Instituto Tecnológico de Costa Rica

Escuela de Computación, Escuela de Electrónica

Curso CE2103

Algoritmos y Estructuras de Datos II

Grupo 1

Proyecto Programado #1

RMLib

Elaborado por:

Erick Obregón Fonseca

2016123157

Profesor del curso:

Jose Isaac Ramirez Herrera

Semestre II

2017

Índice

Introducción

El presente proyecto se lleva a cabo con el fin de desarrollar una biblioteca, en el lenguaje C/C++, que sea capaz de encapsular las funcionalidades necesarias para el manejo de memoria de manera remota. La memoria disponible, tanto en el nodo primario como el secundario, es gestionada por un servidor, que escucha por sockets.

El interés de este proyecto se desarrolló en torno a la idea de implementar una biblioteca genérica empaquetada como una biblioteca de C/C++ que permita que otros programas, creados en este lenguaje, puedan ser enlazados con ella y utilizar sus funcionalidades. La biblioteca se llamará simplemente rmlib y funciona como una interfaz a la memoria, la cual se ejecuta como un servidor por sockets.

Como sistema operativo se utilizó la distribución de Linux, Ubuntu 14.06 LTS. Para la codificación de la biblioteca “rmlib” se utilizó el lenguaje C/C++ y, como editor de código, Qt Creator para facilitar la creación de la interfaz gráfica, tanto del usuario cuando esté manejando la memoria, como del servidor para mostrar los elementos que han sido guardados en la memoria que posee. En la documentación del código fuente se utilizó Doxygen para generar la documentación de manera automática.

El proyecto contiene la librería que le permite enlazar una aplicación con ella y dotarla de funcionalidades que permiten la conexión del servidor por medio de un ip y un puerto tanto del servidor activo como del pasivo. La biblioteca permite integrar las funcionalidades de agregar un elemento a la memoria, eliminarlo y poder ver que hay dentro de ella por medio de una llave.

Descripción del problema

El proyecto consiste en un programa cliente-servidor. El cliente es una biblioteca que encapsula funciones de manejo de memoria. El servidor es un programa que gestiona la memoria disponible en el nodo principal y el nodo secundario.

La biblioteca cliente, llamada “rmlib”, es una biblioteca genérica que podrá ser empleada en el lenguaje C++. Debe permitir a otros programas ser enlazados y poder utilizar sus funcionalidades para conectarse a una memoria remota. La memoria remota se ejecuta como un servidor por sockets, el cual cuenta con failover automático. Antes de enviar una petición, la biblioteca verifica que el servidor esté disponible, de lo contrario, la remitirá al servidor HA (High Availability).

El servidor es un programa que escucha por sockets y que conoce de la existencia de un servidor pasivo de alta disponibilidad y se encarga de sincronizarlo. Cuando se inicia el programa servidor, se indica si será pasivo o activo. Cuando se especifica que el servidor será pasivo, se debe indicar el IP y puerto del activo. El cual se reporta con el activo y empieza a recibir los datos de sincronización.

El servidor activo y el pasivo constantemente verifican que el otro esté funcionando. En caso de que el activo no esté disponible, el pasivo asume el rol del activo. Cuando el activo original regresa, el pasivo lo sincroniza y entrega el rol de activo.

La memoria se comporta en esencia como un mapa llave-valor, por así decirlo, es una lista enlazada en la que hay una llave que permite buscar el valor guardado. A parte del valor, se guarda un conteo de referencias que permite, posteriormente, hacer recolección de basura. La memoria remota tiene un proceso que corre cada cierto tiempo, buscando la memoria que no tenga referencias activas, y la elimina automáticamente.

Para mejorar el tiempo de respuesta, la memoria debe tener un caché pequeño de no más de 5 entradas. Se debe implementar un algoritmo para controlar este caché. También se le solicita que implemente un mecanismo de control de concurrencia, puesto que la memoria puede ser accedida por varios programas a la vez. Además, se debe implementar un mecanismo de encripción/desencripción entre “rmlib” y el servidor, de tal forma que el envío de datos sea seguro.

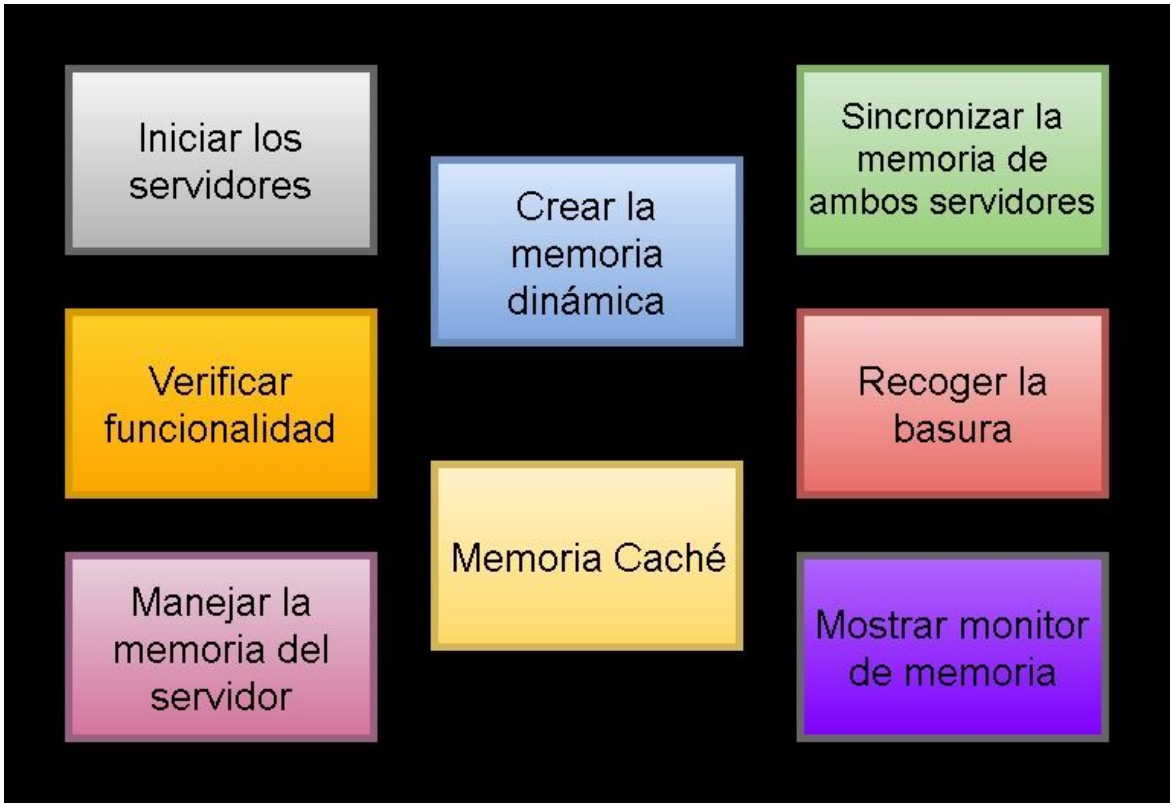
Para finalizar, se debe implementar un monitor con interfaz gráfica que permita ver el estado de la memoria y del caché, donde se muestren todos los datos necesarios para demostrar la funcionalidad.

Planificación y administración del proyecto

Lista de historias de usuario

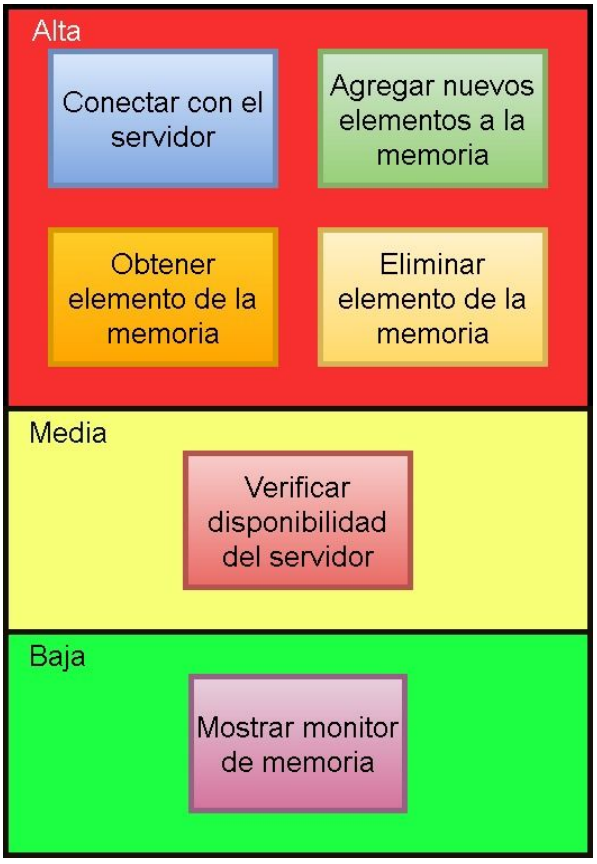
- **RmLib:**

- **Servidor:**

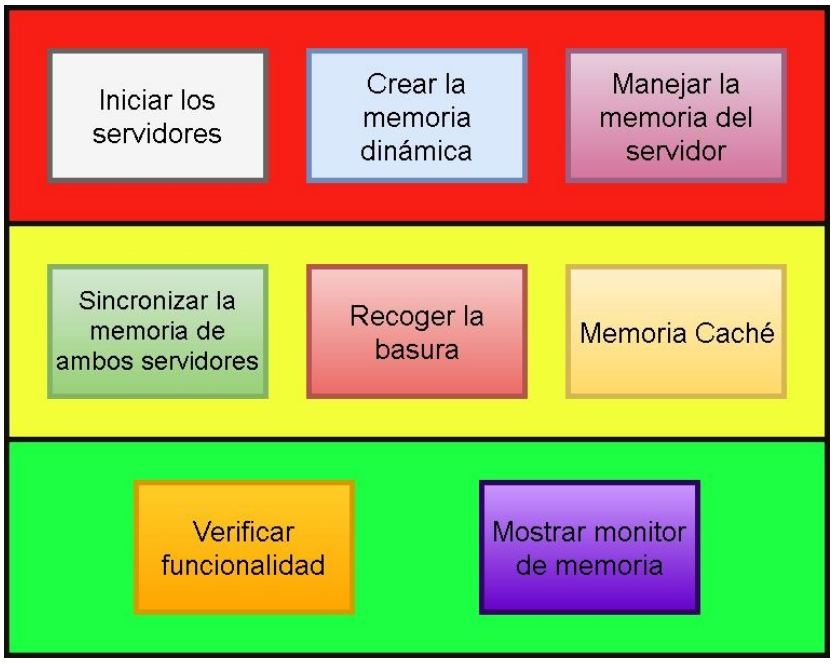


Distribución de historias de usuario

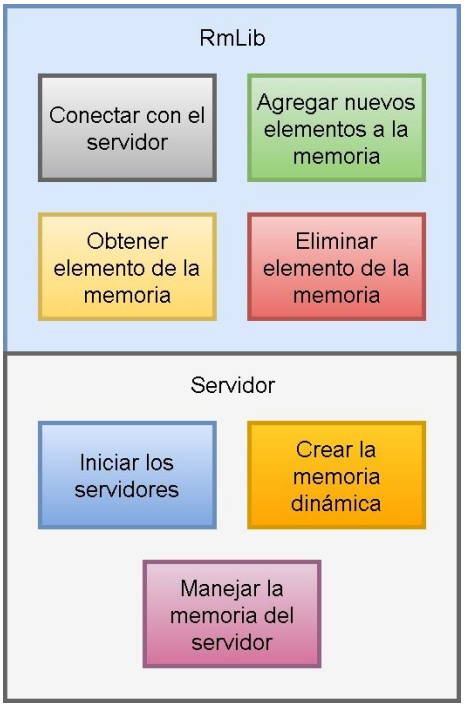
➢ **RmLib:**



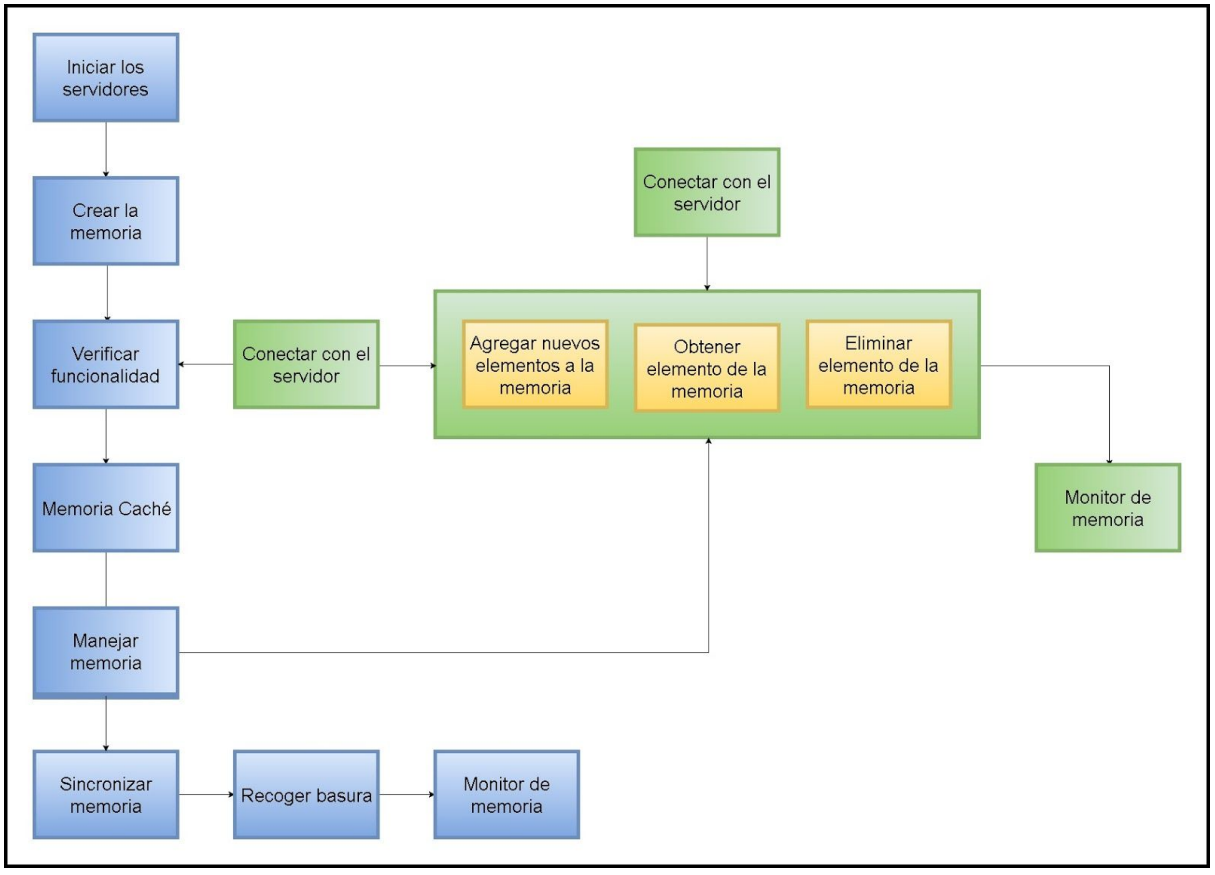
➢ **Servidor:**



Minimal System Spam



Plan de iteraciones



Descomposición de las historias de usuario en tareas

- **RmLib:**
  - Conectar con el servidor:
    - Recibir los IPs y los puertos.
    - Realizar la conexión.
  - Agregar nuevo elemento:
    - Solicitar key, data y size.
    - Verificar si la llave existe.
    - Enviar la información al servidor.
    - Crear nuevo elemento.
    - Agregar a la memoria.
    - Arrojar excepción si la llave ya existe.
  - Obtener elemento:
    - Solicitar la llave.
    - Mandar la solicitud al servidor.
    - Retorna un elemento rmRef_h.
    - Sobrecargar operador =.
    - Sobrecargar operador ==.
    - Sobrecargar operador !=.
  - Borrar elemento:
    - Verificar si el elemento existe.
    - Enviar petición al servidor.
    - Eliminar el elemento de la memoria.
  - Verificar disponibilidad.
    - Mandar petición al servidor activo, si no contesta, enviar al pasivo.
    - Conectar al servidor.
  - Mostrar monitor de memoria.
    - Mostrar caché.

- **Servidor:**
    - Iniciar los servidor.
        - Crear sockets.
    - Crear memoria.
        - Crear nodo llave valor.
        - Crear lista enlazada.
        - Crear conteo de referencia.
    - Sincronizar memoria.
        - Reportar el servidor pasivo con el activo.
        - Comparar memoria del activo y el pasivo.
        - Sincronizar la memoria.
    - Verificar funcionalidad.
        - Verificar que el servidor activo funcione.
        - Si el activo no funciona, poner pasivo como nuevo activo.
    - Recoger basura.
        - Buscar referencia inactivas.
        - Eliminar elemento sin referencias.
    - Crear el caché.
        - Crear el caché con solo 5 entradas.
        - Guardar los últimos 5 elementos accedidos.
        - Eliminar elemento si se accede a uno que no esté en el caché.
    - Manejo de memoria.
        - Recibir petición del cliente.
        - Agregar/obtener/eliminar un objeto de la memoria.
        - Responder petición del cliente.
    - Mostrar monitor de memoria.
        - Mostrar memoria.

Bitácora de Trabajo

Fecha: Sábado 5 de Agosto.

Tareas:

- ➢ Se inició el trabajo escrito con la elaboración de la portada, el índice, la descripción del proyecto y la bitácora de trabajo. Se diseñó un esqueleto previo con todos los componentes que debe incluir el mismo.
- ➢ Se comenzó con la investigación sobre funciones, clases y programación orientada a objetos, así como los punteros dentro de la clases. También se investigó más a fondo sobre herencia, sobrecarga de operadores y funciones, polimorfismo, manejo de memoria y la importación de las librerías, así como las más usadas y con las funciones más importantes.
- ➢ Se realizó la identificación de las historias de usuario tanto para el servidor como para la biblioteca cliente.
- ➢ Se ordenaron las historias de usuario por criticalidad y secuencia de uso para, posteriormente, diseñar el Minimal System Span.

Fecha: Domingo 6 de Agosto.

Tareas:

- ➢ Se diseñó el plan de iteraciones para mostrar la secuencia que debería seguir la ejecución del programa.
- ➢ Se dividieron las historias de usuario en subtareas más pequeñas y se agregaron al trabajo escrito.
- ➢ Se investigó acerca de qué es el failover, cómo es su funcionamiento y en qué momentos se utiliza.

Fecha: Sábado 12 de Agosto.

Tareas:

- ➢ Se empezó con la investigación de cómo se implementa del esquema cliente-servidor por sockets en C/C++.
- ➢ Se inició con la implementación del cliente-servidor, así como el intento envió de información entre ellos.
- ➢ Se creó el cliente y el servidor, los cuales son capaces de mandarse información entre ellos y se muestra en consola como parte de las pruebas.
- ➢ Se agregó en el trabajo escrito, en la parte de la descripción de las librerías utilizadas, la documentación de las librerías que fueron necesarias en la implementación tanto del cliente y como del servidor.

Fecha: Sábado 19 de Agosto.

Tareas:

- ➢ Se empezó con la creación de una lista simplemente enlazada, donde se usarán nodos para guardar la información que el usuario desee.
- ➢ Se investigaron algunos errores relacionado con el paso de punteros como parámetros.
- ➢ Se implementó un algoritmo de inserción de datos a la lista.
- ➢ Se sobrecargó el operador de salida << para mostrar la salida de los datos dentro del nodo.
- ➢ Se sobrecargó el operador de comparación == para verificar si la información dentro de dos nodos es igual.
- ➢ Se sobrecargó el operador de diferenciación != para verificar si la información dentro de dos nodos es diferente.
- ➢ Se encontró un error a la hora de agregar elementos a la lista enlazada, ya que al unir los elementos, los unía con direcciones de memoria con información que no era la esperada. La solución del problema se detalla en la sección correspondiente.
- ➢ Se instaló Doxygen.
- ➢ Se investigó sobre cómo funciona Doxygen para empezar la documentación del código fuente.
- ➢ Se realizaron varias pruebas con Doxygen para ver su funcionamiento y su estructura cuando se documenta.

Fecha: Domingo 20 de Agosto.

Tareas:

- ➢ Se implementó el método get(int index) para buscar un nodo en una posición específica. Si el índice no se encuentra se devuelve una excepción.
- ➢ Se implementó el método get(string key) para buscar un nodo específico por su llave. Si la llave no existe se devuelve una excepción.
- ➢ Se implementó el método remove(int index) para buscar un nodo en una posición específica y eliminarlo. Si el índice no se encuentra se devuelve una excepción.
- ➢ Se implementó el método remove(string key) para buscar un nodo específico por su llave y eliminarlo. Si la llave no existe se devuelve una excepción.

Fecha: Lunes 21 de Agosto.

Tareas:

- ➢ Se sobrecargó el operador de subíndice [], que recibe un índice entero para buscar en la lista el nodo en esa posición.
- ➢ Se sobrecargó el operador de subíndice [], que recibe una llave para buscar en la lista el nodo que se identifica con esa llave.

- ➢ Se agregó la lista lista enlazada a la parte del servidor como memoria para empezar a intentar mandar las peticiones desde el cliente al servidor.
- ➢ Se logró la comunicación de datos entre el servidor y el cliente por medio de palabras claves para las acciones y se separa con algún carácter especial, poco usado, la información que se necesite usar para dicha acción.
- ➢ Se implementó una pequeña interfaz gráfica, en el cliente, para la probar el envió de la información desde el cliente hasta el servidor.
- ➢ Se agregó, en el trabajo escrito, en la parte de la descripción de las librerías utilizadas, la documentación de las librerías que fueron necesarias en la implementación de la lista enlazada y el nodo.
- ➢ Se documento el código fuente del cliente y del servidor con Doxygen.
- ➢ Se creó un repositorio git y se subieron los primeros archivos del proyecto.

Fecha: Viernes 25 de Agosto.

Tareas:

- ➢ Se implementó en el servidor la interfaz gráfica para ver los elementos que se agregan, eliminan y borran de la memoria.
- ➢ Se le implementó un pequeño caché (modificando la clase LinkedList previamente hecha) al cliente para guardar los cinco elementos más accedidos y así no tener que pedirselos al servidor.
- ➢ Se le agregó a la lista enlazada y al caché, un método que verifica si un nodo existe dentro de ellas por medio de sus llaves.
- ➢ Se le implementó una interfaz gráfica al cliente para poder agregar, obtener y eliminar elementos de la memoria, además de poder observar los elementos que se encuentran en el caché.
- ➢ Se le implementaron los algoritmos necesarios para que el cliente le pida una acción al servidor (agregar, obtener, eliminar elemento) y este le contesté si la acción es realizable o no. Además, se verifica en agregar y obtener elemento si los nodos están en el caché antes de pedirselos al servidor. Luego se le solicita al servidor que realice dicha acción.

Fecha: Sábado 26 de Agosto.

Tareas:

- ➢ Se realizó la documentación interna del código fuente de las clases nuevas y los métodos nuevos agregados a las clases ya existentes.
- ➢ Se agregó a la documentación, la descripción de las nuevas librerías utilizadas en el proyecto.
- ➢ Se revisó el código para eliminar cualquier tipo de mala práctica o espacio en memoria que se haya pedido y no se haya liberado.

- ➢ Actualizó el repositorio Git con los nuevos cambios realizados y las versión actual del proyecto.

Fecha: Domingo 27 de Agosto.

Tareas:

- ➢ Se sobrecargó el operador de asignación = para asignar a un nodo la información que esté guardada en otro.
- ➢ Se le implementó el conteo de referencia de los nodos en la memoria. Se le asigna un conteo inicial de 300 y cada segundo se le resta 1 al conteo del nodo. Cuando el nodo vuelve a ser referenciado, se reinicia el conteo en 300 y se repite el proceso. Si el conteo llega a 100 el nodo se elimina de la memoria.
- ➢ Se implementó un recolector de basura para la memoria el cual siempre está recorriendo la lista enlazada cada segundo y elimina todos los nodos cuyos conteos de referencia hayan llegado a 0. Luego de ello notifica a todos los clientes conectados que ese nodo ya no existe.
- ➢ Se documentaron todos los cambios realizados dentro del código fuente y se actualizaron en el repositorio Git.

Fecha: Jueves 31 de Agosto.

Tareas:

- ➢ Se empezó a investigar cómo convertir las lista enlazadas en listas enlazadas genéricas con para poder trabajarlas de manera más general.

Fecha: Martes 5 de Septiembre.

Tareas:

- ➢ Se le implementó al servidor la funcionalidad de reportar, a todos los clientes, cuando algún elemento es eliminado, ya sea, porque nadie lo ha referenciado o fue eliminado por alguno de los clientes.

Fecha: Domingo 10 de Septiembre.

Tareas:

- ➢ Se eliminó del servidor la funcionalidad de reportar a todos los clientes cuando un elemento es eliminado.
- ➢ Se quitó el caché del servidor del cliente y se movió al servidor activo y pasivo, así como su monitor.
- ➢ El cliente solo quedaron los sockets para la conexión y las funciones para agregar un elemento a la memoria, obtener un elemento de la memoria y eliminar un elemento de la memoria.

- ➢ Se implementaron todas las funcionalidades del Caché para que los elementos fueran buscados ahí antes que en la memoria principal del servidor remoto.
- ➢ Se diseñaron los diagramas de arquitectura y componentes y se agregaron a la documentación.

Fecha: Lunes 18 de Septiembre.

Tareas:

- ➢ Se diseñaron los diagramas de despliegue, de clases y de secuencia y se agregaron a la documentación.
- ➢ Se arregló un pequeño error en el servidor en el cual cuando se agrega un nuevo no se cerraba abruptamente.
- ➢ Se modificó la funcionalidad de la instrucción obtener, en la cual primero se busca en el caché si existe el elemento que se busca, de lo contrario, se procede a buscar en la memoria y enviar el elemento para mostrarlo al cliente o enviarle un mensaje indicando que no se encuentra en memoria.
- ➢ Se cambia el constructor del socket cliente para que reciba la IP y el puerto tanto del servidor activo como pasivo.

Fecha: Martes 19 de Septiembre.

Tareas:

- ➢ Se asiste a tutorías con el fin de buscar y arreglar un error en el servidor, en el cual, el mismo, se cierra inesperadamente cuando se le solicita que agregue un nuevo a la memoria, pero para las demás funciones no hay ningún problema encontrado.

Fecha: Viernes 22 de Septiembre.

Tareas:

- ➢ Se encuentra el error del Servidor, el cual es un puntero que se ha eliminado y después de eso seguí utilizando.
- ➢ Se documenta el error correspondiente y se agrega en los anexos la foto del error en la consola.
- ➢ Se finaliza la documentación externa con la completación de la introducción, las conclusiones, los anexos, los algoritmos desarrollados y los problemas encontrados.

Diseño

Diagrama de arquitectura

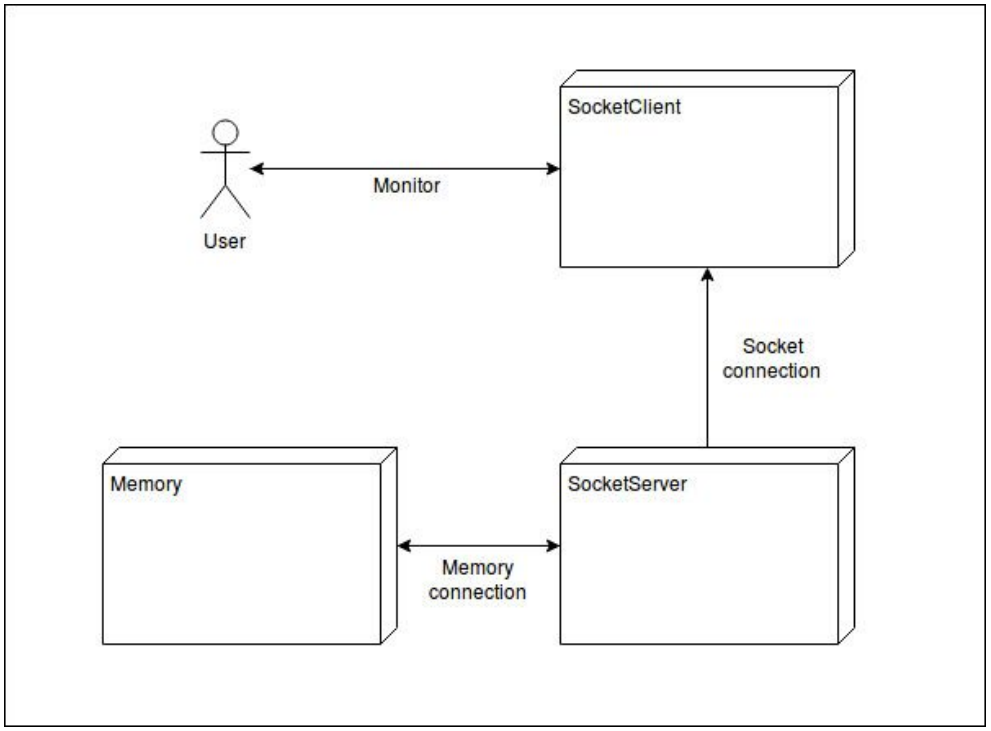


Diagrama de componentes

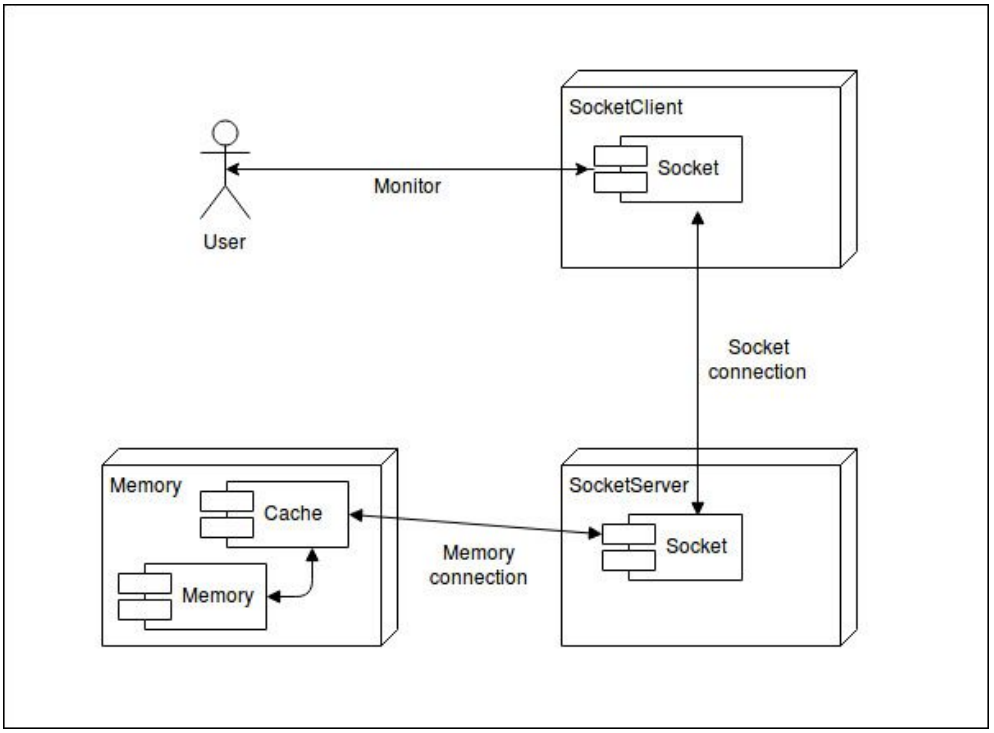


Diagrama de despliegue

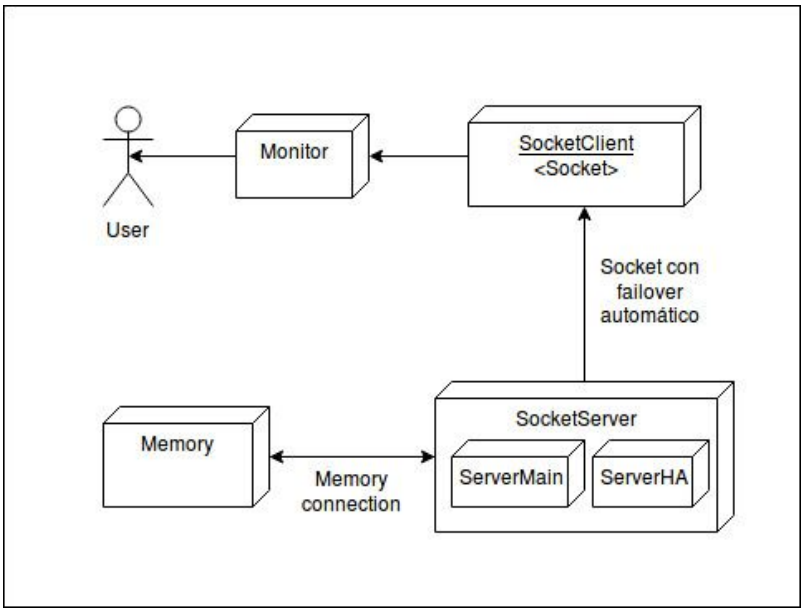


Diagrama de clases

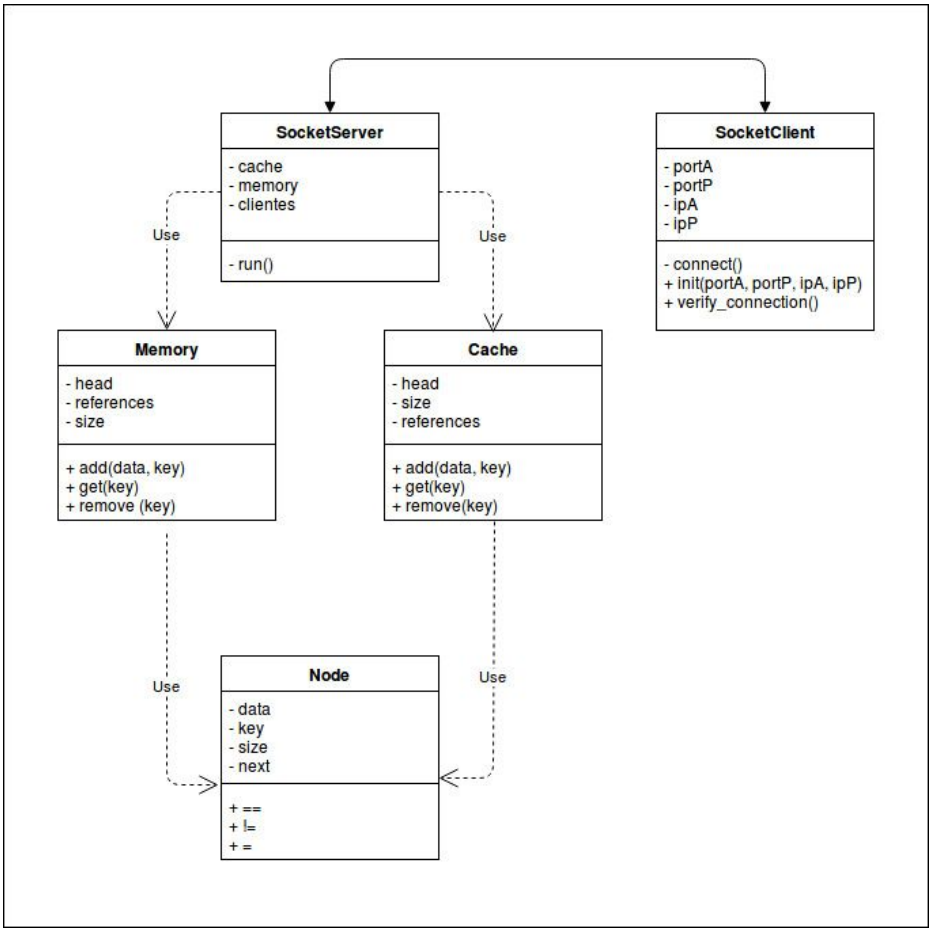
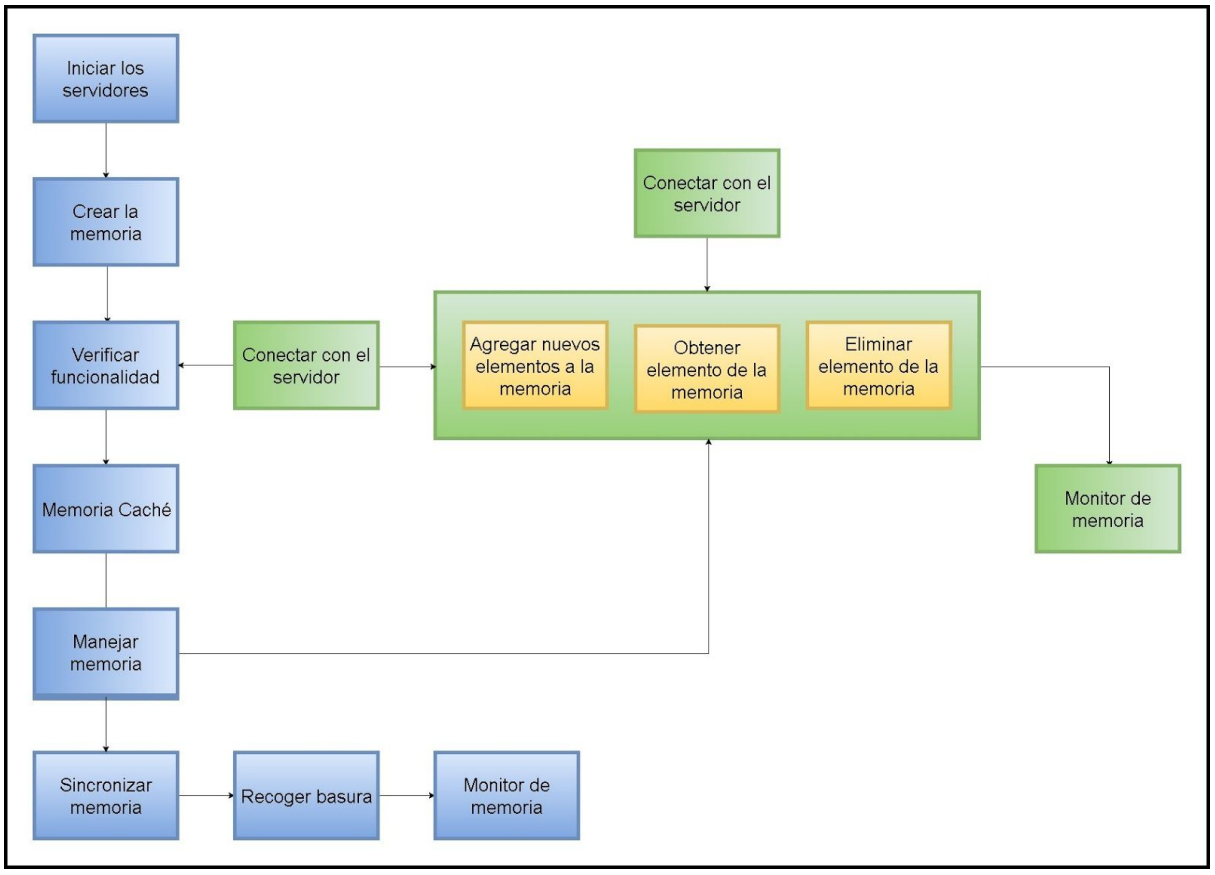


Diagrama de secuencia



Implementación

Descripción de las bibliotecas utilizadas

<arpa/inet.h> (Internet operations): el header pone a disposición los tipos *in_port_t* (un tipo entero sin signo de exactamente 16 bits) y *in_addr_t* (un tipo entero sin signo de exactamente 32 bits). El header pone a disposición la estructura *in_addr* (incluye al menos *in_addr_t* y *s_addr*). Cuando se incluye el header <arpa/inet.h> deberían hacerse visibles todos los símbolos de <*netinet/in.h*> y <*inttypes.h*>.
<chrono> (Time library): es el nombre de un header pero también de un sub-namespace: todos los elementos de este header (excepto para las especializaciones de tipo común) no se definen directamente bajo el namespace std, pero sí bajo el namespace std::chrono. Los elementos de esta cabecera se relacionan con el tiempo. Esto se hace principalmente por medio de tres conceptos; duraciones: miden intervalos de tiempo (1 min, 2 h o 10 ms), puntos de tiempo: una referencia a un punto específico en el tiempo (cumpleaños, el amanecer o cuando pasa el siguiente tren), relojes: un marco que relaciona un punto de tiempo con el tiempo físico real.
<iostream> (Standard Input / Output Streams): header que define los estándares de entrada/salida por pantalla de los datos. Incluyendo esta librería, automáticamente se incluye también <*ios*>, <*streambuf*>, <*istream*>, <*ostream*> y <*iosfwd*>.
<netdb.h> (Network Database Operations): el header puede definir los tipos *in_port_t* y *in_addr_t* como se describe en <*netinet/in.h*>. El header definirá la estructura del host.
<pthread.h> (Threads): el header permite la utilización de símbolos y funciones para las operaciones multi tareas. Incluyendo el header <*pthread.h*> hará que estén visibles símbolos definidos en los headers <*sched.h*> y <*time.h*>. Los tipos *pthread_attr_t*, *pthread_cond_t*, *pthread_condattr_t*, *pthread_key_t*, *pthread_mutex_t*, *pthread_mutexattr_t*, *pthread_once_t*, *pthread_rwlock_t*, *pthread_rwlockattr_t* and *pthread_t* son definidos como se describe en <*sys/types.h*>.
<QApplication>: la clase QApplication gestiona el flujo de control y la configuración principal de la aplicación GUI. QApplication especializa QGuiApplication con algunas funcionalidades necesarias para las aplicaciones basadas en QWidget. Se encarga de la inicialización del widget específico y su finalización.
<QInputDialog>: la clase QInputDialog proporciona un diálogo de conveniencia simple para obtener un único valor del usuario. El valor de entrada puede ser una cadena, un número o un elemento de una lista. Se debe establecer una etiqueta para indicar al usuario qué es lo que debe introducir.
<QMainWindow>: una ventana principal que proporciona un marco para crear la interfaz de usuario de una aplicación. Qt tiene QMainWindow y sus clases relacionadas para la administración de ventanas principales. QMainWindow tiene su propio diseño al que puede

agregar QToolBars, QDockWidgets, QMenuBar y QStatusBar. El diseño tiene un área central que puede ser ocupada por cualquier tipo de widget.
<QMessageBox>: la clase QMessageBox proporciona un cuadro de diálogo modal para informar o hacerle una pregunta al usuario y recibir una respuesta del mismo. Un cuadro de mensaje muestra un texto principal para alertar al usuario de una situación, un texto informativo para explicar mejor la alerta o para hacerle una pregunta al usuario y un texto detallado opcional para proporcionar aún más datos si el usuario lo solicita. Un cuadro de mensaje también puede mostrar un icono y botones estándares para aceptar una posible respuesta del usuario.
<QObject>: la clase QObject es la clase base de todos los objetos Qt. QObject es el corazón del modelo de objetos Qt. La característica central de este modelo es un mecanismo muy potente para la comunicación transparente de objetos llamados señales y ranuras. Puede conectar una señal a una ranura con connect () y destruir la conexión con disconnect(). Los QObjects se organizan en árboles de objetos.
<QString>: la clase QString proporciona una cadena de caracteres Unicode. QString almacena una cadena de QChars de 16 bits, donde cada QChar corresponde un carácter Unicode 4.0. QString utiliza implicit sharing (copy-on-write) para reducir el uso de memoria y evitar la copia innecesaria de datos. Esto también ayuda a reducir la sobrecarga inherente de almacenar caracteres de 16 bits en lugar de caracteres de 8 bits.
<stdexcept>: este header define un conjunto de excepciones estándar que la biblioteca y los programas pueden utilizar para informar de errores comunes.
<string>: la clase de string estándar proporciona soporte para dichos objetos con una interfaz similar a la de un contenedor estándar de bytes, pero añadiendo funciones diseñadas específicamente para operar con cadenas de caracteres de un solo byte. La clase de cadena es una instancia de la plantilla de clase basic_string que usa char (es decir, bytes) como su tipo de carácter, con sus tipos de char_traits y allocator por defecto.
<string.h> (cstring.h): este header define múltiples funciones para manipular cadenas y arrays en C.
<sys/socket.h> (Internet Protocol family): el header pone a disposición un tipo, socklen_t, que es un tipo entero opaco sin signo de longitud de al menos 32 bits. Para evitar problemas de portabilidad, se recomienda que las aplicaciones no utilicen valores superiores a 232 - 1. El header define el tipo *integral unsigned sa_family_t*.
<sys/types.h> (Data Types): el header incluye algunas definiciones para algunos tipos de datos. Todos los tipos se definen como tipos aritméticos de una longitud apropiada, con las siguientes excepciones: *key_t, pthread_attr_t*, *pthread_cond_t*, *pthread_condattr_t*, *pthread_key_t*, *pthread_mutex_t*, *pthread_mutexattr_t*, *pthread_once_t*, *pthread_rwlock_t* y *pthread_rwlockattr_t*. No existen operadores de comparación o de asignación definidos para los tipos *pthread_attr_t*, *pthread_cond_t*, *pthread_condattr_t*, *pthread_mutex_t*, *pthread_mutexattr_t*, *pthread_rwlock_t* y *pthread_rwlockattr_t*.
<thread>: este header declara la clase thread y el namespace this_thread. La clase Thread representa los hilos individuales de ejecución. Un thread es una secuencia de instrucciones

que se pueden ejecutar simultáneamente con otras secuencias de este tipo en entornos multithread, compartiendo un mismo espacio de direcciones.
<unistd.h> (Standard Symbolic Constants and Types): el header define diversas constantes y tipos simbólicos y declara diversas funciones.
<vector>: header que define el contenedor de la clase vector. Los vectores son contenedores de secuencias representando arreglos que pueden cambiar de tamaño. Al igual que los arrays, los vectores usan ubicaciones de almacenamiento contiguas para sus elementos, lo que significa que también se puede acceder a sus elementos usando desplazamientos en punteros regulares a sus elementos, y con la misma eficacia que en arrays. Pero a diferencia de los arrays, su tamaño puede cambiar dinámicamente, con su almacenamiento que se maneja automáticamente por el contenedor.

Descripción de las estructuras de datos desarrolladas

Lista Simplemente Enlazada:

Se utilizó una lista simplemente enlazada en la implementación tanto de la memoria dentro del servidor como del caché dentro del cliente. Para ello se creó una clase Node con los atributos key (como identificador), data (puntero a donde se almacena la información), size (tamaño en memoria de la información) y next (puntero al siguiente elemento). El constructor recibe como parámetro el key y la data y, a partir de eso, setea el key, pide memoria del tamaño de la data y la setea, finalmente declara el puntero next como NULL. Se implementaron los correspondientes getters y setters, además de la sobrecarga de los operadores de asignación operator = (const Node &node) comparación operator == (const Node &node), diferenciación operator != (const Node &node) y salida &operator << (ostream &output, const Node &node).

La lista enlazada simple, contiene como atributos un head (puntero al primer elemento) y un size (tamaño de la lista) en el caso de la memoria. En el caché adicionalmente se le agregó un conteo de las referencias que han habido a ese nodo y una restricción al tamaño (que sea máximo de cinco). En la clase Node del caché, se agregaron los métodos: count_accessed() el cual le suma uno a la cantidad de referencias del nodo y get_accessed() que devuelve la cantidad de accesos de dicho nodo. Se implementaron los métodos: add(string data, string key), exist(string key), get(int index), get(string key), get_size(), remove(int index) y remove(string key), además de una sobrecarga al operador de subíndice operator[](int index) y operator [](string key). En el caché, adicionalmente, se agregó el método remove_unless(), el cual elimina el elemento con menos referencias.

Descripción detallada de los algoritmos desarrollados

Servidor:

El servidor cuenta con un socket el cual siempre está a la escucha de un nuevo cliente que quiera conectarse a él. Una vez aceptada la conexión con el cliente se guarda en una lista de clientes para poder comunicarse con él posteriormente y se crea un nuevo hilo para la comunicación que ese cliente y cuando se desconecta se elimina ese hilo. El servidor queda a la espera de lo que el cliente quiera hacer y es el encargado de gestionar la memoria compartida por distintos usuarios. Si la petición es ejecutada correctamente, el servidor contesta que la opción fue realizada correctamente, de lo contrario, envía un mensaje con el respectivo fallo que ocurrió al tratar de realizar la acción, ya sea que se quiera crear un nuevo elemento en memoria con una llave que ya existe o que se quiera eliminar u obtener un elemento cuya llave no existe.

El servidor también cuenta con un recogedor de basura. Este sirve para eliminar los elemento que llevan mucho tiempo sin ser usados o referenciados, por lo que se consideren basura e innecesarios, por lo que son eliminados automaticamente de la memoria.

Además, los servidores cuentan con failover automático en el caso que el servidor activo falle.

Caché:

Dentro del servidor se tiene un caché para agilizar la búsqueda en la memoria. El caché guarda los cinco últimos elementos, en memoria, buscados o referenciados por los clientes, de esta manera se tienen más a la mano esos elementos y no es necesario ir a buscarlos en la memoria, en la cual pueden haber muchos elementos y se puede tardar mucho tiempo y consumir muchos recursos en buscarlos. Si el elemento definitivamente no se encuentra en el caché, ahí es cuando se busca en la memoria del servidor. Si se requiere agregar un nuevo elemento al caché, se elimina el que cuente con menos referencias de búsqueda entre los cinco elementos que se encuentran almacenados en el caché. Cabe destacar que cada vez que se agrega un nuevo elemento al caché, su cuenta de referencias inicia en cero y cada vez que uno es referencia su cuenta aumenta en uno.

El caché es implementado como una lista simplemente enlazada, solo que esta tiene un tamaño máximo de elemento. Para agregar un elemento se solicitada la llave que se desea que tenga este elemento y el dato a guardar dentro, después de ello el usuario puede eliminar o ver ese elemento. La lista posee un header y un size para llevar el tamaño que tiene esta, además de poseer la sobrecarga de distintos operadores, como el de salida de datos, el de indexación, el de comparación, el de diferenciación, entre otros.

Memoria:

La memoria es elemento principal en el servidor. En ella se guardan todos los elementos que el cliente quiera. La memoria está implementada con una lista simplemente enlazada. Cuando se quiere agregar un elemento, se solicitada la llave que se desea usar como el identificador del elemento y el dato a guardar dentro. La lista posee un header y un size para llevar el tamaño que tiene esta, además de poseer la sobrecarga de distintos operadores, como el de salida de datos, el de indexación, el de comparación, el de diferenciación, entre otros. Se puede tener tres tipos de acciones con la memoria:

- **Agregar:** se solicita una llave y el dato a guardar, la llave se busca a ver si ya existe en la memoria, si ya existe, se levanta una excepción y se le informa al cliente que la llave actualmente ya está en uso. Si la llave no se encontró se agrega el elemento a la lista y se le contesta al cliente con un mensaje diciendo que el proceso se terminó de realizar de manera correcta.
- **Eliminar:** se solicita la llave del elemento que se quiere eliminar, si la llave no existe en la lista entonces se levanta una excepción y se le informa al usuario que la llave no existe en la memoria. Si la llave existiera, se procede a recorrer la lista en busca de la llave y se elimina ese elemento de la memoria y se decrementa en uno el size de la memoria, además de que se envía al cliente un mensaje diciéndole que el proceso se terminó de realizar de manera correcta.
- **Obtener:** para obtener un elemento, se recibe una llave y se verifica si existe en la lista enlazada, si la llave no existe en la lista entonces se levanta una excepción y se le informa al usuario que la llave no existe en la memoria y por lo tanto ningún elemento fue guardado antes con esa llave. Si la llave existiera, se procede a recorrer la lista en

busca de la llave y se le envía un mensaje al cliente con la información contenida en esa llave.

Socket Cliente:

El socket del cliente, es el que se encarga de comunicar la librería usada por el cliente con el servidor y mandar las peticiones solicitadas. Para iniciar el cliente se requieren los IP's y los puertos tanto del servidor activo como del pasivo. Una vez recibidos, se conecta con el servidor activo y si este no se encuentra disponible con el servidor pasivo.

Posee las funcionalidades para el envío de peticiones, se encarga de decirle al servidor que se quiere realizar una acción, ya sea agregar, eliminar u obtener algún elemento de la memoria y también es la que recibe la información del mismo.

Se implementó un socket que es el encargado de enviar las peticiones y otro que simplemente se encarga de escuchar las respuestas del servidor.

Problemas encontrados

1. Al insertar los datos en la lista enlazada, ocurría un error cuando se trataban de unir los nodos.
    1.1. El primer error que se presentó fue que al crear un nuevo nodo el programa se quedaba pegado. Esto ocurría debido a que en el puntero next siempre se creaba un nuevo nodo, por lo que se enciclaba pidiendo memoria. Se solucionó declarando el puntero next en NULL.
    1.2. El segundo error fue que en el puntero que referenciaba al primer elemento en la lista (head), el cual empieza vacío, al insertar el primer elemento no ocurría error alguno, ya que solo se pedía memoria y se asignaba al head. Pero al insertar los siguientes elementos, ocurría un error y era que el programa se quedaba colgado sin tirar ningún error, esto ocurría porque al recorrer la lista, el puntero temporal quedaba apuntando a NULL y a la hora de setear el puntero next, en el programa se queda pegado o mostrar diferentes datos que no eran del programa, por lo que se cambió la forma de recorrer la lista y el problema se solucionó.
2. Se presentó un error a la hora de ejecutar el servidor. Este funcionaba correctamente hasta que se le solicitaba agregar un nuevo elemento a la memoria entonces se salía sin imprimir ningún error. Tiempo después de estar buscando el error, arrojó una excepción que decía “munmap_chunk(): invalid pointer: 0x00007fce3fffee20” (Ver Anexo 1), después de esto, se buscó información sobre este error y se encontró que esto es debido a un puntero que se inicializa correctamente pero, a la hora de utilizarlo, en algún punto del programa este puntero estaba siendo liberado de la memoria y cuando se quería volver a usar no se podía porque esta dirección de memoria ya no pertecía al programa y el puntero había sido borrado.

Conclusiones

- El manejo de la memoria en C/C++ debe realizar cuidadosamente, ya que si no se libera los espacios de memoria correspondiente, se puede empezar a acumular basura que nadie está utilizando y poner lenta la computadora e impedir su correcto funcionamiento.
- En la programación orientada a objetos, los atributos se crean mediante punteros.
- La herencia se puede realizar de distintas maneras y con diferentes formas de accesarlo.
- La sobrecarga de operadores para las clases que han sido creadas desde 0 es muy importante, ya que incluso una salida de datos no está definida.
- El singleton es sumamente necesario en partes del proyecto donde solo se quiera una instancia del mismo, por ejemplo en la memoria y el caché, donde solo se necesitaba que hubiera una instancia de la mismas para que no se estén manejando diferentes memoria y que todos los elementos se manejaran en una sola gran memoria, así mismo en el socket del cliente.

Bibliografía

Save Time and Improve your Marks with CiteThisForMe, The No. 1 Citation Tool. (2017). *Cite This For Me*. Retrieved 5 August 2017, from http://www.citethisforme.com/

Documentos de Google: crea y edita documentos online de forma gratuita.. (2017). Google.com. Retrieved 5 August 2017, from https://www.google.com/intl/es/docs/about/

Hojas de cálculo de Google: crea y edita hojas de cálculo online de forma gratuita.. (2017). Google.com. Retrieved 5 August 2017, from https://www.google.com/intl/es/sheets/about/

Flowchart Maker & Online Diagram Software. (2017). Draw.io. Retrieved 5 August 2017, from https://www.draw.io/

C++ Classes and Objects. (2017). www.tutorialspoint.com. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_classes_objects.htm

C++ Class member functions. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_class_member_functions.htm

C++ Class access modifiers. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_class_access_modifiers.htm

C++ Class Constructor and Destructor. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_constructor_destructor.htm

C++ Copy Constructor. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_copy_constructor.htm

C++ Friend Functions. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_friend_functions.htm

C++ Inline Functions. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_inline_functions.htm

C++ this Pointer. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_this_pointer.htm

Pointer to C++ classes. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_pointer_to_class.htm

Static members of a C++ class. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_static_members.htm

C++ Inheritance. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_inheritance.htm

C++ Overloading (Operator and Function). (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_overloading.htm

Input/Output operators overloading in C++. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/input_output_operators_overloading.htm

Polymorphism in C++. (2017). *www.tutorialspoint.com*. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_polymorphism.htm

C++ Multithreading. (2017). www.tutorialspoint.com. Retrieved 5 August 2017, from https://www.tutorialspoint.com/cplusplus/cpp_multithreading.htm

Programación en C++/Sobrecarga de Operadores - Wikilibros. (2017). *Es.wikibooks.org*. Retrieved 5 August 2017, from https://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C%2B%2B/Sobrecarga_de_Operadores

¿Qué es Failover o conmutación por error? - Definición en WhatIs.com. (2017). *SearchDataCenter en Español*. Retrieved 6 August 2017, from http://searchdatacenter.techtarget.com/es/definicion/Failover-o-conmutacion-por-error

IP failover: un alojamiento sin interrupción - OVH. (2017). *Ovh.es*. Retrieved 6 August 2017, from https://www.ovh.es/servidores_dedicados/ip_failover.xml

Ucaribe (2017). *Diagramación de Arquitecturas en UML - Arquitectura de Objetos Distribuidos*. Claroline.ucaribe.edu.mx. Retrieved 9 August 2017, from https://claroline.ucaribe.edu.mx/claroline/claroline/backends/download.php?url=L1NlbWFuYV8zL0RpYWdyYW1hc19kZV9BcnF1aXRlY3R1cmEucGRm&cidReset=true&cidReq=ARQ2011

Marin, K. (2014). Socket en c++ linux (Chat)P1. YouTube. Retrieved 12 August 2017, from https://www.youtube.com/watch?v=0DdCOwSsFLY

Marin, K. (2014). *Socket en c++ linux (Chat)P2*. *YouTube*. Retrieved 12 August 2017, from https://www.youtube.com/watch?v=mv9a8Gy0LKE

Cruzot, M. (2014). *Listas Enlazadas Simples en C++*. *Gist*. Retrieved 19 August 2017, from https://gist.github.com/martincruzot/16f1c181b38743e06449

High Tech. (2017). La lista enlazada simple. CCM. Retrieved 19 August 2017, from http://es.ccm.net/faq/2842-la-lista-enlazada-simple

Cruz, M. (2017). *Listas Enlazadas Simples Lineales en C++*. *Blog.martincruz.me*. Retrieved 19 August 2017, from http://blog.martincruz.me/2012/10/listas-enlazadas-simples-en-c.html

Yabar, R. (2009). *Listas enlazadas – Clase Lista,Nodo en c++. Imaginemos un mundo libre*. Retrieved 19 August 2017, from https://ronnyml.wordpress.com/2009/07/04/listas-enlazadas-clase-lista-en-c/

Doxygen. (2017). *Doxygen: Main Page*. *Stack.nl*. Retrieved 19 August 2017, from http://www.stack.nl/~dimitri/doxygen/

Cómo instalar. (2017). *Howtoinstall.co*. Retrieved 19 August 2017, from https://www.howtoinstall.co/es/ubuntu/trusty/doxygen

Booleanbite. (2017). *Doxygen una herramienta para documentar código | | Booleanbite*. *Booleanbite.com*. Retrieved 19 August 2017, from http://booleanbite.com/web/doxygen-una-herramienta-para-documentar-codigo/

tutorialspoint. (2017). *Relational operators overloading in C++*. *www.tutorialspoint.com*. Retrieved 19 August 2017, from https://www.tutorialspoint.com/cplusplus/relational_operators_overloading.htm

cplusplus.com. (2017). *string::find - C++ Reference*. *Cplusplus.com*. Retrieved 21 August 2017, from http://www.cplusplus.com/reference/string/string/find/

tutorialspoint. (2017). Subscripting [] operator overloading in C++. www.tutorialspoint.com. Retrieved 21 August 2017, from https://www.tutorialspoint.com/cplusplus/subscripting_operator_overloading.htm

tutorialspoint. (2017). *Assignment Operators Overloading in C++*. *www.tutorialspoint.com*. Retrieved 27 August 2017, from https://www.tutorialspoint.com/cplusplus/assignment_operators_overloading.htm

Oriol, L. (2017). *Conoce la Mejor Aplicación para Hacer Capturas de Pantalla en Linux*. *ComputerNewAge*. Retrieved 27 August 2017, from https://computernewage.com/2015/03/29/conoce-la-mejor-aplicacion-para-hacer-capturas-de-pantalla-en-linux/

<iostream> - C++ Reference. (2017). Cplusplus.com. Retrieved 9 August 2017, from http://www.cplusplus.com/reference/iostream/

<arpa/inet.h>. (2017). Pubs.opengroup.org. Retrieved 12 August 2017, from http://pubs.opengroup.org/onlinepubs/7908799/xns/arpainet.h.html

<netinet/in.h>. (2017). Pubs.opengroup.org. Retrieved 12 August 2017, from http://pubs.opengroup.org/onlinepubs/7908799/xns/netinetin.h.html

<netdb.h>. (2017). Pubs.opengroup.org. Retrieved 12 August 2017, from http://pubs.opengroup.org/onlinepubs/009695399/basedefs/netdb.h.html

<pthread.h>. (2017). Pubs.opengroup.org. Retrieved 12 August 2017, from http://pubs.opengroup.org/onlinepubs/7908799/xsh/pthread.h.html

QApplication Class | Qt Widgets 5.9. (2017). Doc.qt.io. Retrieved 12 August 2017, from http://doc.qt.io/qt-5/qapplication.html#details

QMainWindow Class | Qt 4.8. (2017). Doc.qt.io. Retrieved 12 August 2017, from http://doc.qt.io/qt-4.8/qmainwindow.html#details

QMessageBox Class | Qt 4.8. (2017). Doc.qt.io. Retrieved 12 August 2017, from http://doc.qt.io/qt-4.8/qmessagebox.html

QObject Class | Qt Core 5.9. (2017). Doc.qt.io. Retrieved 12 August 2017, from http://doc.qt.io/qt-5/qobject.html#details

QString Class | Qt Core 5.9. (2017). Doc.qt.io. Retrieved 12 August 2017, from http://doc.qt.io/qt-5/qstring.html

string - C++ Reference. (2017). Cplusplus.com. Retrieved 12 August 2017, from http://www.cplusplus.com/reference/string/string/

<cstring> (string.h) - C++ Reference. (2017). Cplusplus.com. Retrieved 12 August 2017, from http://www.cplusplus.com/reference/cstring/

<sys/socket.h>. (2017). Pubs.opengroup.org. Retrieved 12 August 2017, from http://pubs.opengroup.org/onlinepubs/7908799/xns/syssocket.h.html

<sys/types.h>. (2017). Pubs.opengroup.org. Retrieved 12 August 2017, from http://pubs.opengroup.org/onlinepubs/7908799/xsh/systypes.h.html

vector - C++ Reference. (2017). Cplusplus.com. Retrieved 12 August 2017, from http://www.cplusplus.com/reference/vector/vector/

The Qt Company. (2017). QInputDialog Class | Qt Widgets 5.9. Doc.qt.io. Retrieved 26 August 2017, from http://doc.qt.io/qt-5/qinputdialog.html#details

C++ Reference. (2017). *<stdexcept> - C++ Reference*. *Cplusplus.com*. Retrieved 26 August 2017, from http://www.cplusplus.com/reference/stdexcept/

C++ Reference. (2017). *<vector> - C++ Reference*. *Cplusplus.com*. Retrieved 26 August 2017, from http://www.cplusplus.com/reference/vector/

C++ Reference. (2017). *<thread> - C++ Reference*. *Cplusplus.com*. Retrieved 27 August 2017, from http://www.cplusplus.com/reference/thread/

C++ Reference. (2017). *thread - C++ Reference*. *Cplusplus.com*. Retrieved 27 August 2017, from http://www.cplusplus.com/reference/thread/thread/

C++ Reference. (2017). *this_thread - C++ Reference*. *Cplusplus.com*. Retrieved 27 August 2017, from http://www.cplusplus.com/reference/thread/this_thread/

C++ Reference. (2017). <*chrono*> - *C++ Reference*. *Cplusplus.com*. Retrieved 27 August 2017, from http://www.cplusplus.com/reference/chrono/

Obregón, E. (2017). *ErickOF/Proyecto1_DatosII*. *GitHub*. Retrieved 22 September 2017, from https://github.com/ErickOF/Proyecto1_DatosII

Anexos

Anexo 1

```
*** Error in `/home/erickof/Documentos/Tec/2017 Semestre IV/Algoritmos y Estructura de Datos II/Proyecto1/build-Sockets-Desktop-Debug/Sockets':
 munmap_chunk(): invalid pointer: 0x00007fce3fffee20 ***
======= Backtrace: =========
/lib/x86_64-linux-gnu/libc.so.6(+0x777e5)[0x7fce6b1b47e5]
/lib/x86_64-linux-gnu/libc.so.6(cfree+0x1a8)[0x7fce6b1c1698]
/home/erickof/Documentos/Tec/2017 Semestre IV/Algoritmos y Estructura de Datos II/Proyecto1/build-Sockets-Desktop-Debug/Sockets[0x4051e8]
/home/erickof/Documentos/Tec/2017 Semestre IV/Algoritmos y Estructura de Datos II/Proyecto1/build-Sockets-Desktop-Debug/Sockets[0x404c4b]
/lib/x86_64-linux-gnu/libpthread.so.0(+0x76ba)[0x7fce6baa66ba]
/lib/x86_64-linux-gnu/libc.so.6(clone+0x6d)[0x7fce6b2443dd]
======= Memory map: ========
00400000-0040e000 r-xp 00000000 08:07 24383896                           /home/erickof/Documentos/Tec/2017 Semestre IV/Algoritmos y Estructura
de Datos II/Proyecto1/build-Sockets-Desktop-Debug/Sockets
0060d000-0060e000 r--p 0000d000 08:07 24383896                           /home/erickof/Documentos/Tec/2017 Semestre IV/Algoritmos y Estructura
de Datos II/Proyecto1/build-Sockets-Desktop-Debug/Sockets
0060e000-0060f000 rw-p 0000e000 08:07 24383896                           /home/erickof/Documentos/Tec/2017 Semestre IV/Algoritmos y Estructura
de Datos II/Proyecto1/build-Sockets-Desktop-Debug/Sockets
00f02000-0137d000 rw-p 00000000 00:00 0                                  [heap]
7fce38000000-7fce38022000 rw-p 00000000 00:00 0
7fce38022000-7fce3c000000 ---p 00000000 00:00 0
7fce3d29b000-7fce3d2a6000 r-xp 00000000 08:07 13374543                   /lib/x86_64-linux-gnu/libnss_files-2.23.so
7fce3d2a6000-7fce3d4a5000 ---p 0000b000 08:07 13374543                   /lib/x86_64-linux-gnu/libnss_files-2.23.so
7fce3d4a5000-7fce3d4a6000 r--p 0000a000 08:07 13374543                   /lib/x86_64-linux-gnu/libnss_files-2.23.so
7fce3d4a6000-7fce3d4a7000 rw-p 0000b000 08:07 13374543                   /lib/x86_64-linux-gnu/libnss_files-2.23.so
7fce3d4a7000-7fce3d4ad000 rw-p 00000000 00:00 0
7fce3d4ad000-7fce3d4b8000 r-xp 00000000 08:07 13374546                   /lib/x86_64-linux-gnu/libnss_nis-2.23.so
7fce3d4b8000-7fce3d6b7000 ---p 0000b000 08:07 13374546                   /lib/x86_64-linux-gnu/libnss_nis-2.23.so
7fce3d6b7000-7fce3d6b8000 r--p 0000a000 08:07 13374546                   /lib/x86_64-linux-gnu/libnss_nis-2.23.so
7fce3d6b8000-7fce3d6b9000 rw-p 0000b000 08:07 13374546                   /lib/x86_64-linux-gnu/libnss_nis-2.23.so
7fce3d6b9000-7fce3d6cf000 r-xp 00000000 08:07 13373804                   /lib/x86_64-linux-gnu/libnsl-2.23.so
7fce3d6cf000-7fce3d8ce000 ---p 00016000 08:07 13373804                   /lib/x86_64-linux-gnu/libnsl-2.23.so
7fce3d8ce000-7fce3d8cf000 r--p 00015000 08:07 13373804                   /lib/x86_64-linux-gnu/libnsl-2.23.so
7fce3d8cf000-7fce3d8d0000 rw-p 00016000 08:07 13373804                   /lib/x86_64-linux-gnu/libnsl-2.23.so
7fce3d8d0000-7fce3d8d2000 rw-p 00000000 00:00 0
7fce3d8d2000-7fce3d8da000 r-xp 00000000 08:07 13374552                   /lib/x86_64-linux-gnu/libnss_compat-2.23.so
7fce3d8da000-7fce3dad9000 ---p 00008000 08:07 13374552                   /lib/x86_64-linux-gnu/libnss_compat-2.23.so
7fce3dad9000-7fce3dada000 r--p 00007000 08:07 13374552                   /lib/x86_64-linux-gnu/libnss_compat-2.23.so
7fce3dada000-7fce3dadb000 rw-p 00008000 08:07 13374552                   /lib/x86_64-linux-gnu/libnss_compat-2.23.so
7fce3dadb000-7fce3dadc000 ---p 00000000 00:00 0
7fce3dadc000-7fce3e2dc000 rw-p 00000000 00:00 0
7fce3e2dc000-7fce3e2dd000 ---p 00000000 00:00 0
7fce3e2dd000-7fce3eadd000 rw-p 00000000 00:00 0
7fce3eadd000-7fce3eae5000 r-xp 00000000 08:07 6824667                    /usr/lib/x86_64-linux-gnu/libpciaccess.so.0.11.1
7fce3eae5000-7fce3ece5000 ---p 00008000 08:07 6824667                    /usr/lib/x86_64-linux-gnu/libpciaccess.so.0.11.1
```

```
7fce3ece5000-7fce3ece6000 r--p 00008000 08:07 6824667                    /usr/lib/x86_64-linux-gnu/libpciaccess.so.0.11.1
7fce3ece6000-7fce3ece7000 rw-p 00009000 08:07 6824667                    /usr/lib/x86_64-linux-gnu/libpciaccess.so.0.11.1
7fce3ece7000-7fce3ecf2000 r-xp 00000000 08:07 6824011                    /usr/lib/x86_64-linux-gnu/libdrm_radeon.so.1.0.1
7fce3ecf2000-7fce3eef1000 ---p 0000b000 08:07 6824011                    /usr/lib/x86_64-linux-gnu/libdrm_radeon.so.1.0.1
7fce3eef1000-7fce3eef2000 r--p 0000a000 08:07 6824011                    /usr/lib/x86_64-linux-gnu/libdrm_radeon.so.1.0.1
7fce3eef2000-7fce3eef3000 rw-p 0000b000 08:07 6824011                    /usr/lib/x86_64-linux-gnu/libdrm_radeon.so.1.0.1
7fce3eef3000-7fce3f5ad000 r-xp 00000000 08:07 7081079                    /usr/lib/x86_64-linux-gnu/dri/i965_dri.so
7fce3f5ad000-7fce3f7ad000 ---p 006ba000 08:07 7081079                    /usr/lib/x86_64-linux-gnu/dri/i965_dri.so
7fce3f7ad000-7fce3f7ef000 r--p 006ba000 08:07 7081079                    /usr/lib/x86_64-linux-gnu/dri/i965_dri.so
7fce3f7ef000-7fce3f7f7000 rw-p 006fc000 08:07 7081079                    /usr/lib/x86_64-linux-gnu/dri/i965_dri.so
7fce3f7f7000-7fce3f7ff000 rw-p 00000000 00:00 0
7fce3f7ff000-7fce3f800000 ---p 00000000 00:00 0
7fce3f800000-7fce40000000 rw-p 00000000 00:00 0
7fce40000000-7fce40022000 rw-p 00000000 00:00 0
7fce40022000-7fce44000000 ---p 00000000 00:00 0
7fce44046000-7fce4418f000 rw-s 00000000 00:05 8192050                    /SYSV00000000 (deleted)
7fce4418f000-7fce44196000 r-xp 00000000 08:07 6824015                    /usr/lib/x86_64-linux-gnu/libdrm_nouveau.so.2.0.0
7fce44196000-7fce44395000 ---p 00007000 08:07 6824015                    /usr/lib/x86_64-linux-gnu/libdrm_nouveau.so.2.0.0
7fce44395000-7fce44396000 r--p 00006000 08:07 6824015                    /usr/lib/x86_64-linux-gnu/libdrm_nouveau.so.2.0.0
7fce44396000-7fce44397000 rw-p 00007000 08:07 6824015                    /usr/lib/x86_64-linux-gnu/libdrm_nouveau.so.2.0.0
7fce44397000-7fce443b9000 r-xp 00000000 08:07 6824009                    /usr/lib/x86_64-linux-gnu/libdrm_intel.so.1.0.0
7fce443b9000-7fce445b8000 ---p 00022000 08:07 6824009                    /usr/lib/x86_64-linux-gnu/libdrm_intel.so.1.0.0
7fce445b8000-7fce445b9000 r--p 00021000 08:07 6824009                    /usr/lib/x86_64-linux-gnu/libdrm_intel.so.1.0.0
7fce445b9000-7fce445ba000 rw-p 00022000 08:07 6824009                    /usr/lib/x86_64-linux-gnu/libdrm_intel.so.1.0.0
7fce445ba000-7fce445bb000 ---p 00000000 00:00 0
7fce445bb000-7fce44dbb000 rw-p 00000000 00:00 0
7fce44dbb000-7fce44df1000 r-xp 00000000 08:07 7079746                    /usr/lib/x86_64-linux-gnu/gvfs/libgvfscommon.so
7fce44df1000-7fce44ff1000 ---p 00036000 08:07 7079746                    /usr/lib/x86_64-linux-gnu/gvfs/libgvfscommon.so
7fce44ff1000-7fce44ff6000 r--p 00036000 08:07 7079746                    /usr/lib/x86_64-linux-gnu/gvfs/libgvfscommon.so
7fce44ff6000-7fce44ff7000 rw-p 0003b000 08:07 7079746                    /usr/lib/x86_64-linux-gnu/gvfs/libgvfscommon.so
7fce44ff7000-7fce45027000 r-xp 00000000 08:07 7209420                    /usr/lib/x86_64-linux-gnu/gio/modules/libgvfsdbus.so
7fce45027000-7fce45227000 ---p 00030000 08:07 7209420                    /usr/lib/x86_64-linux-gnu/gio/modules/libgvfsdbus.so
7fce45227000-7fce45228000 r--p 00030000 08:07 7209420                    /usr/lib/x86_64-linux-gnu/gio/modules/libgvfsdbus.so
7fce45228000-7fce4522a000 rw-p 00031000 08:07 7209420                    /usr/lib/x86_64-linux-gnu/gio/modules/libgvfsdbus.so
7fce4522a000-7fce45290000 r-xp 00000000 08:07 6824382                    /usr/lib/x86_64-linux-gnu/libibus-1.0.so.5.0.511
7fce45290000-7fce4548f000 ---p 00066000 08:07 6824382                    /usr/lib/x86_64-linux-gnu/libibus-1.0.so.5.0.511
7fce4548f000-7fce45491000 r--p 00065000 08:07 6824382                    /usr/lib/x86_64-linux-gnu/libibus-1.0.so.5.0.511
7fce45491000-7fce45492000 rw-p 00067000 08:07 6824382                    /usr/lib/x86_64-linux-gnu/libibus-1.0.so.5.0.511
7fce45492000-7fce45493000 rw-p 00000000 00:00 0
7fce45493000-7fce45499000 r-xp 00000000 08:07 7209419                    /usr/lib/x86_64-linux-gnu/gtk-2.0/2.10.0/immodules/im-ibus.so
7fce45499000-7fce45699000 ---p 00006000 08:07 7209419                    /usr/lib/x86_64-linux-gnu/gtk-2.0/2.10.0/immodules/im-ibus.so
7fce45699000-7fce4569a000 r--p 00006000 08:07 7209419                    /usr/lib/x86_64-linux-gnu/gtk-2.0/2.10.0/immodules/im-ibus.so
7fce4569a000-7fce4569b000 rw-p 00007000 08:07 7209419                    /usr/lib/x86_64-linux-gnu/gtk-2.0/2.10.0/immodules/im-ibus.so
```

```
7fce4569b000-7fce4571a000 r-xp 00000000 08:07 6824219                    /usr/lib/x86_64-linux-gnu/libgmp.so.10.3.0
7fce4571a000-7fce45919000 ---p 0007f000 08:07 6824219                    /usr/lib/x86_64-linux-gnu/libgmp.so.10.3.0
7fce45919000-7fce4591a000 r--p 0007e000 08:07 6824219                    /usr/lib/x86_64-linux-gnu/libgmp.so.10.3.0
7fce4591a000-7fce4591b000 rw-p 0007f000 08:07 6824219                    /usr/lib/x86_64-linux-gnu/libgmp.so.10.3.0
7fce4591b000-7fce4594d000 r-xp 00000000 08:07 6824357                    /usr/lib/x86_64-linux-gnu/libhogweed.so.4.2
7fce4594d000-7fce45b4c000 ---p 00032000 08:07 6824357                    /usr/lib/x86_64-linux-gnu/libhogweed.so.4.2
7fce45b4c000-7fce45b4d000 r--p 00031000 08:07 6824357                    /usr/lib/x86_64-linux-gnu/libhogweed.so.4.2
7fce45b4d000-7fce45b4e000 rw-p 00032000 08:07 6824357                    /usr/lib/x86_64-linux-gnu/libhogweed.so.4.2
7fce45b4e000-7fce45b82000 r-xp 00000000 08:07 6824578                    /usr/lib/x86_64-linux-gnu/libnettle.so.6.2
7fce45b82000-7fce45d81000 ---p 00034000 08:07 6824578                    /usr/lib/x86_64-linux-gnu/libnettle.so.6.2
7fce45d81000-7fce45d83000 r--p 00033000 08:07 6824578                    /usr/lib/x86_64-linux-gnu/libnettle.so.6.2
7fce45d83000-7fce45d84000 rw-p 00035000 08:07 6824578                    /usr/lib/x86_64-linux-gnu/libnettle.so.6.2
7fce45d84000-7fce45d95000 r-xp 00000000 08:07 6821491                    /usr/lib/x86_64-linux-gnu/libtasn1.so.6.5.1
7fce45d95000-7fce45f95000 ---p 00011000 08:07 6821491                    /usr/lib/x86_64-linux-gnu/libtasn1.so.6.5.1
7fce45f95000-7fce45f96000 r--p 00011000 08:07 6821491                    /usr/lib/x86_64-linux-gnu/libtasn1.so.6.5.1
7fce45f96000-7fce45f97000 rw-p 00012000 08:07 6821491                    /usr/lib/x86_64-linux-gnu/libtasn1.so.6.5.1
7fce45f97000-7fce45fc8000 r-xp 00000000 08:07 6824406                    /usr/lib/x86_64-linux-gnu/libidn.so.11.6.15
7fce45fc8000-7fce461c8000 ---p 00031000 08:07 6824406                    /usr/lib/x86_64-linux-gnu/libidn.so.11.6.15
7fce461c8000-7fce461c9000 r--p 00031000 08:07 6824406                    /usr/lib/x86_64-linux-gnu/libidn.so.11.6.15
7fce461c9000-7fce461ca000 rw-p 00032000 08:07 6824406                    /usr/lib/x86_64-linux-gnu/libidn.so.11.6.15
7fce461ca000-7fce46223000 r-xp 00000000 08:07 6824641                    /usr/lib/x86_64-linux-gnu/libp11-kit.so.0.1.0
7fce46223000-7fce46422000 ---p 00059000 08:07 6824641                    /usr/lib/x86_64-linux-gnu/libp11-kit.so.0.1.0
7fce46422000-7fce4642c000 r--p 00058000 08:07 6824641                    /usr/lib/x86_64-linux-gnu/libp11-kit.so.0.1.0
7fce4642c000-7fce4642e000 rw-p 00062000 08:07 6824641                    /usr/lib/x86_64-linux-gnu/libp11-kit.so.0.1.0
7fce4642e000-7fce46433000 r-xp 00000000 08:07 6825505                    /usr/lib/x86_64-linux-gnu/libORBitCosNaming-2.so.0.1.0
7fce46433000-7fce46633000 ---p 00005000 08:07 6825505                    /usr/lib/x86_64-linux-gnu/libORBitCosNaming-2.so.0.1.0
7fce46633000-7fce46634000 r--p 00005000 08:07 6825505                    /usr/lib/x86_64-linux-gnu/libORBitCosNaming-2.so.0.1.0
7fce46634000-7fce46635000 rw-p 00006000 08:07 6825505                    /usr/lib/x86_64-linux-gnu/libORBitCosNaming-2.so.0.1.0
7fce46635000-7fce46637000 r-xp 00000000 08:07 13374555                   /lib/x86_64-linux-gnu/libutil-2.23.so
7fce46637000-7fce46836000 ---p 00002000 08:07 13374555                   /lib/x86_64-linux-gnu/libutil-2.23.so
7fce46836000-7fce46837000 r--p 00001000 08:07 13374555                   /lib/x86_64-linux-gnu/libutil-2.23.so
7fce46837000-7fce46838000 rw-p 00002000 08:07 13374555                   /lib/x86_64-linux-gnu/libutil-2.23.so
7fce46838000-7fce46848000 r-xp 00000000 08:07 6823817                    /usr/lib/x86_64-linux-gnu/libavahi-client.so.3.2.9
7fce46848000-7fce46a47000 ---p 00010000 08:07 6823817                    /usr/lib/x86_64-linux-gnu/libavahi-client.so.3.2.9
7fce46a47000-7fce46a48000 r--p 0000f000 08:07 6823817                    /usr/lib/x86_64-linux-gnu/libavahi-client.so.3.2.9
7fce46a48000-7fce46a49000 rw-p 00010000 08:07 6823817                    /usr/lib/x86_64-linux-gnu/libavahi-client.so.3.2.9
7fce46a49000-7fce46a54000 r-xp 00000000 08:07 6823819                    /usr/lib/x86_64-linux-gnu/libavahi-common.so.3.5.3
7fce46a54000-7fce46c53000 ---p 0000b000 08:07 6823819                    /usr/lib/x86_64-linux-gnu/libavahi-common.so.3.5.3
7fce46c53000-7fce46c54000 r--p 0000a000 08:07 6823819                    /usr/lib/x86_64-linux-gnu/libavahi-common.so.3.5.3
7fce46c54000-7fce46c55000 rw-p 0000b000 08:07 6823819                    /usr/lib/x86_64-linux-gnu/libavahi-common.so.3.5.3
7fce46c55000-7fce46c58000 r-xp 00000000 08:07 6823823                    /usr/lib/x86_64-linux-gnu/libavahi-glib.so.1.0.2
7fce46c58000-7fce46e57000 ---p 00003000 08:07 6823823                    /usr/lib/x86_64-linux-gnu/libavahi-glib.so.1.0.2
7fce46e57000-7fce46e58000 r--p 00002000 08:07 6823823                    /usr/lib/x86_64-linux-gnu/libavahi-glib.so.1.0.2
7fce46e58000-7fce46e59000 rw-p 00003000 08:07 6823823                    /usr/lib/x86_64-linux-gnu/libavahi-glib.so.1.0.2
7fce46e59000-7fce46f7c000 r-xp 00000000 08:07 6821495                    /usr/lib/x86_64-linux-gnu/libgnutls.so.30.6.2
7fce46f7c000-7fce4717b000 ---p 00123000 08:07 6821495                    /usr/lib/x86_64-linux-gnu/libgnutls.so.30.6.2
7fce4717b000-7fce47186000 r--p 00122000 08:07 6821495                    /usr/lib/x86_64-linux-gnu/libgnutls.so.30.6.2
7fce47186000-7fce47188000 rw-p 0012d000 08:07 6821495                    /usr/lib/x86_64-linux-gnu/libgnutls.so.30.6.2
7fce47188000-7fce47189000 rw-p 00000000 00:00 0
7fce47189000-7fce471ae000 r-xp 00000000 08:07 6823960                    /usr/lib/x86_64-linux-gnu/libdbus-glib-1.so.2.3.3
7fce471ae000-7fce473ae000 ---p 00025000 08:07 6823960                    /usr/lib/x86_64-linux-gnu/libdbus-glib-1.so.2.3.3
7fce473ae000-7fce473af000 r--p 00025000 08:07 6823960                    /usr/lib/x86_64-linux-gnu/libdbus-glib-1.so.2.3.3
7fce473af000-7fce473b0000 rw-p 00026000 08:07 6823960                    /usr/lib/x86_64-linux-gnu/libdbus-glib-1.so.2.3.3
7fce473b0000-7fce473b1000 r-xp 00000000 08:07 6824314                    /usr/lib/x86_64-linux-gnu/libgthread-2.0.so.0.4800.2
7fce473b1000-7fce475b0000 ---p 00001000 08:07 6824314                    /usr/lib/x86_64-linux-gnu/libgthread-2.0.so.0.4800.2
7fce475b0000-7fce475b1000 r--p 00000000 08:07 6824314                    /usr/lib/x86_64-linux-gnu/libgthread-2.0.so.0.4800.2
7fce475b1000-7fce475b2000 rw-p 00001000 08:07 6824314                    /usr/lib/x86_64-linux-gnu/libgthread-2.0.so.0.4800.2
7fce475b2000-7fce475bd000 r-xp 00000000 08:07 13373992                   /lib/x86_64-linux-gnu/libpopt.so.0.0.0
7fce475bd000-7fce477bc000 ---p 0000b000 08:07 13373992                   /lib/x86_64-linux-gnu/libpopt.so.0.0.0
7fce477bc000-7fce477bd000 r--p 0000a000 08:07 13373992                   /lib/x86_64-linux-gnu/libpopt.so.0.0.0
7fce477bd000-7fce477be000 rw-p 0000b000 08:07 13373992                   /lib/x86_64-linux-gnu/libpopt.so.0.0.0
7fce477be000-7fce4796f000 r-xp 00000000 08:07 6818585                    /usr/lib/x86_64-linux-gnu/libxml2.so.2.9.3
7fce4796f000-7fce47b6e000 ---p 001b1000 08:07 6818585                    /usr/lib/x86_64-linux-gnu/libxml2.so.2.9.3
7fce47b6e000-7fce47b76000 r--p 001b0000 08:07 6818585                    /usr/lib/x86_64-linux-gnu/libxml2.so.2.9.3
7fce47b76000-7fce47b78000 rw-p 001b8000 08:07 6818585                    /usr/lib/x86_64-linux-gnu/libxml2.so.2.9.3
7fce47b78000-7fce47b79000 rw-p 00000000 00:00 0
7fce47b79000-7fce47bd4000 r-xp 00000000 08:07 6825506                    /usr/lib/x86_64-linux-gnu/libORBit-2.so.0.1.0
7fce47bd4000-7fce47dd4000 ---p 0005b000 08:07 6825506                    /usr/lib/x86_64-linux-gnu/libORBit-2.so.0.1.0
7fce47dd4000-7fce47de3000 r--p 0005b000 08:07 6825506                    /usr/lib/x86_64-linux-gnu/libORBit-2.so.0.1.0
7fce47de3000-7fce47de6000 rw-p 0006a000 08:07 6825506                    /usr/lib/x86_64-linux-gnu/libORBit-2.so.0.1.0
7fce47de6000-7fce47dfc000 r-xp 00000000 08:07 6825510                    /usr/lib/x86_64-linux-gnu/libbonobo-activation.so.4.0.0
7fce47dfc000-7fce47ffc000 ---p 00016000 08:07 6825510                    /usr/lib/x86_64-linux-gnu/libbonobo-activation.so.4.0.0
7fce47ffc000-7fce47ffe000 r--p 00016000 08:07 6825510                    /usr/lib/x86_64-linux-gnu/libbonobo-activation.so.4.0.0
7fce47ffe000-7fce48000000 rw-p 00018000 08:07 6825510                    /usr/lib/x86_64-linux-gnu/libbonobo-activation.so.4.0.0
7fce48000000-7fce48021000 rw-p 00000000 00:00 0
7fce48021000-7fce4c000000 ---p 00000000 00:00 0
7fce4c000000-7fce4c022000 rw-p 00000000 00:00 0
7fce4c022000-7fce50000000 ---p 00000000 00:00 0
7fce50000000-7fce50022000 rw-p 00000000 00:00 0
7fce50022000-7fce54000000 ---p 00000000 00:00 0
7fce54023000-7fce54024000 rw-p 00000000 00:00 0
7fce54024000-7fce54084000 rw-s 00000000 00:05 8224819                    /SYSV00000000 (deleted)
7fce54084000-7fce540e3000 r--p 00000000 08:07 7996335                    /usr/share/fonts/truetype/ubuntu-font-family/Ubuntu-RI.ttf
7fce540e3000-7fce54137000 r--p 00000000 08:07 7996202                    /usr/share/fonts/truetype/dejavu/DejaVuSansMono.ttf
7fce54137000-7fce5418e000 r--p 00000000 08:07 7996334                    /usr/share/fonts/truetype/ubuntu-font-family/Ubuntu-R.ttf
7fce5418e000-7fce541ee000 r-xp 00000000 08:07 6825509                    /usr/lib/x86_64-linux-gnu/libbonobo-2.so.0.0.0
7fce541ee000-7fce543ee000 ---p 00060000 08:07 6825509                    /usr/lib/x86_64-linux-gnu/libbonobo-2.so.0.0.0
7fce543ee000-7fce543f3000 r--p 00060000 08:07 6825509                    /usr/lib/x86_64-linux-gnu/libbonobo-2.so.0.0.0
7fce543f3000-7fce543fe000 rw-p 00065000 08:07 6825509                    /usr/lib/x86_64-linux-gnu/libbonobo-2.so.0.0.0
7fce543fe000-7fce54421000 r-xp 00000000 08:07 6824227                    /usr/lib/x86_64-linux-gnu/libgnome-keyring.so.0.2.0
7fce54421000-7fce54621000 ---p 00023000 08:07 6824227                    /usr/lib/x86_64-linux-gnu/libgnome-keyring.so.0.2.0
7fce54621000-7fce54623000 r--p 00023000 08:07 6824227                    /usr/lib/x86_64-linux-gnu/libgnome-keyring.so.0.2.0
7fce54623000-7fce54624000 rw-p 00025000 08:07 6824227                    /usr/lib/x86_64-linux-gnu/libgnome-keyring.so.0.2.0
7fce54624000-7fce54652000 r-xp 00000000 08:07 6824157                    /usr/lib/x86_64-linux-gnu/libgconf-2.so.4.1.5
7fce54652000-7fce54851000 ---p 0002e000 08:07 6824157                    /usr/lib/x86_64-linux-gnu/libgconf-2.so.4.1.5
7fce54851000-7fce54852000 r--p 0002d000 08:07 6824157                    /usr/lib/x86_64-linux-gnu/libgconf-2.so.4.1.5
7fce54852000-7fce54853000 rw-p 0002e000 08:07 6824157                    /usr/lib/x86_64-linux-gnu/libgconf-2.so.4.1.5
7fce54853000-7fce548b7000 r-xp 00000000 08:07 6825520                    /usr/lib/x86_64-linux-gnu/libgnomevfs-2.so.0.2400.4
7fce548b7000-7fce54ab6000 ---p 00064000 08:07 6825520                    /usr/lib/x86_64-linux-gnu/libgnomevfs-2.so.0.2400.4
7fce54ab6000-7fce54ab9000 r--p 00063000 08:07 6825520                    /usr/lib/x86_64-linux-gnu/libgnomevfs-2.so.0.2400.4
7fce54ab9000-7fce54abb000 rw-p 00066000 08:07 6825520                    /usr/lib/x86_64-linux-gnu/libgnomevfs-2.so.0.2400.4
7fce54abb000-7fce54abc000 rw-p 00000000 00:00 0
7fce54abc000-7fce54ad2000 r-xp 00000000 08:07 6825522                    /usr/lib/x86_64-linux-gnu/libgnome-2.so.0.3200.1
7fce54ad2000-7fce54cd1000 ---p 00016000 08:07 6825522                    /usr/lib/x86_64-linux-gnu/libgnome-2.so.0.3200.1
7fce54cd1000-7fce54cd2000 r--p 00015000 08:07 6825522                    /usr/lib/x86_64-linux-gnu/libgnome-2.so.0.3200.1
7fce54cd2000-7fce54cd3000 rw-p 00016000 08:07 6825522                    /usr/lib/x86_64-linux-gnu/libgnome-2.so.0.3200.1
7fce54cd3000-7fce54ceb000 r-xp 00000000 08:07 6823787                    /usr/lib/x86_64-linux-gnu/libart_lgpl_2.so.2.3.21
7fce54ceb000-7fce54eea000 ---p 00018000 08:07 6823787                    /usr/lib/x86_64-linux-gnu/libart_lgpl_2.so.2.3.21
7fce54eea000-7fce54eeb000 r--p 00017000 08:07 6823787                    /usr/lib/x86_64-linux-gnu/libart_lgpl_2.so.2.3.21
7fce54eeb000-7fce54eec000 rw-p 00018000 08:07 6823787                    /usr/lib/x86_64-linux-gnu/libart_lgpl_2.so.2.3.21
7fce54eec000-7fce54f1f000 r-xp 00000000 08:07 6825517                    /usr/lib/x86_64-linux-gnu/libgnomecanvas-2.so.0.3000.3
7fce54f1f000-7fce5511f000 ---p 00033000 08:07 6825517                    /usr/lib/x86_64-linux-gnu/libgnomecanvas-2.so.0.3000.3456789Press <RET
URN> to close this window...
```