Red meteorológica basada en Arduino y comunicada mediante protocolo Zigbee 802.15.4

Autor: David Morán Jiménez
Estudios: Ingeniería Técnica de Telecomunicaciones, especialidad Telemática

Consultor: Jordi Bécares Ferrés

Fecha: 3 de enero de 2012

Dedicatoria y agradecimientos

Este proyecto está dedicado a toda la comunidad que realiza el esfuerzo de compartir, recopilar y crear información en el portal arduino.cc/playground.

A los creadores de Arduino David Cuartielles y Massimo Banzi por desarrollar una herramienta con tanto potencial en el ámbito del Arte, Educación y Nuevas tecnologías.

Agradecimientos al Profesor responsable de la asignatura Xavi Vilajosana y el consultor Jordi Bécares de la Universidad Oberta de Catalunya por haberme permitido llevar acabo el desarrollo del proyecto sobre Arduino, y también a su interés y ayuda mostrados.

Agradecimientos a mi familia, amigos y en especial a mi pareja Noelia, los cuales han soportado junto a mí el esfuerzo que supone estudiar una carrera a distancia, además de compaginarla con el trabajo y vida personal.

Resumen

Desde la aparición de ***Arduino*** he estado rodeado por esta tecnología de una manera subjetiva, nunca había tenido la oportunidad de trabajar directamente con la plataforma pero si a su alrededor. Este trabajo de fin de carrera ha supuesto para mí una labor de investigación y enriquecimiento personal con la cual he conseguido los resultados propuestos en el planteamiento del proyecto y además la oportunidad de aprendizaje deseada.

El resultado del trabajo ha dado lugar a dos nodos o sistemas empotrados autónomos que en este momento desempeñan funciones de estación meteorológica, pero que pueden desempeñar otras funciones o ampliar sus posibilidades con pequeños cambios a nivel de software y hardware. La característica principal mencionada en repetidas ocasiones durante toda la memoria consiste en la capacidad de proceso **autónomo** de la cual están dotados los equipos. Para mí este ha sido el principio fundamental del proyecto, ya que la existencia de recursos similares es una realidad pero siempre con dependencia de diferentes equipos informáticos para la interpretación de datos.

Como estudiante de Ingeniería de Telecomunicaciones la posibilidad de usar una herramienta de comunicación como es el protocolo ***Zigbee*** ha sido muy enriquecedora, ya que aunque cuenta con años de existencia su uso no ha sido demasiado extendido. Ha sido implementado en los nodos desarrollados, creando una transmisión ***bidireccional***, la cual permite a los equipos certificar la comunicación mediante un sistema de CRC que garantiza el correcto envió de los datos.

El ahorro de ***energía*** ha sido un punto fundamental durante todo el desarrollo, lo cual ha dado lugar un sistema con capacidad para dormir, inhabilitando los procesos cuando sea necesario realizar ningún tipo de operatividad.

Como se ha visto, el trabajo de fin de carrera gira en torno a cinco puntos clave: Arduino, Autónomo, ZigBee, bidireccionalidad y energía, dando lugar al siguiente título de proyecto dentro del área de trabajos de fin de carrera "**Sistemas empotrados**":

Red Meteorológica basada en Arduino y comunicada mediante protocolo Zigbee 802.15.4

Índice de contenidos

Índice de figuras

1. Introducción

Anteriormente al desarrollo del proyecto se han dedicado varios meses de trabajo al estudio tecnológico que ofrecía la plataforma Arduino en combinación con el protocolo 802.15.4 o zigBee. Para ello se han realizado pruebas funcionales y de desarrollo con diferentes ejemplos existentes en la plataforma web de Arduino, todo ello basado en la búsqueda del potencial y posterior fiabilidad que presenta Arduino. Una vez comprobado que tecnológicamente era posible realizar un sistema autónomo basado en un coordinador y receptor a través de la plataforma mencionada y utilizando comunicación zigBee, se redactó una propuesta por la cual se planteó el desarrollo de una estación meteorológica que funcionase autónomamente, con un consumo eficiente de energía y sin necesidad de equipos informáticos.

A partir de la validación de la propuesta inicial, se puso en marcha un plan de trabajo para realizar las estimaciones temporales y tecnológicas necesarias para el correcto desarrollo del proyecto. Dicha estimación colocó la duración del desarrollo del proyecto entre el 26 de octubre de 2011 y el 3 de enero de 2012, incluyendo no solo el desarrollo físico, si no también, las diferentes entregas requeridas y la posterior documentación de la memoria.

Una vez realizada la planificación, se llevó a cabo la primera fase de desarrollo del proyecto, este ha consistido en obtener todos los equipos necesarios como son: sensores de humedad, temperatura y luz; tarjetas Xbee Shield y Ethernet Shield; cajas estancas para la colocación de los equipos y librerías e información necesaria.

A raíz de la obtención de los equipos necesarios se dio paso a la construcción de las motas, en una primera fase se buscaba tener un sistema básico o prototipo que pudiera comunicarse inalámbricamente y fuese capaz de realizar mediciones. Dicho prototipo no contaba con control de errores, control CRC, bidireccionalidad y estructura modularizada de la programación.

A continuación se muestran algunas imágenes del proceso de fabricación hasta llegar a completar el equipo:

Figura 1, sensores utilizados.

En la siguiente imagen se muestra un nodo coordinador funcional, el cual no dispone de carcasa, bidireccionalidad, funcionamiento Ethernet y protección contra errores:

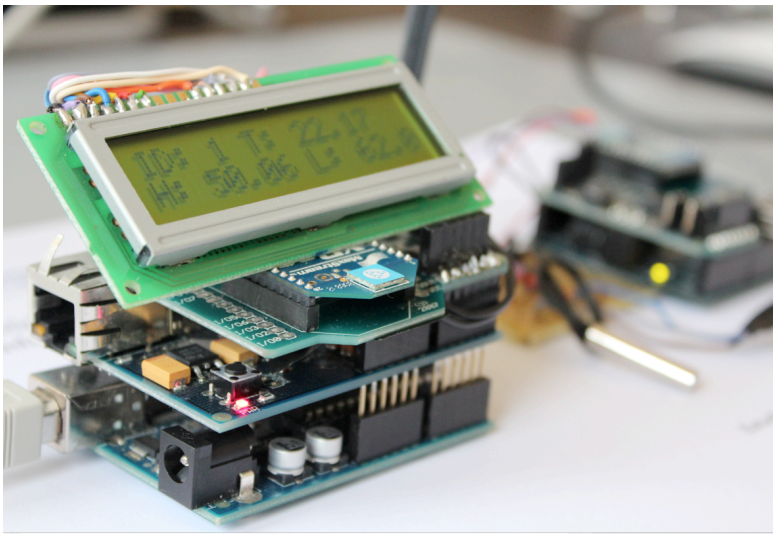


Figura 2, nodo receptor dotado de pantalla LCD.

La siguiente imagen muestra el nodo emisor, dotado de un pequeño circuito donde han sido ensamblados los sensores. No dispone de bidireccionalidad, ni protección frente a errores:

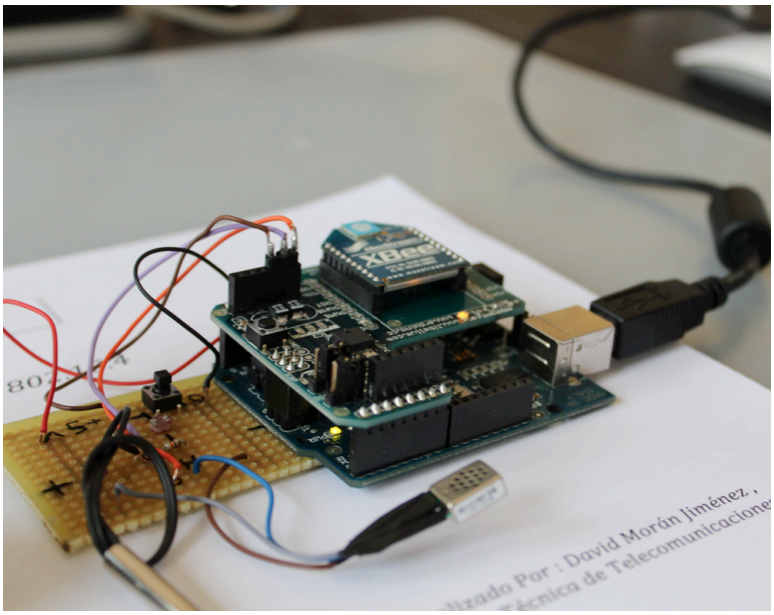


Figura 3, nodo emisor dotado de sensores.

A continuación se pueden ver el sistema funcionando en una primera etapa:

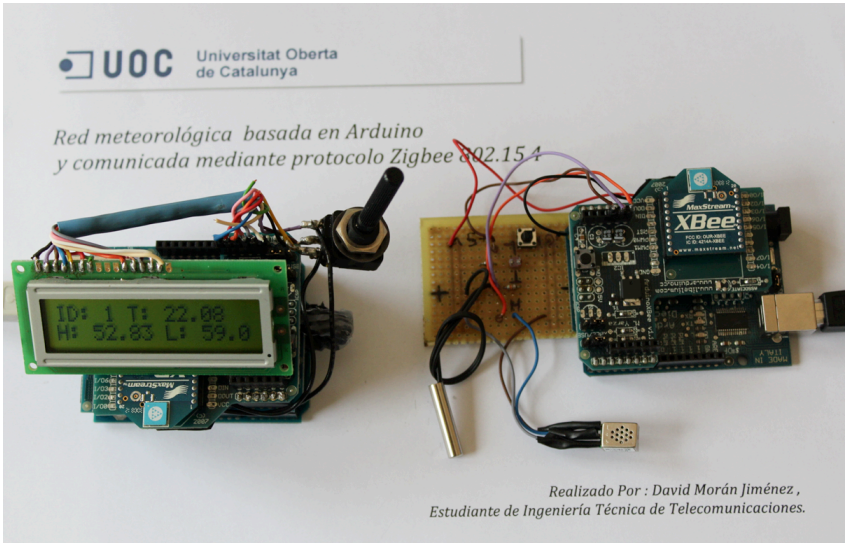


Figura 4, prototipo del sistema

Una vez concluido la primera etapa, se dio paso a la integración de equipos en el interior de unas cajas, y la dotación de pulsadores, interruptores y baterías.

Figura 5, equipo terminado.

Para el desarrollo de los Sketch o códigos se han empleado fuentes como: Comprobación de errores CRC, transformaciones de caracteres, librerías para el LCD, todas ellas obtenidas de la plataforma Arduino. El resto de código ha sido realizado por el estudiante incluyendo un protocolo básico de comunicación por mensajes entre los nodos. El socket UDP servidor ha sido extraído de una práctica de protocolos de Aplicaciones de Internet en la UOC, cursada por el estudiante en 2010 sobre el que se han realizado pequeñas modificaciones.

A continuación vemos un resumen de las características obtenidas en la primera y segunda versión:

Primera versión:

- *Comunicación inalámbrica unidireccional entre los dos nodos.*
- *Captura de datos a través de los sensores de temperatura , luz y humedad en el nodo emisor.*
- *Interpretación de los datos en valores inteligibles relacionados con valores estándar, como grados centígrados y humedad relativa.*
- *Envió de datos desde el nodo emisor al nodo receptor.*
- *Recepción de datos en el nodo receptor.*
- *Procesamiento de los datos en el nodo receptor.*
- *Lectura de datos vía LCD de los valores de temperatura , humedad y luz.*

Segunda versión:

- *Comunicación inalámbrica bidireccional entre los dos nodos y configuración del módulo Xbee modo Coordinador y punto final respectivamente.*
- *Control de errores implementando CRC para controlar que los datos se han enviado y recibido correctamente y envío conjunto de variables.*
- *Control de datos, se interpreta si un dato devuelve valores poco comunes.*
- *Protección contra lecturas incorrectas, si un dato no ha sido recibido correctamente se vuelve a solicitar.*
- *Pequeño protocolo de comunicación por mensajes, con función tipo ACK y mejora de código aplicando programación estructurada para mantenimiento del código y ampliación posterior.*
- *Control de energía, equipo Emisor duerme hasta ser despertado por el coordinador.*
- *Equipo Coordinador preparado para tres equipos emisores, con control de petición de datos por equipo automáticamente.*
- *Adecuación del funcionamiento LCD, utilizando solamente dos funciones para imprimir por pantalla y Botón para el cambio de la frecuencia de medición.*
- *Funcionamiento Ethernet con protocolo UDP envió de datos a través del puerto Rj45.*
- *Adecuación de Sokect UDP modo servidor para Linux.*

1.1. Justificación

La realización de este proyecto ha sido motivada por el bajo o inexistente número de sistemas empotrados basados en Arduino capaces de funcionar sin equipos informáticos. Analizando diferentes proyectos se pudo comprobar que la gran mayoría de los nodos receptores de información (Estaciones meteorológicas, sistemas empotrados, etc.) requerían de un ordenador para poder interpretar los datos recabados, por esta razón se planteó la posibilidad de desarrollar un sistema autónomo, el cual contase con una interfaz para la lectura directa de datos. Además de incluir un pantalla LCD para dicha funcionalidad, se ha dotado al sistema de una función de modificación de los intervalos de medición ambiental.

El sistema cuenta con la posibilidad de variar la precisión temporal con la que se realizan las capturas de valores en relación a la temperatura, humedad y nivel de luz. Para ello se ha desarrollado un sistema el cual permite una comunicación bidireccional, es decir, los equipos emisor y receptor pueden comunicarse entre si. Estas características hacen que el equipo pueda ser modular y en condiciones donde no se disponga de una infraestructura para albergar equipos más potentes, el sistema resulta verdaderamente práctico. Además brinda la posibilidad de su utilización en diferentes usos tanto doméstico o profesional, ya que variando los intervalos de medición podemos recabar más valores, obteniendo resultados más precisos.

Para no cerrar las puertas a que los datos tomados queden encerrados dentro del sistema, el equipo Coordinador o Receptor cuenta con una tarjeta de transmisión Ethernet. Dicha interface permite a Arduino enviar datos a través de una red local o de Internet, por lo que pueden recibirse en un equipo remoto, permitiendo con ello no estar físicamente en el lugar para visionar los datos.

Ha sido de especial importancia la posibilidad que Arduino brinda a la hora de ser modificado para trabajar como un equipo totalmente distinto a su función principal, por lo que en estos momentos la capacidad de estación meteorológica puede ser ampliada en cuanto a funciones se refiere: añadir sensores, nuevos módulos Shield o cambiar totalmente su utilización. Característica abalada por sus entradas y salidas analógicas y digitales de las que Arduino dispone, sin necesidad de utilización de equipos electrónicos como soldadores, etc.

1.2. Descripción del proyecto

El proyecto consiste en desarrollar una estación meteorológica basada en tecnología Arduino, compuesta por un nodo emisor y un nodo receptor, los cuales son comunicados entre si a través del protocolo zigBee 802.15.4 utilizado por los módulos Xbee Shield conectados a las placas. Los nodos constan con la peculiaridad de ser autónomos, es decir, no precisan de ningún ordenador para recibir, enviar e interpretar los datos. El sistema está preparado para soportar tres nodos emisores.

Este modo de funcionamiento dista bastante de los proyectos basados en la misma tecnología (Arduino), los cuales trabajan en una configuración nodo emisor autónomo y nodo receptor conectado a un pc, siendo el ordenador el encargado de la gestión de software y del funcionamiento del nodo receptor o coordinador, como he comentado en el apartado anterior.

El nodo coordinador cuenta con una interface Ethernet Shield para enviar datos a un socket o plataforma a través de la red cableada. Dicho envío sirve para poder ampliar la capacidad del sistema empotrado, contando no solo este con capacidad propia para mostrar los datos obtenidos sino también con posibilidad de envió de datos al exterior de la red.

Para el desarrollo de las funcionalidades ha sido necesario un proceso de construcción, en el cual ha tenido lugar el montaje de un pequeño circuito donde han sido ensamblados los sensores a una tarjeta perforada ubicada en el nodo emisor. Los equipos están contenidos dentro de unas cajas que permiten aislar a estos de ataques externos. Dichas cajas están dotadas con botones de apagado y encendido de los equipos, así como de un botón para la variación del tiempo de medición en el nodo Coordinador.

El nodo receptor se ha equipado con una pantalla LCD así como de un cableado para su conexión al Arduino y de un pequeño bus de cables para poder conectar los pin Serial a la interface Xbee Shield entre el Arduino y el Ethernet Shield. El sistema cuenta con una protección contra errores para evaluar si los datos recibidos son correctos, además de una bidireccionalidad para "hablar" entre si. El nodo coordinador tiene la función de despertar a los nodos emisores para recibir los datos y dormirlos posteriormente (modo sleep), una vez estos hayan enviado la información el coordinador vuelve a dormirlos en cuanto a gestión de procesos se refiere.

El equipo coordinador cuenta con la capacidad de variar el tiempo de toma de valores mediante un botón, el cual permite adaptar los intervalos de captura datos en función al entorno donde se esté trabajando, pudiendo ser este un entorno profesional donde se necesiten por ejemplo intervalos de medición cada 5 segundos, o un emplazamiento que solamente necesite la captura de datos 1 vez cada hora.

1.3. Objetivos del TFC

Para sinterizar todos los objetivos que deben de cumplir las diferentes partes del proyecto, se ha desarrollado una tabla que muestra el valor y justificación que tienen los mismos:

Objetivos	Valores	Justificación
Obtención y transmisión de valores de Temperatura , Humedad y Luminosidad	Registrar temperatura, humedad y luminosidad. Transmitirlos al nodo receptor para ser visionados.	Principal cualidad del sistema, recabar información del entorno y transmitirla a un nodo receptor mediante protocolo ZigBee para ser visualizada en el momento.
Realización de un sistema de bajo Coste	Obtención de una herramienta poco costosa pero eficaz, que se pueda utilizar en un entorno doméstico o profesional. El prototipo siempre contendrá un valor más alto, que los desarrollos posteriores.	El objeto del proyecto no puede se costoso, ya que su aplicación puede ser de tipo doméstico y no sería rentable tener un sistema de alto valor para este fin.
Hardware modificable fácilmente	Sistema que acepte cambios sin utilización de herramientas específicas, ampliación de sensores, etc.	Arduino permite colocar dispositivos sin falta de utilizar soldadores o herramientas, tiene entradas y salidas directas.
Consumo de energía reducido	Efectividad de los recursos energéticos, modos Sleep o de ahorro.	Los nodos emisores pueden estar en lugares de difícil acceso, por lo que debe de realizarse una programación optimizada para evitar el inconveniente de cambiar las baterías frecuentemente.
Red modificable	Ampliación o reducción de estaciones meteorológicas.	Es posible que un entorno profesional se necesiten obtener valores de diferentes ubicaciones, por lo que el sistema debe de brindar la posibilidad de obtener datos de diferentes nodos.

Sistema Autónomo	Sistema Autónomo sin necesidad de equipos para la obtención de datos.	El sistema debe de permitir poder obtener datos sin necesidad de utilizar un equipo informático, ya que puede encontrarse en un entorno complicado, un bosque, etc.
Rápida visualización de los datos	Visualización de los datos en pantalla LCD.	Respaldando el objetivo anterior, el sistema ha de permitir visualizar los datos. Utilización de una interfaz de tipo LCD.
Bidireccionalidad	Los equipos deben de comunicarse en ambas direcciones.	Los nodos han de comunicarse, para comprobar que los datos enviados son correctos, optimización de energía y control de errores.
Control de errores	Debe de existir un CRC.	Apoyado en la plataforma Arduino playground, se implementara al sistema de código CRC.
Envió de datos al exterior de la red	Visualización de los datos desde fuera de la red, internet, red local, host remoto, etc.	Como función complementaria al sistema, este contiene una interface Ethernet que permite enviar los datos a un PC, o Host remoto para almacenar los datos o ser monitorizados.
Sistema protegido ante peligros	Protección de los nodos mediante una carcasa o estructura.	Deben de ser protegidos ante inclemencias climáticas, ataques de animales o posibles riesgos que puedan dañar los sistemas electrónicos.
Utilización de Open Software	Utilización de software libre conlleva abaratamiento del producto.	Al utilizar software libre, el producto tiene un coste inferior, se pueden mejorar los equipos sin ningún tipo de restricción. Con ello se puede replantear el sistema o hacer modificaciones

Utilización de Open Hardware	Flexibilidad de modificación de hardware, e información total de todos sus componentes.	Al igual que en el caso anterior, si el hardware es libre se puede obtener todo tipo de información y prestaciones que los sistemas comerciales no podrían ofrecernos, como por ejemplo modificar su funcionamiento.

Figura 6, cuadro de objetivos.

1.4. Enfoque y método seguido

El enfoque del proyecto ha sido desde sus comienzos una idea basada en el aprendizaje y estudio de la tecnología Arduino. Para poder llevar acabo el mismo, se ha intentado seguir en la totalidad del proceso todos los recursos tales como: librerías, ejemplos, códigos o ideas existentes dentro de la plataforma arduino.cc/playground.

Dado que el sistema empotrado consta de aspectos que han resultado bastante dificultosos, tales como el envío de variables a través del serial, conversiones de caracteres a cadenas, comunicación bidireccional, CRC, la página oficial de Arduino ha resultado de vital importancia para llevar a cabo el proyecto.

En ella se almacena un repositorio para consultar el funcionamiento de las fuentes disponibles, librerías, y ejemplos de desarrollo. Ha sido un pilar fundamental para la construcción del proyecto ya que está dotada de una fuente de información muy grande para la plataforma.

La metodología de desarrollo e implementación ha sido durante todo el proceso un esquema basado en etapas con el seguimiento del diagrama de Gantt desarrollado en planificación del proyecto. Con ello se ha conseguido un trabajo equilibrado en el cual las dificultades abordadas han sido solucionadas en cada momento. Para la constitución de la memoria ha resultado de gran utilidad hacer una recopilación de todas las fuentes utilizadas durante el proceso de construcción y desarrollo del TFC.

La programación ha sido desarrollada estructuralmente, utilizando funciones que modularizan el trabajo dentro de los programas y facilitan la depuración de errores además de la implantación de nuevas funciones. No se ha contemplado el desarrollo de clases o librerías específicas ya que el código existente en cada función no resulta demasiado extenso.

1.5. Planificación del proyecto

A Continuación se detallan las tareas necesarias para la ejecución del proyecto. En el apartado siguiente es posible observar su cronología, fecha de ejecución y duración. Las tareas están distribuidas en función a las necesidades de cada PEC.

- *Propuesta de TFC.*

- *Planificación.*

- *Diseño y búsqueda de hardware.*

    - *Búsqueda de dispositivos.*
    - *Búsqueda de sensores.*
    - *Búsqueda de compartimentos.*
    - *Diseño de circuitos.*

- *Montaje de hardware.*

    - *Montaje de Arduinos y Xbee Shield*
    - *Instalación de sensores en nodo emisor.*
    - *Instalación de LCD en nodo receptor.*
    - *Instalación Ethernet Shield en nodo receptor.*
    - *Montaje de los nodos en compartimentos protectores.*
    - *Montaje de la alimentación, pila 9,6 volts en ambos casos.*
    - *Configuración Xbee Shields, punto a punto.*

- *Diseño y búsqueda de Software.*

    - *Planificación de la programación.*
    - *Diseño de UML.*
    - *Búsqueda de librerías para LCD.*
    - *Búsqueda de librerías para Ethernet Shield.*
    - *Diseño Socket UDP para recibir datos en host remoto.*

- *Desarrollo Programación , Parte 1*

    - *Desarrollo Código Nodo emisor básico.*
    - *Desarrollo Código Nodo Receptor básico.*
    - *Implementación librería LCD.*

- *Desarrollo Programación , Parte 2.*

    - *Código Estructurado y ahorro de energía.*
    - *Implementación CRC.*
    - *Desarrollo de control de valores.*
    - *Configuración Xbee Shield, Coordinador y End Point.*
    - *Desarrollo de Bidireccionalidad, pequeño protocolo de comunicación.*
    - *Cajas para su almacenamiento.*
    - *Implementación de socket UDP y servidor para recepción UDP.*

- *Test de funcionamiento.*

    - *Programación Parte 1*
    - *Programación Parte 2*
    - *Test de funcionamiento en el exterior.*

- *Pruebas Funcionales.*

    - *Prueba de comportamiento durante 2 días , en un entorno apropiado para su objetivo.*

- *Memoria.*

    - *Documentación de todo el proceso.*

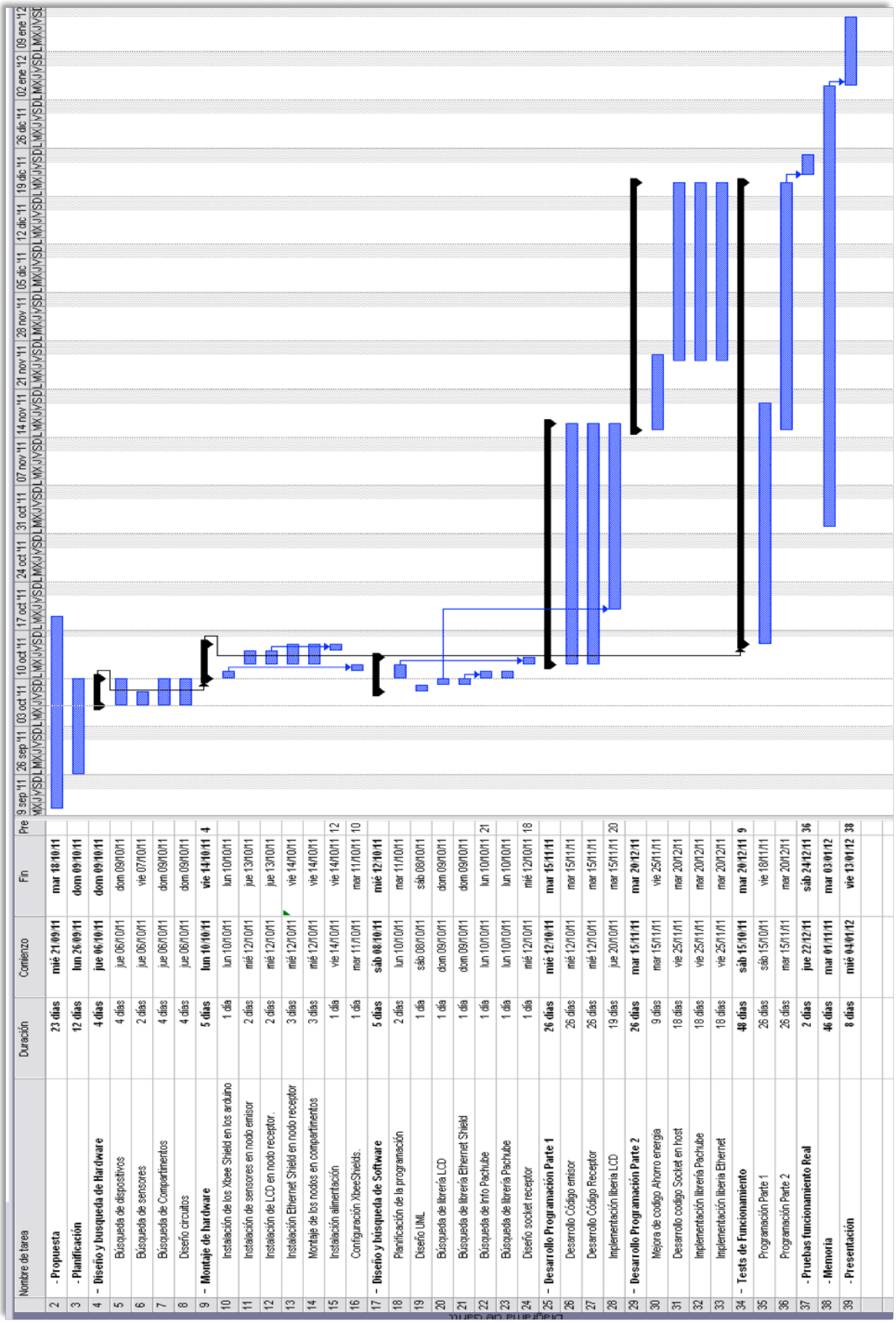
- *Presentación*

    - *Exposición completa del TFC.*

Diagrama de Gantt

En el siguiente diagrama, se realiza la representación temporal de las tareas expuestas en el apartado anterior. Como podemos observar, algunas de las tareas son solapadas ya que pueden ser ejecutadas al mismo tiempo, lo cual genera un aprovechamiento del tiempo. Del mismo modo algunas de las tareas necesitan de trabajos o tareas previas para ser iniciadas, por lo que no se podrán realizar hasta que sus predecesoras sean finalizadas.

Las fechas e hitos estimados en la programación inicial han tenido un concurrencia temporal correcta. No se han detectado grandes retrasos en cada una de las partes. Podemos resaltar como único hito no presentado temporalmente a tiempo la obtención de las carcasas que contienen los equipos.



	Nombre de tarea	Duración	Comienzo	Fin	Pre
2	**- Propuesta**	**23 días**	**mié 21/09/11**	**mar 18/10/11**	
3	**- Planificación**	**12 días**	**lun 26/09/11**	**dom 09/10/11**	
4	**- Diseño y busqueda de Hardware**	**4 días**	**jue 06/10/11**	**dom 09/10/11**	
5	Búsqueda de dispositivos	4 días	jue 06/10/11	dom 09/10/11	
6	Búsqueda de sensores	2 días	jue 06/10/11	vie 07/10/11	
7	Búsqueda de Compartimentos	4 días	jue 06/10/11	dom 09/10/11	
8	Diseño circuitos	4 días	jue 06/10/11	dom 09/10/11	
9	**- Montaje de hardware**	**5 días**	**lun 10/10/11**	**vie 14/10/11**	**4**
10	Instalación de los Xbee Shield en los arduino	1 día	lun 10/10/11	lun 10/10/11	
11	Instalación de sensores en nodo emisor	2 días	mié 12/10/11	jue 13/10/11	
12	Instalación de LCD en nodo receptor.	2 días	mié 12/10/11	jue 13/10/11	
13	Instalación Ethernet Shield en nodo receptor	3 días	mié 12/10/11	vie 14/10/11	
14	Montaje de los nodos en compartimentos	3 días	mié 12/10/11	vie 14/10/11	
15	Instalación alimentación	1 día	vie 14/10/11	vie 14/10/11	12
16	Configuración XbeeShields.	1 día	mar 11/10/11	mar 11/10/11	10
17	**- Diseño y búsqueda de Software**	**5 días**	**sáb 08/10/11**	**mié 12/10/11**	
18	Planificación de la programación	2 días	lun 10/10/11	mar 11/10/11	
19	Diseño UML	1 día	sáb 08/10/11	sáb 08/10/11	
20	Búsqueda de librería LCD	1 día	dom 09/10/11	dom 09/10/11	
21	Búsqueda de librería Ethernet Shield	1 día	dom 09/10/11	dom 09/10/11	
22	Búsqueda de Info Pachube	1 día	lun 10/10/11	lun 10/10/11	21
23	Búsqueda de librería Pachube	1 día	lun 10/10/11	lun 10/10/11	
24	Diseño socket receptor	1 día	mié 12/10/11	mié 12/10/11	18
25	**- Desarrollo Programación Parte 1**	**26 días**	**mié 12/10/11**	**mar 15/11/11**	
26	Desarrollo Código emisor	26 días	mié 12/10/11	mar 15/11/11	
27	Desarrollo Código Receptor	26 días	mié 12/10/11	mar 15/11/11	
28	Implementación libería LCD	19 días	jue 20/10/11	mar 15/11/11	20
29	**- Desarrollo Programación Parte 2**	**26 días**	**mar 15/11/11**	**mar 20/12/11**	
30	Mejora de codigo Ahorro energía	9 días	mar 15/11/11	vie 25/11/11	
31	Desarrollo codigo Socket en host	18 días	vie 25/11/11	mar 20/12/11	
32	Implementación librería Pachube	18 días	vie 25/11/11	mar 20/12/11	
33	Implementación libería Ethernet	18 días	vie 25/11/11	mar 20/12/11	
34	**- Tests de Funcionamiento**	**48 días**	**sáb 15/10/11**	**mar 20/12/11**	**9**
35	Programación Parte 1	26 días	sáb 15/10/11	vie 18/11/11	
36	Programación Parte 2	26 días	mar 15/11/11	mar 20/12/11	
37	**- Pruebas funcionamiento Real**	**2 días**	**jue 22/12/11**	**sáb 24/12/11**	**36**
38	**- Memoria**	**46 días**	**mar 01/11/11**	**mar 03/01/12**	
39	**- Presentación**	**8 días**	**mié 04/01/12**	**vie 13/01/12**	**38**

1.6. Recursos empleados

A continuación se detallan los equipos electrónicos, componentes y herramientas necesarias para el correcto desarrollo del proyecto:

Recursos a nivel de Hardware**:**

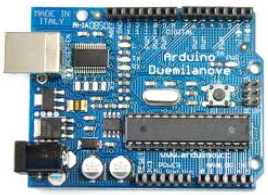
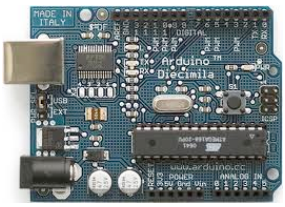
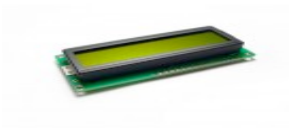
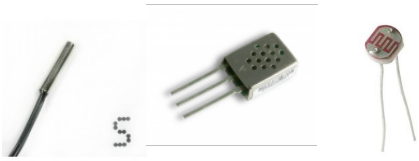
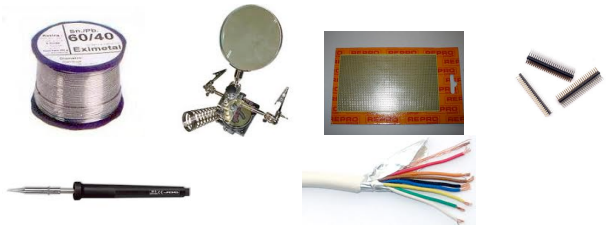
Arduino Duemilanove 	Arduino Diecemila 
Xbee Shield, integra protocolo 802.15.4 	Ethernet Shield, Protocolo UDP,TCP. 
LCD 16 * 2 caracteres Ref LMB162ABC 	Sensores utilizados Temp Ref SEN118A2B, Ref 808HV5 y LDR 
Material y componentes electrónicos varios 	Cajas para componentes electrónicos 

Figura 7, recursos Hardware empleados.

Recursos a nivel de Software:

- **Entorno de programación de Arduino**, el cual ha permitido desarrollar, comprobar, compilar y cargar los programas o sketch a los equipos Arduino mediante el puerto USB. Así como la comunicación serial y acceso a la configuración directa de los módulos Xbee Shield.

- **Librería *LiquidCrystal***, la cual dota a la plataforma de una facilidad para poder realizar funciones de escritura, borrado de la pantalla, encendido y todas las funcionalidades necesarias de una manera muy intuitiva.

- **Librería Ethernet y Liberia SPI,** habilita las funcionalidades a la interface Ethernet Shield instalada en Arduino.

- **Librería UDP,** utilizada para la implementación del socket UDP en el nodo Coordinador soportando así el protocolo UDP.

- **Socket UDP Servidor**, derivado de una práctica realizada por el estudiante en 2010, asignatura protocolos Aplicaciones de Internet en UOC.

- **Librería AVR,** permite al Arduino opciones de control de energía y optimización de utilización de los procesos.

- **Librería Math,** utilizada en los cálculos complejos de los datos obtenidos en los sensores de humedad y temperatura.

- **Código CRC,** permite generar un valor lógico de los datos enviados y una comparación en el destino de los mismos para su correcta certificación.

- **Código de Conversión de datos,** la Liberia estándar de Arduino no contempla el envió de variables con coma, por lo que se han tenido que utilizar diferentes técnicas para el envió de los mismos.

Emplazamientos de prueba:

- Prado de siega en Gijón, Asturias, España, Coordenadas 43.504293, 5.678891.

1.7. Productos

Como resultado del proyecto se ha obtenido una estación meteorológica formada por dos equipos:

Nodo coordinador cuenta con una pantalla LCD que permite la lectura de datos, un pulsador para la variación del tiempo de medición, interruptor para conexión y desconexión, envió de datos a través de Ethernet, un puerto USB y entrada de alimentación para fuente externa.

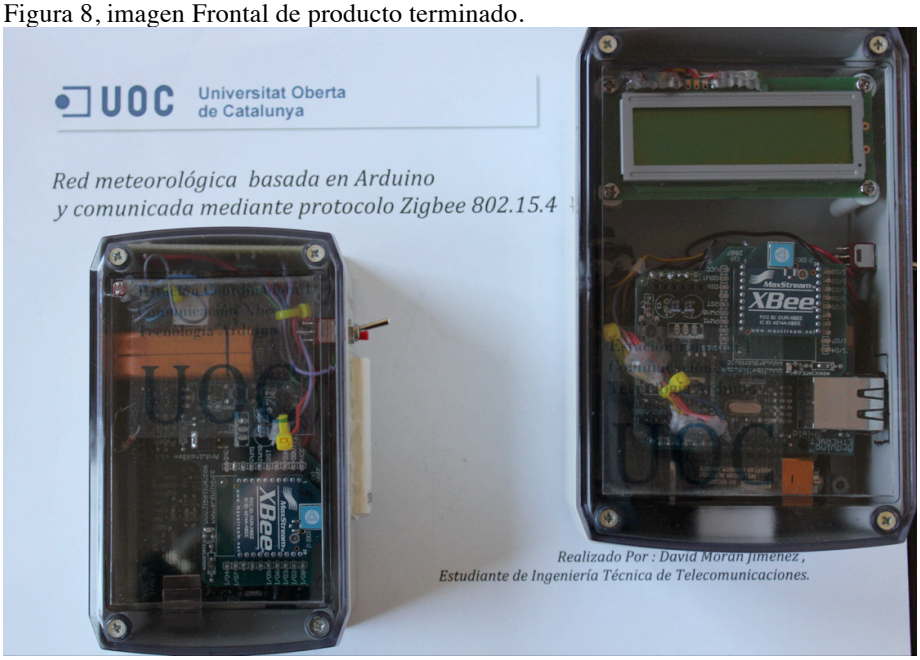
Nodo Emisor es el encargado de realizar las mediciones pertinentes y enviárselas al Coordinador, está compuesto por los sensores, una radio Xbee, un puerto USB y una batería para alimentar al mismo. Además cuenta con un pulsador para el reseteo de la estación así como de un interruptor de conexión y desconexión.

El sistema está preparado para soportar 3 nodos emisores los cuales pueden aportar lecturas de diferentes espacios. Para ello no será necesario ningún tipo de configuración en el equipo coordinador, solo hará falta la fabricación de dos nodos emisores restantes y pequeños cambios a nivel de programación en los nodos emisores (ID correspondiente a cada uno).

Como ha sido comentado anteriormente el sistema cuenta con una interface Ethernet que permite enviar los datos obtenidos mediante un Sokect UDP al exterior de la red, ya se internet o una red local.

A continuación se muestra dos imágenes de los equipos obtenidos:

Figura 8, imagen Frontal de producto terminado.



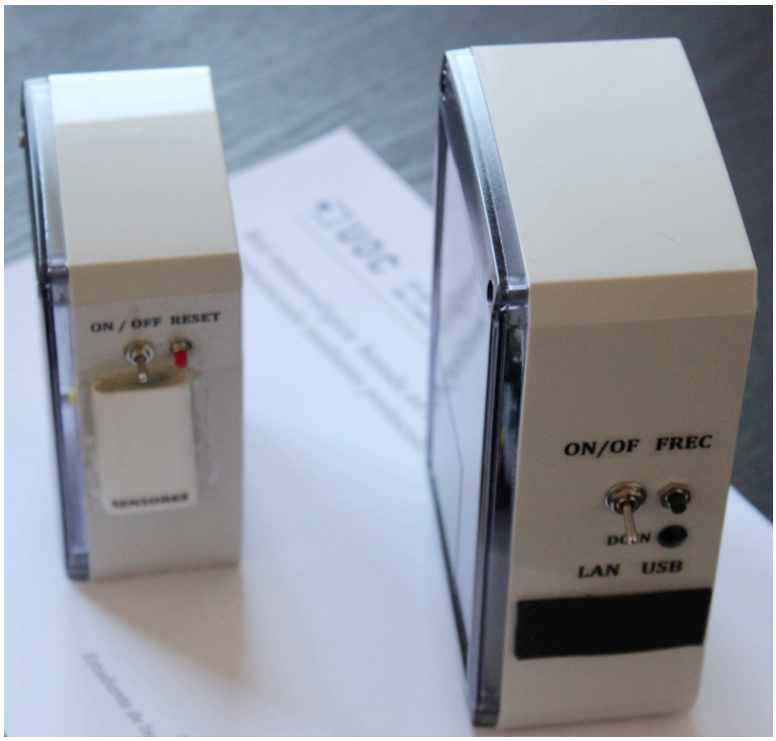


Figura 9, conexiones e Interfaces del producto Terminado.

2. Antecedentes

En los dos capítulos siguientes se realizará una introducción a la tecnología Arduino en el ámbito artístico caracterizado como una potente herramienta de desarrollo cultural, instructivo y formativo. También se realizará una valoración del estudio de mercado de la tecnología utilizada, así como un breve comentario acerca de las diferentes tecnologías existentes.

2.1. Estado del arte

Arduino es una herramienta utilizada por artistas audiovisuales y electrónicos (Arte media). Existen multitud de proyectos entorno a dicha plataforma desde satélites, robots, equipos audiovisuales, instrumentos musicales controlados a través de Arduino. Dado que es una plataforma económicamente asequible, openSoftware y openHardware permite a los artistas tener una herramienta de trabajo de bajo precio (30 euros aproximadamente), la cual cuenta con fuentes de información en comunidades destinadas al desarrollo. Numerosas instalaciones artísticas están dotadas hoy de esta tecnología, remarcando piezas que permiten a los visitantes interactuar o intervenir en el funcionamiento de las obras.

También se ha de destacar que Arduino está basado en un lenguaje de programación llamado wiring, derivado del lenguaje C, resultando así que el aprendizaje no requiera de conocimientos de ingeniería industrial o electrónica para la programación. Todas estas características hacen que dicha plataforma esté siendo utilizada en centros de arte para la realización de workshops de electrónica e instalaciones artísticas. La enseñanza es un campo que se está abasteciendo de las características del dispositivo, centros de educación secundaria cuentan con placas Arduino para instruir a los alumnos en introducción a la electrónica.

Los protocolos y sistemas de comunicación utilizados para el proyecto resultan igual de atractivos, se han utilizado para la transmisión inalámbrica equipos Xbee Shield, los cuales implementan el protocolo Zigbee, estos cuentan con un hardware que se acopla directamente en el Arduino sin ningún tipo de adaptación o similar. Zigbee es un protocolo caracterizado por su bajo coste tanto económico como energético, además de poder ser usado en distancias de entre 30 y 100 metros. Para la comunicación cableada hacia al exterior se ha utilizado Ethernet, usando el protocolo UDP, dotando también al sistema de sencillez ya que no requiere conexión como es en el caso del protocolo TCP.

Existen en el mercado multitud de alternativas hoy en día a Arduino sobre todo después de la aparición de este, van desde la construcción propia de una plataforma casera con un PIC AVR hasta sistemas como Pingüino el cual cuenta con PIC de microchip. PowerJaguar también es una plataforma de aprendizaje que como Pingüino esta compuesta de PIC de Microchip en lugar de AVR. Aunque existen varias, estas dos últimas son prácticamente clones de Arduino en lo que a diseño o estructura se refiere. Son utilizadas de manera habitual por Artistas ya que pueden resultar incluso más baratas que la plataforma escogida para la realización de este proyecto, o como no la plataforma utilizada en UOC TinyOS la cual ya es abalada por una larga trayectoria de proyectos.

2.2. Estudio de mercado

La decisión más importante a la hora de abordar el proyecto ha sido la poca existencia de equipos basados en Arduino que funcionan como estaciones meteorológicas independientes. En el mercado se pueden encontrar incluso equipos comerciales Arduino que actúan como redes Mesh, donde varios nodos envían datos meteorológicos a un nodo Router o coordinador. Estos nodos coordinador son siempre acompañados de un equipo informático el cual hace las veces de interprete.

Para este proyecto se ha pensando la idea de darle más capacidad de procesamiento al nodo receptor, siendo este un estación que permite la lectura de datos, la bidireccionalidad e incluso la interactuación con los nodos despertándolos o durmiéndolos. Además el equipo como se ha comentado cuenta con la capacidad de enviar datos al exterior brindando al proyecto de ser completamente autónomo y no dependiente de un equipo informático. Incluso capaz de realizar comparaciones CRC validar si los datos han sido correctamente enviados.

Arduino fue creado por dos personas, destacando a David Cuartielles. En un primer momento Arduino era una placa con un chip Atmel el cual no contaba con conexión USB, pero que ya tenía entradas y salidas analógicas y digitales. Posteriormente se le doto al Arduino de conectividad USB con un chip FDDI, y en la actualidad se pueden encontrar equipos Arduino que cuentan con chips de radio Xbee integrados en el propia placa, como es el caso del Arduino Nano.

Los equipos desarrollados tienen un destino multidisciplinar, pueden ser utilizados como una estación meteorológica doméstica, industrial, agraria, forestal, etc. Esta permite un número bastante grande de escenarios. Todo ello es debido a la facilidad de uso y su posibilidad de envió fuera de la red hacen al equipo muy flexible. También se ha de destacar que el equipo puede modificarse fácilmente para actuar como una herramienta de sistema empotrado totalmente distinta.

Otra característica fundamental a la hora de valorar la toma de decisión de escoger la plataforma de trabajo ha sido la consolidación de esta en el mercado. Pingüino o PowerJaguar son alternativas que están resultando de un potencial creciente pero que todavía no tienen un calado suficiente en las comunidades de desarrollo. Ambas son hardware y software libre pero sus fuentes de consulta no pueden ser comparadas a las fuentes existentes para Arduino así como todas las librerías o accesorios de mercado. Podría resultar ventajoso utilizar las dos plataformas citadas en el caso de reaprovechar código escrito para PICS los cuales podrían ser utilizados si de esta elección se tratase.

3. Descripción funcional

La base del sistema se ha desarrollado en torno a dos equipos configurados en modo Coordinador y punto final. Para ello se han configurado los módulos Xbee Shield como tal, donde el nodo coordinador tiene la capacidad de gestionar en este momento hasta 3 nodos punto final permitiendo variar su número de equipos modificando el número de nodos máximo en el equipo coordinador y asignado un nuevo numero ID a los nodos punto final.

Se ha decidido encastrar a los equipos dentro de unas cajas para componentes electrónicos, sin ser totalmente estancas, aseguran que los nodos no sufran daños por agentes externos. Se ha procedido al etiquetado de los pulsadores, puertos USB, LAN e interruptores para facilitar la operatividad de los equipos.

Los equipos funcionan con baterías de 9 voltios, aunque el nodo coordinador debe de estar conectado a la red eléctrica ya que su operatividad es alta y el módulo Ethernet Shield produce un gasto de energía extra. En caso de fallo o corte de luz este seguirá funcionando con la batería instalada sin tener que reiniciar los equipos o realizar cualquier tipo de conmutación con el interruptor.

3.1. Sistema total

El sistema tiene una función de estación meteorológica la cual cuenta con un Nodo Emisor o punto Final, Nodo Coordinador o receptor, y Host remoto.

Nodo emisor realiza el trabajo de cálculo, generador de valor CRC y conversión de los datos obtenidos para pode equilibrar la carga computacional al nodo receptor, este está provisto de los sensores de humedad, temperatura y luz, además de la radio Xbee. Como se ha comentado en el apartado anterior el equipo emisor funciona con una batería de 9 voltios, permitiendo también a este si fuese necesario trabajar conectado a la red eléctrica.

Nodo receptor realiza tareas de gestión de energía despertando al nodo emisor cuando precisa de un dato, además también cuenta con una implementación de redundancia cíclica, la cual corrobora que la recepción de datos no contienen errores ni datos inesperados. Este equipo cuenta con una pantalla de 32 caracteres para la presentación de datos además de una tarjeta Ethernet para realizar envíos fuera de la red. Tiene además la capacidad de variar los intervalos de medición sin necesidad de ninguna modificación de código solamente a través de un pulsador se permite cambiar la frecuencia con el que el equipo toma los datos, siempre al principio de la comunicación. Se aconseja que el equipo este conectado mientras esté operativo a la red eléctrica, utilizando la batería de 9 voltios para salvaguardar

fallos o cortes de suministro eléctrico.

Host Remoto con sistema operativo basado en Linux, el cual una vez ejecutado un socket UDP servidor, empezará a recibir datos automáticamente si el sistema está funcionando.

3.1.1. Diagrama de bloques de la aplicación

A continuación se presenta un diagrama que muestra una visión general del funcionamiento de los equipos:

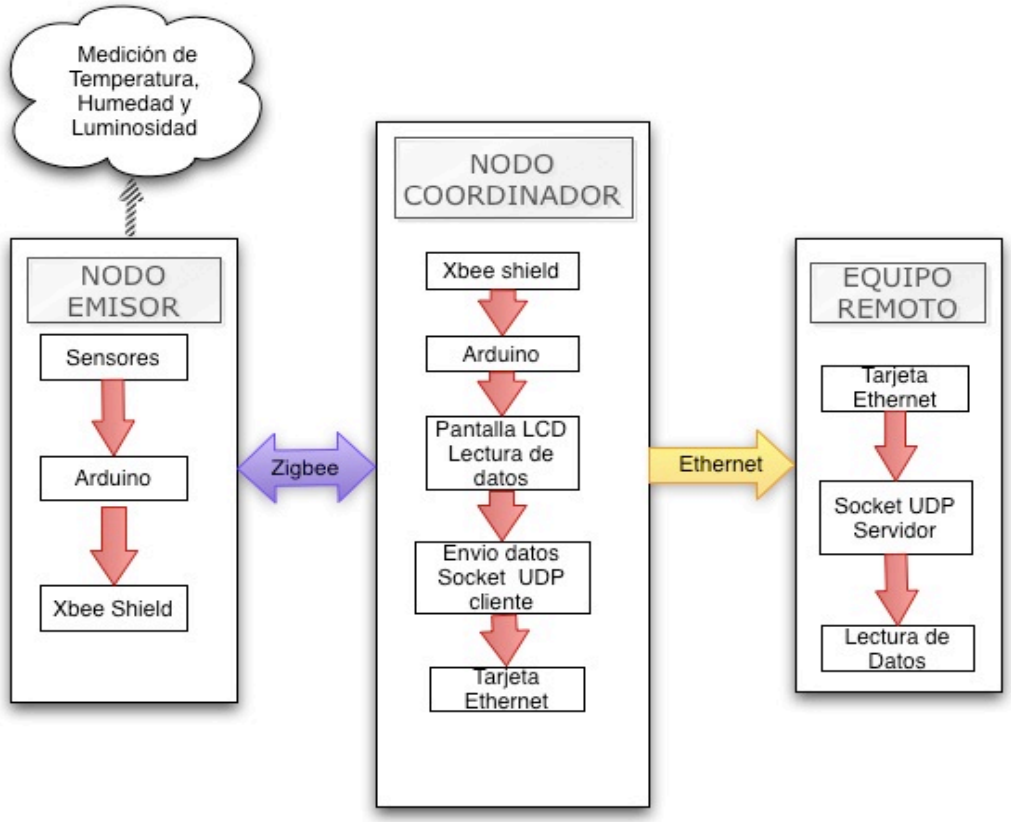


Figura 10, diagrama de bloques.

La figura anterior muestra el medio de interacción entre los tres equipos, por un lado protocolo 802.15.4 existente en la comunicación de las motas y Ethernet entre el nodo Coordinador y el equipo remoto.

3.1.2. Definición de la red

La red de equipos está compuesta por un nodo Coordinador y hasta tres nodos configurados como punto final. Como se ha descrito en apartados anteriores, el equipo Coordinador cuenta con la capacidad de enviar los datos al exterior de la red Zigbee vía Ethernet. Para este caso se cuenta con un socket udp cliente configurado en Arduino y socket udp servidor, desarrollado para entornos Linux.

Para la configuración de red sobre protocolo 802.15.4 se ha utilizado el software X-CTU dispuesto por MaxStream, fabricante de los módulos de radio Xbee, el cual permite realizar test y configuración de radio módems. Para realizar dicha configuración se han programado los equipos con un Identificador personal de red o PAM, común para ambos Xbee coordinador y punto final. El PAM dispuesto en los equipos, define los nodos pertenecientes a una red mediante un ID concreto, para este caso se ha configurado un PAN ID:3332.

Los modos de operación dispuestos en los nodos corresponden a modo coordinador y modo punto final o End Device. El modo Coordinador es el encargado de formar la red, los puntos finales se conectan a él y pudiendo este realizar funciones de enrutamiento entre ellos si fuese necesario. Los puntos finales no pueden comunicarse con otros nodos directamente, estos han de comunicarse hacia un nodo Coordinador. Esta comunicación simple hace que el nodo punto final tenga un coste energético mucho menor que el nodo Coordinador.

En la figura siguiente se muestra la configuración del nodo punto final sobre X-CTU, donde es posible apreciar un PAN ID 3332 y un valor 0 en el modo de operación como Coordinador, indicando así que no esta trabajando en dicho modo.

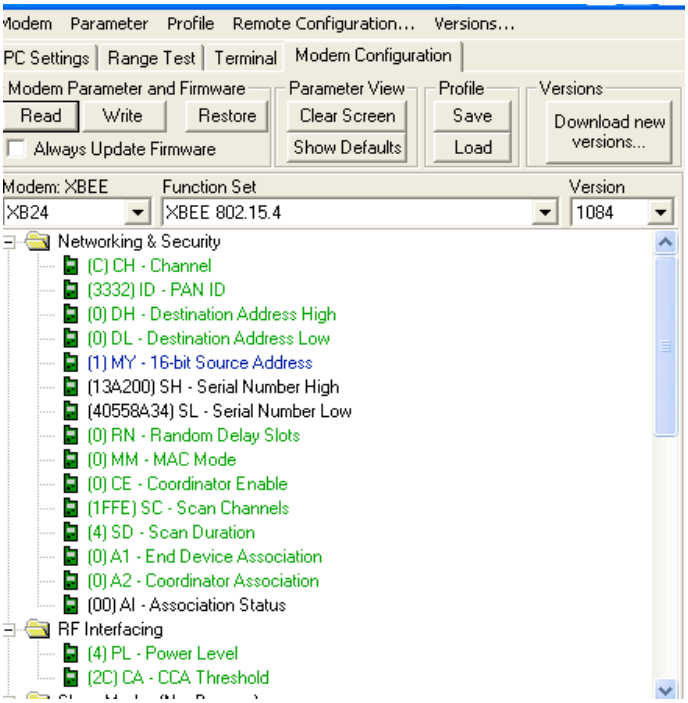


Figura 11, configuración Xbee como end point.

A continuación se muestra la configuración del nodo Coordinador, con un PAN ID 3332 igual que el nodo punto final, y con un valor 1 en modo Coordinador, habilitando a este trabajar como coordinador:

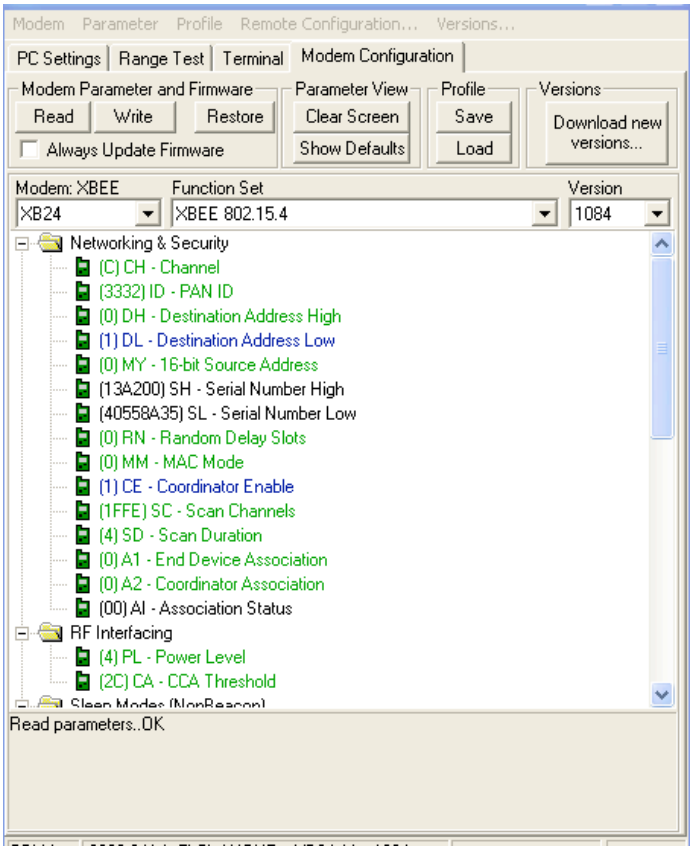


Figura 12, configuración Xbee como coordinador.

El envió de datos a través de Ethernet se ha configurado con una operatividad dentro de una red local, bajo protocolo UDP. Se ha utilizado para este fin parte de un código incluido en la biblioteca de ejemplos de Arduino, la cual está disponible en el entorno de programación del mismo. El direccionamiento de red correspondiente a este socket udp cliente está formado por la dirección IP 192.168.1.177 correspondiente al Ethernet Shield de Arduino, la dirección IP 192.168.1.50 como dirección del socket servidor. La comunicación se realiza en el puerto 8888 udp local , y puerto 6000 udp remoto.

3.1.3. Interacción de lo de diferentes objetos del sistema

La interacción de los diferentes bloques aplicativos resulta diferente en cada uno de los procesos. Algunos componentes tienen una capacidad bidireccional y otros unidireccional. En el caso de las placas Xbee Shield incorporan dos jumper que permite variar el tipo de configuración, USB- XBEE o XBEE-XBEE, para este caso ambos son configurados en el último modo.

Los datos son escritos por el programa o Sketch cargados, sobre el serial de Arduino el cual automáticamente transfiere la información al Xbee. Como es posible observar, el método para enviar datos vía radio es muy sencillo. Si fuese necesario escribir datos manualmente desde el monitor Serial que Arduino incorpora debería de hacerse un cambio al modo USB-XBEE.

El puerto USB tiene la capacidad de transferir el programa desarrollado, monitorizar o interactuar vía serial con los equipos, además de procurar energía al Arduino. Los interruptores o pulsadores están directamente conectados a las entradas disponibles en la placa, la actuación de estos está definida en el Sketch o programa cargado. Del mismo modo, los sensores están conectados a las entradas analógicas de la placa, los lecturas de estos están definidas el software diseñado.

A continuación se muestra gráficamente la interacción de las siguientes partes del sistema en el nodo Emisor.

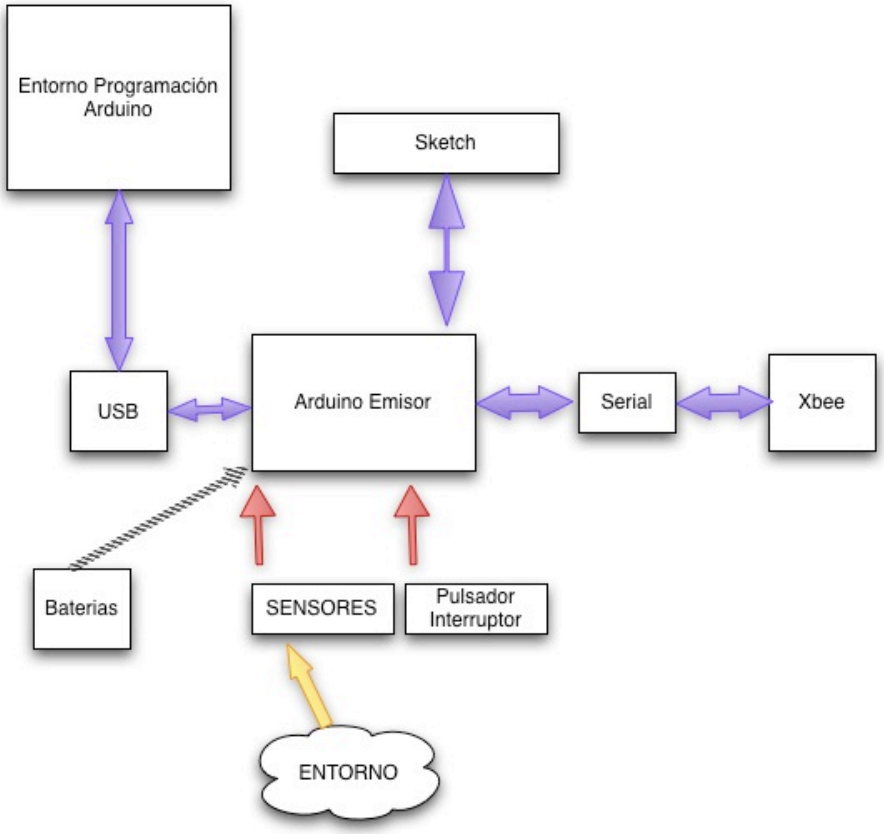


Figura 13, interacción Arduino Emisor.

Del mismo modo que en el nodo emisor, el nodo coordinador interactúa con las diferentes partes del sistema USB, SKETCH, SERIAL y XBEE. Además como se ha comentado este equipo cuenta con una tarjeta Ethernet, la cual envía los datos a través de la red cableada. Para este proyecto no se ha dotado al equipo la posibilidad de recibir datos a través de dicha interface, la cual podría ser habilitada.

La interface LCD está conectada a las salidas digitales que Arduino dispone, desde el Sketch desarrollado se imprimen por pantalla las variables obtenidas del entorno.

Esquema gráfico de la interacción entre los diferentes módulos del nodo Coordinador:

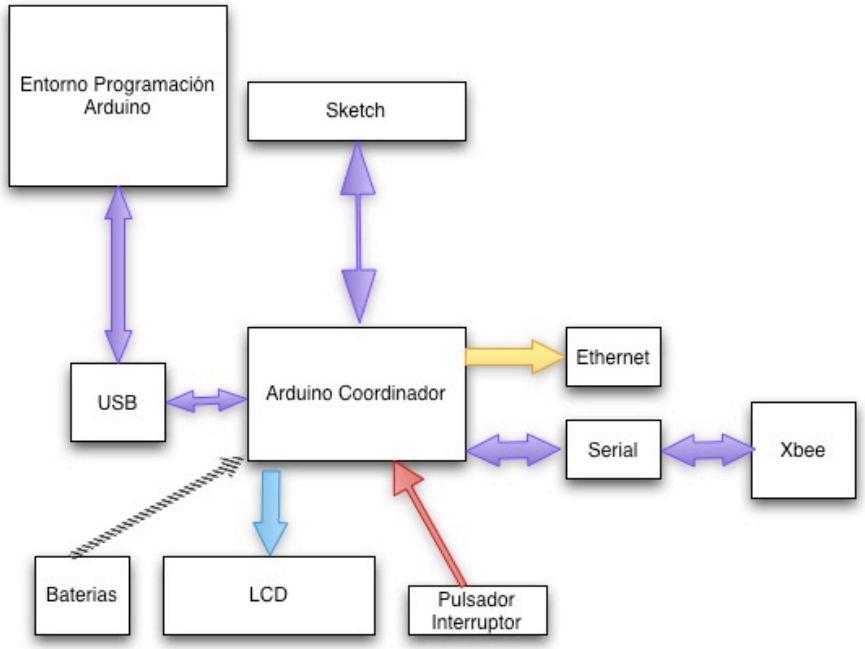


Figura 14, Arduino Coordinador.

3.2. Diseño interface de usuario

Dado que la filosofía seguida durante el desarrollo del proyecto consiste en crear un sistema no vinculado a un pc no se dispone de software o aplicación funcional en un ordenador. La interface de usuario desarrollada está relacionada con la lectura de datos en la pantalla LCD del nodo Coordinador. Como podemos comprobar a continuación, la disposición de los elementos en la pantalla son los siguientes:

- ID el cual especifica la identificación del nodo al que hace referencia la lectura.
- T perteneciente a la temperatura el nodo emisor.
- H humedad relativa en %.
- L cantidad de luz en % siendo 0% oscuridad total y 100 % luz total.

En la siguiente figura se representan los datos citados:

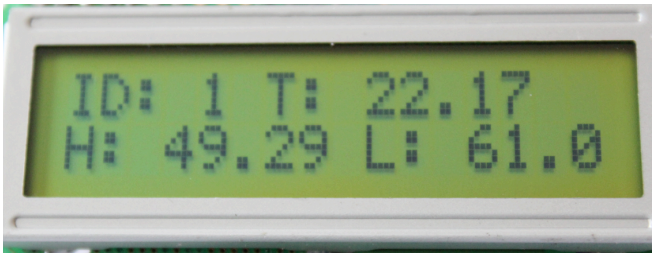


Figura 15, disposición datos LCD.

El socket UDP servidor muestra una salida de valores similar a los descritos en la interface LCD del nodo Coordinador. ID, T, H y L son las variables enviadas, además de indicar si los emisores están apagados, es indicado también el intervalo de tiempo entre lecturas correspondiente al la frecuencia seleccionada al inicio del sistema, se muestra figura socket servidor en entorno Linux.

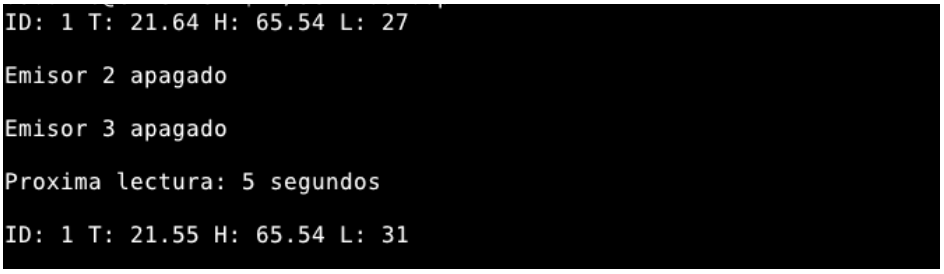


Figura 16, disposición datos Socket Servidor UDP.

3.3. Mota

En este apartado se realiza una descripción de las diferentes partes con las que cuenta Arduino así como de sus principales Características.

Descripción de placa Arduino Duemilanove:

- 14 entradas y Salidas Digitales.
- 6 entradas Analógicas.
- Botón de Reset.
- Conexiones eléctricas de entrada y salida, voltaje de funcionamiento 5 voltios.
- Voltaje de entrada entre 7 y 12v , soportando valores inferiores.
- Led asignado al pin13.
- Leds de Transmisión y Recepción.
- Memoria Flash 16KB.
- Micro controlador ATMega168.
- Pines Seriales.
- Puerto USB.
- Comunicación Vía serie a través de los pines TX y RX.
- Capacidad para detectar interrupciones externas en los pines 2 y 3.

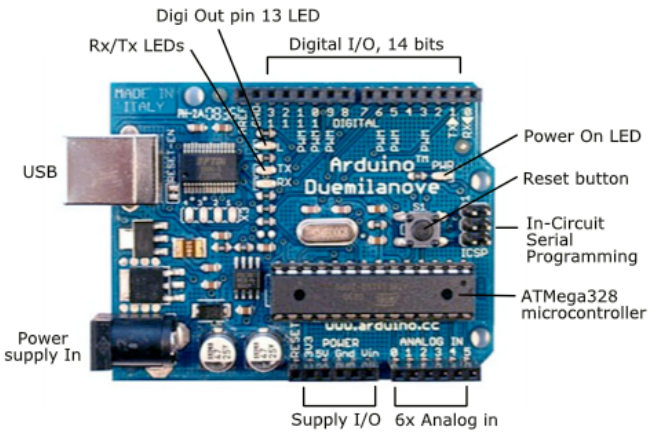


Figura 17, características placa Arduino.

Para el desarrollo del proyecto se han utilizado entradas analógicas en el caso de los sensores, salidas digitales para la conexión del LCD y pulsadores. Además de detectar interrupciones para despertar los equipos. Los lenguajes de programación en los que está basados la plataforma son wiring para el código, el cual a su vez está basado en lenguaje C y Processing para el entorno de desarrollo. Estos permiten la comunicación con otras plataformas, como puede ser Flash, Pure Data, etc.

4. Descripción detallada

En este apartado se muestra la funcionalidad de los dos nodos emisor o punto final y Coordinador, se acompaña para su comprensión dos diagramas de flujo que realizan una aproximación al funcionamiento del sistema.

Para la comunicación de los equipos se ha desarrollado un pequeño protocolo de comunicación basado en la detección de mensajes. Los equipos tienen la capacidad de distinguir si el dato va destinado a él, además de tener un control sobre un envió o reenvió.

Los diferentes caracteres enviados como mensajes son los siguientes:

X -> Despierta todos los nodos, señal genérica.

S-> Pide un primer envío de datos.

D-> Envía datos indicando que es un primer envío.

R -> Realiza petición de reenvío de datos.

E -> Envía datos indicando que es un Reenvío.

O -> Señal para dormir nodos.

Se acompaña figura para una mejor comprensión:

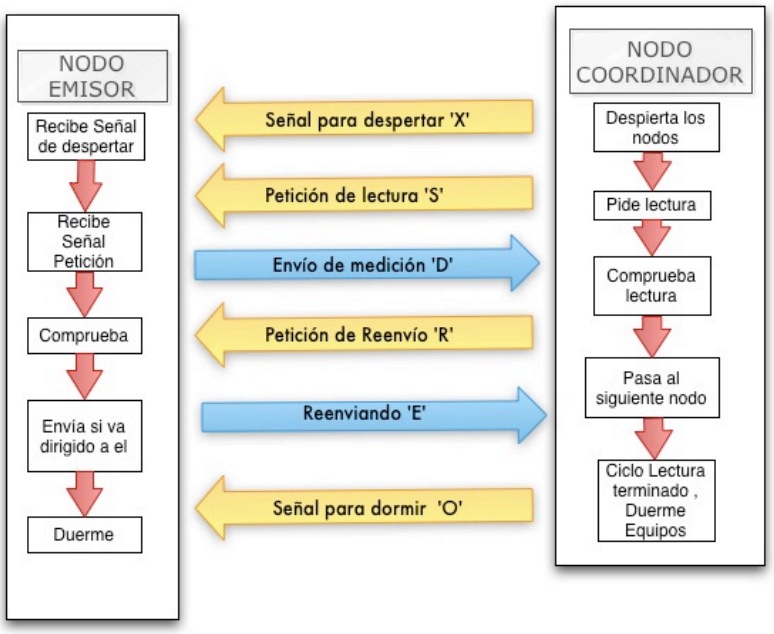


Figura 18, protocolo de mensajes desarrollado.

A continuación se describe el funcionamiento explícito de los equipos, así como de la aplicación del sistema de control comentando anteriormente:

Nodo Emisor

Está compuesto por la estación configurada en modo punto final que consta de los sensores de humedad, temperatura y luminosidad además de la radio Xbee que dota al nodo una capacidad de envío de datos a la estación coordinadora. El equipo pasa por los siguientes estados durante su funcionamiento:

1º El nodo es accionado utilizando el interruptor On/off, este queda a la espera de realizar la primera transmisión, sin pasar a modo Sleep o modo dormir. Es posible ver cuando el Arduino está operativo computacionalmente, el pin13 tiene un led integrado en el propio dispositivo, el cual indica el estado de Arduino. Si el led parpadea Arduino está en funcionamiento, si el led tiene una posición fija ya sea apagado o encendido, Arduino está en modo Sleep o dormido.

Figura 19, led estado Nodo Emisor.

2º El nodo coordinador enviará una señal de broadcast para despertar a todos los nodos participantes en la comunicación en cada ciclo de lectura, como se ha comentado si se está en el primer ciclo, el nodo emisor ya estaría encendido. La señal se corresponde con la letra 'X', una vez el nodo punto final recibe una señal X, la radio Xbee realiza una interrupción al puerto serie que activará al Arduino. Se realiza entonces una lectura de serial hasta recibir una petición que coincida con su Id (número de nodo), compuesta dos caracteres 'Sx" siendo x el numero id del nodo, S1 en el caso del nodo 1. Una vez recibida se realizará la medición y se calculará el valor CRC correspondiente a las variables.

3º Envió de datos a través de Xbee Shield.

El formato de envió de mensaje es el siguiente: ID NODO, TEMPERATURA, HUMEDAD, LUZ, CRC. Es posible observar en la siguiente captura de pantalla un envío a través del puerto serie, monitorizado con la interface de programación de Arduino. El nodo emisor envía en el primer ciclo una medición correspondiente a los siguientes valores: D1, correspondiente a nodo 1; 19,89 º correspondiente a temperatura; 73,3 % Humedad ; 65 % de luz, y 2569888533 valor CRC de los datos enviados.

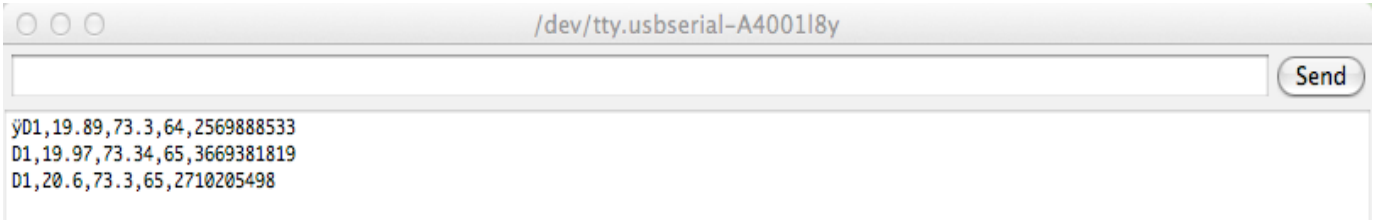


Figura 20, captura serial Nodo Emisor.

4º Si la comunicación ha resultado correcta, el nodo pasará a modo Sleep en el momento que se realice la petición por parte del coordinador al siguiente nodo. Si no ha resultado correcta el nodo Coordinador realizará la petición del dato nuevamente. Dado que el nodo despierta cada vez que un dato es recibido por radio, se genera una interrupción, aunque la petición de dato no vaya dirigida a él. Cada vez que se reciba una señal, el nodo emisor leerá si le pertenece y si no Arduino entrará en modo Sleep. Por ejemplo, nodo 1 recibe petición de nodo 2, en ese momento el nodo 1 analizará que no va a dirigida a el y se dormirá.

5º Al final de un ciclo de lectura, es decir: Petición al nodo 1, 2 y 3, el nodo Coordinador enviará la señal 'O' la cual realizará en todos los nodos un Sleep hasta el próxima petición. Asegurando así que ningún nodo ha quedado despierto entre las peticiones.

Se acompaña diagrama de flujo que representa los datos anteriores:

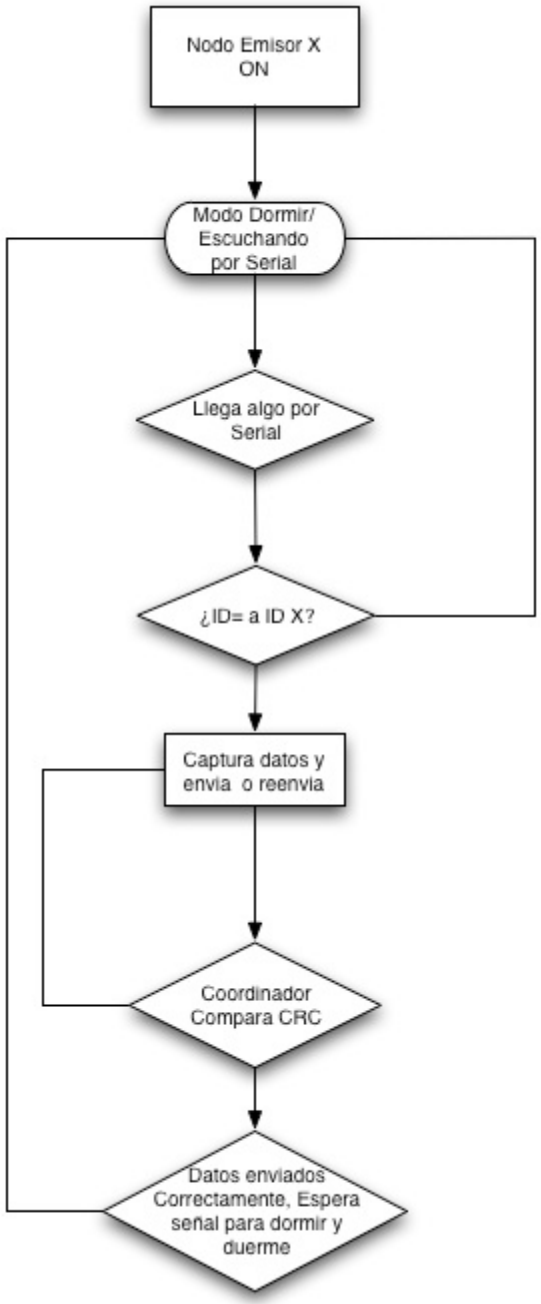


Figura 21, diagrama de Flujo Nodo Emisor.

Nodo Coordinador

El nodo Coordinador realiza diferentes acciones: petición de datos a los diferentes nodos emisores, verificación de datos recibidos, presentación de los datos por pantalla, control para dormir a todos los nodos una vez se ha terminado un ciclo y envío de datos UDP a través de Ethernet Shield.

Para concluir las funcionalidades citadas, el sistema sigue las siguientes fases:

1º El Nodo Coordinador es accionado a través de la palanca ON/OFF.

2º El Nodo nos pide que seleccionamos el intervalo de medición disponible: 5 segundos, 5 minutos, 30 minutos o 1 hora.

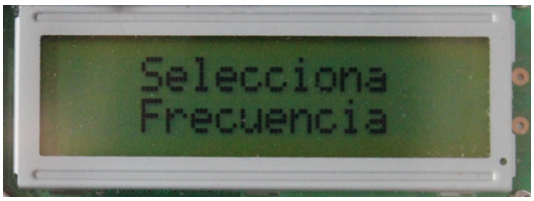


Figura 22, selección Intervalo de medición.

3º Se envía una Señal de broadcast para despertar a todos los equipos, correspondiente al carácter 'X', generando así la espera por serial de la petición de datos en los diferentes nodos emisores e impidiendo que algún nodo esté durmiendo y no reciba la petición de dato.

4º Durante un ciclo de petición, el nodo coordinador realiza el envió a todos los nodos punto final o emisores. Siendo S1 para nodo id=1, S2 nodo id=2, S3 nodo id=3. Entre cada petición existe un tiempo de espera de 15 segundos, una vez finalizado este, el nodo se dará por apagado.

A continuación, se muestra una captura de pantalla del nodo Coordinador, donde se puede apreciar como este envía una señal de broadcast X y realiza la petición a los nodos 1, 2 y 3 con S1, S2 y S3. Una vez terminadas las peticiones, se envía la señal 'O', la cual dormirá a todos los nodos hasta el próximo ciclo. Se ha de remarcar, que existe una espera de 15 segundos para recibir los datos de un nodo, la frecuencia seleccionada manualmente al inicio de la comunicación se aplica en la finalización de cada ciclo, es decir S1, S2, S3 (FRECUENCIA DE LECTURA SELECCIONADA), S1, S2, S3 (FRECUENCIA DE LECTURA SELECCIONADA), etc.

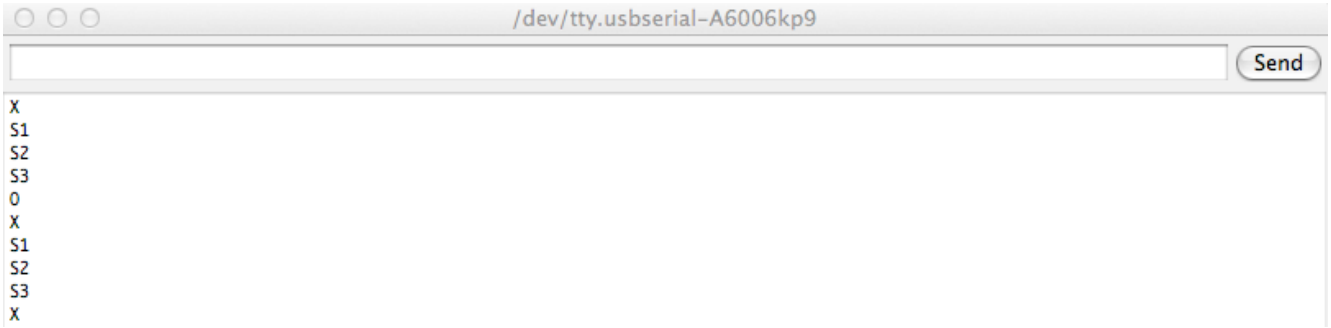


Figura 23, captura Serial Nodo Coordinador.

5º Si el nodo X no contesta, la estación Coordinadora dará al nodo por apagado y continuará al siguiente nodo, el LCD mostrará Emisor X apagado.

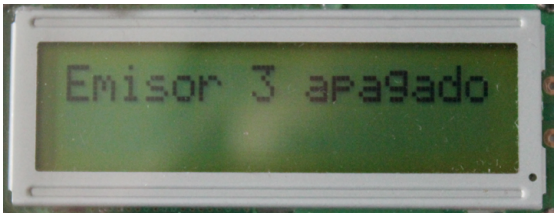


Figura 24, captura Emisor Apagado.

6º Si existe recepción de datos se realizará una comprobación del valor CRC enviado junto a la transmisión de valores, si este no resulta correcto se realizará la petición de datos una vez más y si se vuelve a recibir erróneamente se continuará al siguiente nodo. El nodo Coordinador mostrará por LCD: Dato error. En la siguiente imagen se ha modificado manualmente en el nodo Emisor el dato CRC enviado, provocando un error en la recepción y comprobando que nodo el Coordinador certifica que ha ocurrido un error:

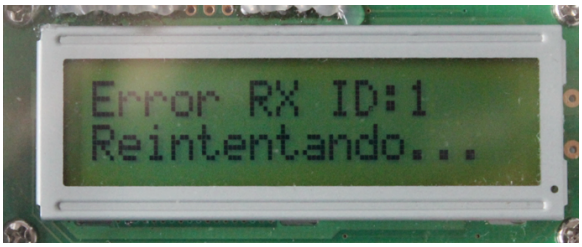


Figura 25, error de CRC.

7º Si todos los datos son correctos, se mostrará por el LCD: ID:X, T:X, H:X, L:X, es decir, Valor ID, Valor Temperatura, Valor Humedad y Valor Luz. Además, se ha especificado un rango numérico para comprobar si un dato llega por debajo o por encima de unos valores concretos, advirtiendo vía LCD: Dato Anómalo.

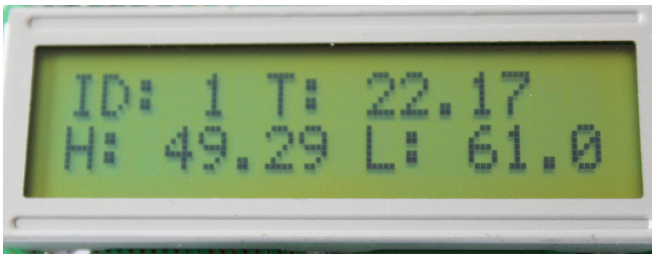


Figura 26, lectura Correcta de datos.

A continuación se ha forzado manualmente la variable temperartura igual a 110º en el nodo emisor para provocar un dato anómalo, y constatar que el nodo Coordinador ha podido certificarlo.

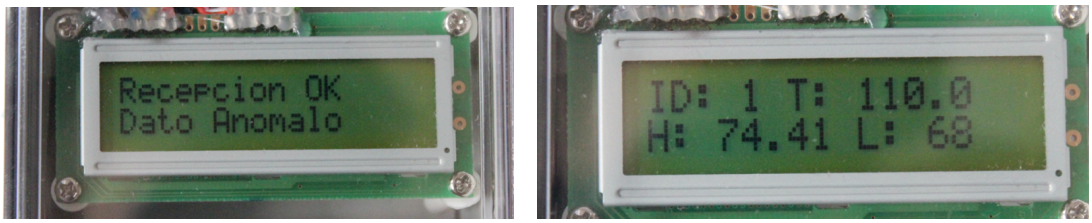


Figura 27, muestra de dato Anómalo.

8º El nodo Coordinador envía la señal "O" para dormir los nodos emisores y se transmiten los datos vía Ethernet a través de un socket UDP cliente / servidor. Los datos son recibidos en un entorno Linux con el socket servidor.

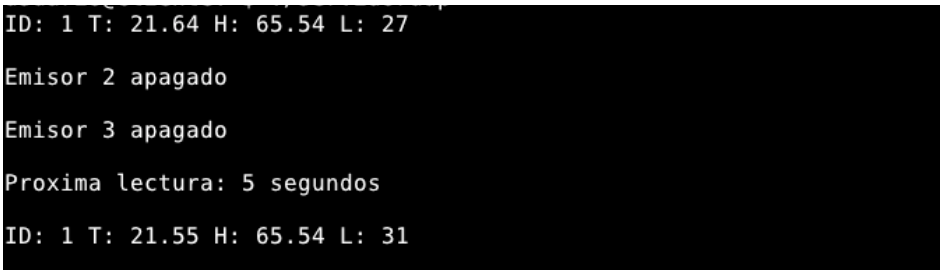


Figura 28, recepción de datos Socket UDP servidor.

9º Se realiza la petición a los 3 nodos y se espera el tiempo seleccionado en el intervalo de medición para volver a solicitar un dato.

Se acompaña figura que representan los datos anteriores:

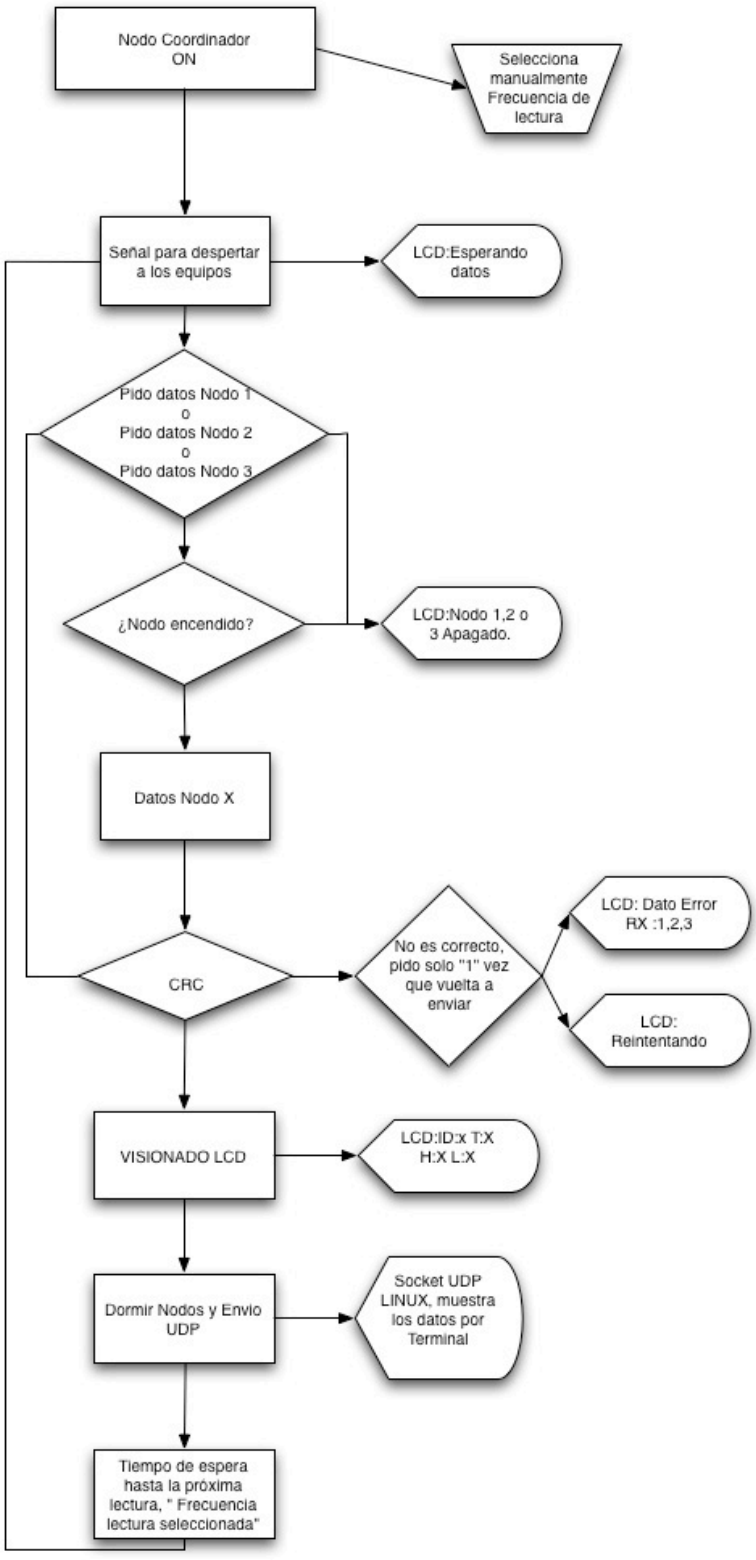


Figura 29, diagrama Flujo Nodo Coordinador.

5. Transmisión.

El sistema está dotado de comunicación vía Zigbee, además de una comunicación vía Ethernet. En este apartado se definirán las principales características con las que cuentan los protocolos utilizados en la transmisión de datos:

- Comunicación vía Ethernet.

Para la comunicación Vía Ethernet se ha decidido implementar el protocolo de comunicación UDP no orientado a conexión. Dicho protocolo está encapsulado dentro del Protocolo IP tal y como está definido dentro del modelo OSI. La decisión de utilizar dicho protocolo es la no necesidad de utilizar una conexión constante tal y como ocurre con el protocolo TCP, esto permite la sencillez en el diseño de los socket implicados en la conexión. No obstante el protocolo UDP no contempla en reenvió de datos incorrectos o no recibidos por lo que resulta ideales en entornos de comunicación broadcast pero no en sistemas que elija una buena fiabilidad en el envió de datos.

Esta comunicación UDP nos permite trasladar el envió de información fuera de la red, configurando los equipos para poder atravesar las diferentes puertas de enlace existentes en la red tales como Routers, Proxys, Firewalls, etc. Característica que dota al sistema del envío remoto de datos a través de internet y por lo tanto poder recibir las mediciones en cualquier parte.

- Comunicación vía Zigbee.

Arduino envía y recibe datos a través de los pines Serial conectados a la interface Xbee Shield, técnica que facilita la manera de transmisión. Si un dato quiere ser enviado solo ha de ser escrito por el serial en el emisor y ser leído por el receptor de forma totalmente transparente al trabajo que realizan los módulos Xbee.

El protocolo 802.15.4 está caracterizado por su gran ahorro de energía llegando incluso a los 2 años de duración de las baterías en algunas motas, su frecuencia de transmisión se en encuentra en 2.4 GHz. En el caso de los módulos Xbee Shield que implementan dicho protocolo tienen diferentes formas de configuración como son: modo Router, Coordinador y Punto final, para las serie 1 Xbee utilizado en este proyecto. La serie 2 de Xbee permite realizar conexiones maya o mesh entre los distintos nodos de la red. Su alcance de actuación se encuentra entre los 30 y 100 metros dependiendo si se trata de entorno exterior o interior, es posible dotar a las interfaces de una antena para mejorar su capacidad de transmisión.

6. Viabilidad técnica

El desarrollo del proyecto ha dado lugar a un equipo con alto nivel de adaptabilidad, la capacidad que Arduino ofrece a la hora de reconducir los proyectos brinda la posibilidad al sistema de ser utilizado no solo para la función acontecida si no para múltiples problemáticas. Dicha justificación corrobora que la inversión en el hardware pueda ser más elevada que la de un equipo diseñado solamente para trabajar en modo estación meteorológica pero a diferencia de este caso, los nodos diseñados no tienen una función cerrada. Esta premisa se cumple también a nivel de Software ya que el diseño de la comunicación entre los equipos permite que se pueda variar con cierta facilidad el cometido de los nodos. Por ello es posible afirmar que la viabilidad del proyecto puede considerarse como positiva cuando el producto requerido necesite tener una capacidad de adaptación, mejora o cambio de función.

Se puede destacar la alta información existente en internet acerca de los equipos utilizados en el desarrollo del proyecto, al contrario que los equipos hardware comerciales, los cuales cuentan con un sistema totalmente cerrado y que en muy pocas ocasiones permiten intervenir en su funcionamiento para mejorar o adaptar su capacidad operativa formalmente. El envío de datos fuera de la red a través de la tarjeta Ethernet capacita al sistema de una ventaja de poder para enviar datos remotamente de una manera sencilla. Todas estas características pueden justificar el desarrollo de más nodos emisores que hagan un sistema completo y que puedan operar en un entorno real, donde su operatividad conjunta proporcione más sentido al proyecto.

Por otro lado la viabilidad del producto en un entorno comercial requiere de un alto nivel de pruebas, donde los equipos deben de ser sometidos a esfuerzos tanto climáticos, operatividad, funcionamiento a largo plazo, etc. Deben de cumplir con una normativa de calidad además de ser un sistema que pueda garantizar que pueden trabajar en condiciones poco ideales como los agentes externos. Para un trabajo sometido a amenazas externas los equipos deberían de cumplir con los estándar IP69 que certifique su funcionamiento sin filtraciones de agua o corrosiones.

7. Valoración económica

A continuación se realiza una valoración del coste de los materiales empleados, así como una valoración estimada de las horas de desarrollo, mano de obra, instalación y mantenimiento del sistema por un año.

Coste de los materiales del Nodo Coordinador

Descripción	Cantidad	Precio Unitario	Precio Total
Arduino Xbee Shield	1	46,02	46,02 €
Arduino Ethernet Shield	1	36,58	36,58 €
Placa Arduino	1	26,00	26,00 €
Pantalla LCD 16x2 Caracteres	1	9,44	9,44 €
Caja para componentes Electrónicos	1	12,43	12,43 €
Interruptores	2	2,20	4,40 €
			Total 134,87 €

Coste de los materiales del Nodo Emisor

Descripción	Cantidad	Precio Unitario	Precio Total
Arduino Xbee Shield	1	46,02	46,02 €
Placa Arduino	1	26,00	26,00 €
Sensor Humedad 808H5V5	1	23,60	23,60 €
Sensor Temperatura	1	12,34	12,32 €
Sensor Luz	1	3,20	3,20 €
Caja para componentes Electrónicos	1	10,54	10,54 €
Interruptores	2	2,20	4,40 €
			Total 126,08 €

Gastos estimados de mano de obra, instalación y mantenimiento de 1 año

Descripción	Cantidad	Precio Unitario	Precio Total
Desarrollo código Emisor	20	8	160 €
Desarrollo código Coordinador	30	8	240 €
Construcción equipo Emisor	5	8	40 €
Construcción equipo Coordinador	4	8	32 €
Instalación de equipos **Incluye desplazamiento**	3	12	36 €
Mantenimiento un año	1	60	60 €
			Total 568 €

Precio total estimado del Proyecto

Descripción	Cantidad	Precio Unitario	Precio Total
Nodo Coordinador	1	134,87	134,87
Nodo Emisor	1	126,08	126,08 €
Horas de desarrollo, construcción Instalación y mantenimiento por un año.	1	568	568 €
			Total 828 €

Figura 30, presupuesto total estimado del producto.

Como se puede observar el coste total del proyecto, instalado y con una mantenimiento que cubra las deficiencias que puede ofrecer el equipo, asciende a 800 euros aproximadamente, teniendo en cuenta las horas dedicadas a la programación, investigación y posterior implementación tiene un coste muy ajustado. Para una segunda versión del equipo la estimación económica sería mucho menor ya que se reaprovecharía el código y solo deberían de ser corregidos bugs y problemas de la versión 1, pudiendo descender hasta los **500 euros.**

8. Conclusiones

En este apartado se comentan las conclusiones obtenidas después de la realización del proyecto, así como un análisis objetivo de las mejoras que deberían de implementarse en una segunda versión de la estación meteorológica. Además, en el ultimo apartado se describen un proceso de autoevaluación describiendo los problemas acontecidos durante el desarrollo.

8.1. Conclusiones

Los objetivos del proyecto han estado fijados ha cinco particularidades descritas al principio de este documento, un sistema **autónomo** basado en **Arduino** con una comunicación **Bidireccional** a través de protocolo **Zigbee** que contemple un ahorro de **energía** eficiente y además como funcionalidad adicional los datos sean enviados a través de Ethernet. Todos los hitos principales han sido cumplidos durante este trabajo. Cabe destacar que no se ha implementado un sistema WatchDog a causa de no disponer de un programador AVR para Atmel. Dicha función hubiera permitido apagar y encender el nodo Coordinador en intervalos cíclicos para optimizar su ahorro de energía mediante interrupciones autogeneradas. No obstante, en el apartado de mejoras se contempla la posibilidad de usar los equipos con sistemas de alimentación eólica o solar, para depender en menor grado de la red de energía eléctrica.

Dado que el objetivo del proyecto no era un desarrollo completo de hardware que acompaña a la plataforma Arduino, tales como Cajas estancas, pasa cables antihumedad, o diferentes interruptores con protección de agentes externos, no se ha llevado a cabo un desarrollo final de esta parte, siendo en estos momentos un equipo ideal para espacios de interior o lugares donde los agentes externos no incidan directamente en los equipos. En una segunda fase de implementación sería un objetivo principal ya que dotaría al sistema de un rango de acción más amplio.

El aprendizaje y conocimientos adquiridos han sido los deseados, se ha conseguido introducirse no solamente en la plataforma Arduino sino también en todo su entorno de desarrollo tales como: investigación de artículos y recursos en arduino.cc o arduinoplayground. Las horas de investigación han sido muy superiores a las de desarrollo, pero se ha conseguido una base para seguir creando proyectos dentro de esta plataforma o mejorando aspectos de este proyecto además de la posibilidad de abarcar otros de nueva índole.

8.2. Propuesta de mejoras

Dado que el desarrollo total del hardware esta fuera de los objetivos del proyecto, en este apartado se realizará una análisis de los diferentes aspectos que debería de cumplir los equipos necesarios para conseguir un producto terminado y adecuado para un mayor número de entornos:

- Cajas estancas que cumplan la normativa IP69, con protección ante chorros de agua. Además debe de contener pasa cables que no permitan la entrada de humedad al interior en el caso del nodo emisor, el cual los sensores deben de estar instalados en el exterior del equipo.

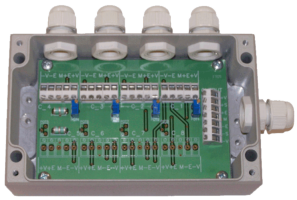


Figura 31, caja estanca IP69.

- Pulsadores e interruptores que cumplan la norma IP69.

Figura 32, pulsadores IP69.

- Módulo solar para Arduino, con un coste alrededor de 60 euros, el equipo incluye una batería de litio, la cual debe de ser cargada 15 horas la primera vez. El sistema produce un tensión de 5 voltios y una corriente eléctrica de 1100 mAh suficiente para el funcionamiento del equipo.

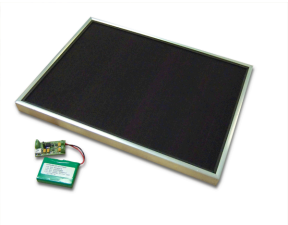


Figura 33, kit energía Solar.

- LCD con retroiluminación, sería posible en este momento instalar un Led en el módulo LCD para la correcta lectura de datos en entornos de poca visibilidad o sustituir este por una visor con retroiluminación.

Figura 34, LCD retroiluminado.

- Implementación WatchDog, la implementación del sistema dotaría a Arduino de un contador que podría resolver problemas como: bloqueos de ejecución realizando un Reset del equipo o generar interrupciones.

- Implementación de un botón para modificar el número de nodos receptores existentes en la red, esto ayudaría al equipo a realizar un coste computacional más óptimo ya que no ha de esperar por la respuesta de los equipos.

- Posibilidad de seleccionar el último dato enviado de un nodo emisor concreto, se podría habilitar a través de un pulsador.

- Posibilidad de incluir más equipos de medición como pluviómetros, veleta para el viento o medición de la velocidad del viento.

8.3. Autoevaluación

A continuación se describen los problemas acontecidos durante el desarrollo del proyecto

- Envío de variables :

En las primeras pruebas funcionales de la comunicación Zigbee 802.15.4 se detectó que el envío de variables entre nodos debía de tener un formato concreto. La primera prueba realizada arrojó que un envío de números enteros podía ser transmitido de diferentes formas. Es posible enviar la variable directamente volcándola sobre el serial, pero en el momento de la lectura se debe de tener un procedimiento estricto para su correcta interpretación.

Otra forma de envío es la interpretación ASCII de cada cifra del número, transformándolo en una cadena de caracteres y leyendo como cadena de caracteres. Esta forma, aunque requiera de más lecturas, posibilita interpretar el dato de una manera directa en tiempo de depuración.

El compilador de Avr y Arduino no contempla una biblioteca de C estándar con soporte para las viables de tipo float. Para solucionarlo se incluyeron en el código las funciones referenciadas en Arduino, con las cuales se realizan las conversiones pertinentes.

-Interpretar datos obtenidos de los sensores:

En el caso de la temperatura se ha recurrido a la ecuación Steinhart-Hart, la cual aproxima el valor obtenido del Thermisor a un valor en grados Celsius, kelvin, etc.

El sensor de humedad 808H5V5 aproxima una respuesta prácticamente lineal, por lo que ha permitido implementar una ecuación para poder interpretar el valor en función de los datos obtenidos.

En el caso del sensor de luz o LDR no se ha podido realizar una conversión a Lumens. La problemática ha sido no disponer de un medidor de luz para calibrar el LDR, por ello se ha decidido usar un valor de 0% oscuridad total a 100% máxima luz que puede absorber el sensor.

- Problemas circuito.

El circuito presentado es el resultado de 3 versiones previas. En las primeras pruebas se encontraron problemas en conexiones y mala conductividad. Los cables utilizados en un primer momento eran de una composición unifilar pero debido a su fragilidad se tomó la decisión de optar por cables multifilares. Para el correcto ensamblamiento de los sensores ha dispuesto de un soporte para soldar, un soldador de 30 W, estaño de dos componentes, varios tipos de componentes electrónicos.

- Problema conexión entre módulos Xbee Shield, Ethernet Shield y Arduino.

Al conectar el módulo Xbee Shield sobre la parte superior del módulo Ethernet Shield y a su vez este último al Arduino, se pierde la conexión serial que el Xbee Shield utiliza directamente de la placa Arduino. Para poder alimentar Xbee se ha implementado un pequeño bus de cables entre el Xbee Shield y el Arduino, el cual realiza una extensión de los pines serial, dejando en medio el Ethernet Shield, el cual no utiliza dicha conexión.

- Watchdog.

Es posible implementar en Arduino un Watchdog para el control de fallos, reinicio automático o contador de operaciones. Para dicha implementación se ha de modificar el bootloader y utilizando uno diferente a Arduino. No se ha podido implementar al no disponer de dicho bootloader.

- Nodo Coordinador.

El nodo coordinador no puede entrar en reposo a causa de no disponer de un Watchdog, ya que este no puede recibir interrupciones internas automáticas, por lo que debe de estar conectado a la red eléctrica para no producir un cambio de baterías frecuente. No obstante, se ha optimizado su consumo y se ha dotado al equipo de una batería de 9 voltios la cual le permite trabajar sin ningún problema, además en caso de caída de tensión, el nodo seguiría funcionando. Esto se debe a que Arduino Duemilanove implementa una función automática en la detección del voltaje de entrada, en caso de tener las dos fuentes de tensión conectadas Arduino trabajará con voltaje por USB o Fuente de alimentación.

- Apagado de las radios Xbee Shield.

Las radios del Xbee deben de estar encendidas para poder recibir los datos y despertar al Arduino por el pin RX. El Xbee Shield contempla ahorros de energía cíclicos pero podrían causar problemas a la hora de transmitir si este se ha dormido en el momento que el nodo coordinador le realiza una petición. Se han encontrado artículos que comentan problemáticas de reseteo en Arduino al utilizar este método, por lo que debería de ser caso de estudio en una siguiente versión.

9. Glosario

Arduino: Plataforma de hardware libre basada en un microcontrolador, la cual cuenta con un entorno de desarrollo propio. Desarrollada para facilitar la creación de proyectos electrónicos.

Arte Media: Arte ligado a las nuevas tecnologías, tales como electrónica, audiovisual, videojuegos, etc.

AVR: Microcontrolador de la familia Atmel diseñado para la ejecución del leguaje C.

CRC: Código de detección de error, este recibe un flujo de datos en la entrada y genera un valor fijo a la salida, el cual debe de ser comparado.

Host: Equipo perteneciente a una red tales como PC, Impresora, etc.

Ip69: Estándar Alemán que describe protección de equipos antes agentes externos.

Jumper: Elemento electrónico usado para interconectar pines pertenecientes a un circuito.

LCD: Pantalla de cristal liquido.

Open Software: Software de tipo abierto, el cual puede ser consultado o modificado sin restricción.

Open Hardware: Hardware de tipo abierto del cual pone a disposición todas sus características, permitiendo además su modificación con total libertad.

Processing: Lenguaje de programación de código abierto basado en Java.

Protocolo 802.15.4: Protocolo Zigbee.

Pam: Identificador de Área personal.

Pic: Microcontrolador fabricados por Microchip Technology.

Shield: Interfaces disponibles para conexión directa Arduino.

Sketch: Nombre de programas diseñados para plataforma Arduino.

TCP: Protocolo de transporte orientado a conexión, definido en la capa 3 del modelo OSI.

UDP: Protocolo de trasporte no orientado a conexión, definido en la capa 3 del modelo OSI.

Wiring: Lenguaje de programación utilizado en Arduino basado en C.

Workshop: Talleres con temáticas de electrónica, Audiovisual, arte media, etc.

Xbee: Estándar de comunicación inalámbrica en Arduino, el cual está basado en Zigbee.

Zigbee: Especificación de protocolos para comunicación inalámbrica en redes de área personal, caracterizado por su bajo consumo y seguridad.

10. Bibliografía

[1] Thermisor Wikipedia, (18/10/2011). Información del funcionamiento del Thermisor, ecuación Steinhart-Hart, url: http://en.wikipedia.org/wiki/Thermistor.

[2] Thermisor Arduino, (18/10/2011). Implementación de la ecuación Steinhart-Hart sobre plataforma Arduino, url: http://arduino.cc/playground/ComponentLib/Thermistor2

[3] Sensor de Humedad, información técnica, (20/10/2011). Información sobre características y grafica de respuesta, url: http://www.cooking-hacks.com/skin/frontend/default/cooking/pdf/Humedad-808H5V5.pdf

[4] Sensor de Humedad ecuación, (20/10/2011). Implementación formula para el calculo equivalente porcentaje de humedad, url: http://netlabmn.unipv.it/wsn/doc/squidbee_interpreting.pdf

[5] Sensor de Humedad Conexión, (20/10/2011). Esquema de conexión para el modelo de sensor tratado en este trabajo, url: http://hfiel.es/proyectos-actuales/sensor-arduino/8-sensor-arduino-materiales-y-conexion-de-componentes

[6] Sensor Luz, características y funcionamiento, (23/10/2011). Características principales del sensor de luz, url: http://www.ladyada.net/learn/sensors/cds.html

[7] Entorno de desarrollo Arduino, (2/10/2011). Manejo del programa, ejemplos y ayuda.
url: http://arduino.cc/es/Guide/Environment

[8] Bus de conexión, entre Xbee Shield, Ethernet Shield y Arduino, (19/11/2011).
url: http://www.sindono.com/stacking-an-arduino-ethernet-shield-and-an-xbee-shield/

[9] Xbee, punto final y Coordinador, (28/11/2011). Configuración de los módulos.
url: http://ftp1.digi.com/support/documentation/90000976_G.pdf

[10] Configuración bidireccionalidad Xbee Shield, (18/11/2011).
url: http://arduino.cc/es/Guide/ArduinoXbeeShield

[11] Modo ahorro energía Arduino (interrupción puerto serie) y Librería AVR, (21/11/2011).
url: http://www.arduino.cc/playground/Learning/ArduinoSleepCode

[12] Modo ahorro energía Arduino, (25/11/2011). Consulta de los diferentes modos de ahorro que dispone la plataforma, url: http://www.nongnu.org/avr-libc/user-manual/group__avr__power.html

[13] Función para realizar printf, (20/11/2011)**.**
Url: http://www.arduino.cc/playground/Main/Printf

[14] Reseteo de Arduino a causa de Sleep de Xbee, (28/11/2011). Problemática derivada de los modos de ahorro de energía en los interface Xbee Shield.
url: http://rubenlaguna.com/wp/2009/03/05/setting-xbee-to-sleep-causes-arduino-reset/

[15] Control de errores, CRC en Arduino, (9/11/2011). Implementación, información de código para detección de errores en información enviada.
urls: http://excamera.com/sphinx/article-crc.html, http://arduino.cc/playground/Main/LibraryList, http://es.wikipedia.org/wiki/Comprobaci%C3%B3n_de_redundancia_c%C3%ADclica

[16] Función ultoa, (10/11/2011). Convertir variable Unsigned long.
url: http://www.nongnu.org/avr-libc/user manual/group__avr__stdlib.html#ga66e31b615d9ef1a19c452d64d7250112

[17] Función ftoa, (3/12/2011). Convertir variable float.
urls: http://www.todopic.com.ar/foros/index.php?topic=34364, http://forums.adafruit.com/viewtopic.php?f=25&t=24077, http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1164927646

[18] Socket UDP, (11/12/2011). Practica Protocolos y Aplicaciones de internet Año 2010 cursada en UOC.

[19] Librerías SPI.h y Ethernet.h para Ethernet Shield, (11/12/2011).
url: http://arduino.cc/en/Reference/SPI

[20] Librería UDP para socket Arduino, (11/11/2011).
url: http://arduino.cc/en/Tutorial/UDPSendReceiveString

[21] Obtención y consulta de equipos, (08/9/2011).
url: http://www.cooking-hacks.com/

[22] Configuración Xbee con XCTU, (18/11/2011).
url: http://blog.make-a-tronik.com/configuracion-basica-modulos-xbee-2-5-xbee-xplorer-usb/

11. Anexos

11.1. Manual de usuario

Se define el modo de operación para la puesta en marcha de los equipos, en las siguientes figuras se indican mediante una numeración las diferentes partes de los equipos:

Figura 35, partes Equipo 1 manual de usuario.

Interruptor de encendido Nodo Coordinador.

1. Pulsador selección Frecuencia de medición.
2. Entrada para conectar trasformador eléctrico.
3. Puerto USB y Puerto RJ45.
4. Interruptor ON/OFF.
5. Pulsador Reset.
6. Sensores y Protección para los mismos.
7. Puerto USB parte inferior del equipo.

Figura 36, partes Equipo 2 manual de usuario.

8. Pantalla LCD para lectura de datos.
9. Batería de 9 voltios.
10. Batería de 9 voltios.
11. Led de estado Arduino Emisor.

Modo de operación:

1. Para poner en marcha los equipos acciónese las palancas **1** y **5** indistintamente. Se recomienda conectar el Nodo Coordinador a una fuente de tensión externa para no cambiar las baterías frecuentemente.
2. Seleccione la Frecuencia de Medición en el nodo Coordinador, accionando el pulsador **2**. Una pulsación rápida alternara entre las diferentes opciones, y una pulsación larga confirmará la selección.
3. Después de unos segundos aparecerá en la pantalla LCD **9** los valores de datos correspondientes a la medición del nodo.
4. Envío de datos vía Ethernet, conecte un cable Cat 5 o 6 al puerto Rj 45 número **4** en un switch o hub.
5. Conecte un PC al switch y configure la dirección de red 192.168.1.50 y ejecute el socket servidor. Automáticamente los datos serán recibidos.
6. Desconectar equipos, accione las palancas **1** y **5**.

Resolución de problemas:

a) No se reciben datos.

- Compruebe el estado de las baterías **10** y **11**.

- Encienda y apague los equipos.

- Compruebe la distancia de los nodos, nunca superior a 100 metros en exterior y 10 metros en interiores con paredes gruesas.

b) Dato Anómalo.

- Compruebe que el equipo no está sufriendo daños externos, excesivo Calor, etc.

- Compruebe el estado de los sensores.

c) No se reciben datos vía Ethernet.

- Compruebe que los cables están conectados y se encuentran en la misma red y mismo direccionamiento IP.

11.2. Ejecución y compilación

ENTORNO ADUINO

Arduino posee un entorno de programación propio, el cual permite edición, depuración, compilación y posterior subida del los programas o Sketch a la placa Arduino. Además de las características tratadas anteriormente, el entorno permite la comunicación con el puerto serie tanto de lectura como de escritura y monitorización. Los programas diseñados son denominados Sketch y como se ha comentado anteriormente pueden ser verificados con el software propietario.

A continuación una imagen del editor descrito :

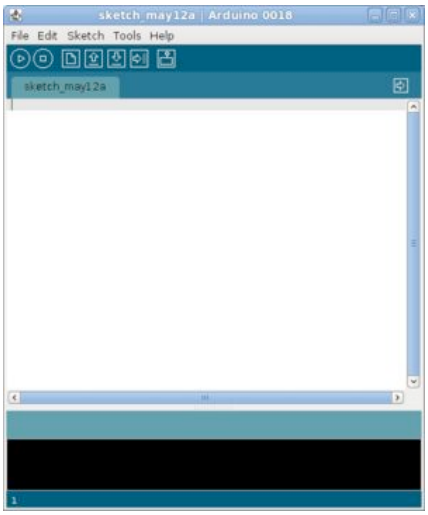


Figura 37, entorno de programación Arduino.

El propio editor contiene las librerías comúnmente utilizadas en Arduino, además de ejemplos tipo: poder comprobar la electrónica o realizar pruebas funcionales. También se ha de destacar que el editor está provisto de un depurador de errores .

Pasos a seguir para poder realizar una programación, verificación, compilación y posterior carga del código al Arduino:

- 1º Abrir el programa Arduino, si no se dispone puede conseguirse en el siguiente enlace: http://arduino.cc/en/Main/Software.

- 2º En el propio editor se puede copiar el código entregado o puede editarse.

- 3º Una vez editado o copiado se presiona en la tecla Verify  y se realiza la depuración.

- 4º Conectar el dispositivo al puerto USB, en ocasiones hay que establecer en que puerto se encuentra el USB y el modelo de placa Arduino del que disponemos. Para ello solo habrá que ir a preferencias y seleccionar los valores correctos.

- 5º Si la verificación ha tenido éxito, pasamos al proceso de compilación y subida del código a la placa, para ello presionamos en Upload to I/O Board  .

- 6º Una vez el programa ha sido subido si se están transmitiendo datos vía Serial podríamos comprobarlos presionando en Serial  , eligiendo la velocidad adecuada del puerto Serie en nuestro caso 9600 baudios.

Nota : En este caso al estar utilizando Xbee Shield debemos de retirar los 2 jumper que se encuentran en la parte superior de las interfaces, para poder realizar la subida del firmware.

SOCKET UDP

El socket UDP, está desarrollado para un sistema Linux. Para compilar dicho socket solo debemos de tener instalado en nuestro sistema Linux un compilador gcc tipo al que es posible encontrar en: http://gcc.gnu.org/ de libre distribución. Una vez instalado debemos de copiar el servidorUdp.c en un directorio y dirigirnos desde una ventana de terminal a dicho directorio.

Nota: Recordatorio para listar y cambiar de directorios en Linux utilizamos: ls lista el directorio actual y cd "directorio" para cambiar al directorio.

Abriendo la terminal del sistema Linux, se navegará hasta el directorio donde se encuentre el socket, una vez dentro se escribirá en la línea de comandos:

gcc –o servidorUdp servidorUdp.c

A partir de este momento el socket ya estará compilado y preparado. Para ejecutar el socket solo se ha de escribir en la terminal la siguiente instrucción:

./servidorUdp

El socket comenzará a funcionar, y esperará a la recepción de datos. En la siguiente captura de pantalla se muestra el proceso:

```
usuario@cliente:~$ gcc -o servidorudp servidorudp.c
usuario@cliente:~$ ./servidorudp
ID: 1 T: 21.64 H: 65.54 L: 27

Emisor 2 apagado

Emisor 3 apagado

Proxima lectura: 5 segundos

ID: 1 T: 21.55 H: 65.54 L: 31
```

Figura 38, compilación y ejecución socket UDP servidor.

Nota: Se debe de tener en cuenta la dirección IP de el equipo Linux además de encontrarse en la misma red que el nodo Coordinador.