SISTEMA DE AHORRO ENERGÉTICO EN CLIMATIZACION E ILUMINACIÓN, PARA GRANDES ESPACIOS COMERCIALES Y DE OCIO, MEDIANTE RED DE SENSORES INALAMBRICOS

José Vázquez Mouzo

Ingeniería de Telecomunicaciones: Especialidad Telemática

Consultor: Jordi Bécares

10 de Junio de 2011

DEDICATORIA Y AGRADECIMIENTOS

A mi mujer Paqui y a mi hijo Alejandro
sin vuestro amor y comprensión nada hubiera sido posible

Agradezco muy sinceramente a todo el equipo docente de la UOC por el magnifico trabajo realizado durante todos estos años que hemos compartido.

También a mi familia por el enorme sacrificio que ha realizado para permitirme un poco de espacio y tiempo para poder enfrentarme a este reto.

Y un agradecimiento especial a mi consultor de proyecto Jordi Bécares por su atención personalizada y diligente.

A mis compañeros de aula. Mucha suerte.

RESUMEN

El principal objetivo de este trabajo final de carrera es adquirir conocimientos y habilidades dentro del mundo de los sensores inalámbricos, experimentar con la Tecnología TinyOs y con el lenguaje de programación NesC. Para conseguir estas metas he realizado una aplicación que desarrolla un Sistema de Control zonal sobre equipos de climatización e iluminación, se pretende que esta gestión sea bajo un planteamiento de ahorro de recursos energéticos y para tal fin se pretende dotar a los nodos sensores de zona, de capacidad de decisión, para que sean ellos, los agentes de campo, los que articulen la sincronización entre la demanda de temperatura y luminosidad del usuario con las prestaciones que ofrecen las unidades de producción. Al mismo tiempo el usuario tiene la posibilidad de variar los puntos de ajuste y control sobre el Sistema asumiendo la dirección del proceso si lo cree conveniente, tanto para labores de mantenimiento y reparación como para decisiones de funcionamiento .

Palabras clave : Motas, Redes de Sensores inalámbricos, WSN, TinyOS, NesC, Ahorro energético

AREA: Sistemas Empotrados

TITULO: SISTEMA DE AHORRO ENERGÉTICO EN CLIMATIZACION E ILUMINACIÓN, PARA GRANDES ESPACIOS COMERCIALES Y DE OCIO, MEDIANTE RED DE SENSORES INALAMBRICOS

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

Capítulo 1

INTRODUCCIÓN

Este sistema esta orientado principalmente como base de aprendizaje personal en el mundo de la programación de los sensores inalámbricos, con la intención de investigar y analizar el comportamiento y las posibilidades de las Motas dentro del campo de la climatización en especial y del control de instalaciones en general, que es a lo que me dedico profesionalmente actualmente con mas de 20 años de experiencia en este campo.

Por ello, esta aplicación esta diseñada para realizar el control de una zona comercial, en cuanto a necesidades de potencia frigorífica y modo de climatización así como en el control de la intensidad lumínica según las condiciones ambientales. Siendo además este sistema la base sobre la que desarrollar los conocimientos de los tutoriales de Tinyos así como el aprendizaje adquirido de otras fuentes, para poder analizar las capacidades de este tipo de nodos sensores en futuras aplicaciones en el campo del Frío Industrial y Climatización.

1.1. CONTENIDO DE LA MEMORIA

El contenido de esta memoria refleja tanto el trabajo realizado con las Motas proporcionadas como la investigación y estudio realizado en el ámbito de las Redes de sensores, desde su programación con NesC hasta el conocimiento detallado de la estructura de Tinyos.

- En el primer capitulo se ha realizado un análisis introductorio sobre cual han sido las motivaciones para la elección de este proyecto así como los objetivos marcados de forma básica desde el principio, con que recursos hemos trabajado y como se ha planificado el esfuerzo para la realización de este proyecto.

- En el segundo capitulo analizamos el estado de la tecnología usada, motas, redes de sensores, particularidades de TinyOS, NesC así como profundizar en el mercado en busca de aplicaciones semejantes.

- El tercer capitulo ahonda en la descripción de funcionamiento del sistema, desde un planteamiento de diseño global y desde un diseño parcial por aplicación individual

- El capitulo cuarto profundiza en detalles técnicos de composición de los distintos módulos de configuración, para dar idea de cómo es la programación NesC desde los ficheros principales.

- En el capitulo cinco se analiza la viabilidad del proyecto desde el presento o en el futuro, se hará si es posible una valoración económica.

- En el seis se hacen efectivas las conclusiones a las que se llega con la elaboración de este proyecto.

- El capitulo siete muestra las referencias bibliografías, así como paginas Web con información de apoyo.

- El capitulo de Anexos conformará todo aquello que no se ha desarrollado en el resto de capítulos

1.2. MOTIVACION DEL PROYECTO

La motivación que me lleva a la realización de este proyecto es analizar las posibilidades que puede aportar la tecnología de sensores inalámbricos aplicada en instalaciones frigoríficas, seguridad, eléctricas y climatización en locales comerciales o de ocio. Este tipo de locales suelen tener unas necesidades ambientales y de confort muy especiales, buscando siempre unas condiciones que equilibren la comodidad tanto de los clientes como de los trabajadores con una cada vez mayor necesidad de ajuste energético, tanto por ahorro económico como por conciencia social..

Habitualmente los que nos dedicamos a la realización de instalaciones comerciales tanto de electricidad como frigoríficas ó de climatización, utilizamos nodos sensores cableados y al mismo tiempo interconectados, pero sin duda la posibilidad de sensores inalámbricos con la ventaja de ser desplazados en cualquier momento sin representar ningún costo extra, tanto en mano de obra como materiales, representaría sin lugar a dudas un aliciente para su utilización, además de un interesante ahorro económico. Otro de los alicientes es que en las instalaciones frigoríficas, de climatización o en general de control, los sensores siempre son unifuncionales, mientras que una Mota nos puede aportar múltiples sensados al mismo tiempo y de índole diversa. También es muy importante el hecho que el propio sensor disponga de capacidad de calculo, aunque sea poca, lo cual nos permitirá que el propio sensor pueda tomar decisiones de funcionamiento. En muchas instalaciones no necesitamos grandes cálculos matemáticos, simplemente adecuar la potencia a las necesidades de una forma armónica.

Hoy en día hay una gran cantidad de programas de gestión instalados en autómatas o PCS, que una vez recogidos los datos de los sensores establecen cual puede ser la mejor decisión de control según las condiciones establecidas, en este proyecto mi intención es que el propio sensor determine, según sus datos, cual es la mejor opción de funcionamiento. De esta forma no seria necesaria la utilización de elementos de gestión pudiendo ser la propia maquina de Climatización

o regulador eléctrico el que reciba las instrucciones directamente del sensor, empleando el ordenador u otro dispositivo solo para comprobación y puesta en marcha, programando las consignas de funcionamiento y después pasar la mota pasarela directamente al controlador del equipo de climatización o de iluminación.

1.3. DESCRIPCION DEL PROYECTO

En este proyecto final de carrera se analiza y crea un sistema para control de equipos de climatización e iluminación. Para ello se implementaran tres aplicaciones, con nombres de constelaciones, Casiopea, Scutum y Hidrus cada una de ellas instaladas sobre uno de los elementos de sistema, formado por dos nodos sensores y un PC. De esta forma tenemos que Casiopea será instalado sobre la mota que se halle en la zona a climatizar y a controlar el estado lumínico, Scutum sobre el nodo que realiza la función de pasarela con el PC e Hidrus, que es una pequeña aplicación en java, simplemente monitorizará por consola los datos recibidos de la zona y nos permitirá programar las consignas de funcionamiento que serán enviadas al nodo sensor ambiente.

En la figura 1 se pueden observar los distintos elementos que componen el sistema, aunque solo disponemos físicamente de dos sensores no pudiendo crear un sistema múltiple.

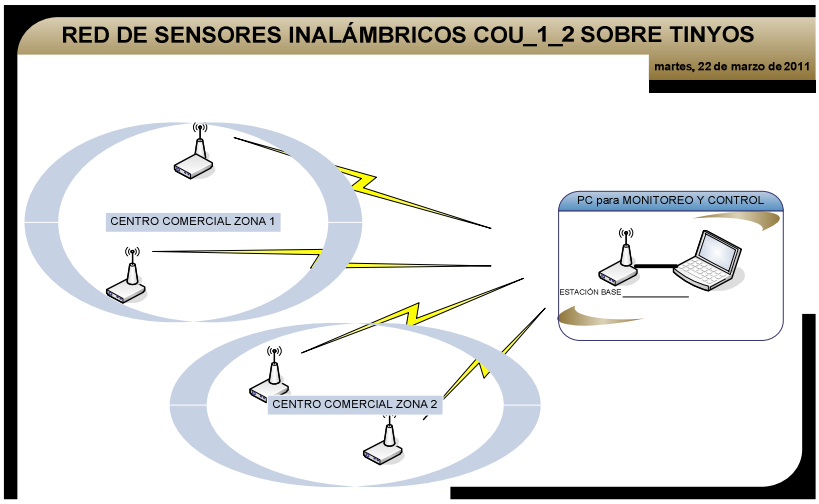

Figura 1. Elementos del sistema

Desde un principio mi idea ha sido la de dotar a la mota de zona de la capacidad de control absoluto, siendo la que efectúa el sensado, ejercita el calculo correspondiente, se intercambian datos sus componentes y además decide cuales son las medidas a adoptar. De esta forma pasaría de ser un elemento casi pasivo que solo realiza funciones de sensado y comunicación, mientras otros elementos toman las decisiones, a ser un elemento activo y coordinador, ejerciendo el control efectivo de las maquinas a las que esta vinculado.

1.4. OBJETIVOS

Los objetivos a conseguir en líneas generales con este proyecto son:

1. Disponer de una visión real del funcionamiento y comunicación de las redes de sensores inalámbricos. Poniendo en operación los motes, instalando los programas creados y así comprobar en tiempo real la operatividad del sistema. Ver y analizar los datos recibidos para implementación en el futuro sobre dispositivos reales.

2. Analizar las posibilidades NesC, un entorno diferente a los utilizados hasta el momento en la carrera, aunque al ser un derivado del lenguaje C hay muchos puntos de unión.

3. Realizar el sensado de temperatura y luminosidad, así como detectar la presencia de un campo magnético cercano a la Mota.

4. Gestionar según los datos sensados el sistema de climatización, definiendo el usuario una consigna y una posición, automática o manual (frío ó calor), la Mota debe establecer el modo de funcionamiento y el porcentaje de potencia necesaria para alcanzar esa consigna. En la posición automática la Mota determinará si el sistema de climatización debe aportar frío o calefacción o debe solo funcionar en modo ventilación con renovación de aire. En posición manual el usuario o instalador fuerza el modo eligiendo enfriamiento o calentamiento, empleando en este caso siempre el 100% de potencia.

5. Mediante tarjeta magnética pasada sobre la mota, esta comunicará el paro inmediato al sistema de climatización y al de iluminación así como el reinicio de la actividad en el momento que se vuelva a pasar la tarjeta.

6. Aplicación en Java de comprobación que sirve de interfaz de usuario. En ella visualizaremos los datos sensados por la Mota, así como las decisiones de control que esta establece.

7. En esta aplicación, tendremos la posibilidad de interactuar con los sensores a través de la modificación de las consignas de trabajo, que podremos establecer en la interfaz de usuario. Analizando la variación a los cambios propuestos.

1.5. ENFOQUE Y METODO SEGUIDO

1.5.1. ENFOQUE DEL SISTEMA

Desde un principio según se fueron ampliando los conocimientos en la tecnología de redes de sensores inalámbricos, el enfoque fue el de agrupar todo el peso de la aplicación en la mota situada en la zona ambiental que ejerce las funciones de sensado, control y de comunicación. Para ello se pretendía una total interrelación entre todos los componentes del la aplicación. En la figura 2 se ve el enfoque sobre el que se trabajo en la segunda prueba de evaluación, que abarcaba la ruta desde el nodo hasta el computador.

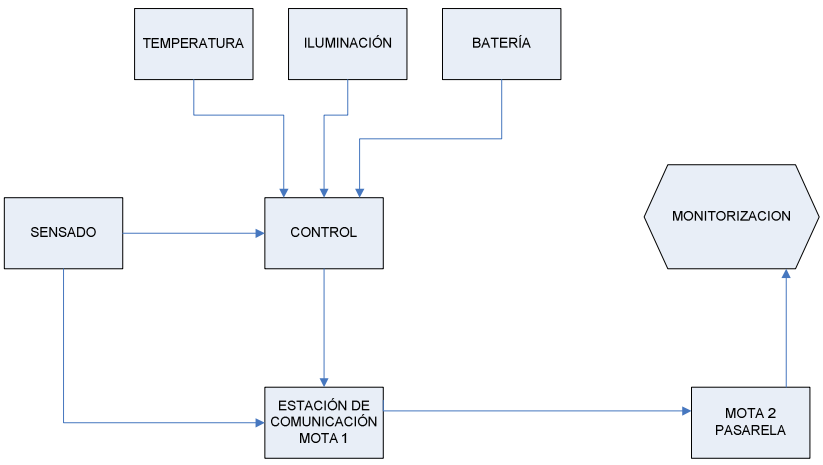

Figura 2. Enfoque dirección mota PC

El enfoque en la tercera prueba se basaba en la recepción de datos por parte de la mota de Zona, como gestionar las consignas de usuario, que influirán directamente sobre el control del sistema. en la siguiente figura, 3, se puede observar el enfoque dado en la dirección contraria.

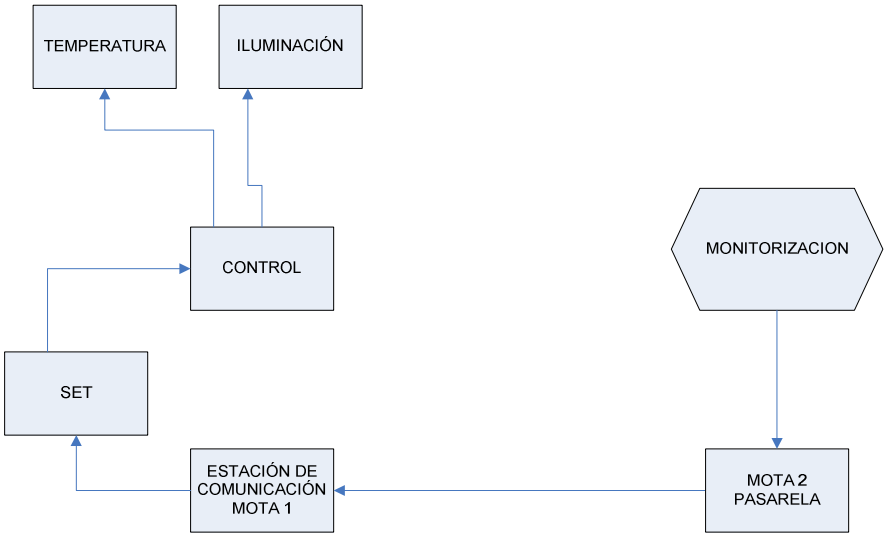

Figura 3. Enfoque dirección PC a nodo sensor

A partir de la implementación de las dos direcciones se adapto el sensado del Efecto Hall sobre el sistema al cual dedicaremos un apartado exclusivo en los detalles del sistema en el capitulo 3.

1.5.2. METODO SEGUIDO

El método seguido en la implementación de la aplicación o del sistema en general, basándome en los enfoques del apartado anterior, ha sido el método de los seis pasos. se trata de un método universal que se puede aplicar a cualquier tipo de proyecto científico o tecnológico, y al que se le pueden asignar distintos tipos de variantes según el modelo sobre el que aplicarlo.

En la figura 4 vemos las distintas etapas realizadas:

- Con una parte organizativa, que engloba el planteamiento del sistema o la idea de proyecto, la búsqueda por distintos medios de la información, el diseño de la aplicación y la planificación con las tareas y el diagrama de Gannt.
- Otra parte tecnológica compuesta por todo el trabajo de codificación como el continuo trabajo de reparación, mediante el sistema prueba error.

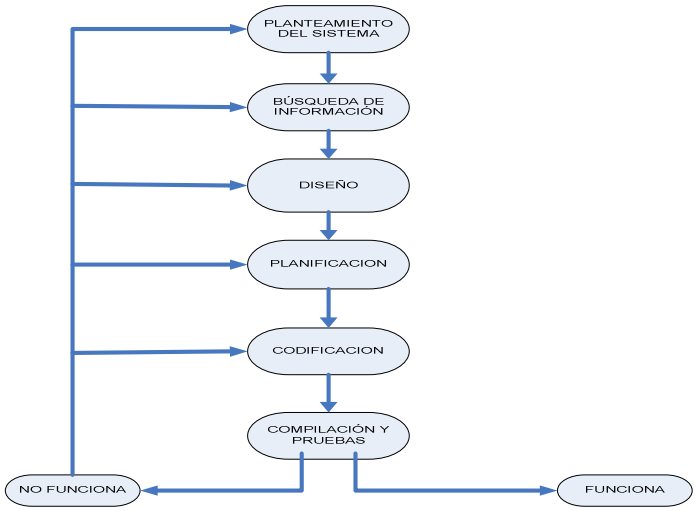

Figura 4. Método de los seis pasos

1.6. PLANIFICACION DEL PROYECTO

1.6.1. TAREAS ACOTADAS Y SU DESCRIPCION

Las tareas a cumplir así como el tiempo estipulado para su realización están reflejadas en la figura siguiente (figura 5), se puede observar que se realizo una modificación con respecto a la preestablecida al comienzo del proyecto, debido a problemas en el desarrollo de la PAC1. Este periodo de ampliación me permitió corregir los errores de funcionamiento

Id	Nombre de tarea	Duración	Comienzo	Fin	Predece
1	**PAC1 Preparacion proyecto**	**17 días?**	**jue 03/03/11**	**vie 25/03/11**	
2	Propuesta TFC	6 días?	jue 03/03/11	jue 10/03/11	
3	Implantacion entorno de trabajo	5 días?	vie 11/03/11	jue 17/03/11	2
4	Pruebas de red con BlinkCOU	1 día?	vie 18/03/11	vie 18/03/11	
5	Plan de Trabajo	15 días?	lun 07/03/11	vie 25/03/11	
6	**PAC 2 Entrega de codigo Parte1**	**21 días?**	**lun 28/03/11**	**lun 25/04/11**	
7	Estudio y practica con NesC	4 días?	lun 28/03/11	jue 31/03/11	
8	Codigo Aplicación COU 1_1(Envio Datos)	8 días?	vie 01/04/11	mar 12/04/11	7
9	Codigo Aplicación Interfaz Usuario(Monitorizacion)	6 días?	mié 13/04/11	mié 20/04/11	8
10	Memoria (Parte 1)	2 días	jue 21/04/11	vie 22/04/11	9
11	Depuracion y Pruebas	2 días?	vie 22/04/11	lun 25/04/11	
12	**AMPLIACION ESPECIAL Para cumplimiento Fase 1**	6 días	mar 26/04/11	mar 03/05/11	11
13	**PAC 3 Entrega del codigo Parte 2**	**21 días**	**mié 04/05/11**	**mié 01/06/11**	**12**
14	Codigo Aplicación COU 1_1 (Recepcion de Datos)	9 días	mié 04/05/11	lun 16/05/11	
15	Codigo Aplicación Interfaz de Usuario (Comunicación con COU)	7 días	mar 17/05/11	mié 25/05/11	14
16	Memoria (Parte2)	2 días	jue 26/05/11	vie 27/05/11	15
17	Depuracion y Pruebas	2 días	lun 30/05/11	mar 31/05/11	16
18	Entrega Código	1 día	mié 01/06/11	mié 01/06/11	17
19	**MEMORIA TFC**	8 días	mié 01/06/11	vie 10/06/11	
20	**PRESENTACION**	**2 días**	**jue 16/06/11**	**vie 17/06/11**	**19**
21	Ultimos detalles y entrega final	2 días	jue 16/06/11	vie 17/06/11	

Figura 5. Tareas y subáreas

Las tareas a cumplir para la realización de este trabajo final han sido:

1. **Preparación e inicio del proyecto**. Esta primera tarea engloba la denominada Propuesta de Trabajo, elección del trabajo a realizar, la implantación del entorno sobre el que trabajaremos en el proyecto, así como las primeras pruebas a las que se ha sometido a este entorno. En estas pruebas se ha utilizado el código suministrado, BlinkCOU y una aplicación del entorno de Tinyos ,BaseStation, de esta forma podremos comprobar el funcionamiento de la red.

2. **Código Parte1**. En esta tarea, se estudia el lenguaje de programación a utilizar, NesC, para poder implementar el código de la Mota. Se diseña el código sensado de datos, de control de temperatura y luminosidad y la transmisión de mensajes El otro COU ya estará preparado para funcionar como pasarela, se hizo en la primera tarea. En esta tarea reflejaremos el funcionamiento en dirección Mota PC.

3. **Ampliación especial.** Se estableció una fase especial para subsanar los problemas derivados de una mala implementación en el envío de datos. Todo el problema partía de que en NesC los componentes son compartimentos estancos que solo se relacionan a

través de la interfaces, yo pretendía unirlos por medio de variables globales en archivos de cabecera.

4. **Código Parte 2**. Aquí continuaremos con la codificación que falta en las dos aplicaciones tanto en la mota como en la interfaz de usuario. Debemos poder comunicar desde la mota al PC y desde el PC a la mota para pasar los set points de trabajo. En esta tarea reflejaremos el funcionamiento en dirección del PC a la Mota zonal.

5. **Memoria.** En esta fase debemos completar la memoria, a la cual hemos ido incorporando datos durante las distintas fases.

6. **Presentación.** Realizar una presentación en video de los trabajos realizados con las motas

1.6.2. DIAGRAMA DE GANNT

El diagrama de Gannt representativo de las tareas asignadas al proyecto seria el representado en la figura 6.

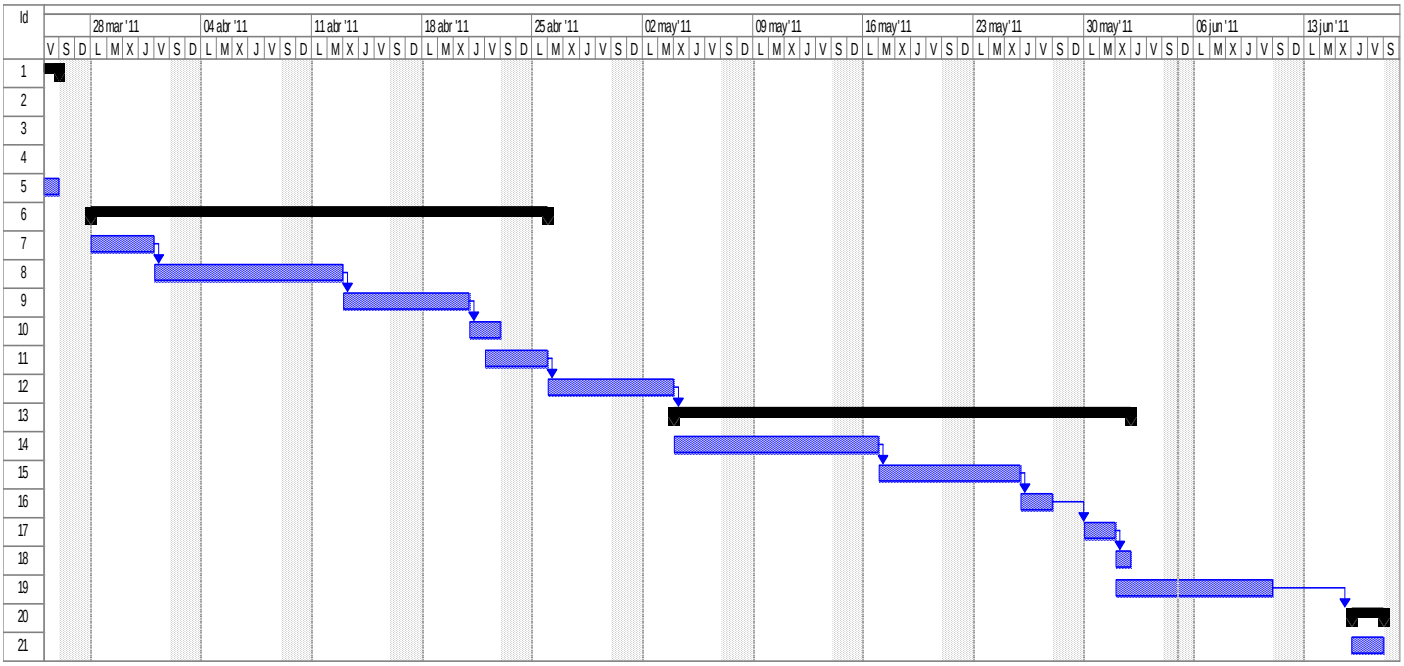

Figura 6. Diagrama de Gannt de este proyecto

1.7. RECURSOS UTILIZADOS

Los recursos básico utilizados para la realización de este proyecto son:

1. **Dos Motas COU 1_1**. Las motas son proporcionadas por la UOC como material de la asignatura, disponen de tres sensores, temperatura, luminosidad y efecto Hall. Utilizaremos una de ellas como Mota remota y otra como Mota pasarela para recepcionar los mensajes así como transmitir las consignas de la aplicación PC.

2. **Ordenador Portátil.** Un ordenador con Ubuntu sobre maquina virtual VMware, con tres puertos USB disponibles

3. **Sistema Operativo Tinyos.** Este sistema será el utilizado para la programación de la red de sensores inalámbricos.

4. **Varios programas necesarios.** Como son Mesprog para programar las motas, Eclipse con Plugin Yeti para codificar los ficheros de la aplicación.

5. **Programa fundamental SerialForwarder.** Se trata de un programa clave para comunicar la mota pasarela con la interfaz de usuario

6. **Aplicaciones Tinyos utilizadas.** Se utilizan en distintas fases Listen, SerialForwarder asi como BaseStation

7. **Libro Tinyos Programming.** La Biblia de los programadores de sensores, escrito por Philip Levis y David Gay, contiene todo lo necesario sobre programación en NesC.

8. **Tutoriales Tinyos.** Pagina de Web de referencia para la realización de este trabajo, sobretodo por el seguimiento de los tutoriales, que son la base para aprender esta nueva tecnología.

1.8. PRODUCTO CONSEGUIDO

El producto resultado, de este Trabajo Final de Carrera es un sistema de control en fase experimental, no podríamos aplicarlo directamente sobre una unidad de climatización, sobretodo sin añadir mas variables a controlar. Llevar el control de una unidad de Aire Acondicionado que ya viene implementada de fabrica con todo tipo de dispositivos de seguridad y control requiere en primer lugar un estudio detallado que analice la relación entre el nodo sensor y el circuito frigorífico. En una maquina frigorífica intervienen principalmente tres elementos compresor, condensador y evaporador, cuando la mota de zona me pida la utilización del sesenta por cien de la potencia yo debo transmitirla en una proporción diferente a cada elemento para mantener el circuito en equilibrio y no actúen seguridades de control de presión o flujo de aire. Por eso digo que eso entraría dentro de una aplicación de mayor calado.

El producto conseguido es un sistema formado por tres aplicaciones:

- Casiopea, es una aplicación que puede llevar el control de un equipo de climatización que lleve incorporado regulador de velocidad con variación de frecuencia o un sistema por capacidad. También podría controlar cualquier tipo de reactancias electrónicas de los lineales fluorescentes de iluminación o focos con regulación.
- Scutum, la misión de esta aplicación es la de recoger los mensajes de radio y pasarlos al puerto serie y en sentido contrario realizar el proceso inverso para ello se ha reutilizado una aplicación de Tinyos, BaseStation que cumplía perfectamente este cometido.
- Hidrus, en un principio iba a ser una aplicación en NetBeans pero por cuestión de tiempo se ha realizado una adaptación de MainTester de BlinkCou para poder comprobar el adecuado funcionamiento del sistema en las dos direcciones.

Capítulo 2

ANTECEDENTES

En este capitulo haremos referencia a todos los conocimientos que de una forma directa o indirecta hemos que adquirir o completar información para poder dar sentido a la tecnología utilizada en este proyecto.

2.1. ESTADO DEL ARTE

Habitualmente los sensores que nos encontraremos en los sistemas electrónicos no tienen ninguna capacidad de procesado limitándose a funcionar como transductor realizando una medición de una variable y enviando la información a un procesador central. Sin embargo la llegada de los sensores inalámbricos representa un gran salto, ya que estos si tienen capacidad de computación, pudiendo organizarse y comunicarse entre ellos lo cual hace aumentar exponencialmente sus posibilidades.

Un sistema WSN (Wireless Sensor Network) de dispositivos sensores sin cables son redes de control de las condiciones de diferentes puntos, temperatura , gases, humedad, campo magnético, presión o intensidad lumínica. Estos dispositivos denominados comúnmente como motas, son autónomos siendo dependientes solo de la autonomía de sus baterías.

.

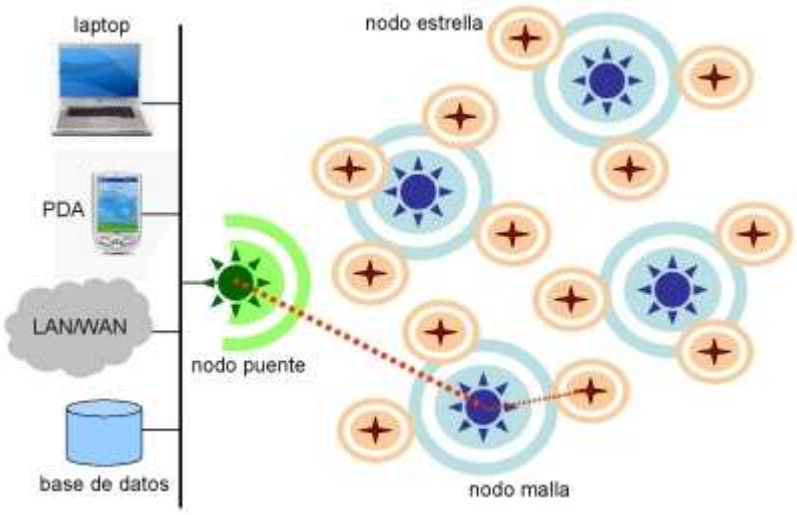

Figura 7. Red de sensores inalámbricos

2.2. NODOS SENSORES (MOTAS)

Los nodos sensores, mas conocidos como Motes o Motas, son dispositivos sensores inalámbricos con unos recursos computacionales y de comunicación limitados, suelen disponer de microcontroladores de 8 ó 16 bits y con pocos kilobytes de RAM. Su radio de baja potencia puede enviar ciento de kilobits por segundo. En la figura 8 se pueden observar los distintos componentes que básicamente están compuestos por transceptor, que dispone del órgano emisor y receptor, una memoria, microcontrolador, sensores y alimentación mediante baterías que le dotan de una autonomía energética que puede permitirle funcionar ininterrumpidamente durante semanas, meses o años, por supuesto dependiendo del tiempo de uso.

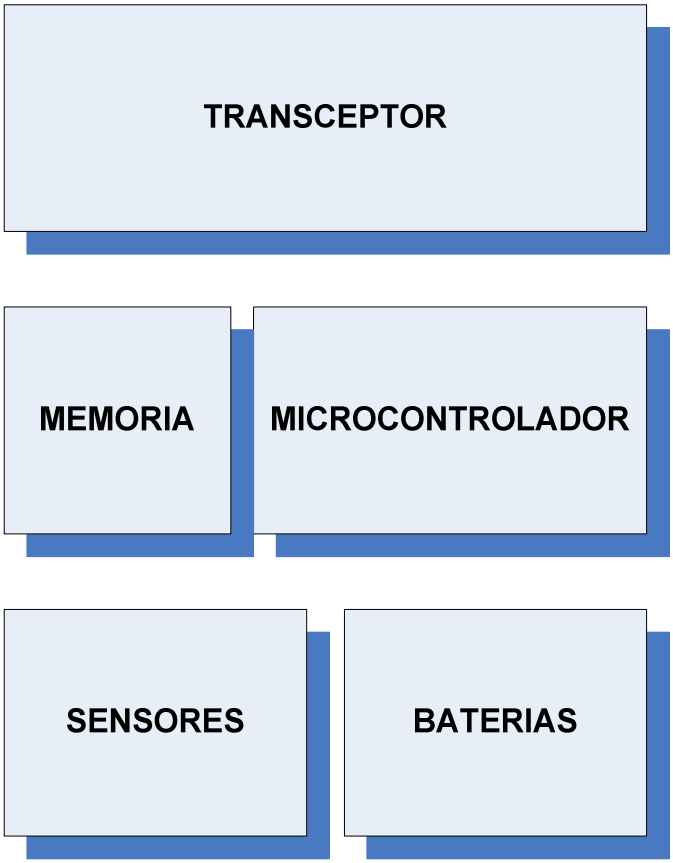

Figura 8. Componentes básicos de un nodo sensor

El equipamiento de sensores es muy variado en cuanto a numero y tipo pudiendo ir prevista con medidores de temperatura, humedad, de campo magnético, gases, luminosidad etc. Como tónica general el dispositivo estará en etapa de bajo consumo o dormido hasta que cualquier evento programado lo invoque, realizará entonces su tarea y vuelve de nuevo al estado inicial de bajo consumo. Los nodos sensores pueden configurar una red sin disponer de una infraestructura previa, con distinta disposición según la forma de interconectarse, dando lugar a redes en:

- Estrella, los nodos tienen esta forma de disposición, donde se emite a un nodo central que funciona como una pasarela.
- Anillo, si la disposición de los nodos en el espacio lo permite seria la topologia mas natural.

- Multisalto, en esta topología cada nodo elige su vecino mas indicado y sobre él realiza su transmisión y de él recibe la información.
- Por grupos, se realizan agrupamientos de nodos que transmiten sobre uno de ellos que formaría parte de un nivel superior y transmitiría sobre los de su misma condición . Este sistema ofrece al posibilidad de utilizar topologías distintas en cada clase.

2.2.1 MOTES COU_1_2 24 A2

La UOC, para la realización de este proyecto nos suministra dos nodos sensores equipado con sensor de temperatura, luminosidad y detector de efecto hall. Una de ellas tendrá como misión servir de pasarela entre la mota situada en el medio o zona de detección y el PC de monitorización de datos. Analizaremos el trafico de datos en una y otra dirección así como la capacidad de procesamiento del nodo sensor de Zona. Los elementos mas importantes de este dispositivo son:

- Modulo Wireless ZigBit de 2,4 Ghz, compuesto de un microcontrolador ATMEGA1281, un transceiver AT86RF230 RF y un chip Antena.
- Chip puente de USB a Serial
- Sensor de temperatura MCP 9701/9701 A , con un rango de temperatura de que va desde los -10 hasta los +125.
- Sensor de luminosidad PDV –P9003-1 se trata de una fotocélula fotoconductiva capacitada para sensar de 400 a 700 nm.
- Sensor de Efecto Hall para detectar la presencia de campos magnéticos.
- Les de tipo rojo, naranja y verde.
- Dos pulsadores, uno de reset y otro de usuario.

En la siguiente fotografía (figura 9) podemos observar la situación de cada elemento en el COU-1-2

Tabla 1

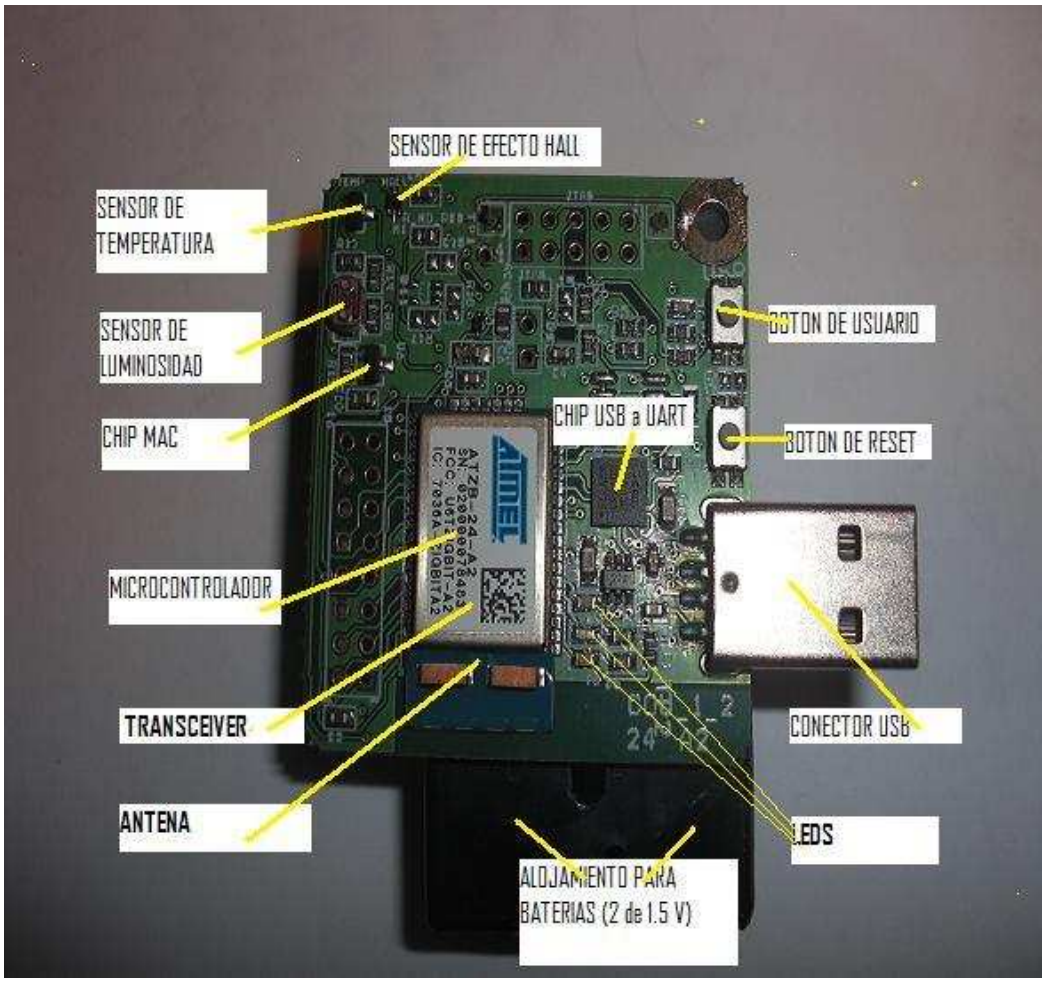

Figura 9. Situación de los elementos primordiales del COU_1_2 24 A2

2.3. TINYOS

Se trata de un Sistema Operativo de libre distribución basado en componentes, pensado especialmente para sistemas inalámbricos empotrados, una de sus peculiaridades es la poca utilización de memoria y adaptarse perfectamente a las características de baja potencia que tienen los microcontroladores que incorporan las motas.

Las aplicaciones y sistemas Tinyos , así como el Sistema Operativo en sí, utilizan como lenguaje de programación , una dialecto del lenguaje C denominado NesC, en el cual profundizaremos mas en el siguiente apartado.

Para la programación fácil de sistemas y aplicaciones Tinyos nos provee de:

- Un modelo de componentes, que define la forma de escribir pequeñas piezas reutilizables de código y componerlas en grandes abstracciones. Conectar los distintos componentes usando un de especificaciones es lo que dará la definición de una aplicación. En la siguiente figura (figura 10) se puede observar como las aplicaciones son básicamente agrupamientos de componentes interrelacionados formando componentes mayores que son las aplicaciones.

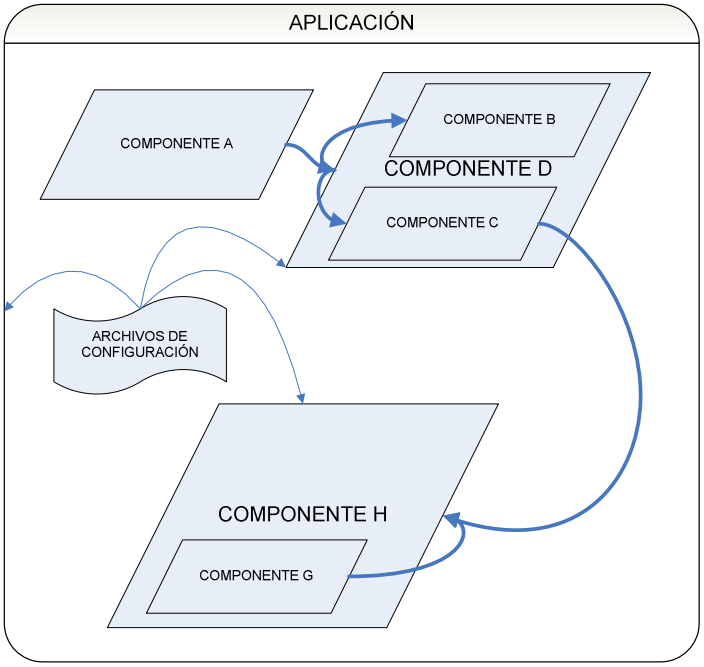

Figura 10. Idea general de Aplicación Tinyos

- Un modelo concurrente de ejecución, que define componentes con distinta sincronía de computación.
- Además de un amplio conjunto de interfaces, servicios, librerías, programas de pruebas, documentación y una estructura general de componentes que simplifica la escritura de nuevas aplicaciones y servicios

Para comprender como funciona este sistema operativo debemos tener en cuenta sus elementos básicos y sus relaciones:

- Los eventos, estos siempre tienen prioridad sobre cualquier tarea, teniendo en cuenta que un evento expulsa a otro ya que solo existe una pila compartida.

- Las tareas, como su propio nombre indica son pequeñas acciones que desarrollan un trabajo de forma asíncrona y se realizaran si no hay ningún evento que reclame la línea de ejecución. Las tareas se guardan en cola, una vez todas ejecutadas el procesador pasara a posición de espera, que se romperá con la activación de algún evento. Hay que tener en cuenta que las tareas se ejecutan en su totalidad y no tiene prioridad sobre otras tareas

- Los comandos, dentro de su componente, realizan llamadas a otros componentes de capa inferior, normalmente para realizar alguna operación. Su uso puede ser de dos tipos, uno para realizar una petición, haciéndose después cargo el planificador de la misma, para terminar señalizando con un evento, el éxito de la operación. Y otro tipo que sea el comando el que haga la operación completamente y no tendremos el evento retornado.

Otra característica de Tinyos es que las aplicaciones que podemos crear se fusionan implícitamente con el propio sistema operativo en tiempo de compilación formando un conjunto de bajo peso que se instalara en el nodo sensor, por lo tanto cada vez que tengamos que realizar modificaciones ó actualizaciones seguiremos volcando en el dispositivo esta fusión de aplicación-Tinyos que compone una aplicación ejecutable. En la siguiente figura, 11, se puede observar la interacción entre los procesos.

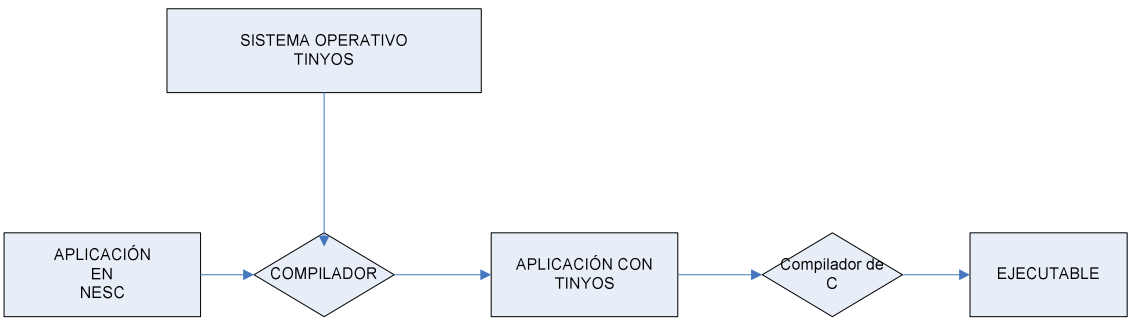

Figura 11. Proceso de compilación de una aplicación en Tinyos

2.3.1. APLICACIONES EN TINYOS: COMPONENTES

Las aplicaciones Tinyos son construidas por componentes, esta provee y usa interfaces. Estas interfaces son el único punto de acceso a la componente. Cada componente estará compuesta de un espacio de memoria y un pila con tareas a realizar. Hay que distinguir dos tipos de componentes por una parte están los primitivos que son los que proporciona el propio SO y los complejos que son aquellos proporcionados por terceros, contribuciones ó librerías.
A continuación se detallan los cuatro elementos que conforman una componente:

1. Controlador de eventos.
2. controlador de comandos
3. Un bloque con información del estado de ejecución del componente y posición de las variables del mismo.
4. Un bloque con la pila de tareas simples.

Un ejemplo de las distintas partes seria la siguiente figura 12 que refleja la composición del componente Active Messages (AM).

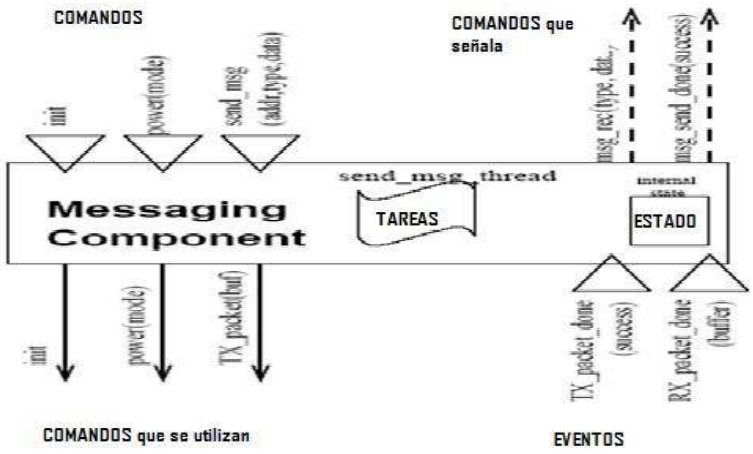

Figura 12. Disección del componente Active Messages

Los componentes pueden ser de varios tipos según sea su función dentro de la aplicación y la relación que tengan con otros componentes relacionados con el hardware o software.

La clasificación sería:

- Componentes de alto nivel, seria viendo la figura 13 el componente ActiveMessages y todos aquellos con funciones de control, enrutamiento o de ejecución de cualquier tipo de calculo.
- Componentes de Hardware sintético, son aquellos que formando parte del software simulan el comportamiento del hardware un ejemplo sobre la figura 9 seria Radio Byte este intercambia datos de entrada o salida del módulo RFM y señaliza cuando un byte ha sido completado.
- Componentes de bajo nivel son aquellos que trabajan directamente sobre el hardware, ya que actúa directamente sobre los pines de entrada y salida a los que esta conectado, el ejemplo seria el modulo RFM.

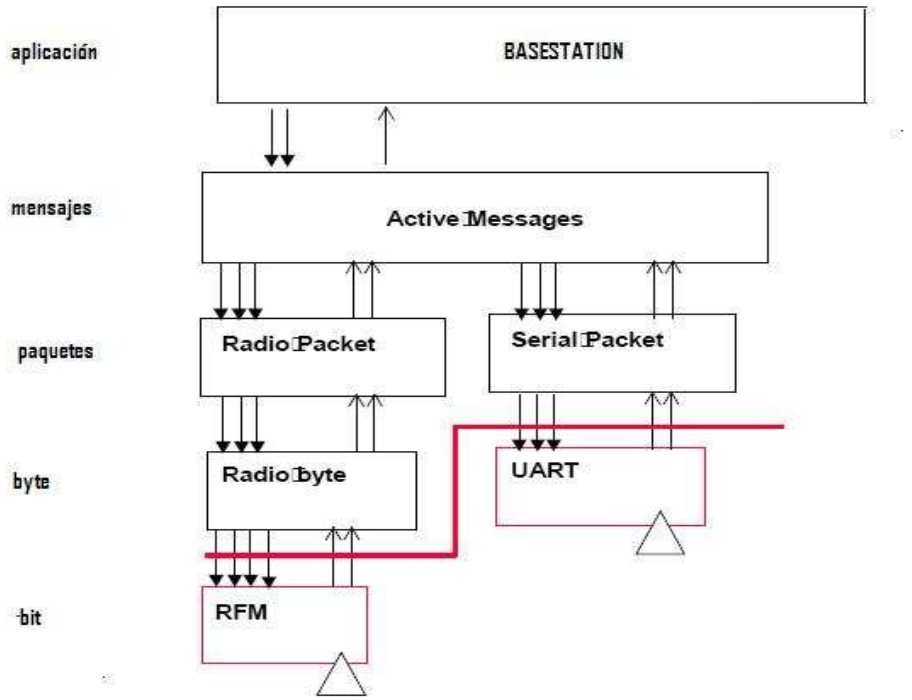

Figura 13. Grafo de componentes de BaseStation

2.3.2 LENGUAJE DE PROGRAMACION NESC

NesC es el lenguaje que se ha utilizado para la realización de este Trabajo Final de Carrera y se utiliza para crear aplicaciones para sistemas inalámbricos embebidos ejecutando el sistema operativo Tinyos. Se trata de un lenguaje orientado a componentes y mas particularmente a interfaces y a eventos. El ensamblado continuo de componentes muchos de ellos ya implementados lo que hace es favorecer al programador a la hora de implementar código y solo tener que centrarse en la funcionalidad del dispositivo.

Desde el punto de vista de la codificación los componentes deben poseer tres secciones que son :

- Configuración e Implementación. Estos deben ir en un fichero que en este proyecto van con el nombre del componente y al terminación AppC como si fuese una aplicación por si solo. Es aquí donde se definen las conexiones de los componentes que se usen, ante todo en relación con las interfaces utilizadas por estos, que deben ser declaradas.

- Modulo. Este fichero en este proyecto termina con el nombre del componente y una C mayúscula al final. Es aquí donde se definen principalmente las interfaces que se usan y aquellas que se proveen y donde se realiza la implementación de las variables globales, tareas y eventos a realizar.

- Fichero de cabecera. Es importante disponer de un fichero de cabecera donde agrupar todas las enumeraciones, registros o tipos de datos creados por el usuario y de los que hace uso la aplicación. Después debemos incluir su referencia en los archivos del Modulo y de Configuración.

- Makefile. trata de un fichero de texto que contiene las reglas de compilación así como las herramientas que se utilizan si se precisa en la misma, para poder crear los ficheros de tipo Java, C, C++ o Python, que son lenguajes soportados por Tinyos; el hecho de generar código fuente en estos lenguajes se debe generalmente a la necesidad de facilitar el intercambio de datos entre la mota y el PC

Habitualmente encontraremos varios tipos de funciones en el fichero de implementación, estas son:

- Tareas (task), son funciones que se ejecutan de forma concurrente. Están declaradas inicialmente con las variables de esta forma " task void Tarea " y después invocadas desde cualquier parte con la palabra reservada "post Tarea", una vez que son llamadas saltamos a su ejecución retornando una vez finalizadas al programa invocador.

- Eventos (event), son llamadas ejecutadas cuando se activa alguna señal del sistema, al ser nesC orientado a eventos le confiere a estos prioridad sobre las tareas. Normalmente saltan como respuesta a algún tipo de llamada a una interface, por ejemplo un temporizador, un envío de Radio, etc. si quisiésemos llamar a un evento de forma manual tiene que ser con la palabra reservada "signal", aunque no es muy habitual

- Los comandos (command), son funciones respuesta a llamadas efectuadas sobre una interface, a la cual pertenecen y aportándonos sus datos para la ejecución de nuestro código. Desde el código la invocación se realiza con la palabra reservada "call".

Si no se indica lo contrario son síncrono, pero si en su declaración anteponemos la palabra reservada " async " de ejecución es asíncrona, que es aquella que se ejecuta cuando se produce una señal de hardware como por ejemplo disponer del canal para la lectura de un sensor, o una interrupción provocada por la detección de un sensor.

Las variables en NesC son muy similares al lenguaje C, con algunas particularidades:

- Cualquier variable declarada por un componente es privada, no se puede acceder a ella directamente. La única forma que pueden tener los componentes de interactuar es mediante interfaces, cuestión que a mi personalmente me costo descubrir.
- Si una función asíncrona accede a una variable global esta debe declararse con la palabra reservada "norace " de esta forma evitamos warnings de compilación al declarar una exclusión mutua.

- Si a las variables le anteponemos la palabra reservada “atomic” estamos protegiendo esa variable de posibles modificaciones de otros hilos de ejecución.

2.3.3 MIG para NESC

Otra de las herramientas utilizada en el proyecto es MIG, con ella se puede crear una interfaz de forma automática en Java, Python o C, construyendo el objeto a partir del paquete que recibe. Para ello debemos implementar la utilidad en el archivo makefile, aunque antes debemos poseer una estructura de mensajes en un archivo de cabecera.

Figura 14.Partes de la implementación MIG en el Makefile

En la figura 14 vemos las partes mas importantes de una muestra de la implementación de la herramienta MIG, a partir de ese momento ya tendremos generados los archivos punto java y

punto class que podremos utilizar en nuestra aplicación Java, hemos conseguido generar de forma fácil las clases para codificar y descodificar paquetes con los mensajes TinyOS.

Es importante tener en cuenta, por que a mi me ocurrió, que al igual que con javac debe haber un tabulador antes de MIG y no espacios.

Para cada campo nombre de la estructura de tipo de mensaje , se encuentran los siguientes métodos (el bit de Offset y métodos son útiles para el tamaño de las estructuras que contienen campos de bits):

- get_ nombre : obtiene el valor del campo
- set_ nombre : establece el valor del campo
- offsetBits_ nombre : cambio de desplazamiento de campo en el mensaje
- offset_ nombre: bytes de desplazamiento de campo en el mensaje
- sizeBits_ nombre : tamaño en bits del campo
- size_ nombre: tamaño en bytes del campo
- isSigned_ nombre : devuelve true si nombre es de un tipo con signo
- isArray_ nombre : devuelve true si nombre es una matriz

2.4. ESTUDIO DE MERCADO

La mayoría de las aplicaciones con tecnología TinyOS para redes de sensores inalámbricos se centran especialmente en determinados campos, como por ejemplo:

- Aplicaciones en el ámbito medico sanitario, destinadas al cuidado de las personas mayores, control y localización de pacientes dentro del hospital, toma continua de parámetros médicos a pacientes, pulsaciones, niveles de azúcar en sangre etc. Con este tipo de aplicaciones sabemos exactamente en que posición del hospital se encuentra el paciente así como generaremos alarmas ante cualquier anomalía en su salud.

- Aplicaciones para entornos agrícolas, sobretodo muy relacionados con la agricultura de precisión, aquella que requiere de unas condiciones ambientales especiales. es posiblemente un campo donde hay ya aplicaciones en marcha y con muy buenos resultados. Por ejemplo en el control de invernaderos, por la importancia que tiene el mantener constantes unas determinadas variables en el ambiente. También se suelen emplear para el control de viñedos, para alarmar en condiciones de heladas.

- Medición de variables Naturales, aplicaciones para la detección de incendios, actividades de los volcanes o movimientos de tierra ó pozos petrolíferos son algunas de las posibilidades donde trabajan, su bajo consumo, su mimetismo con el entorno y su autonomía nos permiten tener datos en lugares de difícil acceso y de alto valor ecológico.

- En aplicaciones militares también están ya integrados, siendo usados para localización de tropas ó vehículos y seguramente otras posibilidades de las que es difícil tener conocimiento por el secretismo que siempre reina en este ámbito.

- Para entornos Urbanos o Públicos, como podrían ser control de plazas en aparcamientos, control de trafico, video vigilancia, localización de vehículos públicos.

- En el hogar para el control de la iluminación, climatización, accionamiento de actuadores por ejemplo el cierre y apertura de persianas. Sin duda en este campo es donde mas tipos de aplicaciones se están desarrollando quizás por que el control sobre el pequeño electrodoméstico o dispositivos de poca complejidad es mas fácil de implementar. es aquí donde encuentro aplicaciones mas parecidas a lo que yo he intentado implementar.

- En el sector transportes esta teniendo bastante incidencia las continua aparición de aplicaciones tanto para control de la carga, analizar su situación en cada momento así como la evolución de las condiciones ambientales durante el desplazamiento y posibles variaciones en el producto.

2.4.1 SECTORES A LOS QUE VA DIRIGIDO EL SISTEMA

Hay muchos mas campos o ejemplos de aplicaciones con redes de sensores inalámbricos, pero donde menos sistemas he encontrado ha sido en el ámbito de las instalaciones comerciales e industriales, donde se utilizan redes de sensores muy parecida como por ejemplo WirelessHart, que aun tratándose de una tecnología inalámbrica, se trata de un estándar completamente distinto, solo se parecen en la tecnología radio utilizada.

Mi intención es la implementación de una tecnología TinyOS dentro del campo de las instalaciones industriales y comerciales realizando una función similar a la de WirelessHart con toda su potencia para ello se puede observar en la figura 15 un esquema de instalación con esta tecnología donde combina cableado para bus de campo y transmisión radio para los sensores.

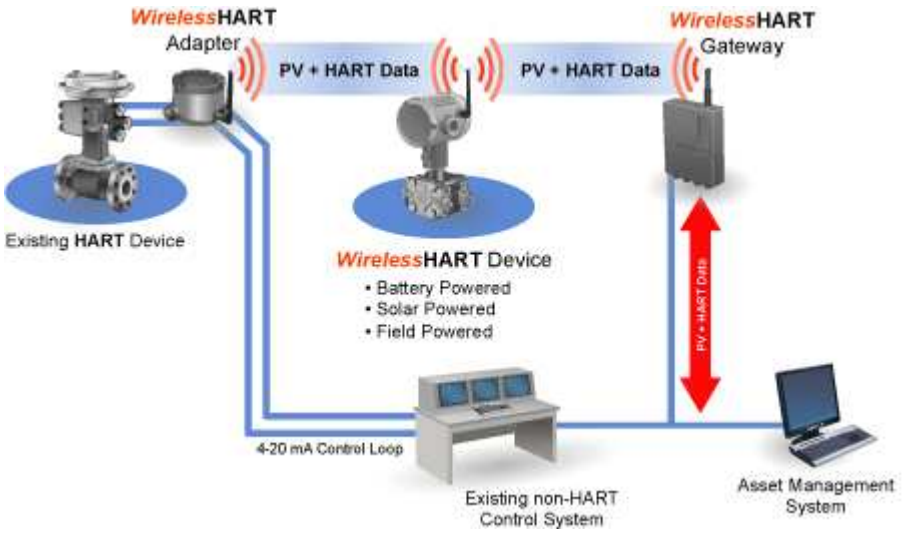

Figura 15. Esquema instalación con tecnología WirelessHart

Capítulo 3

DESCRIPCION FUNCIONAL DEL SISTEMA

El motivo principal de esta aplicación es ser mi base de aprendizaje en el mundo de la programación de los sensores inalámbricos, con la intención de investigar y analizar el comportamiento así como las posibilidades de las Motas dentro del campo de la climatización en especial y del control de instalaciones en general.

3.1. DESCRIPCION DEL SISTEMA DE FORMA GLOBAL

Por ello esta aplicación esta diseñada para realizar el control de una zona comercial, en cuanto a necesidades de potencia y modo de climatización como en el control de la intensidad lumínica según las condiciones ambientales. Siendo esta la base sobre la que desarrollar los conocimientos de los tutoriales de Tinyos así como el aprendizaje adquirido por otras fuentes para poder conocer las capacidades de este tipo de sensores para futuras aplicaciones en Frío Industrial y Climatización o en regulación de aplicaciones eléctricas.

Desde un principio mi idea ha sido la de dotar a la mota de la capacidad de control absoluto, siendo la que efectúa el sensado, ejercita el calculo correspondiente, se intercambian datos sus componentes y decide cuales son las medidas a adoptar. De esta forma pasaría de ser un elemento casi pasivo que solo realiza funciones de sensado y comunicación, mientras otros elementos toman las decisiones, a ser un elemento activo y coordinador ejerciendo la gestión efectiva de la maquinas a las que esta vinculado.

3.1.1 COMPOSICION GENERAL DEL SISTEMA

El sistema esta compuesto por tres aplicaciones, una de ellas instalada en la denominada Mota sensora que se denominará Casiopea. Otra que será instalada en la Mota pasarela situada en el puerto USB del PC que se denominará Scutum cuya base principal y única es la aplicación de Tinyos BaseStation ya que cumplía los requisitos que necesitaba en el proyecto. El tercer pilar denominado Hidrus es una aplicación básica y sencilla, creada y modificada sobre la base del archivo MainTester de la aplicación de estudio COUTester, que realiza las funciones de interfaz de usuario donde podemos observar los datos recibidos y además podemos modificar las consignas de trabajo que inmediatamente son enviadas a la Mota de Zona. Resaltar que para la comunicación entre Scutum en el puerto USB y la aplicación de interfaz de usuario hemos utilizado la aplicación de Tinyos SerialForwarder.

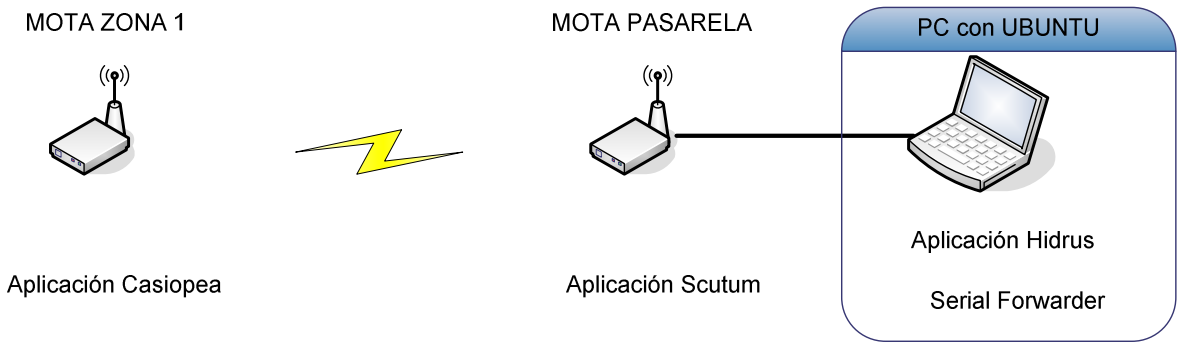

Figura 16. Diseño de Red y sus aplicaciones

En la figura 16 se puede observar la distribución de la red con las aplicaciones asociadas. Desde un punto de vista funcional el diagrama de bloques con las líneas básicas de sus funciones dentro del sistema seria el reflejado en la figura 17.

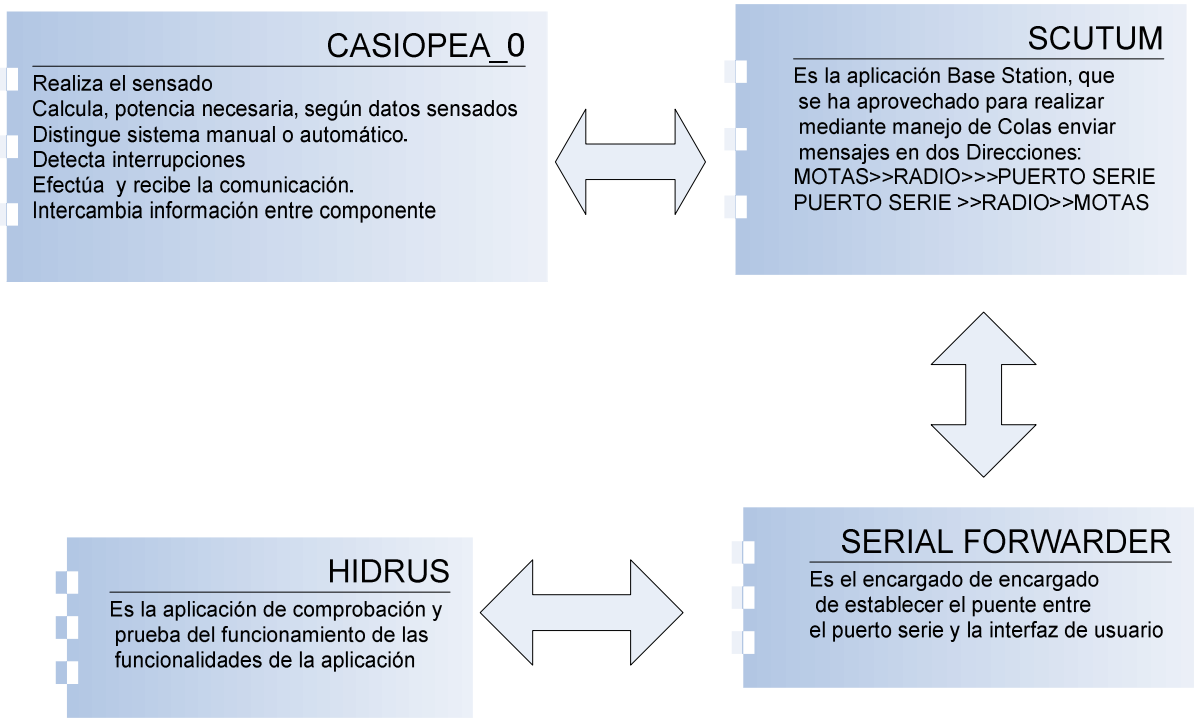

Figura 17. Diagrama de bloques del Sistema

3.1.2. APLICACION CASIOPEA_0

La aplicación Casiopea que esta instalada en la Mota de campo tiene por misión según los componentes implementados

- Recoger los datos de temperatura, luminosidad y carga de su batería. Estos datos serán entregados a otros componentes de la aplicación para su uso, mediante una interfaz propia.

- Calcular el punto PID, al final solo es el punto proporcional por exceso de complicación matemática.. Este punto se calcula según distancia al set. Añadir que dentro de las motas

no trabajamos con temperaturas Celsius sino en unidad counts con la intención de no perder eficacia al realizar el cambio en la mota y la consecuente perdida de decimales. Es en la interfaz de usuario donde se realiza el cambio

- Calcular el valor de la señal que pasar a los balaustres (reactancias de regulación) de los fluorescentes variará según cuatro posiciones o estados, que van desde apagado hasta máxima potencia. También establecemos función manual o automática

- Comunicara el nivel de batería que dispone la Mota, además debe comunicar si ha bajado por debajo de un nivel determinado que consideraremos de peligro.

- Cuando pasamos una tarjeta magnética por la mota Zonal esta comunicara el estado de OFF tanto de la unidad de Climatización como del lineal de Iluminación. El sensado de temperatura así como información de los niveles de batería siguen siendo efectivo. Esta posición de paro se establece para la realización de ajustes o mantenimiento en la unidad de Aire Acondicionado y en la iluminación de la zona. Al mismo tiempo nos avisara que el Efecto de campo ha sido activado. En el momento que pasemos de nuevo la tarjeta magnética el equipo volverá a las condiciones programadas informando que el efecto Campo esta deshabilitado.

- Todos los datos son enviados una vez completados, por un componente ComStation, por radio a la Mota pasarela con la aplicación Scutum.

- Tenemos distintas interfaces creadas en los distintos componentes que sirven para ser utilizados en cualquier momento según las necesidades de la aplicación.

- Disponemos de un componente de comunicación que es el encargado de recibir los sets de trabajo de la aplicación y al mismo tiempo envía los datos.

3.1.3. APLICACION SCUTUM

Esta aplicación esta compuesta por la Aplicación de Tinyos BaseStation. Cumplía perfectamente las necesidades del sistema y solo me he limitado a comprender su funcionamiento de colas. Ha sido una forma de aprovechar y reutilizar código de total confianza.

3.1.4. APLICACIÓN HIDRUS

Esta es la interfaz de usuario, en un principio empecé trabajando sobre una interfaz de usuario con Netbeans pero por problemas de tiempo, además de las dificultades que me provocaba el utilizar Java me han obligado a emplear como base el fichero de CouTester y realizar una interfaz de prueba del sistema. De esta forma podremos ver los datos enviados por la Mota y enviarle las consignas a las que debe funcionar.
Trabajara en colaboración con la aplicación SerialForwarder, estableciendo un enlace entre las dos, mediante TCP/IP.

Consta de dos partes:

- Una vez iniciada la aplicación por consola hace una solicitud de consignas de trabajo, una vez introducidas, recibe los mensajes pendientes y los muestra.

- Envía las consignas que nosotros hemos nosotros hemos programado previamente y se pueden ver las modificadas en breves mensajes por pantalla.

3.2. DISEÑO INTERFAZ DE USUARIO (APLICACIÓN HIDRUS)

En un principio se trabajo con la idea de implementar una interfaz de usuario en Netbeans (que ya estaba en proceso avanzado, figura 18), lo que hubiera ofrecido un resultado mas visual al sistema, pero la falta de tiempo para poder realizarla con suficientes garantías sin provocar retraso en el resto del proyecto me obligo a optar por implementar una sencilla aplicación por consola

que permitiese ver los resultados de monitorización de datos y la recepción de consignas por parte de la mota remota

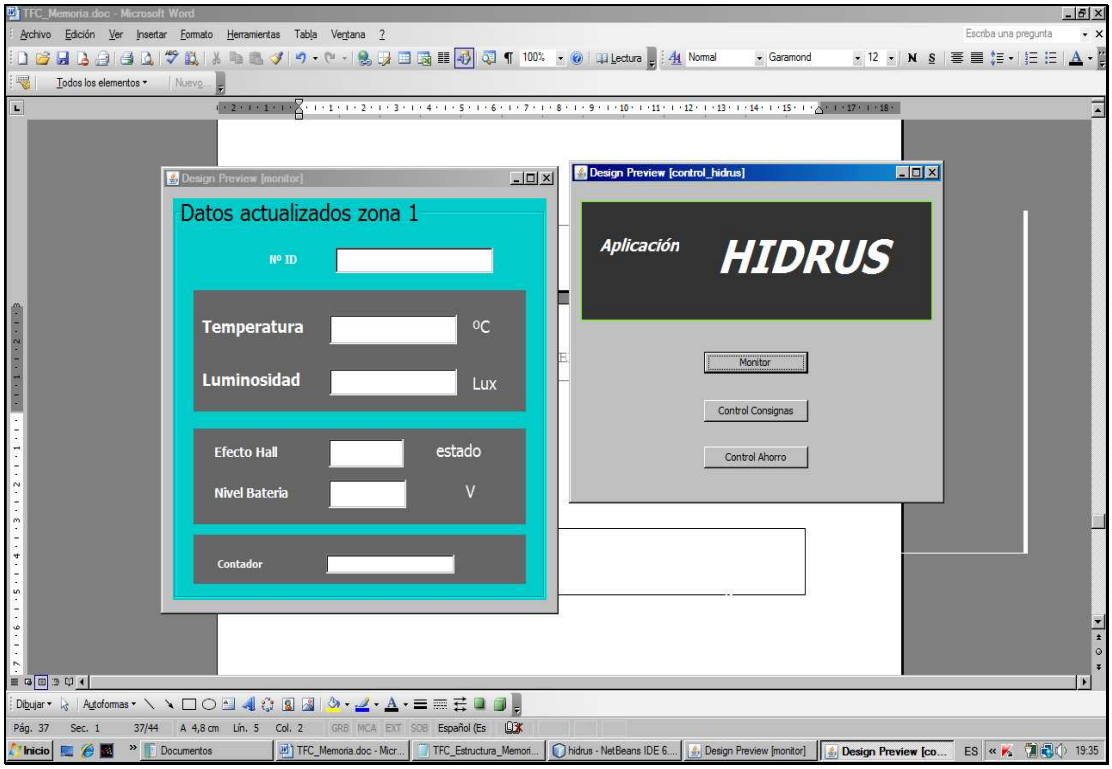

Figura 18. Futura interfaz de usuario Hidrus

Por lo tanto la aplicación Hidrus es una derivación de la aplicación de prueba MainTester ofrecida para aprendizaje por el equipo docente, como ya explique anteriormente.

3.2.1. DIADRAMA DE BLOQUES DE HIDRUS

En esta aplicación tenemos dos grupos diferenciales, los ficheros creados con la herramienta MIG, que genera una clase java en este caso para codificar o descodificar un paquete TinyOS basándose en la estructura de datos creada Msgs.h, para cada campo de la estructura de tipo mensaje se encuentran una relación de métodos para modificar valores, acceder a los mismos, obtener tamaño de bits, etc.

En la aplicación Casiopea ya hemos generado los ficheros punto class y punto java que utilizaremos MainMidrus que es la aplicación principal, en la figura 19 podemos observar el diagrama de bloques.

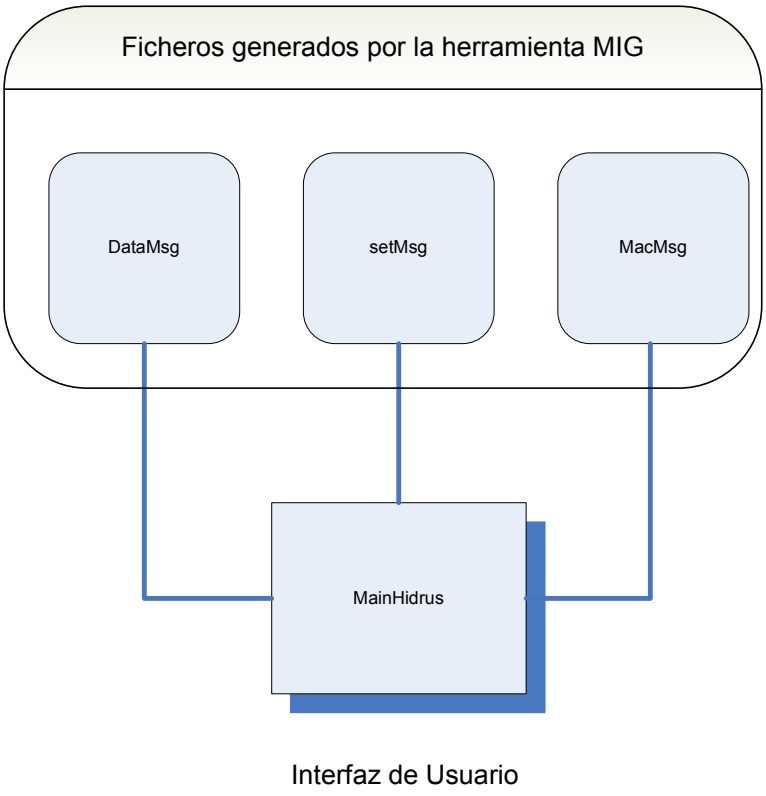

Figura 19. Bloques de la aplicación Hidrus

MainHidrus es la aplicación que se encarga de monitorizar por consola los datos recibidos, pero antes en el inicio del programa nos solicitará las consignas de trabajo tanto de temperatura como los modos de funcionamiento que el usuario establece.

3.2.2. FUNCIONAMIENTO DE LA INTERFAZ

Esta aplicación nos pide al principio que introduzcamos las consignas de funcionamiento, que serán enviadas cuando efectuemos el primer envío de datos. Por lo tanto la aplicación comienza Monitorizando datos de consignas anteriores, esto se ha diseñado para ver las variaciones solo es meramente didáctico. La figura 20 hace referencia a la petición por consola de los datos de

funcionamiento, al mismo tiempo hacemos referencias a los valores recomendados y que es lo que hay que introducir por teclado.

```
jose@jose2:/opt/tinyos-2.x/apps/Hidrus$ java MainHidrus
*********************************************************
*********************************************************
SISTEMA DDE AHORRO ENERGETICO PARA LOCALES COMERCIALES
------------------------------------------------
------------------------------------------------
INTERFAZ DE USUARIO HIDRUS
------------------------------------------------
------------------------------------------------
AÑADIR CONSIGNA tª VALORES RECOMENDADOS (DESDE 306 hasta 316 >>21 a 24 ºC):
308
SETPOINT TEMPERATURA= 308
AÑADIR MODO CLIMATIZACION ( 0=AUTOMATICO, 1 COOL, 2 HEAT, OFF ):
0
MODO CLIMA= 0
AÑADIR CONTROL ALUMBRADO (0= AUTOMATICO, 1= 100% POTENCIA, 2= OFF:
0
MODO ILUMINACION= 0
Enviamos un mensaje con consignas a la Mota: numero mensaje= 0
******************************************************************************
********
```

Figura 20. Cargar valores de consignas

Con el envío de los datos de SET ya recibimos las modificaciones emitidas por la Mota, en la siguiente figura la 21 se puede observar los distintos datos

```
BATTERY: 1412   EFECTO HALL INACTIVE    NIVEL DE BATERIA: OKEY NIVEL
****************************************************************************************
ID MOTE: 1      COUNTER: 194
TEMPERATURE: 24,9       PID : 27,0%     AIR CONDITIONING MODE: HEAT
DATE MODE_TEMP: 1       DATE MODE_LIGHT : 7     TEMPERATURA EN COUNTS: 319
PHOTO:254       MODE LIGHT SYSTEM:STANDARD ILLUMINATION
BATTERY: 1412   EFECTO HALL INACTIVE    NIVEL DE BATERIA: OKEY NIVEL
****************************************************************************************
ID MOTE: 1      COUNTER: 194
TEMPERATURE: 24,9       PID : 27,0%     AIR CONDITIONING MODE: HEAT
DATE MODE_TEMP: 1       DATE MODE_LIGHT : 7     TEMPERATURA EN COUNTS: 319
PHOTO:254       MODE LIGHT SYSTEM:STANDARD ILLUMINATION
BATTERY: 1412   EFECTO HALL INACTIVE    NIVEL DE BATERIA: OKEY NIVEL
****************************************************************************************
ID MOTE: 1      COUNTER: 194
TEMPERATURE: 24,9       PID : 27,0%     AIR CONDITIONING MODE: HEAT
DATE MODE_TEMP: 1       DATE MODE_LIGHT : 7     TEMPERATURA EN COUNTS: 319
PHOTO:254       MODE LIGHT SYSTEM:STANDARD ILLUMINATION
BATTERY: 1412   EFECTO HALL INACTIVE    NIVEL DE BATERIA: OKEY NIVEL
```

Figura 21. Monitorización de Datos

Como se puede observar recibimos por filas:

1. La identidad de la mota y contador de mensajes.
2. Temperatura ambiental, tanto por ciento en el control de capacidad de la unidad de climatización, el modo en que nos hayamos de funcionamiento en este caso hemos solicitado una consigna elevada y estamos calentando el ambiente.
3. En la tercera fila recibimos solo datos de comprobación, para comprobación del programa en la mota, numero de modo de temperatura, numero de modo calefacción y unidades counts de calculo en el nodo sensor.
4. Recibimos sensado de luminosidad y posición calculada de funcionamiento.
5. Batería, capacidad en este momento de la misma en milivoltios y estado del sensor Hall, en el apartado correspondiente veremos su forma de funcionamiento y un sistema de aviso de alarma en caso de que el estado de la batería estuviese por debajo de un determinado valor

Esta implementado para recibir la dirección Mac y numero de identificación de la Mota sensora una vez pulsado el botón de usuario, pero no como algo importante en el sistema, simplemente para probar en una red la comunicación de cada mota y para futuras implementación de alarmas por interrupción.

3.3. DISEÑO APLICACIÓN MOTA ZONA1 (APLICACIÓN CASIOPEA)

La aplicación Casiopea es la parte principal del sistema esta construida sobre la base de que todos los componentes que forman la aplicación interactúen entre ellos, sobretodo en posibles fases de crecimiento de la misma. Podemos observar la relación de bloques en la figura 22, junto con la interfaz para compartir datos que cada bloque aporta.

3.3.1. DIADRAMA DE BLOQUES DE LA APLICACIÓN CASIOPEA

Los distintos bloques que podemos observar dentro de la aplicación están perfectamente diferenciados en la figura 22, el componente NexoMain es el que agrupa todos los módulos y

realiza la salida exterior como aplicación, por lo tanto todos se dirigen hacia ese punto. La utilización de las interfaces asociadas a los distintos componentes es la idea de crear una aplicación abierta que dentro de sus posibilidades se hacer crecer.

Se puede observar que es una aplicación muy modular, ya que se hizo la construcción así con al idea del reaprovechamiento de código para otras aplicaciones de control. La formula modular permitirá quitar módulos o añadirlos según deseemos adaptar el código creando aplicaciones distintas. Por eso se puede observar por los nombres de los ficheros que son como pequeñas aplicaciones juntas y acopladas, que como explique en el Capitulo dos es la base de TinyOS.

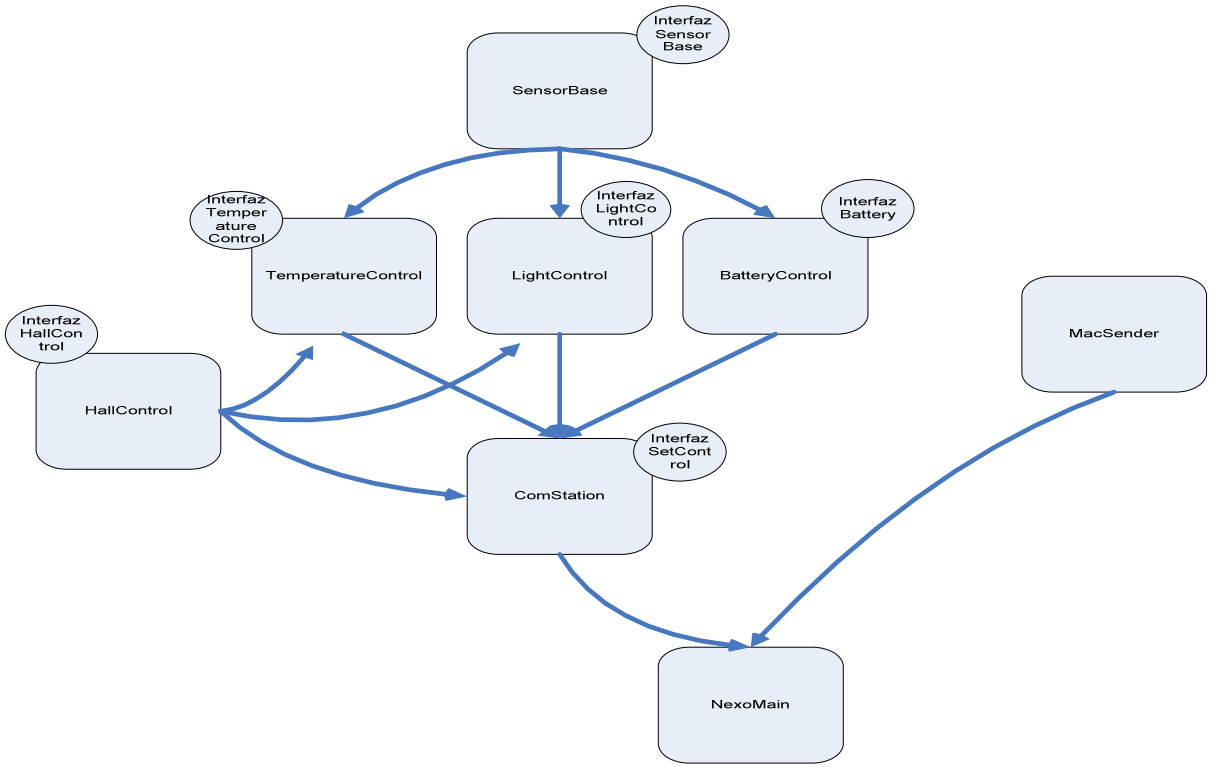

Figura 22. Diagrama de Bloques de la aplicación Casiopea

3.3.2. FUNCIONAMIENTO DE LA APLICACION

El funcionamiento de la aplicación en general depende de la aportación de cada uno de los componentes de la misma por ello en la figura 23 tenemos el flujo de comunicación de la aplicación de una forma sencilla y escueta, que nos muestra el sentido y la interrelación de los módulos según su función.

- **SensorBase**, su misión es simplemente recoger los datos de temperatura, luminosidad y batería, asociado a cada uno de los canales y mediante la interfaz SensorBase retornará ese valor a quien lo necesite dentro de la aplicación.

- **TemperatureControl**, su actividad principal es recoger el dato de temperatura obtenido y realizar distintos cálculos que serian:

    - Calcular potencia Climatización según diferencia con consigna, hacerlo de forma elástica para evitar arrancadas y paradas de compresores y ventiladores ya que estos son los puntos de mayor consumo. Las variaciones de amortiguadas permitirán llegar a consigna de forma acompasada evitaran problemas de estratificaciones de aire y aumentaremos el tiempo de vida de los elementos frigoríficos.
    - Elegir modo, si estamos en versión AUTOMATICA el sistema elige si enfría, calienta o solo ventila dependiendo de la distancia con el setpoint.
    - Si recibe dato de activación de sensor Hall para la Climatización, el calculo proporcional estará a cero.
    - En consigna de MANUAL elegirá frío, calor ó OFF dependiendo de la petición recibida del cliente, esta posición es incluida porque en todos los sistemas de regulación automática nunca se debe dejar totalmente el control al dispositivo.
    - Retornara a través de su interfaz los datos tanto de temperatura en counts, como PID de funcionamiento y el modo de funcionamiento (FRIO, CALOR, VENTILACION) en que se encuentra.

- **LightControl,** es el responsable de una vez recogido el dato de luminosidad y reciclado a unidades lux de:

    - Aplicarle un rango de señal si este esta entre unos determinados valores establecidos. este rango de señal seria para establecer un dato de 0 a 10 Voltios a las reactancias de control. En la interfaz de usuario cambiamos ese valor por un dato mas visual (ILUMINACION ESTÁNDAR, ILUMINACION MINIMA etc.).
    - En caso de activación Hall, se debe producir detección del sistema.
    - Retornamos a través de la interface propia el dato de luminosidad y la señal calculada.

- **BateryControl**, se ocupa simplemente de adaptar el dato del sensor de batería y además calcula:

    - En caso de bajar el nivel de batería de valor determinado lo comunicara al usuario
    - Retorna por su interfaz el dato de batería en milivoltios y además un estado para que el usuario sepa si debe sustituirlas con urgencia o no.

- **HallControl**, se ocupa simplemente de crear una interrupción en caso de estar sometido a un campo magnético, aprovechamos la variación del estado del Led para obtener una variable de estado, ocurriendo lo siguiente:

    - Con campo magnético se apaga Led rojo en mota de zona1, se envía estado por interfaz para paro climatización e iluminación, ese estado se comunica en la interfaz de usuario.
    - Sin campo magnético se enciende Led rojo en mota de zona1, se envía estado por interfaz para activación climatización e iluminación, situándose en las consignas preestablecidas antes de la detección Hall-efect ese estado se comunica en la interfaz de usuario.

- **MacSender,** para futuras implementaciones solo envía identidad mota, y MAC de la misma.

- **ComStation**, su responsabilidad es la comunicación, es por lo tanto una estación de comunicación envía y recibe datos, a los que accede o por acceso propio como componente o a través de las distintas interfaces a las que tiene acceso. Le he dotado de una característica por no implementar un componente mas, y es que provee una interfaz con los consignas de usuario remitidas por la aplicación Hidrus para que sean utilizadas por el resto de componentes.

- **NexoMain**, es el componente que agrupa toda la aplicación.

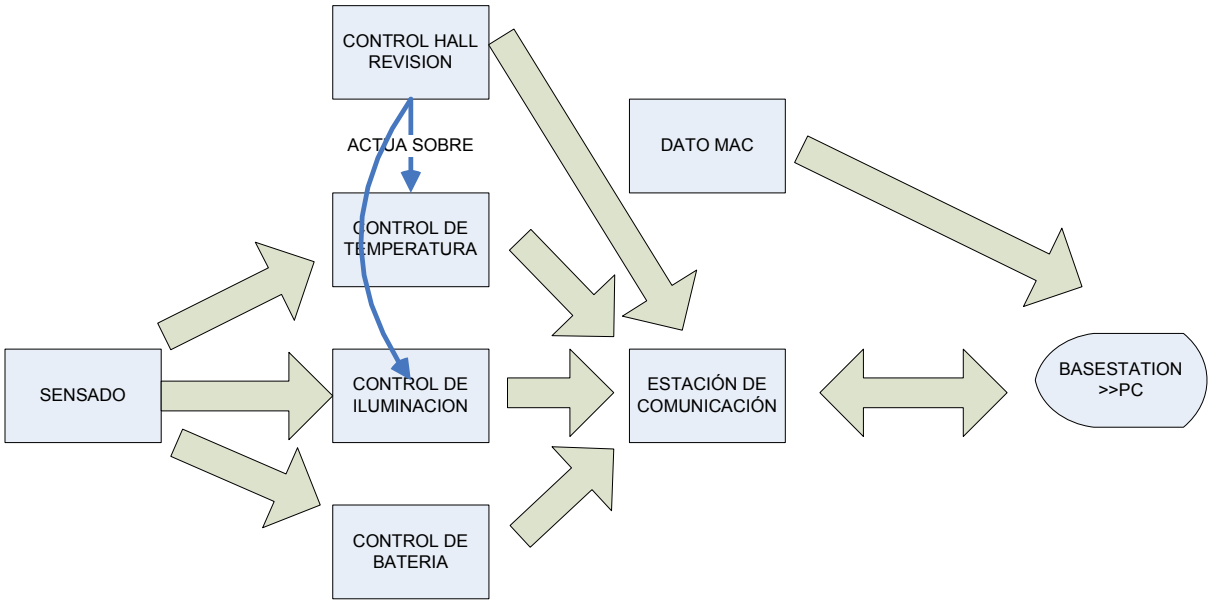

Figura 23. Diagrama de Bloques de la aplicación Casiopea

Capítulo 4

DETALLES TECNICOS DEL SISTEMA

4.1. COMPOSICION APLICACIÓN CASIOPEA

Analizaremos los distintos módulos de la aplicación y observaremos sus detalles de composición interna, para tener referencia de que componentes dispone, que tipo de interfaces, aprovechando el componente grafico de ECLIPSE.

4.1.1. NexoMainAppC

Como se puede observar en la figura 24 la aplicación NexoMainAppC es la que agrupa los distintos componentes de la aplicación, se trata como explique en apartados anteriores de un

modulo de reagrupamiento para esta aplicación en concreto. Su fichero de configuración nos muestra los distintos componentes, con dos temporizadores y como su modulo del mismo nombres solo utiliza la interfaz Leds y Boot, pertenecientes a los componentes MainC y LedsC

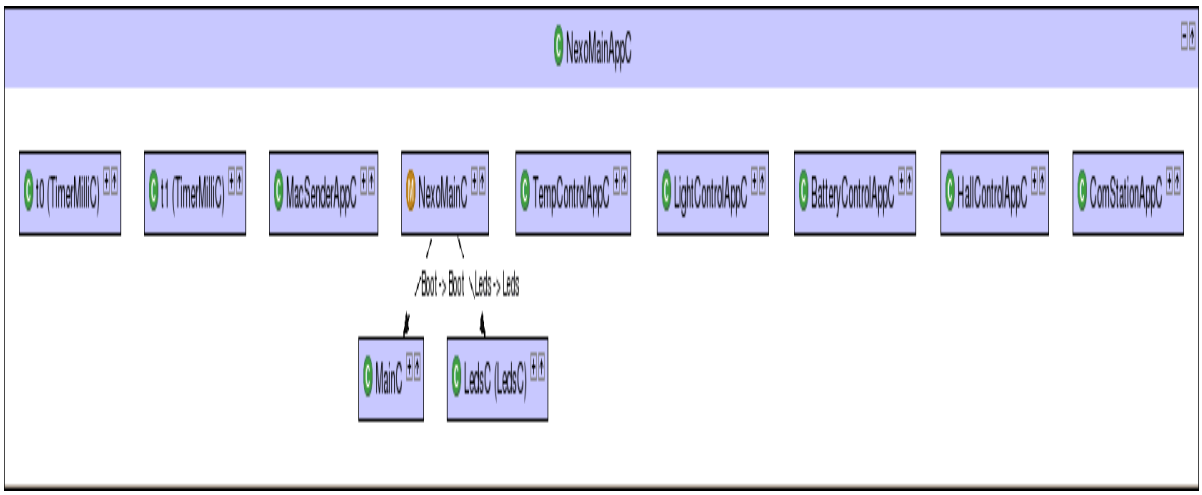

Figura 24. Componentes de NexoMain

4.1.2. SensorBaseAppC

La aplicación SensorBase nos muestra, figura 25, los distintos componentes de los que hace uso, especialmente el ADCReader para acceder a los sensores, para su lectura y el IO para activarlos previamente. También me informa que tengo el ActiveMessage declarado aunque sin uso, se trata de resto de código de realización de pruebas.

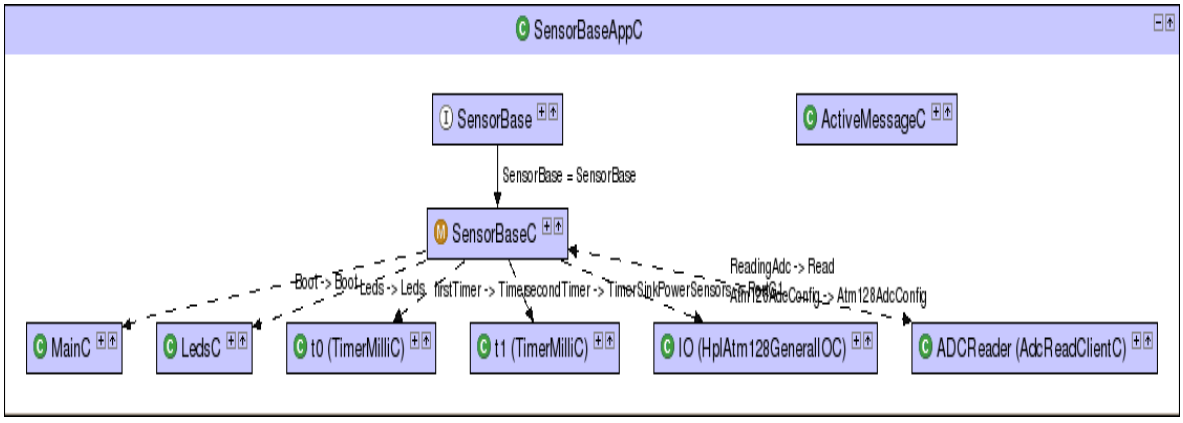

Figura 25. Componentes de SensorBase

4.1.3. BatteryControlAppC

La aplicación de la figura 26 representa la composición de BatteryControl, siendo de gran sencillez, proveemos una interfaz propia, dos temporizadores uno de disparo único y otro de alcance periódico, enlazamos con SensorBase para lectura de nivel de batería mediante su interfaz.

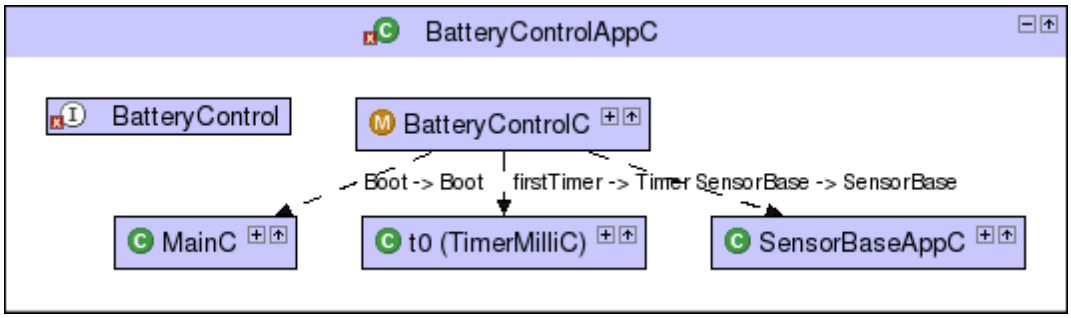

Figura 26 Componentes de BatteryControl

4.1.4. TempControlAppC

La aplicación de la figura 27 representa la composición de TempControl, proveemos una interfaz propia y usamos la de SensorBase y la de HallControl, dos temporizadores uno de disparo único y otro de alcance periódico, enlazamos con ComStation para acceder a las consignas de temperatura para preparación de

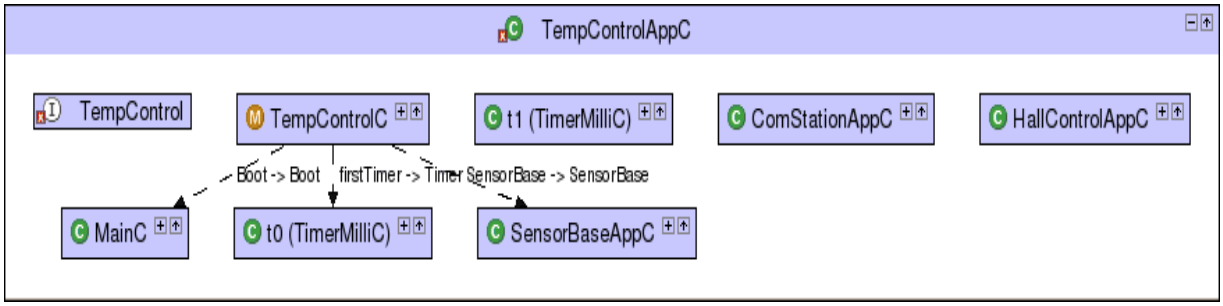

Figura 27 Componentes de TempControl

4.1.5. LightControlAppC

Al igual que en los casos anteriores tiene las mismas particularidades en cuanto a disposición de interfaces de uso o proveimiento en la siguiente figura 28 se deduce con solvencia

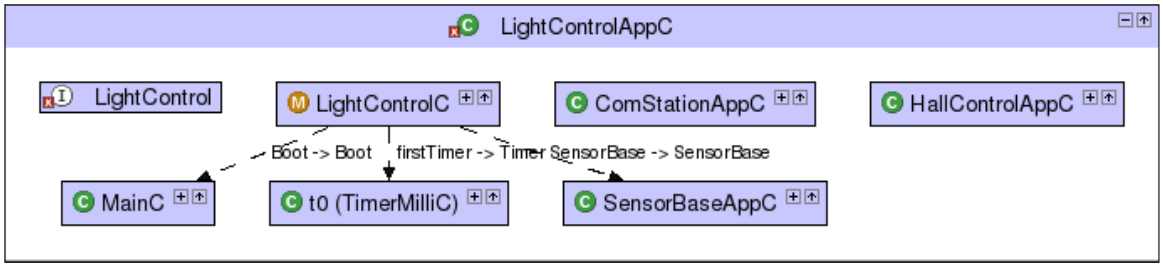

Figura 28 Componentes de LightControl

4.1.6. HallControlAppC

HallControl configura primero el Pin como entrada y será una interrupción en el nivel alto, se resetea y se enciende el sensor, en cuanto se realice la detección se ejecutará un evento el cual actúa sobre el componente LedsC, esto lo aprovecho yo para, sabiendo el estado del Led rojo asociarlo a una variable de estado, cuando haga la presentación are demostración visual. Ver figura 29.

Con esto provoco el paro de la climatización como se muestra en la figura 30 de la monitorización del Sistema.

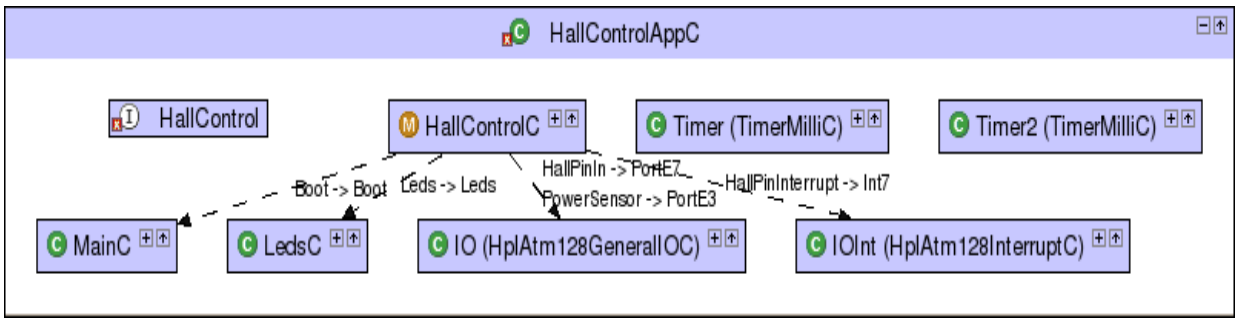

Figura 29 Componentes de HallControl

```
BATTERY: 1417   EFECTO HALL ENABLED     NIVEL DE BATERIA: OKEY NIVEL
********************************************************************************
ID MOTE: 1      COUNTER: 113

TEMPERATURE: 24,9       PID : 0,0%      AIR CONDITIONING MODE: OFF
```

Figura 30 Interfaz de usuario en caso de ENABLED efecto Hall

4.1.7. MacSenderAppC

En el caso de esta aplicación la transmisión la realizamos directamente accediendo a la Radio, no utilizamos en ningún caso ComStation, ya que esta aplicaciones para futuras ampliaciones del Sistema. Preparamos la interrupción en este caso por botón de usuario, y cuando es activado como tenemos acceso al modulo LocalleeeEui64, podemos cargar la dirección Mac i enviarla. Tengo también activado el componente de comunicación Serial aunque se trata de resto de codificación de pruebas, porque no se utiliza. La figura 31 nos muestra gráficamente toda la composición

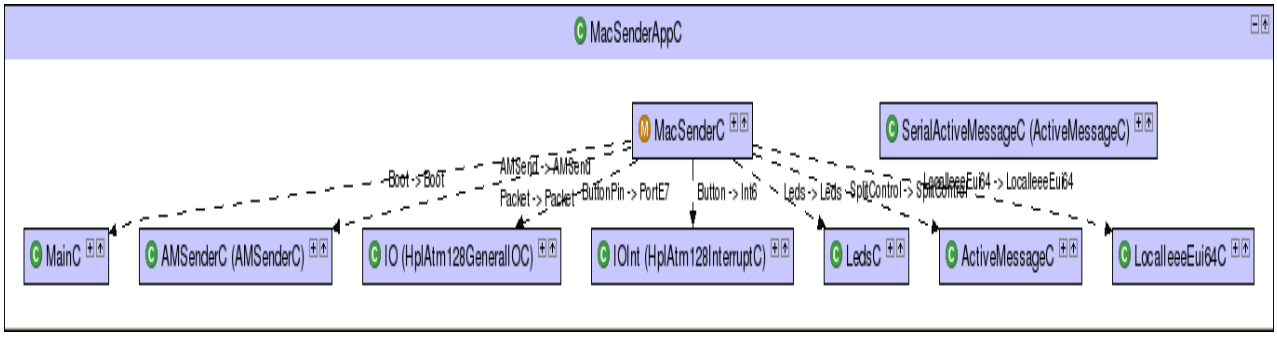

Figura 31 Componentes de MacSender

4.1.8. ComStationAppC

El componente ConStation accede a todas las interfaces y provee la de Set control, para proporcionar las consignas de funcionamiento al resto de componentes. Tenemos acceso a la radio tanto para recibir mensajes como para enviarlos. disponemos de dos temporizadores, Lesds y componente de arranque. se pueden observar sus módulos e interfaces en la figura 32.

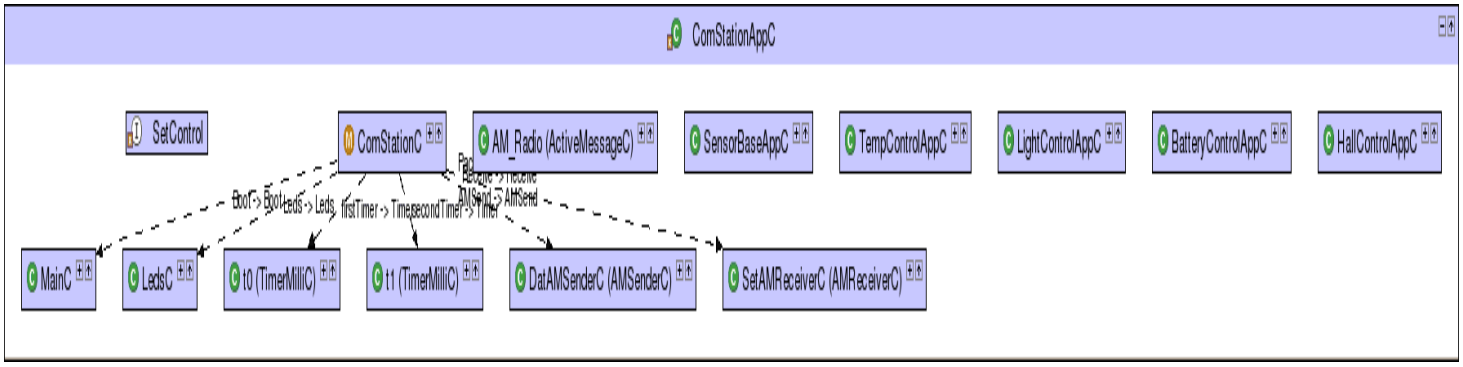

Figura 32 Componentes de CommStation

Capítulo 5

ANALISIS DE LA VIABILIDAD TECNICA Y VALORACION ECONÓMICA DEL SISTEMA

5.1. ANALISIS DE LA VIABILIDAD TECNICA

Para el análisis de la viabilidad técnica del Sistema, se establece un punto de partida por el cual considero que el elemento PC, solo se debe utilizar como comprobación de funcionamiento del Sistema. Una vez efectuada se debería situar la Mota Scutum (pasarela) sobre autómata siendo este el que enlace con la unidad de Climatización, ya que un autómata nos va permitir mayores posibilidades sobre todo para el control de los elementos de la maquina frigorífica. También necesitaríamos que la unidad de climatización dispusiese de regulación de capacidad o de regulación de velocidad, hoy en día no es un problema, ya que la mayoría de unidades llevan dispositivos Inverter de variación de velocidad. El autómata también controlaría la iluminación enviando la señal 0 a 10 voltios requerida a las reactancias.

La implementación del Sistema sobre control lumínico es mas fácil que sobre climatización al no depender de tantas variables como en circuito frigorífico, como puede ser sistema de lubricación

compresor, control presión frigorífica del circuito, elementos eléctricos, axial como presión diferencial de de aire .

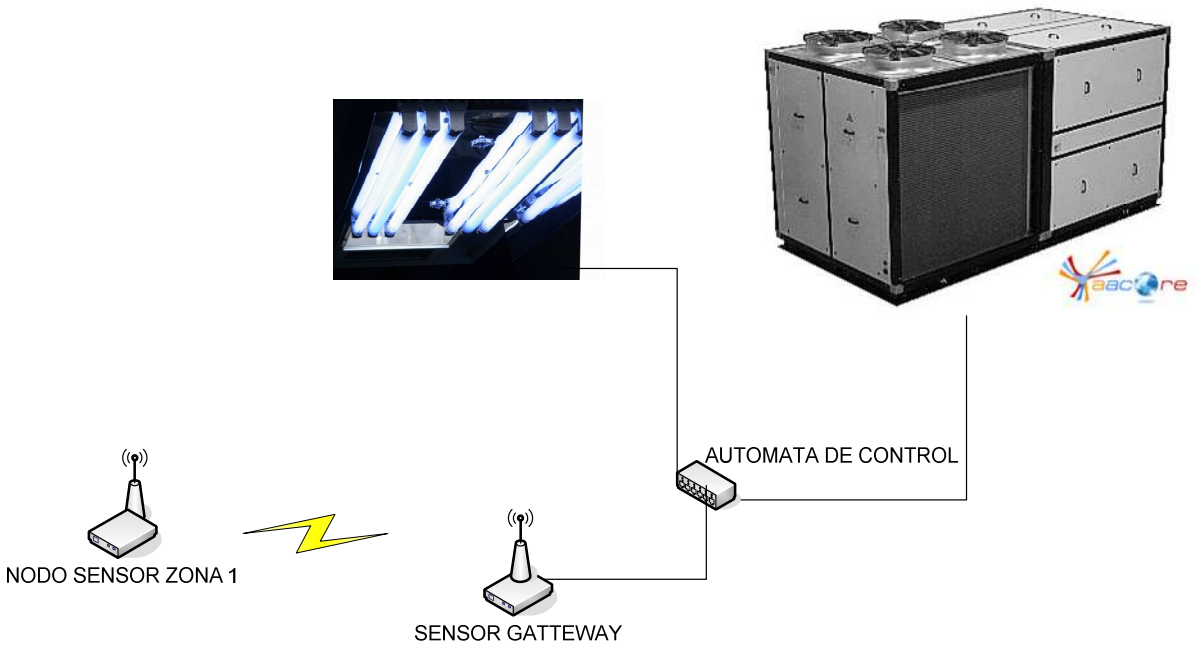

Figura 33 Diseño de Instalación de Campo

En comparativa con los sistemas mas habituales en funcionamiento actualmente en el aspecto de Climatización y control de Iluminación tenemos varias ventajas:

- No necesitamos Bus de Campo para comunicación entre sensores. Eliminamos cableado , tanto de señal, como de corriente con el consiguiente ahorro económico.
- El control que estableceremos se basa en muchos puntos de sensado, actuando sobre pequeñas parcelas autónomas eso representará un ventaja en cuanto a control de las condiciones ambientales, siempre de mayor calidad que un control general.
- El control por PID en grandes superficies proporciona un enorme ahorro energético, ya que adaptamos de forma equilibrada la potencia entregada a las necesidades climáticas.
- Sacamos máximo aprovechamiento a la luz natural.

- Podríamos sacar mas funciones a los sensores, detector de incendios, seguridad, regulación de nivel de humedad, lo cual significaría el disponer de una sola instalación de control para todo el local comercial.

5.2. VALORACION ECONOMICA

No se realiza valoración económica mientras no se realice una prueba de adaptación entre autómata, mota sensora y equipo de climatización, esto requiere de un estudio complementario que no entra dentro del alcance de este proyecto que solo sienta las bases para una futura implementación

Capítulo 6

CONCLUSIONES

6.1. CONCLUSIONES

Estoy satisfecho con la aplicación resultado, posiblemente con mas tiempo podría aumentarle su grado de precisión en cuanto al control de la Zona sensada. No estoy satisfecho con no haber podido entregar una interfaz de usuario mas profesional, pero mis limitados conocimientos de Java y la escasez de tiempo me han hecho retroceder el camino que en un principio seguía.

En cuanto a mi primer contacto con el mundo de los sensores inalámbricos tengo afirmar que ha sido realmente apasionante, y con muchas ganas de seguir profundizando en esta tecnología en el futuro.

6.2. PROPUESTA DE MEJORAS

Se debe establecer un programa de pruebas minucioso y detallado ya que el periodo de pruebas ha sido insuficiente para someter al sistema a todo tipo de situaciones extremas, simplemente se

han comprobado que las funcionalidades básicas estuviera realizadas, sin analizar si sus valores actúan acorde a la realidad, además se debe mejorar en :

- En los primeros mensajes que recibo de la mota el calculo PID me aparecen en centenas, solo son los primeros envíos después se corrige y marca lo habitual en tantos por cien. No disponía de suficiente tiempo para sanearlo.

- Realizar una comprobación de saneamiento para limpiar el código profundamente para eliminar líneas bloqueadas que se utilizaron en algún momento o con las que se hicieron pruebas, así como añadir explicaciones mayor explicaciones, ya que se produjeron variaciones que no están suficientemente explicadas.

- Regular los tiempos de las comunicaciones, no es necesario recibir ni enviar tantos datos continuamente, las variaciones en los sensores no son tan importantes por lo tanto recibimos y enviamos información redundante, con lo cual lo único que hacemos es consumir batería.

- He comprobado una gran variedad de combinaciones, tanto de luz como de temperaturas analizando los cambios que se van produciendo, pero realmente no se si he cubierto todas las posibilidades. Seguramente necesitaría un análisis mas profundo.

6.3. AUTOEVALUACION

Dentro del apartado de autoevaluación hay que destacar varias cosas, sin caer en la autojustificación:

1. Se trata de una tecnología nueva para los que hemos estudiado Ingeniería de Telecomunicaciones, muy interesante pero requiere primero estudio y comprobación de los programas de prueba y después trabajo en nuestro propia aplicación, lo que provoca

mucho tiempo dedicado y siempre las dudas surgen al final cuando aparece el problema y con el tiempo apremiando.

2. Dificultad de depuración, lo que en otro tipo de lenguajes suele ser mas o menos fácil aquí suele ser mas complicado y con el inconveniente que si realizas demasiados cálculos y ejecuciones puedes perder precisión.

3. Sin duda las Motas tienen una gran eficacia cuando realizan funciones básicas, por lo menos las que nosotros tenemos, sin embargo cuando se someten a muchos cálculos, entrañan enormes riesgos en la depuración con el sistema prueba-error y localización de Bugs.

4. En este proyecto es muy importante dominar Java, cuanto mas mejor, creo que en nuestra especialidad seria importante tenerlo en cuenta para enfrentarte a un proyecto con Tinyos. Este aspecto lo tengo que mejorar

5. He visto también que en las motas se pierde mucha precisión en los grandes datos con las operaciones y esto me ha creado algún problema obligándome ha trabajar con unidades counts para los controles en los componentes. Pero me interesaba ver la capacidad de caculo de la Mota y su eficacia, cosa que me ha sorprendido muy gratamente.

6. La capacidad de comunicación de la Mota he podido comprobar que es fantástica desde posiciones dentro de la nevera o congelador en largas distancias y demás así como el poco consumo de la misma a pesar de estar en comunicación intensiva para poder ver resultado y respuesta.
7. Me ha resultado muy sorprendente que a pesar de la cantidad de tiempo que los sensores han estado encendidos el bajo consumo de batería, en todo el tiempo de proyecto no he consumido ni la mitad de la carga de las baterías.
8. La robustez de las motas ante tantas programaciones y reprogramaciones es un dato a destacar.

Por ultimo decir que la dificultad mayor es el tiempo, que es la causa principal de que no se haya podido realizar un proyecto de mayor nivel, no ha sido por ganas ni creo que por compromiso sino solo por tiempo.

Capítulo 7

GLOSARIO

7.1. PAGINAS WEB PARA CONSULTA

Las fuentes de información Web visitadas para este proyecto han sido:

- http://www.atmel.com/dyn/resources/prod_documents/doc8226.pdf
- http://www.sparkfun.com/datasheets/IC/cp2102.pdf
- http://ww1.microchip.com/downloads/en/DeviceDoc/21942e.pdf
- http://www.advancedphotonix.com/ap_products/pdfs/PDV-P9003-1.pdf
- http://www.rohm.com/products/databook/sensor/pdf/bu52001gul-e.pdf
- http://www.tinyos.net/
- cs.acadiau.ca/~shussain/wsn/apps/tos1/
- www. **eclipse** .org / descargas
- http://cv.uoc.edu/app/mediawiki14/wiki/P%C3%A0gina_principal

7.1. BIBLIOGRAFIA

- Estudio y Análisis de las Características de TinyOS. José Ulloa Suárez. Sociedad Agrícola Ojos Buenos Ltda.
- Libro TinyOS Programming. La biblia de los programadores de sensores, escrito por Philip Levis y David Gay
- UNIVERSITAT JAUME I DE CASTELLÓ. Escuela Superior de Tecnología y Ciencias Experimentales. IS31 – Proyectos Informáticos de Sistemas. Memoria técnica del proyecto. Procesamiento de datos en una red de sensores inalámbrica
- INTRODUCCIÓN A LA PROGRAMACIÓN EN TINYOS: NESC.
- TUTORIIAL DE TIINYOS TRABAJO II Asignatura: Redes Móviles 5º Ingeniería Informática. Autor: Profesor: José María Alcaraz Calero Pedro M. Ruiz

Capítulo 8

ANEXOS

8.1. INSTRUCCIONES DE EJECUCION

8.1.1 DESCRIPCIÓN DE COMPILACIÓN Y CARGA

En las APSS de Tinyos-2.x tengo tres carpetas Casiopea_0, Scutum y Hidrus a la cual le paso los archivos una vez modificados y trabajados con mi editor ECLIPSE.

- **Mota Casiopea**

Para compilar hacemos:

1. cd /opt/tinyos-2.x/apps/Casiopea_0

2. make cou24

Instalar la aplicación en la Mota:

1. meshprog –t/dev/ttyUSB0 –f./build/cou24/main.srec

- **Mota Scutum**

Para compilar hacemos:

3. cd /opt/tinyos-2.x/apps/BaseStation

4. make cou24

Instalar la aplicación en la Mota:

2. meshprog –t/dev/ttyUSB0 –f./build/cou24/main.srec

- **Interfaz Hidrus**

Para compilar hacemos:

1. cd /opt/tinyos-2.x/apps/Casiopea_0

2. javac MainHidrus.java

8.1.2 EJECUCION DEL SISTEMA

Pasos para funcionamiento del sistema una vez compiladas y cargadas las aplicaciones

- Poner en marcha la Mota Casiopea_0, para ello solo hay que introducirle las pilas. En cuestión de segundos los Leds van informando del comienzo del sensado, calculo y envío de datos

- Situar la Mota Scutum en el puerto USB y ejecutar el SerialForwarder para que la aplicación de usuario enlace con la Mota.

- Para lanzar el SerialForwarder:

    1. java net.tinyos.sf.SerialForwarder –comm serial@/dev/ttyUSB:19200

- Para lanzar la aplicación Hidrus hacemos:

    1. java MainHidrus

    2. Cuando se inicia pide por consola las consignas de trabajo, se introduce valor y tecla intro, a partir de ese momento ya se ven los datos comunicados.

    3. Para volver a cargar nuevas consignas se para la ejecución y se vuelve a iniciar, yo lo he hecho control+c para y java MainHidrus iniciar, muy rustico pero así podía analizar los datos.

Capítulo 9

CONTRAPORTADA

INGENIERIA DE TELECOMUNICACIONES

ESPECIALIDAD: TELEMÁTICA

TRABAJO FINAL DE CARRERA

SISTEMA DE AHORRO ENERGÉTICO EN CLIMATIZACION E ILUMINACION, PARA GRANDES ESPACIOS COMERCIALES Y DE OCIO, MEDIANTE RED DE SENSORES INALÁMBRICOS