

# **Proyecto de Curso: Análisis, Diseño y Construcción de un Sistema Operativo desde Cero**

Por

Castro Pari, Rayneld Fidel

Mamani Flores, Natan

Mendoza Quispe, Jose Daniel

Polo Chura, Marco Rosauro

Trabajo académico presentado a la

Facultad de Ingeniería

proyecto final de unidad

Ingeniería de Informatica y Sistemas

Departamento Académico de Ingeniería de Informatica y de Sistemas

Asesor: Ugarte Rojas, Héctor Eduardo

Memorial Universidad Nacional San Antonio Abad del Cusco

Semestre 2025-II

## Abstract

This document provides information on how to write your thesis using the L<sup>A</sup>T<sub>E</sub>X document preparation system. You can use these files as a template for your own thesis, just replace the content, as necessary. You should put your real abstract here, of course.

*“The purpose of the abstract, which should not exceed 150 words for a Masters’ thesis or 350 words for a Doctoral thesis, is to provide sufficient information to allow potential readers to decide on relevance of the thesis. Abstracts listed in Dissertation Abstracts International or Masters’ Abstracts International should contain appropriate key words and phrases designed to assist electronic searches.”*

— MUN School of Graduate Studies

## Acknowledgements

Put your acknowledgements here...

*“Intellectual and practical assistance, advice, encouragement and sources of monetary support should be acknowledged. It is appropriate to acknowledge the prior publication of any material included in the thesis either in this section or in the introductory chapter of the thesis.”*

— MUN School of Graduate Studies

# Contents

<b>Abstract</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>List of Tables</b>	<b>vi</b>
<b>List of Figures</b>	<b>vii</b>
<b>1 Marco teorico</b>	<b>1</b>
1.1 Arquitecturas de Sistemas Operativos . . . . .	2
1.1.1 Arquitectura Monolítica . . . . .	2
1.1.2 Arquitectura en Microkernel . . . . .	3
1.1.3 Arquitectura Hibrida . . . . .	4
1.1.4 Arquitectura en Exokernel . . . . .	5
1.1.5 Comparacion entre arquitecturas de Sistemas Operativos . . .	6
<b>2 Revision de S.O. existentes</b>	<b>7</b>
2.1 Fuchsia . . . . .	8
2.2 Haiku . . . . .	10

3 Comparación técnica	12
Bibliography	13

# List of Tables

1.1 Comparación de arquitecturas de SO . . . . . 6

# List of Figures

1.1	Sistema operativo basado en kernel monolítico . . . . .	3
1.2	Sistema operativo basado en microkernel . . . . .	4
1.3	Sistema operativo basado en kernel híbrido . . . . .	5
1.4	Sistema operativo basado en exokernel . . . . .	6
2.1	Icono de S.O. de fuchsia . . . . .	8
2.2	Icono de S.O. de Haiku . . . . .	10

# Chapter 1

Marco teorico

## 1.1 Arquitecturas de Sistemas Operativos

### 1.1.1 Arquitectura Monolítica

El sistema operativo monolítico es un sistema operativo muy simple donde el núcleo controla directamente la gestión de dispositivos, memoria, archivos y procesos. Todos los recursos del sistema son accesibles al núcleo. En los sistemas monolíticos, cada componente del sistema operativo está contenido dentro del núcleo.

En una arquitectura monolítica, el núcleo del sistema operativo está diseñado para proporcionar todos los servicios del sistema operativo, incluyendo la gestión de memoria, la programación de procesos, los controladores de dispositivos y los sistemas de archivos, en un único y gran binario. Esto significa que todo el código se ejecuta en el espacio del núcleo, sin separación entre los procesos del núcleo y los del usuario (GeeksforGeeks, 2024b).

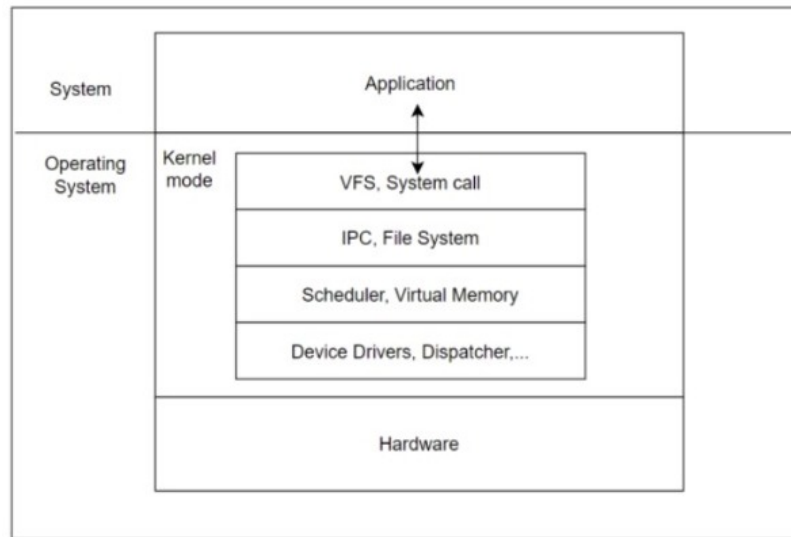


Figure 1.1: Sistema operativo basado en kernel monolítico (Harshvardhan and Irabatti, 2023)

### 1.1.2 Arquitectura en Microkernel

Un microkernel es un enfoque para diseñar un sistema operativo (SO). El microkernel proporciona servicios fundamentales para su funcionamiento, como la gestión básica de memoria, la programación de tareas. El microkernel es un tipo de sistema operativo que proporciona servicios básicos.

como los controladores de dispositivos y los sistemas de archivos, son gestionados por procesos a nivel de usuario. El proceso a nivel de usuario se comunica con el microkernel mediante el paso de mensajes. Esta forma de gestionar el proceso hace que los microkernels sean más modulares y flexibles que los kernels monolíticos tradicionales (GeeksforGeeks, 2024a).

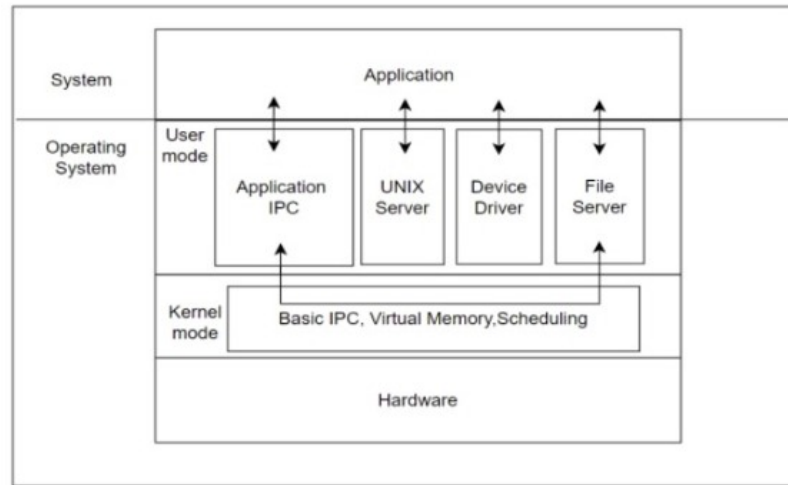


Figure 1.2: Sistema operativo basado en microkernel (Harshvardhan and Irabatti, 2023)

### 1.1.3 Arquitectura Híbrida

Un núcleo híbrido es una arquitectura de núcleo basada en la combinación de aspectos de las arquitecturas de micronúcleo y núcleo monolítico utilizadas en sistemas operativos .

La idea de esta categoría es tener una estructura de kernel similar a la de un microkernel, pero implementada en términos de un kernel monolítico. A diferencia de un microkernel, todos (o casi todos) los servicios del sistema operativo se encuentran en el espacio del kernel . Si bien no hay sobrecarga de rendimiento para el paso de mensajes ni el cambio de contexto entre el kernel y el modo de usuario, como en los kernels monolíticos , no hay ventajas de rendimiento al tener servicios en el espacio de usuario , como en los microkernels (Microsoft Wiki / Fandom, 2024).

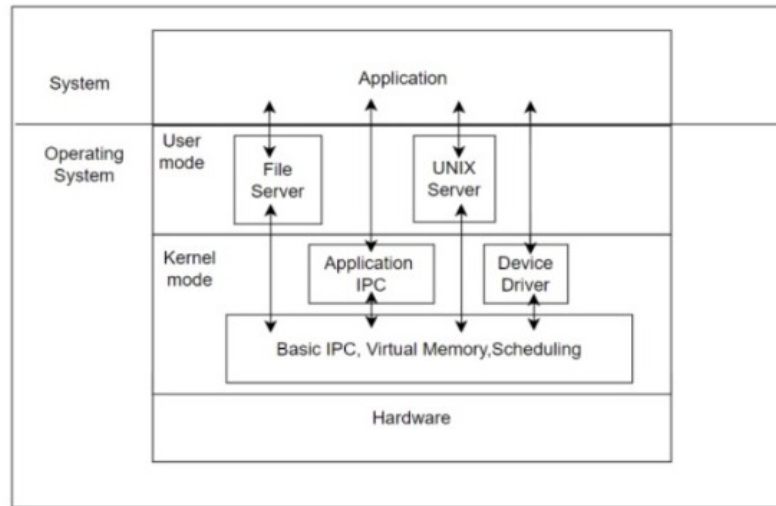


Figure 1.3: Sistema operativo basado en kernel híbrido (Harshvardhan and Irabatti, 2023)

#### 1.1.4 Arquitectura en Exokernel

Exokernel es un sistema operativo desarrollado por el Instituto Tecnológico de Massachusetts (MIT) con el concepto de poner la aplicación bajo control. Los sistemas operativos Exokernel buscan proporcionar gestión de recursos de hardware a nivel de aplicación. La arquitectura de este sistema operativo está diseñada para separar la protección de recursos de la gestión, facilitando así la personalización específica de cada aplicación (Keetmalin, 2017).

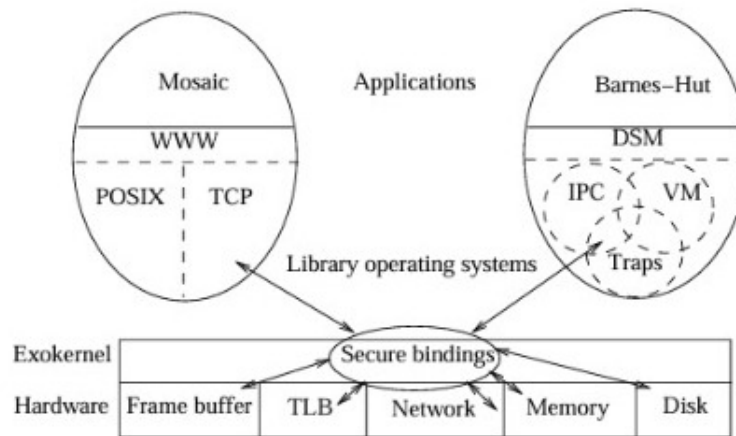


Figure 1.4: Sistema operativo basado en exokernel (Engler et al., 1995)

### 1.1.5 Comparacion entre arquitecturas de Sistemas Operativos

Table 1.1: Comparación de arquitecturas de SO

Arquitectura	Ventajas	Desventajas
Monolítica	Alto rendimiento, simple en diseño inicial	Difícil de mantener, poco modular
Microkernel	Modularidad, mayor seguridad	Mayor sobrecarga de comunicación
Hibrida	Combina rendimiento y modularidad	Complejidad de implementación
Exokernel	Máxima flexibilidad y control	Muy complejo, poco usado en producción

## Chapter 2

### Revision de S.O. existentes

## 2.1 Fuchsia



Figure 2.1: Icono de S.O. de Fuchsia (Fuchsia Project, 2025)

Fuchsia OS es un sistema operativo desarrollado por Google desde 2016, diseñado como una plataforma de propósito general para dispositivos embebidos, móviles, IoT y potencialmente de escritorio. A diferencia de Android, no se basa en Linux, sino en un núcleo propio denominado Zircon (Google Fuchsia Team, 2025b).

Su arquitectura está basada en un microkernel, en el que Zircon gestiona procesos, hilos, memoria y comunicación mediante objetos e IPC, mientras que servicios como sistemas de archivos y controladores se ejecutan en espacio de usuario, favoreciendo la seguridad y modularidad (Google Fuchsia Team, 2025a).

Fuchsia está escrito principalmente en C++, Rust y Dart. Zircon se implementa en C/C++, mientras que la capa de interfaz gráfica se desarrolla con Flutter (Dart), lo que facilita la portabilidad a múltiples plataformas (Google Inc., 2024).

Entre sus componentes clave se encuentran la gestión de procesos basada en objetos, memoria virtual con aislamiento estricto, soporte para múltiples sistemas de archivos (como MemFS, BlobFS y FAT) y un entorno gráfico con Flutter para experiencias multiplataforma (Google Fuchsia Team, 2025b).

El proyecto es mantenido principalmente por Google, con participación de la comunidad de código abierto. Su documentación oficial está disponible en Fuchsia.dev, que incluye guías técnicas, API y manuales de contribución, aunque parte del desarrollo se mantiene cerrado (Fuchsia Open Source Community, 2025).

## 2.2 Haiku



Figure 2.2: Icono de S.O. de Haiku (Haiku Project, 2025b)

Haiku es un sistema operativo de código abierto inspirado en BeOS, iniciado en 2001 con el objetivo de proporcionar un entorno moderno, rápido y sencillo de usar, orientado principalmente a equipos de escritorio (Haiku Project, 2025a).

Su arquitectura es híbrida, ya que utiliza un núcleo monolítico modular, pero con elementos que recuerdan a un microkernel, lo que equilibra rendimiento y flexibilidad (Haiku Project Developers, 2024). El sistema está desarrollado principalmente en C++, con partes en C y ensamblador para las interacciones de bajo nivel (Haiku Project Developers, 2024).

Entre sus componentes clave se incluyen un modelo de multitarea preventiva con planificación por prioridades, gestión de memoria virtual con paginación y aislamiento de procesos, y un sistema de archivos propio llamado OpenBFS, optimizado para indexación rápida (Haiku Project, 2023). Además, cuenta con una interfaz gráfica uniforme gestionada por el `app_server`.

Haiku es mantenido por la Haiku Project Foundation y su comunidad de desarrolladores voluntarios. El proyecto dispone de documentación oficial, foros y repositorios activos en GitHub, con lanzamientos beta periódicos que permiten probar y evaluar

su madurez (Haiku Project Community, 2025).

## Chapter 3

### Comparación técnica

# Bibliography

Engler, D. R., Kaashoek, M. F., and O'Toole, James, J. (1995). Exokernel: An operating system architecture for application-level resource management. In *Proceedings of the Fifteenth ACM Symposium on Operating Systems Principles (SOSP '95)*, pages 251–266, Cambridge, MA, USA. ACM.

Fuchsia Open Source Community (2025). Community involvement in fuchsia. Foros y repositorios. Accedido: 20 septiembre 2025.

Fuchsia Project (2025). Fuchsia — open source operating system. Sitio web oficial. Accedido: 20 septiembre 2025.

GeeksforGeeks (2024a). Microkernel in operating systems. Accedido: 20 septiembre 2025.

GeeksforGeeks (2024b). Monolithic architecture in operating systems. Accedido: 20 septiembre 2025.

Google Fuchsia Team (2025a). Fuchsia concepts and apis. Documentación técnica. Accedido: 20 septiembre 2025.

Google Fuchsia Team (2025b). Zircon kernel — fuchsia documentation. Documentación técnica. Accedido: 20 septiembre 2025.

Google Inc. (2024). Flutter and dart in fuchsia. Blog técnico de Google. Accedido: 20 septiembre 2025.

Haiku Project (2023). Openbfs — be file system in haiku. Documentación técnica. Accedido: 20 septiembre 2025.

Haiku Project (2025a). Haiku — official website. Sitio web oficial. Accedido: 20 septiembre 2025.

Haiku Project (2025b). Haiku — the open source operating system. Sitio web oficial. Accedido: 20 septiembre 2025.

Haiku Project Community (2025). Haiku community and development. Foros y repositorios. Accedido: 20 septiembre 2025.

Haiku Project Developers (2024). Haiku developer documentation. Documentación técnica. Accedido: 20 septiembre 2025.

Harshvardhan and Irabatti, S. (2023). Study of kernels in different operating systems in mobile devices. *Journal of Online Engineering Education*, 14(1s). Article received: 29 January 2023; Revised: 24 March 2023; Accepted: 20 April 2023.

Keetmalin (2017). An introduction on exokernel operating systems. Blog post. Accedido: 20 septiembre 2025.

Microsoft Wiki / Fandom (2024). Hybrid kernel. Accedido: 20 septiembre 2025.