Universidad de las Ciencias Informáticas

Facultad 7

Framework CALIB. Procesos de lectura, procesamiento y visualización de imágenes médicas

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

Autores: Maikel Sánchez Dieguez

Antonio Enrique Vallés Gámez

Tutores: MSc. Héctor Raúl González Diez

Ing. Filiberto López Palenzuela

Ciudad de la Habana, Cuba
Marzo de 2010
"Año 52 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y cedemos 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 25 días del mes de marzo del año 2010.

Maikel Sánchez Dieguez

_____

Firma del autor

Antonio Enrique Vallés Gámez

_____

Firma del autor

MSc. Héctor Raúl González Diez

_____

Firma del tutor

Ing.Filiberto López Palenzuela

_____

Firma del tutor

RESUMEN

En el presente trabajo se muestra el desarrollo de una solución de software que permite la lectura, procesamiento y visualización de imágenes médicas, provenientes de múltiples equipos y de diversas modalidades. Permite, además, por parte de los desarrolladores la inclusión de forma sencilla de nuevos formatos de representación de imágenes médicas.

Para la realización de este trabajo se hizo un análisis del estándar DICOM 3.0 y de otros documentos donde se especifica la estructura de los ficheros de representación de imágenes. Además, se estudiaron soluciones similares para dar a los usuarios una forma más eficaz e intuitiva para realizar sus desarrollos.

Para el desarrollo de la solución se empleó el *Microsoft Visual Studio 2008* y el lenguaje de programación C#, así como el uso de la herramienta case *Enterprise Architect*, para el modelado del sistema.

PALABRAS CLAVES

DICOM, procesamiento, visualización, imágenes médicas.

TABLA DE CONTENIDOS

DEDICATORIA 4
AGRADECIMIENTOS 5
RESUMEN 6
INTRODUCCIÓN 9
CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA 14
1.1 Modalidades de imágenes médicas 14
1.2 Estaciones de visualización de imágenes médicas 19
1.3 Frameworks 21
1.4 Formatos de representación de imágenes médicas 24
1.5 Estado del Arte 33
1.6 Herramientas y tecnologías utilizadas 34
CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA 38
2.1 Breve descripción del sistema 38
2.2 Modelo de Dominio 38
2.3 Especificación de los Requisitos de Software 40
2.4 Actores del Sistema 43
2.5 Diagrama de Caso de Uso del sistema 43
CAPÍTULO 3. DISEÑO DEL FRAMEWORK 50
3.1 Estilo arquitectónico utilizado 50
3.2 Diagramas de Clases 51
3.3 Diagramas de Secuencia 57
CAPÍTULO 4. IMPLEMENTACIÓN 64
4.1 Diagrama de Componentes 64
4.2 Integración dentro del sistema alas PACS 65
4.3 Instanciación del framework 65
4.4 Integridad de los Componentes 66
4.5 Ejemplo de Extensibilidad 67
CAPÍTULO 5. PRUEBAS 69
5.1 Proceso de lectura 69
5.2 Algoritmos de Visualización 70

5.3 Compatibilidad con MONO 2.4 71
CONCLUSIONES 72
RECOMENDACIONES 73
REFERENCIAS BIBLIOGRÁFICA 74
BIBLIOGRAFÍA 76
ANEXOS 79
GLOSARIO 84

INTRODUCCIÓN

Las imágenes médicas complementan en gran medida los diagnósticos de pacientes, tratamientos terapéuticos, la planificación quirúrgica, el muestreo de enfermedades, y a largo plazo para repetir y dar seguimiento a resultados. En las pasadas tres décadas, hubo grandes cambios con la llegada de nuevas técnicas como la tomografía computarizada, la resonancia magnética, la resonancia espectroscópica, la resonancia magnética funcional, la angiografía por sustracción digital, la tomografía por emisión de positrones, entre otras; que han revolucionado la técnica del diagnóstico por imagen [1].

Con la aparición de estas tecnologías y otras asociadas como el ultrasonido, la medicina nuclear, las redes de comunicación, las computadoras personales y los medios de almacenamiento; surgen los sistemas para la transmisión y el almacenamiento de imágenes médicas[1] . Estos sistemas normalmente están formados por elementos que tienen la capacidad de interactuar entre sí (Fig. 1).

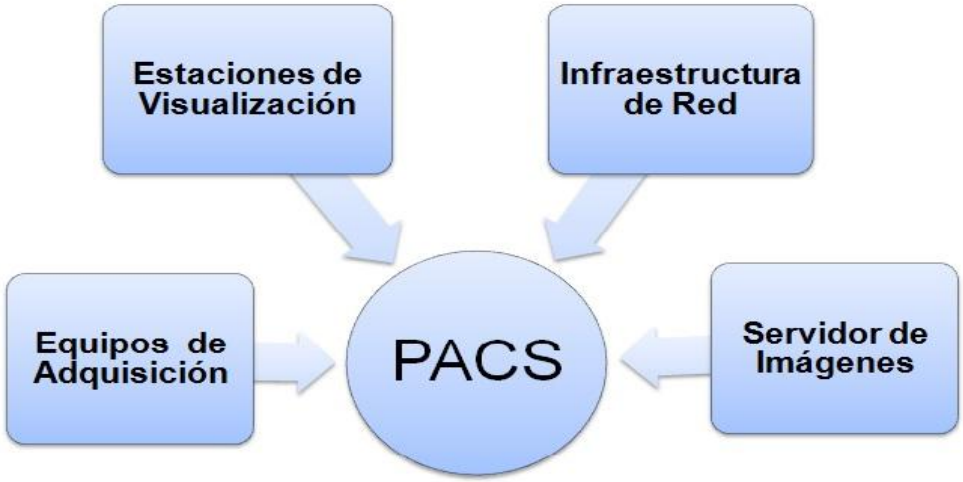


Fig.1 Componentes de un PACS

Las estaciones de visualización de los radiólogos y algunos clínicos, son estaciones de trabajo o PC exclusivamente dedicadas con varios monitores de alta resolución, aunque también suelen utilizarse para estos efectos las estaciones de post procesamiento o las asistentes *(wizards)*, que forman parte del equipo de adquisición de imágenes. Para poder realizar las funciones de visualización y procesamiento, estas

[1]Picture Archiving and Communication System (PACS)

estaciones requieren acceder a la información almacenada en los formatos de almacenamiento de imágenes médicas [2].

Una modalidad de imagen médica define bajo qué términos y condiciones específicas se adquirió dicha imagen, en dependencia de las técnicas y dispositivos utilizados.

Existen diversas modalidades de imágenes médicas y se pueden clasificar de acuerdo a su contenido anatómico (estructurales) y fisiológicos (funcionales), cada una de estas tiene funcionalidades y características que no pueden ser remplazadas por otras modalidades.

Para lograr una mayor integración entre las imágenes de diferentes modalidades estas se pueden representar en formatos diseñados para medicina, como los formatos DICOM[2], ANALYZE 7.5, NifTI[3], entre otros.

DICOM es el estándar más difundido y abarcador existente hasta el momento, por lo que es considerado el estándar industrial para la transferencia y visualización de imágenes médicas digitales y la información asociada a ellas. Es aplicable al terreno de la transmisión, tratamiento e impresión de todo tipo de imágenes médicas, independientemente de la especialidad médica.

Por necesidades específicas de algunas modalidades funcionales como la resonancia magnética funcional, surgen los estándares ANALYZE 7.5 y NifTI, que poco a poco han ido tomando su lugar entre los fabricantes de equipos de estas modalidades.

Como parte del desarrollo del sistema alas PACS[4] se adquirió una biblioteca de clases llamada C# DICOM SDK que facilita, en alguna medida, el manejo de las imágenes médicas en formato DICOM. Esta biblioteca no integra otros formatos de imágenes y no brinda funcionalidades para la descompresión de ficheros en el formato soportado y aunque DICOM es un estándar bien difundido en la imagenología médica, aún quedan proveedores de equipos que mantienen formatos de imágenes comunes.

En Cuba existen numerosos equipos de imágenes digitales que no utilizan formatos DICOM. Es por ello que fue necesario emplear alternativas, no sostenibles en el tiempo, para dar solución parcial a estos problemas dentro del sistema alas PACS.

---

[2] Digital Imaging Communication in Medicine (DICOM).
[3] Neuroimaging Informatics Technology Initiative (NifTI).
[4]Sistema PACS desarrollado por el Dpto. de Software Médico Imagenológico, perteneciente al Centro Especializado en Soluciones para Informática Médica de la Universidad de Ciencias Informáticas.

Además de la diversidad de formatos se han encontrado diferentes formas de compresión de las imágenes según las características de cada equipo y que no es posible cubrir con la biblioteca de clases que se tiene actualmente.

Por otra parte, las empresas que desarrollan soluciones de imágenes médicas en general cuentan con biblioteca de clases o *frameworks*[5] propios para soportar todos sus desarrollos y en este sentido es importante ir basando el desarrollo en soluciones propias.

A raíz de la problemática anteriormente expuesta se plantea el siguiente **problema científico:** necesidad de lectura, procesamiento y posterior visualización de imágenes médicas con diversos formatos.

Con vista a dar solución al problema planteado se propone como **objeto de estudio**: los formatos de representación de imágenes médicas; y como **campo de acción**: los procesos de lectura, procesamiento y visualización en frameworks, biblioteca de clases y aplicaciones de manejo de imágenes médicas; centrando la atención en los formatos ANALYZE 7.5, NifTI y DICOM.

Para dar solución al problema anteriormente planteado, se tiene como **objetivo general de la investigación:** desarrollar un framework que permita la lectura, procesamiento y la visualización de imágenes médicas.

Para dar cumplimiento al objetivo general se proponen las siguientes **tareas a desarrollar**:

- Analizar las especificidades establecidas para los formatos de representación de imágenes ANALYZE 7.5, NifTI y DICOM.
- Analizar el estado del arte en cuanto a frameworks, biblioteca de clases y componentes existentes para el manejo de imágenes médicas en diferentes formatos.
- Especificar qué parte del estándar DICOM cubriría la implementación del framework.
- Modelar el ambiente en el que se incluye el framework como parte de los sistemas PACS.
- Identificar los requisitos de software para la implementación del framework.

[5] Su traducción literal al idioma español es: marco de trabajo, pero en esta investigación para referirse al término se usará la palabra en idioma inglés.

- Realizar una propuesta de arquitectura para el framework.
- Diseñar la solución propuesta.
- Implementar la solución propuesta.
- Diseñar un plan para pruebas de integración, rendimiento y pruebas que soporten las diferentes imágenes médicas según los procesos definidos en el alcance.
- Realizar pruebas a la solución desarrollada.

A continuación se describe de manera resumida el contenido que se expone en cada capítulo del presente trabajo:

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA: Se explican algunas de las características principales de las modalidades de imágenes; así como las de las estaciones de visualización. Se da una breve panorámica de los conceptos de framework, sus beneficios y otras técnicas de reutilización en la programación orientada a objetos. Se hace un análisis de los estándares de representación de imágenes médicas más difundidos como son DICOM, ANALYZE 7.5 y NifTI, con el objetivo de profundizar en los conceptos que brindan soporte a la investigación.

Se hace un análisis de las principales soluciones existentes en Cuba y el resto del mundo; así como las tecnologías utilizadas para el desarrollo del trabajo.

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA: Se presenta la propuesta de solución del framework para la situación problémica. Se muestran los procesos del negocio mediante el modelo de dominio y las características y funcionalidades que tendrá el mismo a partir de los requerimientos funcionales y no funcionales. Se realiza el diagrama de casos de uso del sistema, las especificaciones de los casos de uso asociados al componente y la trazabilidad de los casos de uso con sus correspondientes requisitos funcionales.

CAPÍTULO 3. DISEÑO DEL FRAMEWORK: Se describe cómo implementar el framework, a través del diseño, enfocado a cómo el sistema cumple sus objetivos teniendo en cuenta los requisitos funcionales y no

funcionales. Se realizan los diagramas de clases y los diagramas de interacción según los casos de uso definidos y se explica además la arquitectura utilizada.

CAPÍTULO 4. IMPLEMENTACIÓN: Se describe como fue implementado el framework CALIB. Se presenta el diagrama de componentes, que muestra la interacción de estos después de ser implementados. Se da un breve panorámica de cómo se integraría el framework como parte del sistema alas PACS y su instanciación dentro del Componente de Visualización 3D y su utilización en el Visor WEB de imágenes médicas.

CAPÍTULO 5. PRUEBAS: Se expone el resultado de algunas pruebas realizadas al framework, en cuanto a rendimiento calidad de los algoritmos de visualización y compatibilidad con la plataforma MONO.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

Se explican algunas de las características principales de las modalidades de imágenes; así como las de las estaciones de visualización. Se da una breve panorámica de los conceptos de framework, sus beneficios y otras técnicas de reutilización en la programación orientada a objetos. Se hace un análisis de los estándares de representación de imágenes médicas más difundidos como son DICOM, ANALYZE 7.5 y NifTI, con el objetivo de profundizar en los conceptos que brindan soporte a la investigación.

Se hace un análisis de las principales soluciones existentes en Cuba y el resto del mundo; así como las tecnologías utilizadas para el desarrollo del trabajo.

1.1 Modalidades de imágenes médicas

1.1.1 Tomografía Axial Computarizada

Las imágenes de Tomografía Axial Computarizada (TAC) son particularmente útiles porque puede mostrar con gran claridad diferentes tejidos.

Debido a que el TAC es un procedimiento mínimamente invasivo que proporciona vistas transversales detalladas de todo tipo de tejidos, se está convirtiendo en el método preferido para diagnosticar muchas enfermedades intestinales como apendicitis, y para visualizar el hígado, el bazo, el páncreas y los riñones [3].

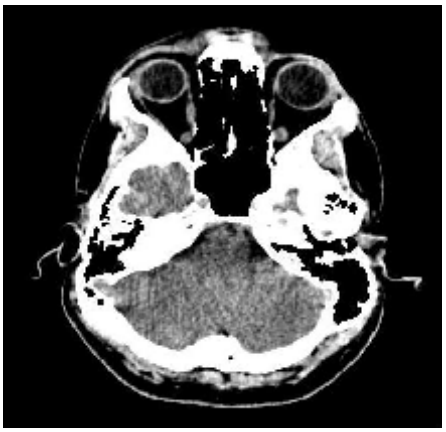


Fig.2 Imagen de TAC

1.1.2 Resonancia Magnética Nuclear

Las imágenes de Resonancia Magnética Funcional (RMN) se usan para diagnosticar lesiones deportivas, especialmente en la rodilla, el hombro, la cadera, el codo y la muñeca. Las imágenes le permiten al médico ver hasta pequeñísimos desgarros y lesiones en los ligamentos y músculos.

Además, la RMN del corazón, la aorta, las arterias coronarias y los vasos sanguíneos es una herramienta no invasiva para diagnosticar enfermedades de las arterias coronarias y del corazón. Los médicos pueden examinar el tamaño y el grosor de las cámaras del corazón y determinar el grado de daño causado por un ataque del corazón o por una enfermedad cardíaca progresiva; también se puede examinar detalladamente los órganos del pecho y el abdomen como los pulmones, hígado, riñones, bazo, páncreas y vasos sanguíneos abdominales, lo que permite diagnosticar y evaluar tumores y enfermedades funcionales [4].

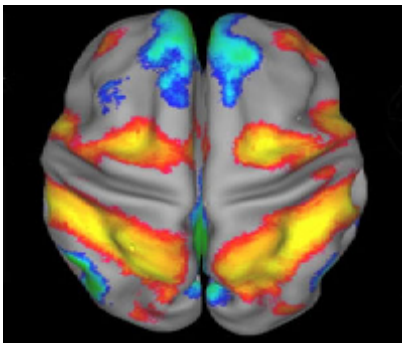


Fig.3 Imagen de RMN

1.1.3 Resonancia Magnética

Las imágenes de Resonancia Magnética (RM) permiten visualizar distintos órganos y tejidos del organismo, por lo que sus indicaciones son varias dependiendo de la zona examinada.

Cuando está indicada para valorar columna vertebral, permite ver mejor los tejidos blandos, es decir, todos los componentes de la columna vertebral que no son hueso. Por tanto, es indicada cuando se sospeche hernia discal, o para valorar discopatías[6], ante sospecha de fractura vertebral o daño medular.

[6]Discopatía: Enfermedad del disco intervertebral.

Cuando está indicada para valorar el cerebro o tejido nervioso, esta proporciona imágenes detalladas del tronco del encéfalo y de la zona posterior del cerebro, la cual es difícil de observar mediante la tomografía axial computarizada (TAC).

Cuando es usada para estudiar el corazón, proporciona imágenes detalladas del mismo y de los vasos sanguíneos, y puede diferenciar tejidos de la sangre en movimiento. También puede diferenciar entre el músculo cardíaco y los tejidos circundantes, y clarificar hallazgos de radiografías o TAC previos. Es buena para mostrar el corazón con imágenes desde múltiples planos. También es útil en el diagnóstico de anomalías congénitas, crecimientos anormales y tumores. Algunas de las enfermedades para las que está indicada son: trastornos de las válvulas cardíacas, estudios de derrames pericárdicos[7], estudio de tumores cardíacos o con invasión de los vasos sanguíneos, estudio de anomalías congénitas del corazón o estudio de fibrosis o cicatrización del músculo cardíaco [4].

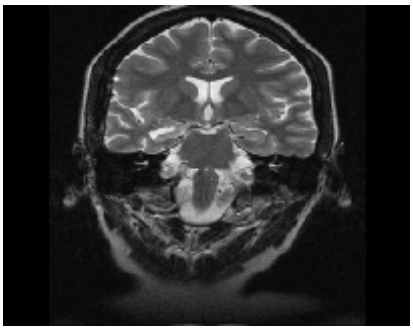


Fig.4 Imagen de RM

1.1.4 Tomografía por Emisión de Positrones

Las imágenes obtenidas de la Tomografía por Emisión de Positrones (PET) se usan frecuentemente para detectar cáncer y evaluar el efecto del tratamiento oncológico.

Los estudios de PET del corazón se pueden utilizar para determinar el flujo sanguíneo del corazón y evaluar los signos de enfermedad coronaria. Se usan también para determinar cuáles zonas de funcionamiento cardíaco reducido están aún vivas, y cuáles son tejido cicatrizal secundarias a un infarto previo. Ello permite

[7] Derrames pericárdicos: acumulación de sangre u otros líquidos una fuerte membrana que rodea completamente al corazón, separándolo de los órganos y estructuras vecinos.

diferenciar entre el músculo cardíaco no funcional y el músculo cardíaco todavía viable, que se podría beneficiar de un procedimiento terapéutico.

Los PET del cerebro se usan para evaluar a los pacientes con trastornos de la memoria de causa no determinada, o ante la sospecha de tumores cerebrales [5].

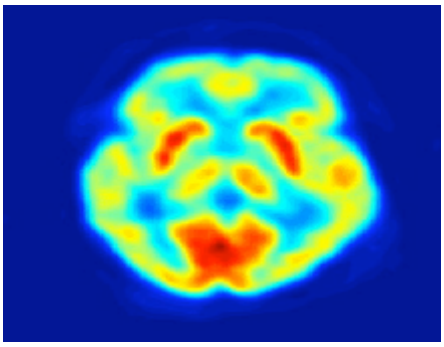


Fig.5 Imagen de PET

1.1.5 Radiografía Digital

Las imágenes de Radiografía Digital (RX) tienen múltiples aplicaciones. No hay ningún síntoma torácico que no pueda ser examinado con una radiografía de tórax. La radiografía de abdomen puede estar indicada ante la presencia de síntomas derivados del aparato digestivo como dolor abdominal, vómitos, alteraciones en las deposiciones, así como alteraciones del aparato urinario como son los cálculos renales.

Las radiografías óseas ponen de relieve posibles fracturas, y luxaciones de las articulaciones. Otra importante indicación es la observación de la evolución de una lesión diagnosticada previamente.

Existen múltiples utilidades y proyecciones que ponen de relieve afecciones de diversos órganos, como es la radiografía craneal para ver alteraciones óseas. En ocasiones se puede usar una sustancia que hace de contraste en la radiografía y nos permiten ver mejor algunas estructuras [3].

Fig.6 Imagen de RX

1.1.6 Angiografías

Las imágenes generadas por los equipos de Angiografías (XA) se utilizan para ver si hay vasos sanguíneos que se han estrechado, bloqueado o tienen algún otro tipo de problema. Los resultados pueden ayudarle al profesional médico a determinar si necesita tratamiento para ensanchar una arteria, eliminar un bloqueo o hacer una derivación (bypass) de la arteria [6].

Fig.7 Imagen de XA

1.1.7 Ultrasonido

Las imágenes obtenidas por el Ultrasonido (US) pueden ayudar a diagnosticar diversas enfermedades y a evaluar el daño en los órganos luego de una enfermedad. Se usa para ayudar a los médicos a diagnosticar síntomas tales como, dolores, hinchazón, infección. Es una forma útil de examinar muchos de los órganos internos del cuerpo, incluyendo en forma enunciativa y no limitativa [6].

El ultrasonido también se usa para:

- Guiar procedimientos como biopsias por aspiración, en las que se usan agujas para extraer muestras de células de un área anormal para realizar pruebas de laboratorio.
- Obtener una imagen de los senos y guiar la biopsia del cáncer de seno.
- Diagnosticar diversas enfermedades coronarias y evaluar el daño luego de un ataque al corazón.

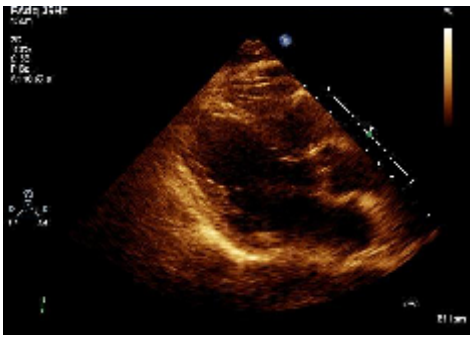


Fig.8 Imagen de US

1.2 Estaciones de visualización de imágenes médicas

Una de las principales características de los PACS es contar con estaciones de trabajo o terminales en la cuales los usuarios puedan visualizar e interpretar las imágenes. En términos de acceso a las imágenes, los radiólogos necesitan pantallas grandes de alta resolución para la interpretación. También estas pantallas deberían ser lo suficientemente flexibles como para permitirles la visualización de estudios previos, correlativos; organizar y reorganizar las imágenes para mostrar una secuencia temporal. Muchas veces los radiólogos interpretan un estudio comparando múltiples imágenes, incluyendo estudios previos. Es por ello que estas estaciones deben ser capaces de redistribuir las imágenes, aumentarlas de tamaño, reducirlas y manipularlas fácilmente a través de la pantalla.

Las estaciones de visualización, en dependencia de las modalidades para las que sean utilizadas, tienen prestaciones diferentes y mayor o menor grado de especialización, pero todas tienen en común la necesidad de mostrar con la mayor calidad posible las imágenes utilizadas para el diagnóstico. Según la definición y cantidad de componentes que tiene cada uno pueden clasificarse en [2]:

- **Estación Diagnóstica de Alta Resolución:** Desde la cual el médico radiólogo realiza el informe generado a partir de la imagen de radiografía digital. Son las de máxima resolución y las que tienen

que tener gran capacidad de interacción con el usuario (procesamiento de las imágenes, zoom, paneo, marcar zonas de interés, etc.). Normalmente están compuestas por 1 o hasta 4 monitores monocromáticos de presentación vertical y muy alta resolución (2048 x 2560 pixel[8]). Debe permitir la visualización de las imágenes a una velocidad mayor que en un entorno tradicional.

Fig.9 Estación diagnóstica de alta resolución

- **Estación Diagnóstica de Resolución Media:** Tiene características similares a las estaciones diagnósticas de alta resolución en cuanto a las posibilidades de interacción por parte del usuario, pero con 1 ó 2 monitores de presentación vertical u horizontal de resolución media (1200 x 1600 pixel). Se utilizan normalmente para diagnosticar sobre imágenes de TAC, MR y US.
- **Estación Clínica:** Tiene características similares a las estaciones diagnósticas de resolución media en cuanto a la resolución, posiblemente con uno de los dos monitores a color pero con menor interacción con el usuario, es decir, que sólo permita consultar los estudios, sin llegar a diagnosticar.
- **Estación de Consulta Web:** Un sólo monitor estándar, con una resolución de (1024 x 768 pixel) o (1280 x 1024 pixel). La imagen llega del servidor de imágenes a pedido del usuario a través de una aplicación WEB de visualización.

Algunas estaciones mencionadas, debido a las características de los equipos de las distintas modalidades para las que se emplean tienen un mayor grado de especialización. Por ejemplo para:

- **Ecografía:** Permitan al profesional el procesamiento y la visualización de imágenes de video.

[8]Pixel: unidad de medida más pequeña en una imagen digital 2D.

- **Quirófanos:** Con control remoto para la ubicación y posición de la terminal en el quirófano, soportes en la pared o en el techo, etc.
- **Ortopedia:** Con plantillas visuales especiales, por ejemplo para prever la posición de una prótesis.
- **Cardiología:** Que permita la visualización de videos (o porciones de videos) de ecocardiografías, visualización dinámica de flujos de angiografías y cateterismos, visualización de electrocardiogramas, herramientas de análisis, etc.

1.3 Frameworks

Los frameworks orientados a objeto son la piedra angular de la ingeniería del software moderna, están ganando rápidamente la aceptación debido a su capacidad para promover la reutilización del diseño y el código fuente. Los frameworks son los generadores de aplicación que se relacionan directamente con un dominio específico, es decir, con una familia de problemas relacionados [7].

Muchos de los que se dedican al desarrollo de software utilizan, conocen o, como mínimo, se han tropezado con el concepto de framework, cuya traducción aproximada sería *marco de trabajo*. Sin embargo, el concepto de framework no es tan simple y mucho menos sencillo de definir; aunque cualquiera con experiencia en la programación captará su sentido de manera casi intuitiva e incluso es posible que esté utilizando su propio framework. Resulta pertinente citar las siguientes definiciones:

"…diseño abstracto orientado a objetos para un determinado tipo de aplicación, que se compone de una clase abstracta para cada componente principal del diseño; contendrá normalmente una librería de subclases que pueden ser utilizadas como componentes del diseño…" [8].

"…una arquitectura genérica que proporciona una plantilla ampliable para su aplicación dentro de un dominio…" [9].

1.3.1 Fases de Desarrollo

El desarrollo de un framework orientado objeto cuenta con tres etapas fundamentales [7]:

1. Análisis del Dominio: en esta fase procura descubrir los requisitos del dominio y los posibles requerimientos futuros. Para completar los requerimientos sirven las experiencias previamente publicadas, los sistemas de software similares existentes, las experiencias personales, y los

estándares considerados. Durante el análisis del dominio, los puntos calientes y los puntos congelados se destapan parcialmente.

2. Diseño del framework: es esta fase se define las abstracciones de éste. Se modelan los puntos calientes, puntos congelados, la extensión y la flexibilidad propuesta en el análisis del dominio se esboza en líneas generales.
3. Instanciación: los puntos calientes del framework son implementados, generando un software del sistema. Es importante observar que cada uno de estas aplicaciones tendrá los puntos congelados del framework en común.

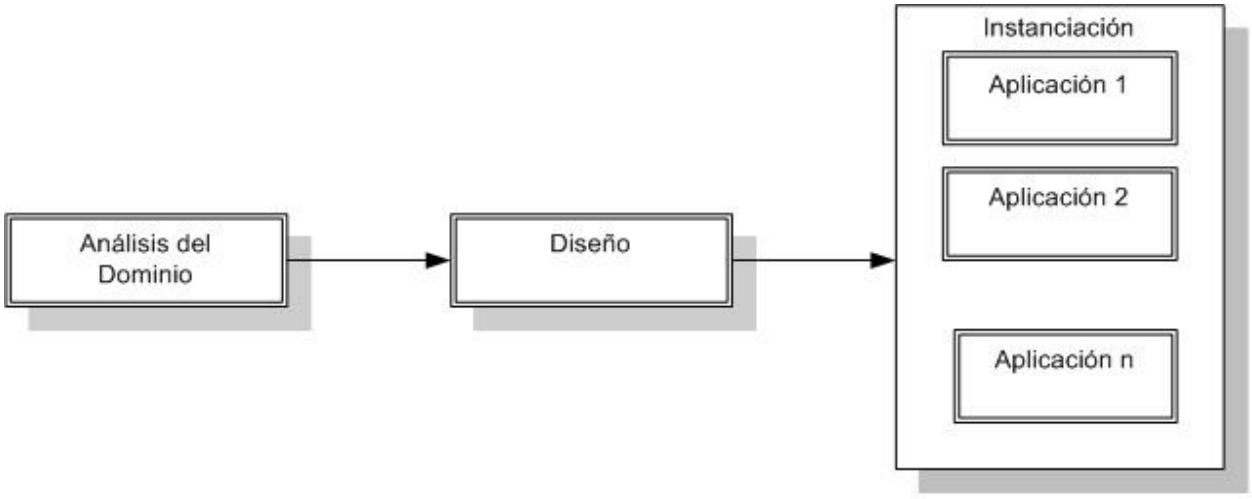


Fig.10 Fases del Desarrollo de un Framework

1.3.2 Beneficios al usar un framework

Se pueden categorizar los beneficios que aporta el uso de los frameworks basado en las siguientes etapas del ciclo de vida del software [7]:

- Diseño

  Adoptar un framework dinamiza el proceso de diseño y como los frameworks están alineados con los requisitos del dominio, el diseño se simplifica. Varias de las funcionalidades que tendrían que ser diseñadas, forman parte del framework, como resultado, el diseño estará centrado en los problemas del dominio y los detalles del mismo.

- Desarrollo y prueba
  - Mayor productividad del desarrollador: Al estar los frameworks enmarcados en un dominio, permiten que los desarrolladores sean productivos inmediatamente.
  - Menos codificación: Muchos de los códigos personalizados, ahora son reemplazados en su totalidad por el framework, o es una configuración del mismo; por lo que existe menos código a probar.
  - Mayor consistencia: Con frecuencia se encuentran varias soluciones al mismo problema entre los miembros del equipo de desarrollo; si se emplea un framework, se crea una propuesta consistente para usar a través del desarrollo del software.
  - Resultados inmediatos: Al dinamizar el proceso de asociación de los requerimientos a funcionalidades se reduce considerablemente el tiempo de desarrollo. Esto permite un mayor número de iteraciones en un corto plazo, lo que conlleva a un producto de más calidad y mayor inmediatez en la entrega.
  - Mayor flexibilidad: El bajo acoplamiento proporcionado por los frameworks aporta una gran flexibilidad, que responde a los cambios de los requisitos tanto del negocio como de tecnología.
  - Arquitectos más eficaces: El esqueleto o estructura proporcionada por los frameworks permite que los arquitectos dediquen más tiempo a la toma de decisiones imprescindibles en vez de gastar una enorme cantidad de tiempo en hacer cumplir las mejores prácticas (el framework hace el trabajo).
  - Líderes más eficientes: El mayor desafío para un líder de proyecto es organizar el trabajo correctamente y asignar los recursos adecuados a las personas idóneas, decisiones que se facilitan extraordinariamente dentro de la estructura proporcionada por el framework [7].

- Producción y mantenimiento

  Los frameworks aportan ventajas a la producción y al mantenimiento debido a la flexibilidad y consistencia que brinda su empleo correcto. Si se usa un framework de forma adecuada el software puede responder fácilmente a los cambios en los requerimientos tanto del negocio como de tecnología. El mantenimiento de las aplicaciones históricamente ha requerido descubrir cómo un desarrollador solucionó un problema determinado y luego cambiar ese comportamiento. Una solución a esto es usar miembros del equipo de desarrollo original para mantener el software, que

evidentemente no hace un uso eficiente de los recursos. En cambio al usar un framework, una solución consistente es usada a través del desarrollo y por ende lleva consigo uno de los principios fundamentales de los frameworks, la flexibilidad. Es muy fácil localizar la parte de la aplicación afectada por los nuevos requerimientos y realizar los cambios de una manera controlada [7].

1.3.3 Otras técnicas de reutilización

La reutilización es una práctica que permite agilizar tanto el trabajo de los arquitectos como de los desarrolladores, uno de los paradigmas de la programación donde por sus características se hace un gran uso de la reutilización es el Paradigma Orientado a Objetos, entre los principales artefactos reutilizables se pueden encontrar:

- Patrón: es una descripción de un problema y la solución, a la que se da un nombre, y que se puede aplicar a nuevos contextos; idealmente, proporciona consejos sobre el modo de aplicarlo en varias circunstancias, y considera los puntos fuertes y compromisos. Muchos patrones proporcionan guías sobre el modo en el que deberían asignarse las responsabilidades a los objetos, dada una categoría específica del problema [10].
- Componentes: Un componente de software es un elemento de software que se ajusta a un modelo de componentes y puede ser desplegado de forma independiente e integrada sin modificación de acuerdo a un estándar de composición [10].
- Biblioteca de clases: Es un conjunto de clases que brindan funcionalidades para resolver un problema en específico.

1.4 Formatos de representación de imágenes médicas

La totalidad de los sistemas de procesamiento y visualización de imágenes médicas utilizan archivos de formatos especialmente diseñados para estas imágenes. A continuación se exponen algunos de dichos formatos con sus principales características.

1.4.1 DICOM

DICOM es el formato contemplado en el estándar DICOM 3.0, para el almacenamiento, procesamiento y transmisión de imágenes médicas, el cual fue creado por ACR[9] y NEMA[10], para lograr una estandarización dentro de los fabricantes de equipos de adquisición de imágenes médicas.

Por su uso a nivel mundial por parte de los sistemas informáticos que implementan el estándar DICOM, constituye uno de los formatos de almacenamiento de imágenes médicas más extendidos y populares que existen.

Este estándar define las estructuras de datos para las imágenes médicas y la información relacionada, ciertos servicios orientados a las comunicaciones (transmisión de imágenes, consulta de un archivo de imagen, etc.) los formatos para el intercambio de los datos almacenados, y los requisitos de compatibilidad para dispositivos y programas.

Internamente un fichero DICOM está conformado por una estructura que permite a los desarrolladores poder acceder fácilmente (Fig. 11).

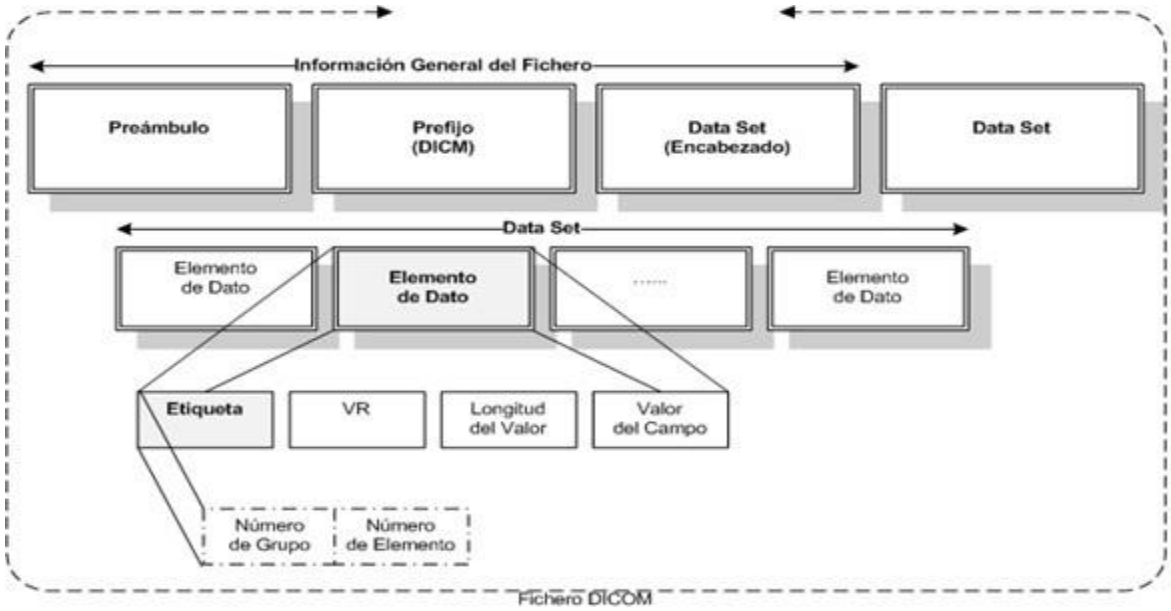


Fig.11 Estructura de un fichero DICOM

[9] American College of Radiology.
[10] National Electrics Manufacturers Association.

Preámbulo: Tiene un tamaño fijo de 128 bytes, y está pensado para tener un uso definido por la implementación. Por ejemplo, puede contener información sobre el nombre de la aplicación usada para crear el fichero, o puede tener información que permita a aplicaciones acceder directamente a los datos de la imagen almacenada en el fichero. En caso de no ser usado, el preámbulo debe estar presente, con todos sus bytes puestos al valor 00h.

Prefijo: Este prefijo consiste en cuatro bytes que contienen la cadena de caracteres "DICM". Esta cadena debe estar codificada siempre con las letras en mayúscula. El propósito de este prefijo es permitir a las implementaciones diferenciar si un fichero es DICOM o no.

Dataset: Es un conjunto de elementos de datos.

Dataset (encabezado): Solo contiene los elementos de datos que pertenecen al grupo 0002H.

Elemento de Datos: Está compuesto por una etiqueta, un valor de representación (VR), la longitud del valor y el valor.

Etiqueta: Sirve para identificar cada elemento de datos de forma única. Su representación es un vector de dos dimensiones, siendo la primera el número de grupo, y la segunda el número de elemento, en hexadecimal con cuatro dígitos. Por ejemplo, si el número de grupo es ocho y el número de elemento es doce, la etiqueta será (0008,000C).

Valor de Representación: Indica la forma en la que se codifica el valor del elemento. Anexo 1

Valor: Es el valor del elemento de datos, codificado según el campo VR y con la longitud que indica el campo Longitud del Valor y que puede contener información vinculada al fichero como por ejemplo información sobre el paciente(nombre, sexo), sobre la imagen, entre otros.

Estructura organizativa de DICOM

Un fichero DICOM contiene mucha información, por lo que el estándar propone una forma de organización basada en IOD[11].

Un IOD es una colección de partes de información relacionada, agrupadas en Entidades de Información o atributos. Cada entidad contiene información sobre un único objeto (mundo real) como un paciente, una

[11]Information Object Definition (IOD)

imagen, etc. Las entidades de información consisten en atributos, describiendo una única parte de información, por ejemplo, el nombre de un paciente. Los atributos que tienen una relación están agrupados en módulos de información de objetos o IOM[12].

Los IOMs están definidos de tal manera que pueden ser usados en más de un IOD. Estos IOMs también tienen la ventaja de que las descripciones semánticas de los atributos descritos pueden ser agrupados juntos [11].

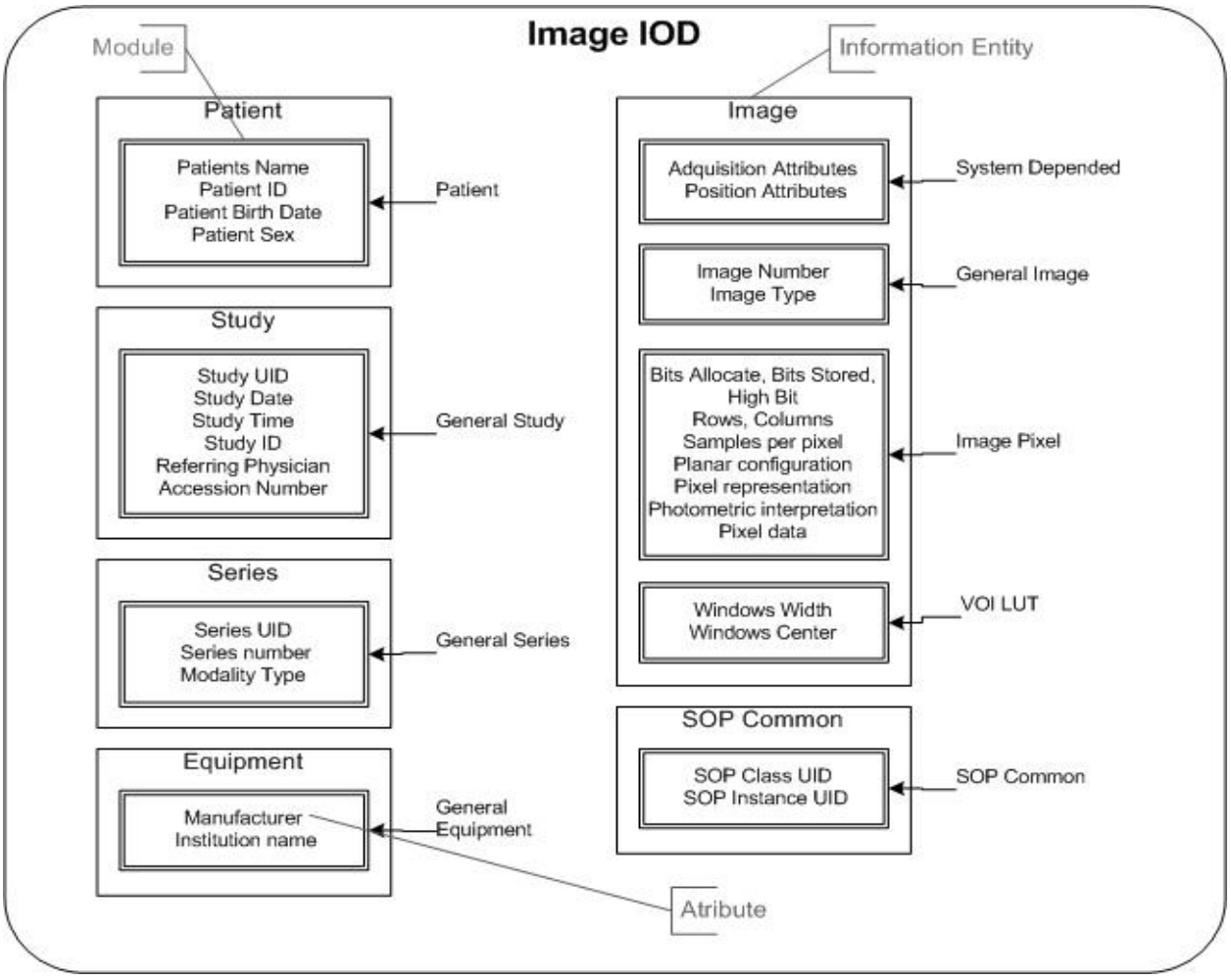


Fig.12 IOD Imagen

[12]Information Object Modules (IOM)

Procesamiento y visualización de imágenes DICOM

El proceso de descodificación de la información almacenada en el *Pixel Data* requiere de varios atributos del *Image Pixel Module*.

- ***Bits Allocated*:** Es la cantidad de bits en los que se puede representar la imagen médica, los más comunes son 8 y 16 bit, lo que posibilita que los valores de los pixels se encuentren entre 255 y 65535, aunque no siempre se utilizan la totalidad de los valores, y los que no se utilizan para representar intensidades de los pixels se pueden utilizar para crear *overlays* dentro de la imagen**.**

- ***Bits Stored*:** Son los bits destinados para almacenar la información de los pixels de cada una de las celdas.

- ***High Bit:*** Indica el bit a partir del cual se representan los bits almacenados en una celda.

- ***Rows* y *Columns*:** Indica el tamaño espacial de las imágenes.

- ***Samples per pixel:*** Indica en cuantos planos viene codificada la información de los pixel de cada celda, normalmente los valores que toma son 1 y 3 en dependencia del ***Photometric Interpretation***.

- ***Photometric Interpretation:*** Los valores que normalmente puede tomar son MONOCHROME1 y MONOCHROME2 para las imágenes en escala de grises y PALETTECOLOR y RGB para las imágenes en colores, aunque existen otros especificados en el capítulo 3 del estándar.

- ***Planar Configuration:*** Permite saber en qué orden vienen codificado los valores del pixel data, este campo solo existe cuando el samples per pixel es mayor que uno.

- ***Pixel Representation:*** Permite saber si losvalores de los pixel son representados con signos o sin signos.

- ***Pixel Data:*** Es el campo donde viene especificada la información visual de la imagen.

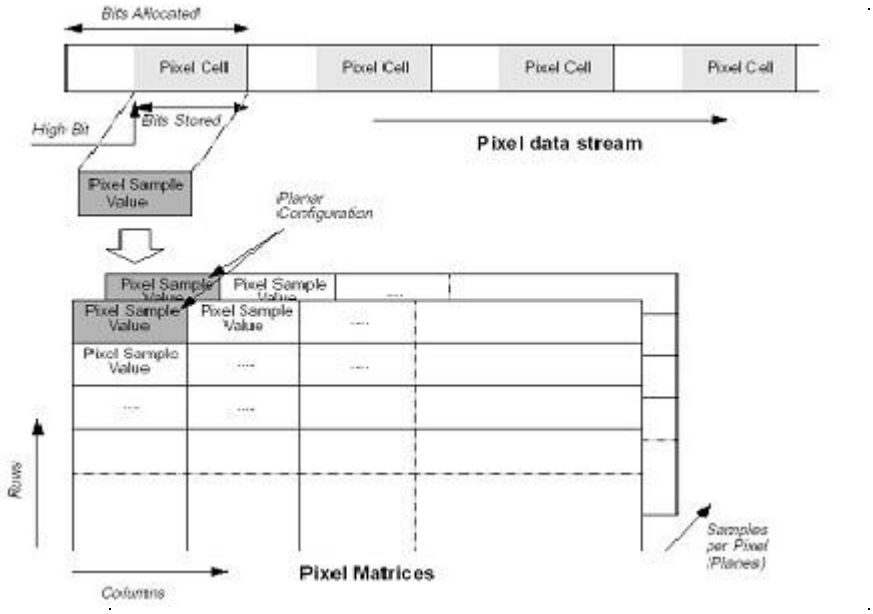


Fig.13 Decodificación del Pixel Data

Normalización

El resultado de aplicar la normalización indica que los valores de entrada de la imagen se modifican a una gama normalizada de los valores de salida según cierta función. Estos valores normalizados dependen del tipo de modalidad y de su uso clínico.

Por ejemplo, para los Rayos X, la intensidad es proporcional a los valores de pixel. Para los sistemas CT los valores de la muestra del *pixel* se convierten a la escala de *Hounsfield*, etc. La función que se utiliza en general es una función lineal que cuenta de dos parámetros que forman parte de los atributos del fichero *Rescale Slope* y *Rescale Intercept*. La figura muestra la función que mapea los valores de *pixel* decodificados a valores de *pixel* normalizados (Fig. 14).

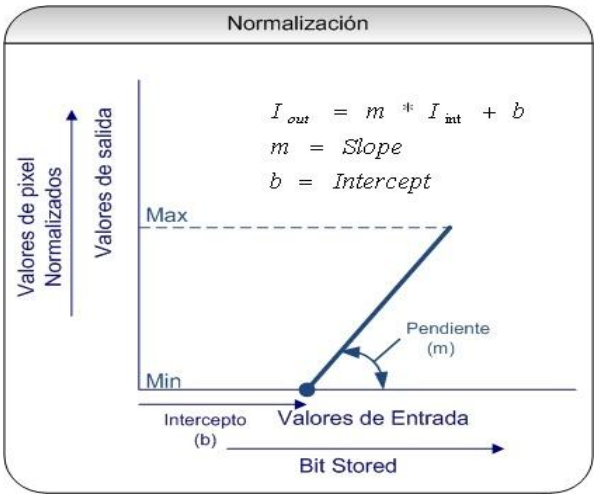


Fig.14 Función de Normalización

Existen algunas modalidades que requieren que la función de normalización sea una función no lineal donde los valores de entrada se mapean a valores normalizado a través de una tabla de valores denominada *Look up Table* y que también viene especificado en el estándar sus valores.

Transformación a escala de grises

En la mayoría de los casos la gama completa de los valores normalizados de la muestra del *pixel* tiene que ser reducido a una gama secundaria que contenga información valiosa para un diagnóstico del estudio realizado. Esto ocurre ya dentro de una misma modalidad donde para representar estructuras específicas en un estudio se requiere de una gama de valores de intensidad del pixel diferentes para cada uno de los tipos de diagnósticos o estructuras a representar.

En el caso, por ejemplo, de estudios de tomografías las estructuras se representan con una gama de valores de intensidad del pixel diferentes de modo que ante un diagnóstico el radiólogo puede moverse de una gama de intensidad a otra según lo requiera la estructura que se está analizando. La función que permite modificar los valores de entrada a valores de salidas se muestra en la siguiente figura (Fig. 15).

$$\begin{cases} Min & I_{in} \le I_1 \\ I_{out} & I_1 < I_{in} \le I_2 \\ Max & I_{in} > I_2 \end{cases}$$

$$donde$$

$$I_1 = C - 0.5 - \left(\frac{W-1}{2}\right)$$

$$I_2 = C - 0.5 + \left(\frac{W-1}{2}\right)$$

$$I_{out} = \left(\frac{(I_{in} - (C-0.5))}{W-1} + 0.5\right) * (Max - Min) + Min$$

Fig.15 Transformación a escala de grises

Esta función depende de dos atributos esenciales que se encuentran agrupados en el módulo *VOI LUT* denominados *Windows Center* y *Windows Width.* De no estar presentes estos valores en el fichero DICOM se pueden determinar a partir de los valores máximo (I_{Max}) y mínimo (I_{Min}) de intensidad del pixel en la imagen.

$$Windows\,Width = I_{Max} - I_{Min}$$

$$Windows\,Center = \frac{I_{Max} + I_{Min}}{2}$$

1.4.2 ANALYZE 7.5

El formato ANALYZE 7.5, fue el formato creado para ser usado en el software de procesamiento de imágenes médicas ANALYZE; es un formato ampliamente usado en el campo de las neuroimágenes funcionales, ha sido implementado por numerosos programas comerciales de procesamiento de imágenes

médicas como son SPM[13], FreeSurfer[14], MRIcro, etc.Entre las principales características del formato ANALYZE 7.5, se encuentran[12]:

- La información se encuentra almacenada en dos archivos con el mismo nombre, un archivo imagen (.img), que posee datos como son el flujo de vóxeles[15], tipo de dato y orden que presentan los mismos, los cuales están descritos por un archivo cabecera (.hdr) Anexo 2, que posee información acerca de las dimensiones de las imágenes, datos del paciente, y otros.
- Es un formato flexible y extensible, pues permite la inclusión de nuevos tipos de datos si el usuario los necesitara.

1.4.3 NifTI

El formato NIfTI 1.1, es el formato propuesto por el DFWG[16] de NIfTI, la estructura del mismo está basada en su gran mayoría en la del formato ANALYZE 7.5, pero NIfTI 1.1 introduce una serie de mejoras enmarcadas sobre todo para facilitar el análisis de datos en las FMRI. Entre las principales características de NIfTI 1.1, se encuentran [13]:

- La información puede ser almacenada ya sea en el formato archivo cabecera-archivo imagen (.hdr Anexo 3 y .img) al igual que en el formato ANALYZE 7.5 o también puede ser almacenada en un archivo único (.nii).
- Mantiene la compatibilidad con programas que utilizan como formato de almacenamiento de imágenes médicas el formato ANALYZE 7.5.

Ha extendido las capacidades que ya existían en el formato ANALYZE 7.5, desde nuevos tipos de datos, códigos y parámetros para lograr una mejor descripción del significado de los datos hasta la modificación del propósito de campos ya existentes en la cabecera del formato ANALYZE 7.5.

[13]Statistical Parametric Mapping: Paquete de software diseñado para el análisis de los datos de secuencias de imágenes del cerebro. Las secuencias pueden ser una serie de imágenes de diferentes cohortes, o series de tiempo de la misma materia.
[14]Freesurfer es un conjunto de herramientas automatizadas para la reconstrucción de la superficie del cerebro de los datos estructurales de la RM y la superposición de datos de resonancia magnética funcional en la superficie reconstruida.
[15]Voxel: Unidad de medida más pequeña en una imagen digital 3D.
[16]Data Format Working Group.

1.5 Estado del Arte

1.5.1 En Cuba

En Cuba el Centro de Biofísica Médica de Santiago de Cuba se dedica al desarrollo de software de visualización médica; éste desarrolla y comercializa el software IMAGIS. También el Centro Internacional de Restauración Neurológica (CIREN) que desarrolla el software STASISS. En ambos casos, los procesos de lectura, escritura y visualización, los realizan utilizando bibliotecas de clases creadas por ellos mismos o utilizan algunas libres.

Con el surgimiento de la Universidad de Ciencias Informáticas la creación de software para la salud se ha potencializado en el país. Muestra de esto es la creación de un sistema PACS que actualmente se instala en algunos hospitales del país y en Venezuela. Este sistema cuenta con un visor de imágenes médicas que para complementar sus funcionalidades utiliza una biblioteca de clases llamada C# DICOM SDK.

1.5.2 En el Mundo

El diagnóstico por imagen es una técnica muy difundida actualmente y existen empresas que se dedican a crear software para apoyar este tipo de técnica.

En la actualidad todas las empresas que se dedican al desarrollo de software de visualización médica utilizan herramientas para soportar sus desarrollos, entre estas se pueden encontrar bibliotecas de clases, SDK[17] y frameworks.

Entre las herramientas propietarias más difundidas actualmente se encuentran:

- LEADTOOLS *Medical Imaging Developer Toolkit*: Con esta herramienta se pueden desarrollar aplicaciones de largo alcance. Entre sus características principales se puede encontrar el soporte para la lectura, comunicación, visualización y procesamiento de imágenes DICOM. Soporta imágenes de 8 y 16 bit, tanto en escala de grises como en colores. Otras de las características principales es que brinda soporte para todas los tipos de compresiones especificadas por el estándar DICOM. Esta herramienta se puede encontrar a un precio de 4.995 dólares por desarrollador [14].

[17] Software Development Kit.

- DICOM C# SDK: Es una biblioteca de clases que implementa parte del estándar DICOM, entre los servicios que implementa se pueden encontrar la lectura, escritura, transmisión y almacenamiento. La licencia de esta herramienta tiene un costo de 6.000 dólares por desarrollador [15].
- Java DICOM Toolkit: Es una biblioteca de clases desarrollada en Java para la construcción de aplicaciones DICOM. Permite que las aplicaciones sean fiables, más eficientemente, librando al programador de tener que lidiar con las complejidades del estándar DICOM. También posibilita la lectura, escritura y transmisión de ficheros DICOM. El costo de la licencia por cada desarrollador es de 495 dólares [16].

También se pueden encontrar soluciones parciales de manipulación de formatos de imágenes médicas con licencia libre; entre estas se encuentran:

- DCMTK: Es una colección de bibliotecas de clases y aplicaciones que implementan gran parte del estándar DICOM; estas incluyen procesamiento, trasmisión, creación y conversión de fichero DICOM. Este conjunto de librerías está escrito con una mezcla de los lenguajes ANSI C y C++. Es utilizado por los hospitales y empresas de todo el mundo para una amplia variedad de propósitos que van desde ser una herramienta para las pruebas, un bloque de construcción para proyectos de investigación, prototipos y productos comerciales [17].
- openDICOM-sharp: Se trata de una implementación completamente nueva de DICOM. En contraste con otras bibliotecas similares, la intención de esta aplicación es proporcionar una clasificación limpia con el apoyo del flujo unidireccional de datos de DICOM. Otra de las características es que soporta DICOM, escribiendo la información de este a un fichero XML; esto no es conforme al estándar, pero muy usado y de gran alcance en el desarrollo de software, almacenamiento y manipulación [18].

1.6 Herramientas y tecnologías utilizadas

1.6.1 Microsoft Visual Studio 2008

Microsoft Visual Studio 2008, es un IDE[18] que plasma la visión de Microsoft acerca de la creación de aplicaciones cliente inteligentes al permitir a los desarrolladores crear de un modo rápido aplicaciones

[18] Integrated Development Environment

conectadas que ofrecen una experiencia de usuario de la máxima calidad. Con Visual Studio 2008, las organizaciones tendrán más fácil que nunca la recopilación y el análisis de información para poder tomar decisiones empresariales eficaces. También permite a las organizaciones, sea cual sea su tamaño, crear de manera rápida aplicaciones más seguras, confiables y fáciles de administrar. Microsoft Visual Studio 2008 ofrece avances fundamentales para desarrolladores en tres áreas principales [19]:

- Desarrollo rápido de aplicaciones.
- Trabajo en equipo eficaz.
- Experiencias de usuario avanzadas.

Microsoft Visual Studio 2008 ofrece herramientas de desarrollo avanzadas, características de depuración, funcionalidad de base de datos y características innovadoras para crear rápidamente las aplicaciones de vanguardia del futuro en una gran variedad de plataformas.

Se decide utilizar este IDE de desarrollo ya que ofrece la visión de las aplicaciones clientes inteligentes, al permitir a los desarrolladores con avanzadas herramientas de desarrollo, y otras características innovadoras, la creación de aplicaciones de manera rápida a través de diversas plataformas.

El sistema se va a desarrollar con el framework 2.0, ya que se considera estable y brinda la posibilidad de migrar a plataformas de software como MONO.

1.6.2 Enterprise Architect 7.5

Enterprise Architect (EA) es una herramienta flexible, completa y potente de modelado en UML bajo plataforma Windows. Provee lo más nuevo en desarrollo de sistemas, administración de proyectos y análisis de negocio.

EA es una herramienta multi-usuario, con bases construidas sobre la especificación UML 2.1 y para soportar UML 2.0. Diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad. Provee una generación poderosa de documentos y herramientas

de reporte con un editor de plantilla completo WYSIWYG[19]. Genera reportes detallados y complejos de EA con la información que se necesita en el formato que su compañía o cliente demanda [20].

1.6.3 Lenguaje de programación C#

C# es un lenguaje propuesto por Microsoft para satisfacer las necesidades actuales y de un futuro cercano. C# es, por tanto, una herramienta, como todos los lenguajes de programación, pero adaptada al trabajo actual. Las principales características que definen al lenguaje C# son [21]:

- Sencillez de uso: C# elimina muchos elementos añadidos por otros lenguajes y que facilitan su uso y compresión, como por ejemplo ficheros de cabecera, o ficheros fuentes IDL.
- Orientado a objetos: C# como lenguaje de última generación es orientado a objetos. Además, C# soporta todas las características del paradigma de la programación orientada a objetos, como son la encapsulación, la herencia y el polimorfismo.
- Orientado a componentes: La propia sintaxis de C# incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular.
- Recolección de basura: Todo lenguaje incluido en la plataforma .NET tiene a su disposición el recolector de basura.
- Compatible: Para facilitar la migración de programadores de C++ o Java a C#, no sólo se mantiene una sintaxis muy similar a la de los dos anteriores lenguajes, sino que también ofrece la posibilidad de acceder a código nativo escrito como funciones sueltas no orientadas a objetos, tales como las DLLs de la API de Win32.

1.6.4 Mono Migration Analyzer

El *Mono Migration Analyzer* (MoMA) es una herramienta que ayuda a identificar los problemas que se encuentran a la hora de portar las aplicaciones realizadas en la plataforma .NET al proyecto MONO. Ayuda a identificar las zonas que aún no están soportadas por el proyecto MONO [22]. Se decide utilizar esta herramienta pues permite a los desarrolladores cuantificar el número de cambios requeridos para ejecutar sus aplicaciones .NET en un entorno Linux.

[19] **WYSIWYG: W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et ("lo que ves es lo que obtienes"). Se aplica a los editores de texto que permiten escribir un documento viendo directamente el resultado final, frecuentemente el resultado impreso.

1.6.5 Metodología de Desarrollo

RUP[20] es una infraestructura flexible de desarrollo de software que proporciona prácticas recomendadas probadas y una arquitectura configurable [23].Es un proceso de desarrollo de software que contiene un conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software. Más que un simple proceso, el proceso de desarrollo de software es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de actitud y diferentes tamaños de proyecto [24].

[20]Rational Unified Process: metodología de desarrollo.

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA

Se presenta la propuesta de solución del framework. Se muestran los procesos del negocio mediante el modelo de dominio y las características y funcionalidades que tendrá el mismo a partir de los requerimientos funcionales y no funcionales. Se presenta el diagrama de casos de uso del sistema, las especificaciones de los casos de uso asociados al componente y la trazabilidad de los casos de uso con sus correspondientes requisitos funcionales.

2.1 Breve descripción del sistema

Debido a las limitaciones que presenta la biblioteca de clases C# DICOM SDK evidenciado en su integración con el sistema alas PACS y la necesidad de poder manipular distintos formatos de representación de imágenes médicas; se propone desarrollar el framework CALIB, el cual permitirá agrupar los procesos de lectura, procesamiento y visualización de los formatos DICOM, ANALYZE 7.5, NifTI en un único formato. Permitiéndole a los sistemas operacionales de procesamiento y análisis de imágenes médicas acceder de manera uniforme a todos los formatos anteriormente mencionados, abstraerse de una serie de pasos demasiados engorrosos como son la lectura de distintos formatos y así mismo desarrollar algoritmos de procesamiento y visualización sobre distintos formatos. Tiene en cuenta además los algoritmos de comprensión de imágenes que puede tener cada formato de imágenes durante el proceso de visualización.

2.2 Modelo de Dominio

Con el objetivo de comprender el ambiente o entorno en el cual está enmarcado el problema que se desea resolver, así como la estructura y la dinámica de la organización e identificar las mejoras potenciales que se pueden lograr, RUP establece en su primera fase de desarrollo la realización del modelo de negocio [1]. Cuando los procesos del negocio no son claramente identificables y no tienen fronteras muy bien establecidas, RUP propone realizar un modelo de dominio.

El modelo de dominio se caracteriza por identificar los objetos más importantes dentro del contexto del sistema, relacionándolos en un diagrama. Estos objetos constituyen conceptos existentes ya en el entorno en que trabaja el sistema y que están bien definidos. El modelo de dominio se describe en diagramas de clases, utilizando el lenguaje UML.

Teniendo en cuenta que los procesos del negocio no son claramente visibles, el negocio se describe utilizando el modelo de dominio (Fig. 16).

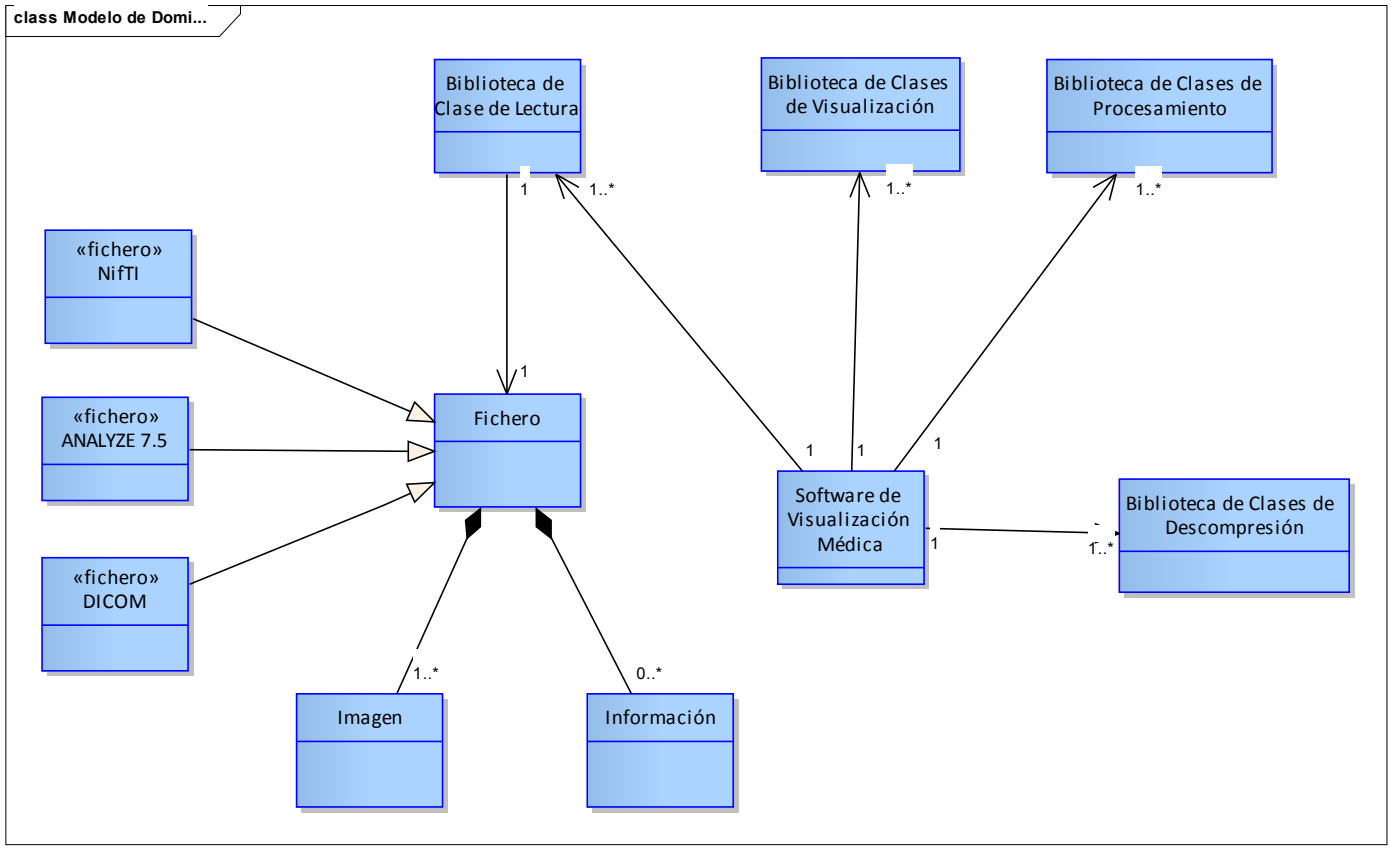


Fig.16 Modelo de Dominio

Imagen: Representa la imagen o las imágenes contenidas en el fichero.

Fichero: Generalización de los ficheros de imágenes médicas, del cual heredan los diferentes formatos (DICOM, NifTI, ANALIZE 7.5).

Información: Datos adicionales que contiene el fichero (nombre, edad, sexo del paciente, cantidad de imágenes, tipo de imagen, etc.).

Biblioteca de clases de lectura: Permite interpretar el juego de datos contenido en un fichero.

Biblioteca de clases de visualización: Permite decodificar los datos de la imagen contenida en el fichero, para poder mostrarlas en el visor.

Biblioteca de clases de procesamiento: Permite aplicar transformaciones a los datos de la imagen contenida en el fichero.

Biblioteca de Clases de Descompresión: Permite interpretar los datos comprimidos en un fichero, y decodificarlos para su posterior visualización.

Software de Visualización Médica: Software que permite visualizar y diagnosticar sobre imágenes médicas.

2.3 Especificación de los Requisitos de Software

Los requisitos funcionales y no funcionales muestran las capacidades o condiciones que el sistema debe cumplir y las propiedades o cualidades que el producto debe tener respectivamente, los cuales en la fase de construcción deben ser medibles.

2.3.1 Requisitos funcionales

- RF 1. Leer fichero.
  - RF 1.1. Leer fichero DICOM.
  - RF 1.2. Leer fichero ANALIZE 7.5.
  - RF 1.3. Leer fichero NifTI.
- RF 2. Post-procesar imágenes.
  - RF 2.1. Aplicar filtros de inversión de los niveles de gris.
  - RF 2.2. Filtrar imágenes RGB a niveles de gris.
  - RF 2.3. Filtrar imágenes por filtros de suavizado.
  - RF 2.4. Filtrar imágenes por filtros paso bajo y paso alto.
- RF 3. Descomprimir.
  - RF 3.1. Descomprimir RLE.
  - RF 3.2. Descomprimir JPEG Lossless J, Non-Hierarchical (Process 14).
  - RF 3.3. Descomprimir JPEG Lossless, Non-Hierarchical, First-Order Prediction.
  - RF 3.4. Descomprimir JPEG Extended.

RF 4. Pre-procesar imagen.

RF 4.1. Aplicar Transformación Lineal.

RF 4.2. Aplicar Transformación a los bits.

RF 5. Visualizar imágenes.

RF 5.1. Visualizar imágenes de 8 bits por pixel.

RF 5.2. Visualizar imágenes de 16 bits por pixel.

RF 5.3. Visualizar imágenes en escala de grises.

RF 5.4. Visualizar imágenes en RGB.

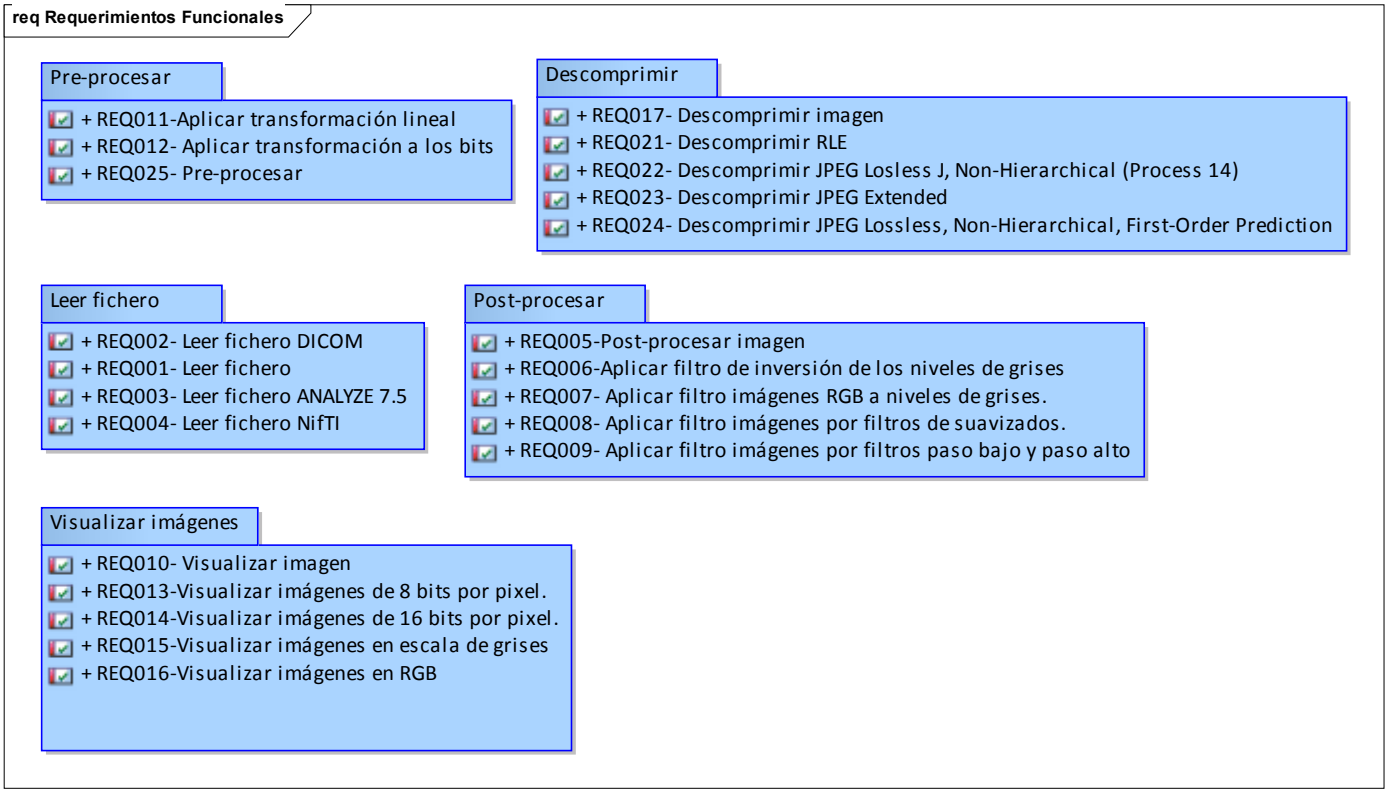


Fig.17 Diagrama de requerimientos funcionales

2.3.2 Requisitos no funcionales

Funcionamiento:

RNFO 1. Sistema Operativo Windows XP, superior o GNU/Linux.

RNFO 2. Framework 2.0, superior o MONO 2.4.

RNFO 3. Procesador: Pentium a 400 MHz o equivalente (mínimo); procesador Pentium a 1 GHz o equivalente (recomendado).

RNFO 4. Memoria RAM: 96 MB (mínimo); 256 MB (recomendado).

RNFO 5. Disco duro: se pueden necesitar hasta 500 MB de espacio disponible.

Usabilidad:

RNU 1. Fácil acceso a las funcionalidades más utilizadas: lectura, procesamiento y visualización.

Eficiencia:

RNE 1. Permitir la lectura de imágenes en tiempo menor a un segundo.

Legal:

RNL 1. El framework y la documentación generada por el mismo son propiedad del Departamento de Software Médico Imagenológico (SWMI).

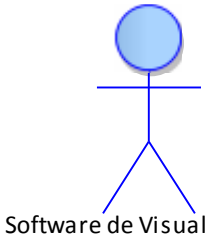
Seguridad:

RNS 1. El framework debe utilizar la infraestructura de llave pública y privada en bien de la integridad de sus componentes.

Interconexión:

RNI 1. El framework debe permitir agregar nuevos lectores sin necesidad de modificar su núcleo.

2.4 Actores del Sistema

Actor	Descripción
 Software de Visualización Médica	Es el programa que interactúa con el framework, utilizando las diversas funcionalidades que este brinda para apoyar y dar cumplimiento a sus funciones.

2.5 Diagrama de Caso de Uso del sistema

Un diagrama de casos de uso del sistema representa gráficamente los procesos y su interacción con los actores, describiendo lo que hace el sistema para cada tipo de usuario.

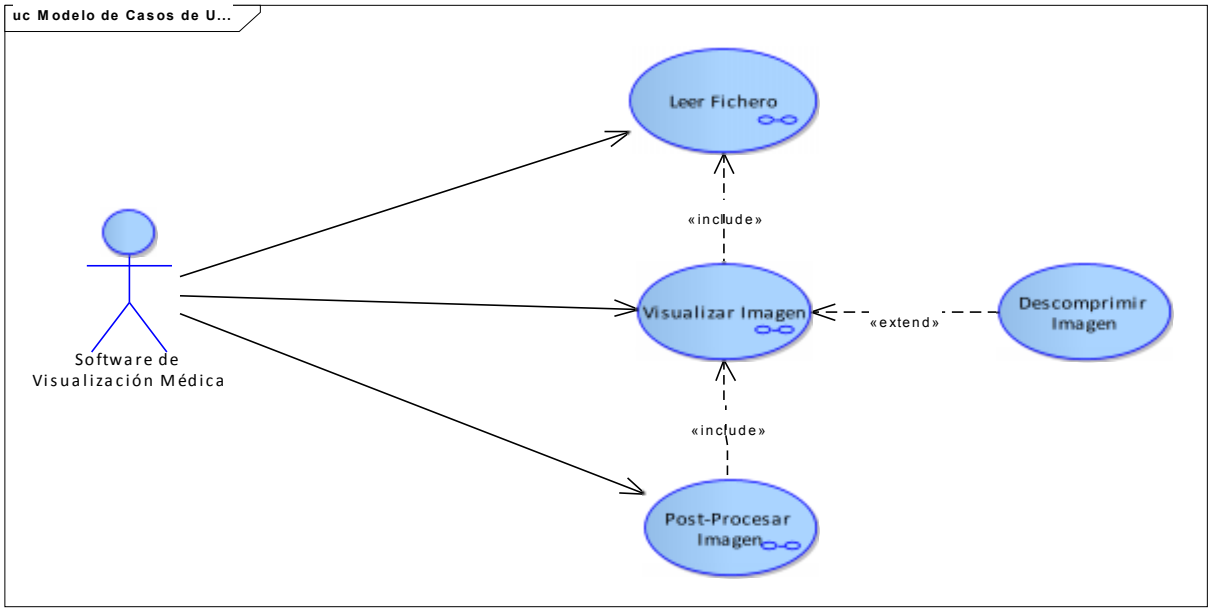


Fig.18 Diagrama de Casos de Uso

2.5.1 Caso de Uso Leer fichero

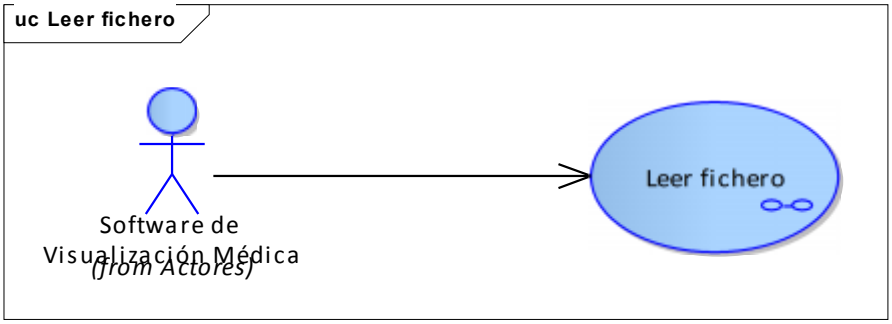


Fig.19 CU Leer Fichero

Caso de Uso	Leer fichero.
Propósito	Interpretar el flujo de datos que compone al fichero con el objetivo de realizar alguna función.
Actores	Software de Visualización Médica.
Resumen	Se comienza por interpretar el formato del fichero al que se le desea dar lectura, y luego con el lector correspondiente se interpretan y se almacenan los datos con el objetivo de realizar alguna acción a partir de los mismos.
Referencias	RF1.1, RF1.2, RF1.3, RF1.4

Acción del Actor	**Respuesta del Sistema**
1. El software de visualización pide el servicio de lectura al framework dándole la dirección del archivo.	**2.** Interpreta el formato del fichero. 2.1. Extrae la extensión del fichero (en caso de que el fichero no contenga extensión ver Flujo Alterno 1). 2.2. Se cargan los lectores. 2.3. Se busca el lector que interprete la extensión obtenida.
	3. Se lee el archivo utilizando el lector genérico correspondiente.
Flujos Alternos:	

1-Fichero sin extensión	
	1. Se prueba a leer el fichero con cada uno de los lectores. En caso de no encontrar ningún lector que interprete el archivo se muestra un mensaje de formato no reconocido.

Fig.20 Descripción extendida del Caso de Uso Leer fichero

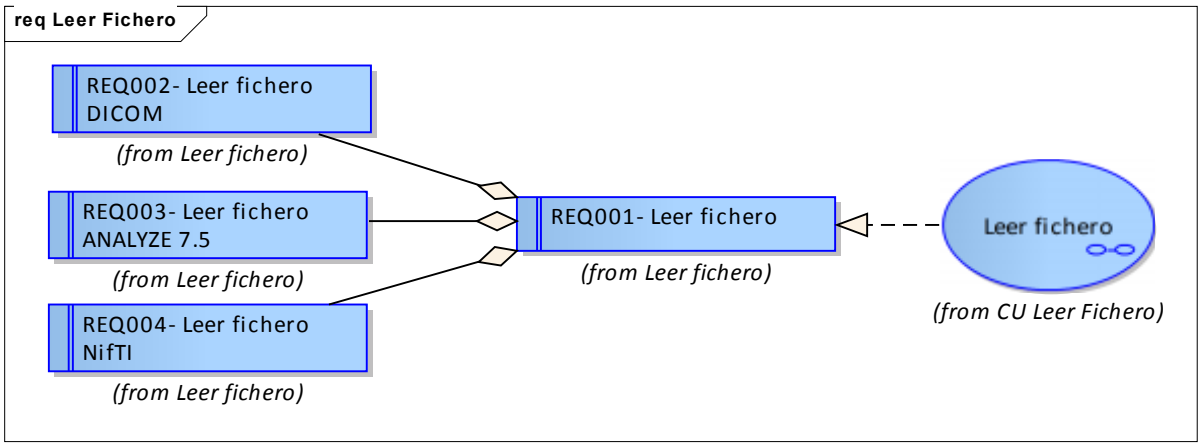


Fig.21 Trazabilidad CU Leer Fichero

2.5.2 Caso de Uso Post-procesar imagen

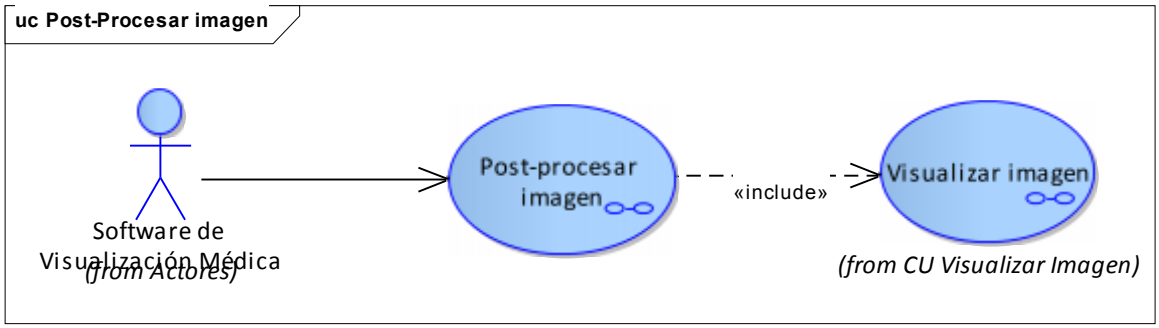


Fig.22 CU Post-procesar imagen

Caso de Uso	Post-procesar imagen.	
Propósito	Aplicar transformaciones a la imagen después de haber sido creada.	
Actores	Software de Visualización Médica.	
Resumen	Se escoge una imagen y se le aplica algún tipo de post-procesamiento.	
Referencias	RF 4.1, RF 4.2, RF 5.1, RF 5.2, RF 5.3, RF 5.4	
Acción del Actor :		**Respuesta del Sistema:**
1. Pide el servicio de "Post-Procesar Imagen" al framework dándole la imagen a transformar.		**2.** Selecciona el algoritmo de acuerdo al pixel type de la imagen.
3. Selecciona el tipo de Post-procesamiento a aplicar.		**4.** Aplica la transformación seleccionada.
5. Muestra imagen transformada.		
Postcondiciones	Se aplica una transformación a la imagen.	

Fig.23 Descripción extendida del Caso de Uso Post-procesar imagen

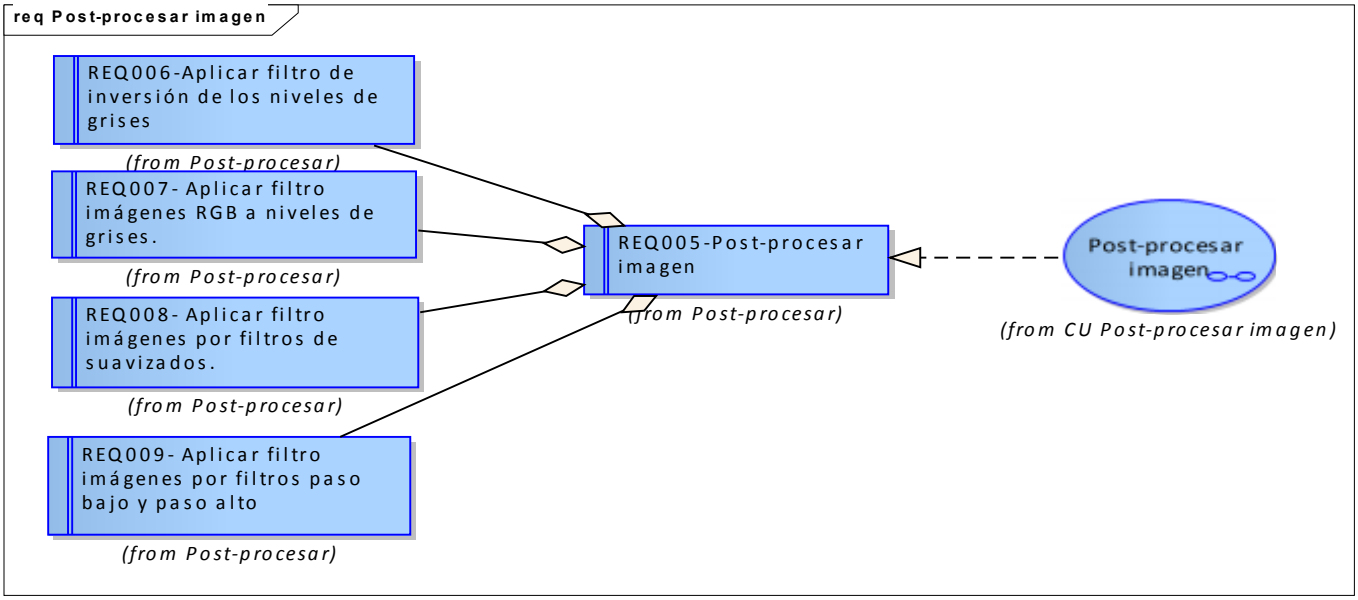


Fig.24 Trazabilidad CU Post-procesar imagen

2.5.3 Caso de Uso Visualizar imagen

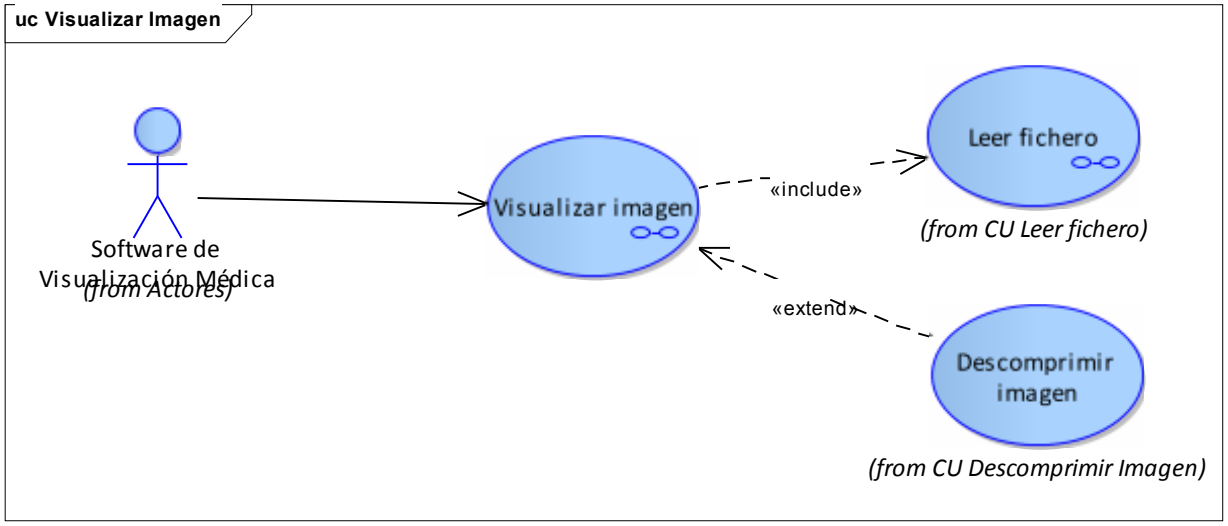


Fig.25 CU Visualizar Imagen

Caso de Uso	Visualizar imagen.
Propósito	Visualizar una imagen.
Actores	Software de Visualización Médica.
Resumen	Se escoge la imagen que se desea visualizar y se muestran en el visor, así como las opciones referentes a realizar cambios a la imagen.
Referencias	RF 4.1, RF 4.2, RF 5.1, RF 5.2, RF 5.3, RF 5.4

Acción del Actor:	**Respuesta del Sistema:**
1. El software de visualización pide el servicio de visualización al framework dándole la dirección del archivo que se desea mostrar el en visor.	**2.** Se lee el archivo utilizando el lector genérico correspondiente.
3. El software pide los *frames* que desea visualizar.	**4.** Devuelve un archivo CalibImage con el o los frames solicitados.
5. Muestra la o las imágenes en el visor. 5.1. Las imágenes pueden ser mostradas	

con ancho y centro de ventana o con paletas.	
Postcondiciones	Se muestra una imagen.

Fig.26 Descripción extendida del Caso de Uso Visualizar imagen

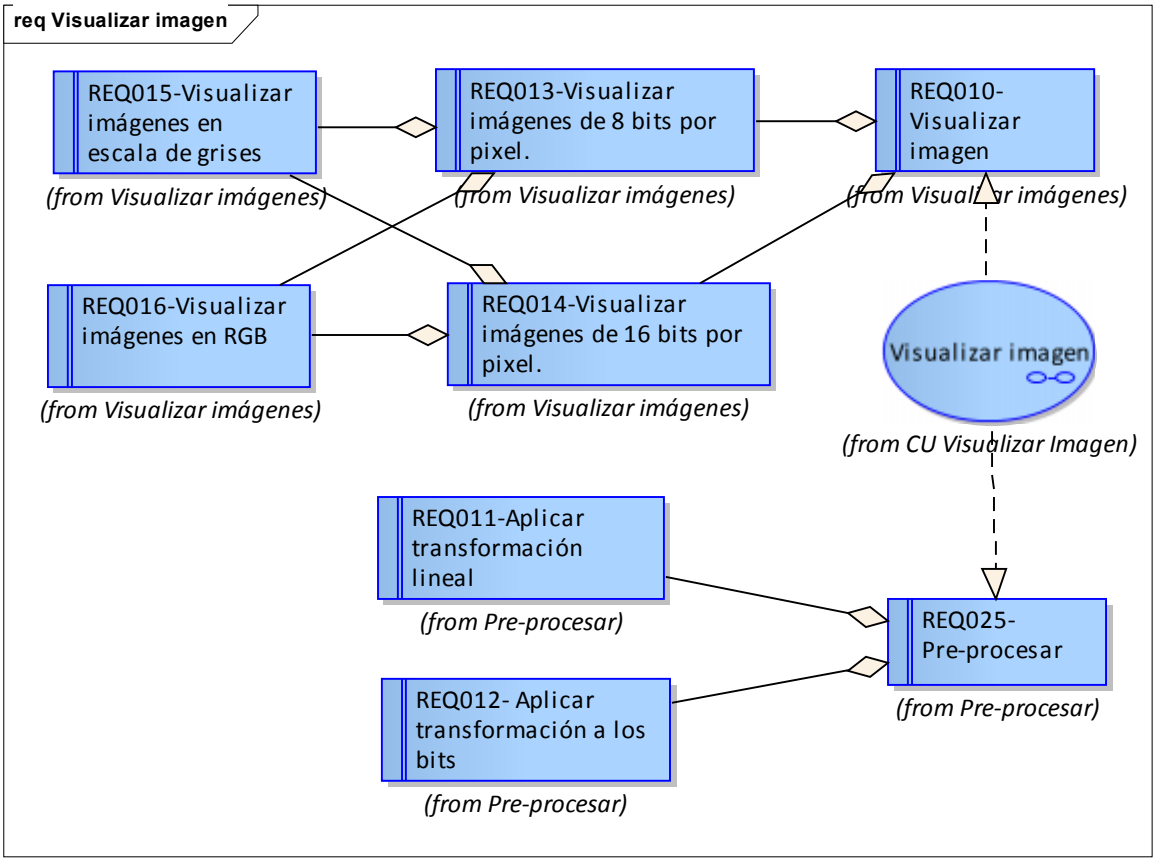


Fig.27 Trazabilidad CU Visualizar Imagen

2.5.4 Caso de Uso Descomprimir imagen

Caso de Uso:	Descomprimir Imagen (Extend)
Propósito:	Descomprimir una imagen con el objetivo de visualizarla.
Resumen:	Se verifica si el flujo de datos correspondiente a la imagen está comprimido

	en alguno de los formatos soportados, si lo está, se procede a descomprimirlo.
Referencias	RF 3.1, RF 3.2, RF 3.3, RF 3.4
Precondiciones	Existe una imagen comprimida que se desea visualizar.
Acciones del Sistema:	
1. Descomprimir Imagen. 1.1. Se comprueba el formato de compresión que contiene la imagen. 1.2. Se cargan los codecs de descompresión. 1.3. Se busca el códec correspondiente al formato de compresión. 1.4. Se procede a descomprimir la imagen.	
Postcondiciones	La imagen queda lista para ser visualizada.

Fig.28 Descripción extendida del Caso de Uso Descomprimir imagen

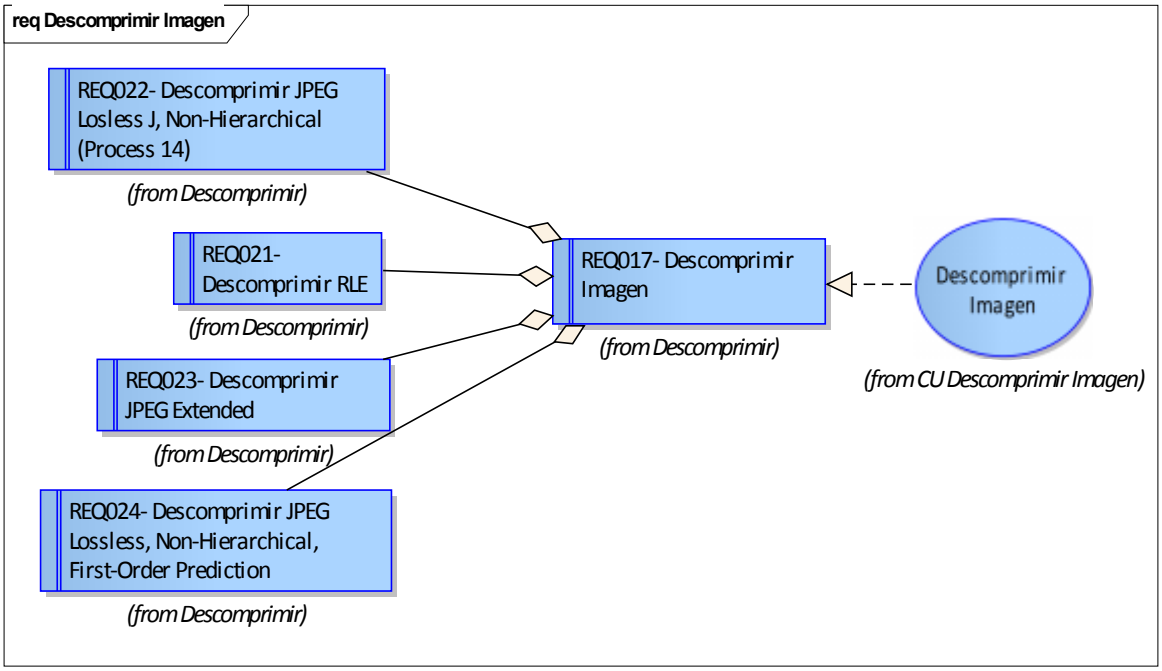


Fig.29 Trazabilidad CU Descomprimir Imagen

CAPÍTULO 3. DISEÑO DEL FRAMEWORK

Se describe cómo implementar el framework, a través del diseño, enfocado a cómo el sistema cumple sus objetivos teniendo en cuenta los requisitos funcionales y no funcionales. Se realizan los diagramas de clases y los diagramas de interacción según los casos de uso definidos y se explica además la arquitectura utilizada.

3.1 Estilo arquitectónico utilizado

Internamente el framework está distribuido en capas, una de negocio y otra de servicios. La capa de negocio contiene las funcionalidades que dan cumplimiento a los requerimientos funcionales ya definidos. La capa de servicios contiene un controlador de plugins, que se encarga de gestionar las funcionalidades que brindan los mismos, e interfaces implementadas por estos plugins.

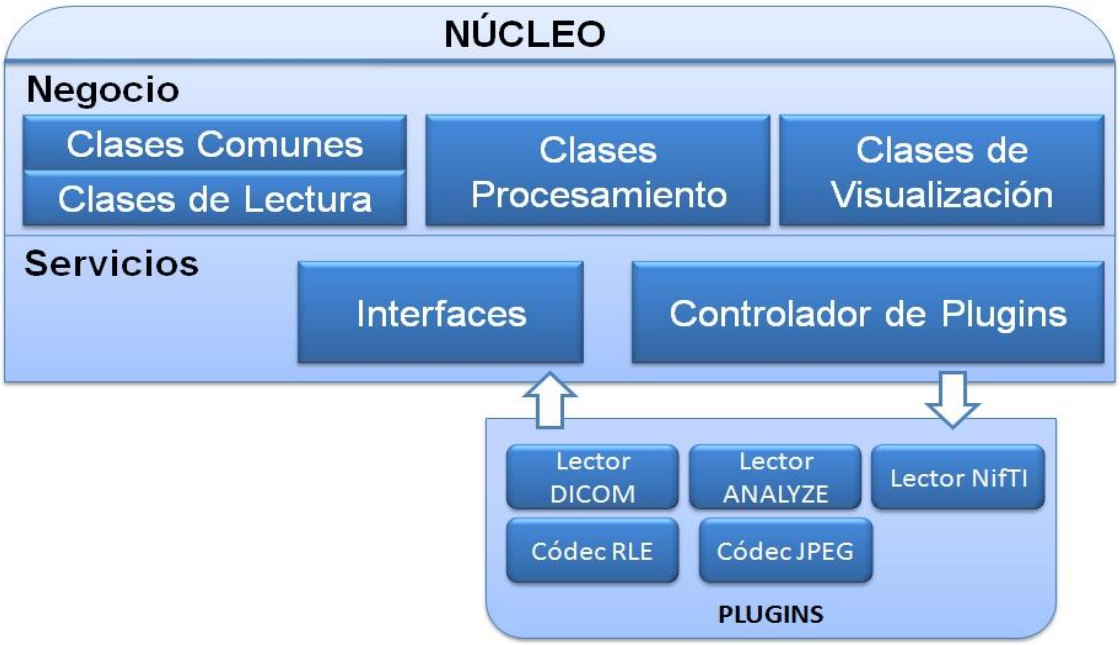


Fig.30 Capas del Framework

La base arquitectónica del framework está soportada sobre una arquitectura basada en plugins. Este estilo arquitectónico permite que el framework desarrollado cumpla con las siguientes características:

- Isolation: Cada plugin debe cargarse y correr en un AppDomain separado. El plugin no debe tener acceso al host, es decir, la comunicación es unidireccional: HOST→PLUGIN.
- Tolerancia o manejo de fallos: Cuando un plugin no funciona o lanza excepciones no esperadas debe descargarse.
- Seguridad: El plugin debe tener los mínimos privilegios necesarios para correr. Estos privilegios varían según el tipo de aplicación que se realice.
- Detección de nuevos plugins o nuevas versiones de los instalados: No es buena idea tener que reiniciar una aplicación o un servicio para instalar un nuevo plugin.

Para lograr que cada plugin corra en un AppDomain separado del AppDomain del HOST es necesario desacoplar en dos partes el sistema, ambas deben conocer y respetar ciertos contratos, de modo que las dos partes deben compartir un conjunto de interfaces. Este patrón que nos permite desacoplar el sistema se llama Service Interface (Fig. 31).

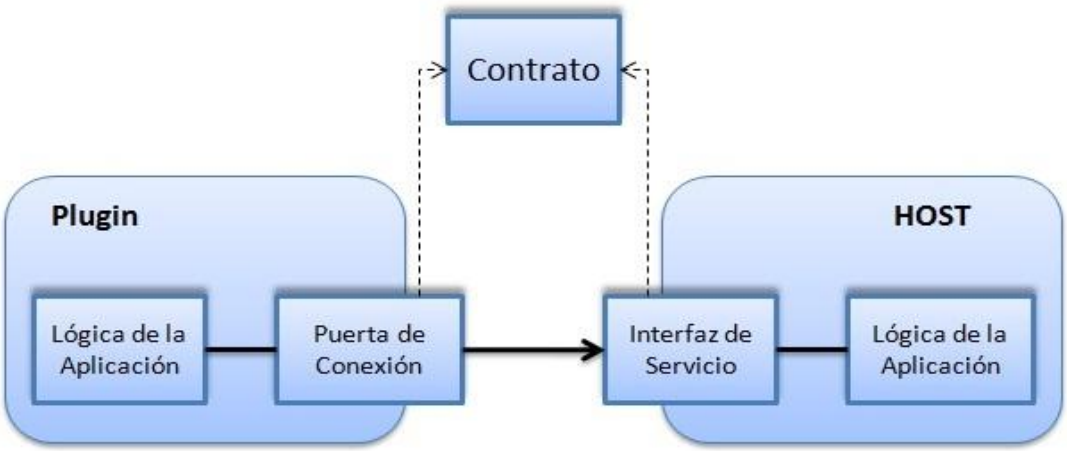


Fig.31 Patrón Interfaz de Servicio (Service Interface).

3.2 Diagramas de Clases

Un diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces de la aplicación. Además visualiza las relaciones entre las clases que se involucran en el sistema.

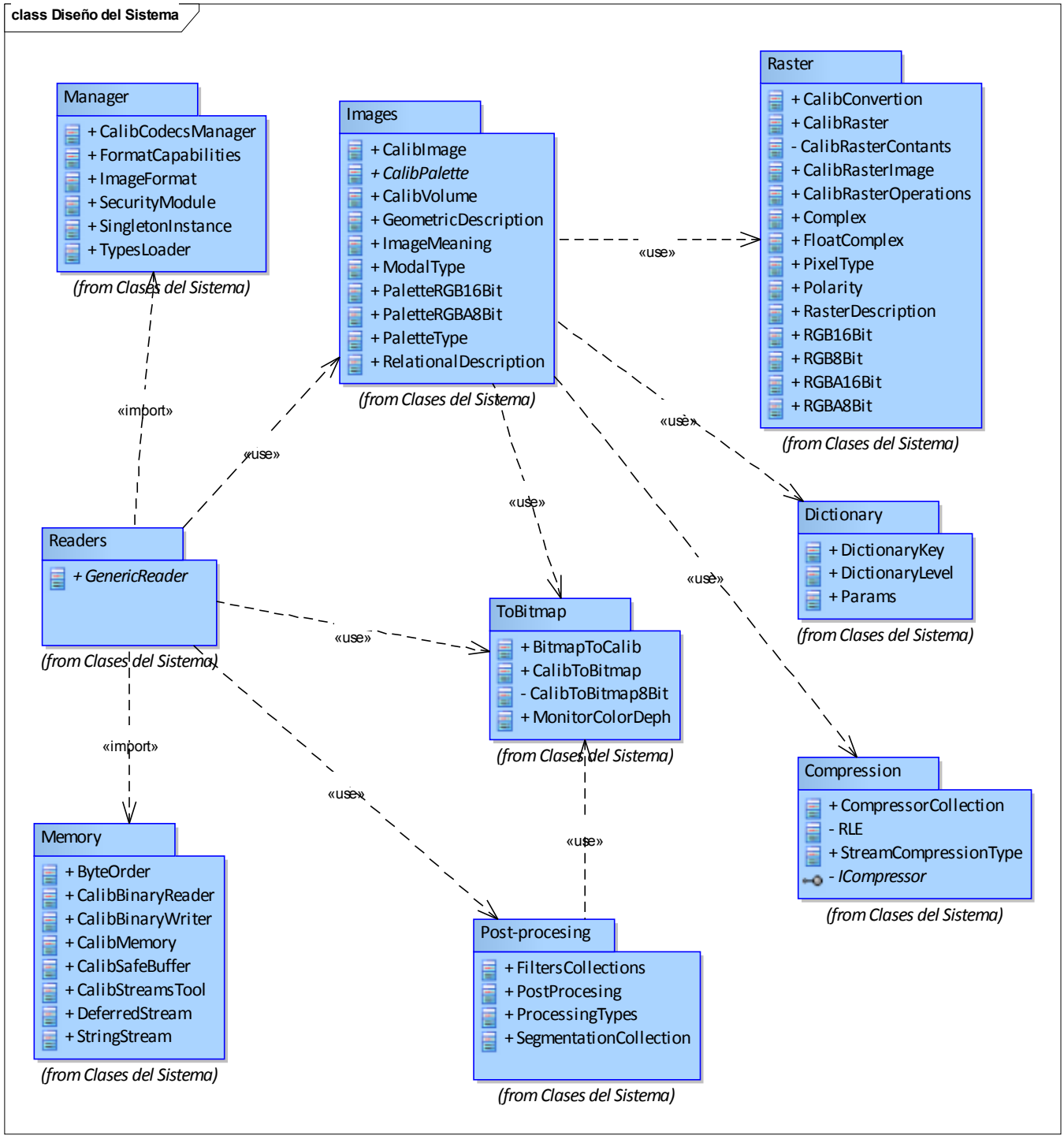


Fig.32 Diagrama de paquetes del framework

El diseño de la aplicación por casos de uso se puede apreciar en las figuras: Fig. 33 hasta la Fig. 36.

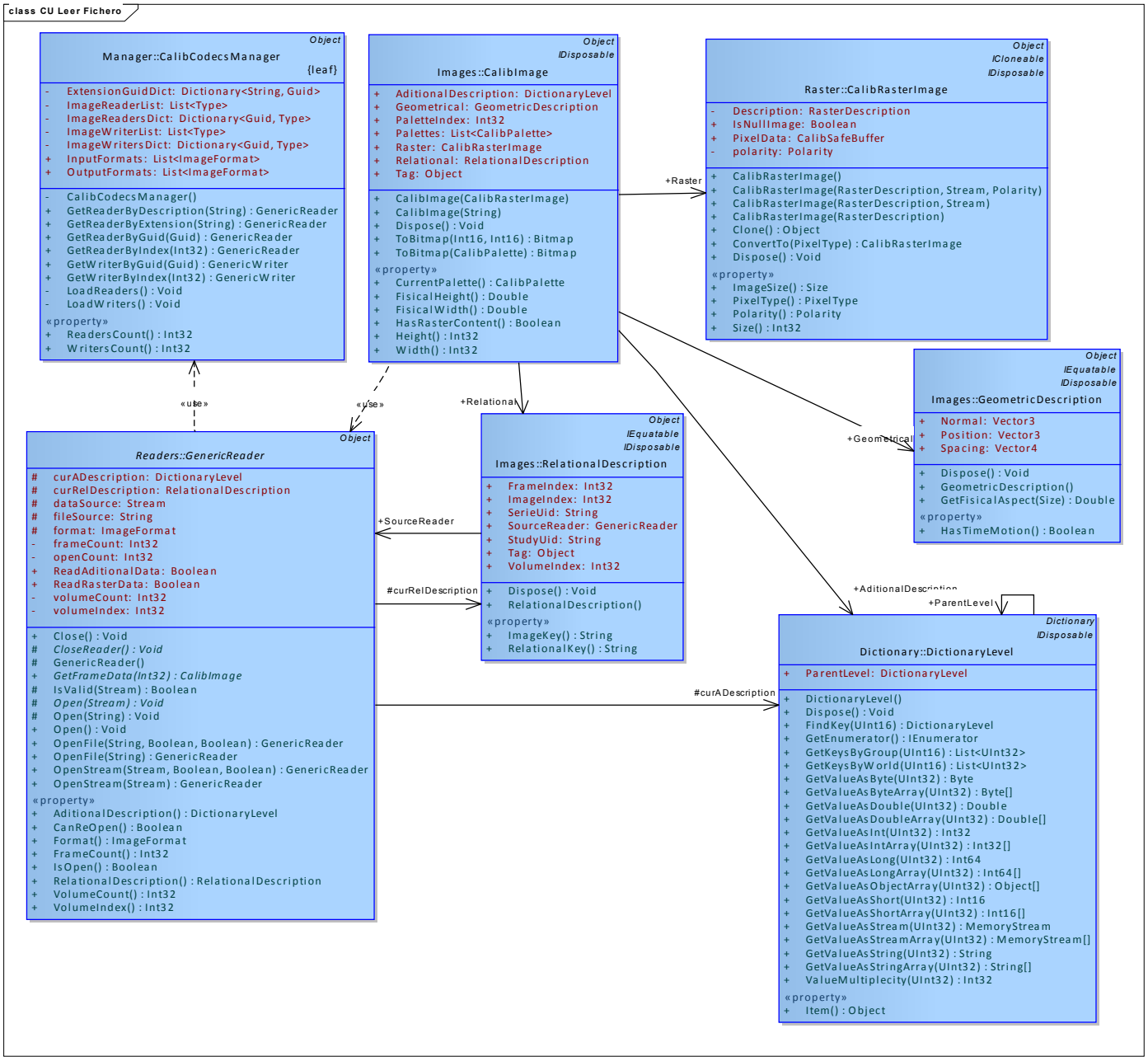


Fig.33 CU Leer fichero

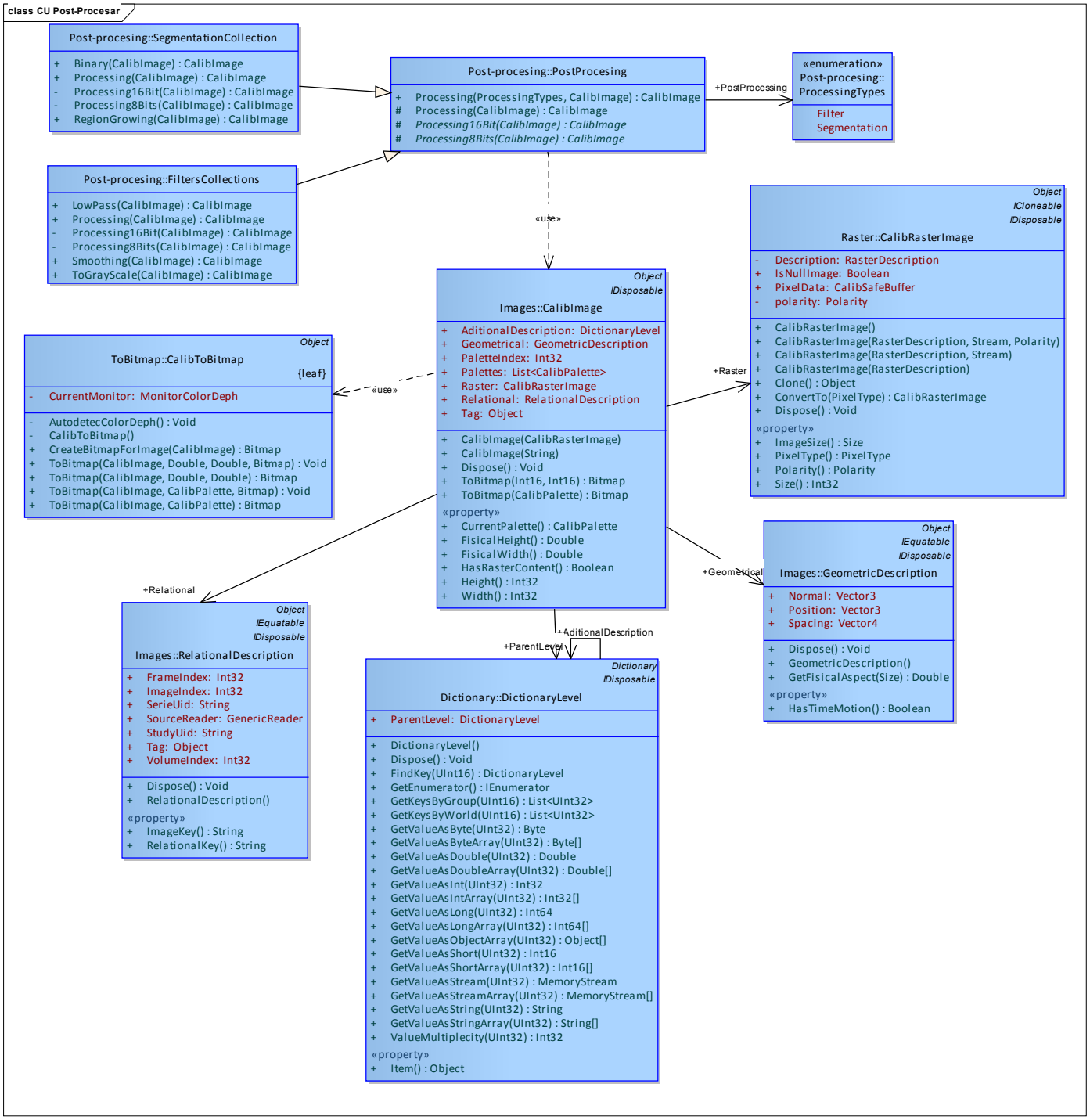


Fig.34 Post-procesar imagen

Fig.35 CU Visualizar imagen

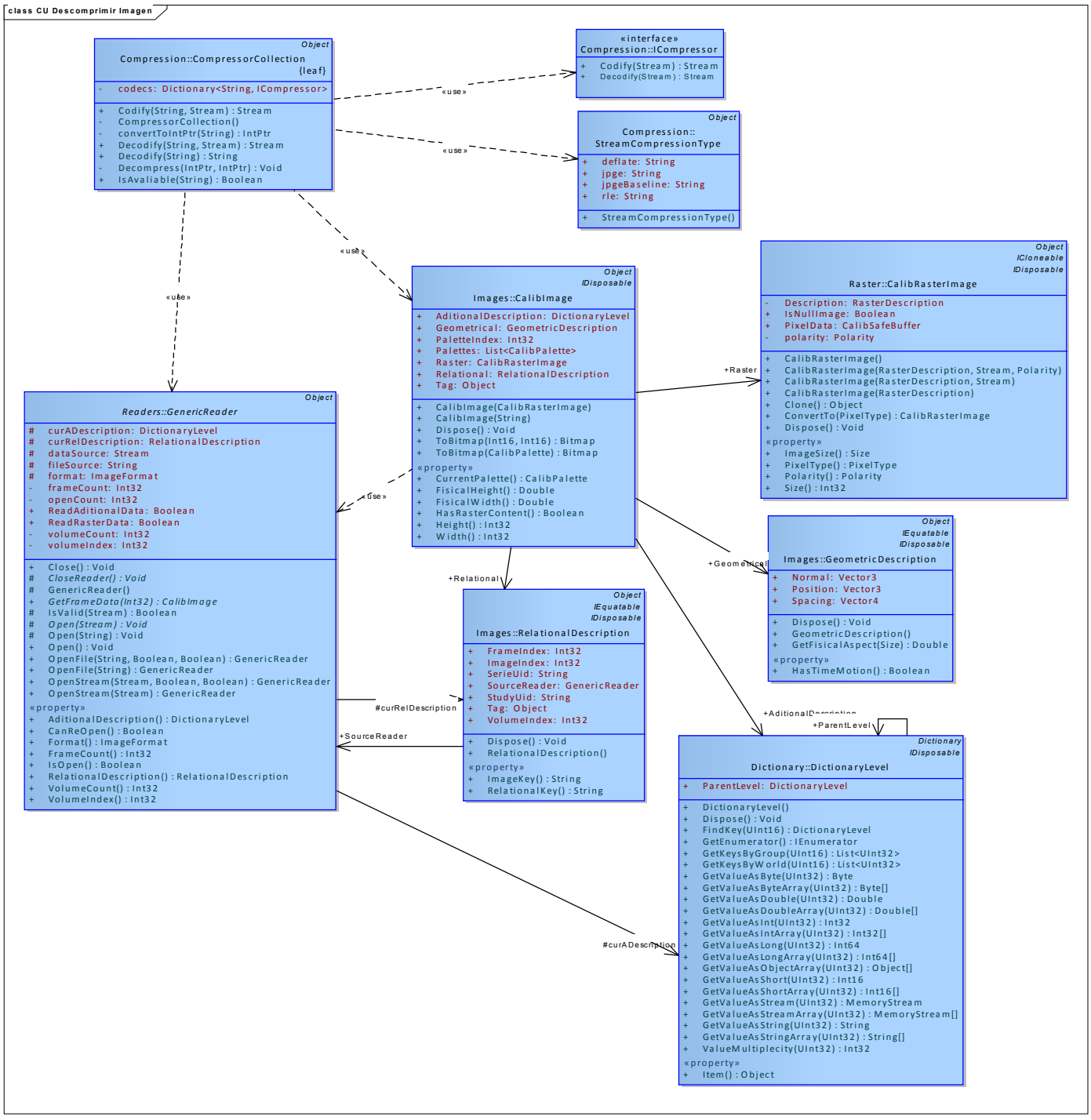


Fig.36 CU Descomprimir imagen

3.3 Diagramas de Secuencia

Un diagrama de secuencia muestra las interacciones entre objetos ordenadas en secuencia temporal; muestra los objetos que se encuentran en el escenario y la secuencia de mensajes intercambiados entre los objetos para llevar a cabo la funcionalidad descrita por el escenario.

Los diagramas de secuencia guían al programador durante la fase de implementación. A continuación se exponen los mismos:

Diagrama de Secuencia	Descripción
Leer fichero (con extensión)	Se interpreta la extensión del fichero, se busca un lector que lo decodifique y se interpretan los datos, si no se encuentra un lector se prueba con todos los que hay disponibles.
Leer Fichero (sin extensión)	Como no se dispone de una extensión que indique el formato del fichero, se prueba a acceder a los datos con cada uno de los lectores hasta encontrar uno que los interprete.
Visualizar imagen (con Ancho y Centro)	Se interpreta el juego de datos del archivo (lectura), se obtienen los frames que se desean visualizar y se muestran con ancho y centro en el visor.
Visualizar imagen (con Paleta)	Se interpreta el juego de datos del archivo (lectura), se obtienen los frames que se desean visualizar y se muestran con paleta en el visor.
Descomprimir imagen	Se interpreta el juego de datos del archivo (lectura), si la o las imágenes están comprimidas, se carga el compresor correspondiente y se procede a descomprimirlas con el objetivo de visualizarlas.
Post-procesar imagen	Se aplica alguna transformación a la imagen para luego visualizarla o bien mientras se visualiza.

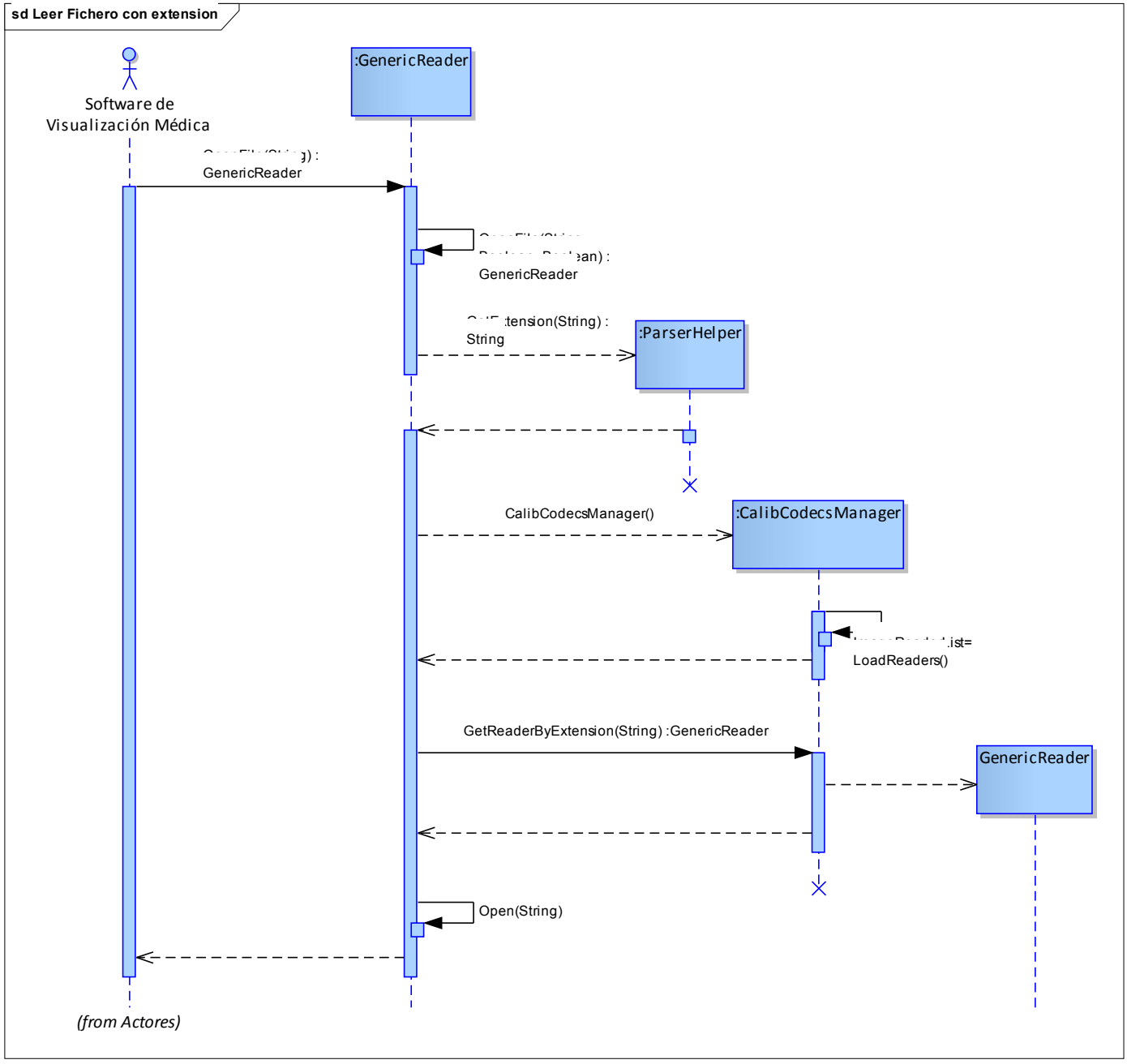


Fig.37 Diagrama de secuencia de CU Leer fichero(con extensión)

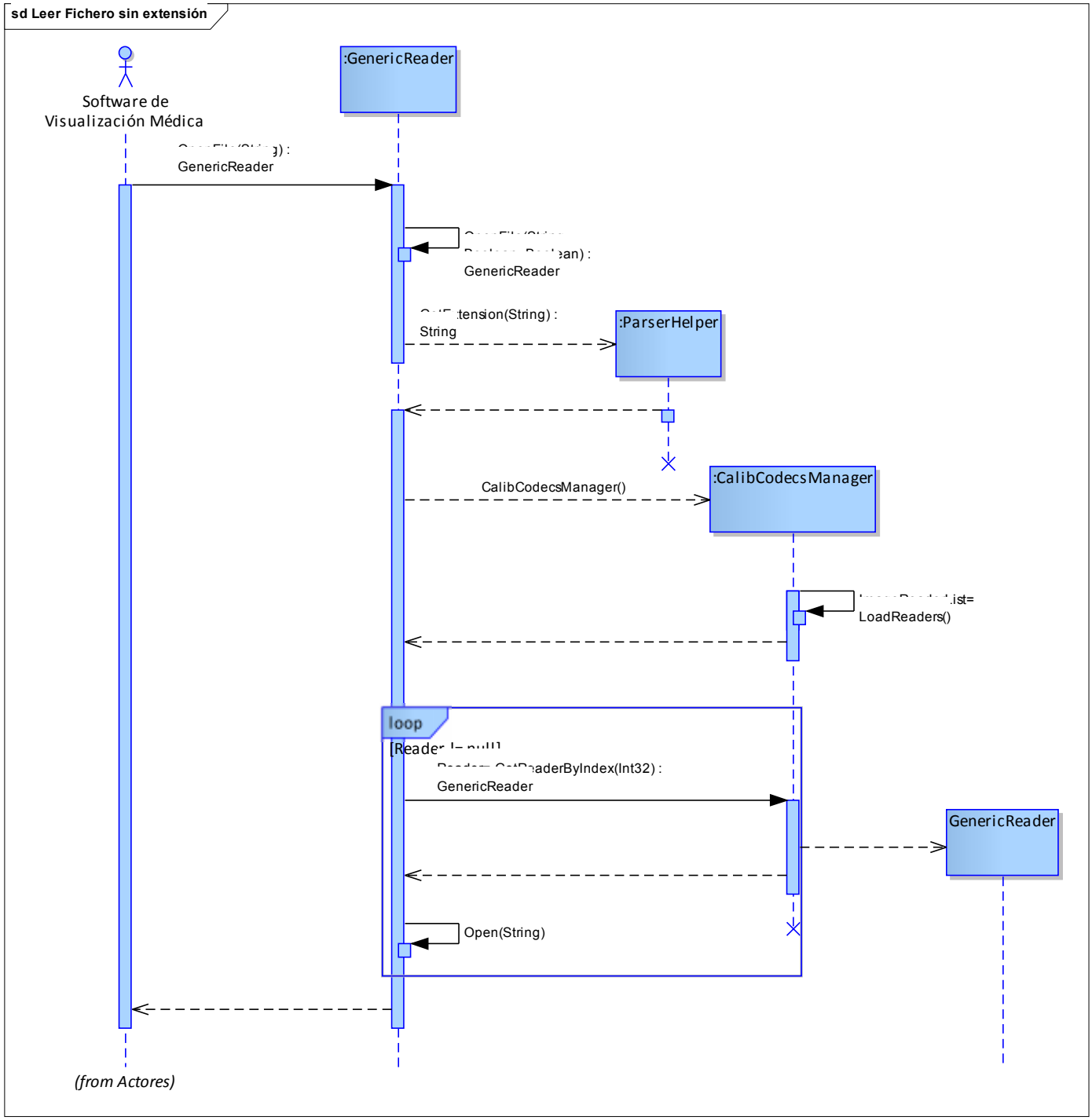


Fig.38 Diagrama de secuencia de CU Leer fichero(sin extensión)

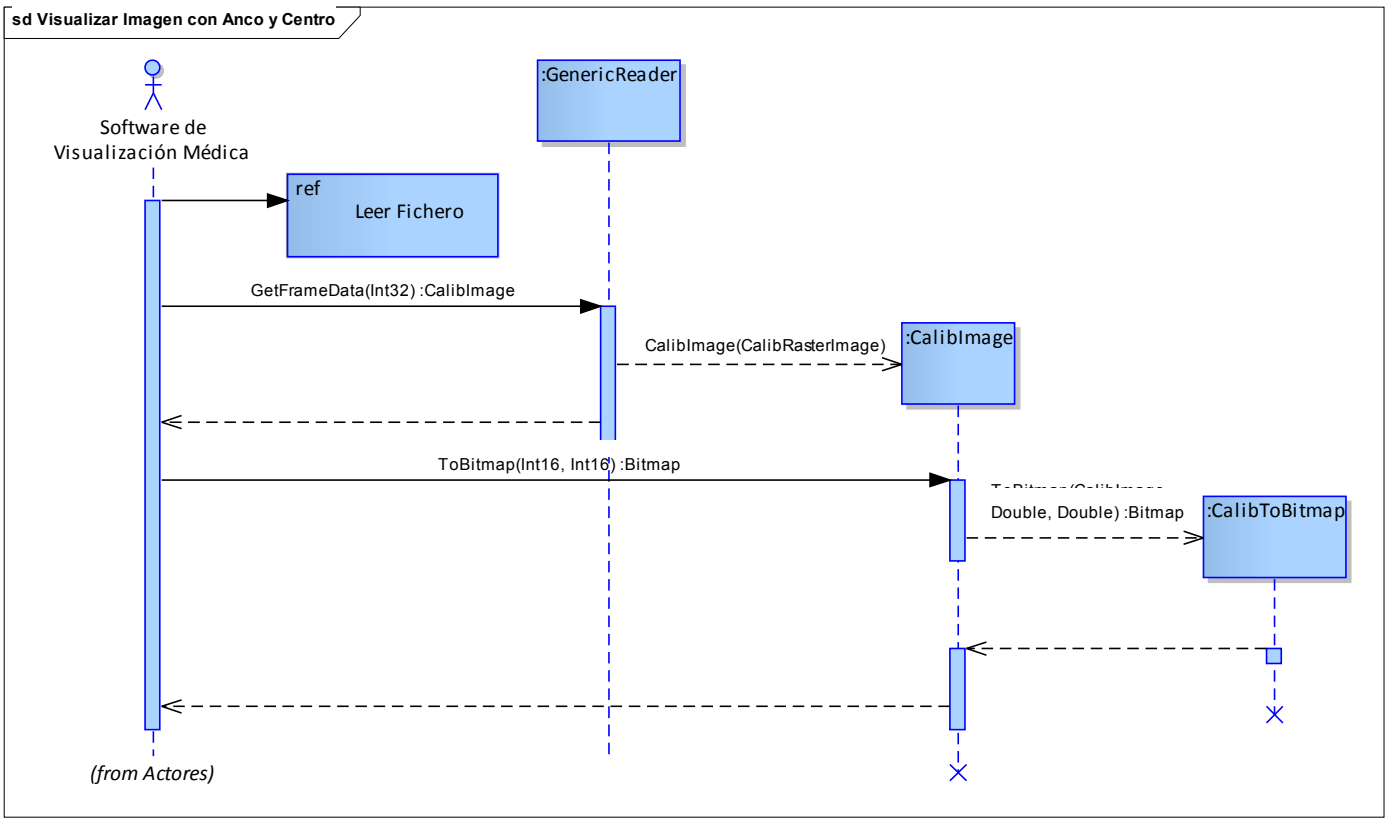


Fig.39 Diagrama de secuencia de CU Visualizar imagen (con Ancho y Centro)

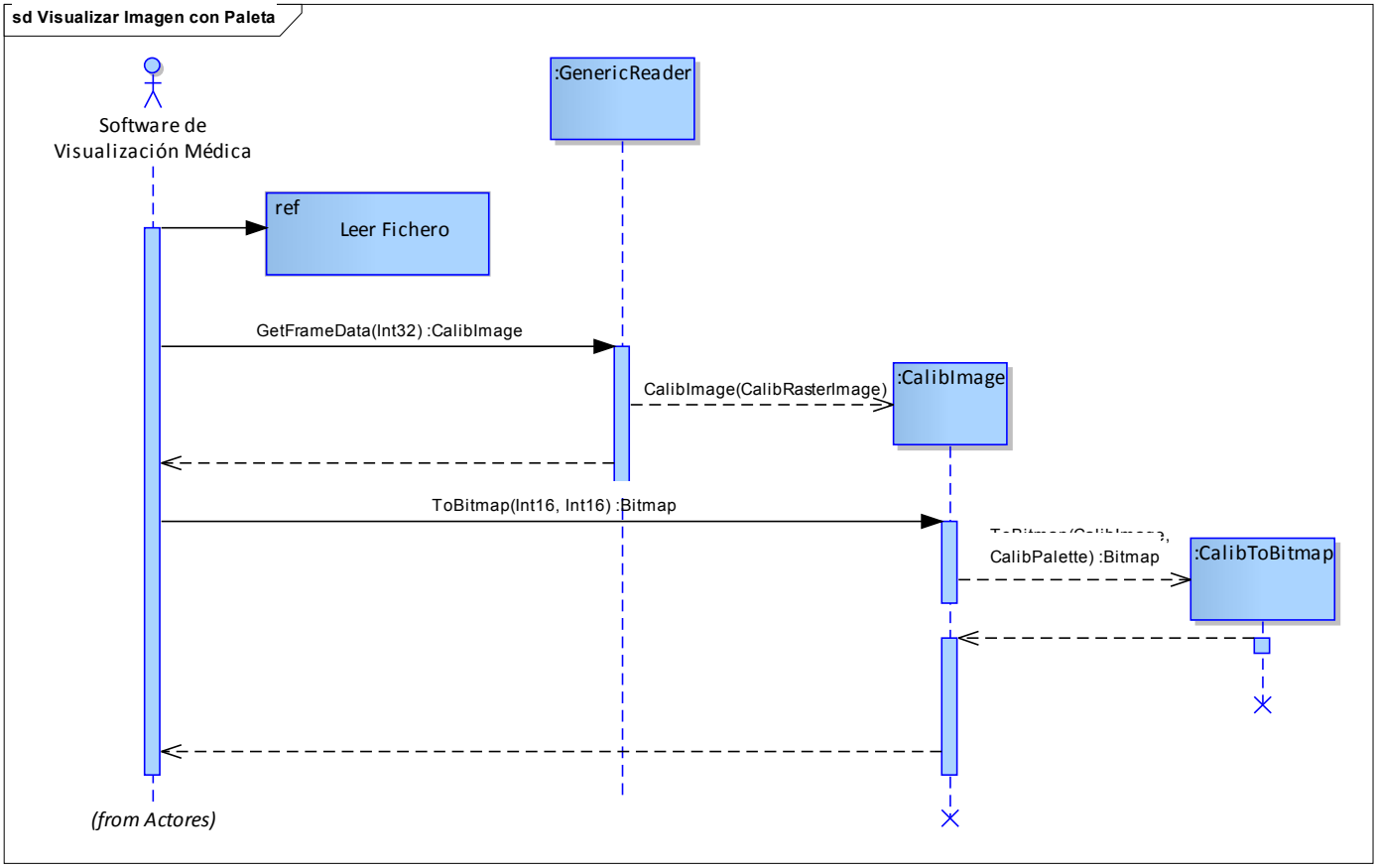


Fig.40 Diagrama de secuencia de CU Visualizar imagen (con Paleta)

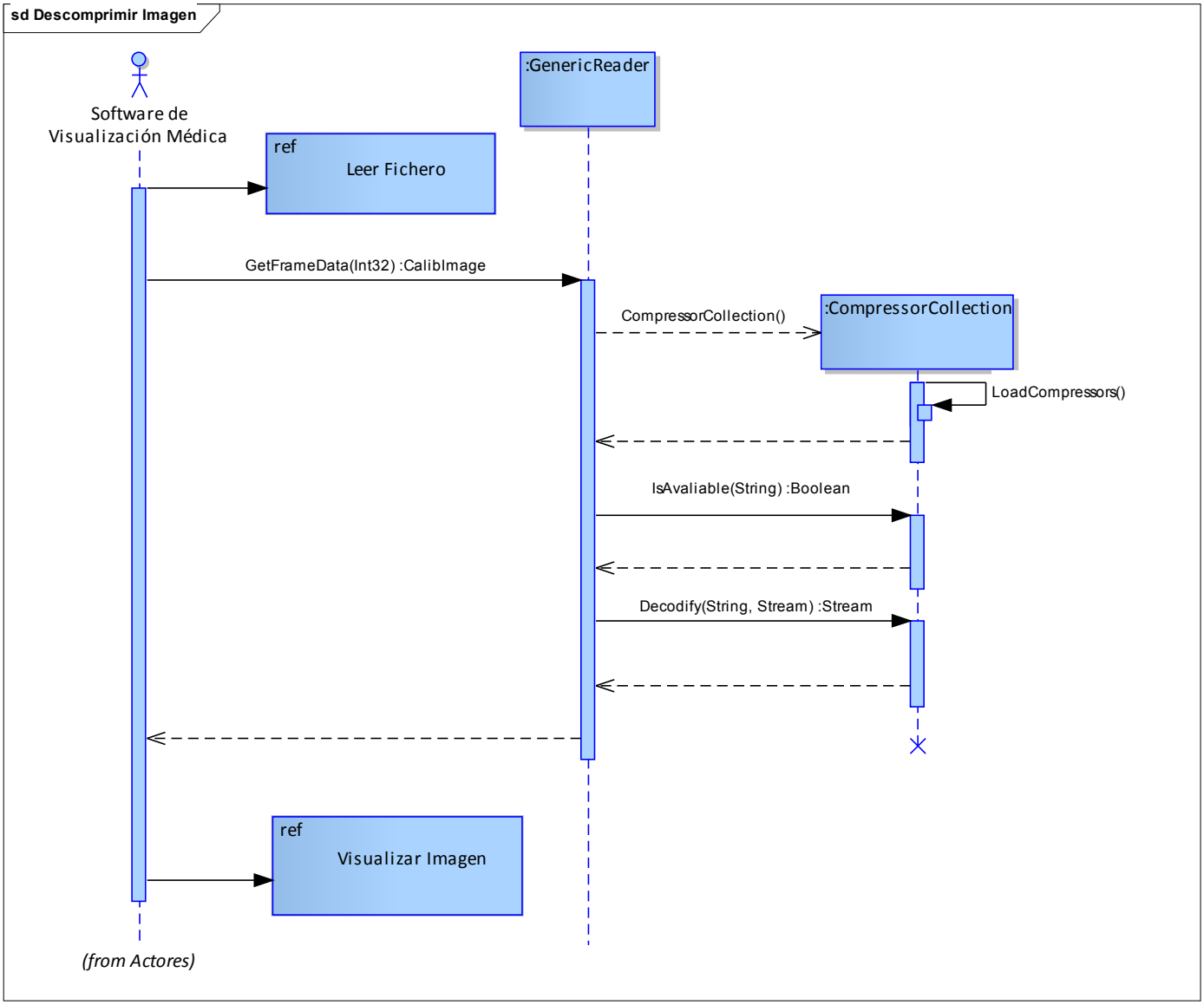


Fig.41 Diagrama de secuencia de CU Descomprimir imagen

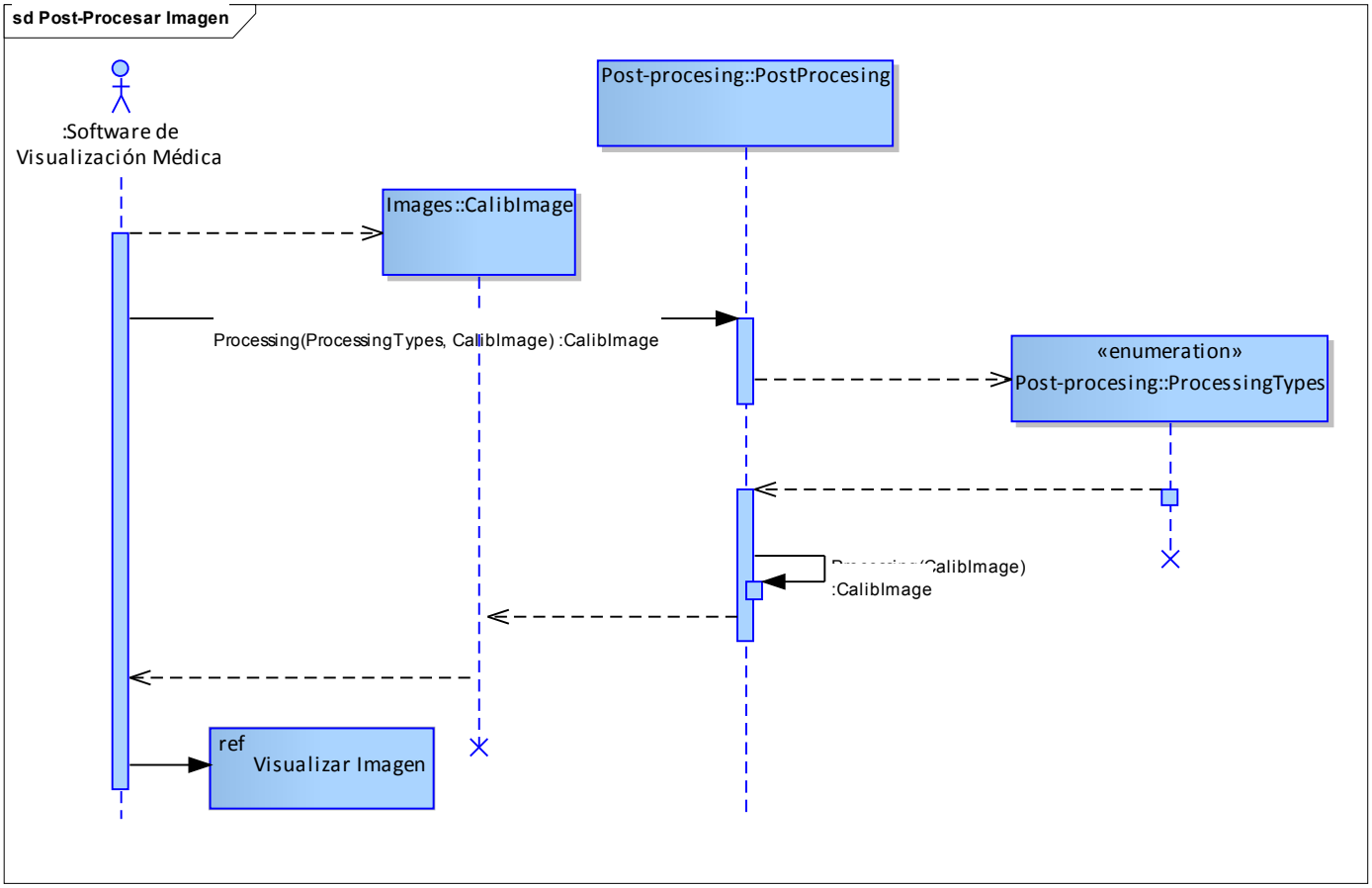


Fig.42 Diagrama de secuencia de CU Post-procesar imagen

CAPÍTULO 4. IMPLEMENTACIÓN

Se describe como fue implementado el framework CALIB. Se presenta el diagrama de componentes, que muestra la interacción de estos después de ser implementados. Se da una breve panorámica de cómo se integraría el framework como parte del sistema alas PACS y su instanciación dentro del Componente de Visualización 3D y su utilización en el Visor WEB de imágenes médicas.

4.1 Diagrama de Componentes

El modelo de componentes ilustra los componentes de software que se usarán para construir el sistema. Se pueden construir a partir del modelo de clases y escribir desde cero para el nuevo sistema o se pueden importar de otros proyectos y de productos de terceros [25].

El framework CALIB se encarga de abstraer todos los procesos de lectura contenidos en el paquete *Readers* y los procesos de descompresión contenidos en el paquete *Codecs Compression,* la cual consume funcionalidades existentes en la biblioteca de clases *dcmview.*

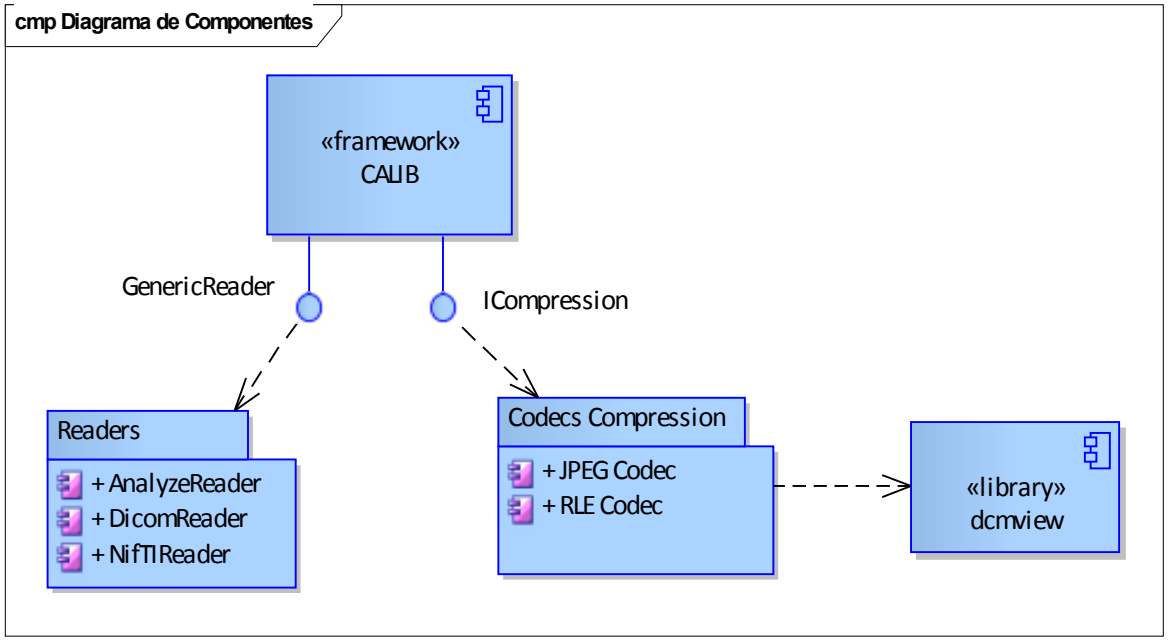


Fig.43 Diagrama de Componentes del framework CALIB

4.2 Integración dentro del sistema alas PACS

El sistema alas PACS en su versión 3.0 cuenta con componentes como el Viewer Manager, que entre sus funcionalidades principales se encuentra la lectura de imágenes médicas; otros de los componentes que se pueden encontrar son visores especializados de distintas modalidades que necesitan visualizar y procesar imágenes médicas; para lograr estos propósitos durante el diseño del sistema se ha tenido en cuenta la utilización del framework CALIB, como componente base para realizar los procesos anteriormente mencionados.

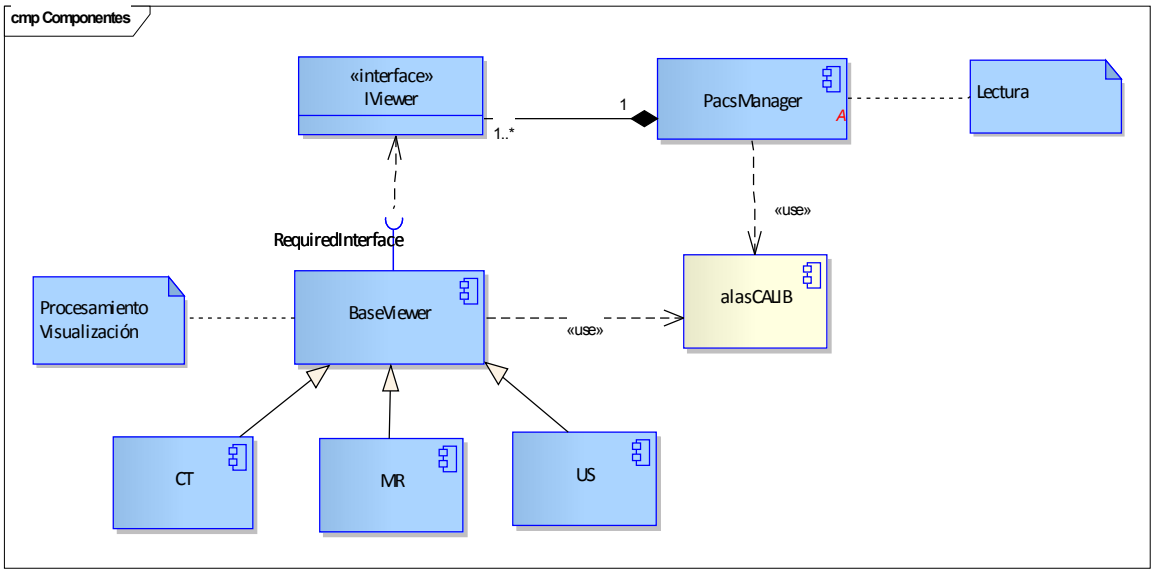


Fig.44 Integración del Framework CALIB dentro del sistema alas PACS v3.0

4.3 Instanciación del framework

En el capítulo uno se abordó las fases de desarrollo de un framework. La tercera fase es la instanciación que no es más que implementar los puntos calientes, llegando a crear un software del sistema. En el Departamento de Software Médico Imagenológico perteneciente al Centro de Informática Médica se desarrolló un Componente de Visualización 3D de neuroimágenes [26] Anexo 4; este componente permite la lectura, segmentación y posterior reconstrucción de neuroimágenes. También está en fase de

desarrollo un visor web de imágenes médicas vinculado a un servidor WADO[21] Anexo 5, este visor debe permitir visualizar imágenes DICOM de cualquier modalidad.

4.4 Integridad de los Componentes

Para lograr la autenticidad e integridad de los plugins que componen al framework se utilizó la criptografía asimétrica o comúnmente conocida como infraestructura de clave pública y privada.

Esta infraestructura está formada por dos claves, una pública que pude ser conocida por todo el mundo y una privada que es sólo conocida por el propietario de la firma; en este método de encriptación para cada clave pública solo existe una clave privada que la pueda desencriptar.

En el framework CALIB cada plugin fue firmado con una clave pública y el núcleo del framework que es la encargada de la gestión contiene la clave privada correspondiente, permitiendo así a la hora de cargar el plugin verificar la autenticidad del mismo.

Para firmar un plugin del framework CALIB se pude hacer de dos formas diferentes pero complementarias: con un nombre seguro o con Signcode[22]. Al firmar un plugin con un nombre seguro, se agrega un cifrado mediante clave pública al archivo. La firma mediante nombres seguros ayuda a comprobar la exclusividad del nombre, impide las simulaciones de nombres y proporciona a los llamadores alguna identidad cuando se resuelve una referencia.

Sin embargo, no hay ningún nivel de confianza asociado a un nombre seguro, lo que hace que Signcode sea importante. Con Signcode, una compañía de software debe demostrar su identidad a una autoridad de terceros y obtener un certificado. Este certificado se incrusta en el archivo y el administrador puede utilizarlo para decidir si debe confiar en la autenticidad del código.

Se puede asignar un nombre seguro y una firma digital Signcode al plugin, o se puede utilizar cualquiera de las dos opciones por separado. Signcode sólo firma los archivos de uno en uno. Un nombre seguro se almacena en el archivo que contiene el manifiesto del plugins, pero la firma Signcode se almacena en una ranura reservada del archivo ejecutable portable que contiene el manifiesto.

---

[21] *Web Access DICOM Object*

[22] Es una herramienta que permite firmar un archivo ejecutable portable (PE), un archivo .dll o .exe, con una firma digital. Es posible firmar un ensamblado o un archivo individual contenido en un ensamblado de varios archivos.

4.5 Ejemplo de Extensibilidad

Una de las características más importantes de un framework es poder extender sus funciones sin necesidad de modificar el núcleo del mismo. A continuación se brinda en forma resumida como crear un plugin de lectura para un formato de representación de imágenes médicas.

```
internal sealed class DICOMReader : GenericReader 1
  {
   2 DictionaryLevel curADescription = new DictionaryLevel();
   3 GeometricDescription curGDesription ;
   4 CalibRasterImage Raster;
   5 protected override void Open(string source)
      {
         //implementación
      }
   6 protected override void Open(Stream source)
      {
         //implementación
      }
   7 protected override void CloseReader()
      {
          //implementación
      }
   8 public override CalibImage GetFrameData(int index)
      {
          //implementación
      }
  }
```

1. Se hereda de la clase GenericReader perteneciente al namespace Readers del framework CALIB.
2. Se crea una instancia de la clase DictionaryLevel, con el fin de almacenar en memoria toda la información contenida en el fichero de imagen.

3. Se crea una instancia de la clase GeometricDescription, este objeto se construye a partir de datos que identifican a la imagen geometricamente, como por ejemplo, ancho y alto, espacio entre los pixels, etc.
4. Se crea una instancia de la clase CalibRasterImage a partir de los valores de grises correspondientes a la imagen y el tipo de pixel almacenado.
5. Este método es implementado para leer los datos a partir de un fichero físico.
6. Este método es implementado para leer los datos a partir de un fichero ya cargado en memoria.
7. Este método es implementado para liberar todos los recursos utilizados durante el proceso de lectura.
8. Se implementa para obtener una imagen lista para ser visualizada o procesada por el framework.

La clase GenericReader contiene atributos y propiedades que durante el proceso de lectura permiten una mejor manipulación de los ficheros.

CAPÍTULO 5. PRUEBAS

Se expone el resultado de algunas pruebas realizadas al framework, en cuanto a rendimiento calidad de los algoritmos de visualización y compatibilidad con la plataforma MONO.

5.1 Proceso de lectura

Se realizó una valoración entre el framework CALIB y la biblioteca de clases DICOM C# SDK en cuanto a los tiempos de respuesta durante el proceso de lectura. Para ello se utilizaron estudios de tomografías axial computarizada (512X512X16) representados en formato DICOM. Se realizaron siete corridas con cada aplicación con cantidades de imágenes diferentes en cada estudio y se registraron los tiempos de respuestas. Para ello se utilizó una PC convencional P4, con 512MB RAM.

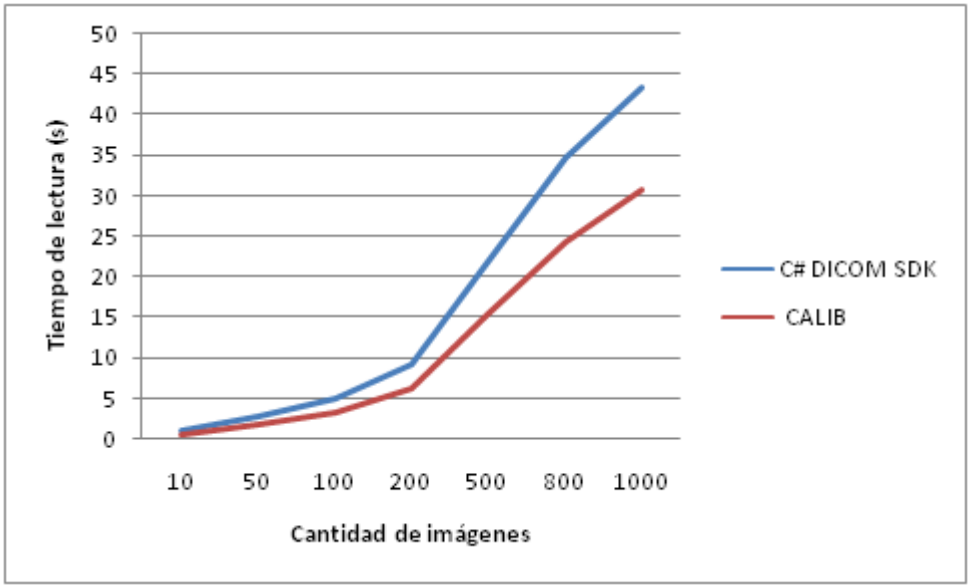


Fig.45 Resultado del proceso de lectura para volúmenes diferentes de imágenes

Adicionalmente se realizó la misma prueba con estudios de radiografía digital (3052X3023X16) el cual posee características diferentes a los de tomografía. En el caso de estudios de tomografía se caracteriza por grandes volúmenes de imágenes pero sin una gran resolución espacial en cambio la radiografía posee como máximo dos imágenes en un estudio pero con una gran resolución espacial. En este sentido los

tiempos de respuesta durante el proceso de lectura mostraron diferencias poco significativas por lo que se consideran que para estudios de pocas imágenes es posible el uso de cualquiera de las dos aplicaciones.

En los resultados se evidencia que existen diferencias significativas entre el uso de cada herramienta desde sistemas operacionales de imágenes médicas, sobre todo cuando desde estos sistemas se requiere acceder a grandes volúmenes de imágenes dentro de un estudio como es el caso de la tomografía axial computarizada durante un diagnóstico clínico. Este es un elemento que siempre requiere la mayor optimización para lograr una mayor eficiencia en el trabajo de los especialistas.

5.2 Algoritmos de Visualización

Para validar los algoritmos de visualización desarrollados se tomó como referencia los algoritmos desarrollados como parte del módulo de visualización del sistema alas PACS Viewer v2.7 y visores de imágenes médicas comerciales.

Se montó un ambiente de prueba que consistió en utilizar un banco de imágenes con estudios de diferentes modalidades diagnósticas. Estos estudios poseen diferentes características en cuanto a los niveles de grises en la imagen que van desde diversas formas de interpretación fotométrica (*Photometric Interpretation*), varios valores de ancho y centro de imágenes así como diversidad en cuanto a la resolución espacial de un estudio. La siguiente tabla muestra las características del banco de estudios empleados para la ejecución de las pruebas.

Modalidad	Cantidad de estudios	Interpretación Fotométrica	Resolución Espacial	Cantidad de imágenes promedio
CT	24	MONOCHROME2	512 x 512 x 12	400
MR	15	MONOCHROME2	256 x 256 x 12	150
US	10	RGB, PALETTE COLOR	512 x 512 x 8	50
DX	17	MONOCHROME1 MONOCHROME2	2048 x 2048 x 12	2

Fig.46 Características del banco de estudios para la ejecución de las pruebas

En todos los casos se obtuvo que los valores de salida del proceso de visualización con cada visor eran los mismos, lo cual asegura una correcta implementación de cada uno de estos algoritmos dentro del

framework CALIB, para una correcta visualización de estudios de imágenes teniendo en cuenta sobre todo que como parte de la muestra de prueba se utilizó una gama diversa de tipos de estudios todos con características diferentes.

5.3 Compatibilidad con MONO 2.4

La figura 47 muestra los resultados proyectados por MoMA, al analizar íntegramente el framework CALIB.

En el framework existen rutinas que son específicas para el sistema operativo que no son soportadas por MONO 2.4, pero se brindan variantes para contrarrestar estos problemas.

En el caso de la biblioteca de clases “Compression”, que ofrece algunas funcionalidades para la descompresión de imágenes, para complementar sus funcionalidades se utiliza una biblioteca de clases desarrollada en C++, que para poder utilizarla en Linux necesita ser recompilada con un compilador para ese sistema operativo.

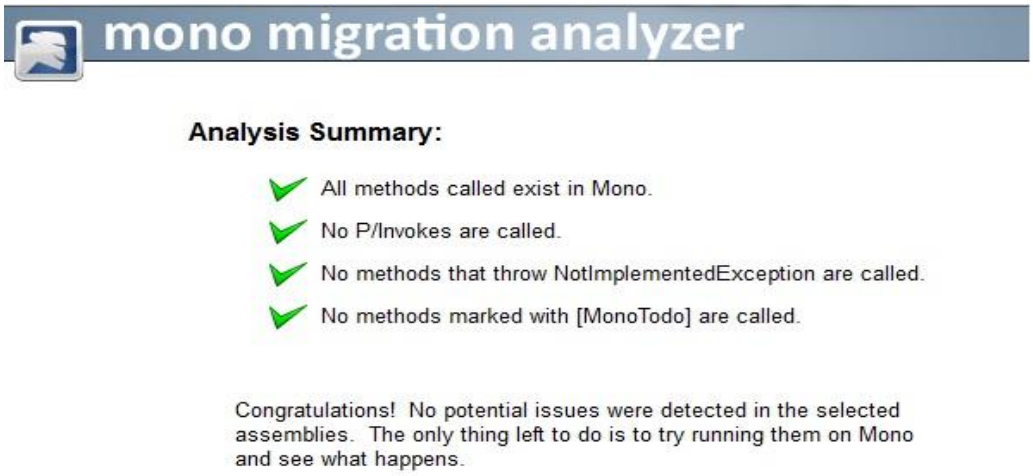


Fig.47 Resultados obtenidos con el software MoMA

CONCLUSIONES

Con el desarrollo de la presente investigación se logró implementar un sistema con una arquitectura robusta y flexible, capaz de realizar funciones de lectura, procesamiento y visualización de imágenes médicas con un rendimiento superior al de las aplicaciones con las que se contaba anteriormente.

El framework CALIB cuenta, entre sus características principales, la de poder hacer extensible sus funciones, de modo que brinda ventajas sustanciales tanto para el usuario como para el desarrollador, pudiéndose integrar nuevos lectores y compresores de manera sencilla y rápida.

La solución desarrollada sirve como base para el desarrollo de sistemas de visualización y procesamiento de imágenes médicas.

RECOMENDACIONES

Con el objetivo de ofrecer una mejor calidad en los servicios que presta el sistema, y teniendo en cuenta la necesidad de ir mejorando el mismo en pos de lograr funcionalidades cada vez más completas, los autores hacen las siguientes recomendaciones:

1. Lograr una arquitectura, capaz de realizar diversas asignaciones de memoria. En el caso de que la computadora cuente con una tarjeta gráfica, el sistema deberá hacer uso de la misma para realizar las funciones críticas de procesamiento.
2. Agregar lectores y compresores capaces de reconocer otros formatos de imágenes que no se tuvieron en cuenta en la primera versión.
3. Agregar puntos de extensibilidad para las funciones de escritura y transmisión.

REFERENCIAS BIBLIOGRÁFICA

[1]"Cassandra Viewer. Algoritmos empleados para la visualizacion de Imágenes Médicas"Memoria Personal Presentada en Opción al Tíıtulo de Máster en Informática Aplicada

[2] Instituo Universitario del Hospital Italiano., Curso Universitario.Sistemas de Información en los Sistemas de Salud. [Document] s.l. : Oregon.Healt&Science.

[3] , RadiologyInfo. *RadiologyInfo.* [En línea] [Citado el: 20 de Noviembre de 2009.] http://www.radiologyinfo.org/sp/info.cfm?pg=genus.

[4] MAPFRE. *Canal Salud.* [En línea] [Citado el: 20 de Noviembre de 2009.] http://www.mapfre.com/salud/es/cinformativo/resonancias-magneticas.shtml.

[5] Texas Heart Institute. *Texas Heart Institute.* [En línea] [Citado el: 25 de 11 de 2009.] http://www.texasheartinstitute.org/HIC/Topics_Esp/Diag/diangio_sp.cfm.

[6] Urología Avanzada en Madrid. *Urología Avanzada en Madrid.* [En línea] [Citado el: 20 de Noviembre de 2009.] http://www.urologiaavanzada.com.

[7] Markiewicz, Marcus Eduardo y de Lucena, Carlos J.P., El Desarollo Del Framework Orientada al Objeto . *El Desarollo Del Framework Orientada al Objeto .* [En línea] [Citado el: 15 de October de 2009.] http://www.acm.org/crossroads/espanol/xrds7-4/frameworks.html.

[8] Jhonson, Ralph E. y Foote, Brian., Designing Reusable Classes. *Journal of Object-Oriented Programming.* [En línea] June de 1988. [Citado el: 15 de October de 2009.] http://www.laputan.org/drc/drc.html.

[9] Larman, Craig., *UML y Patrones.Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* 2da Edición.

[10] Councill, Bill y Heineman, George T., "Definitions of Software Component and Its Elements."

[11] Philips Medical System., *DICOM Cook Book for implementations in Modalities.* 1997.

[12] , Format Analyze. *Format Analyze.* [En línea] [Citado el: 1 de 12 de 2009.] http://imaging.mrc-cbu.cam.ac.uk/imaging/FormatAnalyze.

[13] NifTI: Neuroimaging Informatics Tecnology Innitiative. *NifTI: Neuroimaging Informatics Tecnology Innitiative.* [En línea] [Citado el: 1 de 12 de 2009.] http://nifti.nimh.nih.gov/nifti-1.

[14] LEADTOOLS., LEADTOOLS Technology. *LEADTOOLS Medical Imaging Developer Toolkit.* [En línea] http://www.leadtools.com/sdk/medical-imaging.htm.

[15] My DICOM., DICOM.NET C# SDK. *DICOM.NET C# SDK.* [En línea] My DICOM. http://www.mydicom.net/CSharp.aspx.

[16] TRISPARK., Java Dicom Toolkit. *Java Dicom Toolkit.* [En línea] [Citado el: http://www.trispark.com/index.php/javadicomtoolkit de Enero de 2010.]

[17] DICOM@OFFIS., DCMTK - DICOM Toolkit. *DCMTK - DICOM Toolkit.* [En línea] OFFIS. http://dicom.offis.de/dcmtk.php.en.

[18] openDICOM NET., openDICOM.NET. The DICOM library project. *openDICOM.NET. The DICOM library project.* [En línea] http://www.opendicom.net/.

[19] Microsoft Corporation., Infromación General de Visual Studio 2008. *Visual Studio 2008 Development System.* [En línea] Microsoft Corporation. [Citado el: 8 de Febrero de 2010.] http://msdn.microsoft.com/es-es/vstudio/products/bb931331.aspx.

[20] SPARX System., Enterprise Architect. *Enterprise Architect.* [En línea] [Citado el: 8 de Febrero de 2010.] http://www.sparxsystems.com.ar/products/ea.html.

[21] Pozo, Pedro., clickerar.com. *clickerar.com.* [En línea] [Citado el: 5 de Febrero de 2010.] http://www.clikear.com/manuales/csharp/c10.aspx.

[22] Mono., MoMA. *MoMA.* [En línea] [Citado el: 5 de Febrero de 2010.] http://www.mono-project.com/MoMA.

[23] Grupo Soluciones GSINNOVA., Rational Unifed Process. *Rational Unifed Process.* [En línea] [Citado el: 4 de Febrero de 2010.] http://www.rational.com.ar/herramientas/rup.html.

[24] Jacobson, Ivar, Booch, Grady y Rumbaugh, James., *El Proceso Unificado de Desarrollo de Software.*

[25] SPARX System., El modelo de Componentes. *El modelo de Componentes.* [En línea] SPARX. [Citado el: 16 de Febrero de 2010.] http://www.sparxsystems.com.ar/resources/tutorial/component_model.html.

[26] Palenzuela, Filiberto López y Musulí, Luis Javier Vallejo., *Implementación de un Componente de Software para la visualización 3D de neuroimágenes.* 2009.

[27] , tuotromedico.com. *tuotromedico.com.* [En línea] [Citado el: 25 de 11 de 2009.] http://www.tuotromedico.com/temas/radiografia.htm.

[28] apeXNET. Software Factory., Enterprise Architect. *Enterprise Architect.* [En línea] [Citado el: 8 de Febrero de 2010.] http://www.apexnet.com.ar/index.php/product/viewProducts/24/sl=0.

BIBLIOGRAFÍA

- Diez, Hector Raúl González., "Cassandra Viewer. Algoritmos empleados para la visualizacion de Imágenes Médicas." Julio de 2007. Memoria Personal Presentada en Opción al Tíitulo de Máster en Informática Aplicada.
- Instituo Universitario del Hospital Italiano., *Curso Universitario.Sistemas de Información en los Sistemas de Salud.* [Document] s.l. : Oregon.Healt&Science.
- RadiologyInfo. *RadiologyInfo.* [Online] [Cited: Noviembre 20, 2009.] http://www.radiologyinfo.org/sp/info.cfm?pg=genus.
- MAPFRE. *Canal Salud.* [Online] [Cited: Noviembre 20, 2009.] http://www.mapfre.com/salud/es/cinformativo/resonancias-magneticas.shtml.
- Texas Heart Institute. *Texas Heart Institute.* [Online] [Cited: 11 25, 2009.] http://www.texasheartinstitute.org/HIC/Topics_Esp/Diag/diangio_sp.cfm.
- Urología Avanzada en Madrid. *Urología Avanzada en Madrid.* [Online] [Cited: Noviembre 20, 2009.] http://www.urologiaavanzada.com.
- Markiewicz, Marcus Eduardo and de Lucena, Carlos J.P., El Desarollo Del Framework Orientada al Objeto . *El Desarollo Del Framework Orientada al Objeto* . [Online] [Cited: October 15, 2009.] http://www.acm.org/crossroads/espanol/xrds7-4/frameworks.html.
- Jhonson, Ralph E. and Foote, Brian., Designing Reusable Classes. *Journal of Object-Oriented Programming.* [Online] June 1988. [Cited: October 15, 2009.] http://www.laputan.org/drc/drc.html.
- Larman, Craig., *UML y Patrones.Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* 2da Edición.
- Councill, Bill and Heineman, George T., "Definitions of Software Component and Its Elements."
- Philips Medical System., *DICOM Cook Book for implementations in Modalities.* 1997.
- Format Analyze. *Format Analyze.* [Online] [Cited: 12 1, 2009.] http://imaging.mrc-cbu.cam.ac.uk/imaging/FormatAnalyze.
- NifTI: Neuroimaging Informatics Tecnology Innitiative. *NifTI: Neuroimaging Informatics Tecnology Innitiative.* [Online] [Cited: 12 1, 2009.] http://nifti.nimh.nih.gov/nifti-1.

- Microsoft Corporation., Infromación General de Visual Studio 2008. *Visual Studio 2008 Development System.* [Online] Microsoft Corporation. [Cited: Febrero 8, 2010.] http://msdn.microsoft.com/es-es/vstudio/products/bb931331.aspx.
- SPARX System., Enterprise Architect. *Enterprise Architect.* [Online] [Cited: Febrero 8, 2010.] http://www.sparxsystems.com.ar/products/ea.html.
- Pozo, Pedro., clickerar.com. *clickerar.com.* [Online] [Cited: Febrero 5, 2010.] http://www.clikear.com/manuales/csharp/c10.aspx.
- Mono., MoMA. *MoMA.* [Online] [Cited: Febrero 5, 2010.] http://www.mono-project.com/MoMA.
- Grupo Soluciones GSINNOVA., Rational Unifed Process. *Rational Unifed Process.* [Online] [Cited: Febrero 4, 2010.] http://www.rational.com.ar/herramientas/rup.html.
- Jacobson, Ivar, Booch, Grady and Rumbaugh, James., *El Proceso Unificado de Desarrollo de Software.*
- SPARX System., El modelo de Componentes. *El modelo de Componentes.* [En línea] SPARX. [Citado el: 16 de Febrero de 2010.] http://www.sparxsystems.com.ar/resources/tutorial/component_model.html.
- Palenzuela, Filiberto López and Musulí, Luis Javier Vallejo., *Implementación de un Componente de Software para la visualización 3D de neuroimágenes.* 2009.
- RadiologyInfo. [Online] http://www.radiologyinfo.org/sp/info.cfm?pg=fmribrain.
- ACM. [En línea] http://www.acm.org/crossroads/espanol/xrds7-4/frameworks.html.
- Bio-Formats. [En línea] http://www.loci.wisc.edu/bio-formats-format/analyze-75.
- Clínica Dam. [En línea] http://www.clinicadam.com/salud/5/003330.html.
- ACR-NEMA., DICOM (Digital Imaging and Communication in Medicine). [En línea] http://medical.nema.org/.
- apeXNET. Software Factory., Enterprise Architect. *Enterprise Architect.* [En línea] [Citado el: 8 de Febrero de 2010.] http://www.apexnet.com.ar/index.php/product/viewProducts/24/sl=0.
- "Ibercaja Zentrum." [En línea] http://www.ibercajalav.net/img/imagen_medica.pdf.
- "Imagenología Bio-médica." [En línea] http://www.ipitimes.com/ibm.htm.
- JAMIA. [En línea] http://www.ncbi.nlm.nih.gov/pmc/articles/PMC349388/.

- LeadTool. [En línea] http://www.leadtools.com/.
- NHS Connecting for Health. [En línea] http://www.connectingforhealth.nhs.uk/.
- RadiologyInfo. [En línea] http://www.radiologyinfo.org/sp/info.cfm?pg=headct.
- RadiologyInfo. [En línea] http://www.radiologyinfo.org/sp/info.cfm?pg=PET.
- TRISPARK. [En línea] http://www.trispark.com/.
- tuotromedico.com. *tuotromedico.com.* [En línea] [Citado el: 25 de 11 de 2009.] http://www.tuotromedico.com/temas/radiografia.htm.
- Félix Prieto (Universidad de Valladolid), Yania Crespo (Universidad de la Habana), Francisco J. García (Universidad de Salamanca), Miguel A. Laguna (Universidad de Valladolid)., "Construcción de frameworks basada en análisis de conceptos." [En línea] http://www.giro.infor.uva.es/oldsite/docpub/vmenhir-conceptos-formales.pdf.

ANEXOS

Anexo 1. Valores de Representación (VR).

VR	Definición	Longitud (Bytes)
AE	Cadena de caracteres que identifica una Entidad de Aplicación, siendo los espacios (20h) no significativos.	16
AS	Cadena de caracteres con uno de estos formatos: nnnD, nnnW, nnnM, nnnY; donde nnn es el número de D (días), W (semanas), M (meses), o Y (años)	4
AT	Par ordenado de enteros sin signo de 16 bits (igual que la codificación del campo Etiqueta del Elemento de Datos)	Cadena de caracteres
CS	Cadena de caracteres, siendo los espacios (20h) no significativos.	16
DA	Cadena de caracteres con el formato yyyymmdd, siendo yyyy el año, mm el mes y dd el día	8
DS	Cadena de caracteres representando un número en coma fija o flotante.	16
DT	Representa una fecha.	26
FL	Número en punto flotante de precisión simple.	4
FD	Número en punto	8
IS	Cadena de caracteres que representa un entero en base decimal.	12
LO	Cadena de caracteres.	64
LT	Cadena de caracteres que puede tener uno o más párrafos.	10240 Max.
OB	Cadena de bytes. La codificación del contenido depende del campo Sintaxis de Transferencia.	Según TSN

VR	Definición	Longitud (Bytes)
OF	Cadena de números en punto flotante de simple precisión.	Max.
OW	Cadena de palabras de 16 bits. La codificación del contenido depende del campo Sintaxis de Transferencia.	Según TSN
PN	Cadena de caracteres de cinco componentes: apellidos, nombre de pila, Segundo nombre, prefijo, sufijo.	64 Max.
SH	Cadena de caracteres. 16 caracteres	max.
SL	Entero con signo de 32 bits en complemento a dos.	4
SQ	Secuencia de cero o más ítems.	No aplicable
SS	Entero con signo de 16 bits en complemento a dos.	2
ST	Cadena de caracteres que puede tener uno o más párrafos.	1024 caracteres max.
TM	Cadena de caracteres con el formato *hhmmss.frac.*	16 Max.
UI	Cadena de caracteres que contiene un Identificador Único.	64 Max.
UL	Entero sin signo de 32 bits.	4
UN	Cadena de bytes. La codificación del contenido es desconocida.	Cualquier longitud Válida
US	Entero sin signo de 16 bits.	2
UT	Cadena de caracteres que puede contener uno o más párrafos.	Max.

Anexo 2. Estructura del fichero “Header” del formato ANALYZE 7.5.

Tipo Dato	Campo	Dezplaz.	Tamaño
int	sizeof_hdr	0	4
string	data_type	4	10
string	db_name	14	18
int	extents	32	4
short	session_error	36	2
char	regular	38	1
char	hkey_un0	39	1
Array[8] Short	dim	40	16
short	unused8	56	2
short	unused9	58	2
short	unused10	60	2
short	unused11	62	2
short	unused12	64	2
short	unused13	66	2
short	unused14	68	2
short	datatype	70	2
short	bitpix	72	2
short	dim_un0	74	2
Array[8] float	pixdim	76	32
float	vox_offset	108	4
float	funused1	112	4
float	funused2	116	4
float	funused3	120	4
float	cal_max	124	4
float	cal_min	128	4
float	compressed	132	4
float	verified	136	4
int	glmax	140	4
int	glmin	144	4
string	descrip	148	80

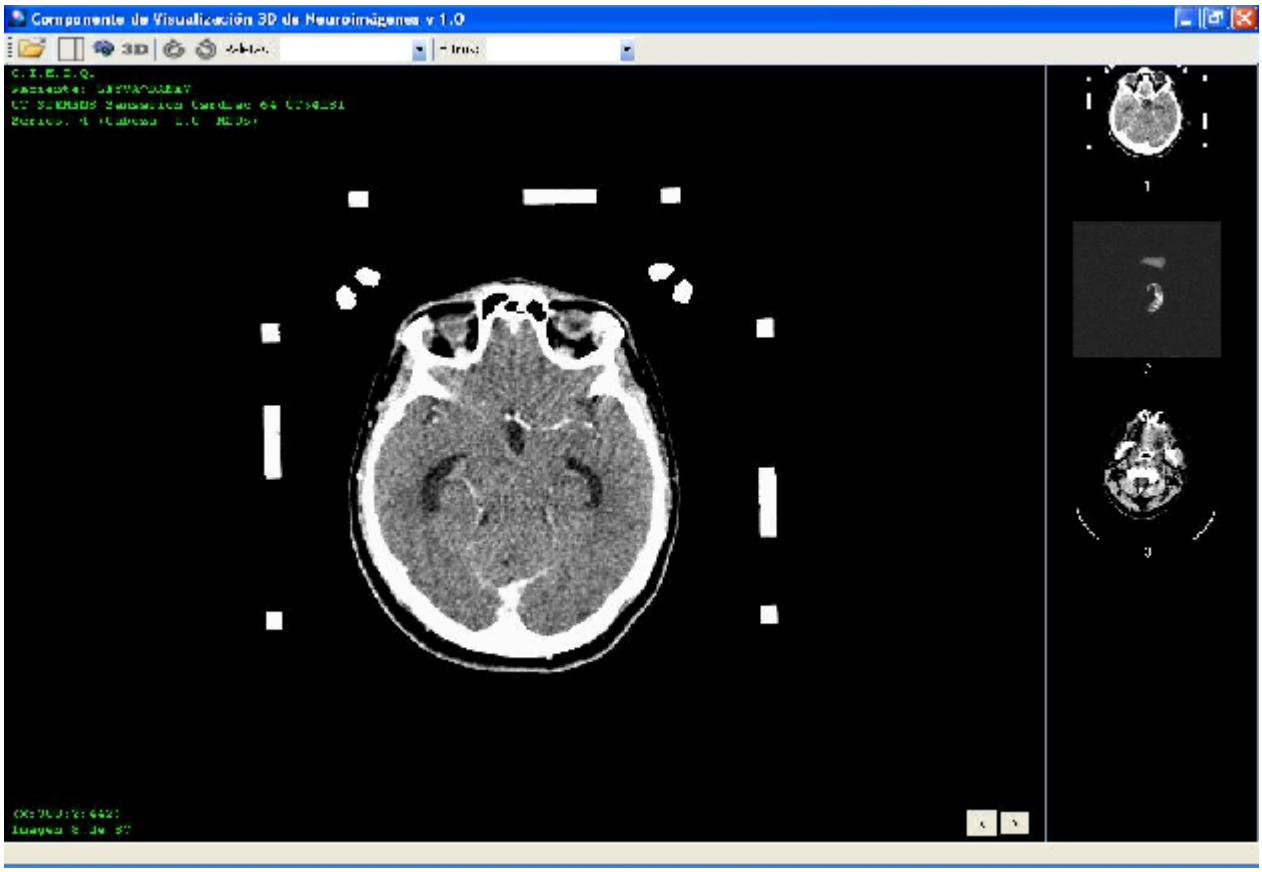
Tipo Dato	Campo	Dezplaz.	Tamaño
string	aux_file	228	24
string	orient	252	1
string	originator	253	10
string	generated	263	10
string	scannum	273	10
string	patient_id	283	10
string	exp_date	293	10
string	exp_time	303	10
string	hist_un0	313	3
int	views	316	4
int	vols_added	320	4
int	start_field	324	4
int	field_skip	328	4
int	omax	332	4
int	omin	336	4
int	smax	340	4
int	smin	344	4

Anexo 3. Estructura del fichero “Header” del formato NifTI.

Tipo Dato	Campo	Dezplaz.	Tamaño
int	sizeof_hdr	0	4
string	data_type	4	10
string	db_name	14	18
int	extents	32	4
short	session_error	36	2
char	regular	38	1
char	dim_info	39	1
Array[8] Short	dim	40	16
float	intent_p1	56	4
float	intent_p2	60	4
float	intent_p3	64	4
short	intent_code	68	2
short	datatype	70	2
short	bitpix	72	2
short	slice_start	74	2
float	pixdim	76	32
float	vox_offset	108	4
float	scl_slope	112	4
float	scl_inter	116	4
short	slice_end	120	2
char	slice_code	122	1
char	xyzt_unit	123	1
float	cal_max	124	4
float	cal_min	128	4
float	slice_duration	132	4
float	toffset	136	4
int	glmax	140	4
int	glmin	144	4
string	descrip	148	80
string	aux_file	228	24
short	qform_code	252	2

Tipo Dato	Campo	Dezplaz.	Tamaño
short	sform_code	254	2
float	quatern b	256	4
float	quatern c	260	4
float	quatern d	264	4
float	qoffset x	268	4
float	qoffset y	272	4
float	qoffset z	276	4
float	srow x	280	16
float	srow y	296	16
float	srow z	312	16
char	intent_name	328	16
char	magic	344	4

Anexo 4. Interfaz principal del componente de visualización 3D de neuroimágenes.



Anexo 5. Interfaz principal del Visor WEB de Imágenes Médicas