

DESARROLLO DEL SOFTWARE DE UN BRAZO MECÁNICO

CURSO 2021/2022



AUTOR/AUTORES

Daniel López López y Daniel González Hernansaiz

Contenido

[1. INTRODUCCIÓN 2](#_Toc90204848)

[1.1 OBJETIVO 2](#_Toc90204849)

[1.2 ALCANCE 2](#_Toc90204850)

[1.3 JUSTIFICACIÓN 2](#_Toc90204851)

[2. IMPLEMENTACIÓN 2](#_Toc90204852)

[2.1 ANÁLISIS DEL BRAZO MECANICO 2](#_Toc90204853)

[2.1.1 información sobre el brazo 2](#_Toc90204854)

[1.1.1 Configuracion 4](#_Toc90204855)

[1.1 ANÁLISIS DE LA APLICACIÓN 9](#_Toc90204856)

[1.2 DISEÑO 10](#_Toc90204857)

[1.3 IMPLEMENTACIÓN 10](#_Toc90204858)

[1.4 IMPLANTACIÓN 10](#_Toc90204859)

[1.5 DOCUMENTACIÓN 10](#_Toc90204860)

[2. RESULTADOS Y DISCUSIÓN 10](#_Toc90204861)

[3. CONCLUSIONES 10](#_Toc90204862)

[4. BIBLIOGRAFÍA 10](#_Toc90204863)

[5. ANEXOS 11](#_Toc90204864)

# INTRODUCCIÓN

Desarrollo de un software Cliente/Servidor para controlar un brazo mecánico a través de un puerto Serie.

Este proyecto se ha desarrollado como parte de la formación del CFG

## OBJETIVO

Se pretende obtener conocimiento básico del funcionamiento y control del brazo mecánico y desarrollar una aplicacion de escritorio que sea capaz de comunicarse mediante un puerto COM al brazo mecanico de Mitsubishi Movemaster de modelo RV-M1 para poder realizar movimientos.

## ALCANCE

Se pretende realizar una aplicación que dispondrá de una parte servidor que funcionara como controlador del programa y recibirá las peticiones de la parte cliente y una parte cliente que funcionara como interfaz del programa y que nos permitirá tanto recibir la información del brazo robótico como enviar peticiones para su movimiento a la parte servidor, ambas partes se pretende realizar en java.

Como parte opcional también se pretende realizar la parte de la interfaz de cliente en Android.

## JUSTIFICACIÓN

Hemos decidido realizar este proyecto para renovar un software de un brazo mecánico, el proyecto consistirá en realizar una aplicación funcional para Windows 10 que sea capaz de controlar un brazo mecánico Mitsubishi Movemaster de modelo RV-M1.

Ahora mismo el desarrollo de aplicaciones esta en alza ya que después de la situación del COVID-19 muchas empresas están requiriendo de mejores softwares o nuevos para un mejor funcionamiento de la empresa. Este proyecto es un ejemplo de una renovación de un software antiguo por uno más moderno.

# IMPLEMENTACIÓN

## ANÁLISIS DEL BRAZO MECANICO

## información sobre el brazo

El robot en el que nos hemos centrado para nuestro trabajo de fin de grado es el Mitsubishi MovemasterEx modelo RV-M1, dicho robot consta de 5 movimientos articulados más la apertura/cierre de la pinza y adicionalmente va montado sobre un carril que le da un movimiento extra pero que no se ha tenido en cuenta durante este TFG.

El robot tiene distintos componentes para su control:

* Brazo articulado.
* Teaching Box o el mando de movimientos.
* Controlador.
* Ordenador con el software para establecer conexión con el robot.

Diagrama

Descripción generada automáticamente

En brazo mecánico dispone de las 5 articulaciones antes mencionadas siendo todas ellas rotacionales, en la siguiente imagen podremos verla con mas detalle todas ellas, deberemos tener en cuenta esto en un futuro para la programación de la aplicación ya que nos centraremos en el movimiento de todas ellas.

Diagrama, Dibujo de ingeniería

Descripción generada automáticamente

A continuación tendremos dos tablas una para hacer referencia a los nombres de las articulaciones del brazo mecánico y otra para ver los grados que puede moverse una articulación tanto positivamente como negativamente.

|  |  |
| --- | --- |
| J1 | Cintura (Waist). |
| J2 | Hombro (Shoulder). |
| J3 | Codo (Elbow) |
| J4 | Inclinación de la muñeca (Pitch). |
| J5 | Giro de la muñeca (Roll). |

Tabla con las referencias a los nombres del brazo mecánico.

|  |  |  |  |
| --- | --- | --- | --- |
| J1 | -150º | A | 150º |
| J2 | -30º | A | 100º |
| J3 | 0º | A | 110º |
| J4 | -90º | A | 90º |
| J5 | -180º | A | 180º |

Tabla con las referencias a los valores máximos y mínimos que puede tomar cada articulación.

## Configuración

En cuanto a la configuración del controlador deberemos de tener varios aspectos en cuenta ya que deberá de coincidir con la siguiente información ya que de lo contrario nuestro programa no será capaz de conectar o enviar la información de forma óptima al robot.

En la parte lateral de la controladora veremos varios interruptores:

**Interruptor ST1**: Este switch establece el control del robot, si se encuentra en la posición superior el control lo posee el microprocesador del controlador mientras que si esta en la posición inferior será el programa del ordenador el que se encargara del control, por lo que este interruptor deberá esta hacia abajo.

**Interruptor ST2:** Si este switch esta en su posición superior la información de la EPROM los datos son transferidos a la RAM de la controladora mientras que si se encuentra en la posición posición inferior la información no será transmitida a la memoria RAM, en nuestro caso mantendremos el interuptor en su posición inferior ya que no haremos uso de la RAM.

Justo encima de los interruptores que acabamos de mencionar veremos que hay otros tres interruptores los cuales tendrán cada uno 8 bits.

**Interruptor SW1:** Se compone por ocho bits y cada bit tendrá un propósito que se montrara en la siguiente tabla.

|  |  |
| --- | --- |
| Bit 1 | Selecciona la terminación para la transmisión de datos desde la conexión RS-232-C. Para la posición superior: CR + LF (“\r” + “\n”) y para la posición inferior: CR (“\r”). Se pondrá en la posición inferior (a no ser que se esté usando MULTI16). |
| Bit 2 | Indica si se revisa o no que los contenidos de la memoria RAM son retenidos cuando no está encendido; si se encuentran en la posición superior se revisa. Si en cambio se encuentra en la posición inferior, no se revisa. Se pondrá en la posición inferior si no se usa la batería. |
| Bit 3 | Selecciona el tipo de tarjeta de entrada/salida usada; para la posición superior es del Tipo A16 o B16 y si está la posición inferior: es del Tipo A8 o B8. |
| Bit 4 | Selecciona si se establecen, se cambiar o se borrar las referencias a las posiciones de los datos en un sistema de coordenadas cartesiano, en la posición superior se activa mientras que en la posición se desactiva. Activaremos esta opción cuando hayamos establecido un sistema de referencia en coordenadas cartesianas y queramos enviar los datos escritos en la EPROM a la RAM. |
| Bit 5 | Selecciona entre el uso de los interruptores de la parte frontal del controlador o el de las señales externas para cargar el programa mientras la tarjeta de tipo A16 o B16 está siendo usada; la posición superior selecciona las señales externas. Mientras que la posición inferior señala los interruptores de la parte frontal del controlador. Si se usa una tarjeta A8 o B8 deberá ponerse en la posición inferior |
| Bit 6 | Nos permite desactivar la tecla ENT del “teaching box” para liberar los frenos del robot. Normalmente, este bit se encontrará en su posición inferior. |
| Bit 7 | No se usa. |
| Bit 8 | Permite activar o desactivar la alarma sonora. En la posición superior se emite un pitido cuando ocurre un error mientras que en la posición inferior no se emite sonido alguno al haber error. |

**Interruptor SW2:** Establece el formato para la transmisión de datos asíncrona.

**Interruptor SW3:** Establece la tasa de baudios, es decir, la velocidad media de transferencia que va a utilizar el controlador.

A continuación podremos ver las posiciones con las que nuestra aplicación funciona.

Imagen en blanco y negro

Descripción generada automáticamente con confianza baja

Con esta combinación, los parámetros del robot son los siguientes:

* Secuencia de terminación: CR + LF (“\r” + “\n”).
* 4800 baudios.
* 7 bits de datos.
* Paridad par.
* 1 bit de parada.

En cuanto al mando o la teaching box deberá de esta apagado en caso de que queramos poder realizar los controles mediante la aplicación.

Comandos del robot.

A continuación, mostraremos todos los comandos que se le podrán pasar al robot mediante la parte de macros del programa, la información estará dividida en secciones dependiendo se su tipo de comando.

**Control de posición y movimiento.**

|  |  |
| --- | --- |
| **DP** | A partir de la posición actual, el robot se mueve a la anterior posición definida |
| **DW** | DW < distancia en x >, < distancia en y >, < distancia en z >.  Conservando la orientación, el robot desplaza la mano desde el punto en el que se encuentra, hasta un nuevo punto a una distancia determinada por los parámetros anteriores, en los ejes X, Y y Z |
| **HE** | HE < número de la posición>.  Guarda la posición actual asignándole el número suministrado como parámetro. Debe cumplirse que: 1 < número de la posición |
| **HO** | Establece la posición de referencia en el sistema de coordenadas cartesianas |
| **IP** | Lleva el robot a la siguiente posición definida. |
| **MA** | MA < posición 1>, < posición 2>, < O ó C >.  Mueve el robot a la posición que se obtiene al sumar las componentes de las posiciones 1 y 2. Estas componentes son las coordenadas X, Y y Z, en las que se encuentra la mano y los ángulos correspondientes a pitch y roll, que determinan su orientación. |
| **MC** | MC < posición 1 >, < posición 2 >.  El robot se mueve en forma continua entre la posición 1 y posición 2, pasando a través de las posiciones intermedias que hayan sido declaradas. |
| **MJ** | MC < posición 1 >, < posición 2 >.  El robot se mueve en forma continua entre la posición 1 y posición 2, pasando a través de las posiciones intermedias que hayan sido declaradas. |
| **MO** | MO< posición >, < O ó C >.  Movimiento a la posición seleccionada con la mano abierta o cerrada (O ó C). |
| **MP** | MP < coord. X >, < coord. Y >, < coord. Z >, < ángulo Pitch >, < ángulo roll > (\*). Mueve la mano del robot al punto dado por las coordenadas X, Y y Z, con orientación definida por los ángulos pitch y roll. |
| **MS** | MS < posición >, < número de puntos intermedios >, < O ó C >  Genera movimiento desde la posición actual hasta la nueva posición pasando través de un número definido de puntos intermedios. |
| **MT** | MT < posición >, < distancia >, < O ó C >  Movimiento en dirección de la herramienta a partir de la posición dada, a lo largo de la distancia definida. |
| **NT** | NT Lleva el robot a su origen mecánico. |
| **OG** | OG Lleva el robot a la posición de referencia en el sistema de coordenadas cartesianas establecido por el comando HO. |
| **PA** | PA < número de la paleta >, < número de columnas >, < número de filas >  Define el número de puntos de red tanto en filas como en columnas para una determinada paleta o rejilla. |
| **PC** | PC < posición 1 >, < posición 2 >  Borra las posiciones definidas en el intervalo comprendido entre la posición 1 y la posición 2. |
| **PD** | PD < posición >, , < coord. Y >, < coord. Z >, < Pitch >, < Roll > (\*)  Crea una posición en las coordenadas dadas y con la orientación definida por los ángulos pitch y roll |
| **PD** | PD < posición >, , < coord. Y >, < coord. Z >, < Pitch >, < Roll > (\*) Crea una posición en las coordenadas dadas y con la orientación definida por los ángulos pitch y roll |
| **PT** | PT < número de paleta >  Calcula las coordenadas de una red de puntos de una paleta e identifica las coordenadas de una posición con una paleta determinada. |
| **PX** | PX< posición 1 >, < posición 2 >  Asigna la posición 1 a la posición 2 y viceversa. |
| **SF** | SF < posición 1 >, < posición 2 >  Asigna a la posición 2 la suma de las coordenadas y ángulos de las posiciones 1 y 2 |
| **SP** | SP < nivel 0 a 9 >, < H o L >  Define la velocidad de movimiento del robot y su aceleración, que puede ser alta (H) o baja (L). |
| **TI** | TI < contador de 0 a 327ó7 >  Espera un periodo de tiempo en segundos equivalente al valor del contador dividido entre 10. |
| **TL** | TL < longitud >  Permite variar la longitud de la herramienta o efector final (la mano) utilizados por el robot, para que los cálculos de posición sean realizados en función de esta nueva dimensión |
| **CP** | CP < contador de 1 a 99 > Permite seleccionar un contador para comparar el valor acumulado con otro valor en una instrucción posterior. |
| **DA** | DA < número de bit >  Deshabilita la interrupción correspondiente al bit de entrada que se encuentre habilitado previamente. |
| **DC** | DC < contador >  Reduce el contador restando 1 al valor acumulado actual. |
| **DL** | DL < número de línea 1 >, < número de línea 2 > (\*)  Borra el contenido de la memoria de programa (2048 líneas), desde la línea 1 hasta la línea 2. |
| **EA** | EA < + ó - > < número de bit > < número de línea >  Permite la interrupción por una señal dada a través del bit específico del terminal de entrada externo, y especifica el número de línea al que el programa salta cuando ocurre la interrupción |
| **ED** | ED  Finaliza el programa. |
| **EQ** | EQ < valor >, < número de línea >  Salta a la línea indicada por el parámetro número de línea, si valor es igual al contenido de un contador seleccionado previamente mediante la instrucción CP. |
| **GS** | GS < número de línea >  Salto a una subrutina que inicia en la línea suministrada como parámetro. |
| **GT** | GT < número de línea >  Salto a la línea indicada |
| **IC** | IC < contador >  Incrementa en 1 el valor acumulado en el contador indicado. |
| **LG** | LG < valor >, < número de línea >  Salta a la línea indicada por el parámetro número de línea, si el parámetro valor es mayor que el contenido de un contador seleccionado previamente mediante la instrucción CP. El rango para el parámetro valor es: -327ó8 < valor < 327ó8. |
| **NE** | NE < valor >, < número de línea >  Salta a la línea indicada por el parámetro número de línea, si el parámetro valor es diferente al contenido de un contador seleccionado previamente mediante la instrucción CP. |
| **NW** | NW  Borra el programa y las posiciones que se encuentren almacenadas en memoria en el controlador del robot. |
| **NX** | NX  Indica el final de un ciclo |
| **RC** | RC < número de repeticiones >  Indica el número de veces que debe repetirse el fragmento de programa consecutivo, cuyo final es la instrucción NX. El parámetro número de repeticiones puede tomar un valor comprendido entre 1 y 32767. |

**Control de la mano:**

|  |  |
| --- | --- |
| **GC** | GC  Cerrar la mano. |
| **GF** | GF < estado 0 ó 1 >  Establece el valor de una bandera que permite mantener el estado (abierto o cerrado) de la mano durante la ejecución de varias instrucciones. |
| **GO** | GO  Abrir la mano. |
| **GP** | GP < fuerza inicial >, < fuerza de retención >, < tiempo de aplicación de fuerza inicial >  Gradúa parcialmente la presión de la mano al momento de cerrar y abrir la mano. |

**Control de Entrada y Salida (I/O) de Datos:**

|  |  |
| --- | --- |
| **ID** | ID  Captura directa de datos en el puerto de entrada |
| **IN** | IN  Captura de datos de forma sincronizada, considerando otras señales disponibles en el puerto de entrada-salida. |
| **OB** | OB , < número de bit >  Activa (+) o desactiva (-) uno de los bits de salida (0 a 15) |
| **OD** | OD < dato >  Fija directamente un dato (1ó bits) en la salida |
| **OT** | OT < dato >  Genera datos de forma sincronizada, considerando otras señales disponibles en el puerto de entrada-salida. |
| **TB** | TB < + ó - >, < número de bit >, < número de línea >  Verifica el estado de un bit de entrada cuyo número es suministrado por el segundo parámetro. Si éste se encuentra en el estado indicado por el primer parámetro, se produce un salto a la línea de programa definida por el tercer parámetro. |

**Control de comunicación a través de un puerto RS-232-C:**

|  |  |
| --- | --- |
| **CR** | CR < contador 1 a 99 >  Lee el valor acumulado en el contador |
| **DR** | DR  Lee el contenido del registro interno. |
| **ER** | ER  Lectura del tipo de error presente durante la ejecución de alguna instrucción. |
| **LR** | LR < número de línea >  Lee el contenido de la línea de programa designada por el parámetro número de línea. |
| **PR** | PR < número de posición >  Lee las coordenadas y ángulos correspondientes a la posición suministrada como parámetro |
| **WH** | WH  Lee las coordenadas y ángulos de orientación de la posición en la que se encuentra el robot |

**Otros comandos:**

|  |  |
| --- | --- |
| **RS** | RS  Puesta a cero del programa y la condición de error. Transfiere el programa y los datos de posición almacenados en la EPROM a la memoria RAM de la controladora |
| **WR** | WR  Escribe el programa generado y los datos de posición almacenados en la EPROM. |
| <cadena de caracteres> | <cadena de caracteres> Permite al programador escribir un comentario – contenido en cadena de caracteres -que no exceda de 7 caracteres |

## ANÁLISIS DE LA APLICACIÓN

Será una aplicación con parte servidor y parte cliente el cual tendrá los controles para manejar el brazo mecánico a través de instrucciones enviadas por un puerto serie conectado al robot. Se necesitará un ordenador servidor con la parte del servidor y un puerto serie o adaptador, y un ordenador cliente que se conectará al servidor mediante una conexión por red.

Los requisitos mínimos son, para el servidor:

* Una conexión a través de socket para conectar los clientes con el servidor
* Un hilo para gestionar las conexiones de los clientes y así poder impedir que se mantengan varias sesiones con el servidor.
* Un conjunto de constantes para las instrucciones del brazo mecánico junto a sus acrónimos y descripción.
* Al establecer conexión que el servidor responda con los puertos series disponibles para comunicarse con el brazo mecánico.

Para el cliente:

La tecnología usada para programar el servidor y el cliente será Java y JavaFX para el cliente. Se va a utilizar socket para realizar las conexiones para no tener interfaces web y para tener mejor velocidad a la hora de enviar y recibir datos. Las conexiones serán monousuarias para impedir contradicciones en el envió de instrucciones al brazo mecánico. Las conexiones al servidor se guardarán en un archivo de configuración para facilitar las conexiones.

Planificación de la realización del proyecto (Diagrama de Gantt o similar)

Se puede utilizar todas aquellas herramientas que sirvan a para mostrar de modo

grafico el análisis, no es obligatorio llegar a un nivel muy detallado, pero sí que nos dé

una idea genérica del proyecto. A modo de ejemplo podremos servirnos de:

* Diagramas E/R o modelo de datos
* Diagramas de clase genérico (Atributos y métodos básicos)
* Casos de uso (El desarrollo de cada caso de uso en anexos)
* Diagramas de flujo de datos: Pueden complementar aquellos casos de uso

complejos.

## DISEÑO

Prototipado: Diseño de pantallas (no el resultado final), bocetos, Mockups

Base de datos. En nuestro proyecto no hay cabida a una base de datos o al menos no hemos identificado una ya que no es necesario guardas grandes cantidades de datos.

Arquitectura:

El proyecto será realizado principalmente en Java ya que es la base de lo que hemos estudiado durante el ciclo, el proyecto será un proyecto Maven con el cual implementaremos las librerías necesarias para la realización del mismo.  
La parte grafica se hará en JavaFX ya que es una forma sencilla y rápida de poder diseñarnos una aplicación de escritorio para Win, además que nos permitirá una fácil comunicación entre la parte de la interfaz y la parte del back de la aplicación.

## IMPLEMENTACIÓN

Desarrollo de la aplicación con los lenguajes y plataformas elegidas: Explicar el código

fuente, scripts, ... más relevantes. Diagrama de clases detallado obtenido por la

aplicación.

La aplicación se ha desarrollado en Java y se han incluido las bibliotecas de JavaFX y de RXTX que se encargar de la comunicación del puerto de comunicaciones. Adicionalmente para su funcionamiento han sido necesarios dll adicionales que se han tenido que introducir en el JDK usado para el desarrollo de la aplicación.

En la siguiente imagen podemos ver el código que se ha utilizado para la creación de los hilos que servirán para la escucha de la parte servidor o cliente dependiendo de donde se encuentren para así poder recibir las instrucciones.

En la siguiente imagen podemos ver el código necesario para la comunicación y utilización del puerto de comunicaciones mediante la librería RXTX.

## IMPLANTACIÓN

El despliegue se realiza en un entorno WIn y la instalación de las aplicaciones solo deberemos de instalar la parte servidor en el ordenador que tendrá el cable de comunicaciones conectado al robot y la parte cliente la tendremos que instalar en cualquier ordenador de la red.

En cuanto a la parte de configuración tendremos que indicar en la parte cliente la IP del ordenador donde este instalado la parte servidor y el puerto será siempre 5555.  
Para mirar la ip del equipo servidor será tan sencillo como abrir un CMD y poner el comando ipconfig de esta forma obtendremos la ip.

Adicionalmente tendremos que ver la configuración de la parte lateral de la controladora como se ha explicado anteriormente en el proyecto y con esto ya tendríamos nuestro programa en funcionamiento.

## DOCUMENTACIÓN

La documentación importante a la hora de la instalación y la utilización de la aplicación se ha proporcionado en la parte de la configuración del robot.

# RESULTADOS Y DISCUSIÓN

Comentarios sobre el desarrollo, detallando la temporalización real respecto a la

Planificada, dificultades más importantes encontradas…

Las mayores dificultades encontradas durante el desarrollo de la aplicación han sido varias.  
La primera de todas ha sido a la hora de realizar la comunicación con el servidor, ya que han sido necesarios la creación de hilos tanto en la parte servidor como cliente para que la escucha de las instrucciones o de los errores que se enviaban desde la otra parte.  
La segunda a sido la comunicación con el robot ya que al ser un ámbito en el que no hemos tratado nunca, ha sido entrar en algo sin casi nada de conocimiento sobre ello, pero respecto a eso lo que mas nos ha dificultado ha sido el poco conocimiento que disponíamos de las funcionalidades y configuraciones del robot ya que se han tenido que modificar varios parámetros, además de dificultades encontradas en el apartado de hardware durante las pruebas ya fuera por problemas con los cables o por desconocimiento de si la parte hardware esta en condiciones óptimas de funcionamiento.

# CONCLUSIONES

Una vez finalizado el proyecto se comentará lo que ha supuesto para tu formación

la realización de este proyecto y en qué medida te ha servido para ampliar

conocimientos y qué destrezas has conseguido con su realización que no se hubiesen

conseguido a lo largo del ciclo.

Este proyecto nos ha supuesto una amplacion de los conocimientos aprendidos en la parte de aplicaciones cliente servidos juntándolo con una aplicación cliente con intefaz grafica, además de introducirnos en el mundo de control de hardware mediante el puerto de comunicaciones obteniendo así conocimiento sobre el traspaso de datos o instrucciones por el mismo, obteniendo así una base en todo lo que eso respecta y abriéndonos unas puertas que antes no habíamos contemplado.

En definitaba

# BIBLIOGRAFÍA

Información genérica del proyecto:  
<http://uvadoc.uva.es/handle/10324/11853>

[PDF con el manual del robot](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjeifLOoN70AhUsyYUKHXULBRQQFnoECAQQAQ&url=http%3A%2F%2Fwww.doyoubuzz.com%2Fvar%2Ff%2F1Z%2F_U%2F1Z_Uax0OM-FbvwXJ6kPgWh8Qu7BtKRiL5r3lqjeINmYn4pTz9S.pdf&usg=AOvVaw2NVzee9eoTPNV49Q-kdzQr).

# ANEXOS

Detalle del código y otra documentación que consideréis oportuna