PROPUESTA PARA PROYECTO DE GRADO

cpPlugins 3.0

Proyecto de Ingeniería de Software

Diseñar y desarrollar una aplicación, basada en una versión anterior de cpPlugins, que proporcione a los usuarios una interfaz "drag and drop", con el fin de procesar y visualizar datos médicos estructurados.

ESTUDIANTE(S)

Ariza Luna, Pablo Andrés _____

Documento	Celular	Correo Javeriano
CC. 1032496430	319-737-3030	pablo_ariza@javeriana.edu.co

Barón León, Carlos Orlando_____

Documento	Celular	Correo Javeriano
CC. 1015470763	321-488-0959	baron_carlos@javeriana.edu.co

Chaustre Perdomo, Santiago_____

Documento	Celular	Correo Javeriano
CC. 1105792535	319-536-1346	santiago-chaustre@javeriana.edu.co

Cocunubo Quintero, Andrés Felipe_____

Documento	Celular	Correo Javeriano
CC. 1022434965	320-416-4194	andres-coconubo@javeriana.edu.co

Rojas Bohórquez, Ailin Ana María_____

Documento	Celular	Correo Javeriano
CC. 1018493614	320-209-2864	ailin.rojas@javeriana.edu.co

DIRECTOR

Leonardo Flórez Valencia PhD. _____

Documento	Teléfono fijo	Correo Javeriano	Empresa donde trabaja y cargo
CC.	3208320 ext 5317	florez-l@javeriana.edu.co	Pontificia Universidad Javeriana; Profesor Departamento de Sistemas

ASESOR

Jaime Andrés Pavlich Mariscal PhD. _____

Documento	Teléfono fijo	Correo Javeriano	Empresa donde trabaja y cargo
CC.	3208320 ext 4731	jpavlich@javeriana.edu.co	Pontificia Universidad Javeriana; Profesor Departamento de Sistemas

Contenido

1 Visión global

1.1 Antecedentes, problema y solución propuesta

1.1.1 Descripción de la problemática u oportunidad

A lo largo del tiempo el ser humano ha demandado el uso de computadores para un gran número de tareas. Sin embargo, la facilidad para realizar estas tareas y su alcance han aumentado gracias a varios factores ligados a la computación gráfica, como la interacción por GUI (Graphical User Interface) o la manipulación directa de objetos gráficos [1]. Algunas disciplinas han tomado ventaja de estas capacidades de la computación para analizar conjuntos de datos de dos o tres dimensiones. Por ejemplo, disciplinas como la medicina, las geociencias y las ciencias de los materiales usan frecuentemente herramientas para la visualización y procesamiento de datos [2].

El análisis y visualización de imágenes médicas ha crecido rápidamente y de igual forma, ha ganado mucha popularidad en varios campos de la medicina [3], como la planificación quirúrgica y radioterapia [4]. A raíz de esto, la medicina ha tenido grandes avances. Se ha pasado de evaluar imágenes médicas bidimensionales, que muestran solo cierta capa de información, a evaluar imágenes médicas en tres dimensiones, con las cuales se puede obtener una visualización completa [5]. Por lo tanto, cada vez que más dispositivos adquieren datos médicos, se necesitan muchos más algoritmos para su procesamiento y herramientas para poder manejarlos y visualizarlos de forma interactiva [3].

Muchas empresas, como SIEMENS, Philips y GE han desarrollado aplicaciones para el procesamiento y visualización de imágenes médicas. El problema es que estos softwares no son abiertos al público y sólo admiten el formato estándar, DICOM, y el formato interno de sus compañías [6].

Ante esta problemática, como parte de la solución, nacen herramientas como The Insight Segmentation and Registration Toolkit (ITK) y se unen otras como The Visualization Toolkit (VTK) [7]. ITK es una librería para el análisis de datos de imágenes médicas [8] y VTK es un software libre que permite realizar procesamiento, visualización y generación de gráficos 3D [9].

Sin embargo, para hacer uso de estas herramientas se requiere tener conocimientos de programación, conceptos avanzados de C++ y dominio de los datos de ITK [7]. Entonces, uno de los principales problemas radica en que los potenciales usuarios de estas herramientas, tales como radiólogos, pueden poseer poca o nula experiencia en la programación y estas herramientas no brindan una interfaz gráfica [10].

Debido a la necesidad de una interfaz gráfica, se han creado aplicaciones para solucionar este problema, como por ejemplo ITKBoard, MeVisLab y MITK [7]. Sin embargo, estas herramientas están desarrolladas para trabajar con una sola tecnología, lo cual brinda muy poca flexibilidad limitando al usuario final a trabajar con las funcionalidades que determinen los desarrolladores de cada aplicación.

Como parte a la solución del problema nace cpPlugins, un framework de alto nivel multiplataforma para encapsular y administrar estructuras de flujos de datos basadas en ITK, VTK y Qt [11]. CpPlugins cuenta con varios componentes para la visualización y procesamiento de datos médicos, entre los que se encuentran filtros y artilugios (widgets).

- Los filtros cuentan con entradas y salidas, los cuales permiten realizar procesamiento de datos.
- Los artilugios permiten interactuar con los datos para captar las entradas del usuario y visualizar información, con el objetivo de observar de manera gráfica un flujo de datos [12].

La versión más reciente de cpPlugins posee un buen diseño de la arquitectura, sin embargo, su implementación requiere de varias funcionalidades relacionadas con la interacción, la visualización, la interfaz gráfica y el procesamiento, para así conseguir una implementación más estable y usable [12].

Finalmente, teniendo en cuenta lo anterior, este trabajo busca ofrecer una solución completa a los problemas expuestos por medio de un software que permita procesar y visualizar datos de manera gráfica. Adicionalmente, brindar la posibilidad de agregar nuevas funcionalidades, y ofrecer una plataforma en donde se puedan aprovechar todas las capacidades del software, para que este pueda ser usado en diferentes contextos.

1.1.2 Formulación del problema

A continuación, se presentan los problemas más relevantes en el ámbito del procesamiento y visualización de imágenes médicas:

- **Usabilidad**
  - Herramientas de uso privado, rígidas y sesgadas a intereses de las compañías.
  - Se necesita un nivel intermedio-avanzado de programación para usar las herramientas de código abierto, como ITK y VTK.
  - Algunas de las aplicaciones de software libre limitan a los usuarios a las características que estas mismas ofrecen.
- **Extensibilidad**
  - Algunas de las aplicaciones de software libre, dificultan el desarrollo e integración de nuevos algoritmos a su sistema.
  - Existen muchos algoritmos de procesamiento de imágenes, para diferentes fines y no existe una unificación entre ellos.
- **Portabilidad**
  - Todas aplicaciones están sesgada a un ambiente de escritorio muy específico.
  - Para la realización de análisis sobre imágenes médicas se necesitan máquinas costosas, las cuales son la única alternativa para ejecutar este aplicativo.

Tomando en consideración todos los problemas mencionados, se plantea una pregunta que encapsula el punto de partida para tener en cuenta al proponer una solución:

"¿Como hacer una herramienta flexible, que permita a un usuario sin conocimientos de programación, realizar procesamientos de flujo de datos estructurados?"

1.1.3 Propuesta de solución

En el marco general del procesamiento de imágenes n-dimensionales, se propone crear una solución que permita diseñar de manera gráfica y sencilla, flujos de datos para ser procesados con diferentes algoritmos seleccionados por el usuario, utilizando para ello el paradigma de programación orientada por eventos.

Para entender la solución, es necesario familiarizarse con dos conceptos base. El primero de ellos se denominará filtro, el cual puede pensarse como una caja negra que se encarga de procesar una entrada y transformarla utilizando algún algoritmo de procesamiento de imágenes haciendo uso de primitivas provistas por *ITK*. El segundo concepto, será denominado tubo, esté último se utilizará para copiar la salida de un filtro en la entrada del siguiente [13]. Estos dos conceptos, se alinean perfectamente en la creación de un *pipeline*, el cual es un conjunto de tubos y filtros interconectados entre ellos para la ejecución de un proceso especifico [14], en este caso el procesamiento y visualización de imágenes médicas.

cpPlugins 3.0, pretende brindar una solución en dos partes. La primera de ellas consiste en un T*oolkit*, que permita integrar nuevos componentes y adaptarlos al sistema. La finalidad de este *Toolkit*, además de brindar una arquitectura modular, es también proveer una interfaz de comunicación, de modo que pueda ser incorporado a otras soluciones a manera de *Plug-In*, para lograr una fácil integración con sistemas externos. Adicional a esta característica, la generación de aplicativos a partir del diseño de tubos y filtros ensamblados por el usuario permite una mayor portabilidad, simplificando la labor de procesar datos con esta configuración cada vez que se necesite.

La segunda parte, consiste en proporcionar una interfaz gráfica, que permita diseñar un flujo de datos, mediante un sistema "*Drag and Drop*", en el que el usuario arrastra los filtros al espacio de trabajo y allí modifica sus parámetros, para luego realizar las conexiones pertinentes entre ellos. Esta interfaz integrada también permitirá visualizar las salidas de los filtros, utilizando para ello *VTK*. De la misma manera, debe facilitar la tarea de cargar pipelines existentes y de persistir los diferentes diseños que se realicen.

Para obtener el aplicativo planteado, se implementará un servidor con cpPlugins y un conjunto de servicios web, que faciliten la comunicación con el cliente, para que este pueda acceder y hacer uso de sus funcionalidades a través de un navegador. En el diagrama de la *Figura 1*, se aprecia la arquitectura que tendría esta solución, integrando una interfaz gráfica de usuario representada por la palabra GUI, un servicio web representado en el diagrama como WS (Web Service), el cual encapsula el comportamiento de un filtro y puede ser consumido desde la máquina del usuario.

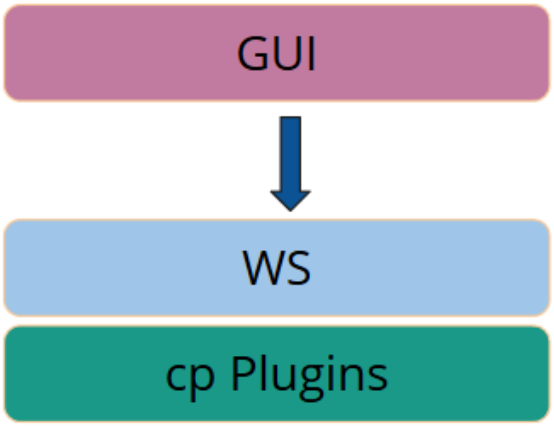

Figura 1. Aplicación Web cpPlugins 3.0

1.1.4 Justificación de la solución

La solución propuesta, es adecuada al problema planteado, debido a que satisface de manera conjunta los problemas de extensibilidad, portabilidad, procesamiento y visualización de datos médicos estructurados de forma gráfica. Por medio de un *Toolkit* modular dotado de esta interfaz gráfica que permite al usuario el uso de tubos y filtros para

crear flujos de procesamiento de datos de manera rápida, y del mismo modo integrar nuevos componentes al sistema original.

La solución propuesta, además considera la aplicación de diferentes paradigmas de programación orientados a la interacción con el usuario. Entre los considerados, se incluye el paradigma de programación reactiva funcional (FRP), el cual es útil en el momento que el usuario modifique los parámetros de entrada de un componente que ya ha procesado la información, posteriormente la aplicación extenderá los cambios a los componentes que se encuentran adelante del que fue modificado [15]

También se consideró el paradigma de programación orientada a eventos, en el que el comportamiento del programa se ve afectado directamente por las acciones realizadas por el usuario. Esto se verá reflejado en los diferentes flujos de procesamiento de datos creados y sus resultados. Además, para complementar, el diseño de pipelines se basa en el paradigma de programación orientada a flujos de datos, ya que cada uno de ellos se representa con un grafo, donde cada vértice es simbolizado como un filtro y está orientado a paralelismo [16].

Para finalizar esta sección, se eligió el paradigma de programación basado en flujos, debido a que engloba los conceptos mencionados anteriormente y su principal funcionamiento se basa en el flujo de datos de un nodo a otro, a diferencia del paradigma de programación basado en bloques, donde sus bases nacen de la programación imperativa y se asemeja mucho más a los lenguajes de programación tradicionales basados en texto [17].

Para escoger la solución adecuada, se consideró una implementación *Stand alone* en comparación con una basada en *Web Services,* tomando en cuenta como factor diferenciador el rendimiento y la portabilidad que pudiera brindar cada una de ellas. Para solucionar el problema de portabilidad, la opción que más favorable es la aplicación web. Ya que, sería compatible con múltiples ambientes y plataformas, de tal manera que el usuario final no debe preocuparse por el entorno en el que se ejecute el aplicativo, y todo el ámbito técnico debe ser transparente para él, ya que solo se necesita de un navegador web para su ejecución. Mientras que, la opción *stand alone* facilita la extensibilidad.

En cuanto al rendimiento, se concluyó que el aplicativo web, daría un mejor rendimiento, ya que en la opción *Stand alone,* existiría una dependencia de las características técnicas de la maquina donde se ejecuta. Por lo tanto, *cpPlugins 3.0,* se implementará a modo de aplicación web.

1.2 Objetivos

1.2.1 Objetivo general

Diseñar y desarrollar una aplicación, basada en una versión anterior de cpPlugins, que proporcione a los usuarios una interfaz “drag and drop”, con el fin de procesar y visualizar datos médicos estructurados.

1.2.2 Objetivos Específicos

- Elaborar la arquitectura de cpPlugins 3.0 para identificar los puntos extensibles.
- Desarrollar un cliente web para acceder a la funcionalidad de cpPlugins 3.0 de manera ligera en diferentes plataformas.
- Identificar atributos de calidad para determinar los recursos físicos que harán parte del sistema.
- Validar con el usuario final las funcionalidades del sistema, brindando un conjunto de algoritmos para el procesamiento de datos médicos estructurados.

1.3 Entregables, estándares utilizados y justificación

A continuación, se define en la tabla 1 la lista de documentos a entregar al final del proyecto. Para cada documento se especifica el o los estándares en los cuales se basa.

Entregable	Estándares asociados	Justificación
Propuesta de trabajo de grado		Documento que contiene los aspectos claves sobre el proyecto a desarrollar, para la aprobación y presentación formal del mismo. Este entregable es el punto de inicio para el desarrollo de trabajo de grado.
SPMP	ISO/IEC/IEEE 16326: 2009 ISO/IEC/IEEE 12207:2017 ISO/IEC 16085:2006 ISO/IEC/IEEE 24748	Documento en el cual se especifica el plan de gestión y proceso de desarrollo del proyecto. Este entregable se basa principalmente en el estándar 16326 de la ISO/IEC e IEEE, ya que provee especificaciones de contenido normativo para los planes de gestión de proyectos de software [18]. Adicionalmente, se utilizarán aspectos del estándar 12207 de 2017, para la definición de los procesos del ciclo de vida del software, actividades y tareas que deben aplicarse durante el desarrollo del producto [19]. Así mismo, para la gestión de riesgos del proyecto se utilizará el estándar 16085 de 2006, para definir el proceso que permita identificar posibles problemas administrativos y técnicos antes de que ocurran, de modo que se puedan tomar medidas que reduzcan o eliminen la probabilidad y/o el impacto de estos problemas en caso de que ocurran [20]. Del estándar 24748 se tomará información sobre los conceptos del ciclo de vida, las descripciones de los propósitos y resultados de las etapas representativas de este [21].
SRS	ISO/IEC/IEEE 29148:2018 DICOM 2019 ISO/IEC 25000:2014	Documento que comprende la especificación de los requisitos de software para el desarrollo del proyecto. Tiene como propósito recopilar y detallar los requisitos funcionales y no funcionales del sistema que constituyen la base para modelar y codificar de manera acertada el software a desarrollar. El estándar principal en el que estará basado este entregable es el 29148 de 2018 de la ISO/IEC e IEEE. Este define la construcción de un buen requisito, proporciona atributos y características de los requisitos y analiza la aplicación iterativa y recursiva de los procesos de requisitos a lo largo del ciclo de vida [22]. Al ser el producto una aplicación de visualización y procesamiento de imágenes médicas, es un requisito usar el estándar DICOM (Digital Imaging and Communications in Medicine) de 2019 [23]. Para el control de calidad del producto, se utilizará el estándar

		ISO/IEC 25000 de 2014, para la especificación de requisitos de calidad del software y evaluación de la calidad de este [24].
SDD	IEEE Std 1016:2009 UML 2.5	Documento que contiene la descripción de diseño de software. Tiene como objetivo dar a conocer la definición de la estructura y comportamiento del sistema. Además, de los detalles más importantes del diseño a bajo nivel de este. Se basa en el estándar 1016 de 2009 de la IEEE, que especifica el contenido de información requerido y la organización para las descripciones de diseño de software (SDD) [25]. Para el modelado de los diagramas se utilizará el estándar UML (Unified Model Language) 2.5. Porque es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema [26].
Memoria		Documento que recopila los aspectos más importantes del proyecto, entre ellos la especificación del problema, el diseño de la solución, su desarrollo, resultados y pruebas. Este entregable certifica el proceso realizado durante las asignaturas planeación de proyecto de grado y trabajo de grado.
Plan de pruebas	ISO/IEC/IEEE 29119-1 ISO/IEC/IEEE 29119-2 ISO/IEC/IEEE 29119-4	Documento que comprende el plan de pruebas de integración continua a ejecutar durante la fase de desarrollo del software. Este entregable se basa en el estándar 29919 de la ISO/IEC e IEEE, es compatible con el diseño y la ejecución de casos de prueba durante cualquier fase o tipo de prueba. La primera parte de este estándar especifica definiciones y conceptos en las pruebas de software [27]. La segunda parte comprende la definición de los procesos de pruebas de software [28]. Por último, la cuarta parte cubre los aspectos más relevantes de las técnicas existentes para realizar pruebas. Define las técnicas de diseño de prueba que se pueden utilizar durante el diseño de la prueba y el proceso de implementación [29].
Reporte de pruebas	ISO/IEC/IEEE 29119-3	La tercera parte del estándar 29119, establece diferentes plantillas de documentación de pruebas de software que pueden ser utilizadas por cualquier proyecto o actividad de prueba más pequeña [30].
Código fuente	Code Conventions for the	Comprende los diferentes archivos que constituyen la implementa-

	Java ™ Programming Language Revised April 20, 1999 JavaScript Standard Style React Standard Style code snippets.	ción de la solución. Se utilizará el estándar de Java para la codificación de los servicios que se ofrecerán, ya que mejora la legibilidad del código permitiendo que los desarrolladores lo entiendan más rápido [31]. Así mismo, para la codificación de la parte web se usará el estándar de JavaScript, dado que ahorra tiempo en la revisión del código para el contribuidor y el auditor [32]. Adicionalmente, se utilizará el estándar de React para el estilo de código de fragmentos. React tiene sus propias guías para la estructuración de código [33].
Producto	Git workflow guideline Semantic Versioning Specification (SemVer) **2.0.0**	Este entregable hace referencia al conjunto de archivos y documentos que soportan el producto final obtenido. Por ende, se utilizarán estándares para las diferentes versiones generadas, con el objetivo de mantener el registro de cambios y orden en los distintos repositorios. Para el desarrollo del producto se utilizará el Git Feature Branch Workflow porque es una recomendación de cómo usar Git para realizar el trabajo de manera consistente y productiva [34]. Adicionalmente, se hará uso de la especificación de versionamiento semántico. Ya que, establece reglas y requisitos que determinan cómo se asignan y aumentan los números de este [35].
Manual de usuario	IEEE Std 1063:2001 (Superseded by ISO/IEC/IEEE 26512 First Edition, 2011-06-01)	Documento que comprende todas las indicaciones paso a paso que debe seguir un usuario para poder usar el programa. Se basa en el estándar 1063 de la IEEE, que establece los requisitos mínimos para la estructura, el contenido de la información y el formato de la documentación del usuario [36].

2 Análisis de Impacto

2.1 A corto plazo

A corto plazo, el impacto que tendrá este proyecto se verá reflejado en *cpPlugins* brindando una mejora considerable a su sistema, mediante una interfaz de interacción más completa y útil hacia el usuario final sin conocimientos de programación. Además, agrega mayor documentación sobre su arquitectura e interfaces de comunicación, facilitando la integración de nuevos plug-ins y otros sistemas.

2.2 A mediano plazo

Se espera que esta aplicación funcione como base para futuros proyectos. Esto con el fin de ofrecer una oportunidad de mejora al proyecto en sí mismo, tal como se ha venido desarrollando a lo largo de los años. También, se espera brindar una herramienta que sea de utilidad en el área de investigación en torno al procesamiento y visualización de imágenes médicas, de tal modo que pueda ser utilizada tanto por estudiantes interesados en la computación visual, como para aquellos inmersos en el área de la medicina.

2.3 A largo plazo

A largo plazo, se espera que el proyecto tenga un impacto positivo en la comunidad médica, siendo una herramienta útil que le permita a los médicos tomar decisiones de manera más rápida, con base en las imágenes médicas diagnósticadas de los pacientes. Así mismo, ayudará a los investigadores de esta área a encontrar implementaciones de diferentes algoritmos, sin tener que indagar en el código fuente de este. Ya que, *cpPlugins 3.0,* provee un framework flexible con gran cantidad de funcionalidades útiles que pueden ayudar en la detección de enfermedades o anomalías, facilitando el análisis y procesamiento de datos médicos estructurados.

3 Proceso

Para el desarrollo del proyecto se utilizarán elementos tanto de la metodología ágil Scrum como Kanban y se adaptarán a las necesidades del equipo. En la siguiente ilustración se observa la implementación de dichas metodologías.

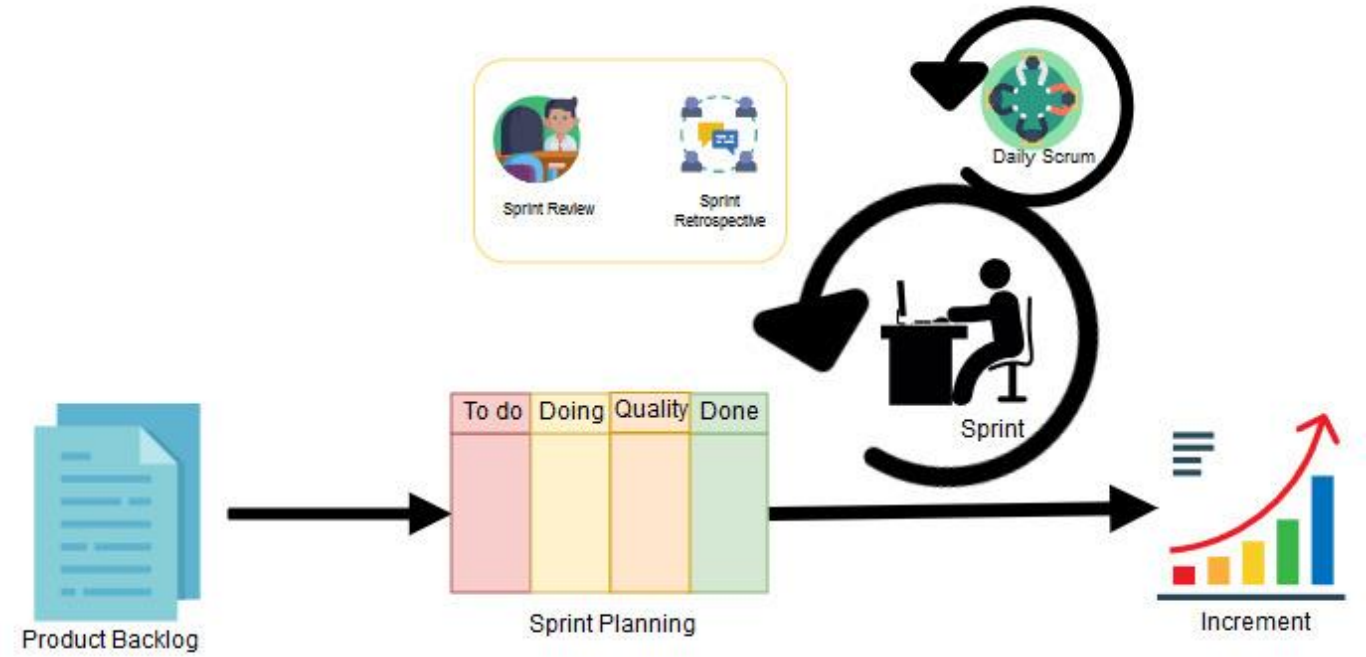

Ilustración 1-Metodología aplicada

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente en equipo, y obtener el mejor resultado posible en un proyecto [37]. Esto resulta útil en el desarrollo de este, ya que una de las necesidades del equipo es presentar avances constantes al director, con el objetivo de tener una retroalimentación y determinar las actividades que se realizarán posteriormente. La metodología Kanban ayudará al desarrollo del proyecto, de acuerdo a los tres principios que brinda: visualización de lo que está ocurriendo, limitación de trabajo en curso y mejora la continuidad del trabajo [38]. Permitiendo tener claro los avances que se obtienen a medida que transcurre el tiempo, evitando la sobrecarga de tareas de los miembros y los tiempos muertos.

Es importante aclarar que cada una de las fases de la metodología escogida se adaptará a las necesidades del equipo de trabajo, es decir, algunas actividades como por ejemplo el daily scrum no se realizará diariamente, sino en el tiempo que se determine por los integrantes del grupo.

3.1 Fase metodológica 1: Planificación del proyecto

Esta fase metodológica comprende todo lo relacionado con la concepción, los objetivos, la planeación de las diferentes actividades y lineamientos correspondientes que se van a realizar durante el desarrollo del proyecto.

3.1.1 Método

Aquí se elabora el plan y administración del proyecto (Software Project Management Plan, por sus siglas en inglés SPMP), al ser una fase corta se usarán los tableros de Kanban de una manera reducida. Cada tarjeta representará la actividad a realizar, se establecerá la cantidad máxima de tarjetas por cada estado (por hacer, haciendo, control de calidad y hecho) y al final se determinarán las actividades para la próxima semana.

3.1.2 Actividades

- Descripción del proyecto: elaborar una introducción que permita entender qué es lo que se busca con el documento, explicando el propósito y el alcance del software que se va a desarrollar.
- Vista general: descripción de los aspectos más importantes del producto.
- Glosario: definición de conceptos y siglas fundamentales para el entendimiento del proyecto.
- Contexto del proyecto: explicación de los aspectos más importantes del funcionamiento interno del proyecto, así como la comunicación con entidades externas.
- Monitoreo y control: presentar los diferentes planes que sean pertinentes para controlar el desarrollo del proyecto y realizar la evaluación necesaria.
- Entrega del producto: presentar y describir los pasos a seguir al momento de la entrega del producto final.
- Procesos de soporte: presentar, documentar y describir a los miembros del equipo los diversos planes transversales para tener en cuenta al momento de realizar los distintos procesos del proyecto.

3.1.3 Resultados esperados

Elaboración del SPMP, en el que se describe el proyecto, además de establecer un esquema de trabajo para los miembros del equipo.

3.2 Fase metodológica 2: Definir requisitos

Se elabora la especificación de requisitos de software (Software Requirements Specification, por sus siglas en inglés SRS), con el objetivo de conocer las funcionalidades que tendrá el sistema. Con esto se pueden tomar decisiones relevantes del proyecto como la selección de herramientas pertinentes, modelos de datos, transformaciones, entre otros aspectos.

3.2.1 Método

Al igual que la Fase Metodológica 1 se usarán los tableros de Kanban, sin embargo, se debe tener en cuenta las limitaciones de tiempo que establezca el cliente.

3.2.2 Actividades

- Preparar la entrevista: programar lugar, fecha, temas a conocer y definir las preguntas que se realizarán al cliente.
- Realizar entrevista al cliente.
- Realizar una retroalimentación con el cliente: validar los requerimientos definidos con este.
- Formalización de requerimientos en diagrama de CU.
- Documentar los requerimientos: documentar detalladamente los requisitos aceptados
- Priorizar los requerimientos: evaluar los requerimientos, con el fin de conseguir un orden de implementación, de acuerdo con los recursos con los que se cuenta o sus limitaciones.

3.2.3 Resultados esperados

Elaboración del documento SRS que especifica los requerimientos del producto, los cuales deben satisfacer al cliente y deben poder ser alcanzables por los desarrolladores.

3.3 Fase metodológica 3: Diseño

En esta fase se elabora la descripción del diseño de software el cual busca dar a conocer la arquitectura del sistema en un alto y bajo nivel de abstracción, utilizando como soporte las vistas lógica y física.

3.3.1 Método

Al igual que la Fase Metodológica 1 se usarán los tableros de Kanban.

3.3.2 Actividades

- Descripción global del sistema: diagrama de contexto que muestra de forma general el producto.
- Definir la arquitectura con el fin de entender la estructura y comportamiento del sistema a un alto nivel de abstracción.
- Descripción detallada de los diferentes componentes, del comportamiento y la persistencia de los datos del sistema.
- Descripción de diseño detallado de bajo nivel, en donde se muestren diagramas de clases de la aplicación.
- Elaborar el árbol de navegación que contiene las pantallas más importantes del software.

3.3.3 Resultados esperados

Elaboración del documento de diseño de software (Software Design Document, por sus siglas en inglés SDD), donde se detalle cada una de las actividades descritas anteriormente, que permitan al equipo de trabajo comenzar a desarrollar la implementación de la solución propuesta, de una manera estructurada.

3.4 Fase metodológica 4: Desarrollo e integración

En esta fase se desarrollará el producto aplicando la metodología ágil explicada al inicio de esta sección. Además, de unificar el código que realice cada miembro del equipo en una versión estable del software.

3.4.1 Método

Esta fase se realiza siguiendo la metodología Scrum modificando el sprint planning por los tableros Kanban. Al finalizar cada sprint, será importante las retroalimentaciones hechas tanto por el director del proyecto como de los miembros del equipo, estableciendo acciones de mejora para el próximo sprint en caso de ser necesario.

3.4.2 Actividades

- Elaboración de los tableros Kanban antes de empezar cada sprint.
- Desarrollar las actividades establecidas en el tablero.
- Comentar problemas presentados durante los daily scrum.
- Ejecutar pruebas de integración continua.
- Presentar y validar funcionalidades realizadas al director del proyecto.
- Integrar a la versión estable de la aplicación las funciones aprobadas por el cliente.

3.4.3 Resultados esperados

Avance del proyecto y versión estable del producto.

3.5 Fase metodológica 5: Presentación del producto

Presentación final del producto, en donde se realizarán las configuraciones para el correcto funcionamiento. Además de entregar la documentación realizada durante el desarrollo del proyecto.

3.5.1 Método

Elaboración de tableros Kanban para las actividades relacionadas con detalles finales del producto y manuales de usuario.

3.5.2 Actividades

- Configuración del sistema: se realizan los ajustes necesarios para la presentación del producto.
- Presentación del producto: realizar una muestra del sistema desarrollado, demostrando el cumplimiento de los objetivos que se buscaban.
- Elaboración de manuales de usuario.
- Entrega de la documentación del proyecto.

3.5.3 Resultados esperados

Presentación funcional del producto y entrega formal de documentos.

4 Aspectos generales del proyecto

4.1 Compromiso de apoyo de la Institución

El proyecto cuenta con el apoyo de la Pontificia Universidad Javeriana.

4.2 Derechos patrimoniales

Los derechos patrimoniales del proyecto serán propiedad de los estudiantes que desarrollan el trabajo de grado, además hace parte del grupo de investigación BASPI de la Pontificia Universidad Javeriana; por lo tanto, los derechos serán compartidos con la institución.

El código se liberará al público bajo la licencia CECILL v2 (acrónimo en francés de "CEA CNRS INRIA Logiciel Libre"), esta es una licencia de software libre, explícitamente compatible con la GPL y GNU [39]. El licenciamiento, permite distribuir copias del software según los términos y condiciones establecidos. Para el licenciamiento de los documentos se utilizará la licencia Creative Commons Attribution 4.0 International (CC BY 4.0) [40], la cual permite compartir y adaptar a partir de los documentos creados, siempre que la persona que los utilice otorgue el crédito a los autores originales de los documentos.

5 Marco teórico

5.1 Fundamentos y conceptos relevantes para el proyecto

El primer concepto importante a tener en cuenta durante la elaboración de cpPlugins 3.0, es *imagen médica*. Una imagen médica se define como una representación del cuerpo humano o partes del mismo, elaboradas a partir de un conjunto de técnicas y procesos establecidos, con el fin de exponer, diagnosticar o examinar enfermedades, o realizar estudios científicos médicos [41].

Cuando se habla de imágenes médicas, en la mayoría de los casos, se relaciona directamente con la radiografía. Una radiografía se define como una representación del cuerpo humano o una parte de él, obtenida a partir de la radiación electromagnética ionizante de los rayos X. Existen diferentes técnicas de la radiografía y otras que, aunque técnicamente no pertenecen a la radiografía, se relacionan a la misma, ya que los departamentos de radiología de los hospitales también trabajan con ellas. Entre estas técnicas se pueden encontrar, tomografías, resonancias magnéticas (MRI), ultrasonografías, entre otras [41, p.].

Como se plantea en el objetivo general (sección 1.2.1), cpPlugins 3.0 pretende brindar una interfaz interactiva para procesar datos médicos estructurados. Se hace uso del término *datos médicos estructurados*, debido a que, para la codificación, almacenamiento y transmisión de imágenes médicas, se usan estándares como DICOM, en donde se almacena información del paciente, la estructura topológica y geométrica del paciente. Por ejemplo, el estándar DICOM posee información en su cabecera acerca del nombre del paciente, el sexo del paciente y otra información relevante y relacionada con el tipo de la imagen médica. Por otro lado, el estándar DICOM también permite almacenar información de las imágenes médicas, como sus dimensiones, el tipo de escáner con el que fue tomada la imagen, las condiciones en las que se tomó la imagen, su formato de compresión, el tipo de imagen médica (e.g. radiografía), la parte de cuerpo a la que pertenece la imagen, entre otros [41, p.].

Para el procesamiento y la visualización de datos médicos estructurados, cpPlugins hace uso de ITK y VTK respectivamente [12].

En primer lugar, ITK *"es un software de código abierto, orientado a objetos para la segmentación y registro de imágenes"* [8]. Por un lado, *"la segmentación es el proceso de identificación y clasificación de datos encontrados en una representación digital"* [8]. Mientras que *"el registro es la tarea de alinear y desarrollar relaciones entre los datos. Por ejemplo, en las imágenes médicas esto se utiliza para encontrar relaciones entre una tomografía computarizada y una resonancia magnética"* [8].

En cuanto a VTK, este *"es un software de código abierto para manipular y mostrar datos científicos. Viene con herramientas de última generación para la representación 3D, un conjunto de widgets para la interacción 3D y una amplia capacidad de trazado 2D"* [9]. Estas funciones de VTK son de gran utilidad a la hora de interactuar y visualizar imágenes médicas.

Para el desarrollo de cpPlugins 3.0, es indispensable hacer uso del concepto de programación de flujos de datos (DFP) debido a que está altamente relacionado con la forma en que se pretende permitir que un usuario diseñe una estructura de procesamiento de datos. La programación de flujo de datos es un paradigma de programación que permite representar programas informáticos como un grafo directo, en donde los nodos son llamados bloques, que comprenden ciertas operaciones, y poseen puertos de entrada y salida. Los bloques son conectados de salidas a entradas por medio de aristas directas que representan el flujo de la información [16].

Muchos lenguajes de programación visual están basados en el paradigma de la programación de flujos de datos [16]. La programación visual se refiere al proceso de construir programas informáticos por medio de representaciones gráficas en lugar de código. Estas representaciones gráficas pueden ser figuras geométricas, figuras diferenciadas por color, entre otras [10].

Otro concepto para tener en cuenta durante el desarrollo de cpPlugins 3.0 es la programación dirigida por eventos ya que parte del objetivo general es desarrollar una interfaz altamente interactiva. La programación dirigida por eventos es un estilo para construir software en donde los programadores registran un conjunto de eventos y programan la respuesta estos en lugar de definir una secuencia de pasos ordenada [15].

De acuerdo con el diseño de cpPlugins 2.0, se plantea una arquitectura de pipes and filters, para el procesamiento de los datos, y una arquitectura modular, para garantizar la extensibilidad [12].

Pipes and filters *"es un patrón de arquitectura que provee una estructura para los sistemas que procesan flujos de datos de forma secuencial"* [42]. En donde, se le conoce como filter a una unidad en donde se enriquecen, se refinan o se transforman los datos de entrada. Por otro lado, un pipe es el canal de comunicación o el flujo de información entre filters [42].

Por último, la arquitectura modular permite que un software se pueda subdividir en pequeñas partes, conocidas como módulos, de manera que se logré que cada parte sea independiente del sistema y de otras partes. Cada módulo resuelve un problema específico y permite la reutilización [43]. A partir de esto, surgen los conceptos de wrapping y plug-ins. El wrapping es una técnica que busca traducir solo la interfaz de una librería de programación, de manera que pueda ser usada en otro lenguaje conservando la implementación en el lenguaje de programación original [7]. Por otro lado, un plug-in es un paquete de software que permite agregar funcionalidades a una arquitectura de extensibilidad bien especificada [44].

5.2 Trabajos importantes en el área

A continuación, se exponen diferentes trabajos que han solucionado un problema similar al que ataca cpPlugins 3.0, o trabajos cuya solución puede realizar un aporte al diseño de la solución de cpPlugins.

Después de realizar una breve descripción de cada trabajo, se procede a realizar un análisis de cada uno, para identificar aspectos importantes que proponen en la solución al problema que plantean.

5.2.1 ITKBoard

ITKBoard es una herramienta que brinda a usuarios, sin conocimiento en programación, la posibilidad de acceder a todas las funcionalidades que brinda ITK por medio de la programación visual y el prototipado rápido. Para lograr su objetivo ITKBoard brinda a los usuarios una interfaz gráfica en donde pueden estructurar flujos de datos de ITK arrastrando elementos visuales que representan las diferentes funcionalidades de ITK [10].

5.2.2 VTK Designer

VTK Designer es una plataforma que permite estructurar, crear y modificar aplicaciones en VTK para visualizar datos científicos. VTK Designer nace de la necesidad de poder ensamblar componentes de VTK sin tener que poseer todos los conocimientos necesarios para poder programar redes de visualización en C++. Provee mecanismos "drag and drop" para la creación y ejecución de estas redes [45].

5.2.3 MeVisLab

Al igual que los trabajos descritos anteriormente, MeVisLab también nace de la necesidad que se posee en diferentes contextos de estructurar flujos de datos en VTK e ITK de manera visual, por medio de diseño "drag and drop" y representaciones visuales de las diferentes herramientas que provee. Sin embargo, MeVisLab pretende brindar al usuario una plataforma flexible que permita al usuario la integración de ITK y VTK de forma genérica, para así evitar problemas a la hora de introducir nuevos módulos o actualizar a una nueva versión de las librerías mencionadas [3].

5.2.4 SimITK

Al igual que ITKBoard, SimITK es una plataforma que brinda a los usuarios la posibilidad de programar visualmente flujos de datos en ITK. Sin embargo, ITK tiene una particularidad, provee su funcionalidad envuelta en bloques del entorno de programación visual de MATLAB, Simulink [7].

MATLAB es un entorno de desarrollo que provee un lenguaje de programación para el análisis número iterativo y procesos de diseño [45]. Por otra parte, Simulink es un complemento de MATLAB que brinda una caja de herramientas para el sistema de programación visual [7].

5.2.5 ParaView Web

ParaView es una aplicación de código abierto para la visualización y análisis de datos. Nace de la necesidad de analizar conjuntos de datos extremadamente largos por medio de recursos de cómputo distribuidos. A partir de ParaView, surge ParaView Web, el cual es un framework que permite a desarrolladores construir aplicaciones web que utilicen visualización interactiva en tres dimensiones. ParaView Web es caracterizado por permite procesar largos conjuntos de datos sobre la web [46].

5.2.6 MITK

MITK es un framework de código abierto desarrollado para proveer de manera sencilla acceso a las funcionalidades de VTK e ITK para el desarrollo de software que soporte al análisis de imágenes médicas con una alta interacción. Por lo tanto, MITK no es un software que pretenda remplazar VTK e ITK, si no una extensión que facilite la integración de ambos y permita agregar funcionalidades que no sean compatibles con los mismos [47].

5.2.7 cpPlugins 1.0

cpPlugins 1.0 es un framework de alto nivel diseñado para encapsular y administrar estructuras de flujos de datos (pipelines) basadas en el kit de herramientas de registro y segmentación ITK, el kit de herramientas de visualización VTK y Qt; este sistema se ha implementado para permitir un rápido desarrollo de aplicaciones de herramientas de procesamiento de imágenes [11].

5.2.8 cpPlugins 2.0

cpPlugins 2.0 es una aplicación inspirada en cpPlugins 1.0 que propone un diseño para una arquitectura extensible destinada a construir tuberías de procesamiento de datos de una manera interactiva. Surge para mejorar aspectos relacionados con la interacción del usuario con el sistema, la flexibilidad del sistema al momento de visualizar distintos tipos de datos y la falta de sincronización de la tubería que presenta la primera versión de cpPlugins [12].

5.3 Análisis Crítico

A partir de los trabajos descritos anteriormente, se hace un análisis en donde se quiere determinar si el trabajo brinda los siguientes criterios:

- **Interfaz interactiva:** El software provee acceso a su funcionalidad por medio de una constante interacción con el usuario y retroalimentación de acciones.
- **Programación visual:** El software permite estructurar programas de manera gráfica.
- **Multiplataforma:** El software puede ser usado en Windows, Linux y MacOS.
- **Extensibilidad:** El software permite la integración de nuevos complementos o funcionalidades sin necesidad de volver a compilarlo.
- **Procesamiento rápido:** El software provee mecanismos para optimizar el tiempo de procesamiento de datos y mecanismos realizar el procesamiento en hardware distribuido.
- **Compatibilidad de archivos:** El software permite obtener datos de diferentes formatos de archivos.
- **Procesamiento de datos médicos:** El software provee mecanismos para procesar datos médicos estructurados.
- **Fácil integración:** El software permite se puede integrar con diferentes sistemas del contexto del problema que aborda sin necesidad de desarrollar un nuevo componente.

Para el análisis de los trabajos se utilizaron los siguientes rangos:

Calificación	Descripción
+	Soporta
-	No soporta
+/-	Soporta parcialmente

Tabla 1 - Categorías de calificación

De acuerdo con los criterios expuestos, se obtienen los siguientes resultados:

Criterios								
	ITKBoard	VTKDesigner	MeVisLab	SimITK	ParaView Web	MITK	cpPlugins 1.0	cpPlugins 2.0
Interfaz Interactiva	+	+	+	+	+	+	-	+/-
Programación visual	+	+	+	+	-	-	-	+/-
Multiplataforma	?	-	+	+/-	+	+	+	+
Extensibilidad	+	+	+	+	+	+	+	+
Procesamiento rápido	?	?	+/-	?	+	+/-	?	?
Programación dirigida por eventos	?	?	?	?	-	+	-	+
Compatibilidad de archivos	+	+	+	+	?	+	+	+
Código abierto	-	+	-	-	+	+	+	?
Procesamiento datos médicos	+	-	+	+	+	+	+	+

Fácil integración	?	?	+	-	?	+	?	?

Tabla 2 - Análisis crítico de trabajos relacionados

En cuanto al primer criterio, interfaz interactiva, ITKBoard, VTK Designer, MeVisLab y SimITK, cuentan con una interfaz interactiva debido a que como se evidencia en su descripción, requieren de la interacción con el usuario para estructurar un flujo de datos y acceder a diferentes funcionalidades que posean. En cuanto a ParaView Web, este provee una interfaz interactiva debido a que permite que el usuario interactúe de forma visual y en tiempo real con los datos que fueron procesados [46]. Por otro lado, MITK provee una interfaz interactiva, ya da soporte a interacciones 3D de la información, y múltiples vistas de los mismos datos [47]. Respecto a cpPlugins 1.0, no provee una interfaz interactiva ya que todo se realiza a través de comandos en una terminal, limitando la interacción a simplemente digitar el comando sin ayuda gráfica [48]. CpPlugins 2.0 soporta parcialmente este criterio, en esta versión se implementó un sistema gráfico que permite construir un flujo de datos. Sin embargo, los desarrolladores expresaron en la documentación de la aplicación que, en trabajos futuros, se debe tener en cuenta una serie de requisitos para mejorar la interacción del usuario y la interfaz gráfica [12].

Para la programación visual, esta es soportada por ITKBoard, VTK Designer, MeVisLab y SimITK, debido a que, como se refleja en su descripción, la programación visual es indispensable para dar solución al problema que plantean. ParaView Web y MITK no soportan la programación visual, ya que no es necesaria en sus respectivos contextos. CpPlugins 1.0 no soporta este criterio, dado que para acceder a las funcionalidades de la aplicación es necesario digitar comandos en una terminal [48]. No provee un sistema gráfico para la construcción de un flujo de datos. En cuanto a CpPlugins 2.0, este criterio lo soporta parcialmente. Pues permite construir pipelines arrastrando módulos, pero algunos aspectos de interacción con los datos no se pueden modificar de manera gráfica [12].

La mayoría de los trabajos relacionados pueden ser usados en las plataformas descritas en los criterios, a excepción de VTK Designer que solo se encuentra compilado para Windows [49], y SimITK que lo soporta parcialmente debido a que se despliega en MATLAB, como ya se había mencionado. CpPlugins 1.0 y CpPlugins 2.0 soportan este criterio, dada la compilación cruzada con CMake y la compatibilidad de librerías de C++ [12].

Sobre la extensibilidad, ITKBoard ha sido diseñado de tal manera que permita la incorporación de nuevos módulos en su solución. Los módulos en ITKBoard son considerados como filtros nuevos de ITK que el usuario desee incorporar para realizar tareas más específicas de procesamiento de imágenes médicas [10]. De igual forma funciona para todos los trabajos con diferencias de acuerdo con su contexto. Por ejemplo, MeVisLab pretende brindar mayor facilidad a la hora de integrar diferentes módulos [50]. VTK Designer permite la integración de componentes de VTK [49]. ParaView Web da a los desarrolladores la posibilidad de agregar cualquier componente que satisfaga necesidades específicas [46]. CpPlugins 1.0 está escrito en C++ y está basado en un sistema de extensiones, las extensiones son adiciones para un programa que agrega funcionalidades nuevas a este. Permite generar extensiones para cargar librerías de procesamiento y visualización [48]. En cuanto a cpPlugins 2.0, se ha diseñado como una arquitectura modular basada en extensiones. Esta nueva versión de la aplicación diseñó e implementó un gestor de extensiones con el objetivo de permitir la inclusión de nuevos componentes a manera de plug-ins y enriquecer la funcionalidad de extensiones de la versión anterior [12].

De acuerdo con el procesamiento rápido, este solo es soportado por ParaView Web, debido a que como se menciona en su descripción, está diseñado para procesar grandes volúmenes de datos vía web. MeVisLab soporta este criterio parcialmente debido a que posee estrategias de paginación y multihilo que permiten la ejecución de estas tareas en un menor tiempo [50]. MITK también soporta parcialmente el criterio, debido a que se apoya en estructuras de datos para organizar los datos y poder procesarlos de manera óptima [47].

Solo se encontró que MITK y cpPlugins 2.0 dan soporte a la programación dirigida por eventos. Para el caso de MITK, todos los elementos en el software poseen estados que están respondiendo a las interacciones del usuario por

medio de dispositivos de entrada y se poseen máquinas de estados que se encargan de distribuir estos eventos en toda la aplicación [47]. Por otro lado, cpPlugins 2.0 utiliza conceptos fundamentales de programación reactiva. Esto hace posible modelar eventos que tienen ocurrencias en puntos discretos en el tiempo donde el sistema puede cambiar en respuesta a eventos [12].

Todos los trabajos expuestos, a excepción de ParaView Web proveen compatibilidad con diferentes formatos debido a que utilizan VTK y/o ITK. Por lo tanto, para el caso de ITK, esta librería da compatibilidad a algunos formatos relevantes como: NIFTI, BMP, DICOM, JPEG, TIFF, VTK, entre otros [51].

En cuanto al código abierto, los trabajos que lo soportan son proyectos con una licencia "open source" y los que no son proyectos con licencias "propietarias".

Todos los trabajos soportan el procesamiento de datos médicos a excepción de VTK Designer. En la descripción de cada trabajo se detalla el problema al que le dan solución. Para el caso particular de ParaView Web, este software permite el análisis de datos en general, en donde los expertos le sacan mayor provecho a un tipo de datos en particular [46].

Finalmente, la fácil integración solo es soportada o solo se posee información de MeVisLab y MITK. MeVisLab posee fácil integración debido a que provee los mecanismos para integrarse con herramientas comunes de radiólogos, tales como servidores PACS, en donde se almacenan las imágenes médicas como radiologías [47]. En cuanto a MITK, posee un servidor para conectar sistemas de seguimiento, la implementación de un concepto que facilita la combinación estructurada de módulos y el código de integración para un sistema PACS [52].

6 Referencias

[1] B. A. Myers, «A brief history of human-computer interaction technology», *interactions*, vol. 5, n.º 2, pp. 44-54, mar. 1998.

[2] B. Fröhler, T. Möller, y C. Heinzl, «GEMSe: Visualization-Guided Exploration of Multi-channel Segmentation Algorithms», *Comput. Graph. Forum*, vol. 35, n.º 3, pp. 191-200, jun. 2016.

[3] J. Rexilius, «A Framework for Algorithm Evaluation and Clinical Application Prototyping using ITK», p. 9.

[4] D. Maleike, M. Nolden, H.-P. Meinzer, y I. Wolf, «Interactive segmentation framework of the Medical Imaging Interaction Toolkit», *Comput. Methods Programs Biomed.*, vol. 96, pp. 72-83, ene. 2009.

[5] H. Dong, L. Xia, J. Zhang, y A. Cai, «Medical Image Reconstruction Based on ITK and VTK», *2013 Int. Conf. Comput. Sci. Appl.*, p. 642, ene. 2013.

[6] «A software development of DICOM image processing based on QT, VTK and ITK», *2013 IEEE Int. Conf. Med. Imaging Phys. Eng. Med. Imaging Phys. Eng. ICMIPE 2013 IEEE Int. Conf. On*, p. 231, 2013.

[7] A. W. L. Dickinson, P. Abolmaesumi, D. G. Gobbi, y P. Mousavi, «SimITK: Visual Programming of the ITK Image-Processing Library within Simulink», *J. Digit. Imaging*, vol. 27, n.º 2, pp. 220-230, abr. 2014.

[8] «ITK - Segmentation & Registration Toolkit». [En línea]. Disponible en: https://itk.org/. [Accedido: 10-feb-2019].

[9] «VTK - The Visualization Toolkit». .

[10] H. D. K. Le, R. Li, y S. Ourselin, «Towards a Visual Programming Environment Based on ITK for Medical Image Analysis», en *Digital Image Computing: Techniques and Applications (DICTA'05)*, 2005, pp. 80-80.

[11] «cpPlugins - CREATIS». [En línea]. Disponible en: https://www.creatis.insa-lyon.fr/cpPlugins/. [Accedido: 10-feb-2019].

[12] C. D. GUERRERO REGINO, R. A. MONTAÑO BARRERA, y M. J. VARGAS MARQUEZ, «Propuesta para Trabajo de Grado - CPPlugins 2.0», p. 25.

[13] «PipelineProcessing Design Pattern». [En línea]. Disponible en: https://www.cise.ufl.edu/research/ParallelPatterns/PatternLanguage/AlgorithmStructure/Pipeline.htm. [Accedido: 09-feb-2019].

[14] «Definition of PIPELINE». [En línea]. Disponible en: https://www.merriam-webster.com/dictionary/pipeline. [Accedido: 18-abr-2019].

[15] X. Fan, O. Sinnen, y N. Giacaman, «Towards an Event-Driven Programming Model for OpenMP», en *2016 45th International Conference on Parallel Processing Workshops (ICPPW)*, 2016, pp. 240-249.

[16] T. Sousa, «Dataflow Programming: Concept, Languages and Applications», 2012.

[17] D. Mason y K. Dave, «Block-based versus flow-based programming for naive programmers», en *2017 IEEE Blocks and Beyond Workshop (B B)*, 2017, pp. 25-28.

[18] «IEEE 16326-2009 - ISO/IEC/IEEE International Standard - Systems and Software Engineering--Life Cycle Processes--Project Management». [En línea]. Disponible en: https://standards.ieee.org/standard/16326-2009.html. [Accedido: 01-abr-2019].

[19] 14:00-17:00, «ISO/IEC/IEEE 12207:2017», *ISO*. [En línea]. Disponible en: http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/06/37/63712.html. [Accedido: 18-abr-2019].

[20] «ISO/IEC 16085:2006, Standard for Software Engineering - Software Life Cycle Processes - Risk Management», *Std ISO IEC 16085 - 2006*, pp. 1-46, dic. 2006.

[21] 14:00-17:00, «ISO/IEC/IEEE 24748-4:2016», *ISO*. [En línea]. Disponible en: http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/05/68/56887.html. [Accedido: 18-abr-2019].

[22] 14:00-17:00, «ISO/IEC/IEEE 29148:2018», *ISO*. [En línea]. Disponible en: http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/07/20/72089.html. [Accedido: 18-ab

[23] «Current Edition – DICOM Standard». [En línea]. Disponible en: https://www.dicomstandard.org/current/. [Accedido: 18-abr-2019].
[24] 14:00-17:00, «ISO/IEC 25000:2014», *ISO*. [En línea]. Disponible en: http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/06/47/64764.html. [Accedido: 18-abr-2019].
[25] «IEEE Standard for Information Technology–Systems Design–Software Design Descriptions», *IEEE STD 1016-2009*, pp. 1-35, jul. 2009.
[26] «About the Unified Modeling Language Specification Version 2.5». [En línea]. Disponible en: https://www.omg.org/spec/UML/2.5. [Accedido: 18-abr-2019].
[27] «ISO/IEC/IEEE International Standard - Software and systems engineering –Software testing –Part 1:Concepts and definitions», *ISOIECIEEE 29119-12013E*, pp. 1-64, sep. 2013.
[28] «ISO/IEC/IEEE International Standard - Software and systems engineering –Software testing –Part 2:Test processes», *ISOIECIEEE 29119-22013E*, pp. 1-68, sep. 2013.
[29] «ISO/IEC/IEEE International Standard - Software and systems engineering–Software testing–Part 4: Test techniques», *ISOIECIEEE 29119-42015*, pp. 1-149, dic. 2015.
[30] «ISO/IEC/IEEE International Standard - Software and systems engineering – Software testing –Part 3: Test documentation», *ISOIECIEEE 29119-32013E*, pp. 1-138, sep. 2013.
[31] «Code Conventions for the Java Programming Language: Contents». [En línea]. Disponible en: https://www.oracle.com/technetwork/java/javase/documentation/codeconvtoc-136057.html. [Accedido: 15-abr-2019].
[32] «JavaScript Standard Style». [En línea]. Disponible en: https://standardjs.com/. [Accedido: 15-abr-2019].
[33] «React Standard Style code snippets - Visual Studio Marketplace». [En línea]. Disponible en: https://marketplace.visualstudio.com/items?itemName=TimonVS.ReactSnippetsStandard. [Accedido: 15-abr-2019].
[34] Atlassian, «Git Workflow | Atlassian Git Tutorial», *Atlassian*. [En línea]. Disponible en: https://www.atlassian.com/git/tutorials/comparing-workflows. [Accedido: 18-abr-2019].
[35] T. Preston-Werner, «Semantic Versioning 2.0.0», *Semantic Versioning*. [En línea]. Disponible en: https://semver.org/. [Accedido: 15-abr-2019].
[36] «IEEE Standard for Software User Documentation», *IEEE Std 1063-2001*, pp. 1-24, dic. 2001.
[37] «Qué es SCRUM», *Proyectos Ágiles*, 04-ago-2008. .
[38] «Kanban: por qué es ágil y en qué aventaja a SCRUM - ITM Platform», *ITM Platform | Projects, Programs & Portfolio*, 20-sep-2017. .
[39] «CeCILL». [En línea]. Disponible en: http://www.cecill.info/. [Accedido: 17-abr-2019].
[40] «Creative Commons — Attribution 4.0 International — CC BY 4.0». [En línea]. Disponible en: https://creativecommons.org/licenses/by/4.0/. [Accedido: 17-abr-2019].
[41] «e-REdING. Biblioteca de la Escuela Superior de Ingenieros de Sevilla.» [En línea]. Disponible en: http://bibing.us.es/proyectos/abreproy/11854/direccion/Volumen+1%252F. [Accedido: 18-abr-2019].
[42] A. Buchanan, *Cycles of child maltreatment: facts, fallacies, and interventions*. Chichester ; New York: Wiley, 1996.
[43] «Arquitectura Modular | Tag | ArchDaily Colombia». [En línea]. Disponible en: https://www.archdaily.co/co/tag/arquitectura-modular. [Accedido: 18-abr-2019].
[44] «Plug-in Architectures». [En línea]. Disponible en: https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/LoadingCode/Concepts/Plugins.html. [Accedido: 09-feb-2019].
[45] «MATLAB - MathWorks». [En línea]. Disponible en: https://www.mathworks.com/products/matlab.html. [Accedido: 18-abr-2019].

[46] S. Jourdain, U. Ayachit, y B. Geveci, «ParaViewWeb: A Web Framework for 3D Visualization and Data Processing», p. 8.

[47] I. Wolf *et al.*, *The Medical Imaging Interaction Toolkit (MITK) – a toolkit facilitating the creation of interactive software by extending VTK and ITK*. .

[48] «Acta de Reunion.docx». [En línea]. Disponible en: https://livejaverianaedu.sharepoint.com/:w:/r/sites/TrabajodeGrado-cpPlugins3.0/_layouts/15/Doc.aspx?sourcedoc=%7BF97A43FA-3924-4D3A-9DE3-8D533DE0D6EB%7D&file=Acta%20de%20Reunion.docx&action=default&mobileredirect=true. [Accedido: 18-abr-2019].

[49] «VTK Designer», *SourceForge*. [En línea]. Disponible en: https://sourceforge.net/projects/vtkdesigner/. [Accedido: 09-feb-2019].

[50] «MeVisLab: MeVisLab». [En línea]. Disponible en: https://www.mevislab.de/. [Accedido: 26-mar-2019].

[51] «ITK/File Formats - KitwarePublic». [En línea]. Disponible en: https://itk.org/Wiki/ITK/File_Formats. [Accedido: 18-abr-2019].

[52] I. Wolf *et al.*, «The medical imaging interaction toolkit (MITK): a toolkit facilitating the creation of interactive software by extending VTK and ITK», presentado en Medical Imaging 2004, San Diego, CA, 2004, p. 16.