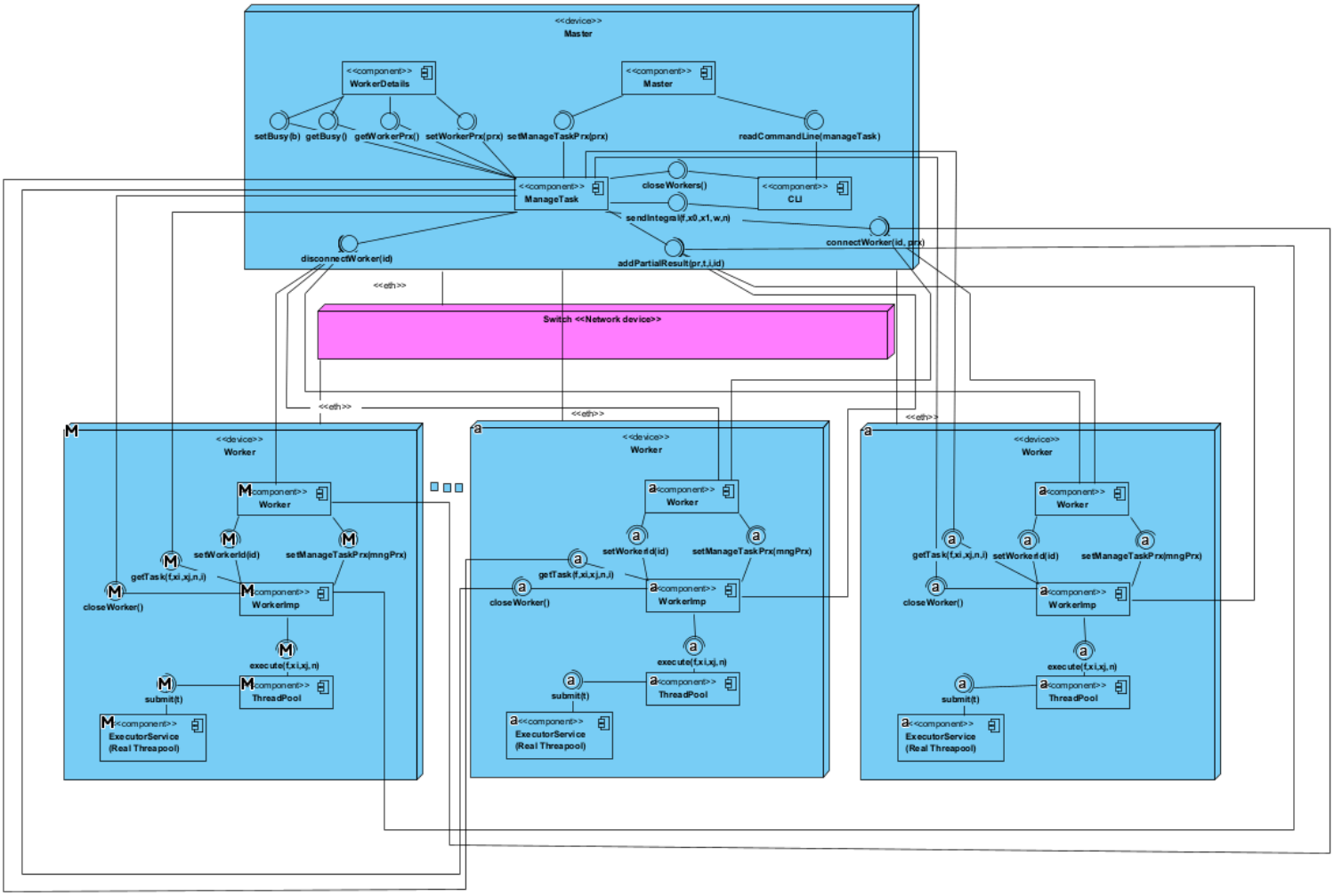
Proyecto Arquitectura de Software

Diagrama de deployment



Estructura de Flynn

Estructura de almacenamiento

La estructura de almacenamiento a utilizar será UMA (Unified memory access), esto debido a que la información obtenida y almacenada estará en un solo lugar, donde estarán los resultados parciales, y finalmente se hará la unión de todos.

Estructura de procesamiento

La estructura de procesamiento a utilizar es la de Single Instruction, Multiple Data (SIMD). Esto se debe a que a cada uno de los procesadores (dispositivos) se le entregará la misma forma de resolver la integral (utilizando el método de montecarlo) pero con una integral definida en distintos rangos (representando al dato a procesar).

Estilos de arquitectura

El estilo de arquitectura a usar es Blackboard. Se tendrá un repositorio centralizado donde se almacenarán las soluciones parciales realizadas por cada una de las máquinas conectadas, de esta manera se realiza una colaboración para resolver el problema de aproximación de integrales.

Patrones de diseño

Patrones a tener en cuenta en el sistema distribuido:

- **Master - Worker**
    - Teniendo en cuenta que es un patrón de distribución, el master worker nos permitirá realizar el procesamiento de la integral separándose en conjuntos de tareas independientes, generando una ventaja al momento de realizar el procesamiento. Cuando se tengan las aproximaciones de cada una de las particiones, se regresarán al componente master donde se agruparán y darán el resultado final.
- **ThreadPool**
    - En este caso que tenemos un problema de concurrencia, los hilos serán una gran herramienta para paralelizar diferentes operaciones, como lo es la aproximación de funciones utilizando monte carlo. Sin embargo, queremos realizar un manejo adecuado de dichos hilos, ya que estos podrán ser reutilizados numerosas veces.

Explicación de la estrategia para resolver el problema de forma distribuida

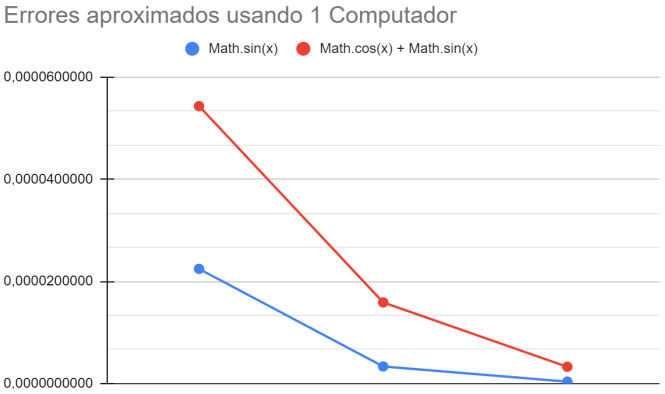
1. El usuario podrá ingresar una integral por medio de la línea de comandos en una sintaxis específica, incluyendo además el rango de dicha integral así como el nivel de precisión deseado.
2. El sistema Master realizará una partición del rango de la integral según el número de máquinas Workers disponibles para procesar y que están conectadas al master.
3. El sistema Master enviará la integral a cada uno de los Workers así como el rango especificado.
4. Los Workers con su integral y rango definido, realizarán una nueva partición del rango según la cantidad de hilos disponibles, de esta manera cada uno de los hilos usando el método montecarlo aproximará la integral.
5. Los Workers sumarán sus aproximaciones y cuando tengan todas listas se le enviarán al Master.
6. Cuando el Master reciba todas las aproximaciones parciales dadas por los Workers, deberá realizar la suma de ellas para que finalmente retorne el resultado aproximado de la integral inicial.

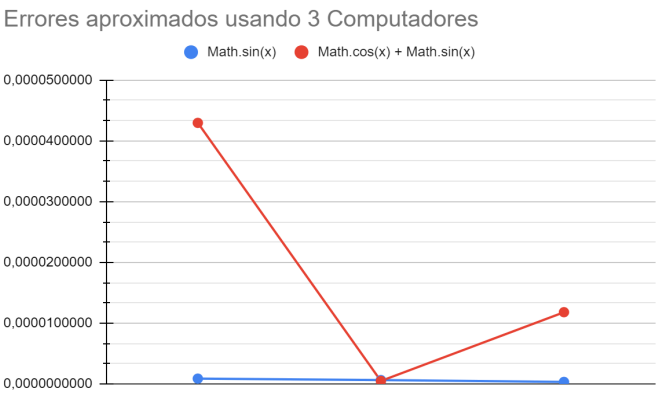
Tabla de valores con parámetros de configuración vs tiempos vs precisión

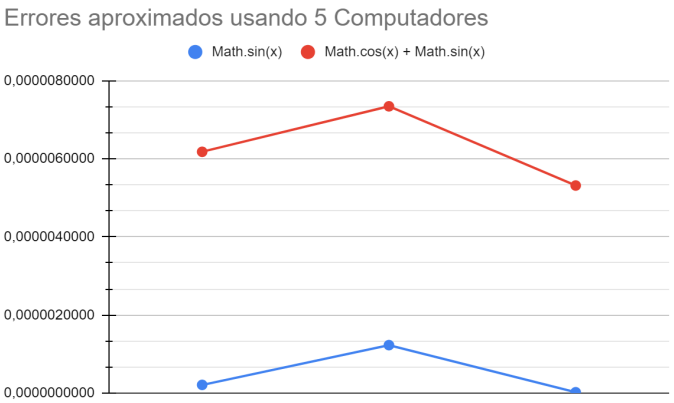
Gráficos comparativos de las n ejecuciones

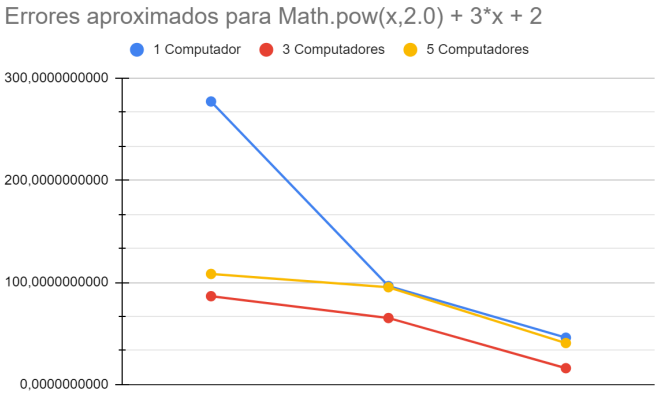
A continuación se presentan los diferentes resultados obtenidos utilizando diferentes configuraciones sobre el sistema distribuido de aproximación de integrales.

Resultados









Análisis final de los resultados

- Se observa una mejora sustancial al aplicar el método de montecarlo utilizando un número elevado de números aleatorios para aproximar las funciones, a mayor cantidad de valores generados, mejor será la precisión en la mayoría de los casos.
- A mayor cantidad de computadores procesando números aleatorios, la velocidad de creación de resultados y la precisión de las funciones mejora considerablemente.
- Debido a que el método Montecarlo se basa en la generación de valores aleatorios, hay ocasiones donde los resultados son mejores en algunas ejecuciones que en otras.