Bizkaiko Foru Aldundia
Diputación Foral de Bizkaia
Berrikuntza eta Ekonomi Sustapen Saila
Departamento de Innovación y Promoción Económica

Programa Ekinberri 2007

SmartMotes

Nodos inalámbricos de redes de sensores con inteligencia semántica

D4.2 Documentación Harware del prototipo RFIDGlove

Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa
robotiker tecnalia

RESUMEN

En este documento se explican las partes más relevantes del código implementado para desarrollar el dispositivo RFIDGlove, así como los pasos dados para su implementación hardware.

Se comentan las diferentes decisiones que se han tomado, como el protocolo de comunicaciones que se utiliza, la cola implementada, los tipos de paquete que se envían entre las motas etc.

Se analiza por un lado las decisiones tomadas para desarrollar el programa del dispositivo RFIDGlove, y por otro la aplicación que se implementa en el ordenador al que se conecta la mota base y la comunicación entre ambas motas.

Tras esto, se presentan los pasos dados en el diseño, así como las incidencias ocurridas, para finalizar presentado los resultados mediante imágenes.

HISTORIAL DE CAMBIOS

Versión	Descripción	Autor	Fecha	Comentarios
V1.0	Primer borrador del documento	Leire Muguira	11/03/2008	
V1.1	Más información sobre el programa, y diagrama de clases añadido	Leire Muguira	01/04/2008	
V1.2	Imágenes añadidas	Leire Muguira	04/04/2008	
V1.3	Documentación del código añadido	Leire Muguira	14/05/2008	
V1.4	Documentación hardware	Asier Arruti	16/05/2008	
V1.5	Revisión y correcciones	Jonathan Ruiz de Garibay Asier Arruti	19/05/2008	
V1.6	Revisión y correcciones	Jonathan Ruiz de Garibay	19/05/2008	

TABLA DE CONTENIDOS

1 INTRODUCCIÓN

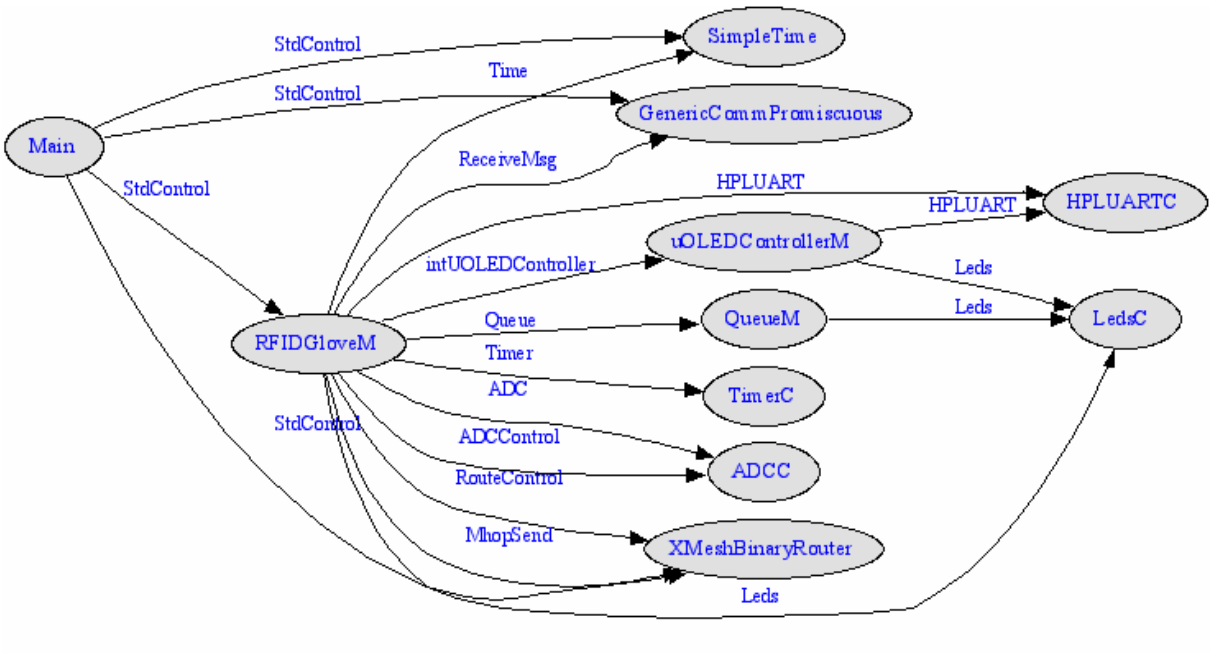
RFIDGlove es un proyecto en el cual un guante dotado de un lector RFID y de una mota MicaZ es capaz de comunicarse con otra mota base para que el sistema logístico conozca las actividades de manipulación de materiales que lleva a cabo el usuario y pueda optimizar el funcionamiento de la organización.

En este documento se explican las decisiones que se han tomado a la hora de desarrollar el código que se ha programado en las motas MicaZ y el código java necesario en el ordenador al que se conecta la mota base.

2 PROGRAMA DE LA MOTA INTEGRADA EN EL RFIDGLOVE

La mota tiene que ser capaz de llevar a cabo diferentes funciones. Por un lado tiene que comunicarse con el lector RFID para capturar lo que éste ha leído. A continuación tiene que enviar el paquete con la información deseada a la mota base, a través de la red Mesh que estará formada por varias motas MicaZ. Además, tiene que actuar en función de lo que la mota base le conteste. Por ejemplo, en la v.2 de RFIDGLOVE, la mota es capaz de visualizar en una pantalla uOLED de 4D Systems el resultado de la lectura, así como notificaciones que se envían desde la estación base en forma de texto.

El programa implementado en la mota integrada en el guante tiene la siguiente configuración:



Componente RFIDGlove

Tal y como se muestra en la figura, el programa hace uso de diferentes componentes. A continuación se explican los detalles más importantes que hay que tener en cuenta.

2.1 Configuración del puerto UART:

El lector RFID que hemos empleado transmite los datos a 9600 baudios y las motas MicaZ, por defecto trabajan a 57600 baudios. Es necesario reducir la tasa de transmisión de las motas.

Una vez que hemos instalado cywing, por defecto, dentro de la carpeta C:\Crossbow\cygwin\opt\MoteWorks\tos\platform\mica2 encontramos dos ficheros HPLUAR0M.nc y HPLUARTC.nc. Para poder modificar el *baud rate*, será necesario copiar ambos ficheros en el siguiente directorio:

C:\Crossbow\cygwin\opt\MoteWorks\tos\platform\mica2

Si nos fijamos detenidamente, vemos como en el fichero HPLUARTM.nc, dentro del método init(), existe una línea de código en el que se especifica la tasa de transmisión en baudios.

Call Setbaud(TOS_UART0_BAUDRATE);

Por defecto está configurado a 57600 baudios.

Para nuestra aplicación, nos interesa que funcione a 9600 baudios, de modo que sustituiremos la línea anterior por la siguiente:

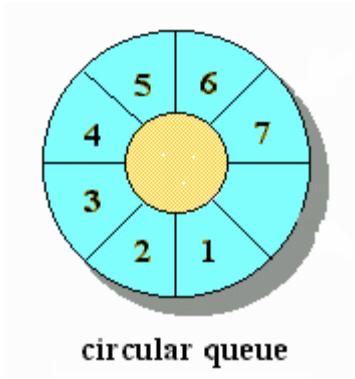
Call Setbaud(9600u);

Una vez realizadas estas modificaciones, lo único que hay que hacer es especificar en nuestro fichero RFIDGloveM.nc que vamos a utilizar el interfaz HPLUART e iniciarlo dentro del método init().

2.2 Implementación de la cola:

Puede haber situaciones en las que el RFIDGlove no pueda comunicarse con la mota base, como por ejemplo la falta de cobertura. En caso de que no se pueda realizar la comunicación, el guante tiene que ser capaz de encolar la información para transmitirlo más adelante, y perder la menor información posible.

La solución que se ha adoptado es la implementación de una cola fifo circular. Tal y como su definición específica, el primer elemento almacenado en la cola será el primero en ser enviado cuando sea posible y se tratará de forma circular.



El tamaño máximo de la cola está limitado por el hardware. Las motas MicaZ disponen de una memoria RAM de 4KB. Se han realizado diferentes pruebas para limitar el tamaño máximo de la cola que se puede implementar.

Por un lado se ha elaborado un programa que únicamente cree una cola del tamaño máximo posible. En cada posición de la cola se almacenará un paquete XDataMsg. Se ha comprobado que en este caso el tamaño máximo es de 189 posiciones. Al añadir el código necesario para desarrollar el RFIDGlove, el tamaño máximo de la cola se reduce a 97 posiciones. Cuando la cola se llena, no se almacenan más datos en ella, se descartan todos los datos recibidos.

- result_t **init**(void)
- uint8_t **enqueue**(XDataMsg msg)
- result_t **dequeue**(void)
- XDataMsg **get**(void)

Métodos del interfaz Queue

2.3 Control del reloj de la mota:

En esta aplicación nos interesa saber la hora en la que se ha realizado cada lectura. Para ello hemos utilizado el interfaz Time del componente SimpleTime.

Realizando la siguiente llamada capturamos los 32 bits de menor peso del tiempo en milisegundos binarios en formato *little endian*.

```
call Time.getLow32();
```

Para convertir milisegundos binarios en segundos hay que dividirlos entre 1024.

A lo largo del programa para cumplir los requisitos del RFIDGlove como por ejemplo el

parpadeo de los leds, se ha tenido que implementar un método *wait(uint16_t ds)* que realiza una llamada a TOSH_uwait. De este modo se puede, por ejemplo, encender un led, esperar un tiempo determinado y apagarlo.

Se ha comprobado que ejecutando el programa sin utilizar este método, el tiempo que transcurre entre la captura consecutiva de dos muestras es la esperada. En cambio, si se emplea el método *wait* se produce un retraso que depende del tiempo que pretendemos esperar en el wait.

Para entender mejor lo que sucede vamos a suponer que en el programa capturamos dos muestras con un intervalo de 5000ms entre ambos y que realizamos una única llamada al método wait para que espere 2000ms. Si el tiempo en que se realizó la primera muestra es 1000ms (desde el instante en que se encendió la mota), la segunda muestra debería dar un tiempo de 6000ms, pero en lugar de eso el tiempo que se obtiene es de 8000ms.

Para solucionar este problema se ha empleado una variable llamada offset, que se encarga de ir acumulando la sumatoria de todos los tiempos que se le pasan al método wait a lo largo de las sucesivas llamadas realizadas al método entre dos capturas consecutivas. De este modo, cuando se recupera el tiempo del reloj interno de la mota, se calcula el tiempo transcurrido desde la captura anterior y se le suma el offset antes de enviar el tiempo a la mota base.

2.4 Ahorro del consumo energético:

Una buena gestión del consumo energético es de gran importancia en este tipo de dispositivos, para que funcionen de forma autónoma durante el mayor tiempo posible.

El guante integra además de la mota MicaZ, un lector RFID y una pantalla µOLED. Las motas MicaZ están optimizadas para funcionar en redes de bajo consumo, y consumen poca potencia. En el caso del lector RFID, tiene que estar todo el rato alimentado para ser capaz de leer tags pasivos en cualquier instante de tiempo. En cuanto a la pantalla, es posible reducir el consumo activando la pantalla únicamente cuando va a ser utilizada para visualizar una imagen. El resto del tiempo, mientras no se está utilizando la pantalla, está permanece apagada.

Para el control del µOLED se ha implementado un interfaz llamado intUOLEDController. Este interfaz permite actuar sobre la pantalla para poder encenderla, mostrar imágenes, escribir texto, borrar la pantalla, apagarla, etc. De modo que a través del método *turnOff* es posible apagar y encender la pantalla cuando sea necesario y ahorrar batería.

- void **display**(bool onoff)
- void **clear_screen**(void)
- void **device**(bool onoff)
- void **turnOff**(bool onoff)
- void **draw**(char position[])
- void **displayImageFromMemoryCard**(char x, char y, char width, char height, char colourMode, char sectHi, char secMid, char sectLo)
- void **writeText96**(char text[], uint8_t colour)
- void **writeText160**(char text[], uint8_t colour)
- void **establishContrast**(char level)
- void **fadeToBlack**(void)
- void **fadeIn**(void)

Métodos del interfaz intUOLEDController

3 COMUNICACIÓN ENTRE RFIDGLOVE Y MOTA BASE:

La comunicación entre el dispositivo RFIDGlove y la mota base se realiza a través del servicio de enrutado multihop XMesh. Para ello se han añadido los componentes GenericCommPromiscuous y MULTIHOPROUTER.

El protocolo de comunicaciones que se ha implementado para la comunicación entre RFIDGlove y la mota base es la siguiente:

- Cada vez que el guante recibe datos, los almacena en la cola y los envía a la mota base. Estos paquetes son de tipo XDataMsg.

- La mota base procesa el paquete recibido y envía un paquete de confirmación al RFIDGlove. Este paquete contiene el código asociado al tag leído por el lector RFID para que el RFIDGlove pueda visualizar la imagen correspondiente. Estos paquetes son de tipo DisplayImage.

- El RFIDGlove, cuando recibe el paquete que viene desde la mota base, comprueba si es la contestación al que él ha enviado anteriormente. En caso afirmativo, borra el paquete de la cola porque ya ha sido tratado.

- Si no recibe la contestación correspondiente, el paquete no se elimina de la cola y el RFIDGlove sigue enviando el paquete hasta que no reciba la contestación desde la base.

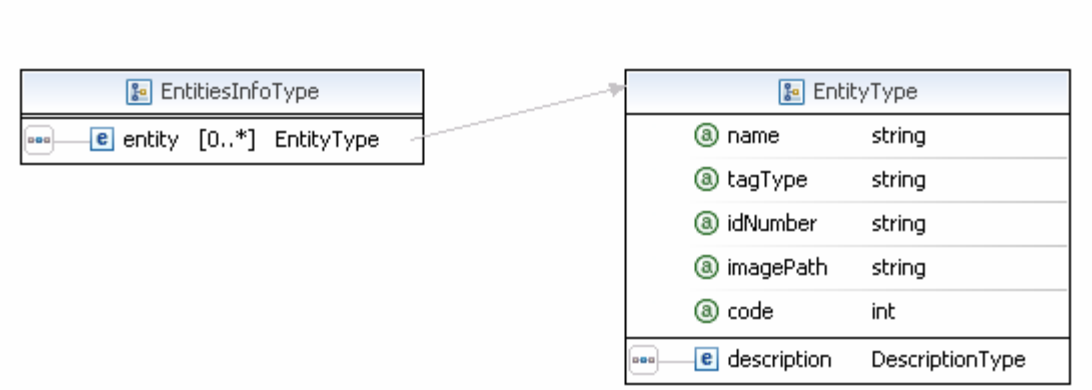
- Además la mota base también puede enviar otro tipo de paquetes que sirven para enviar texto desde la estación base hasta el RFIDGlove. Estos paquetes son de tipo DisplayText y se emplean cada vez que desde la estación base se solicita un envío de texto pulsando el botón de la GUI correspondiente.

3.1 Mota base:

El código implementado en las motas MicaZ está programado en nesC sobre Tinyos.

La mota base, interactúa con java a través de las librerías tinyos.jar que se encargan de gestionar las operaciones de bajo nivel. Desde java se realiza el tratado de los paquetes recibidos.

Cuando la mota base recibe un paquete desde el RFIDGlove, coge el identificador de la tarjeta que se ha leído y comprueba en un fichero xml si reconoce esa tarjeta. Cada tarjeta además de un número de identificación (idNumber), tiene un código asociado (code).



Diseño del esquema xml

La mota base, coge el código asociado a la tarjeta del fichero xml, y envía un mensaje de contestación al RFIDGlove, con el código correspondiente, para que este pueda buscar la posición de memoria con la que se corresponde en el µOLED y lo pueda visualizar.

Además, los datos relacionados con el tag que ha sido leído se visualizan en el interfaz gráfico de la aplicación local (anexos 4.5). Para calcular la información de la hora en la que fue realizada la lectura, hay que tener en cuenta la fecha y hora actual, que se obtienen del ordenador y la información de tiempo relativo que llega desde el RFIDGlove. El tiempo relativo indica el momento en que fue leído el tag, teniendo como origen el instante en que se encendió el guante. A partir de estos datos, y sabiendo la hora actual se calcula en que hora se encendió el guante, así como la hora en la que fue realizada la lectura.

Hay que tener en cuenta que el guante puede ser apagado en cualquier instante por el operario, de modo que el tiempo relativo que llega desde el RFIDGlove volvería a empezar de cero. En este caso, se calcular el nuevo instante en que fue encendió el guante, y de este modo se puede precisar cuando fue realizada la lectura del tag.

Por otro lado, para tener el historial de todos los paquetes enviados por el RFIDGlove desde que se puso en marcha la aplicación, éstos se van guardando en un fichero csv.

En el fichero csv se guarda la siguiente información:

- El identificador del tag leído

- La hora en la que se produjo la lectura

4 DISEÑO HARDWARE:

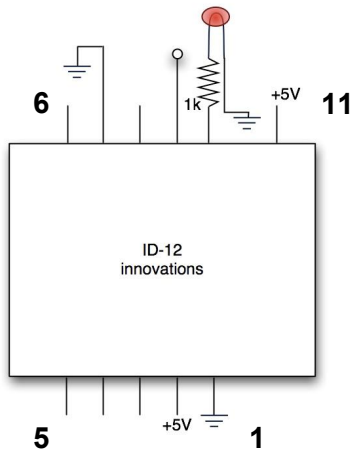
Los principios fundamentales seguidos para el diseño hardware del RFIDGlove son consecuentes con los requisitos del proyecto. Por ello, consultando adecuadamente el documento correspondiente, el primer paso fue la elaboración de un listado de los elementos cuya integración resulta completamente necesaria para cumplir con los objetivos funcionales. Tras esto, en el siguiente apartado se presentará el diseño hardware, seguido de su implementación, y de la redacción de las principales trabas e incidencias surgidas. Finalmente, el último apartado presentará imágenes reales del prototipo resultante.

4.1 Elementos a integrar en función de los requisitos:

Dada la funcionalidad global del guante, es necesario un lector RFID. El modelo seleccionado es el ID-12, lector de baja frecuencia, debido a su reducido tamaño y a su alta capacidad de lectura de tags a distancias razonables. Las razones de su selección frente a otros modelos es el equilibrio entre estas características y su adecuación a un precio coherente. Cabe recordar aquí que uno de los objetivos secundarios en la fabricación del RFIDGlove es la obtención de un producto relativamente económico. La apariencia de este dispositivo se presenta en la siguiente imagen.

ID-12

El esquema electrónico sigue la siguiente configuración, y es importante de cara a la conexión con otros dispositivos tener presente que la comunicación serie asíncrona funciona a 9600 baudios.



Esquema de conexiones del ID-12

Respecto a sus pines, el 1 se conecta a tierra y el 11 a la alimentación de 5 V. El pin 2 es el de RESET, que debe estar conectado a 5 V para mantenerse en funcionamiento de forma continua. El 3 y el 4 están disponibles para conectarles una antena externa y un condensador variable, que permitan al usuario alterar, según el interés de su aplicación, las características de lectura. El pin 5 y el 6 no tienen aún funciones reales definidas, sino que están reservadas para futuros intereses. El 7 es el pin de selección de formato, que conectándolo a tierra proporciona los datos del tag leído en formato ASCII por medio de la línea de datos D0, colocada en el pin 9. La segunda línea de datos D1, situada en el pin 8 no se utiliza. El pin 10 da una función de señalización adicional, emitiendo un pulso lógico de 3.1 kHz cuando se sucede una lectura. En la fase de pruebas, se conecta a un led o a un zumbador, y éste se iluminará o sonará cuando se haya efectuado una lectura. En el caso del RFIDGlove, quedará sin utilidad, y por tanto, desconectado.

Así pues, los pines a conectar en el RFIDGlove serán el 1, 2. 7, 9 y 11.

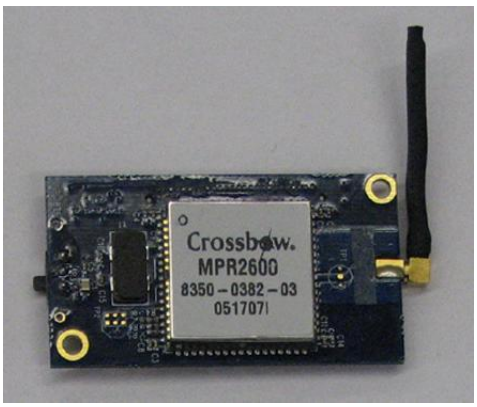
Dado que una de las funciones principales de la aplicación real propuesta para el RFIDGlove es la rápida obtención de información acerca del producto que se ha reconocido, es necesaria la integración de una pantalla, en la que se puedan visualizar tanto imágenes como texto. La elegida es una uOLED 160-GMD1, del fabricante 4DSystems. Las razones son fundamentalmente su alta calidad de imagen (con una resolución de 160*128 píxels) y las múltiples funciones que permite integrar, desde visualizar mensajes remotos de texto hasta la reproducción de vídeos previamente almacenados en la memoria. La forma de emplear esta pantalla es relativamente simple además, ya que basta con enviar por una línea serie asíncrona los correspondientes valores ASCII para los diferentes comandos y datos. Por último, dispone de un lector de tarjetas µSD, en la cual se pueden almacenar contenidos y referenciarlos como parte del funcionamiento de la aplicación.

Sólo consta de 5 pines, ya que la comunicación (también a 9600 baudios) se desarrolla en serie, lo que facilita la integración en el conjunto del RFIDGlove.

uOLED 160-GMD1

Desde la perspectiva de la imagen, el pin 1 es la alimentación, a la que deben suministrarse 5 V. Los pines 2 y 3 son los de transmisión (Tx) y recepción (Rx) respectivamente. Dado que para el RFIDGlove no será necesario enviar ningún valor ni comando desde la pantalla a la mota, solamente se conectará el pin 3 (cruzando el pin de transmisión de la mota con el de recepción de la pantalla). El pin 4 es tierra, y el 5 es el reset, que quedará al aire.

El tercer elemento fundamental es la mota MicaZ. Este género de dispositivos consiste en una plataforma de pequeñas estaciones sensorizadas con capacidad de comunicación inalámbrica, del fabricante Crossbow. La elección de estos dispositivos para mantener la comunicación de datos entre la estación base y el RFIDGlove viene debido fundamentalmente al bajo consumo energético que son capaces de mantener, la relativamente elevada capacidad de procesamiento lógico configurable y las características de comunicación inalámbrica que incorpora. La imagen muestra el aspecto de estas motas.



Mota MicaZ de Crossbow

Se trata de un modelo de prototipado, y son programables en NesC, una variante adaptada de C, y es el elemento encargado de realizar el procesado y gestión del conjunto de dispositivos que integran el guante.

La conexión de la mota con el resto del equipamiento se desarrolla mediante un conector de 51 pines, ubicado en su parte inferior.

Conector hembra de 51 pines

De éstos, los que se utilizarán en la aplicación serán los siguientes:

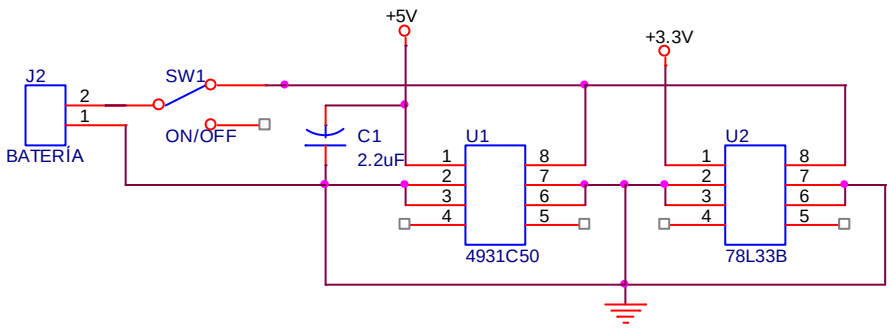
- Los pines 17, 18 y 19 serán las salidas por nivel bajo para los leds y zumbador indicadores; es decir, se activarán cuando se emitan 0 V, para crear una diferencia de potencial con los 3.3 V con que se alimentan.

- Los pines 27 y 28 son tierra y alimentación (3.3 V), respectivamente.
- El pin 37 será la entrada del pulsador que permitirá recuperar la información del último tag leído.
- Los pines 50 y 51 son los pines de la comunicación serie USART, transmisión y recepción respectivamente, y por lo tanto, el 50 se conectará a la pantalla uOLED 160-GMD1 para enviar la información pertinente, y el 51 recogerá la información de las lecturas del tag al estar conectado al lector ID-12.

El resto de componentes al respecto son los complementarios para las funciones de señalización y conexiones. Este conjunto se integra de leds, un pulsador, un zumbador, todos ellos con sus respectivas resistencias, y un interruptor para el sistema de alimentación. A éste también lo acompañan 2 reguladores de tensión de 5 y 3.3 V, con sus consecuentes condensadores. Sobre las conexiones, es necesario un bus que interconecte las 2 placas, situación que se expondrá más adelante.

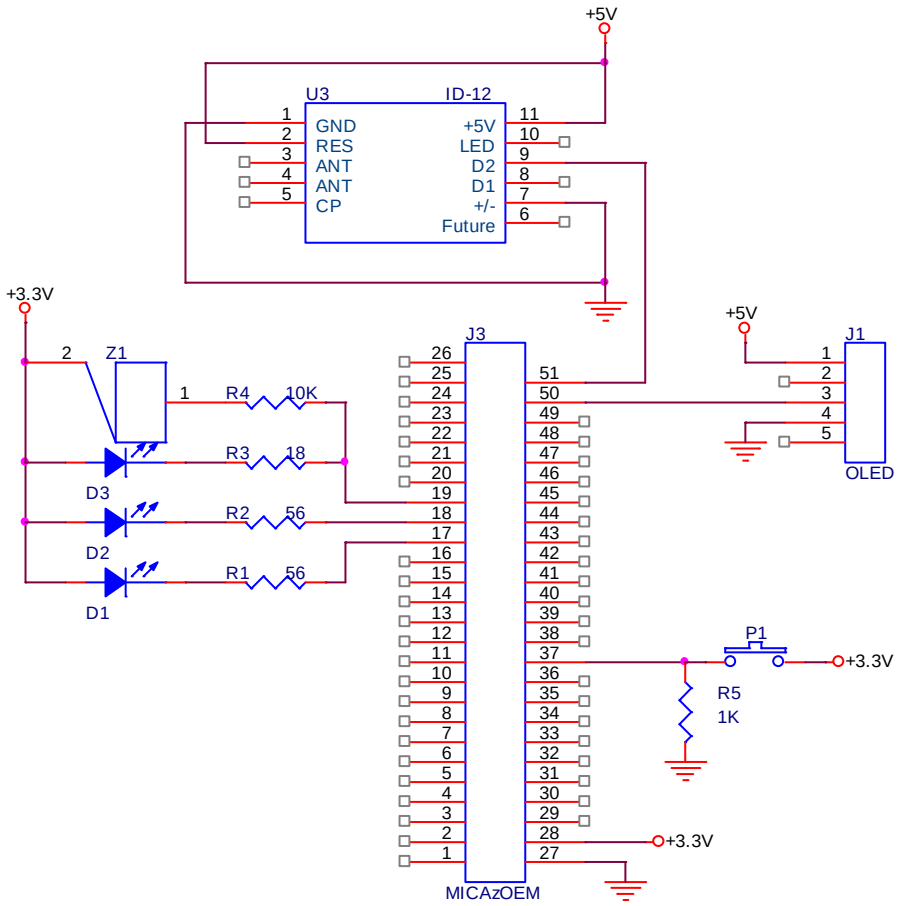
4.2 Diseño:

El bloque de la alimentación se presenta en la siguiente imagen, para la que se utilizará una batería de 9 V.



Bloque de alimentación

La gestión funcional del RFIDGlove requiere, como ya se ha presentado, del control del ID-12 y de la pantalla uOLED, así como de leds y pulsadores, desde el programa almacenado en la mota MicaZ, y por tanto el diseño de las conexiones queda como sigue.



Bloque de gestión

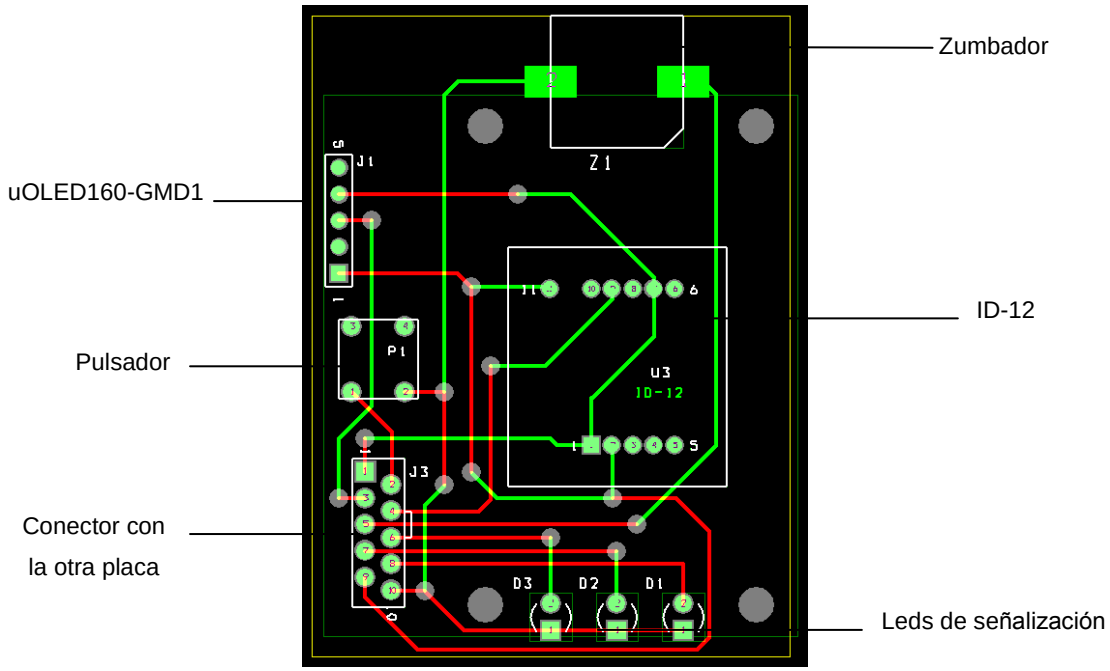
4.3 Implementación:

Antes de mostrar los diseños físicos de las placas, es el momento de comentar la necesidad y justificación de dividir el diseño en 2 placas separadas, en lugar de implementar toda la estructura del RFIDGlove sobre una sóla. Esta decisión viene determinada por el interés en agilizar el movimiento de la mano en el prototipo final del guante, dotándolo de mayor facilidad de movimiento, y por otro lado, la diferenciación de los componentes que lo componen. Se divide por un lado aquellos que conforman la interfaz visual del usuario (junto con el ID-12 para hacer más cómoda la capacidad de lectura por meras cuestiones

geográficas, que ha de estar necesariamente sobre el dorso de la mano), y por otro, transportables a la zona de la muñeca, donde es más cómodo para el usuario la disposición del equipamiento.

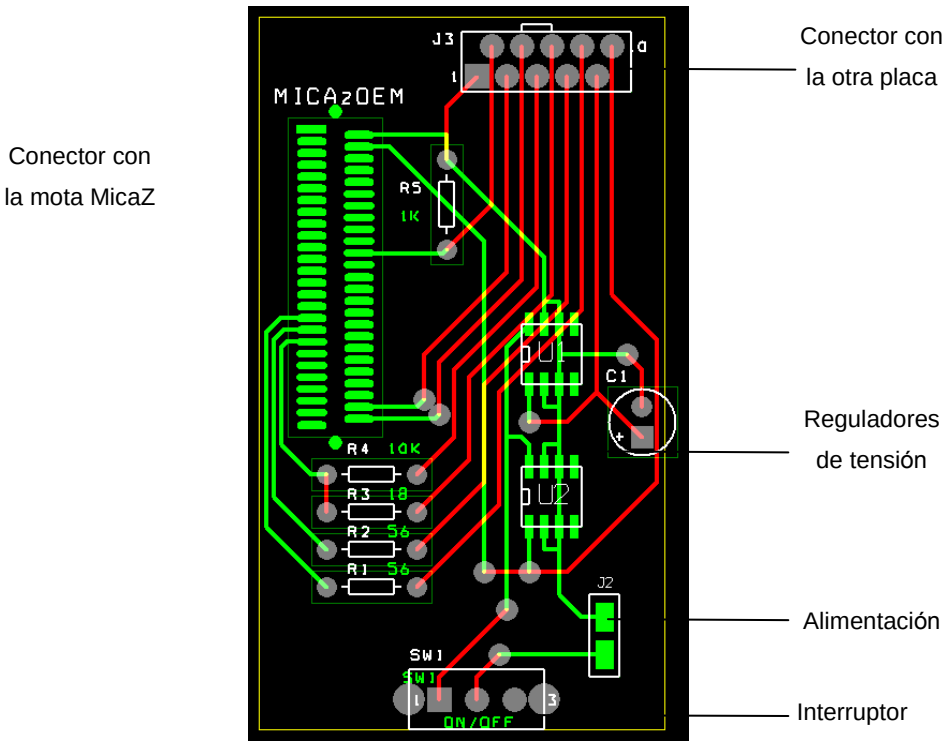
Para la comunicación entre ambas placas, se emplea un bus o conector paralelo de 10 pines, siendo esta cifra lo más reducida posible, dentro de la prioridad de reducir el número de componentes en la placa del dorso, que contiene la parte de interfaz.

Así pues, la siguiente imagen presenta la placa a ubicar el en dorso de la mano. Las ubicaciones principales están señalizadas con los comentarios laterales.



Placa 1 -dorso-

La segunda placa, a ubicar en la muñeca, se puede apreciar en la siguiente imagen.



Placa 2 -muñeca-

Respecto al conector de 10 pines entre las placas, sigue la estructura de conexiones que se describe en la siguiente tabla.

Pin	Nombre	Comentario
1	GND	Tierra del sistema
2	P1	Entrada del pulsador P1
3	TX	Línea serie TX para transmisión a pantalla
4	RX	Línea serie RX de recepción de ID12
5	Z1	Salida al zumbador Z1
+6	D3	Salida al diodo LED D3
7	D2	Salida al diodo LED D2
8	D1	Salida al diodo LED D1
9	+5V	Alimentación para pantalla e ID12
10	+3.3V	Alimentación para diodos, zumbador y pulsador

4.4 Incidencias y soluciones:

Como es habitual, a la hora de desarrollar e implementar un prototipo físicamente, surgen incidencias y problemas que, aún siendo de mayor o menor medida, necesitan solución. Las más reseñables serán expuestas en el presente apartado.

Un incidente leve se produjo con la selección de los leds, para el caso del color azul. El empleo de éste se antojaba de sumo interés, dada su relación convencional con las comunicaciones inalámbricas (acción con la cual está relacionada esta aplicación). Sin embargo, se trata del color de luz de mayor consumo, y el único caso de los deseados en que una caída de 3.3 V no resultaba suficiente para su iluminación. La solución vino de la sustitución de los leds convencionales por otro de bajo consumo y mayor iluminación, recalculando el valor de las resistencias correspondientes. Sin embargo, el encendido constante del led verde resultaba molesto para los ojos del operario, dado que estos leds brillan con mayor intensidad (de forma notoria), así que en este caso del verde se volvió al modelo tradicional.

Otra incidencia vino derivada de las limitaciones de tamaño en las placas. Dado que la reducción máxima de tamaño era uno de los principales objetivos en lo que a la implementación hardware se refería, se tomaron algunas decisiones. Por ejemplo, la idea inicial era aprovechar el espacio que queda debajo de la pantalla para instalar el ID-12 apuntando hacia arriba, en dirección inversa si se considera la lectura desde la palma de la mano. Esto no suponía un problema representativo, ya que el ID-12 lee también por su parte trasera.

Sin embargo, y con el mismo objetivo, también se decidió no incluir condensadores en las entradas y salidas de los reguladores, y asumir de esta forma las pequeñas caídas de tensión que podía sufrir el sistema. Estas dos cuestiones dieron lugar a confusiones que provocaban directamente un funcionamiento incorrecto: el ID-12 veía reducida drásticamente su distancia y rango de lectura de tags, y la pantalla se reseteaba a cada lectura con el ID-12.

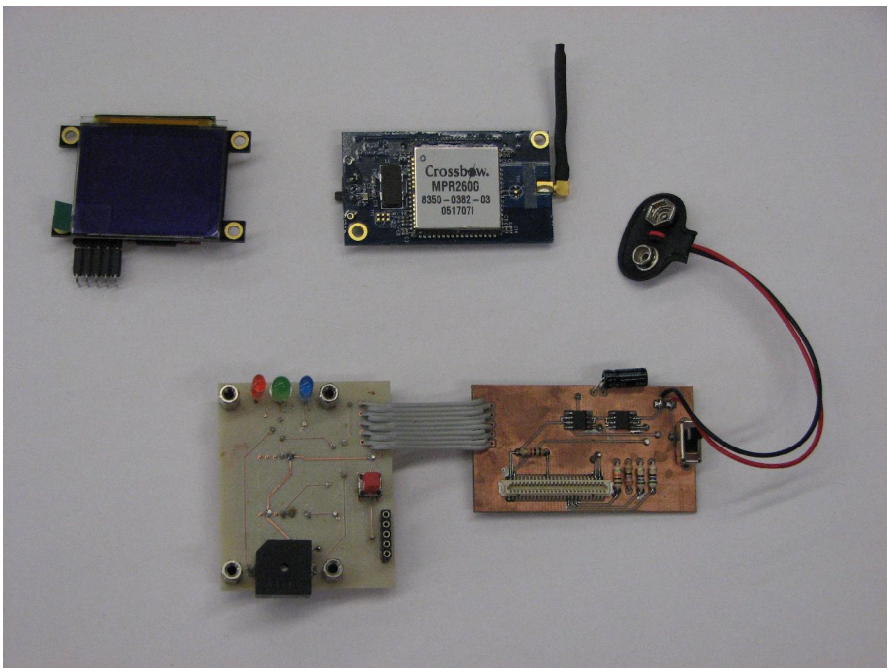
En la creencia inicial de que la ubicación del ID-12 podía afectar a la lectura (leía por su parte trasera, teniendo tras ella una placa metálica, superficies en las que las ondas

electromagnéticas rebotan, y teniendo delante la pantalla, por cuyos circuitos circulaba corriente, pudiendo crear los consecuentes campos), decidió modificarse su posición a la parte inferior de la placa, apuntando hacia la palma, y habiéndose eliminado completamente el cobre de la placa a excepción de las pistas. Sin embargo, tras estos cambios, se descubrió que no era esa la razón del mal funcionamiento de los dispositivos, pero dadas las condiciones expuestas, se tomó la decisión de continuar con el nuevo posicionamiento, ya que las condiciones sólo podían ser mejores que las anteriores. Además, teniendo que integrar pantalla, lector RFID y mota, la reducción del alto y ancho de las placas queda ya bastante limitada, así que tampoco afecta en exceso.

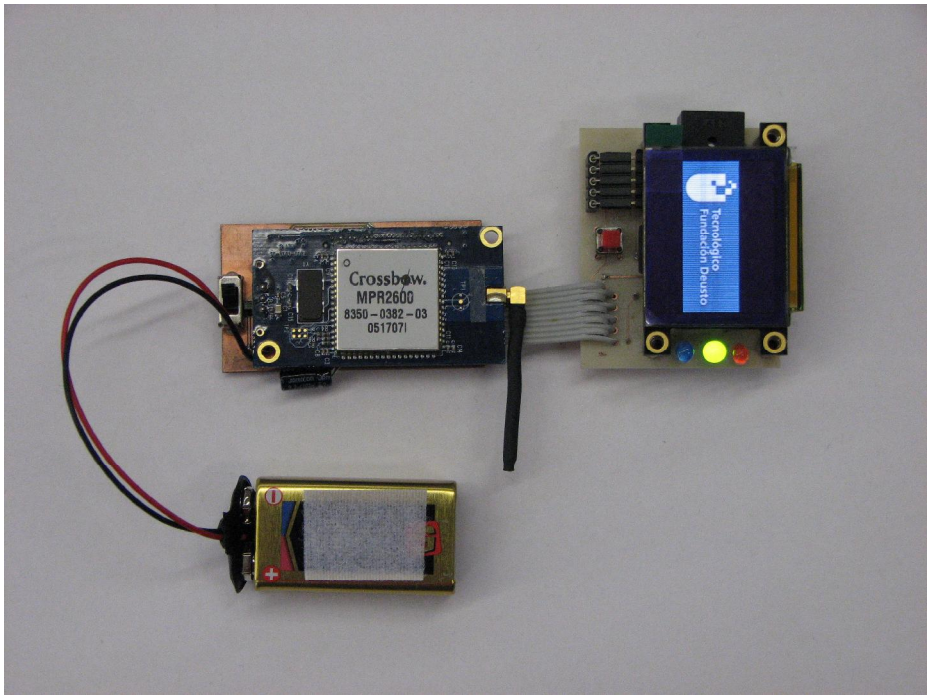
Finalmente, continuando con pruebas reiteradas, se descubrió que la falta del condensador de salida del circuito 4931C50 provocaba el reseteo de la pantalla en cuanto se realizaba una lectura RFID por medio del ID-12, y ambos dejaban de funcionar. La inclusión del condensador puso solución a todo este problema, permitiendo a la aplicación funcionar adecuadamente.

4.5 Prototipo funcional:

Las siguientes imágenes muestran el resultado de la implementación hardware.



Desglose del hardware



Implementación funcionando

Implementación en el guante

5 ANEXOS:

Formato de los paquetes que se utilizan en la comunicación entre el RFIDGlove y la mota base.

5.1 XDataMsg:

Éste es el tipo de paquete que envía el RFIDGlove a la mota base:

```
typedef struct XDataMsg {
        XMeshHeader xMeshHeader;
        union {
                RFIDGlove data;
         }xData;
} __attribute__ ((packed)) XDataMsg;
typedef struct RFIDGlove {
        uint8_t byte1;
        uint8_t byte2;
        uint8_t byte3;
        uint8_t byte4;
        uint8_t byte5;
        uint8_t byte6;
        uint8_t byte7;
        uint8_t byte8;
        uint8_t byte9;
```

```
    uint8_t byte10;

    uint8_t byte11;

    uint32_t timelow; // it is in binary miliseconds (1024 binary miliseconds=1second) and in little endian format.

} __attribute__ ((packed)) RFIDGlove ;
```

5.2 DisplayImage:

Éste es el tipo de paquete que envía la mota base al RFIDGlove para confirmar que la lectura ha sido realizada correctamente y pueda visualizar la imagen correspondiente en la pantalla:

```
typedef struct DisplayImage{

    XMeshHeader xMeshHeader;

    struct{

        int16_t    code;//The corresponding code of the tag to know the display memory position

        int8_t messageDisplTime;//The time to display the image on the screen

        int32_t timelow;

    }xData;

} __attribute__ ((packed)) DisplayImage;
```

5.3 DisplayText:

Éste es el tipo de paquete que envía la mota base al RFIDGlove cuando desde la mota base se solicita enviar texto al RFIDGlove pulsando el botón correspondiente en la GUI de la aplicación local:

```
typedef struct DisplayText{
```

```
    XMeshHeader xMeshHeader;
    struct{
        uint8_t messageDisplTime;
        char data[15];
    }xData;
} __attribute__ ((packed)) DisplayText;
```

5.4 XMeshHeader:

Todos los paquetes que se han mostrado anteriormente incluyen la siguiente cabecera porque la comunicación entre el dispositivo RFIDGlove y la mota base se realiza a través del servicio de enrutado multihop XMesh.

```
typedef struct XMeshHeader{
    uint8_t  board_id;
    uint8_t  packet_id;
    uint8_t  node_id;
    uint16_t parent;
}__attribute__ ((packed)) XMeshHeader;
```

5.5 Aplicación local del ordenador al que se conecta la mota base:

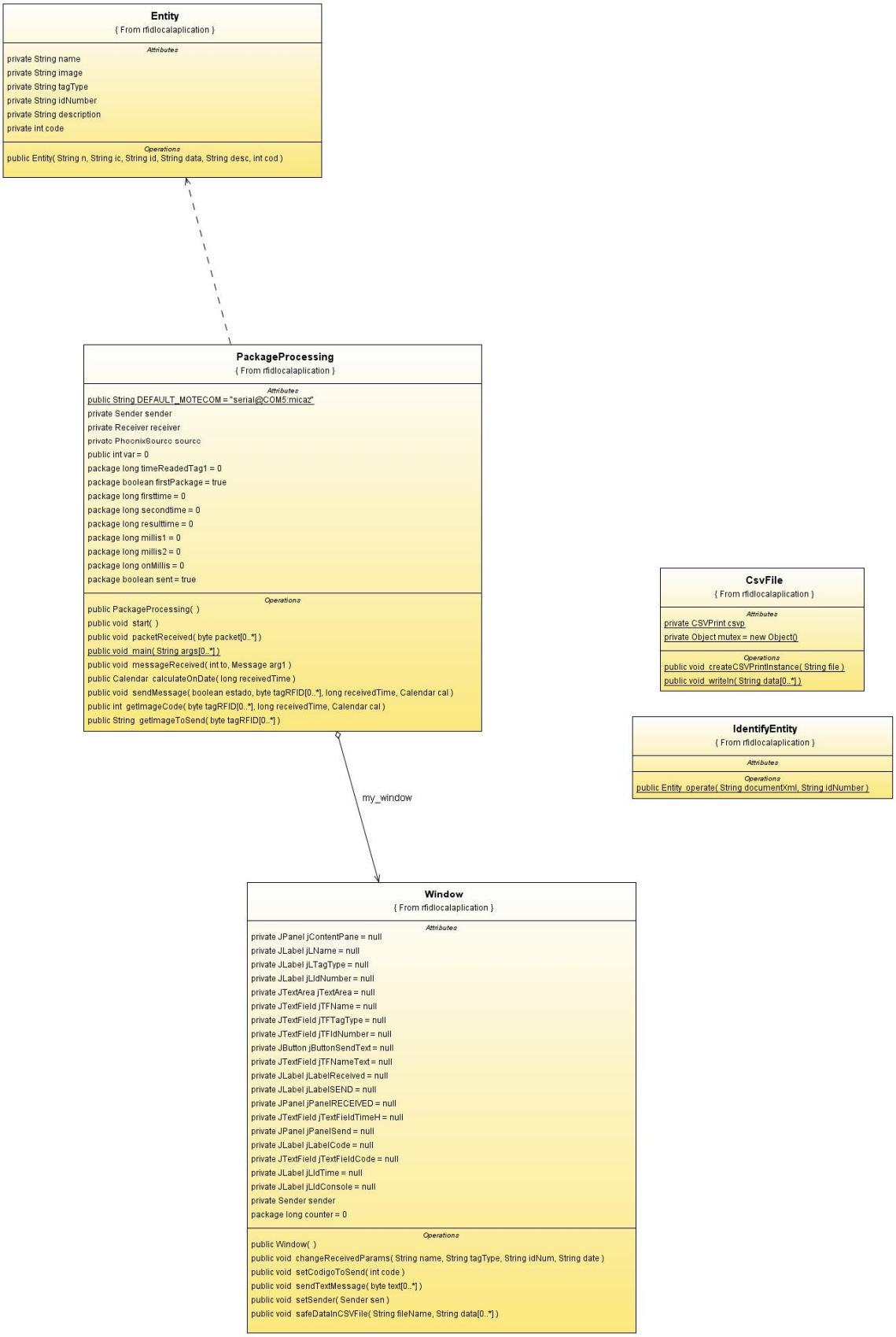
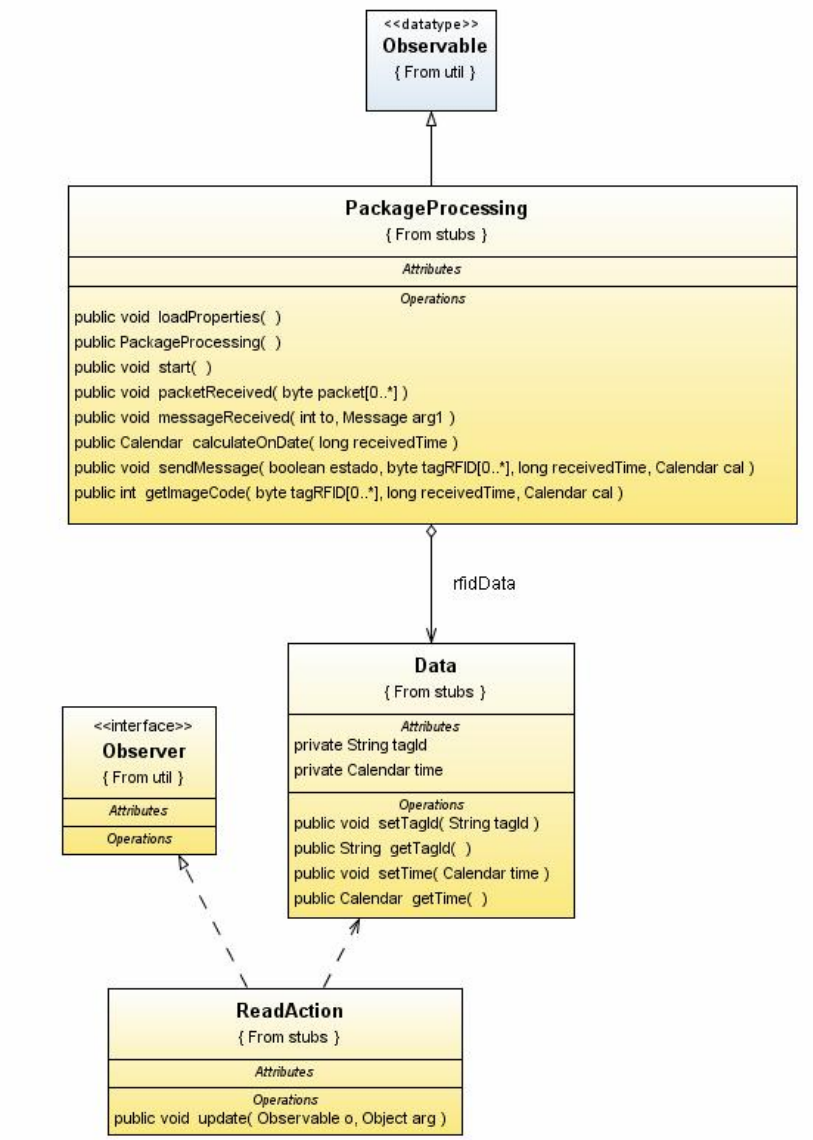
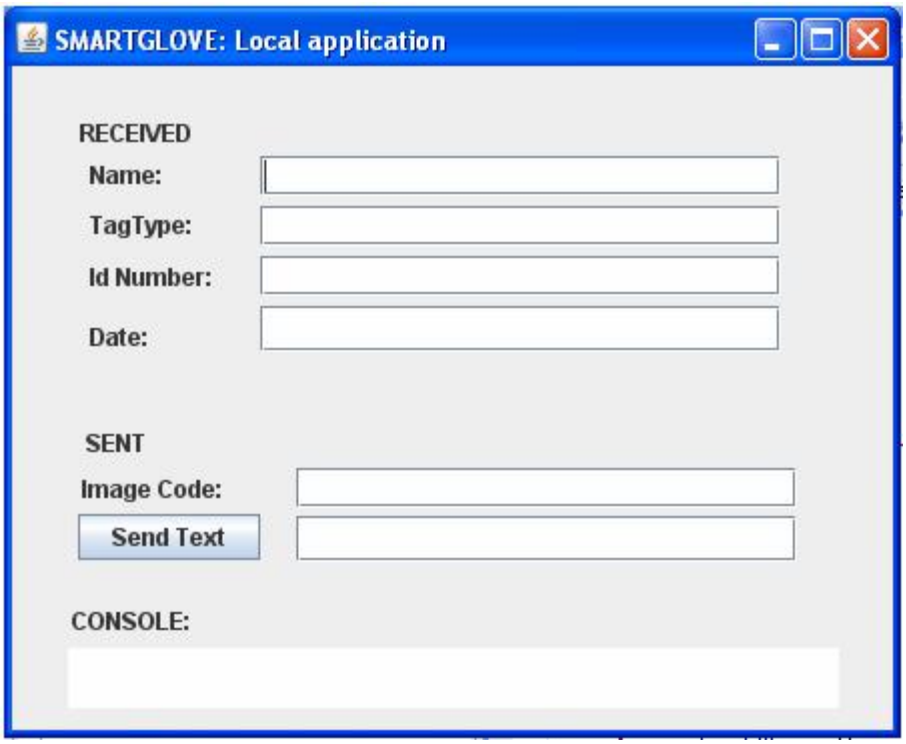


Diagrama de clases de la aplicación local



Patrón Observer para la notificación de lecturas de tags RFID



GUI de la aplicación local