Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI3825 - Sistema de Operación I
Trimestre Enero-Marzo 2025

Proyecto Sincronizando Directorios

1. Introducción

El objetivo del proyecto es el diseño de una solución algorítmica para el problema de sincronizar los archivos de dos directorios.

2. Planteamiento del Problema

Se quiere sincronizar las archivos y directorios que se encuentran contenidos en dos directorios diferentes. Se quiere que realice una aplicación en el lenguaje de programación C, que sincronice dos directorios. La especificación de la aplicación es como sigue:

1. La entrada son dos directorios diferentes que llamamos d_1 y d_2.
2. Solo se va a tomar en cuenta archivos regulares y directorios, en consecuencia podemos ver el sistema de archivos como un árbol con raíz en el directorio /.
3. Se tiene que d_1 es un subárbol, con raíz d_1, del árbol del sistema de archivos. Lo mismo se puede decir para d_2.
4. Se tiene que d_1 y d_2 se comparan en el primer nivel de la altura del árbol. Es decir, se compararan los archivos y directorios del nivel 1.
5. Si en el nivel 1, se tiene que d_1 contiene un archivo x que no se encuentra d_2, se le debe preguntar al usuario si quiere copiar x a d_2, o si por el contrario quiere borrar el archivo x. Lo mismo aplica cambiando d_1 por d_2.
6. Si en el nivel 1, se tiene que d_1 contiene un directorio D que no se encuentra d_2, se le debe preguntar al usuario si quiere copiar recursivamente el directorio D, con todo su contenido, a d_2, o si por el contrario quiere borrar recursivamente el directorio D. Lo mismo aplica cambiando d_1 por d_2.
7. Si en el nivel 1 se tiene que ambos directorios d_1 y d_2, tienen un archivo x con el mismo nombre. Entonces, hay dos casos posibles:

   a) Los archivos tienen el mismo tamaño. En este caso se debe determinar si los archivos son iguales a no, es decir si tienen el mismo contenido. En caso de que los archivos sean iguales no se hace nada. Si son diferentes, entonces, se compara la fecha de la última de modificación de ambos archivos. Luego, se le recomienda al usuario copiar al otro directorio, el archivo con fecha de modificación más reciente. Finalmente, se le pregunta al usuario cual de los dos archivos quiere mantener. Esto es, sean x_1 y x_2

dos archivos diferentes, y si el usuario escoge x_1, entonces x_2 se elimina, y luego se copia x_1 al directorio donde estaba x_2.

b) Los archivos tienen tamaño diferente. En este caso se asumen que los archivos son diferentes. Luego, se le recomienda al usuario copiar al otro directorio, el archivo con fecha de modificación más reciente. Finalmente, se le pregunta al usuario cual de los dos archivos quiere mantener. Se opera como en e lpunto anterior.

8. Si en el nivel 1 se tiene que ambos directorios d_1 y d_2 tienen un directorio D con el mismo nombre, entonces se sincronizará esos dos directorios. Es decir, se sincronizan recursivamente estos dos directorios con el mismo nombre.

Al final de la sincronización se debe tener como resultado que:

1. Los directorios d_1 y d_2, contienen un mismo subárbol de directorios y archivos regulares.
2. Se debe mostrar por la salida estándar, la cantidad de archivos y kilobytes transferidos desde d_1 hacia d_2, y desde d_2 hasta d_1.

3. Actividad a Realizar

Debe realizar una aplicación llamada `syncDir`, que recibe como entrada el camino hasta los directorios, y los sincroniza. La línea de comando a ejecutar es como sigue:

```
>./syncDir [camino hasta el directorio d1] [camino hasta el directorio d2]
```

En caso de que alguna de las entradas consista en un directorio que no exista o en un archivo, se le da un mensaje de error apropiado al usuario. El resultado de la aplicación es que los dos directorios de la entrada, tengan el mismo contendido de directorios y archivos regulares, nivel por nivel del árbol de directorios, tal como se especificó en la sección anterior.

Por ejemplo, suponga que se tiene un directorio de archivos como el de la Figura 1. Se tiene que la aplicación `syncDir` y el usuario se encuentran en un terminal, en el directorio `palma` y se comparara los directorios `USB` y `OLDUSB`. Los siguientes son casos válidos de de la aplicación:

- Caso 1:

  ```
  ./syncDir USB ../Docs/backup/OLDUSB
  ```

- Caso 2:

  ```
  ./syncDir USB /home/palma/Docs/backup/OLDUSB
  ```

- Caso 3:

  ```
  ./syncDir /home/palma/USB ../Docs/backup/OLDUSB
  ```

- Caso 4:

  ```
  ./syncDir /home/palma/USB /home/palma/Docs/backup/OLDUSB
  ```

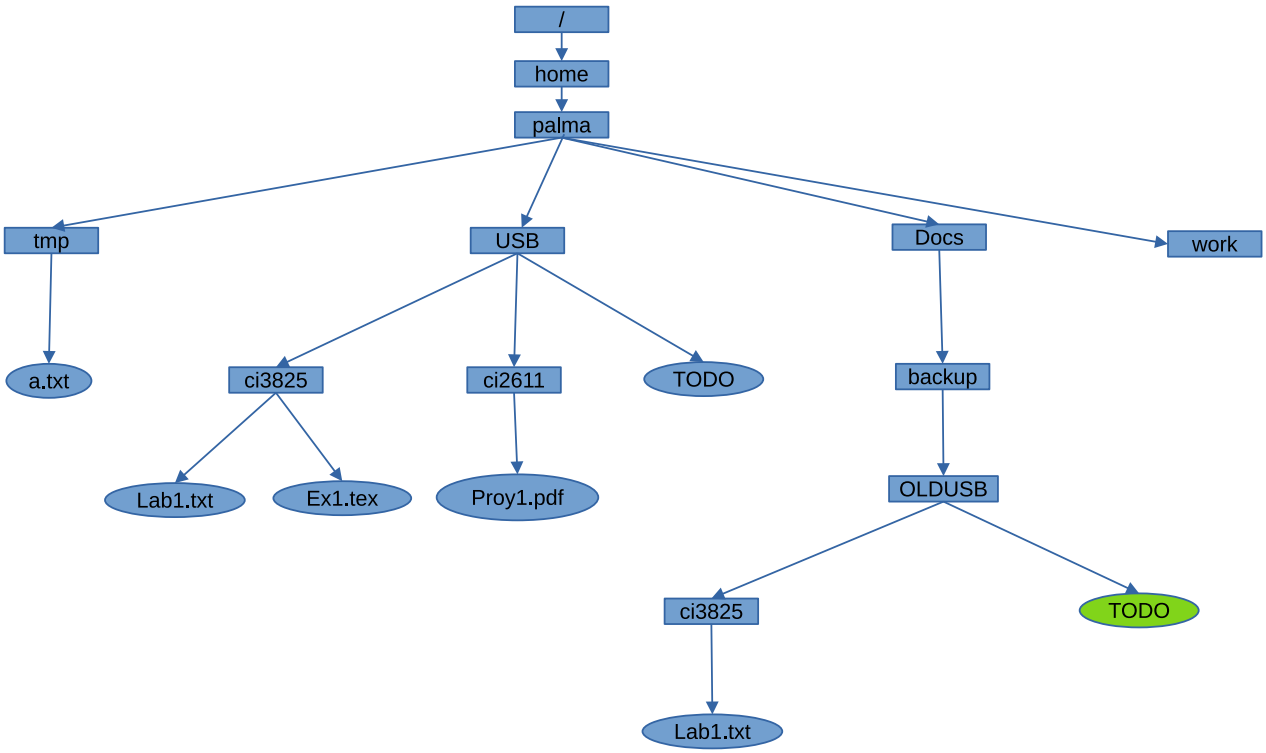

Figura 1: Ejemplo de un directorio de archivos. Los cuadrados son directorios y los óvalos son archivos. Se tiene que el archivo `TODO` en el directorio `USB` tiene una fecha de modificación más reciente que el contenido en `OLDUSB`.

Suponga que cuando de ejecuta `syncDir` se tienen los siguientes eventos y decisiones por parte del usuario, comparando inicialmente los directorios `USB` y `OLDUSB` en el nivel 1:

1. El directorio `ci2611` no está en `OLDUSB`. El usuario decide copiarlo a `OLDUSB`.

2. Los archivos `TODO` en ambos directorios son diferentes, siendo la versión del directorio `USB`, la más reciente. El programa le recomienda al usuario mantener la versión de `TODO` que está `USB`. No obstante, el usuario prefiere la versión que está en `OLDUSB`.

3. Se sincronizan los directorios `ci3825`. En este caso se comparan el nivel 1 de ambos directorios y se tienen los siguientes eventos:

    a) Se encuentra que el directorio `ci3825`, del subdirectorio de `OLDUSB`, no tiene el archivo `Ex1.tex`. El usuario decide que se debe copiar ese archivo al directorio `ci3825`, del subdirectorio de `OLDUSB`.

Al finalizar el procesos de sincronización, los directorios `USB` y `OLDUSB`, contienen los mismos archivos regulares y directorios como se muestra en la Figura 2.

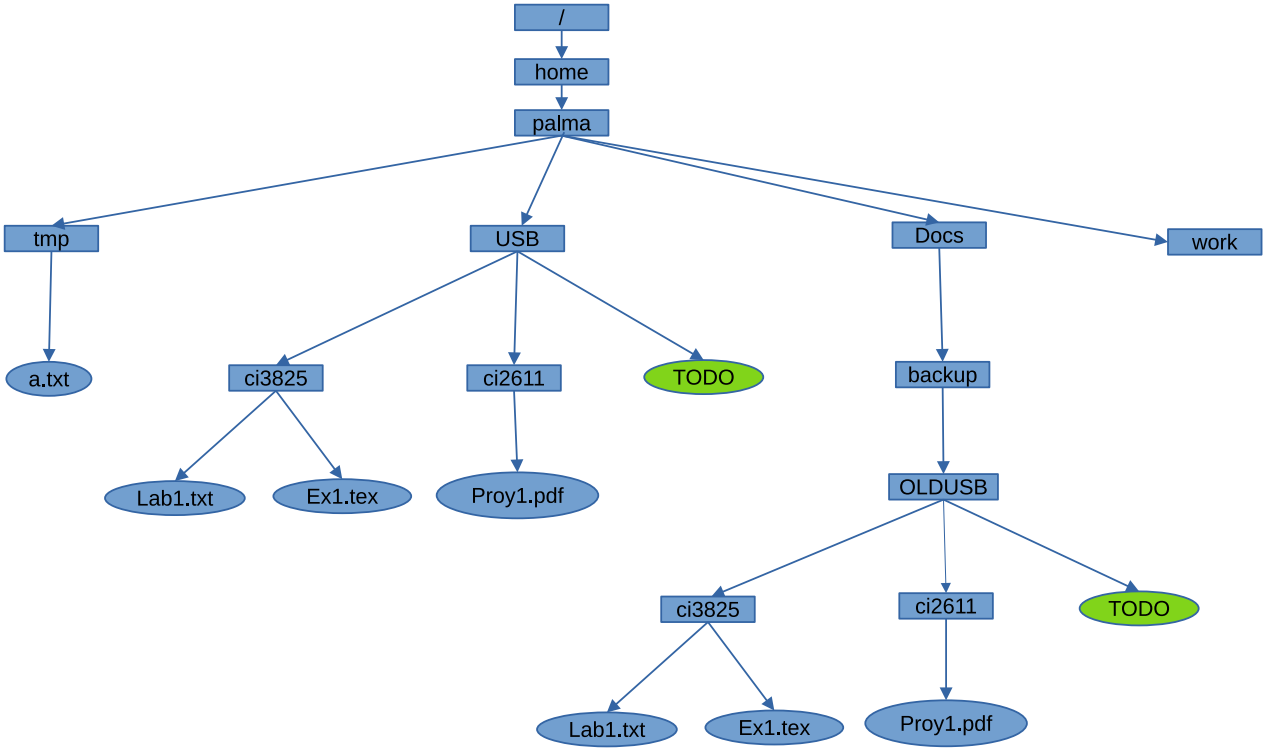

Figura 2: Ejemplo de un directorio de archivos después de la sincronización de los directorios `USB` y `OLDUSB`.

4. Requerimientos de la Implementación

Se quiere que su implementación haga uso de las llamadas al sistema y funciones sobre archivos vistas en clase. No se debe utilizar librerías diferentes a la librería estándar de C con extensiones GNU, para las operaciones sobre Strings. No debe usar librerías externas, a la librería estándar de C con extensiones GNU, para las operaciones sobre archivos y directorios. En específico, debe implementar en su proyecto las siguientes funciones, usando la librería C con extensiones GNU, y las operaciones sobre archivos y directorios vistas en clase:

- Una función que copia un archivo a un directorio, llamada `cp_file_to_dir`.
- Una función que copia un directorio a otro recursivamente, llamada `cp_dir_to_dir`.
- Una función que borra recursivamente, todo el contenido de un directorio, llamada `rm_dir`.
- Una función que retorna `true`, si dos archivos tienen el mismo contenido, de lo contrario retorna `false`, llamada `same_content_file`.
- Una función que sincroniza dos directorios, llamada `sync_dirs`.

Su código debe se modular, debe estar documentado y debe seguir la guía de estilo dada en clase. Debe proporcionar un archivo `Makefile` que compile todo el código, usando los `flags` recomendados en clase, y generar el ejecutable `syncDir`.

5. Informe del proyecto

Debe realizar un *breve* informe, que tiene como objetivo la descripción de su programa. El informe debe contener las siguientes secciones:

Portada: con la información de los autores, nombre, apellido y carné de los estudiantes

Descripción del sistema: Debe explicar el diseño de la solución y las principales estructuras de datos utilizadas. Esta sección no debe ser mayor de dos páginas.

Referencias bibliográficas: En caso de haber utilizado referencias para la elaboración del proyecto.

6. Condiciones de la entrega

Los códigos, el informe y la declaración de autenticidad debidamente firmada, deben estar contenidos en un archivo comprimido *tar.xz*, llamado *Proy3_X_Y.tar.xz*, donde X y Y son los número de carné de los estudiantes. La entrega del archivo *Proy3_X_Y.tar.xz*, debe hacerse por medio de la plataforma *Classroom* antes de las 9:30 A.M. del día viernes 04 de abril de 2025.

Guillermo Palma / gvpalma@usb.ve / Marzo 2025