MUVit

(Brazo Mecánico controlado Remotamente)

Universidad de Cuenca

*Facultad de Ingeniería*

Escuela de Sistemas

|  |  |
| --- | --- |
| INTEGRANTES | RESPONSIBILIDAD DENTRO DEL PROYECTO |

(1)

|  |  |
| --- | --- |
| Freddy Leonardo Abad León | Implementación de la comunicación interprocesos MoM vía Mosquitto MQTT (Publicador y Suscritor) y vía ZeroMQ (p2p) |
| Edisson Santiago Reinozo Trelles | Codificación de las clases usadas para la implementación del brazo mecánico conjuntamente con la construcción de la maqueta 3d, y empaquetamiento de datos para el envío por los canales de comunicación. |
| Jaime Paúl Arévalo Armijos | Asesoría en el control del tráfico (vía WireShark) e implementación de comunicación MoM vía RabbitMQ (p2p) |

(2) HERRAMIENTAS USADAS

*Sublime Text:* (Editor de texto): Editor de texto y editor de código fuente escrito en C++ y Python para los plugins.

*Raspberry PI*: Computador de placa reducida o de placa simple (SBC) de bajo costo que estimula la enseñanza de ciencias de la computación.

*Thonny:* IDE de Python para principiantes. Compatible con diferentes formas de recorrer el código, evaluación de expresiones paso a paso, visualización detallada de la pila de llamadas y un modo para explicar los conceptos de referencias y montón.

*PyCharm:* IDE utilizado en la programación para el lenguaje Python. Está desarrollado por la compañía checa JetBrains. Proporciona análisis de código, un depurador gráfico, un probador de unidades integrado, integración con sistemas de control de versiones.

*Eclipse*: Plataforma de software compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores.

TECNOLOGÍAS USADAS

*Python* *3.5:* Lenguaje de programación en una de sus últimas versiones de desarrollo, que permite el “scripting” necesario en el proyecto.

*MQTT*: Protocolo de mensajería basado en suscripción-publicación estándar y p2p. Funciona sobre el protocolo TCP/IP. Diseñado para conexiones con ubicaciones remotas donde se requiere una "huella de código pequeña" o el ancho de banda de la red es limitado. El patrón de publicación publicación-suscripción requiere un intermediario de mensajes.

*WireShark*: analizador de protocolos utilizado para realizar análisis y solucionar problemas en redes de comunicaciones, para desarrollo de software y protocolos, y como una herramienta didáctica. Cuenta con todas las características estándar de un analizador de protocolos de forma únicamente hueca.

LIBRERIAS USADAS PARA MANIPULAR EL BRAZO

*RPi.GPIO:* Módulo para controlar los canales GPIO de Raspberry Pi

*Xmlrpclib*: XML-RPC es un método de llamada a procedimiento remoto que utiliza XML pasado a través de HTTP (S) como transporte.

LIBRERIAS ESENCIALES PARA LA COMUNICACIÓN PC-BRAZO

*RabbitMQ*: Librería esencial para la comunicación (S.O. : Linux). Este es un software de negociación de mensajes de código abierto, y entra en la categoría de middleware de mensajería. Implementa el estándar Advanced Message Queuing Protocol (AMQP). Middleware de paso de mensajes. Es usado al momento de crear las colas, establecer las conexiones y el envío de mensaje. Es empleado en las clases Send & Receive. Es un intermediario de mensajes: acepta y reenvía mensajes. Se asemeja a una oficina de correos: cuando coloca el correo que se publicara en un buzón, el cartero eventualmente entregará el correo a su destinatario. En esta analogía, RabbitMQ es un buzón, una oficina de correos y un cartero.

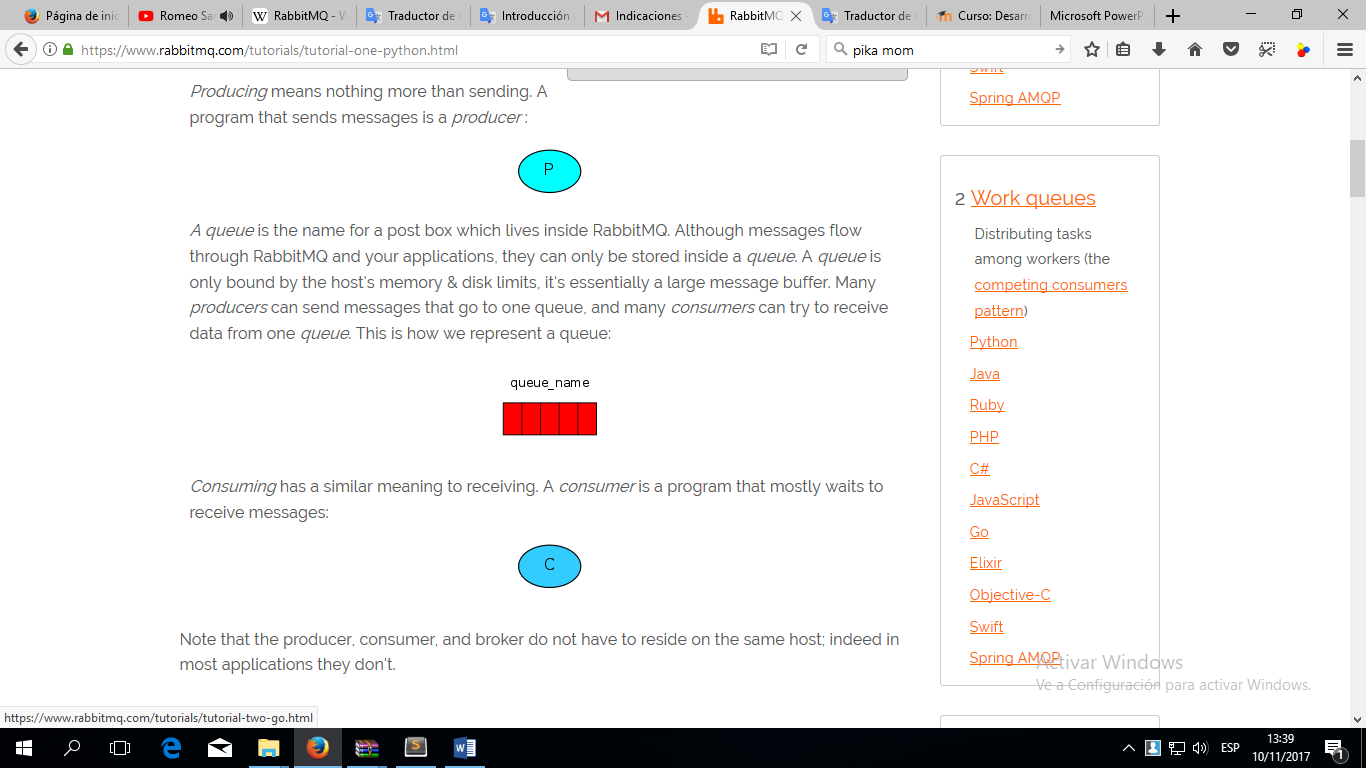
*PIKA*: Implementación pura de Python del protocolo AMQP 0-9-1 que intenta mantenerse bastante independiente de la biblioteca de soporte de red subyacente. Se usa complementando

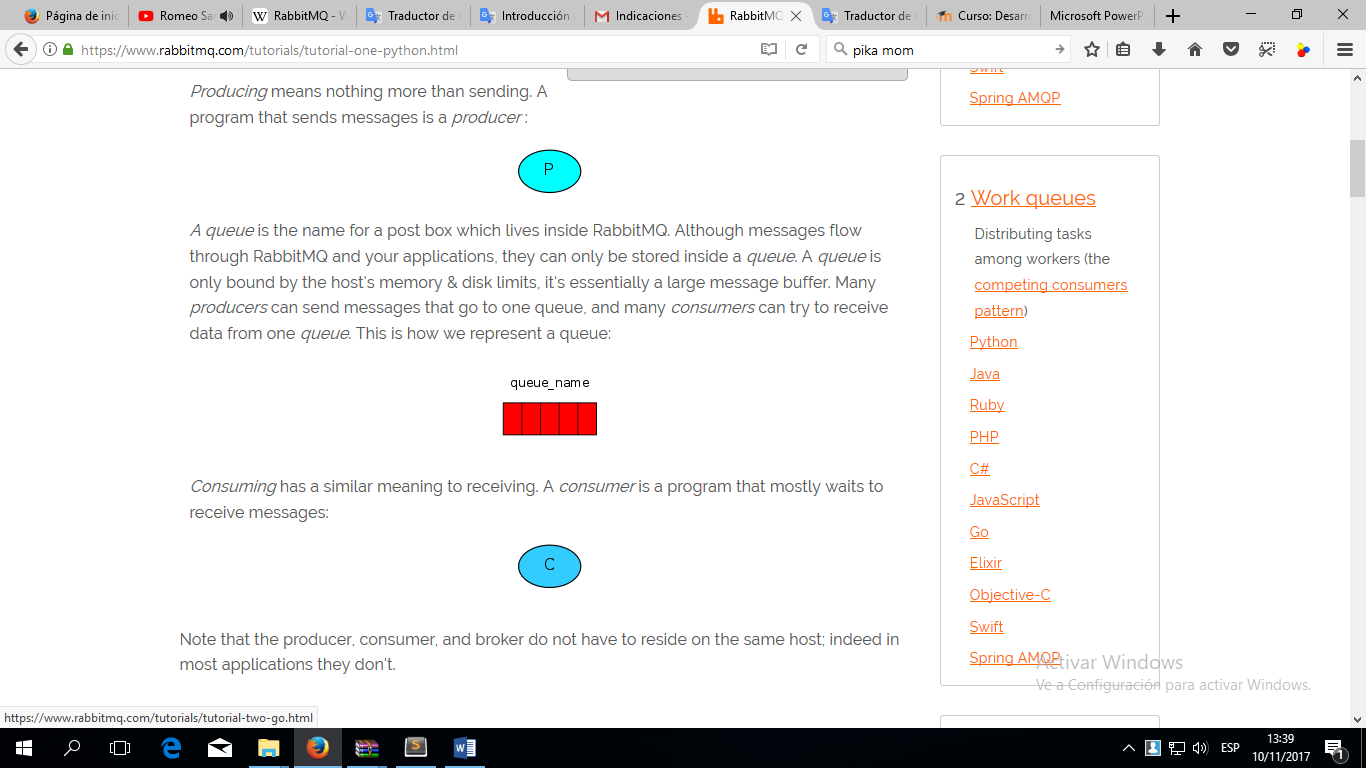
*Mosquitto*: Intermediario de mensajes de código abierto implementa el protocolo MQTT. Proporciona un método liviano para llevar a cabo mensajes utilizando un modelo de publicación / suscripción. Ejemplos de mensajería en "Internet de las cosas": sensores de baja potencia o dispositivos móviles como teléfonos, computadoras integradas o micro controladores.

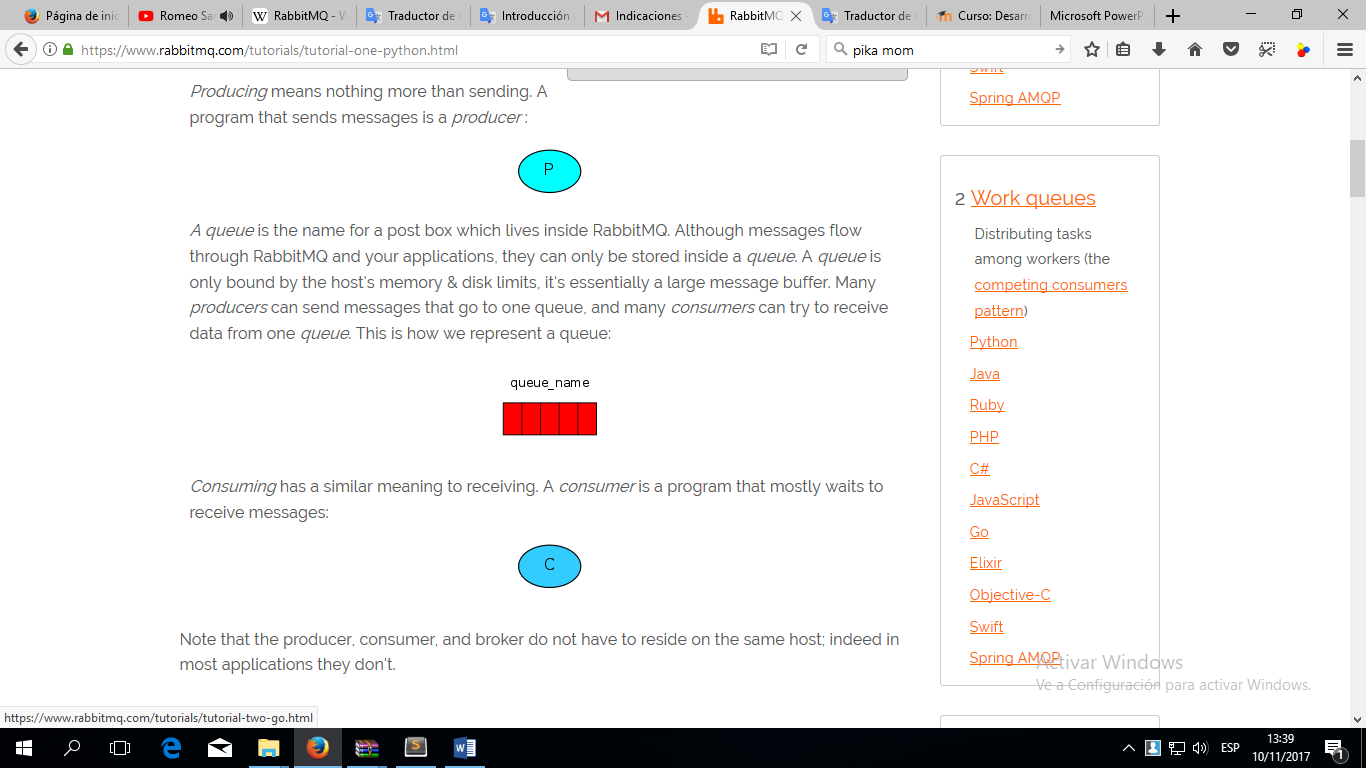
(3) IMPLEMENTACION

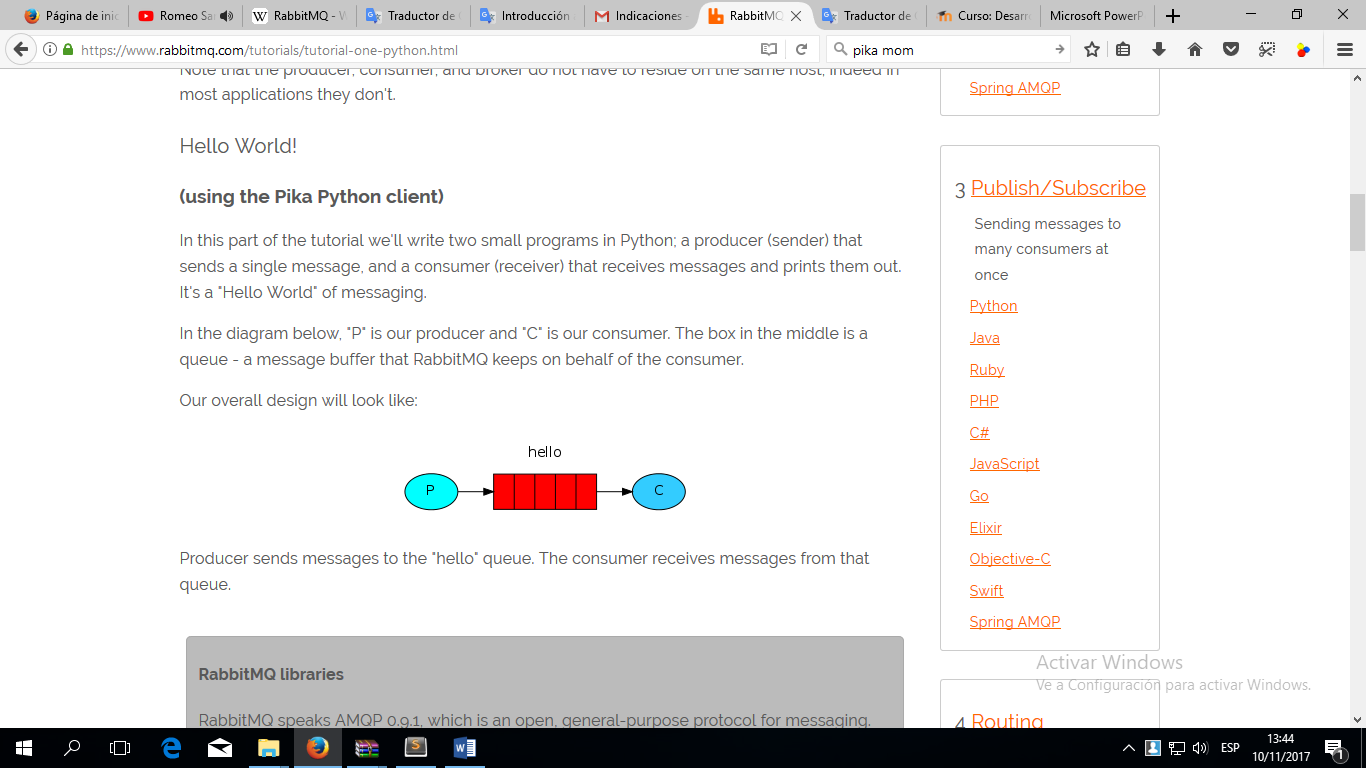
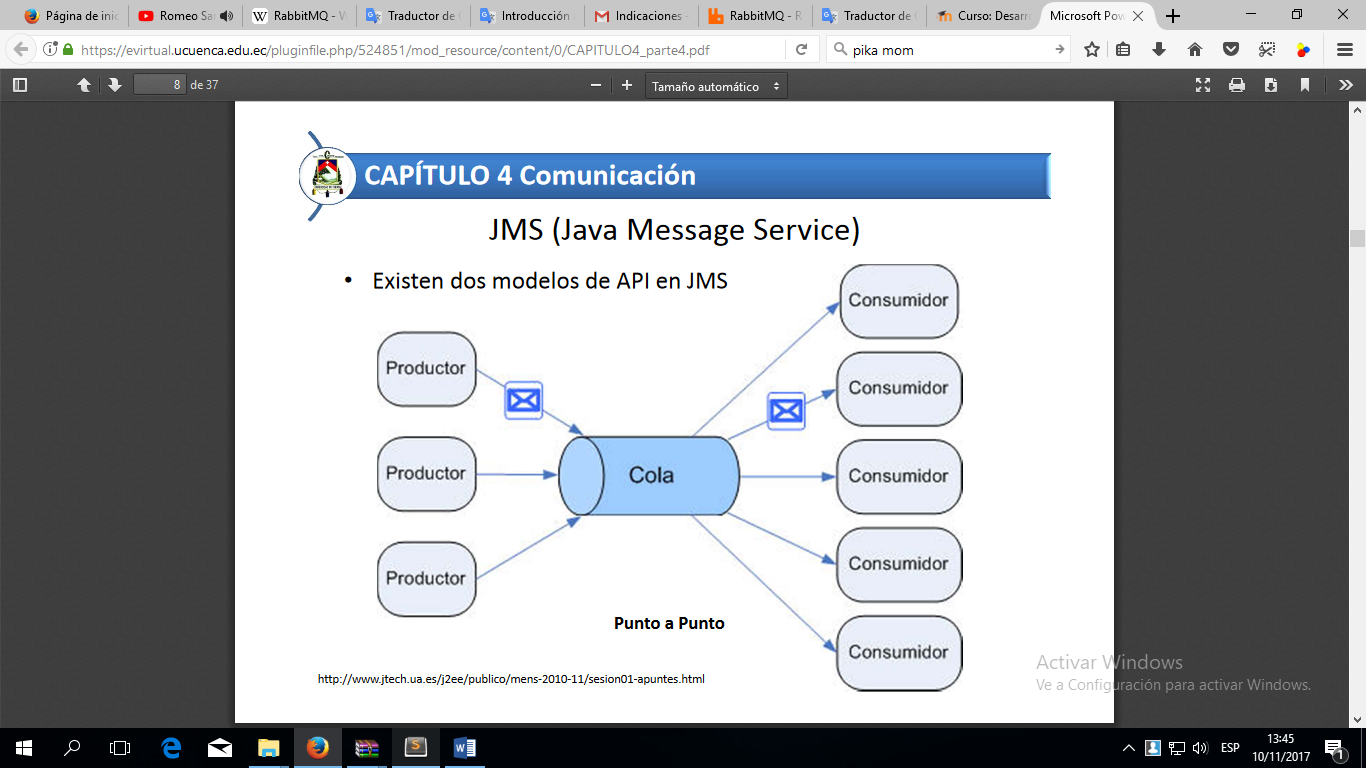
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CLASE | DESCRIPCION | FUNCION | DESARROLLO / REUSO | NODO |
| SEND | Algoritmo que conecta con receive.py.  Requiere acción al usuario. | * Estable conexión. * Escanea datos de ingreso del Usuario. * Apunta a host virtual. * Apunta a una cola de mensajes. * Hace persistente al mensaje. | DESARROLLO | Cliente (PC) |
| RECEIVE | Algoritmo que establece una conexión con send.py y permanece atento a recibir mensajes. | * Establece conexión. * Crea cola de mensajes. * Implementa la librería Pika. * Envía mensaje de respuesta. * Llama a métodos para manipular de brazo. | DESARROLLO | Servidor (Raspberry Pi) |
| BRAZO | Implementa métodos para la manipulación del brazo mecánico. | * Inicializa todos los pines de control de los servomotores. * Limpia los pines de salida. * Transforma el Angulo (sexagesimal) en una potencia para imprimir en cada acción requerida. * Implementa: moverBase, moverCodo, moverBrazo, moverPinza, los cuales reciben ángulo y tiempo, los convierte en potencia, conecta a los pines, los hace correr y los desconecta. | DESARROLLO | Servidor (Raspberry Pi) |

4. Grafique conceptualmente el método asignado

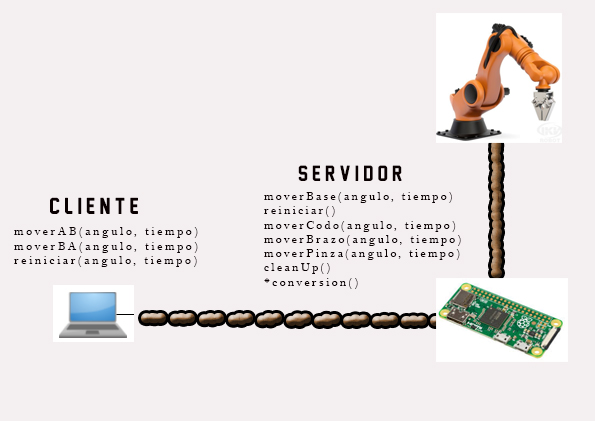
 : Productor - Un programa que envía mensajes

Buzón - Buzón de RabbitMQ, los mensajes fluyen a través de RabbitMQ y sus aplicaciones, pero se almacenan dentro de una cola. Una cola solo está limitada por los límites de memoria y disco del host (gran buffer de mensajes). Muchos productores pueden enviar mensajes que van a una fila, y muchos consumidores pueden tratar de recibir datos de una cola.

 : Consumidor - Programa que espera recibir mensajes

5. Grafique cómo el método de comunicación asignado soporta su proyecto



COLA

receiver.py

send.py

INTEGRANTES

Freddy L. Abad León \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Édison S. Reinozo Trelles \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Jaime P. Arévalo Armijos \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_