Introducción

En esta propuesta se presenta una metodología del control de arduino a través de una interfaz de consola en java, permite crear usuarios y administrarlos, también permite girar un motor de izquierda a derecha.

Materiales

Se utilizó el siguiente material

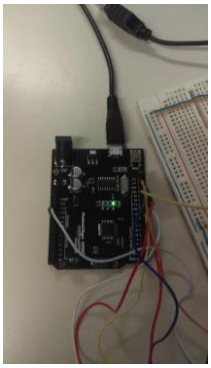
Una placa arduino uno

Un cable USB A-B

Un protoboard

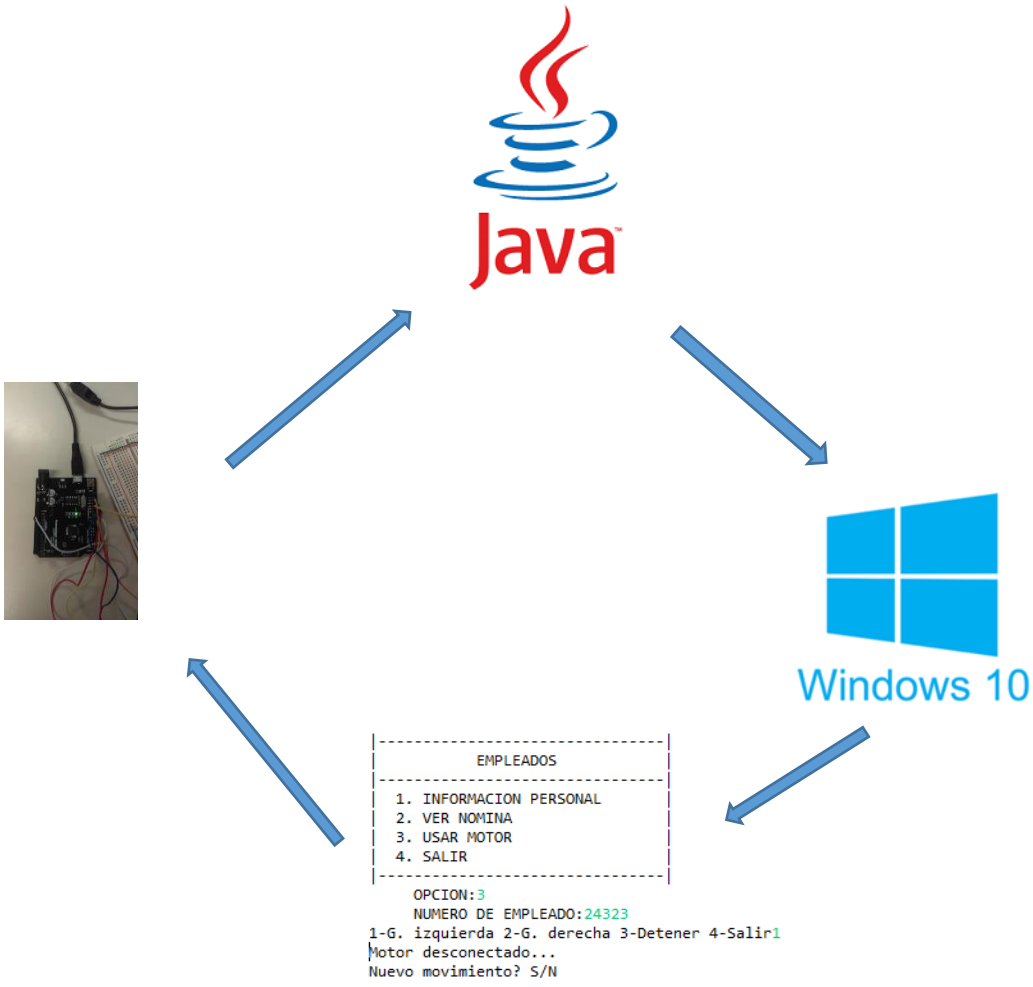
Leds

Resistencias



Conexión de arduino al protoboard

Metodología

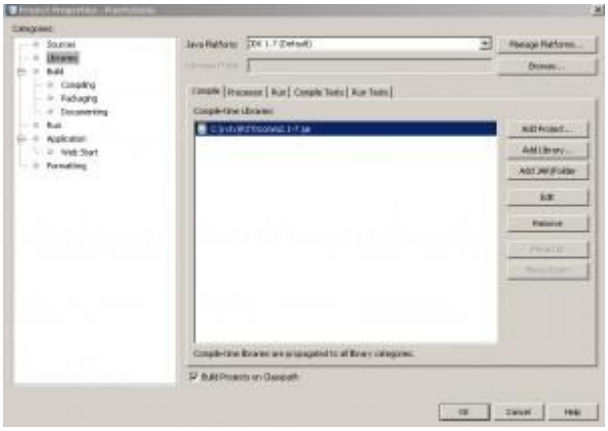


Metodologías y tecnologías usadas, para interactuar con el panel de la aplicación, simula el movimiento de un motor al girar de izquierda a derecha (encendiendo los leds).

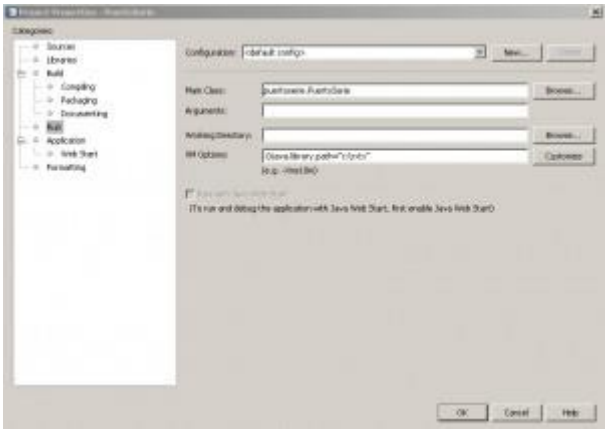
Algoritmo

Paso 1: Configurar la librería RXTX en el entorno de desarrollo.

En las propiedades del proyecto, añadimos la librería:



Añadimos RXTXcomm2.1-7.jar al proyecto
A continuación indicamos donde está la librería nativa. Si la ruta contiene espacios es necesario ponerla entre comillas:



Ruta a la librería nativa rxtx
En Eclipse

En eclipse podemos hacer los dos pasos en la misma pantalla:

Librería RXTX y librería nativa en Eclipse

Paso 2: cargar la librería nativa

Lo primero que hay que hacer es asegurarnos que podemos acceder a la librería nativa. En el caso de que no se cargue correctamente, podemos analizar la excepción producida para diagnosticar el problema. Lo más normal, es que la libreria no haya sido copiada al directorio adecuado, o no le estemos indicando correctamente el path con el parametro "*-Djava.library.path".*

```
try{

System.loadLibrary("rxtxSerial");
System.out.println("Se ha cargado la librería nativa correctamente");

} catch (UnsatisfiedLinkError u) {
System.err.println("No se ha encontrado la librería nativa de puerto serie");
}
```

Paso 3: enumerar los puertos disponibles en el sistema.

Con el siguiente código, enumeramos los puertos serie y paralelo que hay disponibles en el sistema. Si queremos centrarnos solo en los puertos serie, tenemos que comprobar el tipo de cada puerto comparandolo con la constante CommPortIdentifier.PORT_SERIAL.

```
Enumeration listaPuertos = CommPortIdentifier.getPortIdentifiers();
 CommPortIdentifier idPuerto = null;
 boolean encontrado = false;
 while (listaPuertos.hasMoreElements() && !encontrado) {
 idPuerto = (CommPortIdentifier) listaPuertos.nextElement();
 if (idPuerto.getPortType() == CommPortIdentifier.PORT_SERIAL) {
 if (idPuerto.getName().equals("COM1")) {
 encontrado = true;
 }
 }
 }
```

Paso 4: abrir el puerto.

```
SerialPort puertoSerie =null;

try {
puertoSerie = (SerialPort)idPuerto.open( "DescripcionPropietario",2000
);
 } catch( PortInUseException e ) {
System.out.println("Error abriendo el puerto serie");
}



try {
 puertoSerie.setSerialPortParams(9600,
 SerialPort.DATABITS_8,
 SerialPort.STOPBITS_1,
 SerialPort.PARITY_EVEN );
```

```
 puertoSerie.notifyOnDataAvailable(true);
 } catch( UnsupportedCommOperationException e ) {
System.out.println("Error configurando parámetros de configuración");
 }



InputStream entrada = puertoSerie.getInputStream();

String s="";
 while (entrada.available() > 0) {

int dato = entrada.read();
s=s+(char)dato;

}



class SerialListener implements SerialPortEventListener{

@Override
 public void serialEvent(SerialPortEvent arg0) {

 if (arg0.getEventType() == SerialPortEvent.DATA_AVAILABLE) {

//Tenemos nuevos datos disponibles en el puerto, podemos poner aquí
el código que recibe los datos y los procesa.
(...)
}

 }
```

El listener se añade al puerto serie tal que así:

```
SerialListener miListener=new SerialListener();
```

```
puertoSerie.addEventListener(miListener);
```

No podemos olvidarnos de cerrar todos los canales y el puerto una vez finalizada nuestra tarea:

```
puertoSerie.removeEventListener();//En el caso de que lo tenga.
puertoSerie.close();
```

Implementación

Consola de java en Windows 10 para uso del puerto serial hacia arduino

No se pudo probar la aplicación completa por falta de tiempo, se presentaron varios problemas con la compilación java y con la placa arduino, no se logró completamente la comunicación.

Nos percatamos que nunca se puedo abrir el puerto serial en java.

Resultados

El Desarrollo de una aplicación que tiene control de usuarios, altas, bajas, y modificación.

La aplicación es capaz de almacenar los movimientos registrados en el sistema, y controla los movimientos de un motor, en este caso simulado por una placa de arduino.

Referencias

http://monillo007.blogspot.com/2009/04/leer-la-entrada-de-un-puerto-serial.html

http://www.hell-desk.com/acceder-al-puerto-serie-programando-en-java/

http://www.hell-desk.com/acceder-al-puerto-serie-programando-en-java-2a-parte/