Programación Orientada a Objetos

Trabajo Practico Integrador

Ingeniería Mecatrónica

Profesor: Esp. Ing. César Omar Aranda

Integrantes:

- **Bertani**, Matías - 11222
- **Bruno**, Simón - 121777

Introducción

Consigna

En este informe haremos la exposición del trabajo práctico integrador de Programación Orientada a Objetos. Se mostrará brevemente los distintos aspectos teóricos y de diseño involucrados en el desarrollo, además de las dificultades que fueron surgiendo durante el desarrollo del código y algunas conclusiones y observaciones realizadas. El objetivo de la tarea era realizar un aplicativo para controlar un robot de 3 grados de libertad, similar a un ABB IRB 460 con una pistola de pintura como efector final.

Especificaciones

El protocolo de transmisión de datos especificado para la realización del trabajo es XML-RPC y debe comunicar un cliente desarrollado en C++ con un

servidor desarrollado en Python. La consigna especificaba también la implementación de un panel de control de servidor a modo de cmd y un panel de control similar pero más simple para el cliente.

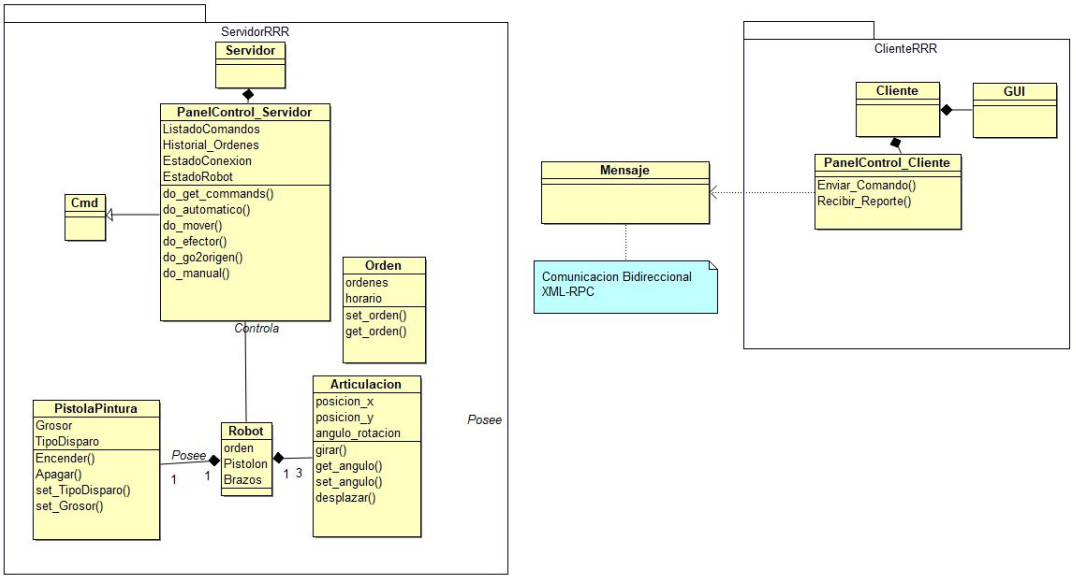
Marco Teórico

Boceto de esquema

La primera instancia del proyecto consistió en realizar un esquema de clases con atributos y métodos en BOUML. Esta parte es fundamental para una comprensión amplia de los alcances del trabajo, ya que establecer todas las relaciones entre clases desde los momentos iniciales de trabajo facilita el desarrollo de todo lo que sigue y ahorra mucho tiempo de confusión. Aquí los puntos más importantes:

- La clase Robot hace instanciaciones de las articulaciones y del efector para representar el hecho de que el robot real está compuesto por esos elementos.
- El Panel de Control del lado servidor, heredando los métodos y atributos de la clase cmd, controla directamente al robot.
- Una clase de comunicación tanto para servidor como para cliente, que comunicará al panel de control que corresponda.
- Dos clases englobadoras de cliente y servidor para contener los paneles de control y las clases de comunicación

Esquema de BOUML



Cabe aclarar que la esquematización es una etapa original y es muy susceptible a cambios y modificaciones a medida que se avanza con el proyecto.

Desarrollo

Etapas de desarrollo

Para lograr la consiga, tuvimos que utilizar el módulo de XML-RPC disponible en python para la creación de servidores, mientras que en C + + utilizamos los archivos disponibles en internet de clases para el cliente. La primera parte de programación del proyecto consistió básicamente en adaptar las clases heredadas tanto en python como en C + + para poder conseguir el

flujo de datos. Por esta razón, los problemas más frecuentes en esta instancia fueron de sobreescritura y polimorfismo en la mayor parte de las clases utilizadas. Además hubo que ajustar algunos conceptos e ideas de la teoría de comunicación por XML-RPC para poder saber qué variables y argumentos era necesario manipular.

Una vez lograda la comunicación, pudimos empezar a generar nuestras propias clases para este robot específico y así ir implementando nuevas funciones en el cmd heredado en python y registrándose en C + + para poder utilizarlas desde el cliente. Mantener siempre presente la organización establecida en BOUML fue fundamental para insertar nuevas clases y métodos en la configuración original.

La parte final del trabajo consistió en ajustar el traspaso de los mensajes de retorno entre objetos con las respectivas variables devueltas como argumento de la comunicación. También fue frecuente la tarea de arreglar detalles en la llamada a todas los métodos y chequear su correcto funcionamiento.

Estructuración de clases

La estructura final de clases dentro del código nos quedó de la siguiente manera:

- Servidor
    - Panel de Control

        - Robot
            - 3 articulaciones y una base para desplazarse
            - Efector Final
        - Orden (clase de registro de órdenes y horario)
    - Configuración de servidor para XML-RPC
- Cliente
    - Configuración de cliente para XML-RPC
    - Panel de Control

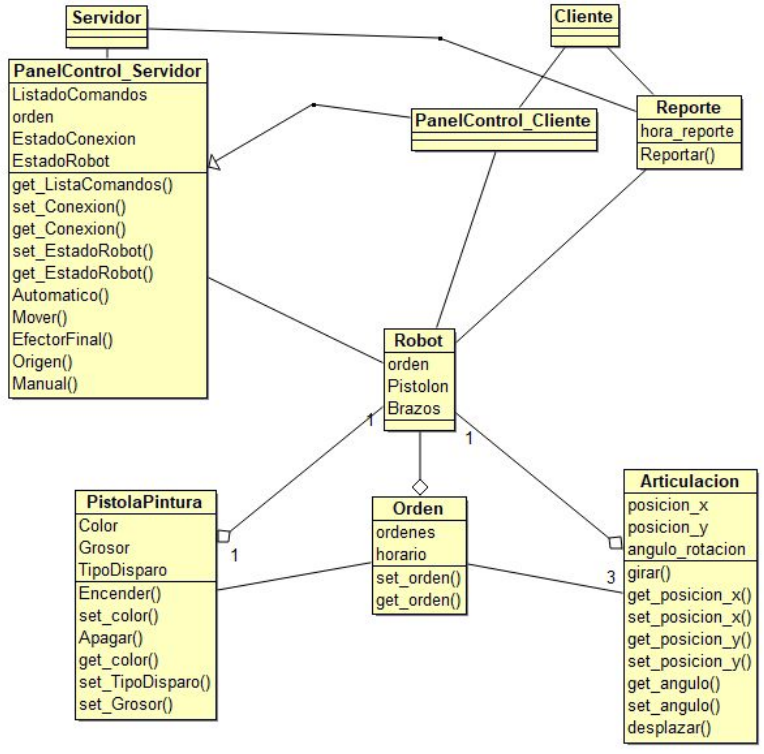
Comentarios y Conclusiones

Opiniones

El proyecto realizado durante el segundo semestre del 2020 tuvo la particularidad de estar acotado a un cursado de 1 mes y medio, por lo cual hubo que adaptarse muy rápido a cualquier cambio o modificación que hiciera falta. Además, creemos que estos temas requieren más tiempo de estudio y análisis para conseguir un conocimiento más acabado. Aun así, consideramos que hemos alcanzado satisfactoriamente los objetivos planteados a principio del curso tanto por el profesor como personalmente. Pudimos agrupar todos los conocimientos aprendidos durante el periodo de introducción y en el cursado oficial, para luego aplicarlos al desarrollo de esta interfaz de comunicación.

Conclusiones de aprendizaje

Una correcta comprensión de la consigna se ve reflejada en una buena estructuración de las clases en BOUML, y nosotros nos dimos cuenta que durante el tiempo de la primera entrega no teníamos muy claro qué hacer ni mucho menos cómo hacerlo. Esto se ve reflejado en el primer esquema que realizamos. Acá encontramos el primer obstáculo y tuvimos que pensar varias cosas antes de poder superarlo, pero notamos la ventaja de haberle dedicado suficiente tiempo a la planificación del código.



A lo largo del proyecto pudimos palpar las ventajas y las propias características que ofrece el Paradigma Orientado a Objetos, por un lado el

poder que supone la herencia y la sobreescritura al facilitar la creación de objetos con mucha funcionalidad y de manera rápida sin preocuparse del "cómo" interno. Esto lo pudimos apreciar al crear el Panel de Control en python aprovechando la clase cmd y heredando de ella todas sus funcionas y solo reescribiendo aquellas que quisiéramos de determinada manera, contrastando esto con el CLI hecho en C++, que nos llevó más tiempo de programación y resultó ser mucho más acotado y más cansador a la hora de realizar modificaciones. De todas maneras, hay que saber distinguir las utilidades de cada lenguaje para no sacarlos de su entorno de funcionamiento natural.

Concluyendo, este proyecto fue muy provechoso para tener un primer contacto estrecho con la programación orientada a objetos, pudiendo visualizar un nuevo paradigma de códigos y adoptarlo como herramienta para nuestro desarrollo futuro.

Referencias

Dudas con C + + y Python: Cpp Reference, Stack Overflow.

Videos subidos a Aula Abierta.