Proyecto 2

RoboticTEC

Instituto Tecnológico de Costa Rica
Escuela de Ingeniería en Computadores
Principios de Sistemas Operativos
I Semestre 2025

Objetivo

Implementar las diferentes capas necesarias para que una aplicación pueda interactuar con hardware a través del sistema operativo, así como fomentar el diseño y creatividad de los estudiantes en la integración del software y hardware.

Aplicar un algoritmo destribuidor de cargas de procesamiento en un sistema distribuido para optimizar el tiempo de respuesta, utilizando OpenMPI con información cifrada

Motivación

Con el desarrollo de este proyecto, se implementará un prototipo hardware, el cual deberá de tener su respectivo device driver y todas las capas de software que este requiere para que puedan interactuar el hardware y SO por medio del mismo. Se desarrollarán y reforzarán técnicas de integración del hardware y software por medio del device driver, además se detallará el proceso de comunicación necesario del sistema operativo hacia el hardware del computador para que la interacción sea adecuada. También se enfocará en el procesamiento distribuido para equilibrar cargas dentro de un sistemas con varios nodos.

Descripción del problema

El proyecto relaciona 4 áreas de los sistemas operativos: drivers, hardware, seguridad y procesamiento distribuido.
En general, el proyecto cuenta con las capas que se muestran en la figura 1, las cuales son de suma importancia por los insumos que necesitan y proporcionan para efectuar de buena manera el procesamiento.

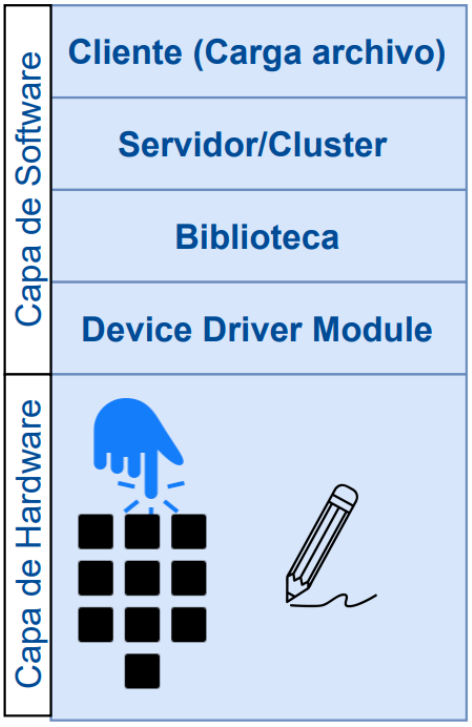

Figura 1: Capas generales del proyecto

Dispositivo Físico

Se deberá crear un prototipo de hardware, utilizando cualquier sistema embebido. Como parte del diseño se deberá elegir alguna interfaz para interactuar con el sistema (GPIO, RJ45, VGA, o cualquier otro), el cual debe contar con una justificación del porqué de dicha elección (Se aconseja que sea GPIO). El dispositivo mecánico será el encargado de presionar en el orden correcto las letras en un dispositivo con algún teclado, también se puede realizar para que sea capaz de escribir una palabra en papel (Esto queda a elección del grupo) . El objetivo es “simular” la capacidad de una mano. El teclado puede ser de cualquier tipo ya sea físico o digital. Una vez que el dispositivo termine de colocar todos los caracteres de la palabra debe hacer alguna señal para indicar que ya terminó (puede encender algún led, hacer un movimiento, incluso poner un punto, esto quedará a criterio de cada grupo)

Hardware

Se debe crear un prototipo de “mano” que sea capaz de realizar las funciones antes descritas. El diseño y la manera mecánica de este prototipo quedará directamente ligado a la creatividad del estudiante. Se requiere los diagramas de los componentes que fueron utilizados para el hardware.

Driver

Se deberá desarrollar un device driver en el lenguaje de programación C y utilizando algún ambiente de Linux. Será el encargado de proveer a las capas superiores las primitivas necesarias para la interacción con el dispositivo físico. La interacción con el dispositivo físico será mediante el device driver (encargado de que el sistema operativo lo reconozca), en otro caso la nota será 0 en el rubros correspondientes. Es importante que considere que el device driver necesita ser un módulo del kernel de Linux (verificar con lsmod que se esté ejecutando). Se debe prestar atención a **NO** utilizar ningún driver del sistema para la interfaz que se utilice. Es importante resaltar que cada grupo debe saber defender el código del driver, en caso contrario, la nota será cero. No se permite únicamente utilizar un driver comercial, si no que debe estar justo a la medida para que lo se vaya a implementar.

Biblioteca

Se deberá crear una biblioteca (se espera un **.a** con los métodos específicos), la cual es la única que interactuará con el driver desarrollado en el punto anterior. La idea fundamental de este módulo es proporcionar un conjunto de funciones consumibles al usuario para que pueda interactuar con el hardware a través del device driver. Se debe generar el **.a**. Esta biblioteca será la encargada de proporcionar funciones como Read y Write, MoveRight, entre otras.

Superficie de donde escribir o digitar la palabra

La superficie donde se debe escribir la palabra quedará a elección de cada grupo, puede ser un teclado físico, un papel o incluso un teclado digital. Lo realmente

importante es que se logre observar en pantalla o en papel la palabra que debe escribir o digitar el prototipo físico.

Servidor-Clúster

Será el encargado de controlar el flujo de información, y es quien le brinda órdenes al hardware utilizando la biblioteca. El servidor será el encargado de orquestar el procesamiento distribuido entre los 3 nodos, además debe de recibir el resultado final proveniente de los nodos de procesamiento. El servidor recibirá por parte del cliente un documento texto cifrado caracter por caracter (el algoritmo de cifrado lo elige cada grupo). La idea es que este servidor divida el proceso en diferentes nodos para que cada uno decifre parte del documento y encuentren la palabra que más se repite en todo el documento. La manera en que se encuentre la palabra que más se repita en el documento quedará a diseño de cada grupo, sin embargo, el algoritmo debe ser paralelizable de manera distribuida.
Una vez que se encuentre la palabra que más se repita, se debe escribir o digitar en la superficie de escritura, junto con la cantidad de veces que se decifra.
Es importante destacar que el servidor debe guardar tanto el archivo cifrado como el decifrado.

Cliente

El cliente deberá de enviar el archivo al servidor para que este divida el procesamiento. También será el encargado de recibir el archivo y cifrar caracter por caracter utilizando el algoritmo de cifrado elegido por cada grupo. En esta parte no se espera interfaz gráfica.

Requerimientos técnicos

- Este proyecto se debe realizar en el lenguaje de programación C o RUST.
- Debe ser implementado en Linux, sin **máquinas virtuales**.
- Se debe proporcionar un Makefile, que genere e instale lo necesario.

- No se permiten soluciones “alambradas”. Esto es, que los distintos módulos de software no esten completamente delimitados y cuya configuración no se dé por una interfaz definida.
- Se debe prestar especial atención a los errores de acceso a memoria o utilización de recursos. Es inaceptable el error ***segmentation fault*** o ***Core dumped***.
- En cada nodo se debe mostrar el monitor del sistema para ver el consumo de recursos.

Documentación externa

Este proyecto debe ser trabajado en un repositorio de GitHub. La documentación se hará en la wiki del mismo repositorio de GitHub. No debe entregar un documento PDF aparte, como usualmente se hace.

- Introducción: teoría necesaria, breve descripción del proyecto y qué es lo que se espera en el escrito.
- Ambiente de desarrollo: software de terceros que se utilizó en el desarrollo del proyecto Esto incluye frameworks, bibliotecas externas o principales, aplicaciones de terceros, herramientas de desarrollo.
- Detalles del diseño del programa desarrollo, tanto del software como del hardware (en caso de que aplique): Diagramas UML, diagramas de flujo, imágenes, descripciones entre otros, todo lo que sea necesario para entender de una mejor manera el diseño y funcionamiento del proyecto.

Entregables

- Código fuente con documentación interna.
- Documentación externa en la wiki del repositorio.
- Cualquier otro archivo o dependencia necesaria para ejecutar el código.

Aspectos administrativos

- No se reciben trabajos después de la hora indicada.
- En la revisión del proyecto pueden estar presentes el coordinador y asistente.

- Es responsabilidad del estudiante proveer los medios para poder revisar la funcionalidad del software, por ejemplo, si no se realiza la interfaz, se debe de proporcionar otro medio para la verificación, de lo contrario la nota será cero en los rubros correspondientes a la funcionalidad faltante.
- En caso de que alguna parte de la especificación sea confusa o ambigua, el grupo de trabajo puede agendar una cita con el profesor para aclarar cada aspecto.
- Fecha de asignación: 31 de mayo del 2025
- Fecha de entrega **y defensa**: 25 de junio
- Grupos: 3 o 4 personas

Rúbrica de evaluación

Funcionalidad (80%)	Valor (%)
Driver	20
Hardware	15
Cliente	5
Servidor	15
Biblioteca	10
Clúster	15
Documentación externa (20%)	
Introducción	5
Ambiente de desarrollo	5
Diseño	10