Universidade Federal de São Carlos

Laboratório de Sistemas Operacionais

Thiago Borges da Silva 613770
Mateus Abreu 612618

Projeto 02 - Gerência de Processos

---

1 Introdução

Este projeto tem como objetivo aprender sobre gerência de processos no Unix e explorar as chamadas de sistema utilizadas para esse fim. A partir de um interpretador de um shell fornecido pelo professor, foram implementadas 3 funcionalidades:

(a) permitir que os comandos executados recebam argumentos

(b) suporte para executar comandos em segundo plano

(c) redirecionamento de entrada e saída padrão

2 Comandos executados com argumentos

Para executar comandos com argumentos, foi fundamental o uso da função *strtok()*, que é uma função da biblioteca *string.h* que ao utilizada em sequência, divide uma sentença em *tokens* separados por um delimitador (no nosso caso, este delimitador foi o *spacebar*).

Feito isso, utilizamos o *execvp()*, que é uma primitiva da familia *exec()*, que recebeu o comando e a lista de *tokens*. A sua diferença para *execlp()* é que a primeira é utilizada quando se desconhece a quantidade de argumentos para o comando e, a segunda, é utilizada quando esta quantidade é conhecida.

3 Comandos em segundo plano

Para implementar esta funcionalidade, utilizamos uma verificação simples, onde se encontrassemos o '&', sinaliza-se uma flag. Esta flag é verificada na execução do processo-pai, onde a chamada *waitpid()*, que suspende a execução até que o processo-filho especificado mude de estado, era executada de duas formas distintas conforme a flag estivesse sinalizada.

Se a flag não estiver sinalizada, a chamada é executada com seu terceiro e último argumento (*options*) possuindo valor 0, caso contrário, seu valor era *WNOHANG*, fazendo com que retorne imediatamente, mesmo se o processo-filho não terminou.

4 Redirecionamento de entrada e saída padrão

De início, é verificado se existe '>' ou '<' no comando e, no caso de ser redireção de saída (>), utilizamos *popen*, que recebe um comando da shell e um tipo de ação, neste caso, a de leitura. O retorno desta função é o *output* do comando.

Para a entrada, os argumentos são lidos a partir do conteúdo de um arquivo e a lista de argumentos é novamente construída a partir do *strtok()*.

5 Conclusão

Por fim, o desenvolvimento do trabalho contribuiu com o entendimento de como funcionam as chamadas de sistema e gerência de processo. Nossa maior dificuldade foi implementar o redirecionamento da entrada, que nos deparamos com um problema onde os argumentos só eram reconhecidos a partir do segundo parâmentro.