Facultad de Ingeniería
Escuela de Ingeniería Civil en Informática

OPTIMIZACIÓN DE CÓDIGO PARA EL PROTOTIPO DE MEDICIONES DE HUESO CORTICAL

Por

Claudio Antonio Araya Valenzuela

Trabajo realizado para optar al Título de
INGENIERO EN INFORMÁTICA
Prof. Guía: Jean-Gabriel Minonzio
Prof. Co-Referente: *
Enero 2019

Resumen

El desarrollo de alternativas mas seguras, portables y económicas a la densitometría ósea, es posible gracias a los avances en técnicas cuantitativas de ultrasonido. La transmisión axial permite medir la porosidad y espesor cortical usando ultrasonido. La optimización de código, permite mejorar el código existente, logrando que este se ejecute mas rápidamente.

Capítulo 1

Introducción

1.1. Principales contribuciones

Los huesos son de vital importancia para la salud y la calidad de vida en general[1],ya que proporcionan al cuerpo funciones estructurales y metabólicas. Las funciones estructurales de los huesos son dar soporte para realizar las acciones mecánicas y entregar protección mecánica a distintos órganos vitales. Metabólicamente, son encargados de producir células sanguíneas y ser la reserva de calcio mas grande del cuerpo humano [2].

Los huesos se componen principalmente de dos distintos tipos de tejidos, una capa exterior compacta compuesta de hueso cortical que rodea el tejido esponjoso de hueso trabecular [3].

Los huesos no saludables, sin embargo, tienen un desempeño deficiente en la ejecución de sus funciones, lo que puede tener consecuencias perjudiciales como las fracturas por fragilidad [1]. En Chile las fracturas de caderas en adultos mayores sus costes y mortalidad equivalen a la suma de costes y mortalidad por enfermedades cardiovasculares y neoplasias [4].

La densidad mineral osea es el marcador biológico mas usado para predecir el riesgo a fracturas, sin embargo características oseas relacionadas a la fuerza también incluyen propiedades del hueso cortical y trabecular. Hallazgos recientes sugieren que la evaluación de riesgo de fractura también debe incluir una evaluación precisa del hueso cortical [5].

Actualmente la empresa Azalée tiene un dispositivo de sonda multicanal para transmisión axial. La transmisión axial es una técnica cuantitativa de ultrasonido que permite cuantificar el espesor cortical y la porosidad del hueso cortical.

La interfaz actualmente implementada despliega el espectro de ondas guiadas en casi tiempo real (con un framerate de 4Hz), siendo el tiempo de respuesta no lo suficientemente menor para el uso requerido, por lo tanto, se propone una revisión de las etapas del algoritmo usado para el análisis de los datos para encontrar secciones que se puedan optimizar.

1.2. Estructura del Documento

El documento posee la siguiente estructura. A continuación, se presenta el Marco Conceptual y el Estado del Arte, en el siguiente capitulo Definición del Problema y Análisis.

Capítulo 2

Marco Conceptual y Estado del Arte

En este capítulo en la sección 2.1 deben presentar el Marco Conceptual, así como en la sección 2.2 es expuesto el estado del arte.

2.1. Marco Conceptual

2.1.1. Ámbito Oseo

Periostio: El periostio es una vaina dura de tejido conectivo denso e irregular que envuelve la superficie ósea en los lugares que no están cubierto por cartílago [6].

Tejido óseo cortical: El tejido oseo cortical contiene poco espacios y es el componente más solido del tejido óseo. Se encuentra debajo del periostio de todos los huesos y forma la mayor parte de las diáfisis de los huesos largos. [6]. En la Figura 2.1, se presenta un Húmero indicando el hueso cortical.

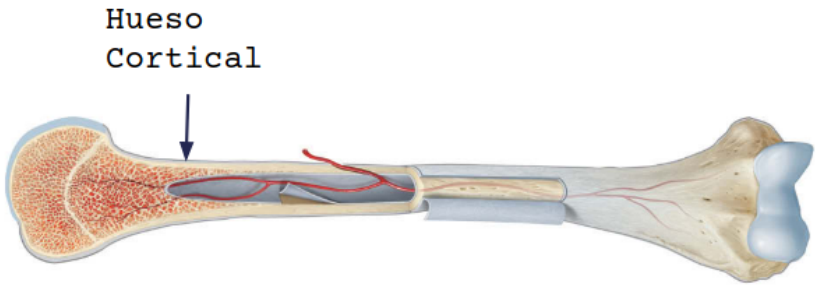

Figura 2.1: Húmero parcialmente seccionado.

Porosidad Cortical: La porosidad cortical consiste en una red de canales que proporcionan conductos para que la vasculatura impregne la corteza y, en última instancia, sostener a los osteocitos [3]. En la Figura 2.2, se muestra un detalle de hueso cortical indicando los canales de la porosidad cortical.

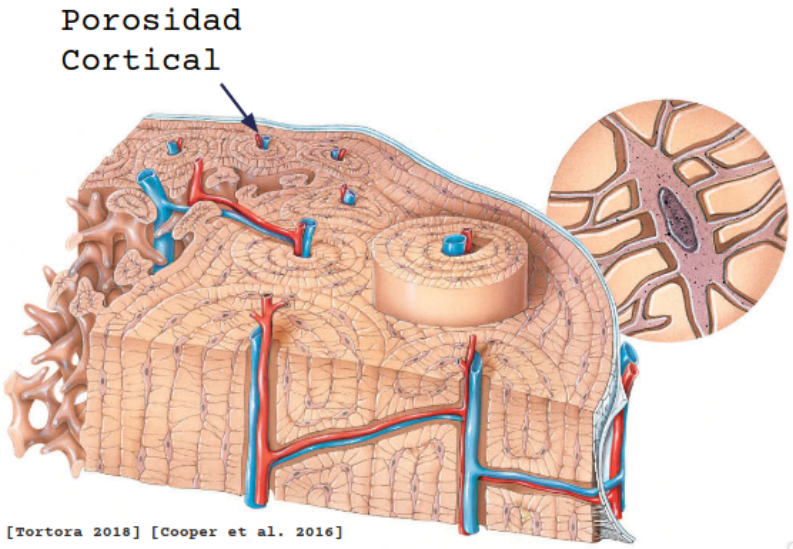

Figura 2.2: Vista ampliada de hueso cortical.

Osteocitos: Los osteocitos son células oseas maduras, que son las principales del hueso y mantienen su metabolismo diario a través del intercambio de nutrientes y productos metabólicos con la sangre [6].

Densidad Mineral Osea: La densidad mineral osea (DMO) es la cantidad de materia mineral en el tejido oseo. Las mediciones de densidad osea son usadas en la medicina clínica como un indicador indirecto de osteoporosis y riesgo de fractura [1]. En la Figura 2.3, se muestra una imagen de Densitometría ósea, donde el nivel de gris indica la cantidad de material mineral en el hueso.

[1] https://meshb.nlm.nih.gov/record/ui?ui=D015519

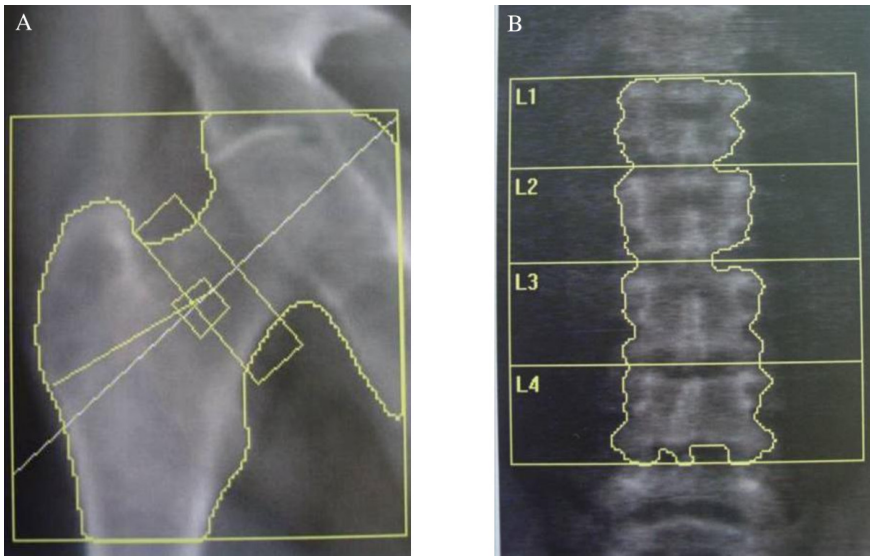

Figura 2.3: Imagen de Densitometría ósea.

2.1.2. Ámbito Ultrasonido

Ultrasonido: El ultrasonido es una onda mecánica con una frecuencia para uso clínico entre 1 y 15 MHz, estas frecuencias son mayores al limite superior audible por un humano (20 KHz). La longitud de onda del ultrasonido en los tejido es entre 0.1 y 1.5 mm [7] [8].

Guía de Onda: Es una estructura que guía ondas, como las electromagnéticas o mecánicas, con una pérdida mínima de energía al restringir la expansión a una o dos dimensiones.

Transductor de Ultrasonido: Son un tipo de sensor acústico, se pueden categorizar como Emisores que convierten señales eléctricas en ultrasonido y como Receptores que convierten el ultrasonido en señales eléctricas [9].

Placa Libre: Una placa es un elemento estructural con dimensiones planas (largo y ancho) son grandes en comparación con su grosor y está sujeto a cargas que causan deformación por flexión además del estiramiento. En la mayoría de los casos, el grosor no es mayor que una décima parte de la dimensión más pequeña en el plano. Por libre se asume que la placa no experimenta momentos de flexión y torsión y no hay fuerzas de corte verticales [10].

2.1.3. Ámbito Procesamiento de Señales

Procesamiento Digital de Señales: El procesamiento digital de señales es el uso del procesamiento digital, por ejemplo, en computadoras o procesadores de señales digitales más especializados, para realizar una amplia variedad de operaciones de procesamiento de la señal. Las señales procesadas de esta manera son una secuencia de números que representan muestras de una variable continua en un dominio como el tiempo, el espacio o la frecuencia.

Transformada de Fourier: La Transformada de Fourier toma una función definida en el dominio del tiempo o espacio y la transforma al dominio de la frecuencia, que provee un ambiente natural para el estudio de muchos problemas. Las técnicas del Análisis de Fourier son usadas en distintos y diversos campos como, por ejemplo el procesamiento de señales, astronomía, geodésica,las imágenes medicas y el análisis de voz [11]. La Figura 2.4, muestra una función y su transformada de Fourier.

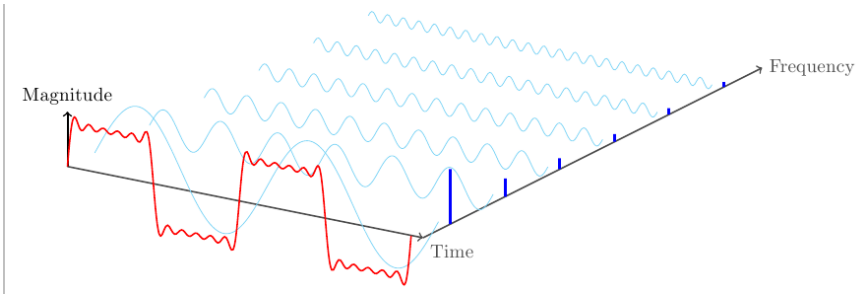

Figura 2.4: Imagen Transformada de Fourier.

Señal: Una señal es una función de uno o más variables independientes, que contienen información sobre el comportamiento o la naturaleza de algún fenómeno [12].

Descomposición en valores singulares: La descomposición de valores singulares (DVS) es una factorización de una matriz M de m filas y n columnas. Esta factorización tiene la forma $M = U\Sigma V*$, donde U es una matriz cuadrada de orden m, donde Σ es una matriz diagonal rectangular n x m con valores reales no-negativos en la diagonal y V es una matriz cuadrada de orden n. Los valores de la diagonal de Σ son los valores singulares de M. Las columnas de U y V son los vectores singulares izquierdos y vectores singulares derechos de M respectivamente. [13]

Transmisión Axial: La transmisión axial es una técnica de ultrasonido cuantitativa que permite medir el espesor y la porosidad del hueso cortical. Esta técnica toma en cuenta

la capacidad de los huesos largos en guiar onda en el rango de frecuencias (200KHz - 2MHz), que significa la coexistencia de varios modos guiados. Usando un sonda compuesta por un arreglo de emisores y receptores dispuestos linealmente, son grabados los modos guiados propagados por el hueso cortical. La curvas de dispersión se obtienen usando una transformada de Fourier bidimensional combinada con una descomposición de valores singulares. Los valores de espesor y porosidad son obtenidos mediante la resolución del problema inverso, donde las curvas de dispersión son predichas desde un modelo de placa libre isotrópica [14].

2.2. Estado del Arte

El estado de arte para el desarrollo especifico de esta tesis, sera separado en dos áreas, análisis de desempeño y alto rendimiento.

2.2.1. Análisis de Desempeño

Thinking Methodically about Performance En este articulo el autor describe dos antimétodos para el análisis de desempeño, luego señala tres alternativas a las descritas anteriormente y finalmente propone como el uso del Método Utilización, Saturación y Errores (USE) como una mejor alternativa. El método USE propone la construcción de un *checklist* que sirve para detectar rápidamente cuellos de botella de recursos o errores [15]. En la Figura 2.5, se muestra el diagrama de flujo de la metodología USE.

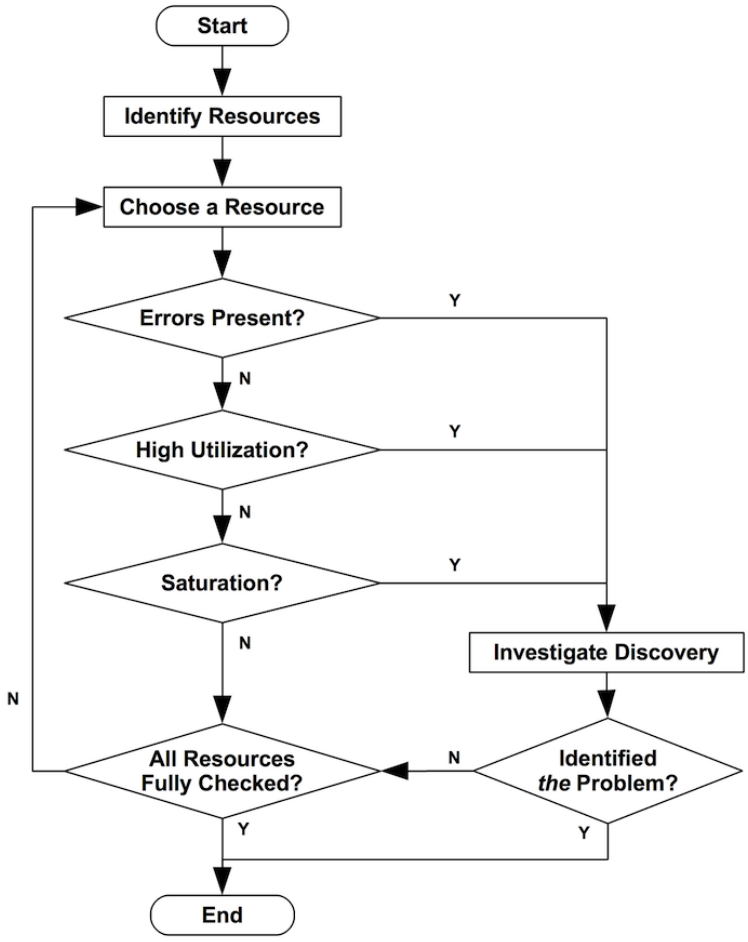

Figura 2.5: Diagrama de flujo de metodología USE.

HPCToolkit - Tools for performance analysis of optimized parallel programs Este articulo describe la herramienta HPCTOOLKIT, desarrollada por la Universidad Rice. HPCToolkit es una suite de herramientas que soporta la medición, análisis, atribución y presentación del desempeño de aplicaciones para programas secuenciales y paralelos. La metodología en la cual esta basada en un conjunto de principios complementarios [16]. A continuación se listan estos principios son:

- Ser independiente del lenguaje.
- Evitar instrumentación del código.
- Evitar puntos ciegos.
- El contexto es esencial para entender el software en capas y orientado a objetos.

- Cualquier medida de rendimiento por si sola produce una vista miope del problema.
- Las métricas de desempeño derivadas son esenciales para un análisis efectivo.
- El análisis de rendimiento debe ser de arriba a abajo.
- La agregación jerárquica es vital.
- La medición y el análisis deben ser escalables.

En la Figura 2.6, se detallan los principales componentes de HPCToolkit y sus relaciones.

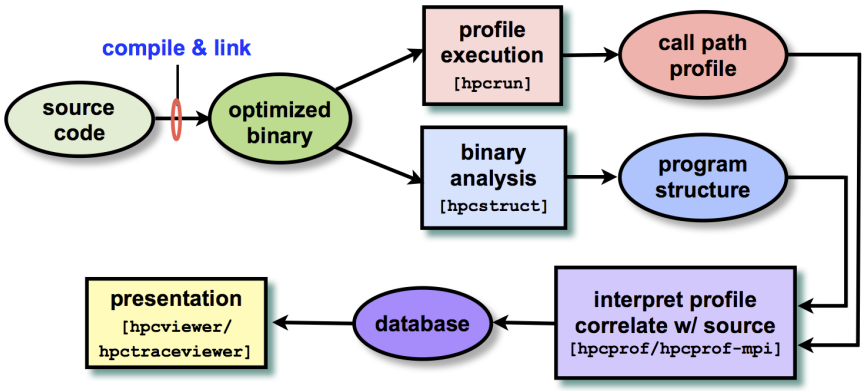

Figura 2.6: Componentes HPCToolkit.

Autoperf - Workflow Support for Performance Experiments El articulo presenta la herramienta Autoperf. Esta permite crear y administras experimentos de desempeño, incluyendo el análisis y posprocesamiento de datos [17]. El diseño de la herramienta lo componen cuatro componentes:

- Especificación del experimento, este componente es el encargado de entregar la información básica para el experimento e.g., el comando usado para invocar el programa.
- Envió del trabajo, encargo de los parámetros que dependen del entorno donde se ejecuta el experimento.
- Recolección de datos, Autoperf se basa en las herramientas de rendimiento existentes para recopilar datos de rendimiento. Actualmente, TAU y HPCToolkit son compatibles con Autoperf.
- Motor de análisis, este modulo es el encargado del análisis estadístico del experimento.

2.2.2. Alto Rendimiento

Anatomy of high-performance matrix multiplication Los autores en este articulo describen los principios que subyacen en la implementación de una multiplicación de matrices que es usada ampliamente en la librería GotoBlas. Un desempeño cerca del optimo es obtenido mediante un entendimiento completo de como la operación debe separada en niveles. La multiplicación de matrices la descompone en distintos casos especiales mas pequeños. La idea es si logra alto desempeño en los casos especiales, obtiene alto desempeño en la multiplicación de matrices [18]. En la Figura 3.1, se muestra modelo simple y refinado de la jerarquía de memoria, estos modelos son necesarios para poder optimizar la multiplicación de matrices, ya que se analiza el costo de mover datos entre capas de memoria.

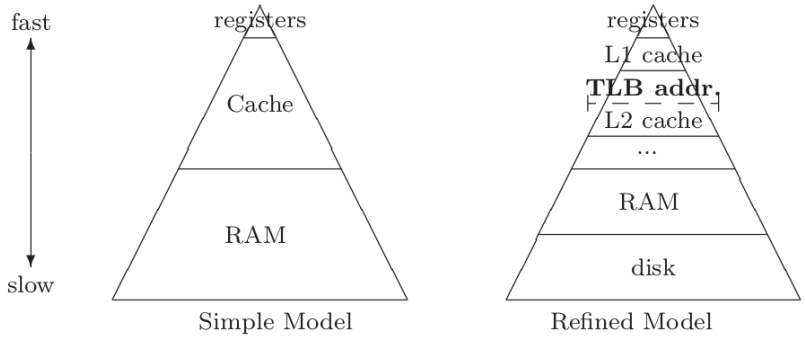

Figura 2.7: Modelos jerárquicos de memoria.

High-Performance Matrix-Matrix Multiplications of Very Small Matrices El documento describe la falta de librerías especializadas capaces de tener un alto desempeño cuando se dan situaciones donde una aplicación tenga que realizar un calculo que sea acumuladamente grande, pero sus partes individuales sean pequeñas. Por eso mismo, lo autores desarrollaron algoritmos innovadores, abstracciones de los datos y las tareas y todas la implementaciones basadas en el estándar MAGMA 2.0 [19]

LIBXSMM: Accelerating Small Matrix Multiplications by Runtime Code Generation En este articulo se presenta un librería obtiene alto desempeño en la multiplicación de matrices pequeñas. La aceleración es obtenida usando distintas técnicas. Siendo la mas importante, usar un generador de código con un modelo de arquitectura integrado que crea código que no necesita un fase de *auto-tunning* [20].

Capítulo 3

Definición del Problema y Análisis

3.1. Formulación del Problema

Actualmente la empresa Azalée cuenta con un prototipo para la medición de la porosidad y espesor oseo, este prototipo usa la técnica de transmisión axial para realizar las mediciones. Esta es una técnica desarrollada para medir la propagación de ondas guiadas de ultrasonido en la capa cortical a lo largo del eje de huesos largos. Los modos guiados propagados en la corteza son grabados con un arreglo de transductor lineal de 1-MHz. La medida de la curva de dispersión es obtenida usando una transformada de fourier de dos dimensiones (espacio, tiempo) combinada con una descomposición de valores singulares. La identificación automática de parámetros es obtenida a través de la solución del problema inverso en donde las curvas de dispersión son predichas con un modelo de placa libre isotrópica transversal bidimensional. La implementación actual de la interfaz humano-computador del prototipo tiene tiempos de respuesta mayor al deseado. El prototipo de medición de hueso cortical se muestra en la Figura 3.1.

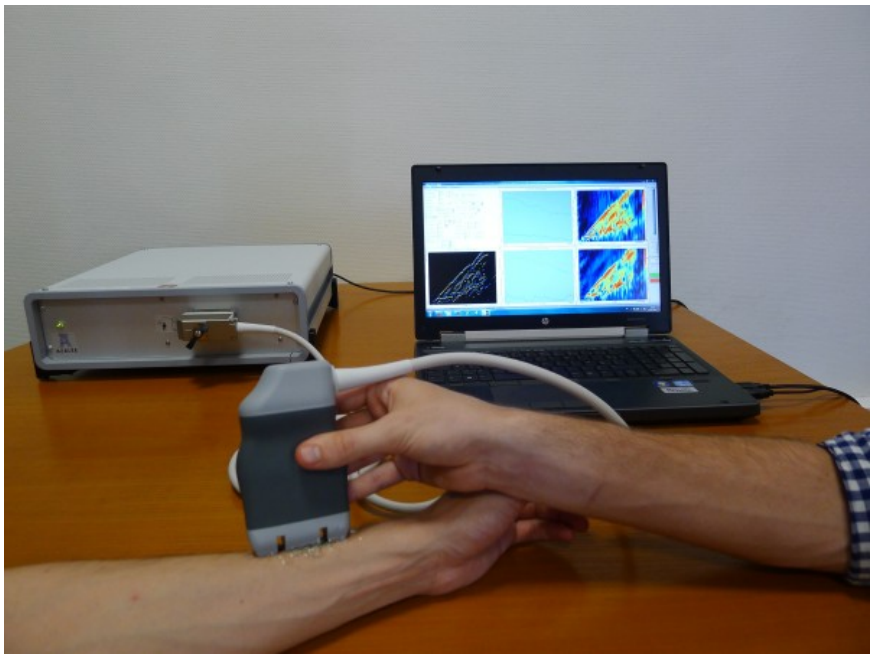

Figura 3.1: Prototipo de Azalée.

3.2. Solución Propuesta

La solución propuesta al problema consiste en revisar las etapas del algoritmo, modificar el código para reducir complejidades temporales. Analizar el rendimiento del software implementado para detectar los puntos problemáticos y las áreas dónde sea posible llevar a cabo una optimización del rendimiento.

3.3. Objetivos

A continuación se detallan los objetivos generales y específicos del trabajo de título

3.3.1. Objetivo General

Disminuir el tiempo de respuesta de la interfaz humano-computador del prototipo de mediciones corticales de huesos, optimizando las etapas del algoritmo.

3.3.2. Objetivos Específicos

1. Reducir complejidades temporales del algoritmo de análisis de datos.

2. Analizar el rendimiento del software implementado (profiling).

3. Paralelizar código a nivel de datos y tareas.

4. Acelerar el acceso a memoria ordenando los datos para tomar ventaja del cache del CPU.

5. Implementar grafico de espesor/porosidad de hueso cortical mediante la resolución del problema inverso.

3.4. Metodología

La metodología a usar es una cascada ad hoc al problema con una fase iterativa al final. Esta metodología la componen las siguientes fases.

1. **Aprendizaje**: Fase donde se familiarizara con los principios de la transmisión axial.

2. **Analizar Código:** Acá el código y el algoritmo serán analizados.

3. **Organizar Modificaciones:** Fase en la cual las modificaciones se ordenaran según las estimaciones de mejoras en el tiempo de respuesta.

4. **Realizar modificaciones** Implementación de las modificación.

5. **Medir** Etapas donde se prueba la modificación anteriormente realizada.

6. Si quedan modificaciones por realizar continuar con la siguiente modificación.

En la Figura 3.2, se presenta un diagrama de la metodología propuesta.

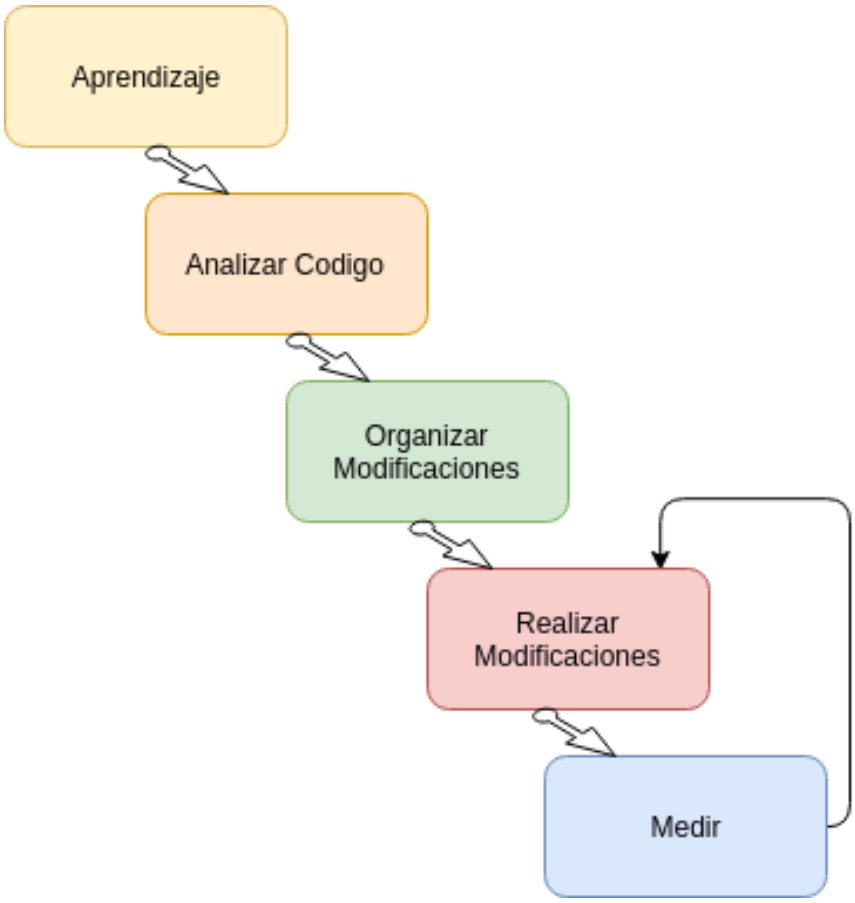

Figura 3.2: Metodología a utilizar

3.5. Especificación de Requerimientos

A continuación, se detallaran los requisitos funcionales y requisitos no funcionales que el sistema de medición de hueso cortical debe cumplir. Los requisito funcionales se entienden como una sentencia que identifica lo que el sistema debe cumplir para producir el resultado requerido deseado[21]. Los requisitos no funcionales son un requisito del software que no describe lo que el software hará, sino *como* el software lo hará[21].

3.5.1. Requerimientos Funcionales

En esta sección se detallan los requisitos funcionales del sistema. La sigla **RFxx** sera usada en el documento para hacer referencia al requisito funcional.

- **RF01** El sistema mostrara las graficas de espacio de Fourier de las señales recibidas por la sonda de ultrasonido.
- **RF02** El sistema mostrara las graficas de espesor/porosidad del hueso cortical que se este midiendo.

3.5.2. Requerimientos No Funcionales

En esta sección se detallan los requisitos no funcionales del sistema. La sigla **RNFxx** sera usada en el documento para hacer referencia al requisito no funcional.

- **RNF01** El sistema mostrará las gráficas de espacio de Fourier con una frecuencia de al menos cuatro gráficos por segundo.
- **RNF02** El sistema mostrará las gráficas de espesor/porosidad del hueso cortical con una frecuencia de al menos cuatro gráficos por segundo.
- **RNF03** El sistema sera desarrollado usando el lenguaje de programación C++.

3.6. Funcionalidades del Sistema

3.6.1. Diagramas de Casos de Uso

El diagrama de caso de uso representa la interacción del usuario con el sistema que muestra la relación entre el usuario y los diferentes caso de uso en donde el usuario esta involucrado. En el caso del sistema de medición de hueso cortical, en la Figura 3.3 se muestra el único caso de uso que es realizar medición.

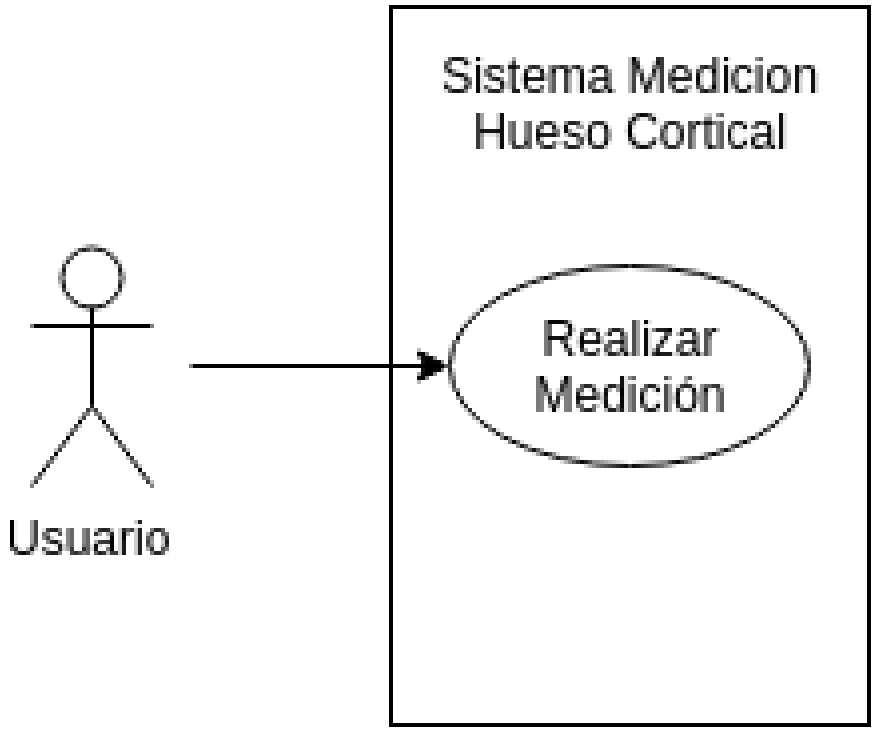

Figura 3.3: Diagrama de Caso de Uso del sistema.

3.6.2. Casos de Uso

En la siguiente sub-sección se presenta tabla de Caso de Uso extendido.

Nombre:	Realizar Medición
Descripción:	Permite al usuario realizar mediciones de espesor y porosidad del hueso cortical
Actores:	Usuario
Precondiciones:	Ninguna
Requisitos No Funcionales:	RNF01,RNF02
Flujo de Eventos	1. El usuario selecciona Realizar Medición 2. Usuario alinea la sonda con el eje del largo del hueso hasta que obtiene una solución del problema inversor 3. Finalizar medición.
Post-Condiciones	Ninguna

3.6.3. Diagramas de Secuencia

3.6.4. Diagramas de Estado

3.6.5. Modelo Conceptual

Capítulo 4

Diseño

4.1. Diseño Arquitectónico

4.1.1. Tecnologías utilizadas

En esta sección se detallaran las tecnologías a utilizar para optimizar el código del prototipo, para poder analizar el código (segunda etapa de la metodología a utilizar), la herramienta AutoPerf sera la utilizada en su ultima versión. Autoperf es una herramienta para crear y administrar experimentos de rendimiento, incluido el procesamiento y análisis de datos. Proporciona un formato simple para definir el entorno del experimento y los datos que se recopilarán, e interactúa con una variedad de herramientas de rendimiento. Junto con lo anterior, la herramienta HPCToolkit es un conjunto integrado de herramientas para medir y analizar el rendimiento del programa en computadoras que van desde sistemas de escritorio multinúcleo a las supercomputadoras más grandes del país. Al utilizar un muestreo estadístico de temporizadores y contadores de rendimiento de hardware, HPCToolkit recopila mediciones precisas del trabajo, el consumo de recursos y la ineficiencia de un programa y las atribuye al contexto de llamada completo en el que se producen. HPCToolkit funciona con aplicaciones que están vinculadas estática o dinámicamente. Dado que HPCToolkit utiliza el muestreo, la medición tiene una sobrecarga baja (1-5 %) y se escala a grandes sistemas paralelos. Las herramientas de presentación de HPCToolkit permiten un análisis rápido de los costos de ejecución, ineficiencia y características de escalamiento de un programa tanto dentro como a través de los nodos de un sistema paralelo. HPCToolkit admite la medición y el análisis de códigos de serie, códigos de subprocesos (por ejemplo, pthreads, OpenMP), MPI y códigos paralelos híbridos (subprocesos de MPI +).

En la etapa de Realizar Modificaciones se utilizara LIBXSMM que es una biblioteca para operaciones especializadas de matriz densa y dispersa, dirigidas a la arquitectura Intel. Los núcleos de multiplicación de matriz pequeña (SMM) se generan para Intel SSE,

Herramienta	Etapa	Dirección Web
HPCToolkit	Análisis de Código	http://hpctoolkit.org/
AutoPerf	Análisis de Código	https://github.com/HPCL/autoperf
LIBXSSM	Realizar Modificaciones	https://github.com/hfp/libxsmm

Tabla 4.1: Tabla Herramientas a Utilizar.

Intel AVX, Intel AVX2, IMCI (KNCni) para coprocesadores Intel Xeon Phi (KNC), e Intel AVX-512 tal como se encuentran en la familia de procesadores Intel Xeon Phi (KNL, KNM).) y procesadores Intel Xeon (SKX). La biblioteca es compatible con el código generado de manera estática en el momento de la configuración (SMM), utiliza rutas de código optimizadas basadas en el código generado por el compilador, así como en las funciones intrínsecas, pero utiliza principalmente la especialización de código Just-In-Time (JIT) para el rendimiento independiente del compilador (multiplicaciones de matrices). , matriz de transposición / copia, funcionalidad dispersa y pequeñas circunvoluciones). LIBXSMM es adecuado para "compilar una vez e implementar en todas partes", es decir, no se necesitan indicadores de destino especiales para explotar el rendimiento disponible. En la tabla 4.1 detalla las herramientas a utilizar junto con las etapas y paginas oficiales.

4.1.2. Flujo de datos / Vista de alto nivel

4.2. Diseño Lógico

4.2.1. Diagrama de despliegue

Los diagramas de despliegue (DD) se utilizan para visualizar los detalles de implementación de un sistema de software. Los diagramas incluyen más que sólo código, sino también bibliotecas separadas, un instalador, archivos de configuración y muchas otras piezas. Para que el software esté listo para ejecutarse, es necesario comprender todos los archivos y ejecutables involucrados y los entornos donde residen.

El DD Ilustra el hardware del sistema y su software. Útil cuando se implementa una solución de software en múltiples máquinas con configuraciones únicas.

Dos tipos especiales de dependencias: la importación de paquetes y la fusión de paquetes.

Pueden representar los diferentes niveles de un sistema para revelar la arquitectura. Deben indicar las dependencias entre paquetes y comunicación entre estos.

En la Figura 4.1, se presenta un ejemplo de un diagrama de despliegue.

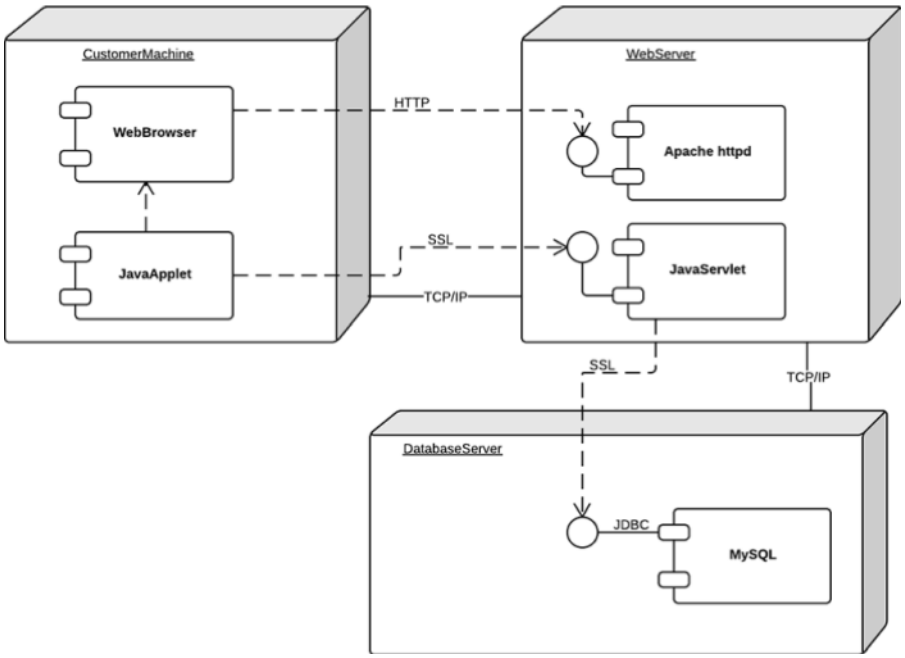

Figura 4.1: Ejemplo Diagrama de Despliegue

4.2.2. Diagrama de componentes

En esta sección se explicara el diagrama de componentes del sistema, el componente paciente carga los datos desde el componente de Persistencia. El primero carga los datos en el Controlador que entrega datos a la Interfaz. Los datos son obtenidos desde el componente Captura Datos, que el componente Medicion se los entrega al modulo DSP. Finalmente, los diagramas de componentes pueden ilustrar una relación de dependencia. Una relación de dependencia ocurre cuando la interfaz provista por un componente coincide con la interfaz requerida por otro componente. La interfaz proporcionada está representada por una bola, y la interfaz requerida está representada por un *socket*.

A continuación, en la figura 4.2, se presenta el diagrama de componentes para el prototipo:

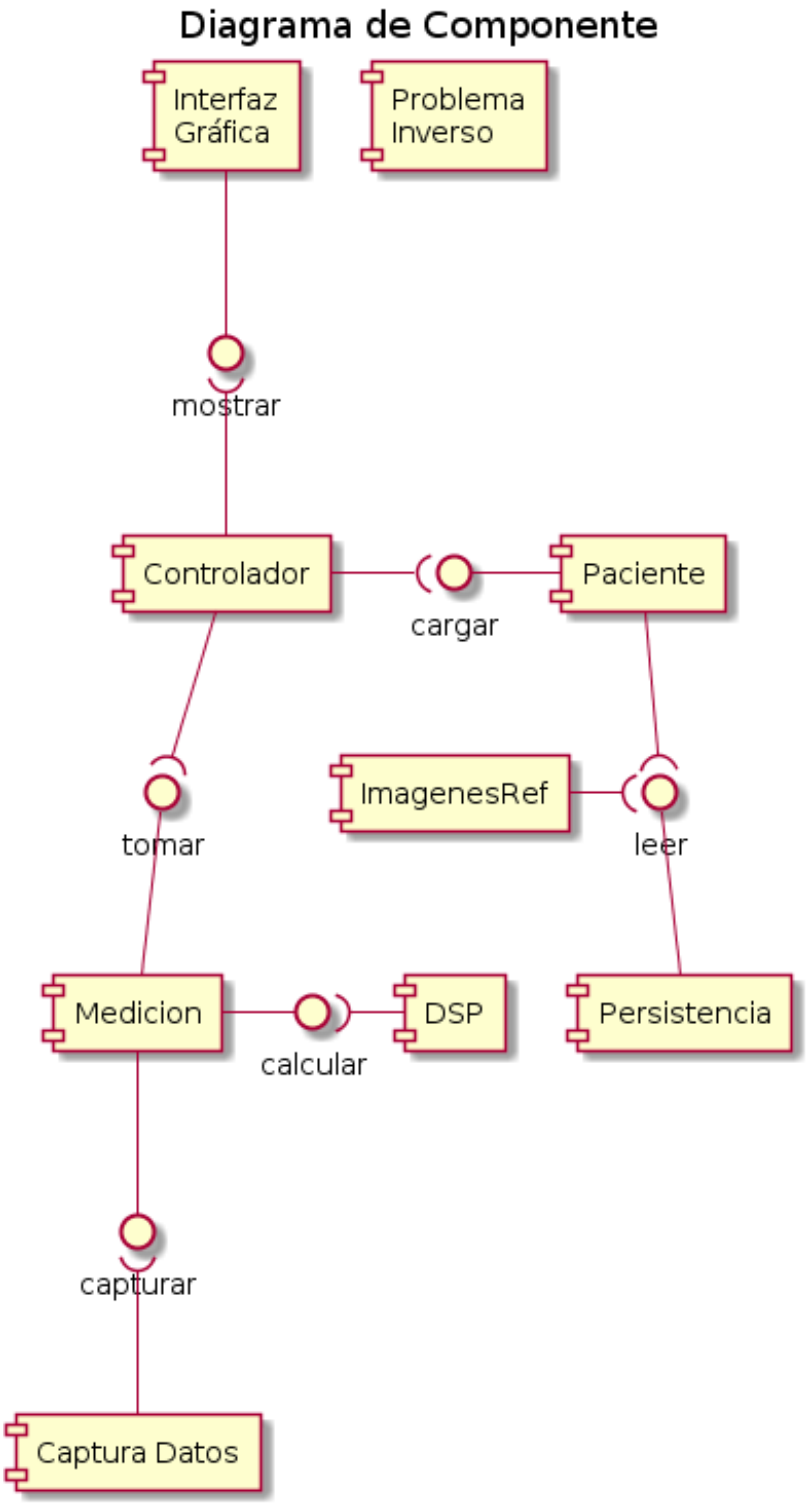

Figura 4.2: Diagrama de Componentes

Los diagramas de componentes son especialmente útiles al principio del proceso de diseño, debido a su énfasis de alto nivel. Se pueden realizar en diferentes niveles y le permiten enfocarse no sólo en los sistemas sino también en los subsistemas.

4.2.3. Diagrama de paquetes

Los diagramas de paquetes muestran los paquetes y las dependencias entre ellos. Estos diagramas pueden organizar un sistema completo en paquetes de elementos relacionados, estos podrían incluir datos, clases o incluso otros paquetes. Los diagramas de paquetes ayudan a proporcionar agrupaciones de alto nivel de un sistema para que sea fácil visualizar cómo un paquete contiene elementos relacionados, así como la forma en que los diferentes paquetes dependen entre sí.

A continuación, en la figura 4.3, se presenta un ejemplo de un diagrama de paquetes para un videojuego:

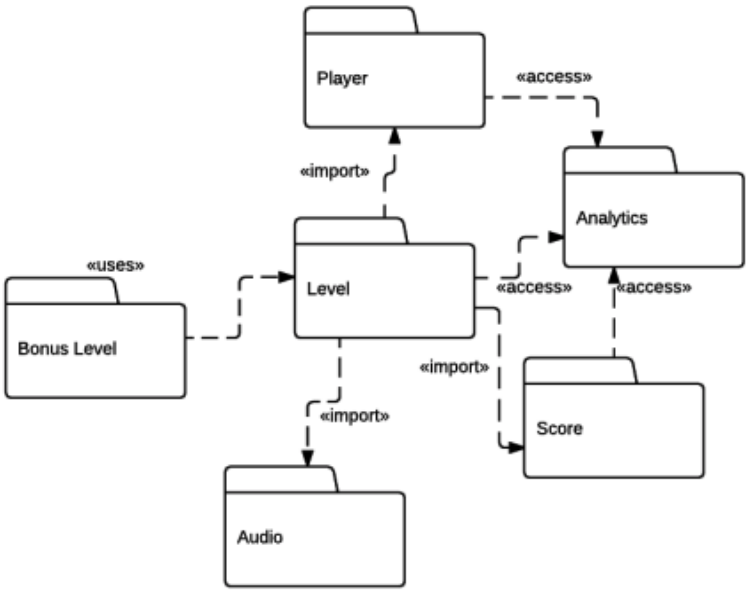

Figura 4.3: Ejemplo Diagrama de Paquetes

4.2.4. Diagrama de clases

El Diagrama de clase da la vista estática de una aplicación. Un diagrama de clase describe los tipos de objetos en el sistema y los diferentes tipos de relaciones que existen entre ellos. Este método de modelado se puede ejecutar con casi todos los métodos orientados a objetos.

El Diagrama de clase ofrece una descripción general de un sistema de software al mostrar clases, atributos, operaciones y sus relaciones. Este diagrama incluye el nombre de la clase, los atributos y la operación en compartimientos designados separados.

Para finalizar, el Diagrama de clase ayuda a construir el código para el desarrollo de aplicaciones de software.

Los elementos esenciales en un diagrama de clases son:

- Nombre de la clase
- Atributos
- Operaciones

A continuación, en la figura 4.4, se presenta un ejemplo de un diagrama de clases:

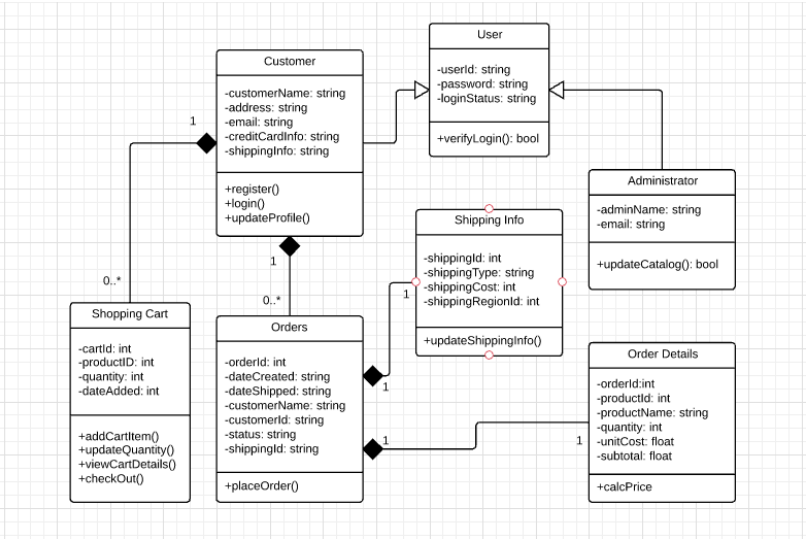

Figura 4.4: Ejemplo Diagrama de Clases

4.3. Diseño de Datos

4.3.1. Diagrama Entidad Relación

4.3.2. Diagrama Relacional

4.3.3. Diccionario de Datos

4.4. Diseño de Interfaz

4.4.1. Arquitectura de la Información

4.4.2. Prototipos de Interfaces Gráficas

4.5. Diseño de Pruebas

4.5.1. Pruebas Unitarias

4.5.2. Pruebas de Integración

4.5.3. Pruebas con Usuarios

4.5.4. Pruebas de Aceptación

4.6. Diagramas Opcionales / Complementario

4.6.1. Diagramas de Actividad

Capítulo 5

Implementación

5.1. Hardware utilizado

El hardware utilizado para el desarrollo de esta tesis corresponde a un notebook **Lenovo B40**. Este notebook cuenta con un procesador **Intel Core i3-5005U** con una frecuencia base de 2.0 Ghz, contando con 2 núcleos. Un núcleo es un termino de hardware que describe el numero de unidades centrales de proceso independientes en un componente único de computo. La frecuencia base del procesador describe la velocidad a la que funcionan los transistores del procesador. Este procesador cuenta con 3 niveles de cache. En el nivel 3 cuenta con 3 Megabytes de memoria, en el nivel 2 con 256 Kilobytes y el nivel 1 cuenta con 32 Kilobytes para los datos y 32 Kilobytes para las instrucciones. La CPU Cache es un área de memoria rápida ubicada en el procesador. Este procesador soporta las extensiones del conjunto de instrucciones Intel® SSE4.1, Intel® SSE4.2, Intel® AVX2. Las Extensiones de conjunto de instrucciones son instrucciones adicionales que pueden aumentar el rendimiento cuando se realizan las mismas operaciones en múltiples objetos de datos. Estos pueden incluir SSE (Streaming SIMD Extensions) y AVX (Advanced Vector Extensions). El notebook utilizado cuenta con 8 Gigabytes de memoria principal del tipo DDR3L.

Todas esta características son importantes para los tiempos de ejecución del programa. La frecuencia base esta asociada a la cantidad de instrucciones que un procesador puede llevar a cabo en un tiempo determinado, la cantidad de nucleos esta ligada a la capacidad de paralelizar tareas. Las memorias cache del procesador son usadas para reducir el tiempo promedio en acceder a los datos desde la memoria principal ya que se encuentran cerca del núcleo del procesador. Las extensiones del conjunto de instrucciones nos permiten obtener paralelismo a nivel de datos.

5.2. Software utilizado

A continuación se detallaran los software utilizados para este trabajo. Se uso MATLAB que es un software para computación matemática, análisis, visualización y desarrollo de algoritmos, para el prototipo del algoritmo de transmision axial permitiendo modificar el codigo de forma mas sencilla. Se utilizo Matlab Coder para transformar el codigo en Matlab a codigo en el lenguaje de programacion C. Esta transformacion permite poder analizar el ejecutable de este codigo con HPCToolkit y asi poder identificar ineficiencias en el codigo, como subutilizacion de la cache. En este trabajo

5.3. Lenguajes de programación

Los lenguajes utilizado fueron el lenguaje propietario de Matlab, junto con el lenguaje de programacion C. La eleccion de Matlab es por la capacidad de convertir el codigo original en codigo en C permitiendo construir binarios que se puedan analizar.

5.4. Estrategia de implementación

La estrategia utilizada para implementar sera primero realizar un analisis del codigo actual y realizar un *refactoring* de este. La refactorización de código es el proceso de reestructuración del código de computadora existente, sin cambiar su comportamiento externo. La refactorización está destinada a mejorar los atributos no funcionales del software. Dentro de estos cambios, primero se buscan *olores de codigo*. A continuacion se listan los mas comunes.

- **Bloaters** son código, métodos y clases que han aumentado a proporciones que son difíciles de trabajar. Por lo general, estos olores no aparecen de inmediato, sino que se acumulan con el tiempo a medida que el programa evoluciona.
- **Previene el cambio** Estos olores significan que si se necesita cambiar algo en un lugar en el código, también se tienen que hacer muchos cambios en otros lugares. El desarrollo del programa se vuelve mucho más complicado y costoso como resultado.
- **Un prescindible** es algo inútil e innecesario cuya ausencia haría que el código sea más limpio, más eficiente y más fácil de entender.
- **Acopladores** los olores en este grupo contribuyen a un acoplamiento excesivo entre clases o muestran lo que sucede si el acoplamiento es reemplazado por una delegación excesiva.

Para solucionar estos problemas se ocuparan distintas técnicas de refactoring.

- **Métodos de composición** gran parte de la refactorización está dedicada a componer correctamente los métodos. En la mayoría de los casos, los métodos excesivamente largos son la raíz de todo mal. El código dentro de estos métodos ocultan la lógica de ejecución y hacen que el método sea extremadamente difícil de entender, y aún más difícil de cambiar.

  Las técnicas de refactorización en este grupo simplifican los métodos, eliminan la duplicación de código y allanan el camino para futuras mejoras.

- **Moviendo características entre objetos** muestran cómo mover de manera segura la funcionalidad entre clases, crear nuevas clases y ocultar los detalles de implementación del acceso público.

- **Organizando datos** ayudan con el manejo de datos, reemplazando los primitivos con una funcionalidad de clase rica. Otro resultado importante es el desenredado de las asociaciones de clase, lo que hace que las clases sean más portátiles y reutilizables.

- **Simplificando Expresiones Condicionales** los condicionales tienden a complicarse cada vez más en su lógica a lo largo del tiempo.

5.5. Interfaces

Bibliografía

[1] Office of the Surgeon General (US). (2004) Bone health and osteoporosis: A report of the surgeon general. rockville (md). [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK45504/

[2] C. Lorincz, S. L. Manske, and R. Zernicke, "Bone health: Part 1, nutrition," *Sports Health*, vol. 1, no. 3, pp. 253–260, May 2009, 10.1177_1941738109334213[PII]. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3445243/

[3] D. M. Cooper, C. E. Kawalilak, K. Harrison, B. D. Johnston, and J. D. Johnston, "Cortical Bone Porosity: What Is It, Why Is It Important, and How Can We Detect It?" *Current Osteoporosis Reports*, vol. 14, no. 5, pp. 187–198, oct 2016. [Online]. Available: http://link.springer.com/10.1007/s11914-016-0319-y

[4] J. L. Dinamarca-Montecinos, G. Améstica-Lazcano, R. Rubio-Herrera, A. Carrasco-Buvinic, and A. Vásquez, "Características epidemiológicas y clínicas de las fracturas de cadera en adultos mayores en un hospital público chileno," *Revista Medica de Chile*, vol. 143, no. 12, pp. 1552–1559, dec 2015. [Online]. Available: http://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0034-98872015001200008&lng=en&nrm=iso&tlng=en

[5] Y. Bala, R. Zebaze, and E. Seeman, "Role of cortical bone in bone fragility," *Current Opinion in Rheumatology*, vol. 27, no. 4, pp. 406–413, jul 2015. [Online]. Available: http://content.wkhealth.com/linkback/openurl?sid=WKPTLP:landingpage&an=00002281-201507000-00013

[6] Tortora/Derrickson, *Anatomy and Physiology, 15th edition*. Wiley Custom Learning Solutions, nov 2018.

[7] A. Webb and N. B. Smith, *Introduction to medical imaging : physics, engineering, and clinical applications*. Cambridge University Press, dec 2011. [Online]. Available: https://doi.org/10.1017/CBO9780511760976

[8] F. M. Abu-Zidan, A. F. Hefny, and P. Corr, "Clinical ultrasound physics," *Journal of emergencies, trauma, and shock*, vol. 4, no. 4, pp. 501–503,

2011. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/22090745https://www.ncbi.nlm.nih.gov/pmc/PMC3214508/

[9] K. Nakamura, *Ultrasonic Transducers: Materials and Design for Sensors, Actuators and Medical Applications*, ser. Woodhead Publishing Series in Electronic and Optical Materials. Elsevier Science, 2012. [Online]. Available: https://books.google.cl/books?id=MZVwAgAAQBAJ

[10] D. Ensminger and F. Stulen, *Ultrasonics: Data, Equations and Their Practical Uses*. CRC Press, 2008. [Online]. Available: https://books.google.cl/books?id=u2XlPFCbVPgC

[11] N. J. Salkind, *Encyclopedia of Measurement and Statistics 3-Volume Set*. SAGE Publications, Inc, oct 2006. [Online]. Available: https://www.amazon.com/dp/1412916119/

[12] A. V. Oppenheim, *Signals, Systems and Inference*. Pearson, apr 2015. [Online]. Available: https://www.xarg.org/ref/a/0133943283/

[13] L. Hogben, *Handbook of Linear Algebra*, ser. Discrete Mathematics and Its Applications. Taylor & Francis, 2007. [Online]. Available: https://books.google.cl/books?id=5GQRnwEACAAJ

[14] J. G. Minonzio, N. Bochud, Q. Vallet, Y. Bala, D. Ramiandrisoa, H. Follet, D. Mitton, and P. Laugier, "Bone cortical thickness and porosity assessment using ultrasound guided waves: An ex vivo validation study," *Bone*, vol. 116, no. June, pp. 111–119, 2018. [Online]. Available: https://doi.org/10.1016/j.bone.2018.07.018

[15] B. Gregg, "Thinking methodically about performance," *Communications of the ACM*, vol. 56, no. 2, p. 45, feb 2013. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2408776.2408791

[16] L. Adhianto, S. Banerjee, M. Fagan, M. Krentel, G. Marin, J. Mellor-Crummey, and N. R. Tallent, "HPCTOOLKIT: tools for performance analysis of optimized parallel programs," *Concurrency and Computation: Practice and Experience*, pp. n/a–n/a, 2009. [Online]. Available: http://doi.wiley.com/10.1002/cpe.1553

[17] X. Dai, B. Norris, and A. D. Malony, "Autoperf," in *Proceedings of the 2015 Workshop on Challenges in Performance Methods for Software Development - WOSP '15*. New York, New York, USA: ACM Press, 2015, pp. 11–16. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2693561.2693569

[18] K. Goto and R. A. van de Geijn, "Anatomy of high-performance matrix multiplication," *ACM Transactions on Mathematical Software*, vol. 34, no. 3, pp. 1–25, may 2008. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1356052.1356053

[19] I. Masliah, A. Abdelfattah, A. Haidar, S. Tomov, M. Baboulin, J. Falcou, and J. Dongarra, "¡," 2016, pp. 659–671. [Online]. Available: http://link.springer.com/10.1007/978-3-319-43659-3_48

[20] A. Heinecke, G. Henry, M. Hutchinson, and H. Pabst, "LIBXSMM: Accelerating Small Matrix Multiplications by Runtime Code Generation," *undefined*, 2016. [Online]. Available: https://www.semanticscholar.org/paper/LIBXSMM%3A-Accelerating-Small-Matrix-Multiplications-Heinecke-Henry/ac473f1674f14253da0e50c25b8cb86f8801a808

[21] K. M. G. Adams, *Non-functional Requirements in Systems Analysis and Design*, ser. Topics in Safety, Risk, Reliability and Quality. Springer International Publishing, 2015. [Online]. Available: https://books.google.cl/books?id=U7-pCAAAQBAJ