Universitat de Vic
Escola Politècnica Superior

Trabajo de Fin de Grado

CONTROL REMOTO DEL ROBOT ABB IRB120 POR TECNOLOGÍA WIFI

José Mª Fernández Sanmiguel

Grado en INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Tutor: Gerard Masferrer Caralt

Vic, Septiembre de 2014

ÍNDICE

Pág.

ÍNDICE FIGURAS, TABLAS Y ECUACIONES

1.- RESUMEN

Durante toda la evolución de la tecnología, se han empleado aparatos interconexionados por cables. Los cables limitan la libertad de movimiento del usuario y pueden captar interferencias entre ellos si la red de cableado es elevada.

Mientras avanzaba la tecnología inalámbrica, se ha ido adaptando al equipamiento electrónico a la vez que se iban haciendo cada vez más pequeños. Por esto, se impone la necesidad de utilizarlos como controles a distancia sin el empleo de cables debido a los inconvenientes que estos conllevan.

El presente trabajo, pretende unificar tres tecnologías que pueden tener en el futuro una gran afinidad.

- **Dispositivos basados en el sistema Android**. Desde sus inicios, han tenido una evolución meteórica. Se han ido haciendo cada vez más rápidos y mejores.
- **Sistemas inalámbricos**. Los sistemas wifi o bluetooth, se han ido incorporando a nuestras vidas cada vez más y están prácticamente en cualquier aparato.
- **Robótica**. Cualquier proceso de producción incorpora un robot. Son necesarios para hacer muchos trabajos que, aunque el hombre lo puede realizar, un robot reduce los tiempos y la peligrosidad de los procesos.

Aunque las dos primeras tecnologías van unidas, ¿quién no tiene un teléfono con conexión wifi y bluetooth?, pocos diseños aúnan estos campos con la Robótica.

El objetivo final de este trabajo es realizar una aplicación en Android para el control remoto de un robot, empleando el sistema de comunicación inalámbrico. La aplicación desarrollada, permite controlar el robot a conveniencia del usuario en un entorno táctil/teledirigido.

Gracias a la utilización de simulador en ambos lenguajes (RAPID y Android), ha sido posible realizar la programación sin tener que estar presente ante el robot objeto de este trabajo. A través de su progreso, se ha ido evolucionando en la cantidad de datos enviados al robot y complejidad en su procesamiento, a la vez que se ha mejorado en la estética de la aplicación.

Finalmente se usó la aplicación desarrollada con el robot, consiguiendo con éxito que realizara los movimientos que eran enviados con la tablet programada.

Como dijo el físico Albert Einstein:
"Si lo puedes imaginar lo puedes lograr / Si lo puedes imaginar lo puedes crear".

2.- ABSTRACT

Throughout the evolution of technology, it have been used devices interconnected by cables. The cables limits the freedom of movement of the user and they may pick up interference between them if the wiring is too high.

While wireless technology progressed, it has been adapted to electronic equipment while were getting smaller and smaller. Therefore, it is imposed the need to use such remote controls without the use of wires due to the disadvantages they entail.

This work intended to unify three technologies that may have a great affinity in the future.

- **Devices based on the Android system**. From the beginning, they have had a meteoric evolution. They have become increasingly faster and better.
- **Wireless systems**. Wifi or bluetooth systems have been incorporated into our lives more and more and they are in practically any device.
- **Robotics**. Any production process incorporates a robot. They are necessary for many jobs although man can make, a robot reduces the time and danger of the processes.

Although the first two technologies go together, who doesn´t have a phone with wifi and bluetooth?, few designs combine these fields with Robotics.

The ultimate goal of this work is to create an Android application for remote control of a robot, using wireless communication system. The developed application allows to control the robot by user convenience on a touch/remote environment.

Through the use of simulation in both programming (RAPID and Android), it has been possible to develop it without the robot being present. Through its progress, it has evolved in the amount of data sent to the robot and complexity in processing, while it has improved the aesthetics of the application.

Finally, the developed application with the robot was used to successfully getting the movements that were shipped with the scheduled tablet.

As physicist Albert Einstein said:
"If you can imagine you can do it / If you can imagine you can create it".

3.- INTRODUCCIÓN

Desde que se fabricaron los primeros robots industriales, hemos dependido de su uso para fabricar prácticamente cualquier cosa.

Vivimos en una era que va evolucionando cada vez más a estar inmerso en el mundo de la robótica. Hacen cualquier proceso de fabricación más fácil, pero aun dependemos de la complejidad que tiene controlarlos. Con los diversos sistemas de programación, pueden hacer un sinfín de operaciones, pero lo que haría todo más sencillo es poder controlarlos de forma remota. Existen controles de robots a pequeña escala, pero pocos a mayor escala que puedan ser controlados de forma inalámbrica.

Por este motivo ha sido querer desarrollar un control inalámbrico por tecnología wifi de un robot industrial, en este caso del IRB120 de ABB situado en el Laboratorio de Automática Industrial en la Universidad de VIC.

3.1.- Motivación

Para realizar el trabajo, se unió el interés del autor por la tecnología robótica y los diseños basados en sistemas Android. En el trascurso de los años, he sido conocedor de la necesidad de que puedan ser creados montajes con el menor número de cables posibles.

Por ello fue que, teniendo acceso a la programación del robot y gran cantidad de información para desarrollar una aplicación en Android, quise unir estas dos programaciones para controlar un robot.

3.2.- Objetivos

El desarrollo de la aplicación empezó como un proyecto de investigación para poder controlar el robot. Pero a medida que fue avanzando, se fue convirtiendo en un hecho debido a la gran cantidad de posibilidades que tiene la programación en Android y RAPID.

Inicialmente se pensó en incluir alguna tarjeta de control que adaptara la información enviada al robot. Aprovechando la posibilidad de utilizar un router wifi, como se verá en el esquema de conexión de la ***Figura 3.2*** del ***apartado 3.4. Montaje***, pudo realizarse la trasferencia de datos sin ningún elemento adicional.

3.3.- Plan de trabajo

El trabajo desarrollado se divide en dos grandes bloques, diferenciados en el tipo de programación.

1.- Programación del dispositivo Android, de una aplicación para enviar peticiones de movimientos del robot ABB y poder controlarlo.
2.- Programación del robot ABB para realizar los movimientos requeridos por el usuario y gestionar los errores que pudieran producirse.

Para poder abordar los dos tipos de programación, se tuvo que subdividir en 3 fases en las que poco a poco se iba añadiendo mayor control en la información transferida.

- **Primera fase**: Comunicación básica, configuración de los elementos para mandar y recibir información entre ellos.
    - Conexión wifi y comunicación básica entre terminal Android y PC.
    - Comunicación entre terminal Android y simulador RobotStudio para el robot ABB.
- **Fase 2**: Programación de las diferentes partes.
    - Programación del robot ABB para interpretar la información recibida vía wifi desde el terminal Android.
    - Introducción de accesorios para mejorar la manejabilidad del robot.
    - Programación de la tablet y accesorios para interactuar el usuario.
- **Fase 3**: Montaje y unión de las distintas partes.
    - Conexión de todos los elementos.
    - Puesta en marcha.
    - Depuración de las aplicaciones, solución de errores.

Las tareas realizadas durante todo el proceso de realización del Trabajo fin de Grado han sido:

1.- Estudio necesidades proyecto.
2.- Estudio comunicación por sockets.
3.- Toma de contacto entorno Eclipse. Instalación y configuración.
4.- Conexión entre PC y aplicación Android.
5.- Transferencia básica datos entre PC y aplicación Android.
6.- Programación RobotStudio para conexión con Android.
7.- Aumento opciones y mejoras Android.
8.- Transferencia paquetes datos para movimientos X-Y-Z.
9.- Añadir pantallas y opciones en Aplicación Android.
10.- Transferencia paquetes datos para movimientos de ejes.
11.- Añadir accesorios externos Android. Lectura de eventos.
12.- Transferencia paquetes datos para movimientos de reorientación.
13.- Transferencia paquetes datos tocando la pantalla o inclinándola.
14.- Resolución de errores programación.
15.- Mejora estética aplicación Android.
16.- Realización memoria y preparación exposición.

Estas, se resumen en el cronograma (***Figura 3.1***) que se muestra a continuación:

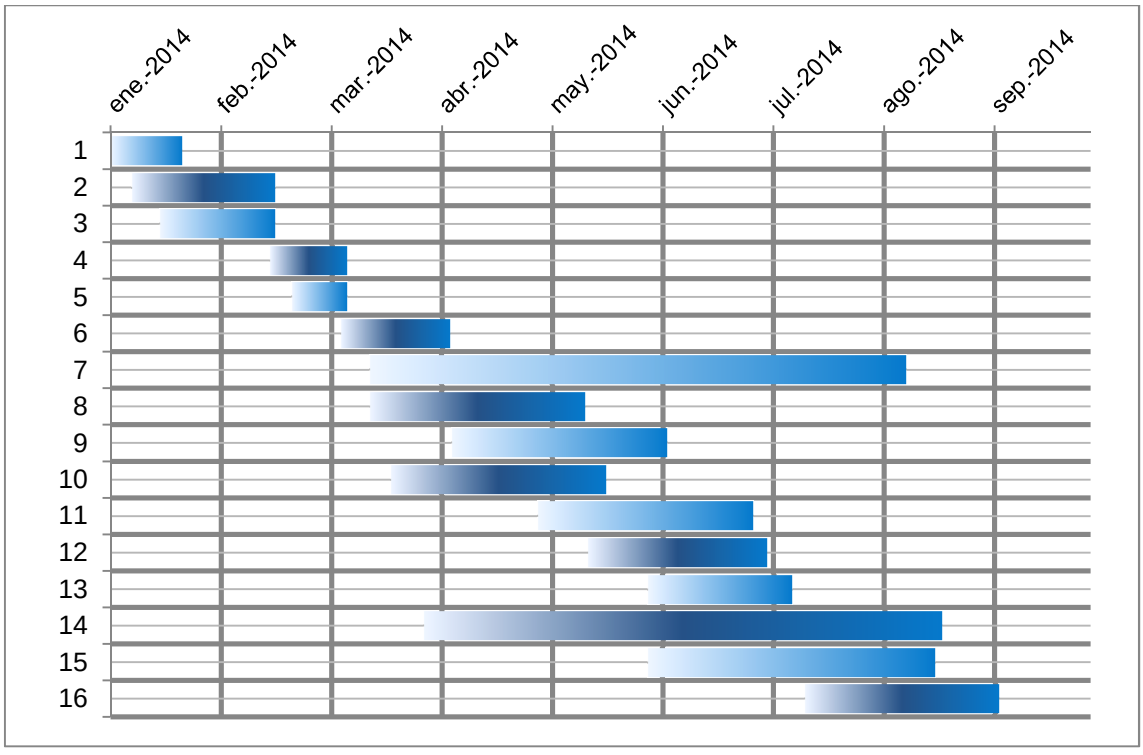


Figura 3.1. Evolución TFG

3.4.- Material usado

Para realizar el montaje, se han utilizado elementos de los que ya se disponían físicamente. El medio para poder unirlos, ha sido la programación para la comunicación entre ellos.

En la ***Figura 3.2*** se muestra el montaje final.

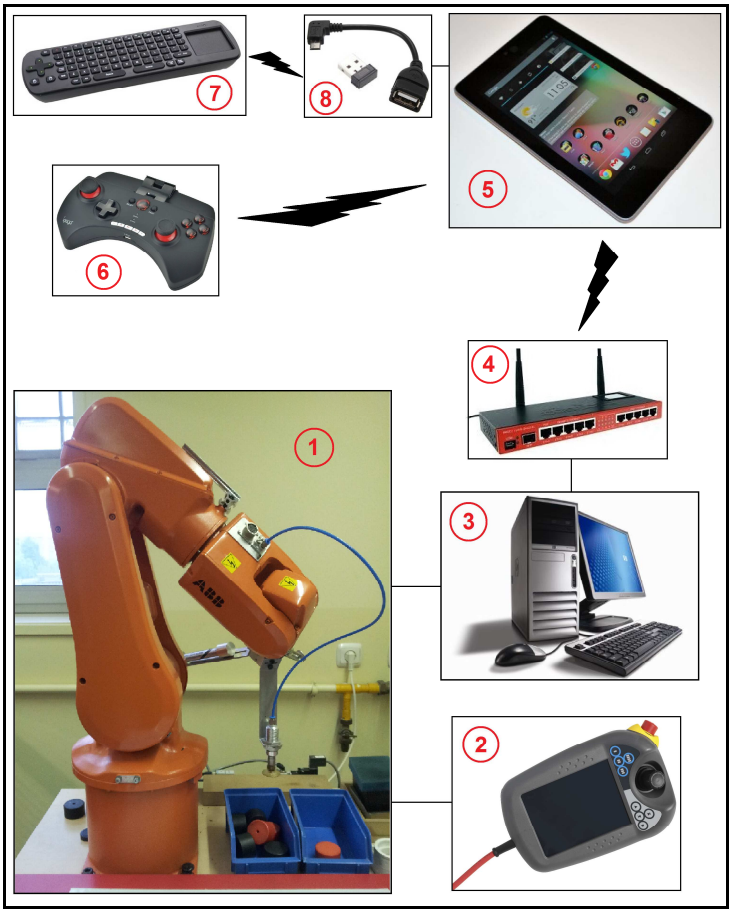


Figura 3.2. Montaje

1.- Robot ABB IRB120.
2.- Flexpendant para el robot.
3.- PC para el control del robot. Para introducir el código en RAPID que será transferido al Flexpendant.
4.- Dispositivo para obtener un punto de acceso wifi de la marca Mikrotik, para realizar la conexión con el robot.
5.- Dispositivo Android. Tablet Nexus 7 de Google.
6.- Gamepad por bluetooth Ipega.
7.- Teclado Smart Remote Measy por RF.
8.- Cable adaptador para Teclado Measy.

3.5.- Puesta en marcha y resultados finales

Conociendo las funciones del terminal Flexpendant que controla el robot, se fueron añadiendo más posibilidades para manejarlo y una mayor gestión de los datos transferidos.

Después de programar la aplicación en ambos lenguajes y realizar las pruebas con el simulador RobotStudio, se probó la aplicación con el robot real. Los resultados fueron los esperados y respondió con éxito a todas las opciones diseñadas en la aplicación.

3.6.- Estructura del archivo "fernandez_sanmiguel_jm.zip"

Los archivos incluidos en ***fernandez_sanmiguel_jm.zip***, al que pertenece este documento, son:

- ***fernandez_sanmiguel_jm_memoria.pdf***: Archivo pdf del presente documento.
- ***fernandez_sanmiguel_jm_anexo.pdf***: Archivo pdf del Anexo de la memoria.
- ***Robotwidroid.apk***: Aplicación Android lista para ser instalada.
- ***Robotwidroid.mov***: Video demostración de la aplicación Android.
  El archivo puede ser abierto con el reproductor de Windows.
- ***fernandez_sanmiguel_jm_app.zip***: Conjunto de carpetas creadas en el programa Eclipse para el desarrollo de la aplicación Android.
  - La distribución de las carpetas se muestra en la ***Figura 6.6, apartado 6.3.1.1. Estructura directorios aplicación***.
  - Tanto los archivos *.java* como los *.xml,* pueden ser abiertos con el "Bloc de Notas" o el "Wordpad" de Windows.
- ***fernandez_sanmiguel_jm_RobotStudio (carpeta):***
  - ***fernandez_sanmiguel_jm_RobotStudio.rspag***: Entorno de trabajo de RobotStudio con la programación incluida. Este archivo será abierto con el programa de RobotStudio.
  - ***fernandez_sanmiguel_jm_Mainmodule.mod***: Programa de RAPID. Este archivo puede leerse con el "Bloc de Notas" de Windows.
  - ***fernandez_sanmiguel_jm_Calibdata.mod***: Programa de RAPID donde están configurados los elementos de la herramienta. Este archivo puede leerse con el "Bloc de Notas" de Windows.

4.- ANTECEDENTES TEÓRICOS

Para poder realizar el montaje y comunicación entre dispositivos, se dispone de una amplia información de todos los elementos que componen el diseño final. Toda esta documentación está disponible en la biografía de la memoria:

- Diseño de aplicaciones Android. ***Bibliografía 10.1.***
- Robot Industrial ABB del Laboratorio. ***Bibliografía 10.2.***
- Accesorios para mejorar la interactuación de los elementos. ***Bibliografía 10.4.***

Con todo ello, se unen en un mismo diseño el control del robot del Laboratorio, mediante una tablet en la que habremos instalado una aplicación en sistema Android diseñada específicamente para este trabajo.

Mediante un conjunto de instrucciones (RAPID tiene una amplia gama), es posible interpretar los paquetes de datos recibidos y realizar movimientos controlados por una pantalla táctil (como es el caso del terminal Android) de una forma muy intuitiva.

La aplicación en Android no pretende sustituir el Flexpendant del robot ABB. La programación en ambos lenguajes (Android y RAPID), sirven de complemento inalámbrico para poder controlar el robot.

4.1.- Sistema Android

4.1.1.- ¿Por qué la elección del sistema Android?

Este sistema se ha introducido en nuestra sociedad desde sus inicios, a la par que Internet influye cada vez más en nuestras vidas.

Entre la gran variedad de características de Android, destacan:

- Plataforma realmente abierta. Es una plataforma de desarrollo libre basada en Linux y de código abierto.
- Adaptable a cualquier tipo de hardware. Android no solo ha sido orientado para su uso en teléfonos y tablets, sino que también ha sido introducido en relojes, cámaras, electrodomésticos, automóviles, y muchos más.
- Portabilidad asegurada. Las aplicaciones son desarrolladas en Java, las cuales podrán ser ejecutadas en prácticamente cualquier tipo de PC. Esto se debe al concepto de máquina virtual.
- Arquitectura basada en componentes inspirados en Internet. Esto permite que la aplicación pueda ejecutarse tanto en una pantalla de móvil como en un televisor.

- Filosofía de dispositivo siempre conectado a Internet. Da la posibilidad de estar conectado a la "nube" para acceder a cualquier dato que queramos.
- Gran cantidad de servicios incorporados. Por ejemplo, las redes sociales, localización GPS, bases de datos, multimedia, etc.
- Aceptable nivel de seguridad. Su núcleo de Linux lo dota de cierta inmunidad, aunque todo dependiendo del uso y acceso que nosotros le demos.
- Optimizado para baja potencia y poca memoria. Gracias al uso de la Máquina Virtual Dalvik, no genera grandes consumos y trabaja con poca potencia.
- Alta calidad de gráficos y sonido. Posee gráficos vectoriales suaves, animaciones inspiradas en Flash, gráficos en 3D basados en OpenGL, incorpora los codecs estándar más comunes de audio y vídeo.

4.1.2.- Su historia

Sus orígenes comienzan en 2005, cuando Google adquiere Android Inc., una pequeña compañía que acababa de ser creada para la producción de aplicaciones de terminales móviles.

Mientras iban apareciendo nuevas versiones, se les ha ido nombrando por nombres de postres en inglés en orden alfabético.

- **A**: Apple Pie (v1.0): *tarta de manzana.* Lanzada en Septiembre de 2008.
- **B**: Banana Bread (v1.1): *pan de plátano*. Febrero de 2009.
- **C**: Cupcake (v1.5): *panqué*. Abril 2009.
- **D**: Donut (v1.6): *rosquilla.* Septiembre 2009.
- **E**: Éclair (v2.0/v2.1): *pastel francés*. Octubre de 2009.
- **F**: Froyo (abreviatura de "frozen yogurt") (v2.2): *yogur helado*. Mayo de 2010.
- **G**: Gingerbread (v2.3): *pan de jengibre*. Diciembre 2010.
- **H**: Honeycomb (v3.0/v3.1/v3.2): *panal de miel*. Febrero de 2011.
- **I**: Ice Cream Sandwich (v4.0): *sándwich de helado*. Octubre de 2011.
- **J**: Jelly Bean (v4.1/v4.2/v4.3): *gominola*. Julio de 2012
- **K**: KitKat (v4.4): *tableta de chocolate con leche*. Octubre 2013.
- **L**: Android L. Próximamente...

Figura 4.1. Evolución Android

A fecha de la realización de esta breve historia, la última versión conocida es la 4.4.4, actualizada en el dispositivo tablet utilizado para pruebas de comunicación con el robot. Posiblemente la próxima versión sea la 4.5 o 5.0 que tiene prevista su aparición en los próximos meses.

4.1.3.- Arquitectura de Android

Está formada por cuatro capas como puede verse en la ***Figura 4.2***:

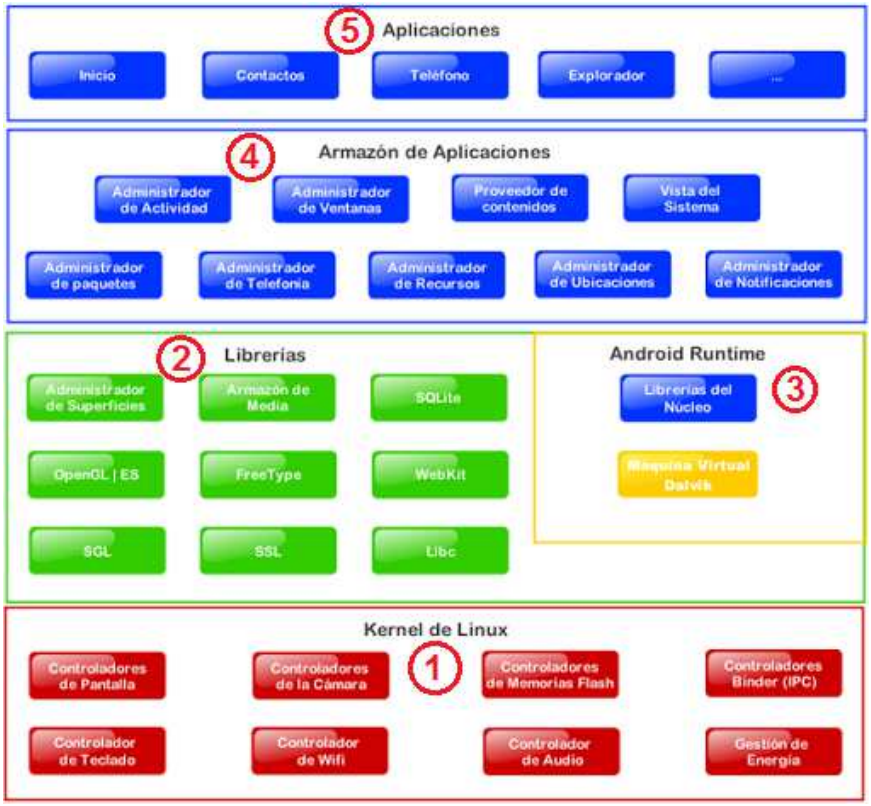


Figura 4.2. Arquitectura Android

1.- **Kernel de Linux:** Contiene todos los controladores del dispositivo de bajo nivel para los diversos componentes de hardware.
2.- **Librerías:** Contienen todo el código que proporciona las principales características de un sistema operativo Android.
3.- **Tiempo de ejecución Android:** En el mismo nivel que las librerías, proporciona un conjunto de librerías principales que permiten a los desarrolladores escribir aplicaciones Android al utilizar el lenguaje de programación java.
4.- **Arquitectura de software de aplicación:** Presenta diferentes posibilidades de uso del sistema operativo Android para los desarrolladores de aplicaciones.
5.- **Aplicaciones:** Estas son teléfono, contactos, explorador, etc.

La característica que destaca de todas las capas, es que están basadas en software libre. Esto quiere decir que está disponible el código fuente del software.

Android está basado en el sistema con núcleo kernel Linux. Kernel es la parte fundamental del sistema operativo y el encargado de facilitar el acceso seguro al hardware por parte de los programas, es decir, es el encargado de administrar todos los recursos del sistema.

Linux es un sistema operativo libre de la familia Unix.

Todo ello unido, forman lo que ya conocemos como Android, un sistema operativo que se ha introducido en el mercado desbancando a otras plataformas.

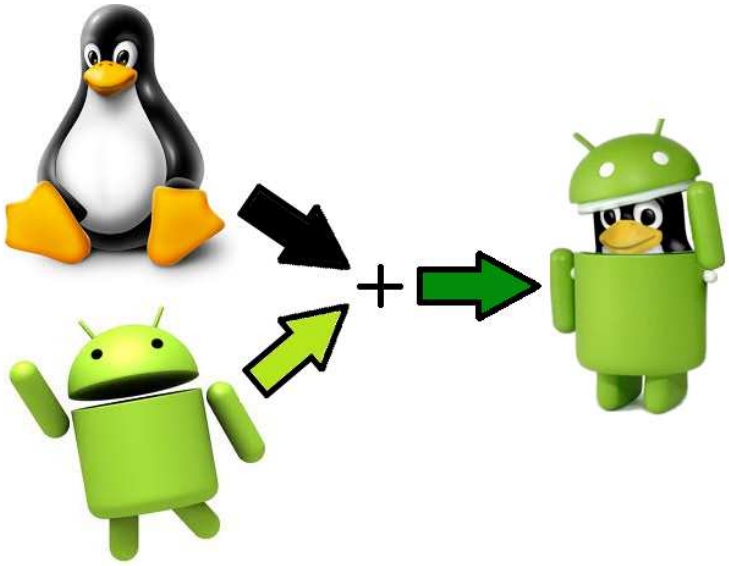


Figura 4.3. Unión entre Linux y Android

4.2.- Robot ABB con Flexpendant

La aplicación en Android, se ha diseñado en base al empleo con el robot IRB120 de la empresa ABB. La programación que interpreta es en lenguaje RAPID (*Robotics Application Programming Interactive Dialogue*).

El lenguaje RAPID, fue creado en 1994 por ABB. Muy similar a otros lenguajes de programación de alto nivel, ha sido diseñado especialmente para controlar robots.

En el programa del robot, está configurada una herramienta con tres terminales. Un rotulador, una punta boli y una ventosa, las cuales están configuradas en el módulo de calibración del robot. En desarrollos futuros de la aplicación en Android y en RAPID, podrían modelarse diferentes configuraciones de robots y herramientas. La gran mayoría de los datos transferidos son de posiciones x, y, z y de ejes. Valores que podrían ser interpretados por otros modelos de robot. Solo habría que adaptar la parte en que se selecciona el tipo de herramienta a utilizar.

4.2.1.- Su historia

Desde sus comienzos en 1883, por Ludvig Fedholm fabricando equipos eléctricos de iluminación y generadores, han desarrollado innumerables avances y maquinaria. Iniciándose por separado como ASEA (*Allmänna Svenska Elektriska Aktiebolaget*) y BBC (*Brown, Boveri y Cie*). Ambas se fusionaron en 1988 para formar ABB (*ASEA Brown Boveri*).

ABB es una empresa que opera en la robótica y sobre todo en las áreas de tecnología y automatización. Presenta cinco divisiones de producción desde su reorganización en 2010.

1.- Productos de transmisión y distribución de electricidad. Transformadores, interruptores, cables, equipos de alta y media tensión como relés de protecciones digitales.
2.- Servicios de transmisión de energía y redes de distribución de las plantas de energía.
3.- Productos y servicios para la producción industrial como motores eléctricos, generadores, electrónica de potencia y robots industriales.
4.- Productos de baja tensión como interruptores, productos de control, accesorios eléctricos, cerramientos y sistemas de cable para proteger a las personas, instalaciones y equipos electrónicos de sobrecarga eléctrica.
5.- Automatización de procesos. Proporcionando optimización de la planta y aplicaciones de automatización específicas de la industria.

4.3.- Comunicación inalámbrica

En el caso que nos ocupa, se han utilizado dos tecnologías de comunicación inalámbrica.

- **Wifi**. Para la comunicación vía sockets entre el terminal Android y el robot.
  - Necesita más potencia de transmisión.
  - Velocidad de transferencia de datos, 11Mbps hasta 54Mbps.
  - Alcance hasta 100m.
- **Bluetooth**. Para sincronizar el mando externo que se utilizará con el terminal Android.
  - Orientada a la sincronización entre dispositivos.
  - Velocidad de transferencia de datos, 1Mbps.
  - Alcance, de 10cm hasta 10m, según potencia.

Por lo tanto, ninguna de las dos es mejor ni peor, simplemente dependiendo del uso que se vaya a hacer se utilizará una u otra.

4.3.1.- Conexión Wifi

Wi-Fi (*Wireless Fidelity*) es una de las tecnologías de comunicación inalámbrica más extendidas. Permite establecer conexiones a Internet sin ningún tipo de cables. Para ello, es necesario tener un punto de acceso o router wifi con el que se pueda establecer la conexión. La gran ventaja es la ausencia de cables, pero tiene la desventaja de que si el sistema puede no estar protegido y es fácil "captar" esta señal para poder utilizarla.

Figura 4.4. Logo WIFI

Otro tipo de tecnología que se utiliza en el desarrollo de la aplicación es la Bluetooth. Esta será usada por el accesorio que será sincronizado con el terminal Android.

4.3.2.- Conexión Bluetooth

Bluetooth es una tecnología inalámbrica destinada a conectar dispositivos que se encuentran a corta distancia mediante un enlace por radiofrecuencia en la banda de los 2,4 GHz. Actualmente, es posible encontrar este tipo de conexión en muy diversos aparatos, tablets, móviles, ordenadores, ratones, teclados, impresoras, cámaras digitales, televisores, etc.

Figura 4.5. Logo Bluetooth

4.4.- Comunicación por sockets

Para poder realizar una comunicación entre el terminal Android y el robot de ABB, se ha empleado la conexión mediante sockets.

Socket podría interpretarse como “enchufe” o “puerta” bidireccional para poder intercambiar información entre dos programas. Uno será creado en el programa servidor (que controlará el servicio) y otro en el cliente (que realizará peticiones al servidor).

Para poder realizarse la conexión, es necesario conocer la IP y el puerto de enlace.

- La IP es un número que identifica a un PC o un terminal con conexión wifi (como el caso del terminal Android o el router empleado para realizar la conexión), dentro de una red que utilice este protocolo.
- El puerto podría asimilarse al canal por el que circularán los datos. Es posible configurar cualquier puerto, pero hay alguna serie de puertos que ya están asignados.
    - Entre el 0 y 1023. Solo pueden utilizarlos procesos del sistema
    - Entre el 1024 y 49151. Pueden utilizarlos procesos de usuario.
    - Entre el 49152 y 65535. Son dinámicos y/o privados.

Por lo tanto nosotros escogeremos un puerto que se encuentre en el segundo rango (1024 a 49151). Este será configurado en el programa de nuestro robot y será suministrado junto a la IP, al usuario del dispositivo tablet que vaya a realizar la conexión. Dicho dispositivo, tendrá conexión wifi para conectarse dentro de la misma red donde se encuentre el robot.

En los programas, se usará un socket orientado a la conexión. Este se denomina socket stream. Esto implica que antes de poder enviar y recibir datos es necesario establecer una conexión entre cliente y servidor. Una vez establecida la conexión, se garantiza que los datos enviados son recibidos correctamente.

Otro tipo de socket, no orientado a la conexión, se denomina socket datagram o UDP (*User Datagram Protocol*), pero nos centraremos en el primero basado en el protocolo TCP (*Transmission Control Protocol)* el cual proporciona un transporte fiable del flujo de bits entre aplicaciones.

El diagrama de flujo básico para la conexión por sockets entre un servidor y un cliente es el que se muestra en la ***Figura 4.6***.

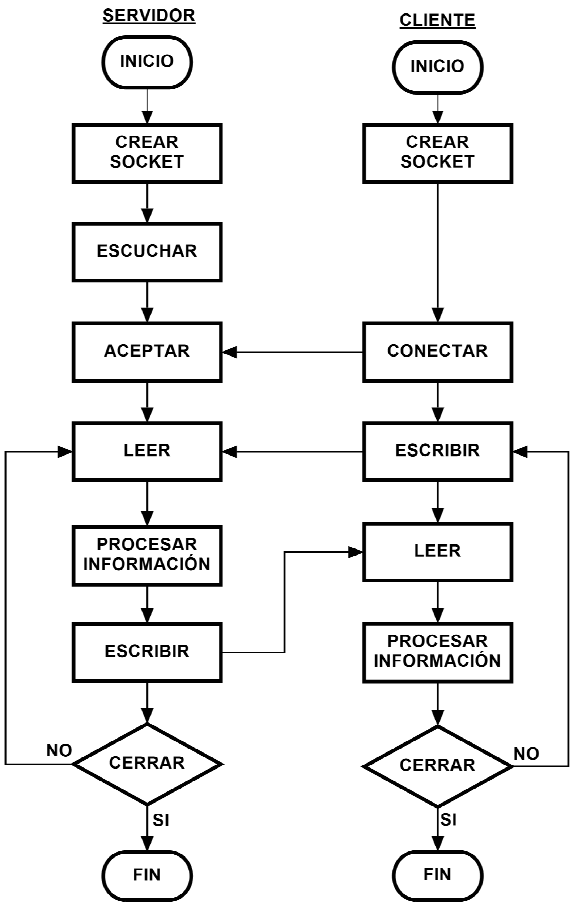


Figura 4.6. Conexión Servidor-Cliente

Según el diagrama mostrado en la ***Figura 4.6***, el servidor crea un socket y queda a la espera de la petición de un cliente.

En este caso, el servidor es el programa creado en el robot que se encuentra en una red y tiene asignada una IP local y un punto de acceso wifi, mientras que el cliente es el programa instalado en Android.

Una vez creado el socket en el servidor, y el cliente ha enlazado su socket con el del servidor, es posible el envío de datos entre ambos programas. El programa de RAPID interpreta los datos enviados por Android para realizar las operaciones solicitadas. A su vez el robot enviará datos al terminal Android cuando este se lo pida. Como se explicará en la sección ***6.3. Funcionamiento software***, el robot no siempre envía datos, solo cuando se le solicita. Esto es debido a que para realizarlo, es necesario cerrar el socket, crearlo y realizar una nueva conexión. Con las pruebas realizadas, cuando esto ocurría, se ralentizaban mucho los movimientos del robot. Por lo tanto se ha optado solo a realizar envío de paquetes de datos del servidor al cliente, cuando este lo solicite.

5.- ENTORNO DE TRABAJO ROBOT ABB IRB120

Antes de entrar en detalle del funcionamiento de la aplicación programada, es necesario conocer el entorno de trabajo donde operará el robot.

En la ***Figura 5.1***, se muestra el espacio de trabajo del robot del software RobotStudio. Este es muy similar al entorno de trabajo real del robot.

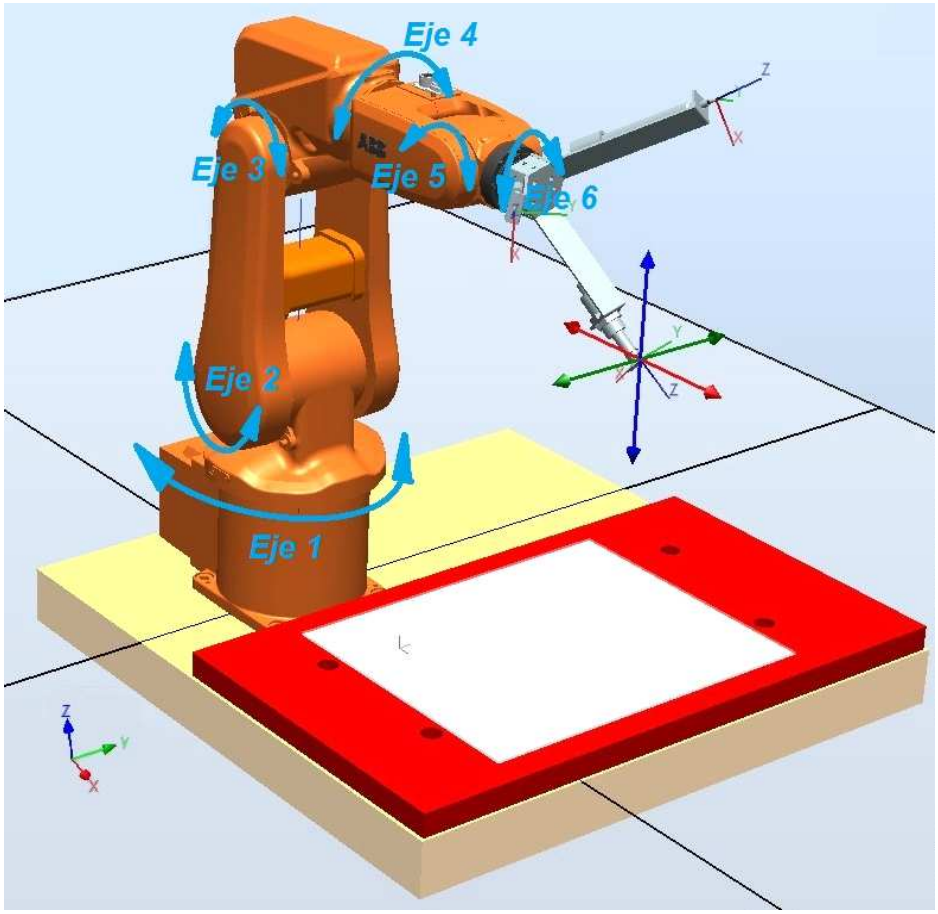


Figura 5.1. Entorno Robot ABB IRB120

El robot utilizado tiene seis articulaciones (Eje 1 a Eje 6), con los que se puede abarcar un gran margen de movimientos. Este tiene conectado en su sexto eje, un terminal de tres brazos que puede ser utilizado cada uno de ellos en la aplicación programada en Android.

En la configuración del robot, están establecidas unas orientaciones de las herramientas respecto al sistema de coordenadas del robot. Estas ya vienen determinadas en su módulo de calibración.

Para la correcta utilización de las herramientas, se han configurado las orientaciones como se muestran a continuación. Aunque estas configuraciones son las que inicialmente están programadas en el robot, es posible enviar desde la aplicación Android otras orientaciones.

La herramienta rotulador, se ha configurado con sus ejes de coordenadas con un giro de 180º en torno al eje X de referencia. En este caso no se aplica ningún giro en los ejes Y y Z, de forma que queda orientada como se muestra en la ***Figura 5.2***.

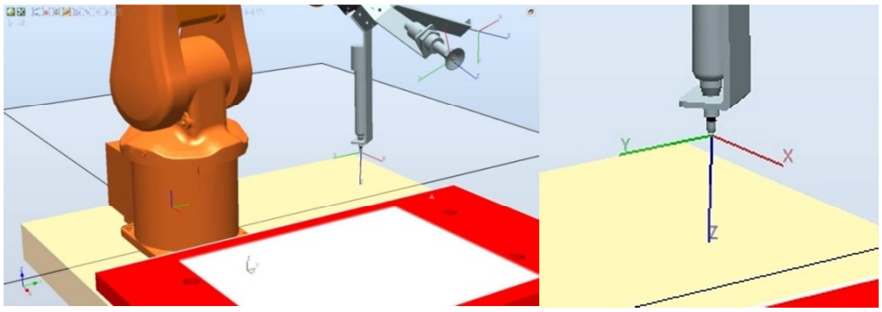


Figura 5.2. Herramienta Rotulador. 180º, 0º, 0º

En la herramienta boli (***Figura 5.3***) se ha aplicado un giro de 180º en torno al eje X y de 90º en torno al eje Z.

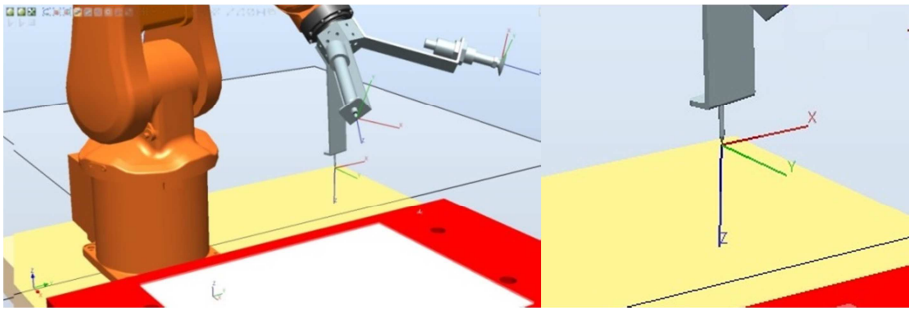


Figura 5.3. Herramienta Boli. 180º, 0º, 90º

En el caso de la ventosa (***Figura 5.4***) se ha aplicado un giro de 180º en torno al eje X y de 180º en torno al eje Z.

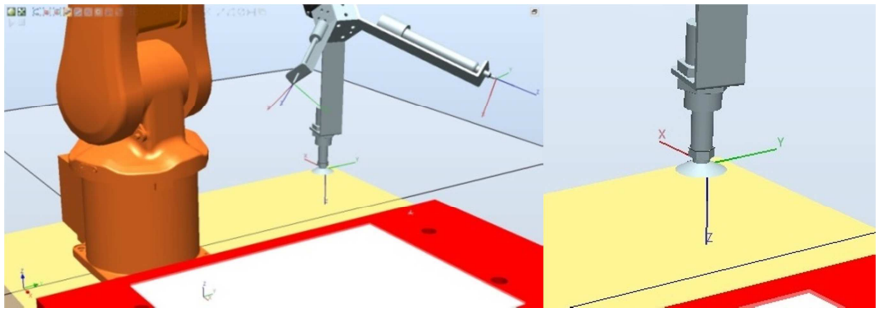


Figura 5.4. Herramienta Ventosa. 180º, 0º, 180º

En la pantalla en la que se podrá dibujar con el rotulador del robot de forma táctil o con la inclinación de la tablet, se utilizará solo el terminal rotulador. De forma que se posicionará como se muestra en la ***Figura 5.5***.

La orientación será la misma en toda el área de dibujo en el panel de la pizarra.

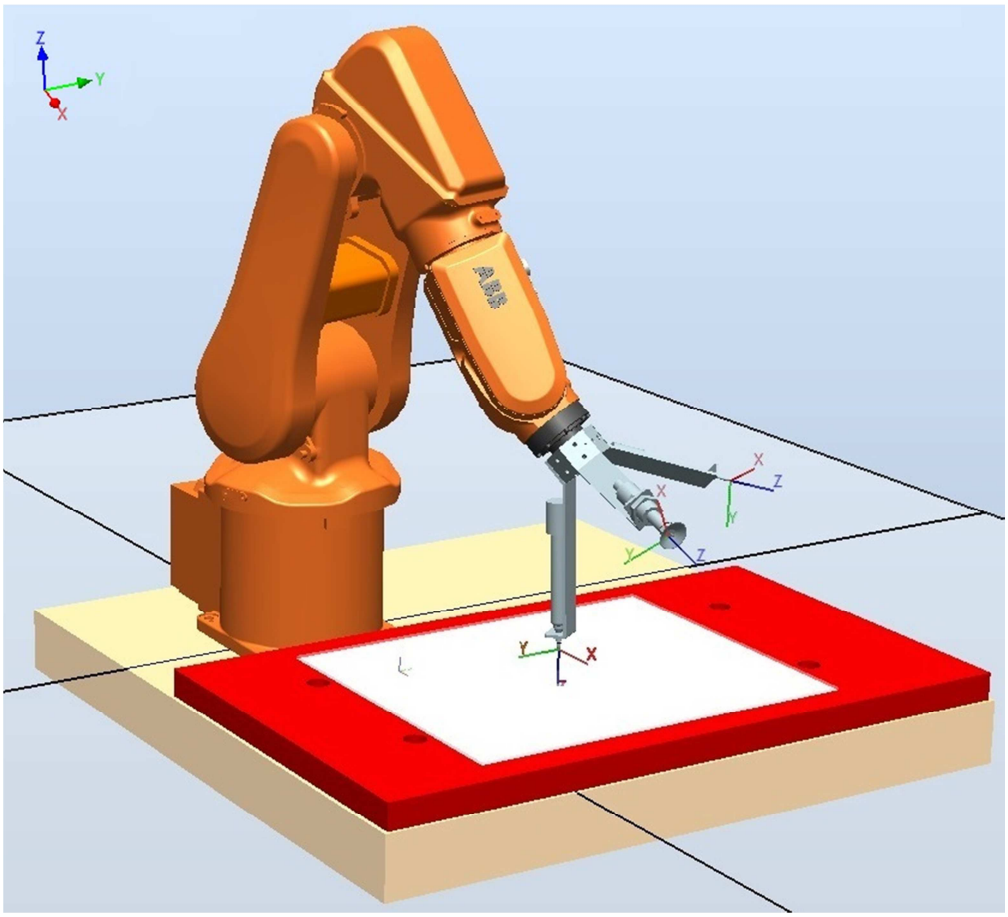


Figura 5.5. Posicionamiento Robot en pizarra

6.- DESCRIPCIÓN DEL HARDWARE Y SOFTWARE UTILIZADOS

6.1.- Hardware

6.1.1.- Robot ABB con punto de acceso Wifi

Las características del robot, pueden consultarse en la ***Figura 1.1*** del ***Anexo 1.Características Robot ABB IRB120***. También puede consultarse en la ***Bibliografía [10]***, pero las que más se han tenido en cuenta a la hora de la programación, han sido los límites de los ejes del 1 al 6. Estos son los que pueden generar más errores, ya que si se producen movimientos más allá de sus límites, el programa se detiene. En la programación se evita que se realicen movimientos que puedan ocasionar errores, calculando el valor de estos ejes de una posición destino antes de que se desplace.

6.1.2.- Terminal Android con conexión wifi y bluetooth

Para la realización del Trabajo de Fin de Grado se ha utilizado la tablet Asus Google Nexus 7 (***Figura 6.1***) con la propiedad de que tiene conexión wifi y bluetooth.

La conexión wifi es imprescindible para la conexión con el robot.

La conexión bluetooth no es obligatoria, pero si recomendable para poder usar el gamepad y aprovechar al máximo las opciones de la programación.

- Pantalla HD de 7,02 pulgadas de 800 x 1280 pixeles (216 ppi) retroiluminada.
- Tamaño 114 x 200 x 8,65 mm.
- Peso 340g.
- Conexiones inalámbricas: Wi-Fi (802.11 b/g/n) y Bluetooth.
- Memoria: 16GB de almacenamiento interno y 1 GB de RAM.
- Conexión por Micro USB.
- Batería. 4325 mAh (hasta 10 horas).
- Sistema operativo 4.4.4.
- CPU: Procesador Tegra 3 de cuatro núcleos.
- Características: Micrófono, NFC, GPS, Acelerómetro, Magnetómetro, Giroscopio.

Figura 6.1. Tablet Asus Nexus 7

6.1.3.- Accesorios

Para mejorar el funcionamiento de la aplicación en Android, se ha añadido la posibilidad de utilizar dos accesorios externos a la tablet. Un teclado externo y un mando gamepad. La configuración de las teclas se muestra en el ***apartado 6.3.1.3. Configuración teclas accesorios***, después de presentar el funcionamiento de cada una de las pantallas de la aplicación.

6.1.3.1.- Teclado externo Measy RC12 con touchpad

Se ha incluido la opción de poder utilizar un teclado externo a la tablet.

La comunicación de este accesorio, es por conexión inalámbrica a través de un receptor conectado a la entrada micro USB de la tablet. Para ello, se ha utilizado un cable OTG (adaptador de micro-USB a USB) como el mostrado en la ***Figura 6.3***.

Las características del teclado externo son:

- Emisión por radio-frecuencia de 2,4GHz.
- Teclado estándar QWERTY con touchpad.
- Utiliza tres pilas AAA de 1,5v.
- Bajo consumo. 21,5mA del receptor USB y 20mA del control remoto.
- Soporta los sistemas de Windows 7/ 8/ XP/ Vista, y de Android 2/ 2.3/ 4.03/ 4.1/ /4.1.2 y 4.4.4.
- La tecnología de ahorro de energía avanzado mejora la vida de la batería.
- Entra en modo Standby después de 2 minutos sin utilizarlo. La corriente de latencia es solo 10µA. Una vez presionado de nuevo un botón vuelve a entrar en funcionamiento.
- Modo “Plug and play”, no necesita driver. Es compatible con USB1.1 y USB 2.0.
- Puede ser utilizado para PC, proyectores, TV, AHFC, Tablet-PC, etc.

Figura 6.2. Teclado Measy RC12

Figura 6.3. Cable OTG

En el ***apartado 6.3.1.3.1. Teclado externo Measy RC12 con touchpad***, se detalla la configuración de las teclas utilizadas en la aplicación Android.

6.1.3.2.- Mando gamepad Ipega PG-9025

Con a la utilización de un mando gamepad similar el usado en las consolas de videojuegos, es posible realizar movimientos del robot con los joysticks y teclas de las que dispone.

Las características más importantes de este mando son:

- Compatible con Android 2.3 o superior, IPod, Iphone, Ipad, así como la gran mayoría de los Tablet PC Android.
- Conexión Bluetooth 3.0.
- Rango de funcionamiento 6 a 8m.
- Batería de Lítio de 380mAh no reemplazable.
- Tiene 2 joysticks, 1 grupo de 4 teclas de dirección, 16 teclas y 4 modos de funcionamiento.
- Entra en modo apagado después de 8 minutos sin utilizarlo.

Figura 6.4. Mando Ipega PG-9025

En el ***apartado 6.3.1.3.2. Mando gamepad Ipega PG-9025***, se detalla la configuración de las teclas utilizadas en la aplicación Android.

6.1.4.- PC para programación

Para la programación en ambos lenguajes, se ha utilizado un PC portátil VAIO SVF1521F6EW con las siguientes características:

- Sistema operativo Windows 8.
- Procesador Intel Core i3-3217U a 1,80 GHz.
- Disco duro de 500GB con 6GB de RAM.

Figura 6.5. PC VAIO

6.2.- Software utilizado

La realización del proyecto ha sido en su gran mayoría programación en dos lenguajes diferentes, Android y RAPID.

El grueso de la programación, se centra en Android, ya que se deben crear varias pantallas, botones, campos de texto, lectura de eventos, etc. Mientras que en RAPID, se debe interpretar la información recibida por el terminal Android, realizar los movimientos y evitar los errores que puedan ocasionarse.

Para ello, se dispone en Internet de gran cantidad de información y tutoriales.

Las versiones utilizadas en la programación son:

6.2.1.- Eclipse

El entorno de programación empleado para la programación de Android ha sido:

- JDK Java. Versión 7(Java SE Developmen kit 7) para Windows x64.
- SDK Android (Kit de desarrollo) que engloba:
    - Eclipse Java EE IDE for Web Developers, para Windows 64 bits. Versión Kepler Service Release 2.
    - ADT plugin para Eclipse, actualizado a la versión 23.0.2 de Junio de 2014.

Es un software de libre uso y que ha sido descargado de sus páginas oficiales. Consultar la ***Bibliografía 10.5 [21 a 24]*** donde se muestran las páginas de descarga.

6.2.2.- RAPID

Para la programación se ha empleado el software de ABB, RobotStudio en su versión 5.15.02, en el que se ha podido simular el funcionamiento del programa sin necesidad de tener el robot. Posteriormente se ha transferido el programa en RAPID al robot para realizar las pruebas. Consultar la ***Bibliografía 10.5 [25]***, para consultar la página oficial ABB de descarga del software.

Con el software de RobotStudio en su versión libre de 30 días, ha sido posible utilizar el programa, mediante el acceso por VPN al servidor de licencias de la Universidad de VIC.

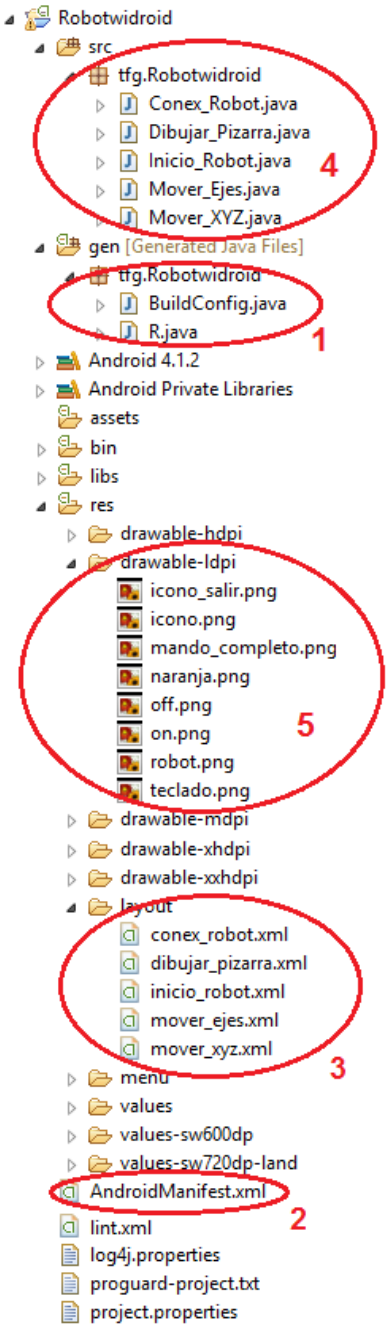
Para la simulación, se ha empleado la estación de trabajo utilizada en las prácticas del curso 2012/2013 de la asignatura de Sistemas Robotizados. En dicha estación ya estaban configurados el espacio de trabajo y la herramienta con los tres terminales.

6.3.- Funcionamiento software

6.3.1.- Android

6.3.1.1.- Estructura directorios aplicación

Los archivos más importantes dentro del directorio que se crea en el entorno de programación Eclipse, son los que se muestran en la ***Figura 6.6***:



1.- Al empezar un proyecto en Eclipse se crean los archivos ***BuildConfig.java*** y ***R.java***. Estos no deben ser modificados ya que se van actualizando a medida que se va creando la programación.

2.- El archivo ***AndroidManifest.xml*** será el encargado de determinar la configuración del proyecto Android. Se establecen las propiedades de cada una de las pantallas, así como el permiso de acceso a internet.

3.- Estos archivos ***.xml*** son los creados para cada una de las pantallas de la aplicación. En ellos se configura el entorno gráfico de cada una. Botones, texto, campos de entrada de texto, etc.

4.- Los archivos ***.java*** son los que contienen todo el código de programación para crear el funcionamiento de la aplicación.

5.- En esta carpeta se guardarán archivos que se usarán en la aplicación como iconos, y gráficos.

En cada una de las carpetas ***drawable (5)***, pueden guardarse diferentes calidades de gráficos para distintas resoluciones de pantallas.

Del mismo modo, la carpeta ***layout (3)***, puede crearse para otros tamaños de pantalla.

El resto de carpetas son de librerías utilizadas en la aplicación y archivos que se van creando automáticamente.

Figura 6.6. Estructura directorios Eclipse

6.3.1.2.- Pantallas aplicación

Se han creado 5 pantallas para la distribución de las diferentes opciones, que son:

6.3.1.2.1.- Inicio

Esta es la pantalla de bienvenida. En ella se muestra el robot para el que se diseñado la aplicación.

Figura 6.7. Pantalla principal

- Desde aquí, es posible acceder a la pantalla donde se permitirá realizar la conexión con el robot de ABB.
- También mostrará una guía resumida del funcionamiento de las diferentes pantallas y de la configuración de los botones de los accesorios utilizados con la tablet. También se muestra información del desarrollador de la aplicación.
- Solo desde esta pantalla, se permitirá salir de la aplicación. En cuyo caso, preguntará si se quiere hacer una desconexión total con el robot.
- El cambio de pantalla, así como en el resto de pantallas, podrá ser realizado desplazando el dedo por zonas donde no haya botones o campos de introducción de texto. Desde esta pantalla solo se podrá pasar a la de conexión.

En el resto de pantallas si se desplaza el dedo de izquierda a derecha será el cambio en el sentido: “Conexión Robot”, “Mover XYZ”, “Mover Ejes”, “Dibujar Pizarra”. En el sentido inverso será cuando lo desplacemos de derecha a izquierda.

En este apartado y los siguientes se muestra el funcionamiento de las pantallas.

En el Anexo de figuras, se muestran las diferentes posibilidades de cada una.

6.3.1.2.2.- Conexión Robot

Figura 6.8. Pantalla Conexión

- Esta es la pantalla que servirá para conectar con el robot, el cual tiene que estar conectado a un punto de red con acceso wifi para la tablet. Tendrá una IP asignada y un puerto de comunicación. Estos dos datos tienen que estar configurados en el programa RAPID para poder realizar la conexión.
- Se introducirán los datos de IP y puerto con los que se realizará la conexión. Estos datos serán guardados en un archivo privado que serán recuperados la próxima vez que se entre en esta pantalla.
- Un botón, permitirá realizar la conexión o desconexión con el robot. Un mensaje en la pantalla, indicará que se ha realizado con éxito. Si no fuera así, se mostraría un mensaje de fallo de conexión. Una vez realizada la conexión, es posible pasar a otras pantallas.
- Se ha incluido un botón para desplegar el menú de cambio de pantalla.
- Solo será posible salir a la pantalla principal desde esta. Para ello, es necesario que se haya realizado previamente la desconexión. Si se intentara salir a la pantalla principal, se mostrará un mensaje de que es necesario desconectar del robot.

Se ha configurado en cada una de las pantallas, que todos los campos de introducción de texto solo sean datos numéricos. Solo en el caso del campo de introducción de la IP, se da la posibilidad de introducir el carácter "**.**" Y en los datos numéricos de posiciones, orientaciones y ejes, que puedan ser valores negativos.

La forma de borrar los datos de las casillas de introducción de texto, tanto en esta pantalla como en la de "Mover XYZ", "Mover Ejes" y "Dibujar Pizarra", es dejando una de las casillas en blanco y enviar el dato. Se indicará que hay un error en la introducción de datos, y el archivo relacionado con esos datos se borrará.

En la ***Figura 6.9***, se muestra el diagrama de flujo del funcionamiento de la pantalla de conexión con el robot.

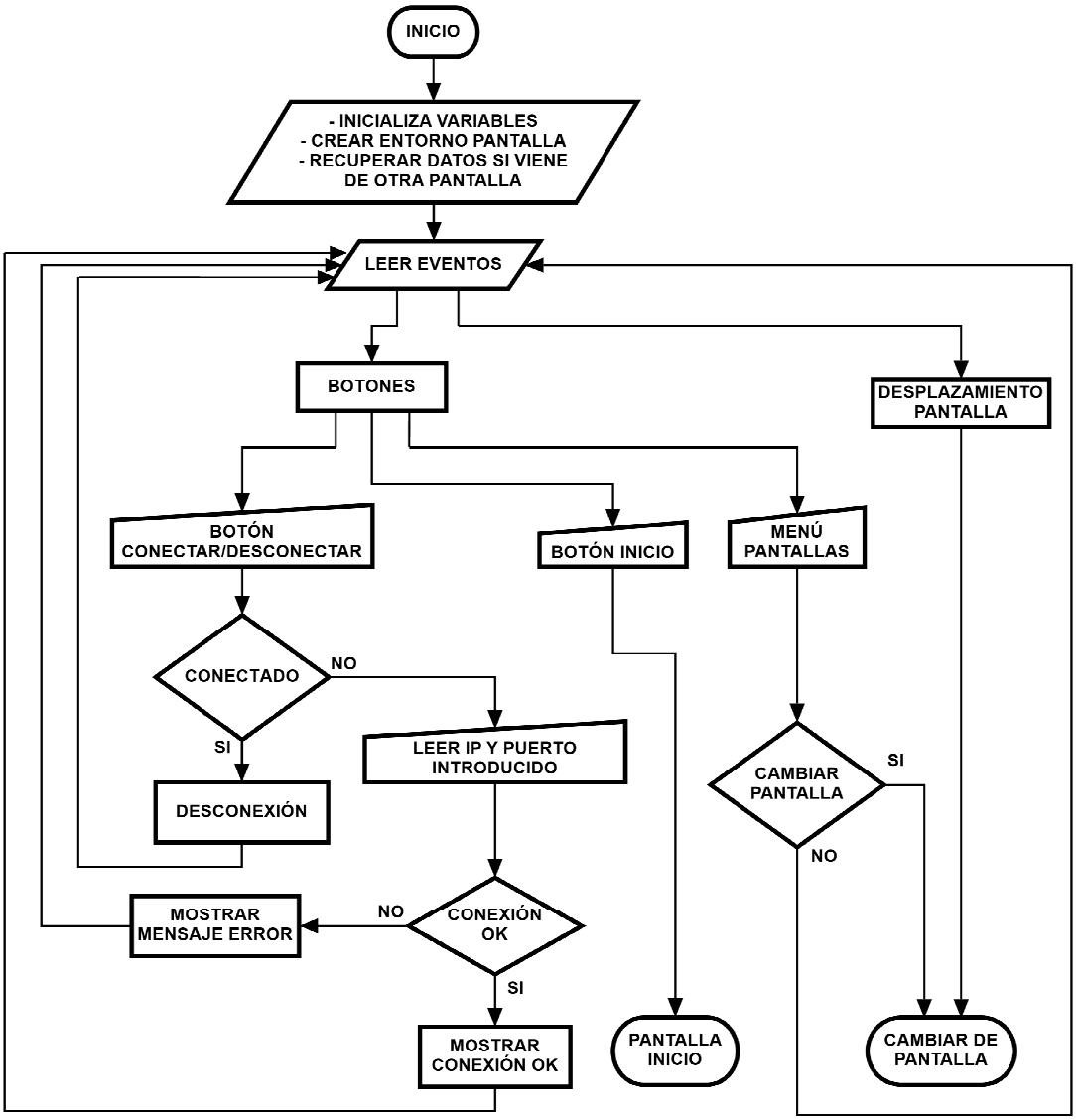


Figura 6.9. Diagrama flujo pantalla Conexión Robot

6.3.1.2.3.- Mover XYZ

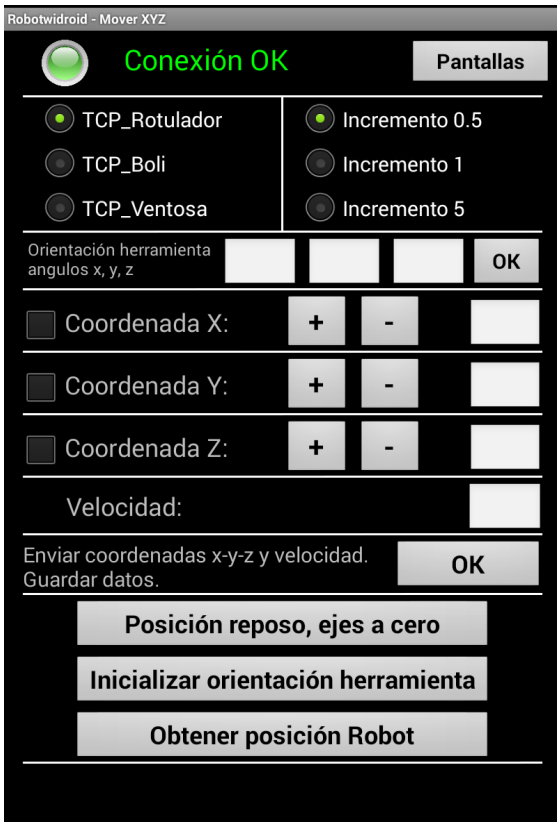


Figura 6.10. Pantalla Mover XYZ

- Desde esta pantalla se podrán realizar los movimientos en X-Y-Z y la reorientación de cada herramienta.
- Será posible trabajar con los dos mandos externos a la tablet, el teclado externo y el mando gamepad. En el ***apartado 6.3.1.3. Configuración teclas accesorios,*** se indicará la configuración de cada uno de los mandos.
- Se podrá seleccionar la herramienta con la que se desea trabajar y el incremento en los movimientos.
- Tres campos de introducción de texto permitirán especificar los grados de orientación de la herramienta que se haya seleccionado.
- Cuatro campos más, para el punto en el espacio X-Y-Z de la herramienta, así como la velocidad que se quiera configurar para los movimientos del robot.

- Los datos de la posición X, Y, Z y velocidad, cuando sean enviados al robot, serán guardados en un archivo privado de la aplicación para la próxima vez que se acceda a esta pantalla. Los datos de la orientación de la herramienta se guardarán en otro archivo distinto cuando se envíen al robot.
- Manualmente, desde esta pantalla será posible hacer los movimientos del robot, especificando la herramienta y el incremento de este.
- Según el botón "**+**" o "**-**" que se pulse para cada coordenada, el campo de texto situado a la izquierda de los botones, cambiará a color verde para indicar que grupo de botones se ha accionado.
- A la izquierda del mencionado campo de texto, se sitúa una casilla de selección que permitirá indicar si se quiere hacer un movimiento de incremento/decremento en cada una de las coordenadas, o una reorientación de la herramienta. En este caso, los campos de texto seleccionados cambiarán para indicar esta operación.
- Un botón nos permitirá indicar al robot para que se coloque en la posición de reposo, esto es, con todos los ejes a cero.

- Otro botón permitirá inicializar la orientación de la herramienta seleccionada. Podría ocurrir que el robot no realizara un movimiento, debido a que generaría un error al moverse a esa posición o porque la orientación de su herramienta no es la correcta para hacer tal movimiento. En este caso, sería posible inicializar la orientación de la herramienta en la posición que se encontrara en ese momento para poder realizar el desplazamiento.

Para comprender el funcionamiento de este botón, se muestra el diagrama de la ***Figura 6.11***.

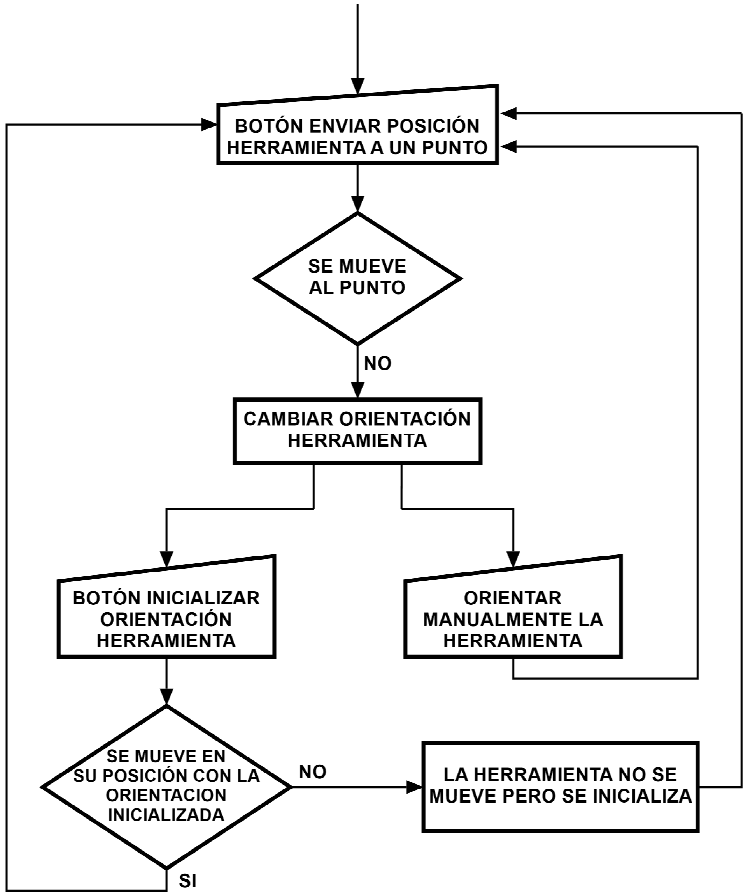


Figura 6.11. Inicializar Orientación Herramienta

- Finalmente, otro botón podrá enviar al robot la instrucción de que le proporcione la velocidad, posición X-Y-Z y orientación de la herramienta seleccionada, que pasarán a los campos de introducción de texto correspondientes. Si se quieren guardar, bastará con enviarlos al robot. El robot no realizará ningún movimiento en este caso porque ya estaba posicionado allí, pero los datos serán guardados.

En la ***Figura 6.12***, se muestra el diagrama de flujo de esta pantalla.

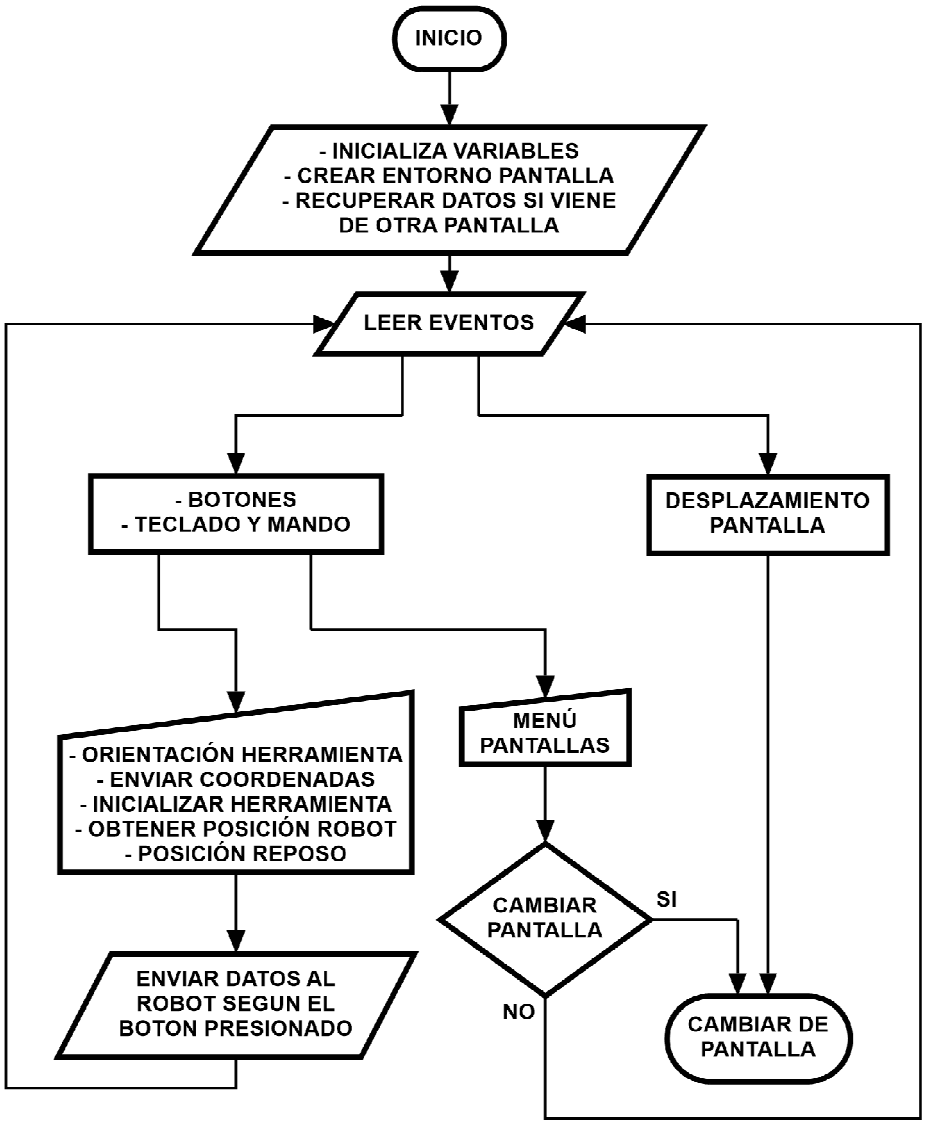


Figura 6.12. Diagrama flujo pantalla Mover XYZ

6.3.1.2.4.- Mover Ejes

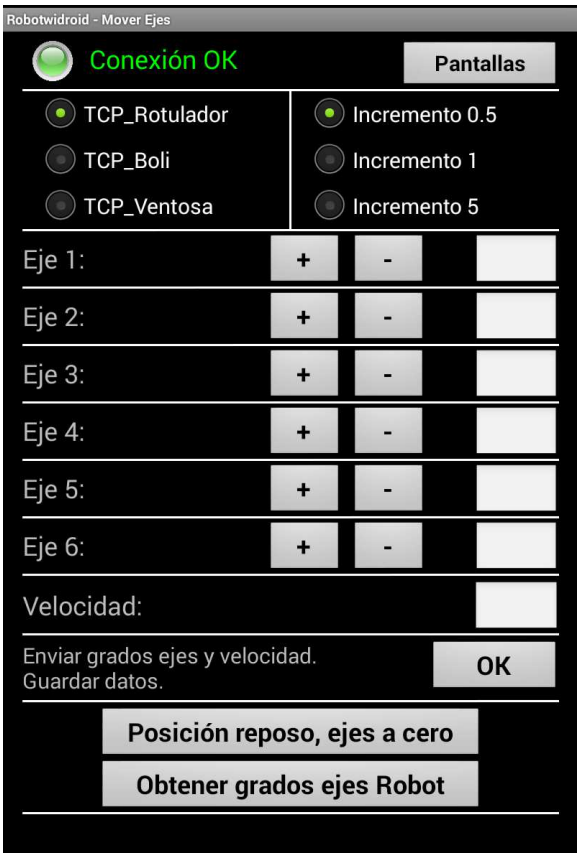


Figura 6.13. Pantalla Mover Ejes

- El funcionamiento de esta pantalla, es similar al de movimiento en X-Y-Z, pero para la variación de los ejes del robot. Su funcionamiento se resume en el diagrama de la ***Figura 6.14***.
- En esta pantalla, el cambio de herramienta no afecta al movimiento de ejes que se quiere enviar, pero se ha dejado para posibles configuraciones de otros robots.
- Se tiene un grupo de introducción de texto de los ejes (del 1 al 6), con el correspondiente botón de envío de los datos y guardado de estos. Otra casilla, permitirá especificar la velocidad del movimiento.
- Un botón permitirá que se obtengan los grados de los ejes y la velocidad. Estos valores se colocarán en los campos de introducción de texto correspondientes.

- Similar a la pantalla de envío de coordenadas X, Y y Z, también será posible variar cada eje con dos botones asignados para cada uno de ellos. Uno para aumentar los grados del eje y otro para disminuirlo. Según el botón que se pulse "**+**" o "**-**" de cada uno de los ejes, el campo de texto a la izquierda, cambiará a color verde para indicar que grupo de botones se ha accionado.
- Los límites de los ejes 1 a 6, se pueden consultar en las características del robot en el ***Anexo*** (***Figura 1.1***).
  - Eje 1: de -165º a +165º
  - Eje 2: de -110º a +110º
  - Eje 3: de -110º a +70º (El robot utilizado tiene un rango de -90º a +70º)
  - Eje 4: de -160º a +160º
  - Eje 5: de -120º a +120º
  - Eje 6: de -400º a +400º

En la ***Figura 6.14***, se muestra el diagrama de flujo de su funcionamiento.

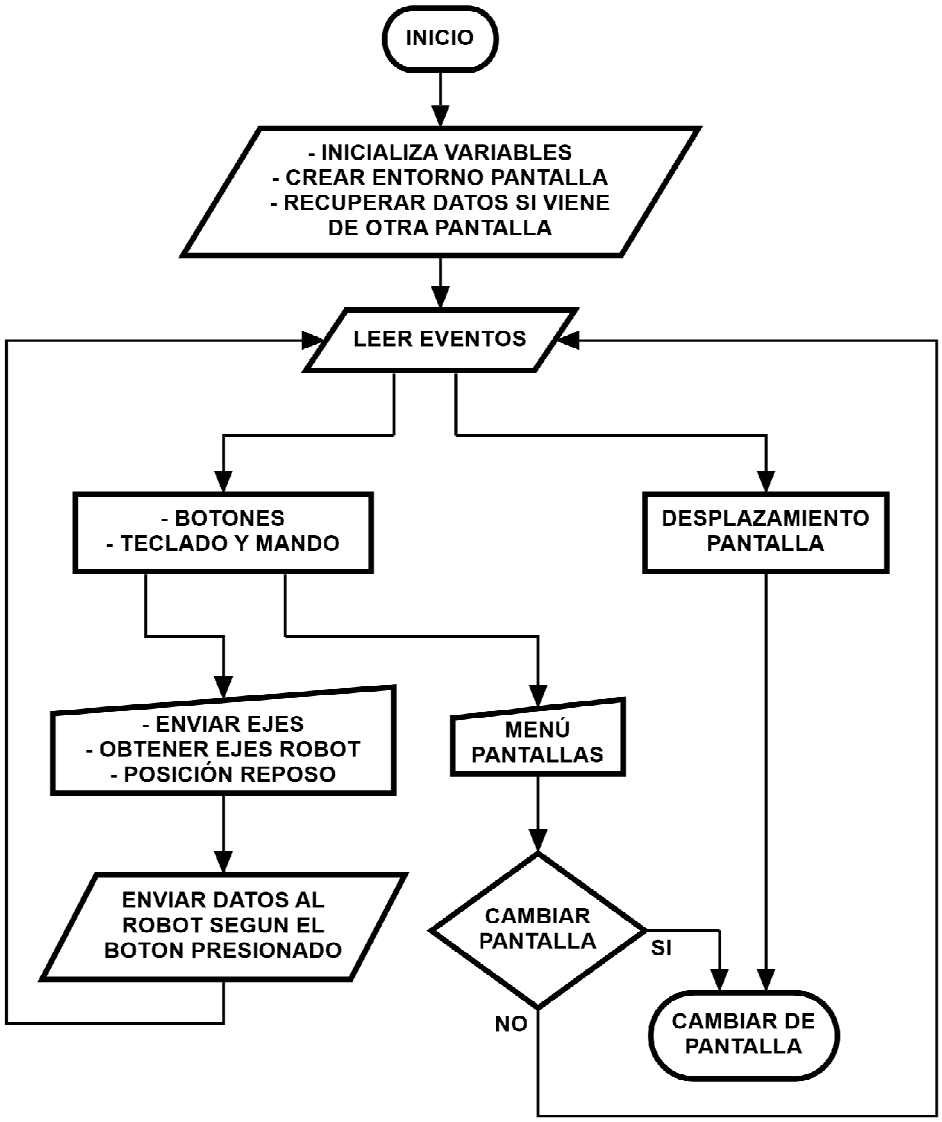


Figura 6.14. Diagrama flujo pantalla Mover Ejes

6.3.1.2.5.- Dibujar Pizarra

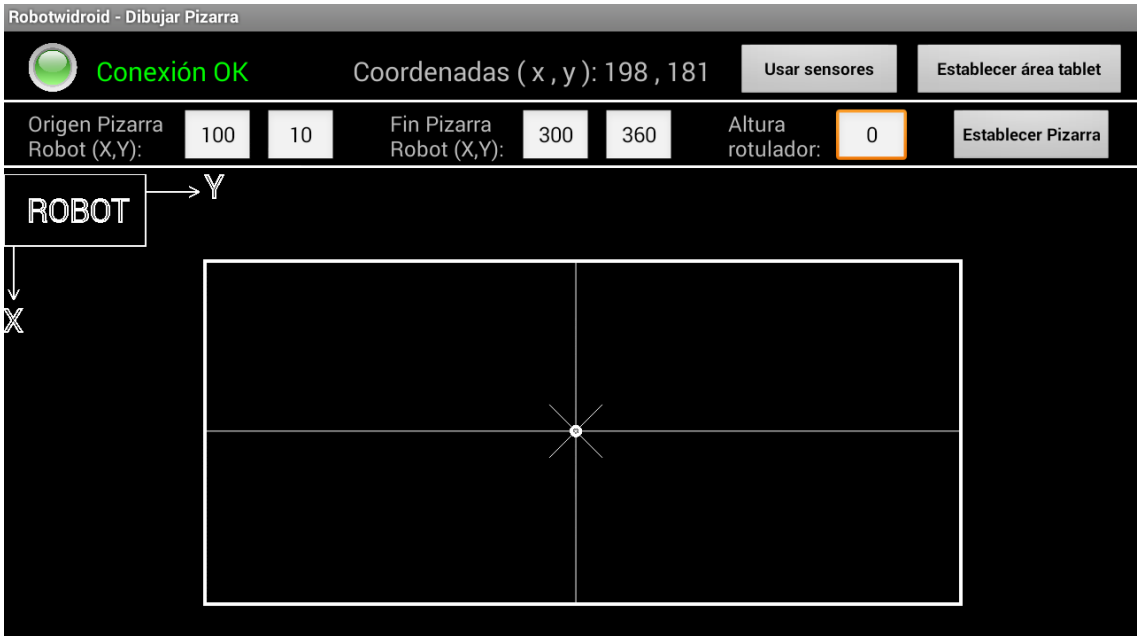


Figura 6.15. Pantalla Dibujar Pizarra

- En esta pantalla no se han configurado los mandos externos (excepto para el cambio de pantalla), ya que está diseñada para operar de forma táctil y para la herramienta rotulador.
- Se ha definido una zona para establecer el área de trabajo del robot. Mediante un rectángulo simulará la pizarra del robot. Una zona del dibujo, indica la situación de la base del robot.
- Si no se ha definido previamente el área de trabajo del robot, es posible mediante el botón habilitado para ello. Esto se hará mediante tres pulsaciones del botón “Establecer área tablet”, situado arriba a la derecha.
  1.- La primera pulsación, si el área estaba definida, la borrará. A la vez, se pedirá que se pulse en el punto más cercano al robot, la esquina superior izquierda del área de la tablet, teniendo la tablet orientada como indica la ***Figura 6.15***. Una cruz inclinada a 45º, permitirá saber dónde se está pulsando.
  2.- Una vez escogido este punto, se pulsará de nuevo el botón. Se pedirá situar un segundo punto que determinará el área de la pizarra del robot, la esquina inferior derecha del área de la tablet. A su vez se irá dibujando el rectángulo de la pizarra mientras se desplace el dedo por la pantalla.
  3.- Una vez decidida el área de la pizarra, se pulsará de nuevo el botón y quedará establecida.

- Cuando se pulse en un punto dentro del área de la pizarra, se dibujarán dos rectas que indicarán donde se quiere que se mueva el robot. Si se pincha fuera de esta área, se mostrará que el punto está fuera de la pizarra.
- Para determinar las coordenadas de la pizarra del robot dentro de su área de trabajo se hará mediante la introducción en unos campos de texto de las coordenadas X-Y de la pizarra (dos para el origen y dos para el final). También es posible determinar la altura del rotulador. Solo son posibles valores numéricos y positivos.
- Según el punto donde se presione en la pizarra, se indicará en un campo de texto, cuales son las coordenadas que se están enviando al robot.
- Si los movimientos son muy rápidos, el robot podría ir como a "tirones". Por eso los movimientos tienen que ser de forma que al robot le dé tiempo a llegar a las coordenadas enviadas. Cuando los movimientos son lentos, y se deja de pulsar en la tablet, el rotulador se elevará.
- Si se pulsa y se levanta en una posición lejana a la que está situado, el robot se posicionará allí pero el rotulador se quedará abajo. Pulsando fuera de la pizarra, el rotulador se levantará.
- Finalmente, un último botón, permitirá usar el sensor acelerómetro de la tablet para realizar los movimientos del robot.
    - Pulsando en él, permitirá que inclinando la tablet se mueva el robot. Esto solo será posible dentro del área establecida de la pizarra. Si se intenta mover fuera, la cruz que indica la posición del robot, no se moverá más allá de ese punto. Pero es posible volver dentro del área.
    - Si posteriormente se pulsa en un área de la pantalla (dentro o fuera de la pizarra), dejarán de utilizarse los sensores. Lo mismo ocurre cuando se pulsa el botón habilitado para los sensores.
- Hay una consideración a tener en cuenta. Para realizar el cambio de pantalla, como se verá en el ***apartado 6.3.3. Transferencia de datos entre programas***, Android debe cerrar y abrir el socket creado. Si se quiere hacer esto mientras el robot está en movimiento, puede producirse un error inesperado en Android. Por seguridad, para evitarlo, hay dos posibilidades en las que se permite el cambio de esta pantalla:
    - Después de un movimiento táctil, pinchando fuera de la pizarra y observando que el rotulador se eleva. En este momento estamos seguros de que ha recibido esta instrucción y el robot se ha detenido.
    - Después de usar los sensores del acelerómetro. Estos movimientos son más pausados, y cuando se deja de usar los sensores, se permite también el cambio de pantalla.

- Para establecer la velocidad de movimiento hay dos posibilidades:
  1.- Si se accede desde la pantalla “Conexión Robot”, la velocidad será de 1000(mm/s).
  2.- Si se accede desde “Mover XYZ” o “Mover Ejes”, es posible configurar otra velocidad. Cuando se acceda a la pantalla “Dibujar Pizarra” se pasará la velocidad configurada. Para que esta velocidad pueda pasar, es necesario que cuando se incluya en el campo de introducción de texto de la velocidad, se realice la petición de enviar.

Todas estas funciones se resumen en el diagrama mostrado en la ***Figura 6.16***.

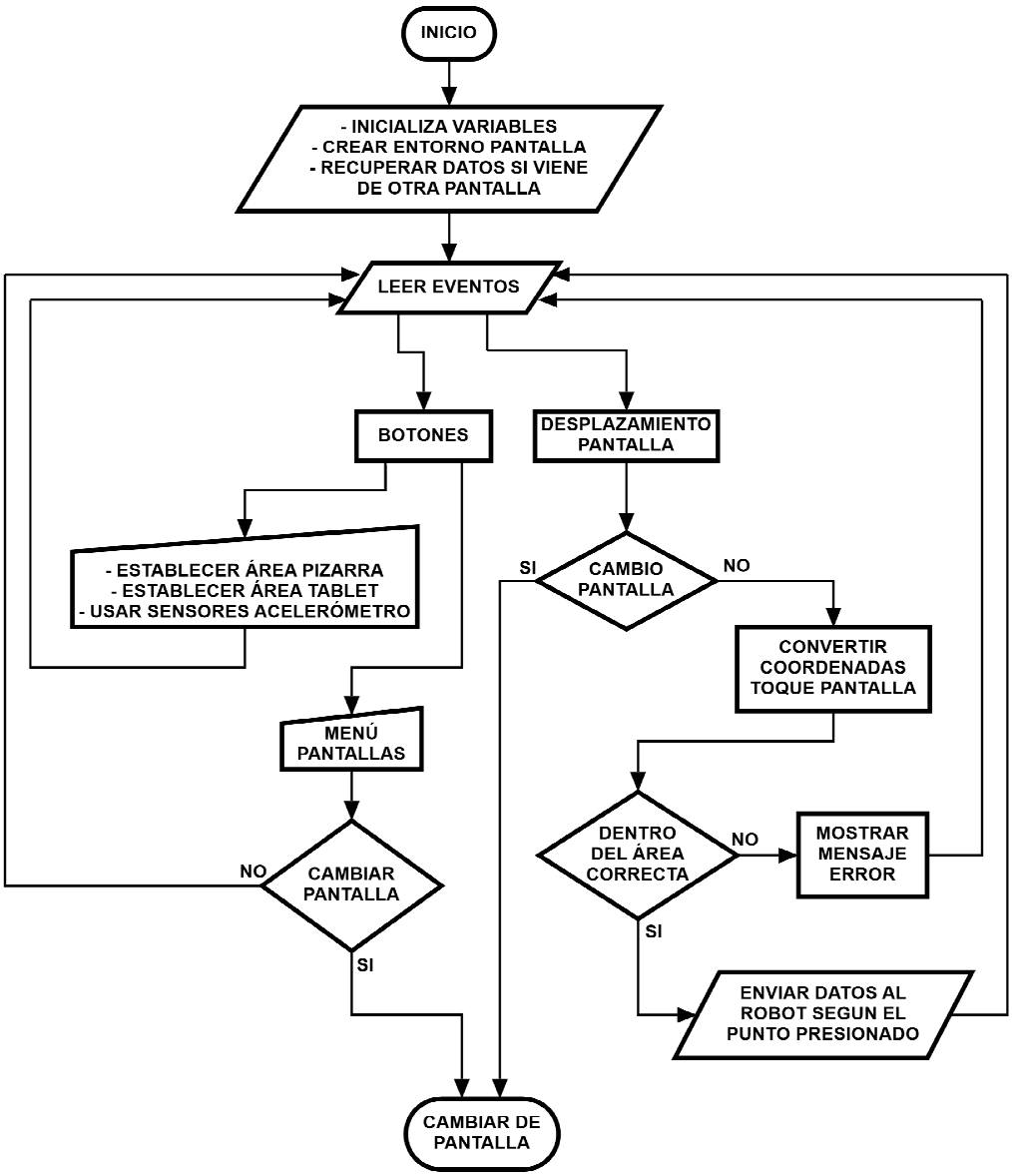


Figura 6.16. Diagrama flujo pantalla Dibujar Pizarra

6.3.1.3.- Configuración teclas accesorios

Hay 2 aspectos importantes a tener en cuenta con estos accesorios:

1.- Para poder utilizar el gamepad en el Nexus 7, es necesario que esté previamente desincronizado. Si usáramos este accesorio, después lo apagáramos y volviéramos a encenderlo, estaría unos segundos vinculado pero después empezaría a parpadear el gamepad hasta apagarse.
También se apaga el dispositivo después de 8 minutos sin utilizarse.
   - Para realizar los pasos de desvincular el gamepad, puede consultarse en el ***Anexo 2.1.1. Desincronizar gamepad.***
   - Una vez Desincronizado, para vincularlo, consultar el proceso en el ***Anexo 2.1.2. Vincular gamepad.***

2.- Utilizando el teclado o el mando, cuando se quieren introducir datos en las casillas de coordenadas, ejes, etc., puede ocurrir que no se abra el teclado del propio terminal Android. Depende del terminal utilizado, ya que se ha comprobado en un Smartphone Samsung Galaxy S2 que esto no ocurre. Pero en la tablet Nexus 7 empleada, cuando se conecta alguno de estos accesorios, no se despliega el teclado táctil. Esto se puede solucionar con una simple configuración del terminal Android de la siguiente manera:
   a. Ajustes.
   b. Idioma e introducción de texto.
   c. Teclado y métodos de introducción.
   d. Predeterminado.
   e. Hardware. Teclado físico.
   f. No.

Puede consultarse como realizarlo, en el ***Anexo 2.2. Ajustes teclado*.**

Tanto el teclado como el mando, han sido configurados en las pantallas de conexión/desconexión y en las de movimientos en x-y-z y de ejes. En la pantalla principal y en la de dibujar en la pizarra, no se han configurado las teclas.

6.3.1.3.1.- Teclado externo Measy RC12 con touchpad

Figura 6.17. Configuración teclado

- **ENTER**: Conecta/Desconecta con el robot en la pantalla "Conexión Robot". Envía la posición o los ejes introducidos en la Tablet.
- **0**: Mueve al robot a la posición de reposo con los grados de los ejes a cero.
- **NUMEROS 1 A 6:** Permite realizar la rotación en torno al eje que seleccionemos. Con las flechas arriba/abajo, aumenta/disminuye los grados.
- **E**: Cambia a la pantalla Mover ejes.
- **G**: Obtiene los valores de X, Y, Z de las coordenadas del robot y la orientación de la herramienta, o los ejes en la pantalla mover ejes.
- **H**: Envía la orientación de la herramienta en la pantalla Mover XYZ.
- **M**: Cambia a la pantalla Mover XYZ.
- **P**: Cambia a la pantalla Dibuja pizarra.
- **R, B, V**: Herramientas Rotulador, Boli, Ventosa. Cambian con la tecla 7.
- **F1, F2, F3**: Incremento 0.5, 1, 5. Cambian con la tecla 8.
- **DELETE**: Reorientar o mover una herramienta.
  Para cambiar entre Reorientación y Movimiento XYZ en pantalla "Mover XYZ".
  Para reorientar, presionar DELETE seguido de X, Y, o Z.
  - **X**: Bloquea la coordenada X. Con los cursores realiza movimientos en Y y Z.
    - Las flechas arriba/abajo, aumenta/disminuye Z.
    - Las flechas derecha/izquierda, aumenta/disminuye Y.
  - **Y**: Bloquea la coordenada Y. Con los cursores realiza movimientos en X y Z.
    - Las flechas arriba/abajo, aumenta/disminuye Z.
    - Las flechas derecha/izquierda, aumenta/disminuye X.
  - **Z**: Bloquea la coordenada Z. Con los cursores realiza movimientos en X e Y.
    - Las flechas arriba/abajo, aumenta/disminuye Y.
    - Las flechas derecha/izquierda, aumenta/disminuye X.

- **ESPACIO**: Inicializa la orientación de la herramienta.
- **ESC**: Salir en “Conexión Robot”. En otra pantalla, pasa a “Conexión Robot”.
- **TOUCHPAD**: Puede ser utilizado en todas las pantallas como si se tratara de un ratón. En la pantalla de Dibujar pizarra es muy útil para mover el robot.

6.3.1.3.2.- Mando gamepad Ipega PG-9025

Figura 6.18. Configuración mando

- **HOME:** Para vincular por bluetooth presionaremos “**A + HOME”** y buscaremos el dispositivo en Android. Ver ***Anexo 2.1.2. Vincular gamepad***.
- **SELECT**: Para cambiar entre Reorientación y Movimiento XYZ en “Mover XYZ”.
- **START**:
  - Conectar/Desconectar en la pantalla “Conexión Robot”.
  - Envía los datos introducidos de posición (Mover XYZ) o de ejes (Mover Ejes), al robot.
- **JOYSTICK IZQUIERDO**:
  - Derecha/Izquierda: Aumenta/Disminuye:
    - Coordenada X. Pantalla “Mover XYZ”.
    - Eje 1. Pantalla “Mover Ejes”.
  - Arriba/Abajo: Aumenta/Disminuye:
    - Coordenada Y. Pantalla “Mover XYZ”.
    - Eje 2. Pantalla “Mover Ejes”.
  - Presionando, obtiene la posición XYZ y orientación de la herramienta (“Mover XYZ”) o los ejes del robot (“Mover Ejes”).

- **JOYSTICK DERECHO**:
  - Derecha/Izquierda: Aumenta/Disminuye Eje 3.
  - Arriba/Abajo: Aumenta/Disminuye:
    - Coordenada Z. Pantalla “Mover XYZ”.
    - Eje 4. Pantalla “Mover Ejes”.
  - Presionando, mueve el robot a la posición de reposo, con todos los ejes a cero en las pantallas “Mover XYZ” y “Mover Ejes”.
- **TECLAS DIRECCION**:
  En la pantalla Mover Ejes:
  - Derecha/Izquierda: Aumenta/Disminuye Eje 5.
  - Arriba/Abajo: Aumenta/Disminuye Eje 6.
- **TECLAS FUNCION MULTIMEDIA**: Para cambiar de pantalla. Cuando se está en una pantalla, la tecla correspondiente a la suya no tendrá ninguna función.
  - **-** : En la pantalla “Conexión Robot” pasa a la de Inicio.
  - ⏮ : Pantalla “Conexión Robot”.
  - ⏯ : Pantalla “Mover XYZ”.
  - ⏭ : Pantalla “Mover Ejes”.
  - **+** : Pantalla “Dibuja Pizarra”.
- **R1**: Cambia el incremento.
- **R2**: Envía la orientación de la herramienta.
- **L1**: Cambia la Herramienta.
- **L2**: Inicializa la orientación de la herramienta.
- **Tecla abajo**: Cuando se introduce texto, si no responde el teclado externo, entonces hay que dar varias veces con la tecla del cursor hacía abajo para saltar las líneas hasta que reconoce el teclado.
- También es posible introducir números en las casillas. La relación entre las teclas y el número asignado a cada una de ellas es:
  - **X** (1), **A** (2), **B** (3), **Y** (4), **L2** (5), **R2** (6), **L1** (7), **R1** (8).
  - **Presionando Joystick izquierdo**: (9).
  - **Presionando Joystick derecho**: (0).
  - **Moviendo Joystick izquierdo**: se desplaza entre las casillas.

Las teclas X, Y, A y B no se han configurado porque coinciden con funciones del teclado externo.

6.3.2.- RAPID

Al ejecutar el programa del robot, se moverá a la posición de reposo (con los ejes 1 a 6 con valor 0º) y se pedirá confirmación de la IP y puerto guardados. Si no es la que se indica, es posible introducirlos. Para la IP, se introducirán por separado cada uno de los 4 grupos que la componen (valor, seguido de "OK").

6.3.2.1.- Procedimiento conexión

En primer lugar, ya que se ha establecido que su conexión sea de Servidor, creará un socket para una conexión entrante, en una IP y puerto que se definirán de la siguiente manera:

- Si la conexión se realiza en red Local:
  1.- Obtendremos la IP Local donde está conectado el robot. En el caso del robot del Laboratorio es *10.1.107.15*.
  2.- Estableceremos un puerto aleatorio. Es conveniente usar puertos que no se usen para otras conexiones. Por ej. pondremos el *12345*.
  3.- Estos datos tienen que suministrarse al usuario que vaya a utilizar el terminal Android objeto de este trabajo.
  4.- El robot quedará a la espera de la conexión.

Una vez que se realiza la conexión con éxito, la transferencia del paquete de datos por parte de Android, puede ser de la siguiente forma:

- La longitud del paquete de datos de la instrucción que envía Android es variable. Estará entre 5 y 33 caracteres, sin contar el carácter de inicio de paquete ("#") y el número de caracteres a leer (este dato será de dos caracteres). Esto es así porque es necesario leer la cadena correcta para realizar las operaciones en RAPID y evitar errores. Por lo tanto, por seguridad se determina si la cadena tiene más de 2 caracteres, a partir del cual estará el carácter "#". Los 2 anteriores los introduce RAPID.
- Seguidamente se comprobará si el movimiento que debe hacer el robot es en las coordenadas X-Y-Z, una reorientación de la herramienta o un movimiento de ejes. También se comprueba si se ha enviado la petición de mover a la posición de reposo, inicializar una herramienta o desconectar.
- También se tiene en cuenta si se ha cambiado de pantalla en la aplicación Android. Esto es porque en Android es necesario que reinicie la conexión, ya que no es posible pasar el dato del socket entre pantallas. Por lo tanto debe indicarle al robot que ha realizado una Desconexión/Conexión. De forma que el robot debe reconectarse a la vez.

El diagrama de flujo de la programación en RAPID puede resumirse en la ***Figura 6.19*** mostrada a continuación:

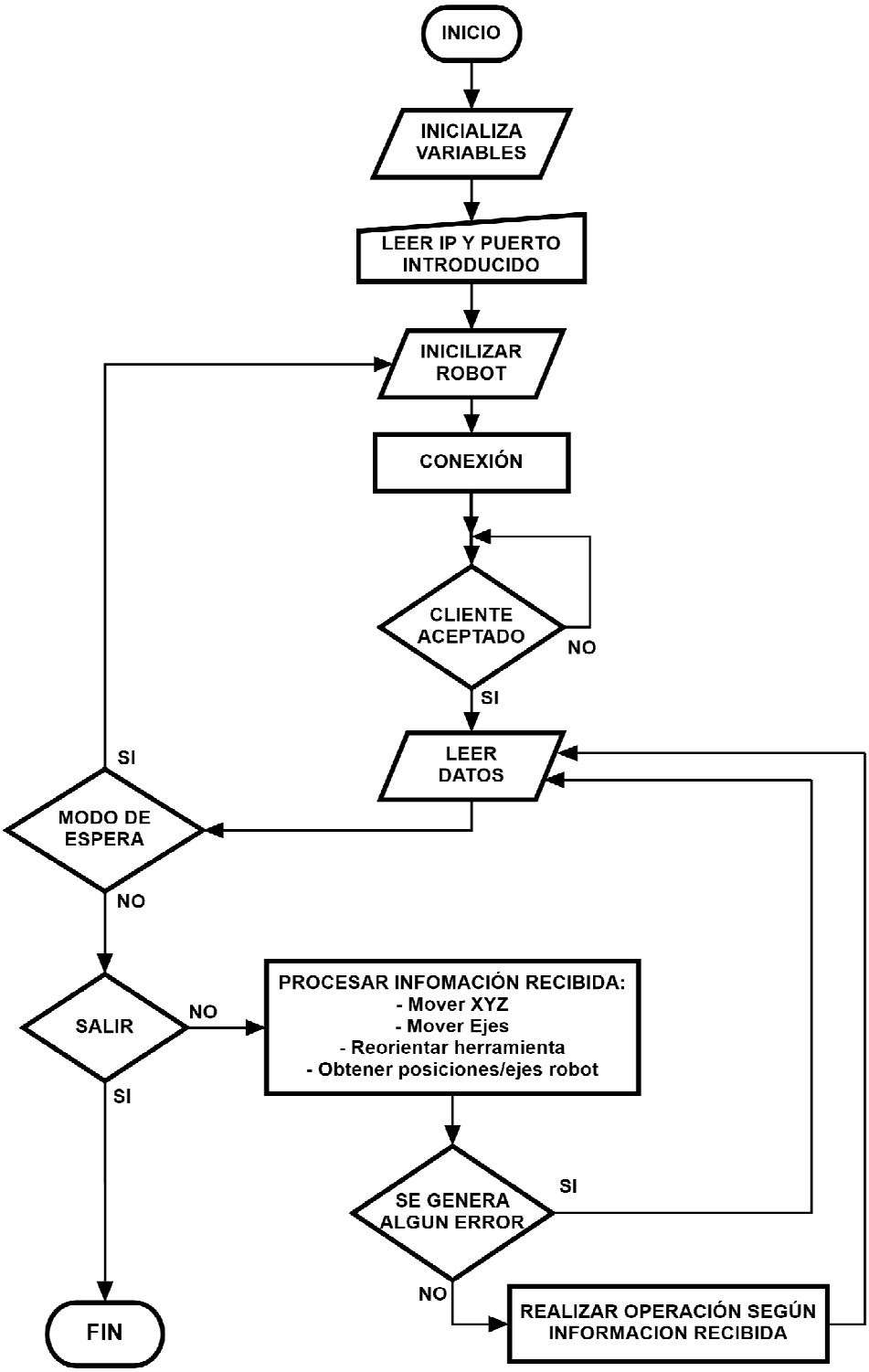


Figura 6.19. Diagrama flujo programa RAPID

6.3.3.- Transferencia de datos entre programaciones

6.3.3.1.- De Android a RAPID

Las posibilidades del paquete de datos son las que se muestran en la **Tabla 1**.

OPERACIÓN		PAQUETE DATOS											
		1	2	3	4	5	6						
1	Desconexión	#	05	DES	C	O							
2	Desconexión total	#	05	SAL	I	R							
3	Cambiar de pantalla	#	05	NEW	A	C							
4	Enviar posición reposo	#	05	000	0	0							
5	Inicializar herramienta	#	05	INI	B	1							
6	Enviar orientación herramienta	#	17	ORI	R	1	+180	-045	+090				
7	Reorientar herramienta	#	05	RY-	V	5							
8	Obtener posición y orientación	#	05	PYO	V	5							
9	Mover X-Y-Z	#	05	PX+	R	1							
10	Enviar posición	#	21	POS	R	1	+200	+100	+300	1000			
11	Dibujar pizarra	#	21	POZ	R	1	+200	+100	+005	1000			
12	Enviar ejes	#	33	JOI	R	5	+010	-020	+050	+045	-060	-200	0100
13	Obtener ejes	#	05	JJJ	R	1							
14	Mover ejes	#	05	J3+	R	1							

Tabla 1. Paquete datos Android a RAPID

- Columna 1 (paquete datos): Carácter inicial (1 carácter).
- Columna 2 (paquete datos): Número caracteres a leer (2 caracteres).
- Columna 3 (paquete datos): Tipo movimiento (3 caracteres).
- Columna 4 (paquete datos): Herramienta (1 carácter).
- Columna 5 (paquete datos): Incremento (1 carácter).
- Columna 6 (paquete datos): Otros datos (hasta 28 caracteres).

El carácter "#" es enviado como marcador para reconocer el inicio del paquete de datos que es de longitud variable. Aunque a continuación se envía la longitud de caracteres a interpretar por el robot, es necesario un marcador debido a que Android puede estar continuamente enviando datos, como en el caso de la pantalla "Dibujar Pizarra". De esta forma RAPID no interpretará una lectura de datos incorrecta.

Los paquetes de datos en que se realizan movimientos, las siglas posibles del tipo de herramienta y de incremento (***Tabla 1, Paquete datos-Columnas 4 y 5***), son:

- **Herramienta.** Columna 4.
  - **R**. Rotulador.
  - **B**. Boli
  - **V**. Ventosa.
- **Incremento/Decremento.** Columna 5.
  - **1**. 0,5mm/0,5º (desplazamiento/grados) en el robot.
  - **2**. 1mm/0,5º (desplazamiento/grados) en el robot.
  - **5**. 5mm/5º (desplazamiento/grados) en el robot.

En el tipo de movimiento (***Tabla 1, Paquete datos-Columnas 3 a 6***), se encuentran diferentes operaciones de envío de datos que son:

1. **DESCO**. Envía la petición de realizar la desconexión con el robot. Si se quisiera hacer una nueva conexión, podría realizarse, ya que el robot se queda en modo de espera.
2. **SALIR**. Se enviará cuando se sale de la aplicación, cerrando definitivamente el socket creado por el robot en espera de una conexión. Si posteriormente se entra en la pantalla de conexión, es necesario reiniciar el programa del robot.
3. **NEWAC**. Debido a que en Android cuando se cambia de pantalla, hay que cerrar, crear sockets y realizar una nueva conexión, debe informarle al robot que debe cerrar y volver a abrir el suyo. Android enviará este paquete de datos. Es muy importante antes de cambiar de pantalla, que el robot esté parado, de lo contrario la aplicación Android podría tener un error inesperado.
4. **00000**. Le pide al robot que se coloque en la posición de reposo. En esta programación se ha establecido que sea con todos los ejes con valor a cero. Se podría configurar otra posición, con solo cambiarlo en RAPID. Además se inicializan las variables del robot.
5. **INIB1**. Se inicializan las variables de la herramienta seleccionada, en este ejemplo la punta boli. Esta instrucción coloca la herramienta perpendicular con respecto al plano horizontal del robot, como puede verse en la ***Figura 5.3*****.**
6. **ORIR1+180-045+090**
   - Esta cadena de datos es utilizada en la pantalla “Mover XYZ”. Enviará al robot una orientación de la herramienta Rotulador con (ángulo x, ángulo y, ángulo z), de (180º, -45º, 90º) en la posición que se encuentre en ese momento. El robot se moverá a esa orientación si es posible, es decir, que no genere ningún error. De lo contrario, se quedará en su misma orientación.

7. **RY-V5**
    - **R**. Realiza una reorientación de la herramienta del robot.
    - **Y-**. Disminuye el eje Y. Las otras posibilidades son X+, X-, Y+, Z+, Z-.
    - **V**. La herramienta seleccionada es la Ventosa.
    - **5**. El incremento es de 5.
8. **PYOV5**
    - **PYO**. Se realiza la petición de obtener las coordenadas x-y-z, la orientación y velocidad de la herramienta especificada.
    - **V**. La herramienta seleccionada es la Ventosa.
    - El valor del incremento puede ser cualquiera.
9. **PX+R1**
    - **P**. Ejecuta un movimiento en x, y, z.
    - **X+**. Realiza un incremento del robot en el eje X. Las otras posibilidades son X-, Y+, Y-, Z+, Z-.
    - **R**. La herramienta seleccionada es el Rotulador.
    - **1**. El incremento es de 1.
10. **POSR1+200+100+3001000**
    - Se realizará un posicionamiento de la herramienta rotulador en x, y, z (200, 100, 300) a una velocidad de 1000 que corresponde en el robot a 1000mm/s (este valor puede ser hasta 7000). Para ver las distintas posibilidades de la velocidad, consultar el manual de instrucciones de RAPID de la página oficial de ABB, ***Bibliografía 10.2 [11]***.
    - Para el posicionamiento no será importante el valor 1 de incremento. Dato posterior al carácter “R”.
11. **POZR1+200+100+0051000**
    - Este paquete de datos es utilizado en la pantalla “Dibujar Pizarra”. Se realizará un posicionamiento del Rotulador robot en x, y, z (200, 100, 005) a una velocidad de 1000 que corresponde en el robot a 1000mm/s. Este último valor no se introduce en la pantalla de “Dibujar Pizarra”. El valor de la velocidad puede ser determinado de dos formas:
        - Pasando desde la pantalla en que se conecta con el robot. El valor de la velocidad será de 1000.
        - Desde la pantalla de mover en x, y, z o desde la de ejes, el valor de la velocidad que se haya configurado y guardado, se pasará a la pantalla de dibujar en la pizarra.
12. **JOIR5+010-020+050+045-060-2000100**
    - Enviará al robot una configuración de ejes (Eje1, Eje2, Eje3, Eje4, Eje5, Eje6) de (10º, -20º, 50º, 45º, -60º, -200º) a una velocidad de 100mm/s.
    - Los caracteres “R5” no serán importantes en esta cadena.

13. **JJJR1**
    - **JJJ**. Realiza la petición de obtener los grados de los ejes del robot.
    - **R1**. Puede ser cualquier valor. Para la configuración de este robot no es determinante. Se ha dejado esta posibilidad por si se quisiera adaptar la programación para otro tipo de robot. Además, para instrucciones como esta, el mínimo de caracteres que se envía al robot es de 5. Se completa el dato enviado para evitar errores en la lectura de datos.
14. **J3+R1**
    - **J3+**. Se realiza una operación sobre el eje 3 del robot. Los ejes van desde J1 a J6. Se hará un incremento en el movimiento.
    - **R**. Este dato es para completar la cadena.
    - **1**. El incremento es de 1.

Los datos de posición x, y, z que se envían al robot se referenciarán de la siguiente manera:

- En la pantalla “Mover XYZ”, la posición en la que se colocará el robot, es referenciado al origen 0, 0, 0 de la base del robot.
- Estas coordenadas se diferencian de las enviadas por la pantalla “Dibujar Pizarra”. En la configuración del entorno de trabajo del robot, está establecido el origen de la pizarra en la posición (132.2, -84.5, 30) con respecto al origen del robot.

Para poder desplazar correctamente sobre el área de la pizarra, sin que el robot tenga que cambiar la orientación ni la configuración de la herramienta, se ha comprobado que el área aceptable esté dentro de las coordenadas X-Y de la pizarra con origen x, y (100, 10) y final (300, 360). Todos estos valores son en milímetros. Estos datos son los que podemos introducir en la pantalla “Dibujar Pizarra”.

De forma que, si posicionamos en la pizarra el robot en la pantalla “Dibujar Pizarra”, estaremos moviéndonos con respecto al origen de coordenadas de la pizarra blanca. Mientras que, como la pizarra esta referenciada con respecto al robot en la posición (132.2, -84.5, 30), la posición del punto en la pizarra con respecto al robot (0, 0, 0), será de (132.2+x, -84.5+y, 30+z).

Para convertir los datos obtenidos de la posición en la tablet a la hora de dibujar en la pizarra, se ha usado el proceso siguiente:

- En el momento de pulsar en el área de dibujo, la posición X de la tablet, corresponderá a la coordenada Y del robot, mientras que la posición Y de la tablet, corresponde a la X del robot.

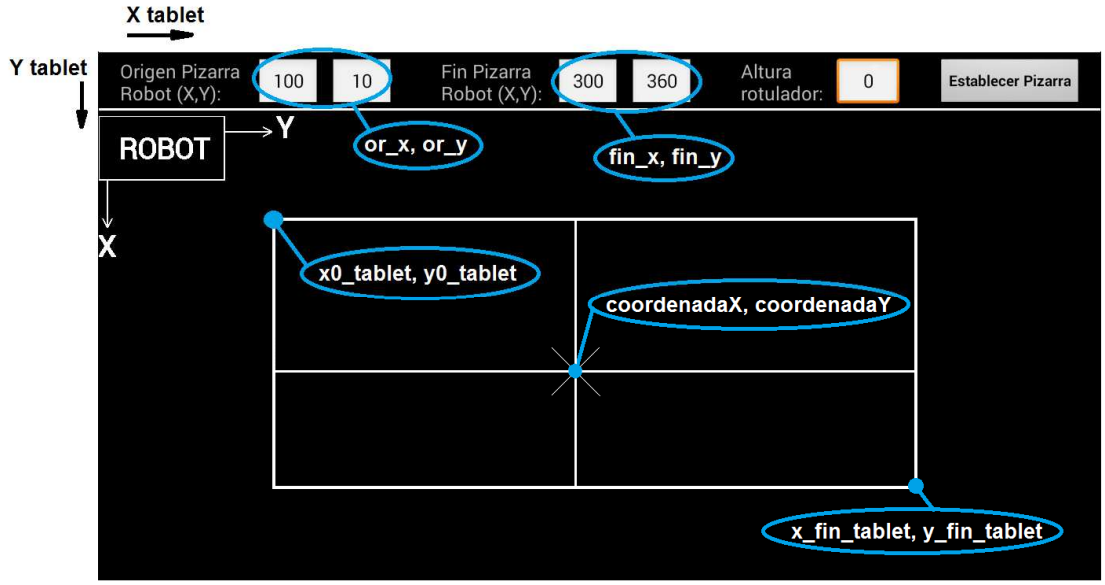


Figura 6.20. Conversión coordenadas

- Para en el caso en el que establecemos el origen de la pizarra en (100, 10) y el final en (300, 360), la conversión de coordenadas se realiza de la siguiente forma:

$$coorx\_corregida = or\_x + \frac{(coordenadaY - y0\_tablet) * (fin\_x - or\_x)}{(y\_fin\_tablet - y0\_tablet)} \quad \textit{Ecuación 6.1}$$

$$coory\_corregida = or\_y + \frac{(coordenadaX - x0\_tablet) * (fin\_y - or\_y)}{(x\_fin\_tablet - x0\_tablet)} \quad \textit{Ecuación 6.2}$$

Los valores con subíndice (tablet), coordenadaX y coordenadaY, son en píxeles. El resto son en milímetros. De la ***Ecuación 6.1 y 6.2***, se define:

- ***coorx_corregida, coory_corregida***: Coordenadas x e y enviadas al robot.
- **or_x, or_y**: Origen para dibujar con respecto al punto origen de la pizarra del robot. Valor obtenido de la introducción de datos que en este caso es (100, 10).
- **fin_x, fin_y**: Punto final para dibujar en la pizarra del robot. En este caso es (300, 360).
- **x0_tablet, y0_tablet**: Punto (x, y) de origen del espacio de dibujo en la tablet. Se establece cuando dibujamos en la tablet el área de trabajo.
- **x_fin_tablet, y_fin_tablet**: Punto (x, y) final del espacio de dibujo en la tablet. Se establece cuando dibujamos en la tablet el área de trabajo.
- **coordenadaX, coordenadaY**: El punto de coordenadas (x, y) de la tablet, obtenido al pulsar en el terminal Android.

6.3.3.2.- De RAPID a Android

RAPID enviará información cuando se lo indique Android de tres formas diferentes, como se muestra a continuación en la ***Tabla 2***.

Estos ejemplos, son extraídos de la ***Tabla 1*** mostrada en el ***apartado 6.3.3.1.***

		OPERACIÓN RECIBIDA	PAQUETE DATOS RECIBIDOS				
			1	2	3	4	5
1	Tabla 1. Fila 3	Cambiar de pantalla	#	05	NEW	A	C
2	Tabla 1. Fila 8	Obtener posición y orientación	#	05	PYO	V	5
3	Tabla 1. Fila 13	Obtener ejes	#	05	JJJ	R	1

Tabla 2. Instrucción Android para pedir datos a RAPID

1. **NEWAC**
   El robot recibe la instrucción de reconectar de nuevo el socket, porque Android lo hace al cambiar de pantalla. RAPID enviará el texto "*NEWACTIVITY*" para que Android interprete que el robot ha recibido la instrucción.
2. **PYOV5**
   Cuando reciba este dato, enviará a Android:
   - Posición x, y, z de la herramienta. En este caso la ventosa.
   - Orientación de la herramienta.
   - Velocidad de movimiento.

   Los datos serán de la forma: *x, y, z, ángulox, ánguloy, ánguloz y velocidad*. Se envían separados por "," y Android los transfiere a las casillas correspondientes como se indica en la ***Figura 2.35***, del ***Anexo 2.3.4 Mover XYZ***.
3. **JJJR1**
   Cuando reciba este dato, enviará a Android:
   - Valores de los ángulos de los ejes (eje 1 a eje 6) que tiene en el momento de recibir la instrucción.
   - Velocidad.

   Los datos se envían separados por "," y Android los transfiere a las casillas correspondientes como se indica en la ***Figura 2.37*** del ***Anexo 2.3.5 Mover Ejes***.

En estos tres casos, en los que se pide al robot que cierre el socket y vuelva a abrirlo, es recomendable que el robot no esté en movimiento. O por lo menos cuando no esté realizando una trayectoria larga. En el caso de que así se hiciera, el dispositivo Android podría quedar a la espera de recibir información del robot y ocasionar una parada inesperada de la aplicación.

7.- PROCEDIMIENTO INSTALACIÓN Y USO

Los pasos para poner en marcha el sistema son:

1.- Si es la primera vez que se instalan ambos sistemas, realizar los siguientes pasos:
   a) Instalar el archivo ***Robotwidroid.apk*** en el dispositivo Android. Para ello, conectar el terminal Android al PC y volcar el archivo como en una unidad de disco.
   b) Desde el terminal Android, buscar la ubicación donde esté el archivo y seleccionarlo para que se instale.
   c) Instalar el código RAPID en el PC que vaya a controlar el robot.
2.- Instalar el router wifi a la red donde esté conectado el robot.
3.- Encender el Flexpendant conectado al robot e iniciar el programa.
4.- Buscar en el terminal Android la red wifi y seleccionarla.
5.- Si se va a utilizar el mando gamepad, sincronizar el dispositivo.
   a) Comprobar que esté desincronizado previamente. Consultar el ***Anexo 2.1.1.***
   b) Presionar **A + HOME** en el mando y seguir los pasos mostrados en el ***Anexo 2.1.2.***
6.- Una vez sincronizado, conectar el teclado externo con el cable adaptador si va a ser utilizado.
7.- Configurar el método de entrada como se muestra en el ***Anexo 2.2*** de ajustes de teclado. De esta forma se desplegará el teclado de Android cuando seleccionemos un campo de entrada de texto.
8.- Iniciar la aplicación en el terminal Android. Es necesario que los pasos 5 a 7 se hagan antes de iniciar la aplicación.
9.- Realizar la conexión según lo indicado en el ***apartado 6.3.2.***

Y ya es posible el control del robot con el terminal Android y los accesorios.

Cuando se realiza el intento de conexión por parte del terminal Android, en el caso de que el robot no estuviera en modo de espera o se han introducido una IP o puerto incorrectos, transcurridos 10 segundos indicará que hay un fallo en la conexión.

Después de que se ha realizado una conexión con éxito, se ha operado con el robot y se realiza una desconexión, el robot quedará a la espera de que se pudiera hacer una nueva conexión. Si se sale de la aplicación, se hará una desconexión total con el robot. Para hacer una nueva conexión sería necesario reiniciar el programa del robot.

8.- CONCLUSIONES Y REFERENCIAS FUTURAS

- En el presente trabajo, la aplicación Android se ha diseñado para un tamaño de pantalla de 800x1280 pixeles. También sería posible diseñarla para diferentes tamaños de pantalla, aunque sería aconsejable que fuese para pantallas más grandes. Para pantallas más pequeñas, debido a que contienen gran cantidad de elementos, el texto podría quedar muy ajustado.
- Aparte del método de conexión en red local, como se vio en el ***apartado 6.3.2.1. Procedimiento conexión***, sería posible realizar la conexión desde un punto de acceso externo a la red. Para ello, debería hacerse lo siguiente.
  1.- Averiguar la IP externa de la red a la que está conectada el robot.
  2.- Por otro lado, debemos conocer la IP Local a la que está conectada el robot y abrir un puerto asignado a esa IP. La apertura del puerto debe realizarse por informáticos.
  3.- Una vez hecho esto, habría que suministrar al usuario Android, la IP externa y el puerto de conexión abierto para la IP Local del robot. Con estos dos datos, una vez introducidos en el terminal Android, podrá realizarse la conexión.
- Otra mejora que podría integrarse, es añadir la posibilidad de observar el robot a través de una cámara. En el entorno del robot del Laboratorio, hay presente una cámara de la empresa y máquinas de visión DALSA. Para ello habría que diseñar la creación de un nuevo socket para la conexión con dicha cámara. Si como se ha comentado antes, se pudiera realizar la conexión desde una IP externa, la cámara sería muy útil para poder controlar el robot. Además, dado que Android posibilita también el tratamiento de audio, podría añadirse un sistema de control micrófono/altavoz para realizar videoconferencia entre los sistemas.
- Existen aplicaciones que permiten controlar un ordenador por seguimiento facial, con los ojos o con movimientos de la cabeza. Esta aplicación podría rediseñarse y adaptarse a otro tipo de robot, permitiendo a personas de movilidad reducida poder controlarlo y la posibilidad de poder dibujar con toques o inclinaciones del terminal Android.

En definitiva, una vez unidas las posibles mejoras mencionadas, podrían dotar al sistema de control remoto desde un entorno ajeno al robot, con la comunicación entre robot (cámara, micrófono y altavoz) y Android (cámara, micrófono y altavoz ya integrados).

9.- PRESUPUESTO

Para el desarrollo del Trabajo Fin de Grado, han sido necesarios los elementos que se desglosan en el listado de precios que se muestran a continuación:

• Terminal Android ASUS NEXUS 7 2012	200,00 €
• Mando gamepad IPEGA PG9025	20,22 €
• Teclado inalámbrico MEASY RC12	30,00 €
• Cable OTG. Adaptador para la conexión del teclado	6,00 €
TOTAL:	***256,22 €***

El robot ABB IRB120 y el router Wifi Mikrotik (Routerboard), pertenecen a la Universidad de Vic.

10.- BIBLIOGRAFÍA Y REFERENCIAS

10.1.- ANDROID

[1] *El gran libro de Android. 3ª Edición 2013. Jesús Tomás Gironés. Editorial Marcombo. Enero-Julio/2014.*

[2] *Desarrollo de aplicaciones para Android. Edición 2013. Joan Ribas Lequerica. Editorial Anaya. Enero-Julio/2014.*

[3] *Sockets en Android Enero-Febrero/2014.*

- http://androideity.com/2012/08/05/sockets-en-android/

[4] *Android para desarrolladores. Enero-Julio/2014.*

- http://developer.android.com/index.html

[5] *Tutoriales. Enero-Abril/2014.*

- www.androidcurso.com/index.php/tutoriales-android-fundamentos
- http://elbauldelprogramador.com/video-tutorial-programacion-android/
- http://elbauldelprogramador.com/curso-programacion-android/
- https://www.youtube.com/results?search_query=Curso+Desarrollo+Aplicaciones+Android
- http://codigoprogramacion.com/android#.U-zqEvl_uSo
- http://www.tutorialandroid.com/basico/
- http://sekthdroid.wordpress.com/2013/02/08/guardar-datos-de-texto-en-memoria-interna-con-android/
- http://www.cristalab.com/videotutoriales/dibujar-pantalla-en-android-c109225l/
- http://www.javaya.com.ar/androidya/detalleconcepto.php?codigo=172&inicio=40

[6] *Consultas programación Android. Febrero-Junio/2014.*

- http://stackoverflow.com/

[7] *Historia Android. Julio/2014.*

- www.elandroidelibre.com/2011/12/un-repaso-a-la-historia-de-android.html
- http://androidzone.org/2013/05/historia-de-android-la-evolucion-a-lo-largo-de-sus-versiones/
- https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android
- http://es.wikipedia.org/wiki/Anexo:Historial_de_versiones_de_Android

[8] *Página Java. Enero-Febrero/2014.*

- www.java.com

10.2.- ABB

[9] *Página oficial ABB. Marzo-Julio/2014.*

- http://new.abb.com/es

[10] *Características robot ABB IRB120. Marzo/2014.*

- http://www.abb.es/product/seitp327/26a64e26204118e5c12576a4004a8497.aspx

[11] Instrucciones de RAPID. Marzo-Agosto/2014.

- http://www05.abb.com/global/scot/scot241.nsf/veritydisplay/688894b98123f87bc1257cc50044e809/$file/Technical%20reference%20manual_RAPID_3HAC16581-1_revJ_en.pdf

[12] *Foros consultas programación. Mayo-Julio/2014.*

- http://forums.robotstudio.com/

[13] *Historia ABB. Julio/2014.*

- http://new.abb.com/es/abb-in-spain/quienes-somos/historia
- http://centrodeartigos.com/articulos-utiles/article_108844.html

10.3.- COMUNICACIONES

[14] *Protocolo TCP. Enero/Febrero-2014.*

- http://es.wikipedia.org/wiki/Transmission_Control_Protocol

[15] *WIFI. Julio/2014.*

- http://redwifi.wordpress.com/definicion-de-wifi/

[16] *Wifi VS Bluetooth. Julio/2014.*

- www.info-ab.uclm.es/labelec/solar/Comunicacion/Wifi/Wifi_vs_Bluetooh.htm

10.4.- CARACTERÍSTICAS HARDWARE

[17] *Características Tablet. Julio/2014.*

- https://support.google.com/nexus/answer/2667663?hl=es&ref_topic=2841129

[18] *Características PC portátil. Julio/2014.*

- http://www.worten.es/store/inicio/informatica/ordenadores/portatiles/portatiles/portatil-15-5-sony-vaio-fit-15e-svf1521f6ew.html#_accessoriesproducts

[19] *Teclado Measy RC12. Julio/2014.*

- http://www.wayteq.eu/files/download/measy-rc12/measy-rc12-air-mouse-manual-en.pdf

[20] *Mando IPEGA PG-9025. Julio/2014.*

- http://www.ebay.com/itm/Wireless-Bluetooth-Game-Controller-Gamepad-iPega-PG-9025-for-Android-iOS-iPhone-/291190715759?pt=Other_Tablet_eReader_Accessories&hash=item43cc51b56f

10.5.- SOFTWARE EMPLEADO

[21] *Eclipse. Enero-Febrero/2014.*

- https://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/lunar
- https://www.eclipse.org/downloads/

[22] *SDK Android (Kit de desarrollo). Enero-Febrero /2014.*

- http://developer.android.com/intl/es/sdk/index.html

[23] *ADT plugin para Eclipse. Enero-Febrero /2014.*

- http://developer.android.com/tools/sdk/eclipse-adt.html

[24] *JDK Java. Versión 7(Java SE Developmen kit 7). Enero-Febrero /2014.*

- http://www.oracle.com/technetwork/java/javase/downloads/jdk-7-netbeans-download-432126.html?ssSourceSiteId=otnes

[25] *Programa RobotStudio. Marzo/2014.*

- http://new.abb.com/products/robotics/robotstudio/downloads

[26] *Droid explorer. Conecta Tablet a PC para capturar las pantallas. Julio/2014.*

- http://de.codeplex.com/releases/view/111225

[27] *Programa diagrama de flujos: Julio/2014.*

- http://www.nchsoftware.com/chart/