SAD (SOFTWARE ARCHITECTURE DOCUMENT) PARA TRABAJO DE GRADO

TÍTULO

Análisis del procesamiento de imágenes médicas pulmonares para el diagnóstico y tratamiento del SDRA

MODALIDAD: Proyecto de investigación

ESTUDIANTE(S)

Cesar Alejandro Guayara Rodríguez _____

Documento	Celular	Teléfono fijo	Correo Javeriano
cc. 1015438560	316-529-6135	7319482	c_guayara@javeriana.edu.co

Erika Jennifer Harker Gutiérrez _____

Documento	Celular	Teléfono fijo	Correo Javeriano
cc. 1018471803	301-341-2386	7599794	eharker@javeriana.edu.co

Juan Manuel Sánchez Lozano _____

Documento	Celular	Teléfono fijo	Correo Javeriano
cc. 1013665691	314-471-2446	-	jsanchez.l@javeriana.edu.co

Juan Miguel Gómez Ganem _____

Documento	Celular	Teléfono fijo	Correo Javeriano
cc. 1020822419	304-574-9104	-	jm-gomez@javeriana.edu.co

Luis David Zárate Castillo _____

Documento	Celular	Teléfono fijo	Correo Javeriano
cc. 1014282537	304-382-3335	-	zarate.luis@javeriana.edu.co

DIRECTOR

Leonardo Flórez Valencia _____

Documento	Celular	Teléfono fijo	Correo Javeriano	Empresa donde trabaja y cargo
cc. -	314-893-2219	-	florez-l@javeriana.edu.co;	Pontificia Universidad Javeriana; Profesor Departamento de Sistemas

Contenido

Lista de tablas e ilustraciones

Lista de tablas

Lista de ilustraciones

1 . Introducción

1.1 Propósito

El propósito de este documento es obtener la representación detallada del sistema a nivel de arquitectura, a través de representaciones que muestre aspectos físicos, de desarrollo, lógico y de procesos. Además, se muestra el diseño de bajo y alto nivel del sistema, con el cual se podrá especificar los componentes con la ayuda de diferentes vistas. [1] [2].

Aquí se especifican los casos de uso o funcionalidades globales que debe tener el sistema, así como demás características que son necesarias describir. Con ello se busca poder tener un panorama general del sistema para desarrollar el proyecto y definir su alcance.

Este documento tiene como principales usuarios los miembros del equipo que estarán involucrados en el prototipo, pues es de vital importancia para los programadores ya que es un referente para la implementación, también será de gran utilidad para poder limitar el alcance del proyecto y de esta manera saber cuáles deben ser las funciones que debe tener el software.

Con el fin de describir el software con la mayor precisión posible, la estructura de este documento se basa en la idea de "4 + 1" modelo de la arquitectura [4]

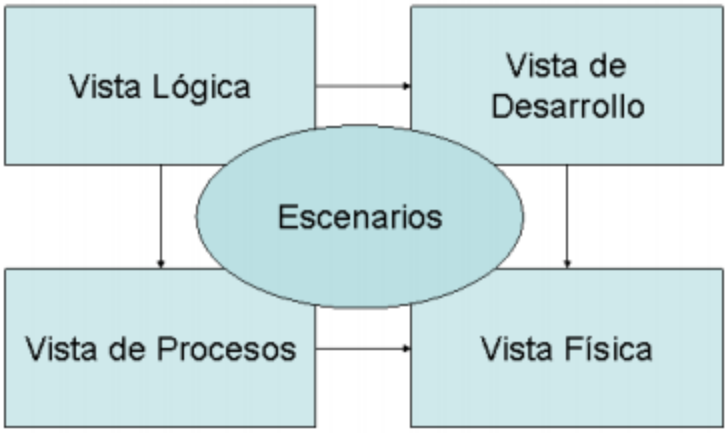

Ilustración 1. Modelo de arquitectura "4 + 1" [5]

1.2 Alcance

El SAD tiene toda la descripción de la arquitectura del sistema QuimeraTK, partiendo de las vistas del sistema (Casos de uso, lógica, desarrollo, física y procesos), adicionalmente es necesario detallar algunos aspectos no funcionales de la arquitectura del sistema.

1.3 Definiciones, siglas y abreviaturas

- RUP: Rational Unified Process
- UML: Unified Modeling Language
- SAD: Software Architecture Document
- **Diagrama de secuencia:** Permiten formalizar el comportamiento del sistema para visualizar la comunicación entre objetos.
- **Diagrama de actividad:** Demuestra la serie de actividades que deben ser realizadas en un uso-caso, así como las distintas rutas que pueden irse desencadenando en el uso-caso.
- **Usuario:** concepto que se usa para mencionar a aquellos que acceden al sistema de manera directa.

2 Vista general del proyecto

Lógica:

Esta vista tiene como objetivo modelar el diseño y dar soporte a los requerimientos funcionales que debe proveer el sistema en términos de servicios de los usuarios. Esta vista arquitectónica se enfoca a la funcionalidad del sistema, mostrar abstracciones claves que permitan descomponer el sistema en subsistemas claves. [3]

Audiencia: Arquitecto y diseñadores de Software
Área: Requerimientos funcionales: describe el modelo de diseño de objetos. Describe también las más importantes realizaciones de casos de uso.
Artefactos Relacionados: Modelo de diseño

Proceso:

Esta vista tiene como objetivo representar los requerimientos no funcionales del sistema. Esta vista tiene en cuenta algunos requerimientos no funcionales como desempeño, disponibilidad, concurrencia y distribución, integridad del sistema, tolerancia fallas. La vista se centra por tanto en la concurrencia y distribución de procesos. [3]

Audiencia: Líder del proyecto
Área: Requerimientos no funcionales
Artefactos Relacionados: Modelo de diseño

Implementación

Se representan requerimientos internos del sistema como facilidad de desarrollo, administración de software, reutilización de código y las limitaciones técnicas que pueden presentar las tecnologías de desarrollo y sus herramientas. Su objetivo es presentar una representación modular del sistema y con esto facilitar el proceso de desarrollo del software y la administración de sus configuraciones. [3]

Audiencia: Diseñadores y Desarrolladores
Área: componentes de software: describe las capas y subsistemas de la aplicación.
Artefactos Relacionados: Modelo de Implementación y componentes

Despliegue o física

La vista física contempla la implantación del software sobre hardware. Se identifican los elementos (redes, procesos, tareas, y objetos) que conforman la topología del hardware usado. Se centra en requerimientos no funcionales como disponibilidad, fiabilidad, escalabilidad y ejecución. [3]

Audiencia: Administradores de Hardware y el Arquitecto de software.
Área: Implementación e Instalación. Topología: describe la asignación del software en el hardware y muestra aspectos distribuidos del sistema.
Artefactos Relacionados: Diagrama de Despliegue

Escenarios o casos de uso

Esta vista unifica las demás vistas, los escenarios que la componen son instancias de los casos de uso que representan escenarios del sistema. Así, desde casos de uso se debe poder realizar la trazabilidad a los componentes del sistema de software. [3]

Audiencia: Todos los stakeholders del sistema, incluidos los usuarios finales.
Área: describe el conjunto de escenarios y / o casos de uso que representan algunas significativa la funcionalidad, la central del sistema.
Artefactos relacionados: Modelo de Casos de uso, documentación de casos de uso

3 Objetivos y restricciones de la arquitectura

A continuación, se describen los requerimientos que deben ser considerados en la definición de la arquitectura del sistema QuimeraTK. En general la arquitectura deberá ser diseñada teniendo en cuenta los siguientes objetivos:

- Procesar imágenes médicas cargadas desde el computador donde se encuentra el sistema
- Integrar 2 o más algoritmos de procesamientos de imágenes en el mismo sistema
- Utilizar los algoritmos integrados sobre una imagen cargada desde el computador donde se encuentra el sistema

3.1 Desempeño

Puesto que el proceso de realizar un TAC demora entre 20 y 30 minutos, el procesamiento de las imágenes obtenidas del TAC no debe ser mayor a 30 minutos.

3.2 Usabilidad

El sistema debe proveer una interfaz de usuario que permita manipular la imagen cargada en el sistema mediante los algoritmos integrados en este

3.3 Mantenibilidad

Para hacer que el software sea fácilmente sostenible, se mantendrá la mayor independencia, entre clases, que sea posible para así evitar que pequeños cambios puedan afectar todo el sistema.

3.4 Portabilidad

Los diferentes módulos que componen el sistema deben ser soportados por los sistemas operativos más comunes: Windows y Ubuntu -Linux.

3.5 Seguridad

Al ser un software que maneja imágenes médicas, las cuales tienen información sensible y confidencial de los pacientes, la información no va a estar en servidores públicos, por lo que la aplicación va a ser standalone para conservar la información en los computadores del cliente.

3.6 Adaptabilidad

El software debe ser capaz de integrar varios algoritmos y a la vez poder representar los resultados en la interfaz gráfica. Mientras que el algoritmo este envuelto en el filtro que maneja el sistema, QuimeraTK debe ser capaz de representar los datos

3.7 Plataforma Tecnológica

La arquitectura de software del sistema QuimeraTK, será implementada en el lenguaje de programación C++.

4 Vista de Casos de Uso

En esta sección se realiza la descripción de la vista de e casos de uso, del sistema QuimeraTK. Se muestra y describe cada caso de uso y sus actores alineados a las funcionalidades del sistema. A continuación, se muestra el diagrama de casos de uso del sistema:

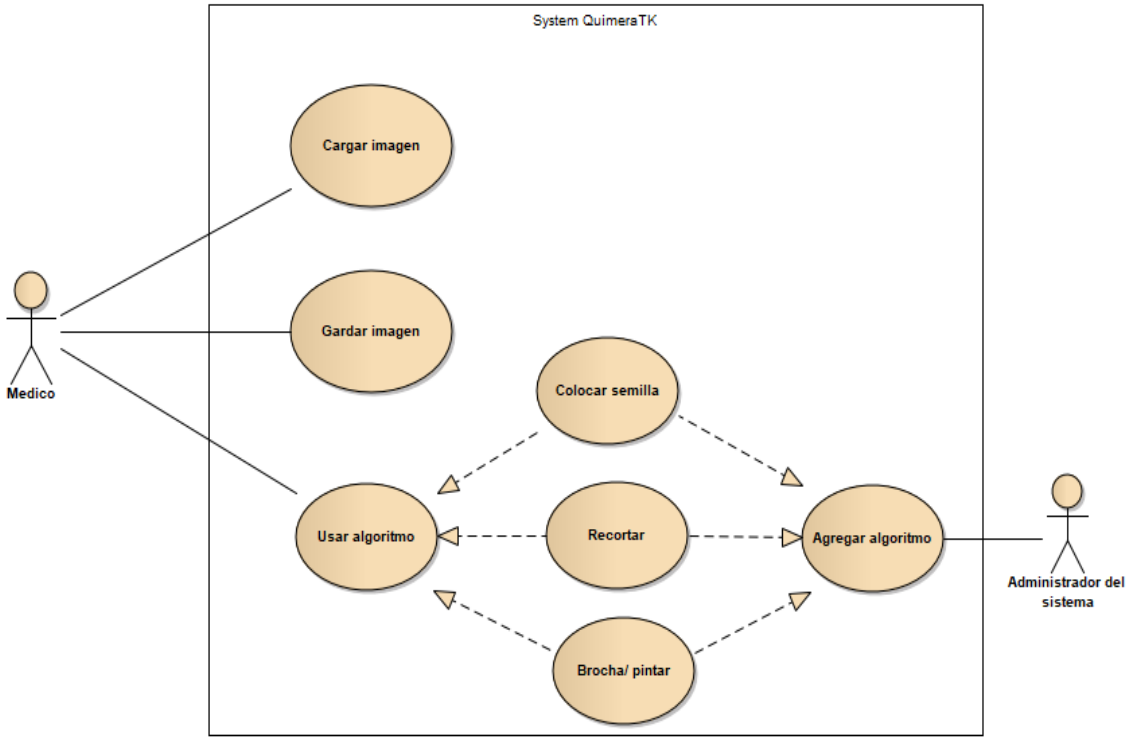

Ilustración 2. Diagrama de Casos de Uso de QuimeraTK

NOTA: Los casos de uso "Colocar Semilla", "Recortar" y "Brocha/ pintar" representa los algoritmos que se encuentran integrados en el sistema. Por lo tanto, de acuerdo a los algoritmos que se encuentren en el sistema, así serán estos casos de uso.

4.1 Actores

Actor	**Descripción**
Medico	Usuario principal del sistema. Realiza las acciones sobre las imágenes médicas con los algoritmos que se encuentren integrados en QuimeraTK. La interfaz de gráfica de usuario esta diseña para el medico
Administrador del sistema	Se encarga de actualizar el sistema y administrar los algoritmos que estarán integrados en el sistema

Tabla 1. Actores del sistema QuimeraTK

4.2 Casos de uso

Caso de Uso 1.	Cargar imagen		
Objetivo	Permitir al usuario cargar al sistema una imagen y aceptar los formatos de las imágenes medicas		
Precondición	La imagen debe está en el computador donde se ejecuta QuimeraTK		
Condiciones de salida en caso de éxito (Poscondición)	Se muestra en la pantalla de QuimeraTK la imagen cargada para poder aplicar los diferentes algoritmos de procesamiento de imagen que se encuentren integrados en el sistema y que desee aplicar el usuario (medico)		
Condiciones de salida en caso de fallo	Mostrar mensaje de error indicando que no se cargó la imagen y mostrar la pantalla donde debería aparecer la imagen en negro		
Actores	Medico		
DESCRIPCIÓN	**Paso**	**Medico**	**Sistema**
	1	El medico elige la imagen que desea cargar en el sistema	
	2	El usuario da click en “cargar imagen” o “aceptar”	
	3		El sistema muestra la imagen en la pantalla
	4		Se activan las opciones integradas de procesamiento de imagen

Tabla 2. Caso de uso 1: Cargar Imagen

Caso de Uso 2.	Guardar Imagen
Objetivo	Una vez modificada la imagen a través de los algoritmos de procesamiento de imágenes utilizados por el usuario, guardar la imagen como un nuevo archivo con los cambios respectivos
Precondición	Cargar una imagen en QuimeraTK
Condiciones de salida en caso de éxito (Poscondición)	Crear un nuevo archivo con los cambios realizados por el usuario sobre la imagen a través de los algoritmos de procesamiento de imágenes existentes en el sistema
Condiciones de salida en	Si la imagen no se puede guardar, se debe mostrar un mensaje de

caso de fallo	error al usuario y no generar el nuevo archivo ni reemplazar el antiguo		
Actores	Medico		
DESCRIPCIÓN	**Paso**	**Medico**	**Sistema**
	1	Seleccionar la opción de "guardar imagen"	
	2	Seleccionar ruta y nombre del archivo que se va a guardar	
	3		Generar archivo en la ruta especificada

Tabla 3. Caso de uso 1: Guardar Imagen

Caso de Uso 3.	Usar algoritmo		
Objetivo	Usar uno de los algoritmos integrados en el sistema sobre la imagen y generar los cabios respectivos		
Precondición	- Cargar una imagen en QuimeraTK - Estar integrado el algoritmo que se desea utilizar - Solo se puede tener activo un algoritmo a la vez sobre la imagen		
Condiciones de salida en caso de éxito (Poscondición)	- Modificaciones realizadas sobre la imagen cargada - Nueva imagen ya procesada con los parámetros dados por el usuario		
Condiciones de salida en caso de fallo			
Actores	Medico		
DESCRIPCIÓN	**Paso**	**Medico**	**Sistema**
	1	Selecciona el algoritmo (acción) que desea hacer sobre la imagen	
	2		El sistema conecta el algoritmo deseado con el sistema
	3	Utiliza los comandos para activar el algoritmo e introducir los parámetros (los comandos pueden estar dados por una combinación de teclas en el teclado o	

		una secuencia de clics con el mouse)	
	4		Recibe los parámetros y realiza el algoritmo seleccionado
	5		Modifica la imagen según el resultado del algoritmo
	6		Actualiza la vista del usuario para mostrar el resultado del algoritmo
EXTENSIONES	3a	Caso de uso: Colocar semilla	
	3b	Caso de uso: Recortar	
	3..	… …	
	3n	Caso de uso: Brocha/Pintar	

Tabla 4. Caso de uso 1: Usar algoritmo

Caso de Uso 4.	Agregar algoritmo		
Objetivo	Agregar un nuevo algoritmo al sistema		
Precondición	- El algoritmo debe estar envuelto en un filtro con la estructura que el sistema QuimeraTK utiliza para integrar los algoritmos - El sistema no debe estarse ejecutando		
Condiciones de salida en caso de éxito (Poscondición)	- Nueva opción en el panel de herramientas que representa el nuevo algoritmo integrado		
Condiciones de salida en caso de fallo			
Actores	Administrador del sistema		
DESCRIPCIÓN	**Paso**	**Administrador**	**Sistema**
	1	Generar XML que contiene la estructura del filtro que envuelve el algoritmo	
	2	Colocar la información en donde se encuentran los algoritmos a integrar en el sistema	

	3	Compilar nuevamente el sistema QuimeraTK	
	4		Generar accesos en la barra de herramientas a los algoritmos disponibles en el sistema
EXTENSIONES	4ª	Caso de uso: Colocar semilla	
	4b	Caso de uso: Recortar	
	4..	… …	
	4n	Caso de uso: Brocha/Pintar	

Tabla 5. Caso de uso 1: Agregar Algoritmo

5 Vista Lógica

La vista lógica debe soportar todos los requerimientos funcionales, así como aquello que el sistema debe proveer en términos de servicio a sus usuarios. Se dividen los paquetes lógicos que estructura la arquitectura. Esta vista muestra los componentes principales de diseño y sus relaciones sin tener en cuenta detalles técnicos. [9]

5.1 Descripción

5.1.1 Estructura del sistema

Arquitectura del sistema: Modelo- Vista – Controlador

El Modelo es el objeto que representa los datos del programa y a la vez la lógica del negocio. Por lo tanto, aquí se maneja todas las transformaciones de los datos,[7] es decir, los algoritmos de procesamiento de imágenes. Como tal, el Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni contiene referencias a ellos. Así, el modelo es alimentado por los algoritmos desarrollado por externos, sin necesidad de conocer el sistema en donde serán integrados.

La Vista es el objeto que maneja la presentación visual de los datos representados por el Modelo. Interactúan preferentemente con el controlador, pero es posible que interactúe con el Modelo.[7] Está basado en los diseños que utiliza los actuales softwares de procesamiento de imágenes médicas.

El Controlador es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo, centra toda la interacción entre la Vista y el Modelo.[7] Así, a través del Controlador se gestiona las entradas del usuario.

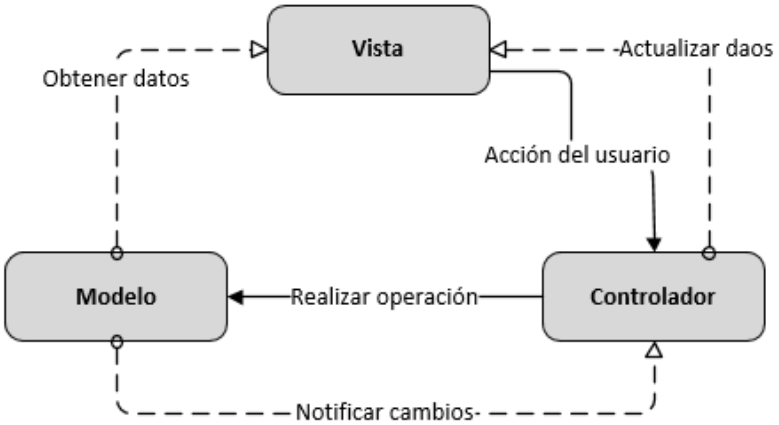

Ilustración 3. Estructura básica Modelo-Vista-Controlador

5.1.2 Flujo de información

Patrón de transferencia de datos: Filtros y tuberías

El objetivo principal del sistema es integrar diferentes algoritmos. Para este fin, se utiliza el patrón de filtros y tuberías, el cual se aplica cuando los datos de entrada requiere una transformación para mostrar una respuesta en los datos de salida.[8] Esta transformación se realiza mediante una serie de componentes que realizarán los cálculos o la manipulación.

Una de las características más importantes del patrón para el sistema es que cada filtro trabaja de manera independiente de los componentes que se encuentren situados antes o después de él, lo que permite la integración de diferentes algoritmos, al igual que retiras los que sean obsoletos.

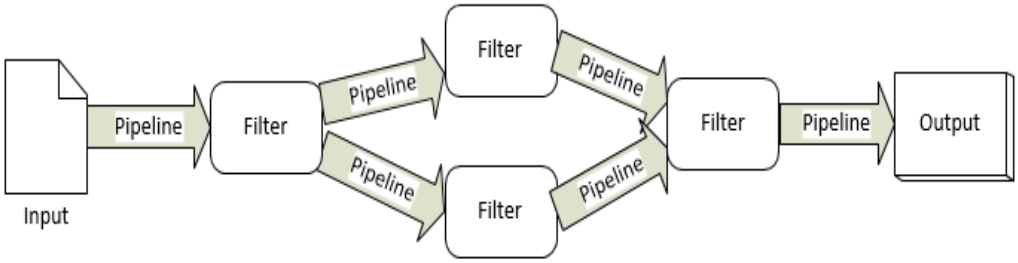

Ilustración 4. Funcionamiento básico del patrón filtros y tuberías (filters and pipelines)

5.1.3 Vista lógica

En la siguiente imagen se muestra la distribución del sistema y los diferentes componentes que se encuentran en esta.

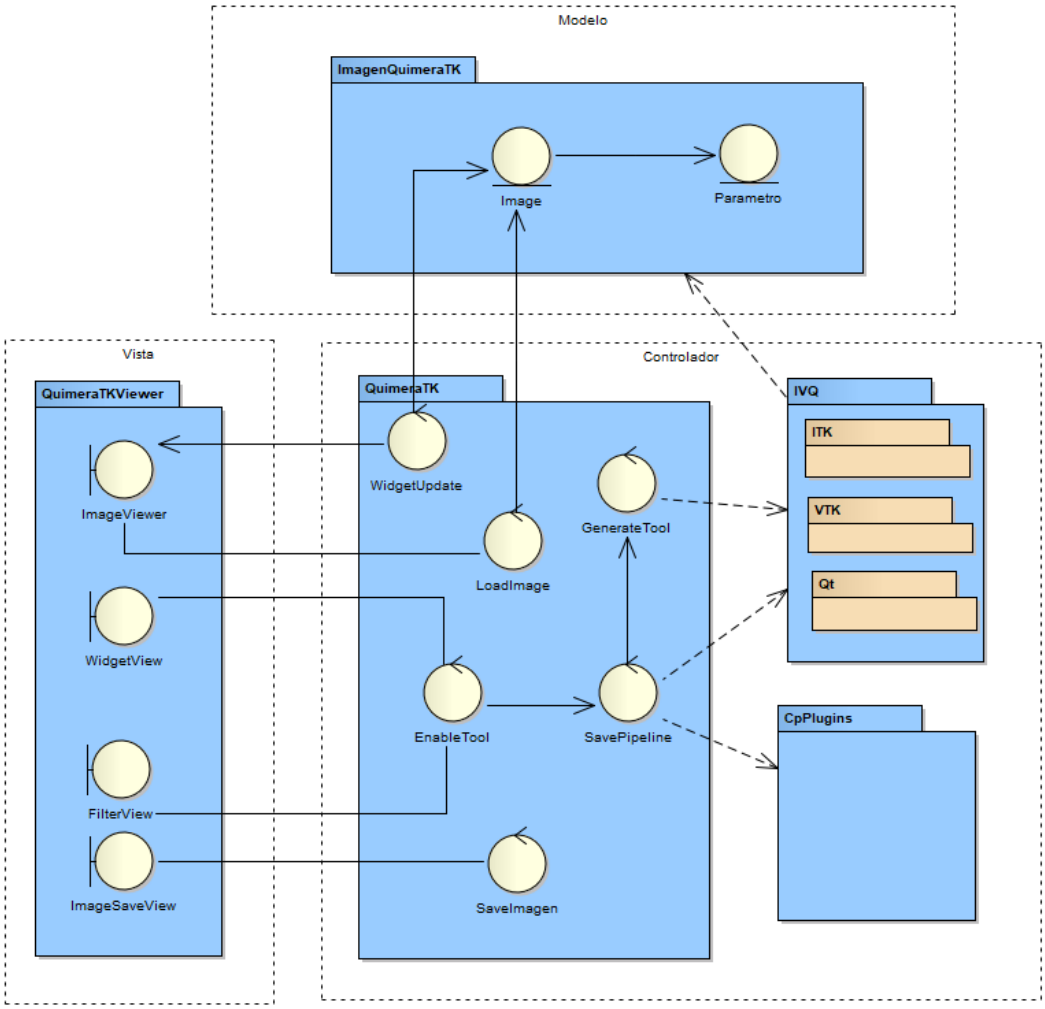

Ilustración 5. Vista lógica de QuimeraTK

Paquete	**Elemento**	**Descripción**
QuimeraTKViewer	ImageViewer	Es la vista que muestra la imagen que está disponible para ser procesada a través de los algoritmos integrados en el sistema
	WidgetView	Muestra los Widgets disponibles en el sistema.
	FilterView	Muestra los Filtros disponibles en el sistema
	ImageSaveView	Muestra la estructura de carpetas del computador donde se encuentra ejecutándose el programa para mostrar la ruta en donde se guardara la nueva imagen

QuimeraTK	LoadImage	Permite cargar en QuimeraTK la imagen que se va a procesar
	EnableTool	Habilita el algoritmo que se va a utilizar sobre la imagen
	SavePipeline	Genera los Piepelines necesarios para conectar los procesos (filtros y/o widgets) para realizar el algoritmo
	GenerateTool	Carga en QuimeraTK los algoritmos que se van a integrar
	SaveImage	Guarda la imagen ya procesada en una ruta especifica en el computador donde se está ejecutando el programa
ImagenQuimeraTK	WidgetUpdate	Es el encargado de observar los cambios en la imagen y reflejarlos en la interfaz del usuario
	Imagen	Representa la imagen dentro del sistema
	Parámetro	Son los parámetros que recibe los algoritmos para modificar la imagen
IVQ	ITK	Contiene filtros y widgets de procesamiento de imágenes y visualización especializadas en segmentación
	VTK	Contiene filtros y widgets de procesamiento de imágenes y visualización
	Qt	Provee herramientas para mostrar en la vista los resultados de los algoritmos utilizados en las imágenes
CpPlugins		Sistema de plugins utilizado para la creación de pipelines que proporcionan las funcionalidades de estructuración, procesamiento y visualización de imágenes.

Tabla 6. Descripción de elementos de la vista logica

NOTA: Para los Widgets y Filters solo mantiene activo uno a la vez entre los dos

5.2 Diseño base

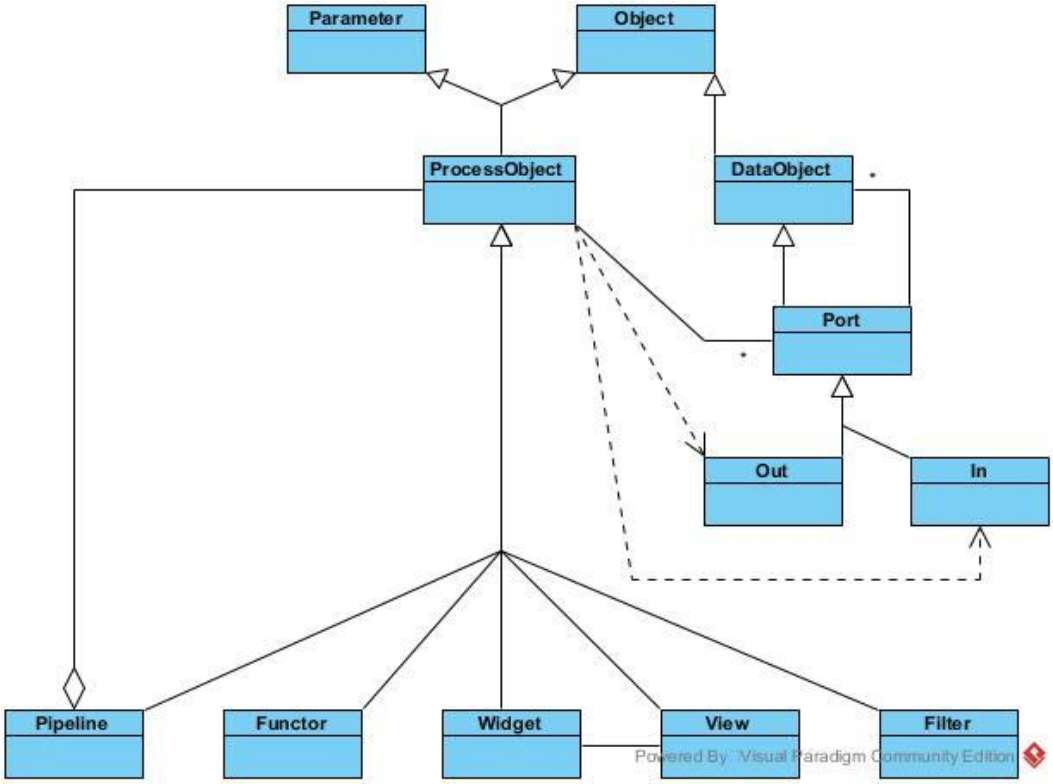

Ilustración 6. Diagrama de dominio base del funcionamiento de los filtros y pipelines de QuimeraTK

5.2.1 Clases base

Nombre	**Parameter**
Descripción	Representa un conjunto de valores escalares.
Herencia	
Nombre	**Object**
Descripción	Clase que implementa un recolector de basura. Es la base de toda la jerarquía. Gracias a esta clase se obtiene el poliformismos que se utiliza en el sistema, además de proporcionar el uso de marcas de tiempo (timestamp) y permite que todo el modelo cpPlugins pueda utilizar el recolector de basura
Herencia	

Tabla 7. Descripción de clases base del modelo

5.2.2 Clases para el manejo de información

Nombre	**DataObject**
Descripción	Clase que representa y proporciona acceso a los datos que son utilizados en las transferencias para los procesos. [6]
Herencia	Object
Nombre	**Port**
Descripción	Clase que representa un espacio por el cual hacer una conexión entre ProcessObjects. Permiten el flujo de datos entre ProcessObjects
Herencia	DataObject
Nombre	**Out**
Descripción	Clase encargada de generar la conexión de salida ya sea hacia otro componente, como la salida final del procesamiento.
Herencia	Port
Nombre	**In**
Descripción	Clase encargada de generar la conexión de entrada hacia un componente.
Herencia	Port

Tabla 8. Descripción de clases relacionadas al manejo de información

5.2.3 Clases relacionadas con los procesos

Nombre	**ProcessObject**
Descripción	Clase que representa un procesamiento de datos. Tiene puertos de entrada que actúan como entrada de datos para ser procesados y un puerto de salida con el resultado del procesamiento. Es decir, ProcessObjects operan en objetos de datos de entrada, produciendo nuevos objetos de datos como salida.
Herencia	Parameter, Object
Nombre	**Pipeline**
Descripción	Clase encargada de generar el puente de comunicación entre componentes.
Herencia	ProcessObject
Nombre	**Functor**
Descripción	Clase que representa un objeto matemático que recibe parámetros y retorna un resultado. Un functor representa un cálculo
Herencia	ProcessObject
Nombre	**Widget**

Descripción	Clase que representa una funcionalidad que requiere de interacción con el usuario. Esta funcionalidad puede ser activada o desactivada. Solo un widget puede estar activo al tiempo. La interacción se da a través de Views
Herencia	ProcessObject
Nombre	**View**
Descripción	Clase que representa una vista de los datos siendo procesados en un ProcessObject. Un View se conecta y funciona como interfaz entre usuario y ProcessObject, haciendo que las interacciones entre el usuario y la vista actúen como entrada a dicho ProcessObject.
Herencia	ProcessObject
Nombre	**Filter**
Descripción	Clase que envuelve un procesamiento el cual se aplica sobre la entrada que recibe, puede arrojar un resultado final o conectar con otro filter para otro procesamiento. Es decir, envuelve la función de un determinado algoritmo para posteriormente ser enviada para su respectivo procesamiento. De esta forma la interacción con el algoritmo se facilita al solo enfocarse en los datos necesarios para ejecutar la función. [6]
Herencia	ProcessObject

Tabla 9. Descripción de clases relacionadas a los procesos

6 Vista de implementación

En esta vista se muestra el sistema de la forma de cómo se debe gestionar del software. Se va a mostrar cómo está dividido el sistema software en componentes y las dependencias que hay entre esos componentes. [5] En el siguiente diagrama se muestra los componentes que constituyen QuimeraTK

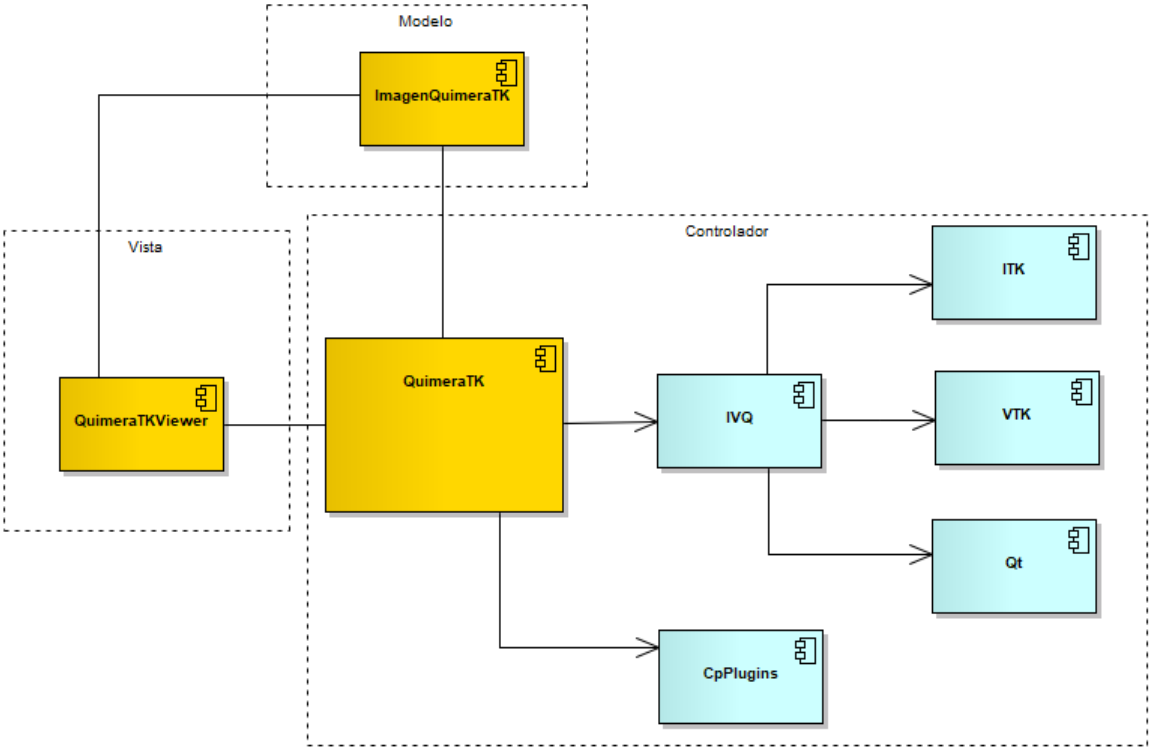

Ilustración 7. Diagrama de componentes de QuimeraTK

Componente	**Descripción**
QuimeraTKViewer	Es el encargado de generar la interfaz gráfica para el médico, donde este podrá manipular la imagen mediante los algoritmos integrados y guardar el resultado final
QuimeraTK	Su objetivo es conectar la interfaz gráfica con los algoritmos de procesamiento de imágenes
ImagenQuimeraTK	Su objetivo es guardar la imagen en el sistema mientras se realiza los procesamientos elegidos por el usuario e informar a la vista los cambios que existan en la imagen
IVQ	Es una un sistema para la integración de pipelines que utiliza ITK-VTK-QT. Este componente proviene de un proveedor

	externo pero es instalado en el computador donde se va a ejecutar QuimeraTK
ITK	Proviene de un proveedor externo
VTK	Proviene de un proveedor externo
Qt	Proviene de un proveedor externo
CpPluggins	Es el encargado de crear los pipelines del sistema. Proviene de un proveedor externo

Tabla 10. Descripción de componentes

7 Vista de Procesos y de Despliegue

7.1 Vista de procesos

Los procesos de QuimeraTK están diseñados para ser usados bajo el patrón de filtros y tuberías (pipelines), y teniendo en cuenta la distribución en las clases y la generación de los procesos representados por objetos dentro del sistema (filter, widget, functor). Así, el uso de los algoritmos que se encuentran integrados dentro del sistema realiza un uso similar de los filtros y los pipelines, como se muestra en la siguiente imagen.

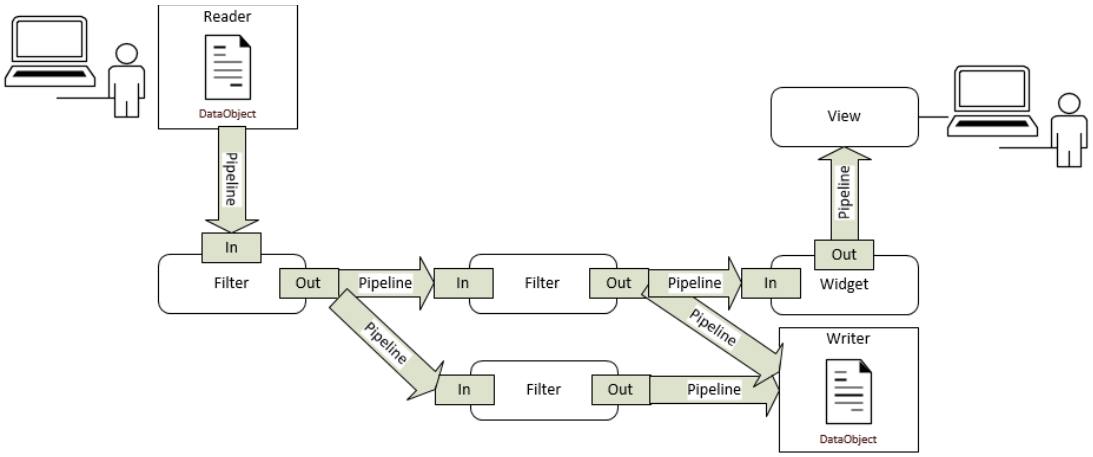

Ilustración 8. Funcionamiento básico de los filtros y pipelines de QuimeraTK

7.2 Vista Fisica

En esta vista se muestra la distribución física del sistema y la topología del hardware. Sin embargo, puesto que QuimeraTK es un sistema standalone, el cual se ejecuta en una sola máquina, solo es necesario un nodo de despliegue para el sistema.

8 Referencias

[1] IBM, «Rational Software Architect for WebSphere Software,» [En línea]. Available: http://www-03.ibm.com/software/products/en/swarchitect-websphere. [Último acceso: 23 Febrero 2016].

[2] IBM, «IBM Rational Software Architect,» [En línea]. Available: http://www.ibm.com/developerworks/downloads/r/rup/index.html. [Último acceso: 23 Febrero 2016].

[3] O. Montenegro, E. Burgos y J. A. Moreno, *DOCUMENTO DE ARQUITECTURA DE SOFTWARE (SAD)*, 2009.

[4] P. Kruchten, «Architectural Blueprints—The "4+1" View Model of Software Architecture,» Noviembre 1995. [En línea]. Available: http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/Pbk4p1.pdf. [Último acceso: 23 febrero 2016].

[5] R. Moya, «Modelo "4+1" vistas de Kruchten,» 31 Marzo 2012. [En línea]. Available: http://jarroba.com/modelo-41-vistas-de-kruchten-para-dummies/. [Último acceso: 23 Febrero 2016].

[6] «The ITK Software Guide». [En línea]. Disponible en: https://itk.org/ITKSoftwareGuide/html/. [Accedido: 06-may-2019].

[7] Y. D. González y Y. F. Romero, «Patrón Modelo-Vista-Controlador.», Rev. Telemática, vol. 11, n.o 1, pp. 47-57, jun. 2012.

[8] M. D. Meyer y D. P. Agrawal, «A modular pipelined implementation of a delayed LMS transversal adaptive filter», en IEEE International Symposium on Circuits and Systems, 1990, pp. 1943-1946 vol.3

[9] J. P. Zuluaga, «Solución GEC.ISO: Software Architecture Document», [En linea]. Disponible en https://pegasus.javeriana.edu.co/~CIS0730IS01/Anexos/Anexo%20D%20SAD.doc [Accedido: 08-may-2019].