

Instituto Tecnológico  
de Costa Rica.

Administración de  
Tecnología de  
Información.

Lenguajes de  
Programación.

Profesor:

Andrei Fuentes.

Tutor/Asistente:

Evelyn Madríz.

Brayan Blanco M.  
Christian Chaves M.  
Cristopher Vásquez V.

---

**Tabla de contenido**

Objetivos del Proyecto ..... 2

- Objetivo General ..... 2
- Objetivos Específicos ..... 2

Descripción del Problema ..... 2

- Problema que se presenta..... 2
- Solución del Problema ..... 3

Descripción del Programa..... 3

Diseño del Programa ..... 5

- Decisiones de Diseño..... 5
- Diagrama Lógico..... 6
- Librerías Utilizadas..... 6

Análisis de Resultados..... 7

- Objetivos Alcanzados..... 7
- Objetivos No Alcanzados ..... 7

Manual de Usuario ..... 7

Conclusión Personal..... 9

## Objetivos del Proyecto

### Objetivo General

Ejecutar un programa que permita la interacción entre usuarios registrados mediante envío y recepción de mensajes.

### Objetivos Específicos

- Investigar funcionalidades específicas y necesarias para un chat.
- Elaborar un servidor central que permita el registro de los usuarios.
- Confeccionar un método que admita tanto el envío como el recibimiento de mensajes de texto o archivos de manera simultánea.

## Descripción del Problema

### Problema que se presenta

El objetivo es lograr la creación de un programa de mensajería similar a Google Chat, o a Facebook Messenger, que permita enviar tanto mensajes como archivos con cualquier extensión y con un tamaño predefinido por el programador, utilizando el lenguaje C en el sistema operativo Linux (Ubuntu).

El Messenger debe cumplir con las características principales de cualquier chat que permita la interacción entre dos usuarios o más, algunas de ellas son:

- Enviar mensajes instantáneos.
- Intercambiar archivos.
- Mantener una comunicación simultánea con varios usuarios a la vez.
- Permitir una interacción fluida mediante texto síncrono.

Para ello es necesario un servidor central que almacene toda la información de los usuarios, además será el encargado de recibir el mensaje enviado por el usuario remitente antes que el destinatario lo reciba.

Además para su completo entendimiento es necesario conocer los siguientes conceptos aplicados para la realización del chat:

- IP
- Puerto
- Socket
- Fork()
- Programación de Múltiples Procesos
- Manejo de archivos
- Modelo TCP/IP

## **Solución del Problema**

- Inicialmente se debe plantear un proceso que permita conocer los requisitos básicos del programa y sus deficiencias, como es en el caso del envío y recibo de archivos donde se deben usar sockets. Esto procede a una investigación de cómo usarlos, cuáles requisitos necesitarán y cuáles librerías deberán ser incluidas para su funcionamiento.
- A partir de los sockets se debe crear un programa que pueda enviar los archivos y que pueda recibirlos, además de que pueda realizar las dos funciones al mismo tiempo, para esto será necesario usar la función fork (), la cual permite dividir un proceso en dos mediante una bifurcación del programa, uno que se encargue de enviar y otro que se encargue de recibir.

## **Descripción del Programa**

- **Front End:**

El programa es un chat que permitirá a los usuarios que lo utilicen interactuar entre sí, mediante mensajes de texto o envío de archivos.

En el cual la aplicación solicita como primera instancia el registro del usuario aportando ciertos datos que le permitan el acceso al programa cuando lo desee utilizar nuevamente.

Al finalizar el registro el usuario podrá utilizar el chat, para ello es necesario que se encuentren al menos dos usuarios conectados para poder interactuar.

El interactúo entre ambos usuarios se pueden dar mediante envío de texto que el mismo chat permitirá redactar, o bien, al enviar archivos.

El funcionamiento del chat se centra que cuando el usuario A (remitente) envíe un mensaje al usuario B (destinatario), este último lo reciba inmediatamente o durante un tiempo razonable; de igual manera si el usuario destinatario se encuentra desconectado al momento que el usuario remitente envía el mensaje éste no se pierda, por lo contrario que el mensaje sea guardado hasta que el usuario destinatario se conecte y así pueda leer el mensaje. Lo mismo sucederá si el mensaje enviado es en sí un archivo.

- **Back End:**

Se refiere al control, gestión y administración que se utiliza para implementar el sistema.

El programa consiste en el envío y el recibimiento de mensajes de texto y/o archivos que permite la interacción entre los usuarios registrados.

Sin embargo enviar mensajes no se efectuará directamente de un usuario a otro, sino que el mensaje primeramente debe ser transferido a un servidor donde se encuentra los datos de todos los usuarios registrados, y así el servidor se encarga de enviar el mensaje al destinatario que el usuario remitente seleccionó.

Algunas de sus funcionalidades consisten en:

- El registro de los usuarios en el servidor central mediante una IP.
- El envío de mensajes entre contactos, pasando primeramente por el servidor.
- Envío y recepción de mensajes simultáneamente.
- Descifrado de los mensajes y archivos enviados de forma cifrada.
- Diferenciación entre mensajes enviados y recibidos mediante colores distintos que permitan su clara visualización.
- En caso que el mensaje enviado corresponda a un archivo, el programa acepta y comprende la ruta del mismo.

## Diseño del Programa

### Decisiones de Diseño

#### ❖ Editores de Texto Utilizados:

Se decidió usar como herramientas de programación los editores de texto “Geanny” (un software que soporta el desarrollo de varios lenguajes en Linux tales como Python, java, C++ entre otros) y “Gedit” (el cual enfatiza la simplicidad y facilidad de uso incluyendo herramientas para la edición de código fuente y textos estructurados, como lenguajes de marcado).

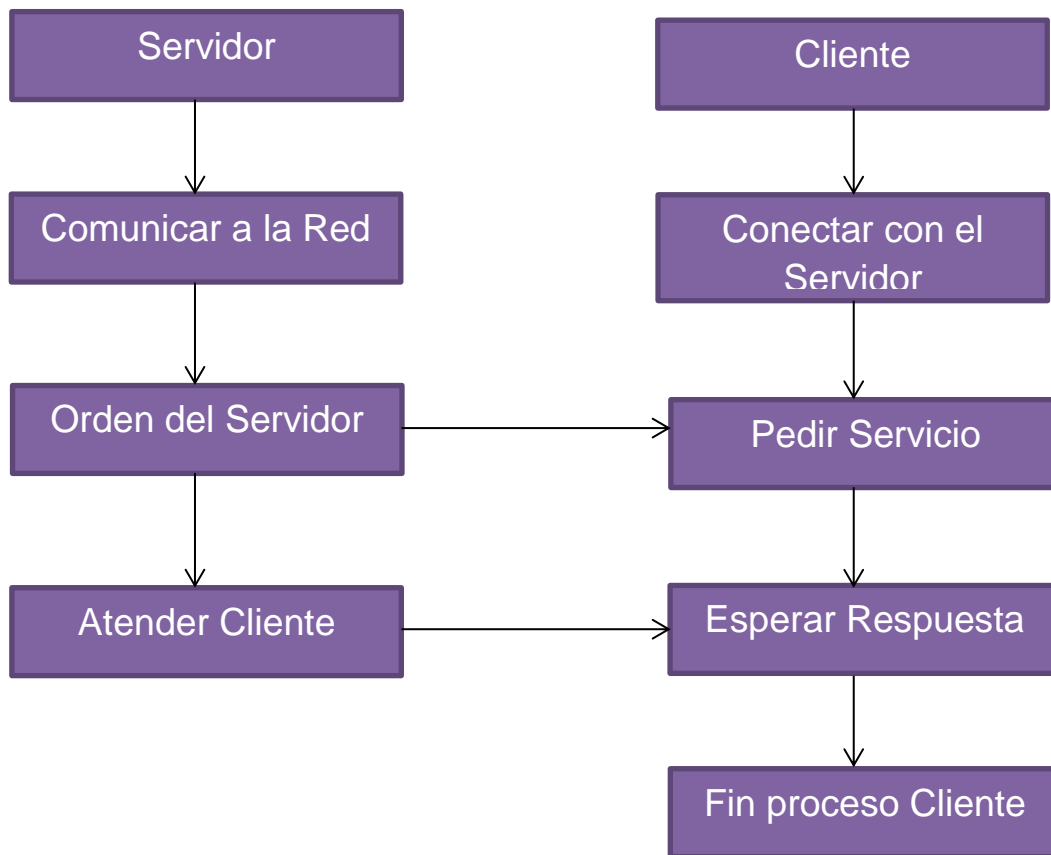
#### ❖ Distribución del Trabajo:

Para el diseño del programa y aprovechamiento del tiempo, se decidió repartir las funciones entre los miembros del equipo. Donde uno se encargó de la investigación e implementación de envío y recepción de archivos, otro se encargó del método de los sockets, y el tercer miembro investigó sobre la función fork() y el funcionamiento del servidor central.

#### ❖ Funcionalidades:

1. **Registrar Usuario:** los usuarios deben registrarse en el servidor central para poder acceder al programa, para ello el servidor debe obtener la IP del usuario automáticamente y guardarla junto a los otros datos correspondientes a cada usuario registrado.
2. **Envío y recepción de mensajes:** se utilizó el protocolo IP para la creación de los sockets. Para permitir el envío y recepción de mensajes de forma simultánea hay que manejar dos sockets mediante la función fork() que se encargará de dividir el programa en dos procesos facilitando dicho objetivo.
3. **Enviar mensajes:** el programa permitirá a los usuarios enviarle mensajes a alguno de sus contactos, para lo cual se especificará el nombre de usuario y el mensaje, y el programa deberá enviar el mensaje al puerto y a la IP asociados a dicho usuario.
4. **Recibir mensajes:** el programa deberá estar en algún puerto para poder recibir mensajes. El puerto que usará el programa deberá estar especificado en un archivo de configuración, para que el usuario no deba digitarlo.
5. **Envío y recepción de archivos:** igualmente que con el envío y recepción de mensajes, para enviar o recibir archivos fue necesario el uso del protocolo IP para la creación de sockets. Dichos archivos deben ser enviados de forma cifrada.

## Diagrama Lógico



## Librerías Utilizadas

- **#include <sys/types.h>** utilizada para la creación de sockets y salidas del sistema con exit().
- **#include <sys/socket.h>** utilizada para la creación de sockets.
- **#include <stdio.h>** utilizada para las entradas y salidas.
- **#include <unistd.h>** utilizada para el uso de fork().
- **#include <netinet/in.h>** necesaria para la implementación de la función. bind().
- **#include <netdb.h>** utilizada para el manejo de errores.
- **#include <stdlib.h>** necesaria para las salidas del sistema con exit().
- **#include <string.h>** utilizada para el manejo de chars como strings.
- **#include <ctype.h>** utilizada para el manejo de chars.
- **#include <signal.h>** utilizada para kill de un proceso.
- **#include <arpa/inet.h>** utilizada para definiciones de operaciones de internet.

## Análisis de Resultados

### Objetivos Alcanzados

- Manejo de sockets.
- Creación del servidor central.
- Registro del usuario.
- Envío y recepción de mensajes.

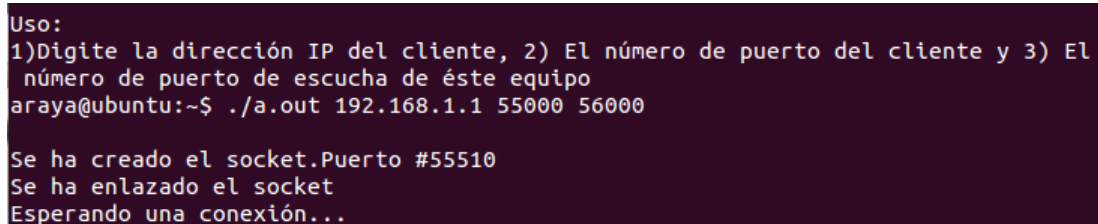
### Objetivos No Alcanzados

- Envío y recepción de archivos: por falta de conocimiento e ignorancia del error que el programa nos informada, nos fue imposible lograr el envío el de archivos a través del chat.

## Manual de Usuario

1. Iniciar el sistema operativo Linux.
2. Abrir la terminal.
3. Se busca la dirección del programa.
4. Para utilizar el chat se debe compilar el código.
5. Se debe ejecutar el archivo, el programa solicitará que se ingrese el número del puerto por el cual se va a dar el envío de los mensajes.
6. Seguidamente se solicitará la inserción de al menos un contacto, debe de ingresar: nombre del contacto, dirección IP (Ej: 192.168.33.46) y puerto.
7. Nuevamente se debe volver a correr el programa para permitir la creación del archivo de contactos.
8. El sistema mostrará un menú principal donde se debe seleccionar una de las opciones que se presentan.
9. Escribiremos el nombre del archivo que deseamos enviar.
10. Enviamos el archivo.

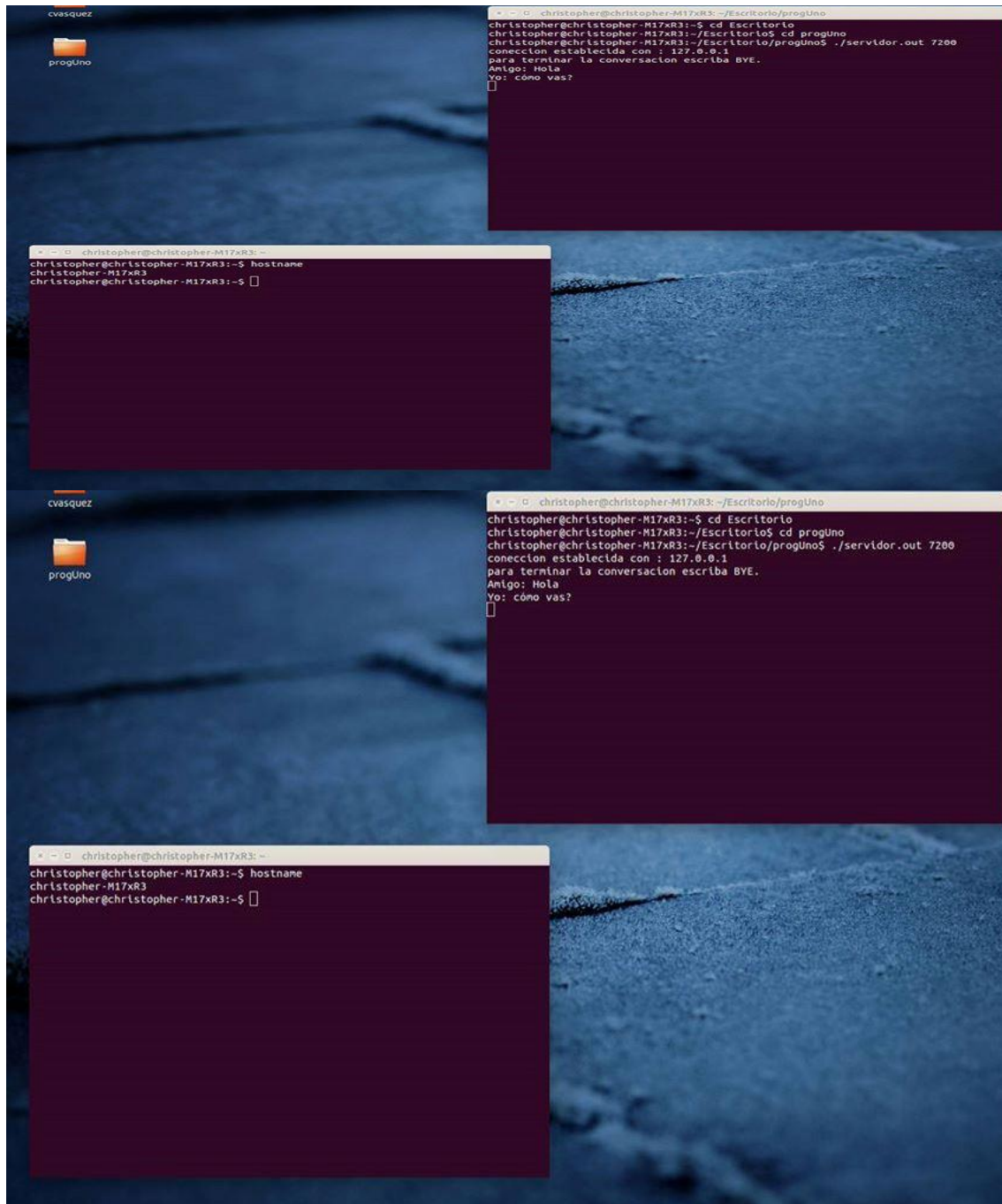
A continuación se adjunta unas imágenes sobre cómo se verá el chat al utilizarlo.

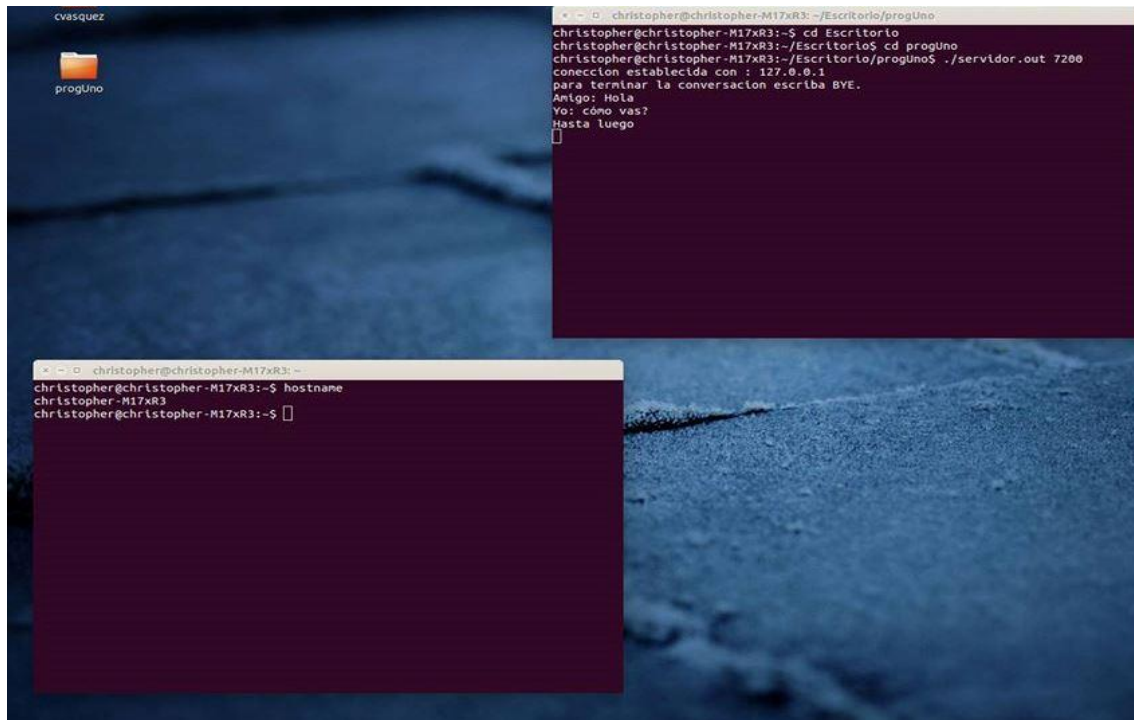


```
Uso:
1) Digite la dirección IP del cliente, 2) El número de puerto del cliente y 3) El
número de puerto de escucha de éste equipo
araya@ubuntu:~$ ./a.out 192.168.1.1 55000 56000

Se ha creado el socket. Puerto #55510
Se ha enlazado el socket
Esperando una conexión...
█
```







## Conclusión Personal

El objetivo de la tarea programada es bastante interesante, primeramente por el lenguaje utilizado en ella, ya que C es un lenguaje que está orientado a la implementación de sistemas operativos en lugar de orientación a objetos que es a lo que nos encontramos acostumbrados, además de poseer datos estáticos, y que es considerado un lenguaje de nivel medio-bajo. Seguidamente, por el sistema operativo (Linux) que se debió utilizar ya que para muchos fue un reto adaptarse a él y lograrlo usar sin problema alguno.

En otras palabras el proyecto está orientado desde el inicio a impulsar a los estudiantes a que mejoren sus habilidades investigativas y de programación.

Fue bastante atrayente el obtener conocimiento en temas de programación totalmente desconocidos para nosotros, como lo fue el manejo de sockets desde la función `fork()`. Además que la aplicación es de los pocos programas en los que podemos ver una verdadera función empleada a la vida real, ya que hoy en día para nadie es totalmente nuevo enviar o recibir mensajes y/o archivos mediante un chat, principalmente porque estamos en una era tecnológica dominada por redes sociales que tienen ésta funcionalidad.

Finalmente, este proyecto programado permitió mejorar las habilidades personales tanto en la administración del tiempo, el trabajo en equipo y entre otras necesarias para el desarrollo del programa lo cual nos prepara aún más como futuros profesionales.