Instituto Politécnico Nacional   
Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas.

Ingeniería Telemática.

Sistemas Distribuidos.

Practica 4A Y 4B.

Dibujo en blanco y negro

Descripción generada automáticamente con confianza baja



DATOS GENERALES.

NOMBRE DE LA PROFESOR: Mata Rivera Migue Felix.  
NOMBRE DE LA TAREA: “Llamada a Procedimiento Remoto (RPC)”

FECHA DE ENTREGA: \_06/10/25\_.

NOMBRE DE LOS INTEGRANTES:

* Martinez Aparicio Eduardo.
* Suarez Ángeles Danna Paola.

GRUPO: 2TV7

REQUERIMIENTOS: AMBIENTE DE LINUX PARA EJECUTAR EL CÓDIGO CON PRIVILEGIOS DE

ADMINISTRADOR EN EL SISTEMA OPERATIVO. Los ejercicios se comprobaron exitosamente usando UBUNTU DESKTOP 18.04

SIN DISTRIBUIR.

1) un archivo de definiciones de funciones (random\_local.h)

2) un archivo con la función main (random\_local.c)

3) un archivo con la definición de dos funciones que se transformarán en RPC posteriormente.  
  
Texto

El contenido generado por IA puede ser incorrecto.

- LOCAL 5 6 -

III.- Codificación, Compilación y Ejecución DISTRIBUIDA

Imagen de la pantalla de un ordenador portátil

El contenido generado por IA puede ser incorrecto.

* SERVER -

Imagen de la pantalla de un ordenador portátil

El contenido generado por IA puede ser incorrecto.

* CLIENT 6 8 -

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

* ARCHIVOS GENERADOS -

EJERCICIO-TAREA para incluir en el reporte:

El reporte en ejecución de estos programas se puede hacer de forma local (una única computadora), pero la tarea debe realizarse al menos con dos computadoras o máquinas virtuales, tanto en una misma red como en redes diferentes ( sugerencia: use las maquinas virtuales provistas para ello, con su equipo de trabajo).

Es el ejercicio de la sección IIa, el cual únicamente lo que hay que hacer es mover los archivos cliente y el servidor para que vivan en computadoras diferentes (pero hay que revisar que tenemos que hacer para que sigan funcionando, tal y como lo hicieron en entorno local)

Basándose en el ejercicio de esta práctica, desarrolle un programa que se llamara calculadora que

realice cuatro operaciones básicas:

suma(int i, int j)  
resta (int i, int j)  
multiplica (int i, int j)  
divide (int i, int j)

Los tipos de datos pueden ser de tipo int o de otro tipo, quedan a elección del programador. Explique dentro de su reporte, que debe modificarse para que este programa se ejecute desde diferentes computadoras conectadas en diferentes redes.

Si su respuesta es afirmativa o negativa, justifique la respuesta, y en caso de que el código requiera ser modificado, también indique estas modificaciones en el código, y señálelas en el reporte explicando las razones e ilustrando la ejecución con capturas de pantalla y con un video no mayor a 5 minutos donde se muestre la ejecución. El código, reporte y el video ( no mayor a 5 min) envíelo en un enlace drive o de su nube de preferencia, a través del classrom de la unidad de aprendizaje, asimismo el código puede subirlo a github y referenciarlo.

Empezamos a modificar el archivo para llamarlo   
calculadora.x – el cual usaremos como punto de partida para generar los demas archivos.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

La estructura de este archivo es:

Texto

El contenido generado por IA puede ser incorrecto.

Posteriormente generamos en la terminal nuestros archivos a traves de la ejecucion de este archivo calculadora.x  
  
Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.  
- Generacion de archivos –

Posteriormente modificamos los achivos calculadora\_server.c y calculadora\_client.c  
a este modo.

|  |  |
| --- | --- |
| #include "calculadora.h"  #include <stdio.h>  #include <stdlib.h>  /\* Implementación del servicio SUMA \*/  int \*suma\_1\_svc(operandos \*ops, struct svc\_req \*rqstp) {  static int resultado;  resultado = ops->a + ops->b;  printf("Servidor: SUMA(%d, %d) = %d\n", ops->a, ops->b, resultado);  return &resultado;  }  /\* Implementación del servicio RESTA \*/  int \*resta\_1\_svc(operandos \*ops, struct svc\_req \*rqstp) {  static int resultado;  resultado = ops->a - ops->b;  printf("Servidor: RESTA(%d, %d) = %d\n", ops->a, ops->b, resultado);  return &resultado;  }  /\* Implementación del servicio MULTIPLICA \*/  int \*multiplica\_1\_svc(operandos \*ops, struct svc\_req \*rqstp) {  static int resultado;  resultado = ops->a \* ops->b;  printf("Servidor: MULTIPLICA(%d, %d) = %d\n", ops->a, ops->b, resultado);  return &resultado;  }  /\* Implementación del servicio DIVIDE \*/  double \*divide\_1\_svc(operandos \*ops, struct svc\_req \*rqstp) {  static double resultado;  if (ops->b == 0) {  printf("Servidor: DIVIDE(%d, %d) -> ERROR: división entre cero\n", ops->a, ops->b);  resultado = 0.0;  } else {  resultado = (double) ops->a / (double) ops->b;  printf("Servidor: DIVIDE(%d, %d) = %.2f\n", ops->a, ops->b, resultado);  }  return &resultado;  } | #include "calculadora.h"  #include <stdio.h>  #include <stdlib.h>  int main(int argc, char \*argv[]) {  CLIENT \*clnt;  operandos ops;  int \*resultado\_int;  double \*resultado\_double;  if (argc < 2) {  printf("Uso: %s <host\_servidor>\n", argv[0]);  exit(1);  }  clnt = clnt\_create(argv[1], CALCULADORA\_PROG, CALCULADORA\_VERS, "udp");  if (clnt == NULL) {  clnt\_pcreateerror(argv[1]);  exit(1);  }  printf("=== Cliente RPC Calculadora ===\n");  printf("Introduce dos números: ");  scanf("%d %d", &ops.a, &ops.b);  resultado\_int = suma\_1(&ops, clnt);  if (resultado\_int == NULL)  clnt\_perror(clnt, "Fallo en SUMA");  else  printf("Suma: %d\n", \*resultado\_int);  resultado\_int = resta\_1(&ops, clnt);  if (resultado\_int == NULL)  clnt\_perror(clnt, "Fallo en RESTA");  else  printf("Resta: %d\n", \*resultado\_int);  resultado\_int = multiplica\_1(&ops, clnt);  if (resultado\_int == NULL)  clnt\_perror(clnt, "Fallo en MULTIPLICA");  else  printf("Multiplicación: %d\n", \*resultado\_int);  resultado\_double = divide\_1(&ops, clnt);  if (resultado\_double == NULL)  clnt\_perror(clnt, "Fallo en DIVIDE");  else  printf("División: %.2f\n", \*resultado\_double);  clnt\_destroy(clnt);  return 0;  } |
| Calculadora\_server.c | Calculadora\_client.c |

Despues de generar los archivos y modificarlos, vamos a ejecutar los archivos y a levantar el enlace del servidor con el cliente en dos maquinas distintas.   
  
en este caso   
computadora de Eduardo sera servidor y computadora de Danna sera cliente.Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

* Cliente -

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.

* Servidor -

La direccion ip del servidor es:  
10.86.28.230  
  
adjuntamos LINK del video mostrando la ejecucion el ejercicio

<https://youtu.be/U7EE3wRTC9g?si=uxBwAZBaZhrr64IK>

conclusiones

Durante el desarrollo de esta práctica se logró comprender de manera integral el funcionamiento del modelo cliente-servidor y su aplicación en entornos distribuidos mediante el uso de llamadas a procedimientos remotos (RPC). A través del programa “calculadora”, se comprobó que el mecanismo RPC permite ejecutar funciones en otra máquina como si fueran locales, simplificando la comunicación y abstrayendo los detalles del envío de datos a través de la red. La generación automática de archivos con rpcgen facilitó la estructuración del sistema, garantizando la correcta serialización de datos y la interoperabilidad entre cliente y servidor. Al trasladar la aplicación de un entorno local a uno distribuido, se evidenció la importancia de una configuración adecuada de la red, las direcciones IP y los servicios del sistema, como rpcbind, para permitir la interacción entre equipos ubicados en diferentes redes. En conjunto, esta práctica reforzó conceptos fundamentales de la programación distribuida, como la transparencia, la modularidad y la cooperación entre procesos, destacando la relevancia de los RPC como base para la construcción de sistemas escalables y comunicados en red.

Se adjunta de la misma formal el link de la carpeta en github