VOLTMETRO BLUETOOTH Y DESPLIEGUE EN SMARTPHONE

Fernando Reyes Avilés
Universidad Autónoma Metropolitana
fera@azc.uam.mx

Carlos Avilés Cruz
Universidad Autónoma Metropolitana
caviles@azc.uam.mx

Resumen

La instrumentación electrónica ha tenido avances significativos tanto en precisión, tamaño, costos, comunicación inalámbrica y movilidad, principalmente. En el presente trabajo se analizó, desarrolló y se construyó un voltmetro basado en tarjeta de desarrollo arduino-UNO, con capacidad de medir hasta 4 voltajes en el rango de 0 a 5 V de cd, con una precisión de 5 mV y un tiempo de lectura-despliegue de 1000 ms. El despliegue de los voltajes leidos se lleva a cabo en un teléfono inteligente corriendo en sistema operativo android. La conexión entre el voltmetro y el teléfono inteligente se hace a través de Bluetooth. La aplicación desarrollada para el teléfono permite desplegar hasta 4 valores simultáneamente (4 canales). El prototipo desarrollado mide y despliega los voltajes en tiempo real y es de bajo costo.

Palabras Claves: Aplicaciones móviles, instrumentación, smartphone, voltímetro.

Abstract

Electronics instrumentation has made important progress in precision, size, cost, wireless communication and mobility. In this work, we have analyzed, developed and built a voltmeter based on Arduino UNO development card. The prototype has the following features: measures up to 4 nodal voltages, 5 mV precision and one second sampling for recording-displaying voltages. Measured voltages are transmitted to a smartphone via Bluetooth connection and displayed on it.

Smartphone application (running on Android operating system) allows displaying 4 voltage values at the same time (4 channels). Our proposal works on real time and it is a low-cost device.

Keywords**:** *Instrumentation, mobile app, smartphone, voltmeter.*

1. Introducción

Los voltímetros tradicionales, ya sean analógicos o digitales, sólo permiten medir una diferencia de potencial (voltaje) a la vez, lo cual vuelve cansado y tardado las mediciones de los voltajes nodales de un circuito eléctrico relativamente grande. Por otro lado, los dispositivos móviles se han vuelto muy populares en los últimos años gracias a que permiten realizar una gran cantidad de tareas con simplicidad. Hoy en día, todo el mundo tiene un teléfono inteligente ”smartphone“ o tableta ”tablet”, por esta razón, se desarrolló una aplicación móvil que se comunica, a través de bluetooth, con una tarjeta de desarrollo Arduino UNO, con el fin de medir varios voltajes nodales, de un circuito eléctrico, a la vez. La facilidad de manipular la información, desde el punto de vista del usuario, que aparece en la pantalla de un dispositivo móvil, y la libertad para presentar información en ella, desde el punto de vista del desarrollador, hacen que los dispositivos móviles puedan ser usados como herramientas de trabajo eficaces.

Gracias a la capacidad de la tarjeta Arduino UNO, de poder medir varios voltajes de manera simultánea, se pensó en utilizarlo de manera conjunta con un dispositivo móvil para agilizar el proceso de medición, y que de esta forma, no se invierta demasiado tiempo en este proceso.

En la figura 1 se presenta un diagrama general de la secuencia de pasos que se efectúan con el fin de tomar las mediciones de los voltajes nodales.

El proceso mediante el cual se miden los voltajes nodales, se recibe en el teléfono y se muestran en la pantalla del dispositivo móvil es el siguiente:

- El teléfono inteligente envía una solicitud de medición de voltajes a la tarjeta Arduino UNO mediante la aplicación Voltmetro Bluetooth.
- La tarjeta Arduino UNO recibe la solicitud, a través del módulo bluetooth, y mide los voltajes nodales del circuito eléctrico.

- El dispositivo móvil recibe los datos enviados por el Arduino UNO y los coloca en pantalla.

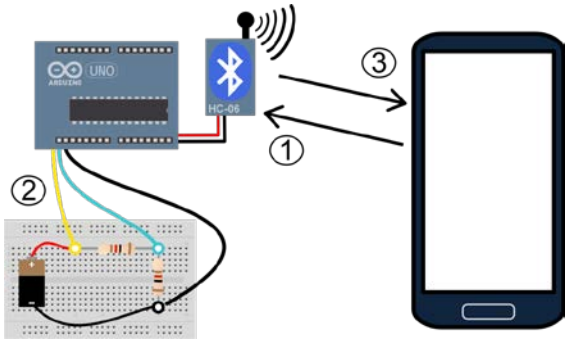
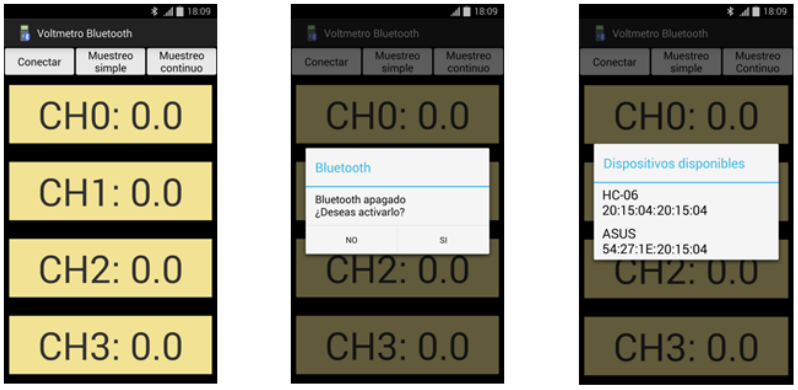


Figura 1 Esquema general del voltmetro bluetooth.

En la figura 2a se muestra una captura de pantalla de la aplicación Voltmetro Bluetooth, la cual contiene tres botones: “Conectar”, “Muestreo Simple” y “Muestreo Continuo”, y 4 cajas de texto. El botón Conectar, se encarga de buscar el módulo bluetooth del Arduino, para posteriormente conectarse con él para poder intercambiar información. En la figura 2b se muestra la ventana emergente que aparece cuando el adaptador Bluetooth del dispositivo móvil está apagado. En la figura 2c se observa la ventana emergente que aparece cuando la aplicación comienza la búsqueda de los dispositivos bluetooth disponibles. En ella se listan todos aquellos dispositivos que estén al alcance del dispositivo móvil.



(a) Pantalla principal de la aplicación

(b) Ventana emergente para el encendido del Bluetooth

(c) Ventana emergente de búsqueda de dispositivos

Figura 2 Pantalla de la aplicación en el teléfono inteligente.

La aplicación tiene dos modos de operación: “Muestreo Simple” y “Continuo”. En modo de “muestreo simple”, solamente se toma una medición de los voltajes cada vez que se oprime el botón “Muestro Simple”. En cambio, cuando el botón “Muestreo Continuo” es oprimido, se toma una medición de los voltajes cada 1000 ms, es decir, los voltajes desplegados en las 4 cajas de texto amarillas se actualizan una vez por segundo.

Se decidió, de forma arbitraria, que solamente se mostraran 4 mediciones simultáneas, pero en realidad no existe una limitante en cuanto al número de mediciones multiplexadas que se pueden tomar. Para aumentar dicha cantidad, solamente sería necesario modificar el circuito de medición, es decir extenderlo, y modificar la forma en la que se presentan los datos, es decir, la interfaz gráfica. La estructura general de la aplicación, la cual se describe a detalle en las siguientes secciones.

Estado del Arte

Los dispositivos móviles se han vuelto muy populares en los últimos años gracias a que permiten realizar una gran cantidad de tareas con simplicidad. La facilidad de manipular la información que aparece en la pantalla de un dispositivo móvil hace que los dispositivos móviles se puedan usar como herramientas de trabajo eficaces.

Con respecto al estado del arte de éste tipo de dispositivos, comercialmente se venden voltímetros con aplicaciones muy específicas, tal es el caso del voltímetro para medir el voltaje en una batería de automóvil [WirelessMultimeter, 2016] el cual se conecta vía bluetooth a un dispositivo teléfono inteligente (smartphone) con la limitante que solo se puede desplegar una medida de voltaje (un solo canal), con alto costo. Por otro lado, se tiene a la venta un multímetro de la reconocida marca Fluke [Fluke, 2013] para realizar medidas inalámbricas, en este caso se tienen que comprar dos dispositivos, uno para realizar la medición y otro para el despliegue; en este caso el despliegue se lleve a cabo en un dispositivo desarrollado por la empresa Fluke. El costo de éste multímetro es alto, con la limitante que no se puede desplegar la información que en el mismo dispositivo.

Se tiene un trabajo de tipo académico [Malik, 2010] en el que se desarrolló un voltímetro digital inalámbrico (Wireless digital voltmeter), la principal desventaja de éste desarrollo radica en la utilización de dispositivos de gran volumen para la conexión inalámbrica, provocando mayor consumo de corriente y voltaje, usa un batería de gran tamaño y realmente no es muy portátil; si bien se puede desplegar la información en un dispositivo móvil, el dispositivo que hace las mediciones es de gran volumen y peso.

[Whong, 2017] desarrolló un multímetro de dos canales que permite medir voltaje, corriente y resistencia. Este multímetro envía las lecturas tomadas vía bluetooth a un dispositivo móvil iOS o Android. Otro ejemplo de un multímetro que utiliza un dispositivo móvil para mostrar la interfaz de usuario es el desarrollado por [Wang, 2013]. Este multímetro se comunica vía USB con el dispositivo móvil. Permite medir voltaje, corriente, resistencia, continuidad, capacitancia, frecuencia y temperatura.

Bluetooth Multimeter es un multímetro de un canal desarrollado por [Seedstudio, 2013]. Este multímetro se comunica inalámbricamente con un dispositivo móvil para mostrar las lecturas obtenidas. Permite medir voltaje, corriente y resistencia.

2. Métodos

El voltímetro se construyó con una tarjeta de desarrollo Arduino UNO, un circuito de referencia y un módulo Bluetooth, figura 3. La tarjeta Arduino UNO contiene un convertidor Analógico Digital (ADC) de 10 bits. El rango de conversión del ADC es de 0 a 5 V. Este convertidor puede manejar 6 entradas analógicas (de manera multiplexada) conocidas como canales analógicos, identificados en la tarjeta con las etiquetas A0, A1, A2, A3, A4 y A5.

Para transmitir los datos por medio del módulo bluetooth, se utilizó un módulo con matrícula HC-06. El Arduino UNO se conecta al módulo bluetooth por medio de las entradas TX y RX. En [Mok, 2011] se describe a detalle la forma de conectar el Arduino y módulo bluetooth. Ya que la transmisión hacia el módulo bluetooth es serial, no se necesita utilizar ninguna biblioteca ni sentencia especial para enviar los datos al dispositivo móvil

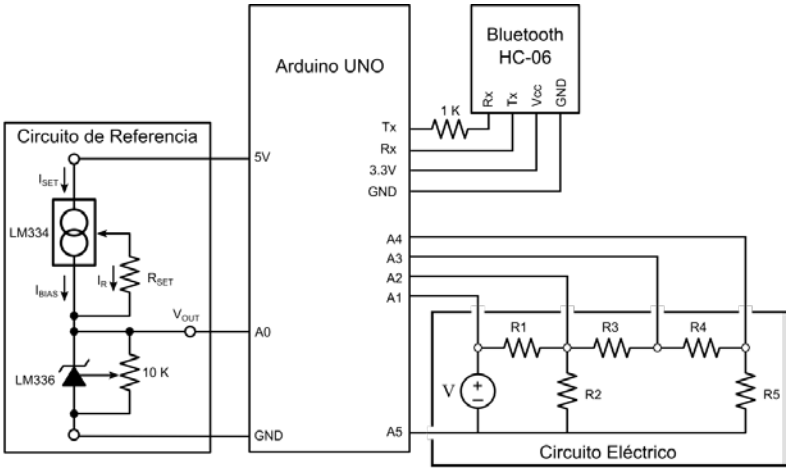


Figura 3 Circuito electrónico del instrumento de medición basado en Arduino-UNO.

La forma de medir voltajes por medio del Arduino es muy fácil. Sólo se debe conectar alguno de los 4 canales disponibles a algún nodo del circuito eléctrico que se está examinando. Una vez que se han conectado los canales del Arduino a los puntos en los que se desea medir voltaje, se debe presionar, en el dispositivo móvil, alguno de los botones de muestreo. Cuando las mediciones están listas y se envíen, se recibirá una cadena de la siguiente forma: ch1, ch2, ch3, ch4.

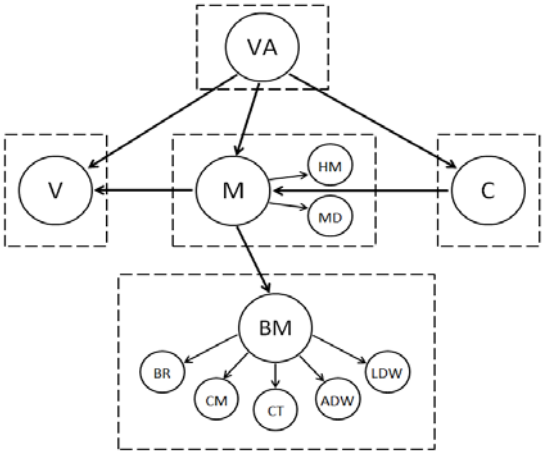
El ADC de la tarjeta Arduino UNO tiene 10 bits de resolución, es decir, los voltajes analógicos que mide los convierte en datos digitales en el intervalo [0, 1023]. Una vez que se mide un voltaje en el circuito eléctrico, se ajusta de acuerdo a la lectura que se obtiene del circuito de referencia, ya que el circuito de referencia genera un voltaje constante de 2.5 V.

Por ejemplo, si en un nodo del circuito se obtiene una lectura digital de 226, y de la salida del circuito de referencia se obtiene una lectura digital de 512, el voltaje medido en el nodo del circuito se ajusta de la siguiente forma:

$$voltaje\ nodal = \frac{226}{512} x2.5\ V = 1.104\ V$$

Desarrollo de la Aplicación en el Teléfono Inteligente

La aplicación se desarrolló para el sistema operativo Android, siguiendo el patrón de programación Modelo-Vista-Control. Para construirla, se necesitó un conjunto de clases, los cuales se muestran en la figura 4.



- VA: VoltmetroActivity
- M: Modelo
- V: Vista
- C: Control
- HM: HiloDeMuestreoContinuo
- MD: ManejadorDatos
- BM: BluetoothManager
- BR: BluetoothReceiver
- CM: CommunicationThread
- CT: ConnectThread
- ADW: AlertDialogWindow
- LDW: ListDialogWindow

Figura 4 Diagrama a bloques de las clases usadas en la aplicación bajo el paradigma *Modelo-Vista-Control*.

VoltmetroActivity es la única *Activity* de la aplicación. El objeto, de tipo VoltmetroActivity, se encarga de crear a los tres objetos principales de la aplicación (los que manejan la aplicación) Modelo, Vista y Control. La aplicación sólo tiene una pantalla, es por eso que sólo se utilizó una *Activity*.

Modelo es un conjunto de clases, conformado por *Modelo, HiloDeMuestreoContinuo* y *ManejadorDatos*, que contiene toda la lógica necesaria para administrar el estado de la aplicación. Los objetos de tipo *VoltmetroActivity*, *Control* y *BluetoothManager*, le informan al objeto de tipo *Modelo* sobre los cambios que ocurren en la aplicación y éste toma las decisiones de lo que ocurrirá a continuación, basándose en la lógica que está contenida en él.

Control es la clase que contiene el registro de los métodos que serán ejecutados cuando algún elemento de la interfaz gráfica de la aplicación cambie (e.g. un botón ha sido oprimido). El objeto de tipo *Control* que es creado por el objeto *VoltmetroActivity*, le avisa al objeto de tipo *Modelo* de cualquier cambio en la interfaz gráfica.

Vista contiene todo lo que hace falta para configurar la interfaz gráfica de la aplicación. El objeto de tipo *Vista* se comunica con el objeto de tipo *Control* para

indicarle cuales son los objetos interactivos (i.e. botones) que pueden sufrir cambios debido a la interacción del usuario con la aplicación, es decir, los eventos a los cuales debe de estar atento.

BluetoothManager y todas las clases que tienen comunicación directa con él (todas las clases que se encuentran en el mismo rectángulo que *BluetoothManager*), contienen toda la lógica necesaria para establecer un canal de comunicación con un dispositivo bluetooth, enviarle información, recibir información de él, y cerrar el canal de comunicación una vez que sea necesario.

Descripción de la clase VolmetroActivity (VA)

Es una clase que extiende a la clase *Activity*. El objeto de este tipo que se crea cuando la aplicación se inicia, crea a los objetos de tipo *Modelo, Vista y Control*. También, por ser un objeto *Activity*, contiene los métodos constructores que son ejecutados cuando el ciclo de vida de la *Activity* cambia de un estado a otro. En este caso, los únicos métodos constructores que se sobre-escriben son los métodos *onPause()* y *onDestroy().* Cuando alguno de ellos es ejecutado, el objeto *VoltmetroActivity* avisa al objeto de tipo *Modelo*, que fue creado en el método *onCreate*(), que el estado de la *Activity* ha cambiado.

Descripción de la clase Modelo (M*)*

En *Modelo* se encuentran definidos los métodos que se ejecutan cuando hay un cambio en:

- El estado de la *Activity*
- El estado de la conexión Bluetooth
- El estado de la interfaz gráfica.

Ya que *Modelo* administra el estado de la aplicación, las únicas clases, que no forman parte de *Modelo*, que tienen comunicación directa con él son: *VoltmetroActivity*, *Vista*, *Control* y *BluetoothManager*. Además de la clase *Modelo*, las clases *HiloDeMuestreoContinuo* y *ManejadorDatos* también contienen métodos que realizan tareas importantes para que la aplicación funcione.

Descripción de la Clase HiloDeMuestreoContinuo (HM)

HiloDeMuestreoContinuo es una clase de tipo Thread, cuya tarea es muy sencilla. Cuando el usuario oprime el botón “Muestreo Continuo”, *Modelo* le notifica esta acción, y comienza a enviarle al Arduino una solicitud de medición cada 1000 ms. Una vez que el usuario oprime nuevamente el botón, deja de enviar estas peticiones (figura 5).

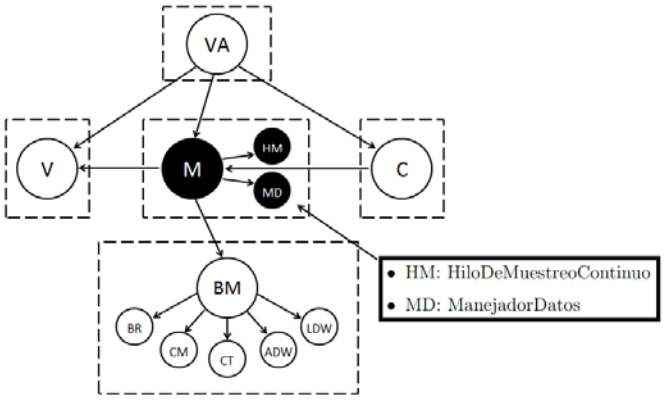


Figura 5 Diagrama de clases y su inter relación: HiloDeMuestreoContinuo y ManejadorDatos.

Descripción de la Clase *ManejadorDatos (MD)*

Una vez que el Arduino ha tomado las mediciones que le fueron solicitadas y las envía al dispositivo móvil, *Modelo* recibe la cadena de caracteres, a través de *BluetoothManager* y la envía al *ManejadorDatos*. El único trabajo que se efectúa aquí, es la interpretación de la información, es decir, convierte la cadena de caracteres recibida desde el Arduino y la convierte en números de punto flotante, los cuales contienen los valores de los voltajes nodales que fueron medidos, y que serán mostrados en pantalla. Una vez que termina la interpretación, le envía los datos a la *Vista* para que los coloque en pantalla (figura 5).

Descripción de la Clase Vista (V)

La clase *Vista* se encarga de ordenar todos los elementos de la interfaz gráfica de la aplicación (figura 6). Para lograr que la aplicación luzca igual en cualquier dispositivo, en esta clase se encuentran declarados una serie de métodos que

ajustan el tamaño de la interfaz gráfica, basándose en el tamaño de la pantalla del dispositivo en el que se ejecuta la aplicación. Esto se logra asignando un tamaño proporcional a la pantalla, a cada elemento que aparece en pantalla, en lugar de asignarles un tamaño fijo en pixeles.

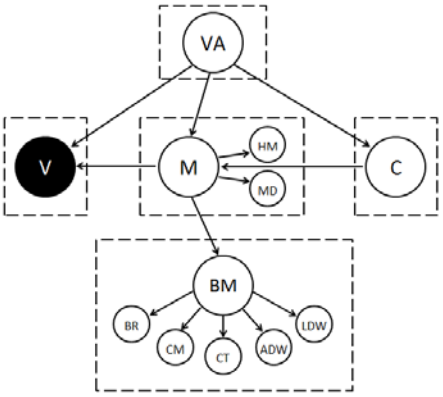


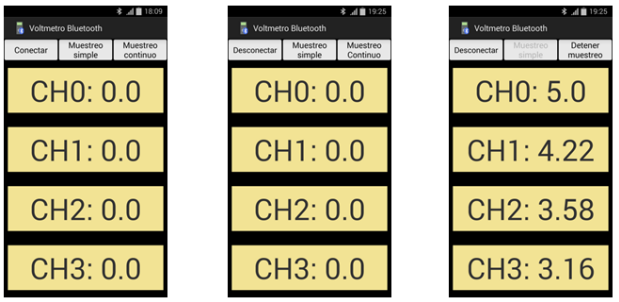
Figura 6 Diagrama de clases que interactuan con la clase *Vista.*

La tarea de colocar en pantalla todos los elementos que aparecen en pantalla, utiliza algunos recursos de tipo Layout (codificados en archivos XML), que no son más que las declaraciones de los botones y las cajas de texto que se muestran en pantalla. A través de la clase *Activity* de la aplicación, obtiene una referencia de estos elementos y los coloca en pantalla. Además, le envía al *Control*, esta referencia, para que reciba notificaciones de las interacciones, del usuario, con estos elementos.

La clase *Vista* también se encarga de actualizar el estado de la interfaz gráfica cuando el usuario oprime algún botón, y cuando las mediciones recibidas desde el Arduino, están listas para ser mostradas en pantalla. Cuando el usuario oprime algún botón, *Control* le notifica de esta acción a *Modelo*, que a su vez le indica a *Vista* que debe actualizar el estado de los botones. El estado inicial de los botones se muestra en la figura 7a, y pueden cambiar en los siguientes casos:

- El texto del botón "*Conectar"* cambia a "*Desconectar"* una vez que se establece una conexión con algún dispositivo bluetooth, figura 7b.
- Una vez que está conectado, y se oprime el botón *"Muestreo Continuo"*, el texto de este botón cambia a *"Detener Muestreo"*, figura 7c.

- Cuando la aplicación entra en el estado de *"muestro continuo"*, el botón *"Muestreo Simple"* se desactiva, es decir, no puede ser oprimido por el usuario, hasta que la aplicación sale del estado de muestreo continuo, figura 7c.



(a) Estado inicial de los tres botones (b) Cambio de texto del botón Conectar (c) Cambio de texto del botón Muestreo Continuo

Figura 7 Cambios en los botones de la aplicación*.*

Por último, la *Vista* también se encarga de colocar en pantalla los 4 valores de los voltajes medidos, que recibe del *ManejadorDatos*.

Descripción de la Clase *Control (C)*

Voltímetro Bluetooth es una aplicación muy sencilla, visualmente hablando, que sólo tiene tres elementos con los que el usuario puede interactuar, en otras palabras, sólo hay tres botones en pantalla. Utiliza las referencias de los botones, que obtuvo de *Vista*, para notificar a *Modelo* cuando el usuario oprime alguno de los botones. Su diagrama de interacción de clases se muestra en la figura 8.

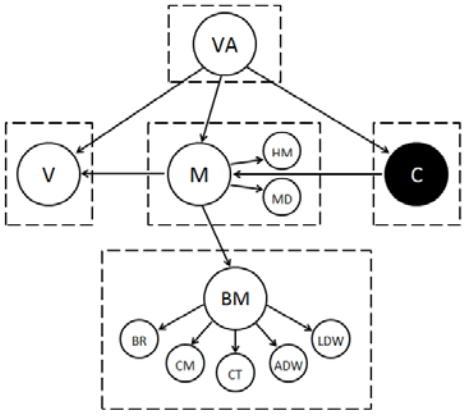


Figura 8 Diagrama de clases que interactuan con la clase *Control.*

Descripción de la Clase *BluetoothManager (BM)*

El grupo de clases que tienen comunicación directa con la clase *BluetoothManager* (BR, CM, CT, ADW, LDW) se encargan de establecer comunicación con un dispositivo bluetooth. En la figura 9 se muestra su interacción con las demás clases.

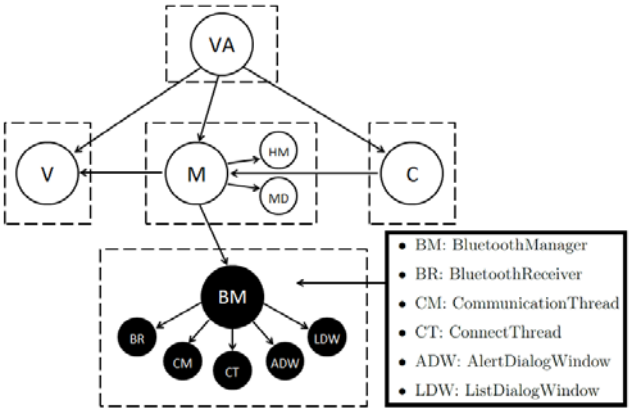


Figura 9 Diagrama de clases que interactúan con la clase *BluetoothManager (BM).*

BluetoothManager es una extensión de *Modelo*, la cual se encarga de administrar el estado de la conexión bluetooth y de notificar a *Modelo* de ciertos eventos relevantes para otros objetos (e.g., *Vista*). La instancia de *BluetoothManager* que *Modelo* crea en su constructor, se encarga de lo siguiente:

- Encender/Apagar el modulo Bluetooth del dispositivo móvil.
- Notificar a *Modelo* cuando no se encontró ningún dispositivo durante la búsqueda. Recibir, desde *Modelo*, los mensajes que se deseen enviar al dispositivo bluetooth.
- Enviarle a *Modelo* los mensajes recibidos del dispositivo bluetooth
- Crear instancias de las clases BluetoothReceiver(BR), Communication Thread(CM), ConnectThread (CT), AlertDialogWindow (ADW) y ListDialogWindow (LDW)

Clase BluetoothReceiver (BR)

BluetoothReceiver es un receptor de notificaciones del sistema operativo (SO).

Esta clase extiende a la clase *BroadcastReceiver*, la cual está diseñada para estar pendiente de las notificaciones que el SO emite cada vez que un suceso importante ocurre. Para atender a estas notificaciones, se deben registrar cuales son los tipos de notificaciones que se desean recibir, ya que el SO emite muchas notificaciones de muchos tipos, así que, cuando se desea poder escuchar un cierto tipo de notificaciones dentro de una aplicación se deben registrar mediante una instancia de la clase *IntentFilter*, la cual permite hacer este tipo de registro.
Las notificaciones a las que se atiende en la aplicación *Voltmetro Bluetooth* son:

- Un dispositivo bluetooth ha sido encontrado.
- El proceso de búsqueda ha comenzado.
- El proceso de búsqueda ha finalizado.
- El dispositivo móvil se ha conectado con éxito al dispositivo bluetooth.
- El dispositivo móvil se ha desconectado del dispositivo bluetooth.

Clase ConnectThread (CT)

ConnectThread es una clase de tipo Thread. El hilo que se crea a través de ésta clase, genera una instancia de la clase *Socket*, con el fin de conectar el dispositivo móvil con el dispositivo bluetooth. La clase *Socket* permite conectar dos dispositivos, además, de conectar dispositivos, permite cerrar el canal de comunicación creado.

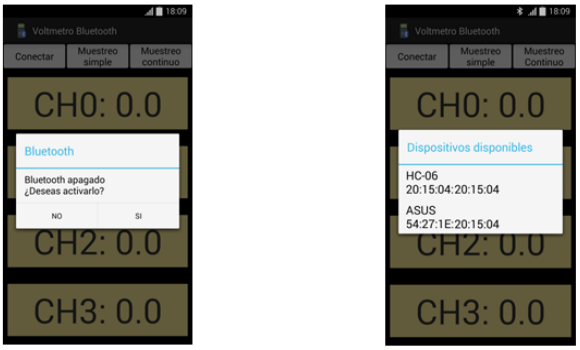
Clase CommunicationThread (CM)

CommunicationThread, al igual que *ConnectThread*, es una clase de tipo Thread. Con el hilo que se crea mediante esta clase, los dos dispositivos que se conectaron a través de la clase *ConnectThread* pueden enviar y recibir mensajes, es decir, se pueden comunicar. En resumen, la única función de las clases *ConnectThread* y *CommunicationThread* es permitir al dispositivo móvil y bluetooth comunicarse (en este caso, el voltmetro).

Clase AlertDialogWindow (ADW)

AlertDialogWindow es una clase que crea una ventana emergente de tipo pop-

up. Cuando la aplicación se abre, la instancia de la clase *BluetoothManager* revisa el estado del adaptador Bluetooth del dispositivo móvil. Si está apagado, se muestra en pantalla la ventana creada por *AlertDialogWindow*. Esta ventana, que se puede observar en la figura 10a, permite al usuario encender, o no, el adaptador Bluetooth. Si decide no encender el adaptador, la aplicación se cierra, de lo contrario, permanece abierta.



(a)Ventana emergente para el encendido del Bluetooth

(b) Ventana emergente de búsqueda de dispositivos

Figura 10 Diagrama de clases que interactuan con la clase *BluetoothManager (BM).*

Clase ListDialogWindow (LDW)

ListDialogWindow también es una clase que crea una ventana emergente de tipo pop-up, figura 10(b). Esta ventana muestra la lista de los dispositivos bluetooth que están al alcance, es decir, los dispositivos bluetooth con los que es posible conectarse. Los elementos de esta lista son seleccionables, en otras palabras, el usuario puede presionar cualquiera de los elementos de la lista, para conectarse con el dispositivo que es listado en la casilla que el usuario presiona.

Hasta aquí se tiene desarrollado ya en su totalidad tanto el hardware como el software de todo el sistema, es decir se cuenta ya con el voltímetro basado en la tarjeta de desarrollo Arduino-uno, así como la aplicación que corre en el teléfono inteligente. La siguiente fase es evaluar nuestro prototipo propuesto.

3. Resultados

A continuación, se presentan los resultados obtenidos tanto en voltímetro bluetooth, como del despliegue en el teléfono inteligente. El muestreo del voltaje

es cada segundo y la resolución es de 5 mV. En la figura 11 se presenta tal como quedó el medidor de voltajes basado en la tarjeta de desarrollo arduino UNO, en dicha figura, el elemento vertical más grande corresponde al módulo de conexión bluetooth. Las especificaciones técnicas se describen en la tabla 1.

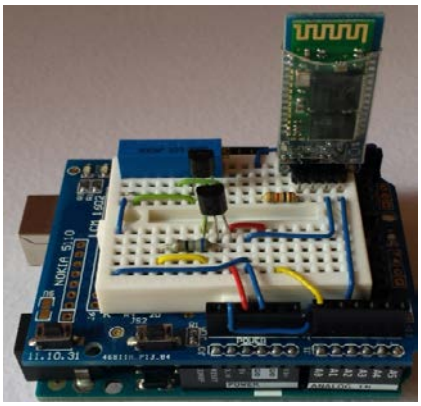


Figura 11 Parte superior del voltímetro, circuito electrico montado sobre una tarjeta de desarrollo.

La validación de las lecturas tomadas con el voltímetro construido se hizo mediante la comparación de las lecturas obtenidas con dicho instrumento y, las lecturas obtenidas con un multímetro digital Fluke 179 True RMS. Se generaron 25 voltajes diferentes con una fuente de alimentación de c.d. GW INSTEK GPS-4303.

Tabla 1 Especificaciones técnicas del dispositivo de medición.

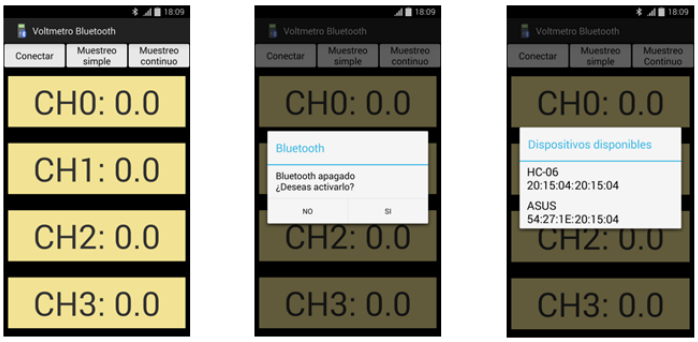
Especificaciones Técnicas	
Microcontrolador	Arduino UNO
Canales analógicos	6
Rango de medición	De 0 a 5 V
Resolución	4.89 mV (10 bits)
Bluetooth	HC-06
Estándar inalámbrico	Bluetooth 2.0
Alcance	< 20 metros
Banda de operación	2.4 GHz
Circuito de referencia	
Voltaje de referencia	2.5 V
Fuente de corriente ajustable	LM334
Diodo de referencia	LM336

El error absoluto promedio entre las mediciones tomadas con la tarjeta Arduino y las mediciones tomadas con el multímetro digital Fluke fue menor al 2%.

Con respecto a la aplicación para el teléfono inteligente móvil, ésta fue desarrollada en lenguaje de programación JAVA y compilada en eclipse. La aplicación puede correr en cualquier dispositivo móvil (teléfono o tablet) bajo el sistema operativo Android 6.0.

La figura 12a muestra la pantalla inicial con sus tres botones de operación:

- Conectar.
- Muestreo simple.
- Muestreo continúo.



(a) Pantalla principal de la aplicación

(b) Ventana emergente para el encendido del Bluetooth

(c) Ventana emergente de búsqueda de dispositivos

Figura 12 Configuración de la conexión Bluetooth entre el voltmetro y el teléfono inteligente.

Para configurar la conexión Bluetooth entre el voltmetro y el teléfono inteligente es necesario oprimir el botón "*conectar*", si el dispositivo bluetooth del teléfono está apagado, se preguntará por el encenderlo, figura 12b. Posteriormente, se buscará los dispositivos bluetooth activados más próximos y se mostrarán en pantalla para poder seleccionar uno de ellos. Para enlazar al medidor voltmetro, se debe seleccionar "HC-06" con dirección mac 20:15:04:20:15:04, figura 12c. Ahora si, ya se está en posibilidad de recibir y mostrar las lecturas que haga la tarjeta Arduino UNO.

También se podría usar para crear otros instrumentos de medición, como amperímetros e incluso osciloscopios, ya que la aplicación descrita aquí, contiene

todo lo necesario para hacerlo, es decir, se puede utilizar como la base de cualquier tipo de aparato de medición de variables eléctricas. Para hacer esto lo único que se necesitaría sería aumentar funcionalidades, no sería necesario modificar nada, gracias a la forma en la que fue construida.

4. Discusión

Una serie de mediciones fueron llevadas a cabo para validar nuestro prototipo propuesto, solo como ejemplo se presenta un circuito resistivo, como el mostrado en la figura 13. En dicho circuito se mide el voltaje de la alimentación y los voltajes nodales de las resistencias. En la figura 14a se muestra cuando la aplicación inicia en el teléfono inteligente. Una vez conectado el voltmetro con el celular, se está en posibilidad de tomar las medidas. La figura 14b muestra el momento en el que se pude ya realizar medidas, ya sea una solo o bien, en modo continuo. En la figura 14c se muestran las mediciones de voltaje del circuito de la figura 13.

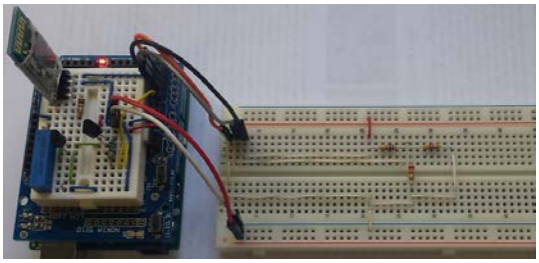
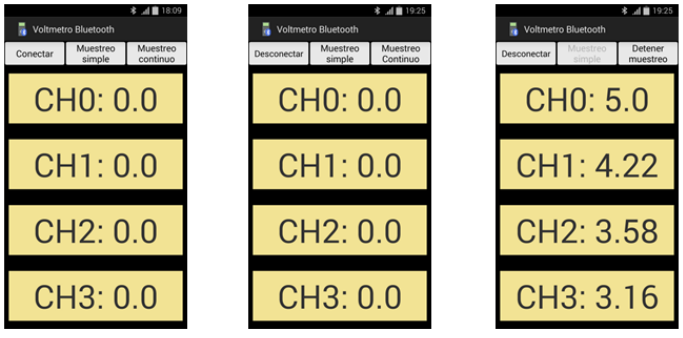


Figura 13 Voltímetro y circuito resistivo de ejemplo.



(a) Estado inicial de los tres botones

(b) Cambio de texto del botón Conectar

(c) Cambio de texto del botón Muestreo Continuo

Figura 14 Despliegue de valores de voltaje en el teléfono inteligente.

En los valores de las medias reales de un circuito resistivo, se debe tomer en cuenta tanto la tolerancia de las resistencias, como la precisión de nuestro instrumento de medición (en nuestro caso es de 5 milivolts).

5. Conclusiones

Hemos logrado la implementación real del diseño y construcción de un voltímetro bluetooth de bajo costo, trabajando en tiempo real. La tarjeta base del desarrollo fue Arduino-UNO y un protoboard de uso genérico. El voltímetro permite medir hasta 4 nodos (canales) de forma simultánea con una resolución de 5 mV en tiempo real. Por otro lado, en cuanto al despliegue, éste puede ser en cualquier teléfono inteligente (smartphone) lo cual abre la posibilidad a que cualquier profesor o estudiante pueda usar éste prototipo y en general éste tipo de tecnología emergente. Las ventajas que presenta el prototipo desarrollado son:

- Bajo costo (alrededor de 400 pesos).
- Sin necesidad de usar cables en las mediciones
- Portátil (cuenta con su propia batería de 9 volts)
- Facilidad de instalar la aplicación en cualquier Smartphone corriendo en sistema operativo Android.
- Facilidad en la conexión bluetooth entre el prototipo y el teléfono inteligente.

Como trabajo futuro se plantea el poder medir corrientes y resistencias de forma multiplexada y poder desplegarlos en una aplicación sobre teléfonos inteligentes (smartphones), lo cual lo convertiría en un multímetro.

6. Bibliografía y Referencias

[1] Arduino–AnalogRead, Arduino.cc, 2017: https://www.arduino.cc/en/Reference/AnalogRead.

[2] Developer.android.com, Bluetooth | android developers, 2016: http://developer.android.com/guide/topics/connectivity/bluetooth.ht.

[3] Java Platform SE 7, Docs.oracle.com, 2017: https://docs.oracle.com/javase/7/docs/api/.

[4] Eclipse.org, Eclipse, 2017. [Online]: https://eclipse.org/.

[5] Fluke, Wireless Multimeter Fluke CNX 3000 for Safer Measurements, fluke.com, 2013: http://en-us.fluke.com/products/digital-multimeters/fluke-cnx-3000-wireless-tester.html..

[6] Malik M. A., Wireless Digital Voltmeter, Workcestor Polytechnic Institute, MA-USA, tesis 2010: https://web.wpi.edu/Pubs/E-project/Available/E-project-050210-221402/unrestricted/FinalReport.pdf

[7] Mok, S., Huang, E. and Xin, L., BLUETOOTH-SERIAL-HC-06. Olimex, 2011: https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf.

[8] eedstudio, Bluetooth Multimeter, Wiki.seeed.cc, 2013: http://wiki.seeed.cc /Bluetooth_Multimeter/.

[9] Wang T., Voltset - The Earth's Smartest Multimeter, Make: DIY Projects and Ideas for Makers, 2013: http://makezine.com/2013/09/21/voltset-the-earths-smartest-multimeter/.

[10] Whong J. and VanWyk E., Mooshimeter, Dragoninnovation, 2017: https://www.dragoninnovation.com /projects/34-mooshimeter.

[11] WirelessMultimeter, Wireless Multimeter Bluetooth, WIRELESSMULTIMETER, 2016: http://www.wirelessmultimeter.com/ ProductDetails.asp?ProductCode=WMM-01.