

# Restricción del Uso de Teléfonos Celulares en Ambientes Controlados

Maria Aurora Molina Vilchis, Ramón Silva Ortigoza,  
Yasania Joselín Escalona Bautista y Héctor Oscar Ramos García

**Resumen**—Es común que se provoquen interrupciones o interferencias por el uso indiscriminado de teléfonos celulares en eventos académicos, culturales o sociales, de ahí que surja la necesidad de evitar o disminuir la recepción o transmisión de llamadas. Otras restricciones pudieran estar relacionadas con el uso de las cámaras fotográficas que incorporan estos dispositivos, la transmisión de mensajes o grabaciones de videos sin autorización. En este artículo se presenta una aplicación basada en Bluetooth para el control del uso de estos dispositivos en ambientes con restricciones.

**Palabras clave**—Bluetooth, restricción del uso de teléfonos celulares, protocolo de comunicación.

## Restriction of the Usage of Mobile Phones in Controlled Environments

**Abstract**—It often happens that interruptions or interferences occur due to indiscriminate usage of the mobile phones during academic, cultural or social events. Thus, there is a necessity for avoiding or diminishing transmissions of phone calls. Another important restriction is related with the unauthorized usage of cameras integrated in these devices, transmission of messages or video capture without permission. In this paper, we present a Bluetooth based application for the restriction of usage of the mobile phones in specially controlled environments.

**Index Terms**—Bluetooth, usage restrictions for mobile phones, communication protocol.

### I. INTRODUCTION

LA importancia que han cobrado las Redes de Área Personal, se basa en la popularidad que han alcanzado los dispositivos móviles tales como los teléfonos celulares, PDA y computadoras portátiles, ya que permiten la comunicación eficiente en cualquier momento y lugar en un entorno personal, de tal manera que se perciben estos dispositivos como partes de un sistema de comunicación integral a lo que se le ha dado en llamar computación pervasiva. Este concepto novedoso se le atribuye a Mark Weiser quien lo anticipó en

sus escritos en 1988, cuando trabajaba para Xerox en el laboratorio de Palo Alto (PARC) en EUA.

La evolución que han tenido los teléfonos celulares ha logrado que este concepto sea ahora una realidad, al incorporar funcionalidades que no hace mucho parecían futuristas como la transmisión y recepción de archivos multimedia, ejecución de juegos, reproducción de archivos MP3, correo electrónico, Web, envío de mensajes de texto y fotografías, recepción de radio y televisión digitales, entre otros. Lo anterior los ha hecho ser los dispositivos móviles más populares por excelencia, sólo en México existen casi 65 millones de teléfonos celulares.

No obstante, el abuso indiscriminado de estos dispositivos ha creado serios problemas, sobre todo la recepción o transmisión de llamadas telefónicas en los ambientes con restricciones de comunicación, pues generan interrupciones no deseadas sobre todo en eventos culturales, ruedas de prensa, ámbitos académicos, etc. donde existen prohibiciones de uso.

En este artículo se presenta el desarrollo de una aplicación de una red Bluetooth para el control de comunicaciones telefónicas celulares en ambientes controlados. La sección II trata de los aspectos técnicos de la red Bluetooth. En la sección III se presenta el diseño de la aplicación. En la sección IV se presenta su desarrollo y los resultados de las pruebas de operación.

### II. TRABAJO PREVIO

La tecnología Bluetooth se ha incorporado de manera natural a estos dispositivos facilitando su interacción y comunicación, y se erige como el estándar de facto de las redes de área personal. Sin embargo, el origen de esta tecnología está relacionado estrechamente con las investigaciones en el campo de las comunicaciones inalámbricas [1]. Así, desde su origen en la década de los 70 hasta la actualidad el interés por estas redes se ha visto progresivamente incrementado convirtiéndola en la tecnología más popular de los últimos años. Los recientes avances en la materia se han centrado en las redes *ad hoc*. Este término hace referencia a redes flexibles inalámbricas sin infraestructura, donde los dispositivos se conectan de forma autónoma.

Estas redes deben poder adaptarse dinámicamente ante cambios continuos, como la posición de los dispositivos, la potencia de la señal, el tráfico de la red y la distribución de la carga. Su principal reto estriba en los continuos e impredecibles cambios de topología.

Manuscrito recibido el 18 de febrero del 2008. Manuscrito aceptado para su publicación el 20 de agosto del 2009.

M. A. Molina Vilchis es del Área de Telemática del Departamento de Posgrado, CIDETEC, Instituto Politécnico Nacional, México (mamolinav@ipn.mx).

R. Silva Ortigoza es del Área de Mecatrónica del Departamento de Posgrado, CIDETEC, Instituto Politécnico Nacional, México (rsilva@ipn.mx).

Y. J. Escalona Bautista y H. O. Ramos García son del Departamento de Comunicaciones y Electrónica, ESIME-Z, Instituto Politécnico Nacional, México (jossy\_259\_1@hotmail.com, kzper16@hotmail.com).

Recientemente la tecnología Bluetooth se ha mostrado como una plataforma de soporte prometedora para estas redes, su principal ventaja es su habilidad para localizar de forma transparente dispositivos móviles cercanos, así como los servicios que ofrecen.

Bluetooth ha es el estándar impulsado por Ericsson, IBM, Intel, Nokia y Toshiba. Las que formaron el Grupo de Interés Especial (*Special Interest Group*, SIG) Bluetooth en Mayo de 1998 [2]. Definida dentro del estándar IEEE 802.15 que se refiere a las redes inalámbricas tipo personal. La conexión inalámbrica Bluetooth opera en el rango de radiofrecuencia de los 2,4 GHz (2,400 a 2,485 GHz) [3] que no requiere licencia de uso en ningún lugar del mundo. Con una banda de guarda de 2 MHz a 3,5 MHz para cumplir con las regulaciones internacionales. Transmite en espectro disperso, con salto de frecuencia, en dúplex y hasta 1,600 saltos/s. La señal salta entre 79 frecuencias en intervalos de 1 MHz para tener un alto grado de tolerancia a las interferencias y obtener comunicaciones robustas. Además se dispone de comunicaciones punto a punto y multipunto, donde un dispositivo puede establecer de forma simultánea hasta siete canales de comunicación a la vez con un solo radio de cobertura.

Bluetooth transmite a una tasa de 1 Mbps en su funcionamiento básico, y de 2/3 Mbps en modo mejorado (Bluetooth 2.0). Utiliza modulación Gausiana por Desplazamiento de Frecuencia (*Gaussian Frequency Shift Keying*, GFSK) para el modo básico, y en el modo mejorado la Modulación Diferencial por Desplazamiento de Fase en Cuadratura (*Differential Quadrature Phase-Shift Keying*, DQPSK) y la Modulación Diferencial por Desplazamiento de Fase (*Differential Phase-Shift Keying*, 8-DPSK). Para lograr que la comunicación sea en dúplex se divide el tiempo de transmisión en ranuras por medio de la técnica Dúplex por División de Tiempo (*Time-Division Duplex*, TDD). La cobertura es de 100 metros para los dispositivos de clase 1 (100 mW), de 20 metros para los de clase 2 (10 mW) y de 10 metros para los de clase 3 (1 mW). Siendo los dispositivos de clase 3 los mayormente usados.

Bluetooth permite conectarse casi con cualquier dispositivo compatible que se halle en las proximidades, cuando los dispositivos se conectan se forma una piconet, en esta los dispositivos conectados comparten el mismo canal y adquieren dos roles distintos: maestro o esclavo. En cada piconet solamente puede existir un maestro y un máximo de siete esclavos, estos últimos no pueden establecer enlaces entre sí, por lo que todo el tráfico es enviado al maestro. Para que el dispositivo maestro logre establecer la comunicación en un principio, deberá ejecutar el protocolo de descubrimiento, este envía una señal en multidifusión para solicitar, básicamente a los dispositivos cercanos, su dirección MAC y el PIN. De esta manera es posible registrar a los dispositivos que se encuentran en estado de activo, sin importar el servicio que estén realizando.

Diversas aplicaciones, basada en Bluetooth, se han propuesto como soluciones a diferentes problemas del

quehacer humano. Así en [4] se presenta el desarrollo de un sistema de información contextual para terminales móviles que sirve como mecanismo de localización en ambientes tales como museos, monumentos, etc. Como apoyo a la red de transporte público en Italia, para llevar a cabo tareas de diagnóstico inalámbrico y mantenimiento preventivo en los autobuses [5]. Otra aplicación interesante es una alarma, como botón de pánico, que envía un mensaje de texto desde un teléfono móvil a un número de emergencia, en caso de que el portador del móvil sufra algún percance en la vía pública [6].

Por otro lado, se han desarrollado aplicaciones en el campo de la medicina, en el Centro Noruego para la Telemedicina, se ha desarrollado una aplicación que hace posible vigilar el nivel de glucosa en la sangre de manera remota. Esta aplicación está dirigida al auto cuidado de la diabetes en pacientes menores de edad [7]. Otras aplicaciones se centran en el mantenimiento y operación de estaciones de bombeo de agua en áreas urbanas para el control del agua [8].

No obstante, se han desarrollado muy pocas aplicaciones para el monitoreo, identificación y rastreo de dispositivos móviles y sus servicios, ese es el caso de *BlueSweep* de AirMagnet Inc que facilita a los usuarios de dispositivos móviles localizar sus dispositivos e identificar los servicios en los que se encuentran trabajando en tiempo real, con lo que se puede identificar las posibles amenazas de seguridad y llevar un registro de incidentes [20].

### III. DISEÑO DE LA APLICACIÓN

#### A. Antecedentes del Problema

Entre las soluciones propuestas para resolver el problema del control de teléfonos celulares en ambientes con restricciones de uso, podemos mencionar a los sistemas de bloqueo de llamadas que obstruyen la recepción en un radio de 30 metros aproximadamente, y que son utilizados principalmente para los sistemas penitenciarios. Entre los equipos que se ofertan podemos citar el *Radio Capsule SRC-300* [9] y el *Portable Palm Phone Jammer* [10] que cuentan con un alcance aproximado de 10 a 50 metros, los cuales pueden trabajar en frecuencias de 800 a 900 MHz y de 1930 a 1990 MHz, con un consumo de 5 voltios y son de dimensiones muy pequeñas.

Si partimos del hecho de que muchos de los teléfonos celulares incorporan la tecnología Bluetooth, entonces es posible plantear una solución para el problema que nos ocupa, por medio de una aplicación que permita detectar o identificar distintos marcas de teléfonos celulares activos y sus servicios, de una forma rápida y automática en un radio de hasta 100 metros utilizando su propio protocolo de descubrimiento de dispositivos.

Respetando el derecho que todos los seres humanos tenemos para comunicarnos, esta aplicación no deberá contraponerse a este derecho, por lo que se establece como requisito en su diseño las siguientes características:

Que los teléfonos celulares cuenten con la tecnología Bluetooth para que puedan ser detectados mediante su protocolo de descubrimiento.

Una vez detectados se va a transmitir una aplicación que al ser aceptada por el usuario desplegará un menú de texto en donde se podrán ver las opciones a seguir.

La consola de control inicia la comunicación enviando mensajes de texto a los teléfonos celulares que estén dentro de su alcance, para obtener su identificación, es decir su dirección física (MAC).

### B. Contexto de Operación

El sistema propuesto utilizará una consola de control para ser instalada en la cabina del auditorio, para propósitos de prueba. Esta consola enviará las señales de descubrimiento de los teléfonos celulares con un alcance de hasta 40 metros [11], si el auditorio sobrepasa la distancia estimada, se usarán equipos repetidores para establecer varias conexiones Bluetooth simultáneas con un alcance de hasta 200 metros.

### C. Interfaz de Usuario

Los requisitos para el diseño de la interfaz de la aplicación del lado de la consola son: que cuente con la tecnología Bluetooth, con una tarjeta inalámbrica y que el sistema operativo sea compatible con el software utilizado.

La interfaz de los teléfonos celulares deberá ser diseñada de tal manera que sea compatible con las características de despliegue y operación de la mayoría de las marcas de teléfonos, es decir en modo de texto y con selección por menú, utilizando los botones de desplazamiento convencionales. Ver Fig. 1.



Fig. 1. Interfaz del Usuario.

### D. Diseño de los objetos del sistema

El diseño de aplicaciones para teléfonos celulares es en base a la programación orientada a objetos, de esta manera la aplicación se basará en dos clases: *Listener* y *Auditorio*. La primera es esencial para el funcionamiento de la tecnología Bluetooth y la segunda es la aplicación en sí.

La función de *Listener*, propia de la tecnología Bluetooth, es detectar de forma automática los teléfonos celulares activos, confirmar su presencia y mostrarlos en la pantalla, además de reservar los servicios de operación del dispositivo maestro, en este caso de la consola de control. Utiliza el atributo *Vector Disp\_Encontrados* para poder realizar la comunicación automáticamente. También se emplearán funciones relacionadas con los servicios como son

*deviceDiscovered()* que realiza la función de detectar los teléfonos celulares con tecnología Bluetooth, *serviceSearchCompleted()* que recibe una afirmación a la búsqueda de servicios y *servicesDiscovered()* para confirmar la localización de servicios y mostrarla en pantalla [12].

La clase llamada *Auditorio* es usada para ejecutar la aplicación, en esta se incluyen los atributos y servicios que tienen que ver con la operación y el despliegue de mensajes presentados en la interfaz del usuario del teléfono celular.

### E. Funcionamiento esperado

En la Fig. 2., se muestra el protocolo para la instalación de la aplicación. La computadora de control inicia la comunicación enviando un mensaje en multidifusión (*Req\_Inquiry*) a los teléfonos celulares para su descubrimiento y con ello identifica su nombre y dirección física.

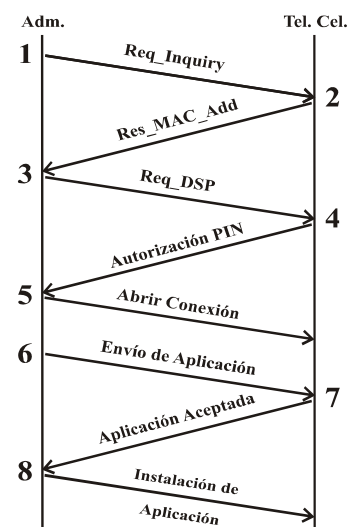


Fig. 2. Protocolo para la instalación de la aplicación.

En consecuencia, los teléfonos celulares envían el mensaje *Res\_MAC\_Add* como respuesta al requerimiento, enviando su dirección física y nombre. A continuación la consola de control enviará el mensaje *Req\_DSP* para decidir con que dispositivo establecerá la conexión y el servicio que usará. El teléfono celular enviará la *Autorización* del PIN, para que el dispositivo responda con los diferentes servicios que puede ofrecer. Una vez recibido el PIN se abre la conexión con el mensaje *Abrir Conexión*. Esto permitirá enviar la aplicación al dispositivo seleccionado. Finalmente sólo es necesario que el usuario instale la aplicación y haga uso de ella.

## IV. DESARROLLO DE LA APLICACIÓN

### A. Plataforma de desarrollo

El desarrollo de aplicaciones para teléfonos celulares se basa en la plataforma Java [13], con los programas Java para móviles: *Java 2 Micro Edition (J2ME)* que utiliza un subconjunto reducido de componentes Java SE, máquinas virtuales y la interfaz de programación de aplicaciones (*Application Programming Interface, API*) [14], Sun

Microsystems, Java JDK (*Java Development Kit*) para aplicaciones de teléfonos celulares [15] y *NetBeans* [16].

## B. Funciones

Para el desarrollo de la aplicación se definieron tres funciones: 1) Constructor de *Auditorio*, 2) Inicialización de la aplicación, y 3) Ejecución de la acción asignada a *Mobile Device*; mostradas en la Fig. 3.

```

* The Auditorio constructor.
*/
public Auditorio() { //Constructor
}
//<editor-fold defaultstate="collapsed"desc="Generated Method: initialize">
/**
 * Initalizes the application.
 */
private void initialize() { //Comenzar a Ejecutar el Programa
    // write pre-initialize user code here
}
//</editor-fold>

//<editor-fold defaultstate="collapsed"desc="Generated Method: startMIDlet">
/**
 * Performs an action assigned to the Mobile Device - MIDlet Started point.
 */
public void startMIDlet() { //Ejecución de Aplicación
    // write pre-action user code here
    switchDisplayable(getAlertaBienvenida(), getMenu());
    // Muestra la alerta Bienvenido y después el Menú en Pantalla
}
//</editor-fold>

```

Fig. 3. Funciones de Inicialización.

Adicionalmente se definieron otras funciones: *switchDisplayable* para el despliegue de mensajes en la pantalla del celular, ver Fig. 4.; *commandAction* que es la que especifica la aplicación que se enviará al dispositivo, ver Fig. 6.; *exitMIDlet* mostrada en la Fig. 5, que realiza la salida definitiva de la aplicación.

```

public void switchDisplayable(Alert alert, Displayable nextDisplayable) {
    // write pre-switch user code here
    Display display = getDisplay(); //Se asigna la variable de tipo Display
    if (alert == null) { //Se compara si Alert es de tipo nulo
        display.setCurrent(nextDisplayable); // Cambia de pantalla
    } else { // Si no se cumple la condición vuelve a cambiar de pantalla
        display.setCurrent(alert, nextDisplayable);
    }
    // write post-switch user code here
}
//</editor-fold>

```

Fig. 4. Función *switchDisplayable()*.

```

/**
 * Exits MIDlet.
 */
public void exitMIDlet() { //Sale del MIDlet
    switchDisplayable (null, null); //Coloca en nulo el switch
    destroyApp(true); //Recibe verdadero
    notifyDestroyed(); //Notifica la destrucción de la aplicación
}
/**

```

Fig. 5. Función *exitMIDlet()*.

```

public void commandAction(Command command, Displayable displayable) {
    if (displayable == Alerta) {
        if (command == ComandoRegresaAlerta) {
            switchDisplayable(null, getMenu());
        } else if (command == ComandoSalirAlerta) {
            exitMIDlet();
        }
    } else if (displayable == Menu) {
        if (command == ComandoContinuar) {
            int Seleccion;
            Seleccion = GrupoElementos.getSelectedIndex();

            //La variable Seleccion contiene el índice seleccionado del grupo de
            //elementos switch(Seleccion)
            {
                //Selecciona la opción alerta.
                case 0:
                {
                    switchDisplayable(null, getAlerta());
                    // Función de NetBeans que permite vibrar al seleccionar la opción
                    javax.microedition.lcdui.Display.getDisplay(this).vibrate(1000);
                    break;
                }
                //Selecciona la opción Recomendación.
                case 1:
                {
                    switchDisplayable(null, getRecomendacion());
                    // Función de NetBeans que permite vibrar al seleccionar la opción
                    javax.microedition.lcdui.Display.getDisplay(this).vibrate(1000);
                    break;
                }
                //Selecciona la opción Precaución.
                case 2:
                {
                    switchDisplayable(null, getPrecaucion());
                    // Función de NetBeans que permite vibrar al seleccionar la opción
                    javax.microedition.lcdui.Display.getDisplay(this).vibrate(1000);
                    break;
                }
                //Selecciona la opción Salir.
                case 3:
                {
                    exitMIDlet();
                }
            }
        } else if (command == ComandoSalir) {
            exitMIDlet();
        }
    } else if (displayable == Precaucion) {
        if (command == ComandoRegresaPrecaucion) {
            switchDisplayable(null, getMenu());
        } else if (command == ComandoSalePrecaucion) {
            exitMIDlet();
        }
    } else if (displayable == Recomendacion) {
        if (command == ComandoRegresaRecomendacion) {
            switchDisplayable(null, getMenu());
        } else if (command == ComandoSalirRecomendacion) {
            exitMIDlet();
        }
    }
}
//</editor-fold>

```

Fig. 6. Función *commandAction()*.

En la Fig. 7. se especifican las Funciones para *Pausar* y *Salir* del MIDlet, con tres opciones: 1) para el reinicio de la aplicación, después de que ha sido pausada cuando alguna acción del teléfono tiene una mayor prioridad, como por ejemplo la entrada de una llamada telefónica o un mensaje de texto. 2) Pausa del MIDlet, da un valor verdadero para que siempre que suceda alguna de las acciones antes mencionadas automáticamente realice la pausa y no exista conflicto con la acción prioritaria. 3) Destrucción de la aplicación. Es decir, borra la aplicación en el dispositivo.

```

public void startApp() { // Reinicia la aplicación pausada
    if (midletPaused) { resumeMIDlet ();}
    else {
        initialize ();
        startMIDlet ();
    }
    midletPaused = false;
}
public void pauseApp() { //Pausa la aplicación cuando es necesario
    midletPaused = true;
}
/**
 * Hace el llamado para poner fin al MIDlet.
 */
public void destroyApp(boolean unconditional) {
}
}

```

Fig. 7. Funciones para Pausar y Salir del MIDlet.

### C. Pruebas

Para efectos de prueba, se utilizó un auditorio común, durante la realización de un evento cultural, de 213 m<sup>2</sup> de área total. Se propusieron y se realizaron tres pruebas: 1) descubrimiento, 2) transmisión y 3) compatibilidad.

#### 1) Prueba de Descubrimiento

Las primeras pruebas de alcance y descubrimiento tienen como finalidad determinar los teléfonos celulares reconocidos por la consola de control. Para iniciar la búsqueda, la consola de control envía una señal en multidifusión para localizar los teléfonos por su dirección física y su nombre. Ver Fig. 8.

#### 2) Pruebas de Transmisión

A continuación se hace la transferencia de la aplicación en modo punto a punto. Sin embargo, hay que considerar que la transferencia puede ser interrumpida por diferentes causas: el rechazo en la transferencia por parte del usuario, pérdida en la conexión debido a ese rechazo, distancia mayor al alcance de la señal o interferencias con otras señales de radio. Para esos casos, la aplicación mostró en pantalla un mensaje de error.



Fig. 8. Teléfonos Celulares Seleccionados para la Transmisión de la Aplicación.

### 3) Pruebas de Compatibilidad

Aquí se verificó si la aplicación, es compatible con los diferentes fabricantes de teléfonos celulares. En este caso se probaron las marcas Sony Ericsson, Nokia, LG, Samsung y Motorola. Esta prueba se realizó porque, cada fabricante tiene instalados diferentes sistemas operativos, plataformas y versiones de Bluetooth.

## V. RESULTADOS

Para la obtención de resultados de las pruebas se empleó el programa SysNucleus USBTrace [17]. Los resultados obtenidos de la primera prueba se muestran en la Fig. 9., donde se puede observar del lado izquierdo de la imagen los dispositivos detectados. Mientras a la derecha se muestran los detalles del registro, entre ellos el más importante para nuestros propósitos es el estado de la conexión [18].

Los resultados de la segunda prueba se muestran en la Fig. 10., en esta se observa el éxito o interrupción de la transferencia. Los teléfonos celulares que no pudieron recibir la aplicación fueron los modelos SGH-J700 de Samsung, MG300 y MG800c de LG.

| Seq | Type   | Time     | Request                  | I/O | En... | Device Object    | IRP         | Status         |
|-----|--------|----------|--------------------------|-----|-------|------------------|-------------|----------------|
| 0   | STA... | 0.000000 | START OF LOG             |     |       |                  |             |                |
| 1   | URB    | 1.505278 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x8455CE... | STATUS_SUCCESS |
| 2   | URB    | 1.505282 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x8455CE... | STATUS_SUCCESS |
| 3   | URB    | 1.505313 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x8455CE... | STATUS_PENDING |
| 4   | URB    | 1.545280 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x8455CE... | STATUS_SUCCESS |
| 5   | URB    | 1.545310 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x8447BB... | STATUS_SUCCESS |
| 6   | URB    | 1.545319 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x8447BB... | STATUS_SUCCESS |
| 7   | URB    | 1.545381 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x8447BB... | STATUS_PENDING |
| 8   | URB    | 1.930284 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x8447BB... | STATUS_SUCCESS |
| 9   | URB    | 1.930332 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x8457CE... | STATUS_SUCCESS |
| 10  | URB    | 1.930348 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x8457CE... | STATUS_SUCCESS |
| 11  | URB    | 1.930409 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x8457CE... | STATUS_PENDING |
| 12  | URB    | 2.960261 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x8457CE... | STATUS_SUCCESS |
| 13  | URB    | 2.960469 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x84AA6...  | STATUS_SUCCESS |
| 14  | URB    | 2.960477 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x84AA6...  | STATUS_SUCCESS |
| 15  | URB    | 2.960542 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x84AA6...  | STATUS_PENDING |
| 16  | URB    | 3.009848 | CLASS_DEVICE             | OUT | 0     | \Device\USBPD... | 0x8457CE... | STATUS_SUCCESS |

Fig. 9. Resultados de la prueba de detección.

| Seq | Type   | Time     | Request                  | I/O | En... | Device Object    | IRP         | Status         | Buffer Snippet           |
|-----|--------|----------|--------------------------|-----|-------|------------------|-------------|----------------|--------------------------|
| 0   | STA... | 0.000000 | START OF LOG             |     |       |                  |             |                |                          |
| 1   | URB    | 0.904112 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x84AB6...  | STATUS_SUCCESS |                          |
| 2   | URB    | 0.904121 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x84AB6...  | STATUS_SUCCESS |                          |
| 3   | URB    | 0.904186 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x84AB6...  | STATUS_PENDING |                          |
| 4   | URB    | 0.904205 | BULK_OR_INTERRUPT_TRA... | OUT | 82    | \Device\USBPD... | 0x8D768E... | STATUS_SUCCESS |                          |
| 5   | URB    | 0.904212 | BULK_OR_INTERRUPT_TRA... | OUT | 82    | \Device\USBPD... | 0x8D768E... | STATUS_SUCCESS |                          |
| 6   | URB    | 0.904263 | BULK_OR_INTERRUPT_TRA... | IN  | 82    | \Device\USBPD... | 0x8D768E... | STATUS_PENDING |                          |
| 7   | URB    | 0.909360 | CLASS_DEVICE             | OUT | 0     | \Device\USBPD... | 0x844D3E... | STATUS_SUCCESS | 0C 08 02 2A 00           |
| 8   | URB    | 0.909364 | CLASS_DEVICE             | OUT | 0     | \Device\USBPD... | 0x844D3E... | STATUS_SUCCESS | 0C 08 02 2A 00           |
| 9   | URB    | 0.909401 | CONTROL_TRANSFER         | IN  | 0     | \Device\USBPD... | 0x844D3E... | STATUS_PENDING |                          |
| 10  | URB    | 0.914100 | CONTROL_TRANSFER         | IN  | 0     | \Device\USBPD... | 0x844D3E... | STATUS_SUCCESS |                          |
| 11  | URB    | 0.917072 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x84AB6...  | STATUS_SUCCESS | 0E 08 01 0C 08 00 2A ... |
| 12  | URB    | 0.917098 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x84AB6...  | STATUS_SUCCESS |                          |
| 13  | URB    | 0.917105 | BULK_OR_INTERRUPT_TRA... | OUT | 81    | \Device\USBPD... | 0x84AB6...  | STATUS_SUCCESS |                          |
| 14  | URB    | 0.917146 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x84AB6...  | STATUS_PENDING |                          |
| 15  | URB    | 0.924966 | CLASS_DEVICE             | OUT | 0     | \Device\USBPD... | 0x8454EC... | STATUS_SUCCESS | 0D 08 04 2A 00 05 00     |
| 16  | URB    | 0.924979 | CLASS_DEVICE             | OUT | 0     | \Device\USBPD... | 0x8454EC... | STATUS_SUCCESS | 0D 08 04 2A 00 05 00     |
| 17  | URB    | 0.925100 | CONTROL_TRANSFER         | IN  | 0     | \Device\USBPD... | 0x8454EC... | STATUS_PENDING |                          |
| 18  | URB    | 0.933085 | BULK_OR_INTERRUPT_TRA... | IN  | 81    | \Device\USBPD... | 0x84AB6...  | STATUS_SUCCESS | 1B 03 2A 00 05           |

Fig. 10. Resultados de la transferencia de la aplicación.

## VI. CONCLUSIONES

Los teléfonos celulares han facilitado en gran medida nuestra vida diaria, pero también se han convertido en un problema dado el abuso que se ha hecho de estos, sobre todo en aquellos ambientes donde está restringido su uso, las soluciones en hardware inhiben la recepción y transmisión de llamadas y atentan contra el derecho a la comunicación. Una alternativa es una aplicación de software basada en la tecnología Bluetooth, para la detección de teléfonos celulares activos y sus servicios, es decir que no sólo detecte la actividad en la recepción y transmisión de llamadas, sino en otras facilidades que requieren de autorización, y que además pueda adecuarse a las restricciones impuestas en distintos ambientes. Siendo la compatibilidad con distintas marcas de teléfonos celulares, el costo y la ubicuidad esenciales en su diseño.

## AGRADECIMIENTOS

M. A. Molina agradece el apoyo económico recibido del programa COFAA del IPN. R. Silva agradece el soporte económico recibido por la SIP y del programa EDI del IPN, así como del Sistema Nacional de Investigadores (SNI), México.

## REFERENCIAS

- [1] J. C. Cano, C. T. Calafate, P. Malumbres, M. Pietro Manzoni, "Redes inalámbricas ad hoc como tecnología de soporte para la computación Uuicua," Departamento de Sistemas Informáticos y Computacionales, Universidad Politécnica de Valencia, disponible en: [http://www.grc.upv.es/calafate/download/main\\_novatica.pdf](http://www.grc.upv.es/calafate/download/main_novatica.pdf), marzo 2008.
- [2] J. Muller Nathan, "Tecnología Bluetooth," Interamericana de España, McGraw Hill, 2002. Consulta: Febrero 2008.
- [3] M. P. Mata Ramírez, "Tecnología Bluetooth", GestioPolis, disponible en: <http://www.gestipolis.com/canales8/ger/tecnología-bluetooth.htm>, marzo 2008.
- [4] P. Pece-Juan, C. Fernández, C. Escudero, "Bluesic: Sistema de información contextual para terminales móviles basado en tecnología

Bluetooth," Departamento de Electrónica y Sistemas, Universidad de A. Coruña, marzo, 2008.

- [5] Centro Riserche FIAT Scpa de Orbassano y Digigrup Sri de Torino, "ATM Spa-Reference Application", disponible en: <http://spanish.bluetooth.com/NR/rdonlyres/235EC2BC-644F-4806-85ED-DD1666FEF43B/0/atm.pdf>.
- [6] O-S. Kell, "Head of business development, Bluetooth panic button potential to save lives," Securecom Technologies Ltd., disponible en: <http://spanish.bluetooth.com/NR/rdonlyres/EC33B6D5-A831-4D3B-886B-48EDEF4E60D8/0/securecom.pdf>, abril, 2008.
- [7] The Norwegian Center for Telemedicine (NST), NST reference application, disponible en: <http://spanish.bluetooth.com/NR/rdonlyres/BA25A6D5-15F0-437F-84C1-AABA58818BFD/0/Ddiabetes.pdf>, consulta: diciembre 2008.
- [8] Mr. O. Roland, ABBAS, Power Technologies Division, Norway, Oslo Municipaly Reference Application, disponible en: <http://spanish.bluetooth.com/NR/rdonlyres/4BCB7259-F643-4A27-855C-56875A78CDF8/0/oslo.pdf>, consulta: diciembre 2008
- [9] G. A. González Palacio, G.R. Peláez Gómez, "Bloqueo señal de celulares," *Informática Jurídica*, Universidad Autónoma Latinoamericana, Facultad de Derecho, Medellín, Octubre 2005, disponible en: <http://bloqueocelular.blogspot.com>, consulta: marzo2008.
- [10] A. Alegría, "Bloquea la señal de celular con Portable Palm Phone Jammer," Gadgets, disponible en: <http://tecnoday.net/2007/09/29/bloquea-la-senal-de-celular-con-portable-palm-phone-jammer/>, consulta: marzo 2008.
- [11] Toshiba. Satellite serie A210-A215 "Manual de Usuario," pp. 124-140.
- [12] . P.D. Borches Juzgado, C. Campo Vázquez, "Java 2 Micro Edition Soporte Bluetooth," Versión 1.0, Universidad Carlos III de Madrid, 20 de Marzo del 2004, consulta Octubre 2008.
- [13] Página Oficial de Java, "Java para Windows – Internet Explorer," disponible en: [http://java.com/es/download/windows\\_ie.jsp?locale=es&host=java.com:80&bhpc=1](http://java.com/es/download/windows_ie.jsp?locale=es&host=java.com:80&bhpc=1), consulta: octubre 2008.
- [14] "Java ME Applications Learning Trail," Página Oficial de NetBeans <http://www.netbeans.org/kb/trails/mobility.html>, consulta: Octubre 2008.
- [15] Página Oficial de Java, "Conozca más sobre la tecnología Java," disponible en: <http://java.com/es/about/>, consulta: octubre 2008.
- [16] "Página Oficial de NetBeans IDE 6.1," Disponible: <http://www.netbeans.org/index.html>, consulta: octubre 2008.
- [17] "USBTrace Tour: User Interface," Página Oficial de SysNucleus USBTrace, disponible en: [http://www.sysnucleus.com/usbtrace\\_tour.html](http://www.sysnucleus.com/usbtrace_tour.html), consulta: octubre 2008.
- [18] "USBTrace Tour: Viewing captured data", Página Oficial de SysNucleus USBTrace, disponible en: [http://www.sysnucleus.com/usbtrace\\_tour3.html](http://www.sysnucleus.com/usbtrace_tour3.html), consulta: octubre 2008.