

Priorización (paso 1)

Paso 1:

RNF-1: El sistema implementa el 100 % de los casos de uso identificados.

CU-1: Entrenar Modelo.

RF-1: Cargar Imágenes.

- **C-5:** Las imágenes de entrada y los resultados de las segmentaciones generadas por el sistema deben poder ser visualizadas en cualquier software que se especialice en su visualización y/o manipulación (formato NIfTI).

• **RF-4:** Entrenar Modelo.

- **C-4:** El entrenamiento se hace por medio de técnicas de aprendizaje profundo (CNN).
- **RF-5:** Evaluar Modelo

• **RF-7:** Guardar Modelo.

- **C-6:** El sistema solo puede generar y cargar modelos de formato “Hierarchical Data Format” en su quinta versión (.h5).

• **RNF-4:** El sistema debe optimizar el uso de recursos como la memoria RAM y la memoria RAM dedicada de la GPU (VRAM) para que el entrenamiento de modelos se realice de forma satisfactoria.

- **RF-3:** Preprocesar Imágenes.

CU-2: Segmentar Imágenes.

• **RF-1:** Cargar Imágenes.

- **C-5:** Las imágenes de entrada y los resultados de las segmentaciones generadas por el sistema deben poder ser visualizadas en cualquier software que se especialice en su visualización y/o manipulación (formato NIfTI).

• **RF-2:** Cargar Modelo.

- **C-6:** El sistema solo puede generar y cargar modelos de formato “Hierarchical Data Format” en su quinta versión (.h5).

• **RF-6:** Segmentar Imágenes.

• **RF-8:** Guardar Segmentaciones.

- **C-5:** Las imágenes de entrada y los resultados de las segmentaciones generadas por el sistema deben poder ser visualizadas en cualquier software que se especialice en su visualización y/o manipulación (formato NIfTI).

RF-9: Acciones por Comandos.

Nota: Los requerimientos no funcionales **RNF-2** y **RNF-3** fueron excluidos de esta priorización, esto dado que si bien son factores determinantes de la calidad del sistema, su naturaleza es de carácter experimental, por ello las estrategias para satisfacerlos se abordan en el plan de pruebas del sistema y en la ejecución del mismo.

Nota 2: El requerimiento no funcional **RNF-5** no está listado en la priorización, ya que pertenece a la categoría de atributos de calidad: seguridad y a la subcategoría: integridad. Teniendo en cuenta su descripción y las características en común de los elementos priorizados (Datos), se toma como un requerimiento asociado al sistema en todos sus niveles de abstracción y jerarquización.

Iteración I

Paso 2 y 3: Se va a refinar el sistema
El elemento del sistema elegido es el mismo sistema.
Drivers: RNF-1

Paso 4: Escoger conceptos de diseño que satisfagan el o los drivers seleccionados.

- Patrones arquitectónicos
 - Pipes and Filters
 - Repository
 - Black Board

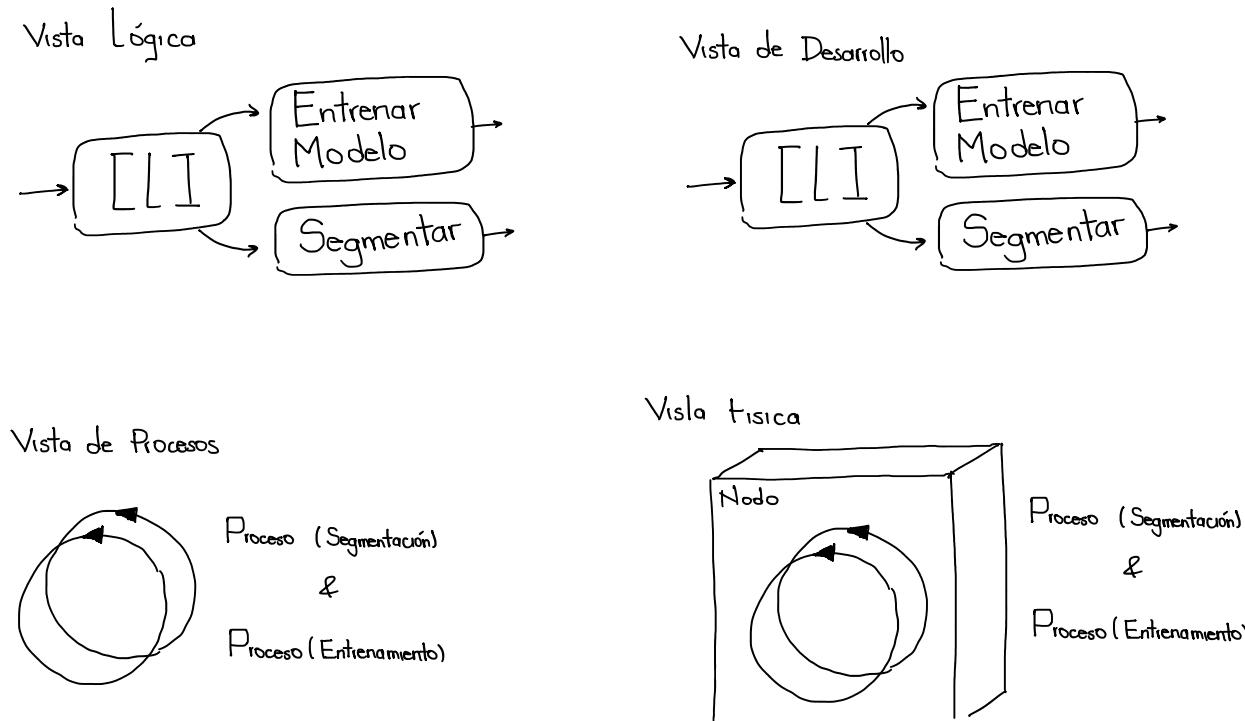
Por el método de priorización realizado se identificó que los componentes que están contenidos dentro del driver arquitectónico son los casos de uso del sistema.

Después de analizar los casos de uso del sistema, se pudo determinar que la característica común de los casos de uso es el procesamiento de datos, dicho esto se Determinaron tres patrones arquitectónicos como posibles opciones para el diseño del sistema.

De estas tres opciones se escogió Pipes and Filters, esta arquitectura se enfoca en el flujo de datos y su implementación es la más intuitiva de los tres patrones planteados.

Por otro lado tanto Repository como Black Board son patrones arquitectónicos más propensos a generar errores sobre los datos.

Paso 5 y 6:



Iteración II

Paso 2 y 3: Se va a refinar el sistema

El elemento del sistema elegido es el mismo sistema.

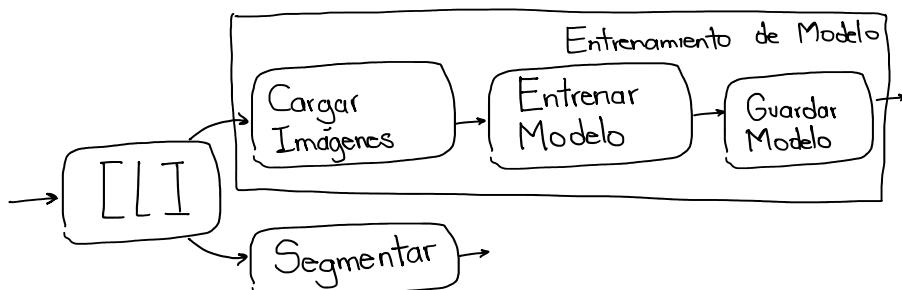
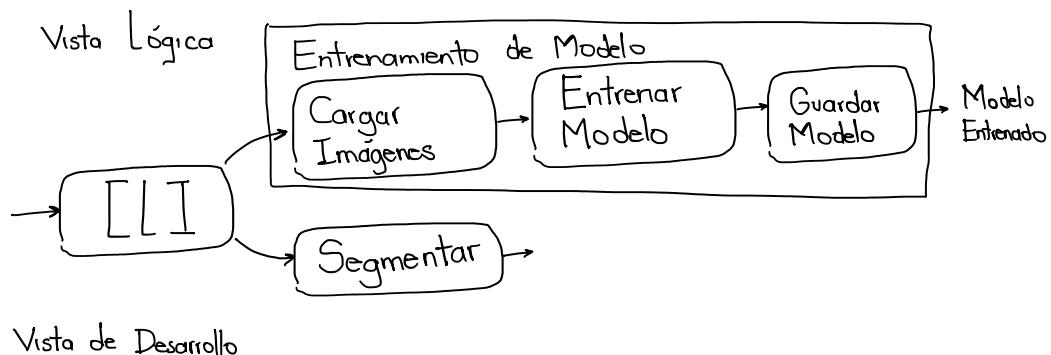
Drivers: CU-1

Paso 4: Escoger conceptos de diseño que satisfagan el o los drivers seleccionados.

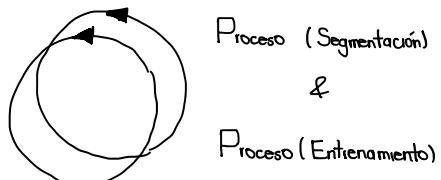
- Descomposición

El caso de uso “entrenar modelo” requiere de ciertos pasos para realizar la tarea objetivo, dichos pasos en primera instancia son los siguientes:

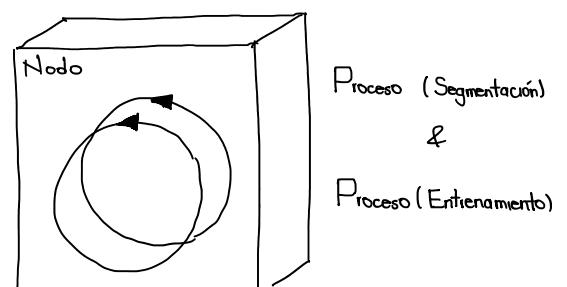
- Cargar imágenes
- Entrenar modelo
- Guardar modelo



Vista de Procesos



Vista Física



Iteración III

Paso 2 y 3: Se va a refinar el sistema

El elemento del sistema elegido es el mismo sistema.

Drivers: C-4.

Paso 4: Escoger conceptos de diseño que satisfagan el o los drivers seleccionados.

- Arquitecturas de redes neuronales
 - Unet
 - CNN
 - CNN + SVM
- Descomposición

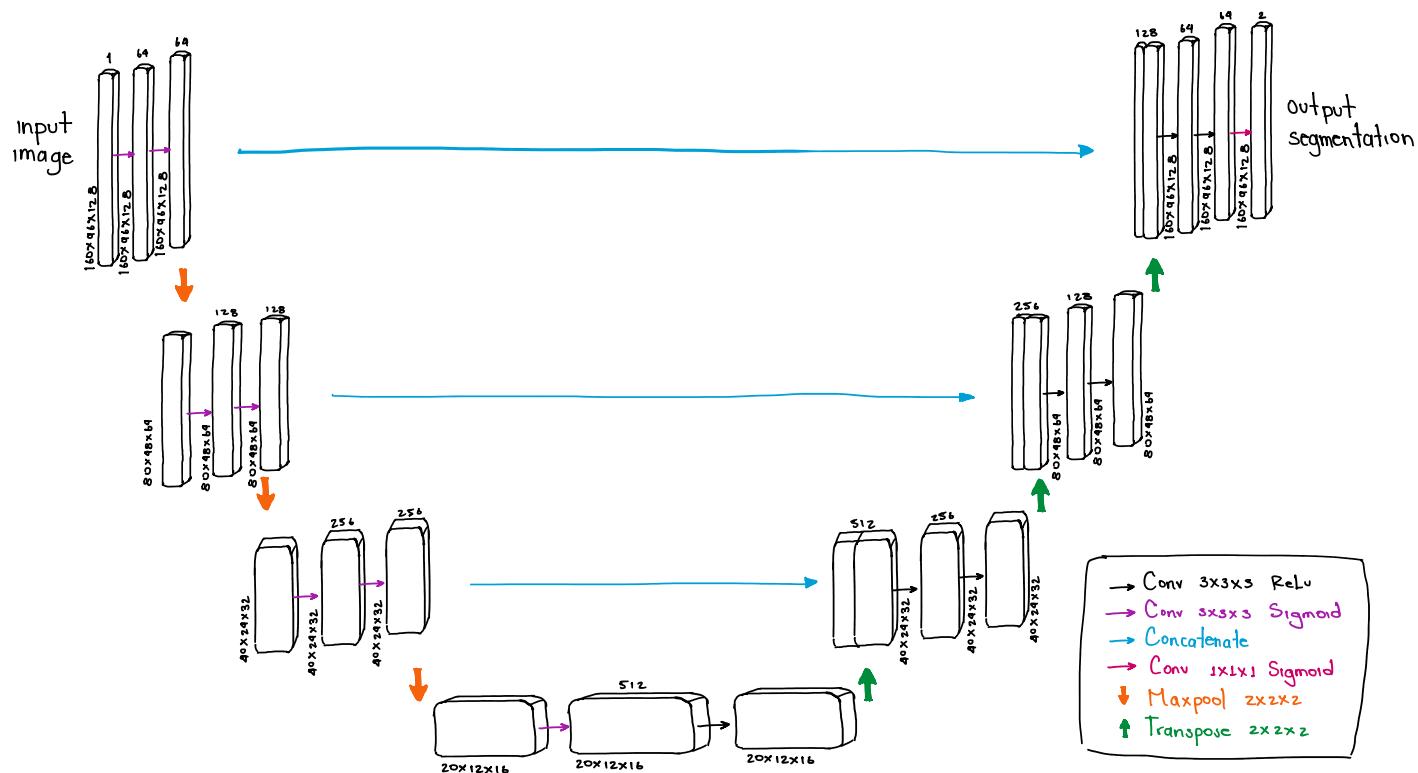
Dada la investigación acerca de las arquitecturas más utilizadas para realizar segmentación de imágenes 3d o segmentación semántica, se encontró que la arquitectura con mejores resultados y más estudiada es la arquitectura Unet.

Por motivos de tiempos de desarrollo y agregado a la arquitectura seleccionada, el módulo de “entrenar modelo” se ha descompuesto en dos componentes “entrenar” y “Unet 3d”.

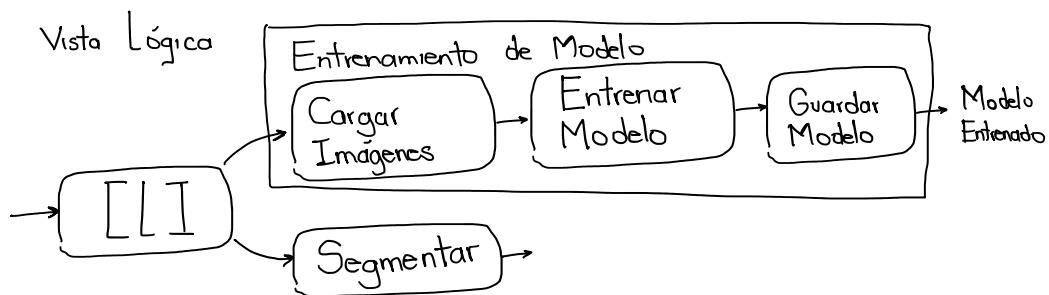
Adicionalmente, se seleccionó el uso de la tecnología “TensorFlow” para reducir la complejidad de la implementación de la arquitectura de la red neuronal. Asociado a esto también se decidió delegar a la tecnología “Keras” la implementación del módulo de guardar modelo.

Arquitectura de Red Neuronal (Unet 3d)

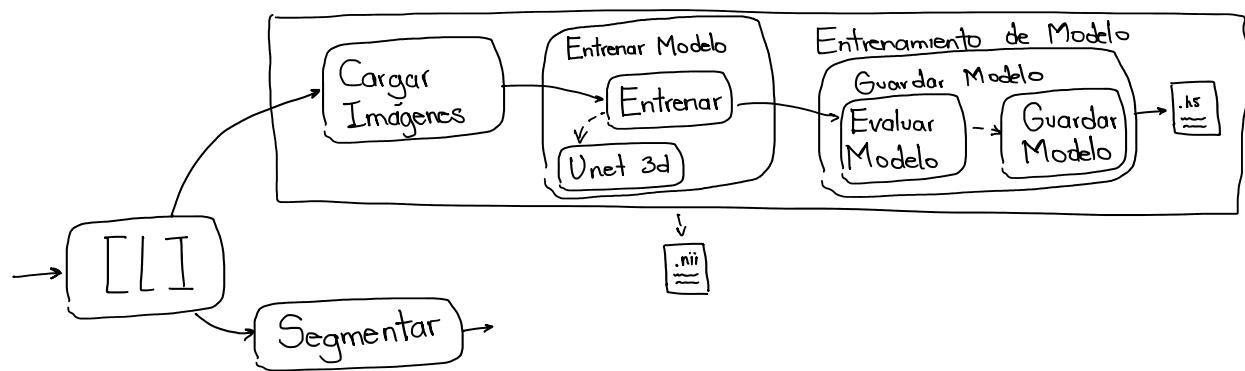
Vista de Arquitectura de Red Neuronal



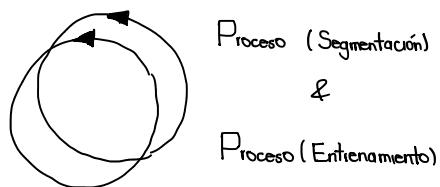
Iteración III (vistas)



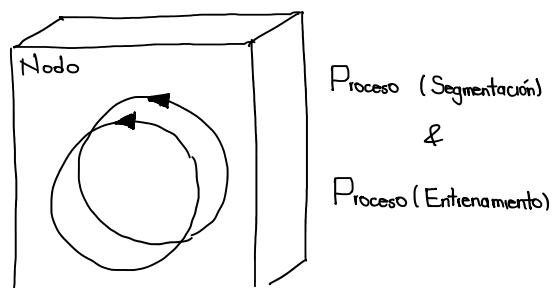
Vista de Desarrollo



Vista de Procesos



Vista física



Iteración IV

Paso 2 y 3: Se va a refinar el sistema

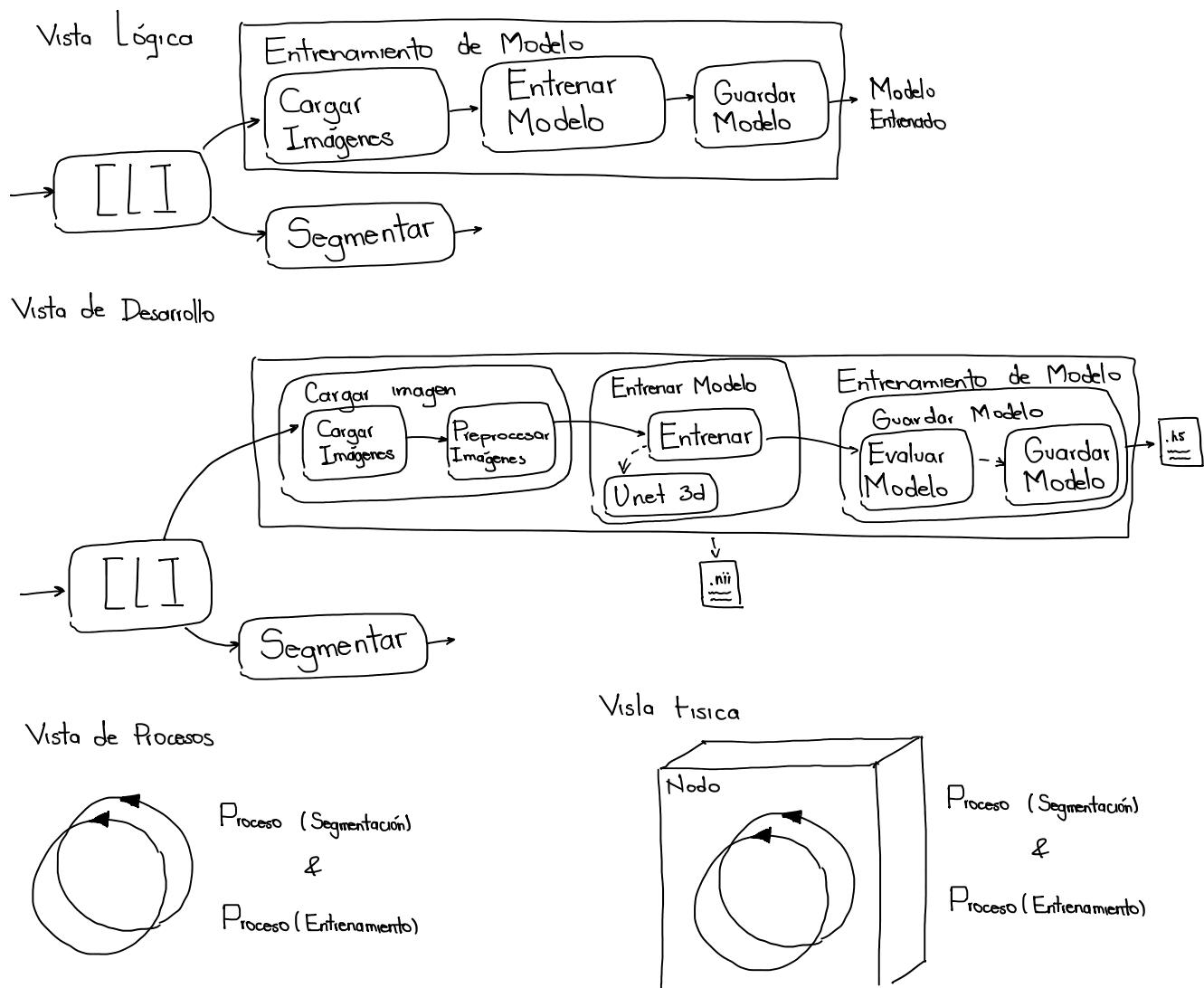
El elemento del sistema elegido es el mismo sistema.

Drivers: RNF-4.

Paso 4: Escoger conceptos de diseño que satisfagan el o los drivers seleccionados.

- Descomposición
- Optimización de los recursos

Dado que la ejecución de la arquitectura carga completamente la imágenes y la procesa en la red neuronal es una tarea que exige muchos recursos y el acceso a una máquina con los suficientes recursos para correr la arquitectura propuesta es difícil, se optó por realizar un preprocesamiento a las imágenes con el objetivo de reducir la carga que las imágenes y la red suponen al hardware sobre el que se ejecutan. Para esto este componente recorta las imágenes en el área de interés



Iteración V

Paso 2 y 3: Se va a refinar el sistema

El elemento del sistema elegido es el mismo sistema.

Drivers: CU-2, RF-1, RF-2, RF-6, RF-8.

Paso 4: Escoger conceptos de diseño que satisfagan el o los drivers seleccionados.

- Descomposición
- Restringir sistema

El caso de uso "segmentar imágenes" requiere de ciertos pasos para realizar la tarea objetivo, dichos pasos en primera instancia son los siguientes:

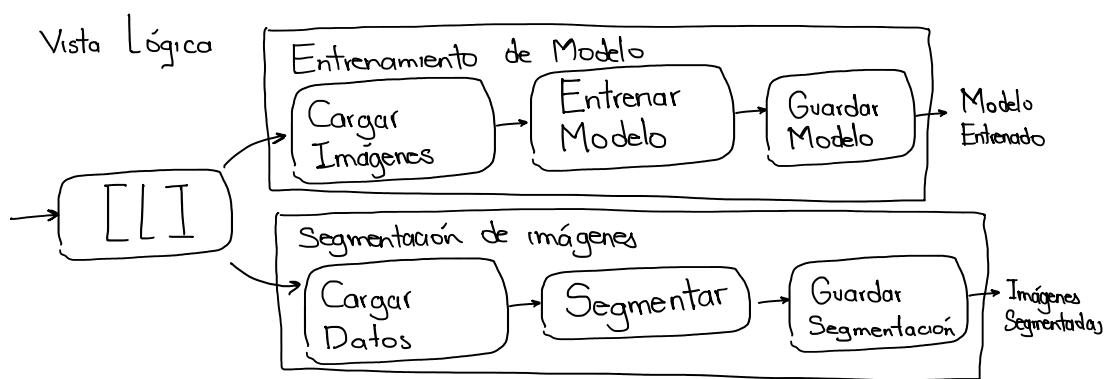
- Cargar datos
- Predecir segmentación
- Guardar segmentación

Fueron satisfechos las restricciones asociadas respecto a las funcionalidades de "cargar datos", "predecir segmentación" y "reconstrucción". Dichas restricciones consisten en el formato de los datos que pueden ser cargados por el subsistema y que pueden ser generados por el mismo.

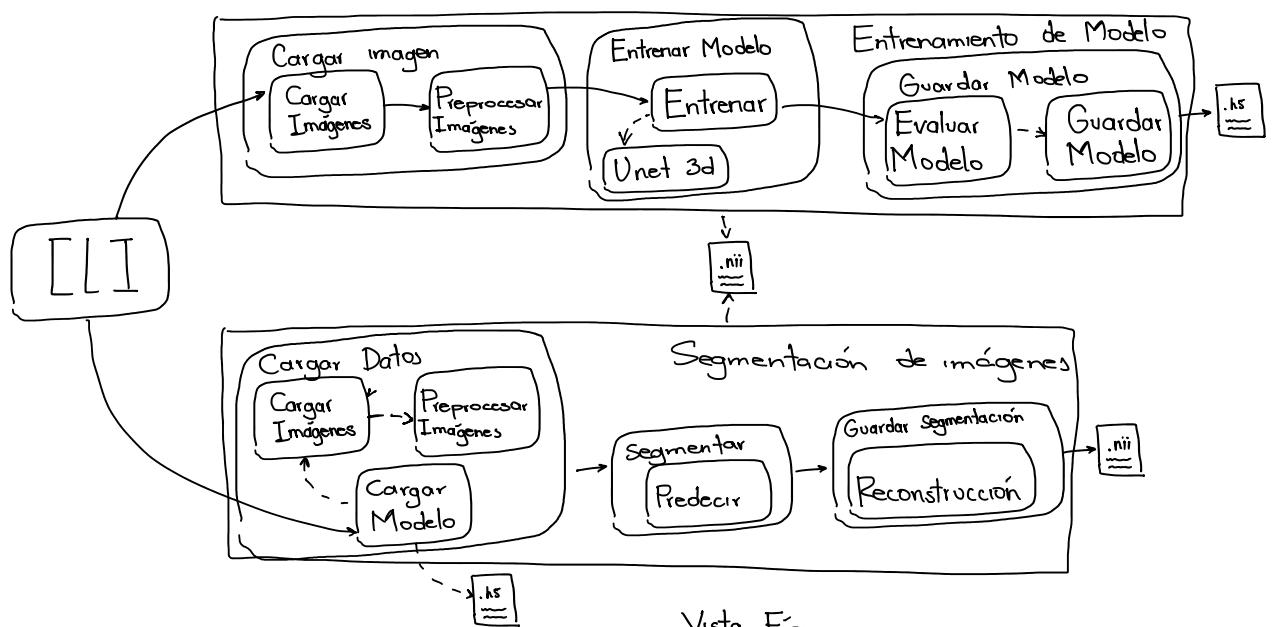
Una vez identificados los diferentes módulos que componen el subsistema también se identificaron y descompusieron los diferentes componentes de dichos módulos. Dichos componentes son: predecir, cargar imágenes, cargar modelo y guardar segmentación.

Finalmente, dado que el modelo se entrena con base en imágenes preprocesadas, el uso del modelo sin estas imágenes puede dar un resultado insatisfactorio en las predicciones de las segmentaciones. Por ello, es necesario preprocesar las imágenes previamente a la predicción y reconstruirlas después de esta, para posteriormente ser guardadas.

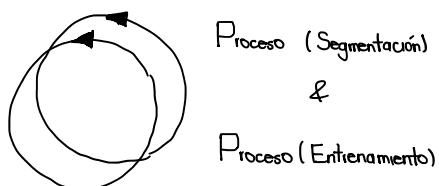
Iteración V (vistas)



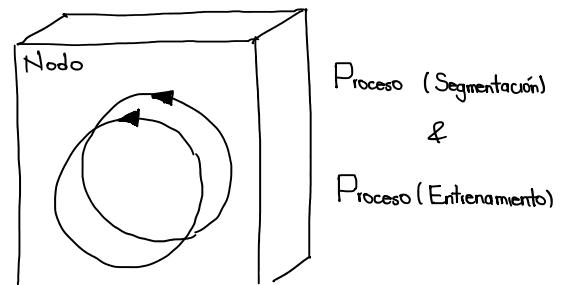
Vista de Desarrollo



Vista de Procesos



Vista Física



Iteración VI

Paso 2 y 3:

El elemento del sistema elegido es el mismo sistema.

Drivers: No se ha seleccionado un driver en específico, ya que esta iteración se propone refinar el sistema generado de las iteraciones previas.

Paso 4: Escoger conceptos de diseño que satisfagan el o los drivers seleccionados.

- Descomposición
- Mapeo de dependencias
- Mapeo de invocaciones
- Composición

El sistema planteado en la iteración previa carecía de detalle, por ello en esta iteración se procedió a refinar la arquitectura propuesta, inicialmente se hizo descomposición de las funcionalidades cuyas responsabilidades no eran claras respecto al funcionamiento que debían realizar, resultado de esto origino que el componente “Preprocesar Imágenes” se descompusiera en “Preparar Datos”, “Recortar” y “Dividir Datos” en el caso del subsistema de entrenamiento, para el caso del subsistema de segmentación se descompuso el componente de “Preprocesar Imágenes”, la descomposición de este resultó en los siguientes componentes: “Preparar Datos” y “Recortar”, una vez realizada esta descomposición fue evidente el notar que para ambos subsistemas tenían funcionalidades similares, sin embargo, el detalle de dichas funcionalidades para ambos subsistemas no era el mismo para todas las similitudes, el único componente que su funcionalidad era exactamente la misma para cada subsistema es la que desempeña el componente “Recortar” agregado a esto se detectó que los componentes que hace uso de esta funcionalidad dependen de los resultados de este, por ello se decidió aislar el componente en su propio módulo, con el fin de que el funcionamiento y el preprocesamiento de los datos sea consistente entre ambos subsistemas.

Se determinó que la funcionalidad del componente “Guardar Segmentación” puede estar contenida en la funcionalidad de reconstruir, por ello se decidió componer un único componente el cual lleva el nombre de “Reconstrucción” el cual se encarga de la funcionalidad de reconstruir la imagen para que se acople a la resonancia magnética y de guardar esta misma.

Después de realizar las composiciones y descomposiciones pertinentes, se procedió a mapear todas las dependencias e invocaciones del sistema, esto con el fin de determinar que componentes y/o módulos interactúan unos con otros, cabe aclarar que con dependencia nos referimos a cuando un componente invoca una funcionalidad de otro y espera una respuesta de este para proceder con su propio funcionamiento, por el contrario, con invocación nos referimos a cuando un componente realiza la invocación de una funcionalidad de otro, componte, pero no espera una respuesta para proceder con su propio funcionamiento, si no, por el contrario, le delega la responsabilidad del ciclo de vida del proceso global del sistema y/o subsistema en ejecución.

Como se determinó en iteraciones previas la implementación del módulo de guardar modelo no será realizado por el equipo, esto se debe a que tecnologías como TensorFlow, Keras, Sklearn ya poseen esas funcionalidades implementadas; sin embargo, se decide dejar expresado en la vista de desarrollo a motivo de evidenciar el flujo que podría realizar el sistema en ejecución. También cabe resaltar que los componentes que se denotan en la vista de desarrollo son archivos .py, el motivo de esto es gracias a su facilidad de uso y compatibilidad con esas tecnologías. En la vista de desarrollo se incluyen y modelan las relaciones entre los elementos del sistemas y las herramientas utilizadas.

Iteración VI (vistas)

