Índice

1. Introducción 1

1.1 Presentación del problema 1

1.2 Estructura de la memoria 2

2. Estado del arte 3

2.1 Radio Cognitiva 3

2.2 Redes de sensores inalámbricas 3

2.3 Simuladores de redes de sensores 3

3. Fases del proyecto 3

3.1 Primera fase 3

3.1.1 Adaptación del entorno de MiWi 4

3.1.2 Pruebas con el entorno de MiWI 5

3.2 Segunda Fase 7

3.3 Tercera Fase 7

4. Adaptación Hardware 8

4.1 Requisitos 9

4.2 Diseño PCB 10

4.3 Adaptación 11

4.3.1 Acoplamiento Físico 11

4.3.2 Configuración Hardware 13

4.3.3. Configuración Software 19

Bibliography 20

1. Introducción

1.1 Presentación del entorno

En el mundo actual, en el sector de la electrónica y fundamentalmente en la electrónica de consumo el número de dispositivos que poseen interfaces radio está creciendo a un ritmo muy alto. Esto es debido a que poder dotar de conectividad a dispositivos que no disponían de ella, en la mayoría de los casos, proporciona nuevas características a los mismos y normalmente hace que su campo de aplicación se expanda y mejore.

Existen, además, gran cantidad de dispositivos electrónicos que poseen diferentes interfaces radio. La mayoría de estos interfaces tiene propósitos bien definidos y normalmente restringidos a determinados usos por lo que no presentan estrategias colaborativas entre ellos para mejorar su funcionamiento. Este hecho representa una limitación y un gran inconveniente para el correcto aprovechamiento del entorno de la comunicación.

1.2 Presentación del problema

Un ejemplo ilustrativo es que en la actualidad las personas se relacionan en su día a día con dispositivos móviles como pueden ser las tabletas electrónicas (*tablets*) y los teléfonos móviles. Éstos disponen de interfaces inalámbricos en diferentes bandas o incluso algunos que comparten la misma banda. Es algo común hacer uso de un enlace con un dispositivo de manos libres vía interfaz Bluetooth y que a su vez se esté requiriendo descarga de datos mediante la interfaz WiFi. Ésta es una situación que se puede dar, por ejemplo, en algo tan habitual como una llamada o videollamada con voz sobre IP. Tanto Bluetooth como WiFi operan en la banda de 2.4 GHz por tanto es susceptible de que ambos interfaces se interfieran.

Tanto es así que, de hecho, Bluetooth utiliza un esquema de comunicación basado en que la señal portadora cambia de frecuencia central 1600 veces por segundo y condiciona los canales de los saltos en parte a criterios relacionados con el nivel de señal de los mismos y de esa forma consigue evitar interferencias con otros dispositivos que trabajen en esa misma frecuencia.

Puede parecer, a priori, que es una forma elegante de solucionar un problema de interferencia con otros dispositivos, sin embargo, observando este hecho con una perspectiva más amplia, este esquema de comunicación representa una solución a un problema que se debería haber evitado en su raíz, por ejemplo utilizando estrategias colaborativas entre los diferentes interfaces implicados.

Además, la gran cantidad de saltos por segundo suponen un consumo de potencia extra en algo que no tiene que ver directamente con la información que se desea transmitir, presentando así un uso ineficiente de la energía.

El ejemplo que hemos ilustrado ha tenido como escenario un dispositivo móvil, es decir, esta ausencia de colaboración ya existe en un dispositivo que en principio debería estar diseñado para presentar un funcionamiento óptimo.

Si se escala el entorno de estudio y se observa el funcionamiento de redes inalámbricas a nivel, por ejemplo, de un bloque de edificios, el escenario es más desolador.

En sociedades muy desarrolladas es común que, en bloques de viviendas, exista un enrutador WiFi por vivienda, varios dispositivos con ese tipo de interfaz por enrutador (móvil, *tablets*, ordenadores, etc.), además varios dispositivos con comunicación Bluetooth (manos libres, equipos de sonido inalámbricos, etc.) también por vivienda, e incluso la tendencia es ir incorporando a cada vez más aparatos domésticos la conectividad inalámbrica. Con todo esto se consigue que el escenario en el que hay que tratar las interferencias es y será cada vez más complejo por lo que se puede llegar al punto en que las soluciones para rodear el problema como la explicada en caso del Bluetooth dejen de ser viables y funcionales.

1.3 Causas del problema

El problema, en primer lugar, reside en que el espectro radioeléctrico es un recurso limitado, con el paso de los años se han ido reservando bandas a diferentes propósitos como son la transmisión de señal de TV, radio, bandas de frecuencia reservadas a uso militar, geo-posicionamiento, comunicaciones marítimas y un largo etcétera, propiciando de esta forma una situación en la que la mayor parte del espectro está reservada a determinados usos y agentes. Precisamente por la gran cantidad de agentes afectados por este reparto y por la tecnología y coste asociado a la misma se ha llegado a una situación en la que no se puede retroceder y que acarrea una enorme segmentación del espectro radioeléctrico.

A fin de ilustrar la magnitud de esta segmentación se presenta la figura 1, donde se muestra el reparto del espectro en E.E.U.U. en el año 2003. La figura muestra el espectro en la banda desde los 3 KHz hasta los 300 GHz y aunque resulta imposible, por el tamaño de la imagen, leer el propósito para el que está reservada cada banda, muestra a grandes rasgos el grado de saturación y segmentación existente en aquel momento.

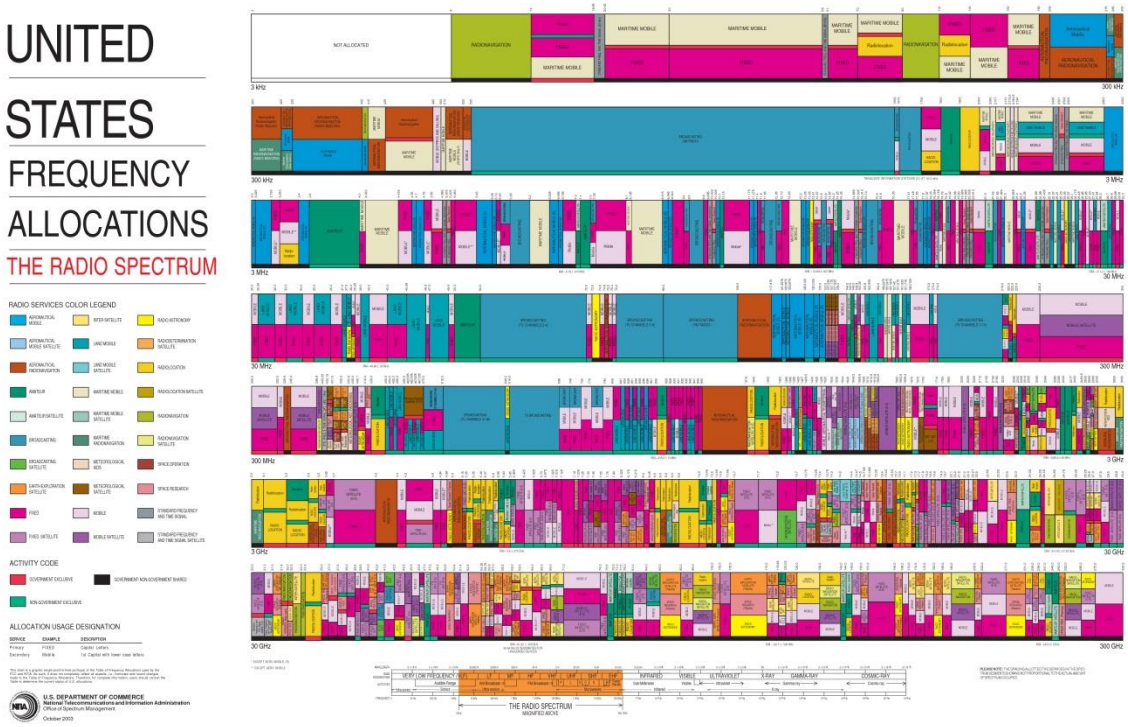

Fig.1. Reparto del espectro radioeléctrico en E.E.U.U. [1]

Esta situación no ha mejorado ni es diferente en Europa. Se puede encontrar información más detallada para la región europea en la tabla del reparto del espectro radioeléctrico [2].

Si ahora se tienen en cuenta los rangos de frecuencia libre y en los que a día de hoy tiene sentido por coste, consumo, alcance y tasa binaria, transmitir de forma inalámbrica, resulta que estaremos en la banda de los 300 MHz a los 7 GHz. En la figura 2 se resaltan en verde las bandas de uso libre en ese rango para el reparto mostrado anteriormente.

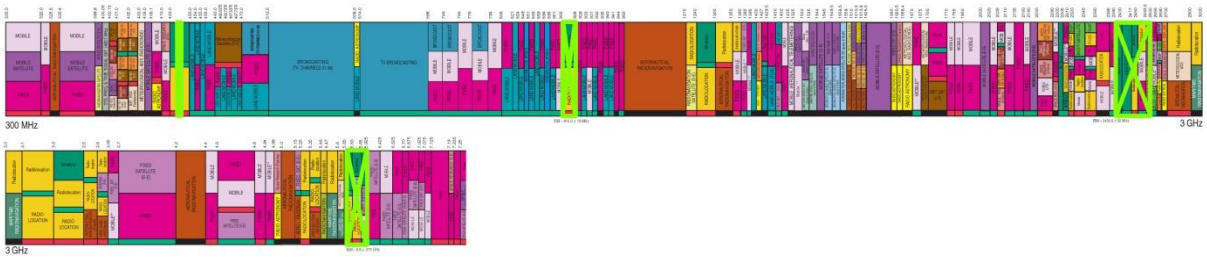

Fig. 2. Bandas ISM resaltadas en el rango de los 300 MHz a los 8 GHz.

Se observa que la figura 2 corresponde al detalle de las filas 5 y 6 de la figura 1 y se puede ver en ella, que las bandas de uso libre, marcadas en verde, también conocidas como bandas ISM (*Industrial*, *Science and Medical*) por sus posibles aplicaciones, representan una porción muy pequeña del espectro.

En consecuencia se presenta una situación en la que gran cantidad de dispositivos intentan transmitir en un determinado y pequeño rango de frecuencias, bandas ISM, y por tanto terminan en una lucha en la que dichos dispositivos intentan hacerse escuchar aumentando el nivel de potencia de sus transmisiones o mediante el uso de ingeniosas técnicas como se ha mostrado en el caso del Bluetooth. Pero en cualquiera de los casos muestran técnicas que a pesar de eficaces no son eficientes energéticamente hablando.

1.4 Presentación del campo de trabajo

Tanto en entornos industriales, como en el hogar, entornos militares, naturales y casi cualquier tipo de entorno que se pueda imaginar, la tendencia que tiene el ser humano en su búsqueda para entenderlo es a caracterizarlo.

Mediante la caracterización se puede comprender lo que ocurre en el medio, predecir lo que puede ocurrir, crear modelos matemáticos que representen las tendencias y buscar actuaciones para prevenir consecuencias indeseadas.

Surgen entonces conceptos y desarrollos como el de las redes inalámbricas de sensores (WSN), que es una idea suficientemente generalista como para ser adaptada a gran cantidad de escenarios diferentes.

Las WSNs (Wireless Sensor Network) están formadas por varias unidades autónomas que se denominan nodos y que constan a su vez de la electrónica necesaria para medir diferentes parámetros del entorno como pueda ser humedad, temperatura, luz, viento, etc. Se dice que estos nodos son autónomos porque solo por ellos mismos constan de cierta funcionalidad, que, como se deduce de los sensores que los componen, consistirá en poder realizar medidas del entorno.

Un ejemplo de aplicación y de gran utilidad para las WSN es la predicción de incendios forestales. Mediante el despliegue de una red inalámbrica de sensores se pueden predecir situaciones de especial riesgo como puede ser una alta temperatura del ambiente, humedad relativa baja, viento y parámetros de ese estilo. De esa forma se podría actuar antes de que ocurriera el posible incendio humedeciendo el ambiente mediante el arrojo de agua desde aeronaves.

Incluso si el incendio ya se ha iniciado, las redes inalámbricas de sensores pueden ser útiles para su pronta detección o incluso para predecir y modelar el comportamiento del fuego a nivel de propagación, intensidad, etc.

Otro posible campo de aplicación muy de moda de las WSNs es la domótica. Es una tendencia cada vez más en auge que diferentes acciones en el hogar se realicen de forma automatizada. Mediante redes de sensores en una casa se puede controlar la iluminación y adecuarla a las condiciones y a la presencia, controlar la temperatura, optimizar consumos energéticos y en general tomar decisiones predictivas, basadas en la experiencia adquirida por la red, con el objetivo de facilitar y abstraer al usuario de realizar tareas para que su lugar de residencia se adapte a sus necesidades.

En la figura 3 se puede ver un ejemplo de un posible elemento para una red sensores en el hogar como es un termostato inteligente.

Fig. 3. Termostato inteligente fabricado por Nest [3].

Este último campo de aplicación expuesto para WSNs es muy conveniente para ser enlazado con lo expuesto en el apartado anterior sobre la utilización de las comunicaciones en las bandas ISM, que es precisamente las que deben utilizar los nodos que forman las redes de sensores.

Se puede concluir este apartado, que presenta el entorno en el que se centra este proyecto, diciendo que si se busca el desarrollo y despliegue sostenible de redes inalámbricas de sensores en cualquier ámbito en el rango de frecuencias libres, en realidad se busca una red que además de esas características tenga una comunicación inalámbrica inteligente y adaptativa a su entorno. Dicho de otra forma, que la comunicación entre los nodos también esté sujeta a un proceso cognitivo.

1.5 Búsqueda de soluciones

Una vez se ha presentado el entorno y anteriormente los problemas que conciernen a las comunicaciones inalámbricas se pueden concentrar los esfuerzos en la búsqueda de soluciones más concretas.

Como ya se ha adelantado en el apartado anterior, dicha búsqueda debe centrarse en la comprensión del entorno y posteriormente en la búsqueda de la colaboración entre los diferentes agentes, que en el caso que concierne, como se expondrá más adelante, serán únicamente nodos.

Los problemas mencionados no son nuevos y ya se empezó a buscar soluciones y desarrollos que los mitigaran a finales del siglo XX. Dos de las líneas de investigación más importantes en las que se comenzó a trabajar fueron la radio cognitiva y más adelante la red cognitiva.

El concepto de radio cognitiva aparece por primera vez en [4] y posteriormente lo define Joseph Mitola como “El punto en el que los PDAs y las redes asociadas son lo suficientemente inteligentes, computacionalmente hablando, sobre los recursos de red y las comunicaciones entre dispositivos para tomar las necesidades en la comunicación entre usuarios como una función del contexto de uso y adaptar así los recursos de red y servicios inalámbricos más apropiados a dichas necesidades” en [5].

El concepto de red cognitiva aparece por primera vez en una tesis también de Joseph Mitola, pero no se define hasta más tarde en otros documentos. Una de las primeras definiciones de red cognitiva la proporciona Thomas en [6]. Define red cognitiva como “una red que lleva a cabo un proceso cognitivo que le permite percibir las condiciones actuales de la red, planificar, decidir, actuar de acuerdo a ellas, aprender de las consecuencias de sus acciones y todo de acuerdo a unos objetivos punto a punto.”

Además apunta que “el ciclo cognitivo, sensa el entorno, planifica acciones de acuerdo a las entradas de sus sensores y a las políticas de la red, decide qué escenario encaja mejor en sus propósitos punto a punto utilizando un motor de razonamiento y finalmente actúa en el escenario elegido. El sistema aprende del pasado (situaciones, planificaciones, decisiones, acciones) y utiliza su conocimiento para mejorar las decisiones en el futuro.”

La radio y la red cognitiva representan el punto de partida para combatir algunos de los problemas que hemos nombrado que se encuentran en las WSNs. Además presentan las bases para el desarrollo de las denominadas CWSNs (Cognitive Wireless Sensor Network) que serán las WSNs ya explicadas con capacidades cognitivas añadidas.

Es fundamentalmente el concepto de ciclo cognitivo el que permite realizar otra definición que será utilizada habitualmente a lo largo de esta memoria y que es el

concepto de estrategia cognitiva. Se ha de entender como estrategia cognitiva la realización del ciclo cognitivo en búsqueda de unos objetivos fijados con anterioridad. Esos objetivos deben determinar el fin último por el cual se ejecuta el ciclo cognitivo y que por tanto justifican su existencia. Por citar algunos ejemplos, dichas metas podrían ser: reducción de consumo, aumento de seguridad, aumento de tasa binaria, etc.

1.6 Objetivo del Proyecto

Una de las líneas de investigación del laboratorio B105 de la ETSI de Telecomunicación de la Universidad Politécnica de Madrid son precisamente las redes inalámbricas cognitivas de sensores. Además son varios los desarrollos que se han hecho involucrando a redes de sensores o simplemente redes de nodos con comunicación inalámbrica.

Con el fin de intentar mejorar y añadir robustez a las comunicaciones de ese tipo de redes se les pretende añadir capacidades cognitivas a los nodos. Estas capacidades deberán coexistir con la aplicación original de la red. Se trata por tanto, como se ha dicho, de mejorar la comunicación en la red con el fin de terminar mejorando la aplicación llevada a cabo, ya sea en términos de consumo, seguridad u otro ámbito dependiente de la estrategia o estrategias cognitivas que se implementen.

Precisamente para realizar la implementación de dichas estrategias es necesario disponer de un entorno de desarrollo de las mismas. Además hay que tener en cuenta que las estrategias pueden ser dinámicas, es decir, cambiantes en función a unos intereses que pueden variar en el tiempo.

Con la disposición de proporcionar ese entorno para implementar estrategias cognitivas nace el objetivo final de este proyecto que trata de construir una arquitectura software que permita la inclusión de dichas estrategias en un nodo de una red de sensores y que, en la medida de lo posible, sea transparente al diseño de la aplicación que se quiera que ejecute ese nodo. De tal forma que se consiga que las aplicaciones desarrolladas puedan funcionar tanto con estrategia cognitiva como sin ella sin que apenas haya que adaptarla para cada caso.

Para conseguir dicho objetivo se plantea una línea de trabajo compuesta por objetivos más pequeños que permitan en último término alcanzar el objetivo final. Dichos sub-objetivos serán tales como familiarizarse con un nodo ya existente en el laboratorio, reconocer sus fortalezas y debilidades para caracterizar una posible evolución de dicho nodo, realizar pruebas con las nuevas características que se hayan especificado y finalmente centrar el trabajo en el diseño, implementación y test de una arquitectura software para realizar estrategias cognitivas.

1.7 Estructura de la memoria

Ésta memoria se estructura en nueve apartados diferenciados. En este primer apartado se ha realizado una breve introducción al entorno y a algunos problemas de las redes inalámbricas de sensores y al camino a seguir para intentar solucionarlo. En el segundo apartado se describirá el estado del arte de las redes cognitivas inalámbricas de sensores. A continuación, en el apartado, tercero se presentan las diferentes fases en las que se ha desarrollado este proyecto. En el cuarto se explica una adaptación *hardware* que se ha realizado con el fin de adaptar una plataforma existente a unas nuevas necesidades. En el quinto se presentará y se justificará el diseño del *software* que se ha desarrollado. En el apartado sexto se describe la implementación del *software* realizada especificando también su estructura. Un séptimo apartado concreta lo anteriormente explicado en un ejemplo demostrador de la estrategia con el fin de presentar el funcionamiento del sistema. El octavo redacta las conclusiones extraídas de la realización de este proyecto. Y por último el apartado noveno presenta unas posibles líneas futuras.

2. Estado del arte

Para repasar el estado del arte de las redes cognitivas inalámbricas de sensores hay que abordar el estudio desde diferentes frentes por la multitud de factores que están involucrados en dicha temática.

Por esa razón en este apartado de la memoria estructuramos el estado del arte en varios apartados según aborden el estado de desarrollo de:

- Nodos para redes de sensores inalámbricas, cognitivas o no.
- Radio cognitiva y aplicable a nodos de bajo consumo.
- Simuladores de redes de sensores.
- Arquitecturas para redes de sensores.

2.1 Nodos para CWSNs y WSNs

En el año 2001 ya se realizó una demostración en la universidad de Berkeley, donde una red a gran escala de 800 nodos autónomos se autodescubría y configuraba. Dicha red de sensores estaba formada por unos nodos de bajo consumo y una forma circular con diámetro de 2.5 cm aproximadamente. Los nodos tenían capacidad para medir iluminación, temperatura, nivel de batería y nivel de señal de radio. La información se recogía en un ordenador y mostraba en tiempo real las condiciones de iluminación de la sala. Además la secuencia de descubrimiento y formación de la red duraba apenas una fracción de segundo. El proyecto, financiado por DARPA (*Defense Advanced Research Project Agency*). Esta demostración mostraba el sorprendente potencial en cuanto a consumo,

complejidad de la topología de red, auto-configurabilidad y capacidad de monitorización de una red de sensores de bajo consumo y coste. Descripción más detallada sobre este proyecto se puede ver en la web del proyecto [7].

Otro proyecto también de la universidad de Berkeley es Epic. Epic se define como una plataforma abierta que se puede adecuar al diseño que se establezca por la aplicación requerida. Consiste fundamentalmente de muchos módulos que se pueden interconectar para cumplir el propósito requerido. Entre esos módulos se pueden destacar el modulo central o *core* donde se encuentra la unidad de procesamiento, módulo de almacenamiento, conectividad USB, módulos de desarrollo para poder programar el micro-controlador de la unidad central y en general una infinidad de módulos para dar capacidad de monitorizar parámetros ambientales, darle conectividad inalámbrica, cámara, etc. Se pueden ver todos los módulos en la web [8].

Un desarrollo muy versátil de un nodo para una red de sensores es el Waspmote. Desarrollado por una empresa llamada Libellium se trata de una plataforma de desarrollo formada por una placa principla a la que se le pueden adaptar diferentes módulos para dotar al nodo de diferentes funcionalidades. Dichos módulos pueden ser tales como módulo GPS, GPRS y otras interfaces de comunicación y en general sub-plataformas para añadir capacidad de medir parámetros tales como gases del tipo CO, CO2, CH4, etc. o eventos de luminosidad, presión, PIR, etc. El nodo además presenta la posibilidad de configurarlo con dos interfaces de comunicación inalámbrico al mismo tiempo, por lo que a priori podría servir para experimentar con algunas estrategias cognitivas. El principal problema de esta plataforma es que tiene un coste alto, y aunque puede representar una opción a tener en cuenta como paso previo a un desarrollo comercial, no parece tan apropiado para un entorno de investigación universitaria.

En general se puede ver una relación de los desarrollos de nodos para redes de sensores en la lista de nodos sensores en [9]. Por citar algunos se compone la siguiente tabla.

Nombre del sensor	Microcontrolador	Transceptor	Memoria de programa+datos	Información adicional
Arago Systems WiSMote Dev	MSP430F5437	CC2520	RAM : 16 Kbytes flash : 256 Kbytes	Contiki and 6LoWPan supported
AVRraven Atmel AVR#Raven wireless kit	AtMega1284p + ATmega3290p	AT86RF230	128 Kbytes + 16 Kbytes	

COOKIES	ADUC841, MSP430	ETRX2 TELEGESIS, ZigBit 868/915	4 Kbytes + 62 Kbytes	Platform with hardware reconfigurability (Spartan 3FPGA based or Actel Igloo)
BEAN	MSP430F169	CC1000 (300-1000 MHz) with 78.6 kbit/s		YATOS Support
BTnode	Atmel ATmega 128L (8 MHz @ 8 MIPS)	Chipcon CC1000 (433-915 MHz) and Bluetooth (2.4 GHz)	64+180 K RAM	BTnut and TinyOS support
EPIC mote	Texas Instruments MSP430 microcontroller	250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver	10 KB RAM	TinyOS
Eyes	MSP430F149	TR1001		PeerOS Support
IMote 2.0	Marvell PXA271 ARM 11-400 MHz	TI CC2420 802.15.4/ZigBee radio compatible	32 MB SRAM	Microsoft .NET Micro, Linux, TinyOS Support
Iris Mote	ATmega 1281	Atmel AT86RF230 802.15.4/ZigBee radio compatible	8 KB RAM	Mote Runner, TinyOS, MoteWorks Support
MicaZ	ATMEGA 128	TI CC2420 802.15.4/ZigBee radio compatible	4 KB RAM	TinyOS, SOS, MantisOS and Nano-RK Support
SenseNode	MSP430F1611	Chipcon CC2420	10 KB RAM	GenOS and TinyOS Support
Tinynode	Texas Instruments MSP430 microcontroller	Semtech SX1211	8 KB RAM	TinyOS
Waspmote	Atmel ATmega 1281	ZigBee/802.15.4/DigiMesh/RF, 2.4 GHz/868/900 MHz	8 KB SRAM	GPRS, Bluetooth, módulo GPS, placas de sensores…
FireFly	Atmel ATmega 1281	Chipcon CC2420	8 KB RAM	Nano-RK RTOS

Tabla 1. Relación de nodos para redes de sensores.

En la actualidad no hay soluciones de nodos que integren y realicen un uso inteligente de diferentes interfaces de comunicación inalámbrica para redes de sensores. Debemos entender las palabras “uso inteligente” como la capacidad de un nodo inalámbrico para utilizar sus interfaces en función de lo que su proceso cognitivo le dicte, teniendo en cuenta el máximo número de variables posible acorde a la capacidad computacional del nodo. Sin embargo algunos de los nodos citados en el apartado si serían aptos como base para desarrollar un nodo para redes inalámbricas cognitivas de sensores, como es el caso del Waspmote, aunque hay que añadir que no está exento de limitaciones al no haber sido concebido para ese propósito, como el hecho de que sus interfaces inalámbricas estén limitadas a dos.

2.2 Radio Cognitiva

Escalando a un nivel inferior entramos en el territorio de la radio cognitiva. Otro campo de desarrollo interesante para ser estudiado es el de la radio cognitiva.

A grandes rasgos aquí se trata de tener un transceptor radio que es capaz de ajustar parámetros de transmisión y recepción tales como la modulación, protocolo, banda de frecuencia, topología, etc. al entorno.

Por tanto para realizar desarrollos de radio cognitiva es necesario disponer de radios con parámetros configurables como puede ser el caso de la SDR (*software-defined radio*). La SDR es fundamentalmente una radio donde algunos componentes que habitualmente están implementados en *hardware* como pueden ser mezcladores, amplificadores, moduladores, etc. han pasado a estar implementados y configurados mediante *software* y sistemas empotrados. Éste hecho propicia que en general se trate de radios altamente configurables aunque con costes normalmente en las centenas de euros.

Si solo consideráramos receptores hay desarrollos abiertos de SDR muy interesantes como es el caso del RTL-SDR que se trata de un dispositivo USB concebido para la recepción de señal digital de televisión pero que se ha reutilizado como receptor SDR. Resulta que muchos modelos de este dispositivo basado en el chip Realtek RTL2832U tienen un sintonizador operable en un rango de frecuencias bastante amplio puede ir desde las pocas decenas de mega Hertzios hasta casi los 2000 MHz. Además el coste de estos receptores es muy bajo rondando los 17 € probablemente debido a la alta penetración de mercado de los receptores de DVB-T.

Sin embargo en SDR con capacidad tanto de transmisión y recepción el precio aumenta y se sitúa entorno a los 500 €. Tal es el caso, por ejemplo, de BladeRF, una plataforma de desarrollo para SDR con una frecuencia de trabajo de entre los 300 MHz y los 3.8 GHz, con una alta configurabilidad. Su principal problema es el precio que se situa dependiendo de la versión entre los 420 $ y los 650 $.

Pasando por alto desarrollos que por su coste, tamaño y consumo no resultan interesantes para ser adoptados en nodos, a nivel de circuitos integrados existen desarrollos como el de IMEC, una plataforma SDR (Software Defined Radio) en banda base y que está listo para soportar las nuevas tecnologías de interfaces inalámbricos como son IEEE 802.11ac, LTE-Advanced o DVB-T2 entre otros [10]. En la figura 4 se puede observar el esquema en el que se basa la plataforma Cobra de IMEC para recoger y enviar datos por diferentes interfaces así como para procesar varios *streams* de datos simultáneamente.

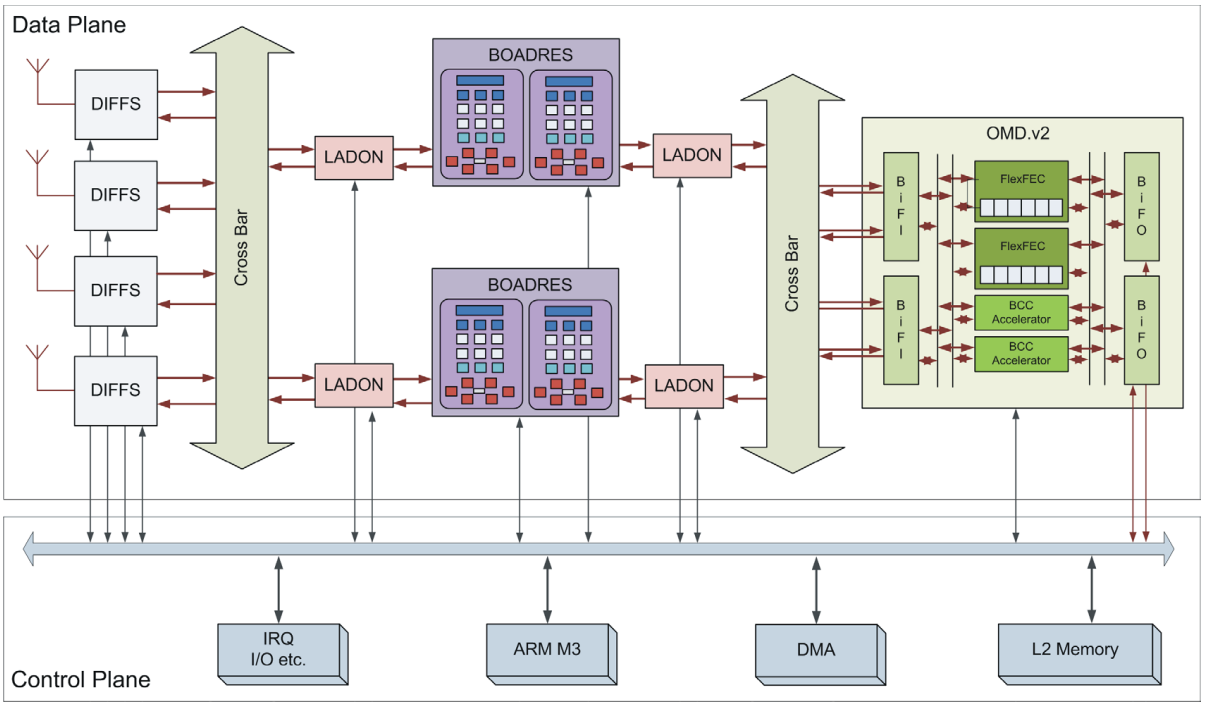

Fig. 4. Esquema de la plataforma Cobra de IMEC.

Otra posibilidad para implementar una SDR de una forma mucho más económica aunque a costa de reducir funcionalidad es adaptar varios interfaces radio que trabajen en diferentes frecuencias en una misma plataforma y diseñar un *software* para poder trabajar simultáneamente con ellos. Se trata de una solución al estilo Waspmote pero eliminando la limitación de dos transceptores. También se puede ver como una forma de adaptar lo que hace la plataforma Cobra de IMEC pero llevado al nivel de sistema empotrado. Así se consigue una plataforma que permite trabajar en distintas bandas, que aunque muy acotadas, tendrán la ventaja de estar restringidas a aquellas en las que es legal realizar comunicación inalámbrica para labores de investigación. Tendrá la limitación sin embargo que los esquemas de modulación quedan restringidos a aquellos que proporcione la interfaz.

Nota: la duda es que no sé si esto es estado del arte o ya estoy aventurando como realizar una SDR.

2.3 Arquitecturas software para implementar estrategias cognitivas en CWSNs

Un aspecto fundamental que deben poseer los nodos pertenecientes a redes de sensores que pretendan ser cognitivas es un software o, siendo más precisos, una arquitectura software que permita la implementación de estrategias cognitivas. El hecho de denominarlo arquitectura tiene su fundamento en que debe proporcionar un entorno capaz de ejecutar diferentes estrategias, más en concreto, debe

proporcionar las herramientas necesarias para que se puedan implementar aquellas que se necesiten. Se trata por tanto de una arquitectura para posibilitar la implementación de estrategias cognitivas concretas y tanto variables en el tiempo como dependientes de la aplicación.

Sin llegar al detalle de la arquitectura *software* y solo explicando la arquitectura general de funcionamiento de los nodos, NWCL presenta un modelo en [11] en el cual se realiza un sensado y posterior caracterización del entorno de la comunicación para de esa forma adaptar los parámetros y características de sus transceptores radio a unos necesidades de comunicación establecidas. Por ejemplo, transmitiendo en canales poco saturados se puede transmitir con menos potencia y de esa forma ahorrar energía. En la figura 5 podemos ver un diagrama con este modelo.

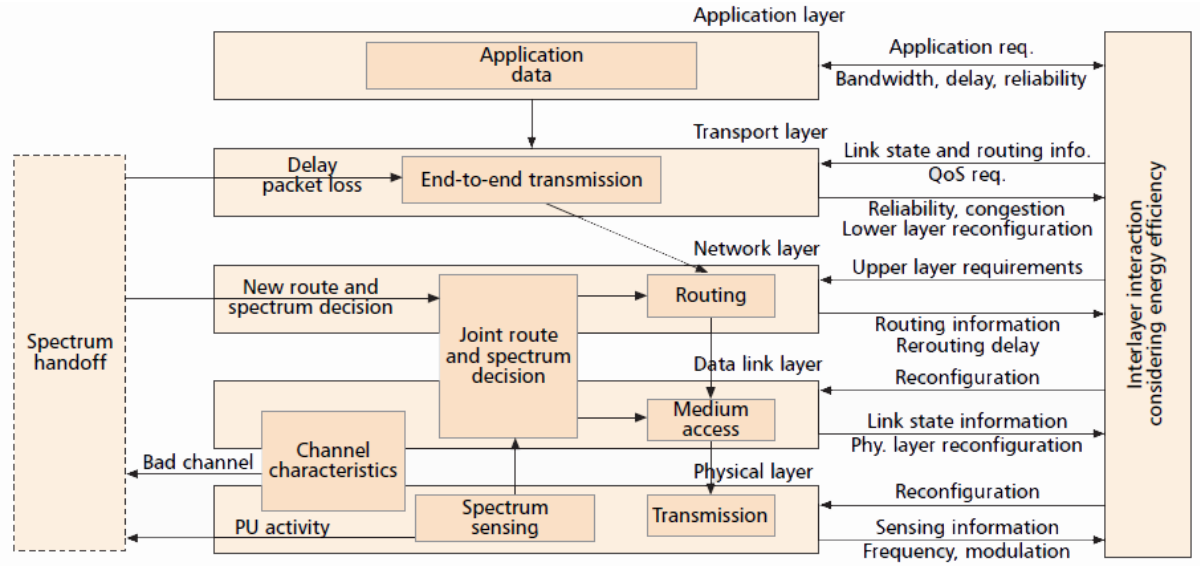

Fig. 5. Diagrama general con el funcionamiento del modelo propuesto por NWCL.

Como se observa en la figura 5 cada uno de los niveles de funcionamiento reportan información al "*Interlayer interaction considering energy efficiency*" que devuelve una respuesta al nivel correspondiente habitualmente con información para su reconfiguración.

Una de las primeras arquitecturas para redes cognitvias también fue presentada por Mitola en su *Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio*. Como es de esperar la arquitectura allí presentada está estrechamente relacionada con la definición que expone el mismo autor para red cognitiva y que se ha citado anteriormente en esta memoria. También se puede observar en la figura 6

que el ciclo cognitivo posee grandes similitudes con la definición dada por Thomas y que también se ha citado anteriormente en esta memoria.

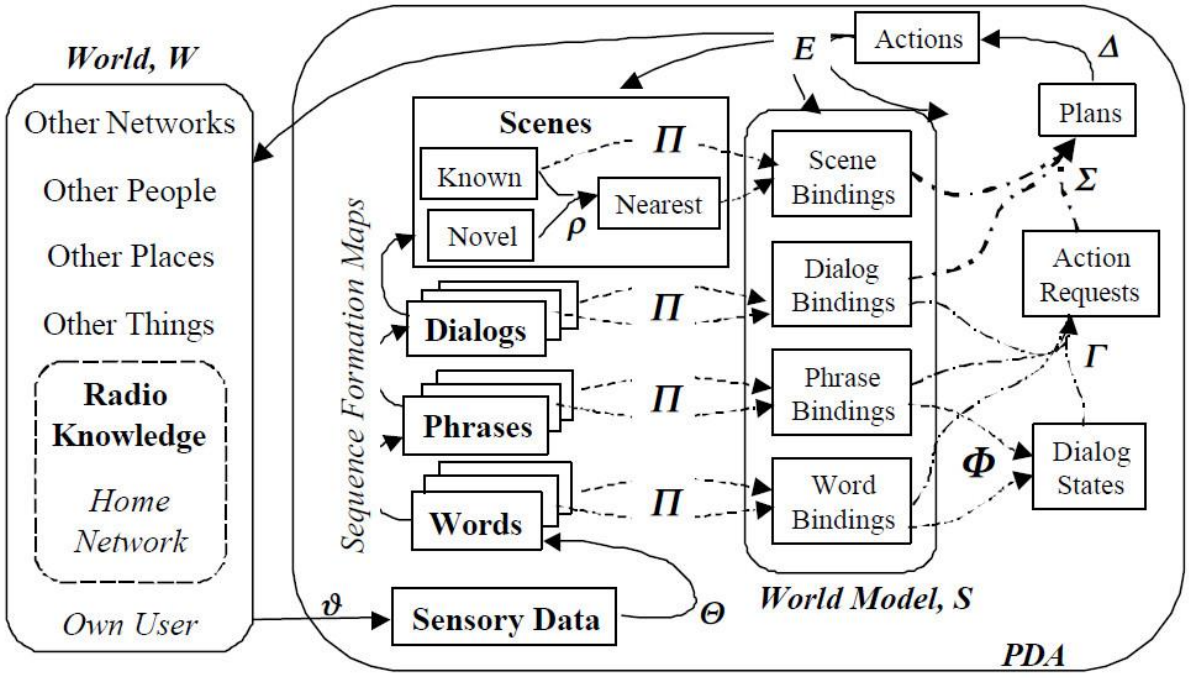

Fig. 6. Arquitectura software propuesta por Mitola.

Se observa en la figura 6 que aunque se trata de un modelo complejo, a grandes rasgos el modelo recoge información del exterior a través de "*Sensory Data*" y se forma un modelo de como es el "mundo exterior" con los datos recogidos. A continuación termina diseñando un plan y ejecutando acciones que a su vez modificarán el entorno y así el ciclo vuelve a empezar. Aunque el diagrama mostrado se aplicaba directamente al lenguaje natural, de ahí que lo que se recogen sean palabras que terminan formando frases, diálogos y contextos, tal y como se expone en el documento dicha arquitectura es, por ejemplo, directamente aplicable a información sobre las características radio del entorno. Es decir, lo que subyace en la figura mostrada es el hecho de que la información que se recoge se utiliza para terminar formando un idea del contexto en el que se encuentra el nodo, o PDA para Mitola, para poder actuar en consecuencia y de forma coherente. En la figura 7 se puede observar dicho ciclo más conceptualizado.

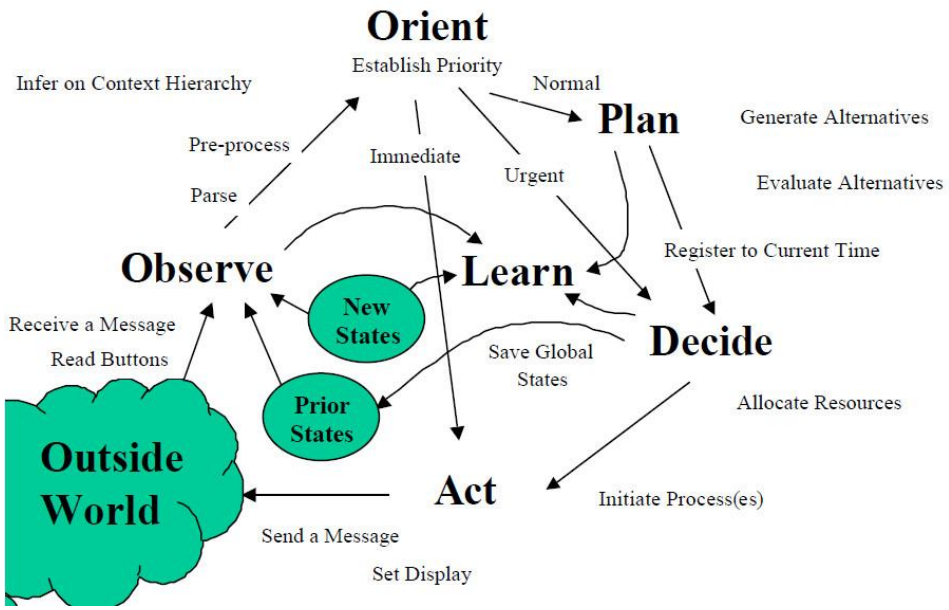

Fig. 7. Esquema del ciclo cognitivo.

En el artículo *A Context Driven Architecture for Cognitive Radio Nodes* [12] se presenta otra arquitectura donde los autores ponen especial énfasis en que el dispositivo o nodo debe "enteder" su estado y el de su entorno y que, además debe poseer mecanismos para mantener la consistencia de este "entendimiento" tanto en el tiempo como en el espacio. Se puede ver un diagrama explicativo de esta arquitectura en la siguiente figura.

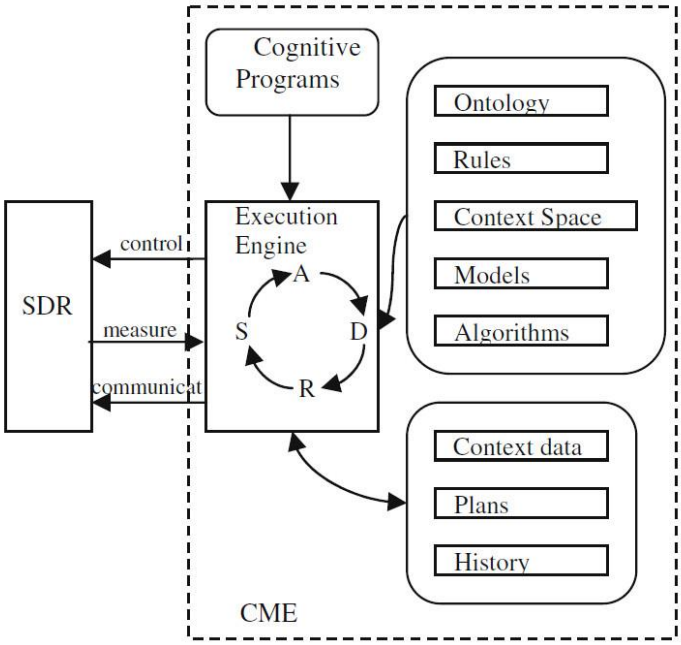

Fig. 8. Arquitectura de Nodo de una red cognitiva.

En la figura 8 se observa la estructura del "*Cognitive Management Entity*" que consta de un motor de ejecución que lleva a cabo las tareas de Sensado, Análisis, Decisión y Reconfiguración y que se alimenta del contexto, algoritmos, modelos y a su vez tiene en cuenta y modifica la historia, planes, etc. Este ciclo de ejecución, además, se realiza de acuerdo a las órdenes de los "*Cognitive Programs*" que es el que contiene las descripciones abstractas de los procesos cognitivos implementados. El gestor de la entidad cognitiva interactúa con la SDR modificando sus parámetros de acuerdo con los resultados del ciclo cognitivo y de la misma forma la utiliza para medir el entorno y para comunicarse.

Otra arquitectura software para procesos cognitivos fue presentada en el *Connectivity Brokerage* [13] que se realizó con la colaboración de las universidades de *UC Berkeley*, *TU Berlin*, universidad de Ozyegin, Universidad Politécnica de Madrid e IMEC. En la figura 9 se puede ver un diagrama explicativo de este modelo.

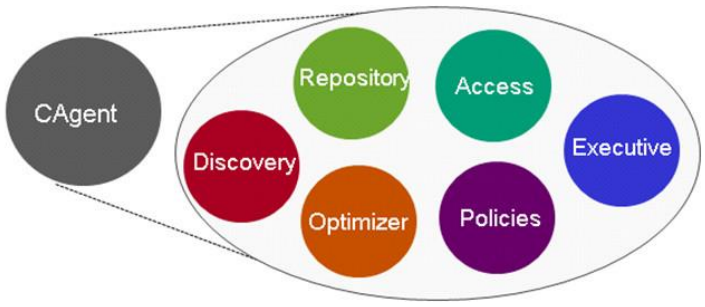

Fig. 9. Módulos de un agente cognitivo.

En el diagrama del *CAgent* se puede ver que está compuesto de varios módulos cuya función es, en la mayoría de los casos, bastante obvia teniendo en cuenta el nombre de los mismos. Además dado que esta es la arquitectura elegida para ser desarrollada en este proyecto, se omitirá la función de cada uno de ellos hasta el apartado en el que se detalla el diseño del módulo cognitivo y que se encuentra más adelante en esta memoria.

Esta arquitectura queda menos detallada por la figura que, por ejemplo, la anterior que se ha presentado, pero, a grandes rasgos, se puede adelantar que simplifica el modelo anterior en algunos aspectos. De esta forma, por ejemplo, cualquier información que caracterice parámetros del nodo, red, entorno o incluso cualquier cosa que sea útil para el proceso cognitivo y susceptible de parametrizar se podrá almacenar en el módulo llamado *Repository*. Ésta información, junto con la información que contenga el módulo *Policies* concerniente a unas determinadas políticas, determinarán, en la mayoría de los casos, la estrategia cognitiva que lleve a cabo el módulo *Optimizer*.

2.4 Simuladores de redes de sensores

Otro de los aspectos importantes a tener en cuenta a la hora de estudiar el estadio en el que se encuentra el desarrollo de las CWSNs es la cantidad y calidad de simuladores para las mismas.

Los simuladores tienen una gran importancia para caracterizar el comportamiento de las redes inalámbricas de sensores puesto que ayudan a predecir el comportamiento de la red sin la necesidad de desplegarla con los costes económicos que ello implica. Además se pueden forzar situaciones y condiciones que no siempre son fáciles que sucedan en un entorno real pero en las que es necesario conocer la reacción de la red, permitiendo así, ahorrar gran cantidad de tiempo. Por ello es fundamental el desarrollo de simuladores para CWSNs de la misma forma que lo han sido y están siendo para WSNs. En consecuencia, la observación de la evolución y calidad de los simuladores proporciona una idea de la importancia y desarrollo de estas redes.

A continuación se describen algunos de los simuladores más importantes de WSNs y CWSNs.

NS es una serie de simuladores del que hay tres versiones y una de sus particularidades es que se trata de *software* libre bajo licencia GNU GPLv2. Actualmente se encuentra en desarrollo la tercera versión (NS-3) aunque la más usada sigue siendo la segunda (NS-2). Estos simuladores se denominan como simuladores de red de eventos discretos y permiten definir topología de la red, protocolos y aplicaciones, configuraciones de tamaño de paquetes y así mismo proporcionan herramientas de análisis de los datos para obtener conclusiones.

Entre sus principales inconvenientes destacan la carencia de interfaz gráfica y la necesidad de poseer conocimientos de lenguaje de *script*, así como de teoría de colas y técnicas de modelado.

De este simulador para WSNs ha surgido otro simulador para CWSNs llamado SENDORA (Sensor Network for Dynamic and Cognitive Radio Access). Consiste en la adición de un módulo conocido como Miracle al simulador NS-2. Dicho módulo le proporciona al simulador la capacidad de albergar una arquitectura con varias capas y múltiples pilas de protocolos además de un modelo de propagación más real y que puede simular varios protocolos de red sobre el mismo canal físico. De esta forma SENDORA tiene capacidad para simular redes cognitivas.

Un simulador de WSNs que resulta interesante por su campo de aplicación es TOSSIM puesto que está diseñado para simular redes cuyos nodos poseen el

sistema operativo TinyOS. Dicho SO (Sistema Operativo) está muy expandido en el ámbito de nodos con procesadores con recursos de memoria y computación limitados como es habitualmente el caso de los nodos de una red de sensores. Sin embargo posee limitaciones como, por ejemplo, que no es apropiado para predecir consumos energéticos reales. Además no está preparado para simular redes cognitivas.

OMNET++ por su parte proporciona un entorno de trabajo con una librería con multitud de módulos de tal forma que cada investigador pueda construirse su propio simulador o escenario de trabajo de acuerdo a las necesidades de su simulación.

Sobre OMNET++ se desarrolla Castalia, que es un simulador para redes de sensores con un desarrollo muy activo por parte de la comunidad científica pues es de código abierto. A las ventajas de modularidad que proporciona OMNET++ se le añade la buena caracterización y modelado del canal real que posee Castalia. Es por estas particularidades por las que Castalia es un buen punto de partida para desarrollar un simulador de CWSNs mediante la adición de nuevos módulos a dicho simulador.

Actualmente en el laboratorio donde se ha desarrollado este proyecto se está trabajando activamente en añadir módulos al simulador Castalia que permitan la simulación de estrategias cognitivas en redes inalámbricas de sensores. De esta forma se podrán predecir comportamientos de redes cognitivas en entornos reales.

3. Descripción de las fases del proyecto

A continuación se describirán las fases en las que se puede dividir el presente proyecto.

La primera de ellas se describe brevemente en este mismo punto ya que, a pesar de no haber sido poco extensa en el tiempo, las metas que en ella se han conseguido estaban ligadas con la familiarización con el entorno, por lo que no es de especial interés una descripción muy detallada de la misma (DEBO JUSTIFICARME??).

Las otras dos fases tienen apartados reservados en esta memoria. En concreto la segunda fase tiene asignado el apartado número 4 y la tercera, que es la más extensa, los apartados 5, 6 y 7.

3.1 Fase de adaptación al entorno

3.1.1 Objetivo

En esta primera fase del proyecto el objetivo fue familiarizarse con la placa existente en el laboratorio y desarrollada por Fernando López [14] que se puede ver en la fotografía. Una descripción más detallada de la plataforma se realizara más adelante pero se puede adelantar que el fin de este periodo del proyecto era adaptarse a un entorno más o menos desconocido y que fundamentalmente involucra la programación de un micro-controlador y gestionar su comunicación con una serie de módulos que le dotan de conectividad inalámbrica.

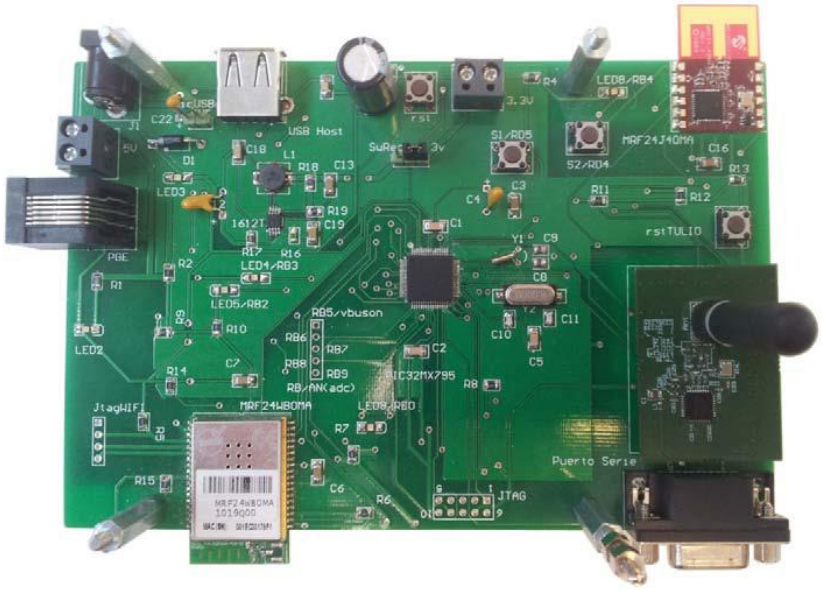

Fig. 10. Placa de desarrollo del laboratorio.

3.1.2 Descripción

La plataforma de desarrollo, B105CNBOARD a partir de ahora, consta de un MCU (Micro-Controller Unit) el PIC32MX795F512H, con 512 kB de Flash y 128 kB de SRAM y diferentes interfaces inalámbricos como son: un transceptor WiFi, un transceptor MiWi™ a 2.4 GHz y una sub-plataforma denominada Tulio [15], desarrollada por AWD, que consta de un SoC (System on Chip) CC1110 de Texas Instruments preparado para funcionar en 868 MHz y que está formado por un MCU y un transceptor.

En la toma de contacto con la plataforma se ejecutaron diversos programas de ejemplo que proporciona el fabricante Microchip tanto para su módulo WiFi como para el módulo MiWi™. Estos ejemplos habían sido adaptados previamente por Fernando López en su PFC (Proyecto Fin de Carrera) para poder ser utilizados en

la plataforma mencionada puesto que originalmente y para MCUs de 32 bits de Microchip solo se contemplaba su uso en la placa comercial del fabricante Explorer 16 con el módulo de adaptación para PIC32 “PIC32MX360F512L Plug-In-Module (PIM)”.

3.1.2.1 Adaptación al entorno de MiWi™

MiWi™ proporciona un entorno de desarrollo para aplicaciones inalámbricas en las que estén involucradas MCUs del mismo fabricante. Este entorno de desarrollo está formado por una serie de funciones que Microchip asocia a diferentes capas de su pila de protocolos, más en concreto funciones de la capa de control de acceso al medio (MAC) y funciones de la capa de aplicación.

Las primeras son funciones que, en general, incluyen directamente lecturas o escrituras de registros de los transceptores y que, por tanto, los modifican directamente, de ahí que se asocien con la capa MAC. Las funciones asociadas a la capa de aplicación son aquellas que, normalmente, poseen llamadas a varias funciones del primer tipo y además poseen algún algoritmo con el fin de proporcionar alguna funcionalidad o simplemente dar consistencia al entorno de desarrollo. Al juego de instrucciones de la capa MAC Microchip lo bautiza como funciones de MiMAC (MiWi™ MAC) y a las de capa de aplicación como MiApp (MiWi™ Application).

Debido al trabajo previo realizado en el laboratorio se consideró que la mejor alternativa era estudiar el funcionamiento de los ejemplos que Microchip proporciona para probar su entorno de desarrollo MiWi™. A priori el problema de esta opción era que dichos ejemplos no están pensados para ser ejecutados en cualquier plataforma sino solo en algunas determinadas de Microchip.

Éste trabajo de adaptación había sido realizado previamente por Fernando Lopez lo que sin duda facilitó el proceso. Sin embargo existía un déficit de información para replicar dicha adaptación, por ejemplo, con cada actualización del entorno de desarrollo.

Por ello, a continuación, se concretan brevemente los ficheros que hay que modificar para adaptar los ejemplos de Microchip y, en general, la implementación de MiWi™ a la placa B105CNBOARD u otra plataforma diferente a las propias de desarrollo del fabricante:

- Microchip/WirelessProtocols/console.c: para configurar correctamente la salida del puerto serie, UART en concreto, que se utiliza para enviar mensajes de *debugueo* hacia un PC.
- Microchip/Transceivers/<Nombre del transceptor>.c: porque en él se define la rutina de interrupción que se debe atender cuando el transceptor

correspondiente lo requiere. Por tanto la definición de dicha rutina deberá ser coherente con el conexionado físico de la plataforma. Habitualmente se utilizará una interrupción externa del MCU para tal efecto dado que el transceptor generará un nivel bajo o alto en el pin que corresponda.

- Microchip/WirelessProtocols/MSPI.c: en este fichero se utilizan dos funciones que deben ser definidas en el proyecto y cuya función es la de enviar y recibir datos por el SPI que conecta el MCU con el transceptor.
- Microchip/Include/WirelessProtocols/SymbolTime.h: en este archivo hay algunas definiciones temporales basadas en el supuesto de que se utilizan unos valores para la frecuencia de reloj del sistema y para el factor de división del bus de periféricos concretos y, por tanto, cuando se utiliza una plataforma con valores diferentes, esas definiciones temporales no corresponden a valores de tiempo “real” y se producen inconsistencias en la pila fundamentalmente porque hay tiempos de guarda que dejan de respetarse.

3.1.2.2 Pruebas con el entorno de MiWi™

Una vez listo el entorno con los ficheros de los que depende la pila de MiWi™ el siguiente paso fue probar y evolucionar en lo necesario un ejemplo de Microchip para terminar enviando datos entre nodos.

Por ello se eligió como punto de partida un ejemplo del entorno de desarrollo de MiWi™ llamado *Feature Demo* y se modificó para ir cumpliendo diferentes objetivos intermedios hasta llegar a un programa que simulara un envío continuo de datos y que, por tanto, permitiera obtener resultados reales como, por ejemplo, de tasa de envío efectivas de datos o de número de paquetes que ha sido necesario retransmitir cada cierto volumen de información enviada. De esta forma se podría adquirir una idea del comportamiento de los nodos en situaciones de aplicaciones semejantes a las reales.

Se detallan a continuación algunos aspectos de este programa de envío de datos pseudo-continuo:

- Es pseudo-continuo porque en realidad lo que hay es un envío de un buffer de datos grande, en concreto 30.000 bytes por lo que se pueden extrapolar algunos resultados.
- El envío del buffer se realiza en paquetes de tamaño configurable aunque restringido a un máximo teórico de 128 bytes que en la práctica será mucho menor para dejar espacio para las cabeceras de la pila de MiWi™.
- El envío de cada uno de los paquetes se hace de forma segura, lo que significa que se activa el cifrado que proporciona el entorno de desarrollo de MiWi™.

3.1.2.3 Otras pruebas

Con el fin de familiarizarse lo máximo posible con la plataforma de desarrollo y con los interfaces inalámbricos se realizaron otras pruebas adicionales como, por ejemplo, ejecutar envíos simples pero utilizando un nodo de pasarela para así probar que no existían problemas cuando se realizará un rutado. También se hicieron algunos ejemplos con la interfaz WiFi como uno en el cual se puede acceder a una página *web* alojada en la plataforma B105CNBOARD a través de dicha interfaz y utilizando la pila de protocolo de TCP/IP.

Dichos ejemplos realizados con la interfaz WiFi, aunque interesantes y didácticos, no fueron de gran utilidad puesto que no se siguió profundizando en ellos debido a que se empezó a poner en duda la conveniencia de tener en un nodo para CWSNs varios interfaces en la misma banda de frecuencia de trabajo y además con un consumo considerable en el caso de WiFi.

Por último y para completar esta primera fase y además servir de apoyo a las siguientes se realizaron otras dos tareas:

- La primera de ellas consistió en crear unas librerías con unas funciones para realizar el envío de las trazas de *debug* a través del USB en vez de la interfaz UART, y de este modo reducir el número de cables y conexiones necesarios para trabajar con la plataforma.
- La segunda se centró en buscar la forma de hacer un escaneo de canales en busca de portadora. Por defecto el transceptor y el entorno de desarrollo proporcionan las herramientas para realizar el escaneo de energía en los canales, pero no el de portadora. Existen en el transceptor varios registros para configurar dicho tipo de escaneo, sin embargo no aparecen documentados los que permitan leer el resultado del mismo. Se realizaron volcados de los registros del transceptor en busca de algún bit no documentado de alguno de los registros que proporcionara dicha información pero no se consiguieron resultados concluyentes.

3.2 Fase de caracterización de la evolución del nodo existente

3.2.1 Objetivo

En esta fase el objetivo fundamental era el de caracterizar los requisitos que debería tener un nuevo el diseño de un nuevo nodo.

El nodo desarrollado por Fernando López cumplía sobradamente la función para la que fue diseñado y que era fundamentalmente servir como prueba de concepto de un nodo para una red de sensores cognitiva.

Como evolución natural de este nodo y teniendo en cuenta el objetivo del diseño aparecían nuevos requisitos, algunos imprescindibles y otros deseables. Dichos requisitos junto con las decisiones finales sobre las especificaciones del nodo fueron determinados por consenso entre varios miembros del laboratorio y con Alvaro Araujo a la cabeza.

3.2.2 Descripción

Uno de los primeros pasos a dar sería, por tanto, el de la definición general del nuevo nodo.

En la tabla 2 se muestra un resumen de los elementos que se identificaron como importantes en la concepción del nuevo nodo.

<table>
<tr><th>Objetivo</th><th>Requisitos imprescindibles</th><th colspan="3">Requisitos deseables</th><th>Decisiones de diseño</th></tr>
<tr><td rowspan="3">Tiene que ser un nodo de una WSN</td><td>Bajo consumo</td><td rowspan="3">Utilizar un único MCU (menor consumo)</td><td colspan="2" rowspan="3">Tamaño justo para desarrollo</td><td rowspan="3">Hacer toda la placa con MCU PIC32 (Microchip)

Eliminar Tulios

Añadir transceptor de Microchip</td></tr>
<tr><td>Recursos escasos</td></tr>
<tr><td>Bajo coste</td></tr>
<tr><td rowspan="4">Debe ser apto para una red que además tenga capacidades cognitivas</td><td>Trabajar en, al menos, dos bandas ISM (868MHz, 2.4GHz) modificando todos los parámetros posibles</td><td rowspan="4">Trabajar en 3 bandas ISM (433MHz, 868MHz, 2.4GHz)</td><td colspan="2" rowspan="4">Posibilidad de conmutación real de cabezales radio. Un transceptor para distintas frecuencias</td><td rowspan="4">Añadir un transceptor a 434MHz

Fusionar e integrar lo más posible todas las pilas

Analizar la influencia de los distintos tipos de antena</td></tr>
<tr><td>Posibilidad de conectarse algún nodo con 802.11</td></tr>
<tr><td>Conexión para antenas externa</td></tr>
<tr><td>Utilizar un único entorno de desarrollo</td></tr>
<tr><td rowspan="4">Tiene que servir como prueba de concepto de diferentes estrategias</td><td>Seguridad</td><td rowspan="4">Cargador de aplicaciones remoto</td><td rowspan="4">Monito-rización de carga de batería</td><td rowspan="4">Conexión para batería</td><td rowspan="4">El procesador y los transceptores proporcionen herramientas y capacidad para cumplir esos requisitos</td></tr>
<tr><td>Bajo consume</td></tr>
<tr><td>Sincronización</td></tr>
<tr><td>Calidad de Servicio (fiabilidad)</td></tr>
<tr><td>General</td><td>El tiempo de desarrollo de cada módulo del dispositivo tiene que verse compensado por los beneficios que la obtención de este módulo nos proporcione</td><td colspan="3"></td><td>Se utilizará el hardware disponible hasta que no se pueda seguir trabajando y se haga una nueva placa</td></tr>
</table>

Tabla 2. Descripción de requisitos y decisiones en la caracterización del nuevo nodo.

Algunas de las conclusiones más importantes que se extrajeron fruto de ese trabajo y que además afectarían al transcurso de este proyecto fueron se justifican a continuación:

- Continuar con PIC32 como plataforma de desarrollo frente a otras alternativas de MCUs como puede ser Atmel principalmente debido al trabajo ya realizado sobre la plataforma existente que además sería aprovechable y, así mismo, debido a la experiencia adquirida con Microchip por otros miembros del laboratorio.
- Eliminar WiFi como interfaz inalámbrica por ser redundante con la banda de trabajo de MiWi™ 2.4 GHz y además poseer un mayor consumo. Se ha marcado como requisito deseable que exista algún tipo de nodo que sea compatible con esta interfaz por el hecho de servir de pasarela de la red de sensores con otros tipos de redes que no posean interfaz del tipo MiWi™.
- Eliminar Tulio y poner un transceptor de Microchip para 868 MHz. Principalmente debido a una objetivo de homogeneizar el nodo hacia un mismo fabricante para MCU e interfaces inalámbricos. Esto facilita aspectos como la interacción MCU con dicho interfaces, soluciona potenciales problemas de compatibilidad, elimina un SoC y lo sustituye por un módulo solo transceptor reduciendo el consumo y además permite incluso reducir el tamaño de la plataforma.
- Añadir transceptor para 434 MHz para tener de esa forma otra posible banda de trabajo para las comunicaciones y aumentando las posibilidades desde el punto de vista cognitivo que proporciona el nodo.
- Elegir los módulos transceptores de tal forma que sean solo transceptor en vez de MCU+transceptor, como es el caso del Tulio, para minimizar el consumo y la complejidad del sistema. Como es necesario en cualquier caso que el nodo posea un MCU central para ejecutar la aplicación deseada, si es posible que ese mismo MCU gestione los interfaces se estará consiguiendo cumplir requisitos ya mencionados como reducción de consumo, complejidad, etc.
- Además los nuevos transceptores se eligieron todos de Microchip y todos compatibles con el entorno MiWi™ para de esta forma facilitar la integración del software de las interfaces en uno solo que formaría parte de la labor de Juan Domingo y expuesta en su PFC [16].

Una vez concretados unos requisitos hardware y, como paso previo a la próxima fase, se decidió probar las nuevas interfaces que se habían elegido para el nuevo nodo. Además era deseable hacerlo antes de que se desarrollara la nueva plataforma para detectar posibles puntos de fallo e intentar evitarlos en el diseño de la nueva PCB (*Printed Circuit Board*) y para tener un prototipo a corto plazo que

permitiera realizar pruebas y así estudiar su comportamiento sin tener que esperar a todo el ciclo de desarrollo, construcción y adaptación al nuevo nodo.

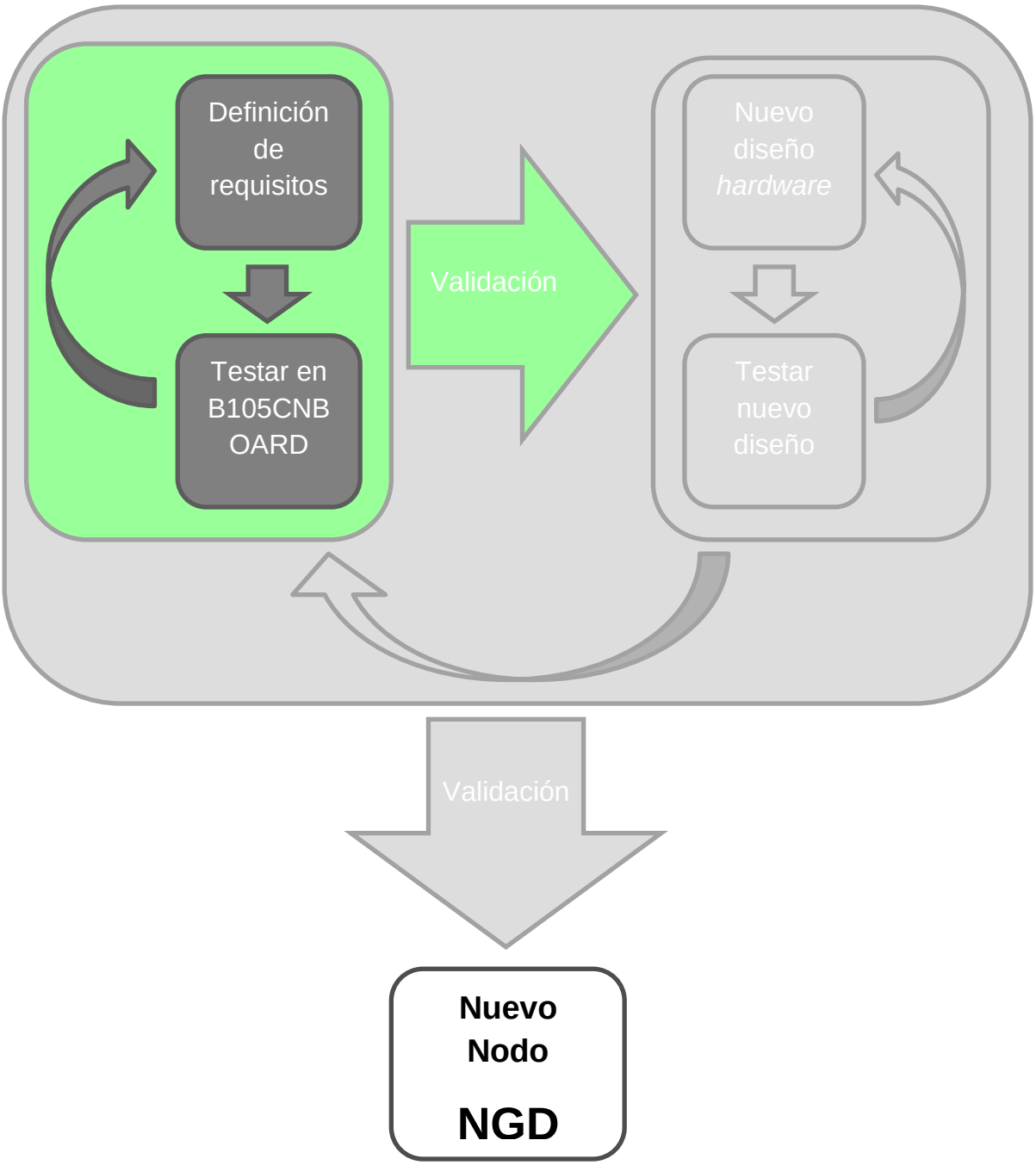

Fig. 11. Ciclo de diseño tentativo para el nuevo nodo.

En la figura 11 se puede observar el ciclo de creación del nuevo nodo que se tuvo en consideración. En él, se puede ver un cuadro de color verde que incluye las etapas que se realizaron en este proyecto respectivas a este proceso de creación, y que serán detalladas más adelante en esta memoria en el apartado correspondiente.

3.3 Fase de diseño e implementación de una arquitectura para CWSNs

3.3.1 Objetivo

El objetivo de la última y tercera fase de este proyecto era la implementación de una arquitectura *software* para poder implementar estrategias cognitivas y poder así, potencialmente, formar redes cognitivas inalámbricas de sensores.

Se presenta, por tanto, como una necesidad el hecho de definir una arquitectura para CWSNs como paso previo a poder formar una red de este tipo. El ámbito de las arquitecturas *software* para simular procesos cognitivos cuenta con una historia relativamente larga y diferentes propuestas de implementación. Sin embargo en el ámbito de las redes de sensores el número de alternativas disminuye, no están muy desarrolladas y además ya se han mostrado algunas en el estado del arte.

En esta fase se eligió una de las posibles arquitecturas en base a unos criterios qure se describirán cuando corresponda y se procedió a su implementación y adaptación a la plataforma existente B105CNBOARD.

3.3.2 Descripción

La tercera parte de este proyecto es además a la que más tiempo se dedicó en su desarrollo. Se diseñó y desarrolló una plataforma software que facilitara la implementación de estrategias cognitivas en el nodo. Este entorno estaba basado en la estructura presentada en el *Connectivity Brokerage*. Además de ser fiel a dicho modelo se han incluido algunos desarrollos extra que serán presentados en el apartado correspondiente con el fin de adaptar la arquitectura a la plataforma *hardware* y a requisitos que también serán descritos.

El desarrollo de este entorno se ha centrado en crear un esqueleto para todos los módulos que se describen en el *Connectivity Brokerage* como parte del *CRModule* (*Cognitive Radio Module*) y, además, añadir algunas funciones a cada uno de ellos para sentar las bases de un demostrador que ejecute una estrategia de optimización y simule algún comportamiento propio de una CWSN con al menos dos nodos.

En la figura 12 se puede observar la arquitectura que ya se mostró en el estado del arte y cuya función es acoger los elementos y procesos necesarios para realizar procesos cognitivos en la red.

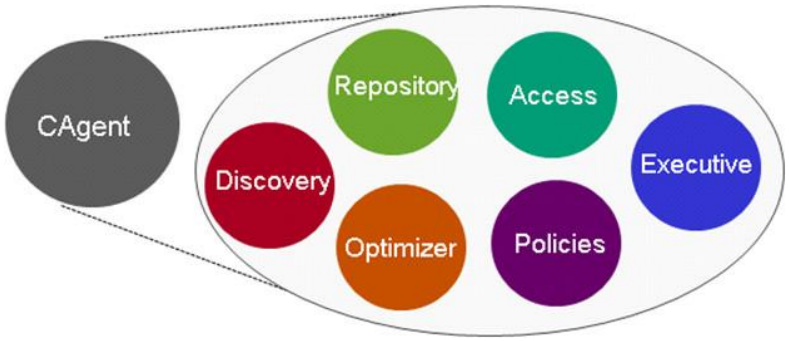

Fig. 12. Esqueleto de la arquitectura software para CWSNs.

Se pueden agrupar los diferentes hitos que se han logrado en esta fase en los siguientes puntos:

- Elección de la arquitectura.
- Implementación de la arquitectura.
- Ejecución de intra-nodo de una estrategia cognitiva.
- Desarrollo de módulos adicionales para conexión de módulos cognitivos entre diferentes nodos.
- Adaptación de la estrategia cognitiva para convertirla en una estrategia inter-nodo.

Por tanto, en los apartados en los que se detalla esta fase de diseño e implementación de la citada arquitectura, se mostrará también la implementación de otros módulos que ha sido necesario desarrollar para poder llevar a cabo estrategias cognitivas que involucren a más de un nodo y que permitan realizar demostraciones reales de las mismas.

4. Adaptación Hardware

Como se ha comentado, durante la realización de este proyecto se rediseñaron algunos aspectos de la plataforma de desarrollo. Uno de esos aspectos eran los interfaces inalámbricos, en los que se suprimieron los módulos WiFi y Tulio de la plataforma B105CNBOARD para dar paso a un transceptor de Microchip capaz de trabajar en las bandas de 434 MHz y 868 MHz.

4.1 Descripción del problema

Para ser coherente con el ciclo de diseño expuesto en la figura 11, previo al desarrollo de una nueva plataforma que adoptara los nuevos interfaces, estos se deberían probar en la placa B105CNBOARD.

Para ello se eligieron unos módulos de evaluación que proporciona el fabricante Microchip llamados “MRF49XA PICTail/PICTail Plus Daughter Board”. Dichos

módulos están compuestos por el transceptor MRF49XA con su cristal correspondiente de 10 MHz, el circuito de adaptación a la antena, una memoria para almacenar la dirección que tendrá la interfaz, puertos para medir la corriente eléctrica consumida, pines de conexión y la propia antena. El transceptor es el mismo tanto para el módulo a 868 MHz como para 434 MHz siendo el circuito de adaptación y la longitud de la antena lo que cambia.

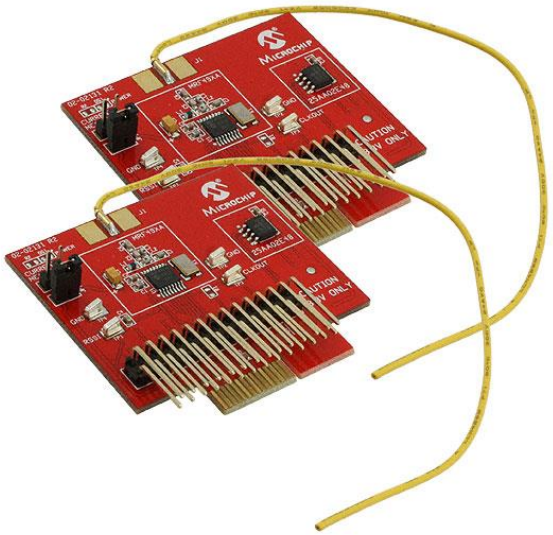

Fig. 2. Placas auxiliares de Microchip para comunicación en 434 MHz.

Existían dos alternativas para realizar el prototipado, bien conectando los nuevos módulos con cables y que estos fueran soldados a la placa o bien realizando una nueva placa auxiliar que se pudiera acoplar a la B105CNBOARD, de tal forma que muchas de las conexiones se realizaran directamente por el trazado y otras mediante cables específicos de prototipado que permitieran conexiones más fiables.

Fig. 3. Cable específico de prototipado.

A continuación se describen varios aspectos de la fase de adaptación hardware.

4.2 Descripción de la solución

La solución adoptada se puede descomponer a su vez en varias etapas que van desde los requisitos que llevaron a la toma de decisiones, diseño de PCB, adaptación *hardware* y hasta la adaptación *software*.

4.2.1 Requisitos

Antes de tomar una decisión sobre cómo adaptar los nuevos interfaces en la plataforma existente para realizar diferentes pruebas se tuvieron en cuenta que restricciones y requisitos se presentaban. Por eso fue necesario definirlos con antelación:

- **Tiempo**: un factor muy importante es que el diseño para adaptar los nuevos módulos a la plataforma fuera eficiente en tiempo. Es decir, realizar una adaptación un poco mejor que otra pero que supusiera un coste adicional en tiempo de un factor diez no resultaría apropiada.
- **Fiabilidad**: dado que se realizarían un número de pruebas considerable con los nuevos transceptores e incluso con varios a la vez era importante poder descartar que los posibles problemas que surgieran no vendrían dados por una conexión poco fiable con la plataforma B105CNBOARD.
- **Adaptabilidad**: era deseable que hubiera que hacer los mínimos cambios en la plataforma a la hora de adaptar esta expansión. Cambios a evitar podían ser cortar pistas del trazado de la placa, soldar a pines del MCU y en general cualquier operación que complicara la adaptación y que hiciera la misma poco replicable o incluso comprometiera la fiabilidad del dispositivo. Se puede decir que en parte se buscaba desarrollar un mecanismo de adaptación “natural” a la plataforma.
- **Versatilidad**: aunque no se trate de un requisito imprescindible sí que sería deseable que se pudieran probar los nuevos módulos simultáneamente. Incluso que fuera posible que coexistieran con el módulo WiFi puesto que, aunque no forme parte del futuro nuevo nodo (NGD, Next Generation Device) si se contempla la posibilidad que algunos nodos “especiales” lo mantengan para dotarles de conectividad.

Por estos requisitos expuestos se decidió tomar la decisión de realizar un diseño de una plataforma auxiliar en un PCB que se adaptara a la B105CNBOARD por ser la solución que mejor se adaptaba globalmente a las necesidades expuestas.

4.2.2 Diseño PCB

Inicialmente uno de los factores a estudiar era si sería posible adaptar a la plataforma existente una nueva PCB para adaptar los nuevos módulos y si habría pines suficientes para hacerlo.

Dado que el nodo B105CNBOARD disponía de un número considerable de pines de propósito general y otros específicos que no estaba siendo utilizados conformaba una buena base para el nuevo prototipo. Además, algunos de los pines de propósito específico (UART, SPI) que no estaban libres a priori, en realidad estaban siendo utilizados por los módulos Tulio y WiFi por lo que, al no contar ya con ellos, dichos pines quedarían disponibles para ser usados.

Por otro lado se disponía de algunas placas B105CNBOARD parcialmente soldadas, a falta de algunos componentes como el módulo WiFi. Precisamente es la huella en la PCB (*footprint*) de este componente la que se decidió utilizar para conectar la nueva placa de expansión debido a que proporciona muchos pines, hacia diferentes entradas del MCU, que podrían ser utilizados por los nuevos módulos inalámbricos que se conectaran a esta PCB de expansión.

En la placa de expansión diseñada podemos distinguir varias partes diferenciadas como podemos ver en la figura 2.

- **Zona de acoplamiento**: etiquetada como “Exp. Conn” y remarcada en color amarillo es la zona que debe ser soldada directamente a la huella del módulo WiFi de la placa B105CNBOARD.
- **Huellas para módulos**: aquí tenemos tres elementos remarcados en color rojo y su objetivo es albergar los módulos WiFi, en la huella etiquetada como MRF24WB0MA, y los dos módulos para 868 y 434 MHz en las huellas marcadas como “PICTail Plus Conn. 1” y “PICTail Plus Conn. 2”.
- **Zonas de interconexión**: estas zonas, marcadas en color azul claro, son zonas donde se colocarán *jumpers*. Dos de estas filas de jumpers, en concreto las etiquetadas como “MRF49XA Conn. 1” y “WiFi Conn.”, sirven para habilitar el módulo con conexión en “PICTail Plus Conn. 1” o el módulo WiFi respectivamente. Hay que notar que estos dos módulos no pueden estar habilitados por sus *jumpers* respectivos simultáneamente puesto que comparten líneas que van a los mismo pines del MCU. La fila de *jumpers* etiquetada “MRF49XA Conn. 2” tiene como objetivo conectar cables de prototipado y que irán a otros conectores en la placa B105CNBOARD.

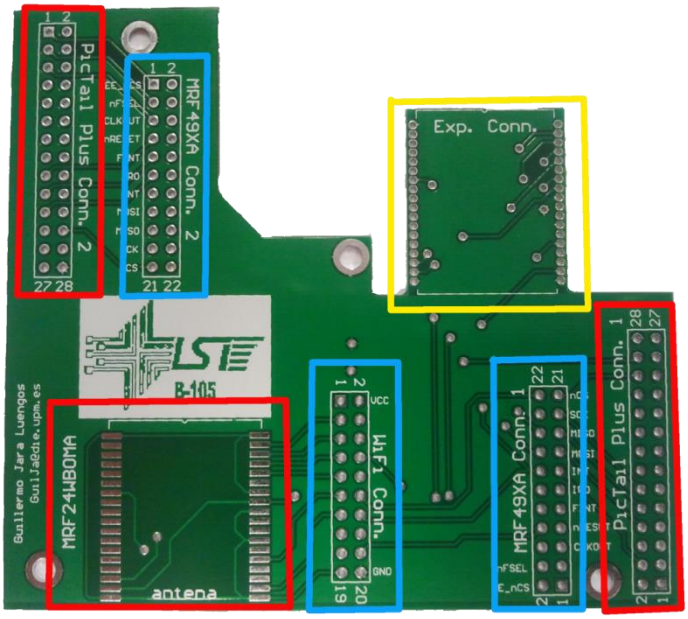

Fig. 4. PCB de expansión.

4.2.3 Adaptación

Con el fin de describir la metodología seguida para realizar la adaptación de la placa de expansión a la original se concreta el proceso en tres etapas: acoplamiento físico, configuración del hardware, configuración del software.

4.2.3.1 Acoplamiento Físico

Como se ha comentado se utilizó la huella del módulo WiFi de la placa B105CNBOARD para acoplar la de expansión. Para poder realizar la soldadura se realizaron vías de un tamaño suficiente para que el estaño fluyera con facilidad a través de ellas y mojara los *pads* de la huella del WiFi al aplicar calor con el soldador de punta fina.

Por otra parte se añadió una vía de sujeción haciéndola coincidir con otra de la placa original cuyo objeto era atornillar en ella una pata. Este anclaje mecánico reduciría el stress al que sería sometida la zona soldada y se evitaría así que saltaran las soldaduras de interconexión. Por último también se añadieron otras vías para colocar patas y que el conjunto fuera estable incluso al conectar los módulos de los nuevos interfaces. En la figura 5 podemos ver sobreimpresionada la zona de la placa BC105CNBOARD donde está el módulo WiFi y que sería donde en su lugar se acoplaría la placa de expansión. Es en la figura 6 donde vemos simuladamente, con un efecto de solapamiento y transparencia, como se acoplaría la nueva placa.

Fig. 5. Zona del módulo WiFi de la placa B105CNBOARD.

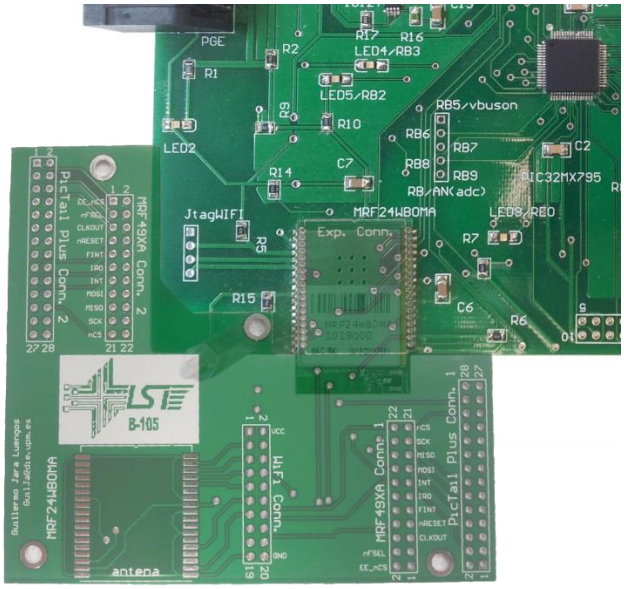

Fig. 6. Solapamiento figurado de la placa de expansión con la placa B105CNBOARD.

Una vez que se realiza la soldadura de ambas placas y se fija con una pata y una tuerca el resultado es el mostrado en la figura 7. Además se observa que ya se han soldado en la placa de expansión las tiras de *jumpers* para interconexión así como los conectores para los módulos con conector PICTail Plus.

Fig. 7. Placa de expansión soldada a la placa B105CNBOARD.

4.2.3.2 Configuración Hardware

En este apartado se abordará la configuración que se eligió para conectar el nuevo hardware con la MCU y la repercusión que tiene en algunos archivos del entorno de desarrollo cuyo propósito es precisamente adaptarse a dicha configuración.

Como se ha mencionado anteriormente, las placas de los módulos con el transceptor MRF49XA se conectan mediante el conector PICTail Plus cuya ordenación de pines podemos ver en el siguiente diagrama.

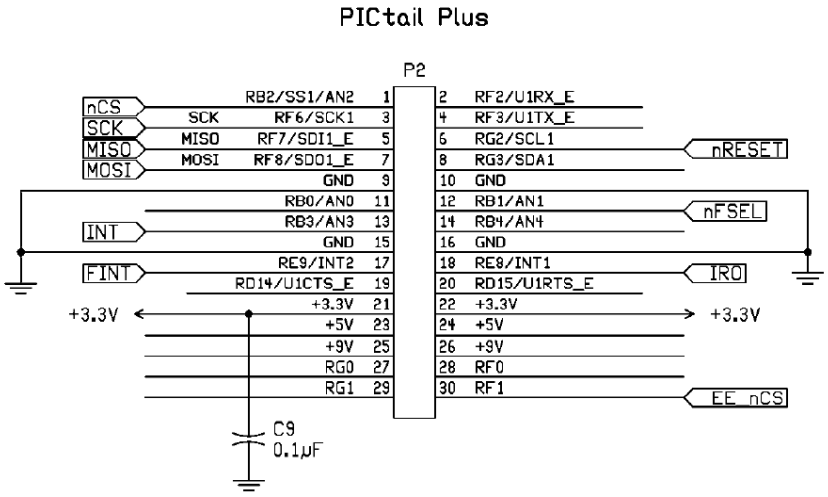

Fig. 8. PICTail Plus pinout.

Donde se observa que algunos de los pines tienen al final de la línea una etiqueta con el nombre del pin del transceptor al que están conectados. La utilidad de estos pines es la siguiente:

- SDI: señal de datos de entrada de la interfaz SPI.
- SCK: señal entrada del reloj de la interfaz SPI.
- CS: *Chip Select* para la interfaz SPI.
- SDO: señal de datos de salida de la interfaz SPI.
- IRO: salida de petición de interrupción hacia la MCU.
- FSEL: selector de la cola FIFO.
- FINT: interrupción asociada a la cola FIFO.
- CLKOUT: salida de reloj.
- RESET: activa a nivel bajo para resetear el transceptor.
- INT/DIO: entrada de interrupción para notificar de la misma al MCU por el pin IRO.
- EE_CS: activo a nivel bajo es el *chip select* de la memoria EEPROM.

Hay que notar que ni el pin EE_CS ni el CLKOUT se conectaron puesto que no era necesario leer o escribir en la memoria EEPROM porque no se utilizó para la realización de este proyecto ni, por otro lado, era necesario una salida de reloj puesto que la placa B105CNBOARD ya tenía un cristal para generar el reloj del sistema.

Observando las conexiones disponibles en el módulo WiFi de la placa B105CNBOARD se puede concluir que dispone de todos los pines necesarios para interactuar con uno de los módulos. Solo por una cuestión de versatilidad se decidió conectar la línea de FINT a un pin de interrupción CN (Notificación de Cambio) que no está disponible en el módulo WiFi por si en algún momento resultaba interesante generar en la MCU una interrupción con ese evento, a pesar de que el entorno de desarrollo de MiWi^{TM} no lo hace y solo utiliza para ello la línea IRO.

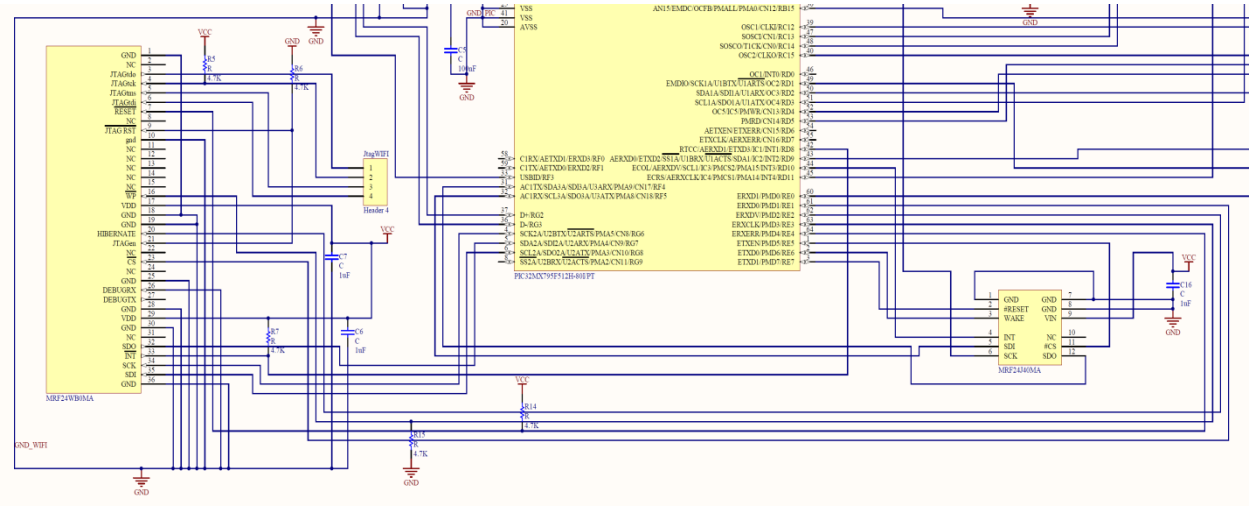

Fig. 9. Esquemático de la zona del módulo WiFi y MCU.

En la figura 10 se pueden ver las conexiones necesarias para que interactúen el módulo con transceptor y la MCU.

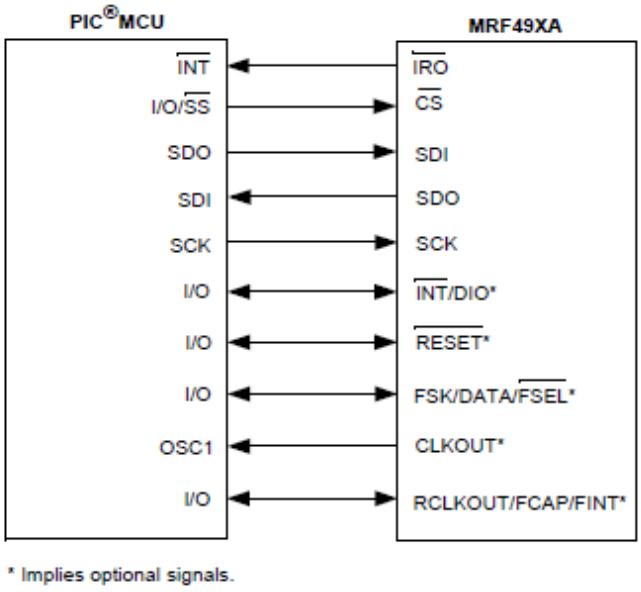

Fig. 10. Conexión de los módulos con el MCU.

En la figura 11 se ven, resaltadas en verde, las líneas que se han utilizado en el esquemático para la interconexión con el módulo con transceptor que se conecta en el conector “PICTail Plus Conn. 1”.

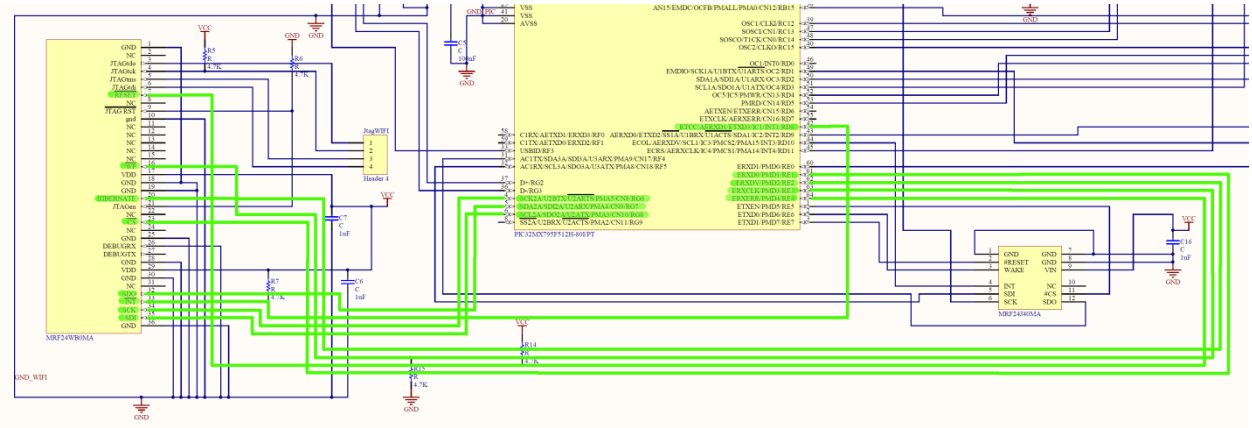

Fig. 11. Esquemático con las líneas utilizadas para interconectar reslatadas.

Como se ha mencionado falta la conexión con el pin IRO que se llevó a la línea donde estaba conectado el LED5 y que en la MCU es un pin capaz de generar una interrupción por CN (Change Notification).

En la siguiente tabla se pueden ver en detalle las conexiones realizadas. Se especifican el pin del módulo “MRF49XA PICTail/PICTail Plus Daughter Board”, el módulo de la placa B105CNBOARD donde está el pin al que se quiere conectar, el nombre del pin en dicho módulo y el pin de la MCU al que pertenece dicha línea.

PICtail Plus Conn.	Placa B105CNBOARD (Módulo)	Pin en el módulo	Pin en PIC32MX
INT***	WiFi	Hibernate (20)	RE2 (62)
FINT	LED5	LED5	CN4 (14) (RB2)
IRO	WiFi	/I/N/T (33)*	INT1 (42) (RD8)
nFSEL	WiFi	/W/P (16)**	RE3 (63)
nRESET	WiFi	/R/E/S/E/T (7)*	RE4 (64)
nCS	WiFi	/C/S (23)	RE1 (61)
MISO	WiFi	SDO (32)	SDI2A (5) (RG7)
MOSI	WiFi	SDI (35)	SDO2A (6) (RG8)

SCK	WiFi	SCK (34)	SCK2A (4) (RG6)

Tabla 1. Conexión entre B105CNBOARD y "PICTail Plus Conn. 1"

* Estos pines están soldados en la placa B105CNBOARD con un pull-up.

** Estos pines están soldados en la placa B105CNBOARD con un pull-down.

***En este proyecto no se ha configurado puesto que no utilizamos ese INT, que es para generar una interrupción en el transceptor (y no viceversa que es lo normal y para lo que sirve IRO).

Se ha comentado anteriormente que la placa de expansión también proporciona la posibilidad de conectar otro módulo "MRF49XA PICTail/PICTail Plus Daughter Board" para tener un total de dos (en 868 y 434 MHz). En la figura 12 se ven resaltadas, de nuevo en verde, las líneas de la placa B105CNBOARD que se utilizan para la comunicación con este otro módulo que a partir de ahora será referido como "alternativo" o "alt." para abreviar.

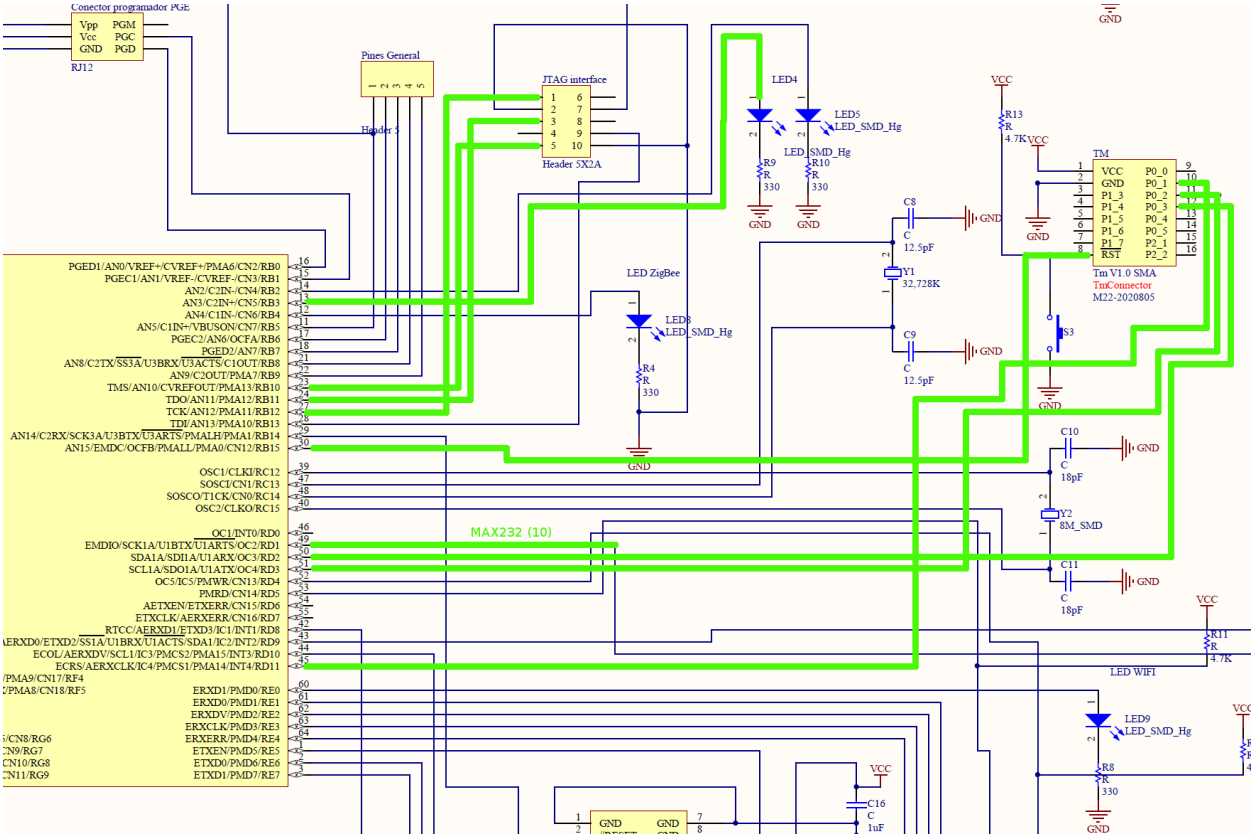

Fig. 12. Conexiones utilizadas por el módulo "MRF49XA PICTail/PICTail Plus Daughter Board" alt.

Hay que tener en cuenta que todas estas conexiones, como se muestra en la siguiente tabla, no se han realizado a través del módulo WiFi y que por lo tanto se han hecho con la ayuda de cables de prototipado, conectados por un lado al *header* de la placa de expansión etiquetado con “MRF49XA Conn. 2” y por el otro a las líneas mostradas en la figura 12. Para elegir estas líneas se tuvo en cuenta por un lado que fueran compatibles con el propósito del pin de la placa “MRF49XA PICTail/PICTail Plus Daughter Board” al que se conectaban (SPI, INT o IO) y por otro que la línea fuera fácilmente accesible. Por esta última razón es por lo que, tal y como se aprecia en la figura 12, la mayoría de las líneas terminan en algún conector. Las líneas que no terminan en ningún conector como pueden ser las líneas de LED4 y la etiquetada en la figura 12 como MAX232 (10) tienen una vía por la que se hizo pasar un *header* y a continuación se soldó dicho *header* a la vía. Un ejemplo de esto se puede ver en la figura 13 rodeado por un rectángulo de color rojo.

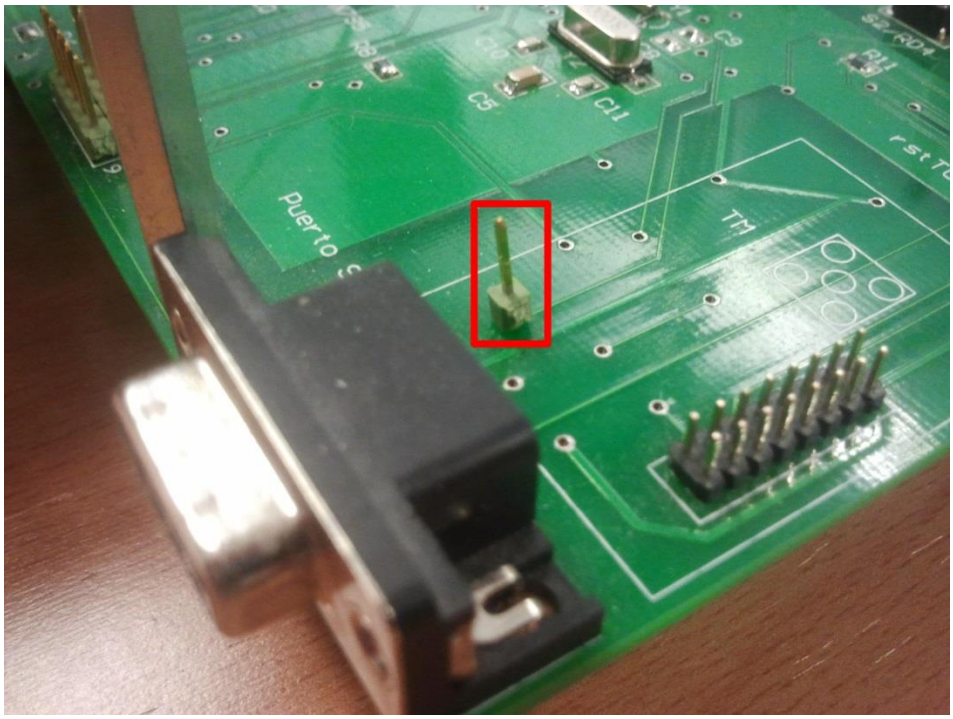

Fig. 13 Detalle de header insertado en una vía perteneciente a la línea MAX232 (pin 10).

En la tabla 2 se observan las mismas relaciones mostradas en la tabla 1 pero ahora para el módulo alternativo.

PICtail	Placa B105CNBOARD (Módulo)	Pin en el módulo	Pin en PIC32MX

INT***	JTAG	JTAG(1)	RB12 (27)
FINT	LED4	LED4	CN5 (13) (RB3)
IRO	Tulio	P0_1 (10)	INT4 (45) (RD11)
nFSEL	JTAG	JTAG(3)	RB11 (24)
nRESET	Tulio	/R/S/T*	RB15 (30)
nCS	JTAG	JTAG (5)	RB10 (23)
MISO	Tulio	P0_3 (12)	SDI1A (50) (RD2)
MOSI	Tulio	P0_2 (11)	SDO1A (51) (RD3)
SCK	MAX232	MAX232 (10)	SCK1A (49) (RD1)

Tabla 2. Conexión entre B105CNBOARD y "PICTail Plus Conn. 2"

En la figura 14 se puede ver una imagen con ambos módulos "MRF49XA PICTail/PICTail Plus Daughter Board" conectados en sus conectores y con todos los cables y *jumpers* necesarios para su correcto funcionamiento con la placa B105CNBOARD.

Figura 14

4.2.3.3 Configuración Software

Para realizar la configuración del software hubo que redefinir algunas definiciones y cambiar las variables que se pasaban a algunas llamadas a funciones.

El entorno de desarrollo de MiWi™ hace uso de aliases, asignados mediante *defines* de lenguaje C, a registros del MCU en los que se configura la dirección del puerto correspondiente para que sea de entrada o de salida y a registros de donde se lee o escribe el nivel lógico que se ha recibido por ese puerto o el que se quiere enviar por él. Éstos aliases se utilizan, por ejemplo, para hacer referencia de una forma rápida e intuitiva a registros de periféricos del MCU.

Por otra parte dicho entorno también realiza llamadas a funciones definidas en la librería de periféricos de Microchip que sirven precisamente para configurar periféricos que utiliza el entorno para interactuar con el transceptor. Dichos periféricos pueden ser independientes de donde está conectado el transceptor o no. Un ejemplo de dependencia es el periférico SPI (*Serial Peripheral Interface*). Puesto que la MCU posee varios módulos SPI cuyos registros están asociados

directamente a unos pines concretos, se debe configurar aquel que corresponde a los pines donde se encuentra conectado el transceptor físicamente.

Previo a la realización de este proyecto ya se disponía de la configuración *software* para utilizar el módulo MRF24J40MA (con transceptor MRF24J40) por lo que el código existente pudo ser utilizado a modo de guía. Sin embargo hay que tener en cuenta que los módulos transceptores "MRF49XA PICTail/PICTail Plus Daughter Board" no poseen los mismos pines que el MRF24J40MA y que hubo que tener especial cuidado en la correcta configuración de las llamadas a funciones mencionadas.

Así mismo, para dotar a la configuración software de mayor versatilidad en cuanto a la adaptación del software a diferentes transceptores se tomó la decisión de que fuera posible configurar cuál de los tres módulos iba a ser utilizado por el programa principal mediante un *define* inicial en un archivo de cabecera.

Referencia a la parte del apéndice donde está el código??

4.3 Pruebas

Para realizar las pruebas con los nuevos transceptores se desarrolló un programa genérico en el sentido que podía funcionar de igual manera para los tres interfaces radio mediante una selección previa a la programación de la plataforma. El objetivo de las pruebas era por una parte probar que las dos nuevas interfaces se comportaban de una forma similar a la que ya existía (MRF24J40), probar la búsqueda, creación y unión a una red por parte del transceptor y por último la realización de escaneos de energía de los canales que disponga la banda de trabajo correspondiente.

4.3.1 Descripción

El programa configuraba la plataforma para que esta estuviera en un estado en el cual esperaba una orden a través de la interfaz USB (como CDC para emular RS-232). Dichas órdenes ejecutarían acciones de nivel de aplicación del entorno MiWi™ que en concreto serían:

- Escaneo en búsqueda de redes.
- Unión a una red conocida.
- Creación de una red nueva.
- Escaneo de nivel de señal en la banda.
- Envío de un mensaje de tamaño igual al definido como "tamaño máximo de paquete" que está en el rango de los 128 Bytes.
- Capacidad para recibir mensajes por dicha interfaz inalámbrica e imprimirlos por pantalla enviando la información al PC de nuevo por USB.

En la figura 14 se observa el flujograma de dicho programa de prueba, donde se pueden ver 3 cajas blancas representando al *hardware* del PC, de un nodo externo que puede enviar por mensajes de forma inalámbrica por la interfaz MiWi^{TM} y de la plataforma B105CNBOARD. Las cajas en tonos verdes con reborde gris representan respuestas del B105CNBOARD a las flechas grises que son excitaciones en forma de caracteres recibidos por USB. La flecha roja represente un mensaje enviado de forma inalámbrica por interfaz MiWi^{TM} y que es recibido por B105CNBOARD. Por último la flecha azul representa información enviada por B105CNBOARD hacia el PC, por medio de USB CDC, y con información procedente de los resultados de la ejecución de las cajas verdes o con la información recibida por MiWi^{TM} a través de la flecha roja.

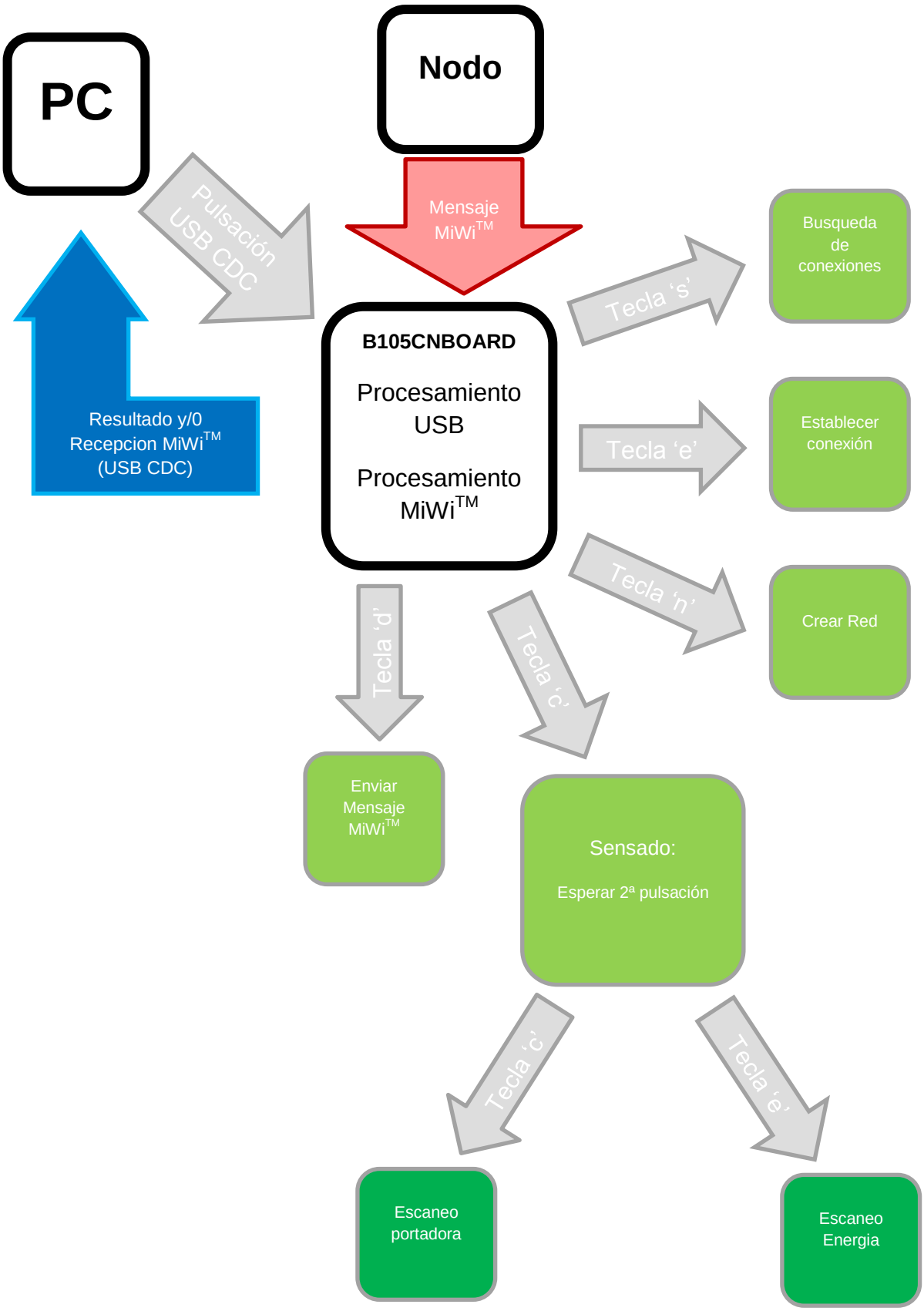
PC
Nodo
Pulsación USB CDC
Mensaje MiWi™
Resultado y/0 Recepcion MiWi™ (USB CDC)
B105CNBOARD
Procesamiento USB
Procesamiento MiWi™
Tecla 's'
Busqueda de conexiones
Tecla 'e'
Establecer conexión
Tecla 'n'
Crear Red
Tecla 'd'
Tecla 'c'
Enviar Mensaje MiWi™
Sensado:
Esperar 2ª pulsación
Tecla 'c'
Tecla 'e'
Escaneo portadora
Escaneo Energia

Fig. 14 Flujograma del programa de prueba

Por tanto, con la realización de esta prueba, se podía concluir que dichos módulos cumplían requisitos desde un punto de vista de aumentar las posibilidades del nodo como “nodo cognitivo” así como desde el punto de vista de aumentar sus posibilidades de comunicación y que, por tanto, eran aptos para ser adoptados en una posible futura versión del nodo.

4.4 Conclusiones

Se puede resumir este apartado notando que se lograron realizar pruebas de los nuevos interfaces requeridos por el nuevo diseño en la plataforma existente B105CNBOARD, además dichas pruebas proporcionaron resultados suficientemente satisfactorios.

Por tanto, si se recuerda la figura 11 en la que se mostraba el ciclo de diseño elegido, se puede concluir que se cumplieron los objetivos marcados como verde y que por tanto el proyecto estaría listo para entrar en su siguiente fase.

5. Diseño Software

Ya se puntualizó en el apartado 3 que los apartados 5, 6 y 7 constituían la tercera fase de este proyecto, en la cual, el objetivo es el diseño, implementación y demostración de una arquitectura para realizar estrategias cognitivas en una red de sensores.

La necesidad de diseñar una arquitectura *software* proviene del hecho de pretender que las redes cognitivas inalámbricas de sensores sean capaces de perseguir diferentes objetivos con la ejecución de sus ciclos cognitivos, lo que se ha definido al principio de esta memoria como estrategia cognitiva. Debido a la complejidad de dichas estrategias en las que, habitualmente, se tratan estímulos externos, respuestas, historia y otros factores propios de los ciclos cognitivos ya explicados, resulta apropiado abstraer esos agentes comunes a todas las estrategias, definir e implementar una arquitectura que los englobe y, habilitar de esa forma, la posterior implementación de diferentes estrategias de acuerdo a las necesidades de la red en unas circunstancias particulares.

Primero se abordará la etapa que constituye el diseño del software y que se subdivide a su vez en varios apartados a fin de, especificar los requisitos, la arquitectura base, la descripción de los sub-módulos, interconexión entre los mismos e interconexión entre nodos.

Para acotar esta discusión, en la figura 14 se puede ver una diagrama general presentado en el *Connectivity Brockerage*, y anteriormente en esta memoria, y que

muestra los sub-módulos que forman el módulo cognitivo (CRModule). Este CRModule deberá estar implementado en *software* y sus sub-módulos tendrán además una funcionalidad acotada y bien definida como veremos más adelante. Por tanto un nodo que sea programado con el CRModule, aparte de con su aplicación claro, estará listo para realizar procesos cognitivos y colaborativos fruto de una estrategia que persigue unos objetivos concretos y, en consecuencia, se le denominará *Cognitive Agent* (CAgent).

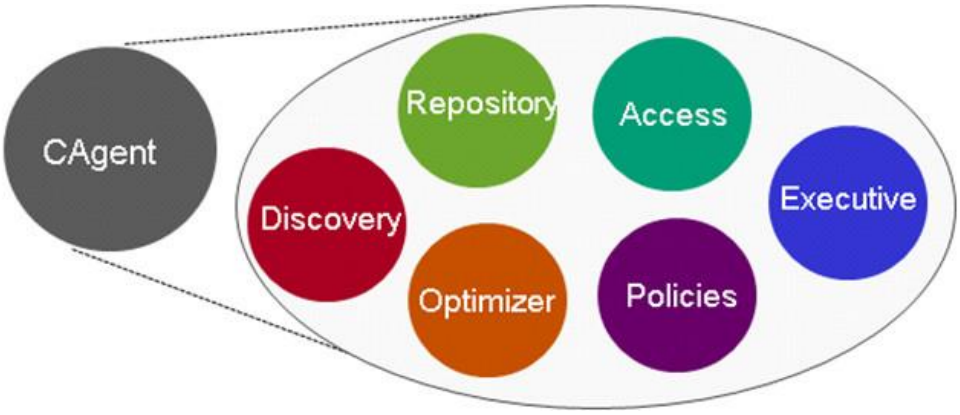

Fig. 16. Detalle de los módulos que forman el CRModule en un CAgent.

5.1 Requisitos

Antes de proceder al diseño de la arquitectura del *software* en sí misma es fundamental establecer unos requisitos que debe cumplir dicho diseño y que son descritos a continuación.

- **Fidelidad con lo presentado en el *Connectivity Brokerage***: puesto que algunos miembros del laboratorio en el que se ha realizado este proyecto han colaborado en el desarrollo del *Connectivity Brokerage* junto con otras universidades de especial relevancia en el sector de las *Cognitive Wireless Sensor Networks* (CWSNs) y por tanto poseen gran experiencia en el campo, unido a que en el laboratorio se está implementando un simulador de CWSNs basado en el mismo modelo y que ya se mencionó en el estado del arte en esta memoria, se consideró de especial relevancia que el diseño del *software* que se realizara fuera acorde con aquel propuesto en dicho artículo.
- **Escalabilidad**: sería deseable que de una forma sencilla se pudieran añadir más sub-módulos a la arquitectura en el futuro. No se puede descartar el hecho de que en el futuro se identifiquen nuevos procesos deseables para el CRModule y que se quieran separar en un nuevo sub-módulo. Por ello es

necesario que el diseño de la arquitectura permita realizar modificaciones con facilidad.

- **Testable y realizable**: se debe poder trasladar con facilidad a una plataforma física para poder comprobar su funcionamiento. Un ejemplo de que no sea fácilmente testable y realizable sería, por ejemplo, que el diseño, por su concepción, necesitará multitud de hebras de ejecución, o que cada sub-módulo tuviera su envío y recepción de mensajes como tal pensando en programación concurrente, etc. *Marcado porque no sé si hay que justificar como debería ser para que no fuera testable.
- **Transparente a la aplicación**: el objetivo de las estrategias cognitivas que se pueden realizar gracias al CRModule es mejorar diferentes aspectos de la comunicación (consumo, seguridad, etc.). Por tanto, un requisito es que la arquitectura diseñada no dependiera para nada de las acciones realizadas en el nivel de aplicación. Dicho de otro modo, en la ejecución de la aplicación, las acciones que debe tomar esta para su coexistencia con la estrategia cognitiva deben de ser prácticamente inexistentes.

5.2 Arquitectura base

Antes de diseñar los sub-módulos que componen el CRModule se consideró que se debía diseñar un modelo en el que acoger la arquitectura general del CRModule. Este modelo debía cumplir con los requisitos mencionados en el apartado anterior.

Por ello, siendo fiel a la arquitectura presentada en el Connectivity Brokerage y mostrada en la figura 16, en principio se deberían tener seis sub-módulos en el CRModule diseñado que serían Repository, Discorvery, Optimization, Execution, Access Control y Policy Support.

Para aumentar la escalabilidad y la manejabilidad de la arquitectura se tomó la decisión de incluir en el diseño del CRModule un nuevo sub-módulo denominado Messenger. Como se explicará más adelante, este nuevo sub-módulo representa una pasarela para el envío de peticiones entre el resto de sub-módulos.

Este sub-módulo Messenger, a su vez, homogeneiza las llamadas que se realizan entre el resto de sub-módulos y proporciona un sencillo *front-end* al programador para implementar esas llamadas añadiendo los parámetros deseados.

Por otra parte, para conseguir que la arquitectura fuera testable, había que habilitar el envío y recepción de mensajes entre sub-módulos pertenecientes a diferentes nodos. A este tipo de mensajes se les asigna el nombre de “mensajes de control” puesto que su propósito es el de controlar la correcta cooperación entre nodos acorde a lo que les dicte su proceso cognitivo. Para dicha tarea se decidió añadir a la estructura un sub-módulo llamado *Virtual Control Channel* (VCC). El VCC

realizará las tareas necesarias para interpretar correctamente los mensajes de control recibidos y de la misma forma se encargará de dar el formato adecuado a aquellos que quieran ser enviados.

Se pude ver el resultado de las modificaciones realizadas en la arquitectura en la figura 17 donde se observa el nuevo CRModule del que dispondrán los CAgents.

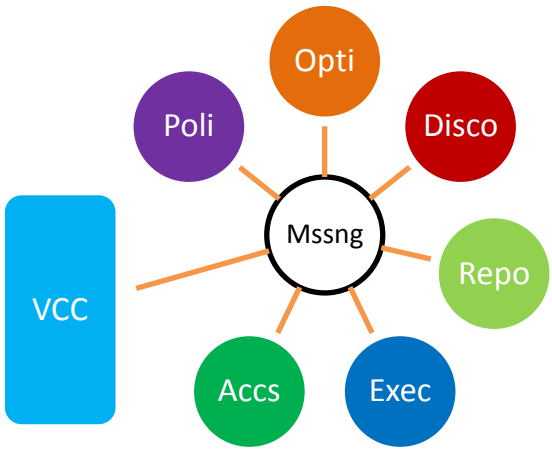

Fig. 17. Arquitectura del nuevo CRModule.

5.3 Descripción de los sub-módulos

En este apartado se realiza una descripción de cada uno de los módulos que componen el CRModule. El objetivo de la misma es detallar cual es la misión de cada uno de los sub-módulos que se han añadido a la arquitectura y de esta forma justificar su existencia.

5.3.1 Repository

El sub-módulo repositorio debe ser el encargado de almacenar información que resulte de interés para la estrategia cognitiva. Ya se mencionó en la introducción de esta memoria que el ciclo cognitivo, entre otras cosas, toma decisiones basadas en parámetros de su entorno y que “el sistema aprende del pasado (situaciones, planificaciones, decisiones, acciones) y utiliza su conocimiento para mejorar las decisiones en el futuro”. Este conocimiento debe estar almacenado en algún lugar y además debe tener algún gestor que conozca cómo debe almacenar la información y de donde tomar la información requerida. Esta es la labor del sub-módulo Repository dentro del CRModule, en el cual otros sub-módulos podrán solicitar información almacenada en él o guardar determinados datos.

5.3.2 Discovery

Es el encargado de caracterizar diferentes parámetros del entorno. El nodo dispondrá de diferentes elementos *hardware* con la capacidad de medir diferentes parámetros del entorno como, por ejemplo, los transceptores inalámbricos tienen la capacidad de medir los niveles de señal o de realizar escaneos en busca de nuevos nodos. Existe la posibilidad, igualmente, de añadir sensores al nodo con lo que, implementando sus funciones correspondientes en este sub-módulo, se aumentaría la capacidad de caracterización del medio por parte del nodo.

5.3.3 Optimization

El sub-módulo Optimization es el cerebro de la estrategia cognitiva, de hecho es el que coordina dicho proceso. Éste sub-módulo puede poseer algoritmos para realizar diferentes estrategias en base a diferentes criterios. Mientras este sub-módulo realiza el proceso cognitivo podrá realizar peticiones a otros sub-módulos del CRModule incluso a otros sub-módulos en otros nodos.

5.3.4 Execution

Execution es el encargado de ejecutar las medidas que decida el sub-módulo Optimization. Si durante el curso de la estrategia cognitiva se toma, por parte del sub-módulo Optimization, la decisión, por ejemplo, de poner el nodo a dormir, será el sub-módulo Execution el que termine ejecutando la instrucción correspondiente que duerma al MCU, al transceptor o a ambos. Y lo mismo se cumple para todas las acciones que se hayan considerado oportunas añadir a la implementación del CRModule. Execution hace las veces por tanto de un actuador.

5.3.5 Access Control

Debido a la naturaleza cooperativa de las estrategias cognitivas realizadas por los CAgents es fundamental contar con mecanismos de seguridad y control para tener constancia de que nodos tienen permisos para realizar qué acciones en otros nodos y, evitar así, que nodos foráneos a la red sean capaces de ejecutar acciones en contra de los intereses de la misma. El sub-módulo Access Control tiene información para relacionar a nodos conocidos con una tabla de permisos que permite o no realizar acciones a los mismos en los sub-módulos del CRModule al que pertenece el mencionado Access Control.

5.3.6 Policy Support

Las estretegias cognitivas que se llevan a cabo y las decisiones que se toman en base a sus resultados deben estar limitadas y determinadas por una serie de valores que tienen unos parámetros definidos y que componen lo que se denominan políticas. El sub-módulo Policy Support tiene información sobre dicho valores y por tanto será consultado por otros sub-módulos y principalmente por el sub-módulo Optimization. Este último, en gran parte, basará sus decisiones para realizar los

procesos adecuados a una estrategia cognitiva en los criterios “anotados” en Policy Support.

5.4 Interconexión de los sub-módulos

Como ya se ha mencionado en esta memoria los sub-módulos poseen, en general, una estrecha relación entre ellos. Esto es así puesto que la estructura diseñada en el Connectivity Brokerage tiene como fin organizar y clasificar tareas diferenciadas del proceso cognitivo pero que a su vez deben trabajar conjuntamente para realizar dicho proceso.

Cuando se hace referencia a interconexión entre sub-módulos lo que se está tratando es como se realizan las peticiones entre los mismos, dicho de otro de modo, como el sub-módulo Optimization solicita al sub-módulo Execution que realice una determinada acción sobre el nodo.

Durante el proceso de decisión sobre que diseño era el más adecuado para la arquitectura se tomaron en consideración las tres opciones que se identificaron y que son descritas a continuación. La implementación realizada será explicada con más detalle en el próximo apartado de esta memoria pero para justificar la decisión de diseño tomada es necesario repasar las diferentes alternativas de programación.

5.4.1 Mensajes por funciones específicas

Es la opción seguramente más sencilla y rápida en cuanto a tiempo de implementación. Consiste en que cada sub-módulo tenga definidas una serie de funciones que sean precisamente las que doten de la funcionalidad deseada al sub-módulo.

Cuando otro sub-módulo quiera o necesite de algunas de esas funciones lo único que tendrá que hacer será llamar o ejecutar dicha función.

A nivel de implementación, la forma más sencilla de hacer esto es añadir las funciones de un sub-módulo, que se quiera que puedan ser ejecutadas por otros, a su archivo de cabecera correspondiente y que, los archivos de cabecera de los otros sub-módulos, incluyan, a su vez, al archivo de cabecera del primero. De esa forma, se pueden hacer llamadas desde un sub-módulo a las funciones definidas en otros.

Como se ha dicho el principal beneficio de este método consiste en la facilidad de ser implementada su arquitectura pues solo habría que preocuparse de definir los módulos y sus funciones.

Como inconvenientes existe uno que es consecuencia directa de su ventaja. El código de algunos sub-módulos sería poco claro y poco limpio. Dentro del fichero

correspondiente a un sub-módulo se podrían encontrar, además de sus definiciones correspondientes, funciones de otros sub-módulos, lo que ya en un primer momento hace que realizar modificaciones en el código pueda convertirse en una tarea poco intuitiva. Además, conceptualmente, este modelo se aleja de la pretensión de que cada sub-módulo se represente de una forma completa puesto que al mirar el código, lo que se observaría, sería, en muchos casos, un popurrí de funciones de otros sub-módulos y por tanto una gran dependencia de unos respecto de otros.

5.4.2 Mensajes por memoria

Es probablemente la implementación que más estrictamente se acogía al concepto de los sub-módulos bien diferenciados mencionado anteriormente. Aquí lo que se contempló fue reservar una determinada cantidad de memoria para ser utilizada por los sub-módulos para comunicarse entre ellos mismos. Esa memoria podía ser compartida por todos o que cada uno tuviera su propio *buffer* reservado. Cuando un sub-módulo quisiera enviar una petición a otro escribiría en el sub-modulo correspondiente o, en caso de tener uno común se indicaría el destinatario en una cabecera del mensaje. Por otra parte los sub-módulos tendrían un proceso de comprobación de mensajes para saber si algún otro sub-módulo les hubiera enviado un mensaje.

Se tendrían, por tanto, una o varias colas FIFO para almacenar mensajes, y dimensionadas apropiadamente. Para las rutinas de comprobación de mensajes serían necesarios otras tantas rutinas que podrían ser disparadas cada cierto tiempo para comprobar si había algún mensaje disponible para los sub-módulos que representaban.

De cualquier forma, esta era una de las posibilidades para realizar este tipo de comunicación pero es fácil pensar en otras compatibles como pudiera ser que solo hubiera una rutina de comprobación de mensajes para todos los sub-módulos u otras similares.

Se representa en la siguiente figura un ciclo de envío de petición mediante una de las posibles implementaciones de este método.

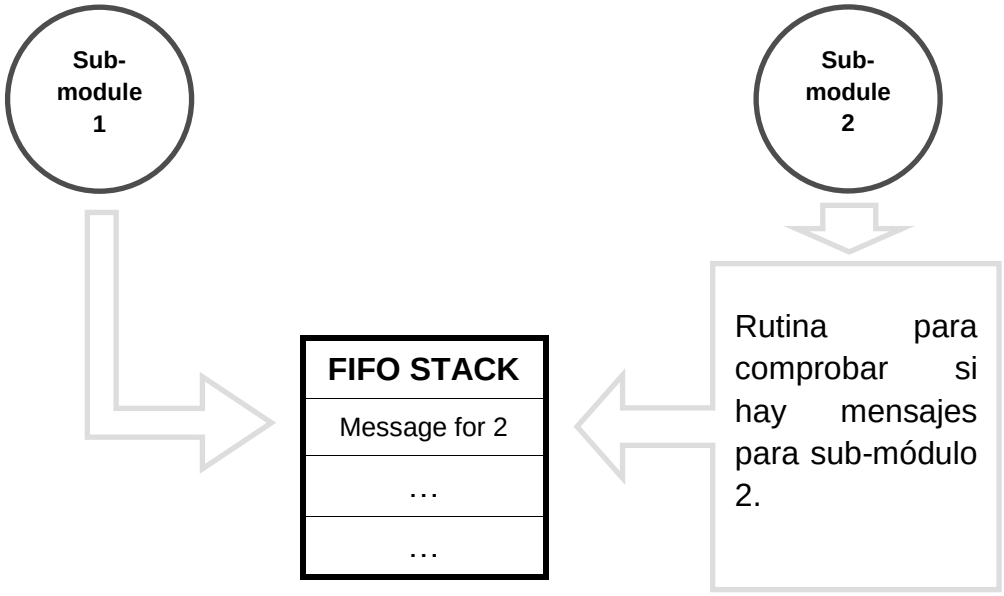

Fig. 18. Ciclo de envío de petición.

Una vez que el sub-módulo 2 recoge y procesa el mensaje que le ha enviado el 1 generaría un mensaje de respuesta para el sub-módulo 1 repitiendo el proceso mostrado en la figura 18.

El principal inconveniente de esta implementación es su complejidad, tanto para implementarla como para que un MCU la procese. Además, las latencias desde que un mensaje es enviado por un sub-módulo hasta que es procesado y una respuesta generada, podrían verse muy incrementadas. Se podría pensar, de igual forma, en problemas típicos de procesos con múltiples hebras en el que hubiera sub-módulos que estuvieran periodos de tiempo grandes sin poder comprobar si tenían mensajes.

Otro inconveniente en el caso de una sola cola FIFO sería que hubiera varios mensajes acumulados y el mero hecho de comprobar el destinatario de los mensajes fuera más costoso de lo deseado puesto que habría que analizar las cabeceras de los mensajes una por una.

La mayoría de estos problemas, si no todos, probablemente sean solucionables, sin embargo, es indudable que representan una dificultad extra y un coste computacional probablemente por encima de lo deseado, más aun teniendo en cuenta que el CRModule pretende dar soporte y mejorar aspectos de la comunicación de nodos que poseen un propósito determinado por su aplicación

principal, y por tanto, los recursos que tome la ejecución de la propia estrategia cognitiva deben de ser limitados.

5.4.3 Mensajes por función genérica

Esta es la opción elegida, si bien se puede ver como una híbrida entre las dos anteriores, está más cercana a la primera opción.

Antes de describir en profundidad el método elegido en el próximo punto, es importante notar que, en el diseño de esta implementación, se tuvo en cuenta principalmente dos factores:

- Por una parte se deseaba que, el hecho de enviar peticiones a otros sub-módulos, fuera perfectamente visible ojeando el código del programa, y no que estuviera repleto de funciones específicas de diferentes sub-módulos.
- Además se quería hacer a un coste computacional bajo.

Por tanto, la forma en la que se realizó la tarea debe ser vista como una adición de un nivel de abstracción respecto a la primera forma descrita en “Mensajes por funciones específicas”.

De esta forma lo que se buscaba era crear una entidad que se encargue de englobar y traducir peticiones de sub-módulos a funciones de otros.

Es así como se decidió la creación de otro sub-módulo que se llamaría Messenger y que se encargaría de hacer de pasarela en la comunicación entre sub-módulos.

5.4.3.1 Messenger

Como se ha mencionado el objetivo de este sub-módulo será el de conectar el resto de sub-módulos. Sin embargo todavía quedaban en el aire dos cuestiones a definir. La primera era si todas las peticiones entre sub-módulos deben tener el consentimiento del sub-módulo Access Control y la segunda es como gestionar las respuestas a peticiones. Pensar más alguna.

Respecto a comprobar los permisos en cada petición que realiza cada sub-módulo a otro se tomó la decisión de no hacerlo. Solo se comprobarán los permisos cuando las peticiones se realicen entre diferentes nodos. En contra de esta decisión juega la versatilidad del sistema puesto que se pueden imaginar supuestos en los que resultara interesante aislar sub-módulos de un nodo del resto de sub-módulos del mismo. Tales supuestos pueden ser, por ejemplo, que se quiera que un nodo pase a hacer las funciones de almacenar datos de nodos colindantes y que deje de guardar información del propio. Se podría, mediante Access Control, restringir el acceso del resto de sub-módulos del nodo al sub-módulo Repository. Luego, al no existir este tipo de permiso, se limita en cierto grado la funcionalidad del sistema.

De cualquier forma, se descartó controlar el acceso dentro de un mismo CAgent por una cuestión de no sobrecargar con tanto protocolo todas las peticiones, pues hay que recordar que uno de los requisitos de la arquitectura es controlar el coste computacional.

Otra cuestión es como los sub-módulos responden a peticiones que les han llegado de otros. Aquí existen dos posibles soluciones:

- Que en el sub-módulo destino se genere otra petición al sub-módulo origen que incluya la respuesta solicitada.
- Que la respuesta, generada en el sub-módulo destino, se incluya de alguna manera en la petición proveniente del sub-módulo origen.

La primera opción presenta dos problemas fundamentales. Por una parte se deduce que cada sub-módulo debería tener implementadas funciones para recoger las respuestas y almacenarlas donde correspondiera. Sería conveniente tener un registro de cuáles son las peticiones que se han realizado desde el origen para que cuando llegara la respuesta del sub-módulo destino se supiera que ha llegado lo que se ha pedido y que por tanto no se modifican variables por las que no se ha preguntado.

Por otra parte, con la primera posibilidad se hace más complejo delimitar el ciclo de ejecución de una petición. Según lo descrito, una posibilidad sería la descrita en la siguiente figura.

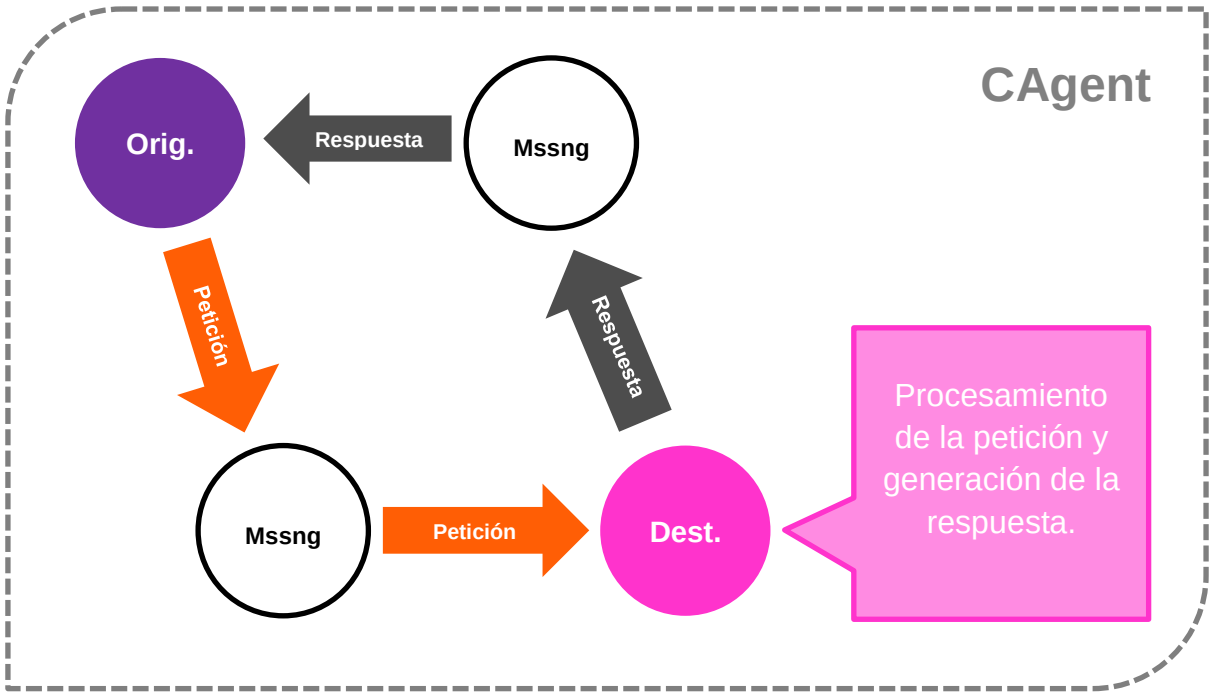

Fig. 19. Ciclo de ejecución de una petición.

Donde cada flecha seguro que implica una llamada a función por lo que como mínimo en esta forma se estarían anidando al menos cuatro llamadas a funciones, sin contar las que existan dentro de cada sub-módulo. Esto, a priori, no parece que represente una implementación muy eficiente puesto que realizar una tarea conlleva muchas llamadas a funciones, lo que requiere de recursos del procesador y, en concreto, memoria fundamentalmente.

La segunda posibilidad mencionada, que es la que se ha desarrollado, consiste en pasar como parámetro en la petición la variable o puntero donde se desee que el sub-módulo destino guarde su respuesta. Con lo que el ciclo de una petición resulta en lo expresado en la figura 20. Como se puede observar en este caso, el número de anidamientos de llamadas consecuencia de la arquitectura se reducirá al menos en dos. Así mismo, en el momento de programar el código, resultará más limpio y será más fácil de seguir porque las rutinas para realizar peticiones serán mucho más claras a consecuencia de lo mismo.

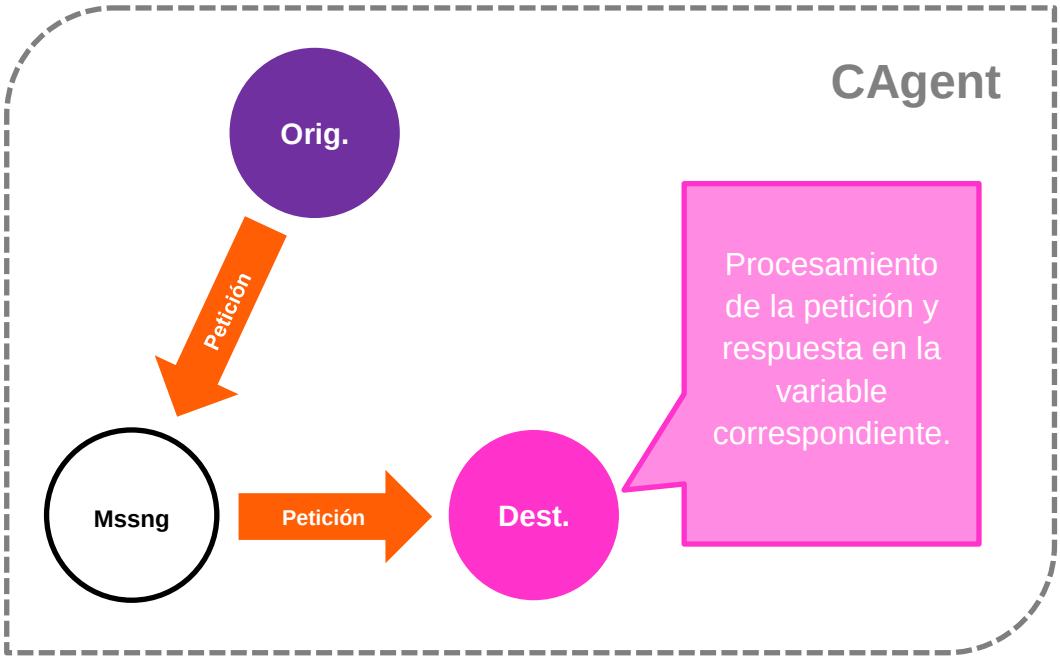

Fig. 20. Ciclo de petición adoptado para el diseño.

5.5 Interconexión entre nodos

Cuando se habla de interconexión entre nodos en este apartado se está haciendo referencia a la comunicación entre sub-módulos del CRModule de diferentes nodos *hardware* y por tanto no es objeto de este documento explicar cómo se realiza la comunicación ordinaria entre nodos para el intercambio de datos comunes.

En este tipo de comunicación entre sub-módulos se deben distinguir dos aspectos bien diferenciados de la misma. Uno es la especificación referente al canal real de comunicaciones y otro al canal lógico si es que lo hay. En el caso que concierne, lo primero se refiere a todo lo relativo al entorno de desarrollo MiWi^{TM} e interfaz radio utilizada y, lo segundo, a la utilización de un servicio que ya aparece en el *Connectivity Brokerage* denominado *Virtual Control Channel* (VCC) y cuya función es la de gestionar el envío y recepción de mensajes entre CAgents.

5.5.1 Canal Real

Esta comunicación ha de hacerse por alguno de los interfaces inalámbricos que posee el nodo y por tanto se realizará utilizando el entorno MiWi^{TM} pues en ese momento todos los interfaces disponibles ya se acogían a él. La arquitectura proporciona, por tanto, la posibilidad de mandar mensajes de control por tres interfaces diferenciadas fundamentalmente por su banda de trabajo y, por tanto, por sus parámetros asociados de alcance, tasa binaria, etc.

El canal real puede, o no, estar reservado a los mensajes de control, lo que significa que el mismo interfaz que se utilice para enviar estos mensajes puede ser que se utilice también como canal para enviar los datos que necesite la aplicación.

A modo de proporcionar una visión general del entorno de desarrollo MiWi^{TM} [17] se puede ver en la siguiente figura como está organizada en capas su estructura.

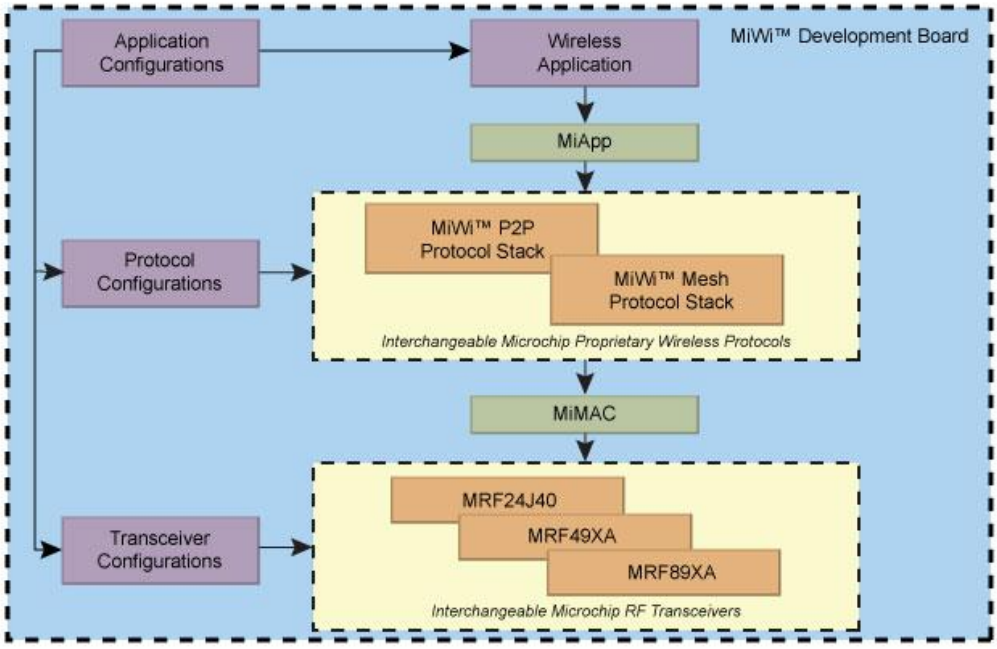

Fig. 19. Estructura del entorno de desarrollo de MiWi™.

Donde se observan los diferentes niveles del protocolo como el nivel de acceso al medio (MAC) con los transceptores, el de red y el de aplicación.

Otros detalles concernientes al canal real se tratarán más adelante en el apartado de implementación.

5.5.2 Canal lógico, el VCC

A parte del canal real de comunicaciones existe el denominado canal lógico. Éste debe ser entendido como una capa de abstracción más de la comunicación inalámbrica entre nodos, situada encima del entorno de desarrollo de MiWi™, pero que solo se utilizará para el envío de mensajes de control. Si, por tanto, se vuelve a mirar la figura 19, el canal lógico habría que situarlo en el cuadro llamado "*Wireless Application*".

Este canal lógico es lo que ya hemos mencionado con anterioridad bajo el nombre de *Virtual Control Channel* (VCC) y es un concepto que se presenta y describe en el *Connectivity Brokerage*.

Allí se presenta como una abstracción mediante la cual los CAgents pueden comunicarse sin barreras e ignorando detalles propios del canal real u otros que

conciernan al entorno o a la naturaleza de los otros nodos. Así mismo, se hace hincapié en la importancia de establecer y mantener este servicio

El VCC puede utilizar transceptores dedicados o por el contrario puede estar montado sobre el mismo canal por el que se envían los datos. Como se presentará en próximos puntos, en este proyecto se ha realizado un demostrador que utiliza el VCC sobre el mismo interfaz y canal utilizado para enviar datos de aplicación. La otra aproximación también es perfectamente válida, sin embargo, la presentada posee la ventaja de demostrar que, en caso de que se quisiera insertar la arquitectura software para realizar estrategias cognitivas en nodos con una sola interfaz inalámbrica, o que, por alguna razón, en nodos con varias estuviera temporalmente disponible solo una, dichas estrategias podrían ser llevadas a buen puerto.

La adopción de este servicio en la arquitectura creada ha sido mediante la inclusión de un nuevo sub-módulo llamado con el mismo nombre (VCC). Aunque debe estar claro que no es un sub-módulo como tal en tanto que no forma parte de los sub-módulos pertenecientes al CAgent, en esta arquitectura, a efectos de su interfaz con los sub-módulos del mismo, se comporta como si lo fuera.

Por ejemplo, si un sub-módulo desea enviar una petición a otro sub-módulo en otro nodo, el primero enviará la petición al sub-módulo VCC del CAgent donde se encuentra mediante el sub-módulo Messenger, igual que se haría para comunicar otros dos sub-módulos cualesquiera pertenecientes al mismo CAgent. El comportamiento en el nodo receptor del mensaje de control se detalla un poco más adelante puesto que es algo distinto.

A fin de visualizar la relevancia y utilidad del VCC en las comunicaciones se presenta la figura 19.

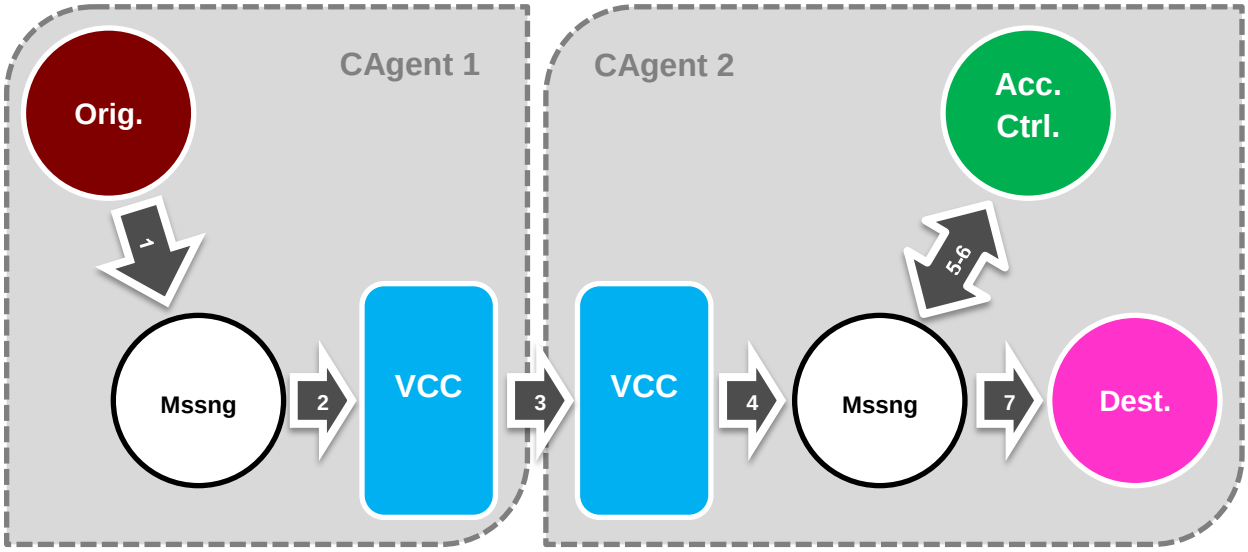

Fig. 19. Recorrido de un mensaje de control entre nodos.

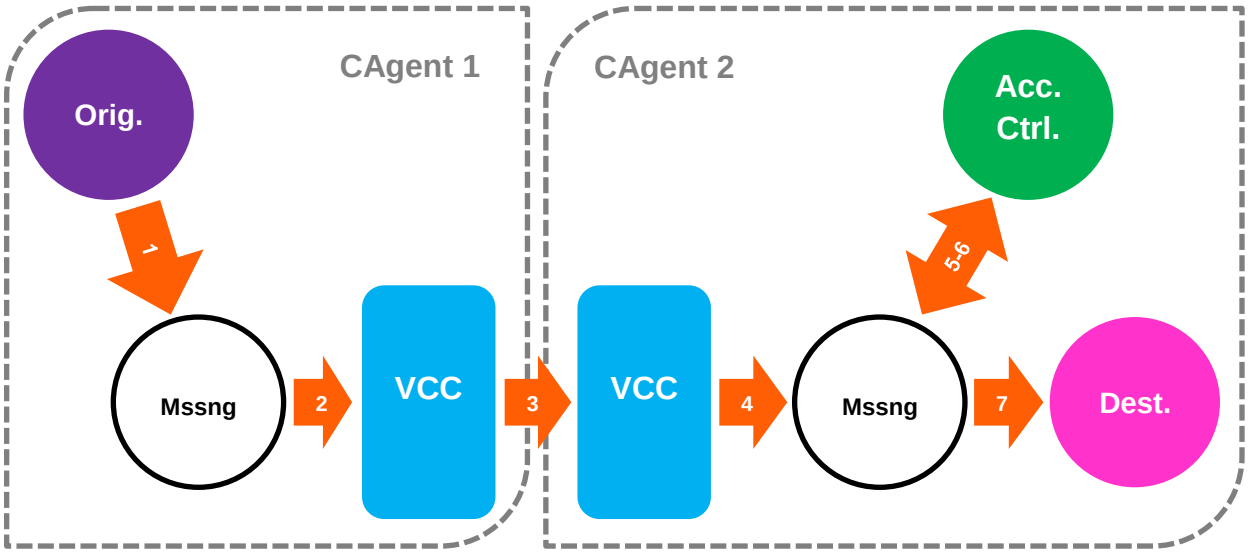

Fig. 19. Recorrido de un mensaje de control entre nodos.

En la figura 19 se observa que, el sub-módulo origen, alojado en el CAgent 1, envía su petición al sub-módulo VCC, perteneciente al mismo CAgent, a través del sub-módulo Messenger (flechas 1 y 2). Dicha petición constará de la información oportuna para que llegue al nodo destino, y en él, al sub-módulo correspondiente. Una vez que llega el mensaje al nodo destino y el VCC recibe el mensaje (flecha 3),

éste lo desgranará para obtener información sobre el sub-módulo al que va dirigido, creará una petición para él y lo enviará a través de Messenger (flecha 4). Messenger, cuando recibe una petición cuyo sub-módulo origen es VCC, automáticamente, requiere de la aprobación del sub-módulo Access Control, para lo que le consulta y recoge su respuesta (flechas 5 y 6). Si la respuesta es positiva, termina enviando la petición al sub-módulo destino (flecha 7).

Por tanto, como se había comentado anteriormente, si se presta atención a la forma en la que otros sub-módulos se comunican con el VCC y viceversa se observa que hay una gran analogía con la comunicación entre sub-módulos puros de los CAgents. La principal diferencia reside en que cuando el VCC realiza una petición a otro sub-módulo ésta siempre debe ser aprobada por el sub-módulo Access Control puesto que al ser VCC el que realiza la petición, es inequívoco que la petición original viene de un nodo externo y, por tanto, se requiere el permiso del control de acceso.

Con esto que queda definido el diseño y función del pseudo sub-módulo VCC.

5.6 Conclusiones

Se ha presentado, en este apartado, el diseño elegido para la posterior implementación de la arquitectura *software* para desarrollar estrategias cognitivas. Para ello se ha detallado la arquitectura general de lo que se ha llamado módulo cognitivo, se han descritos los sub-módulos que lo componen especificando sus funciones y por último se ha descrito como ocurren las peticiones e intercambios de información tanto entre sub-módulos pertenecientes a un mismo CAgent como entre sub-módulos que pertenecen a distintos CAgents.

6. Implementación del software

SOBRANTES

Respecto a medidas del programa de envío continuo

- Los paquetes van numerados con una cabecera que indica su índice.*
- El contenido de los paquetes son números crecientes en decimal.*
- Cada paquete tiene los números de 0 al tamaño de paquete menos uno.*
- En recepción se va rellenando un buffer principal del mismo tamaño que en el origen. Se hace mirando la cabecera del índice para colocarlo en el lugar adecuado.*
- Algunos resultados de este programa se muestran a continuación:

TX_BUFFER_SIZE (Bytes)	RX_BUFFER_SIZE (Bytes)	Tamaño Gran Buffer RX/TX (Bytes)	Número de paquetes enviados.	Tiempo de envío (segs)	Paquetes retransmitidos	Nivel de ruido RSSI
10	10	30.000	3.000	29,22	1-6	<H'10
44	44	30.000	681	7,39	0-2	<H'10

Debería poner algún resultado con el tamaño en 79 Bytes que es el máximo.*

Bibliography

[1] U.S. Department of Commerce, Office of Spectrum Management, United States Frequency Allocations, 2003.

[2] Electronic Communications Committee (ECC), The European Table of Frequency Allocations and Applications, 2013.

[3] Nest. Nest, The Learning Thermostat. [Online]. http://nest.com/

[4] Jr. Joseph Mitola III And Gerald Q. Maguire, "Cognitive Radio: Making Software Radios More Personal," 1999.

[5] Joseph Mitola III, "Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio," 2000.

[6] Luiz A. DaSilva, Allen B. MacKenzie Ryan W. Thomas, "Cognitive Networks," 2005.

[7] University of California e Intel Berkeley Research Lab. Largest Tiny Network Yet. [Online]. http://webs.cs.berkeley.edu/800demo/

[8] Berkeley. Epic: An Open Mote Platform for Application-Driven Design. [Online]. http://eecs.berkeley.edu/~prabal/projects/epic/

[9] Wikipedia. (2013) List of Wireless Sensor Nodes. [Online]. http://en.wikipedia.org/wiki/List_of_wireless_sensor_nodes

[10] IMEC, Digital Baseband Solutions for Cognitive Radio, 2011.

[11] NWCL (Next-generation and Wireless Communications Laboratory). Cognitive Radio Sensor Networks. [Online]. http://home.ku.edu.tr/~nwcl/crsn.html

[12] Balamuralidhar P. and Ramjee Prasad, A Context Driven Architecture for Cognitive Radio Nodes, 2008.

[13] Jan Rabaey et al., Connectivity Brokerage - Enabling Seamless Cooperation in Wireless Networks, 2010.

[14] Fernando Lopez Lara, Diseño e Implementación de un Nodo para Red de Sensores con Capacidades Cognitivas, 2011.

[15] AWD (Advanced Wireless Dynamics), Tulio RF Modem.

[16] Juan Domingo Rebollo, Diseño, Optimizaión y Prueba de un Nodo para una Red de Sensores Inalámbrica con Capacidades Cognitivas, 2013.

[17] Microchip. Microchip Personal Area Networks. [Online]. http://www.microchip.com/pagehandler/en-us/technology/personalareanetworks/home.html

[18] Elena Romero, Javier Blesa, Octavio Nieto-Taladriz Alvaro Araujo, Cognitive Wireless Sensor Networks Framework for Green Communications Design.

[19] Microchip, MRF24J40 Data Sheet, 2010.

[20] Microchip, PIC32MX Reference Manual, 2008.

[21] Texas Instruments, CC1110f32 Data Sheet, 2008.

[22] Microchip, MRF24WB0MA/MRF24WB0MB Data Sheet, 2013.

[23] Microchip, MRF49XA PICTail/PICTail Plus Daughter Board User's Guide, 2009.

[24] Wikipedia. (2011) Wireless Sensor Network. [Online]. http://en.wikipedia.org/wiki/Wireless_sensor_network

[25] Libelium. Libelium. [Online]. htttp://www.libelium.com/es/products/waspmote/

[26] Tatiana Bokareva. Mini Hardware Survey. [Online]. http://www.cse.unsw.edu.au/~sensar/hardware/hardware_survey.html

[27] Michael Johnson et al., "A Comparative Review of Wireless Sensor Network Mote Technologies," 2009.

[28] Microstrain. Wireless Nodes. [Online]. http://www.microstrain.com/wireless/sensors

[29] Wikipedia. Software-Defined radio. [Online]. http://en.wikipedia.org/wiki/Software-defined_radio

[30] Nuand. bladeRF. [Online]. http://nuand.com/

[31] Fereshteh Aalamifar, Design And Implementation of a Cognitive Wireless Sensor Network: Application To Enviroment Monitoring, 2011.