Autores: Elier David Ibarra Medina & Samuel Adrian Montaña Linares
Sistemas Operativos - Pontificia Universidad Javeriana
Profesor John Jairo Corredor Franco
Tema: fork() - Comunicación entre procesos

TALLER 01

Introducción

Este informe documenta la implementación de un sistema multiproceso en C que utiliza la llamada al sistema *fork()* para crear procesos hijos y nietos, implementando comunicación mediante pipes anónimos. El sistema está diseñado para calcular sumas de números contenidos en archivos de manera concurrente.

Arquitectura del Sistema

- ***Estructura de procesos***
  - La jerarquía de procesos es la siguiente
    - Proceso Padre (***main.c***)
      - Primer Hijo (c1)
        - Grand-hijo (g) : Ejecuta ***suma1.c***
      - Segundo Hijo (c2): Ejecuta ***suma2.c***
- ***Comunicación entre procesos***

  Se utilizan tres pipes anónimos para la comunicación:
  - ***Pipe p1:*** Comunica el gran-hijo (***suma1.c***) con el padre
  - ***Pipe p2:*** Comunica el segundo hijo (***suma2.c***) con el padre
  - ***Pipe p3:*** Comunica el primer hijo (***sumatotal.c***) con el padre

Análisis de Componentes

- ***Proceso Principal***
  - Funcionalidad:
    - Validación de argumentos de entrada
    - Creación de pipes de comunicación
    - Gestión de la creación de procesos hijos
    - Recolección y presentación de resultados
  - Características técnicas:
    - Uso de fork() para creación de procesos
    - Redirección de E/S con dup2()
    - Gestión adecuada de descriptores de archivo
    - Mecanismo de espera con wait()
- ***Módulos de Cálculos***
  - ***suma1.c*** y ***suma2.c***
    - Calculan la suma de números en archivos individuales
    - Utilizan la función leerArchivo() para procesamiento de entrada
    - Retornan resultados mediante redirección de stdout
  - ***sumatotal.c***
    - Calcula la suma combinada de ambos archivos
    - Demuestra reutilización de código mediante la función leerArchivo()
- ***Módulo de Utilidades***
  - Función leerArchivo():
    - Lee números desde archivos de texto

        - Maneja formato flexible (números separados por espacios/tabs)
        - Asignación dinámica de memoria
        - Control de errores robusto

Flujo de Ejecución

- Fase de Inicio
    - Validación de parámetros: N1 archivo00 N2 archivo01
    - Creación de tres pipes de comunicación
    - Fork del primer hijo que a su vez crea un gran-hijo

- Fase de Procesamiento Concurrente
    - Gran-hijo: Ejecuta suma1 con archivo00, escribe en p1
    - Primer hijo: Ejecuta sumatotal con ambos archivos, escribe en p3
    - Segundo hijo: Ejecuta suma2 con archivo01, escribe en p2

- Fase de Recolección
    - Padre lee resultados desde los tres pipes
    - Presenta sumas individuales y total
    - Espera terminación de todos los procesos hijos

Conclusiones

El sistema demuestra efectivamente los conceptos fundamentales de:

- Creación y gestión de procesos con fork()
- Comunicación interproceso mediante pipes
- Redirección de entrada/salida