** los autores han identificado una serie de aspectos, los que en futuras iteraciones podrían ser mejorados:

Para mayor variedad de dispositivos de almacenamientos se podrían implementar interfaces para otros medios de almacenamientos.

Se hace indispensable someter el software en cuestión a procesos de calidad, para empezar un despliegue del mismo.

Actualmente en el ámbito internacional existen determinadas tecnologías que permiten la serigrafía de los discos. Una de las más robustas es LightScribe, que es desarrollada por Hewlett Packard y consiste en la escritura en el anverso del disco con laser del dispositivo grabador. Sería de gran utilidad un estudio más profundo en este aspecto y evaluar su aplicación en una futura versión.

El software libre se ha convertido en una alternativa real a la dominación y trabas que impone el software propietario. El Grupo de Procesamiento de Imágenes se ha propuesto como política la migración de sus principales productos a alguna plataforma libre. Por lo que se hace necesario investigar y probar tecnologías de quemado en sistemas libres.

Otra cuestión significativa que vale la pena considerar es la implementación de un sistema complementario para la gestión de los discos grabados, lo cual aportaría un gran valor agregado a la solución presentada.

### Referencias Bibliográficas.

- Booch, Grady, James Rumbaugh, y Ivar Jacobson. *El lenguaje de modelado unificado*. Madrid: AddisonWesley Iberoamericana, 1999.
- Bryan, S. *The Cost and Benefits of Hospital-wide PACS networks: an overview of a comprehensive evaluation exercise*. 1999.
- ECMA. *ECMA 167 Universal Disk Format 3rd*. 1998.
- —. *Volume and File Structure of CDROM for Information Interchange*. 1987.
- Flower, Martín. *UML, gota a gota*. México: Addison Wesley, 1999.
- Huang, H.K. *PACS: Basic Principles and Applications*. New York: Wiley-Liss, 1999.
- Jacobson, Ivar. *Object-Oriented Software Engineering: A use case driven approach*. Addison Wesley, 1992.
- Kendall, Kendall &. *Análisis y Diseño de Sistemas*. México: Prentice Hall, 1997.
- Larman, Graig. *UML y Patrones. Introducción al análisis y diseño orientado a objetos*. México: Prentice May, 1999.
- Mesias, J. C. *Patris: Sistema Intra-Hospitalario de Imagenología y Telemedicina*. . 1998.
- National Electrical Manufacturers Association. *Digital Imaging and Communications in Medicine (DICOM) Part 1: Introduction and Overview*. Virginia 22209 USA: Rosslyn, 2007.
- NEMA. *DICOM Standard Part 1: Introduction and Overview*. Virginia, 2007.
- —. *DICOM Standard Part 12: Media Formats and Physical Media for Media Interchange*. Virginia, 2007.
- —. *DICOM Standard Part 11: Media Storage Application Profiles*. 2007.
- —. *DICOM Standard Part 3: Information Object Definitions*. 2007.
- Nobel Foundation. *The Nobel Prizes 1979*. Stockholm: Nobel Foundation, 1980.
- Rincón, Marcela, y Alejandra Rodríguez. *bioinstrumentacion.eia.edu.co*.  
[http://bioinstrumentacion.eia.edu.co/docs/signals/pacs\\_dicom.pdf](http://bioinstrumentacion.eia.edu.co/docs/signals/pacs_dicom.pdf).
- Sáez, Cristina. "Llega la ciberciudad." *Muy Interesante*, 2007.
- Stryker. 2007. <http://www.stryker.com/en-us/products/MedicalSoftwareWorkflowManagement/PACS/PACSSoftware/006025>.



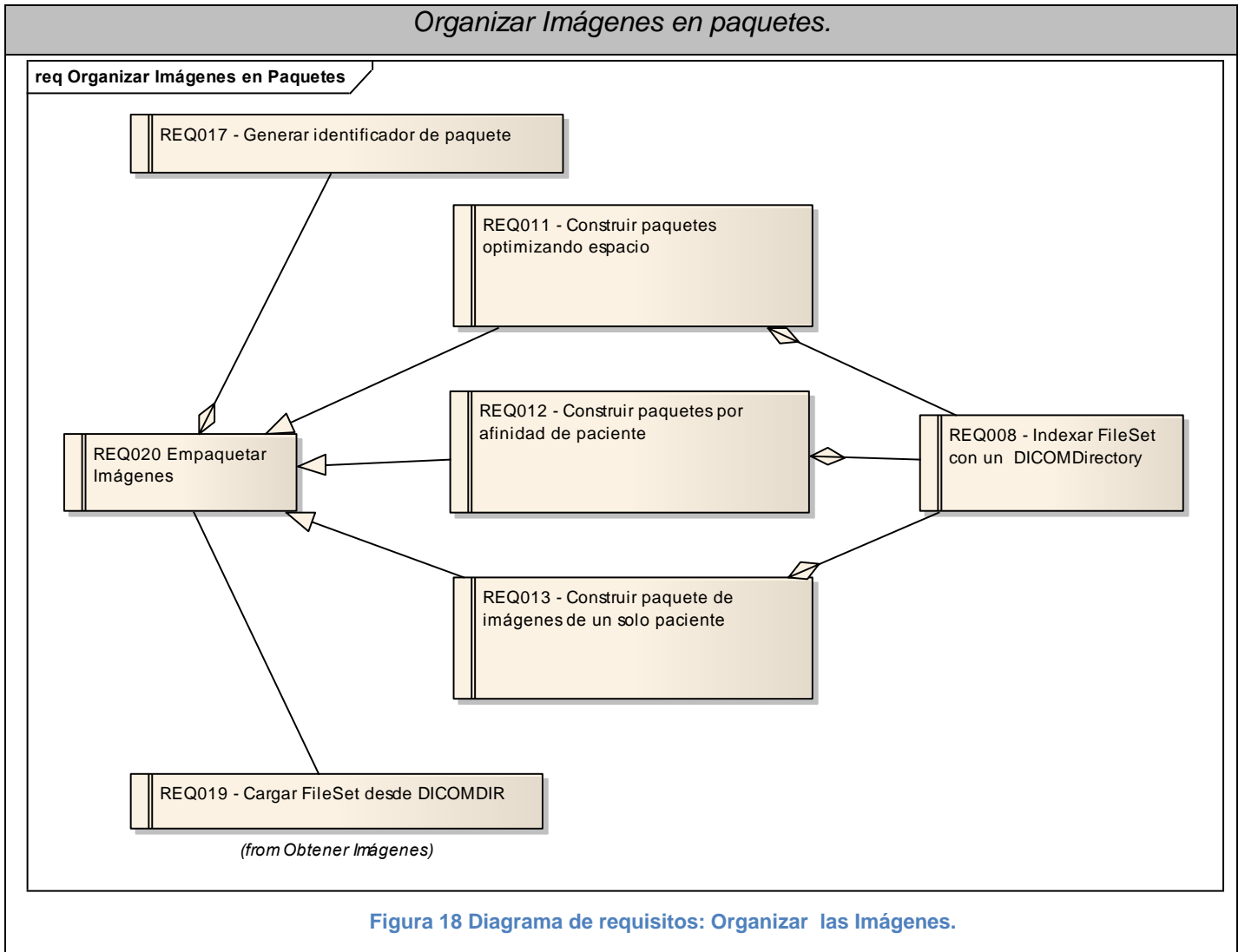
### BIBLIOGRAFÍA

1. Aho, Alfred V., John E. Hopcroft, and Jeffrey D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.
2. Booch, Grady, James Rumbaugh, and Ivar Jacobson. *El lenguaje de modelado unificado*. Madrid: AddisonWesley Iberoamericana, 1999.
3. Bryan, S. *The Cost and Benefits of Hospital-wide PACS networks: an overview of a comprehensive evaluation exercise*. 1999.
4. Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press y McGraw-Hill, 2001.
5. Codeproject. *The Code Project*. 1999. (accessed 2008).
6. ECMA. *ECMA 167 Universal Disk Format 3rd*. 1998.
7. —. *Volume and File Structure of CDROM for Information Interchange*. 1987.
8. Flaming, Bryan. *Practical Data Structures in C++*. John Wiley & Sons, 1993.
9. Flower, Martín. *UML, gota a gota*. México: Addison Wesley, 1999.
10. Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reuseable Object-Oriented Software*. Addison-Wesley, 1995.
11. Huang, H.K. *PACS: Basic Principles and Applications*. New York: wiley Liss, 1999.
12. Jacobson, Ivar. *Object-Oriented Software Engineering: A use case driven approach*. Addison Wesley, 1992.
13. Kendall, Kendall &. *Analisis y Diseño de Sistemas*. México: Prentice Hall, 1997.
14. Larman, Graig. *UML y Patronos. Introducción al análisis y diseño orientado a objetos*. México: Prentice May, 1999.
15. Leiss, Ernst L. *A Programmer's Companion to Algorithm Analysis*. Texas: Taylor & Francis Group, 2006.
16. Martin, Robert C., and Micah Martin. *Agile Principles, Patterns, and Practices in C#*. Massachusetts: Prentice Hall, 2006.
17. Mesias, J. C. *Patris: Sistema Intra-Hospitalario de Imageonología y Telemedicina*. . 1998.

18. Microsoft. *Microsoft Developer Network*. 2008. <http://msdn.microsoft.com> (accessed 2008).
19. National Electrical Manufacturers Association. *Digital Imaging and Communications in Medicine (DICOM) Part 1: Introduction and Overview*. Virginia 22209 USA: Rosslyn, 2007.
20. NEMA. *DICOM Standard Part 1: Introduction and Overview*. Virginia, 2007.
21. —. *DICOM Standard Part 10: Media Storage and File Format for Data Interchange*. 2007.
22. —. *DICOM Standard Part 11: Media Storage Application Profiles*. 2007.
23. —. *DICOM Standard Part 12: Media Formats and Physical Media for Media Interchange*. Virginia, 2007.
24. —. *DICOM Standard Part 3: Information Object Definitions*. 2007.
25. —. *DICOM Standard Part 5: Data Structure and Encoding*. 2007.
26. —. *DICOM Standard Part 6: Data Dictionary*. 2007.
27. Nobel Foundation. *The Nobel Prizes 1979*. Stockholm: Nobel Foundation, 1980.
28. Rincón, Marcela, and Alejandra Rodríguez. *bioinstrumentacion.eia.edu.co*.  
[http://bioinstrumentacion.eia.edu.co/docs/signals/pacs\\_dicom.pdf](http://bioinstrumentacion.eia.edu.co/docs/signals/pacs_dicom.pdf).
29. Sáez, Cristina. "Llega la ciberciudad." *Muy Interesante*, 2007.
30. Stryker. 2007. <http://www.stryker.com/en-us/products/MedicalSoftwareWorkflowManagement/PACS/PACSSoftware/006025>.

ANEXOS

Anexo 1 Diagrama de requisitos: Organizar las imágenes



## Anexo 2 Casos de Uso expandidos

## Crear DicomDir

<b>Caso de Uso:</b>	CU04- Crear DicomDir	
<b>Actores:</b>		
<b>Resumen:</b>	Abrirá las imágenes y tomará del metadato de la misma los campos relevantes. Creando con esto un DICOMDirectory que se incluirá en el paquete que se cree.	
<b>Referencia:</b>	REQ008	
<b>CU asociados:</b>	CU05	
<b>Precondiciones:</b>	La existencia de ficheros DICOM-Compatibles.	
<b>Flujo Normal de Eventos</b>		
<b>Acción del Actor</b>		<b>Respuesta del Sistema</b>
1. Especifica los ficheros que se utilizarán.	1.1 Saca los atributos de interés (de los pacientes, los estudios, las series y las imágenes) 1.2 Organiza la información. 1.3 Retorna el DicomDir.	
<b>Flujos Alternos</b>		
<b>Acción del Actor</b>		<b>Respuesta del Sistema</b>
<b>Poscondiciones:</b>	La creación de un fichero DicomDirectory.	
<b>Prioridad:</b>	Alta	
<b>Especificaciones Complementaria:</b>		

Tabla 11 Caso de uso expandido Crear DicomDir

## Empaquetar ficheros

<b>Caso de Uso:</b>	CU05- Empaquetar	
<b>Actores:</b>	Usuario	
<b>Resumen:</b>	Este caso de uso realizará la selección y organización de las imágenes que formarán parte de los paquetes. Esto se debe de regir por el tipo de empaquetado que el usuario haya definido.	
<b>Referencia:</b>	REQ011, REQ012, REQ013, REQ017, REQ020	
<b>CU asociados:</b>	CU04, CU06	
<b>Precondiciones:</b>	La selección y disponibilidad de los ficheros que se organizarán.	
<b>Flujo Normal de Eventos</b>		
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>	
1. El usuario solicita organizar las imágenes.	1.1	Muestra la interfaz del empaquetado.
2. El usuario ordena comenzar.	2.1	Organiza las imágenes según el criterio de organización.
	2.2	Crea DicomDir.
	2.3	Agrega el DicomDir
	2.4	Informa culminación;
<b>Flujos Alternos</b>		
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>	
2.a El usuario cambia el tipo de empaquetado.	2.1 a	Actualiza el tipo de empaquetado.
3.a El usuario ordena comenzar.	3.1 <sup>a</sup>	Organiza las imágenes según el criterio de organización.
		Crea DicomDir.
	3.2 <sup>a</sup>	Agrega el DicomDir
	3.3 <sup>a</sup>	Informa culminación;

<b>Poscondiciones:</b>	Un compilado, que contenga los ficheros que se quemarán.
<b>Prioridad:</b>	Alta
<b>Especificaciones Complementaria:</b>	

Tabla 12 Caso de uso expandido Empaquetar.

### Quemar Disco

<b>Caso de Uso:</b>	CU12- Quemar Disco
<b>Actores:</b>	Usuario
<b>Resumen:</b>	El proceso de almacenamiento, de los paquetes creados, en medios de almacenamiento. En este proceso se utiliza como medio de almacenamiento el CD y el DVD.
<b>Referencia:</b>	REQ005, REQ007, REQ018
<b>CU asociados:</b>	CU13
<b>Precondiciones:</b>	Existencia de una compilación de imágenes indexadas con un DICOMDIR y un medio de almacenamiento disponible para el quemado. En el caso del medio de almacenamiento debe de ser un CD o DVD. Además de todo esto la computadora donde se realiza el caso de uso debe poseer un quemador de CD o DVD según corresponda.
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario solicita el almacenamiento de los paquetes.	1.1 Se le muestra la interfaz gráfica correspondiente. 1.2 Se listan los paquetes.
2. Selecciona los paquetes que desee quemar.	2.1 Hace las <b>imágenes</b> <sup>25</sup> de los datos. 2.2 Verifica si están listo el o los quemadores. 2.3 Quema los discos.

<sup>25</sup> Imágenes de Datos, compilado de todas las imágenes que se guardarán.

	2.4 Muestra información.
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	2.2ª Se le informa al usuario la incapacidad de proseguir.
<b>Poscondiciones:</b>	Un CD o DVD quemado, con un DicomDirectory que referencie a la información que el mismo contenga.
<b>Prioridad:</b>	Alta.
<b>Especificaciones Complementaria:</b>	

Tabla 13 Caso de uso expandido Quemar Disco.

### Obtener Imágenes DICOM desde Servidor

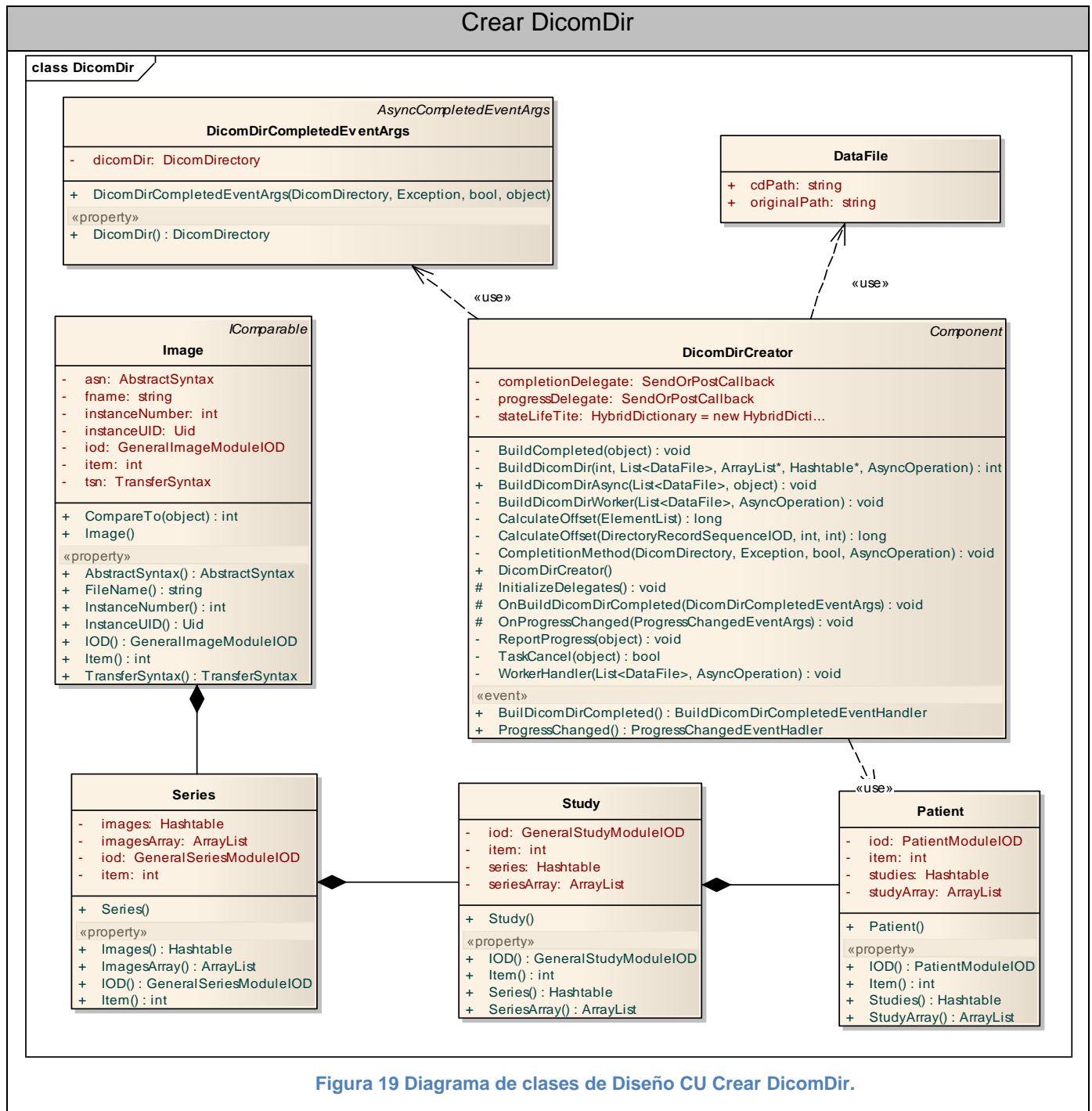
<b>Caso de Uso:</b>	CU10- Obtener Imágenes DICOM desde Servidor.
<b>Actores:</b>	Usuario
<b>Resumen:</b>	Luego de tener localizado los datos de las imágenes que se encuentran en el servidor, se procede a copiarlas hacia la máquina cliente. Esta operación se realiza con la implementación de un C-STORE.
<b>Referencia:</b>	REQ003, REQ004
<b>CU asociados:</b>	
<b>Precondiciones:</b>	Existencia de una conexión con un servidor DICOM-Compatible. Y la figuración en el mismo de las imágenes con los criterios solicitados.
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario solicita obtener imágenes de un servidor.	1.1 Muestra la interfaz visual correspondiente.
2. Selecciona las imágenes que se copiarán desde el servidor.	2.1 Copia las imágenes a la máquina local.

	2.2 Informa la finalización.
<b>Flujos Alternos</b>	
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>
<b>Poscondiciones:</b>	Tener imágenes copiadas desde un servidor de imágenes en la maquina local.
<b>Prioridad:</b>	Alta.
<b>Especificaciones Complementaria:</b>	

Tabla 14 Caso de uso expandido Obtener Imágenes desde Servidor.



Anexo 3 Diagramas de Clases de Diseño



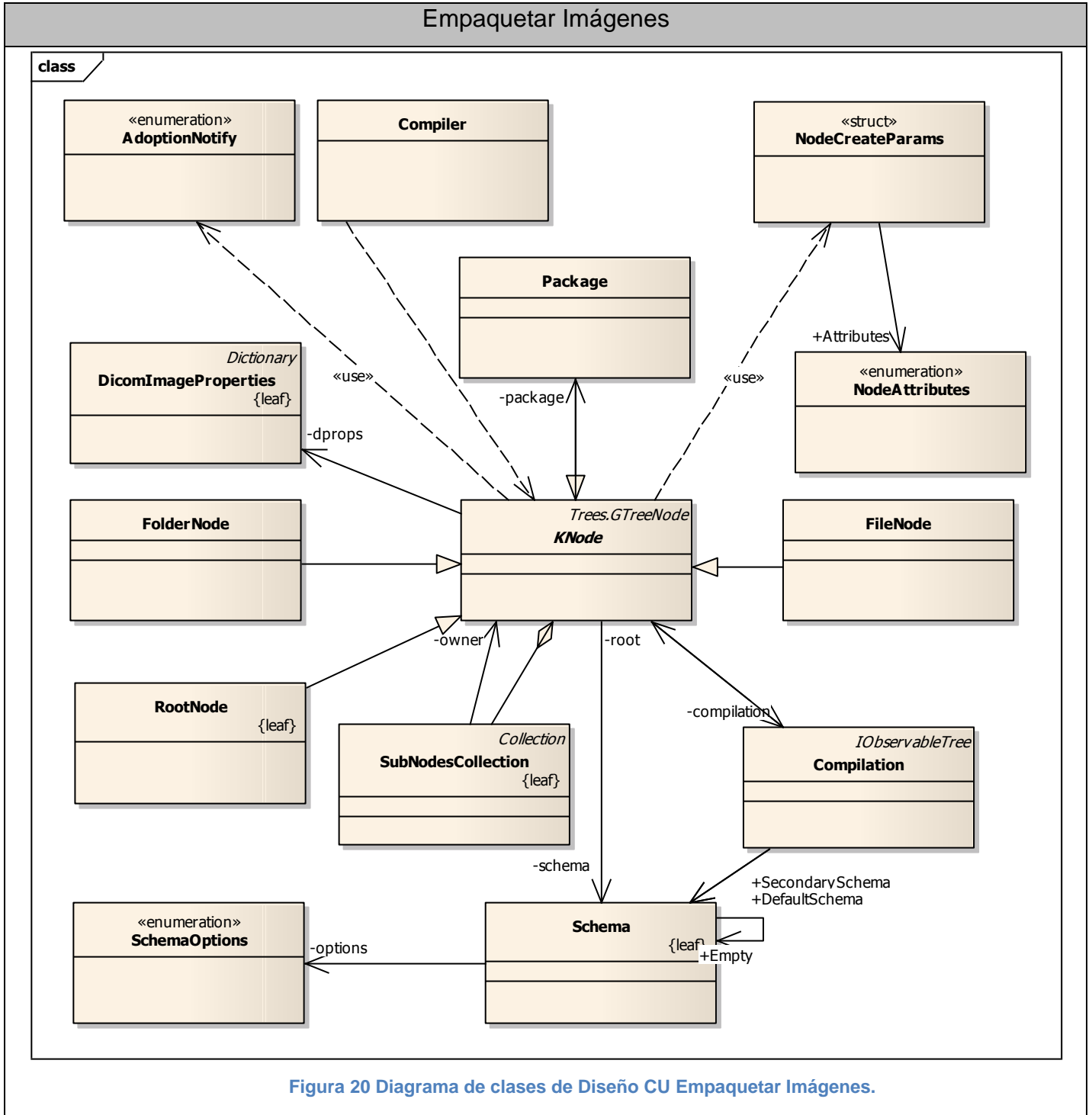
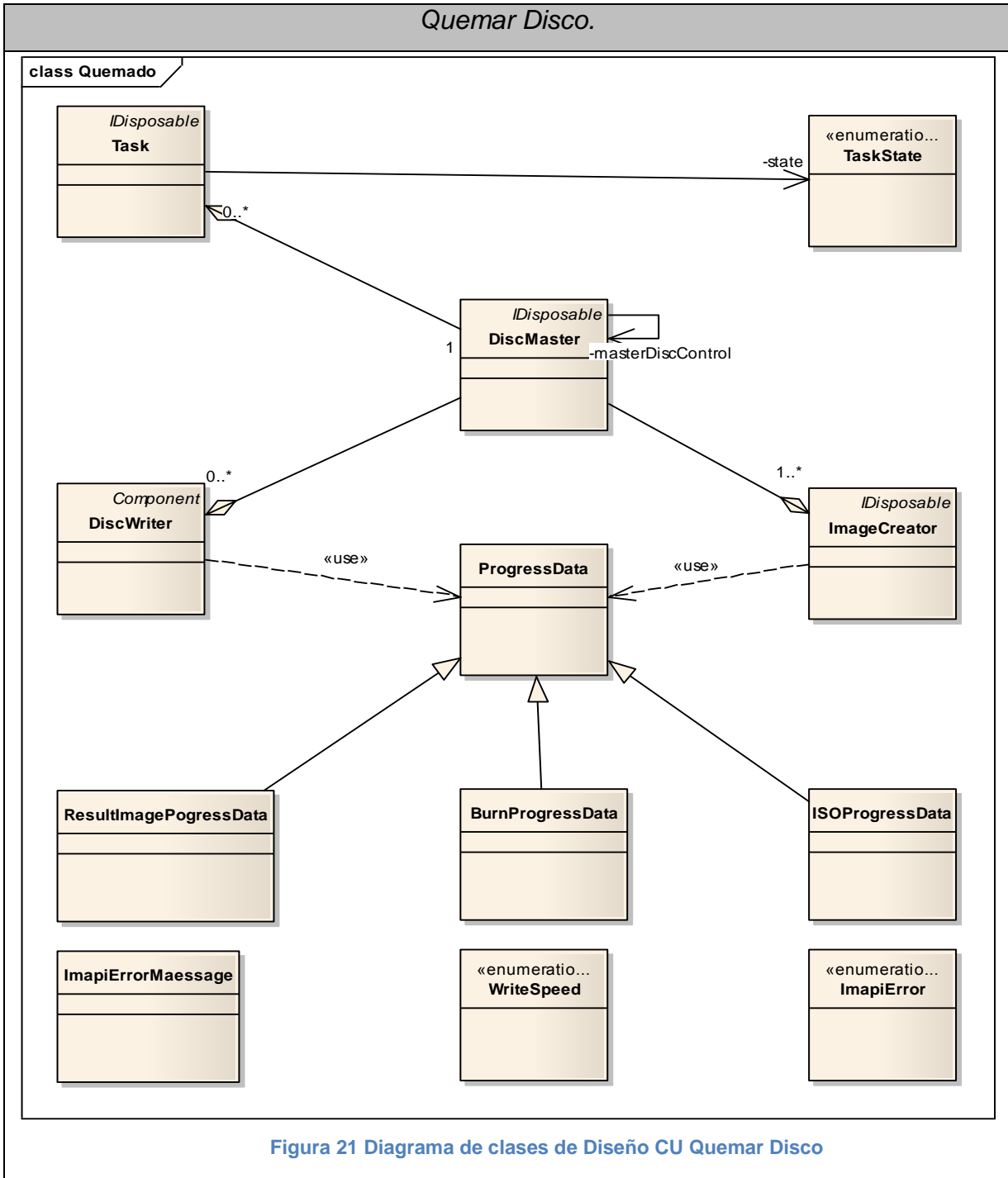


Figura 20 Diagrama de clases de Diseño CU Empaquetar Imágenes.



## Anexo 4 Descripción de Clases de Diseño

## Descripción de clases del CU Crear DicomDir

<b>Nombre:</b>	DataFile	
<b>Tipo de Clase:</b>	Entidad	
<b>Atributo</b>		<b>Tipo</b>
cdPath	string	
originalPath	string	
<b>Responsabilidades:</b>		
Nombre:		
Descripción		

Tabla 15 Descripción textual de la clase DataFile.

<b>Nombre:</b>	DicomDirCompletedEventArgs	
<b>Tipo de Clase:</b>	Entidad	
<b>Atributo</b>		<b>Tipo</b>
dicomDir	DicomDirectory	
<b>Responsabilidades:</b>		
Nombre:	DicomDir()	
Descripción	Obtiene o modifica el objeto DicomDir que se devolverá en el evento.	
Nombre:	DicomDirCompletedEventArgs(DicomDirectory dicomDir, Exception e, bool canceled, object state, )	
Descripción	Inicializa una instancia de la clase.	

Tabla 16 Descripción textual de la clase DicomDirCompletedEventArgs.

<b>Nombre:</b>	DicomDirCreator	
<b>Tipo de Clase:</b>	Controladora	
<b>Atributo</b>		<b>Tipo</b>
completionDelegate	SendOrPostCallback	

progressDelegate	SendOrPostCallback
stateLifeTite	HybridDictionary
<b>Responsabilidades:</b>	
Nombre:	BuildCompleted(object operationState, )
Descripción	Manejador del evento para el completado de un DicomDir.
Nombre:	BuildDicomDir(int skey, List<DataFile> files, ArrayList patientsArray, Hashtable patients, AsyncOperation asyncOp, )
Descripción	Obtiene los datos de interés de las imágenes, dichos datos son los que se incluirán en el Directorio DICOM.
Nombre:	BuildDicomDirAsync(List<DataFile> files, object dicomDirId, )
Descripción	Construye el DicomDir de manera asincrónica.
Nombre:	BuildDicomDirWorker(List<DataFile> files, AsyncOperation asyncOp, )
Descripción	Construye el objeto DicomDir.
Nombre:	CalculateOffset(ElementList el, )
Descripción	Calcula el tamaño del elemento.
Nombre:	CalculateOffset(DirectoryRecordSequenceIOD drs, int start, int end, )
Descripción	Calcula el tamaño de un elemento multievaluado.
Nombre:	CompletionMethod(DicomDirectory directory, Exception exception, bool canceled, AsyncOperation asyncOp, )
Descripción	Elimina la tarea y termina su ciclo de vida.
Nombre:	DicomDirCreator()
Descripción	Inicializa una instancia de la clase.
Nombre:	InitializeDelegates()
Descripción	Inicializa los delegados.
Nombre:	ReportProgress(object state, )
Descripción	Método para manejar el reporte del progreso en la creación de un DicomDir.
Nombre:	TaskCancel(object taskId, )
Descripción	Cancela la creación del DicomDir que se especifica.

Tabla 17 Descripción textual de la clase DicomDirCreator.

<b>Nombre:</b>	Patient	
<b>Tipo de Clase:</b>	Entidad	
Atributo		Tipo
iod	PatientModuleIOD	
item	int	
studies	Hashtable	
Responsabilidades:		
Nombre:	IOD()	
Descripción	Devuelve o modifica los atributos de interés del paciente.	
Nombre:	Item()	
Descripción	Devuelve o modifica el nivel del objeto.	
Nombre:	Patient()	
Descripción	Inicializa una instancia de la clase.	
Nombre:	Studies()	
Descripción	Devuelve o modifica el diccionario donde se verifica la no duplicidad de un estudio.	
Nombre:	StudyArray()	
Descripción	Devuelve o modifica el arreglo que contiene los estudios en el orden en que se agregaron.	

Tabla 18 Descripción textual de la clase Patient.

<b>Nombre:</b>	Study	
<b>Tipo de Clase:</b>	Entidad	
Atributo		Tipo
iod	GeneralStudyModuleIOD	
item	int	
series	Hashtable	
seriesArray	ArrayList	
Responsabilidades:		
Nombre:	IOD()	
Descripción	Devuelve o modifica los atributos de interés del estudio.	

Nombre:	Item()
Descripción	Devuelve o modifica el nivel del objeto.
Nombre:	Series()
Descripción	Devuelve o modifica el diccionario donde se verifica la no duplicidad de una serie.
Nombre:	SeriesArray()
Descripción	Devuelve o modifica el arreglo que contiene las series en el orden en que se agregaron.
Nombre:	Study()
Descripción	Inicializa una instancia de la clase.

Tabla 19 Descripción textual de la clase Study.

<b>Nombre:</b>	Series	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	images	Hashtable
	imagesArray	ArrayList
	iod	GeneralSeriesModuleIOD
	item	int
<b>Responsabilidades:</b>		
Nombre:	Images()	
Descripción	Devuelve o modifica el diccionario donde se verifica la no duplicidad de una imagen.	
Nombre:	ImagesArray()	
Descripción	Devuelve o modifica el arreglo que contiene las imágenes en el orden en que se agregaron.	
Nombre:	IOD()	
Descripción	Devuelve o modifica los atributos de interés de la serie.	
Nombre:	Item()	
Descripción	Devuelve o modifica el nivel del objeto.	
Nombre:	Series()	
Descripción	Inicializa una instancia de la clase.	

Tabla 20 Descripción textual de la clase Series

<b>Nombre:</b>	Image	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	asn	AbstractSyntax
	fname	string
	instanceNumber	int
	instanceUID	Uid
	iod	GeneralImageModuleIOD
	item	int
	tsn	TransferSyntax
<b>Responsabilidades:</b>		
Nombre:	AbstractSyntax()	
Descripción	Devuelve o modifica el Abstract Syntax usado en la imagen.	
Nombre:	FileName()	
Descripción	Obtiene o modifica el nombre completo de la imagen en el FileSet.	
Nombre:	Image()	
Descripción	Inicializa una instancia de la clase.	
Nombre:	InstanceNumber()	
Descripción	Obtiene o devuelve el número de instancia correspondiente a la imagen.	
Nombre:	InstanceUID()	
Descripción	Obtiene o devuelve el SOP instance UID usado en la imagen.	
Nombre:	IOD()	
Descripción	Devuelve o modifica los atributos de interés de la imagen.	
Nombre:	Item()	
Descripción	Devuelve o modifica el nivel del objeto.	
Nombre:	TransferSyntax()	
Descripción	Devuelve o modifica el Transfer Syntax usado en la imagen.	

Tabla 21 Descripción textual de la clase Image.



### Descripción de clases del CU Empaquetar Imágenes

<b>Nombre:</b>	Compilation	
<b>Tipo de Clase:</b>	Entidad	
<b>Atributo</b>		<b>Tipo</b>
AlternatePathSeparator		char
DefaultSchema		Schema
packsize		long
PathSeparator		char
root		KNode
SecondarySchema		Schema
<b>Responsabilidades:</b>		
Nombre:	Changed()	
Descripción	Evento. Ocurre cada vez que cambia la compilación. Por ejemplo: cuando un nodo es adicionado o eliminado.	
Nombre:	Clear()	
Descripción	Vacía la compilación eliminando todos los nodos excepto la raíz.	
Nombre:	Compilation()	
Descripción	Constructor. Inicializa una nueva instancia de la clase.	
Nombre:	DefaultPackageSize()	
Descripción	Obtiene o establece el tamaño que tendrán por defecto los nuevos paquetes.	
Nombre:	GetNode(string path, )	
Descripción	Obtiene un nodo dada su dirección. Devuelve el nodo si existe, o "null" en caso contrario.	
Nombre:	GetPackage(string name, )	
Descripción	Obtiene un paquete dado su nombre. Si no existe ningún paquete con ese nombre devuelve "null".	
Nombre:	OnChanged(TreeChangedEventArgs e, )	
Descripción	Ejecuta el evento "Changed".	
Nombre:	Packages()	
Descripción	Obtiene la lista de paquetes.	

Nombre:	RemoveNode(string path, )
Descripción	Excluye un nodo de la compilación.
Nombre:	Root()
Descripción	Obtiene el nodo raíz de la compilación.
Nombre:	Schema()
Descripción	Obtiene o establece el esquema usado para organizar la compilación.

Tabla 22 Descripción textual de la clase Compilation.

<b>Nombre:</b>	Compiler	
<b>Tipo de Clase:</b>	Controladora	
	<b>Atributo</b>	<b>Tipo</b>
<b>Responsabilidades:</b>		
Nombre:	AccommodateNode(KNode root, KNode node, )	
Descripción	Acomoda un nodo en la compilación creando la estructura jerárquica dictada por el esquema activo.	
Nombre:	CompareCandidates(KNode c1, KNode c2, )	
Descripción	Compara dos nodos para ver cuál es mejor para albergar otro nodo específico.	
Nombre:	Compile(string[] paths, KNode root, )	
Descripción	Adiciona a la compilación los archivos cuyas direcciones se especifican en el parámetro "paths".	
Nombre:	Compile(string filename, KNode root, )	
Descripción	Adiciona un archivo a la compilación.	
Nombre:	Compile(KNode root, KNode node, )	
Descripción	Organiza un nodo dentro de la compilación.	
Nombre:	CompileDirectory(string path, KNode root, )	
Descripción	Adiciona a la compilación todos los archivos DICOM encontrados en una carpeta y sus subcarpetas.	
Nombre:	DiscoverCandidates(KNode root, KNode node, )	
Descripción	Encuentra todos los nodos descendientes de "root" que pueden albergar a un nodo específico.	

Tabla 23 Descripción textual de la clase Compiler.

<b>Nombre:</b>	DicomImageProperties	
<b>Tipo de Clase:</b>	Entidad	
<b>Atributo</b>		<b>Tipo</b>
<b>Responsabilidades:</b>		
Nombre:	EnsureKey(DicomTag tag, )	
Descripción	Crea una llave en el diccionario en caso de que no exista.	
Nombre:	FromFile(string filename, )	
Descripción	Extrae las propiedades DICOM de un archivo.	
Nombre:	FromInfo(DicomFileInfo dinfo, )	
Descripción	Extrae las propiedades DICOM de un archivo.	

Tabla 24 Descripción textual de la clase DicomImageProperties.

<b>Nombre:</b>	FileNode	
<b>Tipo de Clase:</b>	Entidad	
<b>Atributo</b>		<b>Tipo</b>
<b>dinfo</b>		DicomFileInfo
<b>Responsabilidades:</b>		
Nombre:	Attributes()	
Descripción	Obtiene los atributos del nodo.	
Nombre:	FileNode(string name, )	
Descripción	Inicializa una nueva instancia de la clase con un nombre específico.	
Nombre:	FileNode()	
Descripción	Inicializa una nueva instancia de la clase.	
Nombre:	NewSubNode(NodeCreateParams args, )	
Descripción	Método sobrescrito. En esta clase solo lanza una excepción porque un nodo que representa un archivo no puede tener descendientes.	
Nombre:	Refresh()	
Descripción	Relee el archivo de origen y refresca el nodo con las nuevas propiedades.	

Nombre:	SchemaIndex()
Descripción	Obtiene o establece el índice del nodo en su esquema.
Nombre:	SourceFile()
Descripción	Obtiene o establece la dirección del archivo de origen del nodo.
Nombre:	Tag()
Descripción	Obtiene la etiqueta DICOM del nodo.
Nombre:	ValidateSubNode(KNode subnode, )
Descripción	Valida si un nodo puede ser insertado como hijo de este nodo.

Tabla 25 Descripción textual de la clase FileNode.

<b>Nombre:</b>	FolderNode	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	<b>sindex</b>	int
<b>Responsabilidades:</b>		
Nombre:	Attributes()	
Descripción	Obtiene los atributos del nodo.	
Nombre:	FolderNode(string name, )	
Descripción	Inicializa una nueva instancia de la clase con un nombre específico.	
Nombre:	FolderNode()	
Descripción	Inicializa una nueva instancia de la clase.	
Nombre:	NewSubNode(NodeCreateParams args, )	
Descripción	Crea un nuevo nodo hijo.	
Nombre:	SchemaIndex()	
Descripción	Obtiene o establece el índice del nodo en su esquema.	
Nombre:	Tag()	
Descripción	Obtiene la etiqueta DICOM del nodo.	

Tabla 26 Descripción textual de la clase FolderNode.

<b>Nombre:</b>	KNode	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	compilation	Compilation
	dprops	DicomImageProperties
	name	string
	nodeNameComparison	StringComparison
	package	Package
	schema	Schema
	size	long
<b>Responsabilidades:</b>		
Nombre:	Attributes()	
Descripción	Obtiene los atributos del nodo.	
Nombre:	AutoSuggestName()	
Descripción	Sugiere un nombre automático para el nodo.	
Nombre:	AutoSuggestName(KNode adopter, )	
Descripción	Sugiere un nombre para el nodo como hijo de otro nodo.	
Nombre:	Compilation()	
Descripción	Obtiene la compilación a la que pertenece el nodo.	
Nombre:	Contains(string subname, )	
Descripción	Averigua si el nodo tiene algún hijo con un nombre específico.	
Nombre:	CreateSubNodesCollection()	
Descripción	Crea una lista para almacenar los nodos hijos.	
Nombre:	DicomProperties()	
Descripción	Obtiene la colección de propiedades DICOM asociadas al nodo.	
Nombre:	FormatFileSize(long size, )	
Descripción	Obtiene una cadena de caracteres que representa el tamaño del nodo.	
Nombre:	GenerateFilename(NodeCreateParams cparams, )	

Descripción	Genera un nombre de archivo para un nodo.
Nombre:	GenerateFolderName(NodeCreateParams cparams, )
Descripción	Genera un nombre de carpeta para un nodo.
Nombre:	GetPackage()
Descripción	Obtiene el paquete en el cual el nodo se encuentra.
Nombre:	GetSubNode(string subname, )
Descripción	Obtiene un nodo hijo dado su nombre.
Nombre:	InvalidatePackage()
Descripción	Borra la referencia del nodo a su paquete.
Nombre:	IsContainer()
Descripción	Indica si el nodo puede o no contener otros nodos.
Nombre:	IsFile()
Descripción	Indica si el nodo representa un archivo.
Nombre:	KNode(string name, )
Descripción	Inicializa una nueva instancia de la clase con un nombre específico.
Nombre:	KNode(Compilation compilation, string name, )
Descripción	Inicializa una nueva instancia de la clase dentro de una compilación.
Nombre:	Name()
Descripción	Obtiene o establece el nombre del nodo.
Nombre:	NameGenerator(string format, NameGenOptions options, object[] values, )
Descripción	Genera nombres para los nodos.
Nombre:	NewSubNode(NodeCreateParams args, )
Descripción	Crea un nuevo nodo hijo.
Nombre:	OnSubNodesChanged(CollectionChange change, KNode source, )
Descripción	Notifica a la compilación sobre un cambio en la estructura. Llamado cada vez que la colección de sub-nodos cambia.
Nombre:	Package()
Descripción	Obtiene el paquete al cual pertenece el nodo.
Nombre:	Parent()
Descripción	Obtiene el nodo padre.

Nombre:	Path()
Descripción	Obtiene la dirección del nodo en la compilación.
Nombre:	RaiseCompilationChanged(TreeChangedEventArgs e, )
Descripción	Enruta un evento de cambio hacia la compilación.
Nombre:	Remove()
Descripción	Elimina el nodo.
Nombre:	Schema()
Descripción	Obtiene o establece el esquema usado para organizar el nodo.
Nombre:	SchemaIndex()
Descripción	Obtiene o establece el índice de la principal propiedad del nodo en el esquema.
Nombre:	SetParent(ITreeNode adopter, )
Descripción	Establece el nodo padre.
Nombre:	SetParent(KNode adopter, AdoptionNotify notify, )
Descripción	Establece el nodo padre y notifica sobre el cambio.
Nombre:	Size()
Descripción	Obtiene el tamaño del nodo (en bytes).
Nombre:	SourceFile()
Descripción	Obtiene o establece el camino del origen de datos del nodo.
Nombre:	Tag()
Descripción	Obtiene la etiqueta DICOM del nodo.
Nombre:	TagValue()
Descripción	Obtiene el valor asociado a la etiqueta del nodo.
Nombre:	UpdateSize(long delta, )
Descripción	Actualiza el tamaño del nodo.
Nombre:	ValidateName(string value, )
Descripción	Valida el nombre del nodo.
Nombre:	ValidateParent(KNode parent, )
Descripción	Valida el nodo padre.
Nombre:	ValidateSubNode(KNode subnode, )

Descripción	Valida un nodo hijo.
-------------	----------------------

Tabla 27 Descripción textual de la clase KNode.

<b>Nombre:</b>	NodeAttributes	
<b>Tipo de Clase:</b>	Entidad (Tipo Enumerativo)	
	<b>Atributo</b>	<b>Descripción</b>
	None	Especifica ningún atributo.
	RegularContainer	El nodo contiene archivos y carpetas.
	File	El nodo representa un archivo.
	ContainerParent	Indica que el nodo contiene otros nodos contenedores.
	FileParent	Indica que el nodo puede contener archivos.
	Removable	Indica que el nodo se puede eliminar.

Tabla 28 Descripción textual de la clase NodeAttributes.

<b>Nombre:</b>	Package	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	maxsize	long
<b>Responsabilidades:</b>		
Nombre:	Attributes()	
Descripción	Obtiene los atributos del nodo.	
Nombre:	GetPackage()	
Descripción	Obtiene el paquete al cual pertenece el nodo.	
Nombre:	MaxSize()	
Descripción	Obtiene o establece el tamaño máximo de un paquete.	
Nombre:	NewSubNode(NodeCreateParams args, )	
Descripción	Crea un nuevo nodo hijo.	
Nombre:	Package(string name, )	



Descripción	Inicializa una nueva instancia de la clase con un nombre específico.
Nombre:	Package(string name, long maxsize, )
Descripción	Inicializa una nueva instancia de la clase con un nombre y un tamaño máximo.
Nombre:	Package()
Descripción	Inicializa una nueva instancia de la clase.
Nombre:	SchemaIndex()
Descripción	Obtiene o establece el índice del nodo en su esquema.
Nombre:	Tag()
Descripción	Obtiene la etiqueta DICOM asociada al nodo.
Nombre:	ValidateParent(KNode parent, )
Descripción	Valida el nodo padre.

Tabla 29 Descripción textual de la clase Package.

<b>Nombre:</b>	RootNode	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
<b>Responsabilidades:</b>		
Nombre:	Attributes()	
Descripción	Obtiene los atributos del nodo.	
Nombre:	GetPackage()	
Descripción	Obtiene el paquete al cual el nodo pertenece.	
Nombre:	NewSubNode(NodeCreateParams args, )	
Descripción	Crea un nuevo nodo hijo.	
Nombre:	RootNode(Compilation owner, )	
Descripción	Inicializa una nueva instancia de la clase en una compilación.	
Nombre:	SchemaIndex()	
Descripción	Obtiene o establece el índice del nodo en su esquema.	
Nombre:	Tag()	

Descripción	Obtiene la etiqueta DICOM asociada el nodo.
Nombre:	ValidateName(string name, )
Descripción	Valida el nombre del nodo.
Nombre:	ValidateParent(KNode parent, )
Descripción	Valida el nodo padre.
Nombre:	ValidateSubNode(KNode subnode, )
Descripción	Valida un nodo hijo.

Tabla 30 Descripción textual de la clase RootNode.

<b>Nombre:</b>	Schema	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	Empty	Schema
	MaxLength	int
	options	SchemaOptions
	tags	DicomTag
<b>Responsabilidades:</b>		
Nombre:	Equals(object other, )	
Descripción	Determina si el esquema equivale a otro objeto especificado.	
Nombre:	GetHashCode()	
Descripción	Calcula un código hash para el esquema.	
Nombre:	IsEmpty()	
Descripción	Indica si el esquema está vacío.	
Nombre:	Length()	
Descripción	Obtiene la longitud del esquema.	
Nombre:	Options()	
Descripción	Obtiene las opciones de compilación del esquema.	
Nombre:	Schema(DicomTag[] tags, SchemaOptions options, )	
Descripción	Inicializa una nueva instancia de la clase.	

Nombre:	Schema()
Descripción	Inicializa una nueva instancia de la clase.
Nombre:	this(int index, )
Descripción	Obtiene una etiqueta del esquema.

Tabla 31 Descripción textual de la clase Schema.

<b>Nombre:</b>	SubNodesCollection	
<b>Tipo de Clase:</b>	Entidad	
<b>Atributo</b>		<b>Tipo</b>
Attribute		Type
<b>Responsabilidades:</b>		
Nombre:	ClearItems()	
Descripción	Borra el contenido de la colección.	
Nombre:	InsertItem(int index, KNode node, )	
Descripción	Inserta un nodo a la colección.	
Nombre:	RemoveItem(int index, )	
Descripción	Elimina un nodo de la colección.	
Nombre:	SubNodesCollection(KNode owner, )	
Descripción	Inicializa una nueva instancia de la clase como parte de un nodo específico.	

Tabla 32 Descripción textual de la clase SubNodesCollection.

## Descripción de clases del CU Quemar Disco

<b>Nombre:</b>	ProgressData	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	history	Dictionary<string, string>
	progressPercent	int
<b>Responsabilidades:</b>		
Nombre:	ProgressData()	
Descripción	Inicializa una instancia de la clase.	
Nombre:	ProgressPercentage()	
Descripción	Obtiene o establece el porcentaje de la tarea.	

Tabla 33 Descripción textual de la clase ProgressData.

<b>Nombre:</b>	ResultImageProgressData	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	lastAddedFile	string
	taskID	string
<b>Responsabilidades:</b>		
Nombre:	LastAddedFile()	
Descripción	Obtiene o modifica la dirección del último archivo que se adicionó a la imagen.	
Nombre:	ResultImagePogressData()	
Descripción	Inicializa una entidad de la clase.	
Nombre:	TaskID()	
Descripción	Obtiene o establece el identificador de la tarea que se está ejecutando.	

Tabla 34 Descripción textual de la clase ResultImageProgress.

<b>Nombre:</b>	ISOProgressData	
<b>Tipo de Clase:</b>	Entidad	
Atributo		Tipo
copiedBytes		long
mediaType		MediaPhysicalType
outPutFile		string
taskID		string
totalSize		long
Responsabilidades:		
Nombre:	ISOProgressData()	
Descripción	Inicializa una instancia de la clase.	
Nombre:	OutPutFile()	
Descripción	Obtiene o modifica la dirección del fichero donde se salva el ISO	
Nombre:	TaskID()	
Descripción	Obtiene o establece el identificador de la tarea que se está ejecutando.	

Tabla 35 Descripción textual de la clase ISOProgressData.

<b>Nombre:</b>	BurnProgressData	
<b>Tipo de Clase:</b>	Entidad	
Atributo		Tipo
driveName		string
elapsedTime		long
freeSystemBuffer		long
lastReadLba		long
lastWrittenLba		long
recorderID		string
remainingTime		long
sectorCount		long
startLba		long

statusMessage	string
totalSystemBuffer	long
totalTime	long
uniqueRecorderId	string
usedSystemBuffer	long
volumenLabel	string
<b>Responsabilidades:</b>	
Nombre:	BurnProgressData()
Descripción	Inicializa una instancia de la clase.
Nombre:	BurnProgressData(string recorderID, )
Descripción	Inicializa una instancia de la clase. Y establece un identificador para el quemador.
Nombre:	DriveName()
Descripción	Obtiene o establece el nombre del quemador donde se esté llevando a cabo el quemado.
Nombre:	FreeSystemBuffer()
Descripción	Obtiene o establece el tamaño de la parte del buffer que está libre.
Nombre:	RecorderID()
Descripción	Obtiene o establece un identificador único para el quemador.
Nombre:	StatusMessage()
Descripción	Obtiene o establece un mensaje de estado para la tarea en cuestión.
Nombre:	TotalSystemBuffer()
Descripción	Obtiene o establece el tamaño total de buffer del sistema.
Nombre:	UniqueRecorderId()
Descripción	Obtiene o establece el identificador para quemador en cuestión.
Nombre:	UsedSystemBuffer()
Descripción	Obtiene o establece el tamaño de la parte del buffer que está siendo usado.
Nombre:	VolumenLabel()
Descripción	Obtiene o establece la etiqueta del disco que se está quemando.

Tabla 36 Descripción textual de la BurnProgressData.

<b>Nombre:</b>	DiscMaster	
<b>Tipo de Clase:</b>	Controladora	
	<b>Atributo</b>	<b>Tipo</b>
	busyRecorders	Dictionary<string, DiscWriter>
	completedTask	Dictionary<string, Task>
	DDIRCreator	DicomDirCreator
	delayBurnTask	Dictionary<string, Task>
	delayISOTask	List<Task>
	discMaster	MsftDiscMaster2
	freeRecorders	Dictionary<string, DiscWriter>
	imageCreators	List<ImageCreator>
	IsoCreator	ImageCreator
	IsofileSystem	FsiFileSystems
	IsoMediaType	MediaPhysicalType
	IsoPath	string
	masterDiscControl	DiscMaster
	processingTask	Dictionary<string, Task>
	workPath	string
<b>Responsabilidades:</b>		
Nombre:	AddDiscRecorder(MsftDiscRecorder2 recorder, )	
Descripción	Añade un nuevo Quemador a los que posee la clase.	
Nombre:	burnCreateImageCompleted(object sender, System.ComponentModel.RunWorkerCompletedEventArgs e, )	
Descripción	Captura y manipula el evento lanzado ante la culminación de la imagen que se quemará.	
Nombre:	BurnData(KNode packet, )	
Descripción	Agrega el paquete que será quemado a una cola de paquetes	
Nombre:	BurnTaskInit()	
Descripción	Verifica si hay algún quemador disponible, y asigna al mismo la tarea pendiente.	
Nombre:	CheckForRecorder(long packageSize, )	

Descripción	Busca dentro de los quemadores disponibles uno que este listo
Nombre:	CreateISO(KNode packet, )
Descripción	Crea un ISO a partir del contenido del Paquete.
Nombre:	DDIRCreator_BuildDicomDirCompleted(object sender, DicomDirCompletedEventArgs e, )
Descripción	Captura y manipula el evento de Terminación del DicomDir.
Nombre:	DiscMaster()
Descripción	Inicializa una instancia de la clase.
Nombre:	Dispose()
Descripción	Libera los recursos usados por la clase.
Nombre:	dw_OnBurnDiscCompleted(object sender, System.ComponentModel.AsyncCompletedEventArgs e, )
Descripción	Captura y manipula el evento lanzado ante la culminación del proceso de quemado de alguna tarea.
Nombre:	dw_ProgressChanged(object sender, ProgressChangedEventArgs e, )
Descripción	Captura y manipula el evento de progreso que ocurra en quemado de algún paquete.
Nombre:	GetAllTasks()
Descripción	Devuelve todas las tareas que se encuentren en proceso.
Nombre:	GetTask(string taskID, )
Descripción	Devuelve la tarea que se solicita.
Nombre:	imgCreator_progressChanged(object sender, System.ComponentModel.ProgressChangedEventArgs e, )
Descripción	Captura y manipula el evento de progreso que ocurra en la creación de un ISO.
Nombre:	ISOCreationProgressChanged()
Descripción	Manipula el evento para reportar el progreso.
Nombre:	IsoCreator_createlsoCompleted(object sender, System.ComponentModel.RunWorkerCompletedEventArgs e, )
Descripción	Captura y manipula el evento de creación de un ISO terminada.
Nombre:	ISOFileSystem()
Descripción	Obtiene o establece el Sistema de Archivos de la Imagen que se construirá.
Nombre:	IsoFolderPaht()
Descripción	Obtiene o establece la dirección donde se va a guardar los ISOs que



	se creen.
Nombre:	ISOMediaPhysicalType()
Descripción	Obtiene o establece el tipo físico de los ISOs que se crearán.
Nombre:	IsoTaskInit()
Descripción	Verifica si no hay un ISO creándose e inicia otro si hay alguna tarea pendiente.
Nombre:	MasterControl()
Descripción	Devuelve la instancia de la clase.
Nombre:	Recorders()
Descripción	Devuelve una lista con todos los quemadores que posea la computadora.
Nombre:	WorkingPath()
Descripción	Obtiene o establece la dirección de la carpeta de trabajo.

Tabla 37 Descripción textual de la clase DiscMaster.

<b>Nombre:</b>	DiscWriter	
<b>Tipo de Clase:</b>	Controladora	
	<b>Atributo</b>	<b>Tipo</b>
	AsyncOperation	AsyncOperation
	burnWorkerDelegate	BurnWorkerEventHandler
	canceled	bool
	clientName	string
	completionDelegate	SendOrPostCallback
	DiscFormat	MsftDiscFormat2Data
	exception	Exception
	image	IStream
	imageCreator	ImageCreator
	isBusy	bool
	lastPercent	int
	progressDelegate	SendOrPostCallback
	recorder	MsftDiscRecorder2

taskID	string
<b>Responsabilidades:</b>	
Nombre:	BurnCompleted(object state, )
Descripción	Lanza el evento que notifica que el quemado del disco ha finalizado.
Nombre:	BurnDisc(KNode packet, string volumenID, )
Descripción	Quema sincrónicamente el contenido de la imagen en el disco que se encuentre en el quemador.
Nombre:	BurnDiscAsync(IStream image, string volumenID, )
Descripción	Quema asincrónicamente el contenido de la imagen en el disco que se encuentre en el quemador.
Nombre:	BurnWorker()
Descripción	Ejecuta el quemado de forma asincrónica.
Nombre:	CancelAsync()
Descripción	Cancela, si existe, la tarea que se esté ejecutando.
Nombre:	ClientName()
Descripción	Obtiene o establece el nombre de la aplicación que hace uso de la clase.
Nombre:	CloseTray()
Descripción	Cierra la bandeja del quemador que esté definido, si el mismo lo permite.
Nombre:	DiscFreeSize()
Descripción	Retorna la cantidad de bytes libres que hay disponibles en el disco.
Nombre:	DiscWriter()
Descripción	Inicializa una nueva instancia de la clase.
Nombre:	Dispose(bool disposing, )
Descripción	Libera los recursos usados por la clase.
Nombre:	EjectTray()
Descripción	Expulsa la bandeja del quemador que esté definido.
Nombre:	ImageToWrite()
Descripción	Obtiene o establece el flujo de datos de la imagen que se grabará.
Nombre:	Init()
Descripción	Inicializa los datos de la clase, dejándola lista para ejecutar otra tarea.

Nombre:	InitializeDelegates()
Descripción	Inicializa los delegados relacionados con la ejecución del quemado asincrónico.
Nombre:	IsReady()
Descripción	Obtiene si el dispositivo está listo para proceder al quemado, esto incluye un disco en el quemador.
Nombre:	Recorder()
Descripción	Obtiene o establece la clase donde que representa al quemador.
Nombre:	ReportProgress(object ee, )
Descripción	Lanza el evento que notifica un progreso en el quemado del disco.
Nombre:	TaskID()
Descripción	Obtiene o establece el identificador del paquete que se está procesando.

Tabla 38 Descripción textual de la clase DiscWriter.

<b>Nombre:</b>	ImageCreator	
<b>Tipo de Clase:</b>	Controladora	
	<b>Atributo</b>	<b>Tipo</b>
	actualSize	long
	backWorker	BackgroundWorker
	cancel	bool
	dicomDirPath	string
	fileSystem	FsiFileSystems
	fsResult	IFileSystemImageResult
	image	MsftFileSystemImage
	isBackgroundWork	bool
	isBusy	bool
	IsoDoWorkEventHandler	DoWorkEventHandler
	IsoRunWorkerCompletedEventHandler	RunWorkerCompletedEventHandler
	lastPercent	int
	mediaType	MediaPhysicalType

outputFileName	string
packet	KNode
ResultImageCompletedEH	RunWorkerCompletedEventHandler
ResultImageDoWorkEH	DoWorkEventHandler
STGM_SHARE_DENY_WRITE	uint
totalSize	long
volumeName	string
workingDirectory	string
<b>Responsabilidades:</b>	
Nombre:	AddFilesToImage(KNode packet, )
Descripción	Recorre y adiciona recursivamente los ficheros que contenga un paquete a la imagen que se está construyendo.
Nombre:	CreateFSIFile(string filePath, string filePathOnCd, IFsiDirectoryItem root, )
Descripción	Crea un elemento tipo fichero en el sistema de archivos que se está construyendo.
Nombre:	CreateISOFileAsync(KNode packet, )
Descripción	Crea el ISO, de forma asincrónica, para el paquete en cuestión.
Nombre:	CreateProgressISOFile(System.Runtime.InteropServices.ComTypes.IStream imagestream, )
Descripción	Salva para el disco los datos del flujo en un ISO.
Nombre:	CreateResultImage(KNode packet, )
Descripción	Crea la imagen resultante, de forma sincrónica, para el paquete en cuestión.
Nombre:	CreateResultImageAsync(KNode packet, )
Descripción	Crea la imagen resultante, de forma asincrónica, para el paquete en cuestión.
Nombre:	Dispose()
Descripción	Libera los recursos usados por la clase.
Nombre:	FileSystem()
Descripción	Obtiene o modifica el sistema de archivos con que se va a crear la imagen de datos.
Nombre:	ImageCreator()
Descripción	Inicializa una instancia de la clase ImageCreator.

Nombre:	IsBusy()
Descripción	Devuelve el estado de la clase.
Nombre:	ISOOutputFileName()
Descripción	Obtiene o modifica el nombre del fichero donde se va a almacenar la imagen de datos en el disco.
Nombre:	MediaType()
Descripción	Obtiene o modifica el tipo de medio para el que se va a crear la imagen de datos.
Nombre:	Packet()
Descripción	Obtiene o modifica el paquete con el que se va a trabajar.
Nombre:	VolumeName()
Descripción	Obtiene o modifica la etiqueta que se le va a poner a la imagen.
Nombre:	WorkingDirectory()
Descripción	Obtiene o modifica la dirección de la carpeta de trabajo.

Tabla 39 Descripción textual de la clase ImageCreator.

<b>Nombre:</b>	Task	
<b>Tipo de Clase:</b>	Entidad	
	<b>Atributo</b>	<b>Tipo</b>
	dicomDirPath	string
	history	Dictionary<string, string>
	info	object
	pack	KNode
	state	TaskState
<b>Responsabilidades:</b>		
Nombre:	DicomDirPath()	
Descripción	Obtiene o devuelve la dirección del fichero DicomDir.	
Nombre:	Dispose()	
Descripción	Libera los recursos usados.	
Nombre:	History()	

Descripción	Obtiene o devuelve el registro histórico de la tarea.
Nombre:	Packet()
Descripción	Obtiene o modifica el paquete que define la tarea.
Nombre:	State()
Descripción	Obtiene o modifica el estado de la tarea.
Nombre:	StateInfo()
Descripción	Obtiene o modifica la información sobre el estado de la tarea.
Nombre:	Task()
Descripción	Inicializa una instancia de la clase.
Nombre:	TaskIID()
Descripción	Obtiene o devuelve un identificador para la tarea.

Tabla 40 Descripción textual de la clase Task.

## GLOSARIO

**ACR:** Asociación de Radiólogos Americanos (American College of Radiology). Organización principal de radiólogos, oncólogos y físicos de la medicina, que agrupa a más de 30000 miembros.

**Backup:** anglicismo que se refiere a la copia de datos de tal forma que estas copias adicionales puedan restaurar un sistema después de una pérdida de información. En el contexto de este trabajo se usa para hacer referencia al archivado de imágenes con fines de liberación de espacio.

**byte:** Una secuencia contigua de ocho bits en una computadora binaria que comprende el sub-campo direccionable más pequeño del tamaño de palabra natural (word) de la computadora.

**CASE:** Sigla que corresponde a las iniciales de Computer Aided Software Engineering; cuya traducción significa Ingeniería de Software Asistida por Computación. Se define como la aplicación de las tecnologías informáticas para la automatización de procesos. Algunos procesos que se pueden automatizar están: el desarrollo de software, la documentación, la generación de código, el chequeo de errores, entre otros. El uso de las mismas permite la reutilización del software y la estandarización de la documentación.

**CD:** El disco compacto (conocido popularmente como CD, por la sigla en inglés de Compact Disc) es un soporte digital óptico utilizado para almacenar cualquier tipo de información (audio, vídeo, documentos y otros datos).

**DICOM:** (Digital Imaging and Communications in Medicine) es el estándar reconocido mundialmente para el intercambio de imágenes médicas, pensado para el manejo, almacenamiento, impresión y transmisión de imágenes médicas.

**DVD:** Es un formato de almacenamiento óptico que puede usarse para guardar datos. Su nombre proviene de la sigla de Digital Versatile Disc o Disco Versátil Digital. Se asemeja a los discos compactos (CD) en cuanto a sus dimensiones físicas, pero está codificado en un formato distinto y a una densidad mucho mayor.

**Fichero DicomDir:** Es un fichero único y obligatorio que se encuentra en un **File-Set** que contiene información sobre la organización y contenido del File-Set.

**FileID:** Los ficheros son identificados con un identificador que es único para el contexto del **File-Set** donde pertenezca el fichero. Para un mismo fichero, un conjunto ordenado de componentes (una cadena de uno hasta ocho caracteres) conforman el **File ID** (un máximo de ocho componentes).

**File-Set:** Es una colección de archivos DICOM (y posiblemente no DICOM) que comparten un espacio de nombres donde los identificadores de los archivos (**File ID**) son únicos. En el mismo se debe de encontrar un fichero cuyo identificador es DICOMDIR.

**FSC:** File-Set Creator, denominación para aquellas entidades de aplicaciones que crean un fichero DicomDir a partir uno o varios ficheros DICOM.

**FSU:** File-Set Updater, denominación para aquellas aplicaciones que modifica el contenido de un fichero DicomDir a partir de cambios realizados en los ficheros DICOM que conforman al mismo.

**IDE:** Sigla de las iniciales de Integrated Development Environment, en español: Entorno de Desarrollo de Software. Es un programa compuesto por un conjunto de herramientas para un programador. Los mismos pueden soportar varios lenguajes de programación. Su consistencia está formada por un editor de código, un compilador, un depurador, y un constructor gráfico.

**NEMA:** Sigla de National Electrical Manufacturers Association. NEMA es la asociación comercial para la industria de la transformación eléctrica, es además la mayor asociación comercial para los fabricantes de productos eléctricos en los Estados Unidos de América. Presenta una participación activa en la creación de estándares.

**PACS:** Sigla anglosajona de Picture Archiving and Communication System (Sistema de Archivo y Transmisión de Imágenes). Es un sistema para la gestión, transmisión de imágenes médicas. Está formado por computadoras, redes y software que facilitan estas actividades.



**RUP:** Sigla anglosajona de Rational Unified Process es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. El mismo es un conjunto de metodologías adaptables al contexto y necesidades de cada organización.

**SDK:** Sigla de Software Development Kit o Kit de Desarrollo de Software. Es un conjunto de herramientas de desarrollo que le permiten a un programador crear aplicaciones para un sistema bastante concreto.

**UDF:** Sigla de Universal Disk Format, es una especificación de formato de archivos para el almacenamiento en medios ópticos. Utiliza la tecnología de grabación de paquetes (Packet Printing) soportado por grabadoras CD-RW, DVD-RAM/RW, HD DVD y Blue-ray

## Índice de Figuras

Figura 1 El fichero DICOM está dividido en dos partes. ....	9
Figura 2 Estructura del FileSet. ....	10
Figura 3 Ejemplo de la organización del Directorio (DicomDir) y su contenido. ....	12
Figura 4 Diagrama de casos de uso del negocio.....	26
Figura 5 Diagrama de Actividades CU Realizar Estudio.....	27
Figura 6 Diagrama Actividades CU Realizar Backup.....	28
Figura 7 Modelos de objetos del negocio. ....	29
Figura 8 Diagrama de Casos de Uso.....	39
Figura 9 Realización del CU Crear DicomDir.....	45
Figura 10 Diagrama de Análisis del CU Crear DicomDir .....	45
Figura 11 Realización del CU Empaquetar ficheros .....	46
Figura 12 Diagrama de Análisis del CU Empaquetar ficheros .....	47
Figura 13 Diagrama de secuencia del CU Empaquetar Imágenes. ....	48
Figura 14 Realización del caso de uso Quemar Disco .....	49
Figura 15 Diagrama de Análisis del CU Quemar Disco. ....	49
Figura 16 Diagrama de componentes Alas PACS Burner .....	53
Figura 17 Diagrama de Despliegue para el Alas PACS Burner .....	54
Figura 18 Diagrama de requisitos: Organizar las Imágenes.....	60
Figura 19 Diagrama de clases de Diseño CU Crear DicomDir. ....	66
Figura 20 Diagrama de clases de Diseño CU Empaquetar Imágenes.....	67
Figura 21 Diagrama de clases de Diseño CU Quemar Disco .....	68

## Índice de Tablas

Tabla 1 Actores del negocio. ....	25
Tabla 2 Trabajadores del negocio. ....	26
Tabla 3 RF del paquete de Obtención de DICOM. ....	31
Tabla 4 RF del paquete de Empaquetado de las imágenes. ....	32
Tabla 5 RF del paquete Quemado de las imágenes. ....	32
Tabla 6 RF del paquete Configuración. ....	33
Tabla 7 Actores del Sistema. ....	34
Tabla 8 Casos de uso para el primer ciclo de desarrollo. ....	40
Tabla 9 Casos de uso para el segundo ciclo de desarrollo. ....	42
Tabla 10 Casos de uso para el tercer ciclo de desarrollo. ....	42
Tabla 11 Caso de uso expandido Crear DicomDir. ....	61
Tabla 12 Caso de uso expandido Empaquetar. ....	63
Tabla 13 Caso de uso expandido Quemar Disco. ....	64
Tabla 14 Caso de uso expandido Obtener Imágenes desde Servidor. ....	65
Tabla 15 Descripción textual de la clase DataFile. ....	69
Tabla 16 Descripción textual de la clase DicomDirCompletedEventArgs. ....	69
Tabla 17 Descripción textual de la clase DicomDirCreator. ....	70
Tabla 18 Descripción textual de la clase Patient. ....	71
Tabla 19 Descripción textual de la clase Study. ....	72
Tabla 20 Descripción textual de la clase Series. ....	72
Tabla 21 Descripción textual de la clase Image. ....	73
Tabla 22 Descripción textual de la clase Compilation. ....	75
Tabla 23 Descripción textual de la clase Compiler. ....	75
Tabla 24 Descripción textual de la clase DicomImageProperties. ....	76
Tabla 25 Descripción textual de la clase FileNode. ....	77
Tabla 26 Descripción textual de la clase FolderNode. ....	77
Tabla 27 Descripción textual de la clase KNode. ....	81
Tabla 28 Descripción textual de la clase NodeAttributes. ....	81
Tabla 29 Descripción textual de la clase Package. ....	82
Tabla 30 Descripción textual de la clase RootNode. ....	83
Tabla 31 Descripción textual de la clase Schema. ....	84
Tabla 32 Descripción textual de la clase SubNodesCollection. ....	84
Tabla 33 Descripción textual de la clase ProgressData. ....	85
Tabla 34 Descripción textual de la clase ResultImageProgress. ....	85
Tabla 35 Descripción textual de la clase ISOProgressData. ....	86
Tabla 36 Descripción textual de la clase BurnProgressData. ....	87
Tabla 37 Descripción textual de la clase DiscMaster. ....	90
Tabla 38 Descripción textual de la clase DiscWriter. ....	92
Tabla 39 Descripción textual de la clase ImageCreator. ....	94
Tabla 40 Descripción textual de la clase Task. ....	95