Pontificia Universidad Javeriana
Departamento de Ingeniería de Sistemas
Sistemas Operativos, SIU 4085.
enero-mayo 2023

Proyecto 2: Chattering

Objetivo del Proyecto

Después de realizar este proyecto, los estudiantes estarán en capacidad de utilizar herramientas para la comunicación síncrona y asíncrona entre procesos de sistemas Linux, específicamente con pipes y las señales.

Descripción General

La mayoría de nosotros usa redes sociales como WhatsApp para comunicarnos rápidamente con conocidos, amigos, familiares, etc.

En este proyecto implementaremos un chat, que es una aplicación muy similar a lo que hoy es **WhatsApp o Telegram**, que nos permitirá enviar mensajes directos a una persona o a un grupo de personas. Los mensajes escritos serán observados de forma inmediata por todos los destinatarios. Nuestro sistema se llamará **Chattering**

Implementación

El sistema Chattering está conformado por una arquitectura Cliente/Servidor. *La **arquitectura cliente-servidor** es un modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los programas que requieren o demandan estos servicios, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta (Wikipedia).*

Los clientes y servidores del proyecto a implementar son procesos que estarán ejecutándose en un mismo computador.

Cada usuario que desee interactuar con otros usuarios a través del Chattering, deberá invocar un **proceso *talker*** (el cliente). Ese proceso le permitirá enviar y recibir mensajes, crear grupos, listar los usuarios y grupos en el sistema, etc. Así como sucede en las redes sociales, los usuarios se conectan por un determinado periodo de tiempo y luego cierran o terminan el proceso. Chattering estará formado también por un proceso servidor (**el Manager**), que registrará la información de clientes y grupos en el sistema y será intermediario para la comunicación entre los diferentes talkers.

Todos los procesos se comunicarán a través de **pipe nominales.** Al crear cada tipo de proceso, este recibirá como parámetro el nombre de un único pipe que servirá para las comunicaciones en el sentido cliente->servidor. Posteriormente se podrán crear otros pipes en la dirección servidor->cliente (Ver figura 1).

Talkers

Son los procesos que van a representar a los usuarios registrados en Chattering. Cada proceso usuario se ejecutará desde una consola con los siguientes argumentos:

$ talker –i ID –p pipeNom

Donde

ID: identifica al usuario que se quiere conectar en el sistema y será sencillamente un número del 1 al N (N es el máximo número de usuarios en el sistema y será un parámetro de creación del Manager). El sistema debe validar que el número de usuario sea válido (ID <= N), y si existe, que no esté ya conectado en el sistema.

pipeNom: es el nombre del pipe nominal que se utilizará para comunicarse con el proceso Manager. Todos los talkers se deben crear con el mismo nombre de pipe y este debe corresponder al nombre del pipe que usa el Manager.

Una vez iniciado el proceso talker, éste ofrecerá un prompt al usuario para que solicite alguna de las siguientes operaciones:

- **List**: lista los usuarios actualmente conectados en el sistema. Para obtener esta lista se debe preguntar al Manager.
- **List GID**: lista los integrantes del grupo con identificador GID. Esta operación falla si GID no es un identificador válido, es decir si no ha sido creado un grupo con este identificador. La **información de los grupos se encuentra en el Manager.**
- **Group ID1, ID2, ID3…IDN: Esta operación permite crear un grupo con usuarios.** El grupo creado debe registrarse en el Manager. Todos los miembros del grupo deben estar conectados actualmente o haber estados conectados en forma previa; de lo contrario la operación devuelve error. La operación devuelve a quien la invoca, un identificador del grupo creado y envía este identificador **de forma asíncrona** a todos los miembros del nuevo grupo. Este identificador debe ser único en el sistema. Una vez que se establece un grupo, cualquiera de los usuarios puede enviar un mensaje a sus integrantes. Pueden suponer que ningún talker va solicitar la creación de un grupo ya existente.

1. **Sent msg IDi:** Permite enviar un mensaje corto (máximo 100 caracteres) al talker con identificador = IDi. EL talker debe ver el mensaje de forma inmediata**.** El sistema devuelve un error si el Talker IDi no está conectado actualmente. **Este mensaje se debe enviar a través del Manager.** Los mensajes se envían a los destinatarios, no se almacenan en el Manager.
2. **Sent msg GroupID:** permite enviar un mensaje a todos los miembros del grupo con identificador GroupID. El programa da error sólo si el grupo no existe. Es posible que algunos miembros del grupo estén desconectados. Los miembros desconectados no recibirán el mensaje y lo pierden, este no se almacena en ninguna parte.
3. **Salir.** Con esta opción un talker se desconecta del sistema**.** El usuario puede volver a conectarse con el mismo ID con el que fue levantado inicialmente este proceso.

Note que mientras el talker se ejecuta atendiendo las peticiones del usuario, puede llegar un mensaje del Manager. Los tipos de mensajes que pueden llegar son: mensaje que envía un talker específico, formación de un grupo o mensaje que se envía a un grupo. Estos mensajes se muestran por la cónsola y pueden interrumpir la interacción actual del usuario y el talker por un momento; una vez que se recibe el mensaje, las operaciones deben continuar. **La atención de los mensajes asíncronos en el talker se implementará usando señales (signals).**

El Manager

Es el proceso encargado de gestionar las relaciones entre usuarios, los grupos y el envío de **mensajes** entre usuarios, ya sea en forma directa, entre un usuario y otro, o de un usuario a un grupo. El proceso **Manager** se invocará de la siguiente forma desde el *shell:*

$ ***Manager –n N -p pipeNom***

Donde:

N: es el número máximo de usuarios que pueden estar conectados simultáneamente en el sistema. Puede suponer que 100 será el máximo N con el que se invocará este programa.

pipeNom: es el nombre del pipe nominal por donde los procesos talkers se conectarán inicialmente al proceso Manager.

El Manager debe imprimir por la cónsola todas las operaciones que va realizando para los talkers.

Funcionamiento del sistema y comunicaciones

Al comenzar su ejecución, todo talker debe registrarse con el **Manager (así el Manager sabe que está conectado)**. Una vez registrado, el talker ejecutará un ciclo, en el cual le mostrará el prompt al usuario, para que éste solicite las distintas operaciones ya explicadas. Este programa termina cuando el usuario indique la opción de salir. El Manager debe estar al tanto de todos los usuarios conectados, las relaciones existentes y los grupos creados. El pipe ***pipeNom***, que reciben los dos tipos de procesos al ser inicializados, servirá de comunicación entre los talkers y el Manager. Un solo pipe es suficiente en este sentido. Se recomienda crear también un pipe entre el Manager y cada uno de los talkers, como lo muestra la figura 1, para los mensajes que envíe el Manager a cada talker. Uds. pueden crear otros pipes entre los procesos si su diseño lo requiere. El envío de mensajes entre procesos talkers (de forma directa o grupal) debe hacerse a través del Manager, como lo muestra la figura 2 (**Nota: recuerden que en este proyecto los talkers y los managers son procesos ejecutándose en un mismo computador**)

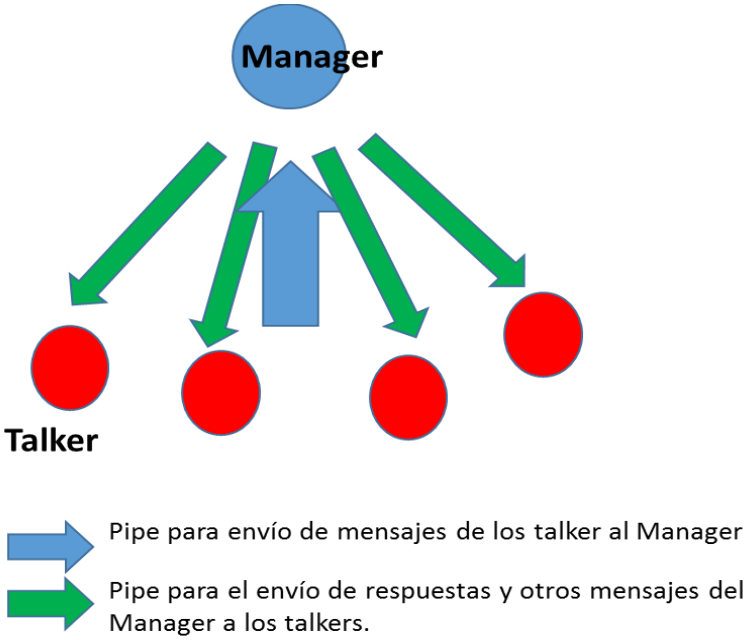

Figura 1: Pipes para las comunicaciones.

Finalmente, noten que un proceso talker está pendiente de las opciones del usuario y no de los mensajes que van llegando del Manager. La recepción asíncrona de los mensajes que envía el Manager a los talkers se puede implementar con una señal del proceso Manager al talker, avisando que tiene un nuevo mensaje para él.

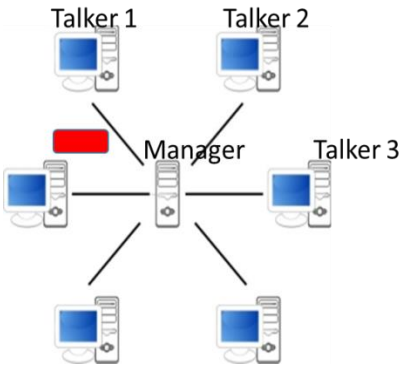

Figura 2a: Envío de un mensaje del talker1 al manager. El manager reenviará el mensaje a los talker 2 y 3

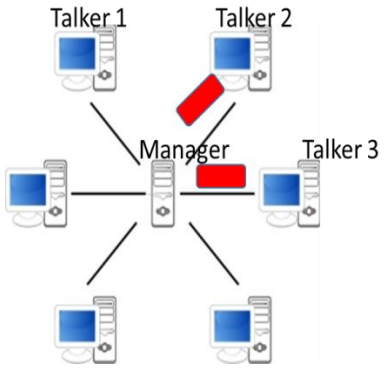

Figura 2b: re-envío del mensaje del talker 1 a los talker2 y talker 3.

Ejemplo de funcionamiento

A continuación, presentamos un ejemplo de ejecución de Chattering. En la primera columna presentamos lo que se está mostrando en el resto de las columnas: ejecución de programas, solicitud de comandos o salida de los programas. Cada columna, a partir de la segunda, muestra el

comportamiento de un determinado programa (Manager, talker 1, 2, ó 5). **En rojo se destacan salidas que son producto de un mensaje asíncrono**. El orden de las operaciones que realiza el Manager dependerá del orden de llegada de las peticiones. Los mensajes asíncronos pueden salir de cualquier manera en la cónsola de los talkers.

Comandos	**$Manager 10 pipecom**	**$Talker 1 pipecom**	**$Talker 2 pipecom**	**$Talker 5 pipecom**
Salidas	Manager iniciado y el sistema podrá tener como máximo 10 usuarios.	Se inicializa un talker con ID = 1	Se inicializa un talker con ID = 2	Se inicializa un talker con ID = 5
Comandos		Sent "hola" 4	Sent "hello" 5	
Salidas	Talker 1 desea enviar mensaje a Talker 4. Talker 4 no está conectado. Se devuelve error. Talker 2 desea enviar mensaje a Talker 5.	Error 4 no conectado		"hello" from 2
Comandos		List		
Salida	Talker 1 solicita los usuarios conectados.	Los usuarios actualmente en el sistema son 1,2,5		
Comandos			Group 1, 5	Sent "Hola como estas" 1
Salida	Talker 2 solicita creación de un grupo con talkers 1 y 5. Se crea un grupo con identificador G1 Talker 5 envía un	"Hola como estas" from 5 Talker forma parte del grupo G1	Se forma el grupo G1 con integrantes 1,2 y 5.	Talker forma parte del grupo G1

	mensaje al Talker 1			
Comandos			Sent “hey” G1	
Salida	Talker 2 envía “hey” a G1	“hey” from G1	“hey” from G1	“hey” from G1

Consideraciones de Diseño e Implementación

- Tenga en cuenta que el Manager y los procesos Talker son independientes. Se comunicarán **a través de pipes nominales y señales. Sólo se ejecuta un Manager, pero pueden existir varios Talkers.**
- El proceso **Manager** y cada proceso **Talker** se inician de la forma indicada y con los parámetros señalados. Una vez que el **Talker** esté en funcionamiento, permitirá a los usuarios realizar las operaciones descritas.
- El **Manager** debe ir informando por pantalla sobre las operaciones que está realizando.
- El proyecto lo pueden realizar en lenguaje C o C++ (en este caso usando las llamadas al sistema vistas en la clase para el manejo de pipes y señales).

Sobre la Entrega

- El proyecto lo deben realizar en grupos de **como máximo 3 estudiantes**.
- La entrega, se realizará por BS. El proyecto que van a subir en BS debe colocarse de forma comprimida usando el comando desde la consola de linux tar czvf (http://ecapy.com/comprimir-y-descomprimir-tgz-tar-gz-y-zip-por-linea-de-comandos-en-linux/index.html), los archivos empleados estarán dentro de un único directorio que tendrá como nombre proyecto II, más cuatro letras que corresponderán a los nombres y apellidos de los integrantes de grupo.
- La entrega consiste de los códigos fuente, el makefile.
- El código debe seguir los lineamientos de la Guía de Programación de C. Esta vez debería usar archivos de encabezado (.h). Debe validar llamadas al sistema y las entradas del usuario.
- **El día de la sustentación deberán realizar, individualmente, una modificación al proyecto,** por tanto, e**l proyecto debe poder ejecutarse en las computadoras de la Universidad (partición con SOP Ubuntu) o en Cocalc.**

  Deben seguir todas las especificaciones que este enunciado. Si tiene alguna duda diríjase al profesor de su sección.

  Prof. Mariela Curiel/Ricardo González/Osberth de Castro.