TEC | Tecnológico de Costa Rica

Instituto Tecnológico de Costa Rica

IC-4700 Lenguajes de Programación
I-2025

Proyecto 1: Sistema de Mensajería C++

Paradigma Imperativo

Integrantes:

- Daniel Alemán
- Luis Meza
- Daniel Zeas

Índice

Enlace de GitHub

Repositorio del Proyecto

Descripción del Proyecto

Este proyecto consiste en un sistema de mensajería cliente-servidor desarrollado en C++ para Linux. El servidor central gestiona los usuarios y los mensajes, mientras que los clientes pueden registrarse, enviar mensajes privados o globales, y recibir mensajes en tiempo real.

Características principales:

- Registro automático de usuarios con IP y nombre.
- Envío de mensajes privados (`/msg`) y globales (`/broadcast`).
- Recepción de mensajes en tiempo real mediante procesos bifurcados (`fork()`).
- Persistencia de mensajes pendientes para usuarios desconectados.

Requisitos

- **Sistema Operativo:** Linux
- **Compilador:** `g++` (C++11 o superior)
- **Librerías:**
  - `sys/socket.h`
  - `arpa/inet.h`
  - `unistd.h`
  - `csignal`
  - `sys/mman.h` (para memoria compartida)

Instalación

1. **Clonar el repositorio:**

```
git clone https://github.com/DanielAR27/Proyecto1-Lenguajes.git
cd  Proyecto1-Lenguajes
```

2. **Compilar el proyecto:**

```
g++ servidor_mensajeria.cpp -o servidor -lpthread
g++ cliente.cpp -o cliente
```

3. **Configurar el servidor:**
   Editar el archivo `config.txt` para cambiar el puerto (por defecto: `5050`).
4. **Ejecutar:**

- Servidor:

```
./servidor
```

- Cliente:

```
./cliente <direccion_servidor> <puerto>
```

Ejemplo:

```
./cliente 127.0.0.1 5050
```

Manual de Usuario

Comandos Disponibles (Cliente):

Comando	Descripción
`/broadcast <mensaje>`	Envía un mensaje a todos los usuarios.
`/msg <usuario> <texto>`	Envía un mensaje privado a un usuario.
`/listar`	Muestra la lista de usuarios conectados.
`/ayuda`	Muestra la lista de comandos disponibles.
`/salir`	Cierra la sesión del usuario.

Ejemplo de Uso:

1. **Registro:**
   Al iniciar el cliente, se solicita un nombre de usuario (sin espacios).
2. **Enviar mensaje global:**

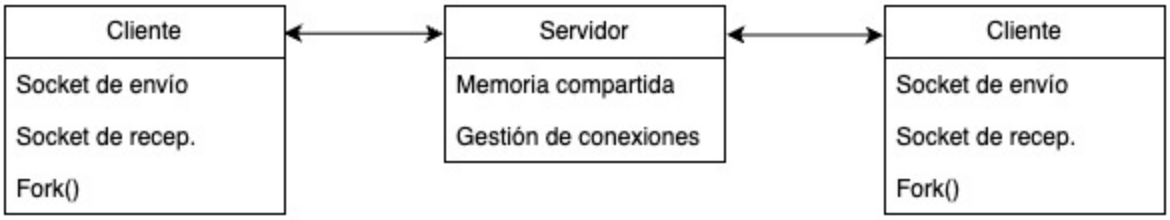
```
/broadcast Hola a todos!
```

3. **Enviar mensaje privado:**

```
/msg usuario2 ¿Cómo estás?
```

Arquitectura Lógica

Diagrama de Componentes:



Estructuras Clave:

- **Servidor:**
  - Usa `select()` para manejar múltiples conexiones.
  - Memoria compartida (`shm_open`) para usuarios y mensajes pendientes.
- **Cliente:**
  - Proceso padre para enviar mensajes.
  - Proceso hijo (`fork()`) para recibir mensajes.

Funcionamiento

1. **Registro de Usuarios:**
   - El cliente envía su nombre al servidor, que lo valida y registra.
2. **Envío de Mensajes:**
   - Los mensajes pasan por el servidor, que los redirige al destinatario o los almacena si está desconectado.
3. **Recepción:**
   - Cada cliente usa un proceso hijo para escuchar mensajes entrantes.
4. **Limpieza:**
   - El servidor elimina usuarios inactivos después de 5 minutos.