UNIVERSIDAD DE JAÉN

ESCUELA POLITÉCNICA SUPERIOR DE LINARES
DEPARTAMENTO DE INGENIERÍA DE TELECOMUNICACIÓN

TESIS DOCTORAL

ARQUITECTURA MULTI-AGENTE Y PROTOCOLO DE COMUNICACIÓN PARA DAR SOPORTE A SISTEMAS BASADOS EN CONOCIMIENTO EN REDES DE SENSORES INALÁMBRICOS

PRESENTADA POR:
JUAN CARLOS CUEVAS MARTÍNEZ

DIRIGIDA POR:
DR. D. JUAN CAÑADA BAGO
DR. D. MANUEL ÁNGEL GADEO MARTOS

JAÉN, 1 DE DICIEMBRE DE 2014

ISBN 978-84-8439-950-6

Para mi esposa, sufrida compañera en todo este periodo,

y mis padres, quienes acertadamente me

guiaron para llegar a lo que soy.

AGRADECIMIENTOS

A mis tutores, el Dr. D. Joaquín Cañada Bago y el Dr. D. Manuel Ángel Gadeo Martos por darme la oportunidad de trabajar con ellos en algo tan apasionante, y al Grupo de Investigación TIC220 de Ingeniería de Sistemas Telemáticos, por el apoyo prestado a mi investigación.

RESUMEN

El auge de las redes de sensores inalámbricos en estos últimos años ha propiciado la aparición de numerosas contribuciones científicas, así como una gran variedad de soluciones y aplicaciones. De entre éstas, gran parte de los esfuerzos investigadores han estado condicionados por la necesidad de conseguir un uso eficientemente de los recursos disponibles en un sensor, para poder así superar las limitaciones intrínsecas de estos: capacidad de proceso, memoria y disponibilidad de energía.

Dentro de las técnicas empleadas para conseguir la gestión eficiente de los recursos en una red de sensores, y más concretamente, la gestión autónoma de los recursos de un sensor, están las basadas en el conocimiento: redes neuronales, algoritmos genéticos, sistemas bio-inspirados, etc. Sin embargo, aunque éstas presentan características apropiadas para esa tarea, no existen contribuciones remarcables en esa área, donde los sistemas borrosos basados en reglas se presentan, a priori, como los más apropiados.

Por lo tanto, con la aplicación de los sistemas borrosos basados en reglas a la gestión autónoma de un sensor, se abre una nueva área de investigación, en la cual se encuadra la presente tesis. Este nuevo escenario necesita de nuevas arquitecturas internas para los sensores, apareciendo como una solución apropiada los sistemas multi-agente. Así mismo, se hace necesario un protocolo de aplicación que dé soporte a este tipo de sistemas, más aun, cuando en las redes de sensores no existe un protocolo apropiado para el envío de bases de conocimiento.

Así, teniendo en cuenta lo anterior, se plantea como objetivo general la definición de una arquitectura multi-agente, cuya finalidad sea conseguir la gestión autónoma de los recursos de un sensor a través de sistemas borrosos basados en reglas. Todo ello se apoyaría en la capacidad de comunicación que aportaría un nuevo protocolo de aplicación.

Por lo tanto, en esta tesis se presentan los trabajos de investigación que han conducido al desarrollo de una arquitectura multi-agente, donde los agentes son modelados a través de sistemas borrosos basados en reglas y cuya función es la de conseguir una gestión eficiente del ciclo de trabajo de un sensor. Así mismo, se han desarrollado nuevas estructuras de información y un protocolo de aplicación específico que da soporte a toda la red de sensores, haciéndola además, disponible desde Internet.

Los resultados obtenidos muestran que la integración propuesta cumple satisfactoriamente con los objetivos marcados. Se han realizado tanto pruebas en simulación, como en equipos reales, los cuales formaban parte de un test-bed diseñado específicamente para poder contrastar los trabajos de investigación desarrollados.

Así pues, se puede concluir que los sistemas borrosos basados en reglas permiten modelar efectivamente el comportamiento de un agente que gestione los recursos energéticos de un sensor, a la par que, gracias al empleo de un protocolo de aplicación específico, se permite su gestión dentro de una red de sensores.

ABSTRACT

The rise of wireless sensor networks in recent years has favoured numerous scientific contributions, as well as a variety of solutions and applications. Moreover, most of those research efforts have been conditioned by the need to ensure efficient use of available resources in a sensor, and to overcome the inherent limitations of these: processing power, memory and energy availability.

A wide variety of techniques have been applied in order to achieve an efficient management of the available resources in a sensor network. Among those techniques, computational intelligence ones like neural networks, genetic algorithms, or bio-inspired systems, appear as suitable solutions to achieve an autonomous management of the resources within a node. However, there are no remarkable contributions in that area, where the fuzzy rule-based systems can be presented as the most appropriate.

Therefore, the application of fuzzy rule-based systems to the autonomous management of a sensor opens a new research area, in which this thesis should be included. This scenario requires new internal architectures for sensors, where multi-agent systems seem to be an appropriate solution. Likewise, it is necessary an application protocol which supports this type of systems, even more when there is not any proper protocol for sending knowledge bases in wireless sensor networks.

So, taking into account the above-mentioned points, the main objective of this research work is the definition of a multi-agent architecture whose purpose is to achieve the autonomous management of the resources of a sensor through fuzzy rule-based systems. Moreover, those systems would rely on the communication services that bring a new application protocol.

Therefore, in this thesis, it is presented the research that has led to the development of a multi-agent architecture, where agents are modelled by fuzzy rule-based systems to achieve an efficient management of the duty cycle of a sensor. In addition, new information structures as well as a specific application protocol that supports the entire sensors network have been developed, making the network available from the Internet, too.

The obtained results show that the proposed integration satisfactorily meets the objectives. They have been tested both in simulations and real devices, which were part of a test-bed designed specifically to contrast the developed research work.

Thus, it can be concluded that fuzzy rule-based systems can effectively model the behaviour of an agent in charge of managing the energy resources of a sensor. Moreover, thanks to the use of a specific application protocol, those agents can be managed within a sensor network.

ÍNDICE GENERAL

CAPÍTULO I. PLANTEAMIENTO DE LA INVESTIGACIÓN Y REVISIÓN DE CONOCIMIENTOS. 1

1 CONTEXTO Y LOCALIZACIÓN DE LA INVESTIGACIÓN 1
1.1 INTRODUCCIÓN 1
1.2 JUSTIFICACIÓN 3
1.3 HIPÓTESIS Y OBJETIVOS 4
1.3.1 Hipótesis primera 4
1.3.2 Hipótesis segunda 4
1.3.3 Hipótesis tercera 4
1.4 OBJETIVOS GENERALES 4
1.4.1 Objetivo 1. Estudio de las plataformas de redes de sensores y soluciones de comunicaciones existentes para WSNs. 5
1.4.2 Objetivo 2. Diseño de una arquitectura multi-agente específica para WSNs. 5
1.4.3 Objetivo 3. Desarrollo de un protocolo de comunicación para la gestión de los sensores. 6
1.4.4 Objetivo 4. Desarrollo de módulos de auto-gestión interna para los sensores con técnicas basadas en conocimiento. 6
1.5 ORGANIZACIÓN/ESTRUCTURA DE LA TESIS 7

2 ESTADO DEL ARTE 8
2.1 REDES DE SENSORES 8
2.1.1 Limitaciones de las redes de sensores 9
2.1.2 Métricas de las redes de sensores 10
2.1.3 Estrategias de eficiencia energética en redes de sensores 12
2.1.4 Aplicaciones de las redes de sensores 17
2.1.5 Tecnologías y Plataformas más usadas en redes de sensores 20
2.1.6 Tendencias actuales en las redes de sensores 24
2.1.7 Carencias e incertidumbres de las redes de sensores 27
2.1.8 Utilización de test-beds en las redes de sensores 31
2.2 SISTEMAS MULTI-AGENTE EN REDES DE SENSORES 36
2.2.1 Teoría de Agentes 36
2.2.2 Estructuras de agentes 37
2.2.3 Arquitecturas multi-agente 40
2.2.4 Lenguajes de especificación y comunicación entre agentes 42
2.2.5 Aplicación de los sistemas multi-agente a las redes de sensores 43
2.3 LÓGICA BORROSA Y SU APLICACIÓN A LAS REDES DE SENSORES 49
2.3.1 Sistemas Borrosos Basados en Reglas 50
2.3.2 Aplicación de los FRBS a las redes de sensores 55

CAPÍTULO II. DESARROLLO Y METODOLOGÍA DE LA INVESTIGACIÓN 59

3 ARQUITECTURA MULTI-AGENTE 59
3.1 REQUISITOS Y JUSTIFICACIÓN 59
3.2 ARQUITECTURA MULTI-AGENTE DE UN SENSOR 60
3.3 ARQUITECTURA DE LOS AGENTES 62
3.4 COMUNICACIONES ENTRE AGENTES 64
3.5 AGENTE DE GESTIÓN 66
3.6 AGENTE DE CONTROL DE APLICACIÓN 67
3.7 AGENTE DE COMUNICACIONES 70
3.8 PROPIEDADES DE LA ARQUITECTURA MULTI-AGENTE 71
3.8.1 Escalabilidad 71
3.8.2 Replicación 71
3.8.3 Adaptabilidad 72

4 PROTOCOLO DE COMUNICACIÓN 73
4.1 INTRODUCCIÓN 73
4.2 ESTRUCTURA JERÁRQUICA DE RECURSOS 74
4.2.1 Formato de definición de recursos y parámetros 76
4.3 RECURSOS 77

4.3.1 Sistema (/system) 77
4.3.2 Red de sensores inalámbricos (/wsn) 78
4.3.3 Registro de actividad de la red (/wsn/log) 78
4.3.4 Registro de actividad de las sondas de un sensor (/wsn/log/<*identificador de sensor*>/probe) 79
4.3.5 Sensores (/wsn/sensors/<*identificador de sensor*>) 79
4.3.6 Tareas (/wsn/sensors/<*identificador de sensor*>/app/task) 80
4.3.7 Sondas (/wsn/sensors/<*identificador de sensor*>/app/probes) 80
4.3.8 Sistema borroso (/wsn/sensors/<*identificador de sensor*>/app /frbs) 81
4.4 PROTOCOLO DE COMUNICACIÓN 82
4.4.1 Servicios ofrecidos por el protocolo 82
4.4.2 Máquina de estados del protocolo 83
4.4.3 Formato de los mensajes del protocolo 85
4.4.4 Comandos del protocolo WISMAP 88
4.5 PASARELA IP DEL PROTOCOLO WISMAP 96
4.5.1 Mensajes del protocolo WISMAPi 97
4.5.2 Método GET 99
4.5.3 Método POST 99
4.5.4 Códigos de estado 100
4.5.5 Uso de la pasarela WISMAP. 100

5 APLICACIÓN DE SISTEMAS INTELIGENTES A LOS SENSORES 102
5.1 SISTEMA DIFERENCIAL 103
5.1.1 Funcionamiento 103
5.1.2 Método general de cálculo 104
5.1.3 Parámetros del sistema diferencial 105
5.1.4 Sistema diferencial para el control del tiempo de sueño de un sensor en una aplicación de monitorización de la presión sonora 106
5.1.5 Mecanismo de evaluación de la calidad del SD 108
5.2 SISTEMA FRBS PARA LA GESTIÓN AUTOMÁTICA DEL TIEMPO DE SUEÑO. 109
5.2.1 Formato para definición de Bases de Conocimiento 110
5.2.2 Descripción del formato XML^{KB} 112
5.2.3 DTD del formato XML^{KB} 114
5.2.4 Formato de datos para la transferencia de Bases de conocimiento 115
5.2.5 Descripción del sistema 118
5.2.6 Base de conocimiento 1 (KB1) 121
5.2.7 Base de Conocimiento 2 (KB2) 122

CAPÍTULO III. RESULTADOS 125

6 BANCO DE PRUEBAS (TEST-BED) 125
6.1 JUSTIFICACIÓN DE LA TECNOLOGÍA 126
6.2 ARQUITECTURA DEL TEST-BED 128
6.2.1 Aplicación web WISMAPWEB. 130
6.2.2 Aplicación wismaphost 131
6.2.3 Aplicación wismapsensor 134
6.3 PRUEBAS DEL TEST-BED 134
6.3.1 Prueba 1 - Lectura del valor de una sonda. 135
6.3.2 Prueba 2 - Comprobación de los valores leídos por una sonda. 136
6.3.3 Prueba 3 - Cambiar el tiempo de sueño por defecto de un sensor. 136
6.3.4 Escenario de pruebas. 137
6.3.5 Resultados de las pruebas. 137
6.4 CONCLUSIONES DE LAS PRUEBAS. 138
6.5 RED DE SENSORES DESPLEGADA EN LA E. P. S. DE LINARES 139

7 PRUEBAS REALIZADAS A LOS SENSORES CON ARQUITECTURA MULTI-AGENTE 141
7.1 PRUEBAS A LA APLICACIÓN WISMAPSENSOR 141
7.1.1 Descripción de las Pruebas. 142
7.1.2 Escenario de pruebas. 142
7.2 ESTABLECIMIENTO DEL VALOR BASE DE FIABILIDAD. 142
7.2.1 Cálculo del tiempo máximo de vida en función del tiempo de proceso. 143
7.2.2 Cálculo del tiempo máximo de vida con ciclos de sueño y proceso alternados. 144

7.2.3 Análisis de consumo con comunicaciones. 147
7.3 RESULTADOS DE LAS PRUEBAS. 149
7.4 CONCLUSIONES DE LAS PRUEBAS. 151

8 PRUEBAS DE DISTRIBUCIÓN DE BASES DE CONOCIMIENTO CON EL PROTOCOLO WISMAP. 152
8.1 PRUEBAS. 152
8.1.1 Condiciones de éxito y fallo 152
8.1.2 Posibles errores y/o fallos 153
8.1.3 Protocolo de pruebas 153
8.1.4 Escenarios de las pruebas 153
8.2 ANÁLISIS PRELIMINAR DE LOS RESULTADOS 155
8.3 ANÁLISIS ESTADÍSTICO DE LOS RESULTADOS 158
8.3.1 Justificación de la hipótesis de partida 159
8.3.2 Caso 1. ANOVA para el contraste de las medias en función de la base de conocimiento enviada. 160
8.3.3 Caso 2. ANOVA para el contraste de las medias en función del número de saltos del escenario de pruebas. 162
8.3.4 Conclusiones de las pruebas estadísticas 164
8.4 ANÁLISIS DE LA TASA DE ÉXITO 164
8.5 CONCLUSIONES DE LAS PRUEBAS 166

9 APLICACIÓN DE SISTEMAS INTELIGENTES A LOS SENSORES 167
9.1 PRUEBAS 168
9.1.1 Señales de entrada 169
9.1.2 Protocolo de pruebas 171
9.1.3 Medida del error 173
9.2 ANÁLISIS DE LOS RESULTADOS 174
9.2.1 Caso 1. ANOVA para la comparación de las medias de error en función del tipo de estimador empleado sin distinción por nivel de carga de la batería. 175
9.2.2 Caso 2. ANOVA para la comparación de las medias de error de los estimadores en conjunto en función del nivel de carga de la batería. 177
9.2.3 Caso 3. ANOVA para la comparación de las medias de error para cada estimador por separado en función del nivel de carga de la batería. 182
9.2.4 Análisis de las pruebas realizadas 183
9.2.5 Estudio del consumo de la batería 185
9.3 CONCLUSIONES DE LAS PRUEBAS 187

CAPÍTULO IV. CONCLUSIONES Y LÍNEAS DE FUTURO 189

10 CONCLUSIONES Y PRINCIPALES CONTRIBUCIONES 189
10.1 PUBLICACIONES 190
10.2 REVISTAS INCLUIDAS EN JCR 190
10.3 CONGRESOS INTERNACIONALES 190

11 LÍNEAS DE FUTURO 191

BIBLIOGRAFÍA 193

ÍNDICE DE FIGURAS

Figura 1. Control del flujo de información para tres tipos de arquitectura por capas para agentes. Fuente (Müller et al. 1995). 41

Figura 2. Taxonomía de la aplicación de los MAS a las redes de sensores. Fuente (Vinyals et al. 2011). 45

Figura 3. Clasificación de tecnologías de agentes en redes de sensores. 47

Figura 4. Ejemplo de controlador borroso. 51

Figura 5. Ejemplo de partición borrosa. 52

Figura 6. Conjunto borroso triangular. 53

Figura 7. Arquitectura multi-agente propuesta. 61

Figura 8. Arquitectura de un agente basado en BDI. 63

Figura 9. Ejemplo de una jerarquía de recursos. 76

Figura 10. Estructura jerárquica de recursos usada en las pruebas del *test-beb*. 78

Figura 11. Máquina de estados del protocolo WISMAP. 85

Figura 12. Formato de la cabecera del protocolo WISMAP. 86

Figura 13. Ejemplo de variable definida en con XML^{KB} y la representación gráfica de sus conjuntos borrosos. 113

Figura 14. Esquema general de un fichero XML^{KB}. 113

Figura 15. Formato raw de bases de conocimiento. 116

Figura 16. Los conjuntos borrosos para las tres variables de entrada y la variable de salida: a) Variable presión sonora, b) Variable de entrada de incremento, c) Variable de entrada batería y d) Variable de salida SMI. 120

Figura 17. Esquema del test-bed desarrollado. 129

Figura 18. Arquitectura software del test-bed. 130

Figura 19. Esquema simplificado de clases. 132

Figura 20. Interfaz de usuario de la aplicación wismaphost. 132

Figura 21. Secuencia de petición de una medida a un sensor. 133

Figura 22. Panel de sensores Sun SPOT del test-bed. 137

Figura 23. Esquema de la res de sensores instalada en la E. P. S. de Linares. 140

Figura 24. Sensor parte de la red desplegada en la E. P. S. de Linares. 140

Figura 25. Comparación entre el tiempo en sueño ligero y de operación. 145

Figura 26. Comparación entre el tiempo en sueño profundo y de operación. 146

Figura 27. Torre de protocolos en un sensor. 148

Figura 28. Ciclos máximos de envío para un sensor para diferentes potencias de transmisión. 149

Figura 29. Esquema de las redes de sensores desplegadas para las pruebas. 154

Figura 30. Ejemplos de redes con topologías equivalentes a los escenarios propuestos. 154

Figura 31. RTT medio para cada prueba. 157

Figura 32. Retardos en función de la base de conocimiento enviada. 157

Figura 33. Retardos en función del número de saltos. 158

Figura 34. Sensor Sun SPOT con micrófono. 169

Figura 35. Ejemplo de la primera señal de cada banco para cada tipo (A, B y C). 171

Figura 36. Intervalo de confianza (95%) de las medias de error de cada estimador para cada nivel de carga de batería para las señales tipo A. 179

Figura 37. Intervalo de confianza (95%) de las medias de error de cada estimador para cada nivel de carga de batería para las señales tipo B. 180

Figura 38. Intervalo de confianza (95%) de las medias de error de cada estimador para cada nivel de carga de batería para las señales tipo C. 181

Figura 39. Ejemplo de señal Tipo A con las SIR para cada estimador usado 186

Figura 40. Ejemplo de señal Tipo B con las SIR para cada estimador usado. 186

Figura 41. Ejemplo de señal Tipo C con las SIR para cada estimador usado. 187

ÍNDICE DE TABLAS

Tabla 1. Comparación de varias arquitecturas de sensores. 21

Tabla 2. Repaso de los paradigmas de Inteligencia Computacional aplicados a las redes de sensores. Fuente (Kulkarni et al. 2011). 25

Tabla 3. Bancos de prueba de redes de sensores. 35

Tabla 4. Parámetros del Agente de Gestión. 68

Tabla 5. Parámetros del Agente de Control de Aplicación. 69

Tabla 6. Parámetros del Agente de Comunicaciones. 71

Tabla 7. Tipos de sondas. 92

Tabla 8. Resumen de los comandos del protocolo WISMAP. 96

Tabla 9. Códigos de estado del protocolo WISMAPi. 100

Tabla 10. Regiones establecidas en el SD para la medición de presión sonora. 107

Tabla 11. Tamaño en bytes y % de ficheros XML^{KB}, comprimidos y en formato raw. 118

Tabla 12. Base de reglas para KB1. 122

Tabla 13. Base de reglas para KB2. 123

Tabla 14. Resultados de la Prueba 1 al test-bed. 138

Tabla 15. Resultados de la Prueba 2 al test-bed. 138

Tabla 16. Resultados de la Prueba 3 al test-bed. 138

Tabla 17. Tasa de éxito de las pruebas al test-bed. 139

Tabla 18. Duración de la batería para su uso de manera continuada en diferentes estados. 143

Tabla 19. Número máximo de ejecuciones en un Sun SPOT en función del tiempo de proceso y del régimen de trabajo. 144

Tabla 20. Consumo del chip CC2420 en función de la potencia de transmisión. 147

Tabla 21. Resultados de las pruebas de ejecución de la aplicación wismapsensor en un sensor aislado. 150

Tabla 22. Resultados de la prueba de fiabilidad. 150

Tabla 23. Experimentos de envío de bases de conocimiento. 153

Tabla 24. Detalles de las bases de conocimiento. 155

Tabla 25. RTT obtenido en el envío de las bases de conocimiento a través de la red de pruebas. 156

Tabla 26. Resultados del test de normalidad Shapiro-Wilk (p-valor) para el conjunto completo de muestras. 159

Tabla 27. Resultados del test de normalidad Shapiro-Wilk (p-valor) para 12 muestras por prueba. 159

Tabla 28. Resultados del ANOVA del RTT en función de la base de conocimiento enviada. 160

Tabla 29. Resultados del test de honestidad de Tukey para los RTT en función de la bases de conocimiento. 161

Tabla 30. Resultados del ANOVA del RTT en función de la base de conocimiento enviada para 12 valores de RTT por caso. 161

Tabla 31. Resultados del test de honestidad de Tukey para los RTT en función de la bases de conocimiento para 12 valores de RTT por caso. 161

Tabla 32. Resultados del contraste Kruskal-Wallis para los RTT en función de la bases de conocimiento. 162

Tabla 33. Resultados del ANOVA del RTT en función del número de saltos del escenario usado para el envío de las base de conocimiento. 162

Tabla 34. Resultados del test de honestidad de Tukey para los RTT en función del número de saltos. ... 163
Tabla 35. Resultados del ANOVA del RTT en función del número de saltos del escenario usado para el envío de las base de conocimiento. ... 163
Tabla 36. Resultados del test de honestidad de Tukey para los RTT en función del número de saltos. ... 164
Tabla 37. Resultados del contraste Kruskal-Wallis para los RTT en función de la bases de conocimiento. ... 164
Tabla 38. Tasa de éxito para el envío de las bases de conocimiento. ... 165
Tabla 39. Tasa de éxito para el envío de las bases de conocimiento sin el escenario de 7 saltos. ... 165
Tabla 40. Tipos de señales para las pruebas. ... 170
Tabla 41. Clasificación cualitativa de las señales para las pruebas. ... 171
Tabla 42. Media del error cuadrático medio obtenido con tiempos de reposo fijos. ... 173
Tabla 43. Valor medio del error cuadrático de las pruebas realizadas agrupadas por tipo de señal. ... 174
Tabla 44. Ciclos de ejecución para la estimación del SMI con el SD y FRBS con ambas bases de conocimiento. ... 174
Tabla 45. Resultados del error cuadrático medio en función del estimador de SMI usado. ... 176
Tabla 46. Resultados del test HSD de Tukey para el 95% de los intervalos de confianza para cada tipo de señal agrupados por el estimador SMI usado. ... 176
Tabla 47. Resultados del error cuadrático medio en función del estimador de SMI usado. ... 177
Tabla 48. Resultados del test HSD de Tukey para el 95% de los intervalos de confianza para cada tipo de señal agrupados por los niveles de batería usados en la estimación del SMI. ... 178
Tabla 49. Resultados de los ANOVA para cada estimador por separado en función del nivel de batería. ... 182
Tabla 50. Resultados del test HSD de Tukey para cada estimador y grupo de señales. ... 183
Tabla 51. Comparativa entre error y ciclos de trabajo. ... 184
Tabla 52. Consumo medio y tiempo de proceso de cada sistema evaluado. ... 185
Tabla 53. Resultados de los ANOVA para el consumo de energía y tiempo de proceso. ... 185

Capítulo I. PLANTEAMIENTO DE LA INVESTIGACIÓN Y REVISIÓN DE CONOCIMIENTOS.

En este capítulo se presentan, en primer lugar, los hechos que han motivado esta tesis así como las hipótesis y objetivos que se plantean conseguir. En segundo lugar se presenta una revisión el estado del arte que abarca las tres áreas fundamentales de esta tesis: redes de sensores inalámbricos, sistemas multi agente y sistemas borrosos basados en reglas.

1 CONTEXTO Y LOCALIZACIÓN DE LA INVESTIGACIÓN

En la presente sección se presenta una introducción al trabajo de investigación desarrollado con objeto de ubicar el contexto de su realización, su motivación, así como presentar las hipótesis y objetivos concretos que lo definen.

1.1 INTRODUCCIÓN

La investigación en el campo de las redes de sensores inalámbricos o WSNs (del inglés *Wireless Sensor Networks*) ha experimentado un importante crecimiento en los últimos años (Karl & Willig 2007). Esto ha propiciado un gran número de aportaciones en ámbitos muy diversos, tales como las radiocomunicaciones, encaminamiento, formación de clústeres, agregación de datos, control de topología, protocolos de comunicación, etc. Todas estas soluciones se apoyan en las características de estas redes, y de los nodos que las forman, para proporcionar diferentes soluciones a innumerables aplicaciones (Yick et al. 2008). Algunos ejemplos de los numerosos ámbitos en los que se han aplicado las WSNs son tan diversos como la salud, vigilancia y seguridad, bio-ingeniería, prevención, detección y control de fenómenos ambientales, automatización y control industrial, domótica, etc.

La gran diversidad de aplicación de las WSNs es posible gracias a la versatilidad de las mismas (Akyildiz & Vuran 2010), la cual les viene dada al estar formadas por dispositivos electrónicos autónomos, de reducido tamaño, con capacidad de proceso y de comunicarse inalámbricamente. Sin embargo, estas características, las cuales posibilitan la aplicación de las WSNs a tan amplia variedad de problemas, son las que suponen sus principales limitaciones (Buratti et al. 2009).

Dichas limitaciones vienen impuestas, en primer lugar, por el reducido tamaño de los nodos, que dependiendo de las tecnologías, puede ir desde unas decenas de centímetros, a apenas un par de ellos. Sin un reducido tamaño, los costes de fabricación, así como las aplicaciones a las que se podrían destinar, harían de las WSNs soluciones poco apropiadas para la mayoría de su ámbito de aplicación actual. Por lo tanto, las limitaciones derivadas de mantener un tamaño reducido obligan a establecer límites en la capacidad de cómputo, memoria, almacenamiento y comunicaciones.

Por otro lado, la segunda característica que es determinante en las limitaciones de las redes de sensores se debe a que éstas suelen estar formadas por nodos aislados.

Normalmente, los nodos que forman parte de una red de sensores, no tienen la posibilidad de estar conectados a fuentes de alimentación externas que aseguren su funcionamiento a lo largo del tiempo. De esta manera, lo habitual es que los nodos empleen fuentes de energía finita, tales como baterías internas o acopladas, por lo que la vida de un sensor está acotada en el tiempo, y por extensión, la de toda la red.

Así pues, aparejada a la limitación de tamaño, está la de una alimentación limitada en el tiempo, normalmente proveniente de baterías, cuya recarga o reemplazo, dependiendo de las aplicaciones, puede resultar inviable técnica o económicamente.

Es evidente que estas limitaciones han marcado la investigación sobre redes de sensores (Wang & Xiao 2006). Aún hoy siguen suponiendo un desafío, ya que se presupone que las limitaciones derivadas del pequeño tamaño y la alimentación finita de los nodos se mantendrán a lo largo del tiempo. Así, mientras más evolucione la micro-electrónica, menores serán los sensores y, previsiblemente, estos consumirán menos, pero a su vez, las baterías, aunque presenten mejoras tecnológicas, también deberán ser más pequeñas. Por lo tanto, la gestión de recursos de un sensor, y más concretamente los recursos más estrechamente vinculados al consumo energético, han sido, y seguirán siendo, parte importante de las investigaciones dentro de las WSNs.

Como más adelante podrá verse con detalle en la sección de Estado del Arte, la mayoría de las características de las redes de sensores están condicionadas por el consumo energético (Anastasi et al. 2009). Así pues, la necesaria gestión de recursos, y más concretamente el consumo energético, es un eje común para casi todas las investigaciones en redes de sensores, tales como la investigación en nuevas técnicas de encaminamiento de mensajes, formación de clústeres o agregación de datos. Sin embargo, la forma en que se ha tenido en cuenta el ahorro de recursos ha sido muy diferente dependiendo de la solución.

De entre las soluciones empleadas para conseguir una gestión eficiente de los recursos están las técnicas basadas en el conocimiento, tales como redes neuronales, lógica borrosa, algoritmos genéticos o sistemas bio-inspirados. Sin embargo, el grado de adecuación de cada una de ellas para las funciones típicas de una red de sensores varía sensiblemente de unas a otras (Kulkarni et al. 2011). Dentro de las técnicas antes enumeradas, los sistemas borrosos, y más concretamente los sistemas borrosos basados en reglas (Cordón et al. 2001) o FRBS[1], se presentan como una solución versátil y apropiada para su aplicación al control de recursos que, sin embargo, no han sido muy empleada para este fin en las redes de sensores.

Como ya se ha comentado, la gestión eficiente de los recursos se ha empleado a diversos ámbitos de las redes de sensores. No obstante, la gestión autónoma de un sensor bajo criterios de eficiencia energética supone un nuevo enfoque dentro de las redes de sensores. Dicha gestión autónoma se basa en que un sensor, en función de la tarea o aplicación a la que se ha destinado, administre sus recursos con objeto de alargar al máximo su vida útil.

Así pues, la gestión autónoma de un sensor requiere de la aplicación de nuevas técnicas a las redes de sensores, tales como los FRBS, así como de nuevas arquitecturas internas que posibiliten la integración de comportamientos complejos y del empleo de nuevos protocolos de comunicación que den soporte a estas técnicas y arquitecturas.

[1] Del inglés *Fuzzy Rule-Based Systems*

Dentro de las posibles arquitecturas internas de un sensor se ha estudiado que la asimilación de los sensores con agentes inteligentes aparece como una tendencia actual y factible, dadas las prestaciones que presentan algunas tecnologías de redes de sensores. Un agente inteligente es visto tradicionalmente como un ente capaz de obrar de manera autónoma y que, partiendo de un conocimiento previo, evalúa el entorno que le rodea, actuando en consecuencia para conseguir un fin concreto (Wooldridge2009). La complejidad que encierra el comportamiento de un agente ha propiciado la coexistencia de muchas teorías, diseños y definiciones, que van desde aquellos que intentan modelar el comportamiento humano (Fagin et al. 1995), a los que buscan, a través de comportamientos simples de muchos agentes en cooperación, un fin mucho más complejo (Stone & Veloso 2000).

Este último tipo de agentes antes mencionado, son un ejemplo de sistemas multi-agente. Su diseño está enfocado a la colaboración entre agentes, lo cuales pueden realizar tareas similares, o totalmente diferentes, con objeto de conseguir una tarea de mayor complejidad. La analogía con las redes de sensores es evidente (Rogers et al. 2009), presentándose como un modelo apropiado para su aplicación. Sin embargo, no son comunes estas integraciones ya que son pocos los trabajos que han buscado esta solución, más aún, cuando se buscan arquitecturas multi-agente internas a un sensor, cuyos agentes colaboren entre sí para conseguir acometer las tareas encomendadas a un sensor de manera energéticamente eficiente.

Las arquitecturas multi-agente internas a un sensor permiten la integración de comportamientos inteligentes de forma modular y escalable, aspectos importantes dentro de las redes de sensores. Pero la aplicación de sistemas multi-agente para la definición de arquitecturas internas de sensores, con la gestión autónoma del sensor como fin, tampoco cuenta con muchos ejemplos significativos en el panorama actual de las redes de sensores.

Por otro lado, la gestión autónoma de un sensor a través de sistemas inteligentes basados en FBRS, e integrados dentro de una arquitectura multi-agente, necesita de un nuevo protocolo de comunicación que permita el intercambio de la información propia de estos nuevos sistemas. Sin embargo, la investigación de protocolos de aplicación no cuenta con contribuciones remarcables en este ámbito (Srivastava2010), más aún, cuando se requiere dar soporte a nuevos sistemas inteligentes con estructuras internas complejas y variables.

Por último, se debe hacer notar que, entre las carencias que más adelante se presentarán de las redes de sensores, se encuentra la falta de pruebas en implementaciones reales, predominando los resultados obtenidos de simulaciones (Raman & Chebrolu 2008). Por lo tanto, todos los desarrollos que han conducido a refrendar las hipótesis del presente trabajo de investigación han sido probados en dispositivos reales dentro de un test-bed, el cual será descrito en la sección 6.

1.2 JUSTIFICACIÓN

Conforme a lo comentado en el apartado anterior, y con la motivación de partida de conseguir un nuevo sistema de gestión autónomo para un sensor, que sea capaz de hacer un uso eficiente de los recursos del mismo, la presente tesis se justifica por lo siguiente:

- La baja integración de los FRBS en las redes de sensores, y más concretamente en la gestión de los recursos internos, presumiéndose que los FRBS son apropiados para dicha tarea.

- La integración en un sensor de una arquitectura multi-agente para modelar los recursos internos del mismo. Se espera que las especiales características de los sistemas multi-agente permitan realizar una gestión inteligente de los recursos del propio sensor.
- La falta de protocolos de aplicación que permitan el soporte a los FRBS y sensores basados en arquitecturas multi-agente, además de otros procesos de intercambio de información, como la comunicación de resultados, actualización de parámetros de configuración, etc.
- Por el bajo número de experimentos y resultados que, dentro de las contribuciones sobre redes de sensores, se han obtenido de implementaciones sobre dispositivos reales.

Estas razones han llevado al inicio del presente trabajo de investigación, para lo cual se han postulado tres hipótesis, así como los objetivos a conseguir con el mismo. Tanto las hipótesis como los objetivos de esta tesis son detallados en el apartado siguiente.

1.3 HIPÓTESIS Y OBJETIVOS

El trabajo de investigación que se presenta en esta tesis, se fundamenta en la consecución de las siguientes hipótesis:

1.3.1 HIPÓTESIS PRIMERA

Si todo sensor o nodo, integrante de una red de sensores tiene unas reconocidas limitaciones en cuanto a capacidad de batería, proceso, memoria y comunicaciones (Karl & Willig 2007), esperamos que con una arquitectura multi-agente (Jennings & Wooldridge 2002) diseñada específicamente para estos dispositivos limitados, la gestión de sus recursos, así como la de la propia red, podrá ser más eficiente, escalable y replicable en otros nodos de la red.

1.3.2 HIPÓTESIS SEGUNDA

Si toda red de sensores tiene como principal función la comunicación de datos referentes al fenómeno bajo estudio entre los nodos que la integran (Akyildiz et al. 2002), y debido a que el mayor consumo, y por tanto factor limitante en cuanto al ciclo de vida de un sensor, es el envío y recepción de información (Anastasi et al. 2009), esperamos que diseñando un protocolo de comunicación específico se puedan soportar sistemas basados en conocimiento en una red de sensores de una manera eficiente.

1.3.3 HIPÓTESIS TERCERA

Si toda red de sensores está destinada a cumplir una función dentro de la monitorización, evaluación y/o modificación del entorno entre otras, y todo esto de una manera eficiente (Karl & Willig 2007), esperamos que aplicando, tanto sistemas analíticos basados en diferencias, como sistemas basados en conocimiento, todos los sensores integrantes de la red puedan gestionarse de manera autónoma o colaborativa, dinámica y eficientemente sin requerir una reprogramación de los mismos, ni un cambio en los protocolos de comunicación.

Por lo tanto, para refrendar lo expuesto en las hipótesis, se detallan en la siguiente sección, los objetivos concretos que guiarán el presente trabajo de investigación.

1.4 OBJETIVOS GENERALES

Los siguientes objetivos tienen como finalidad la confirmación de las tres hipótesis del presente trabajo de investigación. Atendiendo a la división del trabajo propuesta en el

proyecto de tesis y a las hipótesis presentadas, se han definido cuatro objetivos generales:

- Objetivo 1: estudio de los diferentes tipos y plataformas de redes de sensores.
- Objetivo 2: arquitectura multi-agente específica para redes de sensores
- Objetivo 3: protocolo de comunicaciones.
- Objetivo 4: sistemas basados en conocimiento a incorporar en los sensores para la auto-gestión inteligente del sensor.

Cada objetivo general viene seguido de los objetivos específicos, si los hubiere, que se pretenden conseguir adicionalmente.

1.4.1 OBJETIVO 1. ESTUDIO DE LAS PLATAFORMAS DE REDES DE SENSORES Y SOLUCIONES DE COMUNICACIONES EXISTENTES PARA WSNS.

Para un adecuado diseño del protocolo de comunicaciones, es necesario tener en cuenta las características más importantes de las redes de sensores, sus implicaciones en los protocolos de comunicaciones que las gestionan (Verdone & Buratti 2005), además de las más importantes tendencias en las aplicaciones de las redes de sensores inalámbricos (Buratti et al. 2009).

1.4.1.1 Objetivos específicos.

Como parte del estudio necesario y prueba de las WSNs se derivan los siguientes objetivos específicos:

a) Estudio y elección de la plataforma de red de sensores a usar.
b) Especificación de los requisitos del software que dará soporte a la arquitectura multi-agente y al protocolo de comunicaciones, poniendo especial hincapié en la modularidad y comunicación interna entre los agentes.

1.4.2 OBJETIVO 2. DISEÑO DE UNA ARQUITECTURA MULTI-AGENTE ESPECÍFICA PARA WSNS.

Para corroborar la primera hipótesis, se modelará una nueva arquitectura interna del sensor, la cual se basará en los sistemas multi-agente, potenciando la comunicación interna de los distintos agentes, así como la modularidad y escalabilidad de los sistemas internos del sensor y la difusión de los mismos por otros nodos de la red de sensores.

1.4.2.1 Objetivos específicos.

Para poder diseñar la arquitectura muli-agente específica se requerirá el cumplimiento de los siguientes objetivos específicos:

a) Diseño de los agentes internos: definición de funciones, módulos y lenguaje según las técnicas comunes de modelado de agentes, pero teniendo en cuenta las restricciones propias de las redes de sensores.
b) Diseño de un lenguaje uniforme de acceso a los recursos que permita acceder a cada uno de los agentes de la arquitectura y a cualquier subsistema de los mismos.
c) Implementación de la arquitectura muli-agente específica en la plataforma de WSNs elegida.

1.4.3 OBJETIVO 3. DESARROLLO DE UN PROTOCOLO DE COMUNICACIÓN PARA LA GESTIÓN DE LOS SENSORES.

Teniendo en cuenta que las tendencias actuales pasan por la necesidad de tener un modelo interno con varios planos o niveles (Akyildiz et al. 2002) y dado que la información y funcionalidades necesarias para la gestión son muy amplias (Ruiz-Garcia et al. 2009); se hace necesaria una arquitectura interna inicial que permita un control y gestión eficientes de los recursos o subsistemas internos del sensor. Por lo tanto, se necesitan definir las interacciones entre dichos subsistemas para acotar los recursos que el sensor tendrá disponibles a través del protocolo de gestión, así como la caracterización de estos.

Así pues, esta tesis tiene como tercer objetivo la definición de un protocolo de aplicación, necesario para el acceso y gestión de los sensores y sus recursos internos.

1.4.4 OBJETIVO 4. DESARROLLO DE MÓDULOS DE AUTO-GESTIÓN INTERNA PARA LOS SENSORES CON TÉCNICAS BASADAS EN CONOCIMIENTO.

El funcionamiento interno de los sensores aparece en muchas ocasiones guiado por diferentes tipos de algoritmos y protocolos (Karl & Willig 2007, Yick et al. 2008, Akyildiz et al. 2002), si bien el número de estas soluciones que se han basado en técnicas de inteligencia computacional, y concretamente aquellas basadas en lógica borrosa (Kulkarni et al. 2011), no son las más frecuentes. Además, la aplicación de inteligencia computacional se ha enfocado normalmente a procesos de interacción entre nodos, como el enrutamiento y agrupación de estos, y no a aquellos que se derivan en una autogestión de los propios sensores. Por lo tanto, el cuarto objetivo de esta tesis pretende diseñar e implementar subsistemas de control inteligente, basados en lógica borrosa, para el auto control y gestión de los sensores, con objeto de optimizar el uso de recursos limitados, como la batería.

1.4.4.1 Objetivos específicos.

Para la consecución de este objetivo se plantean los siguientes objetivos específicos:

a) Implementación en un sensor de aquellos módulos de gestión inteligente de recursos, basados en lógica borrosa, que se deriven de los estudios destinados a cumplir con el Objetivo 4.
b) Diseño, implementación y puesta en funcionamiento de un banco de pruebas para la red de sensores. Este banco de pruebas, o *test-bed*, tendrá como finalidad la realización de diferentes experimentos en una red de sensores que basará su funcionamiento en la arquitectura y protocolo derivados de la consecución de los objetivos generales previos.
c) Diseño de la aplicación de gestión de la red que centralice la información recibida de la red de sensores.
d) Diseño de una pasarela para el acceso a redes de sensores desde Internet que persiga la máxima compatibilidad con los protocolos actuales de acceso a recursos, tales como HTTP.

1.5 ORGANIZACIÓN/ESTRUCTURA DE LA TESIS

Una vez presentada una introducción a la tesis, así como su motivación, las hipótesis a demostrar y los objetivos a conseguir, se detalla a continuación la composición de los cuatro capítulos que la conforman.

Seguidamente, y como segunda parte del primer capítulo, *Planteamiento de la Investigación y Revisión de Conocimientos*, puede encontrarse el estado del arte que encuadra el presente trabajo de investigación. Dicha revisión de conocimientos se divide en tres partes diferenciadas: las redes de sensores, los sistemas multi-agente y los sistemas de lógica borrosa basados en reglas.

El Capítulo 2, *Desarrollo y Metodología de la Investigación*, se divide en tres grandes bloques, correspondiendo estos con las tres hipótesis de esta tesis doctoral. En primer lugar se detallarán los trabajos conducentes a refrendar la Hipótesis Primera, mostrando la arquitectura multi-agente desarrollada. A continuación se define el protocolo de comunicación que da cobertura a la Hipótesis Segunda, terminando el capítulo con la aplicación de sistemas inteligentes a las redes de sensores, donde se muestra cómo los sistemas borrosos basados en reglas pueden ser incorporados a la gestión interna de un sensor con una arquitectura interna multi-agente.

En el Capítulo 3 se muestran los resultados obtenidos en las pruebas realizadas a los distintos sistemas mostrados en el capítulo precedente. Este capítulo tiene como objetivo el de refrendar cada una de las hipótesis del presente trabajo de investigación, a través de un análisis detallado y pormenorizado de dichos resultados. Este análisis se centrará a su vez en el necesario cumplimiento de los objetivos marcados en la presente tesis. Así pues, en primer lugar aparece la descripción del banco de pruebas desarrollado para dar cobertura a los demás estudios. Las siguientes secciones contienen las pruebas realizadas a la estructura multi-agente, al protocolo de comunicación y a la aplicación de sistemas analíticos y de lógica borrosa para la gestión interna de un sensor.

Por último, en el Capítulo IV, se presentarán las conclusiones y líneas de futuro del presente trabajo, así como las contribuciones científicas fruto del mismo.

2 ESTADO DEL ARTE

En el presente trabajo de investigación se agrupan varias tecnologías, como son las redes de sensores inalámbricos, las técnicas basadas en conocimiento, como la lógica borrosa, y los sistemas multi-agente. Todas estas tecnologías son de muy amplio espectro y múltiples vertientes, por lo tanto, se intentará dar un enfoque integrador de todas ellas, dirigido a ubicar claramente el trabajo realizado, así como la justificación del mismo, optando siempre por remarcar aquello que sea de especial interés para esta tesis.

2.1 REDES DE SENSORES

Las redes de sensores inalámbricos o WSNs, están formadas por un número variable de dispositivos, comúnmente denominados sensores, nodos o "motas". Estos nodos se distribuyen normalmente en un área concreta, sin una topología fija, y usando habitalmente tan sólo el medio aéreo como medio de transmisión (Karl & Willig 2007).

Los sensores están formados por un sistema basado en microprocesador, por lo tanto tienen capacidad de proceso, memoria y almacenamiento, así como capacidad para comunicarse entre sí, normalmente de manera inalámbrica. Una característica común a las redes de sensores es que los nodos no suelen estar conectados a fuentes de alimentación externas, funcionado de manera autónoma con baterías, aunque esto depende de la aplicación, o el entorno de la red.

Otro aspecto a tener en cuenta, y motivo de su denominación como redes de sensores, viene dada porque los nodos suelen tener como principal tarea la de tomar medidas del entorno a través sondas tales como temperatura, humedad, iluminación, etc. Así pues, un nodo se dedicaría generalmente a procesar la información captada por dichas sondas y transmitirla, si fuera necesario, a algún nodo dentro de la red o a una estación base.

Funcionalmente, un nodo de una red de sensores, está comúnmente compuesto por tres unidades, la unidad de monitorización, la unidad de proceso y la unidad de comunicaciones o transceptora (Akyildiz & Vuran 2010), todas ellas alimentadas por una fuente de alimentación común y de capacidad limitada. Esta estructura es la que principalmente proporciona a los sensores su potencial y sus limitaciones.

Para comprender mejor cuales son los aspectos más importantes a tener en cuenta en las redes de sensores, el trabajo de Akyildiz y otros (Akyildiz et al. 2002) es una de las revisiones más reconocidas sobre WSNs vista en sus inicios. En el mencionado trabajo se presenta una revisión del estado del arte, así como los aspectos abiertos para el estudio, dejando claro que los principales aspectos para el diseño de redes de sensores son la tolerancia a fallos, la escalabilidad, las limitaciones del hardware, la topología de las red, el entorno, el medio de transmisión y el consumo de potencia. Teniendo en cuenta estos factores, en los siguientes apartados se comentarán la evolución que han sufrido los aspectos más relacionados con los objetivos del presente trabajo de investigación, tales como la gestión energética del sensor o el control del tiempo de ciclo. Para esto, se partirá de un análisis de las limitaciones de las redes de sensores y se continuará con una clasificación de las mismas en la que se describirán las características más importantes. Además, se comentarán las aplicaciones, tecnologías y tendencias más importantes de las redes de sensores.

Otra de las características habituales de los nodos, es su capacidad de modificación del entorno a través de actuadores, estas redes se denominan por algunos autores WSAN[2] (Akyildiz et al. 2002), aunque no son parte de los objetivos del presente trabajo, como tampoco lo son las redes de sensores sumergidas o subterráneas, las cuales tienen características muy específicas, pudiéndose ver una revisión de ellas en (Akyildiz & Vuran 2010).

2.1.1 LIMITACIONES DE LAS REDES DE SENSORES

Las WSNs se han convertido en una importante área de investigación dentro de las redes de comunicación y los sistemas distribuidos dado el notable avance en el hardware de soporte (Baronti et al. 2007). Esta evolución ha propiciado la reducción del tamaño de los nodos, a la par que se han incrementado las posibilidades en su aplicación, tanto por la mejora en las comunicaciones, como en el proceso interno. Sin embargo, desde los comienzos de su estudio (Heinzelman et al. 1999), se ha tenido muy en cuenta que los nodos son dispositivos con capacidades limitadas de proceso, consumo energético y comunicaciones, debido precisamente a que sus características principales son su autonomía y pequeño tamaño.

A su vez, teniendo en cuenta que el proceso de información y las comunicaciones inciden sobre la autonomía energética de los sensores, especialmente las comunicaciones (Lindsey & Raghavendra 2002), se han derivado gran número de trabajos en el estudio del consumo energético y su optimización en los sensores; algunos de estos trabajos aparecen recogidos en (Wang & Xiao 2006) y (Mahfoudh & Minet 2008). Es por eso, que la principal característica limitante de un nodo aislado es su suministro de energía, aunque no se deben descuidar el procesado de la información y las comunicaciones, además de otros aspectos funcionales como el despliegue y la diseminación de los nodos, el enrutamiento o la agregación de datos (*data aggregation*), orientando todos ellos a una gestión eficiente de recursos.

La limitación en el suministro de energía antes comentada viene derivada, principalmente, por el uso de baterías incorporadas al propio sensor como fuente primaria de alimentación. Esto proporciona al sensor un tiempo de operación finito, teniéndose estas baterías que recargarse tras el tiempo de operación si se quiere mantener éste en servicio. Sin embargo, en determinadas aplicaciones, la posibilidad de recarga es muy difícil, peligrosa, extremadamente costosa o totalmente imposible, como por ejemplo en aplicaciones bajo el agua (Akyildiz et al. 2005) o en volcanes en actividad (Werner-Allen et al. 2006), por lo que se hace necesario el uso de técnicas que hagan un uso eficiente de la limitada energía disponible.

La limitación del suministro energético no es la única (Akyildiz & Vuran 2010), si bien es crucial para entender la mayoría de las investigaciones sobre redes de sensores. Así pues, la necesidad de tener un tamaño reducido limita en gran medida las prestaciones disponibles, y está siempre en constante compromiso con las tecnologías de integración de dispositivos electrónicos, las de diseño de baterías, así como la integración de antenas y sondas. Por otro lado, están las propias limitaciones del despliegue de los sensores, generalmente aleatorio; y las inherentes a las comunicaciones, como son interferencias, desvanecimientos, acceso al medio, etc.

De esta manera, la investigación en el área de las redes de sensores tiene que llegar continuamente a compromisos y escenarios concretos, en los que las hipótesis

[2] Siglas de las palabras en inglés *Wireless Sensor and Actor Networks*.

propuestas se puedan definir de manera precisa para una serie de condiciones determinadas. Es por eso, que la gran mayoría de trabajos en redes de sensores siempre toman de inicio una serie de consideraciones concretas, tales como topología, distancia entre nodos, tipo de despliegue (determinista o aleatorio), movilidad, etc., todas ellas encaminadas a cubrir una aplicación o tipo de aplicaciones específicas.

2.1.2 MÉTRICAS DE LAS REDES DE SENSORES

En el apartado anterior se han comentado diversos aspectos de las redes de sensores, así como de los propios nodos. Así pues se hace necesaria una revisión sobre los parámetros y características más significativos de las WSNs para poder enmarcar adecuadamente el presente trabajo de investigación. También se debe tener en cuenta que la evolución de las redes de sensores está encuadrada en el periodo que va desde la última década del siglo XX, hasta la actualidad, lo que ha derivado en una ingente cantidad de trabajos, gracias en parte, a la facilidad de la comunicación de resultados científicos por el avance de Internet, así como a la amplia disponibilidad de dispositivos, dada la evolución de la microelectrónica y las comunicaciones. Por lo tanto, la revisión que se presenta en este apartado y los siguientes, se hace aún más necesaria, con objeto de aclarar los aspectos clave de las redes de sensores, así como su evolución y tendencias.

A lo largo de los últimos tiempos se han analizado las redes de sensores desde muchos puntos de vista. Una de las primeras taxonomías puede encontrarse en (Tilak et al. 2002), la cual se ha tenido en cuenta para guiar este apartado por su impacto en publicaciones sucesivas. El análisis del autor, sin ánimo de ser exhaustivo, puede usarse para guiar el estudio de los distintos aspectos a tener en cuenta en las redes de sensores. Del mencionado estudio se pueden destacar los parámetros de los sensores que el autor denomina métricas de rendimiento: **eficiencia energética**, **latencia**, **precisión**, **tolerancia a fallos** y **escalabilidad**, y que el autor refiere a protocolos de red para WSNs, aunque dichas características pueden ser extrapoladas a otros ámbitos de las redes de sensores. De entre éstas, la eficiencia energética está estrechamente ligada a todas las demás, al ser éstas dependientes del consumo energético, ya sea por tiempo de proceso, tiempo de operación o necesidad de comunicaciones. Con objeto de que sirvan de guía para los apartados posteriores, se presenta a continuación una breve revisión de estas métricas, y de cómo son influenciadas por los comportamientos internos de una red de sensores, poniendo especial énfasis en el efecto de cada métrica en el consumo energético.

- **Eficiencia energética:** dicha métrica se puede definir como el tiempo máximo de operación de un nodo para la aplicación a la que se destine la red de sensores. De esta manera, a mayor eficiencia, mayor es la vida del sistema. Como ya se ha mencionado esta métrica está afectada por todas las demás, al requerirse por cualquier elemento dentro del sensor alguna cantidad de energía. Así, una máxima permanente en la bibliografía es que en las redes de sensores inalámbricos, todo gasto tiene que estar justificado.
- **Latencia:** en las redes de sensores, la latencia puede definirse como el tiempo que transcurre entre la aparición de un evento del caso bajo estudio, y la recepción por parte del sistema destinatario de dicho evento, siempre y cuando esto sea necesario (Lu et al. 2004). Por lo tanto, esta latencia depende, en primer lugar, de la topología de la red, la cual determinará el número de nodos en la ruta y la potencia necesaria para llegar a cada uno de esos nodos. En segundo lugar, la latencia también estará marcada por el procesado en los nodos intermedios, ya

sea en forma de protocolos de enrutamiento, agregación de datos, seguridad, protección de la información, etc. Por último, otro factor que puede incidir negativamente en la latencia es la posible existencia, en la ruta que debe seguir un paquete, de nodos que se encuentran reactivándose tras periodos funcionando en modo de ahorro de energía (o sueño), ya sea a nivel MAC o de aplicación. Por lo tanto, a mayor número de nodos intermedios, mayor será la latencia, aun cuando el procesado en cada uno de ellos sea el mínimo imprescindible para el reenvío del paquete recibido. A su vez, el procesado en los nodos intermedios, así como los tiempos de reactivación de los que estuvieran en reposo, además de ser factores con una inherente variabilidad, podrán hacer aumentar significativamente la latencia, incluso en redes con pocos nodos.

- **Precisión:** esta característica no está tan enfocada a tener una medida precisa por parte de una sonda conectada un nodo, sino al hecho de que un sensor no suele estar el 100% de tiempo tomando medidas. Por lo tanto, la precisión vendrá determinada por los ciclos de sueño del sensor, la latencia y, por supuesto, la eficiencia energética. Los ciclos de sueño provocarán la pérdida de los eventos o medidas que se produzcan mientras el sensor esté en reposo. Así pues, la existencia de nodos en reposo lleva a una indefectible pérdida de precisión en procesos de estudio continuado del entorno. Por otro lado, la latencia en la transmisión de las medidas añade incertidumbre entre el momento de la detección del evento y la llegada a su destino, lo cual debe ser tenido en cuenta en aplicaciones que requieran de muestreo en tiempo real, debiéndose llegar a un compromiso entre la frecuencia de las medidas y los envíos. Además, datar una medida obliga a mantener algún sistema de sincronización de tiempo, lo que añade complejidad al conjunto de la red de sensores. Por último, conseguir sistemas con una alta eficiencia energética conlleva el uso de mecanismos o técnicas que necesiten un menor consumo de energía o a compromisos entre eficiencia y precisión. Estos compromisos pueden derivar en pérdidas de precisión al intentar aumentar la eficiencia con muestreos de menor frecuencia o a usar sondas o sistemas captadores con menor número de bits, y que por consiguiente necesiten menor tiempo de proceso.
- **Tolerancia a fallos:** fundamentalmente, como aparece reflejado en (Akyildiz et al. 2002), la tolerancia a fallos de una red de sensores viene marcada por la premisa de que el fallo de un nodo no debe dañar significativamente la tarea para la cual se ha desplegado dicha red. Así pues, esta métrica depende de la aplicación concreta a la que se haya destinado la red y de los requisitos establecidos para la misma. Al igual que todas las demás características, la pérdida de un nodo requiere un esfuerzo adicional en el consumo de recursos, ya que puede requerir una reorganización en la topología de la red, cambios en el encaminamiento de paquetes, así como aumento de potencias de transmisión y cómputos adicionales como la reorganización de grupos de sensores (clústeres).
- **Escalabilidad:** al igual que la tolerancia a fallos, la escalabilidad de una red está estrechamente ligada a la arquitectura de red elegida y a la aplicación a la que se destine. La escalabilidad puede ser crucial para alguna de estas aplicaciones, ya que el efecto del aumento del número de nodos, como se concluye en (Egea-Lopez et al. 2006), no es aún contemplado adecuadamente en muchos modelos de redes de sensores, quedando por tanto indeterminado el consumo adicional de recursos que puede suponer un aumento del número de nodos, sobre todo en las comunicaciones. Este problema es uno de los que normalmente intenta paliar las técnicas de **agregación de datos**, cuya influencia en el consumo de energía es

analizada ya en (Krishnamachari et al. 2002) y cuya finalidad es la de agrupar la información generada por los sensores que pertenecen a una ruta, o clúster, en un solo mensaje en su camino desde el nodo originario hasta su destino. Normalmente esto se acomete en forma de nuevos datos añadidos al mensaje, ya sean medidas absolutas, relativas o el resultado de la aplicación de funciones estadísticas, tales como media o la desviación típica de los datos agregados al mensaje.

Como se ha podido comprobar, las métricas descritas están estrechamente ligadas con el consumo energético, así como con la aplicación a la que se destine la red de sensores. Por eso, a continuación se revisan cuáles han sido las estrategias de eficiencia energética más empleadas dentro de las WSNs.

2.1.3 ESTRATEGIAS DE EFICIENCIA ENERGÉTICA EN REDES DE SENSORES

Del apartado anterior se puede concluir que la limitación del suministro energético en los nodos es determinante para el desarrollo de las redes de sensores y de nuevas técnicas, algoritmos y aplicaciones. Esto también queda patente en (Mahfoudh & Minet 2008), donde los autores hacen una clasificación de las estrategias más habituales en las redes de sensores para optimizar el uso energético. En la mencionada contribución, se plantean cuatro estrategias:

- Enrutamiento eficiente en energía.
- Planificación del tiempo de sueño en los sensores.
- Control de topología a través del afinado de la potencia de transmisión de los nodos
- Reducción del volumen de información transferida.

Estas cuatro estrategias han sido estudiadas y desarrolladas de diversa manera en los últimos años, y dado que la integración de algunas de ellas con los sistemas basados en conocimiento, son parte de las contribuciones del presente trabajo de investigación, se pasan a comentar con más detalle.

Las **técnicas de enrutamiento eficiente**, así como los protocolos derivados en las redes de sensores se enfrentan con requisitos distintos a los que se podrían aplicar a otros tipos de redes similares como por ejemplo las *Mobile Ad-hoc Networks* (MANET). De entre estas diferencias se podrían destacar el empleo de algoritmos de inundación o ruteado jerárquico (Watteyne et al. 2011), menos apropiados para las WSNs. Esto queda patente en los recientes esfuerzos del *Internet Engineering Task Force* (IETF) al crear el grupo de trabajo *Routing Over Low power and Lossy networks* (ROLL) para el estudio más concreto de las características de las redes de sensores inalámbricos. Entre estos requisitos (García Villalba et al. 2009)se encuentra la necesidad de algoritmos autónomos, eficientes energéticamente, escalables, robustos y resistentes antes caídas de nodos, que soporten una amplia heterogeneidad de dispositivos, así como que sean capaces de adaptarse a entornos con movilidad de los nodos.

Todas estas restricciones han llevado a un profundo y extenso estudio de nuevas técnicas de enrutamiento, algunas de las cuales aparecen en (Watteyne et al. 2011, Al-Karaki & Kamal 2004, Akkaya & Younis 2005), siendo en el último de estos trabajos donde, por su más reciente realización, se incluye una clasificación más clara. En esta última publicación se presenta un análisis cronológico que parte de los primeros algoritmos de inundación, pasando por los algoritmos de agrupación de nodos

(*clustering*), enrutamiento geográfico, y por último, protocolos de enrutamiento con sistemas de posicionamiento auto-organizados. Actualmente, el enrutamiento en sistemas de posicionamiento auto-organizados ha centrado el trabajo del grupo de trabajo ROLL del IETF, lo cual posibilita que se puedan llegar a soluciones que se conviertan en estándar dentro de las redes de sensores. El estudio pormenorizado de todos y cada uno de ellos queda fuera del propósito de este trabajo de investigación.

La **planificación del tiempo de sueño**, como técnica general de gestión energética se rige por una máxima evidente: *si algo no va a ser usado durante un tiempo es mejor que se apague o reduzca su régimen de funcionamiento*. Este principio básico puede aplicarse igualmente a las redes de sensores, y en concreto a los nodos, con el objetivo de aumentar el tiempo de vida de los mismos, y por tanto el de la red. Sin embargo, la planificación de los tiempos de inactividad, o sueño, no es trivial. Entre los problemas más importantes derivados de la inactividad de un nodo estarían, entre otros, los siguientes:

- Pérdida de cobertura de la red si se desconectan los interfaces radio
- Posibles fallos de detección de eventos, con el consiguiente aumento en la latencia de estos.
- Retrasos en la actualización y re-cálculo de las rutas.

Por lo tanto, han sido muchos los autores que han propuesto diferentes soluciones para el problema, las cuales se podrían dividir en dos grupos:

- Gestión del ciclo de sueño para minimizar el número de transmisiones manteniendo la cobertura de la red.
- Gestión del ciclo de sueño en aplicaciones de monitorización y detección de eventos de magnitudes externas para ahorro de energía.

Estos dos tipos aparecen denominados en (Anastasi et al. 2009) como control de topología y gestión de potencia, respectivamente. En ambos casos existen soluciones tanto centralizadas, como distribuidas según se comenta en (Mahfoudh & Minet 2008).

En el caso de la **gestión del tiempo de ciclo basada en el mantenimiento de cobertura**, la mayoría de las soluciones se apoyan principalmente en el funcionamiento por multiplexación por división en el tiempo (TDM) de la capa MAC IEEE 802.15.4 (IEEE2003) que concretamente usa CSMA-CA. Este protocolo, el más usado actualmente en la redes de área personal (PAN) y particularmente en muchas arquitecturas de redes de sensores, permite la planificación de los tiempos de espera y actividad del interfaz radio gracias a su modo de operación basado en balizamiento (*beacom-enabled*). Concretamente, cuando este modo está activo, se envía una baliza que sirve para alinear los tiempos de espera de transmisión (*backoff time*) de los demás miembros de la red. De esta manera, eligiendo cuando enviar estas balizas, se puede hacer una gestión ordenada de las comunicaciones, permitiendo que los nodos puedan desconectar su interfaz radio y entrar en modo de sueño entre periodos de espera.

Entre los trabajos pioneros que afrontaban la gestión del tiempo de ciclo (actividad frente a reposo) en las comunicaciones por radio de las redes de sensores, está el protocolo Sensor MAC (Wei Ye et al. 2002, Ye et al. 2004), conocido como S-MAC. El trabajo de Wei y otros comienza antes de que fuera lanzado el primer estándar IEEE 802.15.4 y enfatiza la necesidad de períodos de sueño entre fases de actividad de las comunicaciones para ahorrar energía. Esto último se consigue a costa, por supuesto, de

aumentar la latencia en la transmisión de paquetes por la red. En concreto, puede verse una comparativa de este tipo de soluciones en (Demirkol et al. 2006).

Hasta la aparición del estándar IEEE 802.15.4, aparecieron diversas propuestas de este tipo de protocolos de gestión de los ciclos de sueño en la capa MAC, como queda reflejado en el trabajo antes mencionado de Demirkol y otros, donde se pone de manifiesto la transcendencia del propio S-MAC, al aparecer varias propuestas de modificación a éste en esta misma publicación. Este tipo de gestión del tiempo de sueño pretende alargar la vida de los nodos no usando los interfaces radio de estos durante ciertos periodos de tiempo. La conectividad de la red se mantiene, en algunas propuestas, gracias al aumento del número de nodos, consiguiendo así una cobertura total, a la vez que se alarga la vida útil de la misma, a costa eso sí, de una mayor inversión en el número de sensores desplegados. Como no podría ser de otra forma, también se han propuesto soluciones que no consiguen una cobertura total en el área en la que se haya desplegado la red de sensores. Esta pérdida de cobertura se permite en pos de un mayor ahorro en el número de nodos y un menor consumo de energía, lo cual se consigue adaptándose al tipo de evento concreto que la red estudia (Cao et al. 2005, Shansi Ren et al. 2007, Balister et al. 2009) .

Otros de los esfuerzos llevados a cabo en el desarrollo de las redes de sensores es el **ahorro de energía a través del despliegue ordenado de los nodos**. En ese despliegue prima la eficiencia en las comunicaciones, así como en una posible auto-organización de los mismos durante el periodo de funcionamiento de la red. El objetivo que persiguen ambas técnicas es el de minimizar el número de sensores activos simultáneamente, como por ejemplo con el uso de algoritmos de planificación del tiempo de sueño de los sensores. Algunas de las soluciones pueden revisarse en (Anastasi et al. 2009, Mahfoudh & Minet 2008), si bien es conveniente aclarar que existen diferentes interpretaciones de lo que puede denominarse como control de topología, el cual es confundido a veces con el control de potencia de transmisión. Esto puede verse en (Anastasi et al. 2009), donde se define claramente cuáles son las funciones del control de topología.

Así pues, **el control de topología** persigue que, en redes de sensores con redundancia de nodos, se pueda encontrar una organización jerárquica que permita el ahorro máximo de energía a través del ajuste de la potencia de transmisión y de la actividad de los nodos, activando solamente los que sean estrictamente necesarios para desempeñar la función de la red. Una clasificación de las más reconocidas en la literatura es la presentada en (Santi2005), donde se muestra un esquema de tres niveles, distinguiendo en primera instancia entre los sistemas que usan una potencia de transmisión fija, y aquellos que usan esquemas de potencia de emisión variable. Los sistemas de potencia de transmisión variable, según el autor, se clasifican en aquellos que la ajustan en función de la localización (a través por ejemplo de receptores GPS), la dirección en la que se encuentran los nodos más cercanos o del conocimiento de los nodos vecinos, es decir, la topología se ajusta en función de cuantos nodos rodean al sensor y de con qué potencia es necesario transmitir para llegar hasta cada uno de ellos.

En (Zheng Gengzhong & Liu Qiumei 2010) se puede encontrar una revisión más actual de algunas de las técnicas de control de topología. Los autores plantean como conclusión que aún existen muchos problemas para medir el rendimiento real de las soluciones, así como de resultados aportados por experimentos reales, dado que abundan aquellos que se derivan de simulaciones o modelos informáticos, y no los

probados en redes de sensores físicas. Para esto, propone que las soluciones se orienten hacia aplicaciones concretas, aspecto que se ha tenido presente en la elaboración de esta tesis.

El principio de **reducción de la información** a transmitir es básico en las telecomunicaciones, por lo tanto, su aplicación a las redes de sensores, como aspecto a tener en cuenta, no necesita mayor justificación. Sin embargo, hay que tener en cuenta que la compresión en el canal está normalmente delimitada por la modulación establecida en un estándar, como sería el IEEE 803.15.4, Bluetooth o Wifi, los cuales definen tanto el nivel físico como el de enlace. Por lo tanto, los esfuerzos para la **reducción del volumen de información en las redes de sensores** se puede agrupar en las siguientes vertientes según (Anastasi et al. 2009):

- Predicción de datos.
- Compresión de datos.
- Procesamiento en red (conocido en inglés como *in-network processing*).

La **predicción de datos** en las redes de sensores se puede afrontar desde diversas perspectivas (Goel & Imielinski 2001), como por ejemplo:

- Estimar las medidas que podrán haber tomado ciertos sensores de la red a través de los valores leídos de otros nodos vecinos.
- No enviar actualizaciones de las medidas tomadas por un sensor mientras estas no superen un margen concreto sobre una predicción establecida por un nodo central o el propio sensor.

Esta predicción puede estar ligada a técnicas propias de las redes de sensores, tales como la correlación espacial de los sensores o el *clustering*, o bien depender del conocimiento que se tenga del fenómeno bajo estudio. Así pues, también existe la posibilidad, como se comenta en (Hongbo Jiang et al. 2011), de combinar la predicción sobre el fenómeno estudiado con parámetros típicos de una red de sensores. Por lo tanto, la correlación espacial de los nodos dentro de un clúster o la falta de variaciones significativas en las medidas, pueden evitar comunicaciones innecesarias, aumentando de esa manera la duración de la carga de la batería, alargándose así el tiempo de servicio de un nodo.

Otra de las técnicas mencionadas dentro de la reducción de información es la **compresión de datos**. La compresión requiere un procesado interno en el sensor antes del envío de la información, lo cual no involucra a toda la red. Además, como se pone de manifiesto en (Barr & Asanović 2006)hay que tener en cuenta el coste del propio proceso de compresión. En el trabajo antes mencionado se estima el coste del envío de un solo bit como 1000 veces superior al de la ejecución de una sola instrucción de 32 bits, que si bien permite un amplio margen en la aplicación de la compresión de datos en un sensor, el coste de la compresión podría llegar a ser significativo, o incluso superior, si el algoritmo de compresión, o la cantidad de datos, no son los apropiados. Por lo tanto, aunque los experimentos hechos por Barr y Asanović no sean directamente extrapolables a las redes de sensores, sí son indicativos del coste asociado a la compresión, motivo por lo cual se debe tener siempre presente este consumo.

En (Srisooksai et al. 2012) aparece una reciente clasificación, así como una revisión de los algoritmos de compresión usados en las redes de sensores, haciendo distinción entre dos tipos, la compresión de datos distribuida (compatible con la agregación de datos que

se verá a continuación), y la compresión de datos local, independiente de aspectos como el enrutamiento y la agregación de datos. Esta clasificación es relevante, en tanto en cuanto, la aplicación de cada una de ellas recae en topologías claramente diferenciadas de redes de sensores. Así, la compresión distribuida está pensada para entornos de una mayor densidad de sensores, donde el ahorro de energía no solamente proviene de la compresión de la información, sino de la manera en la que datos de nodos que cubren áreas que se solapan se agrupan [3], a costa evidentemente de perder precisión. La compresión local se aplicaría a redes donde la dispersión de los nodos es mayor, y la posibilidad de solapamiento de las áreas de medida es menor o inexistente. Ejemplos de ambos tipos de algoritmos pueden verse en (Srisooksai et al. 2012).

Por último se comentarán **las técnicas de procesamiento en red** y concretamente, su solución más importante: **la agregación de datos**. Para el correcto funcionamiento de estas técnicas se necesita involucrar a todas las capas por encima de la capa de enlace, desde el enrutamiento a la capa de aplicación, pasando por la agrupación de nodos (clústeres), o la elección de cómo y cuándo se obtienen los valores de las sondas y de si estos datos son susceptibles de ser enviados.

En particular, la agregación de datos pretende mitigar dos efectos adversos:

- La redundancia en la información captada por nodos próximos entre sí.
- El excesivo gasto de energía de los nodos que forman parte de las rutas de actualización de datos. Estos nodos, al estar más cercanos a los sumideros (*sinks*), tienen un gasto mucho mayor de energía al tener que encaminar los mensajes de los demás.

Además, la agregación de datos debe tener en cuenta factores (Akyildiz & Vuran 2010) tales como el almacenamiento temporal, representación y precisión de los datos, la técnica de agregación (la cual puede ser desde una simple media de correlación espacial) y por último, y crucial para el desempeño de toda la red, la elección de los puntos de agregación, los cuales definirán la ruta que deben seguir los datos captados por los nodos. Esta elección es vital para el buen aprovechamiento de la propia técnica, y puede ser altamente compleja debido a la naturaleza dinámica de las redes de sensores. Además, esta elección implica también la elección del tiempo de envío, que marcará la latencia en la actualización de la información, la respuesta ante pérdidas, duplicidades y errores de transmisión.

En (Rajagopalan & Varshney 2006) puede verse una revisión de algunos de los protocolos de agregación de datos más importantes en función de la arquitectura de red, como son, entre otros, LEACH (Heinzelman et al. 2002), basado en la agrupación de nodos en clústeres, o PEGASIS (Lindsey & Raghavendra 2002), basado en el descubrimiento del nodo más próximo y en generar el camino de menos distancia hasta el destino. La elección del camino más apropiado para la ruta de un mensaje hasta su destino ha sido también parte crucial en la investigación de las redes de sensores. Esto es debido a que, a pesar de que se obtenga el camino de mínima distancia a recorrer por un mensaje, ya sea por menor consumo de potencia, número de saltos o latencia, si todos los mensajes recorrieran esa ruta, esto derivaría en un trabajo excesivo para los nodos que la forman, agotando prematuramente sus baterías y provocando el

[3] Esta agrupación puede hacerse a través de muestreo alternado por parte de los sensores que abarcan una misma zona, o a través de calcular la media, u otro tipo de operación que incorpore en un solo valor los obtenidos por varios sensores.

aislamiento de parte de los nodos de la red. Este problema se analiza en (Minhas et al. 2009) donde se plantea un algoritmo que balancea la carga entre varias rutas usando lógica borrosa.

Como ya se ha mencionado, el procesamiento en red, y concretamente la agregación de datos, es una técnica totalmente compatible con la compresión de la información, ya que los resultados de la agregación de datos pueden ser comprimidos en los nodos intermedios. Sin embargo, hay que tener en cuenta que un esquema de compresión en los nodos intermedios implica procesos de descompresión, lo cual es un gasto adicional, que aunque como se puede ver en es significativamente menor que la compresión (Barr & Asanović 2006), se debe tener en cuenta en el proceso global.

Así pues, como ha quedado patente en el presente apartado, las formas en las que se ha afrontado la limitación de suministro energético de los sensores son muy variadas, no existiendo una solución únicamente aceptada, más aún cuando se pueden combinar entre sí.

Por lo tanto, en la presente tesis se ha buscado una solución que tomará como base algunas de estas técnicas, si bien se las ha incorporado el concepto de gestión autónoma, el cual se basa en la definición de comportamientos inteligentes en el sensor, que en función de la tarea que tenga encomendada, permitan gestionar sus recursos de manera eficiente, y así alargar el máximo posible su tiempo de vida.

2.1.4 APLICACIONES DE LAS REDES DE SENSORES

Las redes de sensores pueden presentar características muy diferentes en función de la aplicación a la que se destinan. Por otro lado, la concentración de nodos, la forma de su despliegue, el tipo y potencia de los mismos, la capacidad de ser recargados o no, la movilidad, el entorno, etc., afectan de manera directa al comportamiento de cada sensor, a la red y a su ciclo de vida. Esta es la razón de porqué existe tanta variedad de soluciones y de porqué éstas deben ser acotadas como redes ad-hoc, como se establece en (Akyildiz et al. 2002), una de las primeras y más importantes revisiones en conjunto de las redes de sensores, que a pesar del tiempo transcurrido, sigue siendo un análisis válido, aunque las soluciones hayan ido evolucionando en esta última década.

Por lo tanto, una red de sensores inalámbricos, normalmente, se diseña para cumplir con unos determinados requisitos, es decir, para tener una aplicación concreta. La versatilidad de las WSNs ha propiciado una gran variedad de aplicaciones a las que aplicarlas, y su clasificación ha sido sujeto de estudio dentro de las redes de sensores. Así, en Buratti y otros (Buratti et al. 2009) se establece una taxonomía de aplicaciones desde el punto de vista de la información, así como unos principios de diseño que deben seguir las redes de sensores. Esta clasificación atiende sobre todo a la forma en la que una red de sensores analiza el entorno y genera una respuesta, dividiendo éstas en **redes de detección de eventos** y **redes de proceso de estimación espacial**. Por supuesto, en la aplicación a distintos problemas, existirían redes que recaerían en ambas categorías, pero parece acertado diferenciar, a nivel de aplicación, entre aquellas que intentan conseguir una respuesta unificada para toda la red, como sería la intrusión en un perímetro controlado, y aquellas que intentan estimar la magnitud del fenómeno bajo estudio en la zona cubierta por la red.

Otro aspecto que es oportuno remarcar, en cuanto a las aplicaciones de las redes de sensores, es que la investigación en esta área normalmente elige una de las siguientes dos vertientes: la **investigación que intenta ser independiente de la aplicación**, y la

investigación que está orientada a aplicaciones concretas, tanto derivadas de necesidades de la industria, medicina, etc., como del mercado (Buratti et al. 2009). Este hecho ha sido destacado debido a que en parte, el presente trabajo de investigación aportará nuevas contribuciones en técnicas independientes de la aplicación, pero sin perder de vista la segunda vertiente, dando una aplicación concreta y funcional de los resultados de investigación.

Con respecto a la variedad y tipos de aplicaciones de las redes de sensores, uno de los trabajos que mejor ilustra las distintas soluciones empleadas en estas redes, es el trabajo de Yick y otros (Yick et al. 2008), en el que puede verse una interesante taxonomía de las aplicaciones sobre redes de sensores. En la mencionada clasificación, se diferencia en primera instancia entre aplicaciones de monitorización o seguimiento. Esta visión difiere con respecto al trabajo de Buratti, en la que se priorizaba el tipo de fenómeno bajo estudio. En el segundo nivel de la clasificación de Yick y otros pueden encontrarse, para ambos casos, las aplicaciones militares, para negocios, industriales/servicios, hábitat, y para la monitorización se añaden las de salud y control medioambiental. En el trabajo antes mencionado pueden verse varios ejemplos de todas ellas.

Como se ha podido apreciar, en todos estos ejemplos no aparecen **aplicaciones multimedia**, tales como la captura y procesado de imágenes, grabación y distribución de sonido en tiempo real, etc. Sin embargo, este tipo de aplicaciones sí representan un área importante del campo de investigación de las redes de sensores desde hace unos años. Esto se ha debido fundamentalmente a los avances en la integración de la tecnología CMOS, los cuales han posibilitado la integración en los sensores de cámaras y micrófonos de tamaño muy reducido y elevadas prestaciones (Akyildiz et al. 2007).

Como se ha comentado, este nuevo tipo de aplicaciones multimedia suele implicar la distribución de datos en tiempo real, y por lo tanto, se hace necesaria la aplicación de técnicas de calidad de servicio (QoS) sobre las redes de sensores. Las aplicaciones multimedia y la QoS serán comentadas, junto a las tendencias actuales, en el apartado 2.1.6.

Por lo tanto, según las clasificaciones antes comentadas, las posibles aplicaciones de las redes de sensores son muy diversas, destacaremos entre otras las siguientes, las cuales aparecen comentadas en (Karl & Willig 2007).

- Aplicaciones para la **detección de desastres naturales**. Sin duda éste es un tipo de aplicación típica para las redes de sensores (Werner-Allen et al. 2006, Hughes et al. 2006). En estas redes los sensores serían desplegados en la naturaleza, ya sea de manera ordenada o aleatoria, obteniendo su posición a través de sistemas de localización, como GPS, o con medidas relativas a otros sensores de la red. Su función es tratar de prevenir o detectar la aparición de desastres naturales gracias a la medida de parámetros ambientales, tales como la temperatura, la fuerza del viento, la presencia de gases, etc.
- **Control ambiental y mapeado de la biodiversidad**: entre este tipo de aplicaciones están aquellas destinadas a controlar la calidad del aire en zonas de actividad industrial (Khedo et al. 2010), la presencia de contaminantes en entornos marinos (De Marziani et al. 2011), o incluso a controlar movimientos de fauna en reservas naturales o ganado en explotaciones ganaderas (Yang et al. 2009, Handcock et al. 2009).

- **Edificios inteligentes**: aunque dentro de las aplicaciones para los edificios inteligentes existen soluciones específicas cableadas como KNX (KNX Association) o LonWorks (Echelon Corporation), éstas poseen pasarelas para gran variedad de dispositivos inalámbricos, lugar donde se podrían utilizar los sensores inalámbricos. Además, las redes de sensores pueden ser una solución en sí mismas, dada las amplias posibilidades de proceso y comunicación entre los nodos, como la racionalización de recursos energéticos en los edificios inteligentes (Zheng et al. 2010, Sandhu et al. 2004), lo cual puede acometerse de una manera poco intrusiva en edificios ya existentes, además de ser fácilmente escalable.
- **Gestión de recursos humanos** en grandes empresas y complejos: más allá de la gestión de recursos como la climatización en edificios inteligentes, las posibilidades que ofrecen las redes de sensores permitirían ampliar sus aplicaciones al control de presencia, con pequeños dispositivos presentes en cada trabajador, aplicaciones de seguridad (Wong et al. 2006), prevención de riesgos en el trabajo o detección de intrusiones.
- **Vigilancia de maquinaria y mantenimiento preventivo**: dentro de este tipo de aplicaciones entran aquellas destinadas a controlar, por ejemplo, vibraciones excesivas, sobrecalentamientos, presencia de ruidos que indiquen fallos, con la consiguiente posibilidad de evitar que la maquinaria se vea avocada a un sobreesfuerzo o a un fallo (Di Marco et al. 2010). La ventaja que aportan aquí las redes de sensores es que estas soluciones son inalámbricas y pueden ser aplicadas aun cuando la maquinaria no dispusiera de mecanismo alguno de control o vigilancia.
- **Agricultura de precisión**: la posibilidad de disponer de sensores desplegados en áreas de cultivo, ya sean explotaciones extensivas como intensivas (invernaderos), proporcionaría un control bastante preciso de las condiciones ambientales, tales como temperatura, humedad, así como la actuación sobre los sistemas de irrigación, abonado, fumigación, etc., todo ello con apenas un sensor por cada hectárea (Xuemei Li et al. 2008). Además, el control de estos parámetros puede ser usado también para la detección de condiciones que propician la aparición de plagas en los cultivos (Gadeo-Martos et al. 2011).
- **Medicina y cuidados médicos**: las redes de sensores, aparte de la posible controversia que puede suscitar su uso dentro del cuerpo humano, son de gran utilidad, principalmente por su pequeño tamaño, autonomía y carácter inalámbrico, lo cual posibilita aplicaciones como la vigilancia de personas mayores o con discapacidades psíquicas como el alzhéimer (Huo et al. 2009) o la monitorización de constantes vitales (Corchado et al. 2010).
- **Logística**: las aplicaciones para logística llevan tiempo usando sistemas pasivos, tales como los códigos de barras o dispositivos RFID, en embalajes o etiquetados. Si bien la aplicación de redes de sensores para este tipo de aplicaciones aumentaría el coste de algunas soluciones, estas redes aportarían nuevas ventajas a la logística, como el control en tiempo real de las condiciones de las mercancías (temperatura, presión, vibración, etc.), vigilancia de las mismas o posicionamiento. Además, la capacidad de comunicación bidireccional de los sensores posibilita la automatización de aplicaciones tales como la trazabilidad de productos alimenticios, control de niveles, alarmas, etc.
- **Monitorización distribuida**: serían aquellas que de manera distribuida permiten la monitorización y gestión de procesos gracias a las redes de sensores, y que normalmente estarían interconectadas con otras redes, ya sean de sensores, o de

otro tipo como Internet. Así, dentro de este tipo de aplicaciones estarían aquellas que, a través de sensores desplegados en las calles, medirían el tráfico rodado de una ciudad, posibilitando la gestión del mismo a través de la comunicación a los conductores de zonas para aparcar (Chinrungrueng et al. 2007) y rutas alternativas o de zonas altamente congestionadas (Wenjie et al. 2005).

A pesar de todos estos ejemplos, las aplicaciones para las redes de sensores son innumerables, creciendo día a día, y abriendo nuevos campos de aplicación conforme evolucionan las investigaciones sobre ellas.

Si bien la variedad y potencial de aplicación de las redes de sensores es muy amplia, sus limitaciones se tienen que tener siempre presentes, fundamentalmente las derivadas de su fuente de energía, proporcionada normalmente por baterías de pequeño tamaño. Más aún, cuando en la mayoría de los casos no pueden ser reemplazadas, o el coste, o riesgo de este reemplazo, lo hace inviable.

La capacidad de alimentación y el tiempo de operación de un sensor son cruciales a la hora de diseñar una nueva aplicación, es por esto que la eficiencia energética es uno de los aspectos que más aparece en los trabajos de investigación sobre redes de sensores.

2.1.5 TECNOLOGÍAS Y PLATAFORMAS MÁS USADAS EN REDES DE SENSORES

En este apartado se presenta una revisión de algunas de las tecnologías y plataformas de redes de sensores más representativas usadas hasta el momento. En particular, se pretende remarcar el hecho de que el presente trabajo de investigación persigue introducir mejoras concretas en el funcionamiento y arquitectura funcional de los sensores que aún no se han incorporado a éstos como características básicas. Las mejoras que se propondrán tienen como objeto el conseguir una gestión más eficiente de los recursos en el sensor aumentando la vida útil de cada nodo, y por extensión, de toda la red. Todo esto siempre con la premisa de la independencia de la plataforma o familia de sensores usadas. Así pues, a continuación se presenta una revisión del hardware y de los sistemas operativos más importantes empleados en las redes de sensores.

Las redes de sensores, como ya se ha comentado, deben en parte su auge a la mejora constante en la escala de integración de los circuitos electrónicos, lo que ha permitido la reducción de tamaño de microprocesadores, memorias y todo tipo de circuitos de comunicación. Siendo esa reducción de tamaño la que potencia la versatilidad y capacidad de aplicación de las redes de sensores. Por lo tanto, la evolución de una tecnología, como es la microelectrónica, ha posibilitado la aparición de otra área tecnológica, la cual tiene sus propias características y aplicaciones. Con respecto a lo que se puede entender como una red de sensores, se ha dado una definición al inicio de este capítulo, la cual es compartida por la gran mayoría de autores, salvo mínimos matices. Al igual, existe una gran coincidencia entre los autores en cuanto a los elementos que deben componer un nodo de una red de sensores. Así pues, cada nodo (Karl & Willig 2007, Akyildiz et al. 2002) deberá disponer de una unidad de proceso (microprocesador o micro-controlador), memoria, volátil y/o no volátil, sondas y/o actuadores, capacidad de comunicación inalámbrica y una fuente de energía, normalmente autónoma, en forma de batería. A pesar de que estos requerimientos son básicos, y en realidad, la mayoría de ellos están basados en estándares o patentes, para las redes de sensores, y más en concreto para los nodos que la forman, no existe un estándar o plataforma común que se pueda llamar como tal. Esto se debe en parte, como

se ha visto en el apartado anterior, a la gran variedad de aplicaciones a las que las redes de sensores pueden ser destinadas, haciendo que las necesidades del hardware varíen, así como las necesidades de comunicación, medición de las condiciones externas (temperatura, vibraciones, iluminación, CO2, etc.), y por supuesto de las necesidades de consumo de energía. Aparte de las diferentes soluciones hardware que han aparecido en los años de evolución de las redes de sensores, en algunos casos los sensores usados eran mini ordenadores de decenas de centímetros como los del test-bed Orbit (Raychaudhuri et al. 2005), los cuales distan mucho de los escasos cinco centímetros, incluso menos, de las familias de sensores que se verán a continuación.

Las familias de dispositivos que han ido apareciendo en estos últimos años, ya sea en la industria, de manera comercial o en entornos académicos, suelen tener ciertos aspectos comunes:

- Plataforma hardware con amplias posibilidades de configuración. Generalmente es posible añadir dispositivos de comunicaciones inalámbricas de diferentes tecnologías (IEEE 802.15.4, WIFI o Bluetooth), conexiones cableadas como USB o Ethernet, ampliar el número y tipo de las sondas o actuadores, y la posibilidad de conectar dispositivos auxiliares como receptores GPS.
- Sistema operativo abierto: Para facilitar los trabajos de investigación casi todas las plataformas aportan toda la codificación del sistema operativo que gestiona cada sensor.
- Amplia información y ejemplos: Aunque el nivel de profundidad y detalle de la información disponible es diferente, generalmente es suficiente para poder usar cada una de las distintas familias sin recibir un entrenamiento específico.

A partir de estas características comunes, cada una de las plataformas presentan ligeras diferencias, las cuales aparecen resumidas para algunas de las plataformas más comunes en la Tabla 1, presentada en (Baronti et al. 2007):

Tabla 1. Comparación de varias arquitecturas de sensores.

	Btnode 3	**mica2**	**mica2dot**	**micaz**	**telos A**	**tmote sky**	**EYES**
Fabricante	Art of Technology	Crossbow	Crossbow	Crossbow	Imote iv	Imote iv	Univ. de Twente
Microcontrolador	Atmel Atmega 128L	Atmel Atmega 128L	Atmel Atmega 128L	Atmel Atmega 128L	Texas Instruments MSP430	Texas Instruments MSP430	Texas Instruments MSP430
Frecuencia de reloj	7.37 MHz	7.37 MHz	4 MHz	7.37 MHz	8 MHz	7.37 MHz	5 MHz
RAM (KB)	64+180	4	4	4	2	10	2
ROM (KB)	128	128	128	128	60	48	60
Almacenamiento (KB)	4	512	512	512	256	1024	4
Radio	Chipcon CC1000 315/433/868/9 16MHz 38.4 Kbaudios	Chipcon CC1000 315/433/868/9 16 MHz 38.4 Kbaudios	Chipcon CC1000 315/433/868/9 16 MHz 38.4 Kbaudios	Chipcon CC2420 2.4 GHz 250 Kbps IEEE 802.15.4	Chipcon CC2420 2.4 GHz 250 Kbps IEEE 802.15.4	Chipcon CC2420 2.4 GHz 250 Kbps IEEE 802.15.4	RFM TR1001868 MHz 57.6 Kbps
Máximo alcance	150-300 m	150-300 m	150-300 m	75-100 m	75-100 m	75-100 m	75-100 m
Alimentación	2 pilas AA	2 pilas AA	Pila de botón	2 pilas AA	2 pilas AA	2 pilas AA	2 pilas AA
Conector para PC	Placa de programación conectada a PC	Placa de programación conectada a PC	Placa de programación conectada a PC	Placa de programación conectada a PC	USB	USB	Puerto serie
Sistema Operativo	Nut/OS	TinyOS	TinyOS	TinyOS	TinyOS	TinyOS	PEEROS
Transductores	En placa de adquisición de datos	En placa de adquisición de datos	En placa de adquisición de datos	En placa de adquisición de datos	En placa	En placa	En placa de adquisición de datos
Extras	Bluetooth						

De la tabla anterior se deberían destacar dos familias, los nodos Mica, fabricados por Crossbow, y los Telos, una plataforma abierta desarrollada en la Universidad de California Berkeley fabricados en ese momento por Moteiv (absorbida más tarde por Texas Instruments). Si bien no son los únicos, sí son ampliamente referenciados en la bibliografía, usados en gran número de experimentos y se pueden seguir adquiriendo en la actualidad dispositivos equivalentes a ellos pero fabricados con componentes más actuales por otras compañías, guardando la compatibilidad con dichos sensores.

La clasificación que aparece en la Tabla 1 es bastante representativa (pero no exhaustiva), y recoge claramente la existencia de coincidencias entre alguna de ellas, la más importante es la circuitería del interfaz radio, la cual corresponde con un casi todos con el fabricante Chipcon (adquirida por Texas Instruments), ya sea el modelo CC1000 o el CC2420, que da soporte al estándar IEEE 802.15.4 (IEEE2003) para el nivel físico y capa de acceso al medio, el cual se ha convertido en el más usado por las redes de sensores inalámbricos de área personal (WPAN, del inglés *Wireless Personal Area Network*). Con respecto a la plataforma **EYES**, su presencia en la actualidad es prácticamente nula (no se ha encontrado información actualizada), no ocurre así con los **BTnode** (BTnodes), los cuales aún están disponibles y soportan también **TinyOS** (Nut/OS (Egnite) aún está disponible en la actualidad).

Otros dispositivos que surgieron a mediados de las década pasada fueron los **Intel Mote 2** o **Imote2** (Adler et al. 2005). Estos nacieron de una colaboración entre la Universidad de California Berkeley e Intel. También fueron fabricados por la empresa Crossbow (en el año 2007) y soportaban el sistema operativo TinyOS, tanto como Contiki, y como los anteriormente comentados, usan un chip radio CC2420. En la actualidad su proyecto está cerrado por Intel, aunque se siguen usando en entornos académicos.

Debido a que al momento de aparición del artículo de Baronti y otros es a mediados de 2006, del cual se ha extraído la tabla anterior, no aparecen otras plataformas de interés, como **Sun SPOT** (Oracle Labs) (del inglés *Small Programmable Object Technology*) de Sun Labs (ahora Oracle) que lazaba su primera versión comercial en 2006 y cuyo sistema operativo denominado Squawk incorpora una máquina virtual Java, muy orientados estos al entorno académico. También están ausentes los nodos **Waspmote** (Libelium) de Libelium, una spin-off de la Universidad de Zaragoza fundada también en 2006, que aportan una amplia versatilidad gracias a la gran variedad de módulos disponibles. Siguiendo un orden cronológico, otra plataforma desarrollada por una empresa española, Advancare, S. L., comercializa bajo su marca Zolertia sus **nodos Z1** con micro-controladores de 16 bits y 8 KBytes de RAM, y que usan también los chip CC2420 para las comunicaciones inalámbricas con el estándar IEEE 802.15.4, aparte de la posibilidad de usar a niveles superiores 6LoWPAN o Zigbee.

Vistas cada una de las diferentes plataformas hardware, otro aspecto fundamental de cada una de ellas es el software que las controla. Así, cada familia soporta o es compatible con uno o varios sistemas operativos de redes de sensores. Estos sistemas tienen ciertas características comunes como la abstracción de recursos, gestión de interrupciones, concurrencia y comunicaciones de red (Thang Vu Chien et al. 2011). De entre los sistemas operativos más usados, el predominante es TinyOs (Hill et al. 2000), lo cual queda claramente patente del análisis de las plataformas de redes de sensores presentada en la Tabla 1.

Otro sistema operativo para nodos es **Contiki** (Dunkels et al. 2004), un proyecto liderado por Adam Dunkels del *Swedish Institute of Computer Science*, el cual es compatible con casi todos los nodos que funcionen también con TinyOS.

Los sensores Sun SPOT funcionan con el sistema operativo **Squawk**, también de código abierto, y que proporciona una máquina virtual de Java parecida a la que proporciona Java 2 Micro Edition (Sun Microsystems). Este sistema operativo, al existir coincidencias entre las distintas plataformas de Java, permite el intercambio de aplicaciones de manera más directa siempre que usen los paquetes comunes.

Como se ha podido observar, existe una amplia variedad de plataformas, sistemas operativos, así como lenguajes de programación para el desarrollo de aplicaciones. Teniendo en cuenta todos ellos, la idoneidad de cada uno dependerá de la aplicación concreta, así como de factores externos como recursos económicos, experiencia del equipo desarrollador, capacidad de encontrar repuestos, actualizaciones y mejoras. Todas estas variables pueden condicionar severamente cualquier investigación o desarrollo, por lo que siempre deberán ser tenidas en cuenta.

Con respecto a los protocolos o arquitecturas comerciales o industriales cabe destacar Zigbee, Wireless-HART y el estándar abierto ISA100.11a.

- **Zigbee** (ZigBee Alliance) es un conjunto de protocolos de red y aplicación que se instalan encima de la capa de enlace que usa 802.15.4. Zigbee define los protocolos de red y enrutamiento, y la capa de aplicación. Está diseñado para minimizar el consumo de los dispositivos y posibilita en teoría, la interconexión de miles de dispositivos en redes malladas. En Zigbee existen tres tipos de dispositivos, los dispositivos sencillos Zigbee, como sensores, actuadores o controladores, que interactúan con el entorno; enrutadores que enlazan dispositivos sencillos para formar grupos y son los únicos que reciben la información de los dispositivos sencillos. El tercer grupo de dispositivos son los coordinadores, los cuales tienen una función parecida a pasarelas que interconectan secciones de red diferentes, además de almacenar información de la red e iniciar la formación de la misma. Su estándar está disponible desde 2005.
- **Wireless-HART** (HART Communication Foundation2007) es una extensión de HART (*Highway Addressable Remote Transducer*) añadida en 2007 en su revisión 7.0. Este protocolo, creado por la *HART Communication Foundation*, incluyó en esa revisión un interfaz inalámbrico para los dispositivos de campo. Wireless-HART también se basa en 802.15.4, aportando un acceso confiable, seguro y energéticamente eficiente. Permite topologías en malla, estrella o híbridas. Normalmente los dispositivos inalámbricos están diseminados por factorías o plantas industriales para controlar la maquinaria. La Comisión Internacional de Electrotecnia confirmó a Wireless-HART como el estándar internacional IEC [4] 62591 Ed.1.0 para comunicaciones inalámbricas y automatización de procesos en marzo de 2010 (*Industrial Communication Networks—Wireless Communication Network and Communication Profiles—WirelessHART*, IEC 62591).
- **ISA100.11a** (ISA100 Standards Committee2009) es el resultado de los esfuerzos del comité ISAl00 de la *International Society of Automation* (ISA)

[4] IEC son las siglas de *International Electrotechnical Commission*.

para aunar todas las infraestructuras posibles bajo acceso inalámbrico en una misma especificación. Usa comunicaciones en la banda ISM (*Industrial, Scientific and Medical radio band*, 2,4 GHz a 2,5 GHz). Usa la especificación de capas OSI, incorpora seguridad y gestión de sistemas. El estándar se centra en el bajo consumo de potencia, escalabilidad, robustez y la interoperación con otros dispositivos inalámbricos. Permite topologías tanto malladas como en estrella y tiene una política flexible y escalable de funciones de seguridad. ISA100.11a fue estandarizado en 2009, sin embargo el *American National Standards Institude* (ANSI) lo rechazó como estándar oficial y por lo tanto impidió el progreso a estándar IEC. En la actualidad se ha creado un grupo, el ISA100.12 con el objeto de hacer converger a los dos estándares: WirelessHART e ISA100.11a.

2.1.6 TENDENCIAS ACTUALES EN LAS REDES DE SENSORES

El mayor esfuerzo de investigación durante estos últimos años dentro de las redes de sensores inalámbricos ha sido sin duda en la gestión de recursos, ya sea de manera interna a un sensor, como en conjunto dentro de una red. Así, la capa MAC ha sido profundamente estudiada debido a que son las comunicaciones la fuente del mayor consumo en un sensor. Junto al estudio sobre la capa de acceso al medio están también el control de topología y las técnicas de cooperación entre sensores, como puede verse en (Buratti et al. 2009), donde aparecen las tecnologías más importantes que se han ido desarrollando paralelamente a la evolución de las redes de sensores, tales como el **protocolo IEEE 802.13.**4, o **6LowPan** (Kushalnagar et al. September 2007) que permite el envío de paquetes de IPv6 sobre una red de sensores.

A la par de éstas últimas, hay que tener muy en cuenta las arquitecturas de comunicaciones que han ido apareciendo a mediados de la década pasada y cuyo esfuerzo principal ha sido la estandarización. Así pues, **Zigbee**, **WirelessHART** y el **estándar ISA100.11a**, además de los esfuerzos por unificar y estandarizar de la *Wireless Industrial Networking Alliance* (WINA), han conseguido que dentro del entorno de las redes de sensores aparezcan soluciones industriales que cuentan cada día con más dispositivos. Así pues, las aplicaciones industriales son hoy en día una realidad, si bien el desarrollo de las mismas aún requerirá más estudio en aspectos como seguridad, calidad de servicio y aplicaciones en tiempo real.

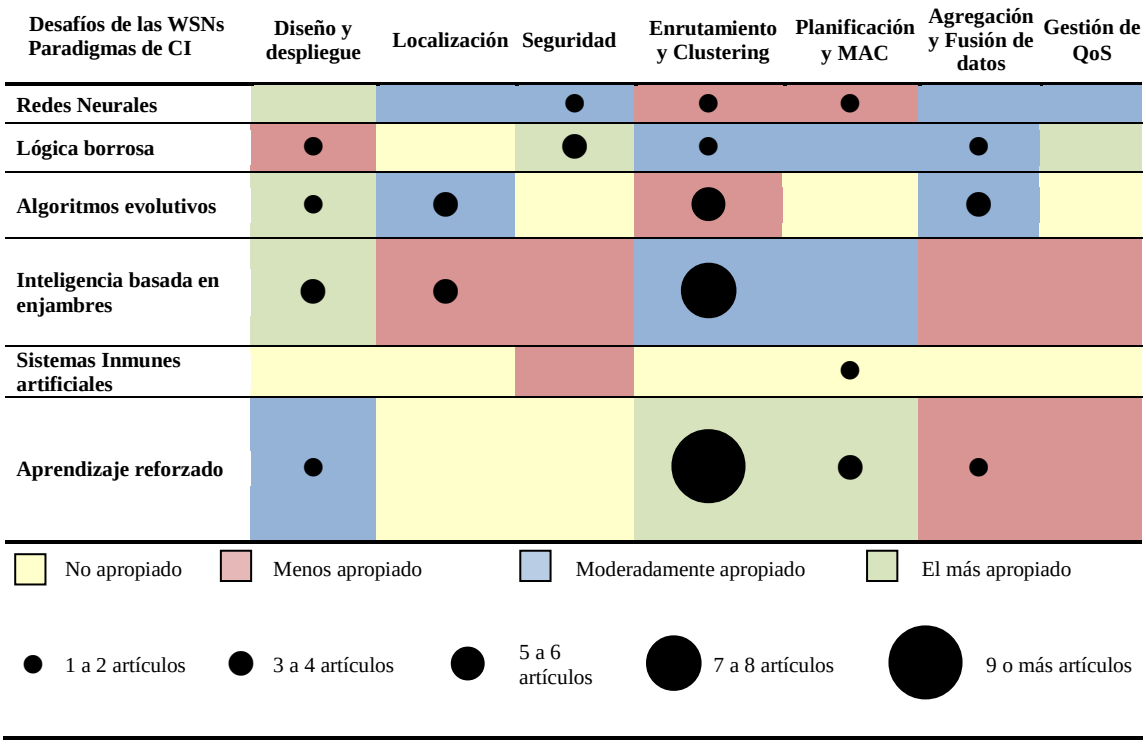
Sin embargo, las aplicaciones industriales no son la única área de expansión de las redes de sensores. A lo largo de la evolución sufrida por dichas redes, han ido apareciendo nuevas áreas de interés, tales como las **aplicaciones multimedia** (Akyildiz et al. 2007), la calidad de servicio (Yigitel et al. 2011), **seguridad y protección de la información** (Xiangqian Chen et al. 2009), así como la aplicación de técnicas de **inteligencia computacional**. Estas nuevas áreas han sido propiciadas en parte por las mejoras en la capacidad de cómputo de los sensores y de los dispositivos de captación de imágenes y audio, además del creciente interés en las mismas.

Con respecto a la **inteligencia computacional**, se ha hecho viable su aplicación en nuevos entornos para los que las redes de sensores son idóneas, como por ejemplo, la monitorización y supervisión de procesos naturales, cuyo modelado requiere una alta complejidad analítica, y que tradicionalmente se vienen abordando con este tipo de técnicas. Por otro lado, el aumento de aplicaciones de redes de sensores, ha hecho necesaria la protección de la información enviada por la red en determinadas soluciones (Kumar et al. 2012).

Concretamente es en la aplicación de técnicas de inteligencia computacional a las redes de sensores donde se ubica parte de las contribuciones de la presente tesis. Como se ha comentado, estas técnicas conforman un medio para poder dar solución a una gran variedad de los problemas que se plantean dentro de las redes de sensores, ya que son potentes herramientas, que gracias a la mejora en las prestaciones de los nodos, están disponibles en la actualidad.

Es evidente que existe una gran variedad de técnicas y métodos dentro de la inteligencia computacional, y que cada uno de ellos afronta de diversa manera los problemas a resolver, así como su aplicabilidad a los mismos. Hecho que conlleva que el consumo de recursos de cada uno pueda ser sensiblemente diferente con respecto a los demás. Por lo tanto, no todas las técnicas de inteligencia computacional son igualmente aplicables a las redes de sensores, lo cual es analizado en (Kulkarni et al. 2011), donde se hace una revisión de las técnicas de inteligencia computacional empleadas en las redes de sensores hasta la fecha de redacción del artículo. La clasificación presentada organiza dichas técnicas por la cantidad de contribuciones científicas en las que se aplican, y por la idoneidad en su aplicación a las redes de sensores que le atribuye el autor a cada una de ellas. En dicho artículo aparece como resultado la siguiente tabla.

Tabla 2. Repaso de los paradigmas de Inteligencia Computacional aplicados a las redes de sensores. Fuente (Kulkarni et al. 2011).



La valoración del autor, en cuanto a la idoneidad de cada una de las técnicas de inteligencia computacional, se hace teniendo en cuenta los desafíos más comunes en las redes de sensores. Cabe destacar la buena aplicabilidad encontrada por el autor para la **lógica borrosa**, la cual aparece especialmente indicada para algunos de los problemas más importantes de las redes de sensores, tales como enrutamiento, formación de clústeres, planificación, gestión de la capa MAC, agregación de datos y QoS. Otro aspecto interesante de la tabla es que aparecen el número de contribuciones científicas

encontradas por el autor en el momento de su confección, si bien este número ha ido creciendo en los últimos años, la fecha de la redacción del artículo coincide aproximadamente con el inicio de los trabajos de investigación de la presente tesis, sirviendo esto como referencia de la novedad de las aportaciones que se presentarán en el Capítulo II.

Las técnicas de inteligencia computacional, como ya se ha comentado, se vienen aplicando a las redes de sensores desde hace algunos años. Concretamente, el uso de lógica borrosa en las redes de sensores, según la revisión bibliográfica llevada a cabo, aparece alrededor de los años 2.003-2.004, y en mayor número, a partir del año 2.005. Cabe destacar que los primeros trabajos con cierta repercusión se centran en el control de topología, como la formación de clústeres (Halgamuge et al. 2003, Indranil Gupta et al. 2005), enrutamiento eficiente (Rea & Pesch 2004, Yusuf & Haider 2005) o detección de eventos (Liang & Wang 2005), entre otros.

En particular, en el presente trabajo de investigación se han usado **sistemas borrosos basados en reglas o FRBS**. Es importante reseñar que el uso de estos sistemas dentro de las redes de sensores, al tiempo del comienzo del presente trabajo de investigación, estaba en sus inicios. Así pues, las contribuciones más destacables en este campo eran los trabajos de (Marin-Perianu & Havinga 2007) aplicados a la detección de incendios, o el trabajo presentado en (Canada-Bago2007), donde el sistema borroso es combinado con un algoritmo genético para el aprendizaje dentro del propio sensor. Estas contribuciones ponen de manifiesto que la aplicación de FRBS a las redes de sensores es una técnica apropiada para gran variedad de soluciones, debido fundamentalmente a que estos sistemas cumplen con dos importantes requisitos, según establece Marin-Perianu en (Marin-Perianu & Havinga 2007):

- Primero, son suficientemente sencillos para ser ejecutados en dispositivos de prestaciones limitadas
- y segundo, toleran que los datos disponibles sean imprecisos o incluso inciertos.

Otra ventaja importante de los FRBS es que, para modificar su comportamiento, la cantidad de datos necesarios, si bien depende del número de reglas, conjuntos borrosos y variables, es relativamente baja[5]. Así, modificar el diseño de un conjunto borroso, los límites de una variable o alguna de las proposiciones de una regla, supondría apenas unas decenas de bytes. Por lo tanto, estas modificaciones podrían ser enviadas a través de la red sin ocasionar un consumo excesivo de energía y conseguir así la mejora o actualización del comportamiento de un sensor de una manera relativamente sencilla. Además de para adaptar el comportamiento del sensor, estos parámetros podrían ser enviados por éste último a otros nodos, o a una estación base, como resultado del aprendizaje obtenido por el mismo. Así pues, las mejoras encontradas en la ejecución de proceso de aprendizaje pueden ser distribuidas a los demás nodos, mejorando el rendimiento total de la red. Sin embargo, esta actualización de los parámetros en los nodos de la red sí debería acometerse con más cautela, pudiéndose beneficiar de esquemas de agregación de datos existentes para la distribución de esos parámetros.

Como se ha venido comentando, el uso de lógica borrosa en las redes de sensores ha ido tomando especial interés en los últimos años, si bien ha sido aplicada de muy diferente

[5] Fundamentalmente los FRBS se modelan con una base de conocimiento, la cual puede ser codificada de diferentes formas, pudiendo llegarse a hacer muy compactas, como quedará de manifiesto en la sección 5.2, donde se proponen dos formatos para su codificación.

manera. Se ha comentado que en (Marin-Perianu & Havinga 2007) se usan FRBS para la detección de incendios, estando dotado cada sensor de su propio motor de inferencia, como en el trabajo de Xia (Xia et al. 2007), donde los nodos encargados de medir las magnitudes bajo estudio incorporan un sistema borroso para la gestión de la QoS de la red de sensores, basándose en la adaptación de los periodos de muestreo a la tasa de pérdidas por tráfico en la red.

Otro tipo de aplicaciones usan la **lógica borrosa de manera centralizada**, ya sea a priori o continua. Entre las soluciones a priori se encuentra aquellas que planifican a partir de ciertas condiciones iniciales el despliegue de una nueva red de sensores, buscando qué protocolos y/o topología serían los más apropiados. Dentro de este tipo están los trabajos de (Pirmez et al. 2007), donde un sistema borroso determina el protocolo de comunicaciones más apropiado para el despliegue de una red en función del número de estaciones base y sensores, tráfico esperado y parámetros de QoS.

Por otro lado, entre los sistemas que usan la **lógica borrosa de una manera reiterada o continua** durante el tiempo vida de la red, está por ejemplo el trabajo de Kim y otros (Kim et al. 2008), donde se usa para la elección de las cabezas de clústeres. Otras aplicaciones donde la lógica borrosa se emplea para el cálculo del coste para rutas en protocolos de enrutamiento pueden verse en (Rea & Pesch 2004, Haider & Yusuf 2009). También aparecen aplicaciones como (Yusuf & Haider 2005, Misra et al. 2009), las cuales describen un protocolo de enrutamiento basado en lógica borrosa.

Cabe destacar que la **aplicación de la lógica borrosa al enrutamiento** ha suscitado mucho interés, debido al hecho fundamental de que un óptimo algoritmo de enrutamiento evitaría el desperdicio de energía que supone el envío de paquetes con más potencia de la necesaria, al usar rutas no adecuadas.

Sin embargo, es en **la gestión autónoma del sensor basada en FRBS** donde el presente trabajo de investigación pretende aportar nuevas perspectivas, tales como la gestión inteligente de las tareas que un sensor debe acometer para cumplir con la función que se le ha otorgado dentro de la red. Además, dicha gestión de comportamientos debe tener siempre en cuenta que un sensor tiene unos recursos limitados, principalmente la energía disponible. Así pues, dentro de este campo no se han encontrado trabajos ajenos al llevado a cabo por el grupo de investigación. Lo más cercano está en el trabajo de Benoit (Benoit et al. 2001), donde se describe un modelo para definir los servicios que los sensores tendrían disponibles en forma de agentes inteligentes, lo cual entronca con la perspectiva que será desarrollada en este trabajo: la coexistencia de agentes controlados por subsistemas inteligentes dentro de cada sensor. Estos subsistemas marcarán el funcionamiento del sensor dentro de la aplicación a la que se destine, así como podrán emplearse para tareas organizativas de la red, tales como la formación de clústeres o enrutamiento.

2.1.7 CARENCIAS E INCERTIDUMBRES DE LAS REDES DE SENSORES

Se ha expuesto anteriormente que los nodos, y por extensión, las redes de sensores, tienen unas limitaciones muy importantes de disponibilidad de recursos energéticos, principalmente derivadas del hardware. Este hecho ha propiciado que gran parte de los trabajos sobre redes de sensores traten sobre métodos de optimización y ahorro de recursos. Sin ánimo de ser reiterativo, a continuación se analizarán algunas de las carencias más importantes de las redes de sensores intentado aportar otros puntos de vista que no sean sólo las limitaciones derivadas del hardware. Dichas carencias, por un

lado alientan, y en cierto modo obligan, a seguir investigando en técnicas, algoritmos y soluciones que logren minimizar su efecto. Sin embargo, por otro lado, pueden generar dudas sobre el éxito de las redes de sensores como solución tecnológica de vanguardia. En (Srivastava2010) se hace un recorrido de los desafíos a los que se enfrentan las redes de sensores en diferentes entornos de aplicación, viendo como aún son necesarios resolver aspectos importantes, como la respuesta en tiempo real o la seguridad, lo cual genera un mínimo de incertidumbre en la integración de las redes de sensores en ciertos entornos como, por ejemplo, los edificios inteligentes o los hogares. Teniendo en cuenta las inquietudes que las redes de sensores generan, se analizarán las carencias desde los siguientes puntos de vista:

1. Futuro incierto de las aplicaciones de las WSNs.
2. Reducida proporción de experimentos con sensores reales.
3. Dispersión en las plataformas de redes de sensores.
4. Falta de protocolos de aplicación genéricos, en concreto para dar soporte a sistemas basados en el conocimiento o de propósito general

Así pues, siguiendo con lo comentado anteriormente, y como primer punto a analizar sobre las carencias e incertidumbres de las redes de sensores, está el hecho de que en la actualidad, y después de más de diez años de evolución, no existen muchas **aplicaciones reales** en el mercado, ya sea planteando soluciones a problemas cotidianos o en la industria. Esto se agrava con hechos como el empleo mayoritario en la bibliografía de simulaciones para mostrar los resultados de trabajos de investigación, sin la correspondiente contrapartida de experimentos con dispositivos físicos. Por lo tanto, las contribuciones dentro del marco de las redes de sensores no suelen incluir la prueba en redes desplegadas en entornos reales que puedan enfrentarse a problemas como sobrecalentamientos, interferencias electromagnéticas, fallos del hardware, etc. Si bien, desde 2007, con la aparición de las familias de arquitecturas y protocolos WirelesHART e ISA100.11a, son cada día más las empresas que ofrecen soluciones industriales basadas en WSNs, tales como Siemens o Honeywell, el potencial y la versatilidad de éstas no están en consonancia con su repercusión en el mercado.

Algunos autores han reflexionado sobre este problema, viendo que donde las redes de sensores encontrarán un importante campo de expansión, es dentro de lo que se ha venido a llamar **la Internet de las Cosas**, del inglés *The Internet of Things* (IoT).

IoT aparece como tal en el trabajo de Kevin Ashton (Ashton2009) y presupone que el culmen de las redes de ordenadores, y por ende de Internet, será cuando cada pequeño equipo o dispositivo electrónico tenga la capacidad de estar interconectado, pudiendo ser controlado y gestionado desde cualquier otro punto. Es en ese escenario donde las WSNs tendrían su apogeo, ya que son los dispositivos idóneos para este cometido. Si bien la idea original de Ashton combina los avances en los sensores con los avances en la tecnología RFID, ambas son complementarias, teniendo presente que las redes de sensores aportan muchas más prestaciones y funcionalidades que ésta última. En este caso, la tecnología subyacente que permitiría esta expansión sería fundamentalmente 6LoWPAN.

Sin duda, con los avances producidos en estos últimos años, y los que aparecerán en poco tiempo, la mayoría de los aspectos técnicos que conciernen a las redes de sensores, y que impiden un despliegue a mayor escala, habrán sido resueltos, es entonces cuando aparecen nuevas preguntas, según recoge Bohli en (Bohli et al. 2009). Por lo tanto, se deberá tener preparadas las respuestas apropiadas, si no, en el caso particular de las

redes de sensores, éstas se pueden quedar relegadas al mundo académico y a contadas aplicaciones industriales experimentales. Así pues, habrá que pensar en:

- ¿Cómo serán los servicios que estarán disponibles en IoT?
- ¿Qué incentivos habrá para proveer información a estos servicios?
- ¿Cómo se instalaran e iniciarán estos servicios?

Estas son preguntas que el autor aborda en la contribución antes mencionada. Sin embargo, las redes de sensores dependen mucho de la expansión de IoT, o lo que es igual, de la aplicación a soluciones tecnológicas reales, como en el campo de la medicina, la automoción, la industria o la domótica.

Así pues, teniendo en cuenta la necesidad que suscita la carencia antes comentada, los **experimentos** que se han realizado como parte de la elaboración de la presente tesis han tenido siempre en mente la aplicabilidad a posibles a problemas concretos, ya sean dentro de la industria, la agricultura o la mejora de la vida de las personas. Más aún, y enlazando con otra de las carencias que se desprenden del estudio de la bibliografía referente a las WSNs, el relativo bajo número de experimentos en los que usan sensores reales, se ha intentado, en la medida de los posible, que el mayor número posible de los resultados aplicables a las WSNs de la presente tesis sean implementados, al menos como prototipo, en una plataforma de redes de sensores real.

El hecho de la existencia de un **bajo número de pruebas en sensores reales**, que complementen los trabajos de investigación sobre los sensores, en parte se debe a que las herramientas de simulación son cada vez más completas y por otro lado, el gasto que conlleva en fondos y tiempo una prueba en equipos físicos es normalmente superior al de preparar una simulación. Además, estas pruebas cuentan con problemas añadidos como los costes de mantenimiento, fallos de implementación y lugar de despliegue, sin contar con las dificultades de poder generar las condiciones reales para las que se haya destinado la investigación. Por eso, si en principio esta carencia puede parecer importante, es sin duda justificable en muchos de los casos, pero no en aquellos que buscan mejoras en aspectos funcionales básicos como enrutamiento, formación de clúster, etc., donde las pruebas pueden ser realizadas con un número abordable de dispositivos y las condiciones del despliegue no requieren más que la diseminación de los nodos en cierta área.

Entroncando con lo comentado en el párrafo anterior, en cierto modo parte de las causas de los aspectos antes comentados, una carencia patente es la **falta de un estándar claro en las redes de sensores**, aunque en la industria están presentes WirelessHART e ISA100.11a, ya se presenta una dualidad, además Zigbee también aporta soluciones tanto industriales, como para el hogar. Además, existe una amplia variedad de plataformas en las que la forma de implementar aplicaciones difiere entre ellas. Si bien en el resumen expuesto en (Baronti et al. 2007), varias plataformas de redes de sensores usan TinyOS, éste sistema operativo no es un estándar. En la actualidad se siguen haciendo dispositivos compatibles con TinyOS, pero también está Contiki o Squawk entre otros.

A pesar de todo, existe un factor importante que une a la mayoría de las plataformas de redes de sensores, el uso como base para las comunicaciones inalámbricas de la **especificación IEEE 802.15.4** (IEEE2003), lo cual en parte podría solventar el problema de los diferentes sistemas operativos, como sucede con las redes de ordenadores, donde se ofrecen servicios que son consumidos de la misma forma por

máquinas Windows, Unix, Linux o Mac. Claro está que esto será posible cuando los protocolos de comunicación de niveles superiores (red en adelante) sean compatibles. Por lo tanto, no sería tan necesaria una estandarización en cuanto a las plataformas de redes de sensores, o en los sistemas operativos, si se consiguiera uniformar las comunicaciones. Y la compatibilidad de la inter-operación de las redes de sensores en parte se podría conseguir usando de manera más extendida la torre de protocolos OSI, o una adaptación de la misma.

Sin embargo, son muchos los trabajos que con el fin de conseguir un mayor ahorro de recursos recurren a soluciones entre-capas (del inglés ***cross-layer***), o lo que es lo mismo, no se definen interfaces de separación entre los niveles tradicionales, como capa de enlace, red o transporte para poder así conseguir un mayor aprovechamiento de recursos. Los diseños *cross-layer* normalmente usan información de los niveles inferiores con objeto de ofrecer un servicio más adaptado a la aplicación concreta y al estado del sensor. Así pues, son comunes los algoritmos de enrutamiento de nivel de red que tienen en cuenta los ciclos de trabajo y sueño del nivel de enlace para ofrecer rutas más eficientes que cuenten con los nodos que estarán activos. Concretamente, los diseños que vinculan el nivel de enlace con el nivel de red son los más habituales.

Por lo tanto, la variedad de plataformas, los diseños *cross-layer*, la fase de experimentación en la que aún se encuentran las redes de sensores, y las especiales características de éstas en cuanto a su limitación de recursos, han dado como resultado otra de las carencias anteriormente enumeradas, la no existencia de un protocolo de intercambio de datos estándar. De hecho, no existe un protocolo de aplicación lo suficientemente extendido y adaptado a las especiales características de las redes de sensores que facilite el despliegue de aplicaciones sencillas de monitorización y control. Este hecho queda de manifiesto en revisiones de las redes de sensores hechas por (Karl & Willig 2007, Akyildiz & Vuran 2010, Zheng & Jamalipour 2009), donde el espacio dedicado a los protocolos de aplicación es testimonial.

A pesar de todo, esta carencia es relativa, ya que como se ha comentado, existen notables avances en la integración de las redes IP y las redes de sensores con el **protocolo 6LoWPan**. Sin embargo 6LoWPAN permite encaminar paquetes IPv6 sobre una red de sensores a gracias a unos mecanismos de compresión de cabeceras que permiten evitar que los 40 bytes de la cabecera de IPv6 tengan que viajar en las limitadas tramas de 802.15.4 de tan sólo 102 bytes. Así pues, según la revisión mostrada en (Silva et al. 2009) las implementaciones hasta ese momento permitían el uso de IPv6/UDP (*User Datagram Protocol*) sobre una red de sensores, e incluso TCP (*Transmission Control Protocol*), si bien este último no es el protocolo más adecuado para un entorno como las redes de sensores[6]. Estos protocolos de transporte permitirían, que a través de la interconexión de la red de sensores con una red IPv6, se pueda consultar el estado de un sensor con protocolos tales como el protocolo de gestión SNMP (del inglés *Simple Network Management Protocol*) sobre UDP, o HTTP (del inglés *Hiper-Text Transfer Protocol*) sobre TCP, como si en este último caso se tratara de un servidor web más de Internet. Sin embargo, estos protocolos de aplicación no

[6] Algunos mecanismos presentes en TCP como la retransmisión en caso de pérdida, el control de congestión o el uso de extensiones como los asentimientos selectivos (SACK), necesarios para conseguir un buen aprovechamiento en redes inalámbricas, conllevarían un elevado gasto en el número de mensajes, así como en el tamaño de los mismos.

están adaptados a las especiales características de las redes de sensores, y su formatos de mensaje distan mucho de lo aceptable en para un sensor[7].

Vistas todas estas carencias, y analizado el impacto en el desarrollo de aplicaciones de las WSNs, así como de los caminos que aún quedan por cubrir, merece la pena tener en cuenta el trabajo de Bhaskaran y Kameswari (Raman & Chebrolu 2008), donde hacen un **análisis crítico de las contribuciones hechas sobre redes de sensores** en cierto número de revistas y congresos, dejando ver que la investigación sobre redes de sensores adolece en muchos casos de un carácter integrador, de resultados probados en dispositivos reales, así como de la aplicabilidad de muchas investigaciones a la industria o al mercado de consumo. Como se ha comentado, el autor señala la falta de estándares como crucial, así como el coste, la no existencia de arquitecturas unificadas o la seguridad, como desafíos a los que se enfrentan hoy en día las redes de sensores. Además, a pesar de que existen gran cantidad de desarrollos sobre aplicaciones concretas en agricultura, medicina, vigilancia, control de incendios, etc., generalmente cada solución intenta aportar algo nuevo y no unificar la investigación hallada en ese campo para hacer un esfuerzo en conseguir un estándar de aplicación. Esto queda de manifiesto en trabajos como (Ruiz-Garcia et al. 2009) en el que se hace un recorrido por aplicaciones de agricultura y alimentación, comentando alrededor de unas 80 soluciones, las cuales presentan un denominador común, son variadas y muchas ellas apenas tienen tiempos de prueba de unos pocos días.

Sin embargo, las carencias aquí presentadas no son barreras insalvables bajo ningún concepto, siendo oportunidades de trabajo e investigación como se señala en (Srivastava2010), donde el autor analiza los desafíos a los que se tendrán que someter las redes de sensores, como por ejemplo, la integración en edificios inteligentes, medicina y agricultura.

Es en este punto donde iniciativas como el IETF ROLL (Watteyne et al. 2011) y el desarrollo de 6LoWPAN, WirelessHART o ISA100.11a, intentan aportar algo de orden en el nivel de red de las redes de sensores, pero aún son necesarios muchos esfuerzos para que el despliegue de cualquier nodo de una red de sensores, sea tan común como añadir un teléfono móvil a una red WIFI.

Como conclusión, con respecto al análisis de las carencias e incertidumbres sobre las redes de sensores, se puede decir que, a pesar del gran recorrido en cuanto a contribuciones científicas y técnicas, las redes de sensores aun necesitan un tiempo para llegar a ser aquello que se pretendió en un principio, que no es más que el paradigma de la integración total de la electrónica, comunicaciones y ser humano.

2.1.8 UTILIZACIÓN DE TEST-BEDS EN LAS REDES DE SENSORES

El uso de bancos de pruebas (o *test-bed*) está ampliamente extendido en la ingeniería en general debido a que aportan un aspecto fundamental en la investigación y desarrollo de nuevos dispositivos, técnicas o algoritmos: pruebas con dispositivos físicos y condiciones reales. En las redes de sensores juegan un papel muy importante, siendo incluso objeto de revisión en contribuciones científicas y técnicas sobre redes de sensores como en (Yick et al. 2008, Imran et al. 2010). Otra funcionalidad que proporcionan los test-beds, además de la manera de probar equipos, protocolos y

[7] HTTP por ejemplo, usa los caracteres “HTTP/1.1” para informar de su versión en cada mensaje, siendo así enviados 64 bits, para algo que bastaría con un par de bits, ya que las versiones existentes de HTTP, son tres (0.9, 1.0 y 1.1), aunque se pueden reducir a dos, la 1.0 y la 1.1.

algoritmos, es que permiten que estos experimentos sean configurados y ejecutados de manera remota, así como permitir su replicación y repetición. Las ventajas que aporta un test-bed, se deben en parte a la incorporación de parámetros configurables o simulados, que intentan reproducir las condiciones más parecidas al entorno final donde se ubicará el dispositivo en pruebas.

Un test-bed puede estar formado por un variado número de nodos, tanto finales como los que forman la red troncal, pasarelas para otros tipos de redes, estaciones base, así como equipos de mayor tamaño y capacidad de proceso como routers o PCs. Todos los equipos que forman el test-bed no tienen por qué ser de la misma tecnología, como se desprende de lo anterior. Además, los elementos que formen la red de sensores pueden mezclar dispositivos de plataformas diferentes, y si no comparten un nivel común de comunicación, podrían interconectarse usando pasarelas específicas para ambas tecnologías.

En particular, en las redes de sensores el uso de test-bed está bastante generalizado desde comienzos de siglo debido a que, en principio, una red de sensores tiene ciertas características en su despliegue que facilitan la instalación de un banco de pruebas. Estas propiedades son:

- **Tamaño reducido**: normalmente los nodos de las plataformas de redes de sensores más habituales, comentadas con anterioridad, son de apenas unos centímetros en todas sus dimensiones, superando en contados casos el decímetro.
- **Alimentación por baterías**: la alimentación autónoma de los nodos les confiere la posibilidad de su distribución en áreas de prueba donde no exista una infraestructura previa de suministro eléctrico, o donde exista, minimiza los costes al no necesitar de inicio tomas adicionales de corriente.
- **Capacidad de almacenamiento y procesado distribuido**: intrínseco al diseño hardware de cada nodo, aunque con prestaciones diferentes según la plataforma usada, es un aspecto fundamental, ya que permite que los programas, así como los datos que maneja el sensor, puedan ser pre-procesados, evitando comunicaciones innecesarias con nodos intermedios (*sinks*) o estaciones base. Además, la capacidad de cálculo distribuida amplía considerablemente las aplicaciones de las redes de sensores.
- **Capacidad de comunicación inalámbrica**: sin duda la capacidad de comunicación inalámbrica, a pesar de que un nodo pueda poseer otros interfaces cableados, es lo que posibilita el despliegue de un test-bed en un entorno no acondicionado previamente. Sin embargo, aspectos como la cobertura e interferencia electromagnética hacen necesario estudios precisos en la ubicación de los nodos.

A pesar de estas ventajas, el diseño y despliegue de un test-bed no es un proceso trivial, ya que, aunque los elementos a desplegar tienen las características antes vistas, las cuales facilitan parte de proceso, el establecer o disponer del número apropiado de nodos para que un experimento tenga validez es complejo. Además, existen otros aspectos adicionales que tienen que ser tenidos en consideración, como se establece en (Handziski et al. 2006), los cuales se desarrollan a continuación, dado que aportan una visión crítica de los test-bed que debe ser tenida en cuenta:

- **Soporte a diferentes arquitecturas o plataformas de sensores**: si bien no es algo obligado, en según qué casos esto puede implicar una mayor complejidad en los experimentos debido a la posible inclusión de pasarelas.
- **Escalabilidad**: no todas las soluciones para redes de sensores necesitan esta propiedad, pero sin duda hay que tenerla muy en cuenta cuando se esté diseñando un banco de pruebas, dado que el coste de tiempo y recursos en introducir modificaciones al test-bed existente puede variar considerablemente.
- **Ejecución y depuración de programas**: normalmente cada nodo en la red tiene asignadas una serie de funciones, las cuales están controladas por programas software que tienen que ser cargados, depurados y modificados. Así pues, dependiendo de las capacidades de los nodos, esto debe de ser muy tenido en cuenta a la hora del diseño ya que pueden surgir los siguientes interrogantes:
    - ¿Pueden los sensores recibir programas software a través de la conexión inalámbrica? En caso afirmativo, aspectos a tener en cuenta sería el tiempo empleado y por tanto la energía consumida, sobre todo en test-bed cuyos nodos no estén conectados a una fuente de alimentación continua. En principio, según indica el autor, la reprogramación de los nodos en test-bed con gran número de ellos debería proporcionarse de manera paralela y no usando la propia red de sensores. Por ejemplo, una solución sería que los nodos dispusieran de otro interfaz, como el Ethernet, y usar una red cableada superpuesta para la carga de programas.
    - ¿Pueden los sensores enviar mensajes de depuración? Esto implicará al menos el poder enviar estos mensajes, o los resultados que normalmente se enviarían a la salida estándar o de error de un programa. Evidentemente, esto implica comunicaciones adicionales y produce una alteración en la duración del proceso bajo estudio, pero según qué casos, puede ser necesario en las fases de prueba del software.
- **Cambios de configuración remotos**: este es un aspecto fundamental para test-beds en los que los nodos estén muy dispersos o difíciles de acceder, y por esta razón cualquier cambio en su configuración debería hacerse de manera remota. En principio cualquier nodo suele permitir cambios en su configuración, y en caso contrario, se arbitrarían mecanismos a través del software diseñado para el test-bed. En particular, los aspectos más comunes a conocer de un sensor desplegado dentro de un test-bed, son su nivel carga de batería, las veces que un proceso concreto se ha iniciado o las medidas tomadas por una sonda instalada en el mismo. Además, suele ser útil el poder simular la pérdida de un sensor o de los mensajes enviados por éste, así como las posibles interferencias entre nodos, entre otras.
- **Mantenimiento de la red**: las acciones de reemplazo de nodos defectuosos o dañados deben de realizarse de manera controlada para evitar, en la medida de lo posible, los cambios en los experimentos. Aparte de lo que señala el autor acerca de un control de identificación basado en la posición, las acciones de mantenimiento deben ser también preventivas, así como incluir el reinicio de las red a un estado conocido después de un fallo, como la avería de varios nodos o la caída de los equipos de recepción de datos.

A pesar de estas consideraciones, los bancos de pruebas pueden diferir en muchos de los aspectos comentados, así como mostrar soluciones que tiendan más a modelar el entorno real donde se aplicará en un futuro la red de sensores, que a facilitar la carga de

aplicaciones o el mantenimiento del mismo. A continuación se mostrarán algunos de los bancos de pruebas de redes de sensores que han conseguido mayor repercusión según diferentes autores.

En la revisión realizada por Yick y otros (Yick et al. 2008) se mencionan tres test-beds, Orbit (Raychaudhuri et al. 2005), Emulab (Johnson et al. 2006) y MoteLab (Werner-Allen et al. 2005), sin duda tres modelos de laboratorios virtuales para la experimentación con redes de sensores. Aparte de las contribuciones mencionadas, los tres mantienen portales web donde presentan las capacidades de cada uno de ellos, ofreciendo servicios de carga de programas y pruebas remotas. Estos test-bed se comentan brevemente a continuación:

- **Orbit** está formado por una malla de dos dimensiones formada por 400 nodos con capacidad de comunicación wifi (802.11). Orbit está instalado y mantenido por el WINLab de la Universidad de Rutgers en Nueva Jersey.
- **Emulab**, además del test-bed original, ha exportado su idea existiendo en la actualidad más de una veintena de laboratorios con su arquitectura por todo el mundo. Emulab permite acceso remoto a través de internet (/www.emulab.net), cambios en la topología y elección de sistema operativo. Los nodos inalámbricos usan 802.11a/b/g, pero tiene una gran variedad de los mismos, tales como equipos PC con Windows o nodos virtuales. Sin embargo, en la actualidad el soporte para las redes de sensores no está disponible. Este soporte estaba basado en pequeños robots XScale con un sistema Startgate y un nodo Mica2 integrado. La documentación sobre ese test-bed puede verse (Johnson et al. 2006).
- **Motelab**, diseñado en la Universidad de Hardvard sigue la misma filosofía de los anteriores, capacidad de cargar programas de manera remota a los sensores, gran cantidad de nodos en interior y acceso a través de internet para la configuración y realización de pruebas. Los sensores usados eran MicaZ, aunque según la última referencia disponible (Imran et al. 2010), el número de nodos era de 190 sensores TMote Sky funcionando con TinyOS. Cabe destacar que al inicio de los trabajos de investigación de la presente tesis la web del test-bed estaba accesible en motelab.eecs.harvard.edu, sin embargo, en la actualidad ha sido desmantelado, concretamente desde enero de 2.014.

Aparte de estos tres test-beds, existen otros muchos que han obtenido diferente repercusión (Imran et al. 2010, Steyn & Hancke 2011), pero se han elegidos estos ya que son mencionados en todas las revisiones que se han hecho en los últimos tiempos, suponiendo un punto de partida para el diseño de muchos test-beds en las redes de sensores.

En la Tabla 3 se presenta el resumen aportado por Steyn en (Steyn & Hancke 2011) en el que se muestran algunos de los bancos de pruebas que se han dado a conocer en los últimos años.

Como ya se ha comentado, un test-bed no es la única herramienta para la prueba de nuevos protocolos o algoritmos en las redes de sensores, la simulación también está disponible y puede ser la mejor solución para algunos casos. Cabe destacar que en (Imran et al. 2010) aparece también una revisión de los simuladores más usados, tales como **NS2** (The Network Simulator ns-2) o **TOSSIM** (Levis et al. 2003).

Tabla 3. Bancos de prueba de redes de sensores.

Tipo de test-bed	Nombre	Resumen del hardware	Características especiales
Servidor central	Motelab 2005	190 nodos MicaZ y Tmote Sky. Conectados con placas Ethernet MIB600 como canal de respaldo.	Código abierto. Disponible online y fácilmente replicable en otras localizaciones
	Kansei 2006	210 nodos XSM y Tmote Sky. Conectados a través de Ethernet	Simulaciones con los datos del test-bed. Los principios de diseño pueden ser usados para test-bed autónomos altamente tolerantes a fallos
	ORBIT 2005	Un grid de 100 nodos desarrollados específicamente Conectados como canal de respaldo a través de Ethernet a múltiples servidores de test.	Soporta múltiples protocolos inalámbricos, nodos móviles y fuentes de interferencia RF. Disponible online
	Mobile Emulab 2006	35nodos MicaZ y Stargate. Conectados a través de placas Ethernet MIB500 y WiFi como respaldo	Código abierto. Disponible online y fácilmente replicable en otras localizaciones
Un solo PC	LabVIEW TB 2006	23 nodos MicaZ y Cricket programados independientemente a través del interfaz serie del PC	Una aplicación de LabVIEW que reduce la complejidad de la programación de nodos y visualizar los datos
	SignetLab 2007	48 nodos EyesIFXv2 a través de una configuración en hilera USB conectados con un PC	Nodos colgados del techo para ahorrar espacio. Software para controlar los nodos y los ajustes dinámicos de arquitectura
	Sensi-uu 2010	Mayormente nodos TelosB, pero también otros tipos de nodos conectados de manera inalámbrica a teléfonos (Smartphones) Android y Symbian	Soporte para software de teléfonos móviles. Puede ser desplegado incluso en nodos móviles Lego
Híbridos	TWIST 2006	Hasta 180 nodos TelosB y EyesIFX conectados a través de USB y Ethernet	El diseño permite cualquier sensor con USB y TinyOS. Control de la arquitectura a través de software
	IBM WSN TB 2006	Nodos desarrollados a media conectados a través de nodos pasarela a la red Ethernet empresarial	Es un ejemplo de usar el backbone de una gran red empresarial para conectar test-beds. También soporta Matlab
	MoteMaster 2009	Nodos TelosB conectados a PC de escritorio a través de USB. Usa la Ethernet existente para conectar con los servidores	Es un ejemplo de uso de software existente en PCs conectados con Ethernet para crear un test-bed. También tiene un interfaz para Matlab
	UTSB 2010	62 nodos Octopus II conectados a PCs de Gestión Distribuida a través de la red Ethernet y WiFi existente	Hardware y software especializado para conseguir consumes muy reducidos y medidas temporizadas
Multi-ubicación	X-Sensor 2009	Más de 178 nodos MicaZ desplegados en múltiples localizaciones conectados a un servidor central y nodos pasarela	Disponible online. Cuando los sensores no están en uso los datos se guardan para uso futuro
	Senslab 2010	Más de 1000 nodos especialmente desarrollados y desplegados en múltiples localizaciones conectados a un servidor central y nodos pasarela	Disponible online. Los usuarios son asignados a máquina virtuales para realizar experimentos. Dispone de un tren de nodos móviles
Gestión intra-banda	SenseNeT 2007	8 nodos MicaZ sin canal de respaldo para funciones administrativas	Usa software basado en nodos para hacer una gestión intra-banda a través de una red inalámbrica. Asequible, sin necesidad de hardware extra
Hardware especializado	TWiNS.KOM 2008	Tubicle tiene un SunSPOT, un Tmote Sky, dispositivo Bluetooth, antena WiFi, webcam y un conversor de serie a Ethernet	Es una solución hardware todo en uno. Soporta aplicaciones para móviles, multimedia y dispone de varias tecnologías inalámbricas
	HINT 2010	Nodos TelosB modificados conectados con places FPGA especialmente diseñadas para capturar datos a nivel primario (desde las patillas de los chips)	Medidas muy precisas de consumo de potencia con mínima interferencia por parte de la operación del nodo
Aplicación industrial	WINTeR 2008	32 nodos especialmente diseñados conectados con un servidor central. Fuentes de interferencia controlables	Es un test-bed en un entorno inalámbrico adverso con fuentes de interferencia electromagnética controlables remotamente
	TB para WSNs	Unos pocos nodos TelosB individualmente programados. PC central con una estación base conectada por puerto serie para el almacenamiento de datos	Desplegado en un canal inalámbrico de condiciones adversas. Software en Java para visualizar los datos del test-bed en 3D

Teniendo en cuenta lo expuesto acerca de los test-bed, y de los recursos necesarios para su despliegue, se intuye que el diseñar, implementar y desplegar un test-bed no es una

tarea sencilla, ya que requiere de un esfuerzo integrador que, aparte del trabajo de desarrollo e investigación, depende de otros factores como la búsqueda de ubicaciones físicas, posibles permisos o licencias, despliegue y protección de equipos, suministro de energía, mantenimiento, etc. Sin embargo, como se comentó al hablar de sus características, si se diseñan apropiadamente, un test-bed tiene la posibilidad de ser escalable y ajustarse a las necesidades de un experimento concreto, sin tener que ofrecer una capacidad de configuración total como otros grandes proyectos, tales como Orbit[8].

Así pues, como parte del presente trabajo de investigación, y como se desprende de sus objetivos, las pruebas que se realicen para la obtención de los resultados de las aportaciones de esta tesis serán realizadas en un test-bed. Dicho test-bed será diseñado e instalado a la par que se acometen los trabajos de investigación. Sus dimensiones y prestaciones estarán ajustadas a las capacidades del grupo de investigación, si bien contará entre sus características y funciones básicas con la posibilidad de acceso a determinadas funciones a través de Internet, carga remota de software y despliegue en un edificio real.

2.2 SISTEMAS MULTI-AGENTE EN REDES DE SENSORES

Los sistemas basados en agentes, y más concretamente, los sistemas basados en varios agentes que colaboran entre sí, o sistemas multi-agente (MAS, del inglés *Multi Agent-Systems*) han sido incorporados por algunos autores a las redes de sensores durante la evolución de las mismas en esta última década. Por lo tanto, se hace necesaria una revisión de los MAS, y su implicación en la Inteligencia Artificial (IA), antes de ver los aspectos más destacados de su aplicación en las redes de sensores. Así pues, a continuación se presenta una breve revisión sobre qué es un agente, aspecto central de la Teoría de Agentes, que es la que intenta unificar todo lo concerniente a este respecto, además de estudiar qué propiedades debería tener, motivaciones y de cómo éstas se representarían formalmente. A continuación se revisarán otros conceptos clave como son la estructura de un agente, arquitecturas de los MAS, los lenguajes de agentes y aplicaciones de los mismos.

2.2.1 TEORÍA DE AGENTES

No es hasta mediados de las década de los ochenta del siglo XX cuando la Teoría de Agentes es tomada en consideración, tanto en la IA, como en la ciencia computacional. Además, su aplicación no se ha restringido a estos dos ámbitos, comenzando a aplicarse en la investigación sobre ingeniería del software, comunicaciones y sistemas concurrentes (Wooldridge & Jennings 1995a).

Dentro de la Teoría de Agentes existen diversas definiciones, todas ellas más o menos coincidentes, aunque normalmente difieren entre sí en ciertos matices. La primera aproximación a la definición de agente en realidad parte de algo elemental, su definición en un diccionario, la cual, en términos generales, podría ser como sigue: “Persona o cosa que produce un efecto”[9]. De esta definición general de un agente, dentro de la Teoría de Agentes, se asume de manera intrínseca que estos son autónomos y racionales. Si bien la definición de autónomo, a pesar de la complejidad que pueda requerir en un sensor, se puede definir como la capacidad de actuar sin la acción directa de un humano, la racionalidad es algo más compleja de describir o acotar, aunque en

[8] Según su portal web, Orbit contaba con unos fondos de 5,45 millones de dólares para su creación, lo cual no está al alcance de cualquier grupo o investigador en redes de sensores.
[9] Definición según el Diccionario de la lengua española de la Real Academia Española.

(Wooldridge & Jennings 1995a) se asume la racionalidad como la capacidad de maximizar su funcionamiento respecto a una función de evaluación. Sin embargo, el autor remarca que esta definición es algo ambigua, mereciendo la asunción de más condicionantes, como el que los agentes sean sistemas de alto nivel (Shoham1993) en tanto en cuanto pueden recibir información, tener planes de actuación, respuesta ante estímulos concretos y la capacidad de comunicarse. De esta manera, los agentes se diferencian de otros sistemas de niveles inferiores como sondas, termostatos o los objetos de la programación orientada a objetos. Entroncando con la consideración de sistemas de alto nivel, aparecen otras características como que un agente es aquel capaz de llevar a cabo acciones (Moore1985), que es un sistema intencional (Cohen & Levesque 1990) o que se rige por creencias, deseos e intenciones (BDI, del inglés *Beliefs, Desires and Intentions*) (Rao & Georgeff 1991).

Realmente, la definición de qué es un agente y de las propiedades que debe tener, ha sido una discusión muy prolífica dentro de la bibliografía, como queda patente en la revisión ofrecida en (Franklin & Graesser 1997), y no todas incluyen específicamente el mismo número de propiedades. Sin embargo, queda claro en todas ellas, ya sea de manera explícita o implícita, que un agente debe tener cierto grado de autonomía, capacidad de interactuar con el entorno, así como de comunicarse, y todo ello gracias a un software específico. Más aun, la discusión sobre qué es realmente un agente, se mantiene hasta los inicios del presente trabajo de investigación, como se refleja en (Barandiaran et al. 2009), donde se vuelve a analizar el concepto de agente, revisando diversas definiciones, llegando a un acuerdo de mínimos, que no dista mucho de lo anteriormente comentado, ya que se llega a que un agente es algo que se distingue de su entorno, capaz de interactuar con el mismo y que lo hace para llevar a cabo una tarea o función concreta.

Por lo tanto, en el presente trabajo se han tenido en cuenta estas características para aplicar los MAS a las redes de sensores. Concretamente, se ha organizado la gestión interna de un sensor como un sistema multi-agente, formando una arquitectura dinámica de agentes dentro de cada sensor, para gestionar de manera autónoma las diferentes funciones del sensor. A continuación se revisarán algunos de los tipos de estructuras más usuales de los MAS, para después particularizar a las redes de sensores.

2.2.2 ESTRUCTURAS DE AGENTES

Así pues, una vez se ha llegado a una aproximación clara de qué es un agente, según antes se apuntó, es necesario ver la manera en la que se pueden construir estos agentes, así como la estructuras válidas para tal fin. Como ha sucedido con la definición de agente, con respecto a las estructuras para la creación de agentes sucede lo mismo, existiendo diversas vertientes y soluciones propuestas en la literatura.

Como punto de partida para el análisis, se tomará la clasificación realizada por Parunak en (Van Dyke Parunak et al. 2004), la cual puede definirse como de alto nivel y que claramente diferencia en primera instancia entre dos tipos básicos de sistemas de agentes: los sistemas de agentes *pesados* y *ligeros*, para luego analizar arquitecturas intermedias.

Los agentes pesados son aquellos, según el autor, que intentan emular en un solo agente el comportamiento lo más parecido a un ser humano posible. Así pues, a estos sistemas de agentes se les atribuyen creencias, deseos o intenciones, además de poseer un lenguaje próximo al lenguaje natural que facilita la interacción con ellos. Este intento de

responder como humanos a través de agentes gobernados por reglas lógicas es una visión típica de la IA. Con respecto a la aplicabilidad, Parunak reseña la complejidad de poner en funcionamiento este tipo de sistemas dada la magnitud de software, la indeterminación provocada por la imposibilidad real de responder a todas las cuestiones por la lógica, además de tender su funcionamiento a degradarse rápidamente.

Por otro lado, los agentes ligeros, provienen de la rama de Vida Artificial, por lo que asumen a cada agente un comportamiento mucho más simple que el de un humano, como por ejemplo el de los insectos. En este ejemplo, a un solo individuo no se le supone la capacidad de actuar como un humano, pero un conjunto suficientemente grande y coordinado de ellos puede llevar a cabo tareas mucho más complejas. Así pues, la programación de este tipo de agentes se puede simplificar, pudiendo consistir en la elección de una serie de parámetros que definen normalmente el proceso bajo estudio y algún tipo de función de evaluación (como las evolutivas). Por lo tanto, debido a que los agentes ligeros son eminentemente procesos matemáticos, generalmente se consigue una mayor eficiencia computacional. Sin embargo, el hecho de basarse su funcionamiento en una parametrización de funciones, hace más compleja su interpretación por las personas, así como el análisis de su comportamiento, el cual, en su mayor parte, no depende de un solo individuo sino de un conjunto.

Además, estas dos clasificaciones pueden combinarse para dar soluciones híbridas de agentes, que combinan los agentes ligeros con los pesados, como por el ejemplo, el uso de agentes en enjambre para resolver problemas como subrutinas de un agente pesado (Van Dyke Parunak et al. 2004).

En las arquitecturas multi-agente, aparte de cómo es en sí cada uno de los agentes, es igualmente importante la relación de cada uno de ellos con el entorno, así como con los demás agentes. Además, según Hoek y Wooldridge (Van der Hoek & Wooldridge 2008), existen dos posibilidades en los MAS en cuanto a la finalidad de los mismos: que todos los agentes compartan el mismo objetivo, o por el contrario, que no lo compartan. En el primer caso se tendría por ejemplo un MAS interesado en controlar la intrusión de personas en un recinto restringido, mientras que en el segundo, podría estar un sistema dentro de un vehículo, que por un lado intenta conseguir la máxima velocidad ajustando elementos aerodinámicos, y por otro el mínimo consumo controlando aspectos como el par entregado a las ruedas por la caja de cambios. Aunque en conjunto ambos agentes se les suponen que comparten un objetivo común, el movimiento eficiente del vehículo, queda patente que sus objetivos particulares son muy diferentes.

Así pues, el siguiente paso, según Hoek y Wooldridge, es cómo llegar a representar y definir los entornos con sistemas multi-agente, para lo cual se puede distinguir entre dos tendencias, los *modelos cognitivos de acción racional* y los modelos de *estructura estratégica*. Los primeros se basan en la atribución de actitudes a los agentes, tales como sus creencias, aspiraciones, intenciones y aspectos similares. El objetivo de este tipo de definiciones es que éstas deriven en un modelo que prediga cómo el agente actuará según sus actitudes.

Por otro lado, los modelos de estructura estratégica no se centran tanto en cómo es un agente por dentro, sino en cómo puede afectar un agente al entorno para cumplir su

tarea, ya sea solo o en conjunto. Estas estrategias suelen provenir del entorno de la Teoría de los Juegos[10].

Si se tienen en cuenta ambas tendencias por sí mismas, es sin duda la segunda la que más puede aproximarse a lo que representa una red de sensores. Sin embargo, no se puede desdeñar la primera, teniendo en cuenta que las capacidades de los nodos están en aumento, y su forma de actuar podría también asimilarse a un agente basado en actitudes. Además, la hibridación entre ambas tendencias es también posible, como se ha visto anteriormente para los agentes pesados y ligeros, que están estrechamente relacionados con esta última clasificación.

A pesar de que la discusión sobre los aspectos filosóficos que rodean la Teoría de Agentes se aparta del objetivo de esta tesis, merece la pena hacer algo más de hincapié en la representación de los estados cognitivos racionales de un agente, ya que se han tenido en cuenta algunos de estos aspectos en el diseño de los agentes de la arquitectura presentada en el presente trabajo de investigación. Por lo tanto, para caracterizar a un agente a través de sus actitudes, éstas se pueden dividir en tres categorías (Van der Hoek & Wooldridge 2008):

- **Actitudes de información**: Estas son las que permiten al agente obtener información acerca del entorno. Algunas de las actitudes que están dentro de esta categoría son el conocimiento y las creencias.
- **Actitudes de intención**: Son las que guían a un agente para la consecución de sus objetivos. De esta manera se tienen a las metas, deseos e intenciones dentro de esta categoría.
- **Actitudes de normativa**: Tales como obligaciones, permisos y autorización.

La elección de, cuáles de estas actitudes son fundamentales y de cómo es la relación entre ellas, no es algo obvio, existiendo en la literatura diversas combinaciones. Sin embargo, la mayoría de los formalismos, según Hoek y Wooldrige, toman en conjunto el conocimiento y las creencias, uniéndolos al menos con las metas y los deseos.

A continuación se enumeran cuatro de los formalismos más conocidos sobre el razonamiento de los agentes racionales. Sin embargo, no es objetivo de este trabajo su revisión en profundidad ya que son aspectos básicos que aparecen reflejados con claridad en la bibliografía.

- Lógica Epistemológica Dinámica, del inglés *Dynamic Epistemic Logic* (DEL). La idea básica de esta tendencia es la de añadirle una componente dinámica (Harel et al. 2000) a la lógica epistemológica (Fagin et al. 1995).
- Lógica intencional transcendental de Cohen y Levesque (Cohen & Levesque 1990).
- Plataforma BDI (*Beliefs-Desires-Intentions*) de Rao y Georgeff (Rao & Georgeff 1991).
- Plataforma KARO de Linder y otros (Linder et al. 1994).

[10] La Teoría de Juegos estudia de manera formal y abstracta, las decisiones óptimas que deben tomar diversos adversarios en conflicto, a través del estudio de modelos matemáticos que describen el conflicto y la cooperación entre entes inteligentes, "jugadores", que toman decisiones. Se reconoce al matemático John von Neumann y al economista Oskar Morgenstern como los precursores de La Teoría de Juegos actual con la publicaron de su libro *The Theory of Games Behavior* en 1944.

2.2.3 ARQUITECTURAS MULTI-AGENTE

A lo largo de estos últimos años han aparecido diversas arquitecturas multi-agente que han intentado llevar a la práctica la Teoría de Agentes. A pesar de la heterogeneidad de soluciones, se puede llegar a la conclusión de que predominan dos grandes categorías de arquitecturas, las Deliberativas y las Reactivas, además de las resultantes de la hibridación entre ambas. En concreto en (Wooldridge & Jennings 1995a), se toma a las arquitecturas deliberativas como las arquitecturas clásicas, mientras que las reactivas aparecen como alternativas. Sin embargo, poco después, Stone y Veloso en (Stone & Veloso 2000), recogen a ambas como primer nivel en la taxonomía realizada por los autores para las arquitecturas multi-agente.

Las arquitecturas deliberativas basan el comportamiento y el conocimiento sobre el entorno en una serie de reglas simbólicas. Por lo tanto, el éxito de este tipo de arquitecturas se fundamenta en la resolución de dos importantes problemas:

- El problema de **la traducción**, es decir, el traducir el entorno a controlar de manera adecuada y precisa usando una representación simbólica lo suficientemente rápida para que ésta sea útil.
- El problema de **la representación/razonamiento**, o de cómo se puede representar de manera simbólica la información que se obtiene del entorno y como conseguir que los agentes obtengan un razonamiento en un tiempo asumible.

Estos problemas, cuya formulación es sencilla, encierran una gran complejidad, por lo que algunos investigadores han buscado alternativas en lo que se conocen como arquitecturas reactivas.

Básicamente, las arquitecturas reactivas se pueden definir como aquellas que no incluyen un modelo central de representación simbólica del entorno, ni poseen un razonamiento simbólico complejo (Brooks1991). Aunque fundamentalmente Brooks fue un crítico acérrimo de la IA tradicional, y no llegó a tener muchos adeptos, sí consiguió demostrar su teoría a través una serie de experimentos con robots basados en lo que él denominaba “arquitectura de subsunción”. Dicha arquitectura se basa en una jerarquía de “comportamientos” que no son más que sencillas tareas con objetivos claros. Cada uno de esos comportamientos compite con los demás para ejercer su control sobre el robot. Las capas inferiores de la jerarquía modelan comportamientos más primitivos y prevalecen sobre capas superiores. Así pues, con esta arquitectura, con un coste computacional extremadamente bajo, sin un razonamiento explícito y sin patrones, consiguió comportamientos equiparables a los conseguidos por los sistemas de IA simbólicos. De esta manera, Brooks dejó patente en sus trabajos: que la inteligencia está en el mundo, no en sistemas incorpóreos que resuelven teoremas; y que la inteligencia no es algo innato y aislado, sino que el comportamiento “inteligente” aflora como resultado de la interacción con el entorno.

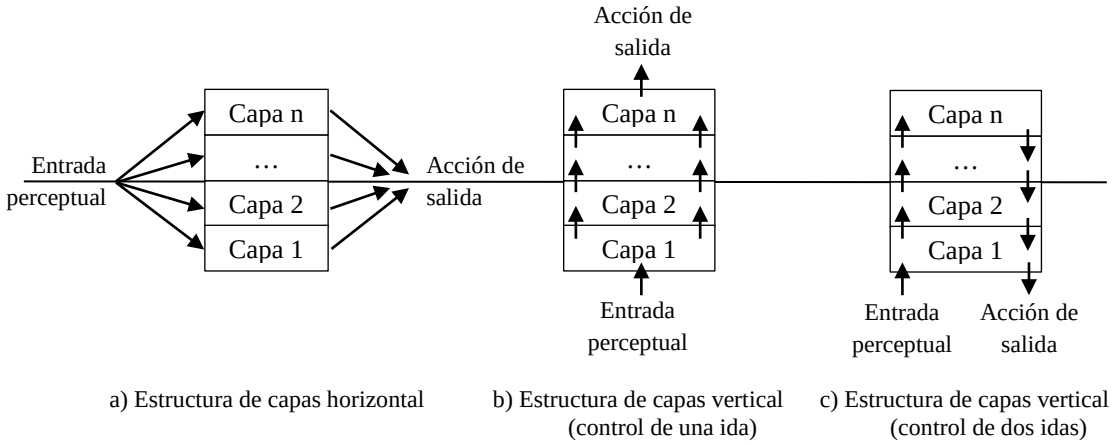
Aproximadamente a la par del trabajo de Brooks, Chapman, como parte de su tesis doctoral, analizó las dificultades de la planificación tradicional de la IA. Más tarde, con su colaborador Agre (Chapman & Agre 1987), comenzaron a explorar otras alternativas. Agre determinó que la mayoría de las tareas diarias, una vez aprendidas, se convierten en rutinas, las cuales pueden ser acometidas con pocas variaciones. Sus investigaciones fueron plasmadas en un videojuego simulado, Pengi (Agre & Chapman 1987), cuyo personaje central se controlaba según sus teorías.

Otros autores que contribuyeron a las arquitecturas reactivas fueron Rosenschein y Kaelbling con su paradigma del autómata ubicado (Rosenschein & Kaelbling 1986, Kaelbling1987, Kaelbling & Rosenschein 1990). Este agente se especificaba con términos declarativos y posteriormente esta especificación se compilaba en una máquina digital. La lógica usada era esencialmente lógica del conocimiento, usando para la especificación del agente dos componentes, percepción y acción (Kaelbling & Rosenschein 1990).

Las arquitecturas reactivas tienen ciertas ventajas con respecto a las deliberativas, tales como la respuesta inmediata del agente, además de no necesitar una representación simbólica del entorno. Evidentemente, estas arquitecturas también presentan inconvenientes (Jennings & Wooldridge 2002), como la dificultad de diseñar agentes puramente reactivos que puedan aprender de la experiencia, así como la complejidad de diseñar las interacciones entre agentes con muchas conductas.

Teniendo en cuenta las características de estas dos arquitecturas, algunos investigadores han desarrollado soluciones híbridas, las cuales no son puramente ni deliberativas, ni reactivas. Estas iniciativas intentaron aunar los mejores aspectos de ambas arquitecturas. En los trabajos sobre arquitecturas híbridas normalmente se realiza la descomposición en subsistemas para cada comportamiento diferente, dando como resultado estructuras basadas en capas (Wooldridge2009, Jennings & Wooldridge 2002). Esta división en capas, debe tener al menos dos de estas capas, una para los comportamientos proactivos (deliberativos) y otra para los reactivos, aunque el número de éstas no tiene por qué estar limitado.

Las estructuras en capas se dividen en horizontales y verticales. La división de capas horizontal propicia una conexión directa de cada capa con la entrada sensorial y con la acción de salida. Por el contrario, las estructuras en capas verticales, la entrada de los interfaces con el entorno, y la salida de acción está, a lo sumo, conectada con una capa. Esto queda de manifiesto en la figura siguiente.



a) Estructura de capas horizontal

b) Estructura de capas vertical (control de una ida)

c) Estructura de capas vertical (control de dos idas)

Figura 1. Control del flujo de información para tres tipos de arquitectura por capas para agentes. Fuente (Müller et al. 1995).

Según apunta Müller (Müller et al. 1995), las arquitecturas horizontales, además de necesitar una unidad central de control, presentan un problema fundamental de escalabilidad. Esto se debe a que las relaciones bilaterales entre cada capa se hacen más

complejas conforme aumenta el número de éstas, ya que cada una de ellas compite por prevalecer en la acción de salida. Por esta razón se propusieron las estructuras verticales, donde se imponen restricciones a las relaciones entre cada capa, ya sea en las de control de una ida, Figura 1 b), o control de dos idas, Figura 1 c).

Como puede deducirse en primera instancia, las arquitecturas reactivas se aproximan más a lo que una red de sensores podría acometer, con cada sensor generando trazas de conocimiento a través de la ejecución de programas sencillos, que en su conjunto permitirían llevar a cabo tareas mucho más complejas. Sin embargo, cabe pensar también que la hibridación ente arquitecturas deliberativas y reactivas puede ser apropiada para algunas aplicaciones de redes de sensores, teniendo en mente además, que la separación en capas de las arquitecturas de agentes permitiría una configuración más flexible de los nodos.

2.2.4 LENGUAJES DE ESPECIFICACIÓN Y COMUNICACIÓN ENTRE AGENTES

Por lenguajes de agentes se entiende al sistema que permite la programación de hardware o software a partir de algunos de los elementos, o características, propios de las diferentes teorías de agentes (Wooldridge & Jennings 1995a). A su vez, el autor también destaca que la separación entre arquitecturas de agentes y lenguajes es artificial, pudiéndose contar las arquitecturas antes vistas como lenguajes en sí mismos.

Incluso en los primeros años de investigación sobre los MAS, diversos grupos de investigadores, en un esfuerzo de estandarización crearon una amplia variedad de lenguajes de especificación de agentes que aportaban su propio entorno de desarrollo como JADE (Bellifemine et al. 1999), del inglés *Java Agent DEvelopment Framework*, uno de los sistemas con más tradición dentro de los MAS y totalmente activo[11].

Otros lenguajes de especificación de agentes aparecen revisados en (Railsback et al. 2006), centrándose en plataformas para modelos científicos basados en agentes (ABMs, del inglés *agent-based models*). Entre estos están Swarm (Minar et al. 1996), RePast (Collier2003), MASON (Luke et al. 2005) y NetLogo. Swarm es uno de las primeras plataformas de desarrollo y simulación de sistemas multi-agente. Proporciona una serie de bibliotecas de funciones escritas en Objective-C y que también dispone de bibliotecas para Java (Java Swarm). RePast surgió como una variante de las bibliotecas Java de Swarm, pero se ha desligado lo suficiente de su origen como para ser tenida en cuenta separadamente, según argumenta Railsback en (Railsback et al. 2006). Mason también está basado en Java, y surge de la necesidad de realizar gran número de simulaciones por parte de sus desarrolladores. Mason es un sistema de un solo proceso, compacto y portable, con la visualización separada de la simulación, especialmente orientado a los problemas de enjambres de agentes (*swarms*). NetLogo es una plataforma de programación y simulación de sistemas multi-agente con fines educativos. Desarrollada en Java, está orientada a la simulación de fenómenos naturales y sociales. Posee un lenguaje propio de la familia de Lisp y es una evolución de StarLogo (Resnick1996), también destinado a entornos educativos.

Merece la pena destacar que todas estas plataformas están en activo y accesibles a través de Internet, además de poder ser utilizadas sin coste. Aparte de éstas, en (Nikolai & Madey 2009) aparece una amplia revisión de plataformas y lenguajes multi-agente, clasificada por varias categorías. La taxonomía presentada por Nikolai da una idea de la complejidad, así como de la gran cantidad de soluciones existentes para la creación y

[11] La última versión disponible es de junio de 2012, pudiéndose encontrar online en jade.tilab.com.

simulación de sistemas multi-agente. Entre ellos, merece la pena destacar ABLE (Bigus et al. 2002) (del inglés *Agent Building and Learning Environment*) desarrollado por IBM, también basado en Java (usa la tecnología JavaBeans para facilitar la visualización por terceros sistemas). Está especialmente diseñado para aprendizaje y razonamiento artificial basado en reglas. Sin embargo, la última versión, aunque aún disponible, data ya del año 2004, si bien sigue siendo la base para nuevos trabajos al igual que JADE u otras plataformas, como queda de manifiesto en (Ivanović et al. 2012).

Por otro lado, cuentan los esfuerzos por estandarizar los lenguajes de comunicación entre agentes, como los de la *Foundation for Intelligent Physical Agents* (FIPA) (O'Brien & Nicol 1998). Este esfuerzo integrador dio como resultado la especificación de un lenguaje de comunicación entre agentes o ACL (del inglés *Agent Communication Language*), que ha sido usado por diferentes autores como base para sus desarrollos en sistemas multi-agente. A pesar de que la última revisión sea del año 2002, y tras una inspección de la actividad en la web oficial de FIPA no se encuentren documentos más allá de 2005, la especificaciones generadas por sus grupos de trabajo sí siguen incorporadas a lenguajes como el antes mencionado JADE (Bellifemine et al. 1999), cuyos trabajos se prorrogan en el tiempo más allá de la última revisión de la especificación de ACL de la FIPA (Bellifemine et al. 2008), o también la revisión de (Ahmed et al. 2009). Entre los lenguajes comentados por Ahmed aparece uno de los lenguajes de comunicación entre agentes más extendidos: KQML (Finin et al. 1994) (del inglés *Knowledge Query and Manipulation Language*). KQML es compacto y versátil, con un formato de delimitación de mensajes con paréntesis que permite que los agentes envíen información sin tener que usar un formato concreto para la parte de contenido, facilitando así la aplicación a diferentes arquitecturas.

Como se puede intuir, y a pesar de que el número de estos lenguajes y entornos de desarrollo es considerable, implementar un MAS sigue siendo una tarea desafiante.

2.2.5 APLICACIÓN DE LOS SISTEMAS MULTI-AGENTE A LAS REDES DE SENSORES

Como se ha podido observar a lo largo de los apartados anteriores, los sistemas basados en agentes y los sistemas multi-agente, como tecnología de inteligencia computacional, es claramente multi-disciplinar, con una gran presencia dentro de la IA en estos últimos años, así como en otras ramas muy diversas, tales como el comercio y la economía (Kauffman & Walden 2001), la ingeniería de producción y control (Daneshfar & Bevrani 2009) o la ingeniería del software (Madejski2007). De entre estas áreas aparece la aplicación de los agentes y los sistemas multi-agente a las redes de sensores, muy ligados con las soluciones de inteligencia ambiental (Weyns et al. 2005).

Las redes de sensores, son sin duda un entorno que puede parecer propicio para la aplicación de los MAS (Tynan et al. 2005). En (Vinyals et al. 2011), de hecho, los autores aportan algunas de las características que hacen a las redes de sensores apropiadas para su integración con sistemas multi-agente, más concretamente en aplicaciones de monitorización. Estas características aparecen comentadas a continuación:

- Son sistemas no invasivos, dado que los nodos de una red de sensores traen incorporadas las sondas y/o actuadores que sirven para interactuar con el entorno, evitando cualquier modificación en el sistema a monitorizar. Esto es

posible debido a que los nodos suelen disponer de sistemas autónomos de alimentación, así como chips de comunicación inalámbrica.
- Pueden cubrir áreas extensas gracias a su bajo coste.
- Gracias a su naturaleza distribuida pueden ser sistemas muy tolerantes a fallos, con una adecuada redundancia, además de poder ser desplegados en entornos hostiles.
- Su versatilidad para aunar diferentes procesos de obtención de datos, ya sea en contenido, resolución y precisión.

Sin embargo, para que una red de sensores sea a la vez un sistema multi-agente, deben concurrir ciertas condiciones, tal y como manifiesta Karlsson (Karlsson et al. 2005), donde un nodo puede tener un comportamiento inteligente y autónomo, como muchos casos de aplicaciones de redes de sensores, pero sin embargo, para que sea considerado un sistema multi-agente, es necesario que exista un lenguaje de comunicación de alto nivel entre los agentes, además de presentar las características típicas de los agentes: autonomía, pro-actividad, colaboración y movilidad. Estas características son asimilables a un nodo de una red de sensores, pero cabe destacar que, dentro de la Teoría de Agentes, la movilidad se enfoca de una manera diferente y lo que tradicionalmente es la movilidad de los nodos en una red. Así pues, los agentes deben ser capaces de migrar de un sistema anfitrión a otro, que es lo que se conoce como movilidad de un agente.

A pesar de los ejemplos existentes, siguiendo con la discusión sobre la idoneidad de las redes de sensores para la implementación de los MAS, existen autores como Rogers, Jennings y Corkill que en su trabajo (Rogers et al. 2009) ponen, como principal problema para la integración directa de los MAS tradicionales en las redes de sensores, las limitadas capacidades computacionales de éstas. Además, apuntan que las comunicaciones entre nodos pueden ser lentas e intermitentes, y que la posible falta de disponibilidad de los sensores, ya sea por estar estos en modo de ahorro de energía o por tener agotada su batería, complican su diseño. Por otro lado, la necesaria escalabilidad de algunas soluciones, puede ser muy compleja de acometer o incluso imposible. Adicionalmente, los entornos pueden ser altamente variables y por ende, difícil de predecir el comportamiento de los nodos. Sin embargo, no es algo irrealizable, presentando en el mismo trabajo una revisión de tres implantaciones reales que adaptaron con éxito ambas tecnologías. Más aun, en (Rogers2011), Rogers recorre, en una breve revisión posterior, otros diversos casos de éxito que combinan ambas tecnologías. Los casos analizados se basan en agentes que gestionan el tiempo de sueño de los sensores con la toma de datos del exterior, y otras dos de control de condiciones atmosféricas.

Así pues, examinando (Vinyals et al. 2011), los autores proponen una taxonomía de las contribuciones de los MAS dentro de las redes de sensores (Figura 2). En dicha clasificación, las características más complejas aparecen señaladas en la figura en negrita, vinculándolas a uno o varios de los desafíos a los que se tienen que enfrentar las redes de sensores, los cuales han sido comentados anteriormente.

Estas características se deben tener en cuenta en cualquier nuevo diseño que pretenda vincular tanto los MAS, como las redes de sensores. Estos aspectos ya se han comentado en parte de esta sección, pero la clasificación presentada en la Figura 2 merece una especial atención y serán revisados a continuación debido a que servirán de guía para los ejemplos de aplicación que se comentarán después.

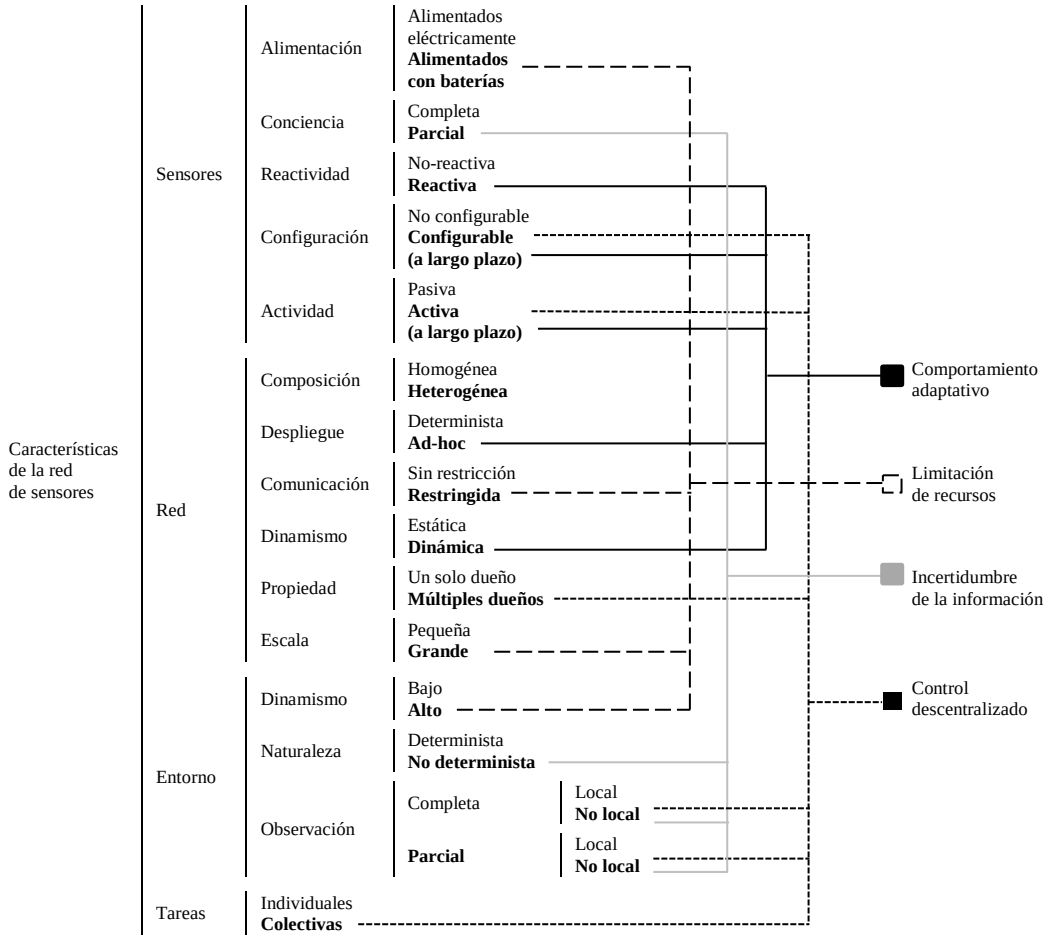


Figura 2. Taxonomía de la aplicación de los MAS a las redes de sensores. Fuente (Vinyals et al. 2011).

Por lo tanto, partiendo de los propios sensores, las características de estos que hace más compleja la aplicación son, en principio, aquellas que añaden incertidumbre a su funcionamiento: disponer de una batería de carga limitada, no tener un completo acceso a todas sus propiedades, ser desplegado en un entorno cambiante y establecer que el sensor se adapte al mismo, necesitar configuraciones a largo plazo, además de que los sensores tengan comportamientos activos, en vez de meros receptores de información. Así pues, la indeterminación del funcionamiento de un sensor (batería, entornos, reactividad), junto con la necesidad de hacer planificaciones a largo plazo, ya sea en la configuración, como en la forma de actuar, hacen que en la integración con un sistema multi-agente se tengan que tener en cuenta todos estos aspectos, y buscarles una solución. La posibilidad de que un nodo no esté disponible, que se vea afectado por efectos meteorológicos que lo desubiquen, lo dañen o destruyan, aparte de la dificultad de encontrar parámetros estables y todas las reacciones posibles a los estímulos externos, complican la especificación de agentes que tengan que responder de manera inteligente a todos estos avatares.

Con respecto a las características de las redes de sensores que dificultan la integración de los MAS con éstas, están la posible heterogeneidad de los dispositivos que las forman, los despliegues no deterministas, bastante usuales en aplicaciones militares, o aquellas que se hacen en entornos naturales, como la protección contra incendios; además, la presencia de canales de comunicación con bajo ancho de banda o de alto coste, harían que las aplicaciones de lenguajes de comunicación de agentes de más alto nivel fueran inviables. A esto hay que añadir la variabilidad en los enlaces de los nodos,

ya sea por cambios en el entorno, interferencias o caídas de nodos, que obligan a usar topologías adaptativas, complicándose aún más en redes donde concurran varios propietarios, así como en redes donde el número de nodos sea muy alto, desde cientos a miles, que hacen que la posibilidad de un control centralizado sea prácticamente inviable.

El tercer elemento que es necesario tener en cuenta es el entorno. Aunque en principio el modelado de éste es uno de los objetivos típicos en los sistemas multi-agente (Weyns et al. 2005), la influencia que el entorno puede tener en una red de sensores añade mayor complejidad, al hecho ya difícil de por sí, de modelar los acontecimientos naturales o artificiales bajo estudio. Así pues, siguiendo la taxonomía de Vinyals y otros, las características de los entornos, que afectan más adversamente a la integración de MAS y redes de sensores son: la alta variabilidad de estos, la presencia de alto número de efectos no deterministas y la existencia de fenómenos parcial o totalmente no observables. Estos tres factores generan un alto grado de incertidumbre a la hora de diseñar la red de sensores, necesitándose por lo tanto sistemas más complejos que puedan prever aquello que no es posible detectar ni predecir de antemano.

Por último, el elemento que queda por evaluar en las redes de sensores son las tareas encargadas a estos sistemas. En este caso la complejidad viene marcada por la necesidad de acometer estas tareas de manera colectiva, y no individualizada por cada sensor. Las tareas que requieren un trabajo coordinado suponen un esfuerzo adicional en coordinación, y por lo tanto en comunicaciones.

Como ha podido observarse, la integración de los sistemas multi-agente en redes de sensores que cumplan con varias de las características vistas anteriormente, dificultará en mayor medida el diseño e implementación de los mismos. Esto abre un campo muy extenso de trabajo e investigación tanto en los MAS, como en las redes de sensores, que en parte se beneficiará de la mejora en las prestaciones del hardware, pero que siempre contará con sistemas de recursos limitados que deberán acometer tareas complejas de supervisión, vigilancia, coordinación, seguridad, control de procesos, etc.

Dentro de las soluciones de aplicación de los MAS a las redes de sensores, los apartados siguientes se centrarán en aquellas que usan los sistemas multi-agente para la gestión interna de un sensor, toma de medidas, así como los procesos de ciclo de trabajo y comunicaciones. Para encontrar una revisión de otras aplicaciones, tanto en (Vinyals et al. 2011) como en (Vukasinovic et al. 2012), aparecen diversas aplicaciones en una amplia variedad de campos dentro de las redes de sensores.

Una de las aplicaciones de los sistemas multi-agente de las antes enumeradas, y que tiene una especial vinculación con los objetivos del presente trabajo de investigación, es la toma de medidas controlada de manera inteligente con cada nodo. Por ejemplo, en (Padhy et al. 2006), en una red de sensores desplegada para controlar un glaciar, cada nodo es un agente encargado de estimar el momento en el que se ha de realizar sus próximas medidas en función de la batería y de un modelo de regresión lineal en una ventana limitada. Además, también modela un sistema multi-agente para calcular el coste de las comunicaciones para encontrar la mejor ruta a utilizar para un mensaje desde el nodo origen hasta la estación base. En este caso la red es fija y no existe interactividad con el entorno, aunque debe enfrentarse a condiciones atmosféricas adversas, así pues, el esquema de predicción usado podría ser comparable con los modelos basados en FRBS propuestos en el presente trabajo de investigación.

Aiello y otros presentan en (Aiello et al. 2009), una arquitectura multi-agente denominada MAPS (*Mobile Agent Platform for Sun SPOT*), que combina las redes de sensores y los MAS, usando concretamente la plataforma Sun SPOT. Esta arquitectura, programada en lenguaje Java, está basada en una serie de subsistemas que controlan la ejecución de tres agentes móviles: aplicación, middleware y red. Así pues, se demuestra que es viable una implementación de sistemas multi-agente, concretamente se definen como agentes ligeros, en los sensores Sun SPOT, que como ya se había comentado, tienen unas prestaciones adecuadas para la integración de sistemas con cierta complejidad, como los sistemas multi-agente. Sin embargo, no se detalla, en el trabajo de Aiello, las funciones concretas que llevan a cabo los agentes, si bien sí se muestra su arquitectura interna basada en eventos y máquinas de estados finitos. Por lo tanto, el presente trabajo de investigación, pretende aportar una nueva perspectiva en las estructuras internas de los agentes al tener estos elementos de decisión basados en FRBS. En un trabajo más reciente (Aiello et al. 2011), los autores aplican la arquitectura MAPS a una red de sensores corporal WBSN (del inglés *Wireless Body Sensor Nerwork*). En este caso, la arquitectura está formada por un coordinador basado en JADE y dos nodos Sun SPOT (en cintura y muslo) con MAPS para el control de posturas de una persona.

Otra solución reciente, Self-StarMAS, presentada por Ayala en (Ayala et al. 2012a), está enmarcada dentro de la Inteligencia Ambiental, como ya se ha visto, una importante área de aplicación de las redes de sensores y los MAS. En este trabajo presentan los inicios de una solución basada en agentes ligeros sobre dispositivos heterogéneos, entre ellos redes de sensores (mencionando la plataforma Sun SPOT). Este sistema es comentado con más detalle en (Ayala et al. 2012b). En ese trabajo la autora describe una aplicación para el apoyo a un geriátrico con el sistema Self-StarMAS empleando terminales móviles (teléfonos) compatibles con la plataforma Android o MIDP [12] (del inglés *Mobile Information Device Profile*) y sensores Sun SPOT. La principal característica de esta arquitectura es que proporciona una gestión homogénea a pesar de disponer de diversos tipos de agentes, así como diferentes plataformas hardware. Sin embargo, la gestión de ahorro de batería que presenta la autora se realiza a través del ajuste del periodo de muestreo usando valores prefijados, y no a través de un sistema adaptativo.

Para completar con esta revisión sobre la aplicación de los MAS a las redes de sensores, se presenta en la Figura 3 la taxonomía descrita en (Dagdeviren et al. 2011), donde se hace una división entre agentes software móviles, agentes hardware móviles y nodos como agentes.

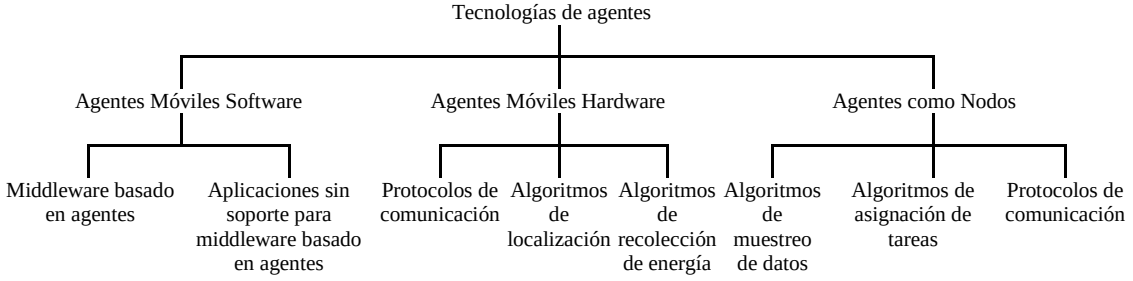


Figura 3. Clasificación de tecnologías de agentes en redes de sensores.

[12] Plataforma software de la versión de Java para dispositivos móviles Java 2ME.

Dentro del análisis realizado por los autores, se tomará en consideración tan sólo el caso concreto de sensores como agentes, dado que está relacionado estrechamente con los objetivos del presente trabajo de investigación. Así pues, las aplicaciones más usadas para este tipo de sistemas multi-agente son: el muestreo del entorno, la construcción de la arquitectura de red, así como la asignación de tareas, quedando como tareas abiertas para la investigación, el enrutamiento energéticamente eficiente y la planificación de rutas. Sin embargo, este tipo de aplicaciones tienen la desventaja ya comentada de tener que ser implementadas en dispositivos con recursos reducidos. Así pues, de entre los trabajos revisados está (Kho et al. 2009), el cual presenta un sistema descentralizado de ajuste del tiempo de muestreo adaptativo a través del uso de la Cantidad de Información de Fisher y regresión gaussiana, aplicado a un problema de detección de inundaciones. Al igual que otros casos que se revisarán a continuación, esta solución no usa técnicas de lógica borrosa, lo cual contribuye a respaldar la novedad de los objetivos del presente trabajo de investigación.

Como ha podido observarse, la cantidad de aplicaciones es amplia y diversa, si bien la conjunción de sistemas borrosos junto con la gestión de sistemas multi-agente en redes de sensores, aun contando con contribuciones, la aplicación concreta de FRBS a elementos de decisión interna a los agentes no está apenas explotada. Existen ejemplos como (Shamshirband et al. 2010) donde los nodos desplegados son meros captadores de datos del entorno, que posteriormente se envían a un nodo central que se encarga de su procesado, aparte de que la consideración como sistema multi-agente tradicional es discutible. También el trabajo de Martyna (Martyna2006) incluye una combinación de técnicas de lógica borrosa, en este caso para el aprendizaje reforzado (algoritmo *Q-learning*) en el enrutamiento en redes de sensores, donde cada sensor se define como un agente. Se destaca que, si bien el uso de las propiedades de un agente aparecen un poco más presentes, no se detallan sus características dentro de la teoría general de agentes, en parte esto se puede deber a la necesidad de limitar éstas, dadas las restricciones que presenta la integración en un nodo de una red de sensores.

Por último se mostrarán algunas de las soluciones middleware que integran agentes en redes de sensores, estas plataformas aparecen revisadas por Younge en (Younge2007), el cual muestra los pros y contras de todas ellas. En primer lugar está JADE (Bellifemine et al. 2008), ya comentada, la cual es compatible con la versión de Java para dispositivos móviles Java 2 Mobile Edition (J2ME) con el perfil 1 (MIDP 1.0), muy extendida en dispositivos como teléfonos móviles y sensores como los Sun SPOT.

Agilla (Fok et al. 2005, Fok et al. 2009) es otro middleware implementado para funcionar sobre sensores MICA2 y TelosB con TinyOS. Los desarrolladores de Agilla afirman que es el primer middleware de agentes móviles que funciona sobre dispositivos de prestaciones reducidas como los nodos de una red de sensores. Agilla presenta especiales capacidades para la movilidad de los agentes a través de la concepción de espacios de memoria basados en tuplas (Gelernter1985), que se comportan como compartimentos estancos para cada agente, lo que facilita la movilidad de estos en busca del lugar de la red donde puedan acometer con mayor eficacia su tarea.

Otro middleware revisado por Younge es Lime (siglas de las palabras en inglés *Linda in a Mobile Environment*) (Murphy et al. 2001), una evolución de Linda (Gelernter1985) que aporta movilidad basándose en el concepto antes comentado de tuplas en memoria.

Sin embargo, según Younge, Lime no simplifica la creación de agentes, limitando muy seriamente la aplicabilidad del mismo.

Por último, Impala (Liu & Martonosi 2003), pensado para el control de la fauna en campo abierto, se implementa sobre un micro-controlador con sensor de temperatura y GPS. Impala proporciona una completa capa de abstracción sobre la que los agentes pueden ser implementados de una manera sencilla y sin necesitar grandes recursos. Estos middleware pueden ser revisados en mayor detalle en la bibliografía propuesta.

Así, pues, las soluciones que integran los MAS sobre redes de sensores son diversas, si bien, la aplicación de lógica borrosa a la gestión interna de un agente no es un aspecto explotado, hecho que avala los objetivos de la presente tesis, teniendo en cuenta que es un campo en total desarrollo durante el periodo de realización del presente trabajo de investigación.

2.3 LÓGICA BORROSA Y SU APLICACIÓN A LAS REDES DE SENSORES

La lógica borrosa[13], en virtud de sus características y de las especiales condiciones de las WSNs, permite su aplicación a una gran variedad de aspectos que se presentan en éstas últimas (Kulkarni et al. 2011). Del análisis realizado por Kulkarni y otros (véase la Tabla 2), se puede suponer que una técnica que intenta emular el razonamiento humano a través de procesos en los que la incertidumbre, junto con el conocimiento experto, son elementos fundamentales, encaja dentro de los problemas más típicos de las redes de sensores. Dichos problemas están normalmente fuertemente vinculados a indeterminaciones propias de las redes de sensores, tales como movimiento, tiempo de vida, pérdida de mensajes, caída de nodos, etc. Además, los datos disponibles a través de sondas siempre contendrán un determinado margen de error debido a tolerancias, condiciones medioambientales, fallos, y por supuesto del hecho de que normalmente muestrean cada cierto tiempo y no de manera continua, dadas las limitaciones de recursos energéticos en los nodos.

Así pues, los sistemas borrosos basados en reglas (FRBS) usados en el presente trabajo de investigación, son una de las más importantes áreas de aplicación (Cordón et al. 2001) de la teoría de conjuntos borrosos presentada por Zadeh (Zadeh1965). Estos sistemas son una extensión de los sistemas clásicos basados en reglas del tipo "SI-ENTONCES" (o en inglés "*IF-THEN*"), cambiando los antecedentes y los consecuentes por reglas borrosas en vez de las tradicionales reglas lógicas.

En los FRBS la lógica borrosa se emplea para modelar el conocimiento disponible, así como las relaciones existentes entre las diferentes variables del problema, ya sean de entrada, como de salida. Esta capacidad les da la posibilidad de aplicarse a gran variedad de problemas (Hirota & Sugeno 1995).

La manera en la que la lógica borrosa se diferencia de la lógica clásica es en la asunción de un cierto grado de incertidumbre en los conocimientos y razonamientos, como de hecho sucede con el conocimiento y razonamiento humano (Zadeh1973). Estos principios fueron inicialmente planteados por Zadeh (Zadeh1965, Zadeh1973) sirviendo como base para todos los estudios sobre lógica borrosa.

El elemento que permite la consecución de un fin, o la resolución de un problema basándose en la lógica borrosa, se denomina sistema borroso o difuso. Así pues, un

[13] También se la conoce como lógica difusa.

sistema borroso, en cuanto a su relación con la lógica borrosa, es aquel que emplea la lógica borrosa, ya sea para la representación de diferentes formas de conocimiento o de la interrelación y dependencia de las variables del problema. En parte, uno de los éxitos de los sistemas borrosos es que han conseguido, a través de una novedosa perspectiva, dar respuesta a problemas que de manera tradicional no hubiera sido posible resolver, ya sea por su complejidad o la imprecisión misma de las variables o datos.

A continuación se revisarán los conceptos clave de los FRBS, concretamente se analizarán los FRBS propuestos por Mamdani (Mamdani1974). Dichos sistemas han sido aplicados a los sistemas internos de gestión autónoma de un sensor, lo cual forma parte de las contribuciones de la presente tesis.

2.3.1 SISTEMAS BORROSOS BASADOS EN REGLAS

Mamdani, en el trabajo ante presentado, sentó las bases del primer FRBS con variables reales, al aplicar las teorías de Zadeh al control de un motor de vapor. En esta primera aproximación, Mamdani propuso un esquema en el que las variables de entrada eran pre-procesadas antes de introducirlas al controlador, y una vez conseguida una solución, son de nuevo procesadas antes de actuar sobre el motor de vapor. Estos controladores de entrada y salida son sistemas borrosos, denominados comúnmente, dentro del ámbito de los FRBS, como fuzzificador para el proceso de entrada y defuzzificador, para el proceso de salida. El sistema completo se denomina Controlador Lógico Difuso o FLC (del ingés *Fuzzy Logic Controlles*) según recogen Mamadani y Assilian en (Mamdani & Assilian 1975).

De estos trabajos se desprende que el papel del ser humano es necesario aún para poder diseñar unas reglas que emanen de la experiencia y de un conocimiento experto, que permita diseñar cada uno de los sistemas borrosos. Más aun, remarca la importancia de la lógica borrosa en su aplicación a problemas de difícil modelado, al poder usar un lenguaje próximo al natural a través del uso de variables lingüísticas (Zadeh1973) para detallar los comportamientos del sistema a controlar.

El conjunto de todo el sistema ideado por Mamdani se puede representar de manera genérica conforme a lo representado en la Figura 4, adaptada de los trabajos de (Cordón et al. 2001, Magdalena & Velasco 1996). En ella ya aparecen un fuzzificador y un defuzzificador, junto con el controlador, denominado motor de inferencia. El cuarto bloque del gráfico lo forma la base de conocimiento. Originalmente la base de conocimiento fue asimilada a la interacción humana (Mamdani1974) debido a que, como ya se ha comentado, deber ser diseñada a partir de la experiencia y de un conocimiento experto.

El gráfico original de Mamdani era, si bien equivalente, sensiblemente diferente, dado que estaba aún en sus primeras etapas de evolución, además de estar vinculado a la aplicación que se usó como base para su investigación. Concretamente, la Figura 4 es una adaptación al esquema usado en el presente trabajo de investigación del gráfico presentado en (Cordón et al. 2001). Cabe destacar que en otros modelos, tanto a la entrada como a la salida de variables, se presentan sistemas normalizadores, dado que se suele operar con magnitudes entre 0 y 1, aunque normalmente ese proceso también se puede integrar en la fuzzificación y la defuzzificación.

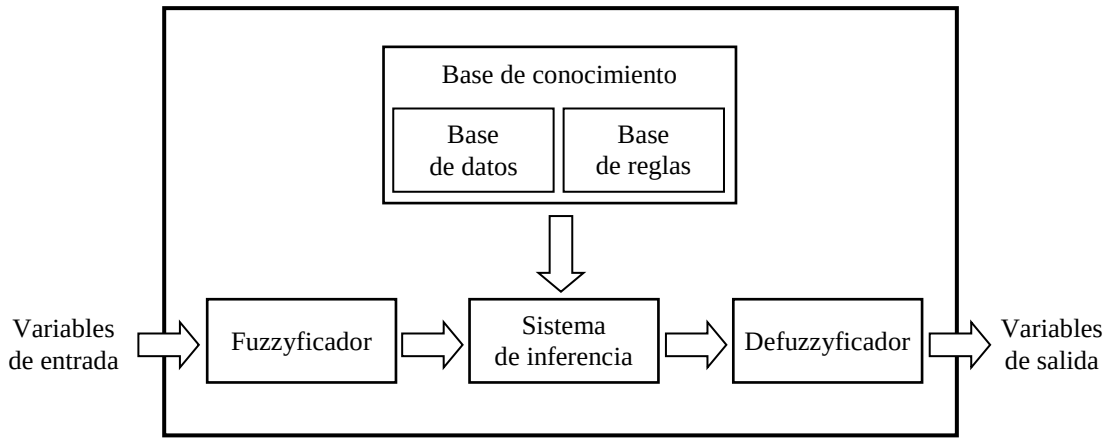


Figura 4. Ejemplo de controlador borroso.

Así pues, a continuación se presentará un análisis detallado de cada uno de los elementos presentados, todos ellos integrantes de un FRBS.

Como se desprende del esquema presentado en la Figura 4, un FRBS se puede dividir en dos partes bien diferenciadas: el conocimiento previo (base de conocimiento) y el motor de inferencia, formado por el sistema de inferencia, el fuzzificador y el defuzzificador.

La **base de conocimiento** o KB (siglas de las palabras en inglés *Knowledge Base*) representa y relaciona todo el conocimiento disponible sobre el problema a tratar, de una manera estructurada y clara. Así pues, la KB modela el conocimiento existente sobre un problema, estableciendo relaciones entre las variables de entrada y las de salida. Estas relaciones se generan a través del razonamiento y la observación de salidas concretas ante una determinada entrada. Por lo tanto, los sistemas borrosos, en gran parte son fruto del conocimiento experto del ser humano, el cual puede contener numerosos aspectos subjetivos, materializándose dentro de un FLC en forma de reglas y funciones de adscripción borrosas (Magdalena & Velasco 1996), además de otros parámetros como rangos o valores de las variables.

Las **reglas borrosas** están conformadas por una serie de variables lingüísticas (Zadeh1975) relacionadas entre sí por elementos de la lógica clásica de la siguiente manera;

$$SI\ X_1\ es\ A_1\ y\ X_2\ es\ A_2\ y\ldots\ X_n\ es\ A_n\ ENTONCES\ Y\ es\ B$$

Donde X_i son las variables de entrada, e Y la variable de salida (en este caso se ha particularizado para un sistema de múltiples entradas y una sola salida). Al conjunto de condiciones de entrada según autores, se les denomina antecedentes y al de salida, consecuentes (Magdalena & Monasterio 1995). El conjunto de todas las reglas se le denomina Base de Reglas.

La definición anterior de una regla es lo que diferencia fundamentalmente a un sistema Mamdani son respecto a un sistema TSK (Takagi, Sugeno y Kang), los cuales definieron que el consecuente fuera una función de las variables de entrada en vez de un valor concreto de otra variable lingüística (Takagi & Sugeno 1985). Así pues, la regla genérica anterior quedaría de la siguiente manera:

$$SI\ X_1\ es\ A_1\ y\ X_2\ es\ A_2\ y\ldots\ X_n\ es\ A_n\ ENTONCES\ Y = f(X_1, X_2,\ldots, X_n)$$

Como ya se ha comentado, la base de reglas es tan solo una de las dos partes de la KB. La otra está formada por la adscripción de cada variable a una partición del espacio borroso, lo cual se consigue a través de conjuntos borrosos. El conjunto de valores de las variables lingüísticas, junto con los conjuntos borrosos conforman la Base de Datos.

Una **variable lingüística** se define a través de un conjunto de valores lingüísticos (palabras o términos del lenguaje natural) en vez de valores numéricos (Zadeh1975). Por lo tanto, una variable que represente, por ejemplo, la carga de un nodo de una red, en cuanto a número de paquetes enviados por minuto, con una variable lingüística podrían ser MUY BAJA, BAJA, MEDIA, ALTA y MUY ALTA, en vez de 10, 20, 30, 40 y 50. El objetivo de este tipo de variables, como resalta Zadeh en el trabajo antes mencionado, es el de poder asignar una cierta incertidumbre a fenómenos que normalmente no pueden caracterizarse fácilmente, a través de valores numéricos. En el ejemplo anterior, una carga MUY BAJA podría ser un valor 10, pero ¿qué sería un valor de 15? Para unos podría ser MUY BAJA y para otros BAJA. Es por eso que en vez de un solo valor numérico, o un rango de valores, las variables lingüísticas, se definen, aparte de por el conjunto de valores disponibles, por una serie de conjuntos borrosos, cada uno correspondiente a un valor concreto de la variable.

Cada **conjunto borroso** se representa por una función de compatibilidad (Zadeh1975) o pertenencia (Cordón et al. 2001)que define una partición borrosa de su dominio de trabajo. Estas particiones se pueden representar a través de diversos tipos de funciones, siendo las más habituales el uso de zonas triangulares, trapezoidales o gaussianas. En la Figura 5 se representa una partición con conjuntos borrosos triangulares para la variable lingüística de carga de un nodo, antes mencionada.

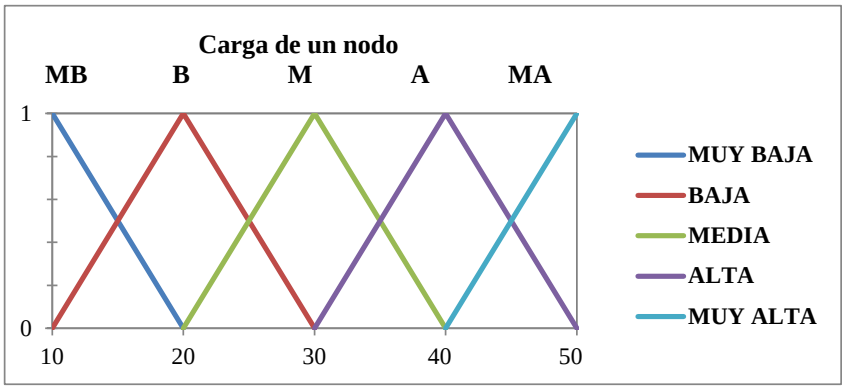


Figura 5. Ejemplo de partición borrosa.

Como puede observarse en la Figura 5, el grado de pertenencia de un valor de carga dentro para un conjunto borroso, en este caso irá de 0 a 1, un rango muy habitual en FRBS. La definición tradicional de un conjunto borroso, dada por Zadeh en (Zadeh1973) es la siguiente:

"Un conjunto borroso *A* de un universo de discurso *U* caracterizado con una función de pertenencia μ_A*: U→ [0,1]* la cual asocia cada elemento *y* de *U* un número $\mu_A(y)$ en el intervalo *[0,1]* el cual representa el grado de pertenencia de *y* en *A*"

Por lo tanto, la atribución de un grado de pertenencia 0 indicará que el elemento *y* no pertenece a dicho conjunto borroso, así como un grado de pertenencia 1 implica una pertenencia completa, siendo los valores intermedios, el grado de pertenencia para elementos que no se encuentren entre los dos casos anteriores.

Así pues, la representación matemática, por ejemplo, de un conjunto borroso de silueta triangular como los presentados en la Figura 5 sería la siguiente:

$$\mu(x) = \begin{cases} 0, & si\ x \leq\ x_1 \\ \dfrac{x - x_1}{m - x_1}, & si\ x\ \in\ (x_1, m] \\ \dfrac{x_2 - x}{x_2 - m}, & si\ x\ \in (m, x_2] \\ 0, & si\ x\ \geq x_2 \end{cases}$$

Ecuación 1

En la siguiente figura se representa de manera genérica un conjunto borroso triangular según la definición de la Ecuación 1.

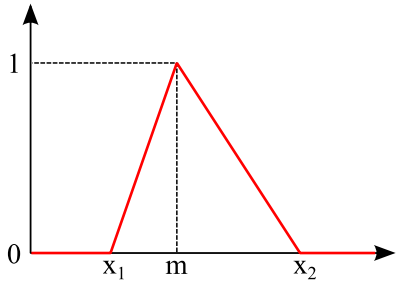


Figura 6. Conjunto borroso triangular.

La teoría sobre conjuntos borrosos puede verse con mayor profundidad en (Zadeh1975).

Una vez descritos los elementos que componen la base de conocimiento queda por detallar la parte encargada del procesado de la información en función del conocimiento previo, el motor de inferencia (MI).

Como ya se ha comentado anteriormente, un motor de inferencia de tipo Mamdani se compone de una interfaz de fuzzificación, un sistema de inferencia, y una interfaz de defuzzificación.

Dado que las magnitudes que normalmente se usan como entradas no son borrosas, la función principal de la **interfaz de fuzzyficación** es la transformación de estos valores, denominados tradicionalmente como valores precisos (del inglés *crisp*, preciso, definido, nítido) a valores borrosos. Para llevar esto a cabo el proceso de fuzzyficación realiza un mapeo de los valores de entrada (*crisp*) a un determinado conjunto borroso dentro del universo de discurso de esa entrada. Este proceso puede venir acompañado también de una normalización, ya que es habitual que los valores de entrada se pasen al motor de inferencia en un rango entre 0 y 1.

Así pues, una variable de entrada, como la vista anteriormente de carga de un nodo, cuyo rango de valores no borrosos va de 10 a 50, para un valor preciso de 20, el valor borroso de entrada al sistema de inferencia sería:

$$(20 - 10)/(50 - 10) = 0{,}25$$

Ecuación 2

El **sistema de inferencia** se encarga de generar una serie de salidas borrosas a partir de los conjuntos borrosos de entrada y de las relaciones marcadas en las reglas borrosas.

Este esquema de inferencia establece una relación entre los conjuntos borrosos de entrada $U = U_1 \times U_2 \times \ldots \times U_n$ en el dominio de entrada $X_1,\ldots,X_n$ y los conjuntos borrosos V en el dominio de salida Y. Este esquema de inferencia borroso usa el *modus ponens* generalizado extendido por Zadeh (Zadeh1973) del *modus ponens* tradicional:

SI X ES A ENTONCES B

X ES A'

Y ES B'

La aplicación de la regla anterior y su desarrollo puede verse con más detalle en (Zadeh1973), así como de manera más resumida en (Cordón et al. 2001). Por lo tanto, resumiendo el proceso, el sistema de inferencia busca la contribución, en una variable de salida, de cada conjunto borroso presente en una regla borrosa, para cada regla existente en la base de reglas. Dicha contribución se obtiene a través del cálculo del **grado de pertenencia** de un valor borroso de entrada para la variable dada. Como resultado del sistema se inferencia, se obtiene un conjunto borroso de salida para cada regla borrosa, lo cual conforma la entrada para el siguiente y último componente del motor de inferencia, la interfaz de defuzzificación.

La labor de la **interfaz de defuzzificación**, como se desprende de lo anterior, es la obtención de un valor preciso a partir de la unificación de los conjuntos borrosos de salida de la aplicación de cada regla borrosa. Esta tarea se puede llevar a cabo a través de dos métodos: **A-FITA**, propuesto por Mamdani (Mamdani1974) y basado en una agregación inicial de los conjuntos borrosos y una posterior inferencia del valor preciso; y el **B-FITA** (Cordón et al. 2001, Bardossy & Duckstein 1995, Wang1994), en el que primero se realiza la inferencia y posteriormente la agregación de los valores inferidos para obtener el valor preciso de salida. Ejemplos de ambos pueden verse en (Cordón et al. 2001).

Por lo tanto, una vez hecho este breve repaso por los FRBS, tan solo cabe remarcar algunas de las ventajas y desventajas que presentan estos sistemas ideados por Mamdani.

Así pues, como principal ventaja y a su vez uno de los aspectos más destacables de este tipo de FRBS es la versatilidad que aportan las reglas borrosas a la hora de expresar el conocimiento experto. Además de esta ventaja fundamental, se pueden identificar las siguientes:

- El amplio margen de libertad que permiten los interfaces de fuzzificación y defuzzificación, así como el propio sistema de inferencia para adaptarse a una gran variedad de aplicaciones.
- La expresión del conocimiento a través de reglas lingüísticas facilita la interpretación por humanos de los condicionantes y acciones que relacionan variables de entrada y salida.
- Capacidad para adaptar o modificar el funcionamiento de un FRBS a nuevas condiciones o aplicaciones a través de cambios concretos en alguno de sus elementos (reglas, conjuntos borrosos, fuzzificador, defuzzificador, etc.) sin tener que cambiar todo el sistema en sí.
- Facilidad para trasladar su representación formal a formatos compactos para su proceso.

A pesar de las ventajas presentadas, existen ciertas desventajas, principalmente derivadas del modelado basado en reglas lingüísticas, lo que conlleva una falta de precisión para problemas complejos. Algunas de estas limitaciones, estudiadas por diferentes autores, aparecen resumidas en (Cordón et al. 2001):

- Falta de flexibilidad dada la rígida partición de los espacios de entrada y salida.
- Cuando las variables de entrada no son mutuamente independientes se dificulta el encontrar una adecuada partición borrosa del espacio de entrada.
- La partición homogénea de los espacios de entrada y salida llega a ser ineficiente y poco escalable ante un incremento de las dimensiones y complejidad del mapeado de los espacios de entrada y salida.
- El tamaño de la KB de un problema crece rápidamente con respecto al número de variables y reglas lingüísticas, lo cual incrementa considerablemente la complejidad total del sistema cuando se quiere aumentar la precisión de los resultados. Esto se debe a que un aumento de la precisión conlleva una mayor granularidad de las particiones borrosas de las variables, que a su vez necesitan un mayor número de reglas para contemplar los nuevos casos, pudiendo hacer más compleja su interpretación.

Para intentar paliar estos problemas surgieron distintas variantes al sistema propuesto por Mamdani, una breve explicación sobre estos puede encontrarse en (Cordón et al. 2001).

2.3.2 APLICACIÓN DE LOS FRBS A LAS REDES DE SENSORES

En la sección 2.1.6, dentro de las tendencias actuales en las redes de sensores, se comentó que las técnicas de inteligencia computacional son un área de expansión de las redes de sensores. En la Tabla 2, extraída de (Kulkarni et al. 2011), quedó reflejada la idoneidad expresada por el autor de las técnicas de lógica borrosa para las redes de sensores.

En particular, en el presente trabajo de investigación se han usado sistemas borrosos basados en reglas o FRBS de tipo Mamdani. Cabe destacar que el uso de estos sistemas dentro de las redes de sensores al tiempo del comienzo del presente trabajo de investigación estaba en sus inicios. Así, se pueden encontrar los trabajos aplicados a la detección de incendios (Marin-Perianu & Havinga 2007), o en (Canada-Bago2007), donde el sistema borroso es combinado con un algoritmo genético para el aprendizaje dentro del propio sensor.

Así pues, la **aplicación de FRBS a las redes de sensores** se empezó a tomar como una técnica apropiada para gran variedad de aplicaciones debido a que estos sistemas cumplen con dos importantes requisitos que se establecen en (Marin-Perianu & Havinga 2007):

- Son suficientemente sencillos para ser ejecutados en dispositivos de prestaciones limitadas.
- Toleran que los datos disponibles sean imprecisos o incluso inciertos.

Otra ventaja importante de los FRBS es que, para **modificar su comportamiento** basta con cambiar la base de conocimiento, ya sea total o parcialmente. Además, si dicha base de conocimiento está adecuadamente definida, en términos de representación de la

información[14], la cantidad de datos necesarios para su modelado, si bien depende del número de reglas, conjuntos borrosos y variables que contenga, es relativamente baja comparada con la capacidad de almacenamiento y transmisión de la mayoría de familias de redes de sensores ya comentadas. Así, modificar el diseño de un conjunto borroso, los límites de una variable o alguna de las proposiciones de una regla, supondría apenas unas decenas de bytes. Por lo tanto, estas modificaciones podrían ser enviadas a través de la red sin ocasionar un consumo excesivo de energía y conseguir así la mejora o actualización del comportamiento de un sensor de una manera relativamente sencilla. Además de para adaptar el comportamiento del sensor, estos parámetros podrían ser enviados por éste último a otros nodos, o a una estación base, como resultado del aprendizaje obtenido por el mismo. Así pues, las mejoras encontradas en la ejecución de proceso de aprendizaje pueden ser distribuidas a los demás nodos, mejorando el rendimiento total de la red. Sin embargo, esta actualización de los parámetros en los nodos de la red sí debería acometerse con más cautela, pudiéndose beneficiar de esquemas de agregación de datos ya existentes para la distribución de esos parámetros.

Como se ha venido comentando, **el uso de lógica borrosa en las redes de sensores ha ido tomando especial interés en los últimos años**, si bien ha sido aplicada de muy diferente manera. Se ha comentado que en (Marin-Perianu & Havinga 2007) se usan FRBS para la **detección de incendios**, estando dotado cada sensor de su propio motor de inferencia, como en el trabajo de Xia (Xia et al. 2007), donde los nodos encargados de medir las magnitudes bajo estudio incorporan un sistema borroso para la **gestión de la QoS** de la red de sensores basándose en la adaptación de los periodos de muestreo a la tasa de pérdidas por tráfico en la red.

Otro tipo de aplicaciones usan la lógica borrosa de manera centralizada, ya sea a priori o continua. Entre las soluciones a priori, se encuentran aquellas que planifican a partir de ciertas condiciones iniciales el **despliegue de una nueva red de sensores**, buscando qué protocolos y/o topología serían los más apropiados. Dentro de este tipo están los trabajos de (Pirmez et al. 2007), donde un sistema borroso determina el protocolo de comunicaciones más apropiado para el despliegue de una red en función del número de estaciones base, sensores, tráfico esperado y parámetros QoS.

Por otro lado, como ejemplo de sistemas que usan la lógica borrosa de una manera reiterada o continua durante el tiempo vida de la red, está el trabajo de Kim y otros (Kim et al. 2008), donde los autores usan los sistemas borrosos para la **gestión de clústeres en una red de sensores**. Otros ejemplos de aplicación de la lógica borrosa a las redes de sensores pueden verse en (Rea & Pesch 2004) y (Haider & Yusuf 2009), donde esta técnica se emplea para el **cálculo del coste de rutas en protocolos de encaminamiento**. También aparecen aplicaciones como (Yusuf & Haider 2005) y (Misra et al. 2009), las cuales hacen uso directamente de un protocolo de enrutamiento basado en lógica borrosa.

Cabe señalar, que **la aplicación de la lógica borrosa al enrutamiento**, ha suscitado mucho interés debido al hecho fundamental de que un óptimo algoritmo de enrutamiento evitaría el desperdicio de energía que supone el envío de paquetes con más potencia de la necesaria al usar rutas no adecuadas.

[14] La representación de la información mencionada hace aquí referencia a la forma en la que una base de conocimiento es codificada en un formato binario inteligible, ya sea para su almacenamiento en medios informáticos o para su transmisión por una red de comunicaciones (como podría ser una red de sensores).

Sin embargo, **es en la gestión autónoma del sensor basada en FRBS donde el presente trabajo de investigación pretende aportar nuevas perspectivas** para el control de las tareas que un sensor debe acometer dentro de la función que se le ha otorgado en una red. Además, la gestión de estos comportamientos debe tener siempre en cuenta que un sensor tiene unos recursos limitados, principalmente la energía disponible. Así pues, dentro de este campo no se han encontrado trabajos relacionados con FRBS para la gestión interna de un sensor ajenos al llevado a cabo por el grupo de investigación.

Recorriendo algunos trabajos que tienen en cuenta lo comentado sobre el uso de FRBS para la gestión interna de un sensor, en el trabajo de Benoit (Benoit et al. 2001) se describe un modelo para definir los servicios que los sensores tendrían disponibles en forma de agentes inteligentes, lo cual entronca con la perspectiva que será desarrollada en este trabajo: la coexistencia de agentes controlados por subsistemas inteligentes dentro de cada sensor. Dichos agentes marcarán el funcionamiento del sensor dentro de la aplicación a la que se destine, así como para tareas organizativas de la red, tales como la formación de clústeres o enrutamiento.

Otra aplicación reciente de FRBS, de tipo Mamdani concretamente, y sobre sensores Sun SPOT es la que se puede encontrar en (Rawi & Al-Anbuky 2011). En este trabajo, el autor remarca el hecho del **muy escaso número de contribuciones en el ámbito de las WSNs que integran FRBS en los nodos**. En lo concerniente a la arquitectura de esta red de sensores, ésta está formada por nodos sensores, nodos actuadores y nodos sumidero. Es en los nodos sensores donde están implementados los FRBS. La KB se envía a través del interfaz que proporcionan los Sun SPOT para la transmisión de información genérica vía radio. Cada nodo sensor tiene encomendada una tarea diferente. Por lo tanto, cada uno se encarga de la inferencia del grado de confortabilidad por un factor diferente: temperatura, iluminación, velocidad del aire y ruido. El resultado de cada sensor se envía a un nodo central que es el que calcula finalmente en grado de confortabilidad de la estancia. Como se desprende de lo anterior, esta arquitectura difiere ampliamente de lo perseguido en la presente tesis, ya que, en primer lugar, no existe una arquitectura definida dentro de cada nodo, así mismo, no se define un protocolo de comunicaciones para el intercambio de información de manera controlada. Y en particular, lo que lo hace diferir más, es que los sistemas FRBS se usan para inferir resultados que no conllevan un ajuste en las tareas o del comportamiento del propio sensor.

Por último, como **reflexión final del estudio del estado del arte**, con respecto a la aplicación de los FRBS a la gestión de sensores inteligentes con una arquitectura multi-agente interna, en este trabajo de tesis se presenta una nueva perspectiva de la integración de tecnologías de inteligencia computacional, como son los FRBS en entornos de limitadas prestaciones. Concretamente se persigue el objetivo de que, tanto el propio comportamiento del sensor, como el del protocolo de comunicaciones, obedezca en la medida de lo posible, a criterios de eficiencia energética controlados por FRBS. Y como se ha puesto de manifiesto, este tipo de aplicaciones son altamente novedosas, contando prácticamente con el trabajo llevado a cabo por el grupo de investigación de Ingeniería de Sistemas Telemáticos, dentro del cual se han desarrollado los trabajos de la presente tesis.

Capítulo II. DESARROLLO Y METODOLOGÍA DE LA INVESTIGACIÓN

Los trabajos de investigación desarrollados en la presente tesis se han clasificado en tres bloques: la arquitectura multi-agente presente en cada sensor, el protocolo de comunicaciones diseñado para gestionar la red de sensores y por último, la aplicación de sistemas analíticos y basados en conocimiento para la gestión interna de un sensor. Cada una de estas partes respaldará una de las tres hipótesis propuestas, intentando cumplir con los objetivos planteados en el capítulo anterior. Así pues, en las siguientes secciones de este capítulo se detallarán convenientemente los trabajos realizados para la consecución de los mencionados objetivos.

3 ARQUITECTURA MULTI-AGENTE

Como se ha podido constatar en el Capítulo I, tanto en la definición, como en las distintas arquitecturas posibles de agentes, existen opiniones y soluciones diversas. Además, su aplicación a las redes de sensores, como técnica de inteligencia artificial, ha contado con diferentes interpretaciones (Vinyals et al. 2011). De entre éstas, la más habitual ha sido la inclusión de un agente dentro de un sensor para la monitorización de un parámetro concreto (Fok et al. 2009), o el uso de una red de sensores para la captación de parámetros del entorno que posteriormente serían transmitidos a un elemento de mayor capacidad, donde realmente estaba implementado el agente. Así pues, en el presente trabajo de investigación se propondrá una arquitectura multi-agente específicamente diseñada para la gestión interna de los limitados recursos disponibles en un sensor, con objeto de aumentar la eficiencia en su uso.

3.1 REQUISITOS Y JUSTIFICACIÓN

En el Capítulo I, concretamente en el apartado 2.2.5, se han mostrado diferentes características de los MAS, y lo que estos implican dentro de las redes de sensores. Como se ha podido ver, es una tarea compleja, marcada por las limitadas prestaciones de los nodos.

Por ese motivo, y como principal requisito, se hace necesario que la aplicación de un sistema multi-agente para la gestión de un sensor no suponga un gasto de recursos que dificulte, o incluso haga inviable, el desarrollo de las tareas destinadas a ser realizadas por el propio sensor dentro de una aplicación concreta. Teniendo en cuenta esta premisa de inicio, se ha diseñado una arquitectura multi-agente interna a un sensor, para llevar a cabo una gestión inteligente de las tareas encomendadas al mismo.

Como primera consecuencia de la aplicación de esta arquitectura multi-agente, se espera que la gestión de los recursos energéticos del sensor resultante derive en un mayor tiempo de vida útil del mismo. Por otro lado, dado que los recursos de proceso y memoria con los que cuentan los sensores son normalmente muy limitados, y la ejecución de determinadas tareas de menor importancia puede poner en riesgo el cumplimiento de otras funciones críticas, se necesita una gestión que priorice entre las

distintas tareas en función de los recursos disponibles. Estas premisas hacen necesario un sistema que aporte la capacidad de gestión de recursos de manera inteligente y que tenga ciertas capacidades especiales:

- Comunicación entre entes de este tipo para llevar a cabo operaciones coordinadas.
- La atribución de características complejas, tales como objetivos, deseos o creencias que respondan a la atribución de ciertos comportamientos basados en el razonamiento humano.
- Capacidad de replicar su comportamiento en cada uno de los nodos de una red a través de la migración de dichas entidades de un sensor a otro.

Como puede intuirse, estas características entran dentro de las ya comentadas para los agentes y arquitecturas multi-agente, por lo que las hace especialmente apropiadas para el cumplimiento de los objetivos del presente trabajo.

Otro factor que justifica el uso de los MAS, es la escalabilidad. Esta propiedad permite que existan diferentes sistemas inteligentes encargados de partes bien diferenciadas de un sensor, tanto software como hardware, y no un solo sistema complejo encargado de todo. Así pues, los agentes funcionarían de manera coordinada pero independiente, pudiéndose desconectar aquellos que no fueran necesarios en un determinado momento.

En resumen, la aplicación de una arquitectura multi-agente a la gestión interna de un sensor viene dada por la capacidad propia de un agente como sistema inteligente, y del hecho de estar formada por más de un agente. Así pues, cada uno de estos agentes conseguirán, en conjunto, controlar áreas diferenciadas de un sensor de manera coordinada, basándose en características ya mencionadas en el Capítulo I, tales como un lenguaje propio, intenciones, creencias, deseos, etc.

3.2 ARQUITECTURA MULTI-AGENTE DE UN SENSOR

Dados los requisitos revisados en el apartado anterior, se ha considerado necesario establecer una división concreta en cuanto a los elementos funcionales que deberían ser controlados por cada agente dentro de un nodo. Para llevar a cabo esta división, se han seguido criterios de simplicidad, dadas las limitaciones de los sensores, así como se han observado las características propuestas por Wooldridge y Jennings (Wooldridge & Jennings 1995b) dentro de lo que se podría denominar como una definición débil, o ligera, de lo que es un agente (agentes ligeros): sistemas software capaces de actuar por sí mismos (autonomía), con capacidad de comunicarse con otros agentes (sociabilidad), conocer y actuar en función del entorno que lo rodea (reactividad) y adelantarse a futuras condiciones a través del estudio del entorno, previendo actuaciones que le ayuden en su labor (pro-actividad).

Dadas las especiales condiciones de un sensor, se ha optado por esta visión de un agente, debido a que la otra vertiente, más presente dentro de la teoría general de agentes, pretende el modelado del razonamiento humano, distando considerablemente de los objetivos del presente trabajo, sumándose que, en principio, sería una tarea de complicada viabilidad en sistemas con recursos tan limitados.

Así pues, atendiendo a las áreas funcionales más importantes en un sensor, y teniendo presente el principio de ahorro de recursos, las tareas a las que serán destinados los sensores y las características presentes en un agente, se propone una arquitectura multi-agente con los siguientes miembros:

- Agente de Gestión (AG): encargado del control general del sensor, como los periodos de funcionamiento y ciclos de sueño, así como del control de los demás agentes.
- Agente de Control de Aplicación (ACA):estaría encargado de la ejecución de las tareas previstas para el sensor, tales como toma de datos de sondas, activación de salidas, ejecución de subprogramas y FRBS, etc.
- Agente de comunicaciones (AC): se encargaría de las comunicaciones, ya sea recepción o envío de información.

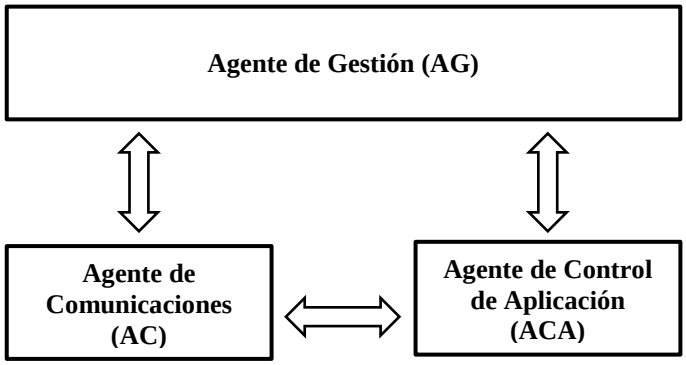


Figura 7. Arquitectura multi-agente propuesta.

El motivo de que sean estos tres agentes concretamente (ver Figura 7) viene dado por el hecho, ya mencionado, de que la propia estructura multi-agente no debe suponer un gasto de recursos superior a las propias tareas a las que esté destinado el sensor. Por lo tanto, se ha buscado que esta división funcional reúna las condiciones necesarias para que cada agente pueda tener una finalidad concreta, precisa, y sea implementado de la manera más compacta posible. Los aspectos tenidos en cuenta se detallan a continuación:

- **Correspondencia con la estructura interna de un sensor.** Los sensores generalmente disponen de subsistemas encargados de la gestión genérica de sus recursos, tales como la carga de la batería, estado de la memoria o el control sobre sus ciclos de sueño y trabajo. Otro bloque típico es el de los subsistemas dedicados a las comunicaciones, los cuales gestionan la torre de protocolos, la transmisión vía radio, el enrutamiento, etc. Por último, están los subsistemas de interacción con el entorno, encargados de tomar medidas, procesarlas, almacenarlas y/o generar algún tipo de salida o decisión. Así pues, se hace evidente que estos tres tipos de subsistemas pueden estar controlados cada uno por un agente diferente y colaborar entre sí para llevar a cabo la función a la que el sensor esté destinado.
- **Complejidad de la jerarquía del sistema multi-agente:** se ha intentado que el número de agentes no suponga una carga excesiva en el conjunto de la implementación en un sensor. Además, la arquitectura diseñada posee una jerarquía simple que permite, si fuera necesario, que tan sólo el AG esté en funcionamiento, encargándose éste de activar a los demás si sus funciones fueran requeridas. De esta manera, se consigue un ahorro importante de recursos al no tener activos aquellos agentes (y por extensión sus subsistemas asociados) que no se necesiten. Cabe destacar, que sería posible cualquier otra arquitectura de agentes y subsistemas (Aiello et al. 2009, Fortino et al. 2012), pero como se

podrá constatar a continuación, la atribución concreta de funciones y tareas a cada uno de los agentes viene dada por la segunda y tercera hipótesis del presente trabajo de investigación.

- **Funciones previstas en las hipótesis y objetivos del presente trabajo:** en concreto, la segunda y tercera hipótesis proponen la creación de un protocolo de comunicaciones y la aplicación de sistemas analíticos y basados en el conocimiento para la monitorización y control del entorno de manera eficiente en cuanto a la gestión de recursos. De ahí la división elegida antes comentada, un agente de comunicaciones para el control del nuevo protocolo, y un agente de control de aplicaciones para dar soporte a los sistemas basados en conocimiento.

En los apartados siguientes se detallará cada uno de los agentes que forman la arquitectura multi-agente de un sensor, sus comportamientos y funciones dentro del sensor, haciendo un recorrido por cada una de las propiedades, que como agente, se les deben reconocer:

- Autonomía.
- Sociabilidad.
- Reactividad.
- Pro-actividad.

Sin embargo, antes de pasar a la explicación detallada de cada uno de los agentes, se pasará a comentar la arquitectura general de los mismos, y las consideraciones que ha llevado al diseño adoptado.

3.3 ARQUITECTURA DE LOS AGENTES

Un aspecto que es necesario abordar, previo a la descripción detallada de cada uno de los agentes, es el modelo que se ha seguido para la especificación del conocimiento de estos, así como la forma en la que los agentes se comunican entre sí.

Como ya se ha comentado, los agentes que forman parte de la arquitectura propuesta siguen el paradigma de agentes ligeros (Van Dyke Parunak et al. 2004), dado que su comportamiento podría modelarse con la configuración de ciertos parámetros y no tratan de representar escrupulosamente el pensamiento humano, sino que intentan responder a las tareas encomendadas de forma mucho más concreta, modelando hasta un nivel preciso su entorno y su comportamiento.

Teniendo en cuenta la dificultad presente en cuanto la definición de un agente, así como de los limitados recursos disponibles en los sensores, se ha buscado una estructura que se pueda beneficiar de la sinergia con los sistemas basados en el conocimiento que se ha propuesto usar en el presente trabajo de investigación. De esta manera, la solución adoptada estaría a medio camino entre los agentes pesados y los ligeros, pudiéndose decir que es una arquitectura híbrida. El motivo de esta elección se debe a que el comportamiento del agente, en algunos casos, debe ser tan sencillo como el paso de parámetros a una función, y en otros, responder a una tarea más compleja definida con criterios más próximos al pensamiento humano.

En concreto, para definir el comportamiento de cada uno de los agentes se ha desarrollado un modelo ligero inspirado en el Razonamiento Procedimental o PRS (del inglés *Procedural Reasoning System*), conocido también como el paradigma Creencias-Deseos (o metas) e Intenciones, o BDI (del inglés *Beliefs, Desires and Intentions*),

próximo a los trabajos de Rao y Georfeff (Rao & Georgeff 1991), cuya representación gráfica puede verse en la Figura 8.

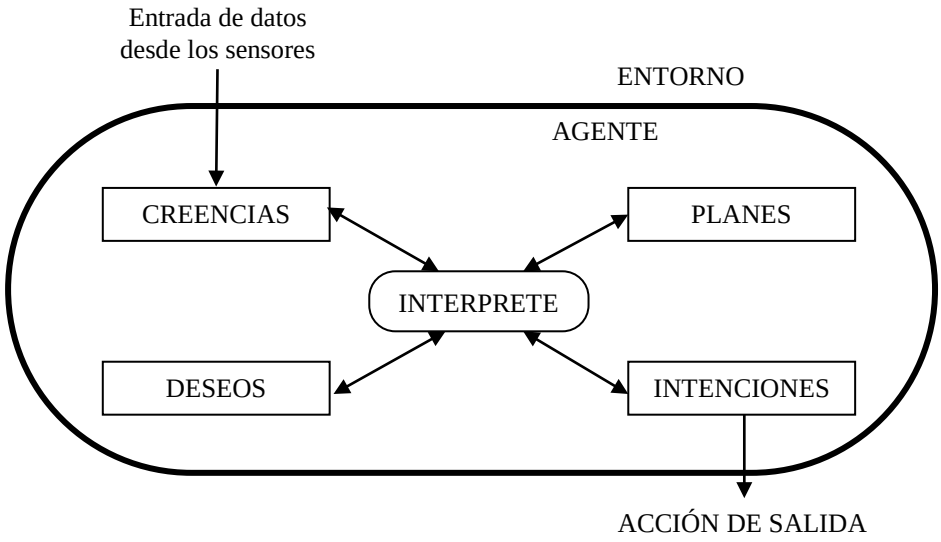


Figura 8. Arquitectura de un agente basado en BDI.

Se ha elegido este modelo como base por su importancia dentro la teoría de agentes, y porque permite definir una gran variedad de comportamientos atribuibles a un sensor a través de una especificación próxima al lenguaje natural, aspecto importante en el presente desarrollo.

Una razón adicional para el uso de la arquitectura BDI, es la posibilidad de combinar estos agentes con sistemas de lógica borrosa. Han existido diversas aproximaciones de este tipo, como las de (Long & Esterline 2000, Shen et al. 2004), que hacen uso de la lógica borrosa para poder solucionar los problemas que presentan las relaciones entre deseos-metas y las intenciones. Sin embargo, teniendo en cuenta las necesidades del bajo coste de recursos que rigen todas las implementaciones en redes de sensores, y dado que se ha decidido implantar un agente ligero, éste se ha desarrollado, apoyándose en agentes BDI y FRBS o FLBDI (del inglés *Fuzzy Light Beliefs Desires Intentions*), dando lugar a, una arquitectura BDI ligera borrosa o FLBDIa (del inglés *Fuzzy Light Beliefs Desires Intentions arquitecture*).

Así pues, se plantea una **asimilación de un agente BDI con un FRBS**, buscando las analogías adecuadas entre los elementos que conforman ambos para aportar conocimiento al sistema. Este tipo de estructuras híbridas han sido estudiadas desde diferentes puntos de vista, como por ejemplo el trabajo de Shen y otros (Shen et al. 2007) en el que se presenta una estructura híbrida en la que se usa la lógica borrosa para generar las creencias del modelo BDI. Otro ejemplo de integración es el propuesto por Shi y Xu (Youqun Shi & Hongyun Xu 2009) en el que se hibrida un agente BDI con un sistema de lógica borrosa dinámica (DFL, del inglés *Dynamic Fuzzy Logic*) para la generación de aprendizaje. En este último caso, al agente BDI se le añade un nuevo elemento de decisión, la credibilidad (*Believability*) que es el resultado del DFL. Sin embargo, la solución que aquí se propone, pretende ahondar aún más en las relaciones inherentes en los procesos de decisión entre un sistema BDI y un FRBS y así optimizar el uso de recursos dentro de un sensor. En consecuencia, las relaciones que se presentan son las siguientes:

- **El conocimiento previo:** las creencias se fundamentaran en el conocimiento experto que permite modelar ciertos fenómenos a emular o controlar, junto con los valores que se captan del entorno (Shen et al. 2007). Por lo tanto, se puede asimilar el conjunto de las variables de entrada, junto con su definición a través de los conjuntos borrosos (base de datos) como las creencias, ya que es una forma normalizada y estrechamente vinculada al razonamiento humano.
- **Resultados deseables (deseos - base de reglas):** los deseos o metas (*goals*) son la expresión de lo que se pretende conseguir en función de cada creencia (Rao & Georgeff 1991), por lo tanto, están estrechamente vinculados con las reglas borrosas, las cuales, para unos valores concretos de las variables de entrada, modelan una serie de metas o deseos en forma de variables de salida.
- **Tarea y propósito al que se destina el agente (intenciones – inferencia):** la intención de un agente se puede asimilar a la inferencia que se realiza en un FRBS. Así, partiendo de una serie de creencias (base de datos), y con unas metas definidas (reglas) en cada momento, en función de entorno (variables de entrada), se intenta determinar cuál es la mejor solución para la tarea a la que se ha destinado el sistema (inferencia), lo cual se puede comparar con la intención del mismo, vista como el conjunto de acciones que llevan a la consecución de una meta concreta.

Se debe destacar que, como sistema PRS, este tipo de agentes no funcionan por principios básicos, sino que disponen de una serie de planes pre-compilados, lo cuales ya están definidos por el programador del agente. Estos planes tienen una serie de elementos comunes (Wooldridge2009):

- Un objetivo, el cual conforma la *pos-condición* del plan.
- Un contexto que forma la *pre-condición* del plan.
- Un cuerpo en el que se detalla el curso de la acción a llevar a término.

Tras presentar la forma en la que se estructura un plan, queda patente que los agentes BDI, cuyo comportamiento sea modelado con FRBS, son factibles.

Este tipo de estructura se aplicará al Agente de Gestión dado que presenta un perfil más idóneo. Los demás se modelarán siguiendo un esquema de agente ligero (Van Dyke Parunak et al. 2004) basados en comportamientos definidos a valores concretos de parámetros internos a cada agente. El motivo de usar diferentes arquitecturas de agentes en el presente trabajo, se debe al necesario compromiso entre consumo de recursos por parte de la propia arquitectura multi-agente, y al desempeño de las tareas asignadas al sensor. Una vez detallada la arquitectura BDI ligera borrosa, ésta podrá ser tomada como base para los demás agentes del sensor, aunque su implementación quede relegada a trabajos posteriores.

El detalle de cada una de estas arquitecturas aparecerá en sus apartados correspondientes del presente capítulo.

3.4 COMUNICACIONES ENTRE AGENTES

La comunicación prevista entre los agentes creados, tiene dos vertientes, la que se lleva a cabo entre los agentes dentro del propio sensor, y la que se entabla entre agentes de diferentes sensores. Al igual que la arquitectura de un agente, la comunicación entre ellos ha propiciado diversas soluciones, si bien, no ha habido tanta controversia como con la definición de qué es un agente o su modelado. En el Capítulo I se mencionan

algunos de estos lenguajes, pudiéndose ver una revisión de estos en (Ahmed et al. 2009). Dada la mayor complejidad de las comunicaciones entre agentes de diferentes sensores, se comentará en primer lugar ésta, dado que la comunicación entre los agentes internos de un sensor se ha intentado minimizar, a través del uso de interfaces definidos a la hora de la implementación de los mismos.

Para la comunicación entre agentes de diferentes sensores no se ha seguido un lenguaje específico, si bien **se ha buscado una aproximación al estándar FIPA** (Fipa ACL2004), el cual, sigue en parte la especificación KQML (Finin et al. 1994), no ha sido una prioridad, dada la limitada capacidad de cómputo de un nodo. Según Wooldridge, ambos lenguajes poseen la misma estructura de definición de mensajes difiriendo en las primitivas usadas para construir el lenguaje (Wooldridge2009)). En este último artículo, el autor también comenta que las primitivas más relevantes en el lenguaje definido en la especificación FIPA son *inform* y *rerquest*, hecho que ha servido como base para la definición del lenguaje entre agentes que se describe a continuación.

Teniendo en cuenta estos dos tipos de primitivas, se ha diseñado una especificación adaptada a unos de los protocolos más usados en Internet, HTTP (Fielding et al. June 1999). La razón de usar este tipo de adaptación es la necesidad de compatibilizar las comunicaciones entre equipos interconectados en Internet y las redes de sensores. Así pues, se podrá proporcionar un acceso a las redes de sensores desde cualquier parte del mundo a través de las herramientas convencionales diseñadas para estos fines, tales como los navegadores web. Esto queda patente, por ejemplo, con el desarrollo de 6LoWPAN (Kushalnagar et al. September 2007) o por el trabajo de proyectos europeos como SENSEI (EU Integrated Project), que perseguía la integración del mundo físico con Internet a través de redes de sensores.

Así pues, la propuesta que se presenta es la **asimilación de cada una de las primitivas FIPA (*inform* y *rerquest*) con un comando HTTP**. En concreto, la primitiva *inform* pretende que el receptor del mensaje "crea" el contenido que se le aporta, es decir, es una primitiva de comunicación de información. Por otro lado, la primitiva *rerquest* se utiliza para requerir una acción por parte de un agente. Teniendo en cuenta la finalidad de cada una de ellas, se ha asignado el método *GET*, del protocolo HTTP, a la primitiva *inform*, y el método *POST*, a la primitiva *request*. Cabe destacar, que tanto el método *GET* como el *POST*, en función del recurso accedido, así como de los campos adicionales que pueden incorporar, podrían ser usados tanto para una primitiva como para otra. Sin embargo, se ha intentado mantener el objetivo fundamental de estos a la hora de proceder y fijar la asimilación con las primitivas mencionadas. Su formato variará sensiblemente cuando se trate de enviar el mensaje dentro de la red de sensores, pero, tanto el método *GET* como el *POST*, tendrán un mensaje homólogo que realizará su misma función. Estos mensajes se detallarán más adelante (véase la sección 4.5).

Por lo tanto, cuando un agente quiera requerir alguna información o la realización de una tarea a otro agente (incluidos agentes que puedan estar en redes de sensores diferentes), podrá hacerlo a través de una petición *GET* o *POST*[15]. Estas peticiones no serán enviadas a cada sensor formateadas concretamente como un mensaje HTTP *GET* o *POST*, pero sí podrán usarse tal cual desde equipos externos a la red de sensores a través de la pasarela que se ha implementado en el presente trabajo de investigación. El

[15] En la vertiente de la comunicación dentro de la red de sensores la correspondencia será con los comandos GET_VALUE y SET_VALUE. Véase la sección 4.4.4.

formato de ambos comandos, así como la estructura de la información que permite acceder a las capacidades de cada agente se verá en la sección 4.2.

Con respecto a la comunicación entre los agentes internos a un sensor, su complejidad se ha intentado minimizar y se realiza a través de la configuración correcta de una serie de parámetros, modelados usando clases e interfaces comunes. La razón de usar este tipo de comunicación es la rapidez y facilidad de implementación en equipos con baja capacidad de memoria y proceso como son los sensores. Sin embargo, que la forma de comunicar una creencia (información) sea a través de una llamada a una función con los parámetros adecuados, no implica que la funciones que puedan desarrollarse, o que la complejidad en cuanto al nivel de detalle de la comunicación, tenga que ser necesariamente sencillo o simple.

Cada uno de estas funciones y parámetros que conforman la comunicación entre los agentes internos a un sensor serán estudiados en la sección correspondiente al agente en cuestión.

3.5 AGENTE DE GESTIÓN

El Agente de Gestión (AG) es el encargado del control de todos los procesos de aplicación dentro del sensor, incluyendo los otros dos agentes. Así pues, todas las operaciones de gestión de recursos que se diseñen para un sensor recaerán en el AG. La descripción del AG será a través de sus propiedades: autonomía, sociabilidad, reactividad y pro-actividad.

El AG posee un comportamiento totalmente autónomo, encargándose del modo de funcionamiento del sensor, siendo el primer sistema que se inicia en el sensor a nivel de aplicación. Su función principal es controlar los ciclos de trabajo. Además, el AG puede iniciar o detener los demás agentes cuando éste estime oportuno.

Como se ha dicho, el Agente de Gestión tiene como **objetivo principal el de controlar y establecer los ciclos de sueño y trabajo en función de la aplicación** a la que se haya destinado el sensor y de la situación propia del mismo. Cabe destacar que ésta es la máxima expresión de autonomía posible, dado que el AG es el único encargado de controlar si el sensor entra en reposo, además de la duración de este periodo. La forma concreta en la que se realiza este control será explicado en la sección 5, donde aparecen propuestos los dos sistemas de control del intervalo de sueño, el método analítico y el método basado en FRBS.

Aparte de la manera concreta en la que se realicen los cálculos para la estimación de los ciclos de sueño del sensor, la cual será detallada en la sección correspondiente, la arquitectura que presenta este agente sigue lo que previamente se ha presentado como la **arquitectura BDI ligera borrosa o FLBDIa**. Esta arquitectura ha sido aplicada al AG a través de un FRBS que modela todo el comportamiento del agente en función de una magnitud bajo estudio concreta. Así pues, el agente actuará de manera autónoma, intentando que su trabajo, el medir dicha magnitud, se realice atendiendo a dos objetivos: minimizar el error (precisión) y alargar la carga de la batería cuanto sea posible (duración).

La precisión en la tarea será buscada a través de limitar el tiempo que el sensor está dormido, dado que a mayores tiempos de sueño, mayor posibilidad existe de la pérdida de un evento importante. Por otro lado, el conseguir que el sensor pueda realizar su tarea durante un largo tiempo se contrapone al anterior objetivo, dado que a menor

tiempo de sueño, mayor número de veces que se despertará el sensor y mayor será el consumo de batería, acortando de esta manera el tiempo de funcionamiento del mismo. He ahí, que es en el compromiso entre ambos, así como en los propios valores captados del entorno y el nivel de batería, lo que conformará el comportamiento del AG.

Lo anteriormente expresado no es más que un resumen de las creencias, deseos e intenciones que modelan este agente. Dado que la FLBDIa necesita emplear un FRBS, su explicación detallada se verá en el apartado 5.2. Además, como elemento de comparación, también se mostrará el sistema analítico que se usó para contrastar el control del ciclo de sueño sin sistemas basados en el conocimiento.

Cabe destacar que el comportamiento modelado por la arquitectura BDI ligera borrosa presenta también dos de las restantes propiedades atribuibles a un agente, la **reactividad, y la pro-actividad**. El comportamiento del AG es claramente reactivo, ya que continuamente basa su comportamiento en elementos externos a él, como son el valor de la magnitud que está monitorizando, así como la carga de la batería del sensor. Por otro lado, la pro-actividad se manifiesta en la modificación del ciclo de sueño, que permitirá ajustar el tiempo que el sensor esté en activo a través del compromiso entre precisión y duración de la batería.

La **sociabilidad**, como se ha comentado en la sección 3.2, le viene dada al AG a través del protocolo diseñado específicamente en el presente trabajo de investigación. Este protocolo, detallado en la sección 4, proporcionará la capacidad de transportar los mensajes del lenguaje entre agentes ubicados en sensores diferentes.

En cuanto a la **comunicación interna** entre los agentes que conforman un sensor, la forma en que esta se lleva a cabo es a través de parámetros de funciones, si bien estos parámetros, así como sus posibles valores, coinciden con los predicados de las dos primitivas usadas en el lenguaje (*inform* y *request*).

En concreto los predicados posibles, o parámetros a informar o requerir aparecen detallados en la Tabla 4.

3.6 AGENTE DE CONTROL DE APLICACIÓN

El Agente de Control de Aplicación (ACA) es el encargado de interactuar con el entorno en función de la tarea que se le encomiende, usando para ello sistemas basados en el conocimiento.

En el diseño del ACA no se ha definido un comportamiento concreto, ya que está diseñado para que pueda incorporar cualquier sistema de decisión basado en FRBS. De esta manera el ACA, recibiendo la base de conocimiento apropiada, ya sea de parte de otro agente o del administrador de la red de sensores, puede modificar o actualizar su funcionamiento para adaptarse a las nuevas necesidades de una aplicación.

A pesar de esta capacidad de configuración, su **arquitectura como agente es equivalente a la detallada para el AG y sigue el modelo BDI ligero borroso**. El ACA posee su autonomía basada en la capacidad de adaptarse a diferentes tareas, a través de la incorporación de bases de conocimiento que modelen su comportamiento. Posee la capacidad de acceso a todas las sondas y actuadores instaladas en el sensor, así como al almacenamiento interno del mismo.

Tabla 4. Parámetros del Agente de Gestión

Parámetro[16]	Código[17]	Acceso[18]	Longitud (bits)	Descripción
MODE	0	R/W	8	Este parámetro controla el modo de funcionamiento general del sensor en el caso de que no se necesite que el AG emplee su comportamiento autónomo. Este parámetro se diseñó para poder dar cobertura a otros tipos de funcionamientos no basados en sistemas multi-agente. El valor del parámetro *MODE* es un byte con signo. Los El modo de funcionamiento puede ser: • MODE_SLEEPANDRUN (valor 0): El sensor está la mayor parte del tiempo dormido, en sueño profundo (o ligero según el caso) y tras este periodo despierta para realizar las acciones en él programadas, y las que tengan encargadas el Agente de Comunicaciones y el Agente de Aplicación. El tiempo en sueño profundo lo marca el parámetro *SLEEPTIME*. • MODE_RUNIFPOWERED (valor 1): El sensor funcionará de manera continua sin tiempo de sueño solo si está alimentado físicamente a una fuente de energía externa. En otro caso se comportará como con el tipo *MODE_SLEEPANDRUN*. • MODE_FORCEDTORUN (valor 2): El sensor permanecerá permanentemente encendido sin entrar en sueño profundo, independientemente de si está conectado, o no, a una fuente de alimentación externa.
SLEEPTIME	1	R/W	64	Tiempo que el sensor debe estar en sueño profundo, o ligero en su caso, entre periodos de actividad. Es un valor de 64 bits y se mide en milisegundos. El valor 0 es un valor de inicio y no es válido, así como los valores negativos resultarán en un informe de error.
SENSORID	2	R/W	8	Identificador único dado al sensor dentro de la red. Se usa para identificar al sensor como recurso dentro de la jerarquía definida para el protocolo de comunicación (véase el apartado 4.2)
ENABLE_ADA[19]	3	R/W	8	Habilita/deshabilita el funcionamiento autónomo del AG. Los valores que puede tomar son los siguientes: • ADA_NOTACTIVATED (valor 0): El AG desactiva su comportamiento autónomo y se rige por la configuración del parámetro *MODE*. • ADA_DELTASYSTEM (valor 1): El AG se comporta de manera autónoma usando un sistema de ajuste dinámico del tiempo de ciclo basado en funciones analíticas y la historia (Véase el punto 5.1) • ADA_FUZZYSYSTEM (valor 2): El AG se comporta de manera autónoma usando la arquitectura BDI ligera borrosa.

Para posibilitar el acceso a la información del entorno desde el modelo de conocimiento que se esté usando, todas las sondas, así como los actuadores posibles, se han introducido en la arquitectura jerárquica de recursos creada en el presente trabajo de investigación (véase la sección 4.2) y que aúna todos los sistemas presentes en la red de sensores. De esta manera, cualquier diseño previsto para un BDI ligero borroso podrá ponerse en contacto con el exterior simplemente definiendo el identificador de la sonda, o actuador, concreto que formará parte de las creencias del modelo.

[16] Dado que la implementación de todos los sistemas se ha realizado en inglés, por motivos de internacionalización de los resultados, todas las referencias a la implementación de los agentes, protocolos, denominación de clases o subsistemas, se mantendrán tal cual aparezcan en el código en todo el presente trabajo de investigación, para así de esa manera, facilitar su cotejo y revisión, ya sea en las contribuciones de la presente tesis, como en la implementación de los mismos.

[17] El código representa el valor numérico asignado a este parámetro dentro de la jerarquía de recursos definida para el protocolo de comunicaciones (véase la sección 4.2).

[18] Para definir el tipo de acceso a un parámetro se han usado las siglas R, W, o R/W, del inglés *Read*, *Write* o *Read and Write*, que significan, respectivamente, que este parámetro puede ser solo leído, sólo modificado, o leído y modificado indistintamente. Así pues si un parámetro se define como R, puede ser predicado de la primitiva *inform*, y si se define como W puede ser predicado de la primitiva *request*.

[19] Acrónimo de *Automatic Dutytime Adjustment* o ajuste automático del tiempo de trabajo.

Con respecto a la **reactividad**, queda patente que el ACA está perfectamente conectado con el entorno a través de todas las sondas disponibles. Además, en función de la aplicación concreta a la que se destine, el comportamiento del agente será **pro-activo**, ya que en general, su función es la de conseguir una modificación en el entorno, en función de la situación actual del mismo, que favorezca la aparición o desaparición de un fenómeno concreto.

Con respecto a las **capacidades de comunicación** se deben hacer las mismas consideraciones que para el AG, si bien los parámetros disponibles para el ACA son diferentes y se presentan en la siguiente tabla:

Tabla 5. Parámetros del Agente de Control de Aplicación.

Parámetro	Código	Acceso	Longitud (bits)	Descripción
MODE	0	R/W	8	Este parámetro controla el funcionamiento del ACA para permitir un funcionamiento determinista, basado en tareas o en FRBS. El valor del modo es un byte con signo El modo de funcionamiento puede ser: • APP_MODE_OFF (valor 0): El agente finaliza completamente una vez termine la última tarea pendiente. El AG podrá iniciarlo de nuevo cuando lo estime oportuno. • APP_MODE_RUNONCE (valor 1): El proceso encargado al ACA se ejecuta una sola vez después de despertarse en sensor. Este proceso es el modo normal de funcionamiento, ya que la tarea a realizar puede tener duración indefinida. El proceso a realizar es el determinado por el agente a través de la definición de una base de conocimiento para su configuración como agente BDI ligero borroso. • APP_MODE_TEST (valor 2): El ACA toma muestras cada vez que el sensor se despierte, pero no ejecuta el proceso principal. El número de muestras que toma lo controla el parámetro TESTS. • APP_MODE_TESTANDRUN (valor 3): La aplicación toma muestras cada vez que el sensor se despierte y ejecuta el proceso principal. El número de muestras que toma lo controla el parámetro TESTS. • APP_MODE_DONOTHING (valor 4): El ACA no realiza ninguna operación concreta esperando un cambio en sus tareas.
TESTS	1	R/W	32	Indica el número de medidas que se deben tomar cuando se está en el modo *APP_MODE_TEST* o *APP_MODE_TESTANDRUN*. Una vez alcanzado el número total de muestras el ACA se pondrá en modo *APP_MODE_OFF*.
PROBE	2	R	8	Indica la sonda que se debe usar para tomar medidas. En este modo de funcionamiento tan solo se ha permitido el acceso a una sonda. La posibilidad de acceder a todas ellas está prevista a través del proceso definido dentro del comportamiento autónomo del ACA como agente.

Cabe destacar que el funcionamiento del ACA está condicionado al momento en el que el sensor esté activo, lo cual depende del AG. Así pues, el proceso que éste tiene que llevar a cabo, estará activo el tiempo determinado por la propia duración del mismo y por el ciclo de actividad determinado por el AG.

Formando parte del ACA existen **tres subsistemas**, entre los cuales están los ya comentados de acceso a las **sondas** (recurso *probes*) y **actuadores** (recurso *actuators*), completados con el **subsistema de lógica borrosa** que modela el comportamiento autónomo de la FLBDIa. Estos sub-sistemas están integrados en la estructura de recursos que modela toda la red de sensores y que se presentan en el apartado 4.2. Concretamente, el recurso que permite el acceso al sub-sistema de soporte a la ejecución de FRBS, se denomina *frbs*.

El recurso *frbs* tiene un elemento que depende de él, el cual es la base de conocimiento asociada (recurso *kb*). Ésta última puede ser modificada remotamente a través del protocolo de comunicaciones.

3.7 AGENTE DE COMUNICACIONES

El Agente de Comunicaciones o AC es el encargado de controlar los aspectos que conciernen a las comunicaciones, desde el interfaz radio instalada en el sensor, al protocolo de comunicaciones usado para la comunicación de datos de aplicación. En general, el AC tiene el encargo de enviar y recibir la información de una manera controlada y eficiente, intentando hacer el menor gasto de recursos posible.

Además, el AC incorporará la función de traductor con el exterior del sensor, a través del empleo del protocolo de comunicaciones diseñado como parte de las contribuciones de la presente tesis. Se ha elegido el modelo de agente traductor por el ahorro de recursos que esto supone. De esta manera, si tan solo hay un punto de acceso de comunicaciones que analice los mensajes recibidos, en contra de un punto de comunicaciones y traductor para cada agente, se evita un mayor número de hebras de ejecución, y por lo tanto, un mayor gasto de memoria y proceso, lo que incrementa el tiempo de ejecución, y en consecuencia el gasto de batería. Así pues, cuando el AC recibe un mensaje para otro de los agentes presentes en el sensor, éste analiza y extrae la información de la unidad de datos del protocolo o PDU (del inglés, *protocol data unit*). Una vez preparada esta información, se envía al agente destinatario a través del interfaz interno que dispone cada uno de ellos.

La **estructura del AC**, al igual que los demás agentes que integran el sensor, será descrita en virtud de las propiedades de autonomía, reactividad, pro-actividad y sociabilidad. Cabe destacar que esta última, la sociabilidad, ha sido comentada con detalle en la sección 3.2 y que será completada con la descripción de la jerarquía de recursos y el protocolo de comunicación en las secciones 4.2 y 4.4 respectivamente.

Con respecto a la propiedad de **autonomía**, el AC tiene un comportamiento concreto, vinculado a las comunicaciones que se establezcan entre los agentes remotos e internos. Este comportamiento está definido por una serie de estados que definen la tarea concreta del agente en cada momento, así como el tipo de información que espera recibir. El motivo de que el AC se rija por una máquina de estados finitos se debe al diseño del protocolo de comunicaciones, por lo que su actividad está estrechamente ligada a las fases definidas en dicho protocolo. La máquina de estados del protocolo de comunicaciones se verá en la sección 4.4.2.

La propiedad de **reactividad**, en este caso queda vinculada a la actividad de la red, respondiendo en función del estado en el que se encuentre y de los mensajes recibidos de otros sensores. Las posibles respuestas vendrán marcadas por el comportamiento definido por los estados del protocolo, y la información disponible en ese momento en el sensor.

La **pro-actividad** del AC le viene dada por su propia capacidad de comunicación.

Los parámetros disponibles para el AC se presentan a continuación en la Tabla 6.

Tabla 6. Parámetros del Agente de Comunicaciones.

Parámetro	Código	Acceso	Longitud (bits)	Descripción
MODE	0	R/W	8	El parámetro *MODE*, al igual que para lo demás agentes, es el que controla el modo de funcionamiento por defecto del AC. Los diferentes modos de funcionamiento se detallan a continuación: • COMMAGENT_MODE_OFF (valor 0): El AC finaliza su funcionamiento de manera ordenada. Puede ser vuelto a activar por el Agente de Gestión. • COMMAGENT_MODE_ALERT (valor 1): El AC tan solo se encarga de comunicar alarmas salientes y no atiende a peticiones del exterior del sensor. • COMMAGENT_MODE_AWAKE (valor 2): Este es el modo de funcionamiento por defecto del AC. En este modo el AC se despierta cada cierto tiempo, (ver parámetro *MODE_AWAKE_TIMEOUT* en esta misma tabla) para comenzar el proceso definido en la máquina de estados del protocolo (véase el apartado 4.4.2). • COMMAGENT_MODE_PERMANENT (valor 3): El AC no entra en reposo en ningún momento y está permanentemente esperando recibir algún mensaje del exterior.
POWER	1	R/W	32	Establece el valor de potencia para las transmisiones salientes. Los valores vienen dados por el estándar IEEE 802.15.4, yendo desde el nivel -32, para la mínima potencia, al 31 para la máxima[20]..
MODE_AWAKE_TIMEOUT	2	R/W	32	Tiempo en milisegundos que permanece el AC en reposo antes de iniciar de nuevo su funcionamiento en el modo *COMMAGENT_MODE_AWAKE*.

3.8 PROPIEDADES DE LA ARQUITECTURA MULTI-AGENTE

Una vez descrita la arquitectura multi-agente de cada sensor, se detallarán a continuación las diferentes propiedades que posee la misma debido a su estructura, las cuales son: escalabilidad, replicación y adaptabilidad.

3.8.1 ESCALABILIDAD

En primer lugar, una de las propiedades que posee esta arquitectura multi-agente, la cual ya se mencionó al principio del Capítulo I, es la escalabilidad. Esta característica es propia de los sistemas multi-agente y se mantiene en la presente arquitectura. En concreto, en la arquitectura presentada, el único agente que debe estar siempre en funcionamiento es el Agente de Gestión, los demás pueden no estar en ejecución si no son necesarios. Además, es posible incrementar el número de agentes, o de redefinir los existentes, asignándoles nuevas funciones o para incorporar nuevos comportamientos. En cualquier caso, el número y la complejidad de los mismos dependerán de los recursos disponibles en el sensor. Los agentes podrán comunicarse entre ellos y con el exterior con los mecanismos existentes, ya que gracias a la jerarquía de recursos diseñada en el presente trabajo (véase la sección 4.2), estarán accesibles simplemente con la definición de nuevos identificadores en el nivel correspondiente. Por lo tanto, no solo se consigue escalabilidad en el número de agentes, sino también en la cantidad de recursos necesarios, así como en la funcionalidad de un sensor.

3.8.2 REPLICACIÓN

La otra propiedad que posee la estructura de agentes presentada es la replicación, entendida ésta como la capacidad de extender el comportamiento de un agente concreto

[20] Dentro de este rango, no se usan todos los posibles niveles dado que el chip de radio usado para las comunicaciones con el estándar IEEE 802.15.4 es el CC2420. Dicho dispositivo posee tan sólo 22 niveles de potencia entre el máximo (31 = 0 dBm) y el mínimo (-32 = -24 dBm), siendo este rango de niveles los que efectivamente se usan

a otro sensor a través de proceso de comunicación simple. La primera condición es posible gracias a la definición de los agentes como agentes BDI ligeros borrosos, ya que su comportamiento está modelado a través de una base de conocimiento. El segundo condicionante, concerniente a la manera en la que se comunican estos parámetros, es posible gracias al protocolo de comunicación desarrollado, el cual permite el intercambio de bases de conocimiento entre sensores, al igual que cualquier otro tipo de dato primitivo, manteniendo así la integridad de la misma. Esta propiedad también se basa en la propia estructura de definición de recursos de la red de sensores, que será vista en la sección 4.2.

3.8.3 ADAPTABILIDAD

La tercera propiedad es también derivada de la propia estructura BDI ligera borrosa. Así pues, esta estructura permite la adaptación del agente en su comportamiento con la simple modificación de alguno de sus elementos, ya sea en las creencias, deseos o metas. Este ajuste se realiza a través de la modificación de la base de conocimiento que modela el agente BDI ligero borroso. De esta manera, añadiendo o cambiando un conjunto borroso, un rango en una variable, o alguno de los antecedentes o consecuentes en una regla, un agente puede adaptar su comportamiento tan sólo con la recepción de una cantidad muy reducida de datos de la red. Además, esta capacidad posibilita el ajuste del comportamiento del agente a través de mecanismos de aprendizaje internos o externos, que determinen nuevos valores para algunos de los parámetros que le afecten.

4 PROTOCOLO DE COMUNICACIÓN

En esta sección se detallará el protocolo de comunicaciones desarrollado dentro de los trabajos de investigación de la presente tesis. En primer lugar, se hará una breve introducción a los protocolos de comunicaciones sobre redes de sensores y la justificación de las decisiones tomadas. A continuación se presentará la jerarquía y los recursos diseñados para dar cobertura a todos los posibles sistemas disponibles en una red de sensores, dando paso a la explicación del protocolo en sí. Por último podrá encontrarse la pasarela con Internet que se ha desarrollado.

4.1 INTRODUCCIÓN

Como se ha podido observar en la revisión de la investigación sobre redes de sensores del Capítulo I, gran parte de los esfuerzos investigadores en el campo de las redes de sensores han estado enfocados a la optimización de recursos, comunicaciones y organización de la red. Evidentemente, también se han dedicado esfuerzos al diseño y/o adaptación de protocolos de comunicación, sobre todo a nivel de red y transporte, siendo un buen ejemplo de esto las iniciativas que ha llevado a la extensión de Internet a las redes de sensores a través del protocolo 6LoWPAN (Kushalnagar et al. September 2007) o la tecnología y protocolos Zigbee (ZigBee Alliance).

Como se puede intuir, **los trabajos sobre nuevos protocolos se han orientado principalmente a la capa de red**, y también, aunque con menor presencia, a la capa de transporte y aplicación. Esto se debe a que la aparición del estándar IEEE 802.15.4 en el 2003, consecuencia de los estudios de finales del siglo pasado y principio de este, afianzó todo lo concerniente a la capa de enlace e inferiores, aunque por supuesto, existen numerosos trabajos que han seguido estudiando e intentando mejorar la capa de enlace y el medio físico. Algunas de las contribuciones más destacables sobre mejoras y/o modificaciones en la capa de enlace son revisadas en (Baronti et al. 2007). Concretamente, en el trabajo de Baronti y otros también aparecen protocolos de capas superiores, tales como aquellos dedicados al enrutamiento o al transporte confiable.

Otro aspecto a tener en cuenta con los protocolos de comunicación es, si siguen o no, el **modelo de capas OSI**. Así pues, el enfoque más común ha sido el mantener la estructura de capas de OSI (Yick et al. 2008), derivado fundamentalmente de la necesidad de comunicaciones radio eficientes y compatibilidad entre redes de sensores. Sin embargo, existe otra vertiente en cuanto al diseño de protocolos de comunicaciones, la cual propugna por un diseño vertical, es decir, que el desarrollo de un nuevo protocolo se realiza involucrando a varias de las capas definidas por OSI. Estos protocolos son los denominados protocolos inter-capa o en inglés, *cross-layer*. Un **protocolo *cross-layer*** pretende optimizar el uso de energía de las comunicaciones en un sensor a costa de perder generalidad y compatibilidad entre diferentes arquitecturas y familias de sensores (Akyildiz et al. 2006). Los protocolos *cross-layer*, además de la optimización de uso de energía interna en el sensor, intentan aprovechar la posible correlación existente entre los nodos de una red, como sucede con la toma de medidas por los sensores, siempre y cuando la densidad de nodos sea tal que propicie dicha correlación. En (Melodia et al. 2006) se puede ver una revisión de algunos protocolos *cross-layer*, y como muestra, están los trabajos descritos en (Xing et al. 2009) para la gestión eficiente de la energía, y (Vuran & Akyildiz 2010), donde se integran las capas desde el nivel físico al de transporte.

Los posibles **problemas de compatibilidad de los protocolos *cross-layer*** se deben a que basan su funcionamiento en una optimización de recursos aunando funciones típicas

de protocolos de enlace, red y transporte, como el enfoque de planos propuesto por Akyildiz y Kasimoglu en (Akyildiz & Kasimoglu 2004).

De esta manera, teniendo en cuenta los objetivos del presente trabajo, la pérdida de versatilidad y/o escalabilidad de los protocolos *cross-layer* a la hora de poder asistir a diferentes aplicaciones, no es asumible, dado que se busca un protocolo de aplicación capaz de dar soporte a sistemas basados en el conocimiento y de gestión de sensores lo más general posible, y que optimice el envío de información. Así pues, en cuanto al protocolo desarrollado, se ha optado por diseñarlo para la capa de aplicación, lo cual no impediría su uso en un dispositivo que siguiera la torre de protocolos OSI o un enfoque *cross-layer*, pero que permita la compatibilidad y escalabilidad en su empleo en redes de sensores de diferentes familias.

El **enfoque de capas OSI** posee detractores entre autores destacados, como se refleja en Akyildiz y otros (Akyildiz et al. 2006), donde se supedita una eficiente gestión de todos los recursos y comunicaciones a las soluciones *cross-layer*, afirmando a la vez que hasta ese momento no existe un protocolo *cross-layer* unificado que abarque desde la capa física a la de transporte, evidentemente por la complejidad que esto conlleva. De esta manera, del estudio de trabajos recientes, como los antes comentados de Xing y otros (Xing et al. 2009) y Vuran y Akyildiz (Vuran & Akyildiz 2010) se desprende que las nuevas orientaciones en el diseño de protocolos para redes de sensores tienden a la integración *cross-layer*. Sin embargo, esta integración no suele llegar al nivel de aplicación, dada la complejidad intrínseca de esta capa por la gran variedad de tipos de aplicaciones y servicios posibles.

Por lo tanto, el **enfoque por el que se opta en este trabajo** se presume adecuado, manteniendo el nivel de aplicación separado de las capas inferiores pero guiado por sistemas inteligentes que en conjunción llevarían a un comportamiento cercano a un protocolo *cross-layer*. Esto sería posible gracias a que estos sistemas inteligentes, tanto internos como externos al sensor, usando información de capas inferiores y del entorno, ayuden a la toma de decisiones de una aplicación concreta para la que se ha destinado el sensor. Este enfoque, sin llegar a ser *cross-layer*, puede parecerse a la arquitectura de planos comentada en (Akyildiz & Kasimoglu 2004), sin embargo, en nuestro caso los planos serán sustituidos por agentes inteligentes.

Así pues, siguiendo el enfoque de capas OSI, en los siguientes apartados se detallará el protocolo de aplicación diseñado para demostrar la segunda hipótesis de la presente tesis. En primer lugar se presentará la estructura jerárquica de recursos en la que se apoya la información accesible por el protocolo, seguida de los tipos de comandos disponibles y su formato correspondiente, la máquina de estados del protocolo y por último la pasarela para acceso desde Internet.

4.2 ESTRUCTURA JERÁRQUICA DE RECURSOS

Dentro de las redes de sensores, al igual que en otros sistemas de información distribuida, un aspecto importante es la manera en la que se define y estructura la información que está disponible, así como la forma de acceder a ella.

Con respecto a la definición y diseño de la estructura de la información disponible en las redes de sensores, teniendo en cuenta las especiales características de éstas, se ha buscado una solución que permita identificar, no sólo a un sensor, sino todos los elementos que integren la propia red, además de todos los agentes, sub-sistemas y

parámetros que se integran en dicho sensor. En el diseño de la citada estructura de información se han tenido en cuenta los siguientes requisitos:

- **Flexibilidad**. Se ha buscado que no haya una vinculación rígida, ni una estructura monolítica para que de esta manera se pueda adaptar a una mayor variedad de redes de sensores.
- **Escalabilidad**. Al igual que con otros aspectos de las redes de sensores, la escalabilidad en la definición y acceso a la información es crucial. Por lo tanto, se ha diseñado una estructura que permite ampliación de la información horizontalmente (nuevos recursos), o verticalmente (detalles y componentes de los recursos).
- **Simplicidad**. Cada elemento se define tan solo por un valor numérico de 8 bits, por lo que a la hora de la transferencia por la red se consigue un considerable ahorro de recursos. Los nemónicos tan solo son usados para facilitar la legibilidad de la propia estructura.
- **Jerárquica**. La estructura de la información es completamente jerárquica, de manera que el nivel al que pertenece un recurso se le suponen unas determinadas propiedades dadas por el propio nivel. Además, cada nivel puede tener niveles inferiores que permiten la definición de elementos que lo componen. En un principio se han previsto sietes niveles dentro de la jerarquía: raíz, sistema, sub-sistema, elemento, agente, componente y sub-componente. Esta denominación es orientativa y en ningún caso es vinculante en cuanto al sistema que puede albergar cada nivel. Concretamente, la denominación presentada es para los niveles de un sensor que contenga una arquitectura multi-agente como la presentada en el presente trabajo de investigación.

Teniendo en cuenta los anteriores requisitos, la estructura de recursos finalmente propuesta se ha diseñado de forma similar a como se estructuran los *Uniform Resource Identifier* (URI) (Berners-Lee et al. January 2005) para el acceso a los recursos en servicios como la *world wide web* (www). Se ha optado por esta estructura, para buscar la mayor compatibilidad posible entre HTTP y la forma en la que el protocolo presentado en esta tesis accede a los recursos disponibles en una red de sensores. Además, esto facilitará la creación y soporte de bancos de pruebas (*test-beds*) a través de una pasarela entre HTTP y el nuevo protocolo para redes de sensores (esta pasarela será comentada en la sección 4.5).

En la Figura 9 puede observarse un ejemplo de recursos dentro de una red de sensores[21]. También aparece como ejemplo un sensor que seguiría la arquitectura multi-agente propuesta.

Como puede verse, cada recurso tiene un identificador numérico asociado, que puede ir desde 0 a 255 (8 bits), además de ser único entre los recursos de su mismo nivel en su rama de la jerarquía. De esta manera, cada recurso puede tener 256 recursos dependientes de él en el nivel inferior.

Adicionalmente, un recurso puede poseer hasta 256 parámetros, los cuales se usarán para configurarlo, modelar su comportamiento o funcionalidad. Estos parámetros, al igual que los recursos, se identifican por un número entero de 8 bits, aunque el tipo de dato o información que contiene depende de cada definición concreta del parámetro. Estos podrán ser vistos en la descripción de cada recurso más adelante. Como ejemplo

[21] Realmente, como se explicará más adelante, los nemónicos usados en la estructura serán en inglés.

inicial, recuérdense los parámetros de configuración de cada agente, que como se ha podido observar están incluidos en la estructura.

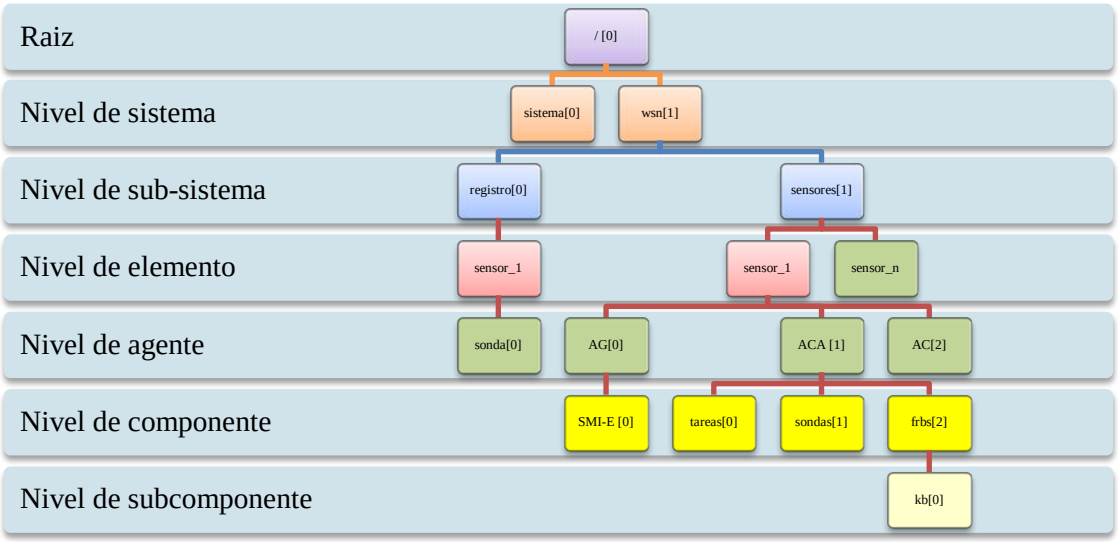


Figura 9. Ejemplo de una jerarquía de recursos.

El hecho de elegir los identificadores de tan solo 8 bits es tan sólo por motivos de ahorro de recursos, tanto en comunicaciones, como en memoria. La posibilidad de elegir identificadores de mayor número de bits, si bien no supondría un cambio drástico en el protocolo y la implementación de las comunicaciones, se ha descartado porque se estima que la capacidad de configurar una red de hasta 256 nodos por cada elemento raíz, es suficiente para los objetivos del presente trabajo de investigación.

Para aumentar la legibilidad de la estructura de recursos, cada uno está identificado con un breve nemónico. Este nemónico tan solo se utiliza para la propia representación y para la formación de la URL que permitiría el acceso a la red de sensores a través de la pasarela HTTP[22].

4.2.1 FORMATO DE DEFINICIÓN DE RECURSOS Y PARÁMETROS

Previa a la descripción de cada recurso, se ha estimado conveniente presentar la estructura que se seguirá a la hora de mostrar los detalles de cada uno. Así pues, en la descripción de cada recurso se podrán encontrar los siguientes elementos:

- Nombre del recurso: el título del apartado coincidirá con el nombre del recurso, y a continuación, entre paréntesis, irá el nemónico del mismo precedido de la ruta completa de recursos[23] hasta el elemento raíz[24]. Se debe remarcar que los nemónicos usados se han definido en inglés por mantener el mismo idioma usado para las publicaciones que avalan la presente tesis y evitar ambigüedades. Cuando un recurso está declarado en plural, implica que el siguiente nivel lo forman un número variable de recursos de ese tipo. Así pues, en el nivel de sub-sistema, el recurso *sensors*, al tener un nemónico en plural, indica que tiene por debajo cada uno de los sensores que forman la red.

[22] Esta pasarela funcionara como un proxy haciendo la traducción de las llamadas HTTP al protocolo de comunicaciones diseñado en el presente trabajo de investigación.
[23] Esta ruta de recursos es la que será usada para las peticiones HTTP a la pasarela.
[24] El elemento raíz se simbolizará simplemente con una barra invertida '/'.

- Descripción: Cada apartado comienza con una descripción del recurso.
- Recursos: Si aparece este apartado, informa de los recursos definidos hasta el momento que están en el nivel inferior de la jerarquía y son descendientes del recurso que se describe.
- Parámetros: Los parámetros que modelan y complementan cada recurso tienen una serie de propiedades comunes que los definen, aparte por supuesto, del significado dependiente de su función y características. En la definición de cada parámetro, aparece en primer lugar su nemónico seguido de un valor numérico entre corchetes. Así pues, los parámetros podrán ser accedidos por ese número en vez de por su nombre. Esto es de particular utilidad a la hora de encapsular un parámetro para su envío por la red de sensores. En segundo lugar se indica el modo de acceso al parámetro. Los posibles valores que puede tomar el modo de acceso aparecen a continuación junto a sus siglas provenientes del inglés:
    - No accesible o NA del inglés *not accesible*. No se lee ni se modifica su valor, tan sólo sirve para identificar un recurso concreto, función o valor a conseguir con un comando.
    - Modo de acceso solo lectura o R del inglés *readable*.
    - Modo de acceso de solo escritura o W del inglés *writeable*.
    - Modo de acceso de lectura y escritura: R/W.

4.3 RECURSOS

En los apartados siguientes se mostrarán los recursos que se han usado para la realización de las pruebas del presente trabajo de investigación, así como para el cumplimiento de los objetivos del mismo, concernientes a la implementación de un test-bed. En la Figura 10 aparece la estructura de recursos empleada con los nemónicos en inglés.

Cabe destacar, que dadas las características de la estructura de la información presentada, la inclusión de un nuevo recurso dentro de la misma tan sólo necesitaría del soporte software adecuado en el sistema físico que dé acceso a dicho recurso. Por ejemplo, si un sensor dispusiera de un nuevo tipo de sonda para medir la presión atmosférica, y a esa sonda se le atribuyera el identificador 16 dentro del recurso *sondas* (*probes*) del nivel de sub-componente, tan sólo se tendría que añadir el soporte para la lectura de dicha sonda en ese sensor, y no se necesitaría modificación alguna en la pasarela HTTP, estación base o protocolo de comunicaciones (como más tarde se verá).

Como se podrá apreciar, en la descripción de algunos recursos se incorpora también la de recursos de niveles inferiores en la jerarquía. Esto se ha decidido poner así por motivos de claridad y ahorro expositivo, ya que no se altera la comprensión del mismo y añade mayor coherencia a la descripción del propio recurso.

4.3.1 SISTEMA (/SYSTEM)

Este recurso modela el equipo que se encarga de hacer de pasarela entre Internet y la red de sensores. Se debe tener en cuenta que este equipo no es parte de la red de sensores, y tan sólo es necesario para poder acceder a dicha red desde Internet o una red IP.

Por debajo de este recurso se deberán añadir los recursos que dependan de la puerta de enlace y que den información de ésta o del test-bed.

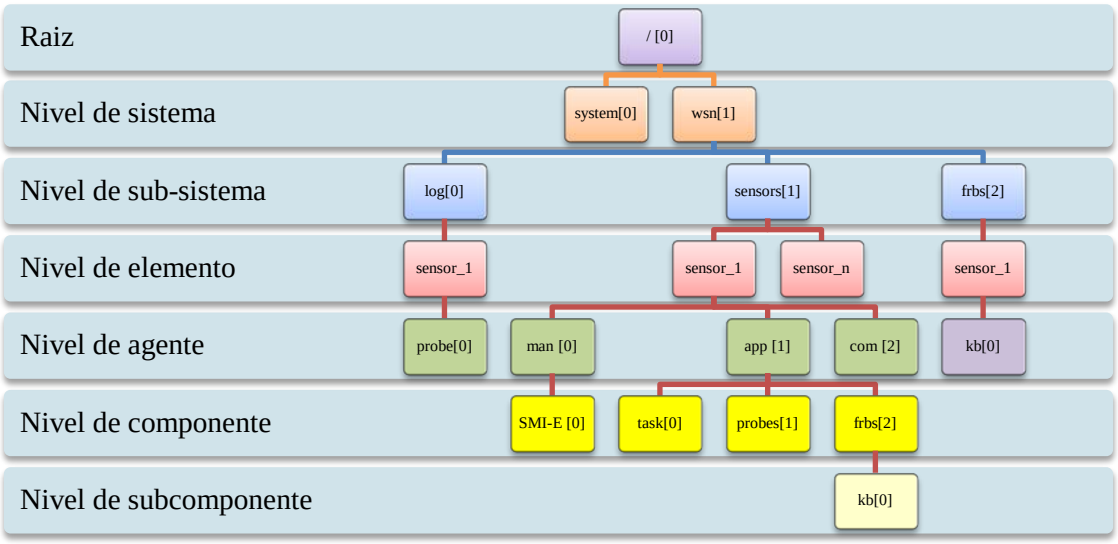


Figura 10. Estructura jerárquica de recursos usada en las pruebas del *test-beb*.

4.3.1.1 Parámetros

- `Location [0] R`: cadena con la ubicación física de la red de sensores o *test-bed*.

4.3.2 RED DE SENSORES INALÁMBRICOS (/WSN)

Bajo este recurso se organizan los diferentes dispositivos (sensores inalámbricos) que forman la red sensores. Este recurso da acceso a cada uno de los nodos presentes en la red a través del siguiente nivel de la jerarquía.

4.3.2.1 Recursos

- `log [0]`: registro de actividad de cada una de las medidas tomadas por cada sensor.
- `sensors [1]`: recurso que aglutina el acceso y control a cada uno de los sensores disponibles en la red.

4.3.3 REGISTRO DE ACTIVIDAD DE LA RED (/WSN/LOG)

Este recurso da acceso a los registros de información de cada uno de los sensores disponibles en la red. Esta información es almacenada en el sistema que hace de puerta de enlace entre la red de sensores e Internet. Fundamentalmente, el tipo de información almacenada son las medidas que los sensores han enviado a la estación base, para que de esta forma puedan estar accesibles en cualquier momento a través de la pasarela de Internet. También son almacenadas todas las peticiones enviadas a los sensores de la red, tanto si han sido atendidas, como si no.

4.3.3.1 Recursos

- `Sensor`: habrá un recurso diferente por cada sensor del que se tengan datos, estando identificados estos nuevos recursos con el identificador correspondiente al sensor. A su vez, cada sensor dispone del nivel de componente, en el cual se puede acceder a su registro de medidas a través de la ruta /wsn/log/<identificador de sensor>/*probe*, que guardarán los registros de las

medidas tomadas por las sondas del sensor. El recurso /wsn/log/<identificador de sensor>/*probe* se detalla en el siguiente apartado.

4.3.4 REGISTRO DE ACTIVIDAD DE LAS SONDAS DE UN SENSOR (/WSN/LOG/<*IDENTIFICADOR DE SENSOR*>/PROBE)

Cada uno de los sensores disponibles en la red contará en el servidor con un registro de peticiones de medida, las cuales pueden haber sido tomadas, estar pendientes o haber fallado. Para solicitar o buscar cada uno de estos grupos de medidas, además de elegir la sonda adecuada, se usan los parámetros *id* y *status*.

4.3.4.1 Parámetros

- `id [0] NA`. Entero positivo de 8 bits mayor que cero: identifica la sonda de la que se quiere obtener el histórico de peticiones de medida.
- `status [1] R`. Entero positivo de 8 bits mayor que cero: indica el estado de las medidas que se desean obtener. El valor por defecto es cero y cualquier valor que no sea un número entre los valores válidos resultará en un error a la hora de hacer una petición. Los valores que puede tomar son los siguientes:
  - Valor 0: todas las medidas disponibles en el servidor para esa sonda. Si no se especifica se toma este valor por defecto.
  - Valor 1: medidas aún no enviadas al sensor.
  - Valor 2: medidas enviadas al sensor a la espera de respuesta. Este estado, en funcionamiento normal durará tan sólo unos segundos.
  - Valor 3: medida tomada completamente y sin errores o comando ejecutado.
  - Valor 4: medida o comando fallido.

4.3.5 SENSORES (/WSN/SENSORS/<*IDENTIFICADOR DE SENSOR*>)

Dentro de la red de sensores, a nivel de aplicación se ha definido un identificador (8 bits) de manera que un sensor pueda ser referenciado desde el exterior de la red sin necesidad de saber su dirección MAC. Este identificador es el nombre que se dará al recurso dentro de este nivel y será usado como tal para la ruta de acceso a los recursos que parten de él en la jerarquía. Así pues, todos los recursos que dependen de él jerárquicamente corresponden a sistemas o elementos que están en un sensor concreto, y por tanto son independientes de cualquier otro nodo de la red. Dado que en el presente trabajo de investigación se ha presentado una arquitectura multi-agente, los recursos que aparecen en el siguiente nivel coinciden con cada uno de los agentes ya detallados en la sección 3. Por lo tanto, no se volverán a repetir los parámetros que estos poseen, haciendo hincapié en que estos podrán ser accedidos a través del recursos correspondiente a un agente.

4.3.5.1 Recursos

- `man [0]`, del inglés *management agent*: permite el acceso a las funciones del Agente de Gestión.
  - Ruta de acceso: /wsn/sensors/<*identificador de sensor*>/man.
- `app [1]`, del inglés *application control agent*: da acceso al Agente de Control de Aplicación**.**
  - Ruta de acceso**:** /wsn/sensors/<*identificador de sensor*>/app.
- `com [2]`, del inglés *communication agent*: permite el acceso al Agente de Comunicación**.**

- Ruta de acceso**:** /wsn/sensors/<*identificador de sensor*>/com.

4.3.6 TAREAS (/WSN/SENSORS/<*IDENTIFICADOR DE SENSOR*>/APP/TASK)

Este nivel de recursos permite acceder a las tareas que tiene programadas el Agente de Control de Aplicación. Las acciones que pueden ser programadas son tan solo muestreo de una sonda concreta, una vez, o por intervalos dados un comienzo y un fin.

4.3.6.1 Parámetros

- `id [0]R/W`. Entero positivo mayor que cero: identificador de la tarea dentro del gestor de tareas que incorpora el Agente de Control de Aplicación.
- `type[1]R/W`. Entero de 8 bits: define el tipo de tarea a realizar. Los valores que puede tomar son:
  - `TASK_REPEAT_ONCE [0]`: la tarea se ejecuta tan sólo una vez, siempre y cuando se haya superado el instante de comienzo marcado por el parámetro *start* (véase más abajo).
  - `TASK_REPEAT_SCHEDULED [1]`: la tarea se repite según los intervalos marcados por el parámetro *interval* (véase más abajo). El inicio y el fin del muestro está marcado por los parámetros *start* y *end* (véase más abajo).
- `start[2] R/W`: fecha y hora de comienzo de la tarea (64 bits[25]).
- `end [3] R/W`: fecha y hora de fin de la tarea (64 bits).
- `interval[4] R/W`: indica los milisegundos tras los cuales la tarea debe ser repetida. Es un valor entero largo (64 bits).
- `probe [5] R/W`: identificador de la sonda de la que se tomará medidas en esta tarea. Es un valor de 8 bits sin signo.

4.3.7 SONDAS (/WSN/SENSORS/<*IDENTIFICADOR DE SENSOR*>/APP/PROBES)

Este nivel de recursos permite acceder a las sondas de captación de información del entorno que tiene instaladas el sensor. Cada sonda se identificará a través del parámetro id (identificador).

4.3.7.1 Parámetros

- `id [0] R`. Entero positivo de 8 bits mayor que cero: identificador de la sonda de la que se quiere obtener el histórico de peticiones de medida. El identificador indica también el tipo de magnitud. El cero es un valor de control que indica ninguna sonda. Las magnitudes que tienen asociado un identificador y se han usado en las pruebas de la presente tesis son las siguientes:
  - Valor 1: temperatura en grados Celsius.
  - Valor 2: luminosidad (las unidades dependen del sensor).
  - Valor 3: acelerómetro de tres ejes, medidos en múltiplos de la aceleración de la gravedad (g, m/s^2).
  - Valor 4: presión sonora de dBA.
  - Valor 255: resultado del FRBS. Por simplicidad se ha incorporado el resultado del FRBS (véase a continuación) como una sonda.

[25] El valor empleado en los parámetros que almacenan fechas indica el número de milisegundos transcurridos desde la medianoche del 1 de enero de 1970. Este es un formato comúnmente usado en los leguajes de programación.

- `value [1] R`: la existencia de este parámetro implica que se desea tomar el valor del sensor. Si *value* no tiene ningún valor asociado, se requerirá la lectura instantánea del momento en el que el sensor reciba el comando.
- `priority [2]R/W`: prioridad de la petición. Puede tomar los valores de 0 a 3 (2 bits). La petición podrá ser descartada por el sensor en caso de no tener la prioridad adecuada y éste no tener un nivel de batería suficiente para procesarla. Los niveles de prioridad son:
  - 0: no procesar si la batería está por debajo del 50%.
  - 1: no procesar si la batería está por debajo del 25%.
  - 2: no procesar si la batería está por debajo del 15%.
  - 3: procesar siempre.

4.3.8 SISTEMA BORROSO (/WSN/SENSORS/<*IDENTIFICADOR DE SENSOR*>/APP /FRBS)

Este recurso permite el acceso al FRBS que incorpora el Agente de Control de Aplicación para la evaluación de cualquier base de conocimiento. El funcionamiento del FRBS está condicionado a la recepción de una base de conocimiento. En el caso de que el modo de funcionamiento del ACA lo permita, y se haya recibido una base de conocimiento, el ACA ejecutará el motor de inferencia con dicha base y almacenará el resultado para que posteriormente pueda ser consultado como si fuera una sonda más (véase el comando *GET_VALUE* en la sección 4.4.4).

El recurso FRBS está gobernado por el comportamiento del ACA, por lo que en principio no se le han definido parámetros. Sin embargo, gracias a las propiedades de la estructura, este hecho podría modificarse tan sólo añadiendo la funcionalidad deseada en el sensor, estando automáticamente disponible a través del protocolo de aplicación.

Dada la importancia de la base de conocimiento para el sistema borroso, y que su inclusión como parámetro complicaría seriamente la definición de los mismos, se ha incluido ésta como un nuevo recurso en el siguiente nivel de la jerarquía.

4.3.8.1 Recursos

- KB [0] W. Este recurso permite enviar, en un formato compacto binario, la información de la base de conocimiento que usa el motor de inferencia del Agente de Control de Aplicación. El formato se genera convirtiendo cada cadena de caracteres que define algún elemento de la base de conocimiento (variables, conjuntos borrosos, etc.) a su correspondiente número entero. Esta codificación será entendida mejor cuando se muestre el formato XML para la definición de bases de conocimiento propuesto en esta tesis (véase la sección 5.2.1).
  - Ejemplo del formato para un conjunto borroso definido con cuatro puntos y cuyo identificador es 2:

    `[identificador=2][número de puntos=4][`x_1`][`y_1`] [`x_2`][`y_2`] [`x_3`][`y_3`] [`x_4`][`y_4`]`

    Los pares x_i e y_i sería valores en coma flotante de doble precisión y los demás, enteros de 32 bits[26].

[26] Se puede optar por una menor precisión para un mayor ahorro de recursos.

4.4 PROTOCOLO DE COMUNICACIÓN

El protocolo de comunicación desarrollado tiene como principal objetivo, según propone la hipótesis segunda del presente trabajo, el dar soporte a los sistemas basados en conocimiento en las redes de sensores. Es por ese motivo por el que se ha diseñado una estructura de recursos como la presentada en la sección 4.2. Con ella se pretende proporcionar el marco de información necesario para la definición completa de todos los elementos dentro de la red de sensores, y así poder cumplir con los requisitos de la segunda hipótesis.

La descripción del protocolo que se presenta a continuación se inicia con la especificación de los servicios que proporciona, seguido por la máquina de estados que gobierna el protocolo y las comunicaciones en el sensor. Por último, se presentará con detalle cada uno de los mensajes que forman parte del protocolo, así como sus comandos.

Antes de pasar a la descripción, cabe destacar que el protocolo que aquí se presenta se ha denominado WISMAP, acrónimo de *WIreless Sensor Management Application Protocol*. Así pues, dado que esta denominación ha sido usada en todas las publicaciones que avalan la presente tesis, se usará WISMAP siempre que se referencie el protocolo aquí presentado.

4.4.1 SERVICIOS OFRECIDOS POR EL PROTOCOLO

En concreto, tanto el protocolo WISMAP, como la estructura de recursos, se ha diseñado para ofrecer dos tipos de servicios: los ofrecidos por el protocolo a nivel de aplicación en una red de sensores y los ofrecidos como red en conjunto (*test-bed*) hacia el exterior (Internet) proporcionados por la pasarela WISMAPi que se describirá en la sección 4.5. Esta doble vertiente es necesaria para cumplir con los objetivos marcados para la presente tesis (véase el apartado 1.4).

Cada servicio se instrumenta a través de uno o más mensajes, los cuales serán definidos más tarde, y vinculados a su correspondiente servicio.

Los servicios ofrecidos por el protocolo WISMAP para el usuario del nivel de aplicación en un sensor son los siguientes:

- **Notificación de actividad:** cada sensor avisa a la estación base cuando está listo para recibir información. Este servicio está justificado por el modelo de funcionamiento propuesto, que es el de que cada sensor esté el mayor tiempo posible en reposo y tan sólo se despierte cuando tenga que tomar medidas del entorno. Como ya se ha explicado, este funcionamiento está controlado por el Agente de Gestión, el cual, a su vez, puede inhibir el envío de notificaciones desactivando el Agente de Comunicaciones. Así pues, un sensor podrá realizar su trabajo de medición, y solo notificar si está listo para recibir otros comandos cuando se estime conveniente. De esta manera, se evita el tener que mantener las comunicaciones activas durante periodos en los que no se necesite enviar información.
- **Servicio de vinculación con estación base:** este servicio permite la instalación y configuración automática de sensores dentro de una red con una estación base. Cuando un sensor se inicia por primera vez dentro de una red, difunde un mensaje de solicitud de vinculación que es atendido tan sólo por las estaciones base que estén activas. La estación base que esté configurada para atender a ese

sensor será la única que responda, enviando un nuevo mensaje de confirmación que hará que el sensor predefina a esa estación base como sumidero (*sink*). Este servicio permite también la actualización de la configuración de los sensores adscritos a un sumidero a través de un comando de rechazo por parte de la estación base, así como la reconfiguración automática por parte del sensor ante un número determinado de intentos[27] de vinculación fallidos.

- **Servicio de solicitud de información:** una estación base puede solicitar información a un sensor que le haya notificado que está despierto a través de una **sesión de control**[28]. Estas solicitudes pueden ser de diferente tipo. Una vez se han recibido todas las respuestas, o la estación base lo estima oportuno, esta última envía el mensaje que informa al sensor de que debe volver a entrar en el estado de reposo.
- **Servicio de configuración:** a través de este servicio, durante una sesión de control, un sensor puede recibir mensajes que modifiquen el valor de parámetros de recursos de su jerarquía interna que tengan permiso de escritura (W).

Todos los servicios antes presentados están vinculados a un estado concreto del Agente de Comunicaciones y a una secuencia concreta de mensajes del protocolo WISMAP. La máquina de estados se presentará en la siguiente sección, y a continuación, cada uno de los diferentes tipos de primitivas, mensajes y su correspondiente formato.

4.4.2 MÁQUINA DE ESTADOS DEL PROTOCOLO

El comportamiento del protocolo de comunicaciones WISMAP, y por extensión, el Agente de Comunicación, está regido por una máquina de estados. Por mantener la coherencia con la denominación de los mensajes del protocolo, estos se mantienen en la máquina de estados en inglés, que como ya se ha comentado, ha sido el idioma usado para todas las implementaciones.

Se debe destacar que la máquina de estados también regula el funcionamiento de la estación base, o de cualquier otro sensor que en un momento se comportara como sumidero. Así pues, los estados propios de un sensor aparecerán diferenciados de los de una estación base o sumidero usando un sencillo código de color: azul para un sensor, rojo para la estación base o sumidero. Los estados comunes a ambos aparecerán en color morado.

Así mismo, las transiciones entre estados dentro de un sensor serán representadas por líneas de trazos, y las de la estación base o sumidero por líneas continuas. Para identificar las transiciones se ha optado por una notación típica en las máquinas de estados de protocolos (como por ejemplo la del protocolo TCP). Esta notación consiste en dos términos separados por una línea horizontal, en la que el término superior identifica el evento o acción que provoca la transición, y en la parte inferior se resume la acción a llevar a cabo, si la hubiera.

La revisión de estados se hará usando el orden en el que se procesarían, en primer lugar en un sensor, y a continuación en una estación base.

[27] La constante que establece el número máximo de fallos en un sensor antes de considerar un este aislado se denomina MFThreshold, del inglés *Maximum Failed Threshold*.

[28] Una sesión de control se define como el tiempo que está un sensor conectado con la estación base. Este concepto será descrito en el estado `OPERANDO`.

- **Aislado (sensor):** este es el estado inicial de las comunicaciones en un sensor justo después de su despliegue. En este estado el sensor **aún** no tiene una estación base asignada, por lo que su acción es solicitar la vinculación con una estación base a través de la primitiva *Link*[29]. Como consecuencia, el sensor envía un mensaje *SNR_ISOLATED*. El envío de este mensaje provoca que el sensor cambie al estado `ESPERANDO ESTACIÓN BASE`.
- **Esperando estación base (sensor):** en este estado el sensor espera recibir un mensaje de vinculación por parte de la estación base que lo tenga asignado (mensaje BST_LINK). En este caso el sensor responderá a la estación base con la confirmación SNR_OK y cambiará al estado OPERANDO que le permitirá recibir los parámetros de configuración necesarios. Si el mensaje BST_LINK no llega antes de que expire un temporizador (Timeout B), el sensor volverá al estado AISLADO. Como consecuencia de esta última transición no se volverá a realizar un nuevo intento hasta que el Agente de Comunicaciones vuelva a iniciar la máquina de estados llamando a la primitiva Link.
- **Operando (sensor y estación base):** en este estado, común a un sensor y a una estación base, se permite el intercambio de cualquier número de solicitudes de información o configuración por parte de las estación base hacia un sensor. Una vez terminado el intercambio, la estación base envía el mensaje *BST_SLEEP* y pasa al estado `ESPERAR SNR_OK`. Por otro lado, cuando un sensor en el estado `OPERANDO` recibe el mensaje *BST_SLEEP* responde automáticamente con un mensaje *SNR_OK* y pasa al estado `REPOSO`. El tiempo que el sensor y la estación base están en el estado `OPERANDO` se denomina **sesión de control.** Durante una sesión de control la estación base podrá requerir toda la información que se necesite recabar de un sensor mientras éste ha estado en reposo. Además de peticiones de medidas del entorno, el sensor puede recibir mensajes de configuración que permitan modificar su comportamiento.
- **Reposo (sensor y estación base):** en este estado, un sensor pasará a una desconexión de comunicaciones que tan sólo será reiniciada cuando el sensor vuelva de su estado de sueño (hecho controlado por el Agente de Gestión). Por parte de una estación base, en este estado tan sólo estará mientras no reciba una petición de algún otro sensor.
- **Esperar SNR_OK (estación base):** este estado es un paso intermedio entre el estado OPERANDO y el estado REPOSO en una estación base. Su función es tan sólo la de esperar la confirmación *SNR_OK* enviada por un sensor tras la recepción del mensaje *BST_SLEEP*. En cualquier caso, si no se recibiera dicho mensaje, este estado finalizaría tras un periodo de espera determinado. Este tiempo de espera está vinculado al tiempo máximo de espera para una conexión, establecido a nivel de transporte, el cual se ha dispuesto, de manera empírica, en 8 segundos.
- **Sensor perdido (estación base):** la estación base entra en este estado tras recibir un mensaje *SNR_ISOLATED* por parte de algún sensor. Estos mensajes son enviados por los sensores cuando no tienen ninguna estación base asignada, ya sea por ser la primera vez que se ha ejecutado el Agente de Comunicaciones, o porque la variable `Fallo` ha superado el valor de reintentos permitidos (*MFThreshold*). La acción que toma la estación base al entrar en este estado es enviar un mensaje *BST_LINK* (o *BST_REJECT* si no le corresponde atender a ese sensor). Del estado `SENSOR PERDIDO` se evoluciona al estado `OPERANDO`

[29] Las primitivas serán comentadas en la siguiente sección junto con los mensajes del protocolo.

al recibir un mensaje de aceptación por parte del sensor (*SNR_OK*). Por no complicar en exceso el diagrama de la máquina de estados, se ha omitido la transición desde este estado al estado REPOSO en el caso de que no se reciba contestación por parte del sensor en 8 segundos.

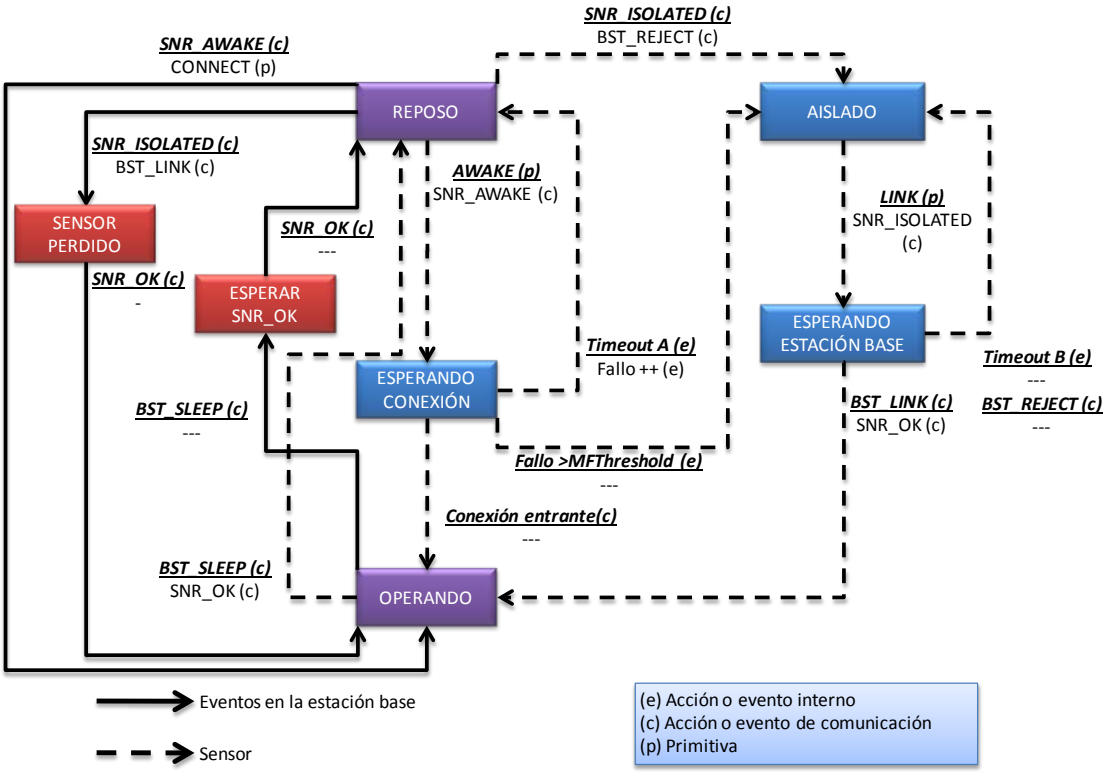


Figura 11. Máquina de estados del protocolo WISMAP.

4.4.3 FORMATO DE LOS MENSAJES DEL PROTOCOLO

El protocolo WISMAP ofrece cuatro primitivas diferentes al usuario de la capa de aplicación, cada una coincidente con uno de los servicios antes presentados. Dichas primitivas son: Awake, *Link*, *Set* y *Get*; al igual que con los demás elementos del protocolo, se han mantenido su denominación en inglés al ser parte de la implementación. Para poder llevar a cabo los servicios invocados por las primitivas, el protocolo WISMAP proporciona doce mensajes diferentes.

Cada mensaje está definido por una serie de características concretas. En primer lugar, un mensaje puede estar vinculado a un tipo concreto de dispositivo[30] o dispositivos, siendo posible su envío tan sólo desde ese tipo de dispositivo. A estos dispositivos se les denomina dispositivos activos. Así pues, los dispositivos que se prevén en el protocolo para recibir un mensaje concreto se denominan dispositivos pasivos.

Además de los tipos de mensaje, el número de destinatarios del mismo también dependerá del mensaje, condicionando en principio las capas de transporte subyacentes. Es por eso que se recomienda disponer de capas de transporte o red que permitan envíos de difusión, así como punto a punto. En la descripción de cada mensaje se han identificado a los dispositivos que pueden emitir dicho mensaje como dispositivos activos, y los que lo reciben como pasivos. Se debe remarcar que no es obligatorio que

[30] Se entiende por dispositivo a un sensor, estación base o sumidero.

las capas inferiores permitan conexiones punto-a-punto confiables, pero sí debería estar disponible la función de envíos de difusión, ya que muchos de los comportamientos del protocolo dependen de esta capacidad.

A continuación se detallará el formato de la unidad de datos del protocolo, o PDU, antes de describir uno a uno cada uno de los mensajes del mismo.

4.4.3.1 Formato de la PDU

La PDU del protocolo WISMAP es común para todos los tipos de mensaje, con una cabecera de nueve bytes y un campo de datos que es opcional. En la Figura 12 se presenta el formato de la cabecera.

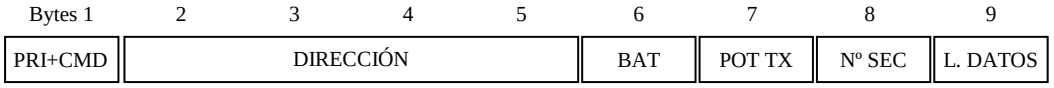


Figura 12. Formato de la cabecera del protocolo WISMAP.

Como se desprende del diseño de la PDU mostrado en la figura anterior, un mensaje del protocolo WISMAP puede contener un máximo de 264 bytes (9 bytes de cabecera + 255 bytes de datos).

A continuación se detallan los campos que forman la cabecera de los mensajes:

- Byte 1:
  - Bits 0-1: prioridad de una petición, su valor tan solo es efectivo en mensajes cuyo destinatario es un sensor. El funcionamiento del sistema de prioridad del protocolo WISMAP se comentará más adelante.
  - Bits 2-7: comando. Permite diferenciar entre los tipos de mensajes. Dado sus especiales características se explicará más adelante.
- Byte 2-5: dirección del dispositivo. Se han reservado 32 bits para disponer de una amplia capacidad de direccionamiento. En particular se han reservado tantos bits para que se pueda desarrollar un formato de direccionamiento jerárquico dentro de una red de sensores que atienda a criterios de aplicación.
- Byte 6: nivel de batería del dispositivo en el momento de enviar el mensaje.
- Byte 7: potencia de transmisión relativa. El valor 255 equivale a la potencia máxima con la que es capaz de transmitir el dispositivo, mientras que el valor 0 corresponde al valor mínimo de transmisión. Se ha usado un valor de potencia relativo para que puedan ser usados dispositivos con capacidades de transmisión diferentes dentro de la misma red y ésta pueda ser ajustada con un margen de libertad suficiente.
- Byte 8: número de secuencia del mensaje para identificación de respuestas. El orden de secuencia de los mensajes no es crítico ya que la mayoría de estos serán encapsulados en una única trama MAC y no suelen requerir de una confirmación explícita. A pesar de todo se dota de este identificador para dar soporte a envíos puntuales de información que requieran de más de un mensaje WISMAP.
- Byte 9: longitud de los datos. No todas las PDUs llevan datos, una PDU sin datos tendrá este campo a 0 y no se interpretará para otro propósito.

4.4.3.2 Prioridad

El protocolo WISMAP establece un sistema de prioridad para los mensajes de petición basado en criterios de ahorro energético. Así pues, se definen cuatro categorías de prioridad que establecen un comportamiento concreto en función de la batería disponible del dispositivo destinatario. Los bits que indican la prioridad son los bits menos significativos del primer byte de la cabecera. Estos dos bits permiten definir, como ya se ha comentado, cuatro clases de prioridad, correspondiendo 00b para la menor y 11b para la mayor. De esta forma, un mensaje con prioridad baja no será atendido si el nivel de carga de la batería no está por encima de un valor concreto, mientras que el nivel máximo establece que esa petición debe ser siempre atendida. A continuación se detalla cada uno de los niveles de prioridad.

- Prioridad baja (00b): las peticiones no serán atendidas cuando la batería esté por debajo del 75% del total de su carga.
- Prioridad media (01b): las peticiones no serán atendidas cuando la batería esté por debajo del 50% del total de su carga.
- Prioridad alta (10b): las peticiones no serán atendidas si la batería está por debajo del 25% de su carga máxima.
- Prioridad máxima (11b): este tipo de peticiones siempre serán atendidas.

4.4.3.3 Comando

Cada mensaje está definido por un comando que indica la acción principal a realizar por el dispositivo destinatario. Los códigos de los comandos se codifican con los 6 bits más significativos dentro del primer byte de la PDU (los dos bits menos significativos, como ya se ha comentado, corresponden con la prioridad del mensaje).

La estructura del campo comando es a su vez jerárquica, dividiéndose en dos partes:

- Los dos bits más significativos informan del origen del comando.
- El resto, del tipo de comando en sí.

Se vio conveniente realizar esta distinción para poder discriminar de manera más rápida los mensajes que pueden interesar a un nodo.

A continuación aparecen relacionados los cuatro posibles casos en cuanto al origen de los mensajes (aparecen tan sólo los dos bits más significativos con valor, el resto aparece con el carácter ‘x’ que indica un valor indiferente):

- 00xxxxb: Mensajes que provienen de una estación base.
- 01xxxxb: Mensajes que provienen de un sensor.
- 10xxxxb: Mensajes que provienen de un sumidero (*sink*).
- 11xxxxb: Mensajes comunes a varios dispositivos o que provienen de una petición WISMAPi[31].

En la siguiente sección se describen cada uno de los diferentes comandos y los mensajes a los que dan lugar.

[31] La denominación WISMAPi es el acrónimo de WISMAP Internet, denominación que recibe el protocolo WISMAP en su vertiente de red pública TCP/IP. Esta especificación será descrita en la sección 4.5.

4.4.4 COMANDOS DEL PROTOCOLO WISMAP

Como ya se ha comentado, cada mensaje del protocolo WISMAP está definido por un comando y por una prioridad. Mientras que la prioridad establece el mismo comportamiento para todos los mensajes, cada comando define un formato y una serie de diferentes acciones a llevar a cabo.

La descripción de cada comando vendrá presentada por el nemónico usado dentro del protocolo WISMAP, la máquina de estados y la implementación software. A continuación se hace referencia a la versión del protocolo en la que apareció. Se debe hacer constar que la versión que se ha establecido durante la redacción de la presente tesis es la 0.7, ya que el trabajo está aún abierto a posibles ampliaciones, las cuales serán convenientemente descritas en las líneas de futuro del presente trabajo de investigación.

La parte que sigue a la versión es la descripción del comando, precediendo ésta a la consignación de los dispositivos que toman parte en la comunicación. Estos dispositivos se han definido de entre dos categorías, activos, aquellos que enviarán el comando, y pasivos, que serán los destinatarios típicos del mensaje.

A continuación se detalla el tipo de envío, pudiendo ser difusión o punto a punto. La distinción se ha implementado a nivel de transporte, usando un protocolo no orientado a conexión para los envíos de difusión, y un protocolo orientado a conexión para los envíos punto a punto. En concreto, los protocolos usados pertenecen a la familia Sun SPOT, si bien no se deriva de esto más que las dependencias típicas de uso de unas clases de comunicaciones concretas a la hora de la implementación.

Los dos últimos apartados que aparecen en la descripción de cada comando son la codificación binaria del mismo y sus parámetros, si los hubiera. En el caso de tener parámetros, aparecerá también su formato concreto.

La codificación del comando se muestra en binario, hexadecimal y en octal.

A continuación se muestra la descripción detallada de cada comando, ordenados en orden ascendente según el valor de su campo “Código”.

4.4.4.1 BST_LINK, mensaje para enlazar un sensor a una estación base.

Nemónico: BST_LINK

Versión de aparición: 0.2

Descripción: este mensaje es enviado por una estación base cuando recibe un mensaje de SNR_ISOLATED de un sensor que está configurado para pertenecer a su red. El mensaje BST_LINK es enviado por difusión debido a que el sensor no puede establecer una comunicación punto a punto sin conocer la MAC de la estación base. Este es el mensaje generado por la ejecución de la primitiva *Link*.

Dispositivos:

- Activos: Estaciones base.
- Pasivos: Sensores, sumideros.

Envío: difusión.

Código (6 bits msb[32]): 0000 01XX - 04h - Octal 01

Parámetros:
- 64 bits: dirección MAC IEEE 802.15.4 de la estación base.

4.4.4.2 BST_SLEEP, mensaje de poner sensor en modo de ahorro de energía.

Nemónico: BST_SLEEP

Versión de aparición: 0.1

Descripción: este mensaje lo envía la estación base al sensor destinatario cuando ya no necesita que éste haga ninguna operación más, haciendo que el sensor entre en modo de ahorro de energía. El envío de este mensaje determina el fin de una sesión de control.

Dispositivos:
- Activos: Estaciones base.
- Pasivos: Sensores, sumideros.

Envío: punto a punto.

Código (6 bits msb): 0000 10XX – 08h - Octal 02

Parámetros: no tiene.

4.4.4.3 BST_REJECT, mensaje de rechazo de vinculación para un sensor.

Nemónico: BST_REJECT

Versión de aparición: 0.2

Descripción: este mensaje es enviado por una estación base cuando recibe un mensaje de SNR_ISOLATED de un sensor que no pertenece a su red. El mensaje BST_ REJECT es enviado por difusión debido a que el sensor no puede establecer una comunicación punto a punto sin conocer la MAC de la estación base.

Dispositivos:
- Activos: Estaciones base.
- Pasivos: Sensores, sumideros.

Envío: difusión.

Código (6 bits msb): 0000 11XX – 0Ch - Octal 03

Parámetros: no tiene.

4.4.4.4 BST_KB, mensaje de envío de una base de conocimiento

Nemónico: BST_KB

Versión de aparición: 0.3

[32] Del inglés *most significant bit.*

Descripción: este mensaje es enviado por una estación base para actualizar la base de conocimiento del sistema borroso basado en reglas de un sensor. Este envío puede responder a una acción del usuario, a un comando del protocolo WISMAPi o a un mensaje de un sensor.

Dispositivos:
- Activos: Estaciones base.
- Pasivos: Sensores, sumideros.

Envío: difusión.

Código (6 bits msb): 0001 00XX – 10h - Octal 04

Parámetros:
- Base de conocimiento en formato binario compacto[33].

4.4.4.5 SNR_AWAKE, mensaje sensor despierto.

Nemónico: SNR_AWAKE

Versión de aparición: 0.1

Descripción: mensaje enviado por el Agente de Comunicación cuando despierta del modo de funcionamiento de ahorro de energía. Este mensaje informa de la intención de comunicar del sensor y de su disponibilidad para la recepción de peticiones de información y configuración. El tiempo que permanece en este estado está limitado por el *Timeout A*. Una vez transcurrido este tiempo el sensor vuelve a ponerse en reposo (ver máquina de estados). Este es el mensaje enviado por la ejecución de la primitiva *Awake*.

Dispositivos:
- Activos: Sensores y nodos sumidero.
- Pasivos: Estaciones base.

Envío: difusión o punto a punto.

Código (6 bits msb): 0100 00XX - 40h - Octal 20

Parámetros: no tiene.

4.4.4.6 SNR_ISOLATED, mensaje de sensor aislado.

Nemónico: SNR_ISOLATED

Versión de aparición: 0.2

Descripción: este mensaje lo envía un sensor cuando no tiene una estación base activa. En concreto el sensor se encuentra en el estado asilado (*isolated*). Los motivos que existen para que un sensor no tenga una estación base activa son dos:

[33] Este formato binario compacto se genera a partir de la definición de una base de conocimiento en formato XML. Se usa este formato, y no el XML directamente, debido a que la codificación de caracteres del XML sería demasiado extensa para ser enviada por una red de sensores. Estos formatos, ambos contribuciones de la presente tesis, serán descritos en la sección 5.2.

- Ser la primera vez que se inicia el sensor, por lo que no tiene una estación base asignada (no existe ninguna estación base prefijada en el código ya que esto impediría la autoconfiguración del Agente de Comunicación).
- El sensor ha perdido contacto con su base durante varios intentos notificando su disponibilidad con el mensaje SNR_AWAKE.

De esta manera los sensores no necesitan tener una pre-configuración de base a la que conectarse, o ser reprogramados tras perder contacto con una. Por lo tanto, sólo las bases necesitan saber qué sensores se pueden unir a ellas, lo cual, al estar normalmente conectadas a equipos de mayor capacidad de proceso, no supone ningún problema.

Un sensor, tras el envío del mensaje que informa de su estado de aislamiento, queda a la espera de una respuesta, la cual puede ser de los siguientes tipos:

- Enlazado con nueva base (BST_LINK): este mensaje lo envía la estación base al recibir SNR_ISOLATED de un sensor que está dentro de su red. La información de este mensaje podrá verse más abajo, pero básicamente informa de la MAC de la estación base.
- Rechazado por la estación base (BST_REJECT): La estación base, o estaciones base, que reciben la petición de vinculación no tienen a ese sensor como miembro de su red. Si ninguna de las estaciones base dentro del rango del sensor (o de la red) tiene al sensor como miembro, el sensor permanece aislado, por lo que la próxima vez que termine su estado de reposo volverá a intentar vincularse con una estación base.

Dispositivos:
- Activos: Sensores y nodos sumidero.
- Pasivos: Estaciones base.

Envío: difusión.

Código (6 bits msb): 0100 01XX - 44h - Octal 21

Parámetros: no tiene.

4.4.4.7 SNR_OK, mensaje de orden recibida y procesada correctamente

Nemónico: SNR_OK

Versión de aparición: 0.1

Descripción: este mensaje lo envía un sensor ante una petición que no implica el envío de más datos, como ante la recepción del comando BST_SLEEP. Los destinatarios de este mensaje no esperarán indefinidamente la recepción de este mensaje por criterios de ahorro de recursos.

Dispositivos:
- Activos: Sensores.
- Pasivos: Estaciones base, sumideros.

Envío: punto a punto.

Código (6 bits msb): 0100 10XX- 48h - Octal 22

Parámetros: no tiene.

4.4.4.8 SNR_VALUE, mensaje de respuesta de valor desde un sensor

Nemónico: SNR_VALUE

Versión de aparición: 0.1

Descripción: este mensaje es la respuesta por parte de un sensor ante la petición GET_VALUE recibida desde la estación base tras la ejecución de la primitiva *Get*.

Dispositivos:
- Activos: Sensores.
- Pasivos: Estaciones base, sumideros.

Envío: punto a punto.

Código (6 bits msb): 0100 11XX- 4Ch - Octal 23

Parámetros:
- Secuencia (1 byte): número de secuencia de la petición GET_VALUE.
- Tipo de sonda (1 byte): tipo de sonda. Las sondas disponibles en un sensor están listadas en la Tabla 7.

Tabla 7. Tipos de sondas.

Código	Sonda	Descripción
00h	Valor no válido	
01h	Temperatura	Temperatura en grados Celsius. Se representa como un valor en coma flotante de doble precisión (64 bits) encapsulado en un entero de 64 bits (*long*) usando el formato de bit *IEEE 754 floating-point "double format"* (IEEE2008)
02h	Iluminación	Luminancia medida en Lux. Se representa por un valor entero de 32 bits
03h	Acelerómetro de 3 ejes	Aceleración en cada eje medida en múltiplos de la aceleración de la gravedad de la Tierra (g). Se compone de tres valores contiguos en coma flotante de doble precisión (64 bits) encapsulado en un entero de 64 bits (long) usando el formato de bit *IEEE 754 floating-point "double format"*. Primero irá la aceleración en el eje X, luego el eje Y, y por último, el eje Z.
04h	Presión sonora 8K Desde la versión 0.2	Valor de la presión sonora, en dBA, captada por el sensor muestreando a 8KHz. El valor se codifica en coma flotante de doble precisión (64 bits) encapsulado en un entero de 64 bits (long) usando el formato de bit *IEEE 754 floating-point "double format"*
05h	Presión sonora 16K Desde la versión 0.2	Valor de la presión sonora, en dBA, captada por el sensor muestreando a 16KHz. El valor se codifica en coma flotante de doble precisión (64 bits) encapsulado en un entero de 64 bits (long) usando el formato de bit *IEEE 754 floating-point "double format"*

- Se ha optado por la predefinición de valores para los tipos de sonda para ayudar a la interpretación de la información entre redes de sensores de

diferentes familias. De esta manera, una petición para la lectura de la temperatura, por ejemplo, para un sensor Sun SPOT o para un MicaZ será idéntica, al igual que la codificación de dicha temperatura. Dado que se ha consignado un byte para este propósito y que se ha establecido el valor 0 como un valor de control que no debe ser usado, se pueden definir un máximo de 255 tipos diferentes de sondas. Las que se presentan en la Tabla 7 son aquellas que se han usado durante los trabajos de investigación de la presente tesis.

- Fecha y hora (64 bits): Fecha y hora de la medida.
- Valor (tamaño variable): Valor medido por la sonda.

4.4.4.9 SNR_ERROR, mensaje de orden recibida y no procesada o incorrecta

Nemónico: SNR_ERROR

Versión de aparición: 0.2

Descripción: informa de un error ante un mensaje enviado con anterioridad al sensor. El mensaje al que se hace referencia se identifica por su número de secuencia dentro de los parámetros del comando SNR_ERROR.

Dispositivos:
- Activos: Sensores.
- Pasivos: Estaciones base, sumideros.

Envío: punto a punto.

Código (6 bits msb): 0101 00XX- 50h - Octal 24

Parámetros:
- Código de error (32 bits): Identificador del error producido al realizar un comando o por un problema interno. Los códigos de error están formados de una manera jerárquica, así el byte más significativo indica el dispositivo o agente que lo ha generado, el siguiente byte informa de la severidad del error y los restantes 16 bits para el código del error. En la versión actual no se ha establecido un tratamiento diferenciado para los códigos de error dado que no ha sido necesario para la consecución de los objetivos de la presente tesis.

4.4.4.10 SNR_TASK, mensaje de datos de tarea realizada

Nemónico: SNR_TASK

Versión de aparición: 0.5

Descripción: informa del valor medido por una tarea. Incorpora la información del identificador de la tarea, fecha y hora en la que tomó la medida y el valor medido. Como nota aclaratoria, la forma de enviar una tarea es con el comando SET_VALUE que se verá más adelante.

Dispositivos:
- Activos: Sensores.
- Pasivos: Estaciones base, sumideros.

Envío: punto a punto.

Código (6 bits msb): 0101 01XX- 54h - Octal 25

Parámetros:
- Identificador (1 byte): Identificador de la tarea que generó la medida.
- Fecha y hora (8 bytes): Fecha y hora de la medida. Se usa el formato de entero largo[34] (*long* de la clase de java Date).
- Valores (variable): Valor de la medida correspondiente con la sonda vinculada a la tarea.

4.4.4.11 GET_VALUE, mensaje de petición de medida de un sensor

Nemónico: GET_VALUE

Versión de aparición: 0.1

Descripción: este mensaje es consecuencia de la ejecución de la primitiva *Get* la cual implica que se ha solicitado información a un sensor. Una estación base envía este mensaje a un sensor que haya comunicado que está disponible para recibir información a través del mensaje SNR_AWAKE.

Dispositivos:
- Activos: Estaciones base.
- Pasivos: Sensores.

Envío: punto a punto.

Código (6 bits msb): 1100 01XX - C4h - Octal 61

Parámetros:
- Identificador de la sonda (1 byte): número de la sonda a leer. Las sondas disponibles en un sensor dependen del soporte físico de este mismo, por lo que podría variar su número y tipo. De esta forma no se consigna un tipo de medida al número de sonda, si no que se accede a sus medidas tan sólo por el identificador. Si se quiere obtener el tipo de medida que aporta se puede solicitar la información de instalación del sensor[35].
- Campo de opciones (1 byte, desde la versión 0.5):
  - Bit 0 - Almacenar la medida en el sensor o no (desde la versión 0.5):
    - Valor 0: No almacenar el dato requerido en el sensor, tan sólo devolver por la red.
    - Valor 1: Almacenar el valor requerido en el sensor además de enviarlo.
  - Bit 1 al 7: reservado para futuro uso.

4.4.4.12 SET_VALUE, mensaje de configuración para un sensor

Nemónico: SET_VALUE

Versión de aparición: 0.2

[34] Este valor contiene el número de milisegundos transcurridos desde las 00:00 horas del 1 de enero de 1970 en el huso horario del meridiano de Greenwich (GMT, del inglés *Greenwich Mean Time*) el cual no cambia con las estaciones.

[35] Esta información se ha dejado pendiente de definir. Una posibilidad es que sea en XML y se cargue al sensor desde la estación base, siendo almacenada de manera persistente por este.

Descripción: este mensaje lo envía la estación base al sensor destinatario para establecer el valor de alguno de los recursos o parámetros internos del sensor. Este mensaje es consecuencia de la ejecución de la primitiva *Set*.

Si el valor ha sido actualizado satisfactoriamente el sensor enviará un mensaje SNR_OK, si por el contrario se ha producido algún error, el sensor emitirá un mensaje SNR_ERROR.

Los recursos en el sensor están organizados según la estructura jerárquica que aparece en el apartado 4.2. Los elementos disponibles en un sensor aparecen en el nivel `/wsn/sensor/` con el correspondiente identificador de sensor. De esta manera, lo recursos accesibles serían aquellos del nivel de componente con parámetros modificables ('W'), como el recurso `com`, `man`, etc. Los componentes se identificarán por su identificador dentro de la petición según aparece también en el esquema siguiente:

Dispositivos:
- Activos: Estaciones base.
- Pasivos: Sensores.

Envío: punto a punto.

Código (6 bits msb): 1100 10XX – C8h - Octal 62

Parámetros: el formato de los mensajes de modificación de valores es algo más complejo, y estos pueden llevar encapsulados varias peticiones en un mismo mensaje:

> `[1 byte – identificador del agente]`[36] `[1 byte – identificador del componente] [1 byte – identificador del subcomponente]`[37] `[`**`lista de parámetros`**[38]`]`
>
> **`Lista de parámetros:`** `[1 byte nº parámetros][`**`parámetro`** `1]…[parámetro n]`
>
> **`Parámetro`**[39]**`:`** `[1 byte - id parámetro][`**`valor`**`]`
>
> **`Valor:`** `[1 byte – tipo valor][1 byte - longitud valor] [n bytes – valor]`
>
> Los tipos de parámetros dependen del recurso al que se esté accediendo, de ahí el identificador de componente que es obligatorio. Los componentes y sus identificadores que pueden ser accedidos dentro de un sensor aparecen en el punto 4.3.5.

[36] Introducido en la versión 0.5
[37] Introducido en la versión 0.5
[38] Cuando el valor es a nivel de subcomponente, la lista de parámetros se reduce normalmente a uno (byte número de parámetros = 1) y el valor es un vector de bytes producto de la serialización de subcomponente.
[39] En la versión 0.5, se ha eliminado el campo número de valores por considerarlo redundante, si es necesario varios valores enteros mejor ponerlos como parámetros diferentes.

Los tipos de valores disponibles van desde los tipos simples como enteros, a complejos, como cadenas de caracteres o reglas borrosas de bases de conocimiento.

Tabla 8. Resumen de los comandos del protocolo WISMAP.

Nemónico	Versión aparición	Código	Parámetros	Descripción
BST_LINK	0.2	0000 01XX	64 bits: dirección MAC de la estación base	Enlazar un sensor a una estación base
BST_SLEEP	0.1	0000 10XX	-	Fin de la sesión de control
BST_REJECT	0.2	0000 11XX	-	Rechazo de vinculación para un sensor
BST_KB	0.3	0001 00XX	Base de conocimiento en formato binario compacto	Envío de una base de conocimiento
SNR_AWAKE	0.1	0100 00XX	-	Sensor despierto
SNR_ISOLATED	0.2	0100 01XX	-	Sensor aislado
SNR_OK	0.1	0100 10XX		Orden recibida y procesada correctamente
SNR_VALUE	0.1	0100 11XX	Secuencia (1 byte): número de secuencia de la petición GET_VALUE. Tipo de sonda (1 byte): tipo de sonda. Las sondas disponibles en un sensor están listadas en la Tabla 1. Tipos de sondas. Valor (tamaño variable): Valor medido por la sonda.	Este mensaje lo envía un sensor ante la petición GET_VALUE recibida desde la estación base
SNR_ERROR	0.2	0101 00XX	32 bits: Código de error	Orden recibida pero no procesada o incorrecta
SNR_TASK	0.5	0101 01XX	Formato de datos de tareas: id, fecha, hora y valor	Valor de una tarea
GET_VALUE	0.1	1100 01XX	8 bits: identificador de la sonda	Mensaje de petición de medida de un sensor
SET_VALUE	0.2	1100 10XX	[1 byte – id del agente] [1 byte – id del componente] [1 byte – id del subcomponente] [lista de parámetros] Lista de parámetros: [1 byte nº parámetros][parámetro 1]…[parámetro n] Parámetro: [1 byte - id parámetro][valores] Valores: [1 byte – tipo valor][1 byte - longitud valor] [n bytes – valor]	Establecer valor en el sensor

4.5 PASARELA IP DEL PROTOCOLO WISMAP

Dadas las necesidades actuales de acceso a la información, las cuales requieren soluciones integrales que permitan la movilidad total de terminales con acceso a Internet, los diseños y desarrollos que se presenten sobre de redes de sensores deben tener dichas necesidades muy presentes. Ya se ha comentado la existencia de protocolos como 6LoWPAN (Kushalnagar et al. September 2007), el cual permitiría extender sobre una red de sensores el protocolo IPv6, y su familia de capas superiores como UDP y TCP, o en aplicación HTTP. Sin embargo, además de tener en cuenta que 6LoWPAN es un protocolo de nivel de red y que, podría ser compatible con el uso de WISMAP, la mayoría de protocolos de transporte y aplicación convencionalmente usados en Internet no son apropiados para su uso en redes de sensores.

Así pues, en el presente trabajo se ha buscado el aprovechamiento de recursos que proporciona el protocolo WISMAP y su funcionalidad (medición, configuración, soporte FRBS) desarrollando una pasarela entre Internet y la red de sensores que use el protocolo WISMAP. Esta pasarela permite hacer peticiones de información y cambios en la configuración de los sensores a través de una URI (Berners-Lee et al. January 2005) con una sintaxis equivalente al protocolo HTTP, si bien el protocolo que aparece en dicha petición, en vez de ser el tradicional HTTP/1.1, es WISMAP/0.1. Esta distinción se hace para facilitar el análisis del tráfico cuando se usa en Internet. Por otro lado, cabe destacar que si simplemente se cambia esta diferencia, las peticiones serían

idénticas (seguirían existiendo diferencias en cuanto a las cabeceras permitidas) y podría usarse en el caso de que otro tipo de tráfico no reconocido estuviera limitado o filtrado.

A esta variante del protocolo WISMAP, para el acceso desde Internet a la pasarela con la red de sensores, se ha denominado WISMAPi (de WISMAP para Internet).

Siguiendo con las similitudes con HTTP, WISMAPi implementa dos de los métodos del protocolo HTTP, *GET* y *POST*. El soporte a los demás métodos no ha sido necesario, ya que la pasarela, si bien es un servidor TCP, no tiene la funcionalidad completa de un servidor web convencional. Se debe hacer constar que el equipo en el que esté alojada la pasarela debe tener al menos un punto de conexión con la red de sensores. En el caso de la pruebas del presente trabajo de investigación se ha usado una estación base de la familia Sun SPOT conectada a través de una interfaz USB.

A continuación se detallará el formato de las peticiones y las respuestas del protocolo WISMAPi.

4.5.1 MENSAJES DEL PROTOCOLO WISMAPI

Dado que los mensajes del protocolo WISMAPi están pensados para ser enviados en redes TCP/IP, para el acceso remoto a las redes de sensores o test-bed que implementen el protocolo WISMAP, siguen la recomendación de la RFC 822 (CrockerAugust 1982), actualmente RFC 5322 (ResnickOctober 2008). Sin embargo, se ha buscado una mayor compatibilidad que HTTP con las extensiones MIME (Freed & Borenstein November 1996), al no incluir información que no esté codificada en caracteres imprimibles de la norma US-ASCII[40], tanto en las peticiones, como en las respuestas. Así pues, todas las respuestas a peticiones WISMAPi, que conlleven el envío de información, incorporarán estos datos en formato HTML.

El formato de los mensajes WISMAPi, que como ya se ha comentado, sigue en gran medida el especificado para HTTP (Fielding et al. June 1999), se ha definido igualmente con la notación ABNF[41] (Overell & Crocker Enero 2008). En la definición se parte de un elemento común, tanto para peticiones como respuestas, denominado WISMAPi-message, el cual se compone de los siguientes elementos (se ha mantenido la definición en inglés con objeto de compatibilizarla con la que se da en la RFC 2616[42]):

```
WISMAPi-message = start-line
     *(message-header CRLF)
     CRLF
     [ message-body ]
```

Siendo:

```
start-line = Request-Line / Status-Line
```

[40] Denominación recomendada por la especificación MIME del juego de caracteres usado en el alfabeto anglosajón en su versión estadounidense del estándar ASCII (American Standard Code for Information Interchange).
[41] Siglas de su denominación en inglés, *Augmented Backus-Naur Form.*
[42] Durante la finalización de la redacción de la presente tesis las RFC que cubren el protocolo HTTP/1.1 se han actualizado, ampliándose dicha especificación a una serie de RFCs, de la 7230 a la 7235. Concretamente, es en la RFC 7230 donde aparece la nueva especificación de los mensajes, la cual no presenta diferencias sustanciales con la anteriormente definida.

Lo cual corresponde con:

- `start-line`: para las peticiones se denominará línea de petición (`Request-Line`), y para las respuestas línea de estado (`Status-Line`).
- `message-header`: ninguna o más cabeceras. La cabecera que se usa en la versión 0.1 son tan solo la `Content-Length` y la `Conten-Type`, ambas con el mismo formato que su contrapartida en HTTP.
- `CRLF`: Una línea en blanco (caracteres `CR`[43]`LF`[44]) que indican el fin de las cabeceras y el comienzo del cuerpo del mensaje.
- `message-body`: el cuerpo del mensaje es opcional. En la versión 0.1 solamente es usado en los métodos POST, siendo dónde se envía la información necesaria para poder acometer la acción a la que se destina el mensaje.

Para la definición de la línea de petición se hace necesaria la definción previa de los elementos que la van a conformar: versión, direccionamiento de los recursos a través de la ruta de acceso (ver sección 4.2) y el método a emplear en el mensaje:

```
WISMAPi-Version = "WISMAP" "/" 1*DIGIT "." 1*DIGIT

WISMAPi_resource = resource_path [ "?" query ]

Method = "GET" / "POST"

resource_path = absolute-path [ "?" query ]

absolute-path = 1*( "/" segment )

query = <query>[45],
```

La versión que se ha empleado durante la realización de las pruebas de esta tesis es la 0.1, por lo que la regla WISMAPi-Version se reemplazaría con WISMAP/0.1. Así, la línea de petición se definiría como:

```
Request-Line = Method SP[46] WISMAPi_resource SP
     WISMAPi-Version CRLF
```

En el caso de las respuestas, la línea de estado se compone de la versión del protocolo, seguida de un código de estado y una frase de explicación. La línea de estado es terminada con CRLF al igual que en las peticiones.

```
Status-Line = WISMAPi-Version
     SP Status-Code
     SP Reason-Phrase CRLF
```

[43] CR, del inglés *Carriage Return*, retorno de carro, carácter ASCII número 13, en ABNF CR = %d13.
[44] LF, del inglés *Line Feed*, avance de línea, carácter ASCII 10.
[45] Básicamente es el formato que inicia la lista de peticiones con el carácter "?" y las encadena con "&", en el punto 4.5.2 se puede ver un ejemplo. Para más detalles ver la RFC 3986 (Berners-Lee et al. January 2005), sección 3.4,
[46] SP, del inglés *SPace* o *blank space*, espacio en blanco, carácter ASCII 32. La aparición de espacios en blanco en los formatos de los mensajes no implica que estos deban usarse. Tan solo serán obligados aquellos marcados con SP.

4.5.2 MÉTODO GET

El método *GET* es el comando de petición de información genérico del protocolo WISMAPi, ya sean valores leídos por las sondas, configuración del sensor o del test-bed, ficheros, etc. Con el método *GET* no se puede establecer el valor de ningún parámetro de la jerarquía de recursos que implique un cambio en el funcionamiento del recurso destino. Para eso se utilizará el método *POST*. El envío de un mensaje WISMAPi con el método *GET*, si el recurso destinatario es un sensor, o un elemento dentro del mismo, generará un mensaje WISMAP *GET_VALUE* en la red de sensores.

Formato

```
GET SP resource_path SP WISMAP/0.1 CRLF
```

Ejemplo de petición GET:

GET /wsn/sensors/6/app/probes?id=1&value&priority=0 WISMAP/0.1

Petición del valor leído por la sonda 1 del sensor 6 de la red de sensores con la mínima prioridad (previamente se ha conectado con el servidor del test-bed)

4.5.3 MÉTODO POST

El método *POST*, al igual que en HTTP, se utiliza dentro del protocolo WISMAPi para el envío de mayor cantidad de datos al test-bed o pasarela, así como para comandar un cambio en el valor de algún parámetro de funcionamiento del test-bed o de un sensor.

Formato

```
POST SP resource_path SP WISMAP/0.1 CRLF
[ parameter ":" SP value CRLF]
```

Tanto `parameter`, como `value`, son cadenas de texto US_ASCII que representarán el nombre y valor del parámetro correspondiente (véase la sección 4.3).

Ejemplo de petición POST:

POST /wsn/sensors/6/man WISMAP/0.1
sleeptime=20000

Petición para establecer el tiempo de sueño del Agente de Gestión del sensor 6 en 20 segundos.

Si el recurso destinatario de un comando POST es un sensor o un elemento dentro del mismo, conllevará el envío de un mensaje WISMAP *SET_VALUE*.

4.5.4 CÓDIGOS DE ESTADO

Los códigos de estado empleados en el protocolo WISMAPi tienen un formato análogo al de HTTP, estando formados por tres dígitos, donde el primero indica la categoría y los otros dos el tipo de estado dentro de la categoría. En la versión 0.1 tan solo se han definido tres categorías:

- Mensajes de estado que informan de que la petición se ha procesado correctamente. Estos comienzan con el dígito '2'.
- Mensajes que indican un error en la petición, atribuible al usuario. Comienzan con '4'.
- Mensajes provocados por fallos en el servidor. Estos comienzan con '5'.

Los códigos de estado que se han definido aparecen reflejados en la Tabla 9 (se acompañan de la `Reason-Phrase`):

4.5.5 USO DE LA PASARELA WISMAP.

La pasarela WISMAP está compuesta por una aplicación servidora TCP, la cual recibe peticiones a un puerto concreto. Esta aplicación es a su vez la que controla la estación base que hace de interfaz con la red de sensores. Al conjunto de la pasarela WISMAP y la aplicación interfaz con la red de sensores se le ha denominado WSN^{MAS}, del inglés *Wireless Sensor Network Management Application System*. Al sistema WSN^{MAS}, junto con una red de sensores es lo que se denominará un test-bed WISMAP.

Tabla 9. Códigos de estado del protocolo WISMAPi

Código	Reason-Phrase	Descripción
210	OK	La petición se ha ejecutado correctamente.
401	Method not supported	Método no soportado para un recurso concreto.
402	Method not recognized	Se ha enviado un método que no pertenece al protocolo.
404	No such resource available	Se ha realizado una petición sobre un recurso que no existe en el test-bed.
405	No such parameter available	El parámetro sobre el que se ha hecho una petición no existe.
406	That resource do not admint POST request	Se ha intentado modificar un recurso que no lo permite.
407	Depth in resource list exceeded	Demasiados elementos en la ruta de recursos.
409	No data for that request	Faltan datos en la petición.
420	Syntax error	Error genérico. No se reconoce la sintaxis.
421	Missing CRLF	Falta el CRLF en la petición.
422	No root element	No se ha enviado el elemento raíz en la ruta de recursos.
423	Not enough bytes length in request	La longitud marcada
424	Bad protocol	Protocolo no reconocido.
425	Bad protocol version	Versión del protocolo WISMAPi no soportada.
510	Memory exception, the server can not store commands	Se ha producido un error por falta de memoria en el servidor y el comando no será almacenado.

Como ya se ha comentado, las peticiones *GET* o *POST* que tengan como destino un recurso presente en un sensor de la red, implicará necesariamente el envío de un mensaje WISMAP, ya sea *GET_VALUE* o *SET_VALUE*. Se debe remarcar que la

obtención de datos a través de la pasarela WISMAP no es normalmente en tiempo real, dado que un sensor está habitualmente configurado para pasar la mayor parte del tiempo en estado de reposo. Así pues, el proceso normal es hacer una petición a la pasarela usando el protocolo WISMAPi, ya sea de obtención de datos o de configuración. A continuación se debe esperar un tiempo hasta que se estime que el sensor ha podido despertarse y atender a la petición enviada, para después hacer una petición WISMAPi al recurso `/wsn/log/` indicando en el siguiente nivel el sensor y demás datos para identificar el valor a leer, tales como la sonda o estado de la petición.

5 APLICACIÓN DE SISTEMAS INTELIGENTES A LOS SENSORES

En la revisión de conocimientos del Capítulo I se han presentado, tanto los sistemas basados en conocimiento usados en esta tesis, como su repercusión dentro de las redes de sensores. Como se comentó en el apartado 2.3.2, las técnicas de lógica borrosa se presumen apropiadas para su aplicación en la gestión interna de un sensor. Esto se podrá dar, siempre y cuando, se mantenga la complejidad de la base de conocimiento dentro de unos límites que permitan un procesado abordable dentro de un sensor. Por supuesto, este requisito podrá ir ajustándose al hardware disponible, ya sea por mejoras en el mismo o por usar tipos o familias de sensores diferentes.

Así pues, se pretende refrendar la tercera hipótesis del presente trabajo con el diseño de sistemas analíticos, y otros basados en conocimiento, para la gestión autónoma de un sensor.

En primer lugar, es necesario determinar qué tipo de proceso, o procesos, dentro de un sensor son los apropiados para la incorporación de sistemas de control inteligentes. Por lo tanto, tomándose como base la revisión de conocimientos inicial, se decidió que se buscarían procesos vinculados a la aplicación a la que esté dedicado un sensor, en vez de buscar procesos de ámbito más general (como formación de clústeres o enrutamiento).

Esta decisión se tomó después de analizar que la aplicación a la que se destina un sensor queda normalmente relegada a un segundo plano en pos de la consecución de objetivos generalistas, más dependientes de las comunicaciones y la distribución de la información, que del cometido del sensor. Si bien es cierto que esos son aspectos muy importantes a tener en cuenta, la tarea de un sensor, la cual es fundamentalmente tomar medidas del exterior para poder monitorizar uno o varios fenómenos, no es tenida en la misma consideración en la mayoría de la literatura. De esta manera, si un sensor no puede cumplir con su función principal, tomar medidas, o si aun tomándolas, éstas no son significativas, su trabajo es inútil; no importando en ese caso cómo podrían ser enviadas, así como otros aspectos como el encaminamiento o las técnicas de agregación de datos a emplear.

Por lo tanto, la pérdida de medidas por parte de un sensor, que no sean derivadas de una avería o del agotamiento de su batería, se debe fundamentalmente a los ciclos de reposo programados en base a criterios de ahorro de energía. De entre los criterios más comunes, en la programación de ciclos de reposo, están aquellos que se rigen por cobertura, latencia o agrupación de datos. Evidentemente, esto es así siempre que no exista la posibilidad de hacer una medición continuada, o que no existan limitaciones en cuanto al consumo de energía, pero no es el caso del que se ocupa el presente trabajo de investigación.

Así pues, con objeto de integrar la monitorización a la que se destine un sensor, con la gestión eficiente de sus recursos energéticos, se planteó añadir a éste un sistema inteligente que controlara su ciclo de sueño de manera autónoma, adaptativa y totalmente dinámica, ajustando los tiempos de reposo en función de la magnitud del entorno que debería controlar.

Se debe destacar, que a pesar de tener un sistema inteligente que controle los tiempos de sueño, el sensor no tendrá la posibilidad de medir durante los tiempos de reposo, por lo que siempre será posible perder valores importantes. Sin embargo, estos tiempos de sueño estarán regulados en función de la magnitud bajo estudio, por lo que en casos de altas fluctuaciones, valores cerca de umbrales críticos, etc., el tiempo de sueño será menor. Por el contrario, cuando el entorno esté más estable, o los valores tomados estén dentro de regímenes de bajo riesgo, los tiempos de sueño serán más largos.

Por lo tanto, en los siguientes apartados de esta sección, se presentarán dos sistemas, uno basado en cálculos analíticos, y otro usando FRBS. Ambos realizarán un control autónomo, adaptativo y dinámico de los tiempos de sueño, con objeto de dar cumplimiento a la tercera hipótesis de la tesis.

En la siguiente sección se describe un sistema analítico que basa el tiempo de sueño de un sensor en las diferencias entre las medidas históricas y unos umbrales concretos. A este sistema se le ha denominado Sistema Diferencial. A continuación, se detallará el sistema FRBS que controlará el tiempo de sueño de un sensor, así como al formato de definición de bases de conocimiento, el cual permite su diseño y envío, ya sea a través de Internet o de una red de sensores.

5.1 SISTEMA DIFERENCIAL

Con objeto de desarrollar un sistema capaz de gestionar de manera autónoma un sensor en función de la magnitud bajo estudio, se ha propuesto un sistema analítico basado en diferencias. Este sistema se ha tomado como punto de partida para el desarrollo del posterior FRBS y el diseño de la base de conocimiento correspondiente. Al final de este apartado se muestra un sistema de evaluación de la calidad para el estimador diferencial. Este evaluador se usa como indicador para la propagación del valor de SMI calculado por el sensor con objeto de que pueda ser usado como base en la configuración de otros nodos del entorno.

5.1.1 FUNCIONAMIENTO

El funcionamiento general del Sistema Diferencial, o SD, se basa en cálculos sencillos (sumas, restas, multiplicaciones y divisiones) tomando como entrada valores instantáneos de la magnitud bajo estudio, así como valores históricos. El resultado del sistema será el tiempo que el sensor estará en reposo[47] hasta que deba tomar una nueva medida. Este tiempo será mayor si el sistema determina que el fenómeno bajo estudio está arrojando valores normales y lejos de zonas críticas. Por el contrario, el tiempo de reposo será más corto si el SD estima que el entorno puede fluctuar próximamente acercándose a zonas donde se necesite mayor precisión (más medidas por unidad de tiempo). A este periodo de tiempo de reposo se le ha denominado en las contribuciones producidas por la presente tesis como SMI, del inglés *Sleep Mode Interval*. Este tiempo se obtendrá en función de una serie de comparaciones y diferencias entre los valores de entrada y unos umbrales prefijados, que formarán los parámetros de ajuste del sistema.

El requisito fundamental de este sistema es que lleve a cabo su tarea de la manera más sencilla posible, computacionalmente hablando. Cabe destacar que el sistema diseñado es un esquema, el cual debe ser adaptado a cada magnitud y entorno en concreto.

[47] A este tiempo también se le suele denominar tiempo de sueño, de inactividad, o con la denominación proveniente del inglés *idle*. Al conjunto de tiempo de reposo y tiempo de actividad también se le conoce como ciclo de trabajo (en inglés *duty time*).

El segundo requisito que debe cumplir el SD está estrechamente relacionado con la magnitud bajo estudio. Así pues, este sistema debe ser empleado en magnitudes o entornos que presentan una moderada o alta inercia. A pesar de esto, el sistema, en función del ajuste de sus parámetros, podría adaptarse a una amplia variedad de magnitudes y entornos, siempre y cuando estos fueran lo suficientemente estacionarios para tiempos superiores al mayor tiempo de reposo deseado. Por lo tanto, los límites de aplicación vendrían dados por los tiempos mínimo o máximo de sueño que el sistema produciría como salida ante una evaluación de sus entradas.

Aplicaciones típicas de estos sistemas serían el control de la climatización, desgaste de piezas, control de fluidos, concentración de gases, etc. Además, como se podrá comprobar, este sistema también se comportará de manera satisfactoria antes magnitudes como la presión sonora [48], cuyo comportamiento es, a priori, poco estacionario.

5.1.2 MÉTODO GENERAL DE CÁLCULO

Como ya se ha comentado, el método de cálculo se basa en la variación de la variable, o variables de entrada, con respecto a valores tomados con anterioridad. Estas variaciones, junto con la evaluación del valor absoluto de la magnitud, son sometidas a la comparación con una serie de umbrales concretos, fruto del ajuste del sistema. El proceso que se sigue se detalla en los siguientes tres pasos:

- Primero: el sensor mide a través de una sonda el valor actual de la magnitud bajo estudio y calcula la diferencia entre este valor y el obtenido en el ciclo anterior. En función del valor absoluto de esta diferencia, se podrá estimar si el cambio producido corresponde a una fluctuación menor, o por el contrario, la variación de medidas es tan grande que implica un cambio importante en el entorno.
- Segundo: se comprueba la región en la que se encuentra el valor medido por la sonda. Las regiones están delimitadas por los umbrales definidos en el modelo. Cada región comprende los valores que se pueden considerar que deben tener un mismo tratamiento. De esta manera se pueden encontrar regiones de valores normales, moderadamente fuera de lo normal o críticos.
- Tercero: este es el paso de cálculo en el se tienen en cuenta los siguientes factores:
    - Las diferencias obtenidas en el primer paso.
    - La región en la que se encuentra el valor medido.
    - Si el valor medido implica un cambio de región.
    - Nivel de batería.
    - Tiempo de reposo (SMI) obtenido en ejecuciones anteriores.

Como se puede apreciar, los puntos que se han presentado son tan solo un esquema general. Esto se debe a que este método debe particularizarse para cada magnitud y entorno concreto.

Los parámetros que describen el método se detallaran en el siguiente apartado, y a continuación de éste, se mostrará la aplicación de los mismos a un sistema para el control del tiempo de sueño en un sensor que mide la presión sonora en interiores.

[48] Nivel de presión sonora (dB): En el aire, 20 veces el logaritmo (de base 10) de una presión sonora determinada con respecto a la presión sonora de referencia de 20 µPa, considerado el umbral de audición humano. El umbral del dolor se establece comúnmente en 20 Pa.

5.1.3 PARÁMETROS DEL SISTEMA DIFERENCIAL

Como se ha comentado, el SD elabora una previsión de tiempo de reposo (SMI) en función a unos parámetros dependientes del entorno, de la historia reciente y del propio sensor. Además, el fenómeno bajo estudio debe mantener cierta estacionariedad, aunque los ajustes propios del sistema le permiten ser capaz de adaptarse a gran variedad de entornos. Esto se consigue gracias a los dos parámetros principales del SD, SMI máximo (SMI_{MAX}) y SMI mínimo (SMI_{MIN}). Así pues, con valores apropiados de estos dos parámetros se puede acotar el máximo y mínimo periodo de muestreo de la magnitud bajo estudio. Se debe destacar, que si los valores de SMI_{MIN} son cercanos al segundo, el propio tiempo que normalmente tarda un sensor en cambiar entre el estado de activo a reposo y viceversa, ronda los cientos de milisegundos, con lo cual este sistema perdería su sentido y su efectividad al poder estar el sensor entrando y saliendo del reposo continuamente, no obteniéndose ninguna ventaja del uso de este sistema.

A continuación se enumeran los demás parámetros usados por el método. La forma abreviada de todos ellos proviene de su denominación en inglés, la cual será mostrada para cada uno.

Diferencia (*d*)

El parámetro *d*, del inglés *difference*, es la diferencia en valor absoluto entre la nueva medida tomada y la medida tomada en el anterior ciclo.

SMI previo (*ps*)

El parámetro *ps* (del inglés *previous SMI*) es el valor de SMI calculado por el método la vez anterior que este se ejecutó.

Delta (Δ)

El parámetro Δ establece el ajuste al valor del SMI previo (*ps*) en función de la diferencia entre la medida actual y la anterior (*d*).

Salto de región (*rh*)

El parámetro *rh* (del inglés *región hops*), mide la variación del valor leído con respecto al anterior de manera cualitativa. Así, *rh* funciona como un factor de ajuste al método, que toma el valor del número de regiones establecidas para la magnitud bajo estudio que se hayan atravesado, entre la medida actual y la anterior. Así pues, si la medida anterior estaba en una región diferente, pero contigua a la que está la medida actual, este valor será 1. Estas regiones son parámetros del modelo y por tanto deben ser ajustadas en función de la magnitud y de la aplicación concreta que se quiera dar al sensor. Un mayor número de regiones conllevará una mayor probabilidad de que el parámetro *rh* pueda tomar valores mayores y por tanto influir en mayor medida en el nuevo SMI. Para obtener este valor de manera analítica, a cada región se le debe asignar un valor entero. Como convención se ha establecido que estos valores sean positivos para las regiones por encima del valor tomado como referencia, y negativos para las regiones con valores inferiores.

$$rh = \text{abs}(\textit{región actual} - \textit{región previa}) \qquad \textbf{Ecuación 3}$$

Estabilidad (*st*)

El parámetro *st* (del inglés *steady*) informa de la estabilidad en las muestras tomadas a lo largo del tiempo, aumentando en uno cada vez que una nueva muestra se mantiene dentro de una misma región que la muestra tomada anteriormente. En el caso de que se produzca un cambio de región este valor se reinicia a cero.

Nuevo SMI (*ns*)

El parámetro *ns* (del inglés *new SMI*) es tanto un valor de cálculo intermedio, como el resultado final que da el SD.

Ajuste por nivel de batería (*B*)

El parámetro B depende del nivel de batería del sensor, y su función es modelar el ajuste debido a la batería que tendrá el nuevo SMI calculado.

Como se puede apreciar, no se ha mostrado ningún valor numérico concreto. Esto es como consecuencia de que, como se verá a continuación, el método de cálculo concreto se debe desarrollar para una aplicación en particular. Se debe destacar que el sistema desarrollado es para el control de maquinaria en funcionamiento a través de la medición del ruido ambiente (magnitud presión sonora). Este ejemplo de desarrollo incorpora las fórmulas básicas del SD. Se ha preferido mostrar estas fórmulas junto con la aplicación concreta, dado que no se busca la generalidad, sino la particularidad, ya que una aplicación de control de gases en una planta química, distará mucho de la mencionada aplicación de medición de polución sonora, siendo a priori no abordable, un método general de control del tiempo de sueño basado en funciones analíticas.

5.1.4 SISTEMA DIFERENCIAL PARA EL CONTROL DEL TIEMPO DE SUEÑO DE UN SENSOR EN UNA APLICACIÓN DE MONITORIZACIÓN DE LA PRESIÓN SONORA

A continuación se mostrarán las fórmulas y valores concretos de ajuste del SD para una aplicación de control del ruido ambiental a través de la presión sonora, la cual se usa típicamente en aplicaciones de contaminación acústica (Filipponi et al. 2008).

El objeto de esta medición es la de controlar un entorno en el que el sonido estaría producido por equipos o maquinaria en funcionamiento. Así pues, la red de sensores sería desplegada, por ejemplo, en una sala de máquinas donde se ubiquen equipos de climatización o los equipos de bombeo de aguas, etc. Además, este tipo de aplicaciones son no invasivas, ya que no requieren de modificaciones en los equipos en funcionamiento. De esta manera, podrían ser desplegadas en cualquier tipo de instalación cuyo funcionamiento sea continuado y los patrones de ruido estén vinculados al régimen de trabajo de la maquinaria. El control de sistemas basados en el sonido no es algo nuevo (Vijayraghavan & Krishnan 1998, Tandon & Choudhury 1999), pero junto con las redes de sensores se abre la posibilidad de incorporar, al menos, sistemas de monitorización y alarma para maquinaria que no dispone de esta posibilidad, a través del análisis del sonido (por supuesto también podría compatibilizarse con la medición de la temperatura ambiente, vibraciones con los acelerómetros, etc.)

El hecho de usar la medición del ruido ambiente, con objeto de monitorizar el régimen de trabajo de maquinaria en funcionamiento, se debe a la necesidad de ajuste subjetivo y

experto que requiere, sobre todo en su aplicación a maquinaria que no haya sido pensada para ser monitorizada de manera automática.

Así pues, los valores que se muestran a continuación son un ejemplo de ajuste para el SD en el que se define un régimen de funcionamiento normal ente los 55 dBA[49] y los 65 dBA, coincidiendo este rango con la región central. Este nivel sonoro coincidiría con un zumbido suave o una conversación tranquila. Se han definido otras cuatro regiones, dos para valores superiores y otras dos para valores por debajo de la región central. Las regiones adyacentes a la central se consideran como rango de funcionamiento permitido pero que indica que el régimen de trabajo ha aumentado o disminuido respectivamente. Las dos regiones restantes, la superior y la inferior se establecen para marcar funcionamientos anómalos, ya sea por un excesivo ruido[50], que puede indicar regímenes de trabajo forzados o averías (piezas que rozan o chirrían); o por un defecto de ruido, lo cual indicaría que la maquinaria no está funcionando. Como puede verse en la Tabla 9, a cada una de las regiones se le ha asignado un valor numérico para el cálculo del parámetro *rh*.

Tabla 10. Regiones establecidas en el SD para la medición de presión sonora.

Región	Crítica inferior	Inferior	Normal	Superior	Crítica superior
Umbrales (dBA)		45	55	65	75	
Valor de región	-2	-1	0	1	2

Adicionalmente a los parámetros mostrados en el apartado anterior, el método de cálculo del nuevo SMI se ha dividido en cuatro pasos, los cuales necesitan de la definición de los valores de los parámetros Δ y B.

Valor de Δ:

$$\Delta = \begin{cases} +10\%, & si\ d < 10\ dB \\ -10\%, & si\ 10\ dB \leq d < 20\ dB \\ -20\%, & si\ d \geq 20\ dB \end{cases}$$ **Ecuación 4**

Valor de B:

$$\mathrm{B} = \begin{cases} +0\%, & si\ el\ nivel\ de\ batería > 75\% \\ +10\%, & si\ el\ 50\% > nivel\ de\ batería \leq 75\% \\ +30\%, & si\ el\ 25\% > nivel\ de\ batería \leq 50\% \\ +60\%, & si\ el\ 10\% > nivel\ de\ batería \leq 25\% \\ +100\%, & si\ el\ nivel\ de\ batería \leq 10\% \end{cases}$$ **Ecuación 5**

Primer paso, **cálculo del valor de incremento *i*:**

$$i = i_o - (rh \cdot 0{,}1) + (st \cdot 0{,}05)$$ **Ecuación 6**

[49] Nivel de potencia sonora con ponderación A (dBA): Es la medida de la presión sonora con ponderación A. La ponderación A tiene en cuenta que el oído humano no es igual de sensible a todas las frecuencias dentro de la banda de audición de los 20 Hz a los 20 KHz. Por ejemplo, un tono de 100 Hz con presión sonora de 70 dB es percibido por el oído como si fuera 19,1 dB menos, dando una presión sonora equivalente de 50,9 dBA.

[50] Un nivel de presión sonora de 100 dBA se considera perjudicial, así como una exposición prolongada a sonidos con niveles de 85 dBA.

Donde i_0 es el valor de incremento calculado en la ejecución anterior del método. El efecto del cambio de región provoca una disminución en el incremento de una décima por cada región de cambio de la nueva medida. De esta manera, el nuevo SMI será menor dado que se ha detectado una importante variación. Por el contrario, el factor de estabilidad aumenta *i* en cinco centésimas por cada nueva ejecución que no ocasione un cambio de región, indicando que las medidas se encuentran estables dentro de una región de funcionamiento. Se debe hacer notar, que tan sólo uno de estos valores será diferente de 0 en una ejecución.

Segundo paso, primera estimación:

Una vez calculado el factor de incremento i, se calcula la primera estimación del nuevo SMI, ns_0.

$$ns_0 = ps \cdot i \qquad \textbf{Ecuación 7}$$

Tercer paso, valor de ns_Δ:

Para obtener el resultado final de esta fase ns_Δ, el valor a ns_0 se actualiza con el valor de Δ.

$$ns_\Delta = ns_o\,(1 + \Delta) \qquad \textbf{Ecuación 8}$$

Cuarto paso, obtención de ns:

Se ajusta ns_Δ con el parámetro B dependiente de la carga de batería disponible. De esta manera se obtiene el valor definitivo del nuevo SMI.

$$ns = ns_\Delta\,(1 + B) \qquad \textbf{Ecuación 9}$$

Como se puede apreciar, el cálculo del nuevo SMI no requiere de operaciones complejas, aunque sí de operaciones en coma flotante. El SD ha sido simulado, junto con el basado en FRBS con múltiples señales de entrada. Estos resultados son mostrados y analizados en el Capítulo III.

5.1.5 MECANISMO DE EVALUACIÓN DE LA CALIDAD DEL SD

Adicionalmente al propio cálculo del SMI que realiza un sensor, se ha diseñado un factor de evaluación de la calidad de las estimaciones. Este estimador está basado en los aciertos o fallos en la predicción realizada. Así pues, se define como un acierto el suponer que la magnitud va a cambiar, reducir el SMI, y que esta lo haga; o que se suponga que la magnitud se mantendrá estable, pronosticar un SMI más largo y que efectivamente la siguiente vez que se lea la magnitud siga dentro de la misma región. Con respecto a los fallos, se consideran tales, si no se cumple alguna de las condiciones de éxito anteriores.

Así pues, se define un nuevo parámetro entero, *Q* (del inglés *Quality*), cuyo valor puede ir desde -∞ a +∞. Siendo los valores positivos los que indican, históricamente, que el SD ha acertado en la predicción más veces que ha fallado.

El algoritmo de cálculo de *Q* necesita de la definición de dos parámetros adicionales que indican lo que ha considerado como un SMI corto o largo para compararlo con el

anteriormente calculado y establecer si la predicción anterior ha sido un éxito o no. Estos valores son SMI_{SHORT}[51], y SMI_{LONG}[52], y se definen de la siguiente manera:

$$SMI_{SHORT} = SMI_{MIN} + (SMI_{MAX} - SMI_{MIN})/4 \quad \textbf{Ecuación 10}$$

$$SMI_{LONG} = SMI_{MIN} + (SMI_{MAX} - SMI_{MIN})/2 \quad \textbf{Ecuación 11}$$

El algoritmo de evaluación del factor de calidad *Q* es el siguiente:

$$SI\ \boldsymbol{rh} = 0\ Y\ \boldsymbol{ps}\ NO\ ES\ \boldsymbol{corto}\ ENTONCES\ \boldsymbol{Q} = \boldsymbol{Q} + 1\ SI\ NO\ \boldsymbol{Q} = \boldsymbol{Q} - 1$$

$$SI\ \boldsymbol{rh} = 1\ Y\ \boldsymbol{ps}\ NO\ ES\ \boldsymbol{largo}\ ENTONCES\ \boldsymbol{Q} = \boldsymbol{Q} + 1\ SI\ NO\ \boldsymbol{Q} = \boldsymbol{Q} - 1$$

$$SI\ \boldsymbol{rh} = 2\ Y\ \boldsymbol{ps}\ ES\ \boldsymbol{corto}\ ENTONCES\ \boldsymbol{Q} = \boldsymbol{Q} + 1\ SI\ NO\ \boldsymbol{Q} = \boldsymbol{Q} - 2$$

Donde:

- corto: SMI < SMI_{SHORT}
- largo: SMI > SMI_{LONG}

Así pues, el factor de calidad *Q* es actualizado cada ejecución, y cuando llegue a un determinado umbral positivo, se enviará el valor calculado en la última ejecución para que sea tenido en cuenta como valor de inicio en los sensores que estén controlando la misma magnitud dentro de la red de sensores. En el Capítulo III se mostrarán algunos resultados obtenidos para valores concretos del método.

5.2 SISTEMA FRBS PARA LA GESTIÓN AUTOMÁTICA DEL TIEMPO DE SUEÑO.

Como se avanzó en la sección 3.3, dentro de la arquitectura multi-agente presentada, se definió una arquitectura BDI ligera borrosa (FLBDIa) la cual sería aplicada al Agente de Gestión (AG). Esta arquitectura asimilaba las creencias, deseos, metas e intenciones de un agente BDI (Rao & Georgeff 1991). Así pues, esto requiere de una aplicación concreta a la que destinar el agente, ya que sin la definición concreta de la misma no sería posible llegar a una solución. Esto es consecuencia de la propia definición de un agente como sistema creado para producir un efecto concreto de manera autónoma.

Así pues, dentro del presente trabajo de investigación, la arquitectura BDI ligera borrosa que rige el funcionamiento del Agente de Gestión se ha diseñado para que controle el tiempo de sueño (SMI) de un sensor a través del análisis y predicción del comportamiento de la magnitud bajo estudio. Más aun, debido a que este problema es aún muy general, se mostrará la aplicación para el control de la magnitud de presión sonora presentada en la sección anterior al evaluar el Sistema Diferencial. De esta manera se podrá comparar los resultados obtenidos con ambos sistemas, lo cual será mostrado en el Capítulo III.

Sin embargo, en primer lugar se presentará el formato para la definición de bases de conocimiento, el cual puede ser usado para su diseño y modelado, así como para su envío por una red de sensores.

[51] *SHORT*, en inglés corto o breve.
[52] *LONG*, en inglés largo, de larga duración.

5.2.1 FORMATO PARA DEFINICIÓN DE BASES DE CONOCIMIENTO

Para poder trabajar con las bases de conocimiento de manera que éstas puedan ser procesadas de manera automatizada, y a su vez diseñadas manualmente, se ha elegido para este fin el formato XML (del inglés *eXtensible Markup Language*). El formato XML es ampliamente usado para la definición de todo tipo de información dentro del entorno de la WWW. El formato XML también es usado dentro de protocolos de comunicación para el acceso de servicios web, como SOAP[53].

El lenguaje XML se basa en la definición de una estructura jerárquica de elementos delimitados por marcas. El formato XML es totalmente extensible, escalable y adaptable a casi cualquier tipo de documento. Cuenta con un extenso y probado conjunto de APIs[54], así como ficheros para la definición formal de su contenido como las DTD[55] o los esquemas[56], además de ficheros para la representación final de la información a través de hojas de estilo.

A continuación, como muestra, se presenta un ejemplo de una base de conocimiento definida en XML con el formato desarrollado en la presente tesis, al cual se le ha denominado XML^{KB}. Más adelante se comentará en detalle el mismo. En concreto, el ejemplo presenta una base de conocimiento con dos variables de entrada, temperatura y humedad, y una de salida, tiempo, junto con una sola regla borrosa.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="FRBSxsl.xsl"?>
<!DOCTYPE kb SYSTEM "kb.dtd">
<kb>
	<db>
		<input>
			<var	id="1"
				name="temperatura"
				type="1"
				min="0"
				max="100"
				description="temperatura medida con la sonda interna" >
				<fs id="1" name="baja">
					<point><x>0.0</x><y>0.0</y></point>
					<point><x>0.0</x><y>1.0</y></point>
					<point><x>0.5</x><y>0.0</y></point>
					<point><x>0.5</x><y>0.0</y></point>
				</fs>
				<fs id="2" name="media">
					<point><x>0.0</x><y>0.0</y></point>
					<point><x>0.5</x><y>1.0</y></point>
					<point><x>0.5</x><y>1.0</y></point>
					<point><x>1.0</x><y>0.0</y></point>
				</fs>
```

[53] SOAP, del inglés *Simple Object Access Protocol*, es un protocolo que permite el acceso a diferentes funciones disponibles en un servidor web bajo el concepto de llamada a procedimiento remoto (RPC). Se implementa sobre el protocolo HTTP.

[54] API, del inglés *Application Program Interface*, es un programa o conjunto de programas que ofrecen una serie de servicios accesibles a través de funciones, métodos y/o clases para un lenguaje o sistema operativo concreto.

[55] Un documento DTD (del inglés *Document Type Definition*) mantiene la especificación formal de un documento XML y permite su validación. Se debe hacer notar que un documento XML no necesita de una DTD para su definición o uso, pero sin una DTD, el formato de un documento XML podría hacerse errático y difícil de mantener. Una DTD usa su propio formato de definición de elementos y no sigue el lenguaje XML en sí mismo.

[56] Un esquema XML es también un documento XML que presenta la estructura formal de dicho documento y permite la validación del mismo arreglo al esquema.

```
                        <fs id="3" name="alta">
                                <point><x>0.5</x><y>0.0</y></point>
                                <point><x>0.5</x><y>0.0</y></point>
                                <point><x>1.0</x><y>1.0</y></point>
                                <point><x>1.0</x><y>0.0</y></point>
                        </fs>
                </var>
                <var    id="2"
                        name="iluminación"
                        type="2"
                        min="0"
                        max="700"
                        description="iluminacion medida con la sonda interna">
                        <fs id="1" name="baja">
                                <point><x>0.0</x><y>0.0</y></point>
                                <point><x>0.0</x><y>1.0</y></point>
                                <point><x>0.4</x><y>0.0</y></point>
                                <point><x>0.4</x><y>0.0</y></point>
                        </fs>
                        <fs id="2" name="alta">
                                <point><x>0.0</x><y>0.0</y></point>
                                <point><x>0.5</x><y>1.0</y></point>
                                <point><x>0.5</x><y>1.0</y></point>
                                <point><x>1.0</x><y>0.0</y></point>
                        </fs>
                </var>
        </input>

        <output>
                <var id="1" name="tiempo">
                        <fs id="1" name="corto">
                        <point><x>0.0</x><y>0.0</y></point>
                        <point><x>0.0</x><y>1.0</y></point>
                        <point><x>0.5</x><y>0.0</y></point>
                        <point><x>0.5</x><y>0.0</y></point>
                        </fs>
                        <fs id="2" name="medio">
                        <point><x>0.0</x><y>0.0</y></point>
                        <point><x>0.5</x><y>1.0</y></point>
                        <point><x>0.5</x><y>1.0</y></point>
                        <point><x>1.0</x><y>0.0</y></point>
                        </fs>
                        <fs id="3" name="largo">
                        <point><x>0.5</x><y>0.0</y></point>
                        <point><x>0.5</x><y>0.0</y></point>
                        <point><x>1.0</x><y>1.0</y></point>
                        <point><x>1.0</x><y>0.0</y></point>
                        </fs>
                </var>
        </output>
</db>

<rb>
        <rule id="1">
                <antecedent>
                        <proposition id="1">
                                <pvar id="1">temperatura</pvar>
                                <pfs id="3">alta</pfs>
                        </proposition >
                        <proposition id="2">
                                <pvar id="2">humedad</pvar>
                                <pfs id="1">baja</pfs>
                        </proposition>
                </antecedent>
                <consequent>
                        <proposition id="1">
                                <pvar id="1">tiempo</pvar>
                                <pfs id="1">corto</pfs>
                        </proposition>
                </consequent>
        </rule>
</rb>
</kb>
```

5.2.2 DESCRIPCIÓN DEL FORMATO XML^{KB}

El formato XML^{KB} se ha diseñado para que sea lo más flexible posible, y a la vez, fiel a la forma tradicional de definir una base de conocimiento. Así pues, en primer lugar, dentro de una base de conocimiento (elemento `kb`) la información se divide en base de datos y base de reglas, manteniéndose estos dos bloques en XML^{KB}. Cada base de conocimiento permite la definición de un identificador (atributo *id* de `kb`, obligatorio[57]) y un nombre (atributo *name* de `kb`) para diferenciarlas en posibles envíos al mismo sensor.

La base de datos (elemento `db`) permite la inclusión de dos tipos de elementos, variables de entrada (elemento `input`) y variables de salida (`output`). Dentro de cada uno de estos elementos se pueden definir cualquier número de variables (elementos `var`).

Cada elemento `variable`, ya sea de entrada o salida, contiene un número diverso de conjuntos borrosos (elementos `fs`) que la definen. Para diferenciar cada variable dentro de la base de datos, cada una tiene un identificador (atributo *id*, obligatorio), un nombre (atributo *name*) y una descripción a efectos de legibilidad y mantenimiento del documento (atributo *description*). Adicionalmente, se han definido tres parámetros que modelan la magnitud que el sensor debe usar para el cálculo y el rango de valores a aplicar al proceso de fuzzificación y defuzzificación. Estos parámetros son:

- Tipo de variable (atributo *type*, obligatorio).
- Valor inferior del rango de la magnitud (atributo *min*, obligatorio).
- Valor superior del rango (atributo *max*, obligatorio).

Cada conjunto borroso, al igual que las variables, posee tanto atributo *id* (obligatorio) como *name* y pueden contener un número variable de puntos (elementos `point`). Dada la definición de un conjunto borroso según este formato, tan sólo se podrían definir formas poligonales. Sin embargo, bastaría con definir otros elementos que permitieran la definición de conjuntos borrosos con formas no poligonales, como por ejemplo normales. Cada elemento `point` consta de dos elementos, `x` e `y`, los cuales tienen como contenido el valor de cada coordenada.

Para mostrar de manera gráfica como se ha definido la variable temperatura del anterior ejemplo del formato XML^{KB}, en la Figura 13 se muestran los conjuntos borrosos de la misma.

La base de reglas (elemento `rb`) se basa en los elementos existentes dentro de la base de datos, ya que se usan los identificadores previamente definidos para las variables y conjuntos borrosos. Así pues, la base de reglas está formada por un conjunto de reglas borrosas (elementos `rule`), cada una con antecedentes (elemento `antecedent`) y consecuentes (elemento `consequent`). Tanto los antecedentes, como los consecuentes están compuestos por un número variable de proposiciones (elementos `proposition`). Cada proposición está identificada por un atributo *id* (obligatorio), y detalla un conjunto borroso concreto para una variable de las que intervienen en la regla borrosa. El conjunto borroso se define a través del elemento `pfs`, y la variable a la que pertenece éste con el elemento `pvar`. Ambos elementos poseen un atributo, *id*

[57] Los atributos obligatorios aparecen marcados como `#REQUIRED` en la DTD (ver en el apartado siguiente), mientras que los atributos opcionales están marcados como `#IMPLIED`.

(obligatorio), que debe coincidir con el identificador del conjunto borroso y variable definidos previamente.

```
<var     id="1"
         name="temperatura"
         type="1"
         min="0"
         max="100"
         description="temperatura  medida  con  la
         sonda interna" >
     <fs id="1" name="baja">
         <point><x>0.0</x><y>0.0</y></point>
         <point><x>0.0</x><y>1.0</y></point>
         <point><x>0.5</x><y>0.0</y></point>
         <point><x>0.5</x><y>0.0</y></point>
     </fs>
     <fs id="2" name="media">
         <point><x>0.0</x><y>0.0</y></point>
         <point><x>0.5</x><y>1.0</y></point>
         <point><x>0.5</x><y>1.0</y></point>
         <point><x>1.0</x><y>0.0</y></point>
     </fs>
     <fs id="3" name="alta">
         <point><x>0.5</x><y>0.0</y></point>
         <point><x>0.5</x><y>0.0</y></point>
         <point><x>1.0</x><y>1.0</y></point>
         <point><x>1.0</x><y>0.0</y></point>
     </fs>
</var>
```

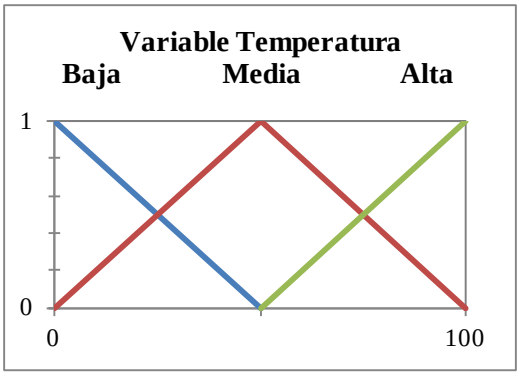


Figura 13. Ejemplo de variable definida en con XML^{KB} y la representación gráfica de sus conjuntos borrosos.

Una vez presentada la estructura completa del formato XML^{KB}, en la Figura 14 se puede observar un diagrama representativo de dicho formato. En el siguiente apartado se presenta la DTD del formato XML^{KB}, elemento necesario para la validación[58] de documentos XML.

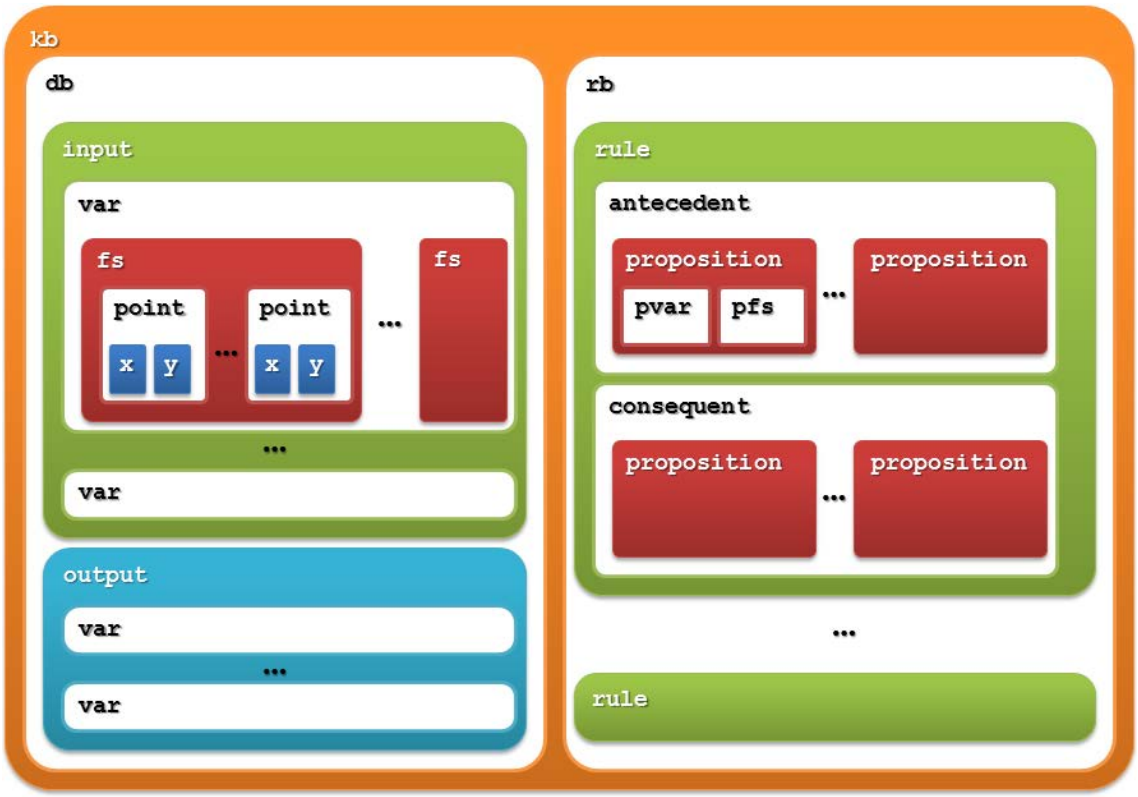


Figura 14. Esquema general de un fichero XML^{KB}.

[58] La validación de un documento XML consiste en comprobar si su estructura se corresponde con lo marcado en una DTD o esquema de referencia.

5.2.3 DTD DEL FORMATO XML^{KB}

A continuación se presenta la DTD[59] del formato XML de definición de bases de conocimiento XML^{KB}. Esta DTD puede ser usada para la validación de cualquier fichero XML^{KB}.

```
<?xml version='1.0' encoding='UTF-8'?>
<!ELEMENT kb (rb|db)+>
<!ATTLIST kb
 name CDATA #IMPLIED
 id CDATA #REQUIRED>
<!ELEMENT db (output|input)+>
<!ELEMENT input (var)+>
<!ELEMENT var (fs)+>
<!ATTLIST var
 description CDATA #IMPLIED
 max CDATA #REQUIRED
 min CDATA #REQUIRED
 type CDATA #REQUIRED
 name CDATA #IMPLIED
 id CDATA #REQUIRED>
<!ELEMENT fs (point)+>
<!ATTLIST fs
 name CDATA #IMPLIED
 id CDATA #REQUIRED>
<!ELEMENT point (y|x)+>
<!ELEMENT x (#PCDATA)>
<!ELEMENT y (#PCDATA)>
<!ELEMENT output (var)+>
<!ELEMENT rb (rule)+>
<!ELEMENT rule (consequent|antecedent)+>
<!ATTLIST rule
 id CDATA #REQUIRED>
<!ELEMENT antecedent (proposition)+>
<!ELEMENT proposition (pfs|pvar)+>
<!ATTLIST proposition
 id CDATA #REQUIRED>
<!ELEMENT pvar (#PCDATA)>
<!ATTLIST pvar
 id CDATA #REQUIRED>
<!ELEMENT pfs (#PCDATA)>
<!ATTLIST pfs
 id CDATA #REQUIRED>
<!ELEMENT consequent (proposition)+>
```

Lo más significativo con respecto a la DTD es tener en cuenta que existen atributos que son obligatorios, ya que se necesitan para el correcto funcionamiento del motor de inferencia, y deben estar presentes en el fichero XML que describa la base de conocimiento. Con respecto a los elementos, todos están marcados con el símbolo '+', lo cual significa que al menos debe aparecer un elemento de este tipo. Sin embargo, en una DTD no se puede especificar un número concreto de elementos que deben definirse, salvo uno o muchos, o ninguno o muchos.

El hecho de incluir en esta tesis una DTD tiene como objetico el de completar la especificación del formato XML^{KB} y que estos ficheros puedan ser validados antes de su uso. Así pues, se asegura un poco más, el correcto funcionamiento de los FRBS incorporados en un sensor, ya que se requiere que los ficheros que contengan bases de conocimiento, y sean enviados por el

[59] Para una completa información sobre la definición de DTDs puede visitarse http://www.w3schools.com/dtd/.

protocolo WISMAP, no solo estén bien formados[60], sino que correspondan fielmente con la DTD para no generar errores a la hora de su procesado.

Uno de estos procesos es la conversión de una base de conocimiento en formato XML^{KB} a la sintaxis para la transferencia por la red de sensores usando el protocolo WISMAP. Este punto se aborda con detalle en la siguiente sección.

5.2.4 FORMATO DE DATOS PARA LA TRANSFERENCIA DE BASES DE CONOCIMIENTO

La especificación de bases de conocimiento en formatos como XML permite una fácil legibilidad, mantenimiento, modificación, e incluso una rápida adaptación a su visualización en un navegador web a través de transformaciones con hojas de estilo. Sin embargo, el uso de marcas de texto codificadas con caracteres ASCII de 7 bits, lo hace poco apropiado para su envío en redes con dispositivos con prestaciones limitadas, tales como una red de sensores. Por este motivo, para el envío de bases de conocimiento por la red de sensores, se ha usado un nuevo formato de transferencia básico o *raw* [61](en inglés crudo) que compacta la información de las bases de conocimiento, reduciendo considerablemente su tamaño.

La compactación, que no compresión, se realiza a través de la eliminación de toda la información textual no procesable por el motor de inferencia, como marcas de apertura y cierre, caracteres especiales, etc. Así pues, en el formato *raw* resultante, se mantiene todo lo esencial, como coordenadas de los puntos, identificadores de variables, conjuntos borrosos, proposiciones, etc. De esta manera, una regla, la cual en lenguaje natural normalmente se escribiría:

SI *temperatura* ES *alta* Y *humedad* ES *baja* ENTONCES *tiempo* ES *corto*.

Y usando el formato XML^{KB}:

```
<rule id="1">
            <antecedent>
                  <proposition id="1">
                        <pvar id="1">temperatura</pvar>
                        <pfs id="3">alta</pfs>
                  </proposition >
                  <proposition id="2">
                        <pvar id="2">humedad</pvar>
                        <pfs id="1">baja</pfs>
                  </proposition>
            </antecedent>
            <consequent>
                  <proposition id="1">
                        <pvar id="1">tiempo</pvar>
                        <pfs id="1">corto</pfs>
                  </proposition>
            </consequent>
</rule>
```

En formato *raw* sería algo una secuencia de bytes con los valores que identifican a cada elemento por su posición dentro de la secuencia como puede verse en la Figura 15.

[60] Un fichero XML bien formado es aquel cuyas marcas están adecuadamente abiertas y cerradas en su nivel correspondiente (<marca>…</marca>), además de no usar caracteres prohibidos, como normalmente son los que se salen del código ASCII de 7 bits.

[61] Se ha utilizado el término *raw* por ser una denominación típica de formatos de todo tipo de información sin compresión y sin estructuras complejas, como ficheros gráficos o de audio.

Regla										
			Antecedentes					Consecuentes		
Id	Nº antecedentes	Nº consecuentes	Nº prop.	Var. de entrada	Conjunto borroso	Var. de entrada	Conjunto borroso	Nº prop.	Var. de salida	Conjunto borroso
1	2	1	2	1	3	2	1	1	1	1

Figura 15. Formato raw de bases de conocimiento.

Como puede observarse, la información guarda toda su coherencia, además de tener un tamaño sensiblemente menor. Sin embargo, se hace mucho más complejo su análisis al eliminar las referencias textuales. La estructura completa del formato *raw* para bases de conocimiento es la siguiente (cada campo está seguido por el tipo de dato usado).

- Base de conocimiento:
  - Identificador de base de conocimiento (Entero)
  - Número de variables de entrada (Entero)
  - Número de variables de salida (Entero)
  - Número de reglas (Entero)
  - Lista de variables de entrada.
  - Lista de variables de salida.
  - Lista de reglas.

Cada una de las dos listas de variables se compone de la sucesión de éstas con el siguiente formato:

- Variable:
  - Identificador (Entero).
  - Tipo (Byte)
  - Mínimo (Coma flotante doble precisión)
  - Máximo (Coma flotante doble precisión)
  - Número de conjuntos borrosos (Entero)
  - Lista de conjuntos borrosos.

La lista de conjuntos borrosos se compone de un número previamente definido de estos. Un conjunto borroso se codifica igual para todas las variables y lo componen cuatro puntos[62]. Su formato es el siguiente:

- Conjunto borroso:
  - Identificador (Entero)
  - Primer punto:
    - X (Coma flotante de simple precisión)
    - Y (Coma flotante de simple precisión)
  - Segundo punto:
    - X (Coma flotante de simple precisión)
    - Y (Coma flotante de simple precisión)
  - Tercer punto:
    - X (Coma flotante de simple precisión)
    - Y (Coma flotante de simple precisión)

[62] Se ha optado por predefinir el número de puntos a cuatro, ya que cumplía con los objetivos planteados, y de esta manera evitar el tener que añadir el número de puntos que compone cada conjunto borroso. Una posterior versión podría incluir una solución más versátil.

    - Cuarto punto:
        - X (Coma flotante de simple precisión)
        - Y (Coma flotante de simple precisión)

La lista de reglas está compuesta por una sucesión de reglas codificadas como se ha visto en el ejemplo anterior. A continuación se detalla su formato siguiendo el esquema de los demás elementos.

- Regla:
    - Identificador (Entero)
    - Número de antecedentes (Entero)
    - Número de consecuentes (Entero)
    - Antecedentes.
    - Consecuentes.

Tanto los antecedentes, como los consecuentes, se componen de una lista de proposiciones que informan de la variable y conjunto borroso que forma parte de dicha proposición en la regla.

- Antecedentes/Consecuentes:
    - Número de proposiciones (Entero)
    - Lista de proposiciones.

Cada proposición sigue el siguiente formato:

- Proposición:
    - Identificador de la variable (Entero)
    - Identificador del conjunto borroso (Entero)

Como se ha podido apreciar, muchos de los campos son de tipo entero, por lo que el tamaño que ocupan es de 32 bits. Esto se ha mantenido dado que algunas arquitecturas de 32 bits siguen un alineamiento de datos en memoria que hace que un dato de un solo byte, resulte en una ocupación de una palabra de 32 bits. Así pues, se ha preferido dejar este tipo de valores para mayor eficiencia en cuanto al acceso a la información en memoria. Sin embargo, sería razonable pensar que para su envío, en muchos de los casos, bastaría con un byte, lo cual generaría ficheros de base de conocimiento tipo *raw* más pequeños.

A continuación, en la Tabla 10, se muestran una serie de ficheros que contienen bases de conocimiento de diferente complejidad escritos en XML^{KB}, su tamaño en bytes, así como el tamaño de estos ficheros comprimidos con el algoritmo ZIP[63] y RAR[64]. También aparece el tamaño que ocuparía la misma base de conocimiento en formato *raw*, y su compresión con ZIP y RAR.

[63] ZIP es un algoritmo de compresión y un tipo de archivos comprimidos muy popular y extendido, creado por Phil Katz en 1989. Está incorporado a muchos sistemas operativos como Unix, Linux, Windows o Mac OS, entre otros.

[64] RAR es un compresor de información y formato de archivo comprimido muy versátil desarrollado por Eugene Roshal (de ahí el nombre del formato, **R**oshal **AR**chive). En la actualidad es propiedad de win.rar GmbH.

Tabla 11. Tamaño en bytes y % de ficheros XML^{KB}, comprimidos y en formato raw.

Variables	Conjuntos borrosos	Reglas	XML^{KB}	ZIP	RAR	raw	raw+ZIP	raw+RAR
4	10	6	6.835	878	797	760	274	222
			100%	12,85%	11,66%	11,12%	4,01%	3,25%
6	14	10	10.793	964	857	1162	283	231
			100%	8,93%	7,94%	10,77%	2,62%	2,14%
6	18	15	16344	935	745	1723	237	182
			100%	5,72%	4,56%	10,54%	1,45%	1,11%
8	24	21	21.329	1003	772	2168	249	187
			100%	4,70%	3,62%	10,16%	1,17%	0,88%

Como se puede observar en la Tabla 10, la reducción que se obtendría para los ficheros tipo XML^{KB} sería considerable, ya sea aplicando algoritmos de compresión o el formato *raw*, yendo desde, aproximadamente un 10% del tamaño original, hasta menos de un 4%. Esto es debido a la alta redundancia de los ficheros XML^{KB} y su estructura repetitiva con un lenguaje fijo. Sin embargo, los resultados comparados con el formato *raw* no consiguen tanta mejora, e incluso, para bases de conocimiento de tamaño reducido, este formato consigue un menor tamaño (aun usando tipos de datos de 32 bits y no de 8 bits). Así pues, es una posibilidad que queda abierta en el caso de aplicaciones en las que se quiera minimizar las comunicaciones vía radio y no haya importantes limitaciones en el proceso en los sensores.

5.2.5 DESCRIPCIÓN DEL SISTEMA

En la presente sección se describirá el FRBS diseñado para dar cumplimiento completo a la tercera hipótesis de la tesis, en la que se postulaba la posibilidad de que un sensor se pueda gestionar de manera autónoma o colaborativa usando sistemas basados en el conocimiento. Al igual que sucede con el SD, se debe hacer una particularización sobre los factores que se tomarán como referencia para el cálculo del tiempo de sueño. Así pues, para poder realizar comparaciones entre ambos métodos, el sistema FRBS para la estimación del SMI se aplicará a la monitorización de la presión sonora.

En consecuencia, la aplicación de un FRBS para la gestión autónoma del tiempo de sueño en una aplicación de monitorización de presión sonora en un sensor, dará como resultado la implementación de una arquitectura BDI ligera borrosa para el Agente de Gestión.

La implementación del FRBS se ha llevado a cabo usando como base el motor de inferencias desarrollado en (Canada-Bago2007), al cual se le ha dotado de una nueva estructura de clases que dé soporte a la definición de bases de conocimiento presentada en los apartados anteriores. Dicho motor es de tipo Mamdani (véase la sección 2.3.1). El proceso de fuzzificación convierte cada valor de entrada al rango de 0 a 1, estableciendo como 0, al valor más bajo del intervalo de entrada, y 1 al valor más alto de dicho intervalo. Con respecto a la interfaz de defuzzificación se ha seguido un modelo B-FITA.

Para llegar al objetivo de conseguir una gestión autónoma del tiempo de sueño del sensor se han desarrollado dos bases de conocimiento, cada una de las cuales establece

criterios ligeramente diferentes para afrontar el mismo problema. Se debe tener en cuenta que ambas bases de conocimiento parten de las mismas variables de entrada, y por supuesto, obtienen la misma variable de salida, estando su diferencia en las reglas borrosas empleadas. Los parámetros del sistema se describen en los apartados siguientes.

5.2.5.1 Variables de entrada

Las variables de entrada que se han usado para el FRBS, que controla de manera autónoma el tiempo de sueño del AG, son equivalentes a las que se usaron en el SD, convenientemente adaptadas a la lógica borrosa a través de la definición de los correspondientes conjuntos borrosos y rango de valores. Las tres variables de entrada son las siguientes:

- **Nivel absoluto medido de la magnitud bajo estudio (*presión sonora*):** La salida del FRBS dependerá del valor tomado del entorno. Valores demasiado altos o demasiado bajos se tratan como condiciones no deseables y deben ser revisados con más detalle que los valores leídos que caigan dentro del rango de condiciones normales de trabajo. Concretamente, esta variable, al igual que para el SD, es la *presión sonora* y se mide en dBA.
- **Las variaciones de entrada del sensor (*incremento*).** Esta variable mide la variación entre el nuevo valor medido por una sonda, y el valor medido en la ejecución anterior. El comportamiento esperado es que el intervalo de sueño tienda a aumentar si los valores leídos, con respecto a los anteriores, tienen pequeñas variaciones. Por el contrario, el tiempo de sueño tenderá a disminuir si las diferencias entre los valores históricos de la magnitud bajo estudio presentan incrementos grandes. Esta variable se denomina *incremento* y se mide en dB.
- **El nivel de la batería (*batería*).** Esta variable mide el nivel actual de carga de batería del sensor. El objeto de incorporar esta variable es que el comportamiento del sistema se ajuste en función de la carga disponible. Así pues, el intervalo de reposo calculado aumentará cuando el nivel de la batería sea bajo y prolongar así la vida útil del nodo, aunque el nodo sensor puede perder parte de la señal que se esté midiendo. Esta variable se denomina *batería* y se mide en porcentaje.

5.2.5.2 Variable de salida

La variable de salida FRBS es el tiempo de sueño para el próximo intervalo de reposo del sensor. Esta variable se denomina *SMI* y se mide en milisegundos.

5.2.5.3 Conjuntos borrosos

Las variables antes definidas han sido diseñadas, como ya se ha comentado, para el mismo tipo de aplicación que el SD, y así poder realizar una comparación lo más equitativa posible. Por lo tanto, los valores usados para los rangos de las magnitudes de las variables de entrada y salida, así como el diseño de los conjuntos borrosos siguen los mismos criterios que para el SD, con las evidentes salvedades intrínsecas a cada uno. La Figura 16 muestra los conjuntos borrosos correspondientes de las variables de entrada y de salida.

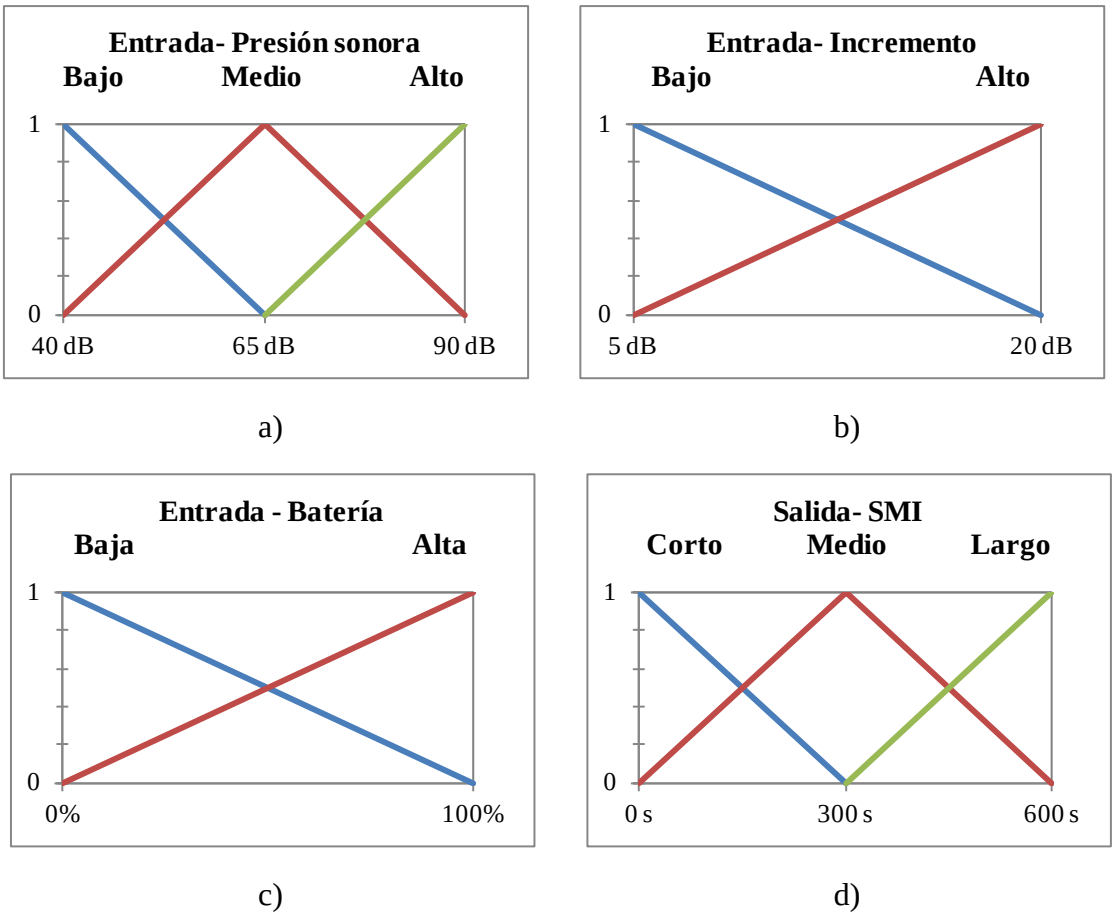


Figura 16. Los conjuntos borrosos para las tres variables de entrada y la variable de salida: a) Variable presión sonora, b) Variable de entrada de incremento, c) Variable de entrada batería y d) Variable de salida SMI.

Los conjuntos borrosos utilizados, y sus características especiales, son solo ejemplos para la aplicación de medición de presión sonora. Además, debido a la versatilidad de los FRBS, y del formato de definición de las bases de conocimiento visto (XML^{KB} y formato *raw*), estos se pueden configurar y adaptar fácilmente a cualquier cambio en la aplicación en curso o para otra solución de monitorización, con solo un mensaje del protocolo WISMAP.

En primer lugar, la variable que se obtiene de la magnitud bajo estudio, la presión sonora, está definida con tres conjuntos borrosos triangulares para modelar el umbral de silencio (BAJO, 40 dBA), la presión sonora de un régimen de trabajo normal (MEDIO, 65 dBA) y el umbral de ruido molesto y que indica una avería o un régimen de trabajo elevado (ALTO, 90 dBA).

Para la segunda variable, *incremento*, el rango elegido para la diferencia entre la magnitud actual y la pasada oscila entre 5 dB y 20 dB. El límite superior del intervalo es de 20 dB porque el sistema tendría como objetivo ser desplegado en entornos controlados, en los que los cambios instantáneos no son habituales. El nivel mínimo de 5 dB se emplea para evitar diferencias mínimas debido a fallas en la calibración del micrófono o ruido eléctrico. Así pues, se han definido dos conjuntos borrosos triangulares, uno denominado BAJO, que prima los valores cerca de 5 dB, y otro ALTO, para los valores de incrementos cerca de 20 dB.

La variable que mide la carga disponible en la batería se ha modelado con dos conjuntos borrosos triangulares, uno para una batería completamente cargada (ALTA, 100%) y el otro conjunto borroso para la batería agotada (BAJA, 0%). Han sido probados modelos de variables más complejas (con mayor número de conjuntos borrosos) y no presentan mejores resultados para el incremento de complejidad que suponen. Para mostrar el efecto de la batería, todas las pruebas, que serán detalladas en el Capítulo III, han sido realizadas para seis cargas diferentes de batería: 5%, 15%, 30%, 75%, 99% y 100%.

La variable de salida, SMI, ha sido diseñada con tres conjuntos borrosos triangulares, los cuales modelan la salida esperada para tiempos de reposo cortos (CORTO, alrededor de los 0 segundos), medios (MEDIO, 300 segundos) o largos(LARGO, 600 segundos). Con estos conjuntos borrosos, el motor de inferencia puede dar resultados que permitan un seguimiento prácticamente continuo de la variable, o reposos de casi diez minutos en el caso de que las condiciones permanezcan estables. Cabe destacar que estos valores podrían ser alterados dentro de un sensor en cualquier momento, necesitándose tan sólo un mensaje del protocolo WISMAP.

En las siguientes secciones se muestra cada una de las bases de conocimiento usadas en el presente trabajo de investigación. Se debe destacar que se ha llegado a estas dos bases de conocimiento después de analizar los resultados obtenidos para el SD; y tras un proceso de estudio y ajuste de las propuestas iniciales de base de conocimiento, se llegó a las siguientes conclusiones:

- Se detectó que no era necesario incrementar la complejidad del sistema con nuevas reglas para contemplar todas las combinaciones de los conjuntos borrosos de entrada.
- La definición de variables con mayor número de conjuntos borrosos para la variable de salida no proporcionó mejoras en el comportamiento, por lo que se mantuvieron los tres que más tarde se mostrarán.

5.2.6 BASE DE CONOCIMIENTO 1 (KB1)

La primera base de conocimiento se ha diseñado teniendo en cuenta las siguientes consideraciones:

- La magnitud a modelar tiene un comportamiento inercial, como es en este la presión sonora.
- Un uso conservador de la batería, pero dando un mayor peso a la precisión (menores tiempos de reposo).

Así pues, según se puede ver en la base de reglas para la base de conocimiento 1 (Tabla 11), se ha establecido que los mayores tiempos de reposo se darán cuando haya una mayor estabilidad en las muestras de entrada (reglas 1, 2 y 3), aunque cada una obedece a unas motivaciones ligeramente diferentes. Cada una de las reglas pasará a ser comentada a continuación:

- Regla 1: es independiente de la presión sonora que se esté registrando, ya que modela los casos de poca variabilidad a la entrada y poca carga de batería, donde se busca un mayor tiempo de reposo.
- Regla 2: intenta alargar el tiempo de vida del sensor no dando prioridad a valores bajos de la magnitud bajo estudio con poca variabilidad, ya que estos valores modelan bajadas en el régimen de funcionamiento de la maquinaria controlada, si bien eso es un evento a controlar, no supone un evento crítico.

- Regla 3: busca una mayor precisión ya que los valores de presión sonora están dentro del rango de funcionamiento deseable y la carga de la batería aun es alta.
- Regla 4: contempla el caso de tener una carga de batería alta y que estar registrando valores altos presión sonora. Así pues, a pesar de que los incrementos sean bajos, se reducen los tiempos inferidos para el SMI dado que interesa tener una mayor precisión en las medidas en esa situación.
- Reglas 5 y 6: contemplan los casos en los que la variable incremento arroje valores altos, y dependiendo de la carga de la batería, se dará un resultado más conservador (regla 5) o que busque una mayor precisión (regla 6). En ambos casos no importa el nivel de presión sonora concreto, ya que las altas fluctuaciones deben ser tenidas siempre en cuenta.

Tabla 12. Base de reglas para KB1.

Regla	Incremento	Batería	Presión sonora	SMI
1	Bajo	Baja		Largo
2	Bajo	Alta	Baja	Largo
3	Bajo	Alta	Media	Medio
4	Bajo	Alta	Alta	Corto
5	Alto	Baja		Medio
6	Alto	Alta		Corto

5.2.7 BASE DE CONOCIMIENTO 2 (KB2)

La base de conocimiento 2 (KB2) se ha diseñado para tener un comportamiento más conservador con respecto al uso de batería. Esto se ha establecido al configurar la variable de salida para que se infieran tiempos más largos para mayor número de reglas. Como se puede comprobar en la base de reglas (Tabla 12), los antecedentes de todas las reglas son los mismos que para la KB1, cambiando tan solo los consecuentes de las reglas 2 a la 5. Como se podrá ver en el Capítulo III, los resultados obtenidos de la aplicación de esta base de conocimiento son sensiblemente diferentes a la KB1, lo cual demuestra que con leves ajustes en una base de conocimiento, se pueden modelar sistemas y comportamientos diferentes. En este caso, tan solo sería necesario modificar 16[65] bytes en el formato *raw*. Así pues, con un ajuste menor, cada sensor podría ir cambiando de comportamiento en función de la aplicación, o de las órdenes de un administrador, con un gasto mínimo de recursos de comunicación.

Al igual que en la base de conocimiento KB1, las reglas 1 y 6 se basan en los mismos criterios y no se han modificado en KB2. Las modificaciones con respecto a KB1 son las siguientes:

- Regla 2: se añade precisión a las medidas tomadas para valores bajos de la presión sonora mientras la batería esté alta. Esto se hace para uniformar los criterios de precisión en ambos extremos del rango de valores de la magnitud bajo estudio.
- Regla 3: se aumentan los tiempos de muestreo para valores de presión sonora que estén dentro del rango de funcionamiento deseable. Así, se espera aumentar la duración del batería, dado que, normalmente, el sistema monitorizado arrojará medidas dentro de este margen.

[65] Este valor proviene de multiplicar 4 identificadores de conjunto borroso, por cuatro bytes que ocupa cada uno.

- Regla 4: se prima el hecho de que los incrementos sean bajos, para asumir que bastará con la inferencia de tiempos a medio plazo para obtener un equilibrio entre precisión y gasto de batería.
- Regla 5: la modificación de esta regla se debe por la priorización en ahorro de batería que marca esta base de conocimiento, por lo que, a pesar de estar registrando altas variaciones en la magnitud de entradas, al estar la batería en niveles bajos, se inferirán tiempos altos para la variable SMI.

Tabla 13. Base de reglas para KB2.

Regla	Incremento	Batería	Presión sonora	SMI
1	Bajo	Baja		Largo
2	**Bajo**	**Alta**	**Baja**	**Medio**
3	**Bajo**	**Alta**	**Media**	**Largo**
4	**Bajo**	**Alta**	**Alta**	**Medio**
5	**Alto**	**Baja**		**Largo**
6	Alto	Alta		Corto

Como se ha podido apreciar, existen unas claras diferencias en el diseño de ambas bases de conocimiento, y como podrá ser comprobado al compararlas entre sí, y al sistema diferencial, los resultados son apreciablemente diferentes. La comparación de los resultados, junto con el análisis estadístico de los mismos, podrá verse con detalle en el Capítulo III.

Capítulo III. RESULTADOS

En el presente capítulo se pasará a detallar tanto el diseño, implementación y resultados obtenidos de los sistemas y pruebas realizadas para la consecución y contrastación de las hipótesis y objetivos del presente trabajo de investigación.

En primer lugar se presentará el banco de pruebas creado (test-bed) el cual está conformado tanto por equipos informáticos como por sensores inalámbricos. A continuación, una vez detallada la plataforma usada para las posteriores pruebas, se mostrará cómo se ha diseñado e implementado la arquitectura multi-agente dentro de un sensor, así como las pruebas de rendimiento y estabilidad realizadas para la misma.

En tercer lugar se detallará cómo se ha desplegado el protocolo WISMAP dentro de los sensores inalámbricos y las pruebas realizadas para la comprobación de su funcionamiento.

En último lugar se presentarán las pruebas realizadas a los sistemas de estimación autónoma del tiempo de sueño, tanto el Sistema Diferencial como el FRBS. Ambos serán comparados entre sí y con muestreos fijos, a través de la simulación de su funcionamiento usando señales generadas de manera pseudo-aleatoria.

6 BANCO DE PRUEBAS (TEST-BED)

Como se ha podido ver en la revisión de conocimientos presentada en el Capítulo I, y más concretamente en su apartado 2.1.8, los bancos de pruebas, en inglés *test-beds*, son una herramienta típica para probar el funcionamiento de tecnologías, técnicas y protocolos de redes de sensores, así como un fin en sí mismo como complementos educativos o plataformas de promoción comercial.

Dada la amplia variedad de características, que como se vio, puede tener un test-bed (véase la Tabla 3), a continuación se comentan aquellas propiedades que se han considerado prioritarias para el diseño e implementación del test-bed creado como parte del presente trabajo de investigación.

- Que la tecnología a usar fuera accesible, para lo que se tuvieron en cuenta los siguientes factores:
    - Accesibilidad a la financiación: se consideró que era prioritario que la inversión en los equipos que iban a formar parte del test-bed fuera asumible por parte del grupo de investigación.
    - Disponibilidad de información y material de soporte.
    - Familiarización con la tecnología.
- Acceso desde Internet. Este punto se consideró importante dado que es una de las características más comunes en todos los test-bed. Además, proporciona la capacidad de ser gestionado y controlado desde cualquier equipo con conexión a Internet, lo cual aporta versatilidad a la solución final.
- Escalabilidad: se debería tener presente como una prioridad básica, que el test-bed debería soportar un número variable de nodos por cada red de sensores, así

como múltiples ubicaciones físicas de estas redes, las cuales podrían ser accedidas usando las mismas herramientas y métodos.

- Tamaño: por motivos de espacio y de disponibilidad de dependencias, el test-bed no debería requerir para su despliegue mucho más que unos pocos metros cuadrados. A pesar de esto, su diseño no ha quedado limitado por tamaño, si bien en su concepción inicial se tendría esta prioridad más en cuenta.

Así pues, estas características han sido tenidas en cuenta para todos los desarrollos realizados como parte de la presente tesis, dando lugar a una arquitectura escalable y replicable de test-bed para redes de sensores con tecnología Sun SPOT.

En el siguiente apartado aparece la justificación de la tecnología usada, para mostrar a continuación la arquitectura del test-bed. Después de ver la arquitectura se detallará el software de soporte para la red de sensores, tanto del acceso al test-bed a través de Internet, dela estación base, seguido del apartado donde se describe la arquitectura interna de un sensor. Seguidamente se mostrarán las pruebas realizadas al test-bed, para finalizar con un breve comentario sobre la instalación de una red de sensores dentro de la Escuela Politécnica Superior de Linares.

6.1 JUSTIFICACIÓN DE LA TECNOLOGÍA

Como ya se ha comentado en la sección 2.1.5, era necesario estudiar previamente qué tecnologías de red de sensores usar para la implementación del test-bed, y así poder tomar una decisión acerca de cuál sería la más apropiada para su uso en la instalación. Este análisis debería conducir a la implantación de una tecnología que permitiera la realización de las pruebas que avalaran las contribuciones del presente trabajo de investigación, más aun, cuando el desarrollo e implementación de dicho test-bed es parte de los objetivos de esta tesis. Las consideraciones y decisiones tomadas se detallan a continuación.

En primer lugar se descartó el hecho de usar varias plataformas, dado que las hipótesis de la tesis no implicaban el uso diferenciado de plataformas, y ello podría conllevar problemas como la incompatibilidad de hardware y software.

Así pues, se evaluaron tres opciones: sensores tipo Sun SPOT, MicaZ e Iris, el primero basado en Java (SO y programación); y los otros dos con sistema operativo TinyOS y programados en NesC. El análisis se limitó a estos tres tipos dado que eran los tres tipos que estaban disponibles dentro del grupo de investigación durante las primeras fases del trabajo, además de ser algunos de los principales tipos de tecnologías usadas dentro de las redes de sensores.

Después de la evaluación del software de gestión de todos ellos, pruebas de carga de programas y familiarización con el entorno, se optó por la plataforma Sun SPOT.

En primer lugar, un aspecto que determinó esta elección fue el software de gestión disponible para carga y modificación de programas. En concreto, tanto para los MicaZ, como para los Iris, no estaba adecuadamente adaptado a las nuevas versiones de sistemas operativos (por encima del sistema operativo Windows XP). Incluso para aquellas versiones de sistema operativo, presupuestas compatibles, no fue posible conseguir una funcionalidad completa. Sin embargo, la plataforma Sun SPOT, si bien también presentaba complicaciones en su fase inicial de instalación, al trabajar sobre una máquina virtual Java, mantenía una alta compatibilidad en sistemas operativos más actuales.

Aparte de este hecho, importante pero no crucial, se tuvieron en cuenta otras consideraciones, las cuales, tras su evaluación, fueron determinantes en la elección de la plataforma Sun SPOT como soporte para las pruebas de los resultados de investigación de la presente tesis. A continuación se detallan los aspectos evaluados:

a) Familiarización: dado que esta plataforma ya había sido usada para los trabajos iniciales dentro de la fase de estudio, se disponía de abundante información, así como de experiencia con ella. A este hecho se unió la amplia familiarización con la programación en Java SE y Java ME (la cual, con ligeras modificaciones es usada por estos sensores). Además, otros miembros del grupo de investigación también tenían experiencia con este tipo de sensores. Este conjunto de factores propició lo siguiente:
    - El tiempo de aprendizaje fue inferior al inicialmente estimado al abordar una nueva tecnología.
    - Disponibilidad de información de uso y ejemplos prácticos.
    - Conocimiento sobre características y problemas propios del desarrollo.
b) Disponibilidad: dado que ya se contaba con un número suficiente de sensores Sun SPOT dentro del grupo de investigación, se facilitaba la realización de pruebas, además de no necesitar nuevas inversiones.
c) Prestaciones: El hecho de disponer de un microprocesador ARM920T de 180 Mhz y 32-bit permite la ejecución mucho más rápida de aplicaciones más complejas que las disponibles en sus competidores. Además, al tener 512Kbytes de RAM (tan sólo superado por el Imote IV de los vistos en el apartado 2.1.5 y que no se tenía disponible), las aplicaciones pueden hacer un uso más intenso de la memoria, abriendo el abanico de posibilidades para estos sensores.
d) Versatilidad: no sólo por su hardware, sino por su sistema operativo y API de programación. Cada sensor incorpora sondas de temperatura e iluminación, acelerómetro de tres ejes, un conversor A/D e interfaces básicos como dos micro-interruptores y ocho diodos led tricolor. Con respecto al API de programación, ésta ha sufrido una evolución que ha propiciado la eliminación de fallos y la inclusión de mejoras que incorporan nuevas funciones a los sensores, como el control mejorado de eventos provocados por las sondas, o la planificación de tareas.
e) Portabilidad de los desarrollos: al usarse para su programación el lenguaje Java, se puede importar y exportar código utilizado en otras plataformas sin modificaciones de consideración (incluso sin ninguna). De esta manera se agiliza el desarrollo y se previenen posibles errores en la adaptación.

Como pudo observarse en la sección 2.1.5, existen soluciones comerciales e industriales que podrían también haber servido como base para la investigación. Sin embargo, estos productos, a pesar de ser estándares abiertos, no son los más idóneos para realizar una investigación científica, ya que suelen estar muy enfocados a una función concreta y no aportan una interfaz de desarrollo como las otras plataformas comentadas en la misma sección. Además, la incorporación de mejoras, como las propuestas en el presente trabajo de investigación, implican una serie de modificaciones que no serían incorporadas a un sistema comercial de una manera directa y, a pesar de que uno de los objetivos de esta tesis sea la prueba de todos los resultados en sistemas reales, la generación de un producto comercial dista mucho de los objetivos del presente trabajo.

Así pues, dado que los resultados que se persiguen dentro de la presente tesis son de ámbito general, y no tienen una vinculación explícita con el hardware o dispositivos

usados, se vio conveniente el empleo de los sensores Sun SPOT, por todos los motivos expuestos. Más aun, existen estudios realizados por otros autores que respaldan la decisión tomada, como por ejemplo la revisión de Johnson y otros (Johnson et al. 2009) o los comentarios favorables en (Potdar et al. 2009).

Además, y como consecuencia de la elección de la tecnología Sun SPOT, el lenguaje de programación usado ha sido Java, el cual coincide con el espíritu del trabajo realizado, buscando la mayor independencia posible de la arquitectura hardware subyacente. Así pues, todos los desarrollos software de la presente tesis podrían ser incorporados a otras plataformas, ya sea de redes de sensores u ordenadores, con relativa facilidad. Este hecho ha sido comprobado, por ejemplo, con las clases de soporte a los FRBS y el motor de inferencia usado en la presente tesis, las cuales han sido empleadas con éxito, tanto en un sensor como en un PC, sin ningún cambio en el código. Como se ha visto en la sección 5.2, esto ha sido posible gracias al uso del lenguaje Java, así como a la definición de los formatos de bases de conocimiento XML^{KB} y *raw*.

Por lo tanto, tomando como base la tecnología Sun SPOT, se ha implementado un test-bed, en primer lugar, como requisito para la cumplimentación de uno de los objetivos de la presente tesis, y en segundo lugar, como elemento necesario para la realización de las pruebas que respalden los resultados de la misma. La arquitectura de dicho test-bed puede verse en la siguiente sección.

6.2 ARQUITECTURA DEL TEST-BED

El test-bed desarrollado como parte de los trabajos de investigación de la presente tesis está configurado, tanto por elementos hardware, como por el software de soporte que asocia cada uno de los dispositivos que lo forman.

La principal característica del diseño realizado es la modularidad. Gracias a esta estructura, todos los test-bed que sigan la arquitectura aquí presentada, podrán ser accedidos y gestionados de igual forma gracias al protocolo WISMAP y a la pasarela para Internet WISMAPi. De esta manera, se consigue que la implementación de test-beds sea totalmente escalable, además de facilitar una gestión distribuida o centralizada.

Cada test-bed está formado por un equipo PC, un estación base, y un conjunto de sensores Sun SPOT que forman la red de sensores. Esta arquitectura puede verse en el esquema presentado en la Figura 17 A continuación se pasará a describir las características más destacables de cada uno de estos elementos.

El equipo que sirve de base para el acceso y gestión de la red de sensores es un PC con el sistema operativo Windows XP para 32 bits con el *ServicePack*3[66], no necesitando prestaciones adicionales a las requeridas por este sistema operativo. Otra ventaja de la tecnología usada es que, dado que existen controladores para los dispositivos Sun SPOT para la mayoría de sistemas operativos generalistas, y todo el software se ha desarrollado en Java, esta máquina podría tener instalado otros sistemas operativos de la familia Windows, Linux o MacOS. A este equipo se le denominará servidor del test-bed o servidor WISMAP (abreviado por Ws).

[66]Concretamente el equipo es un PC con un microprocesador Intel Core 2 Quad Q6600 a 2,4 GHz y 3 GBytes de RAM.

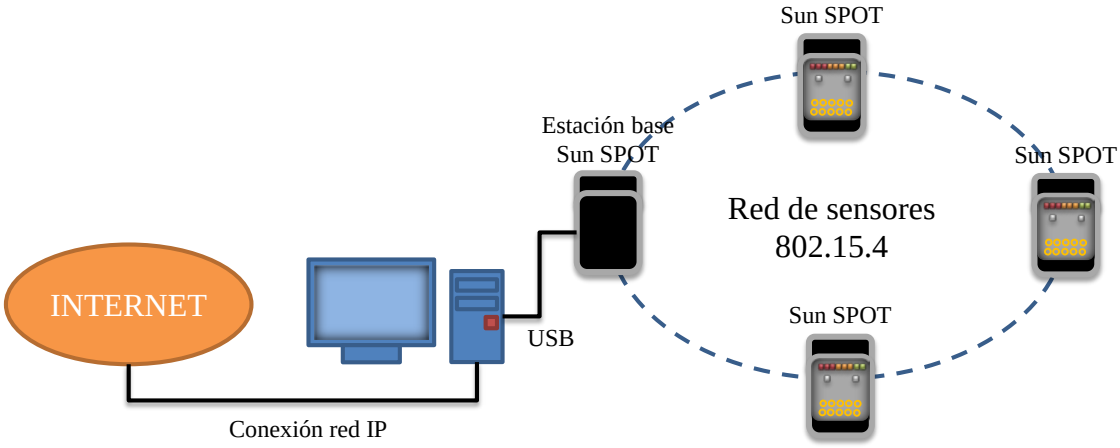


Figura 17. Esquema del test-bed desarrollado.

El servidor del test-bed tendrá conectada en todo momento la estación base Sun SPOT, la cual hace de interfaz entre la red de sensores y el Ws. Una característica adicional, a las ya comentadas de la tecnología Sun SPOT, es que la estación base posibilita la implementación de aplicaciones híbridas, las cuales ejecutan parte de su código en el PC y parte en la estación base[67]. De esta manera se consigue compatibilizar toda la potencia de proceso del equipo anfitrión PC, con las capacidades de comunicación de la estación base. Otra capacidad que deja clara la versatilidad de la tecnología Sun SPOT es que una estación base puede ser configurada como una estación compartida y así ser usada por varias aplicaciones.

Por último, los restantes elementos que conforman el test-bed son los nodos o sensores desplegados, que como ya se ha comentado, son dispositivos Sun SPOT[68]. El número de estos podrá elegirse en función de la aplicación, o aplicaciones, a la que se quiera destinar el test-bed. Se debe recordar que, por los criterios de diseño del protocolo WISMAP, el número máximo de sensores accesibles a través de su dirección de aplicación es de 2^{32}(ver sección 4.4.3), lo cual permite una asignación jerárquica de direcciones, si bien no se ha definido ninguna en concreto, dejando abierto este aspecto a criterios de la aplicación.

Con respecto a la arquitectura software del test-bed (la cual puede verse en la Figura 18), como ya se ha comentado, en el servidor hay una aplicación que permite el acceso a los recursos del PC, así como a la red de sensores a través de la estación base. Este equipo también es el punto de acceso al test-bed desde Internet. A esta aplicación se la ha denominado *wismaphost*. Por otro lado, cada sensor incorpora una aplicación denominada *wismapsensor*, la cual aúna toda la funcionalidad descrita para un sensor: arquitectura multi-agente, soporte al protocolo WISMAP, ejecución de FRBS y soporte para la gestión autónoma del tiempo de sueño dentro del Agente de Gestión, ya sea con el Sistema Diferencial o con FRBS. Cada una de ellas pasará a ser comentada brevemente a continuación.

[67]Estas aplicaciones se denominan Sun SPOT *host applications*.
[68]Durante las pruebas de la presente tesis se han usado sensores Sun SPOT con la versión 6 de hardware y la versión de la SDK red-100104 (más información en www.sunspotworld.com).

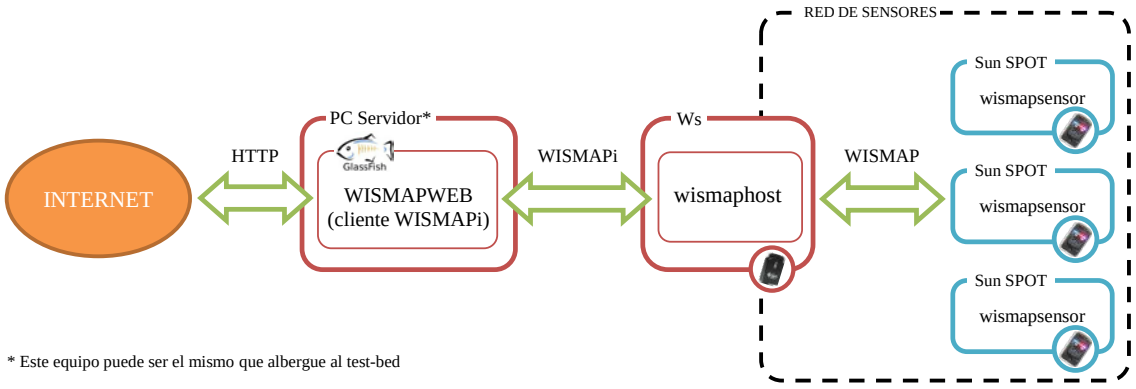


Figura 18. Arquitectura software del test-bed.

6.2.1 APLICACIÓN WEB WISMAPWEB.

En primer lugar, la explicación se centrará en la aplicación web WISMAPWEB, la cual puede definirse como una pasarela HTTP/WISMAPi desarrollada con objeto de poder probar el protocolo WISMAPi, y por extensión, la aplicación *wismaphost* en su función de pasarela Internet/Red de sensores. El motivo de hacer este tipo de pasarela, en vez de una aplicación cliente al uso, estuvo motivada por la versatilidad y facilidad de acceso que aporta al ser una aplicación web. Así pues, el usuario final podría interactuar con ella con cualquier navegador (Internet Explorer, Firefox, Chrome, Safari, Opera, etc.) y sobre cualquier sistema operativo (Windows, Linux, Mac OS, etc.) y/o dispositivo (ordenador, teléfono móvil, etc.). Así pues, esta pasarela HTTP/WISMAPi se ha desarrollado como una aplicación de servidor denominada *WISMAPWEB*. Está basada en tecnología de servidor Java (JSP[69] + Servlets[70]) sobre el servidor GlassFish de Sun[71].

La función principal de la aplicación *WISMAPWEB* es la de proporcionar una interfaz que posibilite la creación de peticiones WISMAPi a través de HTTP. Para esto cuenta con una sencilla interfaz web basada en un formulario, a través del cual se pueden definir toda la información necesaria para realizar una petición completa del protocolo WISMAPi a un test-bed concreto. Los elementos que se pueden configurar para llevar a cabo estas peticiones son los siguientes:

- Dirección IP y puerto TCP del test-bed destino que disponga de la aplicación *wismaphost*, u otra cualquiera que siga el protocolo WISMAPi.
- Numero de intentos de la petición y tiempo de espera entre intentos.
- Método del protocolo WISMAPi a usar, *GET* o *POST*.
- Cadena de recursos que identifica el elemento destinatario de la petición dentro de la jerarquía de recursos.
- Parámetros de la petición para los métodos *GET*.
- Contenido para el cuerpo de las peticiones *POST*.

[69]Siglas de Java *Server Pages*.
[70]Clases de Java que son ejecutadas por el servidor de aplicaciones y que interactúan con el cliente a través de los parámetros de las peticiones HTTP.
[71]Servidor de aplicaciones similar a otros como Apache Tomcat o Microsoft IIS. Ahora se distribuye bajo licencia de código libre y bajo licencia Oracle, ya que Sun Microsystems fue adquirida definitivamente por ésta en 2010.

Como ya se vio en la sección 4.2, la jerarquía de recursos diseñada en la presente tesis posibilita, de una manera clara y flexible, el acceso a toda la información disponible dentro del test-bed y de la red de sensores.

Tras la elección de la acción deseada se construye una petición WISMAPi que es enviada a la pasarela *wismaphost* designada por su IP y puerto. Así pues, la aplicación *WISMAPWEB* se comporta como un cliente WISMAPi gracias a la tecnología servlet de Java. Las respuestas a cada petición son generadas por la aplicación *wismaphost* y recibidas por *WISMAPWEB* para ser mostradas al usuario a través de la interfaz web.

La arquitectura de la aplicación *WISMAPWEB* es relativamente sencilla, a través de la interfaz web generada por el recurso *testbed.jsp* que presenta el formulario de petición, se invoca la ejecución de los métodos de la clase WISMAPquerier que realiza una conexión TCP, envía la petición y recibe la contestación de la pasarela *wismaphost*.

Se debe mencionar que se han desarrollado dos trabajos derivados de la aplicación *WISMAPWEB* a través de sendos proyectos fin de carrera. El primero de ellos emulaba a la aplicación *WISMAPWEB* usando la tecnología de servidor PHP[72], ampliando su funcionalidad a través del uso de una interfaz de usuario dinámica basada en Java Script y almacenamiento en bases de datos MySQL. En el segundo proyecto, dirigido por el autor de esta tesis, se ha implementado una aplicación para terminales telefónicos móviles con el sistema operativo Android[73]. Dicha aplicación permite, a través de una interfaz gráfica, realizar peticiones a un test-bed a través del protocolo WISMAPi, además de registrar los valores tomados y mostrarlos gráficamente.

6.2.2 APLICACIÓN WISMAPHOST

Como ya se ha adelantado brevemente, la aplicación que proporciona toda la funcionalidad al Ws está implementada en Java SE y aúna en un solo programa el acceso a los recursos del PC y a las capacidades de comunicación de la estación base Sun SPOT. Así pues, y en su modo de funcionamiento más completo, se podría definir esta aplicación como una pasarela entre Internet y una red de sensores con tecnología Sun SPOT. Esta pasarela funciona a nivel de aplicación, ya que no se hace una traducción de direcciones de red, ni de transporte, sino que, apoyándose en el protocolo WISMAP, su vertiente para redes TCP/IP, WISMAPi, y la estructura de acceso a recursos, permite el acceso a la información disponible en un sensor, la cual ya quedó definida en la sección 4.2.

Sin embargo, la función de pasarela no es la única, y en concreto, puede no estar activada si no fuera necesaria, dejando al test-bed aislado, para realizar, por ejemplo, tareas de mantenimiento o por seguridad. De este modo, trabajaría tan solamente como servidor para la red de sensores, gestionando la actividad de la red. Además, a través de su interfaz gráfica, un usuario podría hacer peticiones directas a los sensores, tales como leer valores de las sondas, o cargar bases de conocimiento de manera remota. Se debe destacar que la aplicación no se ha desarrollado con el objetivo de proporcionar una interfaz generalista y completamente abierta, sino que se ha centrado en las funciones de

[72]Proyecto fin de carrera realizado por Pedro Antonio González Laínez dentro de la titulación Ingeniería Técnica de Telecomunicación y dirigido por el doctor D. José Ángel Fernández Prieto, con el que el autor de la presente tesis colaboró estrechamente en la definición de las clases que daban soporte al protocolo WISMAPi.

[73]Proyecto fin de carrera realizado por Álvaro Quirós Palacios para la Ingeniería de Telecomunicación.

comunicación e interconexión, añadiéndole los elementos indispensables para permitir la prueba de los diferentes sensores en la red de una manera rápida e intuitiva.

Con respecto al soporte para la ejecución de FRBS en los sensores, se ha implementado la lectura y envío de cualquier base de conocimiento, así como la posibilidad de guardar estas en formato *raw* en disco.

Sin entrar en los detalles concretos de la implementación y de cada una de las clases que forman la aplicación *wismaphost*, en la Figura 19 se muestra un esquema con los paquetes y las clases más importantes de cada uno de estos.

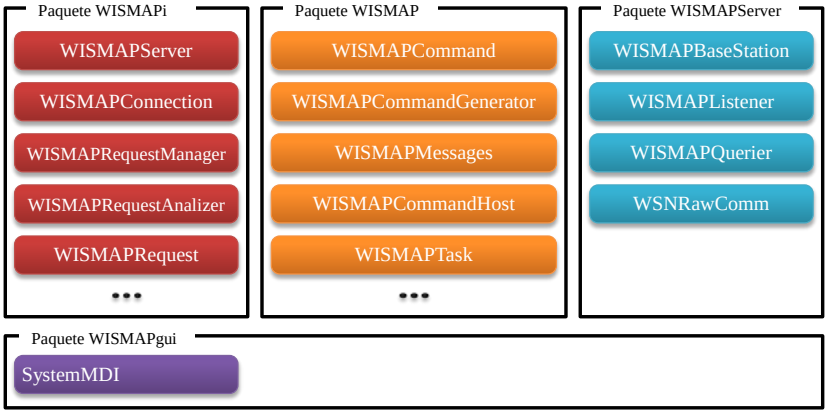


Figura 19. Esquema simplificado de clases.

Adicionalmente, en la Figura 20 aparece una captura de pantalla del interfaz de usuario de la aplicación *wismaphost*. Como se puede apreciar, esta interfaz permite fundamentalmente comprobar los mensajes que se intercambian los sensores y la estación base, así como los que son enviados a través de la pasarela WISMAPi. También permite llevar un registro de la última actividad registrada para cada sensor de la red.

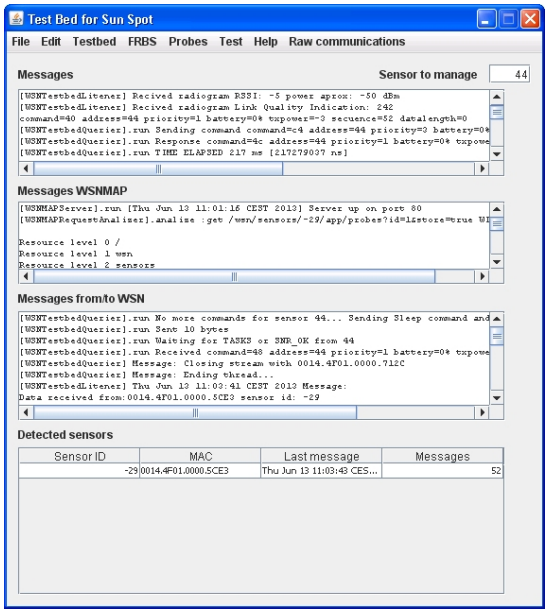


Figura 20. Interfaz de usuario de la aplicación wismaphost.

Para ilustrar el funcionamiento típico de esta aplicación cuando se comporta como pasarela entre Internet y la red de sensores se presenta el diagrama de la Figura 21. En él aparecen las clases más significativas involucradas en cada proceso, así como la secuencia de mensajes y operaciones típicas para tomar una medida de un sensor que forma parte de la red de sensores.

El escenario mostrado en la Figura 21 corresponde con una petición de una medida de temperatura (app/probes?id=1) al sensor 44 (wsn/sensors/44) del test-bed a través del protocolo WISMAPi. Esto provoca que la petición de medida sea almacenada en el Ws para que cuando el sensor 44 despierte se le requiera esa información. Así pues, tras despertar, el sensor y notificarlo a la estación base (SNR_AWAKE), se establece una conexión a nivel de transporte que inicia el proceso de petición de información. Puesto que existe una petición pendiente, se envía al sensor el mensaje GET_VALUE en el que se identifica la sonda de temperatura (tipo 1). Al recibir este mensaje el sensor toma el valor de la sonda de temperatura y envía de vuelta un mensaje SNR_VALUE con la medida requerida. Como el Ws no tiene pendiente más peticiones para el sensor, le envía el comando BST_SLEEP para que entre en reposo y espera para cerrar la conexión a la recepción del mensaje SNR_OK.

Este escenario se completaría con la posterior petición al test-bed, para la lectura del registro por parte de un cliente WISMAPi (como podría ser la aplicación WISMAPWEB) al recurso wsn/log, la cual puede ser la siguiente: get /wsn/log/44/probes?id=1 WISMAP/0.1, y cuya respuesta incluiría información detallada sobre la medida tomada en formato HTML.

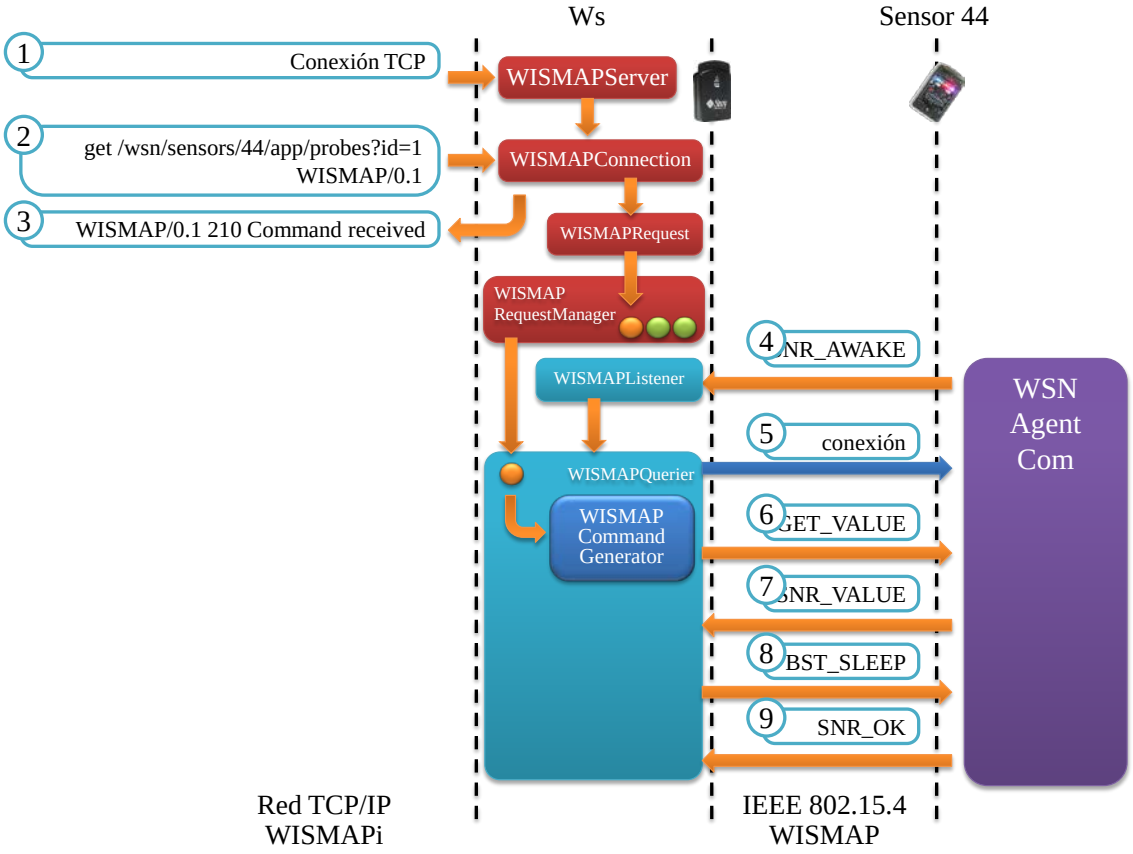


Figura 21. Secuencia de petición de una medida a un sensor.

6.2.3 APLICACIÓN WISMAPSENSOR

La aplicación *wismapsensor* proporciona a un sensor Sun SPOT una arquitectura multi-agente, la capacidad de comunicarse usando el protocolo WISMAP, el soporte para la ejecución de FRBS y la gestión autónoma del tiempo de ciclo, ya sea mediante un sistema diferencial, o basado en lógica difusa. Estas características ya fueron descritas en el Capítulo II, por lo que la explicación se centrará en ubicar cada una de capacidades dentro de la aplicación.

Al igual que la aplicación *wismaphost*, *wismapsensor* ha sido implementada usando el lenguaje Java. En concreto, se ha desarrollado con la variante proporcionada por el fabricante para los sensores Sun SPOT, la cual está derivada de la que se usaba para la programación de terminales móviles, denominada Java ME[74]. La aplicación corre sobre una máquina virtual Java que es a su vez la base del sistema operativo, denominado Squawk (Oracle Labs).

Así pues, la aplicación *wismapsensor* es una parte fundamental de los resultados de la presente tesis, al implementar de forma real y funcional las tres hipótesis de la misma. De esta manera se ha posibilitado la consecución de otros objetivos, así como la realización de pruebas basadas en el despliegue de dispositivos reales.

Las características principales de la aplicación *wismapsensor* son las siguientes:

- Aplicación principal que se encarga de inicializar los diferentes agentes.
- Arquitectura basada en agentes. Cada agente se modela con una clase, la cual tiene una hebra de ejecución, que se encarga del ciclo de vida el agente.
- Clases de abstracción de los recursos físicos del sensor. Se ha diseñado una serie de clases para que el acceso a los recursos dependientes del hardware, como las sondas, para que éstas sean accedidas a través de una interfaz común que aísle de las características peculiares de cada una, dando una respuesta uniforme.
- Clases de soporte al protocolo WISMAP.
- Soporte para la ejecución de FRBS usando las clases de abstracción del hardware.
- Cálculo automático del tiempo de sueño usando el Sistema Diferencial o FRBS.

6.3 PRUEBAS DEL TEST-BED

Como ya se ha comentado, el propio diseño y creación del test-bed tiene un doble cometido: en primer lugar cumplir con uno de los objetivos de la presente tesis, y por otro lado, servir como base para la realización de pruebas y la obtención de resultados que conduzcan a refrendar las hipótesis del presente trabajo.

Dado que el test-bed se compone de diversos elementos hardware y software, las pruebas que se han ido realizando a lo largo del periodo de desarrollo, así como tras la obtención de la versión definitiva, han intentado siempre involucrar al mayor número posible de elementos y de procesos de intercambio de información.

Por lo tanto, las pruebas efectuadas al test-bed se han basado en la realización de peticiones con el protocolo WISMAPi a través de la aplicación web *WISMAPWEB*.

[74]Java ME proviene de Java *Micro Edition*, la cual tuvo su auge a mediados de la década pasada al posibilitar la creación de aplicaciones portables de todo tipo (fundamentalmente videojuegos). Estas aplicaciones estaban destinadas a teléfonos móviles capaces de correr una máquina virtual Java optimizada para dispositivos con prestaciones reducidas, denominada KVM (Kilobyte Virtual Machine).

Estas peticiones serán atendidas por la aplicación *wismaphost*, la cual las procesará, generando, la respuesta a la petición WISMAPi, y si fuera necesario, los mensajes del protocolo WISMAP que deban enviarse al sensor o sensores destinatarios. Así pues, con este flujo de información y las pruebas que se han preparado a tal efecto, se consigue probar todas las entidades de comunicación y proceso que forman parte del test-bed. Estas son:

- Cliente WISMAPi, implementado por la aplicación *WISMAPWEB*.
- Servidor WISMAPi, implementado en la aplicación *wismaphost*.
- Servidor WISMAP, implementado en la aplicación *wismaphost*.
- Agente de Comunicaciones del sensor implementado en la aplicación *wismapsensor*, que procesará las peticiones recibidas con el protocolo WISMAP.
- Agente de Control de Aplicación del sensor, implementado en la aplicación *wismapsensor*, encargado de tomar las medidas de las sondas.
- Agente de Gestión del sensor, implementado en la aplicación *wismapsensor* y que recibirá peticiones de cambio del tiempo de sueño.

Cabe destacar que cada prueba no solo comprueba el buen funcionamiento del software, sino que también verifica el de los equipos que integran el test-bed.

En los tres apartados siguientes se detallan cada una de las pruebas efectuadas al test-bed. Cada una intenta mostrar un escenario de uso común para el test-bed, lo cual ha derivado en la realización de tres tipos de pruebas diferentes. Después de mostrará el escenario donde se han realizado las pruebas, así como un resumen de los resultados y las conclusiones de los mismos.

6.3.1 PRUEBA 1 - LECTURA DEL VALOR DE UNA SONDA.

Esta prueba es la más completa que se puede realizar al test-bed ya que se comprueba todo el proceso de comunicaciones presentado en el Capítulo II. De esta forma, a partir de una petición HTTP a la aplicación *WISMAPWEB*, se genera un mensaje WISMAPi (*GETwsn/sensors/id/app/probes*) que llega al test-bed, a la aplicación *wismaphost*. Ésta registra la petición, y cuando detecte que el sensor destinatario está despierto (recepción del mensaje *SNR_AWAKE*), le enviará, usando el protocolo WISMAP, un mensaje *GET_VALUE* esperando hasta recibir contestación. El sensor, el cual ejecuta la aplicación *wismapsensor*, al recibir este mensaje y procesarlo, tomará una medida de la sonda especificada y montará un mensaje *SNR_VALUE* de respuesta. La medida retornada se almacenará por la aplicación *wismaphost* a la espera de una petición de lectura (prueba 2). Como se puede apreciar, el proceso involucra a todos los sistemas y protocolos desarrollados en la presente tesis.

- Condiciones de éxito: obtención de una medida por parte de la aplicación *wismaphost* y que sea grabada para que así quede disponible para su posterior revisión.
- Posibles fallos: fundamentalmente la probabilidad más alta de fallo se encuentra en las comunicaciones radio entre la estación base y el sensor, pero siempre existirá una mínima posibilidad en las comunicaciones por Internet, al igual que en la ejecución del software, ya sea en el PC o en un sensor. Se debe añadir que la probabilidad de fallo en las comunicaciones aumentará conforme el número de nodos entre la estación base y el sensor destino crezca. Sin embargo, dado que en ese caso los mecanismos que

entran en juego (protocolo de encaminamiento, interferencias, etc.), no son los que se han desarrollado para esta tesis, no habrá nodos intermedios entre la estación base y el destinatario.

- Protocolo de pruebas: realización automática de *n* peticiones, espaciadas un tiempo *t*, a través de la aplicación *WISMAPWEB* al test-bed para solicitar una medida de un sensor. La sonda requerida será la temperatura. El número de intentos y la separación entre peticiones serán variables para modelar diferentes situaciones. Se medirá el número de peticiones no completadas con respecto al total de peticiones.

6.3.2 PRUEBA 2 - COMPROBACIÓN DE LOS VALORES LEÍDOS POR UNA SONDA.

Esta prueba busca la comprobación de las medidas requeridas a un sensor, así como la verificación de la capacidad de almacenamiento de éstas por parte de la aplicación *wismaphost*. Esta prueba, a través de una petición HTTP, genera un mensaje GET (*wsn/log/id/probe*) del protocolo WISMAPi. Este mensaje, en función del parámetro *status*, permite recibir en el cliente WISMAPi (y por extensión en el cliente HTTP usado) una tabla formateada con información completa sobre las medidas solicitadas.

- Condiciones de éxito: obtener una lectura por cada una de las peticiones enviadas en la Prueba 1.
- Posibles fallos: básicamente, y salvando el posible fallo en la petición al servidor WISMAPi, cuando no se haya podido obtener una medida de un sensor, se habrá debido a un fallo en la comunicación en la red de sensores.
- Protocolo de pruebas: comprobación de que las *n* peticiones de la Prueba 1 han sido correctamente atendidas, así como que todas ellas se visualizan correctamente.

6.3.3 PRUEBA 3 - CAMBIAR EL TIEMPO DE SUEÑO POR DEFECTO DE UN SENSOR.

Esta prueba, está destinada a comprobar que las acciones de configuración de un sensor funcionan correctamente. Para esto, desde la aplicación *WISMAPWEB*, se generará una petición WISMAPi (*POST wsn/sensors/id/man?sleeptime=t*[75]), la cual será recibida por el test-bed y puesta en espera hasta que el sensor destinatario comunique su disponibilidad de recibir mensajes. Cuando el sensor se despierte se le enviará un mensaje WISMAP del tipo *SET_VALUE* con el nuevo valor. Si el mensaje es atendido correctamente, el sensor responderá con el mensaje *SNR_OK*, terminando la prueba.

- Condiciones de éxito: obtención de un mensaje *SNR_OK* por parte de la aplicación *wismaphost* como respuesta a un mensaje *SET_VALUE* enviado a un sensor.
- Posibles fallos: al igual que con la Prueba 1, la probabilidad más alta de fallo se encuentra en las comunicaciones entre la estación base y el sensor, sirviendo las mismas consideraciones realizadas antes.
- Protocolo de pruebas: realización automática de *n* peticiones, espaciadas un tiempo *t*, a través de la aplicación *WISMAPWEB* al test-bed para solicitar un cambio de configuración a un sensor. El parámetro a modificar, como ya se

[75]La prueba se hará sobre el parámetro tiempo de sueño por defecto del Agente de Gestión (*sleeptime*), y *t* tomará valores entre 5.000 ms y 20.000 ms.

ha comentado es el tiempo de sueño por defecto del Agente de Gestión. Se medirá el número de peticiones no completadas con respecto al total de peticiones realizadas.

6.3.4 ESCENARIO DE PRUEBAS.

Las pruebas se han realizado sobre el test-bed instalado en el despacho B-207 de la E. P. S. de Linares, el cual consta de un equipo PC[76] conectado a Internet a través de la red de la Universidad de Jaén; una estación base Sun SPOT conectada al PC a través de su interfaz USB y uno o varios sensores Sun SPOT instalados en un panel fijado a la pared (ver Figura 22). Cada sensor puede estar instalado de tres formas diferentes, conectado al PC a través del USB para su monitorización y mantenimiento de suministro eléctrico, conectado a un concentrador USB que le proporcione alimentación ininterrumpida o completamente aislado, funcionando con el suministro de su propia batería.

Figura 22. Panel de sensores Sun SPOT del test-bed.

6.3.5 RESULTADOS DE LAS PRUEBAS.

En las tablas que recogen los resultados de las pruebas realizadas para *n* peticiones a tres sensores diferentes espaciadas por *t* segundos. La tasa de fallo (*e*) se medirá en tanto por ciento, según la siguiente fórmula:

$$e = \frac{fallos}{n} * 100\ \%$$ **Ecuación 12**

Se considerará fallo todo aquello que impida comprobar el resultado de la acción solicitada. Así pues, los fallos podrían deberse a errores en el software, almacenamiento, interferencias, retardos, etc., que derivaran en:

- No conocer el valor de la sonda para la Prueba 1.
- No poder leer los valores leídos históricamente por una sonda para la Prueba 2.
- No haber cambiado el tiempo de sueño en el Agente de Gestión para la Prueba 3.

[76]Características principales: Microprocesador Intel Core 2 QUAD Q6600 a 2,4 GHz, 2,93 GBytes de RAM y sistema operativo Windows XP Profesional versión 2002 con *Service Pack* 3.

Los resultados obtenidos aparecen detallados en las tablas siguientes.

Tabla 14. Resultados de la Prueba 1 al test-bed.

Prueba 1	n	t(s)	fallos	e
Sensor 1	1000	10	0	0,0 %
Sensor 2	1000	10	3	0,3 %
Sensor 3	1000	10	0	0,0 %

Tabla 15. Resultados de la Prueba 2 al test-bed.

Prueba 2	n	t (s)	fallos	e
Sensor 1	100	20	0	0,0 %
Sensor 2	100	20	0	0,0 %
Sensor 3	100	20	0	0,0 %

Tabla 16. Resultados de la Prueba 3 al test-bed.

Prueba 3	n	t (s)	fallos	e
Sensor 1	100	20	0	0,0 %
Sensor 2	100	20	0	0,0 %
Sensor 3	100	20	0	0,0 %

6.4 CONCLUSIONES DE LAS PRUEBAS.

Previo a las conclusiones de las pruebas realizadas, y con objeto de mostrar el éxito de las mismas, se presentan en la Tabla 16 la media de los fallos obtenidos ($\bar{e}$), el límite unilateral superior del intervalo de confianza[77] (i_{us}), así como la tasa de éxito (E) resultante para dicho valor medio[78] y de los límites del intervalo de confianza (E_i). El intervalo de confianza ha sido calculado usando el test T de Student[79] para cada conjunto de muestras. El motivo por el que se ha empleado el límite de confianza unilateral superior es debido a que no se pueden dar valores negativos del número de fallos. El cálculo de i_{us} responde a la siguiente fórmula:

$$i_{us} = \bar{e} + t_\alpha \left(\frac{S}{\sqrt{n}}\right) \qquad \textbf{Ecuación 13}$$

Donde:

- n es el número de muestras (n=3).
- S es la desviación estándar de las muestras (en este caso S= 1,732).
- t_α es el valor crítico de la distribución T de Student para un α y n-1 grados de libertad (en este caso t_α =6,314).

[77] El intervalo de confianza informa del rango de valores obtenidos entre el que se encuentran el 95% de los mismos. Se ha elegido un 95% ya que es un valor típico usado en estadística, vinculado a la tasa de significancia, que en este caso sería de 0,05 (5%).

[78] La tasa de éxito se ha calculado como el número de experimentos menos el número de fallos medio, dividido entre el número de experimentos, expresado en tanto por ciento.

[79] La prueba *T* de Student (Student1908) se emplea para averiguar el intervalo de confianza de los valores de la muestra así como para comparar las medias de dos conjuntos de valores cuando se disponen de pocos valores de entrada y no se dispone de la desviación típica real.

Tabla 17. Tasa de éxito de las pruebas al test-bed.

Prueba	$\bar{e}$	i_{us}	E	E_i
1	1	7,314	99,900%	99,269%
2	0	No hay variación	100%	No hay variación
3	0	No hay variación	100%	No hay variación

Con objeto de verificar que la media de la Prueba 2 (las demás es evidente que es cero) se le ha realizado la prueba *T* de Student con el resultado del p-valor de 1 (lo cual era también predecible). Por lo tanto no se puede afirmar que la media de fallos de la Pruebas 2 sea distinta de 1.

Como se puede observar en las tablas anteriores, se han producido muy pocos fallos, casi testimoniales, durante todo el proceso de pruebas, lo cual deja clara la estabilidad de los sistemas desarrollados. En concreto, la Prueba 1 ha confirmado la capacidad del test-bed para solicitar la realización de tareas en los sensores. Con la Prueba 2 se ha mostrado la capacidad del test-bed para recoger información de la red de sensores y que ésta pueda ser accedida cuando el usuario la necesite. Por último, en la Prueba 3 se permite comprobar la capacidad que ofrece el test-bed para que un usuario, o una aplicación, realicen tareas de configuración en los nodos de la red.

Además, la estructura de recursos propuesta, junto con la pasarela con Internet (protocolo WISMAPi), posibilitan que los escenarios que modelan las tres pruebas: la solicitud de una medida a un sensor (Prueba 1), la lectura de los valores tomados (Prueba 2) y el cambio de la configuración en un sensor (Prueba 3), puedan realizarse desde cualquier navegador web.

Por lo tanto, puede afirmarse que gracias a los resultados obtenidos en las pruebas, el test-bed desarrollado, el cual se ha basado en el diseño e implementación de la arquitectura multi-agente FLBDIa en los nodos y el protocolo de comunicación WISMAP que son aportaciones de la presente tesis, **cumple satisfactoriamente con los objetivos marcados**. Así pues, se da cobertura a lo marcado en el punto b) del Objetivo general número 2 y los puntos b), c) y d) del Objetivo general número 4.

Se debe hacer notar, que los objetivos conseguidos con estas pruebas, complementan las pruebas siguientes en tanto en cuanto las aplicaciones, la pasarela WISMAPi y el lenguaje de acceso a los recursos en un sensor, han sido empleados para poder realizar dichas pruebas.

6.5 RED DE SENSORES DESPLEGADA EN LA E. P. S. DE LINARES

Adicionalmente a los trabajos de desarrollo y prueba del test-bed, se ha instalado una red de sensores en el interior del Edificio B de la E. P. S. de Linares. Dicha instalación formó parte del trabajo realizado por Andrés Maldonado López para su proyecto fin de carrera de la titulación de Ingeniería de Telecomunicación, dirigido por el Dr. D. José Ángel Fernández Prieto y el autor de la presente tesis.

La red desplegada consta de cinco sensores Sun SPOT aislados, distribuidos desde la planta segunda a la planta baja del edificio, formando una red troncal IEEE 802.15.4 en el edificio. En la Figura 23 puede verse un esquema de la red.

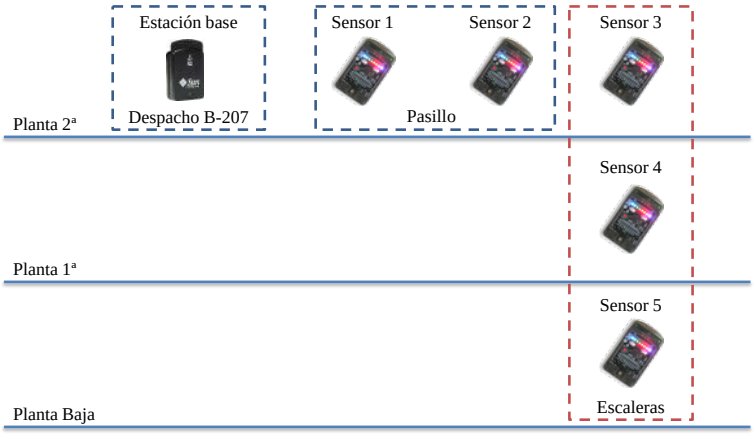


Figura 23. Esquema de la res de sensores instalada en la E. P. S. de Linares.

Con objeto de que cada sensor esté en funcionamiento el mayor tiempo posible, cada uno está instalado en el interior de una caja de PVC IP55[80], convenientemente cerrada, y está conectado a la corriente eléctrica con un alimentador USB para asegurar su funcionamiento continuado (ver Figura 24).

Figura 24. Sensor parte de la red desplegada en la E. P. S. de Linares.

Esta red se ha utilizado tanto para diversos tipos de pruebas, tales como mediciones de alcance de los sensores Sun SPOT, funcionamiento del protocolo WISMAP o la medición del ruido ambiente (Mariscal-Ramirez et al. 2011) (Mariscal-Ramirez et al. 2014).

[80]Según la norma DIN EN IEC 60529, la definición de IP seguida de dos dígitos, indica el grado de protección del contenedor. Cada dígito informa de un nivel de protección, el primero es frente a la entrada de cuerpos sólidos dentro del contenedor, y el segundo frente a líquidos. Concretamente estas cajas (IP55) presentan una alta protección, ya que tan solo podría entrar polvo, y en poca cantidad, además de poder resistir un chorro de líquido dirigido por toda su superficie, sin que éste penetrara al interior.

7 PRUEBAS REALIZADAS A LOS SENSORES CON ARQUITECTURA MULTI-AGENTE

Dado que la arquitectura multi-agente ha sido implementada para su uso dentro de los sensores Sun SPOT [81], y ésta está definida en las clases que forman parte de la aplicación *wismapsensor*, las pruebas realizadas para refrendar el cumplimiento de la Hipótesis primera de la presente tesis, se han basado, en primer lugar, en la valoración del funcionamiento de dicha aplicación, pruebas que aparecen en la presente sección. Además, la propia arquitectura multi-agente FLBDIa, en cuanto a su capacidad de reconfiguración, interacción con el entorno y comunicación, ha sido probada en la sección 6, en la que se han valorado los siguientes aspectos:

- Agente de Gestión: cambios de parámetros de funcionamiento del sensor.
- Agente de Control de Aplicación: toma de muestras.
- Agente de Comunicaciones: soporte al protocolo WISMAP.

Más aún, la capacidad de recepción y procesado de las bases de conocimiento del Agente de Comunicaciones, la cual es objetivo principal de la Hipótesis segunda, será revisada en la sección 8. Y por último, la ejecución por parte de la estructura de sistemas analíticos basados en diferencias, y de sistemas basados en conocimiento, para conseguir una gestión autónoma, dinámica y eficiente de un sensor, objetivo de la Hipótesis tercera, será evaluada en la sección 9. Con todo esto se quiere hacer ver que todas las pruebas realizadas en la presente tesis tienen como base, como no podría ser de otra forma, la estructura multi-agente FLBDIa. Complementando con sus resultados, los obtenidos en la presente sección.

Así pues, a continuación se presentan las pruebas realizadas para evaluar la fiabilidad de la estructura FLBDIa, aspecto fundamental para un sensor, ya que son dispositivos que normalmente trabajan aislados y que deben funcionar sin una supervisión continuada. Por lo tanto, la implementación de la estructura multi-agente FLBDIa debe ofrecer las mayores garantías en cuanto su fiabilidad, conseguida gracias a la robustez de su software.

7.1 PRUEBAS A LA APLICACIÓN WISMAPSENSOR

Como ya se ha comentado, las pruebas se basan en la ejecución de la aplicación *wismapsensor* y su puesta en funcionamiento en sensores Sun SPOT dentro del test-bed descrito en la sección anterior. Estas pruebas tienen como objetivo fundamental el de comprobar la fiabilidad del software que modela la arquitectura multi-agente propuesta y gestiona el funcionamiento de un sensor.

La fiabilidad del sensor, entendida como el tiempo que un sistema funciona correctamente durante un tiempo acotado, se medirá en tanto por ciento y estando éste aislado.

[81] A pesar de que la implementación cuenta con aspectos propios de la tecnología Sun SPOT, el mayor número de dependencias con esta plataforma se localiza, fundamentalmente, en las clases de comunicaciones y de interacción con el entorno (sondas), por lo que el diseño adoptado ha tenido en cuenta este particular, permitiendo una rápida adaptación .a otras plataformas que se basen en Java para la programación de los sensores.

Para poder establecer un criterio objetivo del tiempo que un sensor podría estar en funcionamiento, se usarán los datos ofrecidos por el fabricante de las tecnologías Sun SPOT. Los cálculos concretos se mostrarán más adelante.

7.1.1 DESCRIPCIÓN DE LAS PRUEBAS.

En la prueba diseñada para medir su eficiencia, el sensor funcionará sin alimentación externa, por lo que contará tan solo con su batería, totalmente cargada. Para medir el tiempo que el sensor está activo se tomará como referencia el último mensaje *SNR_AWAKE*[82] que haya enviado.

- Condiciones de éxito: que el sensor siga ejecutando correctamente el ciclo de trabajo programado [83], así como respondiendo adecuadamente a los comandos enviados hasta que se agote su batería.
- Posibles fallos: es evidente que en esta prueba la batería acabará por descargarse, por lo que en sí, el hecho de agotar la batería no se considera un fallo. Por lo tanto, los fallos posibles se derivan de las comunicaciones, problemas del software (bloqueos, desbordamientos de memoria, etc.) y hardware (fallos de memoria, fallos de alimentación de la batería, etc.).
- Protocolo de pruebas: Cada uno de los sensores se pondrá en funcionamiento por periodos indefinidos de tiempo, anotando los ciclos ejecutados y el tiempo transcurrido entre el momento del inicio y un fallo (si lo hubiera). El tiempo máximo de prueba será el que permita la carga de la batería. Este proceso se repetirá al menos 4 veces por sensor.

7.1.2 ESCENARIO DE PRUEBAS.

Las pruebas se han realizado sobre varios sensores Sun SPOT instalados en el panel del test-bed sin alimentación externa. De esta manera se pretende modelar la situación más desfavorable para un sensor, como en el caso de estar desplegado en una red sin posibilidad de acceso a alimentación eléctrica, como sensores desplegados en zonas boscosas remotas o en animales en libertad, entre otras.

7.2 ESTABLECIMIENTO DEL VALOR BASE DE FIABILIDAD.

Para poder definir un valor de fiabilidad acorde con las prestaciones de los sensores, como ya se ha comentado, se hace necesario un estudio del consumo de energía que conllevan diferentes tareas dentro de un nodo Sun SPOT. Así pues, y según la documentación de los sensores Sun SPOT, el régimen normal de funcionamiento (procesador y radio activa, pero sin enviar nada) provoca un consumo de corriente entre 70 mA y 120 mA, que unido a los 720 mAh[84] de carga máxima de batería, da un tiempo máximo de operación de unas seis a diez horas[85]. Si el sensor estuviera en reposo (o

[82]Se debe recordar que los mensajes del protocolo WISMAP incluyen un campo para la carga de batería, por lo que se pueden aislar ciertos tipos de fallos si no se reciben mensajes cuando aún existía suficiente carga, y descartar los derivados de la falta de energía.

[83] El ciclo de trabajo que debe realizar el sensor es despertar del sueño profundo cada 10 segundos, enviar un mensaje *SNR_AWAKE*, esperar la respuesta de la estación base (*BST_SLEEP*) y devolver un *SNR_OK*.

[84]Los datos de consumo son estimaciones para un uso no intensivo, ya que la demanda continuada de altos valores de corriente descargaría más rápido la batería.

[85]El cálculo del tiempo de funcionamiento presentado en esta sección se ha obtenido dividiendo la caga máxima de la batería entre el consumo medio aportado por el fabricante para cada modo de funcionamiento. En este caso, la duración para una carga de proceso normal resulta de dividir 720 mAh entre 70 mA y 720 mAh entre 120 mA, dando como resultado 10h 17’ 8’’ y 6 h, respectivamente. Aunque, como se puede intuir, estos datos son orientativos, permiten establecer unos niveles de referencia aproximados que es lo que se persigue en esta sección.

sueño ligero), consumiría 24 mA (reloj y radio apagados) por lo que podría estar en espera unas 30 horas. Si los ciclos de trabajo incorporan tiempos de reposo profundo[86], la vida del sensor puede ser alargada considerablemente, dado que en este estado el sensor consume tan solo 32 μA. Esto arrojaría un tiempo estimado de funcionamiento de más de novecientos días (solo en tiempo de sueño), lo cual es totalmente inútil ya que el sensor no habría realizado ninguna tarea. Con objeto de resumir los resultados a los que se ha hecho referencia, los tiempos de funcionamiento aparecen resumidos en la Tabla 17. En los apartados siguientes se mostrarán escenarios con ciclos de trabajo más realistas, igualmente basados en los datos de la especificación de los sensores Sun SPOT, con objeto de acotar el máximo tiempo de funcionamiento de un sensor aislado que tuviera un ciclo de trabajo equivalente a su funcionamiento con la arquitectura multi-agente FLBDIa.

Tabla 18. Duración de la batería para su uso de manera continuada en diferentes estados.

	Batería (mAh)	Consumo (mA)	Duración de la batería	
			horas	días
Bajo régimen	720	70,000	10,29	0,43
Alto régimen	720	120,000	6,00	0,25
Sueño ligero	720	24,000	30,00	1,25
Sueño profundo	720	0,032	22.500,00	937,50

7.2.1 CÁLCULO DEL TIEMPO MÁXIMO DE VIDA EN FUNCIÓN DEL TIEMPO DE PROCESO.

Tomando los consumos (*C*) en función del régimen de trabajo de la Tabla 17 como valores medios, en la Tabla 18 se presenta un cálculo estimativo[87]del número máximo de ciclos de trabajo (c_t), a cuya duración se ha denominado tiempo de proceso (t_p). Los ciclos de trabajo se han calculado para un sensor Sun SPOT totalmente cargado (720 mAh), sin contar los periodos de reposo entre los ciclos, los cuales serán contemplados en el siguiente apartado. Se han elegido los tiempos de proceso de 1, 2, 5 y 10 segundos, escenarios A, B, C y D respectivamente, dado que la mayoría de operaciones a realizar por los sensores tienen una duración media con estos valores. Las acciones o procesos que podrían corresponder con los escenarios propuestos aparecen descritos a continuación:

- Escenario A (tiempo de proceso de 1 s): el sensor se despierta y tiene las comunicaciones apagadas.
- Escenario B (tiempo de proceso de 2 s): el sensor está vinculado a la estación base pero no recibe ninguna tarea.
- Escenario C (tiempo de proceso de 5 s): el sensor se despierta y recibe una petición para tomar una medida o un cambio de configuración.
- Escenario D (tiempo de proceso de 10 s): el sensor no está vinculado y envía la petición a la estación base, esperando hasta que expire el temporizador.

[86]Este estado de reposo, del que disponen los sensores Sun SPOT, es el que se induce en el software usado para las pruebas. También se le denomina sueño profundo.

[87] El cálculo del número de ejecuciones se ha obtenido de dividir la carga disponible de la batería (mAh), entre el resultado del producto del consumo de corriente en función del régimen de trabajo (mA) por el tiempo de proceso en horas. El tiempo de proceso en horas se calcula dividiendo el valor t_p entre 3600 segundos que tiene cada hora.

Las diferencias entre un régimen de trabajo alto o bajo, por ejemplo, estribarían en la existencia o no, de alguna tarea adicional encargada al Agente de Control de Aplicación (como tomar medidas de una sonda o inferir un resultado de un FRBS con una base de conocimiento previamente cargada). La fórmula empleada para el cálculo de los tiempos que se muestran en la Tabla 18 es la siguiente:

$$c_t = \frac{720\ mAh}{\left(C \times \frac{t_p}{3600\ s/h}\right)} \qquad \textbf{Ecuación 14}$$

Donde *C* se mide en mA y t_p en segundos.

Tabla 19. Número máximo de ejecuciones en un Sun SPOT en función del tiempo de proceso y del régimen de trabajo.

Escenario	t_p (s)	Consumo (mA)		c_t según régimen*	
		Bajo	Alto	Bajo	Alto
A	1,0	70	120	37.028	21.600
B	2,0	140	240	18.514	10.800
C	5,0	350	600	7.405	4.320
D	10,0	700	1200	3.702	2.160

*Se ha despreciado el tiempo que tarda un sensor en reiniciar desde el estado de sueño profundo, ya que es de apenas unos pocos milisegundos.

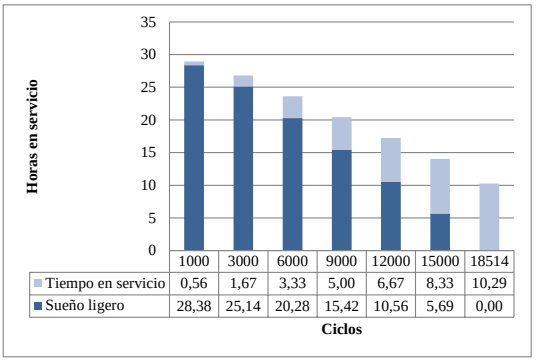
Así pues, un sensor, sin tener en cuenta los tiempos de reposo, podría realizar, como máximo, entre 37.028 y 21.600 ciclos de trabajo para el escenario A. Sin embargo, en ese caso, las tareas ejecutadas por el sensor serían mínimas, siendo poco representativo su empleo como referencia. Por lo tanto, se tomarán como base los valores de los casos B y C, los cuales representan unos escenarios más completos y que implican un mayor número de sistemas y elementos dentro del sensor. De esta manera, el número de ciclos que podría ejecutar un sensor, despreciando el tiempo de sueño, estaría en el rango de los 18.000 a los 4.000 ciclos aproximadamente.

Una vez vistos los ciclos de ejecución máximos previstos para un sensor, en el apartado siguiente se calculará la máxima duración de la batería incorporando el consumo de los tiempos de sueño entre ciclos de trabajo.

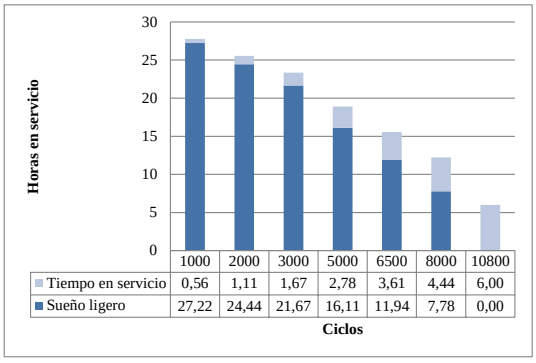
7.2.2 CÁLCULO DEL TIEMPO MÁXIMO DE VIDA CON CICLOS DE SUEÑO Y PROCESO ALTERNADOS.

Como ya se ha adelantado, se van a presentar los tiempos máximos de vida, y por tanto los ciclos de trabajo, contando con el consumo producido por los ciclos de sueño entre ejecuciones, De esta manera, cada ciclo de trabajo constará de un tiempo de proceso y un tiempo de reposo. Así pues, se van a presentar dos series de gráficas (una por cada tipo de reposo, ligero o profundo) y en función del régimen de trabajo (bajo y alto), para los escenarios B y C. El sensor se considerará que está completamente cargado. Se han elegido los escenarios B y C ya que, según los cálculos mostrados en el apartado anterior, representan situaciones más complejas que el escenario A e implican un mayor número de ciclos que el escenario D.

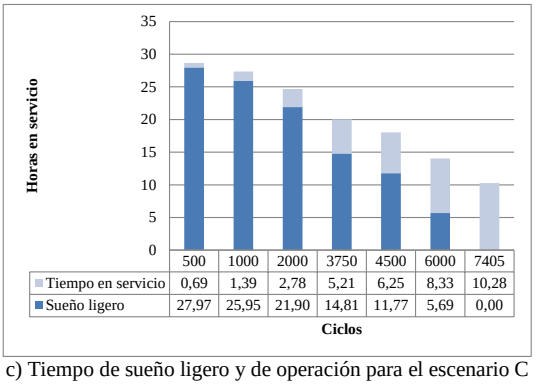
En la Figura 25, se presenta una estimación[88] del tiempo máximo de operación de un sensor que intercalara tiempo de proceso y de sueño ligero, en función del número de ciclos. Los consumos usados son los detallados en la Tabla 17. Se ha contemplado que entre cada ejecución podría haber un tiempo de reposo mínimo (incluso nulo). Así por ejemplo, en el caso de la viñeta a) de la Figura 25, para 9000 ciclos de trabajo, el sensor estaría en servicio un total de 20,42 horas hasta que se agotara su batería. De este tiempo, 5 horas serían el resultado del proceso de 9000 ciclos de trabajo de 2 segundos, mientras que permanece en sueño ligero el resto del tiempo, en periodos de unos 6 segundos (15,42 h / 9000). Además, como puede observarse, en cada gráfica de la Figura 25, se presentan un número creciente de ciclos de trabajo hasta que el tiempo de proceso representa el 100% del consumo de batería, el cual coincide con los valores de ciclos de trabajo presentado en la Tabla 18.



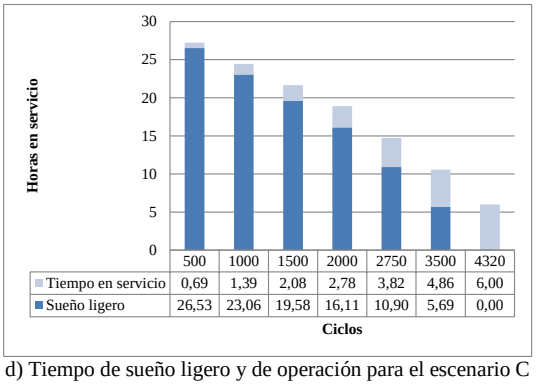
a) Tiempo de sueño ligero y de operación para el escenario B con régimen de proceso bajo



b) Tiempo de sueño ligero y de operación el escenario B con régimen de proceso alto



c) Tiempo de sueño ligero y de operación para el escenario C con régimen de proceso bajo



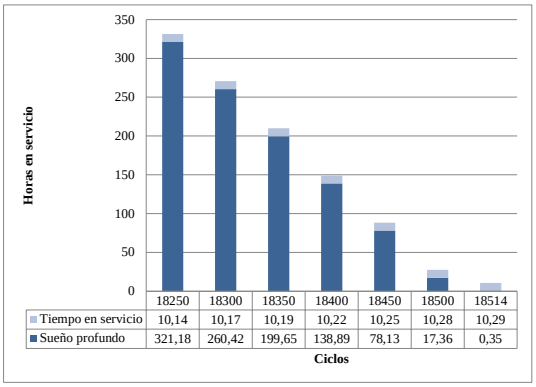
d) Tiempo de sueño ligero y de operación para el escenario C con régimen de proceso alto

Figura 25. Comparación entre el tiempo en sueño ligero y de operación.

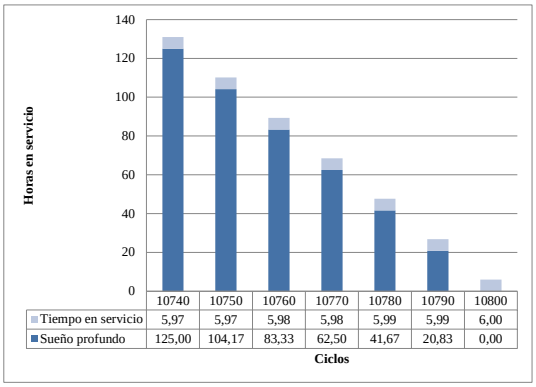
En la Figura 26 se muestra el resultado de las mismas operaciones que en la figura anterior, salvo que los tiempos de reposo son de sueño profundo. Con objeto de que sea apreciable el tiempo de operación, se han elegido valores de ciclos de ejecución cerca de su límite máximo, el cual coincide con la última barra de cada gráfico. Como se puede apreciar, el incremento en el tiempo de operación es significativo con respecto a

[88] La barra del *Tiempo en servicio* es el resultado de multiplicar el tiempo de proceso de cada caso por el número de ciclos. Por otro lado, la barra de *Sueño ligero*, o *Sueño profundo*, se calcula restando del total de la carga de batería (720 mAh) el consumo resultante de multiplicar el número de ciclos por el consumo por ciclo presentado en la Tabla 18 para cada caso. El valor de energía resultante se divide entre el consumo en reposo (ligero o profundo) y este es el valor de tiempo representado en la gráfica.

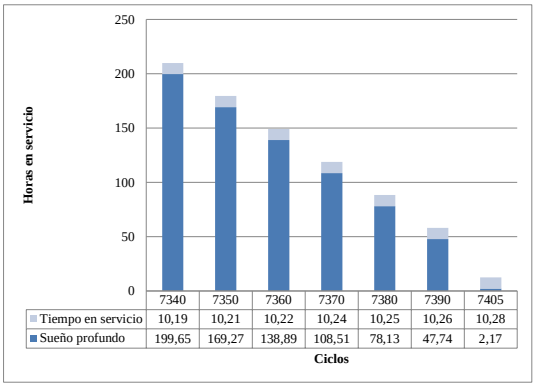
los mostrados anteriormente con reposo en sueño ligero. Esto se debe a que el consumo del sensor en sueño profundo es muy inferior al consumo en sueño ligero.



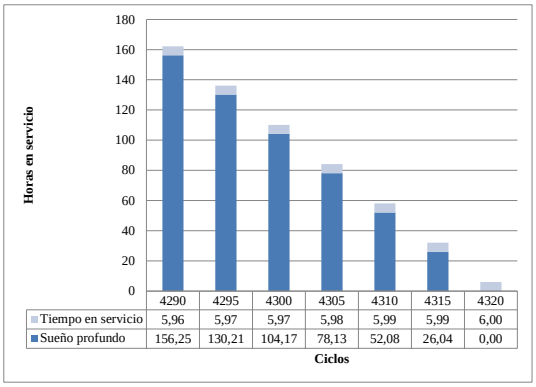
a) Tiempo de sueño profundo y operación para el escenario B con régimen de proceso bajo



b) Tiempo de sueño profundo y operación para el escenario B con régimen de proceso alto



c) Tiempo de sueño profundo y operación para el escenario C con régimen de proceso bajo



d) Tiempo de sueño profundo y operación para el escenario C con régimen de proceso alto

Figura 26. Comparación entre el tiempo en sueño profundo y de operación.

Como puede observarse en las dos figuras anteriores, en el mejor de los casos, un sensor no realizaría muchos más de 18.000 ciclos de proceso (para tiempos de proceso de dos segundos y su batería cargada al máximo). El tiempo en servicio iría desde unas 10 horas para los 18.000 ciclos y reposo con sueño ligero, hasta más de seiscientas, si el sensor entrara en sueño profundo. Por lo tanto, teniendo en cuenta los valores obtenidos en el análisis presentado en este apartado, se tomará como referencia el valor obtenido para ciclos de ejecución de 2 segundos, 18.504. Se dejará un margen de tolerancia del 10%, lo que arroja un valor de 16.654 ciclos, ya que los valores usados se derivan de las especificaciones técnicas del fabricante, y los sensores usados llevan en servicio varios años, lo cual podría haber degradado sus prestaciones, sobre todo la capacidad total de la batería. Así pues, se considerará que la ejecución de una aplicación es fiable al 100% siempre y cuando sea capaz de ejecutarse sin errores 16.654veces. A este valor se le denominará *fiabilidad máxima 1* o fm_1.

En el apartado siguiente se presenta un estudio adicional de consumo en función de las transmisiones y recepciones necesarias para uno de los intercambios de mensajes típicos del protocolo WISMAP.

7.2.3 ANÁLISIS DE CONSUMO CON COMUNICACIONES.

Con objeto de ofrecer una visión completa de las capacidades de un sensor, para así poder tener un criterio adicional para la valoración de las pruebas de fiabilidad, se presenta a continuación un estudio similar a los anteriores en el que se han tenido en cuenta el envío de mensajes. El número de mensajes enviados y recibidos por un sensor serán los mínimos definidos por el protocolo WISMAP. Estos son los siguientes:

- Enviar *SNR_AWAKE*.
- Recibir (en el caso de que no tenga ninguna tarea que realizar) *BST_SLEEP*.
- Enviar *SNR_OK*.

Por lo tanto, un sensor que siga el protocolo WISMAP, teniendo activo el Agente de Comunicaciones, envía al menos dos mensajes y recibe otro. De esta manera, las pruebas que se muestran a continuación reflejarán este hecho, además de incluir un tiempo mínimo de proceso de un segundo con carga de proceso baja (70 mA).

Para calcular el consumo de batería de las comunicaciones se han tomado como base las especificaciones del chip radio CC2420 fabricado por Texas Instruments y que es usado por los sensores Sun SPOT. Con respecto a los datos de transmisión, se ha usado lo marcado por la norma IEEE 802.15.4:

- Banda: 2400-2483,5 MHz.
- Régimen binario: de 250 Kbps y 62,5 Ksímbolos/s. La norma indica que un octeto requiere de dos símbolos para su transmisión.
- Consumo por recepción: 19,7 mA.

Tabla 20. Consumo del chip CC2420 en función de la potencia de transmisión.

Potencia de transmisión (dBm)	Consumo (mA)
0	17,4
-5	14,0
-10	11,0
-15	9,9
-25	8,5

La longitud en bytes de los mensajes enviados se deriva de la pila de protocolos usada, la cual se presenta en la Figura 27, por lo que se hace preciso un breve recuento de los tamaños de las cabeceras de dichos protocolos. En primer lugar se muestra el desglose de la longitud mínima de las cabeceras para todos los niveles:

- IEEE 802.15.4: cada trama tiene 6 bytes fijos a nivel físico[89], 3 bytes más (*Frame ControlField* y *Data SequenceNumber*, entre 0 y 20 de dirección (8 en este caso), en la cabecera MAC, los datos y dos bytes finales (*FrameCheckSequence*). Así pues, cada trama de datos llevará un mínimo de 19 bytes más los datos de usuario[90].

[89] Preámbulo de trama 4 bytes, byte SFD (*Start of Frame Delimiter*), que forman la Cabecera de sincronización y longitud de la PDU del nivel MAC (1 byte) que es la cabecera real de nivel físico.

[90] El tamaño máximo para una trama a nivel MAC es de 127 bytes, por lo que en este caso la cantidad máxima de datos de usuario por trama será igual a 127 – 3 – 8 – 2 = 114 bytes.

- LoWPAN: necesita una cabecera mínima de 2 bytes.
- Nivel de transporte: se han usado los protocolos que implementan los sensores Sun SPOT, uno orientado a conexión y otro no, siendo la longitud de ambas cabeceras 3 y 1 bytes respectivamente. El mensaje *SNR_AWAKE* usa el protocolo sin conexión y los otros dos, el protocolo orientado a conexión.
- Nivel de aplicación: la longitud de la cabecera de los mensajes del protocolo WISMAP es de ocho bytes, y dado que no se necesitan más datos, coincide con la longitud total del mensaje.

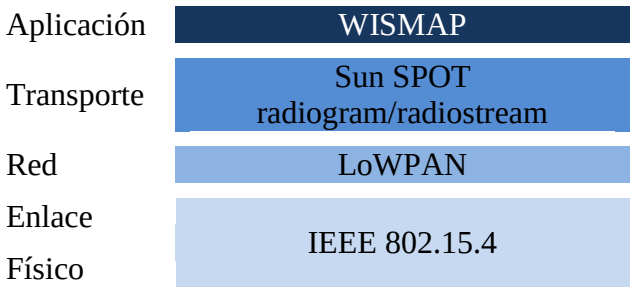


Figura 27. Torre de protocolos en un sensor.

Por consiguiente, la longitud en bytes de cada tipo de mensaje es la que aparece reflejada a continuación:

- Mensajes sin conexión en transporte: Físico y MAC (19 bytes) + LoWPAN (2 bytes) + Transporte (1 byte) + WISMAP (8 bytes) = 30 bytes.
- Mensajes con conexión en transporte: Físico y MAC (19 bytes) + LoWPAN (2 bytes) + Transporte (3 bytes) + WISMAP (8 bytes) = 32 bytes.

Por lo tanto, se enviarían los mensajes *SNR_AWAKE* (30 bytes) y *SNR_OK* (32 bytes) y se recibiría *BST_SLEEP* (32 bytes), siendo estos datos los que se usarán para el cálculo de la estimación de ciclos de trabajo que se podrían llevar a cabo sin descanso (en cualquier caso, si se introdujera un tiempo de reposo, sería un número inferior de ciclos). El cálculo realizado sigue el siguiente proceso:

1. Se calcula el tiempo en segundos requerido para el envío de dos mensajes (62 bytes) y la recepción de otro de 32 bytes. Este tiempo se obtiene dividiendo el número de bytes duplicado (2 símbolos por byte), entre la velocidad por símbolo que define la norma IEEE 802.15.4 de 62,5 Ksímbolos/s
2. El tiempo obtenido para la transmisión se multiplicará por cada uno de los consumos mostrados en la Tabla 19, con objeto de presentar los resultados para diferentes potencias de transmisión. Para la recepción será el consignado previamente de 19,7 mA. Los resultados de estas multiplicaciones, en mAs, se convertirán a mAh, dividiendo entre 3600 s/h.
3. A los valores de consumo de energía obtenidos se añade un segundo de proceso en bajo régimen (70 mA), lo que da un valor de 0,0194 mAh.
4. Para hallar el número de ciclos de envío, se ha dividido la carga total de la batería (720 mAh), entre cada uno de los valores de consumo obtenido para las diferentes potencias de transmisión mostradas en la Tabla 19.

Los resultados se muestran en la Figura 28.

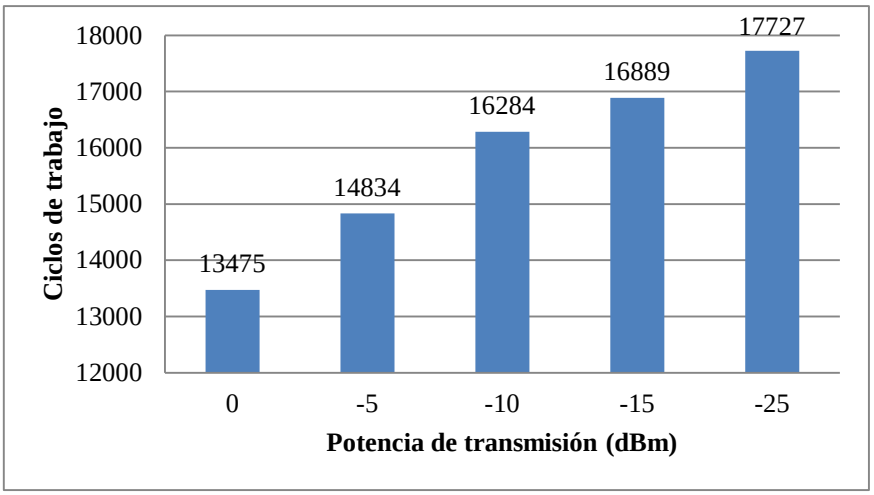


Figura 28. Ciclos máximos de envío para un sensor para diferentes potencias de transmisión.

Como puede observarse, el número máximo de ejecuciones estimadas con transmisión de mensajes WISMAP, incluso con los valores de potencia de transmisión más bajos, no supera los 18.000 ciclos. Por lo tanto, y aplicando el mismo margen de tolerancia que en el apartado anterior, el valor de referencia en función del análisis de consumo de las comunicaciones sería de 15.955 (17.727 -10%). A este valor, con objeto de diferenciarlo del obtenido en el apartado anterior, se le denominará *fiabilidad máxima 2* o fm_2.

Tanto el valor fm_1, como el fm_2, serán usados por separado para contrastar la fiabilidad de las pruebas a las que se ha sometido el software.

7.3 RESULTADOS DE LAS PRUEBAS.

Como se ha establecido en la sección anterior, los valores de referencia para medir la fiabilidad del software que modela la arquitectura multi-agente FLBDIa[91]son 16.654 (fm_1) y 15.955 (fm_2). En la Tabla 20 se muestran los resultados de las pruebas realizadas a dos sensores aislados.

En la Tabla 21 se muestran los valores medios de los ciclos de trabajo ejecutados ($\overline{c_t}$), además de su desviación típica (σ) y el intervalo de confianza (i_c) para cada uno de los valores medios obtenidos en las pruebas. Los valores de Fiabilidad 1 (*f1*) y Fiabilidad 2 (*f2*), en tantos por ciento, corresponden con la relación entre la media de ciclos ejecutados por un sensor y el número máximo de ciclos para los procesos sin comunicaciones (fm_1) o con comunicaciones (fm_2), los cuales se han calculado de la siguiente manera:

$$f1 = (\overline{c_t}/fm_1) \times 100\% \qquad \textbf{Ecuación 15}$$

$$f2 = (\overline{c_t}/fm_2) \times 100\% \qquad \textbf{Ecuación 16}$$

[91]Con objeto de tener una referencia previa de la fiabilidad del sistema, algunas de las pruebas aisladas realizadas han conseguido que un sensor, conectado a un concentrador USB, esté en funcionamiento con la aplicación *wismapsensor* durante cuatro meses. El ritmo de trabajo fue de más de 4000 ciclos de sueño y proceso por día, lo que arroja un total de más de 500.000 ejecuciones del software. Por lo tanto, en el caso de trabajar aislado, queda de manifiesto que las probabilidades de que un sensor ejecutando la aplicación *wismapsensor* no presente un problema son muy altas.

Tabla 21. Resultados de las pruebas de ejecución de la aplicación wismapsensor en un sensor aislado.

Sensor1		Sensor2	
Ciclos	**Tiempo(h)**	**Ciclos**	**Tiempo(h)**
15.553	43,20	16.972	47,14
14.523	40,34	16.609	46,14
14.481	40,23	16.760	46,56
14.292	39,70	16.825	46,74
14.326	39,79	16.589	46,08
14.126	39,24	16.852	46,81
14.434	40,09	16.796	46,66
14.351	39,86	16.759	46,55
14.227	39,52	16.901	46,95
13.816	38,38	16.913	46,98
14.166	39,35	16.876	46,88
14.520	40,33	16.679	46,33
14.863	41,29	16.863	46,84
15.123	42,01	16.796	46,66
14.781	41,06	16.853	46,81

Tabla 22. Resultados de la prueba de fiabilidad.

	$\bar{c}_t$	t	i_c	f1	f2	σ
Sensor 1	14.505,47	40,84	14.266,71 - 14.744,23	87,10%	90,91%	416,53
Sensor 2	16.802,87	45,97	16.742,33 - 16.863,40	100,00%	100,00%	105,6

Al igual que en el apartado 6.4, el intervalo de confianza ha sido calculado usando el test *T* de Student para cada conjunto de muestras. Dado la mayor diferencia en las muestras, y dado que el test *T* necesita que las éstas sigan una distribución normal, se han sometido ambos conjuntos al test de normalidad de Shapiro-Wilk[92], arrojando en ambos casos un resultado satisfactorio.

Como se puede observar en la tabla anterior, tanto el Sensor 1 como el 2 han conseguido un alto valor (incluso el máximo) para la fiabilidad. Las diferencias obtenidas, después de un análisis más detenido de algunas de las pruebas presencialmente, se deben como principal motivo a la variabilidad de los retardos en la respuesta por parte de la estación base. Por otro lado, efectos más difícilmente observables son, por ejemplo, la actualización de las tablas de encaminamiento o interferencias en las comunicaciones que derivan en la no contabilización de algún mensaje enviado por el sensor.

[92]El test de Shapiro-Wilk (Shapiro & Wilk 1965) es utilizado para comprobar la normalidad en conjuntos de pocas muestras (menos de cincuenta usualmente). Este test plantea como hipótesis nula la normalidad de los conjuntos de muestras, por lo que si el p-valor es mayor de 0,05 no se puede descartar la hipótesis nula, y por tanto los conjuntos de muestras presentan normalidad, lo cual es el objetivo que se persigue con el test.

7.4 CONCLUSIONES DE LAS PRUEBAS.

Como se desprende de las pruebas realizadas, se ha conseguido una gran fiabilidad en la aplicación *wismasensor*, encargada de modelar la arquitectura FLBDIa. Por lo tanto, se puede afirmar que **la arquitectura propuesta puede ser trasladada con la suficiente fiabilidad a un sensor, Sun SPOT en este caso, y que su ejecución está prácticamente asegurada para toda la duración de la batería del mismo**. De esta manera se cumplen los puntos a) y c) del Objetivo general 2.

Así pues, teniendo en cuenta los resultados de las pruebas realizadas a la arquitectura FLBDIa, modelada por la aplicación *wismapsensor*, junto con los obtenidos en las pruebas ya presentadas en la sección 6, y el cumplimiento de los objetivos ya comentados, se puede afirmar que la **Hipótesis primera** de la presente tesis queda refrendada.

8 PRUEBAS DE DISTRIBUCIÓN DE BASES DE CONOCIMIENTO CON EL PROTOCOLO WISMAP.

En esta sección se mostrarán los resultados obtenidos con objeto de avalar la Hipótesis segunda de la presente tesis. En dicha hipótesis se proponía la creación de un protocolo de aplicación que permitiera soportar sistemas basados en conocimiento en una red de sensores, de una manera eficiente.

En primer lugar, se debe hacer notar que el protocolo WISMAP ha sido usado en los dos bancos de pruebas anteriores con diferentes propósitos, siendo todos ellos llevados a cabo satisfactoriamente. Sin embargo, queda por contrastar su capacidad de envío de bases de conocimiento en una red de sensores. Así pues, se ha realizado una batería de pruebas que hacen uso de dicha capacidad de envío de bases de conocimiento del protocolo, con objeto de comprobar la fiabilidad de las transmisiones, así como el retardo obtenido en escenarios reales. Además, con las pruebas de transmisión y medición de retardo, se ha comprobado la robustez de la implementación del Agente de Comunicación, complementando así las pruebas realizadas en la sección anterior.

8.1 PRUEBAS.

Las pruebas realizadas para comprobar la efectividad del protocolo WISMAP en el envío de bases de conocimiento se han realizado también sobre sensores Sun SPOT, los cuales fueron desplegados en el Edificio B de la E. P. S. de Linares. Estos sensores tenían instalado el programa *wismapsensor*, que da soporte a la estructura multi-agente FLBDIa. Con respecto a las bases de conocimiento, se emplearon las usadas en (Gadeo-Martos et al. 2011) ya que, además de formar parte de trabajos previos del grupo de investigación, corresponden con un ejemplo real de aplicación para la detección temprana de plagas en el cultivo del olivo[93].

Cabe destacar que estas pruebas se lanzaron desde el equipo PC y estación base que forman parte del test-bed desarrollado como parte de la presente tesis.

El objeto fundamental de estas pruebas es que, dado que la información a enviar obliga al uso de varios mensajes a nivel de enlace (IEEE 802.15.4), el retardo que se produzca esté acotado y no experimente fluctuaciones significativas entre envíos, así como evitar problemas con el re-ensamblado de las bases de conocimiento. Además, se medirá la tasa de fallo, relacionando el número de mensajes perdidos, con respecto al total de enviados. El retardo medido será el tiempo de ida y vuelta o RTT (del inglés *round trip time*) entre el envío de un mensaje WISMAP *SET_VALUE*, y su correspondiente respuesta *SNR_OK*.

8.1.1 CONDICIONES DE ÉXITO Y FALLO

Si durante el periodo de espera se produce un error, o una expiración de los temporizadores de espera (puestos a un máximo de 8 segundos), se considerará que esa prueba ha sido fallida. Si se recibe durante la espera el mensaje *SNR_OK* proveniente del sensor, el experimento se consignará como exitoso.

[93]En este caso se aplican técnicas de lógica difusa y redes de sensores para poder anticipar la presencia de plagas del olivo como el repilo (Spilocaeaoleagina), la mosca del olivo (Bactroceraoleae) o la polilla del olivo (Praysoleae). Así pues, gracias a la medición de parámetros ambientales como la humedad y la temperatura se pueden anticipar a su aparición y prevenirla con tratamientos de fumigación.

8.1.2 POSIBLES ERRORES Y/O FALLOS

El principal problema que puede presentar esta prueba es la pérdida de un mensaje durante el proceso de cálculo de las rutas en la red.

Aparte de este problema, y aunque en menor medida, podrían darse la pérdidas debidas a interferencias con otras emisiones en la misma banda. Pero, con objeto de evitar al máximo este problema, y después de analizar las posibles fuentes de interferencia (fundamentalmente la red WIFI de la Universidad de Jaén), se decidió mantener como canal de emisión el 26 de la norma IEEE 802.10.4, el cual está en la parte más alta de la banda ISM de 2,4 GHz y no se solapaba con ningún otro servicio.

Otro tipo de problemas que se pueden presentar, lo cuales podrían derivar en un fallo, serían los errores en la ejecución del software, tanto el programa *wismaphost*, como el software que gestiona los nodos, *wismapsensor*. Sin embargo, la probabilidad de este tipo de errores, después de las pruebas de fiabilidad realizadas, se asume como casi inexistente.

Por último, estarían los fallos derivados del hardware. Este tipo de fallos se asumen como posibles, sobre todo en sensores cuya vida de uso supera los cuatro años. Sin embargo, tras la detección de este tipo de fallos, el proceso a seguir sería el de reemplazar ese dispositivo por otro en mejores condiciones, y reiniciar la prueba.

8.1.3 PROTOCOLO DE PRUEBAS

Cada base de conocimiento será obtenida de un fichero en formato XML^{KB} con la aplicación *wismaphost*. Una vez leída, se enviará manualmente al sensor destinatario 30 veces, a intervalos regulares de entre 10 y 20 segundos. Este número de repeticiones se ha establecido con objeto de permitir un posterior análisis estadístico de los resultados obtenidos.

La aplicación anotará el instante de envío (mensaje *SET_VALUE*), así como el momento en el que se reciba la respuesta (mensaje *SNR_OK*).

8.1.4 ESCENARIOS DE LAS PRUEBAS

Se desplegaron cinco redes diferentes, una para cada prueba, en las que se contó siempre con la misma estación base y un número variable de nodos, los cuales fueron reutilizados en los diversos montajes. La cantidad de sensores se fue incrementando en cada escenario en 2. Así pues, en el primero de ellos, la red estaba formada por la estación base y un solo nodo (el destinatario), el segundo contaba con tres nodos (dos intermedios y el destinatario), y así sucesivamente hasta el quinto escenario en el que intervenían nueve sensores. El detalle puede verse en la Tabla 22.

Tabla 23. Experimentos de envío de bases de conocimiento.

Prueba	Estación base	Sensores	Saltos	Niveles[94]	Distancia aprox. (m)
1	1	1	1	1	8
2	1	3	3	1	26
3	1	5	5	1	44
4	1	7	7	2	58
5	1	9	9	3	74

[94]Los niveles se refieren a las plantas del edificio involucradas en el experimento.

La distancia entre los nodos que estuvieran en la misma planta fue entre 8 y 10 metros aproximadamente, mientras que para aquellos que servían para cambiar de planta, se redujo a unos 6 metros (red de 7 y 9 nodos). En la Figura 29 puede verse un esquema con el despliegue de los sensores para cada una de las pruebas. La potencia de transmisión se ajustó en cada nodo para evitar, en lo posible, que dos nodos no adyacentes entre sí pudieran ponerse en contacto, alterando así el resultado de la prueba..

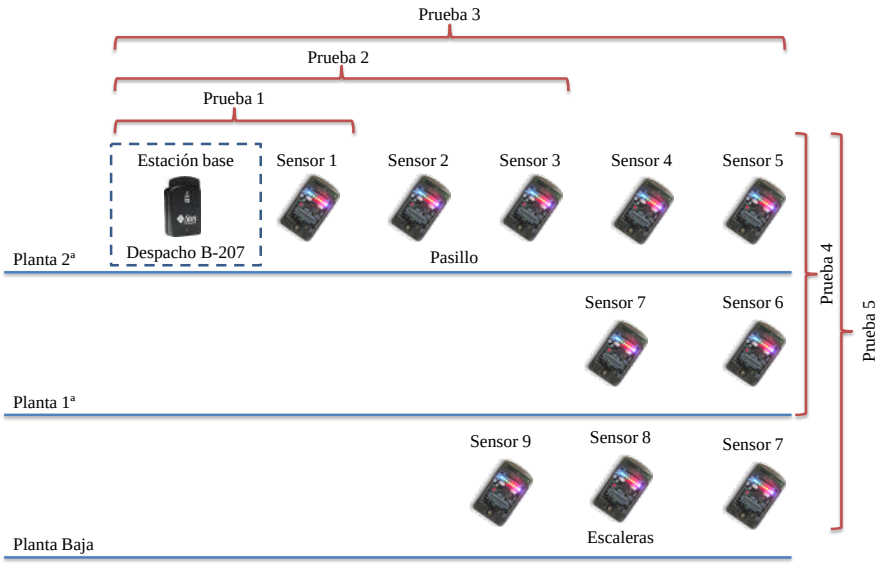


Figura 29. Esquema de las redes de sensores desplegadas para las pruebas.

El objeto de realizar las pruebas con este número de sensores es el de modelar escenarios típicos que se darían en redes reales de mayor tamaño y topologías diversas, sin tener que realizar un despliegue completo de la red. Así pues, desde el caso de un salto, el cual podría modelar una red en estrella de pocos nodos, al de nueve saltos, que podría modelar una red mallada, en estrella o jerárquica de gran cantidad de nodos, se cubren una gran variedad de topologías y tamaños de redes de sensores. En la Figura 30 se muestran algunos ejemplos de topologías que se podrían modelar con los escenarios de pruebas propuestos.

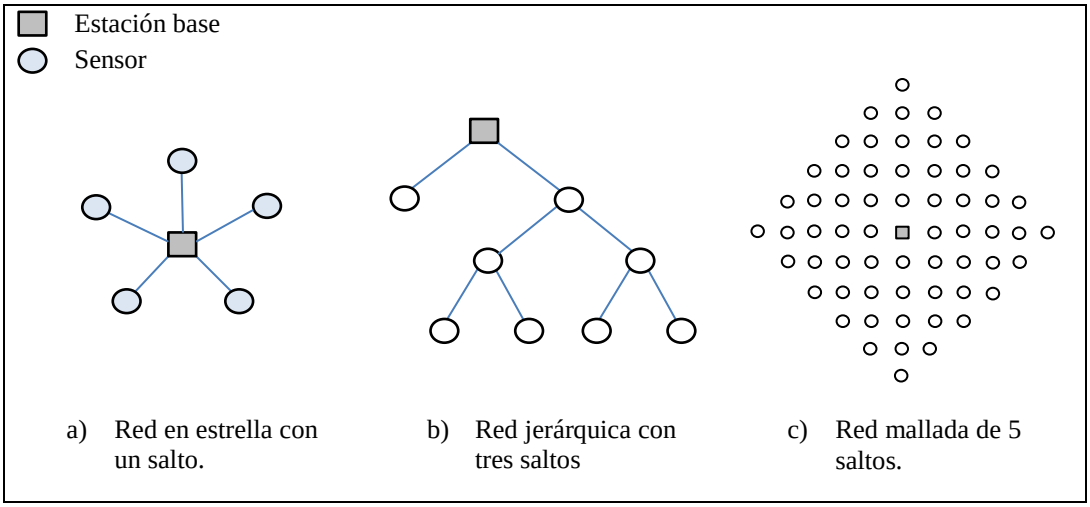


Figura 30. Ejemplos de redes con topologías equivalentes a los escenarios propuestos.

Para cada prueba, tan solo el sensor al final de la red será el destinatario de la base de conocimiento, mientras que los nodos intermedios se comportarán como enrutadores. Esto es posible gracias a que los sensores Sun SPOT proporcionan esta capacidad, usando para este fin el protocolo LQRP[95](Sun Microsystems2008).

Las características de las base de conocimiento usadas, kb1, kb2 y kb3, aparecen reflejadas en la Tabla 23. Para su definición se ha usado el formato XML^{KB} desarrollado en esta tesis, así como para su transmisión se ha empleado el formato *raw*, igualmente definido en el presente trabajo de investigación.

Tabla 24. Detalles de las bases de conocimiento.

	kb1	**kb2**	**kb3**
Variables de entrada	3	2	4
Variables de salida	1	1	1
Conjuntos borrosos	12	7	13
Reglas	2	1	5
Tamaño en bytes	332	492	740

8.2 ANÁLISIS PRELIMINAR DE LOS RESULTADOS

Los resultados obtenidos en la medición del RTT en el envío de cada una de las bases de conocimiento se muestran en la Tabla 24. En dicha tabla se detallan los valores de retardo obtenido para el envío de cada una de las tres bases de conocimiento, en función del número de saltos de la red de pruebas (1, 3, 5, 7 y 9).

Como queda patente después de la revisión de la Tabla 24, el envío de más bytes a través de la red, así como que los mensajes tengan que atravesar más nodos, aumenta el retardo obtenido en la distribución de las bases de conocimiento.

Con respecto a las pruebas en las que no se ha obtenido un valor de retardo (un '-' en la tabla anterior), se debe a que la prueba no fue satisfactoria. Para no desvirtuar la prueba, no se repitieron, considerándolos como fallos de la transmisión. Se puede apreciar que los fallos se localizan en pruebas concretas, fundamentalmente en la distribución de la kb1 para nueve saltos, y la kb2 para tres saltos. Después de un estudio de cada caso se pudo comprobar que se debieron a problemas en la ubicación (y orientación) de los sensores, ya que no se encontraban en soportes fijos, y la propia distribución por la planta del edificio provocaba reflexiones y caminos alternativos por patios interiores que propiciaban cambios en la ruta durante esas pruebas que derivaron en la pérdida de paquetes. En la Figura 31 se muestran los valores medios obtenidos para la transmisión de cada base de conocimiento en función del número de saltos.

[95]Protocolo LQRP, del inglés *Link QualityRoutingProtocol*, es una modificación desarrollada por Sun Microsystems del protocolo AODV (Perkins et al. August 2003). El protocolo LQRP está especializado en enrutamiento en malla y gestiona dicho enrutamiento a través de una función de medición de la calidad del enlace (*LQI*) y de la salud del nodo, derivada del estado de la batería del sensor.

Tabla 25. RTT obtenido en el envío de las bases de conocimiento a través de la red de pruebas.

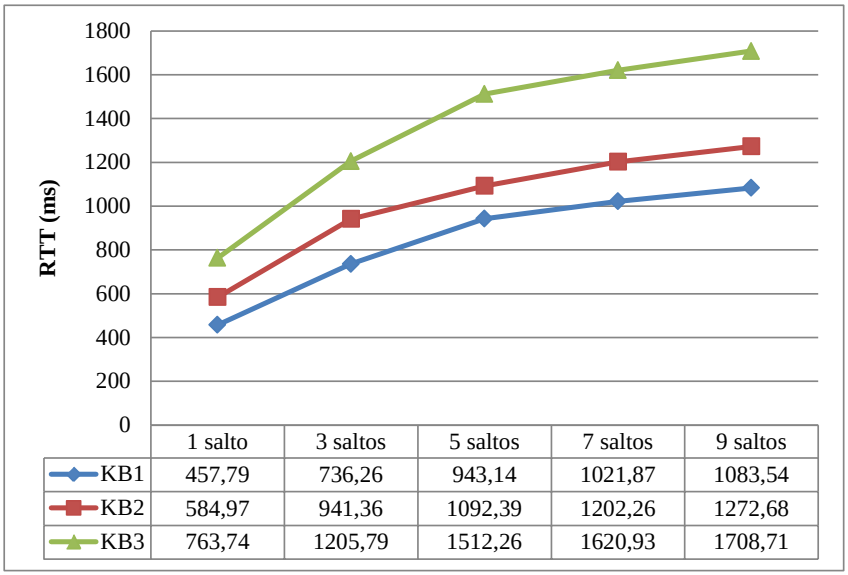
	kb1 RTT (ms)					kb2 RTT (ms)					kb3 RTT (ms)				
Prueba	**1**	**3**	**5**	**7**	**9**	**1**	**3**	**5**	**7**	**9**	**1**	**3**	**5**	**7**	**9**
1	463	755	1066	1209	1180	595	937	1250	1198	1264	770	1031	1503	1606	1818
2	466	781	1026	1150	1219	615	1109	1119	1149	1261	772	1194	1546	1588	1742
3	441	734	918	1077	993	564	1000	1055	1162	1286	845	1171	1442	1674	1757
4	472	734	878	950	1154	564	906	1031	1090	1253	748	1264	1496	1670	1870
5	461	734	897	1052	1051	563	953	1076	1215	1298	748	1205	1579	1661	1610
6	478	745	964	980	1223	627	1000	1121	1246	1370	770	1218	1583	1643	1721
7	472	765	1019	985	968	601	906	1234	1171	1260	752	1140	1605	1544	1875
8	476	703	890	1085	1004	600	890	1274	1211	1332	757	1218	1598	1749	1760
9	470	734	956	950	1067	591	890	1237	1132	1278	757	1234	1497	1570	1644
10	445	718	870	1060	1125	566	937	1118	1130	1260	759	1218	1639	1754	1604
11	446	709	893	1156	1002	575	906	1049	1111	1252	763	1281	1512	1705	1554
12	456	718	882	1092	1022	638	984	1150	1264	1153	765	1359	1398	1573	1750
13	471	690	1098	969	1078	631	968	1100	1178	1264	767	1218	1560	1599	1737
14	444	734	1041	1067	-	570	937	1435	1154	1280	771	1180	1540	1770	1804
15	439	750	966	1107	-	558	875	1124	1374	1360	770	1177	1515	1565	1528
16	468	718	1045	945	-	577	906	1015	1143	1247	773	1210	1440	1842	1677
17	455	765	1112	1148	-	563	921	1067	1244	1324	774	1218	1520	1450	1786
18	440	703	975	967	-	573	921	1158	1204	1431	745	1150	1608	1618	1848
19	450	734	939	954	-	572	906	1017	1283	1191	745	1222	1530	1559	1595
20	451	718	931	971	-	583	937	1031	1351	1322	748	1221	1388	1556	1629
21	468	781	912	945	-	600	953	1140	1330	1196	752	1178	1528	1516	1834
22	458	656	974	933	-	599	968	1213	1240	1285	756	1155	1545	1504	1716
23	444	701	911	960	-	824	-	1262	1160	1301	767	1205	1454	1688	1762
24	478	750	944	921	-	583	-	1097	1231	1364	758	1264	1503	1503	1619
25	455	799	877	1010	-	602	-	1017	1206	1205	760	1235	1617	1753	1651
26	445	687	977	954	-	567	-	1492	1207	1131	761	1234	1603	1560	1597
27	468	734	914	933	-	574	-	1064	1136	1236	765	1180	1436	1648	1607
28	450	796	935	1051	-	586	-	1252	1238	1321	766	1177	1606	1647	1629
29	446	768	880	1020	-	571	-	1102	1254	1159	749	1211	1391	1514	1597
30	-	705	899	1034	-	568	-	1164	1146	1219	772	-	1667	1599	1910

Observando la Figura 31, resulta evidente que el tamaño de cada base de conocimiento es un factor importante, así como el número de saltos del experimento[96]. Estos resultados eran de esperar, aunque sea una apreciación cualitativa, dada la limitación de 127 bytes que como máximo se puede enviar en una trama MAC IEEE 802.15.4. De ese total de bytes disponibles, tan solo quedarían libres para datos de los niveles superiores 114 bytes[97]. Descontando los bytes de cabecera para el protocolo LoWPAN (2 bytes), transporte de Sun SPOT con conexión (3 bytes) y WISMAP (8 bytes), quedan solo 101

[96]Ambas afirmaciones serán corroboradas estadísticamente en el apartado 8.3

[97]Cabecera MAC: 2 bytes (*Frame Control Field*), 1 byte (*Data SequenceNumber*), 8 bytes de dirección en este caso y los dos bytes finales (*FrameCheckSequence*), en total 13 bytes.

bytes disponibles para el envío de la base de conocimiento. Por lo tanto, para kb1 se necesitarán al menos 5 mensajes, 6 para kb2 y 8 para kb3.



	1 salto	3 saltos	5 saltos	7 saltos	9 saltos
KB1	457,79	736,26	943,14	1021,87	1083,54
KB2	584,97	941,36	1092,39	1202,26	1272,68
KB3	763,74	1205,79	1512,26	1620,93	1708,71

Figura 31. RTT medio para cada prueba.

A continuación, con objeto de caracterizar los valores de retardo obtenidos, se presentan dos figuras con diagramas de caja[98]. La primera (Figura 32) representa los valores de retardo obtenidos en función de la base de conocimiento enviada, independientemente del número de saltos, mientras que la segunda (Figura 33) muestra los valores de retardo en función del número de saltos para las tres bases de conocimiento enviadas conjuntamente.

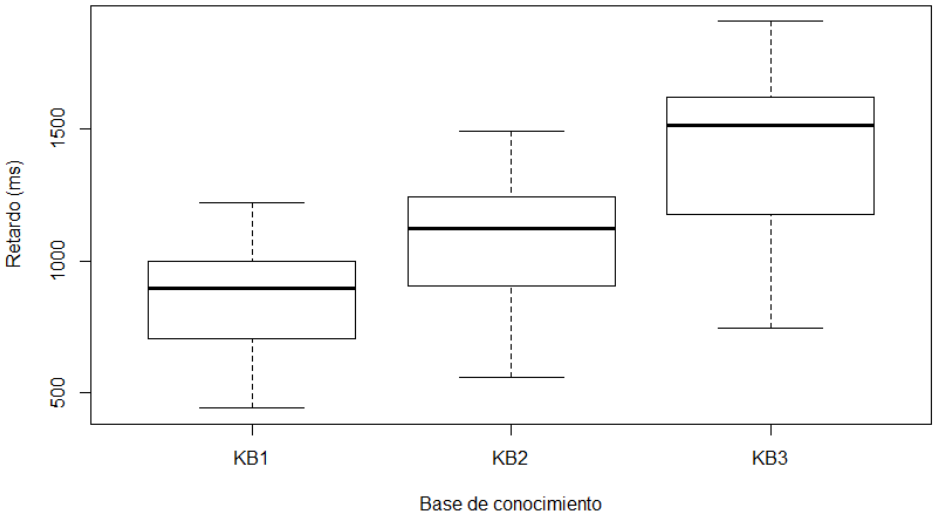


Figura 32. Retardos en función de la base de conocimiento enviada.

[98] Los diagramas de Caja (conocidos también en inglés por *boxplots* o *box and whiskers*) son una representación visual usada para describir las características importantes de un conjunto de datos, a la vez que dan información de dispersión y simetría. En cada gráfico se representan los tres cuartiles y los valores, mínimo y máximo, de los datos, sobre un rectángulo, ya sea alineado horizontal o verticalmente. El segundo cuartil, o Q2 representa la mediana del conjunto de datos, representada con una línea de mayor grosor dentro de la caja.

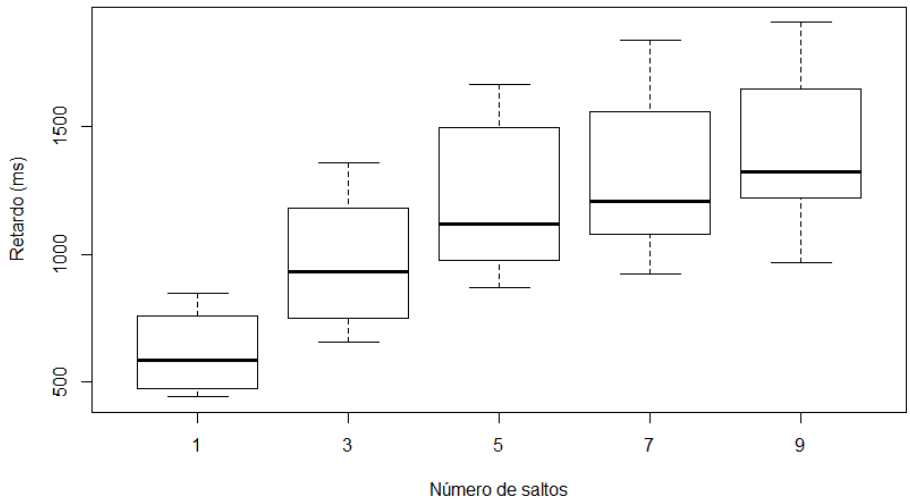


Figura 33. Retardos en función del número de saltos.

8.3 ANÁLISIS ESTADÍSTICO DE LOS RESULTADOS

Como ya se ha introducido, para poder realizar un análisis fiable y representativo a través de la tasa de éxito, se han procesado estadísticamente los datos de RTT obtenidos. La técnica usada para tal fin es un análisis de varianzas, también conocido como ANOVA[99] (del inglés *ANalysis Of VAriance*) de una vía. Así pues, esta herramienta permite la comparación de las medias de dos o más grupos de datos independientes. En concreto de han realizado dos análisis por separado, en el primero se han cotejado las varianzas de los valores del retardo obtenidos en función de la base de datos enviada, mientras que en el segundo se analizan dichos valores de retardo en función del número de saltos. Así pues, en ambos casos se planteará como hipótesis nula la no existencia de diferencias significativas entre las medias de los retardos, buscándose por lo tanto, que se pueda rechazar dicha hipótesis a través de la consecución de un p-valor menor que el intervalo de confianza fijado (α), que se estableció en α=0,05. Sin embargo, es necesario que para poder realizar el análisis de las varianzas con la técnica ANOVA, los conjuntos de muestras cumplan una serie de requisitos:

- **Independencia**: los valores de RTT medidos son independientes, ya que responden a envíos separados en el tiempo lo suficiente para que no exista interferencia entre ellos.
- **Normalidad**: el análisis de las varianzas con ANOVA puede aceptar incluso cierta no-normalidad. De esta manera, el número de pruebas realizadas para los diferentes casos ha sido elegido para propiciar la normalidad del conjunto de valores de retardo. A pesar de todo, se ha comprobado la normalidad de los valores obtenidos para casa caso, realizando el test de Shapiro-Wilk para cada uno. Gracias a este test, se detectó que en siete de los quince grupos se descarta la hipótesis de normalidad (ver Tabla 25), por lo que se procedió a una reestructuración de los valores de retardo y a repetir las pruebas de normalidad con 25, 20 y 12 muestras. En el último caso (12 muestras) se consiguió una

[99]La técnica ANOVA se basa en la comparación de las medias de dos o más grupos de valores, si solo fueran dos grupos podría usarse un test *T* de Student. Pudiendo descartar la hipótesis nula (no hay ninguna media significativamente menor de las demás) si el p-valor es menor de 0,05. Esta técnica fue desarrollada por Ronald. A. Fihser a principios del Siglo XX, comenzando a ser ampliamente conocida a partir de la publicación de su libro *Statistical Methods for Research Workers*(Fisher1970).

mejora sensible en la normalidad de los conjuntos de retardos, ya que tan solo dos de ellos no presentaban una distribución normal (ver Tabla 26). Esto derivó en la realización de una batería complementaria de análisis de varianzas, lo cual será explicado más delante.

Tabla 26. Resultados del test de normalidad Shapiro-Wilk (p-valor) para el conjunto completo de muestras.

Base de conocimiento	Número de saltos				
	1	3	5	7	9
Kb1	**0,048670**	0,796829	**0,013641**	**0,016057**	0,245407
Kb2	**0,048670**	0,796829	**0,013641**	**0,016057**	0,245407
Kb3	0,494519	**0,012237**	0,417676	0,611173	0,200168

Tabla 27. Resultados del test de normalidad Shapiro-Wilk (p-valor) para 12 muestras por prueba.

Base de conocimiento	Número de saltos				
	1	3	5	7	9
Kb1	0,263248	0,685418	0,056902	0,544281	0,169780
Kb2	0,202398	**0,038468**	0,177089	0,926422	0,109960
Kb3	**0,000204**	0,492179	0,788720	0,576069	0,534791

- **Homocedasticidad**: u homogeneidad de las varianzas, de los conjuntos de muestras. Dichas varianzas deben de ser razonablemente iguales para que se pueda aplicar un análisis de las mismas. Esto puede comprobarse, por ejemplo, a través del test de Levene (Levene1960), Bartlett (Bartlett1937) o el de Cocrhan [100], más robusto frente a conjuntos de datos que no presenten la característica de normalidad, lo cual ha sido determinante para elegir este último como método de comprobación de la homogeneidad de la varianza de los conjuntos de muestras. La aplicación del método de Cochran ha confirmado la homogeneidad de las varianzas de los retardos en función del número de saltos para cada una de las bases de conocimiento enviadas. Esto ha permitido afrontar con más garantías el análisis de las varianzas a través de la técnica ANOVA.

Por lo tanto, dado que la normalidad está presente en la mayoría de los conjuntos de datos, en primer lugar se realizará un ANOVA para el conjunto de todos los datos y posteriormente para solo con 12 muestras, con objeto de comparar ambos análisis. Además, para despejar dudas sobre los resultados, se realizará el mismo contraste de varianzas a través de la prueba de Kruskal-Wallis (Kruskal & Wallis 1952), un método no paramétrico, equivalente al ANOVA, pero que no supone normalidad en los datos, pero sí homogeneidad en las varianzas.

8.3.1 JUSTIFICACIÓN DE LA HIPÓTESIS DE PARTIDA

En los apartados siguientes se detallan los dos análisis realizados a los datos de retardo, en primer lugar en función de la base de conocimiento enviada, y en segundo lugar, en función del número de saltos. Para ambos casos se planteará como hipótesis nula la no existencia de diferencias significativas entre las medias del RTT. Esto implicaría, en el caso de no poderse descartar la hipótesis nula, que no existen diferencias significativas

[100] El test de Cochran(Snedecor & Cochran 1980) se usa para comprobar la homogeneidad de las varianzas de los conjuntos de muestras, generalmente de grandes variaciones de la varianza.

en las pruebas para el envío de cada base de conocimiento, ni en función del tamaño en bytes de éstas, ni en el número de saltos en la ruta de los mensajes.

La elección de estas hipótesis nulas tiene como objeto el confirmar que el envío de las tres bases de conocimiento con el protocolo WISMAP, en cinco escenarios diferentes, depende del tamaño de la base de conocimiento y del número de nodos que tiene que atravesar el mensaje[101].

El no poder descartar la hipótesis nula, derivaría en suponer que el envío de información de tamaños, sensiblemente diferentes, con el protocolo WISMAP y en redes con número de nodos variable, no obtiene valores de RTT significativamente diferentes. Esto daría a entender, en primer lugar, que las pruebas no han sido realizadas correctamente, ya que existe suficiente disparidad en los escenarios de prueba propuestos como para esperar variaciones significativas en los valores obtenidos. En segundo lugar, se podría inferir que las bases de conocimiento elegidas no son lo suficientemente diferentes entre sí como para probar un protocolo pensado para su envío, ya que su transmisión no supondría ninguna diferencia significativa con respecto a las demás, y por lo tanto, la tasa de éxito que resulte estaría adulterada. Y por último, si el objeto de esta prueba es demostrar la efectividad del protocolo WISMAP en un entorno real, tampoco serían válidos los resultados de tasa éxito a obtener, al no haber sido probados en entornos suficientemente dispares. Por lo tanto, como consecuencia de no poder descartar la hipótesis nula en las pruebas propuestas, no se podría contrastar que el formateo, envío y procesado de bases de conocimiento por el protocolo WISMAP es viable en entornos reales, y por consiguiente tampoco se podría justificar la Hipótesis segunda de la presente tesis.

8.3.2 CASO 1. ANOVA PARA EL CONTRASTE DE LAS MEDIAS EN FUNCIÓN DE LA BASE DE CONOCIMIENTO ENVIADA.

Como ya se ha comentado, la hipótesis nula que se plantea es la no existencia de diferencias significativas en los retardos obtenidos en el envío de cada base de conocimiento en los escenarios propuestos. Formalmente:

H_0: $\mu_{kb1} = \mu_{kb2} = \mu_{kb3}$

H_1: alguna μ_i es distinta.

En primer lugar se aplicará un ANOVA para todo el conjunto de datos. Los resultados obtenidos aparecen reflejados en la Tabla 27.

Tabla 28. Resultados del ANOVA del RTT en función de la base de conocimiento enviada.

	Suma de Cuadrados	gl	Media Cuadrática	F	P
Base de conocimiento	21.789.451	2	10.894.726	128	<2e−16
Residuos	36.689.857	431	85.127		

[101]Esto se conseguirá si es el resultado de las pruebas a realizar, ANOVA y Kruskal-Wallis permiten, todas ellas, descartar la hipótesis nula.

Como se puede apreciar en la tabla anterior, el factor P es menor de 0,05, lo cual permite rechazar la hipótesis nula y confirmar que existen diferencias significativas entre al menos las varianzas de los retardos de los envíos con dos de las bases de conocimiento empleadas. Para poder averiguar cuáles son las bases de conocimiento cuyos RTT difieren significativamente, así como en qué sentido es esta diferencia, se ha practicado a los resultados el test de honestidad, o HSD (Tukey & Cox 1994)(del inglés *Honestly-significant-difference*), de Tukey, cuyos resultados aparecen en la Tabla 28.

Dado que en este caso, para todas las comparaciones posibles entre bases de conocimiento, el resultado es significativo (P ajustado<0,05), se podría afirmar que existen diferencias significativas en los RTT obtenidos en las pruebas en función de la base de conocimiento enviada.

Tabla 29. Resultados del test de honestidad de Tukey para los RTT en función de la bases de conocimiento.

Bases de conocimiento comparadas	Diferencia	Límite inferior	Límite superior	P ajustado
KB2-KB1	211,1228	129,1911	293,0544	0
KB3-KB1	543,7407	462,7146	624,7669	0
KB3-KB2	332,6180	253,2299	412,0060	0

Sin embargo, dado que algunos de los grupos de valores no presentaban un comportamiento normal, se repitió el análisis para conjuntos de 12 valores de RTT para cada experimento (Tabla 29) y el consiguiente test de honestidad Tukey (Tabla 30).

Tabla 30. Resultados del ANOVA del RTT en función de la base de conocimiento enviada para 12 valores de RTT por caso.

	Suma de Cuadrados	gl	Media Cuadrática	F	P
Base de conocimiento	8.389.606	2	4.194.803	49,8	<2e−16
Residuos	14.909.226	177	84.233		

Tabla 31. Resultados del test de honestidad de Tukey para los RTT en función de la bases de conocimiento para 12 valores de RTT por caso.

Bases de conocimiento comparadas	Diferencia	Límite inferior	Límite superior	P ajustado
KB2-KB1	1.697.667	4.452.354	2.950.098	0,0045632
KB3-KB1	5.186.167	39.337.354	6.438.598	0
KB3-KB2	3.488.500	22.360.687	4.740.931	0

Como puede observarse en la Tabla 30, los resultados vuelven a ser satisfactorios, rechazándose la hipótesis nula en todos los casos de comparación entre bases de conocimiento.

Para completar el análisis de los retardos obtenidos en función de la base de conocimiento, y dada la no normalidad de algunos de los conjuntos de valores de RTT

se ha realizado un contraste de Kruskal-Wallis. Dicho contraste asume como hipótesis nula que todos los valores pertenecen a la misma población, asumiendo igualdad de medianas. Por lo tanto, si se rechaza la hipótesis nula, se puede considerar que los retardos evaluados, en este caso en función de la base de conocimiento enviada, no pertenecen a la misma población. El resultado es el siguiente:

Tabla 32. Resultados del contraste Kruskal-Wallis para los RTT en función de la bases de conocimiento.

	Chi cuadrado[102]	gl	P
Kruskal-Wallis	155,3645	2	<2,2e−16

A la vista del resultado de la prueba de Kruskal-Wallis, se puede afirmar, que como era de esperar, sí existen diferencias significativas para los retardos obtenidos en función de la base de conocimiento enviada. Así pues, queda corroborada la primera asunción para poder considerar relevante la tasa de error obtenida.

En el apartado siguiente se repetirá este mismo proceso para el caso de la búsqueda de diferencias significativas en los retardos en función del número de saltos que atraviesan los mensajes del protocolo WISMAP.

8.3.3 CASO 2. ANOVA PARA EL CONTRASTE DE LAS MEDIAS EN FUNCIÓN DEL NÚMERO DE SALTOS DEL ESCENARIO DE PRUEBAS.

En este caso, se busca encontrar diferencias significativas en las medias obtenidas en los valores de RTT para el envío de las bases de conocimiento en función del número enlaces entre sensores (saltos) que debe atravesar el mensaje. Por lo tanto, la hipótesis nula a plantear es la no existencia de dichas diferencias en las medias, formalmente:

$$H_0: \mu_{1salto} = \mu_{3saltos} = \mu_{5saltos} = \mu_{7saltos} = \mu_{9saltos}$$

$$H_1: \text{alguna } \mu_i \text{ es distinta.}$$

Siguiendo el proceso marcado por el apartado anterior, en primer lugar se aplicará un ANOVA para todo el conjunto de datos. Los resultados obtenidos aparecen reflejados en la Tabla 32.

Tabla 33. Resultados del ANOVA del RTT en función del número de saltos del escenario usado para el envío de las base de conocimiento.

	Suma de Cuadrados	gl	Media Cuadrática	F	P
Número de saltos	35575207	4	8893802	166.6	<2e-16
Residuos	22904101	429	53390		

[102] La distribución Chi cuadrado (χ^2, pronunciada como *ji*) o de Pearson, es una distribución de probabilidad continua, de cola a la derecha, cuyo parámetro k representa los grados de libertad de la variable aleatoria. El contraste Kruskal-Wallis la usa para aproximar el estadístico *H*, resultado de la prueba (de hecho a esta prueba también se la denomina prueba *H*)

Al igual que en el caso anterior, el resultado es satisfactorio. A continuación se comprueba se la diferencia entre las varianzas es así para todas las combinaciones, o solo para algunas, aplicando el test de honestidad de Tukey. Los resultados se muestran en la Tabla 33.

Tabla 34. Resultados del test de honestidad de Tukey para los RTT en función del número de saltos.

Número de saltos comparados	Diferencia	Límite inferior	Límite superior	P ajustado
3-1	349,37443	252,99567	445,7532	0
5-1	602,30178	508,96934	695,6342	0
7-1	675,21576	581,88332	768,5482	0
9-1	812,15297	713,43672	910,8692	0
5-3	252,92735	157,04106	348,8136	0
7-3	325,84133	229,95504	421,7276	0
9-3	462,77854	361,64431	563,9128	0
7-5	**72,91398**	**-19,90984**	**165,7378**	**0,2003415***
9-5	209,85118	111,61569	308,0867	0,0000001
9-7	136,9372	38,70171	235,1727	0,0014423

* Resultado no significativo.

Tras ver los resultados del test de Tukey, tan solo en uno de los casos, el que compara los retardos de 5 saltos con los obtenidos con el escenario de 7 saltos, no se puede descartar la hipótesis nula, lo cual da a entender que, en ese caso, el escenario de siete saltos propuesto no consiguió ser significativamente diferente al de 5. Con respecto a los demás, se puede afirmar que los RTT medidos son significativamente diferentes, y por tanto los escenarios válidos.

Al igual que antes, se repetirá el análisis ANOVA (Tabla 34) para grupos de datos de 12 RTT, haciendo el test de Tukey seguidamente (Tabla 35).

Tabla 35. Resultados del ANOVA del RTT en función del número de saltos del escenario usado para el envío de las base de conocimiento.

	Suma de Cuadrados	gl	Media Cuadrática	F	P
Números de saltos	13.618.329	4	3.404.582	61,55	<2e-16
Residuos	9.680.503	175	55.317		

Como puede observarse en la Tabla 35, los resultados son satisfactorios salvo para los escenarios de 5 y 7 saltos (igual que en la prueba anterior con los grupos completos de muestras), y para la comparación de 9 y 7 saltos.

Tabla 36. Resultados del test de honestidad de Tukey para los RTT en función del número de saltos.

Número de saltos comparados	Diferencia	Límite inferior	Límite superior	P ajustado
3-1	361,13889	208	514	0
5-1	599,80556	447	753	0
7-1	688,44444	536	841	0
9-1	755,61111	603	908	0
5-3	238,66667	86	391	0
7-3	327,30556	174	480	0
9-3	394,47222	242	547	0
7-5	**88,63889**	**-64,167099**	**241,4449**	**0,5000805***
9-5	155,80556	2,999567	308,6115	0,0432479
9-7	**67,16667**	**-85,639322**	**219,9727**	**0,7446692***

* Resultado no significativo.

Completando el análisis presentado para el caso anterior, en la Tabla 36 se presenta el resultado del contraste de Kruskal-Wallis en función del número de saltos.

Tabla 37. Resultados del contraste Kruskal-Wallis para los RTT en función de la bases de conocimiento.

	Chi cuadrado	gl	P
Kruskal-Wallis	105,5269	4	<2,2e−16

8.3.4 CONCLUSIONES DE LAS PRUEBAS ESTADÍSTICAS

Después de las pruebas realizadas, se puede afirmar que la suposición de que las bases de conocimiento elegidas, así como los escenarios de pruebas propuestos, son lo suficientemente representativos como para dar validez a la tasa de error obtenida, la cual se analizará en la sección siguiente. Tan solo cabe destacar que, para el caso de los escenarios con 5 y 7 saltos, las diferencias obtenidas, teniendo en cuenta los resultados conjuntos para el envío de las tres bases de conocimiento, no han sido significativos. Esto puede indicar que, o bien las distancias para esos experimentos no fueron suficientemente largas, o que algunos mensajes recorrieron una ruta más corta. Esto pudo suceder al entrar momentáneamente dos sensores no adyacentes en contacto (probablemente por reflexiones o ligeras variaciones en la orientación) dentro de su rango de cobertura, reduciendo así el número de saltos que recorrieron algunos mensajes. El análisis de la tasa de éxito tendrá en cuenta este hecho en los resultados que se presentan en la siguiente sección.

8.4 ANÁLISIS DE LA TASA DE ÉXITO

Una vez se ha determinado que los experimentos realizados han obtenido resultados estadísticamente significativos, y por lo tanto se pueden considerar representativos para probar el protocolo WISMAP, en tanto en cuanto representan escenarios reales más complejos, se procederá a analizar la tasa de éxito obtenida, para así poder avalar el

funcionamiento de dicho protocolo y por consiguiente la Hipótesis segunda de la presente tesis.

Así pues, teniendo en cuenta los fallos que se reflejan en la Tabla 24, se han obtenido los valores de la tasa de éxito[103] que se muestran en la Tabla 37.

Tabla 38. Tasa de éxito para el envío de las bases de conocimiento.

Caso	Fallos/Saltos						Total de casos	Tasa de éxito
	1	3	5	7	9	Total		
Envío de kb1	1	0	0	0	17	18	150	88,00 %
Envío de kb2	0	8	0	0	0	8	150	94,37%
Envío de kb3	0	1	0	0	0	1	150	99,33%

Se debe hacer constar que cuando se contó con una ubicación estable de los sensores (todas las pruebas menos las dos antes mencionadas, kb1 con 9 saltos y kb2 con 3 saltos), tan solo se dieron dos casos de mensajes perdidos (kb1 con un salto, y kb3 con 3 saltos). Así pues, se ha procedido a calcular una tasa de éxito parcial[104], la cual arroja un 99,49%. Este resultado puede considerarse como muy satisfactorio, teniendo en cuenta que se trata de comunicaciones inalámbricas, en interiores y en la banda ISM, la cual es usada por gran cantidad de dispositivos, más aún en el entorno universitario.

En la Tabla 38 se muestran los resultados del número medio de fallos obtenidos ($\bar{e}$) para el envío de cada base de conocimiento, el límite unilateral superior[105] del intervalo de confianza (i_{ui}), así como la tasa de éxito (E) resultante para el dicho valor medio y del límite del intervalo de confianza (E_i). Estos valores han sido calculados sin tener en cuenta los resultados obtenidos para los escenarios de 7 saltos, dado que no se han obtenido diferencias significativas en la comparación con el escenario de 5 saltos, por lo tanto el número total de casos para calcular la tasa de éxito será solo 120.

Tabla 39. Tasa de éxito para el envío de las bases de conocimiento sin el escenario de 7 saltos.

Caso	$\bar{e}$	i_{ui}	E	E_i
Envío de kb1	4,5	30,85	96,25%	74,29%
Envío de kb2	2	14,63	98,33%	87,81%
Envío de kb3	0,25	1,83	99,79%	98,48%

Como puede observarse, los resultados son suficientemente satisfactorios para los tres casos. Además, la tasa de éxito acumulada[106] resulta un 92,50%, habiendo obtenido un valor parcial para el envío de la kb3 de un 99,79%.

[103] La tasa de éxito se ha calculado dividiendo el número de valores de retardo no medidos, 18, 8, y 1 para la kb1, kb2 y kb3 respectivamente, entre el número total de pruebas, para el envío de cada base de conocimiento, que es de 150.

[104] Sin tener en cuenta los envíos de kb1 con nueve saltos y kb2 con 3 saltos.

[105] Las consideraciones y fórmula para cálculo son las mismas que las mostradas en el apartado 6.4, actualizando los valores de $\bar{e}$, S y n correspondientemente.

[106] La tasa de éxito acumulada se ha calculado comparando todos los fallos obtenidos frente al total de experimentos, 360.

8.5 CONCLUSIONES DE LAS PRUEBAS

Por lo tanto, teniendo en cuenta los resultados de tasa de éxito obtenidos, así como la justificación de la validez de los mismos presentada en esta sección, se puede concluir que el protocolo WISMAP ha sido probado satisfactoriamente en el envío de bases de conocimiento en redes de sensores, cumpliendo con el Objetivo general 3, y por consiguiente refrendando la **Hipótesis segunda** dela presente tesis.

9 APLICACIÓN DE SISTEMAS INTELIGENTES A LOS SENSORES

Para completar el capítulo de pruebas, se presentan a continuación los trabajos realizados para refrendar la Hipótesis tercera de la presente tesis. Así pues, y según lo expuesto en dicha hipótesis, se esperaba que, aplicando tanto sistemas analíticos basados en diferencias, como sistemas basados en conocimiento, los sensores que integren una red puedan gestionarse de manera autónoma o colaborativa, dinámica y eficientemente sin requerir una reprogramación de los mismos, ni un cambio en los protocolos de comunicaciones.

El motivo de que esta hipótesis se formule en último lugar, así como el hecho de que también las pruebas que conducen a refrendarla se presenten al final del capítulo, se deriva de que necesita apoyarse en los trabajos realizados para las dos primeras hipótesis, como son:

- Arquitectura multi-agente, que permita la gestión interna del sensor por medio de sistemas inteligentes.
- Protocolo de comunicación que de soporte a los sistemas inteligentes, así como a la gestión remota de la arquitectura multi-agente y recursos internos del sensor.
- Test-bed de pruebas y aplicaciones que dan soporte tanto al test-bed, como a la arquitectura multi-agente y al protocolo de comunicación en una implementación real sobre sensores Sun SPOT.

Así pues, en esta sección se presentarán los resultados de la aplicación de un sistema basado en diferencias (Sistema Diferencial) y un sistema basado en FRBS para la estimación del tiempo de sueño de un sensor (SMI) en una aplicación de monitorización del entorno, en concreto de la magnitud presión sonora. Para el sistema FRBS se mostrarán los resultados de la aplicación de las dos bases de conocimiento mostradas en la sección 5.

Como ya se explicó con anterioridad (véase la sección 5), ambos sistemas[107] intentan mantener al sensor dormido el mayor tiempo posible, siempre y cuando estimen que las condiciones de la magnitud que están monitorizando se lo permitan. Por lo tanto, si cualquiera de ambos sistemas, en función de sus parámetros de entrada, estima que la magnitud va a sufrir pocos cambios en un futuro, tenderá a incrementar el tiempo de sueño. Si por el contrario las medidas indican que la magnitud se encuentra en un régimen de mayor volatilidad, o en un rango de valores que debe ser controlado con mayor precisión, acortará los tiempos de sueño. Se debe hacer notar que ambos tipos de sistemas se han pensado como sub-sistemas del Agente de Gestión, y en concreto el basado en FRBS, ha dado lugar a la creación de una arquitectura BDI ligera difusa o FLBDIa (véase la sección 5.2). Por lo tanto, como sub-sistemas del Agente de Gestión, ofrecen a éste una estimación del nuevo SMI, la cual puede ser usada, o no, en función del modo de funcionamiento en el que esté configurado el agente.

Así pues, en los apartados siguientes se presentarán las pruebas realizadas a ambos sistemas, así como un análisis estadístico de los mismos. Por lo tanto, y con objeto de

[107] A estos sistemas, se les denominará también estimadores SMI, dado que en parte realizan una estimación de cuánto tiempo tiene que estar el sensor en reposo, entre los instantes en los que sea preciso realizar una medida de la magnitud bajo estudio.

fuera abordable la realización de un número de pruebas suficiente para que los resultados tuvieran relevancia estadística, se ha empleado un software de simulación especialmente diseñado para este propósito. De esta manera, se permite abordar la repetición de los experimentos[108] un número de veces suficiente para que se pueda asumir la propiedad de normalidad[109] del conjunto de los valores de entrada, además de evitar los problemas que se derivarían de unas pruebas en dispositivos reales[110].

Otra ventaja de realizar las pruebas a través de una simulación es que se facilita la adaptación a diferentes escenarios, además de conseguir que ambos sistemas tomen las mismas medidas de la magnitud bajo estudio, aspecto casi imposible si las medidas las realizaran dispositivos reales.

Por otro lado, la razón de diseñar un software específico, y no usar alguno de los disponibles en el mercado, se fundamenta, en primer lugar, en la no existencia de un software especializado para probar este tipo de sistemas, por lo que en definitiva se requeriría la implementación dentro de alguna plataforma conocida, como podría ser Matlab o NS. En segundo lugar, dado que el software desarrollado era totalmente funcional y estaba realizado en Java, la posibilidad de usar exactamente el mismo código implementado en los sensores, descartaría cualquier problema o desajuste derivado de la conversión a cualquier otro lenguaje. Así pues, la aplicación empleada para realizar las pruebas de los sistemas de estimación del SMI se ha implementado en lenguaje Java, apoyándose en todas las clases ya desarrolladas en el caso de los sistemas de estimación del SMI.

Así pues, al usar un software de simulación específico se ha podido realizar una comparación en las mismas condiciones de ambos sistemas, además de emplear exactamente el mismo código que se ejecutaría en los sensores, lo cual evita cualquier tipo de posible ambigüedad o desajuste.

9.1 PRUEBAS

Las pruebas tratan de emular, para cada uno de los estimadores presentados, el funcionamiento de un sensor, cuyo Agente de Gestión establece los ciclos de reposo en función de lo que cada estimador dé como resultado. Esto se ha simulado usado la aplicación antes mencionada. La magnitud bajo estudio, en este caso la presión sonora, se simulará a través de una serie de bancos de funciones pseudo-aleatorias, las cuales serán pre-generadas y aportadas como entrada a los estimadores. Cada banco estará formado por treinta señales, para que así en todos los valores derivados de su uso se pueda asumir normalidad. De esta manera los dos sistemas (Diferencial y FRBS) se probarán bajo las exactas mismas condiciones, pudiendo realizar una caracterización estadística de los resultados obtenidos. Tanto el Sistema Diferencial como el basado en FRBS se ajustan a lo presentado en la Sección 5 del Capítulo II.

[108] Aunque será detallado más adelante, cada prueba implicaría un funcionamiento continuado e ininterrumpido del sensor durante 10 horas, esto supondría que para 30 experimentos se requerirían 12 días y medio, lo cual demoraría en demasía la capacidad de ajuste del modelo, así como la comprobación de errores, muy probables dada la duración del experimento.

[109] El número de experimentos que comúnmente permite asumir la normalidad del espacio de muestras es de 30, por lo que se ha elegido este número. Más adelante se harán consideraciones adicionales acerca de esta asunción.

[110] Estos problemas podrían ser la aparición de fallos no derivados del funcionamiento de los propios sistemas, tales como deterioro del hardware, diferencias en la carga de la batería o errores fruto de una manipulación no controlada, pudiéndose adulterar los resultados de las pruebas realizadas

9.1.1 SEÑALES DE ENTRADA

Con objeto de probar los sistemas bajo estudio en diferentes condiciones, se han realizado tres grupos de pruebas. Cada grupo incluye la evaluación con un banco de treinta señales de entrada, todas ellas diferentes entre sí, las cuales serán usadas para la estimación del SMI con el Sistema Diferencial y el FRBS, con ambas bases de conocimiento.

Las señales de entrada comparten entre sí los parámetros de la generación pseudo-aleatoria, pero difieren ligeramente con respecto a los otros dos bancos. Los tres tipos de señales, denominadas Tipo A, Tipo B y Tipo C, se han generado teniendo en cuenta una alta inercia, pero con la posibilidad de presentar fluctuaciones en intervalos suficientemente cortos. De esta manera podrán aparecer picos o derivas a valores críticos, que permitan poner a prueba la capacidad de estimación de los ciclos de reposo de los sistemas bajo estudio.

Estas señales se han modelado para que emulen el sonido producido por sistemas mecánicos en movimiento, tales como ventiladores, motores, cintas transportadores, los cuales, en condiciones normales de funcionamiento, mantienen un nivel de ruido controlado y dependiente de su régimen de trabajo. A pesar de que se han generado a través de software, los parámetros usados para su creación se han derivado de las mediciones realizadas del sonido de los ventiladores en equipos informáticos. Las medidas de referencia fueron tomadas con un sensor Sun SPOT con una circuitería adicional[111], la cual incluía un micrófono (véase la Figura 34).

Figura 34. Sensor Sun SPOT con micrófono.

Las condiciones bajo las que se generarán las señales de entrada serán las siguientes:

- Rango de valores entre 40 dBA y 90 dBA.
- Nivel medio del régimen normal de funcionamiento: entre los 60 dBA y 65 dBA.
- Cada señal estará formada por muestras tomadas cada segundo hasta completar una duración de 10 horas.

El algoritmo que genera las señales pseudo-aleatorias calcula nuevos valores de la señal para intervalos regulares de *P* segundos. Los valores intermedios de la señal entre los

[111] Este circuito, junto con el algoritmo para la medición de la presión sonora en un sensor Sun SPOT, forma parte del trabajo de investigación desarrollado por Juan Andrés Mariscal Ramírez (Mariscal-Ramirez et al. 2011) dentro del grupo de investigación de Ingeniería de Sistemas Telemáticos, como parte de su tesis doctoral.

puntos de cálculo se obtienen de la interpolación lineal entre dichos puntos. A continuación se presenta el algoritmo de cálculo:

```
i=0
m = 60
REPITE{
   SIs>SENTONCESn=m+(R*signo(i))
   SI NOn=m+(N*signo(i))
   SIn>TENTONCESn=T
   SIn<BENTONCESn=B
   G(i*P)=n
}MIENTRAS(i*P<t)
```

Donde:

- G, señal generada con el algoritmo.
- P, intervalo de tiempo entre las muestras generadas por el algoritmo.
- t, duración total de la señal a generar.
- m, valor resultado de la ejecución previa del algoritmo. Inicialmente su valor es 60 (dBA).
- Muestra (n), valor de la señal a calcular.
- Estabilidad de ciclo (s), valor aleatorio usado para determinar si la nueva muestra a calcular corresponde con una muestra normal (se usa el parámetro N) o inusual (cálculo con el parámetro R). Va de 0 a 1.
- Incremento (i), valor aleatorio para los incrementos en la magnitud. Va de -0,5 a 0,5.
- Base (B), valor mínimo de la señal (40 dBA).
- Tope (T), valor máximo de la señal (90 dBA).
- Incremento normal en dBA (N), valor máximo para las fluctuaciones de la señal cuando ésta se supone que están en un régimen de trabajo normal.
- Incremento inusual en dBA (R), valor máximo para las fluctuaciones de la señal cuando ésta está en un régimen de funcionamiento anómalo.
- Estabilidad (S), probabilidad de que la señal tenga en este ciclo un comportamiento normal. Valores próximos a uno darán señales con menor número de fluctuaciones anómalas (parámetro R).

Los valores aleatorios se han obtenido a través de la función aleatoria, disponible en el lenguaje de programación Java, usando una distribución de probabilidad uniforme.

Cada uno de los tres bancos de treinta señales se ha calculado aplicando al algoritmo los siguientes juegos de parámetros:

Tabla 40. Tipos de señales para las pruebas.

	N (dBA)	R (dBA)	Estabilidad	P (s)
Señales Tipo A	0,5	2,0	0,9	120
Señales Tipo B	0,2	2,0	0,9	25
Señales Tipo C	0,2	0,5	0,9	25

Así pues, de manera cualitativa, como puede verse en la Tabla 40, las señales Tipo A se presuponen con el comportamiento más inercial de los tres tipos, ya que no se calculará

un nuevo valor hasta transcurridos 120 segundos, si bien el nuevo valor puede llegar a ser de 2,0 dBA para las fluctuaciones inusuales. Sin embargo, tanto las señales Tipo B, como las Tipo C, presentan una menor inercia, al calcularse los nuevos valores cada 25 segundos. Las señales Tipo B, son las que presentan la posibilidad de una mayor variabilidad, ya que su parámetro R es de 2,0 dBA y se calculan cada 25 segundos. Por su lado, las Tipo C, a pesar de poder fluctuar cada 25 segundos, lo hacen a una tasa menor que las Tipo A (0,2 dBA) en condiciones normales, al igual que también, para las fluctuaciones inusuales tan sólo permiten incrementos de 0,5 dBA.

Tabla 41. Clasificación cualitativa de las señales para las pruebas.

Inercia*	**Variabilidad***
Tipo A	Tipo B
Tipo B y C	Tipo A
	Tico C

* Ordenadas cualitativamente de mayor a menor.

Como ejemplo de la generación de estas señales de prueba se presenta la Figura 35, donde aparecen los 480 primeros segundos de la primera señal de cada banco.

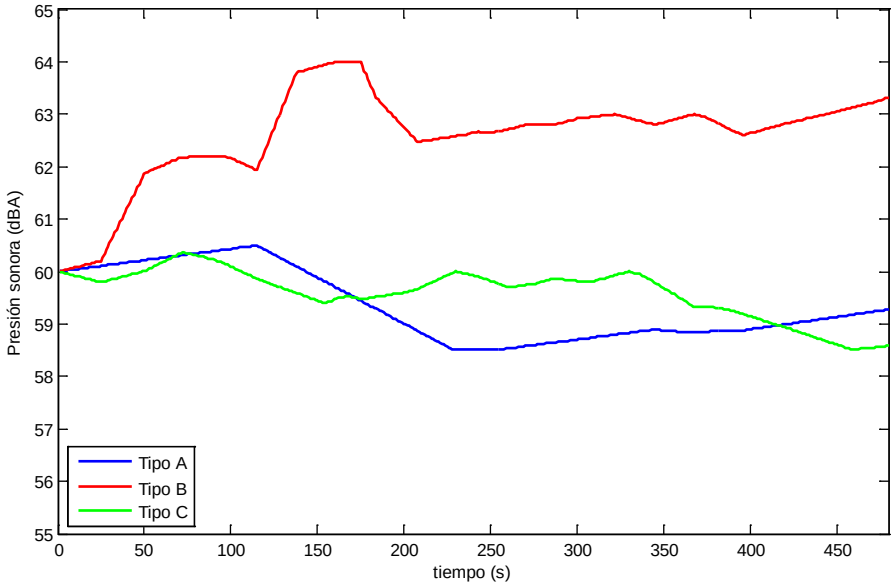


Figura 35. Ejemplo de la primera señal de cada banco para cada tipo (A, B y C).

9.1.2 PROTOCOLO DE PRUEBAS

Las pruebas realizadas a cada sistema (SD, FRBS con KB1 y FRBS con KB2) se dividieron en tres bloques, uno para cada tipo de señal de entrada pre-generada. Además, para cada tipo de señal se hicieron 6 simulaciones diferentes, cada una para una carga de batería diferente del sensor. Los valores de carga de batería que se eligieron son: 5%, 15%, 30%, 75%, 99% y 100%. El objeto de realizar las simulaciones para diferentes niveles de carga de la batería del sensor se debe a que, tanto del Sistema Diferencial, como el basado en FRBS, tienen como parámetro de entrada el nivel de carga de la batería, por lo que se hace necesario ver cómo se comportan ambos ante diferentes niveles de carga. Como puede observarse, los niveles están separados al

menos por un 10%[112]con objeto de que la simulación no tenga que tener en cuenta el consumo de batería, dado que este parámetro depende de un elevado número de factores. Así pues, se han elegido estos niveles para propiciar en ambos sistemas de estimación del SMI comportamientos diferentes que permitan comprobar su efectividad.

Así pues, cada prueba consiste en que cada sistema se pone a funcionar dentro del simulador tomando como entrada los valores de las señales pseudo-aleatorias pre-generadas, repitiendo este proceso para cada uno de los diferentes niveles de batería presentados.

El proceso de cálculo de las estimaciones es el siguiente:

1. Para el primer valor de entrada, y según las condiciones de carga de batería y valores iniciales, se estima un nuevo SMI.
2. Este nuevo SMI se toma como el nuevo tiempo de reposo que el Agente de Gestión fuera a usar.
3. En la siguiente ejecución del sistema por parte del simulador, se tomará como entrada el valor de la señal de presión sonora del instante correspondiente al SMI estimado en la iteración anterior (no se consideran tiempos de latencia en el reinicio del sensor, ya que a efectos prácticos no introducen ninguna diferencia), estimándose un nuevo valor del SMI.

Los pasos 2 y 3 se repiten hasta que no haya más muestras disponibles de la señal de entrada.

Como consecuencia de los pasos anteriores, se habrán tomado una serie de medidas puntuales de la magnitud bajo estudio, correspondientes a valores concretos de la señal de entrada. Estas medidas estarán separadas entre sí por los SMI estimados. Adicionalmente, y como consecuencia del propio proceso, se habrán obtenido el número total de ejecuciones empleadas por cada sistema bajo estudio.

Una vez terminado el proceso de estimación del SMI, a través de la interpolación lineal de los valores obtenidos en los puntos medida, se generará, para cada experimento, una señal, denominada Señal Interpolada Resultado (*SIR*). Estas señales corresponderían con lo que un sensor, controlado por el Agente de Gestión, habría medido si se hubiera colocado en el entorno, cuya magnitud a monitorizar, en este caso la presión sonora, evolucionara conforme a la señal pseudo-aleatoria de entrada.

Se hace evidente que el sensor habrá perdido información de la magnitud bajo estudio al pasar la mayor parte del tiempo en reposo. Sin embargo, es objeto de las presentes pruebas demostrar que, gracias a la gestión inteligente de este tiempo de sueño con los sistemas SD, y fundamentalmente con los FRBS, el error cometido al no poder leer la magnitud de manera continua, con respecto a sistemas de muestro a intervalos fijos, se reduce gracias a la estimación del SMI, y por lo tanto, se refrenda la Hipótesis tercera.

Como recuento total del número de experimentos, se debe tener en cuenta que habrá una señal *SIR* por cada señal de entrada pseudo-aleatoria de cada uno de los tres tipos (30 x 3), además de por cada sistema diferente (3 en total: SD, FRBS con KB1 y FRBS con

[112]Se han usado los niveles 99% y 100%,en primer lugar, para comprobar que ambos sistemas funcionan adecuadamente en los límites de un parámetro como es la carga de la batería, y en segundo lugar, se permite verificar que una mínima variación en la batería, propicia un comportamiento ligeramente diferente, pero predecible.

KB2) y por cada nivel de carga de batería empleado (6 niveles: 5%, 15%, 30%, 75%, 99% y 100%), lo cual arroja un total de 1620 experimentos, cada uno con su correspondiente *SIR*. Por lo tanto, dada la gran cantidad de señales, se usará una medida de error como elemento comparador, así como diferentes estadísticos para analizar y presentar toda la información producida por los experimentos. Así pues, en el siguiente apartado se presentará las medidas usadas para la comprobación de la eficacia de ambos métodos, seguido del análisis detallado de los resultados de las pruebas.

9.1.3 MEDIDA DEL ERROR

La medida que se ha usado para comprobar la eficacia de ambos métodos de estimación del SMI, que además permita realizar una comparación entre ambos, es el error cuadrático medio. La medida del error se realiza sobre la diferencia de cada señal de entrada y la *SIR* correspondiente. Dado que ambas señales están dentro del mismo rango de valores, previamente a la realización de la diferencia, se normalizan entre 0 y 1, tomando como valor de normalización los 90 dBA del valor máximo de las señales de entrada.

Con objeto de poder obtener una referencia de los niveles de error tolerables para los resultados de ambos sistemas, se han obtenido los datos de error para el caso de que los sensores tuvieran un ciclo de trabajo prefijado, es decir, que siempre estuvieran en reposo el mismo tiempo y una vez activos tomaran una medida, volviendo al estado de reposo seguidamente. A estos resultados se les ha denominado como los resultados del Sistema Continuo (SC), y se han usado para tiempos de reposo que van desde los veinte segundos a los cinco minutos[113]. Los resultados, en media, del error cuadrático medio obtenido para las treinta señales de cada tipo pueden verse en la Tabla 41.

Tabla 42. Media del error cuadrático medio obtenido con tiempos de reposo fijos.

Tipo de señal	20 s	40 s	150 s	300 s
A	0,0001	0,0008	0,0343	0,1703
B	0,0066	0,0414	0,4765	1,1478
C	0,0010	0,0061	0,0683	0,1661

Así pues, en el caso de que un sensor estuviera programado para estar en reposo por periodos de tiempo fijos, obtendría, en media, los valores de error presentados, los cuales serán usados como referencia en el análisis estadístico de la presente sección de resultados.

El número de ciclos (proceso y reposo) que realizaría un sensor para los tiempos de sueño de 20, 40, 150 y 300 segundos, serían 1800, 900, 240 y 120, respectivamente. Lo cual, como ya se ha visto en apartados anteriores, puede dar una idea aproximada del consumo de batería de cada uno. Así pues, una vez presentados los niveles de referencia, en el apartado siguiente se mostrará el estudio de consumo de batería realizado a cada método en sensores Sun SPOT.

[113]Se han elegido los 20 segundos ya que es el mínimo establecido, tanto para el SD como para ambos FRBS, como salida de su estimación, motivo por el cual se ha elegido, al igual que los cinco minutos, que coincide con la mitad del valor máximo establecido para la estimación.

9.2 ANÁLISIS DE LOS RESULTADOS

Como ya se ha comentado, la medida usada para comprobar la efectividad de los métodos de estimación del tiempo de sueño, es el error cuadrático medio aplicado a la diferencia entre cada señal pseudo-aleatoria que modela el entorno, y la señal reconstruida con los puntos de medida, denominada *SIR*. Sin embargo, dado que el error cuadrático medio se calculará para cada punto de la señal de entrada, y eso sería difícilmente representable, se ha optado por presentar la media de dicho error para cada tipo de señal, agrupando las medias de las treinta señales que forman parte del experimento. Así pues, se tendrá un valor de error cuadrático medio por cada sistema, tipo de señal y nivel de batería, como puede verse en la Tabla 42.

Tabla 43. Valor medio del error cuadrático de las pruebas realizadas agrupadas por tipo de señal.

		Nivel de carga de la batería					
Sistema	**Tipo de señal**	**5%**	**15%**	**30% s**	**75%**	**99%**	**100%**
SD	**A**	0,5487	0,5494	0,5423	0,5294	0,5062	0,5062
FRBS KB1	**A**	0,3483	0,3196	0,2834	0,2282	0,2062	0,2061
FRBS KB2	**A**	0,3907	0,3460	0,3280	0,2929	0,2755	0,2751
SD	**B**	2,4387	2,4849	2,3878	2,2767	1,9764	1,9764
FRBS KB1	**B**	1,7075	1,6029	1,4711	1,2596	1,1240	1,1265
FRBS KB2	**B**	1,8643	1,7469	1,6388	1,5102	1,3757	1,3784
SD	**C**	0,3614	0,3664	0,3642	0,3585	0,3408	0,3408
FRBS KB1	**C**	0,2595	0,2448	0,2269	0,2020	0,1802	0,1799
FRBS KB2	**C**	0,2829	0,2610	0,2484	0,2438	0,2320	0,2319

En la Tabla 43 aparecen reflejados los ciclos de trabajo y sueño empleados por los sistemas durante el proceso de simulación. En concreto, el valor que se muestra es la media obtenida en para los treinta experimentos de cada tipo.

Tabla 44. Ciclos de ejecución para la estimación del SMI con el SD y FRBS con ambas bases de conocimiento.

		Nivel de carga de la batería					
Sistema	**Tipo de señal**	**5%**	**15%**	**30%**	**75%**	**99%**	**100%**
SD	**A**	64	65	67	70	77	77
FRBS KB1	**A**	82	88	93	106	115	115
FRBS KB2	**A**	78	83	87	94	99	99
SD	**B**	64	65	69	77	92	92
FRBS KB1	**B**	86	92	99	116	134	134
FRBS KB2	**B**	78	84	89	97	105	105
SD	**C**	64	65	68	71	77	77
FRBS KB1	**C**	81	86	93	107	114	114
FRBS KB2	**C**	78	82	85	89	92	92

Los valores presentes en la tabla aparecen redondeados.

Puede observarse que en casi todos los casos se está por debajo de las 120 ejecuciones que correspondería al sistema continuo con tiempo de sueño fijo de 300 s.

Dada la cantidad de datos presentada, así como su heterogeneidad, a continuación se presenta un análisis estadístico exhaustivo que muestra el grado de mejora que implica el uso de cada uno de los sistemas de estimación, con respecto a un muestreo continuo de la magnitud bajo estudio. Todo esto teniendo en cuenta que se prioriza el hecho de que un sistema ahorre energía, es decir, esté más tiempo en reposo, a que obtenga más cantidad de medidas, siempre y cuando el error ocasionado en la magnitud observada (señales SIR) sea asumible.

En este caso, al igual que en el apartado 8.3, se ha elegido como herramienta estadística el análisis de las varianzas o ANOVA de una vía. En concreto de han cotejado las varianzas del error cuadrático medio obtenido, en función del método elegido (SD, FRBS con KB1 y FRBS con KB2) y de seis niveles de carga de batería diferentes (5%, 15%, 30%, 75%, 99% y 100%). El análisis de los requisitos que deben cumplir los conjuntos de muestras se detallan a continuación:

- **Independencia:** las muestras son independientes, cumpliéndose en este caso ya que los valores a contrastar son el fruto de la aplicación de tres métodos diferentes e independientes entre sí.
- **Normalidad:** se ha elegido un número suficiente de valores, como ya se ha comentado, para propiciar la normalidad del conjunto de muestras, a la par que se ha realizado el test de Shapiro-Wilk para cada uno, obteniendo un resultado satisfactorio en todos ellos.
- **Homocedasticidad:** se ha empleado el test de Cocrhan en el presente análisis para todos los conjuntos de muestras, dando resultados satisfactorios en todos los casos.

Los resultados que se prevén obtener deberían justificar el uso de alguno de los sistemas propuestos en esta tesis, a través de la aparición de casos que mejoren significativamente a un sistema estático.

9.2.1 CASO 1. ANOVA PARA LA COMPARACIÓN DE LAS MEDIAS DE ERROR EN FUNCIÓN DEL TIPO DE ESTIMADOR EMPLEADO SIN DISTINCIÓN POR NIVEL DE CARGA DE LA BATERÍA.

En este caso, se plantea la siguiente hipótesis nula: sea cual sea el método empleado no se encontrarán diferencias significativas en la medias del error cuadrático medio calculado. O lo que es igual, formalmente:

H_0: $\mu_{SD} = \mu_{FRBS\ KB1} = \mu_{FRBS\ KB2}$

H_1: alguna μ_i es distinta.

Así pues, se ha aplicado la técnica ANOVA para cada grupo de señales del mismo tipo, buscando la presencia de medias significativamente diferentes entre sí con respecto al método usado para la estimación del tiempo de sueño. Los resultados obtenidos aparecen reflejados en la Tabla 44.

Como puede observar el factor P es menor de 0,05, lo cual nos permite rechazar la hipótesis nula para los tres tipos de señales. Sin embargo, esto tan solo nos confirma que al menos una de las medias de todos los casos es significativamente diferente. Para contrastar qué estimador de SMI es el que aporta una media significativamente diferente, así como en qué sentido es esta diferencia, se ha practicado a los resultados

del método ANOVA el test de honestidad de Tukey, cuyos resultados aparecen en la Tabla 45.

Tabla 45. Resultados del error cuadrático medio en función del estimador de SMI usado.

Tipo de señales		Suma de Cuadrados	gl	Media Cuadrática	F	P
A	**Estimador SMI**	7,086	2	3,5430	606,9	<2e−16
	Residuos	3,135	537	0,0060		
B	**Estimador SMI**	75,52	2	37,7600	330,3	<2e−16
	Residuos	61,40	537	0,1100		
C	**Estimador SMI**	1,9096	2	0,9548	590,9	<2e−16
	Residuos	0,8677	537	0,0016		

Tabla 46. Resultados del test HSD de Tukey para el 95% de los intervalos de confianza para cada tipo de señal agrupados por el estimador SMI usado.

Tipo de señal	Estimador SMI	Diferencia	Límite inferior	Límite superior	P ajustado
A	**FRBS KB1-SD**	−0,26505481	−0,28398367	−0,24612595	0
	FRBS KB2-SD	−0,21227714	−0,23120600	−0,19334828	0
	FRBS KB2-FRBS KB1	0,05277767	0,03384881	0,07170653	0
B	**FRBS KB1-SD**	−0,8753200	−0,9590903	−0,7915498	0
	FRBS KB2-SD	−0,6715803	−0,7553505	−0,5878100	0
	FRBS KB2-FRBS KB1	0,2037398	0,1199695	0,2875100	1e−07
C	**FRBS KB1-SD**	−0,13979145	−0,14975004	−0,12983287	0
	FRBS KB2-SD	−0,10535054	−0,11530913	−0,09539196	0
	FRBS KB2-FRBS KB1	0,03444091	0,02448233	0,04439950	0

Como se puede apreciar, en todos los casos en los que se compara los sistemas SD con los FRBS, estos obtienen una menor media de error, en parte debido al menor número de ciclos del SD.

Con respecto a la comparación entre las dos bases de conocimiento usadas por el FRBS, queda patente que los resultados de la estimación de tiempos de sueño con la KB2 generan un error mayor. Esta diferencia, al igual que la de los FRBS con el SD, se debe en principio al número de ciclos empleado por cada uno. Aunque se explicará convenientemente en las conclusiones de esta sección de pruebas, se puede adelantar que, en igualdad de ciclos, los sistemas FRBS pueden alcanzar un menor error al ajustar de una manera más precisa los ciclos de sueño a la tendencia de la señal.

Así pues, y como resumen de este primer análisis, se extrae que el sistema que consigue un menor error, a costa de un mayor número de ciclos de trabajo, es el FRBS con la

base de conocimiento KB1. Mientras que el SD, con su menor número de ciclos arroja un mayor error medio, quedando por último el FRBS con la base de conocimiento KB2 en un lugar intermedio.

9.2.2 CASO 2. ANOVA PARA LA COMPARACIÓN DE LAS MEDIAS DE ERROR DE LOS ESTIMADORES EN CONJUNTO EN FUNCIÓN DEL NIVEL DE CARGA DE LA BATERÍA.

En el caso de la comparación de las medias, para cada nivel de batería usado en la simulación, la hipótesis nula planteada es la siguiente: sea cual sea el valor del nivel de batería, no se encontrarán diferencias significativas en la medias del error cuadrático medio calculado[114]. O lo que es igual, formalmente:

H_0: $\mu_{100\%} = \mu_{99\%} = \mu_{75\%} = \mu_{30\%} = \mu_{15\%} = \mu_{5\%}$

H_1: alguna μ_i es distinta.

Al igual que el caso anterior, se ha aplicado un ANOVA para cada grupo de señales del mismo tipo, buscando la presencia de medias significativamente diferentes entre sí con respecto al método usado para la estimación del tiempo se sueño. Los resultados obtenidos aparecen reflejados en la Tabla 46.

Tabla 47. Resultados del error cuadrático medio en función del estimador de SMI usado.

Tipo de señales		Suma de Cuadrados	gl	Media Cuadrática	F	P
A	**Estimador SMI**	0,7780	5	0,15565	8,802	4,87e−08
	Residuos	9,4430	534	0,01768		
B	**Estimador SMI**	22,1700	5	4,43300	20,63	<2e−16
	Residuos	114,7600	534	0,21500		
C	**Estimador SMI**	0,1941	5	0,03881	8,023	2,62e−07
	Residuos	2,5833	534	0,00484		

Como se deduce de la tabla anterior, al igual que en el Caso 1, los resultados del ANOVA muestran que existen diferencias significativas en al menos una de medias. Si bien ahora, la cantidad de combinaciones posibles en las comparaciones es sensiblemente mayor, por lo que se hace aún más necesario el test HSD de Tukey, así como un análisis pormenorizado del mismo.

Los resultados del test HSD aparecen resumidos en la Tabla 47, donde para cada tipo de señal se recogen en primer lugar las medias que, comparadas entre sí, ofrecen una diferencia significativa. En una línea al final de cada bloque de señales, aparecen resumidos los valores de P ajustado para los casos en los que no existen diferencias significativas. Para identificar cada nivel de carga de batería se ha usado la letra ‘B’, seguida los dígitos del nivel de carga que representan. Las comparaciones entre dos niveles aparecen reflejadas como ambos indicadores separados por un guion.

[114] Al igual que el caso anterior se asume la normalidad del conjunto de muestras usado.

Se debe hacer notar, que en este caso, los resultados que relacionan cada grupo de medias agrupan a los tres métodos de estimación. Por lo tanto, la información que se deriva de este análisis es, que en conjunto, tan solo cuando la diferencia en los niveles de batería supera el 70% (señales tipo A y C) y el 60% (señales tipo B), los estimadores arrojan diferencias significativas en las medias del error cuadrático medio.

Tabla 48. Resultados del test HSD de Tukey para el 95% de los intervalos de confianza para cada tipo de señal agrupados por los niveles de batería usados en la estimación del SMI.

Tipo de señal	Comparación entre niveles de batería	Diferencia	Límite inferior	Límite superior	P ajustado
A*	B100-B05	−0,09994576	−0,15664227	−0,04324925	0,0000094
	B75-B05	−0,07908576	−0,13578227	−0,02238925	0,0010595
	B15-B100	0,07574261	0,01904610	0,13243912	0,0020514
	B15-B05 (0,82), B30-B05 (0,21), B30-B100 (0,06), B75-B100 (0,89), B30-B15 (0,90), B75-B15 (0,06) y B75-B30 (0,50)				
B*	B100 − B05	−0,5114560	−0,7091050	−0,3138070	0,0000000
	B75 − B05	−0,3213377	−0,5189867	−0,1236887	0,0000613
	B15 − B100	0,4528874	0,2552384	0,6505364	0,0000000
	B30 − B100	0,3405021	0,1428531	0,5381511	0,0000164
	B75 − B15	−0,2627691	−0,4604181	−0,0651201	0,0022015
	B15-B05 (0,95), B30-B05 (0,13), B75-B100 (0,06), B30-B15 (0,58) y B75-B30 (0,25)				
C*	B100-B05	−0,0504013	−0,0800557	−0,0207469	0,0000227
	B75-B05	−0,0331810	−0,0628353	−0,0035266	0,0181305
	B15-B100	0,0398462	0,0101918	0,0695006	0,0018836
	B15-B05 (0,91), B30-B05 (0,30), B30-B100 (0,06), B75-B100 (0,55), B30-B15 (0,9), B75-B15 (0,24) y B75-B30 (0,86)				

* Con objeto de simplificar la tabla, se han omitido las comparaciones con el nivel de carga del 99% dada la gran similitud, en este caso, con el nivel de carga de 100%

Dado que el análisis en conjunto aporta tan solo una información parcial, a continuación se presentan una serie de tres figuras compuestas de seis gráficas cada una, donde se pueden observar las diferencias (significativas o no) de las medias del error obtenido para las combinaciones, dos a dos, de cada uno de los estimadores propuestos.

En cada gráfica de dichas figuras aparecen los tres intervalos de confianza de la comparación de los resultados para un solo nivel de carga de batería a través de la realización de un ANOVA[115] y un test HSD entre los tres estimadores. El objeto de mostrar estas gráficas es para que, de una manera cualitativa más que cuantitativa, se pueda comprobar que en la mayoría de los casos la estimación del SMI con ambos sistemas FRBS, se arrojan unas medias de error significativamente diferentes, salvo en algunos casos con carga de batería baja, donde no se puede hablar de diferencias

[115]La hipótesis nula, cuya formulación formal se omite por ser equivalente a la ya vista para el Caso 1, supondría que no hay diferencia entre las medias de error de cada estimador para un nivel concreto de batería.

significativas entre la estimación usando FRBS y ambas bases de conocimiento (estos casos aparecerán marcados en cada gráfica de las figuras con una flecha).

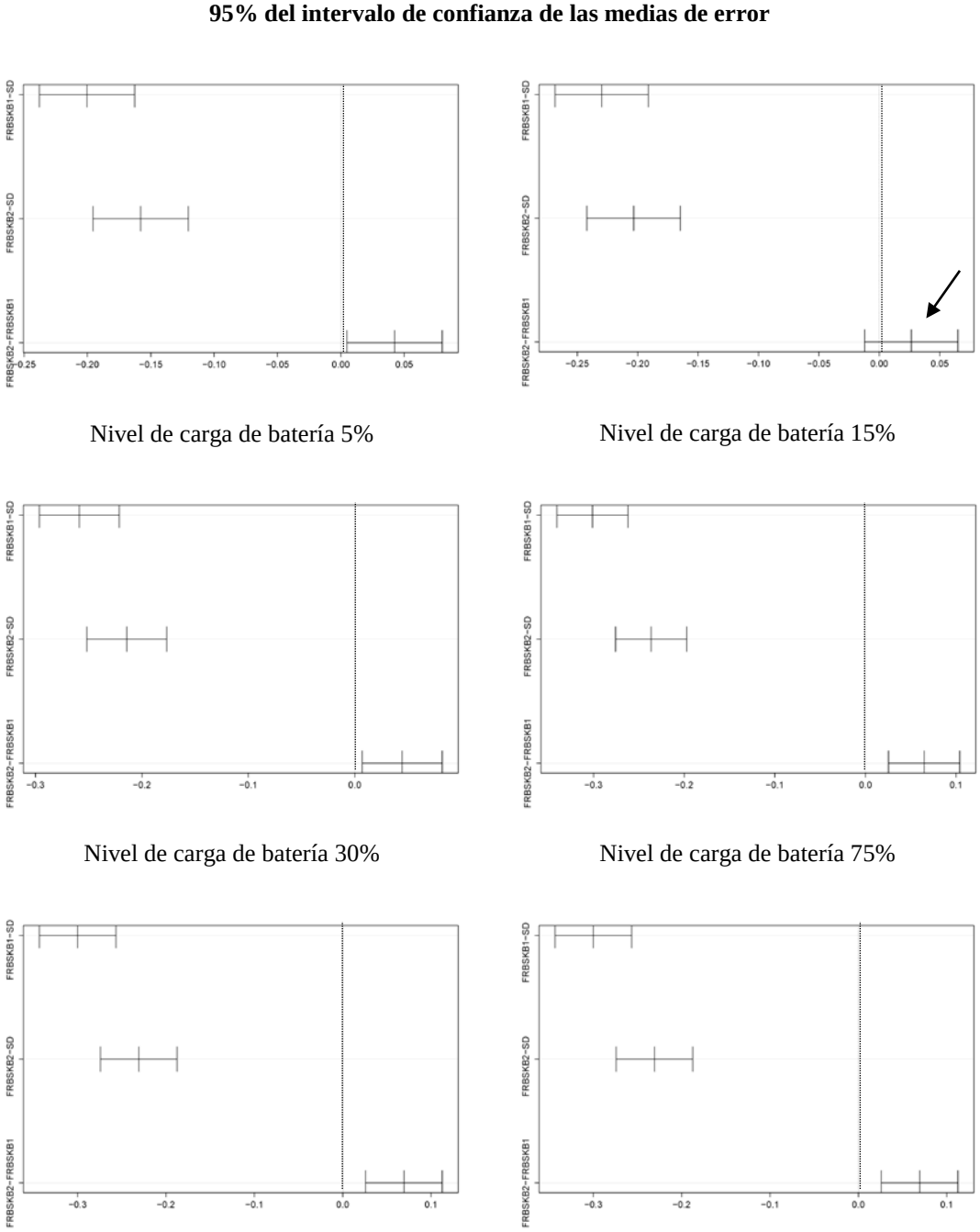
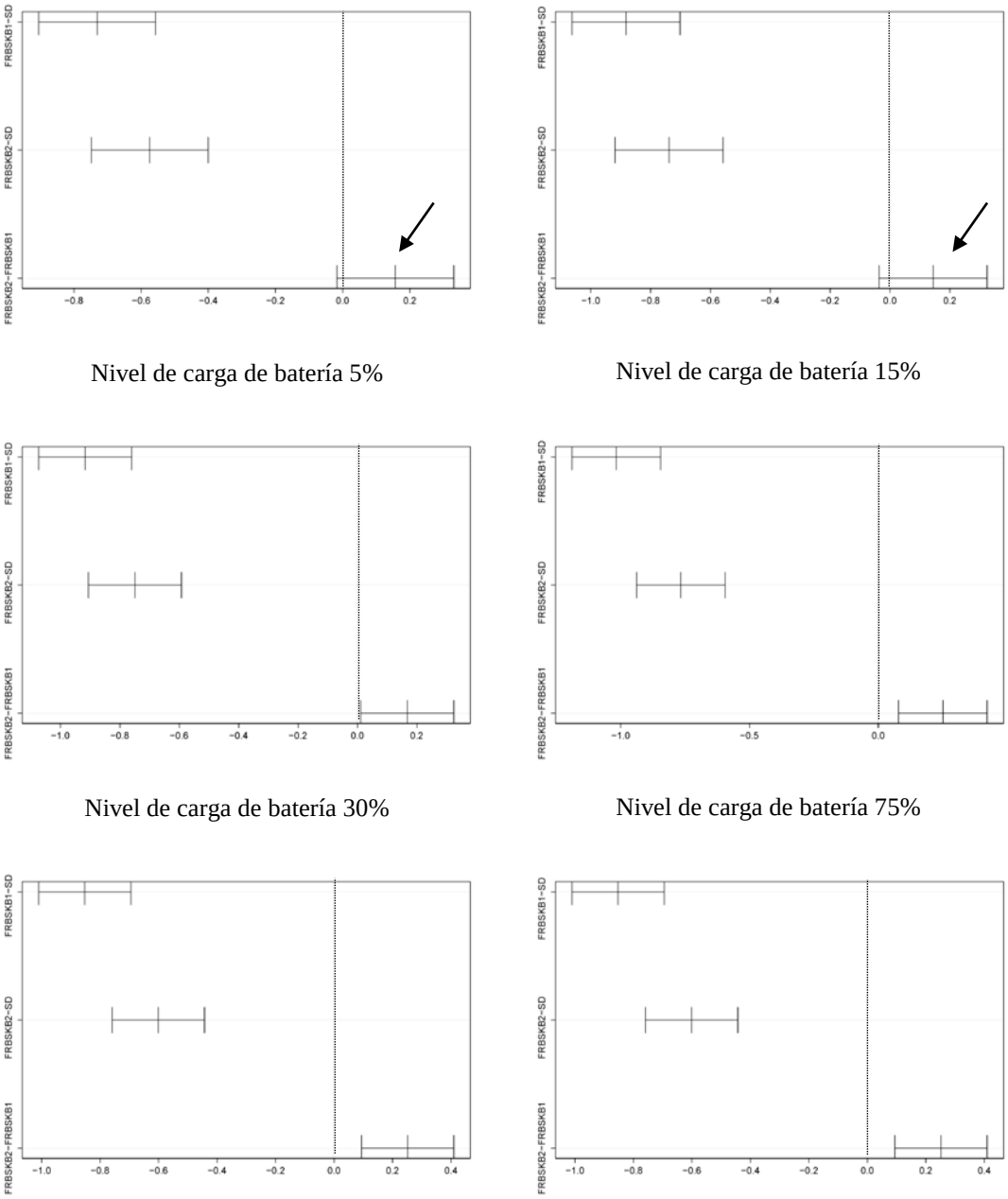


Figura 36. Intervalo de confianza (95%) de las medias de error de cada estimador para cada nivel de carga de batería para las señales tipo A.

95% del intervalo de confianza de las medias de error



Nivel de carga de batería 5%

Nivel de carga de batería 15%

Nivel de carga de batería 30%

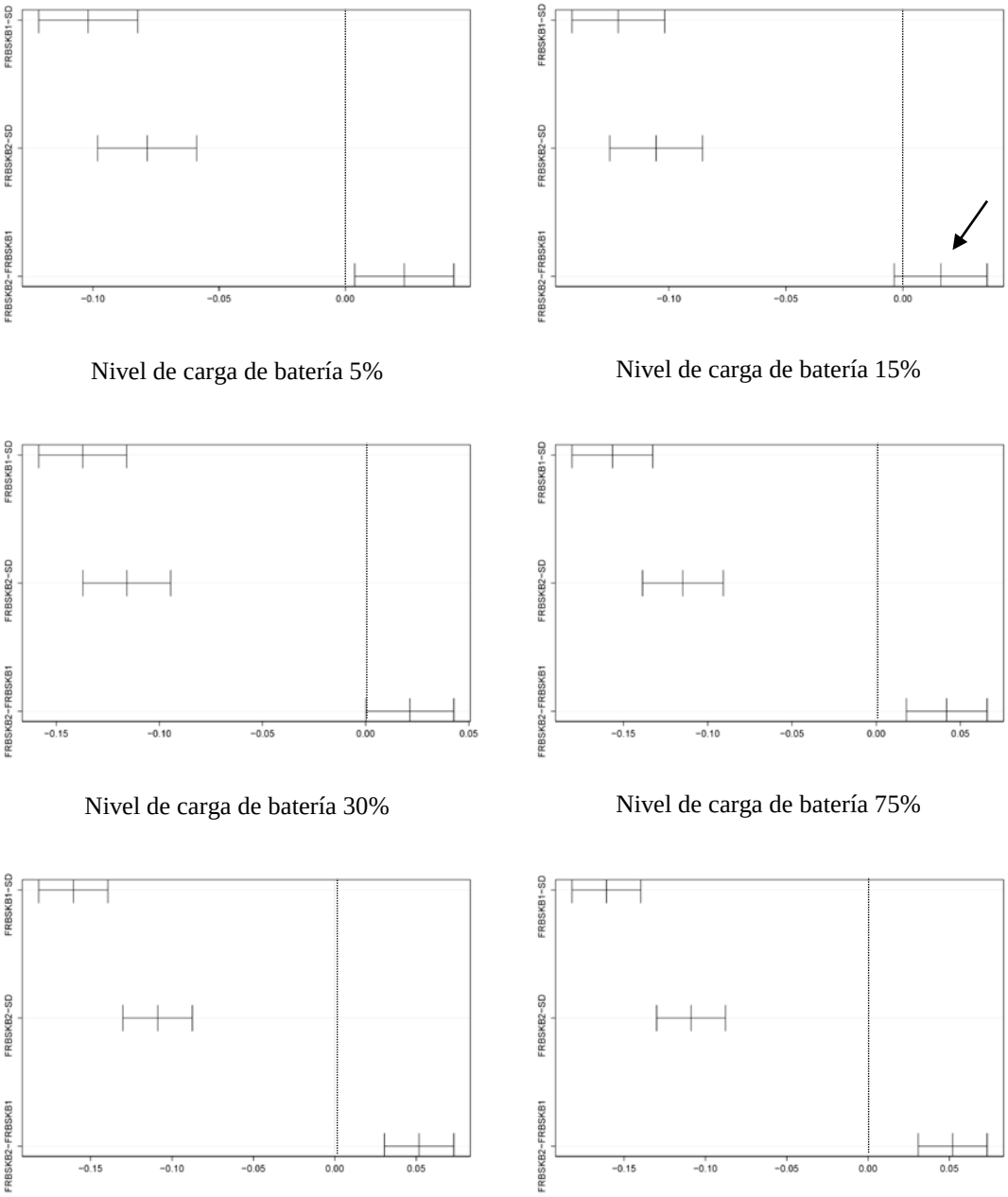
Nivel de carga de batería 75%

Nivel de carga de batería 99%

Nivel de carga de batería 100%

Figura 37. Intervalo de confianza (95%) de las medias de error de cada estimador para cada nivel de carga de batería para las señales tipo B.

95% del intervalo de confianza de las medias de error



Nivel de carga de batería 5%

Nivel de carga de batería 15%

Nivel de carga de batería 30%

Nivel de carga de batería 75%

Nivel de carga de batería 99%

Nivel de carga de batería 100%

Figura 38. Intervalo de confianza (95%) de las medias de error de cada estimador para cada nivel de carga de batería para las señales tipo C.

Para finalizar con el análisis de este caso, se deben destacar los dos aspectos vistos en el mismo. En primer lugar, si se analizan los resultados de los tres estimadores en conjunto en función del nivel de batería, no se obtienen resultados significativos salvo para diferencias grandes (60-70%) del nivel de batería. Dado que esto no ofrece un resultado concluyente, se hace necesario el segundo análisis mostrado previamente. Así pues, cuando se analiza cada estimador en función de un nivel de batería concreto, se

consiguen resultados significativamente diferentes de cada uno con respecto a los demás, incluso entre la aplicación de ambos FRBS (salvo para cuatro casos concretos, como se pueden ver en las Figuras 36, 37 y 38.

Por lo tanto, y como conclusión de este caso, se puede decir que cada estimador tiene comportamientos significativamente diferentes con respecto a los demás, en menor grado si se miran todos los niveles de batería en su conjunto, pero muy claros cuando se analizan estos para un nivel de batería en concreto.

9.2.3 CASO 3. ANOVA PARA LA COMPARACIÓN DE LAS MEDIAS DE ERROR PARA CADA ESTIMADOR POR SEPARADO EN FUNCIÓN DEL NIVEL DE CARGA DE LA BATERÍA.

Una vez analizada la influencia de los niveles de carga de batería para los estimadores en su conjunto, a continuación se presentan los resultados de la aplicación de un ANOVA a las medias de error obtenidas para **cada estimador por separado**, en función del nivel de batería. Así pues, la hipótesis nula planteada será igual para cada caso: no existen diferencias significativas entre las medias de error resultado de la aplicación de cada sistema estimador con un nivel de carga de batería diferente. La formulación formal sería equivalente a la mostrada para el Caso 2, aplicándola a cada estimador por separado (SD, FRBS con KB1 y FRBS con KB2).

El resultado del ANOVA para cada estimador aparece resumido en la Tabla 48, en la que solo se muestra el factor de significancia *P*. Como puede observarse, el sistema SD tan solo consigue diferencias significativas entre las medias de error para las señales tipo B, mientras que el sistema FRBS consigue diferencias significativas para todos los tipos de señales y ambas bases de conocimiento.

Tabla 49. Resultados de los ANOVA para cada estimador por separado en función del nivel de batería.

	P		
Estimador	**Señales tipo A**	**Señales tipo B**	**Señales tipo C**
FRBS con KB1	<2e−16	<2e−16	<2e−16
FRBS con KB2	1,78e−14	1,78e−15	5,03e−10
SD	0,262	5,9e−07	0,204

Siguiendo con el esquema de los casos anteriores y así poder evaluar con mayor precisión los resultados obtenidos, se ha realizado el test HSD de Tukey a cada uno de los ANOVA objeto de ese caso. Los resultados de los tres test HSD se muestran en conjunto en la Tabla 49.

Como puede observarse, el estimador que proporciona un comportamiento más adaptativo en función de la carga de la batería es el FRBS con la base de conocimiento KB1, el cual, para los tres tipos de señales y las 10 combinaciones de carga de batería presenta resultados significativos en 24 de las 30 posibilidades. Le sigue el sistema FRBS con la base KB2, el cual, dado que es más conservador en batería y se ejecuta menos ciclos consigue una menor adaptabilidad para cargas medias de batería (consigue diferencias significativas para el 50% de los casos). Comparando los resultados de ambos sistemas FRBS, se puede apreciar que mientras que los resultados son similares para las señales tipo A y B, para las señales tipo C la base de conocimiento KB1 consigue mejores resultados.

Tabla 50. Resultados del test HSD de Tukey para cada estimador y grupo de señales.

		Comparación entre los niveles de batería*				
Estimador	**Tipo de señal**	**B100-B05**	**B15-B05**	**B30-B05**	**B75-B05**	**B15-B100**
FRBS (KB1)	**A**	0,000**	0,119	0,000**	0,000**	0,000**
	B	0,000**	0,378	0,001**	0,000**	0,000**
	C	0,000**	0,216	0,000**	0,000**	0,000**
FRBS (KB2)	**A**	0,000**	0,011**	0,000**	0,000**	0,000**
	B	0,000**	0,211	0,001**	0,000**	0,000**
	C	0,000**	0,022**	0,000**	0,000**	0,001**
SD	**A**	0,314	1,000	0,998	0,907	0,297
	B	0,000**	0,987	0,982	0,407	0,000**
	C	0,406	0,993	0,999	0,999	0,195
Estimador	**Tipode señal**	**B30-B100**	**B75-B100**	**B30-B15**	**B75-B15**	**B75-B30**
FRBS (KB1)	**A**	0,000**	0,349	0,023**	0,000**	0,000**
	B	0,000**	0,141	0,162	0,000**	0,003**
	C	0,000**	0,015**	0,079	0,000**	0,004**
FRBS (KB2)	**A**	0,001**	0,699	0,671	0,001**	0,076
	B	0,000**	0,109	0,287	0,000**	0,138
	C	0,156	0,469	0,407	0,125	0,969
SD	**A**	0,483	0,834	0,998	0,895	0,978
	B	0,000**	0,013**	0,832	0,168	0,752
	C	0,278	0,559	1,000	0,963	0,989

* Con objeto de simplificar la tabla, se han omitido las comparaciones con el nivel de carga del 99% dada la gran similitud, en este caso, con el nivel de carga de 100%.

** Resultado significativo.

Por último, el sistema SD presenta la menor adaptabilidad con la carga de batería. Este hecho se deriva en parte de la menor capacidad para adaptarse al entorno, así como de la complejidad de modelar una magnitud real a través de un sistema analítico. En particular consigue tan solo 4 casos en los que la diferencia entre las medias de error es significativamente distinta.

9.2.4 ANÁLISIS DE LAS PRUEBAS REALIZADAS

Los resultados mostrados en los tres casos anteriores ponen de manifiesto que los sistemas bajo estudio consiguen resultados diferenciados, cumpliendo además eficazmente con su cometido. El Sistema Diferencial, dada su propio diseño, se adapta con más lentitud a la señal, pero con un menor gasto de energía derivado del menor número de ciclos resultante. El estimador FRBS, con la base de conocimiento KB1, es el que consigue el error más bajo dado que el diseño de la base de conocimiento prima más la precisión con respecto al consumo, al contrario que la KB2, diseñada para ser más conservadora en el gasto de energía. Por lo tanto, el número de ciclos de trabajo obtenido con la aplicación de los estimadores del SMI basados en FRBS para el Agente

de Gestión puede ser elegido entre dos soluciones, ambas eficaces. Además, gracias a la arquitectura de los sensores, y del diseño de las bases de conocimiento, un sensor podría alternar entre ambas configuraciones a través de una acción de gestión externa, de una manera rápida y sencilla.

Sin embargo, el análisis de los resultados no estaría completo sin comparar el error obtenido entre todos los sistemas en los casos en los que el número de ciclos sea similar. De esta manera, se podrá contrastar si algún sistema consigue un menor error con un número de puntos de medida igual o similar, esperándose en ese caso que sean los sistemas basados en FRBS los que consigan un mejor resultado. Para ellos se presenta la Tabla 50, la cual agrupa la información del error y ciclos medios de trabajo obtenidos para los sistemas bajo estudio (Tabla 42 y Tabla 43), así como para el Sistema Continuo de 300 s (se muestran tan solo los datos para la carga de batería donde se pueden realizar comparaciones dada la similitud de los ciclos de trabajo).

Tabla 51. Comparativa entre error y ciclos de trabajo.

		Carga de batería							
	Tipo	**5%**		**15%**		**99%**		**100%**	
Sistema	**de señal**	**Ciclos**	**Error**	**Ciclos**	**Error**	**Ciclos**	**Error**	**Ciclos**	**Error**
SD	A	64	0,5487	65	0,5494	**77**	**0,5062 b**	**77**	**0,5062 b**
FRBS KB1	A	82	0,3483	88	0,3196	115	0,2062	115	0,2061
FRBS KB2	A	**78**	**0,3907 a**	83	0,3460	99	0,2755	99	0,2751
SC 300 s	A	120	0,1703	120	0,1703	120	0,1703	120	0,1703
SD	B	64	2,4387	65	2,4849	**92**	**1,9764 b**	**92**	**1,9764 b**
FRBS KB1	B	**86**	**1,7075 a**	**92**	**1,6029 a**	134	1,1240	134	1,1265
FRBS KB2	B	**78**	**1,8643 a**	**84**	**1,7469 a**	105	1,3757	105	1,3784
SC 300 s	B	120	1,1478	120	1,1478	120	1,1478	120	1,1478
SD	C	64	0,3614	65	0,3664	**77**	**0,3408 b**	**77**	**0,3408 b**
FRBS KB1	C	81	0,2595	86	0,2448	114	0,1802	114	0,1799
FRBS KB2	C	**78**	**0,2829 a**	82	0,2610	92	0,2320	92	0,2319
SC 300 s	C	120	0,1661	120	0,1661	120	0,1661	120	0,1661

a-b: Pares de valores de error comparados para cada bloque del mismo tipo de señal con valores de ciclos similares o menores para los estimadores basados en FRBS.

Como se puede apreciar, existen casos en los que los estimadores basados en FRBS consiguen un menor valor de error que el SD incluso con un número similar o incluso menor de ciclos. Estos casos aparecen marcados en la Tabla 50, como es el caso de las señales Tipo A donde el SD con 77 ciclos consigue un error de 0,5062, mientras que el FRBS con la KB2 para 78 ciclos da un error de tan solo 0,3907. Aunque es mucho más claro el caso de las señales Tipo B, donde el SD en 92 ciclos arroja un error de 1,9764, mientras que el FRBS con la KB1 el error es tan solo 1,6029 con los mismos ciclos, siendo aún mayor la mejora obtenida con el FRBS con KB2 que con tan solo 84 ciclos mejora al SD con un error de 1,7469. Estos resultados ponen de manifiesto la mejor capacidad de adaptabilidad de los FRBS con respecto a sistemas analíticos de igual complejidad (esto podrá verse en el apartado siguiente).

También se puede apreciar que los valores de error del sistema continuo son menores que los demás. Esto se debe a que el tiempo usado para la generación de las señales de entrada es un submúltiplo del tiempo de ciclo usado para el SC. Por lo tanto, la

simulación de la toma de medidas de este sistema hace que el sensor obtenga la medida justo en puntos donde se ha generado la señal, consiguiendo una buena fidelidad con la señal original.

9.2.5 ESTUDIO DEL CONSUMO DE LA BATERÍA

Con objeto de acotar el consumo de energía provocado en un sensor por la ejecución de cada método, se han ejecutado el Sistema Diferencial, el motor de inferencia con ambas bases de conocimiento (KB1 y KB2) y el sistema de muestreo continuo en pruebas de 1000 iteraciones para que el nivel de batería descendiera una cantidad medible. Estas pruebas se han repetido 30 veces para cada estimador consiguiéndose los valores de la Tabla 51.

Tabla 52. Consumo medio y tiempo de proceso de cada sistema evaluado.

	SC	SD	FRBS (KB1)	FRBS (KB2)
Consumo (mAh)	0,02135	0,02137	0,02165	0,02159
Tiempo de proceso (ms)	727,11	730,83	737,60	737,29

Como se puede apreciar, los valores obtenidos para ambas magnitudes son muy similares, debido en parte al consumo derivado del mantenimiento propio del sensor. Pero para poder realizar un contraste estadístico de los mismos, se ha realizado un ANOVA entre las medias obtenidas para ambas magnitudes, mostrándose el resultado en la Tabla 52. En estos casos ambas hipótesis nulas plantean que no habrá diferencias significativas para el consumo de batería o tiempo de proceso en función del tipo de estimador o con el sistema continuo. Formalmente:

H_0: $\mu_{SD} = \mu_{FRBS\ KB1} = \mu_{FRBS\ KB2} = \mu_{SC}$

H_1: alguna μ_i es distinta.

Particularmente, en este caso el objetivo es que no se pueda rechazar la hipótesis nula, lo cual permitiría afirmar que no existen diferencias significativas entre los valores de consumo y tiempo de proceso obtenidos.

Tabla 53. Resultados de los ANOVA para el consumo de energía y tiempo de proceso.

		Suma de Cuadrados	gl	Media Cuadrática	F	*P*
Consumo de energía	**Estimador**	1,620e−06	2	8,089e−07	1,281	0,282
	Residuos	6,442e−05	102	6,316e−07		
Proceso	**Estimador**	1169	2	584,3	1,181	0,311
	Residuos	50,464	102	494,7		

Así pues, una vez vistos los resultados de la tabla anterior, se puede afirmar que no es posible descartar la hipótesis nula, por lo que no existen diferencias significativas en la aplicación de un método de estimación con respecto al otro, incluido el Sistema Continuo.

Por lo tanto, las mejoras introducidas por los sistemas FRBS están completamente justificadas frente a sistemas de medición continua, ya que conseguirán resultados de

error asumibles, a la par que el consumo adicional introducido en el proceso no es significativo. Además, como consecuencia, un menor número de ciclos de trabajo aumentará la duración de la carga de la batería, su vida, y por tanto el tiempo de trabajo del sensor.

Como muestra del modo de trabajo de los estimadores de SMI vistos y del sistema continuo, se presentan a continuación tres gráficas, una para cada tipo de señales usadas (A, B o C), en las que aparece una de las señales de dicho tipo, con las interpolaciones generadas por los puntos de medida de cada estimador (señales SIR).

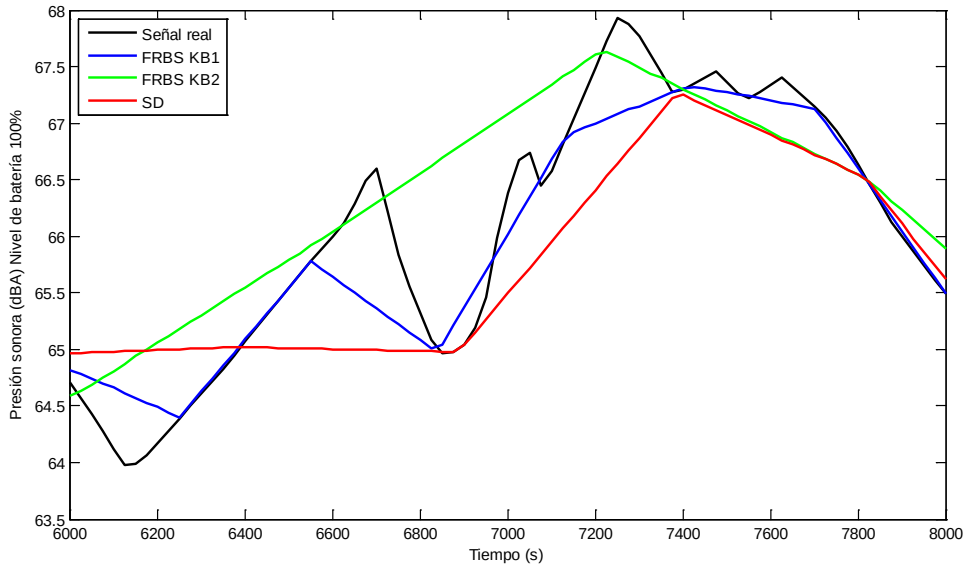


Figura 39. Ejemplo de señal Tipo A con las SIR para cada estimador usado.

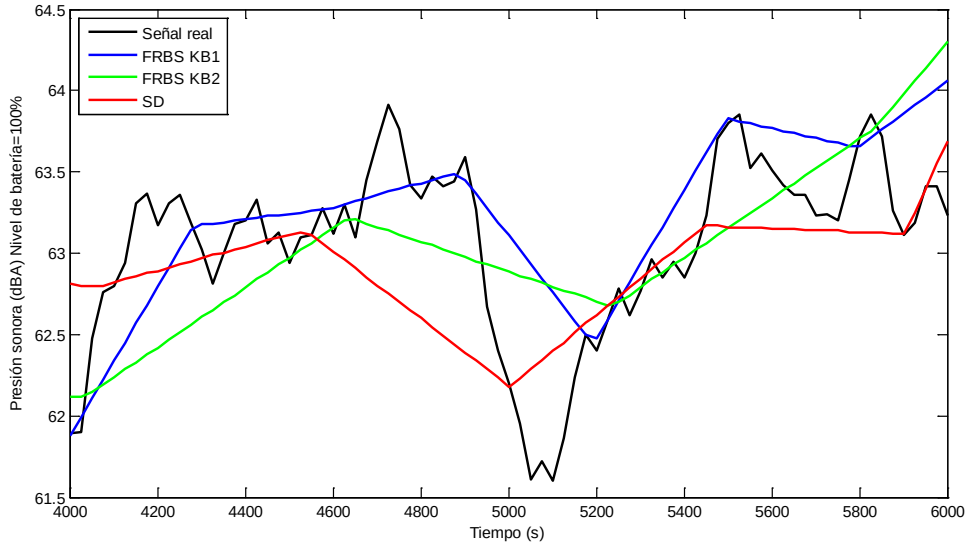


Figura 40. Ejemplo de señal Tipo B con las SIR para cada estimador usado.

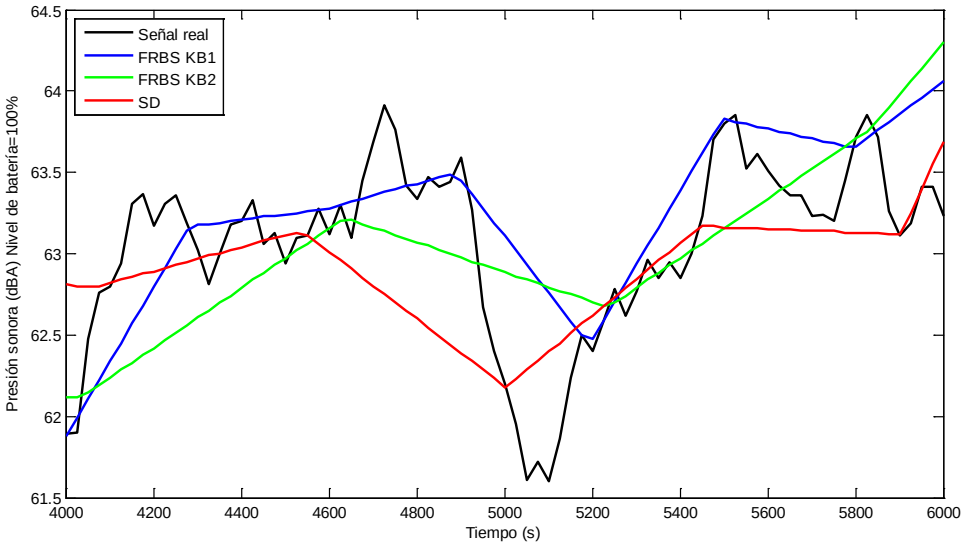


Figura 41. Ejemplo de señal Tipo C con las SIR para cada estimador usado.

Cualitativamente se puede apreciar como ambos sistemas FRBS consiguen adaptarse mejor a la señal que modela la magnitud bajo estudio, sin más información que las medidas y resultados anteriores, claro está, apoyándose en sendas bases de conocimiento.

Esto pone de manifiesto que el diseño de una base de conocimiento a través de un conocimiento experto, permite conseguir resultados mucho más precisos con igual o menor complejidad que un sistema analítico.

9.3 CONCLUSIONES DE LAS PRUEBAS

Teniendo en cuenta los resultados obtenidos en las pruebas presentadas con anterioridad, queda demostrado que es posible la gestión autónoma de procesos dentro de un sensor tanto con sistemas basados en diferencias, como con sistemas basados en conocimiento, dando cumplimiento al punto a) del Objetivo general 4. Estos sistemas han sido capaces de adaptarse al entorno a través de la estimación del tiempo de sueño, en función de los valores medidos de la magnitud bajo estudio. Todo esto de manera autónoma, dinámica y eficiente, por lo tanto, queda refrendada la **Hipótesis tercera** de la presente tesis.

Capítulo IV. CONCLUSIONES Y LÍNEAS DE FUTURO

En este capítulo se detallan, en primer lugar, las conclusiones obtenidas tras los trabajos de investigación y redacción de la presente tesis doctoral, acompañándose dichas reflexiones con las contribuciones que respaldan los resultados obtenidos. A continuación, y como elemento final del presente trabajo de investigación, se presentan las líneas de futuro que abre esta tesis y que perfilan la evolución natural de lo descrito en la misma.

10 CONCLUSIONES Y PRINCIPALES CONTRIBUCIONES

Teniendo en cuenta los resultados presentados en las secciones precedentes se pueden formular las siguientes conclusiones:

Primera. Que la integración de una arquitectura multi-agente FLBDIa en sensores para la gestión de sus recursos internos es factible y que el modelado de comportamientos dentro de los sensores se puede lograr a través de la definición de agentes FLBDI, como se detalló en la sección 3 y queda demostrado en la sección 7.

Segunda. Que el protocolo de aplicación WISMAP permite efectivamente dar soporte a sensores con una arquitectura multi-agente y con sistemas de auto-gestión basados en FRBS para una amplia variedad de casos. Esto ha quedado de manifiesto en la sección 8, derivándose además las siguientes consideraciones:

- El protocolo WISMAP permite el envío eficiente de bases de conocimiento para los FRBS, así como realizar tareas más básicas tales como el envío de muestras o el cambio de parámetros de funcionamiento de un sensor.
- Por otro lado, la definición de una estructura jerárquica de recursos permite que, con el protocolo WISMAP, se pueda acceder a sensores con arquitecturas internas diferentes, dinámicas y de complejidad variable, sin requerir ninguna modificación en el mismo.
- Que la implementación de un test-bed ha sido beneficiosa para poder realizar un número adecuado de pruebas de comunicaciones, así como para la simulación de diferentes configuraciones de redes de sensores.
- Que se ha conseguido una efectiva integración del protocolo WISMAP e Internet a través de la pasarela WISMAPi, la cual ha aprovechado los recursos que brinda el protocolo HTTP para conseguir dicha integración.

Tercera. Que es posible la efectiva integración en los sensores de sistemas de gestión autónoma basados en FRBS para el control de los recursos energéticos. Esto ha sido corroborado por los resultados mostrados en la sección 9, debiéndose además tener en cuenta las siguientes consideraciones adicionales:

- La gestión basada en FRBS es más eficiente en comparación con sistemas analíticos de igual complejidad.

- Los FRBS permiten modelar comportamientos inteligentes basados en el conocimiento experto en sensores, sin requerir grandes modificaciones, ni en el software de estos, ni en la estructura de la información que les da soporte.
- La definición del lenguaje XML^{KB} ha permitido que las bases de conocimiento puedan ser descritas formalmente en un lenguaje fácilmente legible, interpretable y editable, mientras que con el formato *raw* se consigue reducir la información necesaria para su envío por una red de sensores.

Como reflexión final a las conclusiones antes presentadas, cabe reseñar que las redes de sensores son aún un sector en crecimiento que está en continua evolución. Dentro de las WSNs aún existen gran cantidad de problemas por resolver y otro gran número de nuevas aplicaciones donde pueden ser útiles. Las futuras soluciones de redes de sensores deberán ser flexibles, escalables, eficientes y capaces de adaptarse a aplicaciones reales, ya sea dentro de la industria, biomedicina, medioambiente, vehículos o en el hogar. Es por eso, que en la presente tesis se ha buscado que la aplicabilidad de los resultados a entornos reales fuera la máxima posible, para que, de ese modo, se facilitara la trasferencia de los trabajos de investigación aquí expuestos.

10.1 PUBLICACIONES

A continuación se enumeran las publicaciones a las que ha dado lugar la presente tesis, y que avalan los objetivos y resultados presentados.

10.2 REVISTAS INCLUIDAS EN JCR

1. **J. C. Cuevas Martínez**, J. Cañada Bago, J. A. Fernández Prieto y M. A. Gadeo Martos. Knowledge-based duty cycle estimation in wireless sensor networks: Application for sound pressure monitoring. APPLIED SOFT COMPUTING. 13 - 2, pp. 967 - 980. 2013. DOI: 10.1016/j.asoc.2012.10.005
2. **J. C. Cuevas Martínez**, M. A. Gadeo Martos, J. A. Fernández Prieto y J. Cañada Bago. Wireless Intelligent Sensors Management Application Protocol-WISMAP. SENSORS 10 (10), pp. 8827 - 8849. 2010. DOI: 10.3390/s101008827.

10.3 CONGRESOS INTERNACIONALES

1. **J. C. Cuevas Martínez**, J. Cañada Bago, J. A. Fernández Prieto y M. A. Gadeo Martos. Propagation of agent performance parameters in wireless sensor networks. HIGHLIGHTS IN PRACTICAL APPLICATIONS OF AGENTS AND MULTIAGENT SYSTEMS. SERIES: ADVANCES IN INTELLIGENT AND SOFT COMPUTING. 89, pp. 213 - 221.2011.

11 LÍNEAS DE FUTURO

El trabajo iniciado en la presente tesis, como ya se ha comentado, ha perseguido la integración de tecnologías basadas en el conocimiento dentro de las redes de sensores, con el objetivo final de favorecer la integración en aplicaciones que aporten valor a la industria, medicina o el hogar, entre otros.

Así pues, se abren líneas de futuro orientadas al estudio de nuevos sistemas y entornos que permitan, evolucionando la arquitectura de agentes, el protocolo de comunicaciones y los FRBS, conseguir una adaptación e integración eficiente de los sensores, tanto en sus capacidades de auto-gestión, como en nuevas aplicaciones. Las contribuciones futuras se podrían dirigir hacia los siguientes campos:

- **Nuevas arquitecturas de agente.** Dado que las prestaciones del hardware de los sensores experimentan continuas mejoras, se podrán incorporar estructuras de agentes más potentes que permitan una mayor capacidad de decisión y replicar comportamientos más complejos. Algunas de esas nuevas capacidades, que convendría explorar, en conjunción con técnicas de inteligencia computacional (ver a continuación), se introducen a continuación:
    - **Supervivencia de la red de sensores y agentes:** organizar qué agentes deben seguir activos, cuáles migrar, cuáles abandonar, en qué momento, todo ello teniendo en cuenta la aplicación y tareas encomendadas.
    - **Mutación/evolución de comportamientos:** promover la aparición de nuevos comportamientos en los agentes de forma análoga a los algoritmos genéticos. En este caso, los agentes cuyos modelos de comportamiento estuvieran basados en FLBDIa, podrían combinarse con algoritmos genético-evolutivos que promovieran actualizaciones en su comportamiento con cambios en las bases de conocimiento.
    - **Movilidad:** este es un tema recurrente en gran número de contribuciones actuales, sin ser totalmente novedoso, siempre presentará desafíos mientras más sensores se incorporen a una red y la aplicación sea más compleja. La combinación de soluciones centralizadas, distribuidas, colaborativas o algún tipo de hibridación entre ellas, estará marcada por el compromiso entre consumo de energía, provocado por el gasto en comunicaciones y proceso, y las propias limitaciones de los sensores (como pudo verse en el apartado 2.1.1).
- **Ajuste y evaluación de los FRBS.** Los FRBS presentados se basan en reglas y variables borrosas definidas en bases de conocimiento, las cuales, gracias al protocolo WISMAP, pueden ser modificadas dentro de los sensores. Por lo tanto, se podrían desarrollar nuevos sistemas que, de forma conjunta o distribuida, realizaran modificaciones a las bases de conocimiento, con el objetivo de ajustar las respuestas de los FRBS a nuevas condiciones.
- **Aplicación de otras técnicas de inteligencia computacional**, tales como redes neuronales, algoritmos genéticos o sistemas bio-inspirados que en solitario, o junto con los FRBS ya presentados, permitan modelar nuevos comportamientos en los agentes, con el propósito de aumentar su autonomía, ya sea través de sistemas internos, externos o colaborativos. Entre las soluciones más importantes estarían las siguientes:
    - Estimación de la potencia de transmisión en redes, tanto estáticas como dinámicas.

    - Decisiones sobre cuándo enviar notificaciones de alarmas, actualizaciones de medidas, etc.
    - Encaminamiento y organización dinámica de clústeres en función de la aplicación, magnitud, degradación, necesidades del usuario, etc.
- **Integración dentro La Internet de las Cosas (IoT)** (Ashton2009), adaptando las funciones aportadas por el protocolo WISMAP, así como la estructura de recursos presentada para que esta integración de las redes de sensores en Internet, a través de IPv6, sea transparente. En particular, aunque se pueda usar 6LoWPAN6 para el envío de los paquetes IPv6 por la red de sensores, el enfoque propuesto en la presente tesis facilitaría la integración a nivel de aplicación, ocultando en lo posible al usuario las complejas características de una red de sensores, gracias a que WISMAPi (ver sección 4.5) es un protocolo fácilmente compatible con HTTP y basado en una arquitectura de recursos (ver sección 4.2), que se plasma en un formato con el que los usuarios están muy familiarizados.

BIBLIOGRAFÍA

Adler, R., Flanigan, M., Huang, J., Kling, R., Kushalnagar, N., Nachman, L., Wan, C. & Yarvis, M. 2005, "Intel Mote 2: an advanced platform for demanding sensor network applications", *Proceedings of the 3rd international conference on Embedded networked sensor systems*, pp. 298.

Agre, P.E. & Chapman, D. 1987, "Pengi: An implementation of a theory of activity", *Proceedings of the sixth National Conference on Artificial intelligence*, pp. 268-272.

Ahmed, M., Ahmad, M.S. & Yusoff, M.Z.M. 2009, "A Review and Development of Agent Communication Language", *Electronic Journal of Computer Science and Information Technology (eJCSIT)*, vol. 1(1), pp. 7-12.

Aiello, F., Bellifemine, F., Fortino, G., Galzarano, S. & Gravina, R. 2011, "An agent-based signal processing in-node environment for real-time human activity monitoring based on wireless body sensor networks", *Engineering Applications of Artificial Intelligence*, vol. 24, no. 7, pp. 1147-1161.

Aiello, F., Fortino, G., Guerrieri, A. & Gravina, R. 2009, "Maps: A mobile agent platform for wsns based on java sun spots", *Proceedings of the third international workshop on Agent Technology for Sensor Networks (ATSN)*.

Akkaya, K. & Younis, M. 2005, "A survey on routing protocols for wireless sensor networks", *Ad Hoc Networks*, vol. 3, no. 3, pp. 325-349.

Akyildiz, I.F. & Kasimoglu, I.H. 2004, "Wireless sensor and actor networks: research challenges", *Ad hoc networks*, vol. 2, no. 4, pp. 351-367.

Akyildiz, I.F., Melodia, T. & Chowdhury, K.R. 2007, "A survey on wireless multimedia sensor networks", *Computer Networks*, vol. 51, no. 4, pp. 921-960.

Akyildiz, I.F., Pompili, D. & Melodia, T. 2005, "Underwater acoustic sensor networks: research challenges", *Ad Hoc Networks*, vol. 3, no. 3, pp. 257-279.

Akyildiz, I.F., Su, W., Sankarasubramaniam, Y. & Cayirci, E. 2002, "Wireless sensor networks: a survey", *Computer Networks*, vol. 38, no. 4, pp. 393-422.

Akyildiz, I.F. & Vuran, M.C. 2010, *Wireless Sensor Networks*, John Wiley & Sons, Inc, New York, NY, USA.

Akyildiz, I.F., Vuran, M.C. & Akan, O.B. 2006, "A Cross-Layer Protocol for Wireless Sensor Networks", *40th Annual Conference on Information Sciences and Systems*, pp. 1102-1107.

Akyildiz, I.F., Weilian, S., Sankarasubramaniam, Y. & Cayirci, E. 2002, "A survey on sensor networks", *Communications Magazine, IEEE*, vol. 40, no. 8, pp. 102-114.

Al-Karaki, J.N. & Kamal, A.E. 2004, "Routing techniques in wireless sensor networks: a survey", *Wireless Communications, IEEE,* vol. 11, no. 6, pp. 6-28.

Anastasi, G., Conti, M., Di Francesco, M. & Passarella, A. 2009, "Energy conservation in wireless sensor networks: A survey", *Ad Hoc Networks,* vol. 7, no. 3, pp. 537-568.

Ashton, K. 2009, "That 'Internet of Things' Thing", *RFID Journal,* vol. 22, pp. 97-114.

Ayala, I., Amor, M. & Fuentes, L. 2012a, "Self-management of ambient intelligence systems: a pure agent-based approach", *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, pp. 1427-1428.

Ayala, I., Amor, M. & Fuentes, L. 2012b, "Self-StarMAS: A Multi-Agent System for the Self-Management of AAL Applications", *Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS),* pp. 901-906.

Balister, P., Zheng, Z., Kumar, S. & Sinha, P. 2009, "Trap Coverage: Allowing Coverage Holes of Bounded Diameter in Wireless Sensor Networks", *INFOCOM 2009, IEEE*, pp. 136-144.

Barandiaran, X.E., Di Paolo, E. & Rohde, M. 2009, "Defining agency: Individuality, normativity, asymmetry, and spatio-temporality in action", *Adaptive Behavior,* vol. 17, no. 5, pp. 367-386.

Bardossy, A. & Duckstein, L. 1995, *Fuzzy Rule-Based Modeling with Applications to Geophysical, Biological, and Engineering Systems*, CRC Press, Inc, Boca Raton, FL, USA.

Baronti, P., Pillai, P., Chook, V.W.C., Chessa, S., Gotta, A. & Fun Hu, Y. 2007, "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards", *Computer Communications,* vol. 30, no. 7, pp. 1655-1695.

Barr, K.C. & Asanović, K. 2006, "Energy-aware lossless data compression", *ACM Transactions on Computer Systems (TOCS),* vol. 24, no. 3, pp. 250-291.

Bartlett, M.S. 1937, "Properties of sufficiency and statistical tests", *Proceedings of the Royal Society of London.Series A-Mathematical and Physical Sciences,* vol. 160, no. 901, pp. 268-282.

Bellifemine, F., Caire, G., Poggi, A. & Rimassa, G. 2008, "JADE: A software framework for developing multi-agent applications. Lessons learned", *Information and Software Technology,* vol. 50, no. 1–2, pp. 10-21.

Bellifemine, F., Poggi, A. & Rimassa, G. 1999, "JADE–A FIPA-compliant agent framework", *Fourth International Conference on Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM 1999)*, pp. 97-108.

Benoit, E., Dapoigny, R. & Foulloy, L. 2001, "Fuzzy-based intelligent sensors: modeling, design, application", *Proceedings of the 8th IEEE International Conference on Emerging Technologies and Factory Automation*, pp. 401-407.

Berners-Lee, T., Fielding, R. & Masinter, L. January 2005, *Uniform Resource Identifier (URI): Generic Syntax. RFC 3986*.

Bigus, J.P., Schlosnagle, D.A., Pilgrim, J.R., Mills III, W.N. & Diao, Y. 2002, "ABLE: A toolkit for building multiagent autonomic systems", *IBM Systems Journal*, vol. 41, no. 3, pp. 350-371.

Bohli, J.M., Sorge, C. & Westhoff, D. 2009, "Initial observations on economics, pricing, and penetration of the internet of things market", *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 2, pp. 50-55.

Brooks, R.A. 1991, "Intelligence without reason", *Artificial intelligence: critical concepts*, vol. 3, pp. 107-163.

BTnodes, *A Distributed Environment for Prototyping Ad Hoc Networks : Main - Overview browse*. Available: http://www.btnode.ethz.ch/ [2012, 7/13/2012].

Buratti, C., Conti, A., Dardari, D. & Verdone, R. 2009, "An overview on wireless sensor networks technology and evolution", *Sensors*, vol. 9, no. 9, pp. 6869-6896.

Canada-Bago, J. 2007, "From a Genetic Fuzzy Rule-Based System to a Intelligent Sensor Network", *International Conference on Sensor Technologies and Applications*, pp. 373-377.

Cao, Q., Abdelzaher, T., He, T. & Stankovic, J. 2005, "Towards optimal sleep scheduling in sensor networks for rare-event detection", *4th International Symposium on Information Processing in Sensor Networks, IPSN 2005*, pp. 20-27.

Chapman, D. & Agre, P.E. 1987, "Abstract reasoning as emergent from concrete activity" in *1986 Workshop on Reasoning About Actions and Plans*, eds. M.P. Georgeff & A.L. Lansky, Morgan Kaufmann Publishers, Inc., Los Altos, California, pp. 411-424.

Chinrungrueng, J., Sunantachaikul, U. & Triamlumlerd, S. 2007, "Smart parking: An application of optical wireless sensor network", *International Symposium on Applications and the Internet Workshops*, pp. 66-66.

Cohen, P.R. & Levesque, H.J. 1990, "Intention is choice with commitment", *Artificial Intelligence*, vol. 42, no. 2, pp. 213-261.

Collier, N. 2003, "Repast: An extensible framework for agent simulation", *The University of Chicago's Social Science Research*, vol. 36.

Corchado, J.M., Bajo, J., Tapia, D.I. & Abraham, A. 2010, "Using heterogeneous wireless sensor networks in a telemonitoring system for healthcare", *Information Technology in Biomedicine, IEEE Transactions on*, vol. 14, no. 2, pp. 234-240.

Cordón, O., Herrera, F., Hoffmann, F. & Magdalena, L. 2001, *Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases*, World Scientific Pub Co Inc.

Crocker, D.H. August 1982, *Standard for the Format of ARPA Internet Text Messages. RFC 822*.

Dagdeviren, O., Korkmaz, I., Tekbacak, F. & Erciyes, K. 2011, "A survey of agent technologies for wireless sensor networks", *IETE Technical Review*, vol. 28, no. 2, pp. 168.

Daneshfar, F. & Bevrani, H. 2009, "Multi-agent systems in control engineering: a survey", *Journal of Control Science and Engineering*, vol. 2009, pp. 5:1-5:12.

De Marziani, C., Alcoleas, R., Colombo, F., Costa, N., Pujana, F., Colombo, A., Aparicio, J., Alvarez, F.J., Jimenez, A. & Urena, J. 2011, "A low cost reconfigurable sensor network for coastal monitoring", *IEEE-Spain OCEANS*, pp. 1-6.

Demirkol, I., Ersoy, C. & Alagoz, F. 2006, "MAC protocols for wireless sensor networks: a survey", *Communications Magazine, IEEE*, vol. 44, no. 4, pp. 115-121.

Di Marco, P., Park, P., Fischione, C. & Johansson, K.H. 2010, "TREnD: A Timely, Reliable, Energy-Efficient and Dynamic WSN Protocol for Control Applications", *IEEE International Conference on Communications (ICC)*, pp. 1-6.

Dunkels, A., Gronvall, B. & Voigt, T. 2004, "Contiki - a lightweight and flexible operating system for tiny networked sensors", *29th Annual IEEE International Conference on Local Computer Networks*, pp. 455-462.

Echelon Corporation, *LonWorks, estánar y tecnología*. Available: http://www.echelon.com/technology/lonworks/; [2013, 7/13/2013].

Egea-Lopez, E., Vales-Alonso, J., Martinez-Sala, A., Pavon-Mario, P. & Garcia-Haro, J. 2006, "Simulation scalability issues in wireless sensor networks", *Communications Magazine, IEEE*, vol. 44, no. 7, pp. 64-73.

Egnite, *The Ethernut Project*. Available: http://www.ethernut.de/en/software/index.html [2012, 6/20/2012].

EU Integrated Project, *SENSEI: Integrating the physical with the digital world of the network of the future*. Available: http://www.ict-sensei.org/index.php [2013, 2/27/2013].

Fagin, R., Halpern, J.Y., Moses, Y. & Vardi, M.Y. 1995, *Reasoning about knowledge*, MIT press, Cambridge, MA.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. & Berners-Lee, T. June 1999, *Hypertext transfer protocol - HTTP/1.1. RFC 2616*.

Filipponi, L., Santini, S. & Vitaletti, A. 2008, *Data collection in wireless sensor networks for noise pollution monitoring*, Springer.

Finin, T., Fritzson, R., McKay, D. & McEntire, R. 1994, "KQML as an agent communication language", *Proceedings of the third international conference on Information and knowledge management*ACM, New York, NY, USA, pp. 456-463.

Fipa ACL 2004, *FIPA ACL Message Structure Specification*. Available: http://www.fipa.org/specs/fipa00061/SC00061G.html [2013, 10/6].

Fisher, R.A. 1970, *Statistical methods for research workers*, 14th edn, Oliver and Boyd Edinburgh.

Fok, C., Roman, G.-. & Lu, C. 2005, "Mobile agent middleware for sensor networks: an application case study", *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pp. 382-387.

Fok, C., Roman, G. & Lu, C. 2009, "Agilla: A mobile agent middleware for self-adaptive wireless sensor networks", *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 4, no. 3, pp. 16:1-16:26.

Fortino, G., Guerrieri, A. & Russo, W. 2012, "Agent-oriented smart objects development", *16th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 907-912.

Franklin, S. & Graesser, A. 1997, "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents", *Intelligent Agents III Agent Theories, Architectures, and Languages*, vol. 1193, pp. 21-35.

Freed, N. & Borenstein, N. November 1996, *Multipurpose internet mail extensions (MIME) part one: Format of internet message bodies. RFC 2045*.

Gadeo-Martos, M.A., Fernandez-Prieto, J.A. & Velasco, J.R. 2011, "An Architecture for Performance Optimization in a Collaborative Knowledge-Based Approach for Wireless Sensor Networks", *Sensors*, vol. 11, no. 10, pp. 9136-9159.

García Villalba, L.J., Sandoval Orozco, A.L., Triviño Cabrera, A. & Barenco Abbas, C.J. 2009, "Routing protocols in wireless sensor networks", *Sensors*, vol. 9, no. 11, pp. 8399-8421.

Gelernter, D. 1985, "Generative communication in Linda", *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 7, no. 1, pp. 80-112.

Goel, S. & Imielinski, T. 2001, "Prediction-based monitoring in sensor networks: taking lessons from MPEG", *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 5, pp. 82-98.

Haider, T. & Yusuf, M. 2009, "A fuzzy approach to energy optimized routing for wireless sensor networks", *The International Arab Journal of Information Technology*, vol. 6, no. 2.

Halgamuge, M.N., Guru, S.M. & Jennings, A. 2003, "Energy efficient cluster formation in wireless sensor networks", *10th International Conference on Telecommunications (ICT)*, pp. 1571-1576.

Handcock, R.N., Swain, D.L., Bishop-Hurley, G.J., Patison, K.P., Wark, T., Valencia, P., Corke, P. & O'Neill, C.J. 2009, "Monitoring Animal Behaviour and Environmental Interactions Using Wireless Sensor Networks, GPS Collars and Satellite Remote Sensing", *Sensors*, vol. 9, no. 5, pp. 3586-3603.

Handziski, V., K\opke, A., Willig, A. & Wolisz, A. 2006, "TWIST: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks", *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*New York, NY, USA, pp. 63-70.

Harel, D., Kozen, D. & Tiuryn, J. 2000, *Dynamic logic*, MIT press, Cambridge, MA, USA.

HART Communication Foundation 2007, *HART Field Communication Protocol Specification, Revision 7.0*, HART Communication Foundation.

Heinzelman, W.B., Chandrakasan, A.P. & Balakrishnan, H. 2002, "An application-specific protocol architecture for wireless microsensor networks", *Wireless Communications, IEEE Transactions on*, vol. 1, no. 4, pp. 660-670.

Heinzelman, W.B., Kulik, J. & Balakrishnan, H. 1999, "Adaptive protocols for information dissemination in wireless sensor networks", *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pp. 174-185.

Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D.E. & Pister, K. 2000, "System architecture directions for networked sensors", *ACM SIGOPS Operating Systems Review*, vol. 34, no. 5, pp. 93-104.

Hirota, K. & Sugeno, M. 1995, *Industrial applications of fuzzy technology in the world*, World Scientific Publishing Company Incorporated.

Hongbo Jiang, Shudong Jin & Chonggang Wang 2011, "Prediction or Not? An Energy-Efficient Framework for Clustering-Based Data Collection in Wireless Sensor Networks", *Parallel and Distributed Systems, IEEE Transactions on*, vol. 22, no. 6, pp. 1064-1071.

Hughes, D., Greenwood, P., Blair, G., Coulson, G., Pappenberger, F., Smith, P. & Beven, K. 2006, "An intelligent and adaptable grid-based flood monitoring and warning system", *Proceedings of the 5th UK eScience All Hands Meeting*Edinburgh, UK, pp. 53-63.

Huo, H., Xu, Y., Yan, H., Mubeen, S. & Zhang, H. 2009, "An elderly health care system using wireless sensor networks at home", *Third International Conference on Sensor Technologies and Applications, SENSORCOMM'09.*Athens/Glyfada, Greece, pp. 158-163.

IEEE 2008, *IEEE Standard for Floating-Point Arithmetic*.

IEEE 2003, *IEEE 802.15.4: Wireless Medium Access Control (MAC) and Physical layer (PHY) specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs),* IEEE Computer Society, LAN/MAN standards committee.

Imran, M., Said, A.M. & Hasbullah, H. 2010, "A survey of simulators, emulators and testbeds for wireless sensor networks", *International Symposium in Information Technology (ITSim)*Kuala Lumpur, Malaysia, pp. 897-902.

Indranil Gupta, Riordan, D. & Srinivas Sampalli 2005, "Cluster-head election using fuzzy logic for wireless sensor networks", *Proceedings of the 3rd Annual Communication Networks and Services Research Conference*Halifax, Canada, pp. 255-260.

ISA100 Standards Committee 2009, *Wireless Systems for Industrial Automation: Process Control and Related Applications*.

Ivanović, M., Vidaković, M., Mitrović, D. & Budimac, Z. 2012, "Evolution of Extensible Java EE-Based Agent Framework" in *Agent and Multi-Agent Systems. Technologies and Applications*, eds. A. Håkansson, N. Nguyen, R.L. Hartung, R.J. Howlett & L.C. Jain, Springer-Verlag Berlin Heidelberg, Germany, pp. 444-453.

Jennings, N.R. & Wooldridge, M. 2002, *Agent technology: foundations, applications, and markets*, Springer, New York, USA.

Johnson, D., Stack, T., Fish, R., Flickinger, D.M., Stoller, L., Ricci, R. & Lepreau, J. 2006, "Mobile Emulab: A Robotic Wireless and Sensor Network Testbed", *Proceedings of the 25th IEEE International Conference on Computer Communications*, pp. 1-12.

Johnson, M., Healy, M., Van de Ven, P., Hayes, M.J., Nelson, J., Newe, T. & Lewis, E. 2009, "A comparative review of wireless sensor network mote technologies", *Sensors, 2009 IEEE*, pp. 1439-1442.

Kaelbling, L.P. 1987, "An architecture for intelligent reactive systems" in *Reasoning about actions and plans*, ed. M.B. Morgan, Morgan Kaufmann Publishers Inc, Los Altos, California, pp. 395-410.

Kaelbling, L.P. & Rosenschein, S.J. 1990, "Action and planning in embedded agents" in *Designing Autonomous Agents*, ed. P. Maes, The MIT Press/Elsevier, Amsterdam, Netherlands, pp. 35-48.

Karl, H. & Willig, A. 2007, *Protocols and architectures for wireless sensor networks,* Wiley-Interscience.

Karlsson, B., Bäckström, O., Kulesza, W. & Axelsson, L. 2005, "Intelligent sensor networks-an agent-oriented approach", *Proceedings of the Workshop on Real World Wireless Sensor Networks*.

Kauffman, R.J. & Walden, E.A. 2001, "Economics and Electronic Commerce: Survey and Directions for Research", *International Journal of Electronic Commerce*, vol. 5, no. 4, pp. 5-116.

Khedo, K.K., Perseedoss, R. & Mungur, A. 2010, "A wireless sensor network air pollution monitoring system", *International Journal of Wireless & Mobile Networks, IJWMN*, pp. 31-45.

Kho, J., Rogers, A. & Jennings, N.R. 2009, "Decentralized control of adaptive sampling in wireless sensor networks", *ACM Transactions on Sensor Networks*, vol. 5, no. 3, pp. 19:1-19:35.

Kim, J.M., Park, S.H., Han, Y.J. & Chung, T.M. 2008, "CHEF: cluster head election mechanism using fuzzy logic in wireless sensor networks", *10th International Conference on Advanced Communication Technology*, pp. 654-659.

KNX Association, *Estándar KNX*. Available: http://www.knx.org/ [2012, 12/6/2012].

Krishnamachari, L., Estrin, D. & Wicker, S. 2002, "The impact of data aggregation in wireless sensor networks", *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops*Vienna, Austria, pp. 575-578.

Kruskal, W.H. & Wallis, W.A. 1952, "Use of ranks in one-criterion variance analysis", *Journal of the American statistical Association*, vol. 47, no. 260, pp. 583-621.

Kulkarni, R.V., Förster, A. & Venayagamoorthy, G.K. 2011, "Computational Intelligence in Wireless Sensor Networks: A Survey", *Communications Surveys & Tutorials, IEEE*, vol. 13, no. 1, pp. 68-96.

Kumar, Y., Munjal, R. & Kumar, K. 2012, "Wireless Sensor Networks and Security Challenges", *IJCA Proceedings on National Workshop-Cum-Conference on Recent Trends in Mathematics and Computing 2011*, vol. RTMC(9), pp. -.

Kushalnagar, N., Montenegro, G., Hui, J.W. & Culler, D.E. September 2007, *6LoWPAN: Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944*.

Levene, H. 1960, "Robust tests for equality of variances" in *Contributions to probability and statistics: essays in honor of Harold Hotelling*, eds. I. Olkin, H. Hotelling, S.G. Ghurye, W.G. Madow & H.B. Mann, Stanford University Press, Stanford, CA, USA, pp. 278-292.

Levis, P., Lee, N., Welsh, M. & Culler, D.E. 2003, "TOSSIM: accurate and scalable simulation of entire TinyOS applications", *Proceedings of the 1st international conference on Embedded networked sensor systems*New York, NY, USA, pp. 126-137.

Liang, Q. & Wang, L. 2005, "Event detection in wireless sensor networks using fuzzy logic system", *Proceedings of the 2005 IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety*, pp. 52-55.

Libelium, *Waspmote - Wireless Sensor Networks 802.15.4 ZigBee Mote* . Available: http://www.libelium.com/products/waspmote [2012, 7/17/2012].

Linder, B., Hoek, W. & Meyer, J.-.C. 1994, "Communicating rational agents", vol. 861, pp. 202-213.

Lindsey, S. & Raghavendra, C.S. 2002, "PEGASIS: Power-efficient gathering in sensor information systems", *IEEE Aerospace Conference Proceedings*, pp. 1125-1130.

Liu, T. & Martonosi, M. 2003, "Impala: a middleware system for managing autonomic, parallel sensor systems", *Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming*, pp. 107-118.

Long, S.A. & Esterline, A.C. 2000, "Fuzzy BDI architecture for social agents", *Proceedings of the IEEE Southeastcon*, pp. 68-74.

Lu, G., Krishnamachari, B. & Raghavendra, C.S. 2004, "An adaptive energy-efficient and low-latency MAC for data gathering in wireless sensor networks", *Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04) - Workshop 12*, pp. 224.

Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K. & Balan, G.C. 2005, "MASON: A multiagent simulation environment", *Simulation*, vol. 81, no. 7, pp. 517-527.

Madejski, J. 2007, "Survey of the agent-based approach to intelligent manufacturing", *Journal of Achievements in Materials and Manufacturing Engineering*, vol. 21, no. 1, pp. 67-70.

Magdalena, L. & Velasco, J.R. 1996, "Fuzzy rule-based controllers that learn by evolving their knowledge base" in *Genetic Algorithms and Soft Computing*, eds. F. Herrera & J.L. Verdegay, Physica-Verlag, Heidelberg, Germany, pp. 172-201.

Magdalena, L. & Monasterio, F. 1995, "Evolutionary-based learning applied to fuzzy controllers", *Proceedings of the International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium*, pp. 1111-1118.

Mahfoudh, S. & Minet, P. 2008, "Survey of Energy Efficient Strategies in Wireless Ad Hoc and Sensor Networks", *Seventh International Conference on Networking, ICN 2008.*Cancun, Mexico, pp. 1-7.

Mamdani, E.H. 1974, "Application of fuzzy algorithms for control of simple dynamic plant", *Proceedings of the Institution of Electrical Engineers*, pp. 1585-1588.

Mamdani, E.H. & Assilian, S. 1975, "An experiment in linguistic synthesis with a fuzzy logic controller", *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1-13.

Marin-Perianu, M. & Havinga, P. 2007, "D-FLER–a distributed fuzzy logic engine for rule-based wireless sensor networks", *Ubiquitous Computing Systems*, vol. 4836, pp. 86-101.

Mariscal-Ramirez, J.A., Fernandez-Prieto, J.A., Gadeo-Martos, M.A. & Canada-Bago, J. 2011, "Knowledge-based wireless sensors using sound pressure level for noise pollution monitoring", *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on*, pp. 1032-1037.

Mariscal-Ramirez, J.A., Fernandez-Prieto, J.A., Canada-Bago, J. & Gadeo-Martos, M.A. 2014, " A new algorithm to monitor noise pollution adapted to resource-constrained devices", *Multimedia Tools and Applications*, pp. 1-15.

Martyna, J. 2006, "Fuzzy Reinforcement Learning for Routing in Wireless Sensor Networks" in *Computational Intelligence, Theory and Applications*, ed. B. Reusch, Springer Berlin Heidelberg, pp. 637-645.

Melodia, T., Vuran, M.C. & Pompili, D. 2006, "The state of the art in cross-layer design for wireless sensor networks", *Wireless Systems and Network Architectures in Next Generation Internet*, vol. 3883, pp. 78-92.

Minar, N., Burkhart, R., Langton, C. & Askenazi, M. 1996, "The swarm simulation system: A toolkit for building multi-agent simulations", .

Minhas, M.R., Gopalakrishnan, S. & Leung, V.C.M. 2009, "Multiobjective Routing for Simultaneously Optimizing System Lifetime and Source-to-Sink Delay in Wireless Sensor Networks", *Proceedings of the 29th IEEE International Conference onDistributed Computing Systems Workshops, 2009*, pp. 123-129.

Misra, S., Roy, S., Obaidat, M.S. & Mohanta, D. 2009, "A Fuzzy logic-based Energy Efficient Packet Loss Preventive Routing Protocol", *International Symposium on Performance Evaluation of Computer & Telecommunication Systems*, pp. 185-192.

Moore, R.C. 1985, "A formal theory of knowledge and action" in *Formal Theories of the Commonsense World*, eds. J.R. Hobbs & R.C. Moore, Ablex Publishing Corporation, Norwood, New Jersey.

Müller, J., Pischel, M. & Thiel, M. 1995, "Modeling reactive behaviour in vertically layered agent architectures" in *Intelligent Agents*, eds. M. Wooldridge & N.R. Jennings, Springer Berlin, Heidelberg, Germany, pp. 261-276.

Murphy, A.L., Picco, G.P. & Roman, G.-. 2001, "LIME: a middleware for physical and logical mobility", *21st International Conference on Distributed Computing Systems*, pp. 524-533.

Nikolai, C. & Madey, G. 2009, "Tools of the Trade: A Survey of Various Agent Based Modeling Platforms", *Journal of Artificial Societies and Social Simulation*, vol. 12, no. 2, pp. 2-2.

O'Brien, P.D. & Nicol, R.C. 1998, "FIPA — Towards a Standard for Software Agents", *BT Technology Journal*, vol. 16, no. 3, pp. 51-59.

Oracle Labs, *SunSPOTWorld - Home* - . Available: http://www.sunspotworld.com/ [2012, 2/23/2012].

Overell, P. & Crocker, D. Enero 2008, *Augmented BNF for syntax specifications: ABNF. RFC 5234*.

Padhy, P., Dash, R.K., Martinez, K. & Jennings, N.R. 2006, "A utility-based sensing and communication model for a glacial sensor network", *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pp. 1353-1360.

Perkins, C., Belding-Royer, E. & Das, S. August 2003, *Ad hoc on demand distance vector (AODV) routing. RFC 3561*.

Pirmez, L., Delicato, F.C., Pires, P.F., Mostardinha, A.L. & de Rezende, N.S. 2007, "Applying fuzzy logic for decision-making on wireless sensor networks", *IEEE International Fuzzy Systems Conference*, pp. 1-6.

Potdar, V., Sharif, A. & Chang, E. 2009, "Wireless Sensor Networks: A Survey", *International Conference on Advanced Information Networking and Applications Workshops*, pp. 636-641.

Railsback, S.F., Lytinen, S.L. & Jackson, S.K. 2006, "Agent-based simulation platforms: Review and development recommendations", *Simulation*, vol. 82, no. 9, pp. 609-623.

Rajagopalan, R. & Varshney, P.K. 2006, "Data-aggregation techniques in sensor networks: a survey", *IEEE Communications Surveys & Tutorials*, vol. 8, no. 4, pp. 48-63.

Raman, B. & Chebrolu, K. 2008, "Censor networks: a critique of sensor networks from a systems perspective", *SIGCOMM Comput.Commun.Rev.*, vol. 38, no. 3, pp. 75-78.

Rao, A.S. & Georgeff, M.P. 1991, "Modeling Rational Agents within a BDI-Architecture" in *Proceedings of Knowledge Representation and Reasoning (KR&R-91)*, eds. R. Fikes & E. Sandewall, Morgan Kaufmann Publishers, Inc., San Francisco, CA, USA, pp. 473-484.

Rawi, M.I.M. & Al-Anbuky, A. 2011, "Development of Intelligent Wireless Sensor Networks for Human Comfort Index Measurement", *Procedia Computer Science*, vol. 5, pp. 232-239.

Raychaudhuri, D., Seskar, I., Ott, M., Ganu, S., Ramachandran, K., Kremo, H., Siracusa, R., Liu, H. & Singh, M. 2005, "Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols", *IEEE Wireless Communications and Networking Conference*, pp. 1664-1669 Vol. 3.

Rea, S. & Pesch, D. 2004, "Multi-metric routing decisions for ad hoc networks using fuzzy logic", *1st International Symposium on Wireless Communication Systems*, pp. 403-407.

Resnick, P. October 2008, *Internet message format. RFC 5322*.

Resnick, M. 1996, "StarLogo: an environment for decentralized modeling and decentralized thinking", *Conference Companion on Human Factors in Computing Systems*, pp. 11-12.

Rogers, A. 2011, "Agent Technologies for Sensor Networks", *The Computer Journal*, vol. 54, no. 3, pp. 307-308.

Rogers, A., Corkill, D.D. & Jennings, N.R. 2009, "Agent Technologies for Sensor Networks", *IEEE Intelligent Systems,* vol. 24, no. 2, pp. 13-17.

Rosenschein, S. & Kaelbling, L.P. 1986, "The synthesis of digital machines with provable epistemic properties" in *Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning About Knowledge*, ed. J.Y. Halpern, Morgan Kaufmann Publishers, Inc., San Francisco, CA, USA, pp. 83-98.

Ruiz-Garcia, L., Lunadei, L., Barreiro, P. & Robla, I. 2009, "A Review of Wireless Sensor Technologies and Applications in Agriculture and Food Industry: State of the Art and Current Trends", *Sensors*, vol. 9, no. 6, pp. 4728-4750.

Sandhu, J.S., Agogino, A.M. & Agogino, A.K. 2004, "Wireless sensor networks for commercial lighting control: decision making with multi-agent systems", *AAAI workshop on sensor networks*, pp. 131-140.

Santi, P. 2005, "Topology control in wireless ad hoc and sensor networks", *ACM Computing Surveys (CSUR),* vol. 37, no. 2, pp. 164-194.

Shamshirband, S., Kalantari, S. & Bakhshandeh, Z. 2010, "Designing a smart multi-agent system based on fuzzy logic to improve the gas consumption pattern", *Scientific Research and Essays*, vol. 5, no. 6, pp. 592-605.

Shansi Ren, Qun Li, Haining Wang, Xin Chen & Xiaodong Zhang 2007, "Design and Analysis of Sensing Scheduling Algorithms under Partial Coverage for Object Detection in Sensor Networks", *Parallel and Distributed Systems, IEEE Transactions on*, vol. 18, no. 3, pp. 334-350.

Shapiro, S.S. & Wilk, M.B. 1965, "An analysis of variance test for normality (complete samples)", *Biometrika*, vol. 52, no. 3/4, pp. 591-611.

Shen, S., O'Hare, G.M.P. & O'Grady, M.J. 2007, "Fuzzy-set-based decision making through energy-aware and utility agents within wireless sensor networks", *Artificial Intelligence Review*, vol. 27, no. 2-3, pp. 165-187.

Shen, S., O'Hare, G.M.P. & Collier, R. 2004, "Decision-making of BDI agents, a fuzzy approach", *Proceedings of the Fourth International Conference on Computer and Information Technology*, pp. 1022-1027.

Shoham, Y. 1993, "Agent-oriented programming", *Artificial Intelligence*, vol. 60, no. 1, pp. 51-92.

Silva, R., Silva, J.S. & Boavida, F. 2009, "Evaluating 6lowPAN implementations in WSNs", *Proceedings of 9th Conferncia sobre Redes de Computadores Oeiras, Portugal*.

Snedecor, G.W. & Cochran, W.G. 1980, *Statistical methods*, 7th edn, Iwoa State University Press, Ames, Iowa.

Srisooksai, T., Keamarungsi, K., Lamsrichan, P. & Araki, K. 2012, "Practical data compression in wireless sensor networks: A survey", *Journal of Network and Computer Applications*, vol. 35, no. 1, pp. 37-59.

Srivastava, N. 2010, "Challenges of next-generation wireless sensor networks and its impact on society", *Journal of Telecommunications*, vol. 1, no. 1, pp. 128-133.

Steyn, L.P. & Hancke, G.P. 2011, "A survey of Wireless Sensor Network testbeds", *AFRICON, 2011*, pp. 1-6.

Stone, P. & Veloso, M. 2000, "Multiagent systems: A survey from a machine learning perspective", *Autonomous Robots*, vol. 8, no. 3, pp. 345-383.

Student 1908, "The probable error of a mean", *Biometrika*, vol. 6-1, pp. 1-25.

Sun Microsystems 2008, *Link quality routing protocol (LQRP)*. Available: http://www.sunspotworld.com/docs/Red/javadoc/com/sun/spot/peripheral/radio/mhrp/lqrp/package-summary.html [2013, 6/15/2013].

Sun Microsystems, *Java ME Landing Page* . Available: http://www.oracle.com/technetwork/java/javame/index.html [2012, 7/17/2012].

Takagi, T. & Sugeno, M. 1985, "Fuzzy identification of systems and its applications to modeling and control", *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-15, no. 1, pp. 116-132.

Tandon, N. & Choudhury, A. 1999, "A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings", *Tribology International*, vol. 32, no. 8, pp. 469-480.

Thang Vu Chien, Hung Nguyen Chan & Thanh Nguyen Huu 2011, "A comparative study on operating system for Wireless Sensor Networks", *International Conference on Advanced Computer Science and Information System*, pp. 73-78.

The Network Simulator ns-2, *Network Simulator: Technical information* . Available: http://nsnam.isi.edu/nsnam/index.php/Main_Page [2012, 5/10/2012].

Tilak, S., Abu-Ghazaleh, N.B. & Heinzelman, W.B. 2002, "A taxonomy of wireless micro-sensor network models", *ACM SIGMOBILE Mobile Computing and Communications Review,* vol. 6, no. 2, pp. 28-36.

Tukey, J.W. & Cox, D.D.R. 1994, *The collected works of John W. Tukey*, Taylor & Francis US.

Tynan, R., Ruzzelli, A. & O'hare, G. 2005, "A methodology for the deployment of multi-agent systems on wireless sensor networks", *Proceedings of the International Conference on Software Engineering and Knowledge Engineering*.

Van der Hoek, W. & Wooldridge, M. 2008, "Chapter 24 Multi-Agent Systems" in *Foundations of Artificial Intelligence*, ed. D. Kirsh, Elsevier, Amsterdam, Netherlands, pp. 887-928.

Van Dyke Parunak, H., Brueckner, S., Fleischer, M. & Odell, J. 2004, "A design taxonomy of multi-agent interactions" in *Agent-Oriented Software Engineering IV*, eds. J. Odell, P. Giorgini & J.P. Müller, Springer-Verlag, Germany, pp. 123-137.

Verdone, R. & Buratti, C. 2005, "Modelling for wireless sensor network protocol design", *IEEE International Workshop on Wireless Ad-hoc Networks (IWWAN 2005)*, May 23-26.

Vijayraghavan, P. & Krishnan, R. 1998, "Noise in electric machines: A review", *Industry Applications Conference. Thirty-Third IAS Annual Meeting*, pp. 251-258 vol. 1.

Vinyals, M., Rodriguez-Aguilar, J.A. & Cerquides, J. 2011, "A survey on sensor networks from a multiagent perspective", *The Computer Journal*, vol. 54, no. 3, pp. 455-470.

Vukasinovic, I., Babovic, Z. & Rakocevic, G. 2012, "A survey on the use of Mobile Agents in Wireless Sensor Networks", *IEEE International Conference on Industrial Technology*, pp. 271-277.

Vuran, M.C. & Akyildiz, I.F. 2010, "XLP: A Cross-Layer Protocol for Efficient Communication in Wireless Sensor Networks", *Mobile Computing, IEEE Transactions on,* vol. 9, no. 11, pp. 1578-1591.

Wang, L.X. 1994, *Adaptive fuzzy systems and control- Design and stability analysis,* Prentice Hall.

Wang, L. & Xiao, Y. 2006, "A survey of energy-efficient scheduling mechanisms in sensor networks", *Mobile Networks and Applications,* vol. 11, no. 5, pp. 723-740.

Watteyne, T., Molinaro, A., Richichi, M.G. & Dohler, M. 2011, "From MANET To IETF ROLL Standardization: A Paradigm Shift in WSN Routing Protocols", *Communications Surveys & Tutorials, IEEE,* vol. 13, no. 4, pp. 688-707.

Wei Ye, Heidemann, J. & Estrin, D. 2002, "An energy-efficient MAC protocol for wireless sensor networks", *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1567-1576 vol.3.

Wenjie, C., Lifeng, C., Zhanglong, C. & Shiliang, T. 2005, "A realtime dynamic traffic control system based on wireless sensor network", *International Conference Workshops on Parallel Processing*, pp. 258-264.

Werner-Allen, G., Lorincz, K., Ruiz, M., Marcillo, O., Johnson, J., Lees, J. & Welsh, M. 2006, "Deploying a wireless sensor network on an active volcano", *Internet Computing, IEEE*, vol. 10, no. 2, pp. 18-25.

Werner-Allen, G., Swieskowski, P. & Welsh, M. 2005, "MoteLab: a wireless sensor network testbed", *Proceedings of the 4th international symposium on Information processing in sensor networks*.

Weyns, D., Van Dyke Parunak, H., Michel, F., Holvoet, T. & Ferber, J. 2005, "Environments for multiagent systems state-of-the-art and research challenges", *Environments for multi-agent systems*, vol. 3374, pp. 1-47.

Wong, K.H.M., Zheng, Y., Cao, J. & Wang, S. 2006, "A dynamic user authentication scheme for wireless sensor networks", *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, pp. 8 pp.

Wooldridge, M. 2009, *An introduction to multiagent systems*, Wiley.

Wooldridge, M. & Jennings, N.R. 1995a, "Agent theories, architectures, and languages: a survey", *Intelligent agents*, vol. 890, pp. 1-39.

Wooldridge, M. & Jennings, N.R. 1995b, "Intelligent agents: theory and practice", *The Knowledge Engineering Review*, vol. 10, no. 02, pp. 115-152.

Xia, F., Zhao, W., Sun, Y. & Tian, Y.C. 2007, "Fuzzy logic control based QoS management in wireless sensor/actuator networks", *Sensors*, vol. 7, no. 12, pp. 3179-3191.

Xiangqian Chen, Makki, K., Kang Yen & Pissinou, N. 2009, "Sensor network security: a survey", *Communications Surveys & Tutorials, IEEE*, vol. 11, no. 2, pp. 52-73.

Xing, G., Sha, M., Hackmann, G., Klues, K., Chipara, O. & Lu, C. 2009, "Towards unified radio power management for wireless sensor networks", *Wireless Communications and Mobile Computing*, vol. 9, no. 3, pp. 313-323.

Xuemei Li, Yuyan Deng & Lixing Ding 2008, "Study on precision agriculture monitoring framework based on WSN", *Proceedings of the 2nd International Conference on Anti-counterfeiting, Security and Identification.*, pp. 182-185.

Yang, H., Wu, H. & He, Y. 2009, "Architecture of wireless sensor network for monitoring aquatic environment of marine shellfish", *Proceedings of the 7th Asian Control Conference*, pp. 1147-1151.

Ye, W., Heidemann, J. & Estrin, D. 2004, "Medium access control with coordinated adaptive sleeping for wireless sensor networks", *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, pp. 493-506.

Yick, J., Mukherjee, B. & Ghosal, D. 2008, "Wireless sensor network survey", *Computer Networks*, vol. 52, no. 12, pp. 2292-2330.

Yigitel, M.A., Incel, O.D. & Ersoy, C. 2011, "QoS-aware MAC protocols for wireless sensor networks: A survey", *Computer Networks*, vol. 55, no. 8, pp. 1982-2004.

Younge, A.J. 2007, "Research Paper : Using Intelligent Agents in Wireless Sensor Networks", *Department of Computer Science, Rochester Institute of Technology, New York*, .

Youqun Shi & Hongyun Xu 2009, "Research on the BDI-Agent Learning Model Based on Dynamic Fuzzy Logic", *Proceedings of the International Conference onComputational Intelligence and Software Engineering*, pp. 1-6.

Yusuf, M. & Haider, T. 2005, "Energy-aware fuzzy routing for wireless sensor networks", *Proceedings of the IEEE Symposium on Emerging Technologies*, pp. 63-69.

Zadeh, L.A. 1975, "The concept of a linguistic variable and its application to approximate reasoning—I", *Information Sciences*, vol. 8, no. 3, pp. 199-249.

Zadeh, L.A. 1973, "Outline of a new approach to the analysis of complex systems and decision processes", *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-3, no. 1, pp. 28-44.

Zadeh, L.A. 1965, "Fuzzy Sets", *Information and Control*, vol. 8, pp. 338-353.

Zheng Gengzhong & Liu Qiumei 2010, "A Survey on Topology Control in Wireless Sensor Networks", *Proceedings of the Second International Conference on Future Networks*, pp. 376-380.

Zheng, J. & Jamalipour, A. 2009, *Wireless sensor networks. A networking perspective*, Wiley-IEEE Press.

Zheng, T., Qin, Y., Gao, D., Duan, J. & Zhang, H. 2010, "A practical deployment of Intelligent Building Wireless Sensor Network for environmental monitoring and air-conditioning control", *2nd IEEE International Conference on Network Infrastructure and Digital Content*, pp. 624-628.

ZigBee Alliance, *ZigBee Alliance Official site*. Available: http://www.zigbee.org/Home.aspx [2012, 9/11/2012].