



# Universidad Austral de Chile

---

Facultad de Ciencias de la Ingeniería  
Escuela de Ingeniería Civil en Informática

## **PROTOTIPO DE INTERFACE SAFETY FOR FIREFIGHTERS CON MÓDULOS XBEE SOBRE ANDROID Y SQLITE.**

Proyecto para optar al título de  
**Ingeniero Civil en Informática**

PROFESOR PATROCINANTE  
JORGE ANTONIO MORALES VILUGRÓN  
INGENIERO ELECTRÓNICO  
MAGISTER EN ADMINISTRACIÓN, MBA

PROFESOR CO-PATROCINANTE  
JUAN PABLO SALAZAR FERNÁNDEZ  
INGENIERO CIVIL EN INFORMÁTICA  
MAGISTER EN ADMINISTRACIÓN, MBA

PROFESOR INFORMANTE  
MARÍA ELIANA DE LA MAZA WERNER  
INGENIERO CIVIL EN INFORMÁTICA  
MAGISTER EN INFORMÁTICA EDUCATIVA

**JAIME ESTEBAN HAUSDORF DELGADO**

VALDIVIA – CHILE  
2013

## Dedicatoria

*A mis padres, Jaime Eduardo Hausdorf Niklitscheck y María Elena Delgado Cárdenas, todo lo que soy es gracias a ellos, a su apoyo, su educación, sus retos, su cariño. En realidad me faltan palabras para expresar mi eterna gratitud a estas dos grandes personas.*

*Mamá, gracias nunca me olvidaré de cómo me hacías tareas para que desarrolle en el campo, antes de entrar al colegio, cómo me ayudabas a buscar recortes y tareas tiempo después y cómo me ayudabas e incentivabas a estudiar y leer.*

*Papá, gracias por jugar conmigo aunque llegaras cansado del trabajo en el campo, gracias por tus enseñanzas y tus consejos. Y no olvides un papel que te regalé hace un tiempo y que es muy cierto aplicado a ti.*

*“Hay hombres que luchan un día, y son buenos.*

*Hay hombres que luchan un mes, y son mejores.*

*Pero hay hombres que luchan toda una vida, esos son IMPRESCINDIBLES.”*

*También agradecer a mis hermanas, Ximena y Pamela por todo el cariño y apoyo demostrado a lo largo de esta carrera y de toda la vida. Antes peleábamos bastante, pero ¿qué hermanos no pelean cuando chicos?. Muchas gracias a las dos por todo, de verdad.*

*A mi polola, aunque no sé si se aplica esa palabra, ahora me gusta más “Mi Mujer”, Mariela, muchas gracias por todo este tiempo que hemos estado juntos, por quererme, tenerme paciencia, cuidarme y ayudarme. Gracias por ese hermoso hijo que me has dado hace ya 4 meses y medio, medio mañosito pero son la razón de mi vivir, aunque suene cliché jejeje.*

*Te amo mucho y quiero que estemos juntos hasta viejitoos, caminando todos destartalados de la manito.*

*Finalmente, agradecer a los profesores que me han formado, desde kinder a la fecha, especialmente al equipo de trabajo de esta tesis, profe María Eliana, profe Juan Pablo y profe Jorge, por lo jugado que es este último para conmigo.*

*A todos ustedes, dedicado este trabajo, el esfuerzo de varios meses para lograr terminar la carrera y poder decir que soy un **Ingeniero Civil en Informática**.*

*Jaime Esteban Hausdorf Delgado*

## ÍNDICE

|  |     |
|--|-----|
| ÍNDICE .....   | i   |
| ÍNDICE DE TABLAS .....   | iii |
| ÍNDICE DE FIGURAS.....   | iv  |
| RESUMEN.....   | vi  |
| ABSTRACT .....   | vii |
| 1 INTRODUCCIÓN .....   | 1   |
| 1.1 Objetivos .....  | 1   |
| 1.2 Motivación.....  | 2   |
| 1.3 Impactos .....   | 3   |
| 2 NIVEL ACTUAL.....  | 4   |
| 2.1 Prevención de riesgos .....  | 4   |
| 2.1.1 Niveles de la prevención.....  | 5   |
| 2.1.2 Niveles en que se puede actuar para prevenir los riesgos laborales .....                               | 5   |
| 2.2 Seguridad bomberil .....   | 6   |
| 2.3 Certificación de calidad.....  | 7   |
| 3 TECNOLOGÍAS ASOCIADAS.....   | 8   |
| 3.1 Xbee y Zigbee .....  | 8   |
| 3.2 Tarjeta IOIO .....   | 11  |
| 3.3 Android.....   | 13  |
| 3.4 SQLite .....   | 15  |
| 4 DESCRIPCIÓN DE LA METODOLOGÍA .....  | 17  |
| 5 DESARROLLO DEL PROTOTIPO .....   | 18  |
| 5.1 Funcionalidades del producto.....  | 18  |
| 5.2 Requisitos no funcionales.....   | 18  |
| 5.3 Lista de prototipos o experimentos .....   | 19  |
| 5.3.1 Prueba de concepto N°1 “Hacer una aplicación básica en Eclipse sobre Android” .....                    | 19  |
| 5.3.2 Prueba de concepto N°2 “Comunicar la tarjeta IOIO con el teléfono inteligente” .....                   | 22  |
| 5.3.3 Prueba de concepto N°3 “Enviar datos desde la tarjeta IOIO al teléfono y almacenarlos en SQLite” ..... | 25  |
| 5.3.4 Prueba de concepto N°4 “Comunicar dos módulos Xbee de forma transparente inalámbricamente” .....       | 27  |

|  |    |
|--|----|
| 5.3.5 Prueba de concepto N°5 “Comunicar dos módulos Xbee vía API inalámbricamente” .....   | 29 |
| 5.3.6 Prueba de concepto N°6 “Enviar datos de sensores desde el Xbee receptor al teléfono móvil” .....                                   | 31 |
| 5.3.7 Prueba de concepto N°7 “Conectar sensor de luz al Xbee emisor, transmitir data y almacenar ésta en una base de datos SQLite” ..... | 33 |
| 5.4 Diseño.....  | 35 |
| 5.4.1 Casos de uso.....  | 35 |
| 5.4.2 Diagrama de clases .....   | 40 |
| 5.4.3 Diagrama de componentes.....   | 40 |
| 5.5 Solución final .....   | 42 |
| 5.5.1 Procedimiento.....   | 43 |
| 5.5.2 Código de la aplicación.....   | 46 |
| 5.5.3 Esquema del cableado final.....  | 48 |
| 5.5.4 Foto del sistema real.....   | 48 |
| 6 CONCLUSIONES .....   | 53 |
| 6.1 Conclusiones .....   | 53 |
| 6.2 Trabajo futuro.....  | 54 |
| 7 REFERENCIAS.....   | 56 |

## ÍNDICE DE TABLAS

| Tabla  | Página |
|--|--------|
| Tabla 1. Variación participación de mercado en EEUU, abril-julio 2012..... | 14     |
| Tabla 2. Funcionalidades del sistema.....                                  | 18     |
| Tabla 3. Requisitos no funcionales. ....                                   | 18     |
| Tabla 4. Curso normal de los eventos CU: Activar bombero. ....             | 37     |
| Tabla 5. Curso normal de los eventos CU: Ver datos.....                    | 39     |

## ÍNDICE DE FIGURAS

| Figura   | Página |
|--|--------|
| Figura 1. Tecnologías asociadas a la solución propuesta.....                       | 8      |
| Figura 2. Coordinador PAN con múltiples nodos.....                                 | 10     |
| Figura 3. Red mesh para módulos Xbee PRO.....                                      | 11     |
| Figura 4. Placa IOIO y sus componentes.....  | 13     |
| Figura 5. Opción para dejar el celular en modo depuración.....                     | 20     |
| Figura 6. Programa Eclipse con un proyecto Android nuevo.....                      | 21     |
| Figura 7. Pantalla del equipo Android con un mensaje “Hola mundo”.....             | 21     |
| Figura 8. Código necesario para el funcionamiento de la aplicación.....            | 22     |
| Figura 9. Placa IOIO con la luz de estado apagada.....                             | 22     |
| Figura 10. Programa PickIt3.....   | 23     |
| Figura 11. Led de estado encendido en la placa IOIO.....                           | 24     |
| Figura 12. Pantalla del teléfono con <i>IOIO Hardware Tester</i> en ejecución..... | 24     |
| Figura 13. Esquema de conexiones para probar este experimento.....                 | 25     |
| Figura 14. Extracto de código de esta prueba.....                                  | 26     |
| Figura 15. Placa IOIO antes y después del soldado de conectores.....               | 27     |
| Figura 16. Configuración de ambos Xbee para “chat”.....                            | 28     |
| Figura 17. Terminal de los módulos logrando el “chat”.....                         | 29     |
| Figura 18. Trama en el módulo receptor.....  | 30     |
| Figura 19. Conexión entre la IOIO, el Xbee receptor y el teléfono.....             | 31     |
| Figura 20. Parte de código fuente del experimento N°6.....                         | 32     |
| Figura 21. Esquema de cableado de esta prueba de concepto.....                     | 33     |
| Figura 22. Conexión del sensor de luz al módulo Xbee.....                          | 34     |
| Figura 23. Esquema del cableado de la prueba de concepto.....                      | 35     |
| Figura 24. Diagrama general de casos de uso.....                                   | 36     |
| Figura 25. Diagrama de pantalla CU: Activar bombero.....                           | 37     |
| Figura 26. Diagrama de colaboración de Activar bombero.....                        | 38     |
| Figura 27. Diagrama de pantalla CU: Ver datos.....                                 | 39     |
| Figura 28. Diagrama de colaboración de Ver Datos.....                              | 39     |
| Figura 29. Diagrama de clases de la solución propuesta.....                        | 40     |
| Figura 30. Diagrama de componentes.....  | 41     |
| Figura 31. Diagrama de la pantalla inicial.....                                    | 41     |
| Figura 32. Diagrama de la pantalla con la lista de bomberos.....                   | 42     |
| Figura 33. Diagrama de pantalla del detalle de un bombero.....                     | 42     |
| Figura 34. Aplicación en el <i>smartphone</i> .....                                | 43     |
| Figura 35. Lista desplegable de módulos Xbee disponibles.....                      | 44     |
| Figura 36. Lista con las personas enlazadas.....                                   | 45     |
| Figura 37. Datos de una persona seleccionada de la lista.....                      | 45     |
| Figura 38. Código de botón Enlazar.....  | 46     |
| Figura 39. Código de botón Siguiente.....  | 46     |
| Figura 40. Código de seleccionar del <i>ListBox</i> .....                          | 47     |
| Figura 41. Esquema del cableado de la solución propuesta.....                      | 48     |
| Figura 42. Módulo emisor.....  | 49     |

|  |    |
|--|----|
| Figura 43. Módulo receptor.....                              | 49 |
| Figura 44. Sistema completo para la solución planteada. .... | 50 |

## RESUMEN

Este Proyecto de Titulación, tiene como objetivo el desarrollo de un sistema integrado de Software –Hardware, que permita conocer una serie de datos ambientales a los que es sometido un bombero en el desarrollo de la profesión. Es decir, conocer datos que permitan saber que está pasando alrededor del bombero cuando está controlando un siniestro.

Esta información permitiría saber la temperatura a la que está sometido, si está erguido o ha caído producto de un accidente y/o de una falla en su organismo, los niveles de monóxido de carbono en el ambiente y cualquier otra medición para lo cual se cuente con el sensor adecuado.

Por otro lado, tenemos que el sistema operativo para móviles, Android, está teniendo un gran crecimiento tanto en la cantidad de usuarios como de dispositivos que lo han adoptado y un gran volumen de aplicaciones.

Para ello, a cada bombero se le instalará en la cintura una unidad que se encargará de captar y transmitir los datos. Estos datos serán captados por una *tablet* con sistema operativo Android, unidad que se encontrará en el carro y/o en algún lugar cercano al siniestro y donde la señal que se transmita sea captada por ésta.

Esta aplicación tiene como novedad el trabajar con Android, y tener que desarrollar las aplicaciones de captura con tecnología recientemente lanzada al mercado y que es exclusiva para este tipo de dispositivos. Además, los datos de cada bombero se almacenarán en una base de datos SQLite. Esto permitirá que además de ver los datos *online*, éstos se podrán evaluar “ex post”. Esto último dará una herramienta de gestión tendiente a mejorar los procedimientos al abordar un siniestro.

Además, se podría aplicar a otros siniestros con altos niveles de contaminación, sólo incorporando los sensores adecuados al tipo de dato que se estime relevante para la situación.



## **ABSTRACT**

This graduate project aims to develop a software-hardware program enable to know a sequence of environmental data that is submitted to a firefighter in the development of the profession. That means, find data to know what is happening around the firefighter when he is controlling a disaster.

This piece of information would help to know the temperature at which it is subjected, if it is upright or if it has fallen from an accident and/or or a defect in his system, the carbon monoxide levels in the environment and any other measurements for which record with the appropriate sensor.

Furthermore, we have mobile operating system, Android, is having a huge growth in both the number of users and devices that have adopted and a large volume of applications.

For that, to each firefighter will install in the waist a unit responsible for capturing and transmitting data. This data will be captured with Android operative system, a unit that will be in the car and/or somewhere close to the incident where the signals to be transmit and captured by this.

This application has as an innovation work with Android, and having to develop capture applications with recently launched technology that is exclusive to this type of device. In addition data of each firefighter will be stored in a SQLite database. This will also see the data online; this could be evaluated “ex post”. This will give a management tool aim at improving procedures to tackle a disaster.

Moreover, it could be applied to other high level contamination catastrophe only incorporating the suitable sensors for the type of data that is deemed relevant to the situation.

# 1 INTRODUCCIÓN

Existen muchas actividades profesionales que deben lidiar con situaciones con altos niveles de peligrosidad. Éstas pueden ser de altísima contaminación y/o de varios peligros asociados a la actividad de controlar un siniestro por parte de las brigadas de incendios en nuestro país y del mundo en general.

La irrupción de nuevas tecnologías nos desafía a desarrollar aplicaciones que permitan que estas actividades se encuentren más controladas. Una forma de controlarlas y a través de ello proteger a las personas que la desarrollan es conocer datos del entorno en que se encuentra el personal.

Hoy existen una serie de sensores que permiten tomar muestras de datos muy importantes para el personal a cargo de las brigadas, como lo son: niveles de temperatura, nivel de gases, nivel de monóxido de carbono, de posición y si está erguido o caído el sujeto.

Por otro lado, el gran desarrollo que están teniendo las unidades móviles, que cada día se comportan como verdaderas computadoras portátiles, con algunas ventajas de la portabilidad, menores costos, tecnologías asociadas y de duración de baterías. Por lo tanto, la tecnología disponible por un lado y la necesidad de protección del ser humano por otra, nos desafía a desarrollar sistemas que permitan conocer datos ambientales, en especial en trabajos de alta peligrosidad, usando herramientas desarrolladas para móviles y el desarrollar el hardware que se requiera para minimizar el riesgo y almacenar información que pueda ser usada para mejorar los procedimientos en el desarrollo de la profesión.

## 1.1 Objetivos

### **Objetivo general:**

Desarrollar un prototipo de captura y transmisión de sensores, el hardware y software de captura y almacenamiento, para sistemas operativos móviles Android.

### **Objetivos específicos:**

- Analizar las principales tecnologías disponibles para la *tablet* Samsung Galaxy Tab 10.1 que permitan recepcionar las señales de distintos sensores ambientales, mostrarlas en la pantalla y almacenarlas en una base de datos.
- Modelar y diseñar una arquitectura de software para la implantación del prototipo a desarrollar.
- Implementar el prototipo funcional de acuerdo a lo modelado en los objetivos específicos anteriores.
- Validar el resultado del proyecto obtenido al implementar la aplicación sobre una situación lo más real posible.

## **1.2 Motivación**

Existe seguridad limitada en lo que respecta a actividades que ponen en riesgo la vida de una persona, este es el caso de los bomberos, hombres y mujeres que arriesgan su propia integridad para salvar vidas de otros, que la mayoría de las veces, ni siquiera son conocidas.

Ya es hora de apoyarlos de alguna manera, y con esta propuesta podremos ayudar a resguardar la integridad física de los que luchan contra el fuego, ya sea en línea, teniendo los datos en el momento del siniestro para actuar luego de recibir una alerta, o luego de producido el suceso, hacer un análisis del por qué sucede algún evento, actuando finalmente de manera preventiva.

Así, la inclusión de la tecnología se hace necesaria en el desarrollo del proyecto, debido a su aporte a salvar vidas de las personas que arriesgan la suya. Tecnología que va desde la utilización de variados sensores hasta dispositivos móviles como la *tablet* con sistema operativo Android, y una base de datos para almacenar los datos y poder hacer estudios posteriores.

Por último, vemos que la mayor motivación de este proyecto es la ayuda que se prestará a personas que no dudan en arriesgar su vida para salvar otras.

### **1.3 Impactos**

Existen impactos tanto en lo social como en lo tecnológico. Debido a lo mencionado en el apartado anterior, si el objetivo del proyecto se cumple y las motivaciones son satisfechas, el impacto generado en la sociedad y en el cuerpo de bomberos, será alto, al tener un sistema que resguardará la seguridad de ambos grupos de personas.

En el caso tecnológico el proyecto se presenta como una innovación en el país, siendo una de las primeras iniciativas de este tipo, si no la primera ya que no se han encontrado casos de desarrollos similares.

Para la Universidad Austral de Chile significa ser reconocida por la sociedad por su aporte a la labor que ejercen cientos de personas que comprometen su vida sin medidas de seguridad como las que plantea este proyecto.

## **2 NIVEL ACTUAL**

### **2.1 Prevención de riesgos**

Prevención hace referencia a la acción y efecto de prevenir. El concepto, por lo tanto, permite nombrar a la preparación de algo con anticipación para un determinado fin, a prever un daño o a anticiparse a una dificultad, entre otros significados. [Def]

Riesgo, por su parte, tiene su origen etimológico más lejano en el vocablo árabe rizq, que significa “lo que depara la providencia”. El término está vinculado a la proximidad de un posible daño y a la vulnerabilidad. [Def]

Estas definiciones nos permiten comprender que el concepto de prevención de riesgos puede asociarse a la preparación de alguna medida defensiva para anticiparse y minimizar un daño que es posible que ocurra. En otras palabras: ante una situación o actividad que es inherentemente riesgosa por sus propias características, las personas toman ciertos recaudos por si el riesgo se materializa y se convierte en un peligro para la integridad. [Def]

La prevención de riesgos es muy importante en el trabajo, especialmente en aquellos que implican una mayor posibilidad de perjuicio para el trabajador (como la construcción, la minería o la industria química, por ejemplo). [Def]

El objetivo de esta prevención es reducir los accidentes de trabajo y minimizar los daños en caso que ocurran. La prevención de riesgos, por lo tanto, incluye un cierto sistema organizativo de la actividad y la utilización de uniformes adecuados para proteger la salud del trabajador. [Def]

El control de materiales y desechos tóxicos también forma parte de la prevención de riesgos en la industria para cuidar la salud del trabajador y para evitar la contaminación en los alrededores de las fábricas. [Def]

Como se menciona en un apartado anterior, “la prevención de riesgos es muy importante en el trabajo, especialmente en aquellos que implican una mayor posibilidad de perjuicio para el trabajador (como la construcción, la minería o la industria química, por ejemplo)”, ajustándose perfectamente también al trabajo que realizan miles de bomberos cada día.

### **2.1.1 Niveles de la prevención**

a) Prevención Primaria: Dirigida a evitar los riesgos o la aparición de los daños mediante el control eficaz de los riesgos que no puedan evitarse. Es la más eficaz y la más eficiente. [Bue10]

b) Prevención secundaria: Se produce cuando ha comenzado el proceso de alteración de la salud aunque no se manifieste de una manera clara; en general, puede tratarse de una fase inicial, subclínica, muchas veces reversible. [Bue10]

c) Prevención terciaria: Hay que aplicarla cuando existe una alteración patológica de la salud o durante la convalecencia de la enfermedad o posteriormente a la misma. Se trata de prevenir la reincidencia o las recaídas, o las posibles complicaciones o secuelas, mediante el adecuado tratamiento y rehabilitación, como principales medidas. [Bue10]

### **2.1.2 Niveles en que se puede actuar para prevenir los riesgos laborales**

Se puede actuar para prevenir los riesgos laborales a cuatro niveles:

a) En la concepción y el diseño. Aquí es necesario que los profesionales tengan asumida la filosofía preventiva junto con los conocimientos preventivos necesarios en su formación académica y de especialización. Se trata de evitar el riesgo, o en todo caso minimizarlo. [Bue10]

b) Sobre el origen del riesgo, con objeto de eliminar el riesgo, o en su caso reducirlo todo lo posible. [Bue10]

c) Sobre el medio ambiente de trabajo o medio de transmisión del riesgo, con objeto de controlar el riesgo. [Bue10]

d) Sobre la propia persona, con objeto de protegerla mediante equipos de protección individual contra los riesgos existentes, de promover y vigilar su salud y de educarla y formarla adecuadamente. Cabría añadir las instrucciones e información dirigidas individualmente. [Bue10]

Para las dos subsecciones anteriores, la solución planteada apunta en el primer caso al nivel indicado con la letra a), prevención primaria. Y en el caso de la segunda parte, el nivel tratado con este proyecto de titulación es el d), sobre la propia persona.

## 2.2 Seguridad bomberil

¿Cómo se resuelve la problemática de la integridad física de los individuos que participan en actividades de alta peligrosidad, como lo son los incendios?

Hoy en día existen diferencias sustanciales entre los diferentes cuerpos de bomberos del mundo, en Estados Unidos por ejemplo, se crean normas y hay instituciones especialmente dedicadas al cuidado de la integridad física de las personas, por ejemplo el Instituto Nacional para la Salud y Seguridad Ocupacional (NIOSH<sup>1</sup>) que es la agencia federal encargada de hacer investigaciones y recomendaciones para la prevención de enfermedades y lesiones relacionadas con el trabajo, cuya misión es generar nuevos conocimientos en el campo de la salud y seguridad ocupacional y adaptar esos conocimientos a la práctica para la mejora de la situación de los trabajadores. Para cumplir esta misión, NIOSH realiza investigaciones científicas, elabora directrices y recomendaciones de obligatoriedad, difunde información y responde a solicitudes para la realización de evaluación de riesgos de salud en el lugar de trabajo.

Este instituto trabaja por área ocupacional, y que en el caso del sector bomberil<sup>2</sup>, cuenta con gran cantidad de artículos sobre las recomendaciones y directrices para con los *firefighters*.

En el caso de Chile, lo más parecido a esta organización, podría ser la ACHS<sup>3</sup>.

Según NIOSH, existen 4 factores esenciales para seguridad de los bomberos:

- Seguir las políticas y procedimientos establecidos para el combate y extinción de los incendios.
- Seguir el programa de mantenimiento adecuado a los equipos de respiración autónomos.
- Establecer un plan logístico, que incluya la contabilidad de los bomberos en el lugar del incendio.
- Utilizar dispositivos de un sistema de seguridad de alerta personal P.A.S.S. (Sistema de seguridad de alerta personal)

---

<sup>1</sup><http://www.cdc.gov/spanish/niosh/index.html>

<sup>2</sup><http://www.cdc.gov/spanish/niosh/topics/bomberos.html>

<sup>3</sup><http://ww3.achs.cl>

La deficiencia en cualquiera de estos factores puede crear una situación de amenaza en la integridad física de los bomberos. [Bar08]

Así, tanto en el extranjero como a nivel nacional se ha hallado que la vestimenta es a lo que más se apunta, contando con varias prendas para la protección contra el fuego, como botas, esclavinas, cascos, entre otros. Pero lo que más se acerca a nuestra propuesta de solución es un sistema de alarmas llamado P.A.S.S. que da una alerta sonora cuando se está sin movimiento por un período determinado de tiempo, limitándose a que el sonido tenga la potencia necesaria para que un compañero del bombero que está inmóvil lo escuche y vaya en su auxilio.

Finalmente, destacar el adelanto tecnológico y social que significará desarrollar este prototipo para luego llegar a hacer el sistema a gran escala.

### **2.3 Certificación de calidad**

En el caso chileno, para la empresa vinculada a los siniestros forestales, Conaf, la entidad que certifica sus trajes e implementos es Cesmec<sup>4</sup>. Y para la institución dedicada a los incendios urbanos como lo es Bomberos de Chile, para adquirir sus productos se verifica que éstos estén certificados por el Idiem<sup>5</sup> de la Universidad de Chile.

Por lo indicado anteriormente, este prototipo debe primero pasar a la siguiente etapa de la fabricación. Esto implica desarrollar las placas electrónicas y el encapsulamiento con materiales de características industriales. Esto tanto para los dispositivos de captación y transmisión como el de recepción de los datos.

Contactados, vía email, con personal del Idiem, éstos indicaron que no certificaban este tipo de dispositivos. Esto no parece novedoso ya que dispositivos como éste, con tecnologías recién lanzadas al mercado requieren el estudio y creación de nuevas normas y referencia mundiales. Cabe indicar que en Estados Unidos se quiere integrar este tipo de tecnologías, en sus Bomberos Profesionales, para el año 2016.

De igual forma se procedió con Cesmec, quienes solicitaron más datos sobre las características del proyecto desarrollado en esta tesis. Al término de ésta, no se logra una respuesta concreta de dicho centro.

---

<sup>4</sup> <http://www.cesmec.cl/>

<sup>5</sup> <http://www.idiem.cl/>



### 3 TECNOLOGÍAS ASOCIADAS

Para lograr el objetivo planteado se necesitan un módulo Xbee (ver explicación en el siguiente punto) configurado como emisor, conectado a algún(os) sensor(es), otro módulo Xbee configurado como receptor y conectado a una placa IOIO (ver explicación en el subsiguiente punto) que servirá de nexo con el dispositivo de visualización y almacenamiento, que en este caso es un teléfono con sistema operativo Android.

Las tecnologías utilizadas en la solución planteada se muestran en la Figura 1, en donde se aprecia que el bombero deberá tener un módulo Xbee con un par de sensores y una fuente de alimentación. Por otro lado el comandante, tendrá un teléfono inteligente con sistema operativo Android, y una tarjeta IOIO para conectar el módulo Xbee receptor a dicho *smartphone*. Ambos actores estarán comunicados vía señales de radio frecuencia.

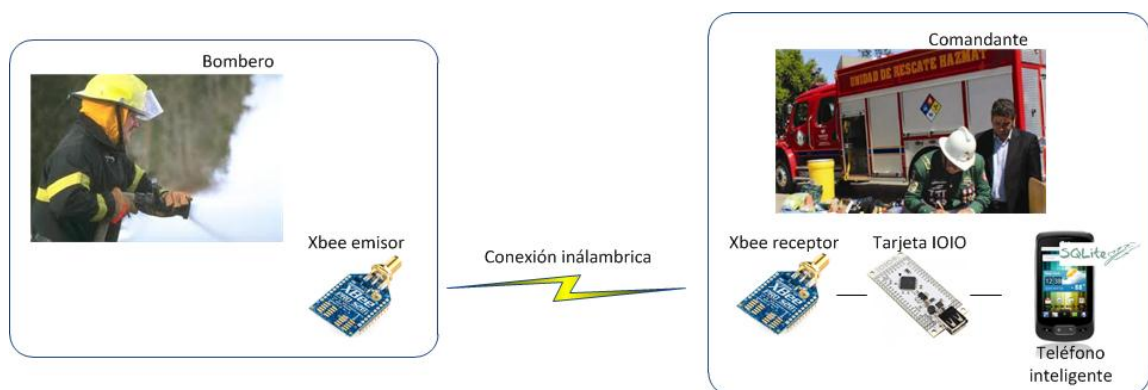


Figura 1. Tecnologías asociadas a la solución propuesta.

#### 3.1 Xbee y Zigbee

Zigbee es un protocolo de comunicaciones inalámbrico basado en el estándar de comunicaciones para redes inalámbricas IEEE\_802.15.4. Creado por Zigbee Alliance<sup>6</sup>, una organización, teóricamente sin ánimo de lucro, de más de 200 grandes empresas (destacan Mitsubishi, Honeywell, Philips, Invensys, entre otras), muchas de ellas fabricantes de semiconductores. Zigbee permite que dispositivos electrónicos de bajo consumo puedan realizar sus comunicaciones inalámbricas. Es especialmente útil para redes de sensores en entornos industriales, médicos y, sobre todo, domóticos. [Oya08]

Las comunicaciones Zigbee se realizan en la banda libre de 2.4GHz, una única frecuencia, es decir, de un canal. [Oya08]

<sup>6</sup><http://www.zigbee.org/>

Una red Zigbee la pueden formar, teóricamente, hasta 65535 equipos, es decir, el protocolo está preparado para poder controlar en la misma red esta cantidad enorme de dispositivos. Entre las necesidades que satisface el módulo se encuentran: [Oya08]

- Bajo costo.
- Ultra-bajo consumo de potencia.
- Uso de bandas de radio libres y sin necesidad de licencias.
- Instalación barata y simple.
- Redes flexibles y extensibles.

El uso del protocolo Zigbee va desde reemplazar un cable por una comunicación serial inalámbrica, hasta el desarrollo de configuraciones punto a punto, multipunto, *peer-to-peer* (todos los nodos conectados entre sí) o redes complejas de sensores. [Oya08]

Una red Zigbee la forman básicamente 3 tipos de elementos. Un único dispositivo coordinador, dispositivos *routers* y dispositivos finales (*endpoints*). [Oya08]

El coordinador.

Es el nodo de la red que tiene la única función de formar una red. Es el responsable de establecer el canal de comunicaciones y del PAN ID (identificador de red) para toda la red. Una vez establecidos estos parámetros, el coordinador puede formar una red, permitiendo unirse a él a dispositivos *routers* y *endpoints*. Una vez formada la red, el coordinador hace las funciones de *router*, esto es, participar en el enrutado de paquetes y ser origen y/o destinatario de información. [Oya08]

Los *routers*.

Es un nodo que crea y mantiene información sobre la red para determinar la mejor ruta para transmitir un paquete de información. Lógicamente un *router* debe unirse a una red Zigbee antes de poder actuar como *router* retransmitiendo paquetes de otros *routers* o de *endpoints*. [Oya08]

Dispositivos finales.

Los dispositivos finales no tienen capacidad de enrutar paquetes. Deben interactuar siempre a través de su nodo padre, ya sea este un coordinador o un *router*, es decir, no puede enviar información directamente a otro *enddevice*. Normalmente estos equipos van alimentados a baterías. El consumo es menor al no tener que realizar funciones de enrutamiento. [Oya08]

Cada módulo Zigbee, al igual que ocurre con las direcciones MAC de los dispositivos Ethernet, tiene una dirección única. En el caso de los módulos Xbee cada uno de ellos tiene una dirección única de 64bits que viene grabada de fábrica. Por otro lado, la red Zigbee, utiliza para sus algoritmos de ruteo direcciones de 16 bits. Cada vez que un dispositivo se asocia a una red Zigbee, el coordinador al cual se asocia le asigna una dirección única en toda la red de 16 bits. Por eso el número máximo teórico de elementos que puede haber en una red Zigbee es de  $2^{16} = 65535$ , que es el número máximo de direcciones de red que se pueden asignar. [Oya08]

Estos módulos Xbee, pueden ser ajustados para usarse en redes de configuración punto a punto, punto a multipunto o *peer-to-peer*. Un ejemplo se puede ver en la Figura 2, donde se muestra una conexión multipunto, con un coordinador, conectado a varios nodos. Digi<sup>7</sup> (compañía que desarrolla éstos módulos) tiene 2 series de módulos disponibles. Los módulos Xbee Serie 2 permite hacer redes *mesh*<sup>8</sup>, la serie 1 no. [Oya08]

También existen los llamados módulos Xbee PRO de la Serie 1 que se diferencian en la capacidad de alcance, permitiendo en algunos casos doblar la distancia de transmisión, ya que poseen una mayor potencia en la señal. Con los módulos Xbee PRO de la Serie 2, es posible crear redes más complejas, como las llamadas *mesh*. Estas permiten acceder a

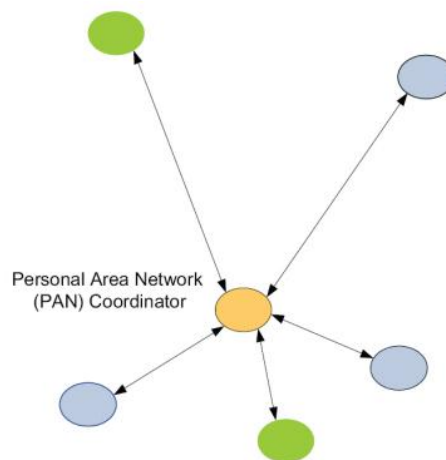


Figura 2. Coordinador PAN con múltiples nodos.<sup>9</sup>

un punto remoto, utilizando módulos intermedios para llegar como *routers*. Además los módulos automáticamente generarán la red entre ellos, sin intervención humana alguna,

<sup>7</sup><http://www.digi.com/>

<sup>8</sup> En español, red malla, es una topología de red en la que cada nodo está conectado a todos los nodos.

<sup>9</sup>[http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia\\_Usuario.pdf](http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario.pdf)

permitiendo la reparación de la red en caso de que algún nodo falle. Al mismo tiempo la red por sí sola resuelve la mejor ruta para un determinado paquete. [Oya08]

La Figura 3 muestra un ejemplo de una red *mesh*. Se observa que se desea acceder al punto B a partir del punto A. Suponiendo que la distancia entre A y B es demasiado para que alcance la señal, se utiliza la red *mesh* para poder alcanzarla, así cada nodo ubicado en medio del camino mostrado en celeste, funciona como módulo transparente, donde todo lo que le llega es retransmitido hacia el punto A. Esto funciona para cada uno de los módulos de la red. [Oya08]

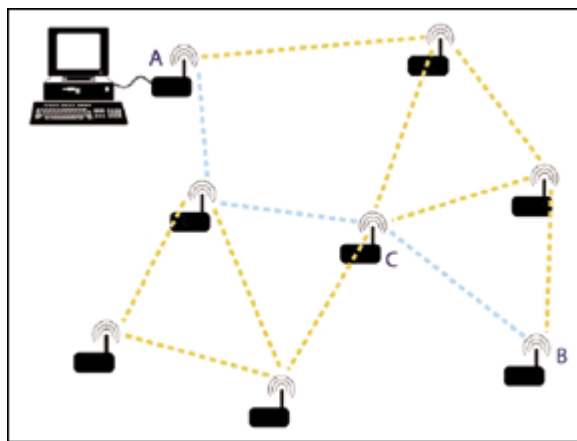


Figura 3. Red mesh para módulos Xbee PRO<sup>10</sup>.

### 3.2 Tarjeta IOIO

El IOIO (pronunciado como yo-yo) es una tarjeta especialmente diseñada para trabajar con dispositivos Android (versión OS 1.5 o mayor). La tarjeta provee una conectividad robusta a cualquier dispositivo Android vía conexión USB y es totalmente controlable desde dentro de las aplicaciones Android usando un API de Java simple e intuitivo – no será necesario ningún programador integrado o externo.

La tarjeta IOIO contiene un microcontrolador único que actúa como un host USB e interpreta comandos desde las aplicaciones Android. Adicionalmente, el IOIO puede interactuar con dispositivos periféricos en la misma forma como lo hacen la mayoría de los microcontroladores. Los códigos para controlar estas interfaces están escritos de la misma forma en la que se escriben las aplicaciones Android con la ayuda de una librería de nivel de aplicaciones fácil de usar. En otras palabras, se puede combinar la increíble

<sup>10</sup><http://www.emb.cl/electroindustria/articulo.mvc?xid=745&edi=42&xit=un-premier-sobre-zigbee>

potencia de cómputo, la conectividad Internet/Bluetooth, pantalla *touch* y una variedad de sensores del dispositivo Android con la habilidad de adicionar fácilmente dispositivos periféricos para interactuar con el mundo externo. También, usar el IOIO no requiere de ninguna modificación de hardware o software de tu dispositivo Android, lo que permite conservar la garantía.

El IOIO actúa como un host USB y se conecta con los dispositivos Android que posean la característica de dispositivo esclavo USB. [Oli]

La placa IOIO contiene los siguientes componentes:

- Conector USB hembra (tipo A): Usado para conectar al dispositivo Android.
- Pines GND: Conexión a tierra.
- Pines Vin: Usado para el suministro de energía a la tarjeta.
- Pines 5V: Normalmente usado como salida 5V cuando la placa se alimenta desde Vin.
- Pines 3.3V: Salida 3.3V
- Pines E/S (48 pines, numerados 1-48): Pines de E/S de propósito general.
- Led encendido: Se ilumina cuando el IOIO está recibiendo corriente.
- Led estado: Se ilumina brevemente durante el arranque y luego se hace bajo control de aplicaciones.
- Pin mclr: Normalmente no se utiliza. Su propósito es para la programación de nuevo gestor de arranque en el tablero IOIO.
- Regulador de voltaje: Ajusta la cantidad de corriente de carga suministrada en la línea VBUS del USB en el dispositivo Android. Girando en la dirección (+) aumenta la corriente de carga.

En la figura 3 se aprecia una tarjeta IOIO, indicándose los principales pines y partes.

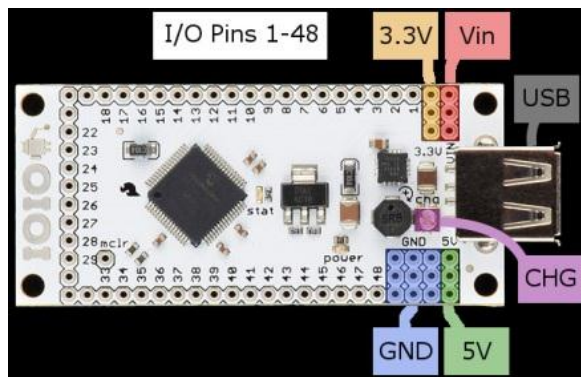


Figura 4. Placa IOIO y sus componentes<sup>11</sup>.

### 3.3 Android

Android es un sistema operativo móvil basado en Linux, que junto con aplicaciones *middleware* está enfocado para ser utilizado en dispositivos móviles como teléfonos inteligentes, tabletas, Google TV y otros dispositivos. Es desarrollado por la *Open Handset Alliance*<sup>12</sup>, la cual es liderada por Google. Este sistema por lo general maneja aplicaciones como Google Play.

Fue desarrollado inicialmente por Android Inc.<sup>13</sup>, una firma comprada por Google en 2005. Es el principal producto de la *Open Handset Alliance*, un conglomerado de fabricantes y desarrolladores de hardware, software y operadores de servicio. Las unidades vendidas de teléfonos inteligentes con Android se ubican en el primer puesto en los Estados Unidos, en el segundo y tercer trimestre de 2010, con una cuota de mercado de 43,6% en el tercer trimestre. A nivel mundial alcanzó una cuota de mercado del 50,9% durante el cuarto trimestre de 2011, más del doble que el segundo sistema operativo (iOS de Apple Inc.) con más cuota.

Tiene una gran comunidad de desarrolladores escribiendo aplicaciones para extender la funcionalidad de los dispositivos. A la fecha, se han sobrepasado las 600.000 aplicaciones (de las cuales, dos tercios son gratuitas) disponibles para la tienda de aplicaciones oficial de Android: Google Play, sin tener en cuenta aplicaciones de otras tiendas no oficiales para Android, como pueden ser la *App Store* de Amazon o la tienda de aplicaciones Samsung Apps de Samsung.

<sup>11</sup><https://github.com/ytai/ioio/wiki/Getting-To-Know-The%20Board>

<sup>12</sup><http://www.openhandsetalliance.com/>

<sup>13</sup><http://www.android.com/>

Cuota de mercado.

La compañía de investigación de mercado Canalys<sup>14</sup> estima que en el segundo trimestre de 2009, Android tendría 2,8% del mercado de teléfonos inteligentes a nivel mundial.

En febrero de 2010, ComScore<sup>15</sup> dijo que la plataforma Android tenía el 9% del mercado de teléfonos inteligentes en los Estados Unidos, como estaba tasado por los operadores. Esta cifra fue superior al estimado anterior de noviembre de 2009, el cual fue del 9%. Para finales del tercer trimestre de 2010, el mercado de Android en los Estados Unidos había crecido en un 21,4%.

En la tabla 1, se muestra lo que arroja un estudio de ComStore, más de 114 millones de personas en los EE.UU. dueños de teléfonos inteligentes, durante tres meses (hasta julio), subiendo un 7 por ciento frente a abril. Google Androides clasificada como la plataforma de teléfonos inteligentes más alta, con un 52,2 por ciento del mercado (aumentando 1,4 puntos porcentuales), mientras que la participación de Apple aumentó 2 puntos porcentuales, hasta el 33,4 por ciento. RIM (BlackBerry) ocupó el tercer lugar con una cuota del 9,5 por ciento, seguido por Microsoft (3,6 por ciento) y Symbian (0,8 por ciento). [Flo12].

Tabla 1. Variación participación de mercado en EEUU, abril-julio 2012.

|                                     | <i>Share (%) of Smartphone Subscribers</i> |               |                     |
|-------------------------------------|--|---------------|---------------------|
|                                     | <i>Apr-12</i>                              | <i>Jul-12</i> | <i>Point Change</i> |
| <i>Total Smartphone Subscribers</i> | 100.0%                                     | 100.0%        | N/A                 |
| Google                              | 50.8%                                      | 52.2%         | 1.4                 |
| Apple                               | 31.4%                                      | 33.4%         | 2.0                 |
| RIM                                 | 11.6%                                      | 9.5%          | -2.1                |
| Microsoft                           | 4.0%                                       | 3.6%          | -0.4                |
| Symbian                             | 1.3%                                       | 0.8%          | -0.5                |

A nivel mundial, según un estudio realizado por International Data Corporation (IDC<sup>16</sup>), de los 152,3 millones de *smartphones* vendidos en todo el mundo, el 59 por ciento utilizan el sistema operativo móvil de Google. A Android le sigue el SO móvil de Apple que está presente en el 23 por ciento de los terminales.

<sup>14</sup><http://www.canalys.com/>

<sup>15</sup><http://www.comscore.com/>

<sup>16</sup><http://www.idc.com/>

Android terminó el trimestre como líder entre los sistemas operativos móviles, lo que representa más de la mitad de todas las ventas de teléfonos inteligentes. Además, Android cuenta con una larga lista de fabricantes que incorporan el SO en sus *smartphones*. La surcoreana Samsung fue la que más ha contribuido al éxito de Android, con el 45,4 por ciento de todas las ventas de teléfonos inteligentes con Android. [Lib12]

Actualmente, la firma asegura que se activan 1.000.000 dispositivos en el mundo cada día. En diciembre de 2011 se activaban 700.000 dispositivos diariamente, anteriormente en julio de 2011 se declaró que se activan unos 550.000 dispositivos Android cada día en comparación con diciembre de 2010 que se activaban 300.000 dispositivos móviles con Android, y los 100.000 que se activaban en mayo de 2010.

### 3.4 SQLite

¿Qué es SQLite?

SQLite es un proyecto de dominio público creado por D. Richard Hipp que implementa una pequeña librería de aproximadamente 500Kb programada en lenguaje C, que funciona como un sistema de gestión de base de datos relacionales. A diferencia de los motores de base de datos convencionales con la arquitectura cliente-servidor, SQLite es independiente, ya que no se comunica con un motor de base de datos, sino que las librerías de SQLite pasan a integrar la aplicación. La misma utiliza las funcionalidades de SQLite a través de llamadas simples a sub rutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un solo fichero estándar, en la máquina local.

Características de SQLite.

SQLite goza de un par de características que lo hacen único, alguna de ellas se detallan a continuación.

Cero configuraciones.

De la forma en que fue concebido y diseñado SQLite, no necesita ser instalado. No prender, reiniciar o apagar un servidor, e incluso configurarlo.



## Portabilidad

SQLite puede ser ejecutado en diferentes sistemas operativos, como ser Windows, Linux, BSD, Mac OS X, Solaris o estar embebido en muchos otros como Android, Symbian, Palm OS. La portabilidad no está dada en sí por el software, sino por la base de datos condensada en un solo archivo, que puede estar situado en cualquier directorio, trayendo como ventaja que la base de datos puede ser fácilmente copiada a algún dispositivo USB o ser enviada vía correo electrónico.

## Registros de longitud variable

Generalmente los motores asignan una cantidad fija de espacio en disco para cada fila en la mayoría de los campos de una determinada tabla. Por ejemplo, tomemos un campo de tipo VARCHAR(100), esto significa que el motor le asignará 100 bytes de espacio fijo en disco, independientemente de la cantidad de información que se almacene en ese campo. En cambio, SQLite realizará todo lo contrario, utilizando para ello la cantidad de espacio en disco necesario para almacenar la información real del campo. Tomando el ejemplo anterior, si se quisiera almacenar un solo carácter en un campo definido como VARCHAR(100), entonces un único byte de espacio de disco se consume.

## Pedagógico

Algunas veces cuando se está en un ámbito educativo, se tiende a enseñar SQL para administrar las bases de datos. Muchas veces transportar una base a casa no es tarea fácil, por eso SQLite está siendo adoptado de una forma pedagógica. Por ejemplo, los estudiantes pueden enviar vía-mail las bases de datos a los docentes para su presentación y los docentes realizarán las observaciones necesarias.

## Limitaciones

Falta de Clave Foránea: se hace caso omiso de las claves foráneas; esto quiere decir, cuando se realiza la creación de la tabla desde el modo consola, está permitiendo el uso de la clausura, aunque no realizará el chequeo de la misma.

Falta de documentación en español: al momento de escribir este documento es muy escasa la documentación disponible en español sobre SQLite. [Cab07]

## **4 DESCRIPCIÓN DE LA METODOLOGÍA**

En general el desarrollo del proyecto se basará en la búsqueda en Internet, manuales de proveedores y el continuo apoyo del profesor patrocinante, mostrando avances periódicos e incrementales.

En particular, para realizar el objetivo 1 se comenzará investigando en Internet lo referente a los sensores a utilizar. Otro punto a buscar por el mismo medio es lo referente a dispositivos que faciliten la comunicación entre sensores, dispositivos de envío y captación de las señales.

En el objetivo específico 2 se deberá trabajar en la “arquitectura” que sostendrá a este prototipo, nótese que arquitectura se refiere a la base sobre la cual se pondrá el sistema. En el objetivo 3 se deberá implementar el prototipo en base a todas las normas y especificaciones descritas anteriormente, con la construcción de una interfaz de usuario amigable, soportado por una base de datos disponible para dispositivos móviles con Android.

Finalmente, en el objetivo 4 se validará la solución realizada haciendo encuestas de satisfacción a los usuarios en un ambiente posterior a la prueba del sistema.

Cabe señalar que todo el desarrollo del proyecto se contempla dentro de un marco de trabajo evolutivo, debido a que el contacto con el cliente/patrocinante será constante para validar que lo que se está realizando concuerda con sus necesidades, además de que al tratarse de un desarrollo innovador que involucra trabajo con hardware y software, deben realizarse constantes experimentos o pruebas de concepto para cada “componente” del sistema a implementar.

## 5 DESARROLLO DEL PROTOTIPO

### 5.1 Funcionalidades del producto

En la Tabla 2 se enuncian las características del producto desarrollado.

Tabla 2. Funcionalidades del sistema.

| Ref. # | Función  | Categoría |
|--------|--|-----------|
| F1     | El dispositivo Xbee debe ser capaz de capturar y enviar información de distintos sensores (temperatura, monóxido de carbono, acelerómetro, entre otros) a un módulo Xbee que funcionará como receptor central. | Evidente  |
| F2     | El Xbee receptor debe ser capaz de enviar los datos recibidos vía cableado a la tarjeta IOIO.  | Evidente  |
| F3     | Desde el dispositivo inteligente se debe obtener desde la tarjeta IOIO los datos de los sensores.  | Evidente  |
| F4     | El comandante debe poder ver el detalle de los últimos cinco datos de un bombero cualquiera.   | Evidente  |
| F5     | El sistema debe ser capaz de asociar un dato de entrada a un módulo Xbee emisor (bombero).   | Evidente  |

### 5.2 Requisitos no funcionales

En la siguiente tabla, Tabla 3, se indican los requisitos no funcionales del sistema a desarrollar en este proyecto.

Tabla 3. Requisitos no funcionales.

| Ref. # | Función   | Categoría |
|--------|---|-----------|
| F6     | El <i>software</i> debe ser desarrollado para la plataforma <i>Android</i> .    | Evidente  |
| F7     | Los datos deben ser almacenados en una base de datos SQLite.                    | Evidente  |
| F8     | Pérdida de datos 30-40 % a cinco segundos<br>Pérdida de datos 0% a 20 segundos. | Evidente  |
| F9     | Temperatura operación: -40 a 85° C (industrial)                                 | Evidente  |

## 5.3 Lista de prototipos o experimentos

En este apartado se indican los experimentos realizados para llegar posteriormente a la solución planteada, partiendo por pruebas sobre desarrollo de aplicaciones en Android, luego comunicar el teléfono con la placa IOIO que servirá de nexo con el módulo Xbee receptor. Posterior a esto, los datos recibidos en el *smartphone* deben ser almacenados en una base de datos SQLite que almacenará los datos enviados por los sensores conectados al Xbee emisor. Luego se prueba la comunicación entre los módulos Xbee de forma transparente y API, para en una nueva prueba enviar datos desde el Xbee receptor hacia el teléfono a través de la placa IOIO. Finalmente, conectar sensores al módulo Xbee emisor, transmitir los datos y almacenarlos en el dispositivo Android.

La configuración de software y hardware para la realización de este desarrollo es la que se describe a continuación.

Configuración de hardware.

- Notebook Dell XPS 15z, con procesador de Intel Core i5-2410M 2.3 GHz.
- Dos módulos XBee Pro 50mW RPSMA - Series 2 (Mesh).
- Tarjeta de desarrollo IOIO para Android.
- Smartphone LG P500h Optimus One con Android 2.3.3

Configuración de software.

- IDE Eclipse v3.7 Indigo
- Java JDK 1.6 (Android requiere esta versión).
- X-CTU software.

### 5.3.1 Prueba de concepto N°1 “Hacer una aplicación básica en Eclipse sobre Android”

El inicio de estas pruebas, contempla el lograr el famoso “Hola mundo” en Android, asunto que no fue logrado en pocos intentos debido principalmente a la versión del JDK de Java puesto en el equipo portátil desarrollador. Esta versión debe ser la 1.6. Además de este traspie, para lograr desarrollar en este ambiente es necesario tener el *smartphone*

en modo de depuración, objetivo que se logra activando la opción del menú que se muestra en la Figura 5. Para llegar allí en el teléfono utilizado es necesario acceder a “Menú > Ajustes > Aplicaciones > Desarrollo > Depuración USB”.

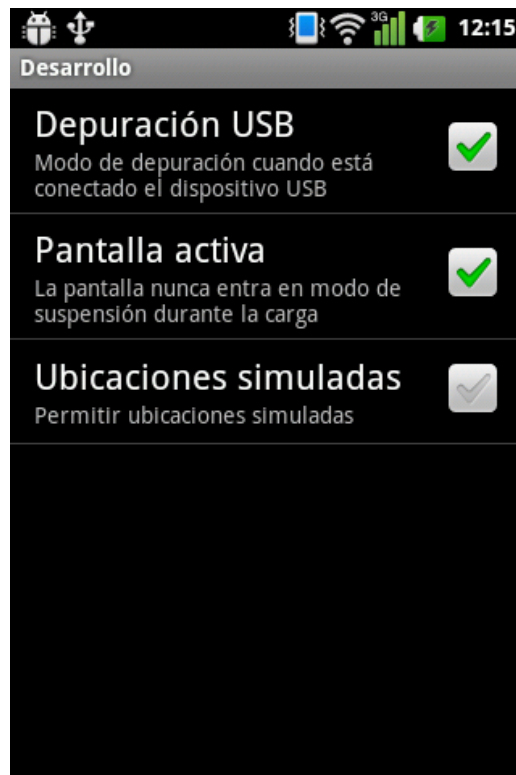


Figura 5. Opción para dejar el celular en modo depuración.

Además de esto, se debe tener instalado en el equipo, el programa para acceso al teléfono móvil, que en este caso es el LG PC Suite IV, software disponible en la página web del fabricante.

Para ayudar al desarrollador inicial sobre Android, existe una página web oficial<sup>17</sup> dedicada a proveer de algunas de las herramientas necesarias para comenzar a trabajar en este sistema móvil.

En la Figura 6, se presenta el programa Eclipse Indigo con un proyecto Android creado, como se ve, es una estructura de árbol de carpetas, donde en SRC, podemos incluir el código principal de nuestra aplicación.

---

<sup>17</sup><http://developer.android.com/index.html>

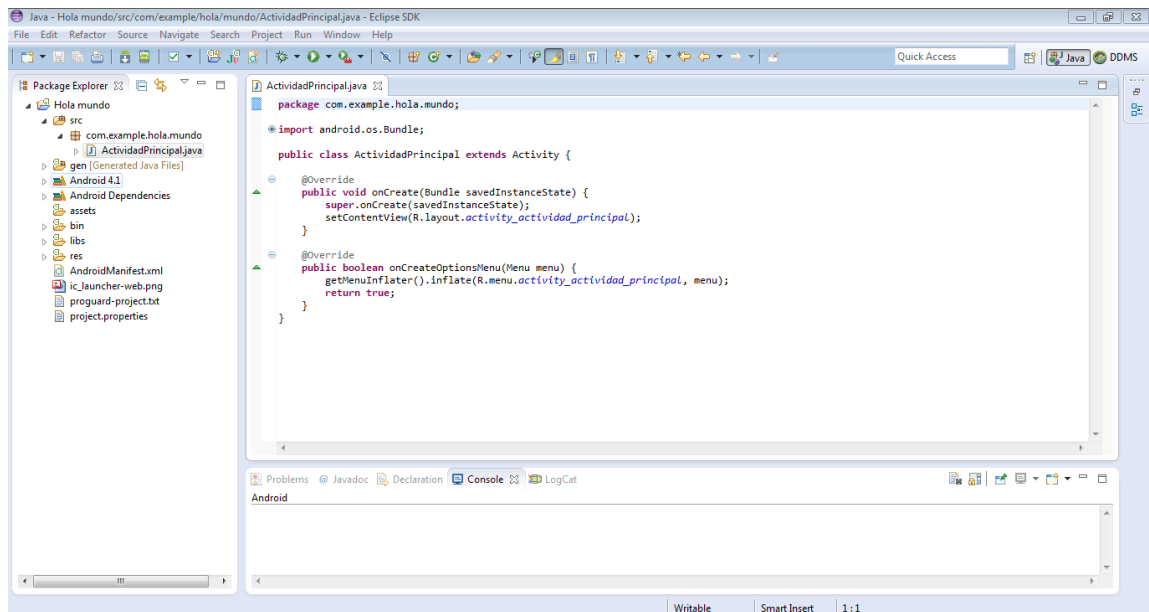


Figura 6. Programa Eclipse con un proyecto Android nuevo.

Luego de todo lo anterior, podemos lograr el objetivo de este experimento, resultado que se muestra en la Figura 7 y que en la Figura 8 se exhibe parte del código necesario para llegar a este fin.



Figura 7. Pantalla del equipo Android con un mensaje “Hola mundo”.

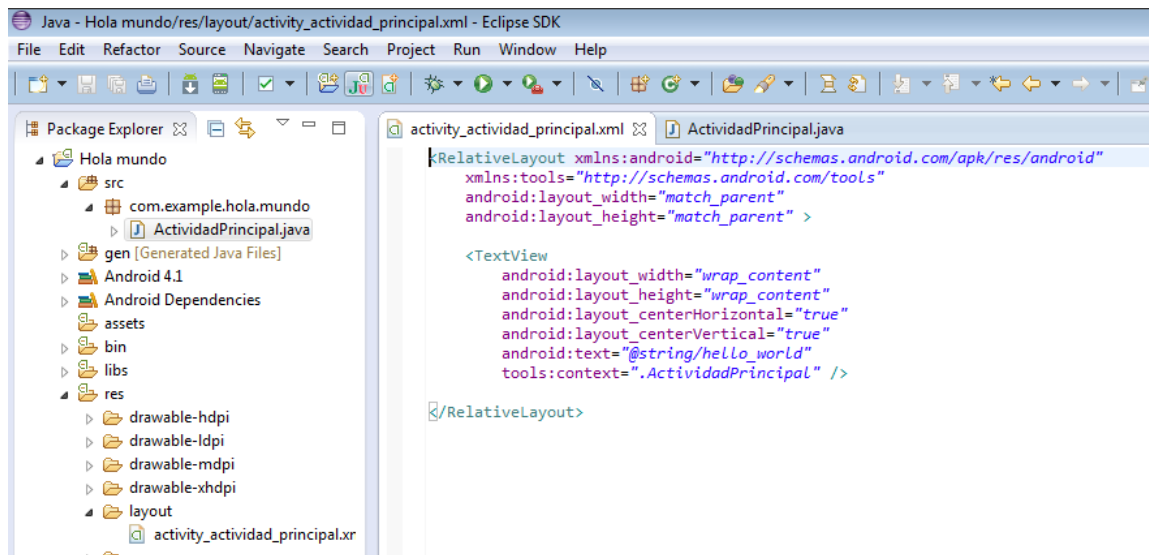


Figura 8. Código necesario para el funcionamiento de la aplicación.

### 5.3.2 Prueba de concepto N°2 “Comunicar la tarjeta IOIO con el teléfono inteligente”

El segundo experimento consiste en lograr que el *smartphone* logre detectar la tarjeta IOIO y por medio de una aplicación disponible en la tienda de Android, *Play Store*, llamada *IOIO Hardware Tester*, la funcionalidad de dicha aplicación es encender y apagar la luz de estado de la placa. En la Figura 9 se aprecia la tarjeta con el led antes mencionado apagado.

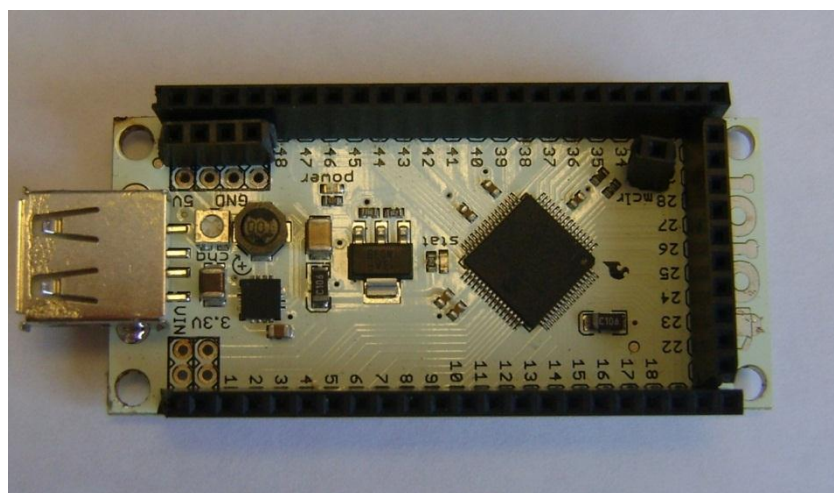


Figura 9. Placa IOIO con la luz de estado apagada.

Luego de varios intentos fallidos se logra determinar que es necesaria una actualización del *firmware* con el que viene la tarjeta, para lo que es utilizado un programa llamado PickIt3, el cual se ve en la Figura 10.

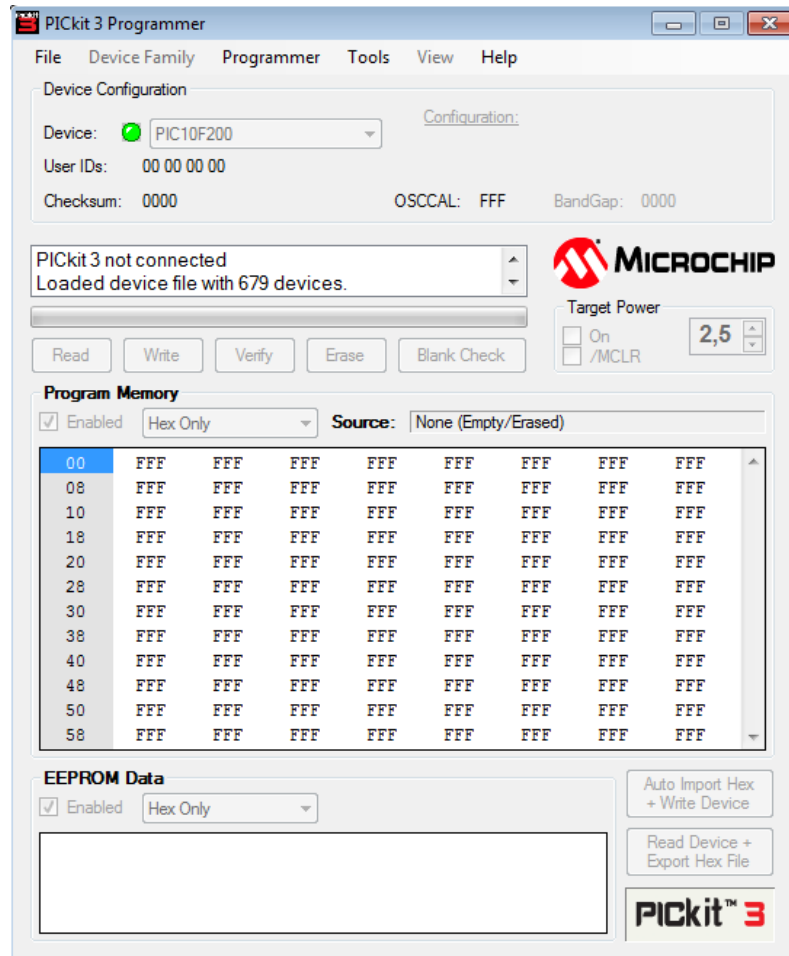


Figura 10. Programa PickIt3.

Una vez actualizado el firmware, la aplicación es probada con resultado satisfactorio. En la Figura 11, vemos la luz encendida en la placa y en la Figura 12 se muestra la pantalla del dispositivo Android con la aplicación *IOIO Hardware Tester* en ejecución.



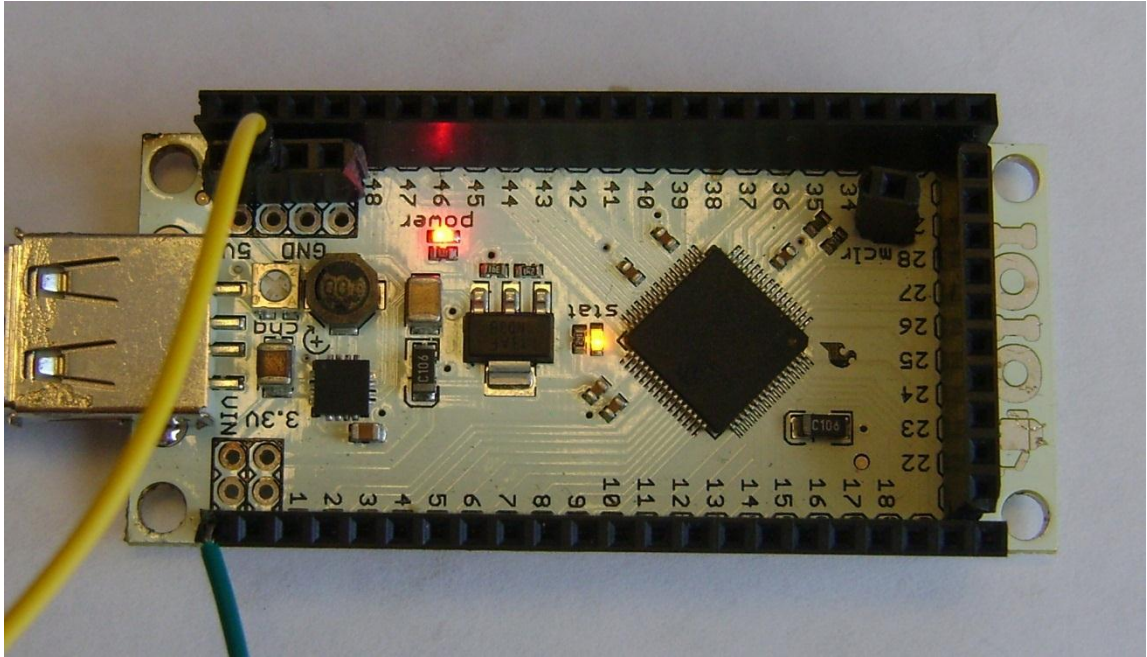


Figura 11. Led de estado encendido en la placa IOIO.

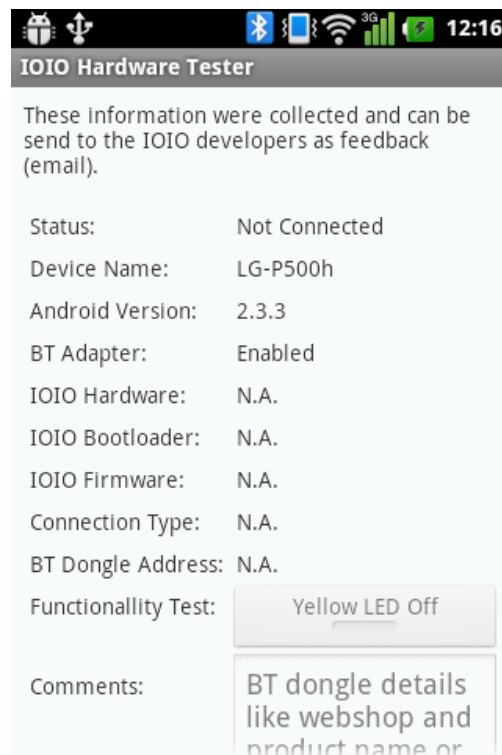


Figura 12. Pantalla del teléfono con *IOIO Hardware Tester* en ejecución.

### 5.3.3 Prueba de concepto N°3 “Enviar datos desde la tarjeta IOIO al teléfono y almacenarlos en SQLite”

El objetivo de este experimento es que teniendo conectada la placa al teléfono, ésta sea capaz de capturar datos de sensores conectados directamente a ella, pasarlos al teléfono y este último ser capaz de almacenarlos en una base de datos. A continuación, en la Figura 13, se muestra el esquema de las conexiones hechas.

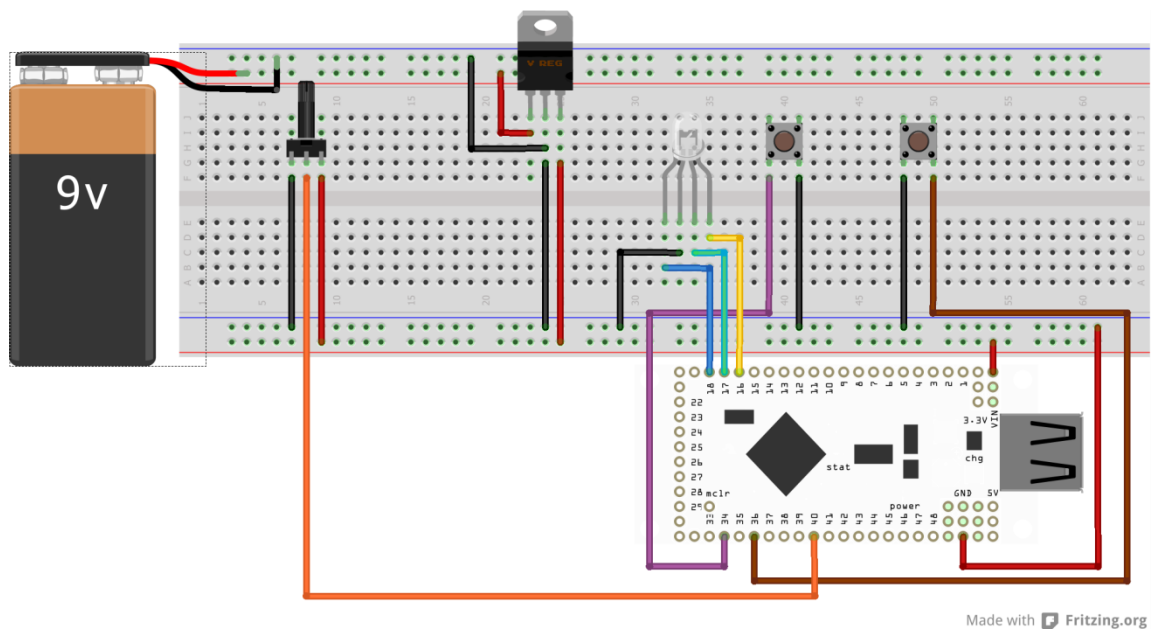


Figura 13. Esquema de conexiones para probar este experimento.

Parte del código necesario para realizar esta prueba se muestra en la Figura 14, donde con la función `setText()`, se va mostrando en pantalla los valores que va tomando el pin de la tarjeta. A su vez, con la función `insertarFila()`, se almacenan estos valores en SQLite.

```

private void setText(final String str1, final String str2) {
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            mBtn1TextView.setText(str1);
            mBtn2TextView.setText(str2);
        }
    });
}

```

```

//Abrir la base de datos, se creará si no existe
abrirBasedatos();
//Insertar una fila o registro en la tabla "bomberos"
//si la inserción es correcta devolverá true
boolean resultado = insertarFila(bombero,pulso/2, temperatura);
if(resultado)
    Toast.makeText(getApplicationContext(),
        "Pulso 1 añadido correctamente", Toast.LENGTH_SHORT).show();
else
    Toast.makeText(getApplicationContext(),
        "No se ha podido guardar el pulso 1", Toast.LENGTH_SHORT).show();

```

Figura 14. Extracto de código de esta prueba.

En esta prueba, hubo dos contratiempos, el primero es que la tarjeta IOIO viene perforada y los cables con los que se contaba quedaban sueltos en los agujeros, por lo que había un comportamiento inestable, para lo que se optó por soldar conectores a dicha placa. En la Figura 15 se aprecia una placa sin conectores y la placa luego del proceso de soldado.

Luego, la aplicación parecía funcionar en condiciones adecuadas, pero al momento de intentar acceder a la base de datos de la aplicación para ver si los datos almacenados eran los correctos, el teléfono no mostraba el directorio en el que se guardan las bases de datos. Para ver estos datos el teléfono debe estar como *root*, con esto se tendrá acceso a todas las carpetas del equipo Android. Para acceder a las bases de datos debemos seguir la ruta “data/data/<nombre del paquete>/databases”. Para ver estas bases de datos en el computador existe un programa llamado SQLiteAdmin, con el que podemos modificar los archivos de SQLite.

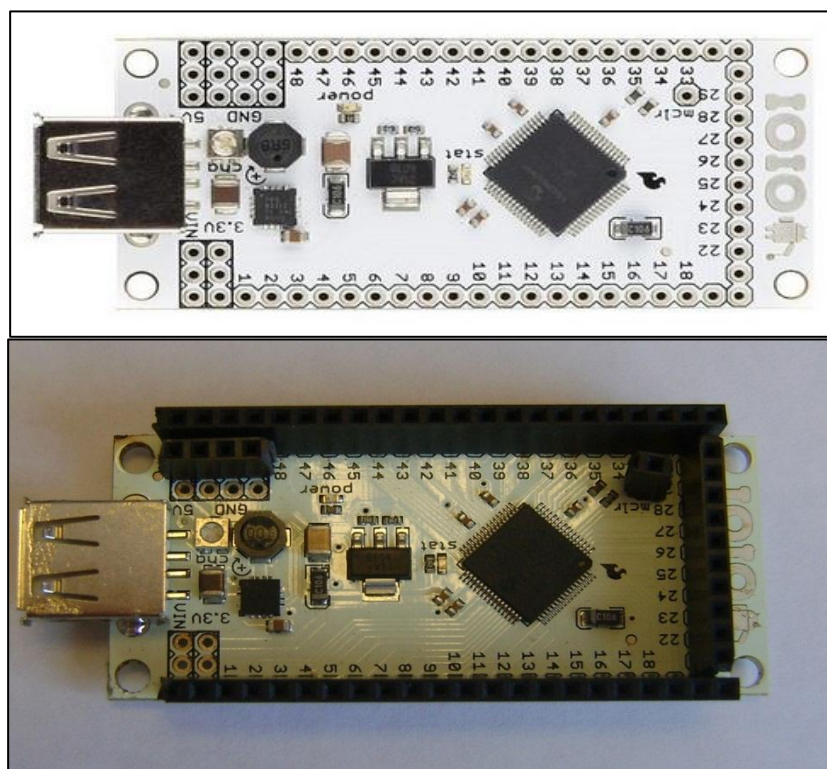


Figura 15. Placa IOIO antes y después del soldado de conectores.

#### 5.3.4 Prueba de concepto N°4 “Comunicar dos módulos Xbee de forma transparente inalámbricamente”

De modo transparente se refiere a que éste está destinado principalmente a la comunicación punto a punto, donde no es necesario ningún tipo de control. También se usa para reemplazar alguna conexión serial por cable, ya que es la configuración más sencilla posible y no requiere una mayor configuración. Es la forma en la que se probó un “chat”, en el que utilizando el programa brindado por el fabricante de los módulos Xbee, X-CTU, podemos enviar y recibir mensajes sin preocuparnos de como sucede esto.

Los problemas ocurren cuando los módulos no se conectan y existen diferentes versiones de cómo configurarlos para que se logre la comunicación. Configuraciones tales como que los dos módulos deben ser *routers*, o uno *router* y el otro coordinador, son algunas de las variantes que complican el proceso. Luego debemos verificar el *PAN ID*, identificador de la red, de ambos que debe ser igual, de la misma forma que el *Operating Channel*, luego el *Serial Number High* y el *Destination Address High* es el mismo para ambos (13A200), en el caso del *Serial Number Low*, está indicado en la parte posterior

de cada Xbee y es único. Este valor del *Serial Number Low* va en el *Destination Address Low* del otro Xbee y viceversa. En la Figura 16 vemos la configuración de los Xbee para lograr una comunicación en modo transparente.

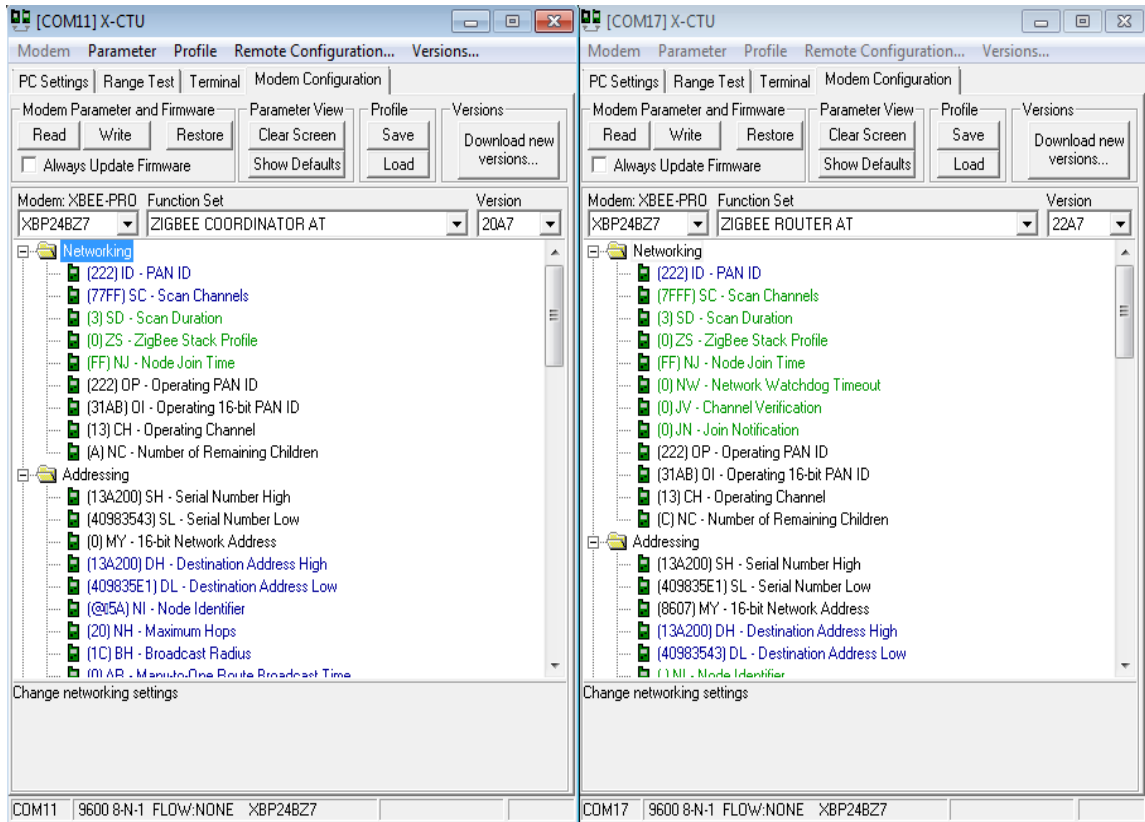


Figura 16. Configuración de ambos Xbee para “chat”.

En la Figura 17 se ve el terminal de ambos módulos en el que en letras azules es lo que se envía y en letras color rojo es lo recibido.

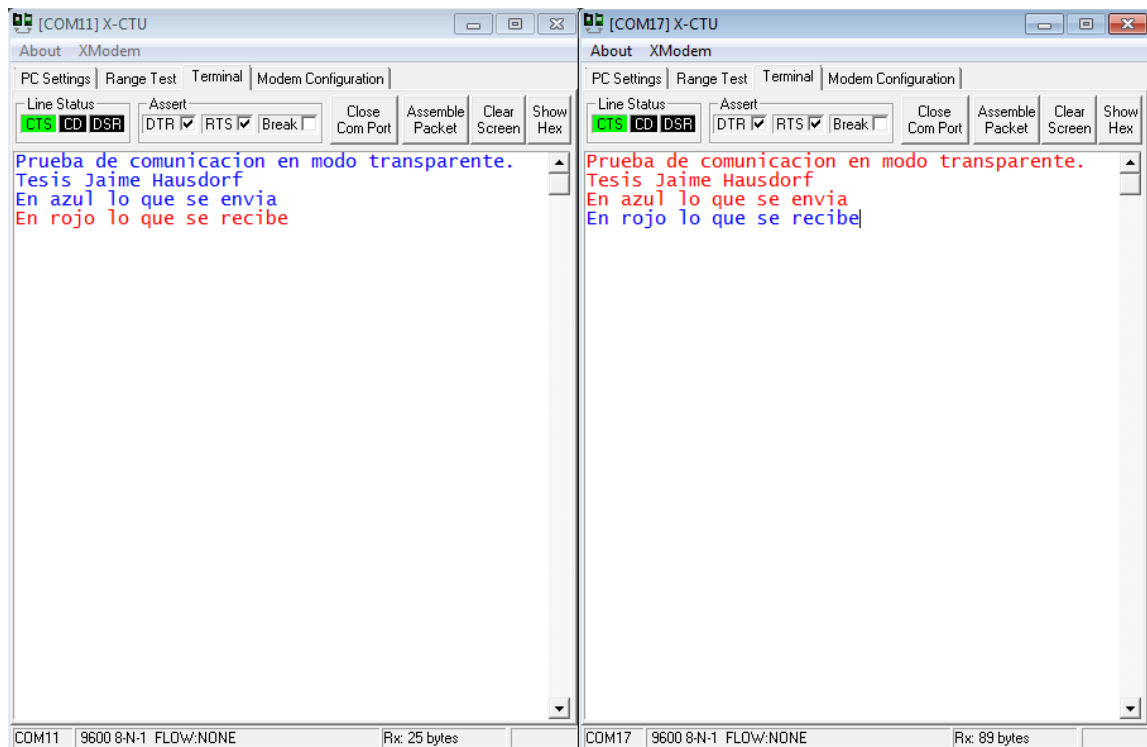


Figura 17. Terminal de los módulos logrando el “chat”

### 5.3.5 Prueba de concepto N°5 “Comunicar dos módulos Xbee vía API inalámbricamente”

Los componentes que deben activarse para que el módulo Xbee transmita una señal analógica, en el X-CTU, en la sección *I/O*, en el D1, por ejemplo, se escoge la opción 3 (ADC). Además en la parte de *I/O Sampling* el *IR* se pone en 3E8 (que son 1000 milisegundos en hexadecimal). Para el transmisor hay que conectar el voltaje de referencia en RES del Xbee, la tierra, energizarlo con 3.3 Volts y en D1 poner el cable con el dato. Así en el receptor llegará una trama, que se desglosa y que hasta ahora los dos penúltimos partes son los datos. Una alternativa es hacer cable virtual (lo que sale en el pin X del emisor, también sale del pin X del receptor), pero esto es válido sólo para conexiones punto a punto, lo que limita la cantidad de “bomberos en acción”. Por lo que esto último no es útil en nuestro caso.

En el módulo receptor podemos ver la trama recibida, y ésta es ejemplificada en la Figura 18.



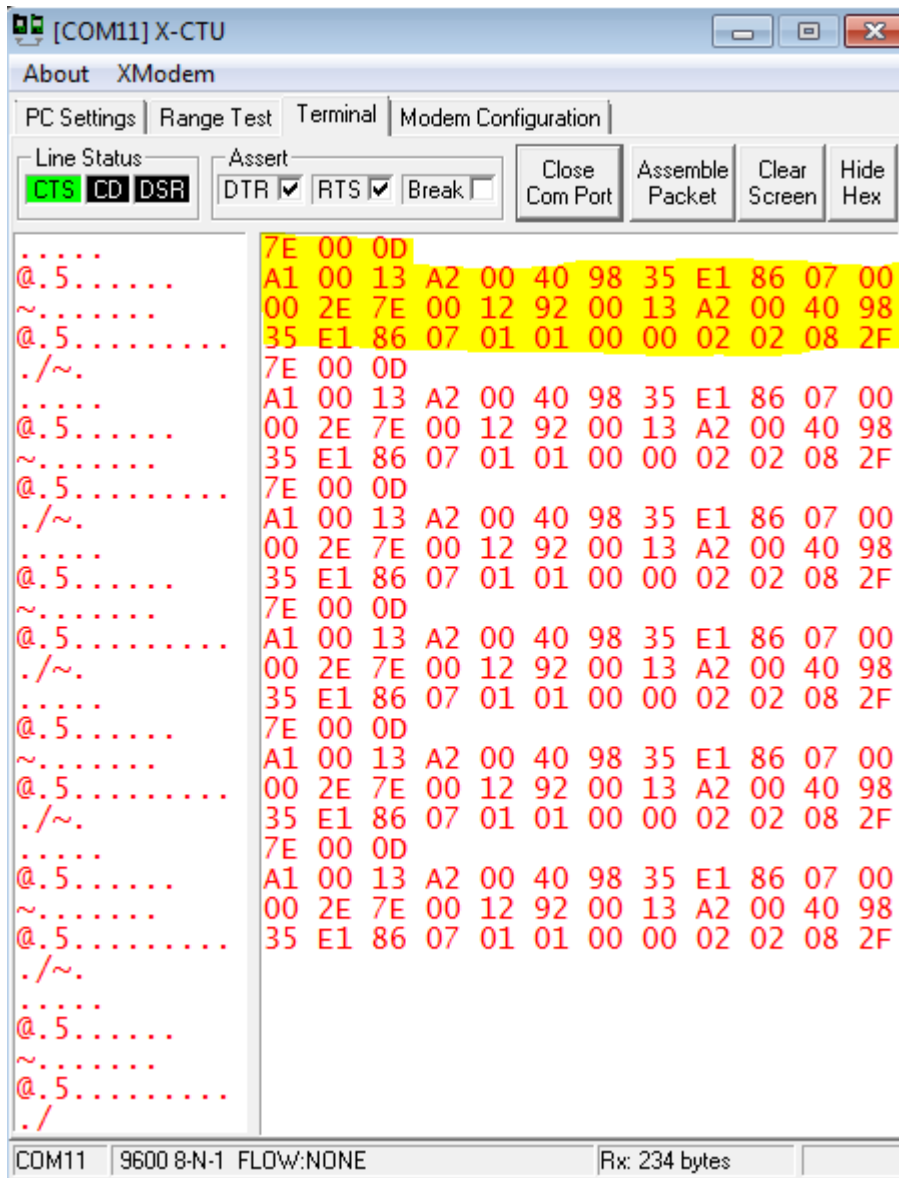


Figura 18. Trama en el módulo receptor.

El formato de estos bloques es el siguiente:

- El primer byte es de referencia para el comienzo de la trama (0x7E)
- El segundo y tercer byte (0x0026 = 38 en decimal) indican el número de bytes de la trama de datos.
- Los siguientes 38 bytes, corresponde a la trama de datos.
- El último byte (0xD7) es el valor de *Checksum* para la comprobación de errores.

### 5.3.6 Prueba de concepto N°6 “Enviar datos de sensores desde el Xbee receptor al teléfono móvil”

En el caso de este experimento, no había transferencia de datos inalámbricamente pero había que lograr manejar la entrada serial de información por un pin de la tarjeta IOIO, dato que era enviado por el pin de transmisión del Xbee receptor.

Una vez configurado el módulo Xbee para que envíe los datos de los sensores conectados a él cada cierto tiempo, se debía lograr ver en el celular los valores de estos sensores, cuestión que involucra una sincronización entre la IOIO, el Xbee y el teléfono. Para lo cual hay que manejar pausas y tiempos de ejecución de la aplicación Android. En la Figura 19, se muestra la conexión realizada para ejecutar esta prueba de concepto.

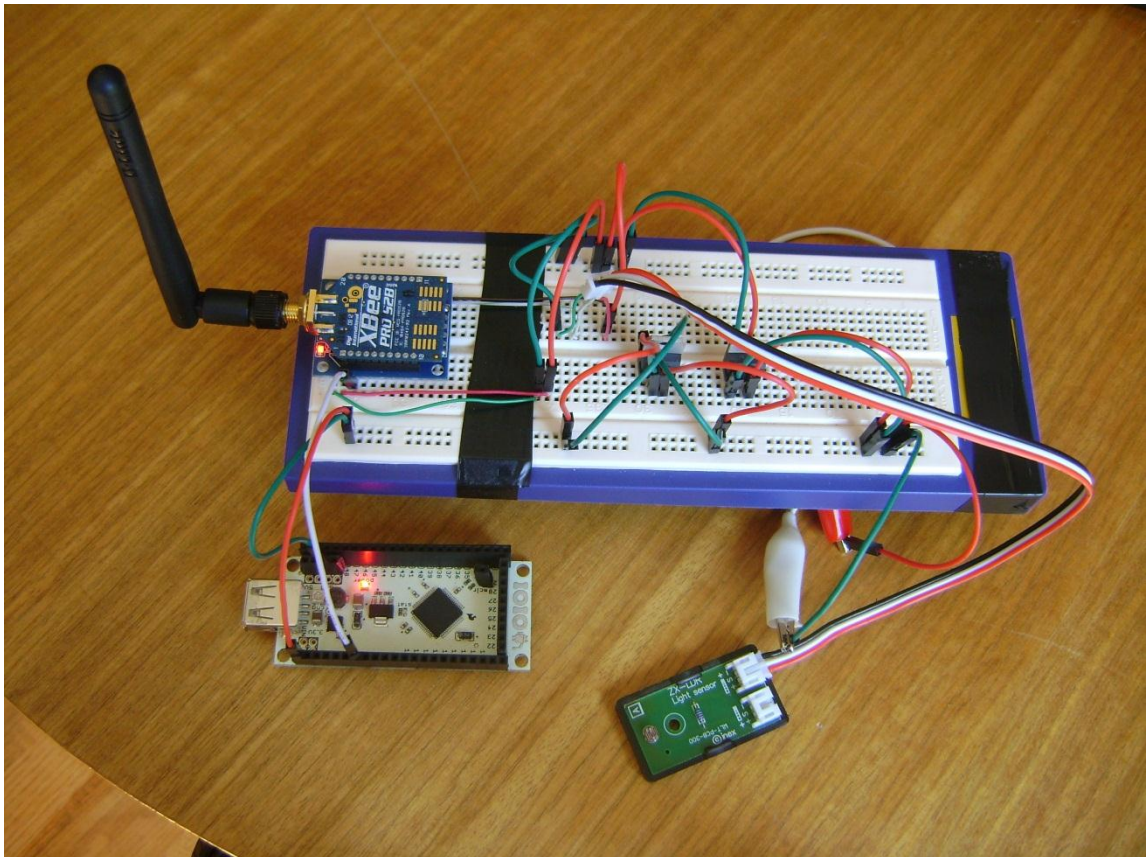


Figura 19. Conexión entre la IOIO, el Xbee receptor y el teléfono.

El código fuente de esta prueba es exhibido en la Figura 20, donde se ve la utilización de un pin configurado para que lea de forma serial, forma por la que es enviada la información desde el módulo.



```

private final int rxPin = 4;
private final int txPin = 5;

private Uart uart;
private InputStream in;

```

```

class IOIOThread extends AbstractIOActivity.IOIOThread {
public void setup() throws ConnectionLostException {
    try {
        //uart = ioio._openUart(rxPin, txPin, baud, parity, stopBits);
        uart = ioio._openUart(rxPin, txPin, 9600, Parity.NONE, StopBits.ONE);
    } catch (ConnectionLostException e) {
        throw e;
    }
}
char[] HEX_CHARS = "0123456789abcdef".toCharArray();
@Override
public void loop() throws ConnectionLostException {
    try{
        in = uart.getInputStream();

        Thread.sleep(100);
        int availableBytes = in.available();
        if (availableBytes > 0) {
            char[] chars = new char[2 * availableBytes];
            for (int i = 0; i < availableBytes; ++i) {
                byte bb = (byte) in.read();
                chars[2 * i] = HEX_CHARS[(bb & 0xF0) >>> 4]; //hace un AND entre el byte bb y 0xF0, y el resultado se mueve 4 espacios,
                chars[2 * i + 1] = HEX_CHARS[bb & 0x0F]; //cont. si el and resulta en 11101000, el resultado final será 1110
            }

            String mensaje = new String(chars);
            setText(mensaje);

            Thread.sleep(700);

```

Figura 20. Parte de código fuente del experimento N°6.

En la Figura 21, se aprecia el esquema de conexiones necesarias para el correcto funcionamiento de esta prueba de concepto.

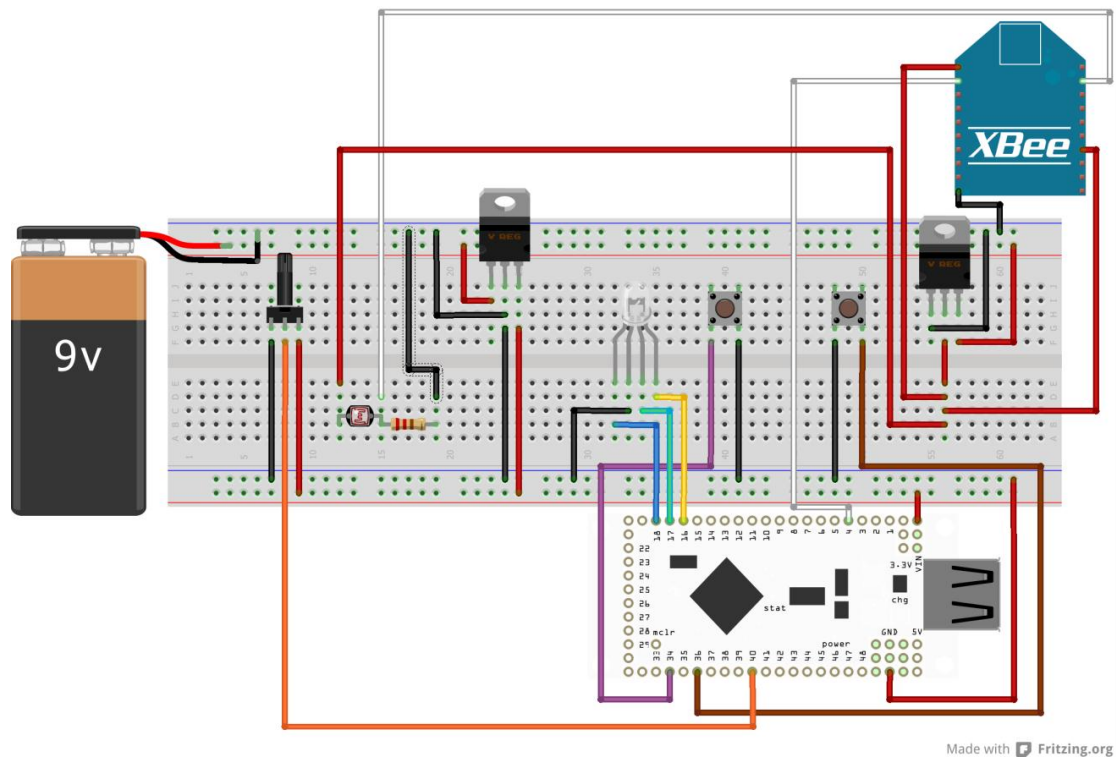


Figura 21. Esquema de cableado de esta prueba de concepto.

### 5.3.7 Prueba de concepto N°7 “Conectar sensor de luz al Xbee emisor, transmitir data y almacenar ésta en una base de datos SQLite”

Se comprueba el proceso completo de la solución propuesta, desde el módulo Xbee emisor hasta que los datos son almacenados en el teléfono inteligente en una base de datos SQLite.

El proceso se inicia con la captura de los datos del sensor de luz que está conectado al Xbee emisor, esta conexión es realizada como se explicó anteriormente y es mostrada en la Figura 22.

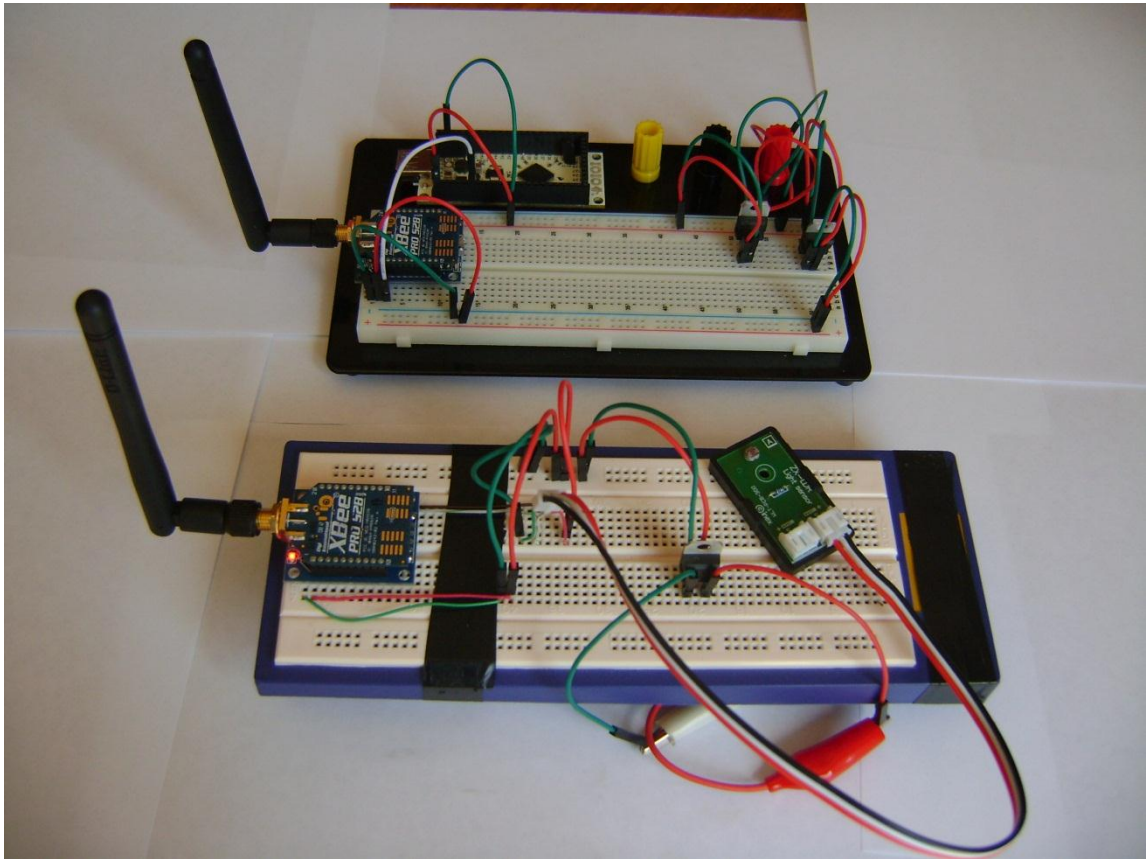


Figura 22. Conexión del sensor de luz al módulo Xbee.

Luego, por otro lado, en el parte receptora, tenemos un Xbee energizado y el pin de transmisión conectado al pin 4 de la placa IOIO, que es definida en el *software* como de entrada serial. Así el dato es captado inalámbricamente por el Xbee receptor y transmitido hacia la tarjeta IOIO, esta a su vez es conectada vía USB al teléfono móvil y éste ejecutando una aplicación, es capaz de leer estos datos y almacenarlos en la base de datos de la aplicación.

En la Figura 23, finalmente, vemos el esquema resultante de este experimento que sirve de muestra de lo que será la solución final.

Finalmente, luego de ejecutadas las siete pruebas anteriores, se llega a una solución que contempla el envío de datos de sensores conectados a un módulo Xbee, que transmite por radiofrecuencia hacia otro Xbee configurado como receptor, para que éste traspase la información hacia el teléfono inteligente a través de la placa IOIO. Los datos son mostrados y almacenados en el *smartphone* con lo que finaliza el trazado del dato.

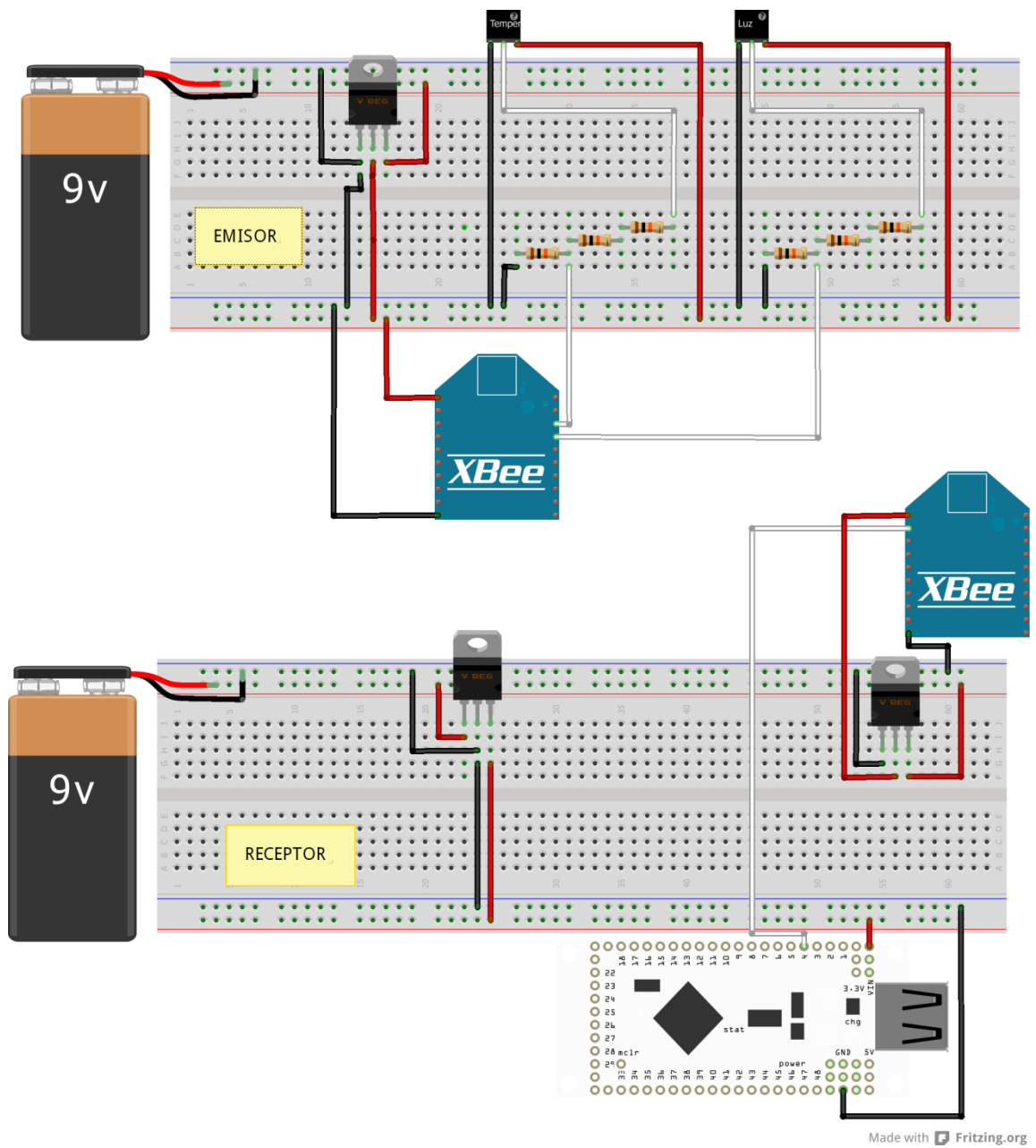


Figura 23. Esquema del cableado de la prueba de concepto.

## 5.4 Diseño

### 5.4.1 Casos de uso.

En la Figura 24, se aprecia el diagrama de casos de uso general.

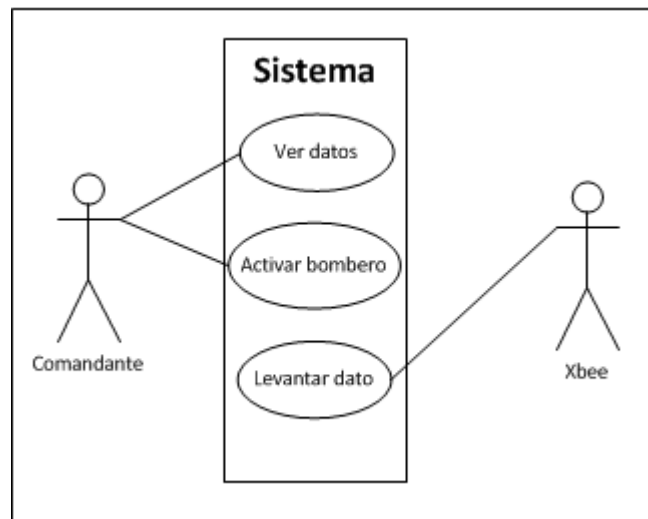


Figura 24. Diagrama general de casos de uso.

#### 5.4.1.1 Actor Comandante

##### 5.4.1.1.1 Caso de uso real

Caso de uso: Activar bombero

Actores: Comandante

Propósito: El comandante activa un bombero para capturar sus datos.

Resumen: El comandante, en la pantalla inicial, activa un bombero al emparejarlo con un Xbee dando clic en Enlazar. Para comenzar a capturar los datos de los bomberos activos, da clic en Siguiente.

Referencias cruzadas: Funcionalidades: F5

En la Figura 25, se despliega el diagrama de pantalla y en la Tabla 4, se muestra el curso normal de los eventos para el caso de uso Activar bombero.



Figura 25. Diagrama de pantalla CU: Activar bombero

Tabla 4. Curso normal de los eventos CU: Activar bombero.

| Acción del actor   | Respuesta del sistema   |
|--|---|
| 1. El comandante hace clic en algún Xbee de la lista desplegable en 1 e ingresa un nombre de bombero en 2, luego da clic en 3 (Enlazar). |   |
|  | 2. El sistema enlaza el Xbee con el bombero, enviando un mensaje para confirmar la acción.          |
| 3. El actor da clic en 4, Siguiente, para comenzar la captura de datos.  |   |
|  | 4. El sistema cambia de pantalla y comienza a recibir los datos y almacenarlos en la base de datos. |

#### 5.4.1.1.2 Diagrama de colaboración

A continuación, se presenta la Figura 26, mostrando el diagrama de colaboración para este caso de uso.

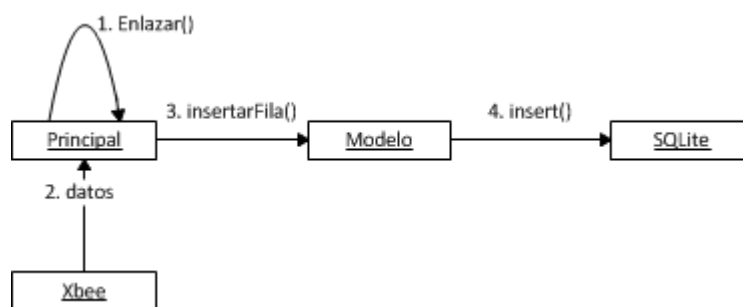


Figura 26. Diagrama de colaboración de Activar bombero.

Los elementos que aparecen en el diagrama de colaboración son clases y componentes del sistema. Principal y Modelo son clases y están en el diagrama de clases, por otro lado SQLite y Xbee son componentes de la solución y están en la Figura 1.

#### 5.4.1.1.3 Caso de uso real

Caso de uso: Ver datos

Actores: Comandante

Propósito: Ver detalle de los últimos cinco datos enviados por un bombero.

Resumen: El comandante puede acceder a los últimos cinco datos de un bombero haciendo clic sobre su nombre, ya sea porque éste cambia de color para alertarlo de la ocurrencia de algún evento o simplemente para monitorear su comportamiento reciente.

Referencias cruzadas: Funcionalidades: F2, F3, F4

Precondición: Debe haber al menos un módulo Xbee emparejado con un bombero

En la siguiente figura, Figura 27, se muestra el diagrama de pantalla y en la Tabla 5, se exhibe el curso normal de los eventos para el caso de uso Ver datos.



Figura 27. Diagrama de pantalla CU: Ver datos

Tabla 5. Curso normal de los eventos CU: Ver datos

| Acción del actor  | Respuesta del sistema  |
|---|--|
| 1. El comandante hace clic en algún nombre de bombero de la lista desplegada, en 1. |  |
|   | 2. El sistema cambia de pantalla para mostrar los últimos cinco datos del bombero seleccionado |

#### 5.4.1.1.4 Diagrama de colaboración

A continuación, se presenta la Figura 28, mostrando el diagrama de colaboración para este caso de uso.

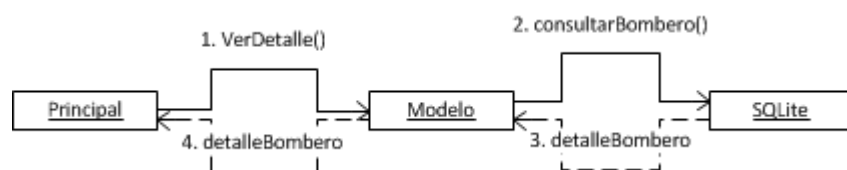


Figura 28. Diagrama de colaboración de Ver Datos.



### 5.4.2 Diagrama de clases

Enseguida se muestra el diagrama de clases de la solución propuesta, esto en la Figura 29. Este diagrama de clases es el resultado de las pruebas y experimentos realizados en donde se trata de aplicar el patrón MVC (modelo, vista, controlador).

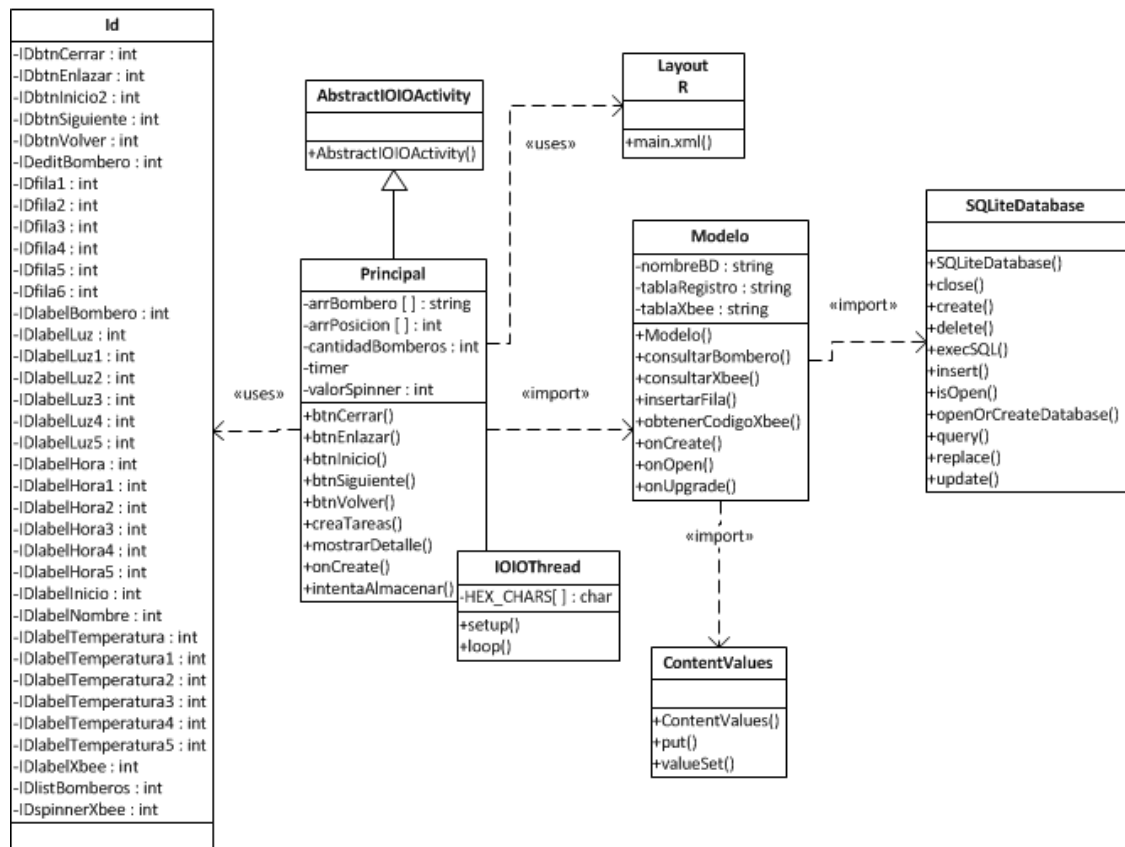


Figura 29. Diagrama de clases de la solución propuesta.

### 5.4.3 Diagrama de componentes

En la Figura 30, que se muestra a continuación se aprecia el diagrama de componentes de la solución propuesta.

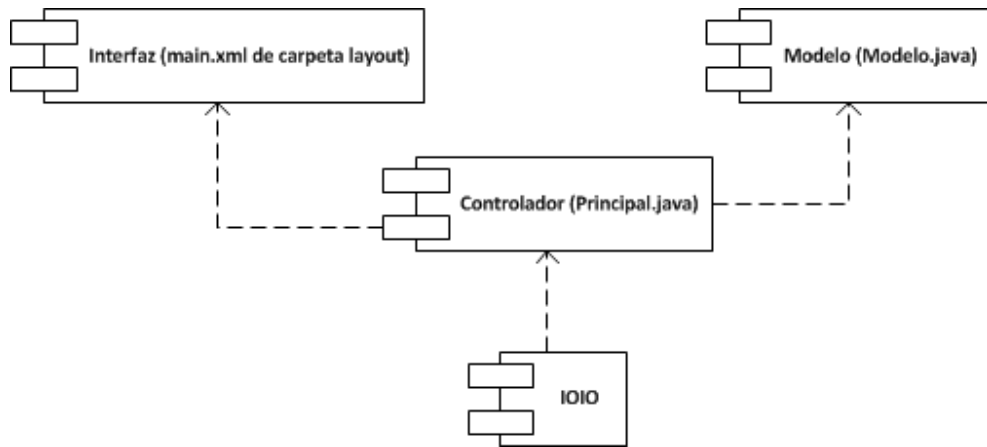


Figura 30. Diagrama de componentes.

#### 5.4.4 Diagrama general de pantallas.

A continuación se muestra un esquema de cómo se presentan las pantallas al usuario. Así, tenemos que en la Figura 31, se aprecia la pantalla inicial en donde se debe seleccionar el módulo Xbee en 1, ingresar el nombre del bombero en 2 y para enlazarlos, se debe presionar 3. Para comenzar la captura de datos, presionar 4.



Figura 31. Diagrama de la pantalla inicial.

Esto llevará a la siguiente pantalla, que está representada por la Figura 32, en donde se muestran, en 1, los nombres de los bomberos enlazados con algún módulo Xbee y que al hacer clic en uno de ellos, pasamos al detalle en la siguiente pantalla.

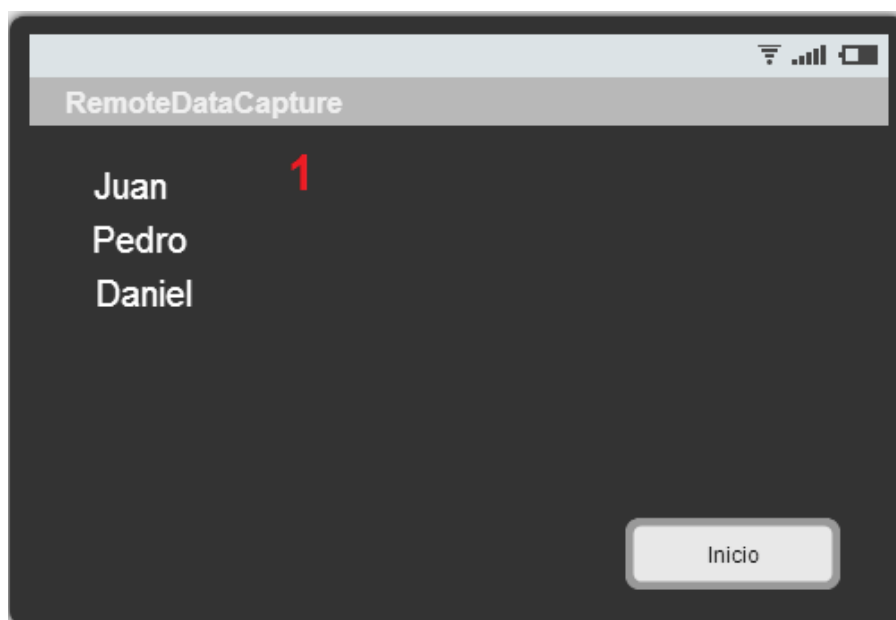


Figura 32. Diagrama de la pantalla con la lista de bomberos

En la tercera pantalla, que se ve en la Figura 33, se muestra el detalle del bombero seleccionado de las últimas 5 mediciones recibidas y almacenadas.

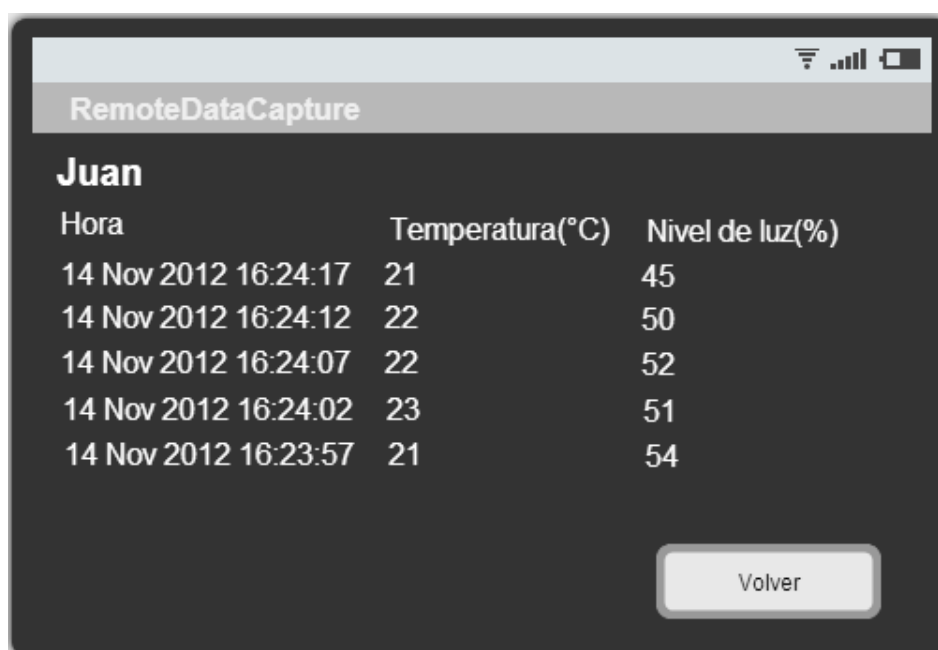


Figura 33. Diagrama de pantalla del detalle de un bombero

## 5.5 Solución final

En esta sección se explica cómo se procede para esta solución desarrollada e instalada en el celular (o *tablet*) con sistema operativo Android.

Como cualquier desarrollo informático, éste presenta una serie de pantallas que el usuario debe leer y responder a las preguntas planteadas y según sean sus respuestas se presentan nuevas pantallas y nuevas opciones para finalmente dejar operativa la aplicación.

### 5.5.1 Procedimiento.

Buscar la aplicación instalada en el celular o *tablet* con sistema operativo Android, denominada RemoteDataCapture y que se muestra en la Figura 34.

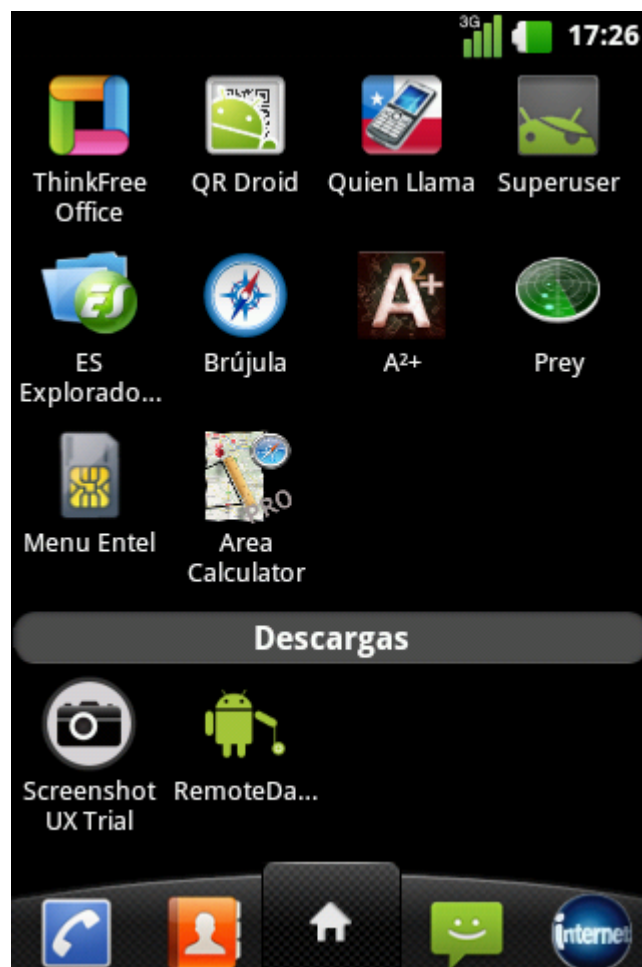


Figura 34. Aplicación en el *smartphone*.

Una vez seleccionado la opción se debe asociar el número del dispositivo transmisor, que se muestra en una lista desplegable (ver Figura 35), a la persona que lo portará, para ello se edita e ingresa el nombre. Luego, se presiona el botón “Enlazar” y el dispositivo despliega un mensaje de enlace realizado. Esto se repite para cada dispositivo transmisor que se disponga.



Figura 35. Lista desplegable de módulos Xbee disponibles.

En este momento se recomienda activar el transmisor para chequear que éste se sincroniza con el receptor, con el celular (*tablet*) que almacenará la información transmitida por los equipos.

Finalmente, una vez ingresado los nombres de las personas, presionar “Siguiete” y el dispositivo inicia la captura de información de los equipos transmisores, mostrando una lista con los nombres de las personas enlazadas, como se aprecia en la Figura 36.

Durante la captura, la aplicación informa como se desarrolla el proceso de almacenamiento en la base de datos SQLite desplegando el mensaje “Añadido correctamente<data>”. De existir problemas en la grabación de los datos indica el mensaje “No se ha podido guardar en la base de datos <data>”, donde <data> es el nombre de la persona que lo porta.

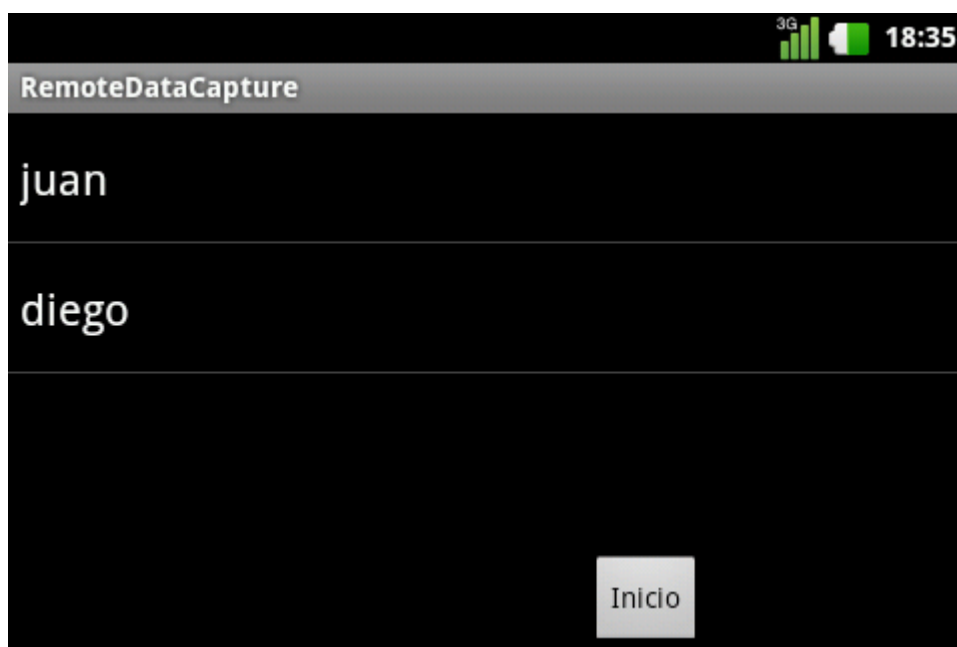


Figura 36. Lista con las personas enlazadas.

Para visualizar los datos de una persona determinada, se debe seleccionar de la lista desplegada, pudiendo ver las últimas 5 lecturas realizadas por el dispositivo en una nueva pantalla, que se muestra en la Figura 37.

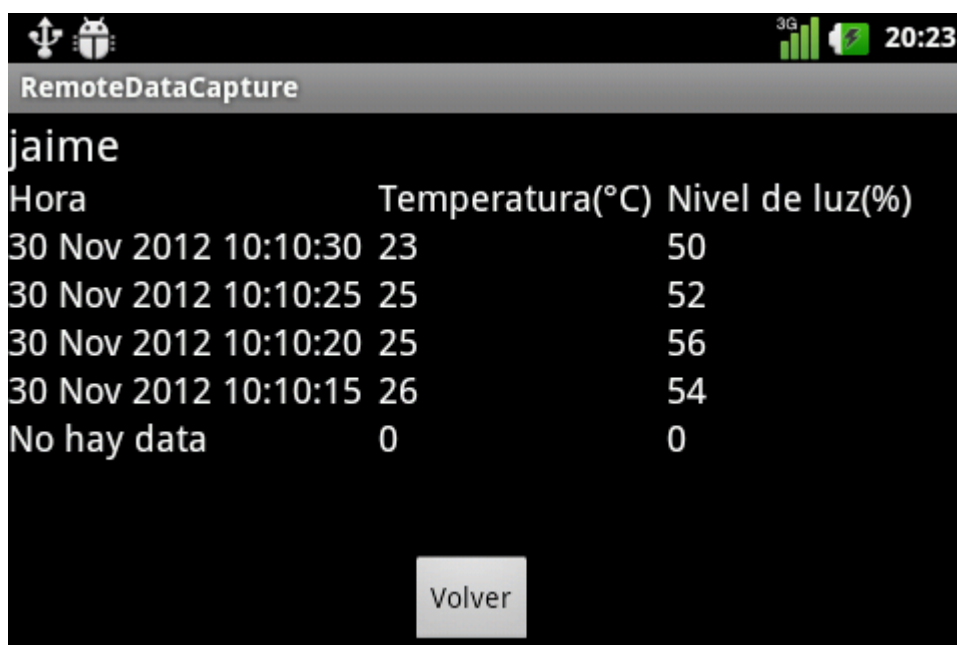


Figura 37. Datos de una persona seleccionada de la lista.

Si los datos recibidos pasan un umbral, se muestra un mensaje de alerta indicando el nombre de la persona en problemas, para proceder a tomar las medidas correspondientes para que la integridad del personal no se vea afectada.

## 5.5.2 Código de la aplicación.

En este apartado se mostrarán los códigos fuentes del desarrollo, en particular de las opciones: Botón Enlazar – en la Figura 38, Botón Siguiente – en la Figura 39 y Seleccionar un bombero de un *ListBox* – en la Figura 40. Todos estos códigos son puestos aquí porque son considerados como las principales partes de líneas de código fuente desarrollados.

Botón Enlazar y Botón Siguiente, están nombrados y utilizados en el caso de uso real Activar Bombero.

```
public void btnEnlazar(View view){
    arrPosicion[cantidadBomberos] = valorSpinner+1;
    arrBombero[cantidadBomberos] = editBombero.getText().toString();

    Toast.makeText(getApplicationContext(),
        "Enlazado correctamente", Toast.LENGTH_SHORT).show();

    cantidadBomberos=cantidadBomberos+1; //para saber cuantos bomberos habran en la siguiente pantalla
}
```

Figura 38. Código de botón Enlazar.

```
public void btnSiguiente(View view){
    //de inicio a pantalla 2
    iniciar=true;
    //lleno la lista con los nombres de los bomberos ingresados en 1
    String[] arrBomberoAjustado = new String[cantidadBomberos];
    for (int j=0;j<cantidadBomberos;j++){
        arrBomberoAjustado[j] = arrBombero[j];
    }
    ArrayAdapter<String> adaptador;
    adaptador = new ArrayAdapter<String>(getApplicationContext(),android.R.layout.simple_list_item_1, arrBomberoAjustado);
    listBomberos.setAdapter(adaptador);

    //irse a la otra pantalla al hacer clic en algun item de la lista
    listBomberos.setOnItemClickListener(new OnItemClickListener() {
        public void onItemClick(AdapterView?> parent, View v, int posicion, long id) {
            String nombreBombero = listBomberos.getItemAtPosition(posicion).toString();
            mostrarDetalle(nombreBombero, posicion);
        }
    });

    //borro lo de inicio y cargo lo de pantalla 2
    btnCerrar.setVisibility(View.INVISIBLE);
    btnSiguiente.setVisibility(View.INVISIBLE);
    btnEnlazar.setVisibility(View.INVISIBLE);
    labelInicio.setVisibility(View.INVISIBLE);
    labelXbee.setVisibility(View.INVISIBLE);
    labelBombero.setVisibility(View.INVISIBLE);
    editBombero.setVisibility(View.INVISIBLE);
    spinner.setVisibility(View.INVISIBLE);

    btnInicio.setVisibility(View.VISIBLE);
    listBomberos.setVisibility(View.VISIBLE);
}
```

Figura 39. Código de botón Siguiente.

El código de Seleccionar del *ListBox*, está vinculado al caso de uso real Ver datos.

```

public void mostrarDetalle(String nombreBomb, int posicionXbee){
    btnInicio.setVisibility(View.INVISIBLE);
    listBomberos.setVisibility(View.INVISIBLE);

    btnVolver.setVisibility(View.VISIBLE);
    labelNombre.setVisibility(View.VISIBLE);
    labelTemperatura.setVisibility(View.VISIBLE);
    labelHora.setVisibility(View.VISIBLE);
    labelLuz.setVisibility(View.VISIBLE);
    btnVolver.setVisibility(View.VISIBLE);
    //15 textview con los "datos"
    labelTemperatura1.setVisibility(View.VISIBLE);
    labelHora1.setVisibility(View.VISIBLE);
    labelLuz1.setVisibility(View.VISIBLE);
    labelTemperatura2.setVisibility(View.VISIBLE);
    labelHora2.setVisibility(View.VISIBLE);
    labelLuz2.setVisibility(View.VISIBLE);
    labelTemperatura3.setVisibility(View.VISIBLE);
    labelHora3.setVisibility(View.VISIBLE);
    labelLuz3.setVisibility(View.VISIBLE);
    labelTemperatura4.setVisibility(View.VISIBLE);
    labelHora4.setVisibility(View.VISIBLE);
    labelLuz4.setVisibility(View.VISIBLE);
    labelTemperatura5.setVisibility(View.VISIBLE);
    labelHora5.setVisibility(View.VISIBLE);
    labelLuz5.setVisibility(View.VISIBLE);

    String[] fechas = new String [5];
    int[] temper = new int[5];
    int[] luces = new int[5];
    //debe llamar a un metodo de Modelo para ver los registros de este bombero
    SQLiteDatabase db = bbdd.getWritableDatabase();
    Cursor d = bbdd.consultarBombero(db,nombreBomb,arrPosicion[posicionXbee]);
    if (d.moveToLast()) {
        //ver si hay 5 registros, si no mostrar los que hayan
        int cont=0;
        do{
            //obtener los registros y ponerlos en los textView de la pantalla 3
            long milisegundos = d.getLong(0);
            Date fecha = new Date(milisegundos);
            String fechaLarga = fecha.toString(); //lo entrega en ingles y con gmt-3 y cosas raras
            String horaAtrasada = fechaLarga.substring(11,19);
            int horaMasTres= Integer.parseInt(horaAtrasada.substring(0, 2)) + 3;
            String horaCorrecta = horaMasTres+" "+fechaLarga.substring(13,19);
            fechas[cont] = fechaLarga.substring(8, 10) + " " + fechaLarga.substring(4, 7) +
                " " + fechaLarga.substring(30, 34) + " " + horaCorrecta;
            temper[cont] = d.getInt(1);
            luces[cont] = d.getInt(2);

            if(!d.moveToPrevious()){
                for(int r=cont; r<5; r++){
                    fechas[r] = "No hay data";
                    temper[r] = 0;
                    luces[r] = 0;
                }
                cont = 6;
            }
            cont++;
        }
        while(cont<5);
    }
    db.close();

    labelNombre.setText(nombreBomb);
    labelHora1.setText(fechas[0]);
    labelHora2.setText(fechas[1]);
    labelHora3.setText(fechas[2]);
    labelHora4.setText(fechas[3]);
    labelHora5.setText(fechas[4]);
    labelLuz1.setText(Integer.toString(luces[0]));
    labelLuz2.setText(Integer.toString(luces[1]));
    labelLuz3.setText(Integer.toString(luces[2]));
    labelLuz4.setText(Integer.toString(luces[3]));
    labelLuz5.setText(Integer.toString(luces[4]));
    labelTemperatura1.setText(Integer.toString(temper[0]));
    labelTemperatura2.setText(Integer.toString(temper[1]));
    labelTemperatura3.setText(Integer.toString(temper[2]));
    labelTemperatura4.setText(Integer.toString(temper[3]));
    labelTemperatura5.setText(Integer.toString(temper[4]));
}

```

Figura 40. Código de seleccionar del *ListBox*.



### 5.5.3 Esquema del cableado final.

A continuación se presenta el esquema del cableado de la solución final obtenida, esto se aprecia en la Figura 41.

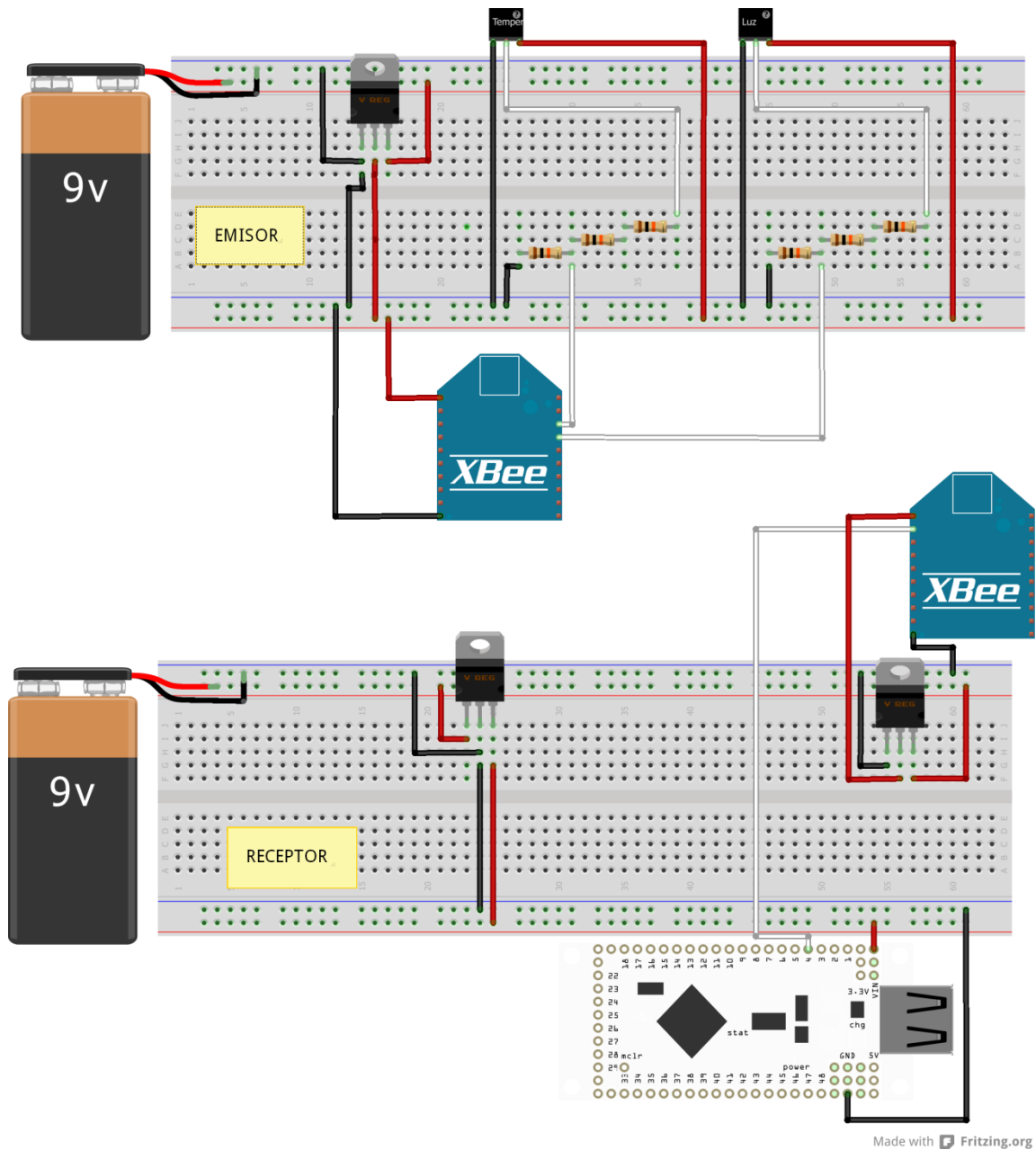


Figura 41. Esquema del cableado de la solución propuesta.

### 5.5.4 Foto del sistema real.

#### 5.5.4.1 Módulo emisor.

En la Figura 42, se muestra una fotografía del módulo emisor para esta solución, con un sensor de luz y uno de temperatura.

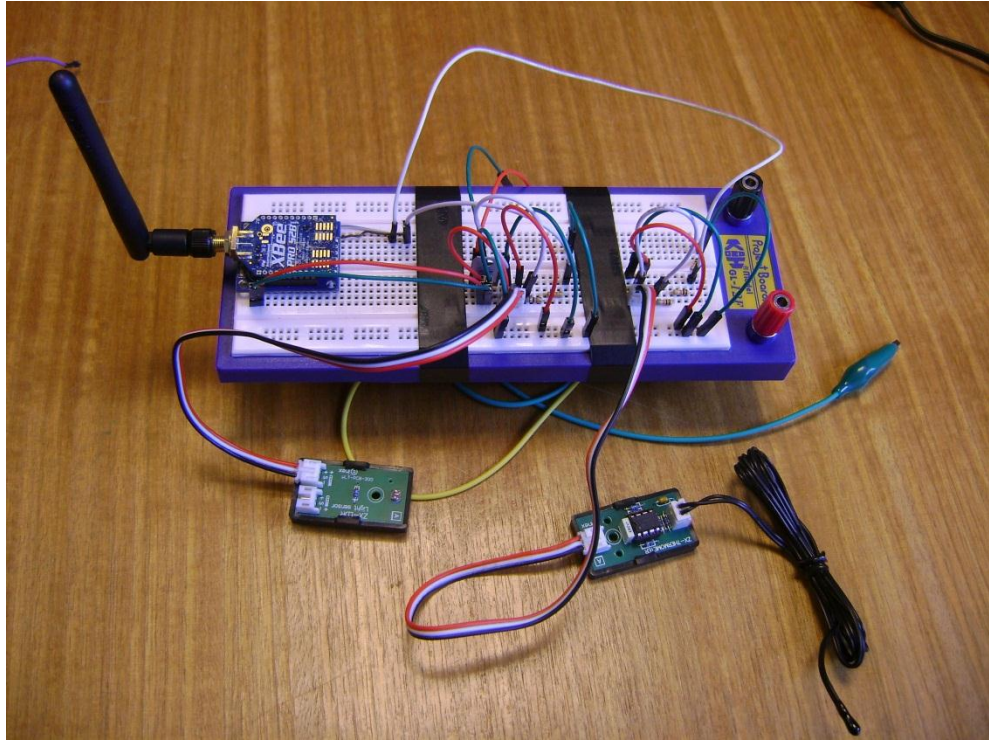


Figura 42. Módulo emisor.

#### 5.5.4.2 Módulo receptor.

En la Figura 43, se aprecia el módulo receptor con sus dos componentes, un dispositivo Xbee y una placa IOIO.

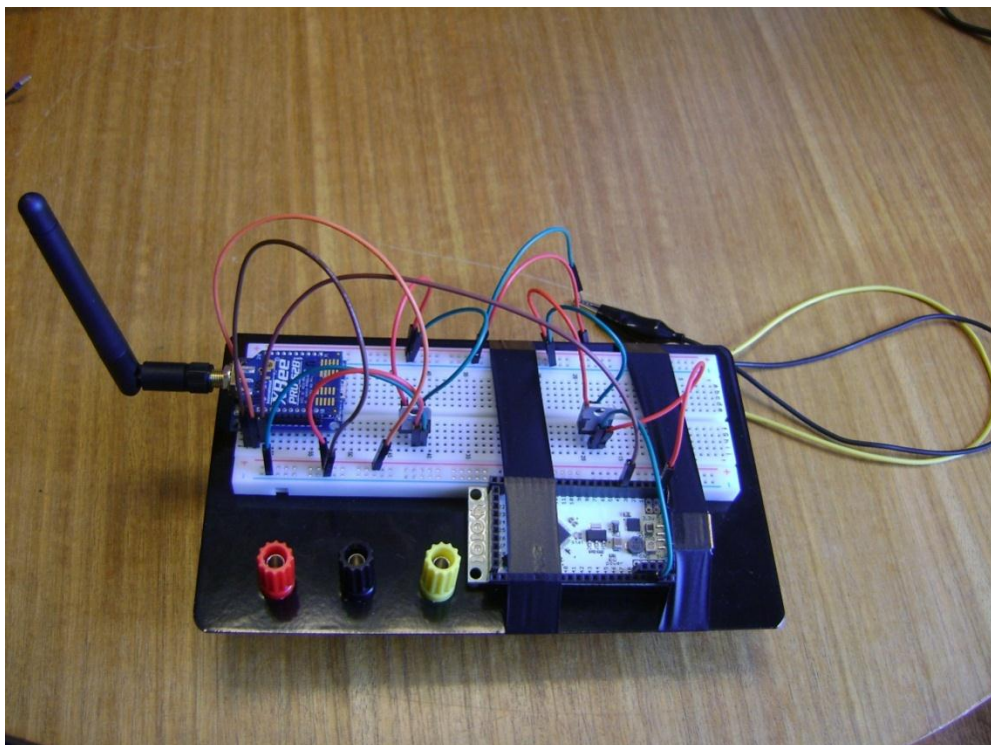


Figura 43. Módulo receptor.



#### 5.5.4.3 Solución completa.

Finalmente, en la Figura 44, se ve el sistema completo, el módulo emisor, receptor y el teléfono inteligente que soporta la aplicación.

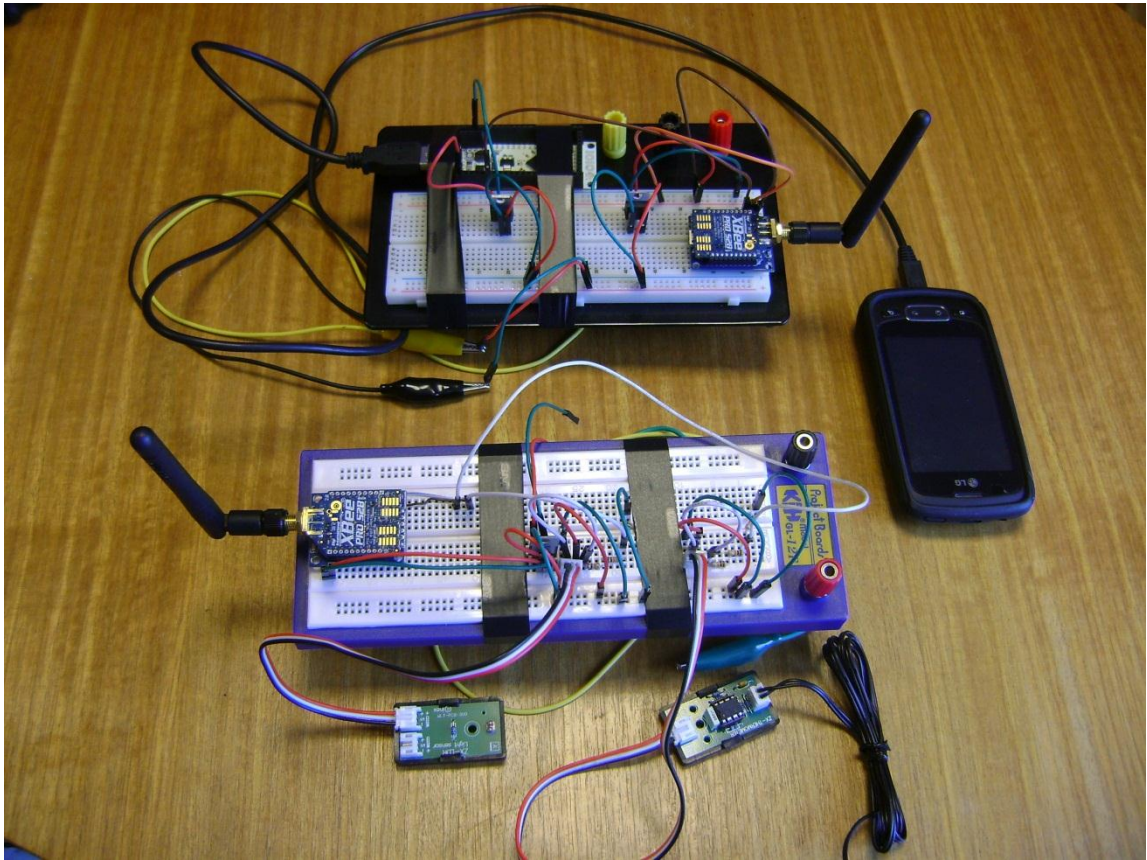


Figura 44. Sistema completo para la solución planteada.

#### 5.5.4.4 Pruebas Realizadas.

Armado todos los componentes de hardware, configurado los Xbee e instalado el software en el receptor se procedió a realizar las pruebas definitivas.

Primero, a una distancia de 3 metros entre el transmisor y el receptor para verificar los niveles de luz y de temperatura. Para el caso de la intensidad de luz de acercó y alejó una linterna led de gran potencia lumínica y para la temperatura se introdujo el terminal en agua caliente y en una cubeta de hielo.

Segundo, se alejó el transmisor a 800 metros aproximadamente, del receptor y se midió la luminosidad ambiente y se cubrió totalmente el sensor de luz, para tener luminosidad cero. Para el caso de la temperatura se midió la temperatura del lugar a 1 metro del suelo y en contacto con el cemento.

Con estas pruebas se verifica que el sistema cumple con el objetivo de levantar un dato en una zona no iluminada (WiFi) y que el dato puede visualizarse con la aplicación desarrollada para celulares con sistema operativo Android.

#### 5.5.4.5 Recomendaciones.

Al trabajar con módulos Xbee, una de las cosas probables que ocurra es que el programa X-CTU no lo reconozca y se dé por “muerto” el módulo, ya sea por sobre voltaje o por alguna otra mala conexión, pero existe una serie de pasos, que a continuación se enumeran, para poder “revivir” los Xbee.

1) En la pestaña “PC Settings”, setear lo siguiente:

Baud rate = 115200

Flow Control = None

Data Bits = 8

Parity = None

Stop Bits = 1

2) En la pestaña “Modem Configuration”, seleccionar el modem y *function set* apropiados. (Para testear el dispositivo, se debería seleccionar “XBP24-ZB” y la *function set* “ZigBee Coordinator API”.)

3) Cambiar a la pestaña “Terminal”. Desmarcar “RST” y marcar “Break” en el área “Assert”. (“DTR” debería estar marcada.)

4) Presionar reset en el dispositivo. (Si no tiene el botón reset, se debe hacer un puente entre el pin 1 con el 5 o 6.)

5) Desmarcar “Break”.

6) Clic en la pestaña “Terminal de la ventana principal, y luego dar Enter. El dispositivo debería mostrar algo como esto:

-----

.EM250 Bootloader. v20 b09

.

.1. uploadebl

.2. run

.3. ebl info

.BL > .

-----

7) Presionar 1. Debería decir ".begin upload", y comenzar a imprimir varias "C"s.

8) Volver a la pestaña "Modem Configuration" y dar clic en "Write".

9) El *firmware* debería escribirse correctamente. Cambiar el "Baud Rate" a 9600 en la pestaña "PC Settings".

10) Volver a la pestaña "Modem Configuration" y presionar "Read".

Esto debería hacerlo funcionar nuevamente.

## **6 CONCLUSIONES**

### **6.1 Conclusiones**

El objetivo general de construir un sistema que permita levantar datos, a través de sensores y que estos se puedan recepcionar en un dispositivo móvil, en particular un celular con sistema operativo Android, está cumplido totalmente.

Del análisis de las tecnologías disponibles para móviles, y en particular para captar una señal externa, la placa desarrollo IOIO para Android, que recepciona la señal del Xbee receptor y la pone a disposición del celular a través del puerto USB, resulto ser la más apropiada.

El desarrollo del software que se instaló en dispositivo móvil, cuyo objetivo es leer el dato que entra al celular, procesar para separar los datos de la trama de transmisión, mostrar en la pantalla y almacenar en la base de datos SQLite, cumple el objetivo para el cual fue desarrollado.

En la etapa de validación del sistema desarrollado, se pudo verificar que la frecuencia de muestreo de cinco segundos, en un período de un minuto deja del 30 a 40 % de la transmisión sin lectura. Esto implica que puede usarse sin problemas en situaciones donde el muestreo entregue información completa cada 20 segundos.

La etapa de análisis de las tecnologías disponibles y la selección de éstas, presenta el inconveniente de usar tecnologías de reciente lanzamiento al mercado que carecen de apoyo técnico, y el apoyo que se encuentra en las redes sociales aún no está actualizado. En este contexto se explican muchos de los problemas de información técnica de los Xbee de largo alcance y de la interface IOIO que permite leer información del exterior con un celular con Android.

Se destaca lo interesante que resulta en lo profesional y personal trabajar en el área de la electrónica más dura. Desde la adquisición de la tecnología, pasando por su configuración y uso de software de control. Y más interesante aún, un lenguaje técnico nuevo que suman a la formación integral de un Ingeniero Civil Informático que en la actualidad y la tendencia está más asociado a la informática y las comunicaciones.

Esto último se traduce en la práctica, en aprender a usar software de configuración de los Xbee y de la interface IOIO. Además, del software de diseño para los esquemas de circuitos necesarios para el proyecto.

Resulta necesario establecer la diferencia entre desarrollos móviles que son *inside* y las aplicaciones que requieren conexión con el mundo exterior y con otras tecnologías. Esto último tiene un impacto importante en los tiempos de desarrollo de los proyectos asociados, dado que para las tecnologías usadas existe muy poca información y menos asesoría de parte de proveedores que limitan sus servicios a la venta de estos dispositivos.

Por varios aspectos antes indicados, se estima necesario para la complementación y actualización de los futuros ingenieros readecuar los planes y programa, para que éstos se hagan cargo de los nuevos desafíos que están planteando las tecnologías móviles. Para ello, se podría integrar un curso taller (práctico) de programación de móviles, considerando las aplicaciones sin relación con el exterior y aquellas que leen dispositivos externos e integran distintas tecnologías.

## **6.2 Trabajo futuro**

Encapsular los dispositivos de recepción y transmisión. Es decir, instalar en placas de desarrollo electrónicas y envolver los circuitos con un material resistente de calidad industrial, que lo proteja de distintos niveles de temperatura, del agua, de ambientes corrosivos y de golpes.

Para que este prototipo pueda ser comercializado y según sea las áreas donde se quiera usar deberá ser certificado por algún organismo competente que tiene como objetivo certificar el cumplimiento de una serie de normas, propias del área donde se quiera usar este dispositivo.

En particular, para integrarse a las labores vinculadas al quehacer forestal, debe ser certificado por Cesmec (Centro de estudios, medición y certificación de calidad). Y para las labores de bomberos en el control de siniestros urbanos, este debe ser certificado por Idiem (Instituto de investigaciones y ensayos de materiales).

Cabe indicar que se puede usar en otras áreas, que no requieran certificación para su uso, como sería el caso de necesitar levantar un dato en zonas no iluminadas de campos de gran extensión.

Dado que puede capturar datos en zonas sin conectividad, se pueden realizar los ajustes para el control GPS de una flota de embarcaciones pesqueras.



## 7 REFERENCIAS

- [Bar08] Barrera C., (2008). Medidas de seguridad para bomberos. Disponible en <http://seguridadindustrial.com.mx.dish14.net.ibizdns.com/revistajulio62-2008.pdf>. Consultado el 12 de julio de 2012.
- [Bue10] BuenasTareas.com., (2010). Prevención De Riesgos Laborales. Disponible en <http://www.buenastareas.com/ensayos/Prevencion-De-Riesgos-Laborales/513020.html>. Consultado el 15 de noviembre de 2012.
- [Cab07] Cabero G. & Maldonado D. (2007). Sqlite: Rápido, ágil, liviano y robusto. Disponible en <http://es.scribd.com/doc/52882068/SQLite>. Consultado el 27 de septiembre de 2012.
- [Def] Definicion.de (n.d). Definición de prevención de riesgos. Disponible en <http://definicion.de/prevencion-de-riesgos/>. Consultado el 15 de noviembre de 2012.
- [Flo12] Flosi S., (2012). comScoreReportsJuly 2012 U.S. Mobile SubscriberMarket Share. Disponible en [http://www.comscore.com/Press\\_Events/Press\\_Releases/2012/9/comScore\\_Reports\\_July\\_2012\\_U.S.\\_Mobile\\_Subscriber\\_Market\\_Share?utm\\_source=feedburner&utm\\_medium=feed&utm\\_campaign=Feed%3A+comscore+%28comScore+News%29](http://www.comscore.com/Press_Events/Press_Releases/2012/9/comScore_Reports_July_2012_U.S._Mobile_Subscriber_Market_Share?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+comscore+%28comScore+News%29). Consultado el 26 de septiembre de 2012.
- [Lib12] Libertad digital, (2012). Android, imparable: 59% de cuota de mercado en 'smartphones'. Disponible en <http://www.libertaddigital.com/internet/2012-05-26/android-imparable-59-de-cuota-de-mercado-en-smartphones-1276459645/>. Consultado el 13 de septiembre de 2012.
- [Oli] MCI Limitada - Olimex Chile, (n.d). IOIO para Android. Disponible en [http://www.olimex.cl/product\\_info.php?products\\_id=854](http://www.olimex.cl/product_info.php?products_id=854). Consultado el 3 de septiembre de 2012.
- [Oya08] Oyarce A., (2008). Guía del Usuario XBEE Series 1. Disponible en [http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia\\_Usuario.pdf](http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario.pdf). Consultado el 25 de septiembre de 2012.