Sistemas de Operación 1
Enero - Marzo 2017

Proyecto 1

Comunicación entre Procesos y Manejo de Archivos

Objetivo

Familiarización con las llamadas al sistema para la creación y acceso a las estructuras del sistema de archivos; y para el envío y recepción de información entre procesos.

Introducción al Problema

Es de gran utilidad en redes UNIX, donde los usuarios tienen asignadas cuotas de disco, contar con una aplicación que recorra recursivamente un árbol de directorios, generando un reporte con la información más importante para cada directorio en el árbol. De esta manera el usuario podrá saber con exactitud en qué directorios está invirtiendo mayor cantidad de espacio.

Este proyecto está enmarcado en el área de los sistemas de archivos y comunicación entre procesos, utilizando las herramientas y llamadas al sistema que han aprendido en las clases de Laboratorio. Se pide desarrollar una aplicación llamada *els* (enhanced – ls) que debe cumplir con las siguientes características:

- La línea de ejecución de la aplicación será:
  > els <*output_file*>
  donde el parámetro output_file será el *nombre base* de los archivos donde se generarán los reportes.

- A partir del directorio donde es ejecutado el comando y para cada directorio a partir de ese punto, deberá aparecer en el reporte la información que se especifica a continuación:

  - Path absoluto del directorio Ej: /preg/1/95-27445/
  - permisos : todos los permisos actuales del directorio en letras, al igual que son mostrados por el comando ls –l.
  - user-id : user id del dueño del directorio.
  - group-id : group id asignado al directorio.
  - Fecha de la última modificación.
  - Fecha del último acceso.
  - Número total de archivos contenidos en el directorio (se deben incluir los subdirectorios).
  - Número total de bytes ocupados por los archivos ubicados en ese directorio (no incluye subdirectorios).

- En caso de encontrar un archivo core durante el recorrido, éste será eliminado y agregado al reporte la información del directorio donde se encontraba el core, su fecha de modificación y el tamaño.

Un archivo core dump es generado cuando un programa termina abruptamente, ya sea por algún error, violación de memoria o por mecanismos de protección de acceso al hardware. Este archivo contiene un vaciado de la memoria y del estado del CPU para el momento en que terminó el proceso. Existen

herramientas que ayudan a examinar el contenido de un core con la intención de identificar el problema que produjo la finalización del proceso.

Implementación

- El proceso principal deberá crear un abanico de procesos de acuerdo a la cantidad de subdirectorios que existan en el directorio donde se ejecute el comando `els <output_file>`. Es decir que deberá existir un proceso hijo por cada subdirectorio existente en el directorio de ejecución.
- Cada proceso hijo deberá generar un reporte con la información descrita en el punto anterior a partir del subdirectorio del cual es responsable. El nombre del archivo reporte debe seguir el siguiente formato: `output_file-directory_name`
- El proceso Padre generará un reporte global para cada uno de los subdirectorios, considerando la información acumulada de sus hijos referente a:
    - Número total de archivos contenidos a partir del directorio de ejecución (incluyendo subdirectorios).
    - Número total de bytes ocupados por todos los archivos ubicados a partir del directorio de ejecución (no incluye subdirectorios).

  El nombre del archivo reporte para el Padre debe seguir el mismo formato: `output_file-directory_name`
- Los Procesos hijos le informarán al Padre la información acumulada necesaria para el reporte del Padre a través de *pipes no nominales*. Por lo tanto, el proceso Padre debe crear tantos pipes como hijos cree.
- El proceso Padre instalará un manejador de señales para evitar que sea interrumpido con CRTL-C desde la consola.

La figura 1 ilustra el esquema de creación de procesos y pipes, y un ejemplo de reportes finales para la estructura de directorio mostarada

Otros aspectos importantes a tener en cuenta para la implementación

- Tener en cuenta que en el recorrido se pueden encontrar directorios sin permiso de lectura o ejecución.

- Verificar que el directorio donde se desee crear el archivo reporte puede no tener permisos de escritura, en este caso dar un mensaje de error y abortar el proceso.

- Recuerden tomar en cuenta aspectos de modularidad, eficiencia y buen uso de la memoria.

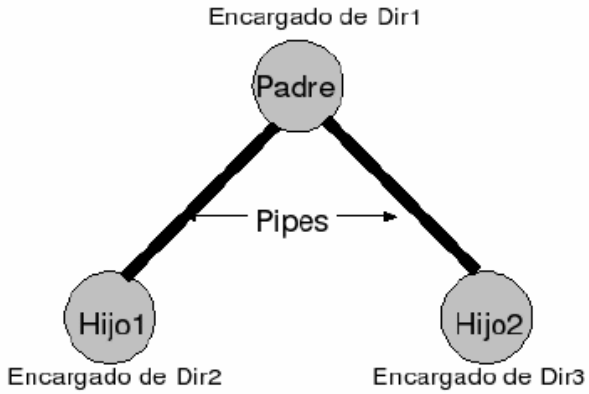
Entrega

El proyecto puede ser entregado el jueves de la semana 7 (23:55). La entrega consistirá de un archivo comprimido en formato *tar.gz* con los archivos fuente de su programa (.c y .h), makefile y README. No incluya el ejecutable ni los archivos objeto generados durante la compilación.

Suponiendo la siguiente estructura de directorios a partir de /home/yudith:

```
Dir1: Dir2:  Dir2.1: Dir2.1.1: M(5)
                     P(4)
                     Q(4)
              A(2)
              B(2)
              C(2)
       Dir3: Dir3.1: R(4)
              X(2)
              Y(2)
        N(2)
```

El nro entre () representa el tamaño en bytes de los archivos. Los DirN son directorios.



Reporte de Padre:
/home/yudith/Dir1 drwxr-xr-x yudith sasl 2008-03-01 2008-03-06 **3 2**
/home/yudith/Dir1/Dir2 drwxr-xr-x yudith sasl 2008-03-01 2008-03-06 **8 19**
/home/yudith/Dir1/Dir3 drwxr-xr-x yudith sasl 2008-03-01 2008-03-06 **4 8**

Reporte de Hijo 1:
/home/yudith/Dir2 drwxr-xr-x yudith sasl 2008-03-01 2008-03-06 **4 6**
/home/yudith/Dir2/Dir2.1 drwxr-xr-x yudith sasl 2008-03-01 2008-03-06 **3 8**
/home/yudith/Dir1/Dir2.1/Dir2.1.1 drwxr-xr-x yudith sasl 2008-03-01 2008-03-06 **1 5**

Reporte de Hijo 2:
/home/yudith/Dir3 drwxr-xr-x yudith sasl 2008-03-01 2008-03-06 **3 4**
/home/yudith/Dir3/Dir3.1 drwxr-xr-x yudith sasl 2008-03-01 2008-03-06 **1 4**

Figura 1