



# MINI-SHELL

SISTEMAS OPERATIVOS

**Presentado por:**

Edgar Jampier Leyva Garcia

Henry Mark Maquera Mamani



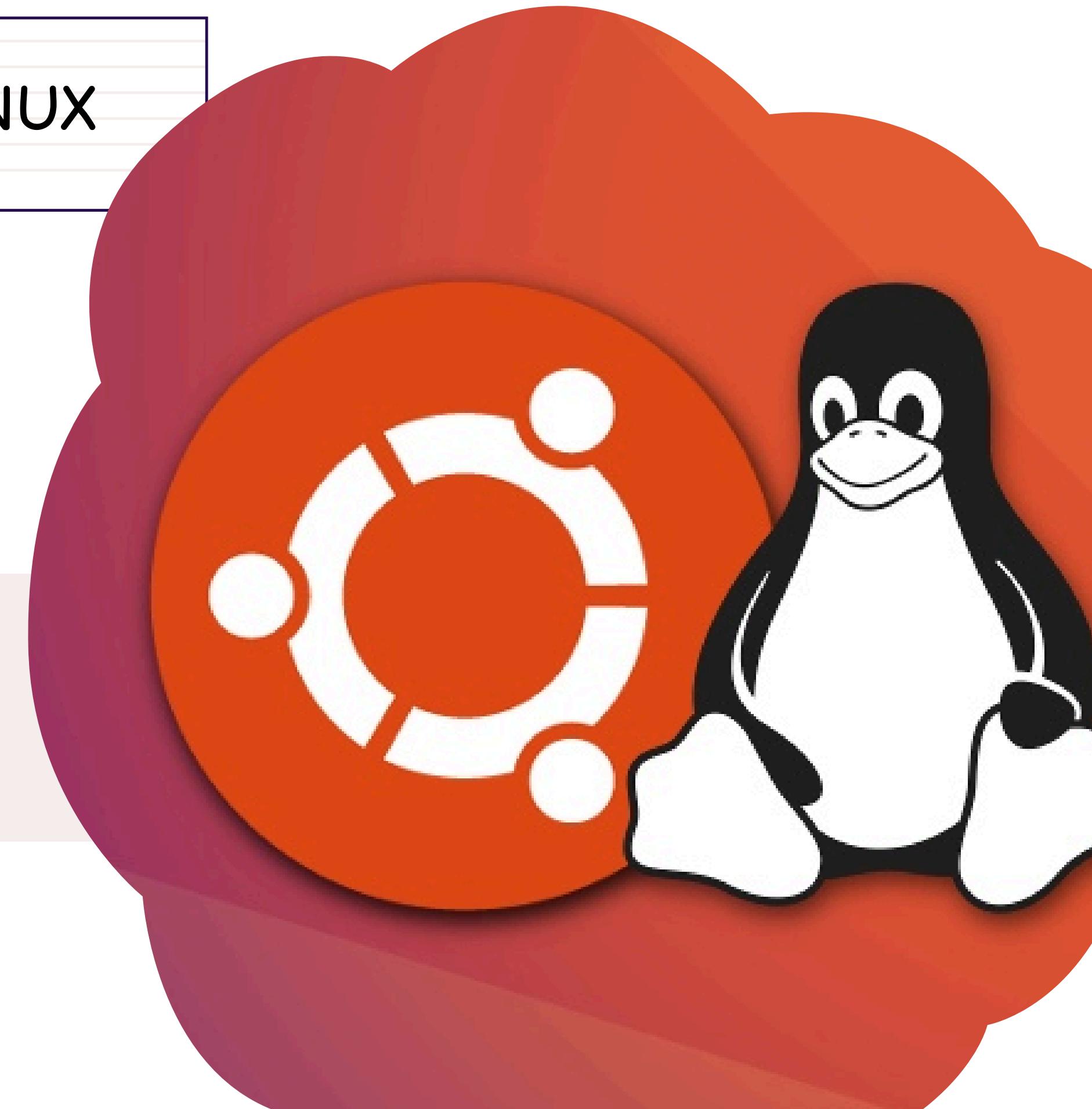
# INTRODUCCIÓN

Nuestro proyecto es la implementación de una mini-shell funcional en C++. El objetivo fue aplicar directamente la teoría de Sistemas Operativos, usando llamadas al sistema como fork, pipe y dup2 para construir desde cero un intérprete de comandos robusto y funcional.

## JUSTIFICACION LINUX

manejo de procesos, hilos y llamadas al sistema.

Incluye las funciones esenciales como fork(), exec(), wait(), pipe(), dup2(), y las bibliotecas pthread para la concurrencia.





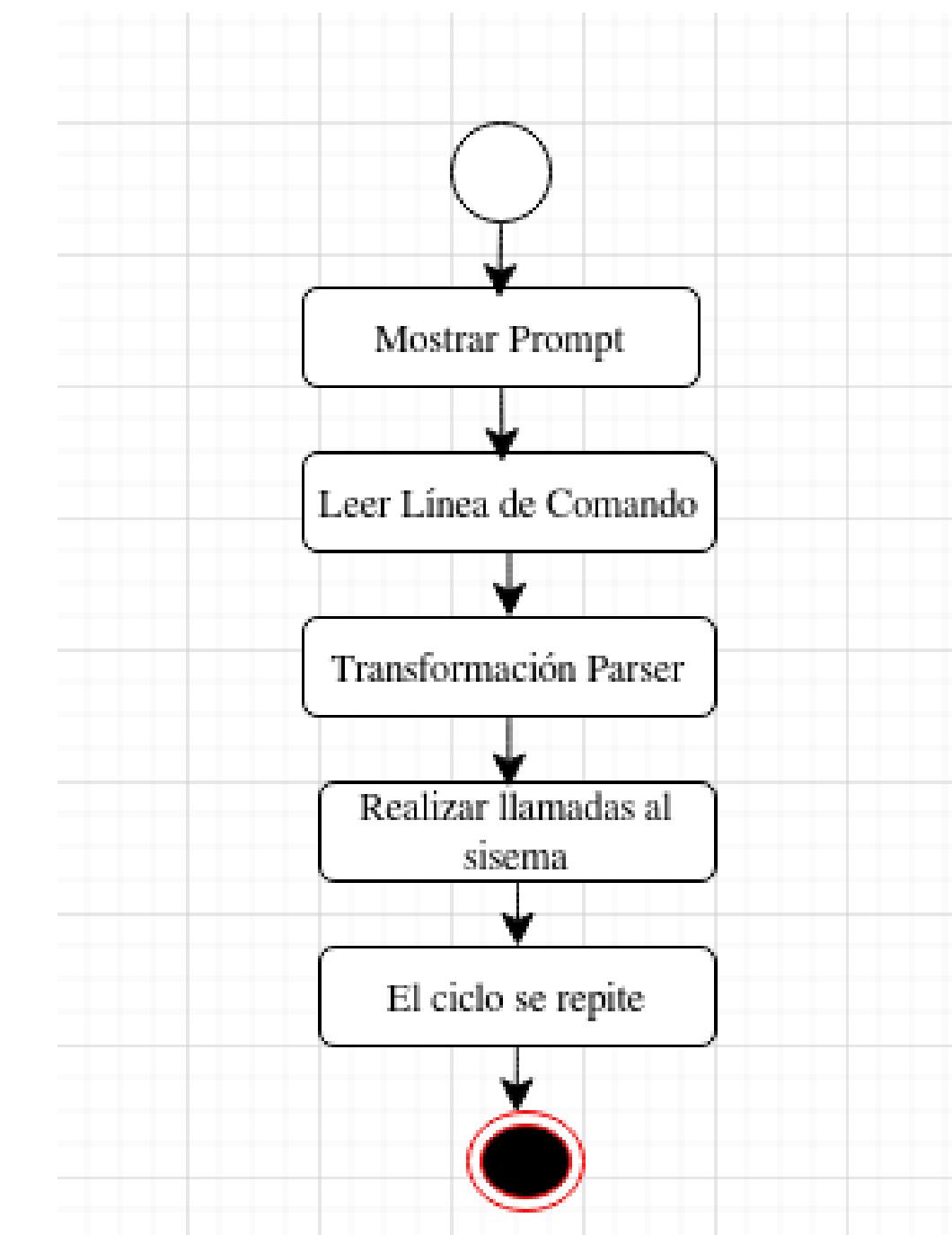
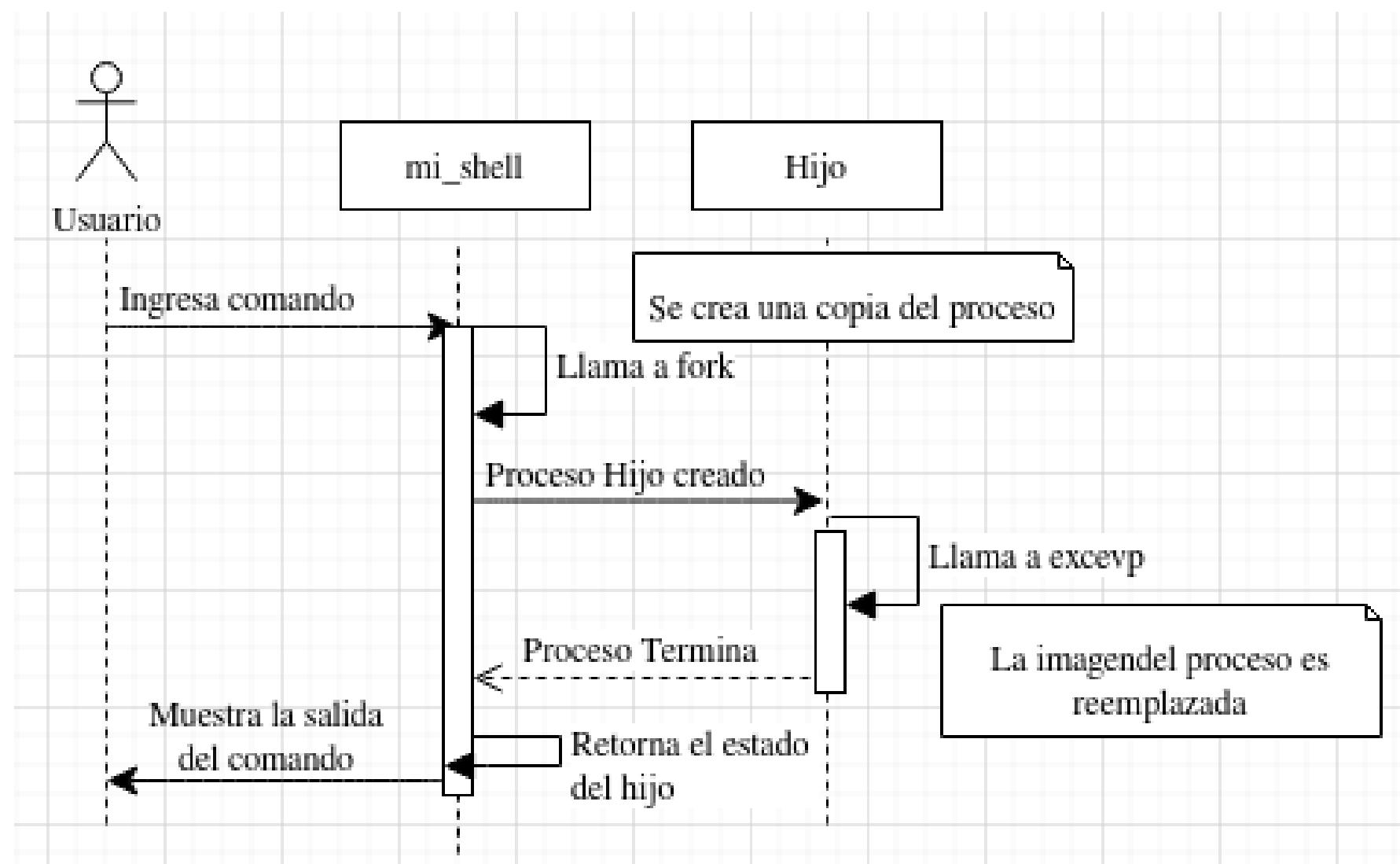
# OBJETIVOS



Diseñar una arquitectura modular y robusta capaz de interpretar y ejecutar comandos complejos, incluyendo tuberías (|), redirecciones (>, <) y ejecución en segundo plano (&).



# DIAGRAMA DE ACTIVIDADES





# FUNCIONALIDADES PRINCIPALES



## 1. Funciones Principales:

- Muestra el prompt personalizado con el directorio actual.
- Separa la línea de entrada en tokens utilizando espacios y tabulaciones.
- Permite cambiar el directorio de trabajo.
- Muestra el directorio de trabajo actual.

Muestra estadísticas de memoria del proceso.



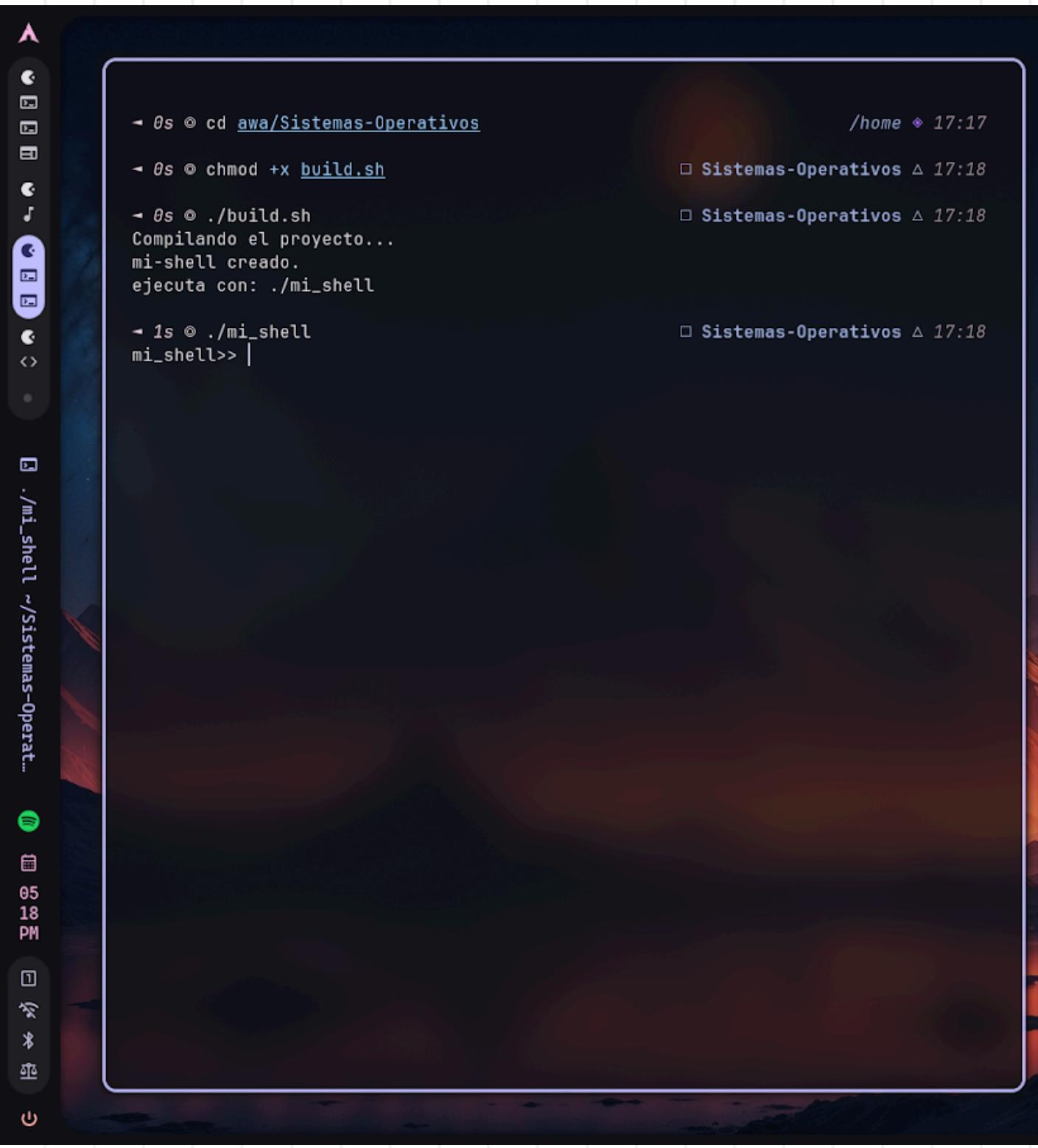


# COMANDOS INTERNOS

Comando	Descripción
<code>cd [directorio]</code>	Cambia el directorio de trabajo actual de la shell.
<code>pwd</code>	Muestra la ruta completa del directorio de trabajo actual.
<code>help</code>	Muestra una lista de los comandos internos disponibles y su función.
<code>clear</code>	Limpia por completo la pantalla de la terminal.
<code>salir</code>	Termina la sesión de la mini-shell y sale del programa.



# ● CAPTURAS DE EJECUCION DE ALGUNOS COMANDOS



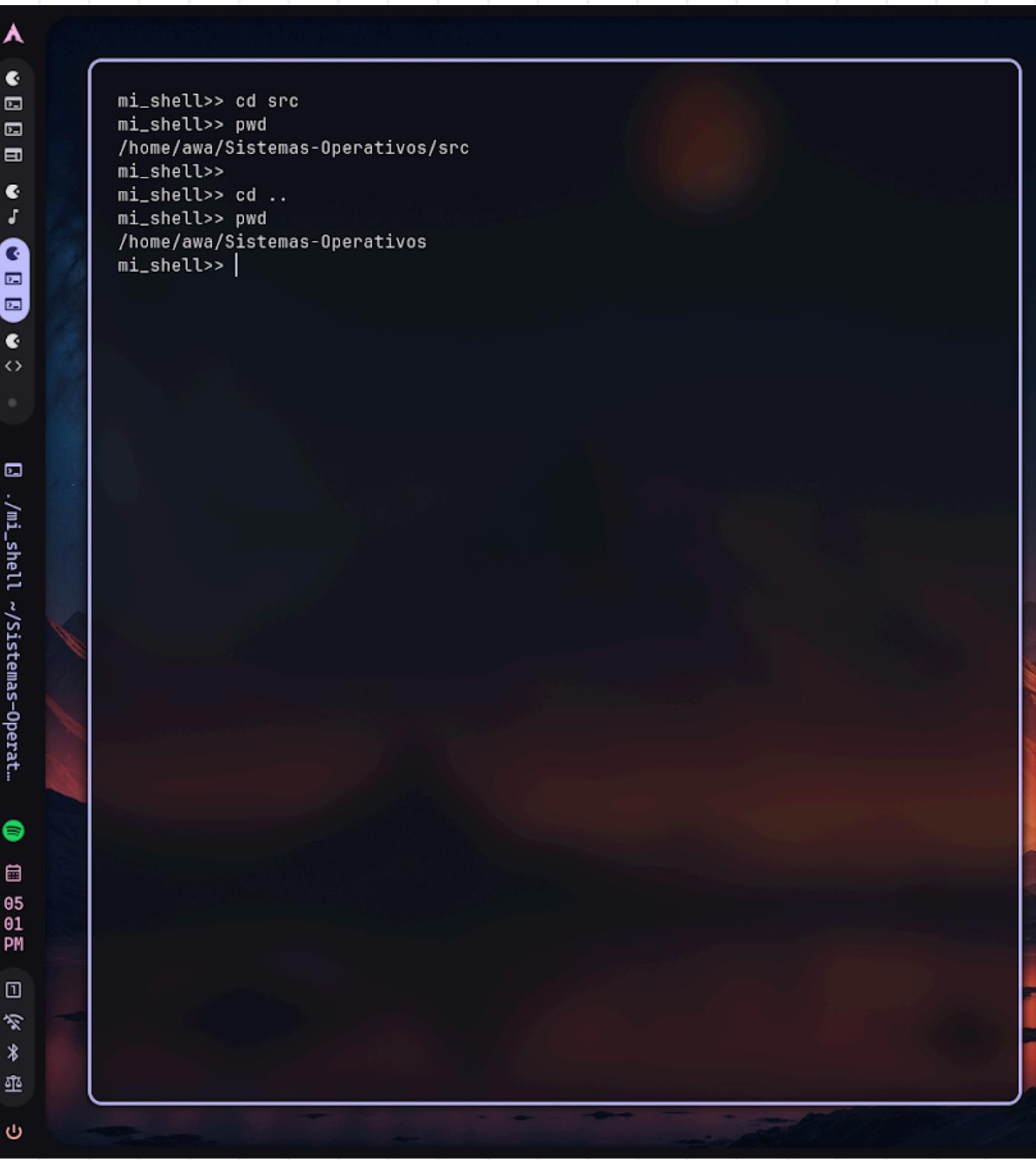
The screenshot shows a terminal window with a dark background and light-colored text. On the left, there's a vertical dock with various icons. The terminal content is as follows:

```
- 0s @ cd awa/Sistemas-Operativos                               /home ✧ 17:17
- 0s @ chmod +x build.sh                                         □ Sistemas-Operativos △ 17:18
- 0s @ ./build.sh                                              □ Sistemas-Operativos △ 17:18
Compilando el proyecto...
mi-shell creado.
ejecuta con: ./mi_shell

- 1s @ ./mi_shell                                              □ Sistemas-Operativos △ 17:18
mi_shell>> |
```



# ● CAPTURAS DE EJECUCION DE ALGUNOS COMANDOS



A screenshot of a terminal window titled "mi\_shell". The window shows a command-line interface with a dark background and light-colored text. The commands entered are:

```
mi_shell>> cd src
mi_shell>> pwd
/home/awa/Sistemas-Operativos/src
mi_shell>>
mi_shell>> cd ..
mi_shell>> pwd
/home/awa/Sistemas-Operativos
mi_shell>> |
```

The terminal window has a vertical toolbar on the left with icons for navigation, file operations, and other shell functions. The title bar shows the path and the title. The bottom status bar displays the current time as 05:01 PM.



# ● CAPTURAS DE EJECUCION DE ALGUNOS COMANDOS



A screenshot of a terminal window with a dark background and light-colored text. The terminal shows a session of a shell named 'mi\_shell'. The user runs several commands to list files and directories, specifically searching for '.cpp' files. The terminal interface includes a title bar with icons and a status bar at the bottom showing the path and current time.

```
mi_shell>> ls | grep .cpp
mi_shell>> ls
build.sh docs include mi_shell README.md src
mi_shell>> cd src
mi_shell>> ls | grep .cpp
executor.cpp
main.cpp
parser.cpp
mi_shell>> |
```



# EJECUCIÓN





# CONCLUSIONES

Diseñar una arquitectura modular y robusta capaz de interpretar y ejecutar comandos complejos, incluyendo tuberías (|), redirecciones (>, <) y ejecución en segundo plano (&).

# Gracias