MPointers 2.0 Documentation

David Obando Blanco y Sofia Gonzales

April 10, 2025

Introducción

MPointers 2.0 es un sistema de gestión de memoria que consta de dos componentes principales: un servicio de gestión de memoria y una biblioteca de MPointers. El servicio de gestión de memoria administra un bloque de memoria reservado, mientras que la biblioteca de MPointers permite a las aplicaciones interactuar con esta memoria a través de una interfaz similar a un puntero.

Contents

1 Breve descripción del problema

El problema abordado por este proyecto es la gestión eficiente de la memoria en aplicaciones que requieren un control preciso sobre la asignación y liberación de memoria.

2 Descripción de la solución

La solución implementada consiste en un servicio de gestión de memoria que maneja la asignación y liberación de memoria, y una biblioteca de MPointers que proporciona una interfaz similar a un puntero para interactuar con la memoria gestionada. Se utiliza un sistema de conteo de referencias para la gestión automática de la memoria.

3 Diseño general

3.1 Diagrama de clases UML

4 Problemas encontrados y soluciones

Durante el desarrollo del proyecto, se encontraron varios problemas que requirieron atención especial. A continuación, se detallan los principales problemas y sus respectivas soluciones:

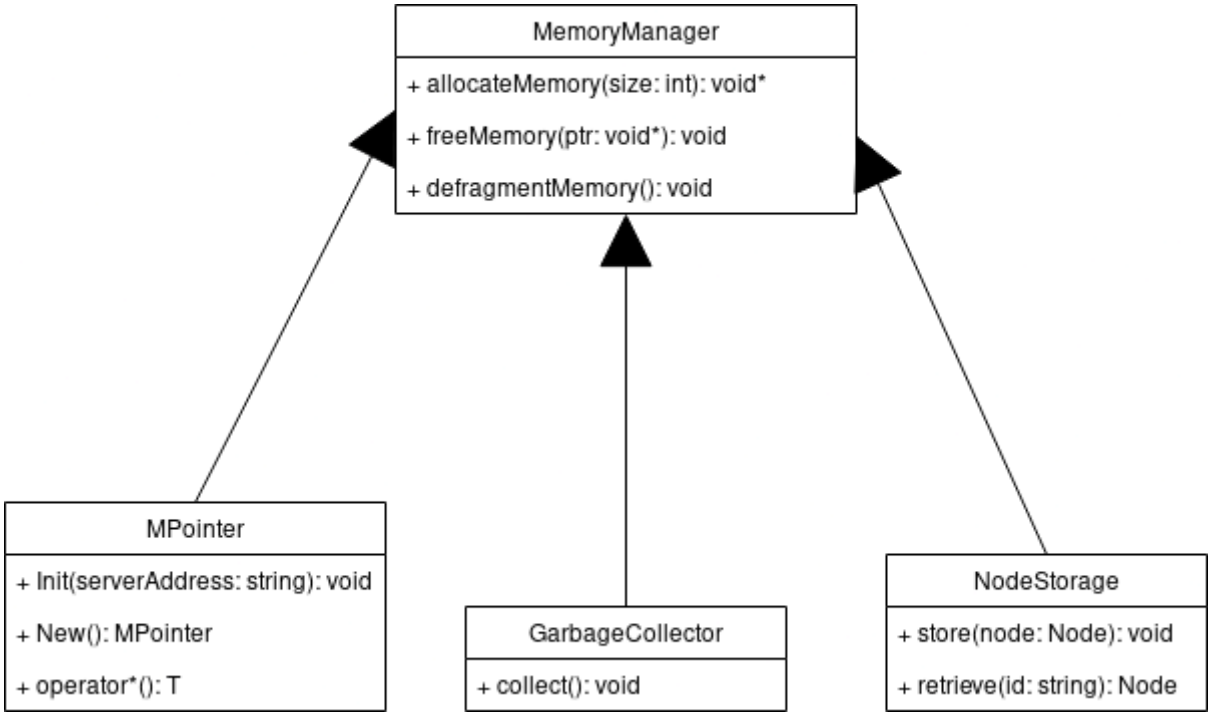

Figure 1: Diagrama de clases UML del diseño orientado a objetos.

4.1 Problema de segmentation fault en el test de lista enlazada

Problema: El programa terminaba con SIGSEGV al ejecutar el test de lista enlazada, lo que indicaba un problema de acceso a memoria inválido.

Solución: Se implementó el operador de asignación de valores en la clase MPointer para manejar correctamente las asignaciones entre punteros, asegurando que las referencias se actualicen apropiadamente.

4.2 Problema con la gestión de memoria en MPointer

Problema: Se detectaron posibles memory leaks en la clase MPointer, lo que podría resultar en una pérdida gradual de memoria disponible.

Solución: Se realizó una revisión exhaustiva y se corrigió la implementación de los operadores de copia y movimiento, asegurando una gestión adecuada del ciclo de vida de los objetos.

4.3 Error de vinculación con libsystemd en Artix Linux

Problema: gRPC requería systemd como dependencia, pero el sistema utilizaba runit como sistema de inicio.

Solución: Se compiló gRPC sin dependencias de systemd utilizando la opción -DgRPC_USE_SYSTEMD=OFF, permitiendo su uso en sistemas que no dependen de systemd.

4.4 Incompatibilidad de versiones entre bibliotecas

Problema: Se presentaron conflictos debido a diferentes versiones de las bibliotecas Abseil, RE2 y gRPC.

Solución: Se identificaron y se utilizaron las siguientes versiones compatibles:

- gRPC 1.60.0
- Abseil 20230802.1
- RE2 20231101
- Protobuf 25.1

Enlace al repositorio de Github

El código fuente del proyecto está disponible en el siguiente enlace: `https://github.com/TheDeGeO/MPointers2`