



Alumnos:

- Francisco de Jesús Méndez Lara
- Emmanuel Muñoz Cerda
- Sandra Olimpia Estrella Prieto García
- Gerardo Femat Delgado

Profesor: Javier Santiago Cortes López

Fecha: 22/06/22

4^{to} SEMESTRE, GRUPO C

Introducción: En este proyecto se realizará un administrador de archivos desarrollado en Shell Script. Mediante el uso de una interfaz de texto para el usuario (TUI). Usando como medio para su creación y programación una máquina de Virtual Box y Visual Studio Code.

Marco teórico:

- ❖ **Tecnología:** Se hizo uso de Linux Mint como sistema operativo, para estar programando los Shells. El tipo de shell que utilizado fue bash.
- ❖ **Dialog:** Es un comando de GNU/Linux que permite crear cuadros de dialogo en el terminal para que se utilice en los scripts. Suele ser muy usado para administración de sistemas, instalación de distribuciones, configuración de aplicaciones, etc. También permite el control de programas en un tiempo de ejecución.
- ❖ **ncurses:** Son librerías que provee una API que permite al programador escribir interfaces basadas en texto y TUI's. También optimiza el refresco de la pantalla, lo que permite reducir la latencia experimentada cuando se usan intérpretes de comandos remotos. Estas librerías ayudan a trabajar con funciones similares a las contenidas en la librería conio.h del viejo turboC de Borland, entre las cuales están: gotoxy(), clrscr(), textcolor(), getch(), etc. Con ncurses se pueden hacer aplicaciones gráficas elegantes en modo texto y de forma muy fácil para ser utilizadas en terminal.

Temas del curso que se usaron a lo largo del proyecto:

- **Shell Script:** Es un grupo de comandos, funciones y variables, tienen la misma sintaxis de varios lenguajes de programación y está fuertemente influenciado por el lenguaje C. Los Shell Scripts son capaces de manejar las mismas tareas que estos. Los scripts deben escribirse en un archivo de texto al que se le otorga permisos de ejecución. Además, existen dos tipos de scripts: los basados en bourne shell y los derivados de C.

Ya que se definió lo que es un Shell Script, ahora se definirá lo que es **Shell Scripting**. Esta es la técnica de diseñar y crear un Script mediante un Shell de un Sistema Operativo, o un editor de texto. Este siendo un tipo de lenguaje de programación que generalmente es interpretado durante la ejecución.

- **Metacaracteres:** El metacarácter son caracteres específicos que interpreta el sistema operativo mandando indicaciones a la computadora cómo proceder. Nuestro sistema operativo leerá un metacarácter e interpretará lo que significa, para posteriormente realizar una acción basada en esa interpretación. Tal acción puede incluir instruir al sistema para que separe las declaraciones o buscar diferentes deletreos de una palabra, así como cambiar los colores e inclusive imprimir cuadros.

Desarrollo de los puntos vistos a lo largo del proyecto:

Se desarrollaron los puntos solicitados en el proyecto, iniciando una investigación sobre "*midnight commander*", una vez observado su funcionamiento, se prosiguió a seguir investigando, pero enfocado a lo que podría servir para crear una interfaz de texto. Esto fue debido a que se consideró más importante aprender el uso del programa para realizar la interfaz ya que no se vieron durante clases, que los scripts mismos. Investigando, se encontró una paquetería llamada "*dialog*", la cual solucionaba gran parte de los problemas planteados con la interfaz, ya que cuenta con herramientas de gran utilidad para configurar el administrador de archivos.

Se comenzó por programar los scripts de menú, tras esto se fue conectando la interfaz para que estuviera ciclada, mediante la etiqueta "*--fselect*" se abre el menú para seleccionar archivos o directorios al iniciar a trabajar con los scripts, ya que esa etiqueta retorna la ruta en la que el usuario esta seleccionando. Se programaron las condiciones para detectar si el usuario está trabajando con un directorio o un archivo dentro de un menú de opciones, esto provoca que se le muestren diferentes opciones: si es un directorio o si es un archivo. Ya teniendo las diferentes opciones, a cada opción se le dio su script correspondiente, de los cuales se desarrollaron 3

tipos: scripts para directorios, para archivos y para ambos (estos los llamamos generales).

Retomando lo mencionado, el `--fselect` proporciona la ruta donde se encuentra el usuario, esto se pasa por parámetro a los scripts y de ahí se valida si es una ruta aceptable, si el usuario tiene permisos para hacer las modificaciones o tratar el directorio del archivo como es debido. Después se investigó la manera en la que se podría ejecutar el script al iniciar sesión en la máquina virtual de ejemplo, para que cada vez que se haga un login se ejecute en pantalla completa. Para lograrlo se usó Fedora, configurándolo para que al arrancar el SO el usuario realice su login e inicie el script de forma automática.

También se programaron las funciones extendidas, estas funcionan tanto para archivos como directorios: cambio de permisos, cambio de propietario y cambio de grupo, únicamente para directorios: agregar prefijos. Para cambiar los permisos, se necesita hacer log in como root o el usuario al que le pertenece ese archivo, para modificar los permisos del grupo, los mismo se puede hacer desde el usuario root o el usuario al que le pertenece ese archivo. Para cambiar el propietario de un archivo o directorio, se debe hacer solamente accediendo con root. Para el prefijo se debe seleccionar una carpeta a la que su contenido se le desea añadir los prefijos.

Finalmente se desarrolló el makefile, el cual revisa que estén los programas necesarios, todos los scripts y cambia a modo de ejecución cada uno de los scripts para que el usuario pueda ejecutar el main.sh.

Problemas encontrados a la hora de desarrollar el proyecto y sus soluciones:

La principal dificultad que se presento fue el saber cómo configurar el dialog, ya que se estuvo leyendo constantemente su manual e investigando etiquetas específicas de éste en internet, pero esto último no fue de mucha ayuda, ya que en su mayoría son ejemplos muy básicos que no cumplía el objetivo, por lo tanto fue batallar a base de prueba y error, hasta que se logrará comprender el funcionamiento de este, una vez comprendido, generar lo demás se volvió bastante sencillo.

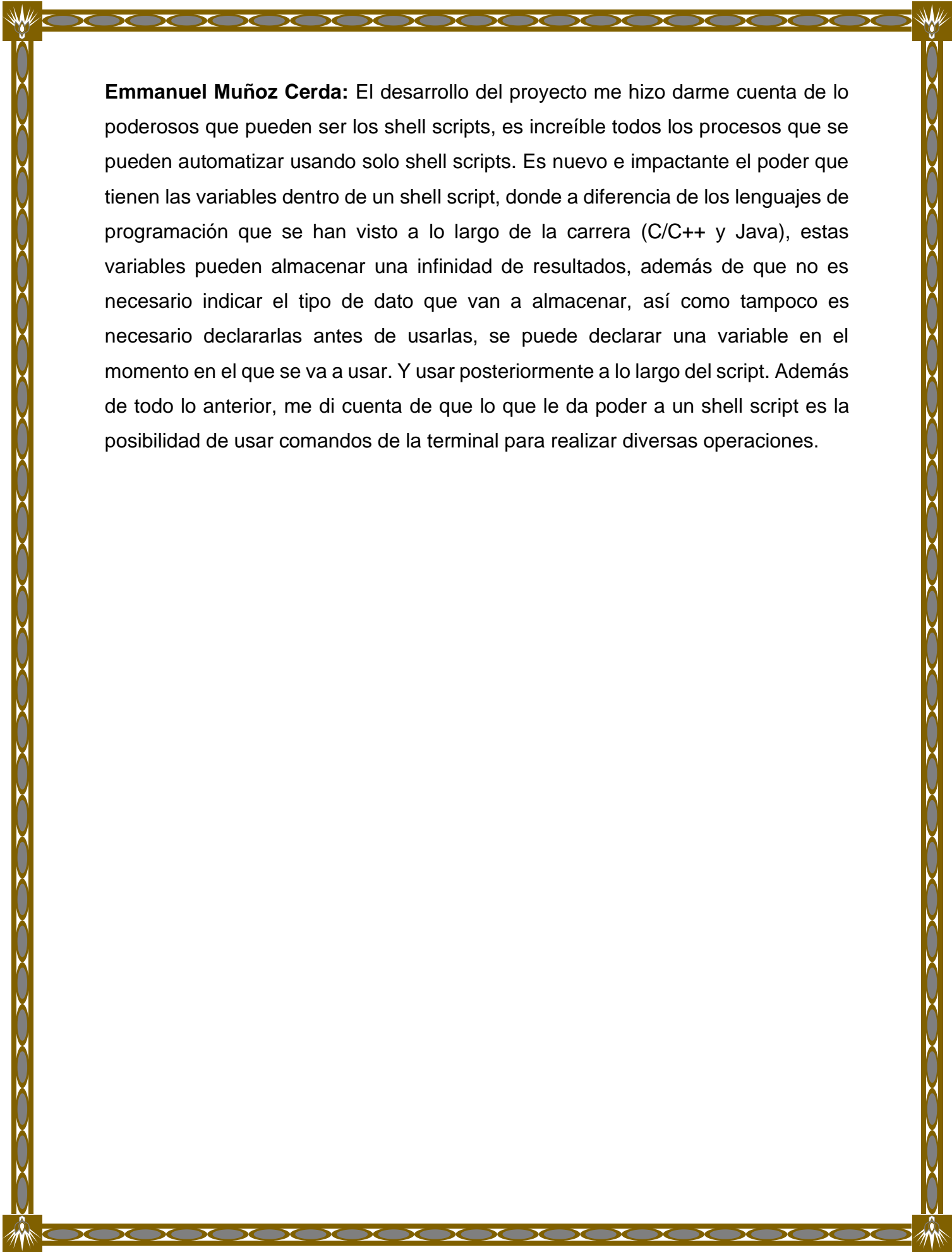
Conclusión: El Shell script es una manera de generar programas de administración muy sencillos, esté proyecto sin duda fue un reto, ya sea por la dificultad que se presentaba de forma lógica al momento de crear el administrador de archivos desde cero, así como la poca información que se encontraba a la hora de investigar. Se tuvieron que realizar investigaciones más profundas o exactas, para guiarnos e incluso a veces ni así se lograba encontrar algo. En gran parte del proyecto, como se mencionó con anterioridad en el desarrollo de este documento, el administrador de archivos fue creado a base de prueba y error, con el fin de aprender el funcionamiento de algunos programas como el dialog o comprender el funcionamiento de los comandos que utilizamos para programar los demás scripts.

Conclusiones por integrante:

Gerardo Femat Delgado: Durante la creación del proyecto reforzamos los conocimientos obtenidos en el curso, tomando conciencia de las capacidades que tiene el shell scripting y su gran importancia en el entorno para administración y mantenimiento de los sistemas. El mayor reto sin ninguna duda fue investigar y comprender el funcionamiento de “dialog” ya que fue muy complicado comprender como retornaba aquello que seleccionaba el usuario y si seleccionaba un botón específico. Una vez comprendido qué, cómo y por qué retornaba valores concretos todo fue más sencillo. Por igual debí investigar sobre las funciones en bash, para que fuera fácil el manejo de la interfaz. Una vez con los conocimientos básicos solo se tuvieron que crear los scripts conociendo como se retornaban los valores y manejarlos con readlink y dirname, comandos muy útiles al manejar archivos o directorios. En cuanto a los scripts a pesar de ser muchos fueron sencillos de realizar, con lo visto en clases y un poco de investigación fue suficiente. El proyecto en un contexto general me gusto hacerlo, el shell scripting es una herramienta muy poderosa en los sistemas like-UNIX y sobre todo imagino en servidores.

Francisco de Jesús Méndez Lara: Durante el tiempo que estuve investigando sobre la problemática planteada a simple vista no parecía un gran reto, ya que con lo aprendido en clases sobre lo que es el Shell Scripting sonaba bastante sencillo, las cosas se tornaron complicadas cuando comenzamos a investigar sobre la interfaz de texto, ya que no había mucha información en internet y la que había era bastante sencilla, tratamos de analizar lo que era el midnight commander, pero al ver fue desarrollada casi en su totalidad en C, decidimos dejar de investigar su código para no cerrarnos a no intentarlo solamente con Shell Scripts. Al investigar sobre dialog también se nos presentó diferentes obstáculos, ya que no terminábamos de comprender como funcionaba, si retornaba valores, que modificaciones hacía, pero al menos yo después de leer todo el manual de usuario brindado por el “man” vi diferentes tipos de ventanas que nos proveía, por lo que ya teníamos por donde comenzar. Fue bastante entretenido experimentar con el dialog, lo cual nos dio un parámetro muy amplio para comenzar a hacer la TUI, donde finalmente adaptar los Scripts para que se armonizara con nuestra interfaz fuera bastante buena, la verdad la capacidad de configuración que tienen los Shell scripts, la verdad me sorprendió, entendí mejor el porque es muy usado para mantenimiento y administración de sistemas.

Sandra Olimpia Estrella Prieto García: A lo largo de este proyecto, me di cuenta de que es de suma importancia el saber investigar bien tus fuentes de consulta, para poder encontrar algo que te guíe u oriente bien. No fue nada fácil el lograr avanzar el proyecto, ya que a pesar de que teníamos las bases para poder estar programando algunos scripts, el saber cómo hacer uso de otras cosas como el “dialog”, fue el que nos detuvo. Ya que tuvo su cierto grado de complejidad el saber cómo funciona y usarlo.



Emmanuel Muñoz Cerda: El desarrollo del proyecto me hizo darme cuenta de lo poderosos que pueden ser los shell scripts, es increíble todos los procesos que se pueden automatizar usando solo shell scripts. Es nuevo e impactante el poder que tienen las variables dentro de un shell script, donde a diferencia de los lenguajes de programación que se han visto a lo largo de la carrera (C/C++ y Java), estas variables pueden almacenar una infinidad de resultados, además de que no es necesario indicar el tipo de dato que van a almacenar, así como tampoco es necesario declararlas antes de usarlas, se puede declarar una variable en el momento en el que se va a usar. Y usar posteriormente a lo largo del script. Además de todo lo anterior, me di cuenta de que lo que le da poder a un shell script es la posibilidad de usar comandos de la terminal para realizar diversas operaciones.

Anexos:

Link para ver todos los códigos de proyecto: https://eduuaa-my.sharepoint.com/:f/g/personal/al289539_edu_uaa_mx/EIF2BKQ2UjxPtnmDT3BKTL8BGOEGpY9SWXdnfPFzPPV85A?e=PluUM1

Link con todo el proyecto: https://eduuaa-my.sharepoint.com/:f/g/personal/al289539_edu_uaa_mx/EmWUfBnZ3oNBmO0LOwU9apIBB-2cyYuzZsAzWlx2SaTYVw?e=LNf4V7

Función de cada código:

- **Main:** Script principal del programa, lugar donde se cicla el fselector y todos los scripts que conforman el proyecto.
- **copiarDir:** Copia un archivo y obtiene la ruta del destino a donde se quiera copiar.
- **creaDir:** Crea un directorio desde la ruta donde se especifica.
- **eliminaDir:** Verifica la dirección que se le envíe y verifica que este sea un directorio, después de encontrarlo y confirmarlo lo elimina.
- **moverDir:** Mueve un directorio con todo su contenido a una nueva dirección.
- **renombrarDir:** Cambia el nombre de un directorio.
- **copiarFile:** Copia un archivo y obtiene la ruta del destino a donde se quiera copiar.
- **creaFile:** Primero verifica que la ruta enviada pertenezca a un directorio, de comprobarse crea un archivo. En caso de que falle la acción, mostrara un mensaje con error.
- **editarArchivos:** Modifica el contenido del archivo al que se le vaya a editar, por el contenido que se le mande.
- **eliminaFile:** Primero comprueba que desde la ruta donde se solicita la función sea un archivo, para después de confirmarlo manda un mensaje confirmando que se desea eliminar ese archivo, en caso de que la respuesta sea afirmativa se elimina, de no serlo se cancela la acción.
- **Mostrar:** Muestra el contenido de un archivo mediante un msgbox.
- **moverFile:** Mueve un archivo con todo su contenido a una nueva dirección.
- **renombrarFile:** Cambia el nombre de un archivo.
- **cambiarGrupo:** Si el usuario que solicita la acción es root o pertenece al grupo de destino, hace el cambio de grupo con el comando chown.
- **cambiarPermisos:** Se verifica que sea un archivo o un directorio lo que se solicita, para después mostrar en un checklist los permisos que se puede otorgar de forma ordenada, para que el usuario los pueda seleccionar.

- **cambiarPropietario:** Manda un mensaje preguntando al usuario con que usuario va a cambiar. Comprueba que exista el usuario con el que se quiera cambiar y de verificarlo, lo hace. En caso de que no exista, manda un mensaje de error.
- **Comprimir:** Comprime un archivo solicitado y muestra en pantalla dónde y cómo fue comprimido. En caso de que falle al comprimir, manda un mensaje de error.
- **Descomprimir:** Descomprime el/los archivo(s) que se seleccione desde la ruta solicitada, mostrando cuales fueron los archivos descomprimidos, en caso de fallar manda un mensaje de error.
- **Desempaquetar:** A los .TAR les saca los archivos que tienen contenidos.
- **Empaquetar:** Junta los archivos, pero no los comprime, como lo hace el comprimir.
- **renombrarTodo:** Tiene la funcionalidad de tener un mejor control sobre archivos generados automáticamente o para archivos/directorios descargados, para así lograr tener un control más organizado sobre archivos específicos en un directorio. Esta función esta solo disponible en los directorios.
- **Makefile:** Revisa que estén los programas necesarios, los scripts y modifica los permisos para que se puedan ejecutar.

Fuentes de consulta:

- <https://conocimientolibre.mx/shell-scripting/>
- <https://sanchezcorbalan.es/que-es-un-shell-script/>
- <https://aplicacionesysistemas.com/en/dialog-crear-menus-tus-scripts/>
- [https://www.ecured.cu/Dialog_\(programa_de_Linux\)](https://www.ecured.cu/Dialog_(programa_de_Linux))
- http://gluc.unicauca.edu.co/index.php/Programación_con_Ncurses#:~:text=Las%20ncurses%20son%20librerías%20que,y%20de%20forma%20muy%20fácil
- <https://www.netinbag.com/es/internet/what-is-a-metacharacter.html>