**Programación Orientada a Objetos**

**Trabajo Integrador**



**Ingeniería en Mecatrónica – Facultad de Ingeniería – Ciclo Lectivo 2024**

**Profesores**

Mg. Ing. César Omar Aranda

Ing. Facundo Martín

**Alumnos**

Calderón Dal Pozzo, Joaquín. Legajo N° 13839

De Miguel Funes, Juan Marcos. Legajo N° 13673

Huertas Coppo, Juan Francisco. Legajo N° 12620

Rayes Cano, Julián Andrés. Legajo N° 13256

Índice

[Introducción 3](#_Toc181020440)

[Cuerpo 3](#_Toc181020441)

[Conceptos 3](#_Toc181020442)

[Componentes de Software 3](#_Toc181020443)

[Descripción 3](#_Toc181020444)

[Diagrama de Clases 3](#_Toc181020445)

[Diagrama de Secuencia 3](#_Toc181020446)

[Diagrama de Actividad 3](#_Toc181020447)

[Desarrollo 3](#_Toc181020448)

[Capturas de pantalla 3](#_Toc181020449)

[Módulo y reporte de Prueba Unitaria 3](#_Toc181020450)

[Conclusiones y comentarios 3](#_Toc181020451)

[Comentarios 3](#_Toc181020452)

[Ventajas y desventajas 3](#_Toc181020453)

[Extensiones 3](#_Toc181020454)

[Referencias 3](#_Toc181020455)

# Introducción

Este informe tiene por objetivo detallar un sistema bajo el paradigma de la Orientacion a Objetos para el control remoto de un robot de tres grados de libertad, implementado en los lenguajes Python y C++. Se tiene por objetivo aplicar los conceptos fundamentales del paradigma de la Programación Orientada a Objetos (POO) para la creación de una aplicación que permita manipular precisamente el robot en comandos de tipo G-Code.

El proyecto abarca el diseño de un servidor en Python y de un cliente en C++. La lógica de control del robot se encuentra implementada en el servidor, y éste proporciona una interfaz de acceso remoto RPC (Remote Procedure Call). Por otro lado, el cliente, desarrollado en C++ proporciona una interfaz de usuario que permite enviar órdenes al robot y monitorear su estado.

La estructura del proyecto sigue una arquitectura de tres capas: modelo, vista y controlador (MVC), repartiendo la lógica en componentes que gestionan datos, interfaz de usuario y flujo de control respectivamente. Esto permite que la solución se pueda adaptar a futuras implementaciones con interfaces gráficas o ampliar su capacidad con agregados externos sin afectar la estabilidad del funcionamiento básico.

Además del control del robot, se incluye un sistema de registro de actividades, donde cada acción realizada es almacenada en un archivo CSV, permitiendo la trazabilidad de operaciones, así como un archivo de usuarios y contraseñas que asegura que sólo personal autorizado pueda controlar el robot.

Las principales funcionalidades abarcan desde la activación y desactivación de motores y del efector final hasta la ejecución de movimientos en modo manual y automático. En modo manual, el usuario envía comandos y en modo automático se cargan y ejecutan archivos en G-Code con las secuencias de trabajo, permitiendo automatizar tareas repetitivas.

A lo largo de este trabajo se detallará la aplicación de la solución escalable y adaptable para el control remoto de un robot.

# Cuerpo

## Conceptos

Arquitectura Cliente-Servidor

Este proyecto se basa en el modelo cliente-servidor, una arquitectura en la que el cliente solicita servicios al servidor, quien los ejecuta y responde. En este caso, el cliente, codificado en lenguaje C++ se conecta al servidor, codificado en lenguaje Python mediante XML-RPC para enviar comandos que controlen al robot

La importancia de esta arquitectura radica en la separación de responsabilidades, donde el cliente maneja la interfaz del usuario cliente y el servidor gestiona la lógica de control y comunicación con el hardware del robot.

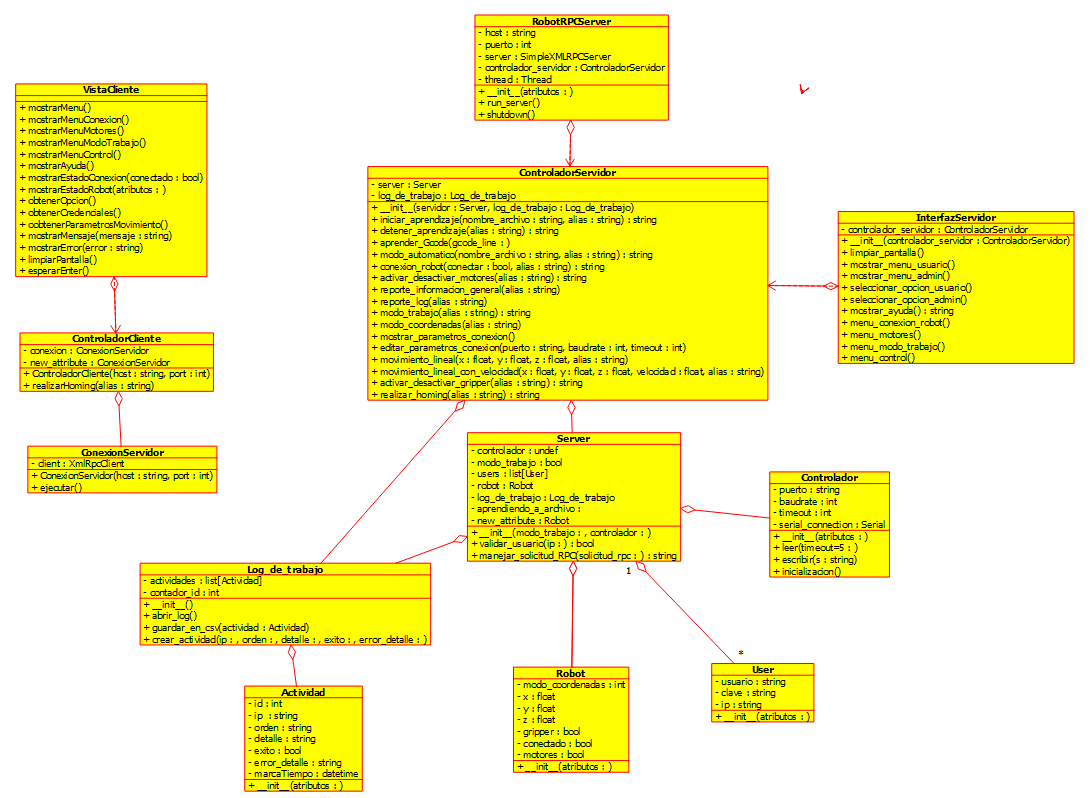
Múltiples hilos

En este trabajo, el servidor implementa múltiples hilos (multithreading) para manejar la comunicación con el cliente y gestionar las operaciones del robot en paralelo. El servidor puede escuchar comandos mientras mantiene otros procesos activos, esencial para un funcionamiento fluido.

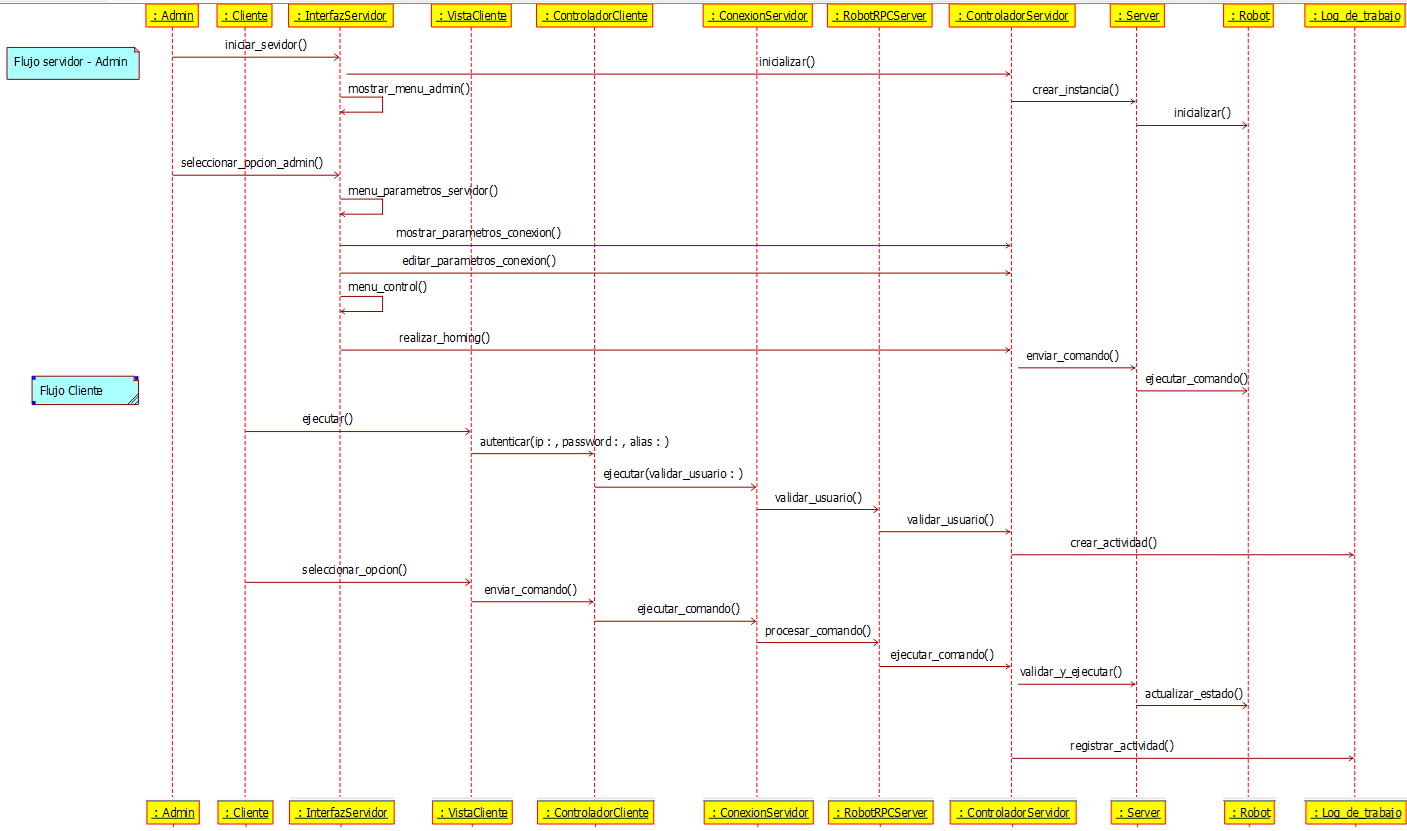
## Componentes de Software

## Descripción

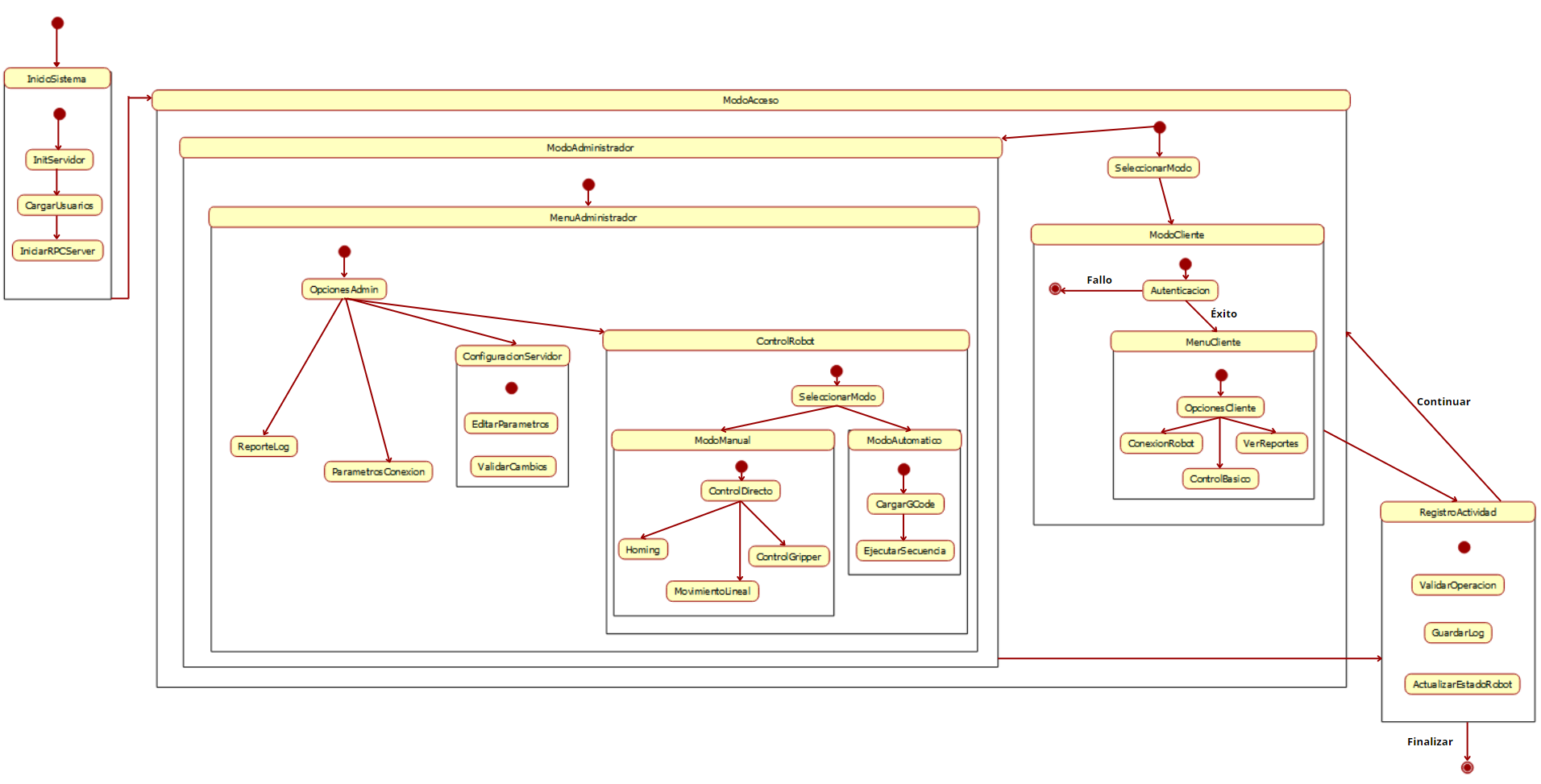
## Diagrama de Clases



## Diagrama de Secuencias



## Diagrama de Actividad



## Desarrollo

## Capturas de pantalla

## Módulo y reporte de Prueba Unitaria

# Conclusiones y comentarios

## Comentarios

## Ventajas y desventajas

## Extensiones

# Referencias

<https://www.geeksforgeeks.org/client-server-model/>

<https://docs.python.org/3/library/threading.html>