UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

PROYECTO DE TESIS DOCTORAL

TECNOLOGÍAS MIDDLEWARE PARA EL DESARROLLO DE SERVICIOS EN ENTORNOS DE COMPUTACIÓN UBICUA

Autora: Mª Celeste Campo Vázquez
Director: Andrés Marín López

Fecha: 13 de enero de 2003

Índice

1. Introducción

Si nos fijamos a nuestro alrededor, la visión futurista que Mark Weiser describió en su artículo "The Computer for the 21st Century" [Weiser, 1991] comienza a ser una realidad. Weiser describe entornos saturados de elementos con capacidades de cómputo y comunicación, totalmente integrados en nuestras vidas y que nos proporcionan información asociada a nuestras necesidades y al entorno en el que nos encontramos en cada momento. En la actualidad, gracias a la evolución de la tecnología hardware, existen un mayor número de dispositivos: teléfonos móviles, PDAs, pagers, que nos acompañan en todo momento debido a su reducido tamaño. Estos dispositivos tienen capacidad de cómputo y además pueden comunicarse con otros elementos sin necesidad de conexiones físicas, gracias al desarrollo y evolución de los protocolos inalámbricos, tanto en redes celulares, GPRS [Andersson, 2001] y UMTS [Kaaranen et al., 2001], como en redes locales, WLAN [Dayem, 1997] y Bluetooth [Bray and Sturman, 2001]. Así, ya es más o menos habitual que estos dispositivos nos proporcionen nuevas aplicaciones además de las que propiamente tenían asociadas, por ejemplo, desde un móvil no sólo mantenemos una conversación telefónica, sino que también podemos ver la información del tiempo, localizar la farmacia más cercana o incluso, algo menos común, pero posible, programar la lavadora [Margherita2000.com, 2000].

La primera aproximación que la tecnología ha dado a la visión de Weiser es lo que se denomina *"anytime, anywhere"*, es decir, ha permitido a los usuarios, a través de sus nuevos dispositivos personales inalámbricos, acceder a las mismas aplicaciones que utilizan habitualmente en sus PCs, pero en cualquier momento y desde cualquier lugar. Así, los usuarios a través de ellos pueden navegar por Internet, WAP [Singhal, 2001] o i-Mode [Frengle, 2001], pueden consultar su correo electrónico, pueden sincronizar la información almacenada entre sus dispositivos, SyncML [Hansmann et al., 2002], para poder gestionar automáticamente las diversas actualizaciones realizadas en diferentes momentos desde diferentes dispositivos.

Pero la visión de Weiser va mucho más allá, involucra introducir capacidad de cómputo y de comunicación en el mundo físico [Hansmann et al., 2001]: en sensores, que capturen información del entorno o del propio usuario, en actuadores que nos permitan ejecutar acciones de forma remota sobre elementos físicos (puertas, interruptores,...), en electrodomésticos del hogar (lavadoras, neveras,...) o de oficinas (impresoras, faxes, aire acondicionado, proyectores, ...) y en sistemas de transporte (coches, autobuses, ...). Además será preciso que estos dispositivos tengan la capacidad de operar sin infraestructura de red fija, lo que implica no sólo la necesidad de emplear protocolos inalámbricos, sino que también será necesario proporcionales autonomía, es decir, que puedan comunicarse con otros dispositivos directamente sin necesidad de dispositivos intermedios (típicamente PCs) para construir nuevos servicios.

En estos nuevos entornos, los dispositivos personales de los usuarios permitirán que el mundo físico se configure según sus necesidades y preferencias, así no sólo podrá utilizar estos dispositivos como un ordenador portátil sino como un mando a distancia que le permita controlar su entorno físico más próximo. Por ejemplo, podemos pensar que la PDA de un usuario al entrar en una sala le indique al aire acondicionado que la temperatura ideal es 25 grados y que sea el propio aire acondicionado el que interaccione con los sensores de temperatura para que, a partir de sus mediciones, pueda mantener la temperatura indicada.

Para que esta visión se haga realidad es necesario realizar un gran esfuerzo de investigación, centrado en aportar soluciones para que la tecnología software nos dé el soporte adecuado para desarrollar aplicaciones en lo que se denominan entornos de computación móvil y ubicua. Las nuevas restricciones y características que imponen estos entornos, nos lleva a cambiar los con-

ceptos tradicionales que se han aplicado en entornos de redes fijas en las que los nodos eran computadores. Parece claro que no es eficente migrar soluciones tradicionales a estos nuevos sistemas [Banavar et al., 2000], existe una mayor diversidad de dispositivos, con diferentes limitaciones hardware para ejecutar aplicaciones y en los que hay que tener en cuenta que las comunicaciones pueden ser costosas y las distancias mucho más significativas que en las redes tradicionales. Además, la mayoría de estos nuevos dispositivos no son multipropósito, sino que proporcionan un serie de servicios concretos, pero de manera espontánea, cuando entran en contacto unos con otros a través de algún protocolo de comunicación, es posible que se compongan estos servicios de forma inteligente, para ofrecer un nuevo servicio que satisfaga las expectativas del usuario de manera transparente al mismo, como quería Weiser.

Este documento contiene los planteamientos para la realización de una tesis doctoral en el campo de la definición de tecnologías middleware para facilitar el desarrollo de servicios en entornos de computación ubicua, centrándose fundamentalmente en proporcionar el soporte adecuado en los nuevos dispositivos que aparecen en estos entornos. Así, en la sección 2 exponemos detalladamente la motivación de la tesis, describiendo los retos que impone el nuevo paradigma de la computación ubicua. A continuación en la sección 3, realizamos una introducción al estado del arte y en la sección 4 analizamos algunas de las investigaciones que se están llevando a cabo en otros centros de investigación, para abordar objetivos similares a los que se proponen en esta tesis, y que se enumeran en la sección 5. Por último en la sección 6 detallamos el plan de trabajo a seguir.

2. Motivación de la tesis

Como hemos introducido en la sección anterior, los avances que se han realizado en el campo de la microelectrónica, permitiendo embeber microprocesadores en cada vez más elementos del mundo físico, unido al desarrollo y esfuerzo de estandarización que se ha realizado en la definición de nuevos protocolos de red, hace posible que los dispositivos que pueden formar parte de un entorno de computación distribuida sean muchos y muy distintos, lo que introduce nuevos retos añadidos a los problemas tradicionales de este paradigma de la computación.

En la literatura [Satyanarayanan, 2001] se habla de la computación móvil y de la computación ubicua como evoluciones de la computación distribuida. Se comenzó a utilizar el término computación móvil a principios de los 90 cuando los dispositivos personales y las redes inalámbricas permitieron la movilidad de los usuarios, así a los problemas tradicionales se añadían los problemas de direccionamiento, de la calidad y capacidad cambiante de las redes de acceso debido al interfaz radio, de las limitaciones de recursos y baterías de los dispositivos finales, etc. Una gran mayoría de las soluciones que se presentaron a estos problemas, se centraban sobre todo en solventarlos de tal manera que las aplicaciones que se desarrollaban y los servicios que se le ofrecían a los usuarios finales, eran iguales que si estuvieran conectados a una red fija.

La computación móvil dió paso a lo que se denomina computación ubicua, cuando los dispositivos con capacidad de computación y comunicación no eran sólo dispositivos personales, sino casi cualquier dispositivo físico que rodea al usuario. En este nuevo paradigma se mantienen los retos de la computación móvil, y además se añaden algunos nuevos: la definición de nuevos protocolos de red inalámbrica para la comunicación directa entre dispositivos; la necesidad de que los dispositivos descubran las capacidades de otros, para poder ofrecer servicios más inteligentes y complejos al usuario, sin necesidad de interaccionar continuamente con él; la necesidad de capturar las preferencias del usuario de manera que los servicios se adapten a ellas, etc.

En los siguientes apartados detallaremos más esta evolución, describiendo primero lo que denominamos nuevos actores, que son los nuevos dispositivos que han aparecido estos últimos años y que permiten la movilidad de los usuarios y la ubicuidad de la tecnología. Después describiremos algunos de los nuevos escenarios en los que estos dispositivos proporcionarán al usuario nuevos servicios. Por último, detallaremos los retos que imponen estos escenarios y las respuestas que debe proporcionar la tecnología para que sean una realidad.

2.1. Nuevos actores

Desde nuestra visión creemos que los nuevos dispositivos con capacidad de computación y comunicación que han aparecido estos últimos años, nunca serán como ordenadores personales, aunque los desarrollos en microlectrónica hagan viable igualar sus capacidades, ya que la mayoría de ellos están dedicados a una tarea concreta y muy determinada. El usuario a lo que aspira es que las nuevas tecnologías permitan configurar esas tareas, adaptándolas a sus propias necesidades y que su interacción con ellas sea más fácil y transparente.

Con esta idea en mente, hemos clasificado estos dispositivos en varios grupos y hemos visto qué posibilidades de integración tendrán en un entorno distribuido.

- **Dispositivos personales**

  Englobamos en esta categoría a los nuevos dispositivos personales, como teléfonos móviles y agendas electrónicas, que, aunque en principio fueron diseñados para proporcionar un servicio concreto, telefonía inalámbrica o agenda, la realidad es que estos dispositivos cada vez proporcionan una mayor funcionalidad al usuario y además lo acompañan en todo momento.

  En estos últimos años existe una clara tendencia a incorporar a estos dispositivos capacidad de comunicación inalámbrica, no sólo sistemas de telefonía como GSM, o GPRS para conectarse a Internet y obtener servicios tradicionales como navegación Web o correo electrónico, sino también de corto alcance como IrDA, Bluetooth o IEEE 802.11 para que puedan colaborar con los dispositivos más próximos, formado redes ad-hoc. El uso de protocolos inalámbricos facilita también la movilidad de estos dispositivos.

  La característica más importante que tienen estos dispositivos es que son personales, de hecho todos ellos ofrecen un conjunto de aplicaciones denominadas *Personal Information Management* (PIM) y por lo tanto, las aplicaciones que se ejecutan en ellos deben estar personalizadas a las propias preferencias del usuario. Para ello los servicios se configurarán basándose en ellas, empleándose perfiles obtenidos de forma dinámica y transparente (o no) a partir de la interacción que tenga el usuario con el dispositivo y con las aplicaciones que en él residen.

  Las capacidades hardware de estos dispositivos varían enormemente, además la mayoría de ellos tienen un sistema operativo distinto y propietario, por lo que el desarrollo de aplicaciones en un principio estuvo cerrado a los propios fabricantes. En la actualidad podemos decir que existen tres grandes alternativas en el mercado que además han permitido que el desarrollo de aplicaciones se extienda, estos sistemas operativos son: WindowsCE, PalmOS y Symbian [Hansmann et al., 2001], los dos primeros centrados más en el mundo de las agendas personales y el último en el de los teléfonos móviles, aunque todos ellos comienzan a dar soporte a ambas funcionalidades. También existen varias iniciativas para introducir Linux en estos dispositivos y en este sentido, este último año ya han aparecido

productos comerciales con una versión reducida de este sistema operativo.

En general, casi todos estos dispositivos tienen displays con buenas resoluciones que facilitan la visualización de información al usuario. La introducción de datos se realiza mediante teclados tradicionales o reducidos, como los de los teléfonos móviles, pero también se han definido nuevos métodos para facilitar esta tarea al usuario como el método Graffiti, el T9 para teléfonos móviles, o reconocedores de escritura y de voz. Funcionan todos ellos con baterías.

- **Dispositivos de función específica**

  Englobamos en esta categoría a lo que se denomina tradicionalmente electrónica de consumo, tanto del hogar (lavadoras, aire acondicionado, ...) como de oficinas (impresoras, fax, ...). Estos dispositivos tienen una función concreta, y el usuario no espera que realicen otro tipo de tareas mas que aquella para la que han sido diseñados y comprados. Entonces la pregunta que se nos plantea es: cómo se pueden emplear las capacidades de computación y comunicación de estos dispositivos, posibles respuestas serían obtener un servicio mejor sin interacción directa y constante del usuario, o realizar funciones de mantenimiento que faciliten a los fabricantes dar un mejor producto.

  En general, estos dispositivos son estáticos y con conexión a la red mediante protocolos inalámbricos o cableados, empleando la red eléctrica, la red telefónica o cableados específicos. Sus características hardware varían de unos a otros y los sistemas operativos son propietarios y diseñados a medida, para que se realice correcta y eficientemente la función específica para la que se han construido. El desarrollo de aplicaciones está bastante cerrado, sobre todo porque no se ha tenido en cuenta la posibilidad que plantea el hecho de que estos dispositivos interaccionen con otros para realizar mejor su tarea, por ejemplo, la lavadora con los identificadores del tipo de ropa, el frigorífico con los identificadores de los productos, etc.

  Una característica común a casi todos ellos, es que su interfaz para interaccionar con el usuario está realizada a medida, con displays reducidos y teclas con funciones específicas. Al contrario que en los dispositivos personales, la mayoría de éstos no emplean baterías sino que se alimentan a través de la red eléctrica.

- **Sensores y actuadores**

  Esta categoría engloba a un gran número de dispositivos que agrupamos en dos tipos, por una parte los sensores, que nos permiten realizar medidas y obtener datos del medio físico que nos rodea, ejemplos de este tipo de dispositivos son un sensor de temperatura o de presión, o una *smart label* que nos proporciona la identificación de un objeto. El otro tipo, es lo que denominamos actuadores que nos permiten realizar una acción concreta sobre el medio físico, ejemplos de este tipo son un sistema de apertura de puertas, una bombilla, una alarma acústica, etc.

  Estos dispositivos son los más limitados en cuanto a capacidad de almacenamiento y procesamiento, pero son imprescindibles para proporcionarle a los usuarios servicios que tengan en cuenta el espacio físico que le rodea y que le permitan actuar directamente sobre él. Suelen ser dispositivos estáticos. En cuanto a los sensores, la tendencia actual es interconectarlos con protocolos inalámbricos, que faciliten colocarlos en cualquier lugar sin necesidad de cableado. Por este último motivo estos dispositivos suelen funcionar gracias a baterías, la optimización de su uso es más crítico en este caso debido a que cambiarlas o recargarlas no puede estar realizándose continuamente, se intenta conseguir que duren

años. Algunos sensores también emplean mecanismos inductivos para su funcionamiento, por ejemplo las *smart labels* transmiten la identificación cuando un sistema lector se aproxima y le suministra la energía necesaria para transmitir esa información. En cuanto a los actuadores, es más habitual que estén conectados a la red eléctrica, y por lo tanto, a veces emplean ésta como medio de transmisión para comunicarse con otros dispositivos. En general, tanto sensores como actuadores no tienen ninguna interfaz para que el usuario interaccione directamente con ellos.

2.2. Nuevos escenarios

En esta sección describiremos algunos escenarios de ejemplo que se basan en la capacidad de movilidad de los dispositivos y a su ubicuidad. Estos escenarios pueden parecer futuristas pero los retos tecnológicos que plantean son abordables en poco tiempo, si damos respuestas a algunas de las necesidades que se plantean, sobre todo a nivel software, y que han sido parte de la motivación de esta tesis.

- **Personalización**

  María dentro de un par de meses comienza sus vacaciones, este año le gustaría ir a Francia siempre tuvo ganas de conocer ese país, sobre todo su capital París. Antes de irse de vacaciones debe entregar un proyecto en su trabajo, y no tiene mucho tiempo para organizar su viaje, así que delega esta tarea al asistente personal que tiene instalado en su PDA. Este software realiza una búsqueda de billetes de avión y hoteles que mejor se adapten a las preferencias de María, además sabe que a María le gusta mucho la pintura impresionista y que las colas en los museos obligan a realizar reservas previas de las entradas, también sabe que recientemente un par de amigos de María han trasladado su residencia a un pueblo cerca de París, por lo que probablemente sea un buen momento para ir a visitarlos.

  Cuando el asistente detecta que María ya se encuentra en su casa, le informa mediante una alerta de algunas de las ofertas de avión y hotel que ha encontrado para saber cuál es la que más le satisface y si quiere realizar la reserva de alguna en concreto. El asistente también le informa de la posibilidad de reservar entradas en algunos museos, y le recuerda que es un buen momento para visitar a sus amigos, pero que debería ponerse en contacto con ellos para saber si estarán en esas fechas. A María le parece una gran idea e indica a su PDA que quiere realizar la llamada en ese momento, la PDA se pone en contacto con el teléfono y marca el número para que María puede hablar. A sus amigos les hace mucha ilusión la visita, así que le indica a la PDA que la incluya en el plan de viaje.

- **Hogar**

  Se aproxima la fecha del viaje de María, y el trabajo no le deja ningún tiempo libre. Pero su PDA sabe que es necesario informar a algunos dispositivos de la casa de que María va a pasar fuera unas cuantas semanas, así que lo anuncia al entorno. La nevera al recibir este anuncio realiza un chequeo de la comida que contiene, todos los productos poseen una *smart label* que los identifica y además almacena la fecha de caducidad, algunos de ellos es preciso consumirlos antes de su viaje, así que emplea su conexión a Internet para ofrecerle a María unas posibles cenas que le permitan consumirlos, la nevera no tiene display pero sabe que María siempre enciende la tele de la cocina antes de ponerse a cocinar así que le manda un mensaje a través de ella, y si María lo desea le enviará también algunos de sus sugerencias para la cena de hoy.

La PDA también informa a la calefacción los días que María no va a estar en casa, para que se desconecte siempre que la temperatura no baje de 13 grados, para que María no encuentre su casa muy destemplada cuando regrese. La calefacción detecta que hay una serie de sensores de temperatura en las habitaciones y los configura para que le comuniquen cuando la temperatura baja de ese valor. A María le preocupa mucho que alguien detecte su ausencia en casa así que ha configurado la televisión para que se encienda un rato todas las noches, además la televisión cuando se enciende se pone en contacto con las luces para que estas se enciendan también y así parezca más real que hay alguien en casa.

- **Aeropuerto**

  María llega por fin al aeropuerto, observa las pantallas y ve que su vuelo a París se ha retrasado una hora, de todas formas se dirige a la puerta de embarque, en el fondo se alegra, así podrá refinar un último apartado del documento del proyecto y enviarselo a su jefe, lo redacta en su PDA y le indica que necesita enviarlo, la PDA busca algún punto de acceso inalámbrico, detecta uno, pero para tener un buena conexión María debe aproximarse más, y así se lo indica mediante la visualización de un mensaje. María se acerca y consigue enviarlo. María sólo ha enviado el apartado que estaba modificando y unas directivas de cómo introducirlo en el documento final y a quién mandárselo, cuando todo esto llegue al PC de su oficina, será allí donde se realizarán todas estas tareas.

  En ese momento, anuncian por los altavoces un cambio de puerta de embarque, María no conoce muy bien el aeropuerto pero su PDA le visualiza un mapa del mismo y le indica cuál es el camino óptimo para ir a la nueva puerta de embarque.

  Cuando llega a la puerta de embarque y espera para poder entrar en el avión recibe una alerta en su PDA que alguien en su entorno busca a un jugador para una partida de damas, María acepta y se le descarga el juego para poder comenzar la partida y así amenizar el tiempo de espera.

- **Turismo**

  Por fin, María llega a París, ha empezado sus vacaciones. En su PDA tiene sus planes de viaje, al llegar al aeropuerto se le descarga un mapa del mismo y así puede llegar antes a la salida. Cuando llega al hotel la recepcionista toma sus datos y le configura a María una tarjeta inteligente inalámbrica. Al entrar en el ascensor no necesita marcar el piso al que va, sino que la propia tarjeta se comunica con el ascensor para indicárselo. Del mismo modo cuando llega a la puerta de su habitación ésta se abre automáticamente en cuanto María se aproxima. Al entrar en la habitación se enciende la televisión automáticamente apareciendo un mensaje de bienvenida para María y una serie de normas del hotel, como horarios del comedor, servicio de habitaciones, etc.

  María deja el hotel para pasear por las calles de París, como es la primera vez que visita la ciudad su PDA le proporciona un mapa de la zona. Al aproximarse por las calles y a través de su teléfono móvil le llegan alertas sobre determinadas tiendas que existen en la zona, el teléfono móvil se las pasa a la PDA que selecciona aquellas que más le interesan a María y se las muestra.

  A María también le gusta disfrutar "sin tecnología" de su viaje, así que deshabilita su PDA y decide callejear por París y fijarse ella misma en lo que le rodea. Pero cuando llega la hora de cenar, vuelve a habilitar su PDA para que le localice el restaurante de comida francesa más cercano en la zona y que tenga una mejor relación calidad/precio.

- **Automóvil**

  Después de un par de días en París, ha llegado el día de ir a visitar a sus amigos, María ha reservado para ello el alquiler de un coche. Cuando se sube en él, la PDA descubre que tiene sistema de navegación así que le indica cuál es la dirección a la que se dirige María para que le ayude a seguir el camino correcto. Además ajusta la música al nivel adecuado, y el coche como detecta que María no es francesa le indica algunas de la normas de conducción que debe tener en cuenta en Francia, mediante los altavoces del coche y por supuesto, en su idioma.

2.3. Nuevos retos

En la sección anterior hemos visto algunos escenarios que ilustran situaciones ficticias pero a las que la tecnología está dando respuesta. En las distintas situaciones hemos visto cómo los diferentes dispositivos que hemos descrito en la sección 2.1 le facilitan a María poder irse de viaje, sin tener que estar pendiente de todo lo que esto implica. Es importante darse cuenta que María, como usuaria, lo que percibe es que el entorno que le rodea se adapta a sus preferencias, a María no le importa que su calefacción sea capaz de regularse cooperando con los sensores que tiene situados en su casa, lo que le importa es que su casa siempre tenga la temperatura que ella desea.

Con el ejemplo anterior lo que queremos ilustrar es que la tecnología debe centrarse en proporcionar servicios adaptados al usuario, y por lo tanto, el **modelo de aplicaciones en computación ubicua debe orientarse a proporcionar y componer servicios personalizados al usuario final**. Esta idea rompe con las respuestas iniciales que se han dado al desarrollo de aplicaciones software en dispositivos móviles y ubicuos, en las que el principal objetivo es que una serie de programas software puedan ejecutarse en dispositivos limitados.

Con esta nueva forma de abordar el problema del desarrollo de servicios, hemos realizado un análisis de los componentes software que es necesario incluir en cada uno de los tipos de dispositivos que hemos considerado anteriormente.

- **Dispositivos personales**

  Son los dispositivos más completos y que además acompañan al usuario, en ellos deberán exisistir los siguientes componentes:

  - Anuncio de los servicios que ofrecen.
  - Descubrimiento de servicios alcanzables.
  - Gestión de perfiles y preferencias de usuario.
  - Personalización de servicios.
  - Composición de servicios.
  - Sincronización de datos almacenados entre diferentes dispositivos.
  - Seguridad.

- **Dispositivos de función específica**

  En este caso, los componentes deben facilitar y estar orientados a la realización de la tarea específica para la que han sido fabricados. Así serán necesarios los siguientes componentes:

- Anuncio de los servicios que ofrecen.
- Descubrimiento de servicios alcanzables que puedan facilitar la realización de su tarea.
- Composición de servicios.
- Configuración para que puedan adaptarse a otros sistemas, a los requisitos del usuario o al entorno en el que se encuentren.
- Seguridad.

- **Sensores y actuadores**

  Su limitación y su propósito específico, para proporcionar o realizar una tarea concreta, hace que los componentes necesarios en estos dispositivos se reduzcan a:

  - Anuncio de los servicios que ofrecen.
  - Configuración para que puedan adaptarse a otros sistemas, a los requisitos del usuario o al entorno en el que se encuentren.
  - Seguridad.

Como se observa en la enumeración realizada, existen una serie de componentes comunes a todos los tipos de dispositivos, el planteamiento de esta tesis doctoral se va a centrar en proporcionar tecnologías middleware aplicables a todos los dispositivos, de tal manera que cada modulo compartido funcione del mismo modo en cada uno de ellos y por lo tanto, permita la interoperabilidad, sin limitar la flexibilidad a la hora del desarrollo de nuevos servicios. Así estos módulos comunes son:

- Descubrimiento y/o anuncio de servicios.
- Configuración.
- Seguridad.

Los módulos dependientes del tipo de dispositivo son:

- Composición de servicios.
- Gestión de perfiles y preferencias de usuario.
- Personalización de servicios.
- Sincronización de datos almacenados.

El análisis y definición de todos estos módulos presenta importantes retos para la investigación e involucra un importante trabajo. Esta tesis abordará en primer lugar el **descubrimiento y anuncio de servicios**, y en segundo lugar el **soporte para facilitar la configuración y composición de servicios**. En tareas como la gestión de perfiles y preferencias de usuario, y la seguridad no se realizarán aportaciones, aunque el estudio realizado sobre los requisitos que imponen los entornos ubicuos serán útiles para realizar la definición de soluciones y propuestas en estos retos.

Para definir esta serie de módulos que serán tecnologías básicas para la realización de un middleware que dará soporte al desarrollo de servicios en entornos de computación ubicua, es necesario tener en cuenta las restricciones que impone el entorno, las características hardware/software que tienen los nuevos dispositivos y las caracterísiticas de los diferentes protocolos de red que se emplean.

En cuanto a los entornos, la característica fundamental es que son muy dinámicos y cambiantes. Utilizando nuestro ejemplo, en su casa María puede tener todos sus electrodomésticos interconectados a una red, probablemente cableada y además los elementos que la forma son más o menos estables. En cambio en el aeropuerto o en las calles de París, los elementos con los que puede interaccionar María dependerán en buena medida de su posición, porque podrá alcanzarlos cuando se aproxime a ellos con su PDA. Además el tipo y características de estos dispositivos variará enormente, algunos pertenecerán a otros usuarios, otros estarán embebidos en el entorno físico proporcionando servicios de terceros, etc. Por lo tanto, es necesario que las tecnologías middleware proporcionen soluciones que se adapten a ambas situaciones, para proporcionar una solución que se adapte tanto a entornos dinámicos como estáticos.

En cuanto a los dispositivos, la característica fundamental es su heterogeneidad y las limitaciones de algunos de ellos. En los escenarios en los que se mueve María tenemos desde dispositivos personales, como su PDA o su teléfono móvil, que poseen interfaces amigables para el usuario, que funcionan con baterías y emplean para comunicarse protocolos inalámbricos; hasta dispositivos fijos, con conexión a red fija y sin interfaz como su nevera, hasta los más limitados como son los sensores de temperatura. Por lo tanto, a la hora de definir tecnologías middleware que deben proporcionarse en todos ellos, debemos proponer soluciones simples y flexibles que sean factibles de implementar hasta en los más limitados dispositivos.

En cuanto a los protocolos de red, la característica fundamental es su heterogeneidad y sus diferentes prestaciones en cuanto ancho de banda, calidad de la transmisión y conectividad. En el hogar los electrodomésticos de María pueden estar conectados empleando una red fija de alta velocidad, pero su PDA, su teléfono móvil, y los sensores situados en su hogar emplean protocolos inalámbricos, que se caracterizan por:

- la calidad cambiante de las transmisiones, debido a los errores en el interfaz radio;
- la conectividad no transitiva, debido al corto alcance de algunos protocolos y a la dificultad de imponer soluciones de routing de múltiples saltos;
- el coste de comunicación, debido fundamentalmente al coste de baterías que suponen la transmisión;

Por lo tanto, las soluciones que proporcionemos deben intentar minimizar el número y cantidad de transmisiones a realizar, de manera que las soluciones propuestas sean eficientes tanto en redes fijas como en redes móviles.

Considerando el análisis realizado, además de que los servicios proporcionados en entornos ubicuos deben estar orientados a realizar las tareas que desea el usuario, es necesario también que tengan las siguientes características:

- autonomía propia para poder alcanzar los objetivos que quiere el usuario, pero sin necesidad de interaccionar continuamente con él;
- capacidad de cooperar con otros sistemas para poder obtener información o ejecutar tareas que las limitaciones del dispositivo no les permiten realizar de forma local;

- movilidad para poder ejecutarse en los sistemas que tengan la información localmente, de forma que se minimice el número de transmisiones, se compense las limitaciones de los protocolos inalámbricos y no se dependa de protocolos de routing multisalto.

Analizando estas características hemos considerado que el paradigma de computación que se adapta a todas estas características es el paradigma de agentes móviles. Éste es el motivo que nos lleva a proponer la aplicación de la tecnología de agentes como tecnología middleware para el desarrollo de servicios en entornos de computación ubicua. Aunque por defínción, esta tecnología se adapta a este tipo de entornos y ya existen muchas implementaciones, estudios y estándares al respecto, no se había involucrado a dispositivos limitados ni se habían tenido en cuenta las restricciones que imponen las redes inalámbricas y la posibilidad de formación de redes ad-hoc entre dispositivos limitados, por lo tanto, es necesario realizar un análisis detallado de todos los aspectos desarrollados hasta este momento para adaptarlos a los nuevos requisitos y retos que se nos plantean. En este sentido, centraremos la segunda de las contribuciones que hemos propuesto en esta tesis doctoral.

3. Estado del arte

Primeramente, en esta sección realizamos un estudio sobre las diferentes tecnologías que es necesario tener en cuenta a la hora de realizar las contribuciones propuestas en esta tesis doctoral. En el apartado 3.1 realizamos un estudio de las nuevas tecnologías de red, tanto las inalámbricas que emplean los dispositivos personales, como las que se emplean en entornos del hogar y que utilizan la red eléctrica o la red telefónica como medio de transmisión. En el apartado 3.2 describimos la tecnología que ha permitido solventar el problema de heterogeneidad a la hora de desarrollar aplicaciones que se ejecuten en dispositivos limitados, que es Java 2 Micro Edition (J2ME). Este estudio nos permitirá analizar las necesidades y restricciones que imponen los entornos de computación ubicua, a la hora de definir tecnologías middleware para el desarrollo de servicios en este tipo de entornos.

Además, en esta sección hemos incluido una descripción detallada de las tecnologías directamente relacionadas con las contribuciones que se van a realizar en esta tesis doctoral. En el apartado 3.3 describimos los protocolos de descubrimiento y anuncio de servicios de forma dinámica y tecnologías relacionadas, y concluimos con el análisis que nos ha llevado a la necesidad de realizar una contribución en este campo, a pesar de la posibilidad de aplicar las soluciones existentes. En el apartado 3.4 hemos realizado un estudio de la tecnología de agentes, centrándonos en los agentes móviles y en las implementaciones de plataformas de agentes desarrolladas en Java, analizando la viabilidad de implantar esta tecnología sobre dispositivos limitados J2ME.

3.1. Nuevos protocolos de comunicación

En este apartado realizaremos una introducción a los nuevos protocolos de comunicación que han aparecido estos últimos años, y que están permitiendo que la computación móvil y ubicua sea una realidad. No va a ser objetivo de esta tesis proporcionar ninguna aportación en este campo, pero se considera fundamental como base para entender algunos de los requisitos que se deben imponer, a la hora de definir tecnologías middleware para el desarrollo de aplicaciones distribuidas en este tipo de entornos.

Tipo	Tecnología	Banda de frecuencia	Tasa de datos
2G	GSM	900/1800/1900 MHz	9.6 kbps
2.5G	GPRS	900/1800/1900 MHz	$\leq$ 171 kbps
	EDGE	900/1800/1900 MHz	$\leq$ 384 kbps
3G	UMTS	2 GHz	$\leq$ 2 Mbps

Cuadro 1: Tecnologías de redes inalámbricas de área extensa

Se realiza un breve estudio de los protocolos más importantes en el ámbito de las redes de área extensa, sección 3.1.1, y área local, sección 3.1.2, cubriendo las más importantes tecnologías inalámbricas que han permitido la movilidad de los usuarios. En el apartado 3.1.3 se cubren los protocolos de menos alcance, que se han denominado de área personal porque han sido diseñados para la comunicación entre dispositivos personales del usuario. Por último, en el apartado 3.1.4 se cubren los nuevos protocolos que se están desarrollando para conectar en red los diferentes dispositivos localizados en un hogar.

3.1.1. Redes de Área Extensa

La movilidad de los usuarios se ha conseguido gracias a la aparición de protocolos inalámbricos de área extensa, en concreto con la aparición de las redes de telefonía celular, que han tenido un gran impacto en estos últimos años por su aceptación por parte de un gran número de usuarios. Estas redes conocidas como redes 2G, como GSM, están dando paso a las denominadas redes 2.5G, como GPRS, y a las 3G, como UMTS y actualmente en la literatura ya están apareciendo trabajos de investigación sobre lo que será la siguiente generación, las 4G.

Debido al gran impacto que han tenido las redes de telefonía móvil de segunda generación, como GSM, varios organismos en los últimos años están estandarizando redes móviles de tercera generación (3G). A nivel mundial el ITU está especificando las características comunes que tendrán estas redes para garantizar su interoperabilidad, es lo que se denomina sistema IMT-2000. UMTS es uno de estos sistemas, y será el sucesor en Europa del sistema GSM.

Una característica común de todas estas redes es que se apoyan fuertemente en la infraestructura de red: la cobertura está dada por estaciones base, los recursos de radio están gestionados desde una ubicación central, y los servicios están integrados en el sistema.

En un principio estos sistemas sólo proporcionaban al usuario final un servicio de voz personalizado al que podía acceder en cualquier momento y en cualquier lugar. La demanda por parte de los usuarios de poder acceder con esta misma flexibilidad a los mismos servicios que tienen en redes fijas, inicialmente navegación web y correo electrónico, ha llevado a importantes retos tecnológicos que se han ido abordando a través de las distintas evoluciones de los sistemas de segunda generación.

En el Cuadro 1 resumimos las características de las diferentes tecnologías que describiremos brevemente a lo largo de este apartado.

GPRS *General Packet Radio Service* (GPRS) [GSM 02.60, 1997] es una evolución del sistema de telefonía móvil GSM y es un estándar de transición al sistema de Tercera Generación UMTS. Este sistema ha sido estandarizado por el ETSI pero su implantación no ha tenido lugar hasta

el año 2000.

GPRS [Bettstetter et al., 1999] básicamente añade conmutación de paquetes de datos a todos los niveles de la red GSM, optimizando la utilización de los canales radio para el tráfico a ráfagas (por ejemplo, el tráfico de Internet), y realizando un uso más eficaz de los recursos de la red, de manera que se pueden alcanzar mayores tasas de datos.

La arquitectura del sistema GPRS no utiliza las centrales de conmutación de GSM para el transporte de paquetes de datos, sino que los controladores de estaciones base de radio están directamente conectadas a la red de datos a través de dos nuevos tipos de servidores, también denominados nodos *GPRS Support Node* (GSN), que son de dos tipos:

- *Serving GPRS Support Node* (SGSN), se encarga de la entrega de paquetes a y desde los terminales móviles, también realiza tareas de gestión de movilidad y gestión de sesión, y lleva a cabo funciones de autenticación y tarificación.

- *Gateway GPRS Support Node* (GGSN), actúa como un interfaz entre la red GPRS y la red de datos externa, y también lleva a cabo funciones de autenticación y tarificación.

En el transporte de voz se siguen utilizando los mecanismos tradicionales de GSM.

Desde el punto de vista de los usuarios, GPRS ofrece unos menores tiempos de acceso, conectividad permanente (*always on*) y una tasa de datos mayor que la proporcionada por GSM, teóricamente se pueden alcanzar tasas de hasta 171 kbps, aunque los valores reales se aproximan a los 40 kpbs. Además, la tarificación se realiza según el volumen de información transmitida y no por tiempo de conexión.

EDGE El paso siguiente en la evolución de los sistemas de telefonía móvil fue el sistema *GSM Enhanced Data rates for Global Evolution* (GSM/EDGE) [3G TS 43.051, 2002] que consiste en utilizar una modulación más eficiente en el interfaz radio para poder obtener mayores tasas de transmisión, utilizando las mismas frecuencias en las que opera GSM, lo que supone una gran ventaja a las operadoras en el camino hacia la implantación de sistemas de tercera generación.

En el sistema EDGE [Muller et al., 2001] se define una nueva red de acceso, la *GSM/EDGE Radio Access Network* (GERAN), que ofrece diferentes posibilidades de modulación, que combinadas con diferentes tasas del codificador de canal convolucional utilizado, dan lugar a tasas de transferencia máximas de 384 kbps. Los principales objetivos en la definición de GERAN fueron poder proporcionar servicios de tercera generación a los usuarios, reutilizando la mayor parte de infraestructura existente y que esta red de acceso fuese capaz de interoperar tanto con la *core network* de GSM-GPRS, como con la de los futuros sistemas UMTS.

El ETSI comenzó a estandarizar el sistema EDGE, pero debido a su proximidad con los sistemas de tercera generación, se delegó este trabajo al 3GPP en verano de 2000. El estándar ha evolucionado de forma paralela a las especificaciones de UMTS, de hecho existe la Release 99, Release 4 y Release 5 del mismo.

UMTS El sistema *Universal Mobile Telecommunications System* (UMTS) se engloba dentro de la iniciativa mundial de estandarización de sistemas de telefonía móvil de tercera generación IMT-2000. El objetivo común de estos sistemas es proporcionar al usuario final:

- Convergencia de servicios, que le permitirá acceder a los mismos servicios que le proporcionan las redes fijas, gracias a mayores tasas de transmisión en el interfaz radio (hasta 2Mbps).
- Movilidad personal, que permitirá acceder a los servicios de la red, independientemente del lugar en el que se encuentre o del terminal a través del que se accede a ellos.
- Portabilidad y movilidad de servicios, a través del concepto *Virtual Home Enviroment* (VHE) que le permitirá mantener la personalización de sus servicios, independientemente del lugar en el que se encuentre o del terminal a través del que accede a ellos.

UMTS es el estándar de facto para la evolución de los sistemas GSM. Técnicamente introduce importantes cambios respectos a los sistemas anteriores tanto en la red de acceso, como en la *core network*. El 3GPP es el organismo encargado de su estandarización, y hasta el momento ha publicado las siguientes versiones:

- **Release 99** [3G TS 21.101, 2000] es un estándar firmemente establecido y será el que se utilice en el despliegue inicial de UMTS en todas las operadoras europeas. El objetivo de esta versión es aumentar las tasas de transmisión para poder proporcionar servicios multimedia a los usuarios, conservando parte de la infraestructura de red de GSM-GPRS/EDGE, para minimizar a las operadoras los costes iniciales de implantación. Así, esta versión se centra en definir un nuevo interfaz radio, denominado *UMTS Terrestrial Radio Access Network* (UTRAN).
- **Release 4** [3G TS 21.102, 2001] en esta versión del estándar, la voz se transporta ya sobre IP y aparecen separadas las funciones de control y conectividad para voz.
- **Release 5** [3G TS 21.103, 2002] ésta es la versión denominada como "all IP". IP será la tecnología de transporte en la *core network* para todo tipo de datos, y posiblemente también se empleará en la red de acceso radio UTRAN. En esta versión se culmina la separación entre los planos de transporte y de control.

3.1.2. Redes de Área Local

En los últimos años también han aparecido una serie de estándares y normas para redes LAN inalámbricas, como IEEE 802.11 o HIPERLAN, que proporcionan movilidad a los usuarios en entornos más limitados, pero que les permite acceso a Internet, por lo que están siendo unos claros rivales tecnológicos de las redes celulares 3G. Su modo de funcionamiento habitual es centralizado a través de puntos de acceso de forma similar a las redes anteriores, aunque también tienen modos en los que los terminales pueden comunicarse directamente entre sí.

Cuando los terminales se comunican entre sí, se dice que los nodos forman una red ad-hoc. En [Frodigh et al., 2000] se define una red ad-hoc, como una red formada sin ninguna administración central, en la que los nodos usan una interfaz inalámbrica para comunicarse. Habitualmente estos nodos son móviles por los que estas redes son muy dinámicas, en ellas los dispositivos que las forman aparecen y desaparecen de forma espontánea, por lo que el funcionamiento de la red no puede depender de un nodo concreto.

Tradicionalmente las redes ad-hoc se han empleado en aplicaciones militares en las que una configuración de red descentralizada, es una ventaja operativa o incluso una necesidad. En los últimos años se han definido una serie de protocolos inalámbricos que se están implementado

Tecnología	Banda de frecuencia	Tasa de datos	Alcance
IEEE 802.11b,g	2.4 GHz	11 Mbps	100 m
IEEE 802.11g	2.4 GHz	54 Mbps	100 m
IEEE 802.11a	5 GHz	54 Mbps	100 m
HIPERLAN/2	5 GHz	54 Mbps	150 m

Cuadro 2: Tecnologías inalámbricas de área local

cada vez más habitualmente en los nuevos dispositivos personales, lo que ha permitido que estas redes ad-hoc se formen entre estos nuevos dispositivos y en nuevas situaciones y por lo tanto, que se empleen en un mayor número de escenarios.

En el Cuadro 2 resumimos las características de las diferentes tecnologías que describiremos brevemente a lo largo de este apartado.

IEEE 802.11 El estándar [IEEE 802.11, 1999] ha sido definido con el objetivo de permitir a usuarios inalámbricos conectarse a redes de área local. Está diseñado para ser compatible con el resto de estándares LAN cableados de la familia 802. El estándar define un mismo nivel MAC que usa el algoritmo *Carrier-Sense Multiple-Access with Collision-Avoidance* (CSMA/CA) y varias posibilidades de transmisión a nivel físico: infrarrojos, y las más populares, de espectro ensanchado por secuencia directa (*Direct-Sequence Spread Spectrum*, DSSS) y de espectro ensanchado por salto en frecuencia (*Frequency-Hopping Spread Spectrum*, FHSS) ambas en la banda ISM[1] (2,4 GHz). Se consiguen de esta forma velocidades de 1 y 2 Mbps.

Posteriormente, apareció el IEEE 802.11a que utiliza *Orthogonal Frequency-Division Multiplexing* (OFDM) en la banda de los 5 GHz y que alcanza tasas de hasta 54 Mbps. Para que la Unión Europea aceptase el uso de esta banda de frecuencia, el grupo tuvo que realizar modificaciones al estándar recogidas en el IEEE 802.11h, en la que se incluyeron también mecanismos de control de potencia de transmisión.

Sobre las formas de transmisión del estándar original en la banda ISM, se han definido varias ampliaciones para alcanzar mayores tasas binarias. Aparecen así:

- IEEE 802.11b que utiliza *Complementary Code Keying* (CCK) sobre DSSS y consigue tasas de 5,5 Mbps y 11 Mbps. Ésta es la versión del estándar más utilizada y también es conocida comercialmente como WiFi.
- IEEE 802.11g, también denominado 802.11b extendido, que pretende conseguir tasas de hasta 54 Mbps. Este estándar todavía está en desarrollo y se pretende que sea compatible con las redes 802.11b y 802.11 DSSS.

Otras iniciativas dentro del grupo, algunas todavía en fase de desarrollo, son:

- IEEE 802.11e que se ha creado para introducir calidad de servicio sobre las implementaciones IEEE 802.11b,a y g.
- IEEE 802.11i que se ha creado para mejorar los mecanismos de seguridad e introducir autenticación a nivel MAC.

[1] Industrial-Scientific-Medical

El IEEE 802.11 define dos modos de arquitectura de red, por una parte el modo ad-hoc, en el que las estaciones se comunican entre sí directamente, y por otra el modo centralizado, en el que las estaciones acceden a la red a través de uno o varios puntos de acceso.

HIPERLAN/2 La norma [HIPERLAN/2, 1999] ha sido definida por el ETSI BRAN[2] con el objetivo de dar acceso inalámbrico de alta velocidad a redes de área local. Opera en la banda sin licencia de 5 GHz y alcanza tasas de transmisión de hasta 54 Mbps. La norma especifica una red de acceso radio que se puede utilizar con una variedad de núcleos de red, gracias a una arquitectura flexible que define la capa física y de control de enlace independiente al núcleo de red con el que opere. Además especifica un conjunto de capas de convergencia que facilitan el acceso a los distintos núcleos. Se han definido varias capas de convergencia, en concreto existen definiciones para interoperar con redes IP, redes ATM, redes celulares de tercera generación y redes IEEE 1394.

La capa MAC de HIPERLAN/2 está basada en la aproximación *Time Division Multiple Access, Time Division Duplex* (TDMA/TDD) y ha sido diseñado para dar soporte a calidad de servicio, esencial para aplicaciones multimedia y de tiempo real. A nivel físico emplea la misma técnica que su estándar competidor el IEEE 802.11a, OFDM. En [Doufexi et al., 2002] se puede encontrar un estudio comparativo de ambas tecnologías.

HIPERLAN/2 depende de una topología de red celular y soporta dos modos de operación básicos, el modo centralizado y el modo directo. En el modo de operación centralizado los terminales se comunican entre sí y acceden al núcleo de red a través de puntos de acceso, este modo se utiliza cuando es necesario cubrir una zona amplia y por lo tanto, existen varios puntos de acceso. En el modo de operación directo existe un único punto de acceso que controla la asignación de recursos radio a los diferentes terminales, en este modo los terminales pueden comunicarse directamente entre ellos, a este modo de operación no se le denomina ad-hoc porque el funcionamiento de la red depende de un elemento central, el punto de acceso, para funcionar.

3.1.3. Redes de Área Personal

Paralelamente a los protocolos anteriores, estos últimos años se han desarrollado otros protocolos de más corto alcance, con un modo de operación principalmente ad-hoc. Aparecen así IrDA y Bluetooth, como mecanismos para comunicar dispositivos personales del usuario entre sí, con el objetivo de compartir y sincronizar información entre ellos. Por ello a este tipo de redes se les ha denominado redes de área personal, aunque estos protocolos también se están utilizando en electrónica de consumo y su uso puede extenderse a áreas locales.

En el Cuadro 3 resumimos las características de las diferentes tecnologías que describiremos brevemente a lo largo de este apartado.

IrDA *Infrared Data Association*[3] fue creada en 1993 con el objetivo de desarrollar una tecnología inalámbrica por infrarrojos, de corto alcance, fácil de usar, de bajo coste e interoperable. El éxito de su iniciativa lo demuestra el hecho de que casi todos los ordenadores portátiles, agendas digitales, teléfonos móviles, cámaras digitales... lo soportan actualmente. Se puede decir que hasta la aparición de Bluetooth, esta tecnología era el único protocolo inalámbrico empleado para la transferencia y sincronización de datos entre este tipo de dispositivos. En la actualidad,

[2]European Telecommunications Standards Institute-Broadband Radio Access Networks
[3]http://www.irda.org/

Tecnología	Banda de frecuencia	Tasa de datos	Alcance
IrDA	Infrarrojos	115 kbps 1.152 Mbps 4 Mbps	2 m
Bluetooth v1.1	2.4 GHz	1 Mbps	10/100m
IEEE 802.15.3	2.4 GHz	55 Mbps	10 m
IEEE 802.15.4	868 MHz 915 MHz 2.4 GHz	20 kbps 40 kbps 250 kbps	10-20 m

Cuadro 3: Tecnologías de área personal

mas que tecnologías competidoras son complementarias dependiendo de la aplicación en la que se utilicen [Suvak, 2000].

El núcleo principal de la plataforma IrDA [Williams, 2000], que se denomina IrDA-Data, abarca tres protocolos obligatorios y uno opcional, estandarizados en el año 1994:

- *Infrared physical layer* (IrPHY), que soporta conexiones entre dispositivos a una distancia de 1 a 2 metros, y que para implementaciones de bajo consumo se reduce a 20cm. Se proporcionan tasas de transmisión de 115 kbps, 1.152 Mbps y hasta un máximo de 4 Mbps.
- *Infrared Link Access Procotol* (IrLAP), que proporciona conexiones fiables entre dispositivos para la transmisión de datos y mecanismos de descubrimiento de dispositivos.
- *Infrared Link Management Protocol* (IrLMP), que proporciona multiplexación de canales sobre una conexión IrLAP, y descubrimiento de servicios a través de Information Access Service (IAS).
- *Tiny Transport Protocol* (TinyTP), que proporciona control de flujo sobre conexiones IrLMP. Siendo éste el protocolo opcional.

Para promover la interoperabilidad de las aplicaciones que emplean la plataforma, se han definido un conjunto de protocolos que proporcionan servicios de comunicaciones: IrCOMM que emula la comunicación a través de puerto serie y paralelo y el IrLAN para el acceso a LANs tipo 802. Así como otros protocolos que proporcionan servicios a nivel de aplicación: IrOBEX que permite el intercambio de objetos (semejante a HTTP), IrTRAN-P para el intercambio de imágenes, IrMC para intercambio de información entre teléfonos móviles e IrJetSend que describe como interactuar con el protocolo HP JetSend.

Los principales inconvenientes de IrDA son su rango corto de cobertura, la necesidad de visión directa entre dispositivos, y su modo de operación punto a punto. Estos inconvenientes están siendo solventados parcialmente por lo que se denomina Advanced Infrared (AIR) [IBM Research Zurich, 1999].

Bluetooth El protocolo inalámbrico [Bluetooth v1.1, 2001] surge como una iniciativa de Ericsson en 1994 con el propósito de sustituir los cables entre los dispositivos electrónicos, tales como teléfonos, PDAs, ordenadores portátiles, cámaras digitales, impresoras,... usando

un chip de radio de bajo coste. En 1998, se unen a esta iniciativa IBM, Toshiba, Nokia e Intel formándose el *Bluetooth SIG*[4] para promover y estandarizar esta tecnología.

El protocolo Bluetooth trabaja en la banda ISM. La especificación define un canal de comunicación de máximo 720Kbps en modo asimétrico o 432,6 Kbps en modo simétrico con un rango óptimo de 10m (opcionalmente 100m).

Dos o más unidades Bluetooth que comparten el mismo canal forman una *piconet*. Dentro de una *piconet*, una unidad Bluetooth puede representar uno de dos papeles: maestro o esclavo. Cada *piconet* debe tener un maestro y puede tener hasta siete esclavos activos. Dos o más *piconets* pueden estar interconectadas formando lo que se denomina *scatternet*. Una unidad Bluetooth puede ser simultáneamente un miembro esclavo de múltiples *piconets*, pero solamente maestra en una de ellas. En el protocolo Bluetooth no hay transmisión directa entre esclavos, todas las comunicaciones deben pasar por el maestro, que organiza de acuerdo a un determinado esquema, las comunicaciones en la *piconet*.

El *IEEE 802.15 Working Group* que surge en 1999 con el objetivo de definir una serie de estándares y normas para redes personales inalámbricas (*Wireless Personal Area Networks*, WPAN), decidió utilizar la especificación Bluetooth como base de su estándar [IEEE 802.15.1, 2002] y así en él se estandariza la capa física y MAC de Bluetooth. Este grupo también está desarrollando nuevos estándares para WPAN.

- El *IEEE 802.15 WPAN Task Group 3* se encarga de la definición de un estándar para soportar mayores tasas de datos, por encima de los 20 Mbps, con el objetivo de dar soporte a aplicaciones multimedia y procesado de imagen [Karaoğz, 2001].
- El *IEEE 802.15 WPAN Task Group 4* se encarga de la definición de un estándar que soporta menores tasas de datos, por debajo de los 200 kbps pero que optimiza el uso de las baterías, con el objetivo de ser utilizado en redes de sensores, etiquetas inteligentes, controles remotos [Callaway et al., 2002].

3.1.4. Redes del Hogar

El entorno del hogar ha sido uno de los primeros escenarios en los que se ha querido implantar el concepto de computación ubicua [Dutta-Roy, 1999]. El hogar plantea retos tecnológicos particulares, el más importante es el de conseguir conectar entre sí los habituales electrodomésticos del hogar sin un coste adicional elevado a los usuarios. Así se han desarrollado una serie de protocolos para utilizar la red telefónica y la red eléctrica como medios de transmisión, también se ha empleado tecnología inalámbrica, y existen algunas iniciativas más invasivas como el uso del cable.

En la Cuadro 4 resumimos las características de las diferentes tecnologías que describiremos brevemente a lo largo de este apartado.

HomeRF El *Home RF Working Group*[5] es un consorcio de las principales compañías fabricantes de PCs y electrónica de consumo y de telecomunicación que surge con el objetivo de permitir la interoperabilidad entre redes de voz y de datos en entornos del hogar, a través de un protocolo inalámbrico. El principal trabajo del grupo en estos últimos años ha sido la especificación de este protocolo, denominado *Shared Wireless Access Protocol* (SWAP), que aúna

[4] http://www.bluetooth.com/
[5] http://www.homerf.org

Tecnología	Banda de frecuencia	Tasa de datos	Alcance	Medio físico
HomeRF	2.4 GHz	20 Mbps	50 m	Inalámbrico
HomePNA	4–10 MHz	10 Mbps	–	Línea telefónica
HomePlug	4.3–20.9 MHz	14 Mbps	–	Red eléctrica

Cuadro 4: Tecnologías del hogar

características del estándar *Digital Enhanced Cordless Telephone* (DECT) para la transmisión de voz y de tecnologías WLAN para la transmisión de datos, pero siempre pensando en su uso para el desarrollo de aplicaciones en entornos del hogar.

SWAP [Negus et al., 2000] emplea a nivel físico la técnica FHSS en la banda ISM. A nivel MAC emplea por una parte *Time Division Multiple Access* (TDMA) para servicios interactivos y de tiempo real, y por otra, CSMA/CD para datos asíncronos y permitiendo alcanzar mayores tasas de transmisión, hasta 1,6 Mbps.

SWAP puede operar de dos formas, como red ad-hoc o como red centralizada. En el modo ad-hoc sólo se soportan comunicaciones de datos, todos los nodos son iguales y las funciones de control de red se distribuyen entre todos los nodos. En el modo centralizado, utilizado para conexiones de tiempo real, se precisa un punto de conexión para coordinar el sistema, el punto de conexión actúa como un gateway a la red telefónica conmutada (PSTN) y puede estar conectado a un PC para dar soporte también a servicios de datos. En SWAP el punto de conexión puede emplearse también para realizar gestión de potencia de los dispositivos estableciendo periodos de *wakeup* y *polling*.

Una red SWAP soporta hasta 127 nodos. Estos nodos puede ser de cuatro tipos diferentes:

- Puntos de conexión.
- Dispositivos de voz o datos síncronos, también denominados I-nodes.
- Dispositivos de datos asíncronos, también denominados A-nodes.
- Dispositivos mixtos de voz y datos.

El protocolo SWAP en su versión 2.0 ha añadido mejoras al estándar inicial, permitiendo tasas de datos de hasta 10Mbps y soporte al *roadming* entre varios puntos de conexión.

HomePNA *Home Phone Line Networking Alliance*[6] fue creada en 1998 por los principales fabricantes de ordenadores y semiconductores para seleccionar, promover y estandarizar tecnologías de red en el hogar empleando la red telefónica tradicional.

La utilización de la red telefónica para la transmisión de datos presenta importantes retos debido a los problemas que plantea de impedancias, reflexiones y atenuaciones. HomePNA 1.0, basado en la propuesta de Tut Systems, solventó estos problemas, empleando el rango de 5.5–9.5 MHz y basándose en el estándar IEEE 802.3, pero encapsulando tramas Ethernet en paquetes mayores, de esta forma se obtienen tasas de datos de hasta 1 Mbps, que permiten la compartición de recursos entre PCs (ficheros, acceso a Internet, periféricos), que eran los requisitos iniciales marcados.

[6] http://www.homepna.org

Esta tasa de datos era insuficiente para dar soporte a tráfico multimedia y de entretenimiento, que eran las nuevas aplicaciones que demandaban los usuarios en su hogar, así el grupo estandarizó la versión 2.0, propuesta conjuntamente por Lucent Tecnologies y Broadcom Corporation, en la que se alcanzan tasas de hasta 10 Mbps y se introduce soporte a calidad de servicio. HomePNA 2.0 [Frank and Hollaway, 2000] emplea el rango de 4–10MHz utilizando a nivel físico *Frequency Diversity Quadrature Amplitude Modulation* (FDQAM) que aumenta la robusted de las transmisiones introduciendo redundancia a QAM, a nivel MAC se sigue empleando CSMA/CD, como en las redes IEEE 802.3, pero se introducen ocho niveles de prioridad y un nuevo algoritmo para mejorar la resolución de colisiones denominado *Distributed Fair Priority Queuing* (DFPQ), de esta forma se garantiza un ancho de banda a las aplicaciones y se reduce la latencia que presenta el nivel MAC original de IEEE 802.3.

La alianza sigue trabajando en una nueva versión del estándar, compatible con las anteriores que permita alcanzar tasas de hasta 100 Mbps. Su implantación sigue teniendo algunos problemas asociados a las propias características del cable telefónico, entre ellas está minimizar la interferencia con las tecnologías xDSL empleadas para la conexión a Internet.

Destacar que HomePNA fue concebido para la conexión en red de PCs en un hogar y posteriormente para interconectar también a la red televisiones, videos y dispositivos de entretenimiento, no ha sido diseñado para interconectar otro tipo de electrodomésticos de consumo.

HomePlug El uso de la red eléctrica como medio de transmisión de datos nos permite interconectar en red casi todos los aparatos del hogar, ya que la mayoría de ellos están conectados a ella para obtener la energía necesaria para funcionar. Debido a la variabilidad en frecuencia de la señal eléctrica, éste es un medio complicado para emplear como transmisor de datos, sobre todo si se quieren conseguir unas tasas elevadas.

En los últimos años han aparecido una serie de iniciativas, la mayoría propietarias, que emplean la red eléctrica para formar redes de control de aparatos eléctricos, y por lo tanto no es necesario alcanzar altas tasas de transmisión. Las tecnologías que más han destacado son:

- **X-10** [X10, 1997], fue diseñado en Escocia entre los años 1976 y 1978 con el objetivo de transmitir datos por las líneas de baja tensión a muy baja velocidad (60 bps en EEUU y 50 bps en Europa) y costes muy bajos. Existen tres tipos de dispositivos X-10: los que sólo pueden transmitir órdenes, los que sólo pueden recibirlas y los que pueden enviar/recibir éstas. Los transmisores pueden direccionar hasta 256 receptores. Se puede decir que a dia de hoy es el protocolo más usado en aplicaciones domóticas.

- **LonWorks** [Echelon, 1999], fue presentada por Echelon en 1992 para implementar redes de control distribuidas y automatización. Es una arquitectura descentralizada, extremo a extremo, que permite distribuir la inteligencia entre los sensores y los actuadores instalados en la vivienda y que cubre desde el nivel físico al nivel de aplicación de la mayoría de los proyectos de redes de control. LonTalk es el protocolo que proporciona los servicios de transporte y routing extremo a extremo. Debido al coste de la tecnología se ha implantado más en entornos empresariales que en el hogar.

- **CEBus** [Wacks, 2002], fue presentada por *Electronics Industry Association* (EIA) Americana en 1992 como un estándar abierto para su uso en redes de control. CEBus usa una modulación en espectro ensanchado obteniendo una velocidad media de transmisión de 7500 bps y a nivel MAC es similar a IEEE 802.3. Como parte de la especificación se ha definido también CAL, un lenguaje a nivel aplicación para la comunicación entre dispo-

sitivos. En Europa existe una iniciativa equivalente denominada *European Home System* (EHS).

En Marzo de 2000 se forma *HomePlug Powerline Alliance*[7], una asociación de las principales compañías del sector del hogar que se unen para crear un estándar de redes de alta velocidad utilizando la red eléctrica, para su uso en el hogar. El objetivo es permitir que los consumidores que empleen esta tecnología puedan compartir periféricos, conexiones a Internet, distribución de audio/video, así como otras aplicaciones de entretenimiento [Gardner et al., 2000].

El grupo decidió apoyar la tecnología PowerPacket de Intellon como base para la primera versión del estándar. PowerPacket utiliza OFDM en el rango de frecuencia entre 4.3 MHz–20.9MHz y emplea CSMA/CD a nivel MAC alcanzando una tasa de datos nominal de 14 Mbps. Para dar soporte a tráfico de voz y multimedia se introduce un protocolo multinivel que permite introducir prioridades a determinados paquetes de datos. Intellon considera que empleando su técnica de modulación se podrán alcanzar tasas de datos de hasta 100 Mbps. PowerPacket ha sido diseñado para no interferir con los protocolos de control que emplean la red eléctrica, como los descritos anteriormente.

HomeCNA *Home Cable Network Alliance*[8] es una agrupación de empresas, formada en 2001, que pretende promover la utilización de cable coaxial para su utilización como medio de transmisión dentro del hogar. El trabajo del grupo hasta la actualidad ha sido proponer el rango de frecuencias que emplearán las diferentes aplicaciones que utilizan el coaxial como medio de transmisión, para de esta forma fomentar la utilización de este soporte en redes del hogar.

El grupo tiene como objetivo definir un estándar común teniendo en cuenta algunos de los estándares claves que ya existen en el mercado, entre estos destacan:

- La norma [IEEE Std 1394-1995, 1996], comercialmente conocida como FireWire, define un interfaz digital de bajo coste que integra productos de entretenimiento, comunicaciones y computación en una misma red multimedia, que opera sobre cables de fibra o cobre. Permite comunicaciones peer-to-peer a velocidades de 100, 200 o 400 Mbps y da soporte a transmisiones de datos síncronos y asíncronos. Para desarrollar aplicaciones interoperables sobre redes IEEE 1394 se han definido una arquitectura denominada Home Audio/Video Interoperability (HAVi) [Lea et al., 2000]. La empresa Broadband Home Inc, perteneciente a HomeCNA, propone una extensión propietaria de la tecnología IEEE 1394 sobre cable coaxial, como estándar para la interconexión de productos de audio y video en el hogar.
- La especificación [Cable Home 1.0, 2002] es una iniciativa de CableLabs, asociación de operadores de televisión por cable de EEUU. Su objetivo fundamental es definir una arquitectura que permita garantizar servicios con una determinada calidad en redes del hogar basadas en cable. Para ello se define un modelo físico y un modelo lógico de red. En [Edens, 2001] se encuentra una descripción detallada de ambos modelos.

3.2. Dispositivos

Los avances que ha realizado la microlectrónica en los últimos años en tres aspectos fundamentales:

[7] http://www.homeplug.org
[8] http://www.homecna.org

- la miniaturización, consiguiendo embeber en chips cada vez más pequeños microprocesadores,
- la evolución de las tecnologías de memoria, que permiten mayor capacidad en chips más pequeños,
- el desarrollo de baterías más pequeñas que duran más y que son menos pesadas,

nos abre la posibilidad de que cada vez más dispositivos tengan capacidad de procesamiento y por lo tanto, que se pueda integrar en el mundo de la computación distribuida para ofrecerle a los usuarios nuevos servicios hasta ahora no proporcionados.

Las limitaciones de estos dispositivos hacen que una gran mayoría de ellos no sean dispositivos multipropósito y que el desarrollo de aplicaciones se tenga que realizar a bajo nivel y en lenguajes propietarios, lo que unido a sus propias limitaciones, dificulta su integración en las soluciones tradicionales de computación distribuida. Gracias a la aparición de [Java 2 Micro Edition, 1999] se ha abierto un importante camino para poder homogeneizar los desarrollos en los diversos tipos de dispositivos, en esta sección realizaremos una introducción a esta nueva tecnología.

3.2.1. J2ME: Java 2 Platform, Micro Edition

En 1999, Sun Microsystems anuncia la aparición de Java 2 Micro Edition con el propósito de permitir que aplicaciones Java se ejecuten en dispositivos con potencia de procesamiento limitada, como teléfonos móviles, *pagers*, *palm pilots*, *set-top boxes*, y otros. Una solución que responde a la amplia difusión que están teniendo estos dispositivos en los últimos años y a la demanda de usuarios y proveedores de servicios de recibir/ofrecer nuevas aplicaciones para aumentar las funcionalidades que aportan estos pequeños dispositivos. J2ME conserva, siempre que las limitaciones de los dispositivos lo permiten, compatibilidad con la versión J2SE, de manera que se eliminan las clases o paquetes que no es posible migrar y se definen unos nuevos adaptados a la funcionalidad concreta de un rango de dispositivos.

J2ME en la actualidad abarca dos categorías de dispositivos, por un parte los que se denominan fijos, que poseen conexiones a la red fija y tienen una capacidad de almacenamiento del orden de 2 a 16 megabytes. Por ejemplo, *set-top boxes*, Internet TV y sistemas de navegación de automóvil. Y por otra parte, los dispositivos denominados móviles, que tienen capacidades de almacenamiento limitadas del orden de los 128 kilobytes, con microprocesadores del 16 o 32 bit RISC/CISC y que se comunican a través de conexiones inalámbricas. Dentro de esta categoría están los teléfonos móviles, *palm pilots* y *pagers*.

Una de las principales ventajas de J2ME es que es una arquitectura modular, Figura 1, que se adapta a las limitaciones de los diferentes dispositivos en los que se quiere integrar. La arquitecura J2ME define tres capas:

- **Máquina virtual**. En la actualidad J2ME soporta dos máquinas virtuales: la Java Virtual Machine que se emplea en ediciones J2SE y en J2EE para los dispositivos con procesadores de 32 bit, y la [KVM, 2000] para arquitecturas de 16/32 bits pero con capacidades de almacenamiento limitado.
- **Configuraciones**. Definen una serie de bibliotecas Java que están disponibles para un conjunto de dispositivos, con similares capacidades de procesamiento y memoria. J2ME soporta varias configuraciones, en la actualidad existen dos estandarizadas:

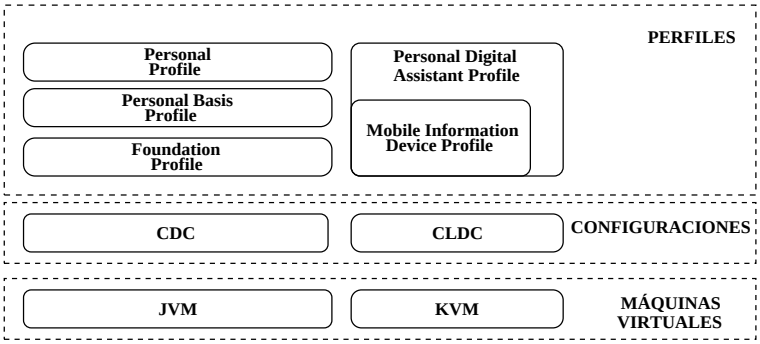

Figura 1: Arquitectura J2ME

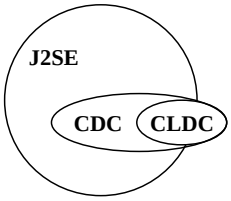

Figura 2: Relación entre CLDC, CDC y J2SE

- **Connected, Limited Device Configuration** [CLDC, 2000], que engloba en general a dispositivos personales móviles.
- **Connected Device Configuration** [CDC, 2002], que engloba en general a dispositivos fijos. Por motivos de compatibilidad es un superconjunto de CLDC.

Ambas configuraciones tiene clases comunes con la J2SE, que permite la compatibilidad, pero poseen además clases específicas para los tipos de dispositivos para los que se definieron, ver Figura 2.

- **Perfiles**. Definen un conjunto de APIs que pueden emplearse para desarrollar aplicaciones para una familia particular de dispositivos. El principal objetivo en la definición de un perfil es garantizar la interoperabilidad de las aplicaciones entre un conjunto de dispositivos que soportan el mismo perfil. Un mismo dispositivo puede soportar diferentes perfiles y los perfiles se desarrollan para una determinada configuración. Ver Cuadro 5.

  Sobre CLDC se han estandarizado:

  - **Mobile Information Device Profile** [MIDP, 2000] para teléfonos móviles, pagers y PDAs de bajo coste.
  - **Personal Digital Assistant Profile**, para asistentes personales. La diferencia añadida respecto al anterior es la parte gráfica que es menos limitada.

  Sobre el CDC se están estandarizado:

  - **Foundation Profile**, para dispositivos embebidos con CPU de 32 bit y sin interfaz de usuario. Este pérfil es la base del *Personal Basis Profile*.

- **Personal Basis Profile**, para dispositivos embebidos con interfaz gráfica limitada. Está construido sobre el *Foundation Profile* y es un subconjunto del *Personal Profile*. Proporciona un subconjunto de AWT, soporte a JavaBeans y un nuevo modelo de aplicaciones: Xlets.
- **Personal Profile**, para dispositivos embebidos pero con una interfaz gráfica mucho más rica que el anterior, este perfil está pensado para *set-top box* y televisión interactiva. Está construido sobre los anteriores e introduce más clases de AWT y soporte a applets.

Ligadas a la arquitectura J2ME, existen una serie de APIs que se han ido definiendo para cubrir funcionalidades específicas de determinados tipos de dispositivos. La mayoría de ellas pueden emplearse como extensión a cualquier perfil J2ME porque se han desarrollado sobre la parte común de CLDC y CDC. A continuación, describimos brevemente algunos de ellos, en [Ortiz, 2002] se puede encontrar una tutorial más detallado:

- Java APIs for Bluetooth, que permite el uso de conexiones Bluetooth y da soporte al protocolo *Object Exchange* (OBEX).
- Wireless Messaging API, que proporciona un API para el envío y recepción de *Short Messaging Service* (SMS).
- Mobile Media API, que da soporte a procesado multimedia.

Dentro de *Java Community Proccess*[9] también se están definiendo paquetes opcionales unidos a determinadas configuraciones, como son el *RMI Optional Package* y el *JDBC Optional Package for Foundation Profile*.

Relacionando la clasificación que hemos realizado de dispositivos en la sección 2 con la arquitectura J2ME, podemos decir que nuestros dispositivos personales se corresponden con la configuración CLDC y los dispositivos de función específica con la configuración CDC. En principio, J2ME no da soporte a los dispositivos más limitados, que nosotros hemos denominado sensores y actuadores, podriamos pensar que en estos dispositivos es difícil llegar a introducir un máquina virtual Java, pero teniendo en cuenta que existe la especificación Java Card parece que, aunque no existe ninguna iniciativa en curso, es viable su realización en el futuro.

Java Card [Chen, 2000] es una iniciativa anterior a J2ME, y consiste en introducir la tecnología Java en una tarjeta inteligente. Las tarjetas inteligentes suelen tener CPUs de 8 bits, RAM de tan sólo 512 bits y memoria no volátil, EEPROM, de 24KB. El API que proporciona Java Card es limitado y muy adaptado a la funcionalidad tradicional de las tarjetas inteligentes que es la seguridad. Del mismo modo creemos que se podrán desarrollar iniciativas semejantes para *smart labels*, sensores, actuadores,...

3.3. Protocolos de descubrimiento de servicios

El descubrimiento de servicios de manera dinámica no es algo nuevo, existen varias propuestas al respecto, más o menos extendidas en entornos de redes fijas [Richard III, 2000] [Helal, 2002]. El objetivo principal con el que surgieron estos protocolos fue facilitar a un dispositivo móvil el descubrimiento, configuración y uso de los servicios que ofrecía la nueva red

[9] Comunidad encargada de la especificación de nuevos APIs y tecnologías Java

Configuración	**Perfil**	**Dispositivo**	**Requisitos**
CLDC	*Mobile Information Device Profile* (**MIDP**)	Teléfonos móviles, pagers, PDAs de bajo coste	256KB ROM 128KB RAM
	Personal Digital Assistant Profile (**PDAP**)	PDAs que gestionan información personal y que se sincronizan con PC o otras PDA	1MB (ROM + RAM)
CDC	*Foundation Profile* (**FP**)	Dispositivos embebidos sin interfaz de usuario.	1MB ROM 512KB RAM
	Personal Basic Profile (**PBP**)	Dispositivos con interfaz de usuario sencilla, electrónica de consumo del hogar lavadores, neveras . . . y de oficina, faxes, impresoras . . . , dispositivos de automóvil. Construido sobre el FP.	2MB ROM 1MB RAM
	Personal Profile (**PP**)	Dispositivos con un interfaz gráfica más compleja, como *set-top boxes*, televisiones, PDAs de alto coste. Construido sobre el PBP	2.5MB ROM 1MB RAM

Cuadro 5: Configuraciones y perfiles J2ME

Funcionalidad	SLP	Salutation	SSDP	SDP
Descubrimiento de servicios	√	√	√	√
Anuncio de servicios	√	√	√	
Registro de servicios	√	√		√
Interoperabilidad	√	√		√
Seguridad	√			√

Cuadro 6: Funcionalidad de los protocolos de descubrimiento de servicios

a la que se conectaba. En general, las diferentes propuestas abordan el problema de distintas formas, algunas han sido diseñados para su uso con un determinado protocolo de red, otras para usar con un determinado lenguaje de programación,... lo que ha provocado que no operen entre sí, este problema ha sido solventado parcialmente con la definición de pasarelas entre algunos de estos protocolos.

En este apartado realizaremos una introducción a los más importantes protocolos de descubrimiento de servicios para ver sus principales características y su estado de desarrollo actual. En concreto, vamos a describir: SLP, Salutation, SSDP de UPnP y SDP de Bluetooth. En la Cuadro 6 podemos ver una comparativa de las distintas funcionalidades que ofrecen cada uno de ellos. También hemos incluido en este apartado la descripción de cómo se realiza el descubrimiento de servicios en algunos entornos de desarrollo de servicios para entornos dinámicos, que han aparecido estos últimos años, así describiremos brevemente el descubrimiento de servicios en Jini, en OSGi y en JXTA.

3.3.1. Service Location Protocol

Service Location Protocol (SLP) [RFC 2608, 1999] es el principal resultado del *Service Location Protocol Working Group* del IETF (SVRLOC). Su objetivo es la definición de un protocolo de descubrimiento automático de servicios en redes IP, descentralizado y extensible. Emplea URLs para la descripción de los servicios, y basándose en ellas los usuarios (aplicaciones) pueden conocer qué servicios existen en la red y acceder a ellos.

La infraestructura SLP consiste en tres tipos de agentes: *User Agents* (UAs), que son los que realizan el descubrimiento de servicios para satisfacer las necesidades que demandan las aplicaciones de los usuarios finales, *Service Agents* (SAs), que son los responsables de anunciar las características y localización de los servicios y *Directory Agents* (DAs), que son los responsables de almacenar información sobre los servicios que se están anunciando en la red. SLP tiene dos modos de funcionamiento, con DAs, en cuyo caso los SAs registran en ellos los servicios que ofrecen y los UAs buscan en ellos los servicios que precisan; o sin DAs, en cuyo caso los UAs envían por multicast peticiones de servicios, a las que los SAs que ofrecen el servicio responden mediante mensajes unicast. En el RFC se comenta que la existencia de DAs mejora sustancialmente las prestaciones del protocolo.

SLP define varios mecanismos para descubrir DAs, el primero de ellos es el modo pasivo, en el que los SAs y los UAs escuchan los mensajes multicast enviados por los DAs en los que anuncian su existencia periodicamente; y el segundo de ellos es el modo activo, en el que los SAs y UAs envían un mensaje multicast o utilizan DHCP para descubrir los DAs existentes, si existe algún DA, los UAs y SAs utilizan comunicación unicast con ese DA para buscar o registrar servicios, respectivamente.

El protocolo sólo proporciona un mecanismo de búsqueda de servicios y en ningún momento aborda cómo los clientes hacen uso de los servicios. Introduce algunos mecanismos de seguridad, sobre todo para garantizar que no se propague información falsa sobre la localización de servicios.

3.3.2. Salutation

Salutation es una arquitectura para el descubrimiento y anuncio de servicios desarrollada por el *Salutation Consortium*[10], agrupación de compañías y centros académicos, que surge con el objetivo de resolver el problema del descubrimiento y acceso a servicios entre un amplio conjunto de dispositivos y equipos en un entorno cambiante, independientemente del protocolo de transporte particular que se esté utilizando.

Salutation [Miller and Pascoe, 2000] es un estándar abierto independiente de sistemas operativos, protocolos de comunicaciones y plataformas hardware. La arquitectura Salutation define tres componentes:

- *Functional Units* que desde el punto de vista de un cliente definen un servicio. Para alguno de los servicios más habituales, como impresoras, faxes o almacenamiento de documentos, el consorcio está definiendo estas unidades de forma que se garantiza la interoperabilidad.

- *Salutation Managers* (SLMs) que permiten a los clientes descubrir y comunicarse con los diferentes servicios proporcionados en la red. El proceso de descubrimiento de servicios puede realizarse a través de múltiples SLMs. Un SLM puede descubrir otros SLMs remotos y determinar los servicios que están registrados allí. De esta manera el descubrimiento de servicios se realiza de una manera mucho más rápida.

- *Transport Managers* (TMs) que permiten aislar al SLM del protocolo de transporte que se está empleando para acceder a él. De esta forma, para dar soporte a un nuevo protocolo de transporte sólo es necesario implementar un nuevo TM, sin modificar la implementación del SLM.

Salutation no sólo define mecanismos de descubrimiento, sino que en la especificación también se describen mecanismos para el acceso a los servicios. Este soporte se proporciona a través del SLM que tiene tres modos de operación: *native personality*, en la que sólo se emplea para descubrir servicios y para establecer la comunicación entre los clientes y el servidor; *emulated personality*, que además de establecer la conexión sirve como pasarela entre el protocolo que emplea el cliente y el que emplea el servidor cuando son diferentes; y *salutation personality*, en el que además de establecer la conexión obliga a que la comunicación entre cliente y servidor se realice en un formato determinado.

Para dispositivos limitados se ha definido una simplificación del protocolo denominada Salutation-Lite, que se centra fundamentalmente en dar soporte al descubrimiento dinámico de servicios. En esta versión no sólo se ha tenido en cuenta la limitación de los dispositivos, sino también el limitado ancho de banda que proporcionan los protocolos de comunicación que suelen emplear, como son IrDA o Bluetooth.

[10] http://www.salutation.org

3.3.3. Simple Service Discovery Protocol

Universal Plug-and-Play (UPnP) es una arquitectura propuesta para el descubrimiento, anuncio y uso de servicios en entornos dinámicos centrada en estandarizar protocolos de comunicación basados en XML, para la interacción entre los diferentes elementos de la arquitectura. El *UPnP Forum*[11] es promotor de esta iniciativa, que está liderada por Microsoft.

Simple Service Discovery Protocol (SSDP) [Goland et al., 1999] es el protocolo que se ha definido dentro de UPnP para el descubrimiento dinámico de servicios. SSDP puede operar con o sin un elemento central, denominado *Service Directory* en la red. SSDP está contruido sobre HTTP, empleando tanto unicast, HTTPU[12], como multicast, HTTPMU[13]. Cuando un servicio quiere unirse a la red, primero envía un mensaje de anuncio con el fin de notificar al resto de los dispositivos su presencia, si está presente en la red un *Service Directory*, éste registra el servicio anunciado. También existe la opción de enviar un mensaje unicast directamente al *Service Directory*, para que éste registre el servicio. Los mensajes de anuncio contienen una URI que identifica el servicio anunciado y una URL a un archivo XML que proporciona una descripción de dicho servicio. Cuando un cliente quiere descubrir un servicio, puede tanto contactar con el servicio directamente a través de la URL que ha obtenido en alguno de los anuncios que ha escuchado anteriormente, o puede enviar activamente un mensaje de búsqueda de servicio por multicast. En el caso de descubrir un servicio a través de un mensaje de búsqueda, la respuesta puede ser proporcionada por el propio servicio o por un *Service Directory*.

SSDP se presentó como un draft al SVRLOC del IETF, con el objetivo de realizar una simplificación de SLP, y que de esta forma se pudiera implementar en un mayor número de dispositivos. En la actualidad este draft está obsoleto.

En UPnP no se aborda la seguridad a nivel SSDP.

3.3.4. Bluetooth Service Discovery Protocol

El SIG Bluetooth además del protocolo de comunicación propiamente dicho ha definido una serie de protocolos a nivel aplicación que facilitan el desarrollo de aplicaciones Bluetooth. Entre ellos se encuentra el *Service Discovery Protocol* [SDP, 2001]. Este protocolo le permite a un dispositivo Bluetooth conocer cuáles son los servicios que proporcionan los dispositivos con lo que tiene conectividad. Este protocolo sigue un esquema clásico cliente/servidor en el que la descripción de los servicios se almacena como un *service record* en el dispositivo que actúa como servidor SDP, que, normalmente se corresponde con el que ejerce de maestro en la *piconet* Bluetooth. SDP soporta búsquedas por clases de servicios, y por atributos de servicios, la distinción entre clases y atributos no está bien definida por lo que es necesario que los fabricantes se pongan de acuerdo en estas definiciones para que este protocolo sea interoperable a nivel práctico.

SDP sólo permite el descubrimiento de servicios y no entra en cómo se accede a ellos, se considera que se debe definir en los protocolos de más alto nivel, que hacen uso de SDP. Sin embargo, dentro de SDP se define un atributo común a todos los servicios, `ProtocolDescriptionList`, en el que se define la lista de protocolos con los que se puede acceder al servicio.

[11] http://www.upnp.org
[12] Variante de HTTP sobre UDP
[13] Variante de HTTP que permite mensajes multicast sobre UDP

3.3.5. Descubrimiento de servicios en Jini

[Jini, 1999] es una arquitectura distribuida orientada a servicios desarrollada por Sun Microsystems, su principal objetivo es convertir a la red en una herramienta flexible y fácilmente administrable en la que los clientes puedan encontrar servicios de un modo robusto y flexible. Jini se apoya fuertemente en Java, de tal forma que en Jini es necesario que todos los dispositivos tengan una máquina virtual Java, o un dispositivo, que si que la tenga, y que actúe en su nombre.

En la arquitectura Jini se definen tres componentes: los servicios, los clientes y los servicios de directorio, que se denominan *Jini Lookup Services* (JLS). Un servicio en Jini está representado por un objeto Java y se define como una entidad software que proporciona algún tipo de cálculo o control sobre un dispositivo. Un cliente es aquel que hace uso de un servicio, y por lo tanto un servicio puede ser a su vez cliente de otro.

El JLS es obligatorio dentro de la arquitectura de Jini, y los clientes y los servicios siempre se descubren a través de él y nunca de forma directa. Para registrar o buscar un servicio primero es necesario localizar algún JLS, para ello se han definido tres tipos de protocolos: *unicast discovery protocol*, empleado cuando ya se conoce un JLS, *multicast request protocol* empleado para buscar un JLS y *multicast announcement protocol* empleado por los JLS para anunciar su disponibilidad. Para soportar el dinamismo de la red y que los servicios registrados en el JLS no se queden obsoletos, Jini implementa un mecanismo de *leasing* que obliga a que los servicios actualicen su registro periodicamente para mantener su entrada en un JLS y de esta forma poder ser localizados por los clientes.

En Jini se define no sólo un mecanismo de descubrimiento de servicios, sino también un mecanismo de uso: un cliente le solicita a un JLS la búsqueda de un servicio proporcionándole un identificador del servicio, un interfaz Java o un conjunto de atributos, el JLS responde a esta búsqueda con un conjunto de objetos *proxies* que permitirán al cliente acceder a implementaciones de dicho servicio. El mecanismo de comunicación entre clientes y servicios se basa en *Java Remote Method Invocation* (RMI).

En cuanto a seguridad, Jini depende del modelo de seguridad de Java.

3.3.6. Descubrimiento de servicios en OSGi

Open Services Gateway Initiative (OSGi[14]) fue creada en Marzo de 1999 con el objetivo de definir una especificación software abierta que permita diseñar y construir plataformas compatibles que sean capaces de proporcionar múltiples servicios en redes locales en general, aunque su uso actual está muy centrado en redes del hogar. Aunque OSGi define su propia arquitectura se ha tenido en mente su compatibilidad con otras iniciativas semejantes como son Jini o UPnP.

La arquitectura de OSGi [Marples and Kriens, 2001] tiene dos elementos fundamentales, por un parte una plataforma de ejecución de aplicaciones basada en Java y por otra, una serie de paquetes (*bundles*) que proporcionan una determinada funcionalidad a otros paquetes o directamente al usuario final y que se ejecutan sobre la plataforma. Estos elementos residen siempre en un elemento central que se denomina *OSGi service platform* situada en la red local y conectada al proveedor de servicios a través de una pasarela en la red del operador. Este elemento además será el encargado de permitir la interacción entre dispositivos o redes de dispositivos que empleen distintas tecnologías para comunicarse.

[14] http://www.osgi.org

En OSGi un servicio está proporcionado por un *bundle* que se ejecuta en la plataforma OSGi. En la plataforma existe el *OSGi service registry* que actúa como un servicio de directorios en el que los *bundles* se registran y a través del cual pueden localizar a otros *bundles* para componer nuevos servicios.

En la especificación de OSGi se han definido una serie de APIs básicas para el desarrollo de servicios, como el de logging, servidor HTTP y el que se denomina *Device Access Specification* (DAS) que permite el descubrimiento y anuncio dinámico de dispositivos y de los servicios ofrecidos por éstos. En DAS se definen los *Network Bundles* que son los encargados de descubrir nuevos dispositivos y servicios empleando el protocolo de descubrimiento correspondiente en esa red, una vez obtenido esta información, deben obtener del proveedor de servicios el *Device Bundle* correspondiente al dispositivo concreto, que se instalará en la plataforma y se registrará en *OSGi service registry* y de esta forma podrá ser descubierto por otros *bundles*.

Mediante este mecanismo OSGi permite la integración de cualquier protocolo de descubrimiento y anuncio de servicios dentro de su plataforma. En [Dobrev et al., 2002] se muestran dos ejemplos detallados de integración de Jini–OSGi y de UPnP–OSGi.

3.3.7. Descubrimiento de servicios en JXTA

El proyecto JXTA[15] es una plataforma de computación distribuida *peer-to-peer* (P2P) concebida y promovida inicialmente por Sun Microsystems a principios de 2001, pero en la que en la actualidad participan un gran número de centros de investigación académicos e industriales. JXTA proporciona una serie de tecnologías middleware por encima de las cuales se pueden construir servicios y aplicaciones. El objetivo principal del proyecto es crear un soporte software para sistemas P2P que permita la interoperabilidad entre diferentes implementaciones de un mismo servicio, la independencia de la plataforma, del lenguaje y entorno de programación en el que se desarrolla el servicio y la ubicuidad de los servicios, permitiendo incluir en el sistema desde los grandes servidores hasta los dispositivos más limitados como PDAs o electrónica de consumo.

Para alcanzar este objetivo JXTA [Gong, 2001] ha definido una serie de protocolos básicos, entre ellos se encuentra el *Peer Discovery Protocol* que estandariza el formato de mensajes que permite a un *peer*, o elemento del sistema, encontrar a otros *peers*, servicios, etc. Este protocolo emplea a su vez el *Peer Resolver Protocol* que es un protocolo genérico que permite enviar y recibir peticiones de búsquedas. A su vez este protocolo emplea el *Rendezvous Protocol* que está basado en que existe un *peer* especial en el sistema que guarda información sobre los *peers*, servicios, etc. que conoce, de tal forma que puede proporcionar esta información a otro *peer* que establezca una comunicación con él.

El *Peer Discovery Protocol* es el protocolo de descubrimiento por defecto que debe implementarse obligatoriamente en todos los sistemas JXTA. Además en el prototipo de implementación de la versión 1.0 se han desarrollado otros mecanismos opcionales para el descubrimiento, que son los siguientes:

- *LAN-based*, en el que el descubrimiento se realiza mediante mensajes broadcast.
- *Invitation*, en el que un *peer* proporciona información del sistema a otro *peer* sin solicitud previa de éste.

[15] http://www.jxta.org

- *Cascade*, en el que los *peers* se van proporcionando información sobre los otros *peers* que conoce, previa autorización de estos.

Dentro del proyecto JXTA, existe la iniciativa JXME para introducir la tecnología JXTA en dispositivos limitados J2ME, la implementación se basa en que estos dispositivos tienen un JXTA *relay* asociado, tipicamente un PC próximo, que actúa en su nombre y que le permite integrarse en la red P2P JXTA.

3.3.8. Pasarelas entre protocolos

A día de hoy la implantación de estos protocolos no ha sido muy amplia, por lo que ninguno de ellos se ha situado como un estándar de facto para el descubrimiento dinámico de servicios. Esto ha llevado a una necesidad mucho más acuciante de conseguir que estos protocolos interoperen entre sí. Un ejemplo de ello, es la especificación OSGi en la que ya se ha tenido en cuenta esta heterogeneidad para integrar los diferentes protocolos existentes dentro de la plataforma. Otras de las propuestas que se han realizado en este sentido han sido:

- **Jini–SLP** [Guttman and Kempf, 1999], en el que se ha creado un *User Agent* SLP especial que registra los *Service Agent* SLP con capacidad Jini.
- **Salutation–SDP Bluetooth** [Miller and Pascoe, 1999] , en el se consideran dos aproximaciones, la primera mapea los APIs de Salutation a los de SDP–Bluetooth para implementar Salutation sobre SDP; la segunda utiliza un *Transport Manager* Bluetooth y reemplaza SDP por Salutation.

3.3.9. Conclusiones

En entornos de computación ubicua es imprescindible proporcionar un mecanismo de descubrimiento y anuncio de servicios, debido al dinamismo de los entornos y al mayor número y diversidad de dispositivos que aparecen y que pueden ofrecer y demandar servicios. En los apartados anteriores, hemos visto que existen varias soluciones planteadas para el descubrimiento y anuncio de servicios. Algunas de las propuestas son protocolos de descubrimiento y anuncio como tal, y otras forman parte de la arquitectura de un *framework* de desarrollo de servicios en entornos dinámicos, por lo que su utilización está muy ligada a la forma en la que se desarrollan y se proporcionan estos servicios.

Aunque estas soluciones han tenido en cuenta el dinamismo del entorno provocado por la movilidad de los dispositivos que ofrecen y demandan lo servicios, la mayoría no han tenido en cuenta las limitaciones de los dispositivos que aparecen en estos entornos, ni la posibilidad de que la redes ad-hoc que se forman entre ellos de forma espontánea, no incluyan ningún elemento fijo sin limitaciones, tipo PC. Así, en la mayoría de ellas aparece un servidor central en el que los que ofrecen servicios se registran y al que los clientes preguntan cuando quieren encontrar un determinado servicio. En entornos de computación ubicua las redes son muy dinámicas y los dispositivos tienen capacidades de almacenamiento limitadas, por lo que es muy costoso que uno de ellos afronte el papel de servidor.

Partiendo de la necesidad de no tener un servidor central, se nos plantean dos posibilidades:

- Métodos *push*: se producen anuncios continuos de los servicios y los clientes escuchan estas peticiones seleccionando el servicio que les interesa.

- Métodos *pull*: son los clientes los que realizan peticiones continuamente para que los servicios se descubran bajo demanda.

Un factor importante a tener en cuenta en estas soluciones en las que, o los anuncios de los servicios, o las peticiones, se envían a todos los dispositivos que están a su alrededor como mensajes de broadcast, suponen un gran gasto de las baterías de los dispositivos y por lo tanto, tiene que optimizarse y valorar cuál es el mejor método para que el descubrimiento de servicios se realice lo antes posible sin implicar un gran número de transmisiones.

Recordemos además, que en entornos tan dinámicos los servicios estarán disponibles un tiempo limitado, por lo que tenemos el compromiso de implementar mecanismos que permitan detectar lo antes posible tanto la disponibilidad como indisponibilidad de esos servicios.

Vemos entonces la necesidad de definir una tecnología para el descubrimiento y anuncio de servicios adaptada a entornos ubicuos, que tenga las siguientes características:

- capaz de adaptarse tanto a entornos dinámicos como estáticos,
- capaz de funcionar sin elementos centrales,
- simple y poco costoso computacionalmente para que sea implementable en dispositivos limitados,
- capaz de minimizar el número y cantidad de información a transmitir para que se adapte a las restricciones de los protocolos inalámbricos.

3.4. Agentes

La tecnología de agentes ha tenido un especial impacto en los últimos años gracias a su aplicación en la computación distribuida. El modelo más ampliamente extendido es el modelo cliente/servidor, en el que un cliente que se ejecuta en un entorno envía un conjunto de datos a un servidor, y espera que éste le envíe los datos de respuesta de la operación realizada, antes de enviar nuevos datos. Cada mensaje intercambiado en la red implica una petición de un servicio y una respuesta a esa petición. La comunicación establecida precisa de una conexión permanente. Esto provoca el consumo de un gran número de recursos.

La introducción de la tecnología de agentes permite realizar las mismas operaciones pero con la ventaja de que sean asíncronas y que además no precisemos conexiones permanentes para la ejecución de tareas, puesto que el agente que migra hacia otro sistema además de llevar los datos necesarios para realizar la operación, conserva la información de estado del proceso.

A pesar de estos beneficios, debemos tener en mente que la tecnología de agentes no sustituye a otros paradigmas de la computación distribuida, una aplicación que se desarrolla con tecnología de agentes puede desarrollarse con las tecnologías convencionales. Lo que debemos analizar es si, para una aplicación determinada, con la tecnología de agentes se obtienen mejores prestaciones, y en ese caso aplicarla. Existen algunos campos de aplicación en los que se ha demostrado que el uso de agentes es mucho más beneficioso: en concreto se han realizado estudios en el caso de acceso a bases de datos [Papastravrou et al., 1999] y en tareas de gestión de red [Baldi and Picco, 1998] [Glitho and Magedanz, 2002].

Existen dos conceptos fundamentales en los sistemas de agentes móviles, que son el de agente y el de plataforma de agente.

3.4.1. Agentes móviles

Un agente [Nwana, 1996] se define como una entidad software que actúa en nombre de otra entidad, por ejemplo, de una persona o de otro agente, que es autónomo y se comporta según los objetivos que debe alcanzar y que reacciona ante eventos externos y puede comunicarse y colaborar con otros agentes.

Un agente móvil es un agente que puede migrar entre dos nodos de una red. Junto a este tipo de agentes, surgieron también los agentes inteligentes, basados en aplicar conceptos de inteligencia artificial al paradigma de agentes. Un agente móvil puede ser inteligente y a la inversa, pero ambas características son independientes.

Para aclarar la definición de agente móvil es necesario indicar que es lo qué entendemos por migración de un agente entre dos nodos. Una entidad software en ejecución está compuesta por el código, que nos proporciona la descripción estática de su comportamiento, y su estado, que nos proporciona su descripción en un momento determinado de ejecución. Migrar un agente consiste en enviar su código y el estado de ejecución al host remoto, de forma que después de su migración en el host remoto se retome su ejecución en el mismo punto en el que se encontraba antes de migrar.

3.4.2. Beneficios de los agentes móviles

Tradicionalmente, los agentes móviles se han contemplado como una alternativa viable en la que la transmisión del código necesario para realizar una tarea resultaba más económico (en términos del tráfico necesario) que la orientación cliente/servidor. En este sentido en [Glitho and Pierre, 2002] se describe una comparativa de sincronizaciones de agendas entre máquinas remotas en tres casos: el primero basado en cliente/servidor, el segundo basado en optimizar el servidor para la operación de sincronización, y el tercero basado en agentes móviles. Las conclusiones del trabajo fueron que, en términos de datos transmitidos, la alternativa de cliente/servidor crecía linealmente con el número de participantes, mientras que las otras permanecían casi constantes y a un nivel muy inferior. Hay que hacer notar que el escenario utilizaba sólo dos servidores que almacenaban las agendas. La segunda conclusión era que el coste de desarrollar el servidor optimizado fue bastante grande, esfuerzo que no fue necesario en el caso de agentes móviles.

Además del menor consumo de recursos de comunicación y del menor coste de desarrollo de aplicaciones concretas, el paradigma de agentes y sistemas multiagentes plantea otros beneficios adicionales, como son la cooperación entre agentes y el diseño de sistemas dirigidos al usuario, en los que el agente personal del usuario cobra una gran importancia. Como última ventaja citaremos la gran diversidad de dispositivos a disposición del usuario, que los agentes móviles pueden integrar para facilitar sus tareas y dar una visión de servicios ubicuos, en lugar de servicios diversos prestados por distinto software corriendo en distintos dispositivos.

3.4.3. Plataformas de agentes

Los agentes móviles requieren un entorno especial para su ejecución. Este entorno es lo que se denomina plataforma. La plataforma además de aportar la capacidad de ejecución propiamente dicha, debe proporcionar una serie de servicios básicos:

- Comunicaciones: que facilite la comunicación de los agentes con el mundo exterior, prin-

cipalmente, que le permitan interactuar con otros agentes en tareas cooperativas.

- Nombrado: que permite nombrar a los agentes de manera que puedan ser indentificados de forma unívoca, y también nombrar a las plataformas y nodos a los que migran los agentes.

- Descubrimiento y localización: que facilite a los agentes descubrir otros agentes y plataformas con los que puede interactuar, o a las que puede migrar, para alcanzar los objetivos que tiene encomendados.

- Movilidad: que facilite la migración de agentes entre nodos remotos.

- Seguridad: que garantice por una parte la protección de los agentes ante ataques del host en el que se ejecuta y por otra, la protección del host ante los ataques de los agentes que se ejecuten en él.

Existen diversas plataformas de agentes móviles entre las que cabe destacar Aglets de IBM [Lange and Oshima, 1998], Voyager de ObjectSpace [Glass, 1999], Grasshopper de IKV [Grasshopper, 1998] y JADE de la Universidad de Parma [Bellifemine et al., 1999].

3.4.4. Estándar FIPA

Foundation for Intelligent Physical Agents[16] (FIPA) comenzó sus actividades en 1995 con el objetivo de estandizar aspectos relacionados con la tecnología de agentes y sistemas multiagente. En la actualidad se puede considerar el estándar más ampliamente reconocido y extendido internacionalmente, y se ha convertido en un referente a seguir a la hora de realizar desarrollos basados en agentes.

Las especificaciones FIPA se han agrupado en tres tipos: *component*, que se encargan de estandarizar todas las tecnologías básicas relacionadas con agentes, *informative* que describen posibles soluciones en aplicaciones realizadas con agentes en un determinado dominio y *profiles* que son conjuntos de especificaciones de tipo *component* que permiten validar cuándo una implementación es conforme al estándar.

En *FIPA Agent Management Specification* [SC00023J, 2002] se describe el modelo de referencia FIPA de plataforma de agentes y se describe la funcionalidad de cada uno de sus componentes, ver Figura 3. Éstos componentes son:

- **Agent Management System** (AMS): que gestiona el ciclo de vida de los agentes, los recursos locales, y los canales de comunicación y proporciona un servicio de páginas blancas, que permite localizar agentes por su nombre.

- **Directory Facilitator** (DF) : que proporciona un servicio de páginas amarillas, que permite localizar agentes por sus capacidades y no por su nombre.

- **Agent Communication Channel** (ACC): que gestiona el envío de mensajes entre agentes de la misma plataforma o de plataformas distintas, y permite la migración de agentes.

[16] http://www.fipa.org

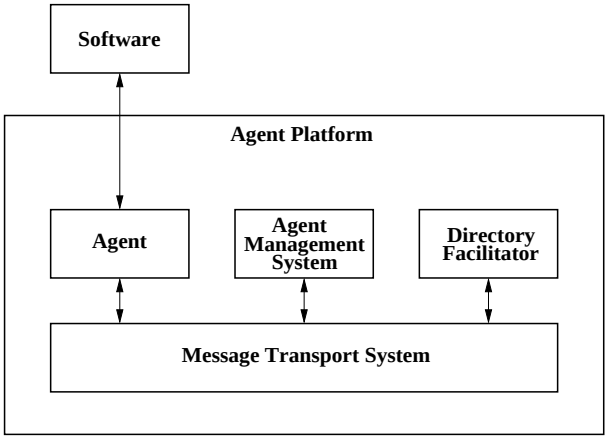

Figura 3: Modelo de referencia FIPA

3.4.5. Lenguajes de programación

Una parte importante del trabajo realizado en el desarrollo de plataformas de agentes ha sido proporcionar o adaptar lenguajes de programación para implementar estas plataformas [Cugola et al., 1997]. Se han utilizado lenguajes de propósito más general como pueden ser Java o adaptaciones de otros, como las versiones realizadas sobre Tcl: Safe-Tcl [Borenstein, 1994], Agent Tcl [Kotz et al., 1999], TACOMA [Johansen et al., 1999] y en algunos casos, se han definido lenguajes específicos para este propósito: Telescript [White, 1995], MO [Tschudin, 1994] y Tycoon [Matthes et al., 1994].

La característica principal que deben proporcionar estos lenguajes es permitir la movilidad de una unidad de ejecución, que consiste en una parte de código, que nos proporciona la descripción estática del comportamiento de un programa, y el estado de la unidad de ejecución. El estado contiene por una parte, lo que se denomina espacio de datos, que son todos los recursos accesibles desde todas las rutinas activas y por otra, lo que se denomina estado de ejecución, que es información de control, como el valor del contador del programa y el estado de la pila, que nos permite retomar la ejecución después de la migración. Cada uno de estos componentes pueden moverse independientemente.

Los diferentes lenguajes han aportado soluciones diferentes para la movilidad de estos componentes, agrupándose en dos clases, por una parte los lenguajes que soportan *strong mobility* y los que soportan *weak mobility*:

- Movilidad fuerte (*strong mobility*): se migra el código y estado, abarcando tanto el espacio de datos como el estado de ejecución.
- Movilidad débil (*weak mobility*): se migra el código y el espacio de datos del estado del código, pero no el estado de ejecución.

Aunque existen lenguajes que proporcionn *strong mobility*, como son Telescript, TACOMA y Ara [Peine and Stolpmann, 1999]. La mayoría de los lenguajes soportan *weak mobility*, entre ellos Java, que a pesar de esta limitación ha sido el lenguaje en el que más plataformas de agentes se han desarrollado, debido a las siguientes ventajas aportadas por el lenguaje:

- **Independencia de la plataforma**: una de las principales ventajas que introdujo Java, fue poder desarrollar software independiente del procesador. La clave consistió en desarrollar un código neutro que pudiera ser ejecutado sobre una máquina virtual, denominada *Java Virtual Machine*, que es la que interpreta este código neutro convirtiéndolo a código particular de cada CPU o plataforma. Esto nos permite, que cualquier agente desarrollado en Java pueda ejecutarse en un host remoto que tenga una máquina virtual Java.

- **Ejecución segura**: Java fue concebido para su utilización en redes como Internet, por lo que la seguridad ha adquirido una importancia vital en su definición. La arquitectura de seguridad de Java hace que sea relativamente sencillo salvaguardar a un host de un agente malicioso, porque no se permite el acceso directo a los recursos del host, es un lenguaje fuertemente tipado y no existen punteros.

- **Carga dinámica de clases**: este mecanismo permite a la máquina virtual cargar clases en tiempo de ejecución, e incluso se puede habilitar su descarga a través de la red. Esto facilita la migración de agentes de un host a otro, ya que sólo es necesario que migre la clase *root* del agente, las demás clases que se precisen en ejecución, y que no existan en el host destino, serán descargadas dinámicamente, bien desde el host origen o bien desde un repositorio de clases habilitado en la plataforma de agentes para proporcionar este servicio.

- **Programación multithread**: los agentes por definición son autónomos, es decir, un agente se ejecuta independientemente de los otros agentes que residen en el mismo lugar. El comportamiento autónomo de los agentes se consigue permitiendo que cada agente se ejecute en su propio proceso ligero, también llamado hilo de ejecución. Java permite la programación multihilo y además proporciona un conjunto de primitivas de sincronización que nos permite la interacción entre los agentes.

- **Serialización de objetos**: el lenguaje Java proporciona mecanismos de serialización que permiten representar el estado de un objeto en forma serializada con el suficiente detalle para que este objeto se pueda reconstruir posteriormente.

- **Reflexión**: El código Java permite obtener información sobre los campos, métodos, y constructores de la clases cargadas. Esta información puede emplearse para crear objetos de una clase que se descubre en tiempo de ejecución. Esto permite dotar de cierta inteligencia a los agentes ya que pueden obtener información de si mismos y de otros agentes.

A pesar de todas estas ventajas, el desarrollo de plataformas de agentes con Java tiene ciertas limitaciones, una de las más importante es la que comentamos anteriormente, que sólo soporta *weak mobility*, pero además:

- No proporciona un soporte adecuado para el control de recursos, ya que en Java no se tiene control sobre los recursos consumidos, tiempo de procesador y memoria consumida, por un objeto.

- No proporciona control de referencias, todos los métodos públicos de un objeto Java son accesibles desde cualquier otro objeto que tenga una referencia a él. Esto presenta un problema ya que los agentes no tienen control sobre qué agentes acceden a sus métodos.

3.4.6. Plataformas de agentes y J2ME

Con la aparición de versiones de Java para dispositivos limitados, J2ME, y considerando que el lenguaje de programación en el que más plataformas de agentes móviles se han desarrollado es Java, las posibilidades de desarrollar plataformas de agentes en estos dispositivos se presenta alcanzable.

Para determinar las restricciones que nos impone J2ME a la hora de desarrollar una plataforma de agentes en dispositivos limitados, nos centramos en las limitaciones que posee la configuración CLDC por ser la más limitada y por ser un subconjunto de CDC, con lo que garantizamos que una solución en CLDC también funcionará en CDC. Por lo tanto, las principales limitaciones que nos impone CLDC son:

- Limitaciones del lenguaje Java:
  - No soporta tipos de datos de coma flotante (`float` o `double`).
  - No soporta la finalización de instancias de clases. El método `Object.finalize()` no existe.
  - Limitaciones en el manejo de errores. La mayoría de las subclases de `java.lang.Error` no están soportadas y en general, la gestión de errores es particular de la implementación realizada para el dispositivo concreto.
- Limitaciones de la máquina virtual:
  - No soporta Java Native Interface (JNI), debido a:
    - Modelo de seguridad limitado del CLDC.
    - Limitaciones de memoria necesarias para el soporte completo de JNI.
  - No soporta cargadores de clase definidos por el usuario, debido a restricciones de seguridad.
  - No soporta reflexión, y por lo tanto, ni serialización de objetos, ni soporte a RMI, ni otras características avanzadas de Java (JVM Debugging Interface, JVM Profile Interface).
  - No soporta grupos de `threads` ni `daemon threads`, las operaciones de arranque y parada de threads sólo se pueden aplicar individualmente.

La mayoría de estas limitaciones se deben a las propias limitaciones de procesamiento y memoria de los dispositivos y a razones de seguridad motivadas porque J2ME/CLDC no soporta el modelo completo de seguridad de J2SE. El modelo soportado por J2ME/CLDC es de tipo "sandbox", es decir las aplicaciones se ejecutan sobre un entorno limitado en el que la aplicación sólo puede acceder a las APIs definidas por el CLDC y por los perfiles proporcionados, o a clases específicas del dispositivo.

En el futuro se pretenden suplir algunas de estas limitaciones, entre ellas la sincronización de threads y el cargador dinámico de clases.

Si recordamos las características que convertían a Java en un buen lenguaje para desarrollar plataformas de agentes, sección 3.4.5, y las comparamos con las restricciones de J2ME, vemos que gran parte de éstas han desaparecido, o se han visto limitadas por motivos de seguridad, en concreto, aquellas que nos permitían implementar movilidad de objetos (carga dinámica de clases, serialización y reflexión).

3.4.7. Conclusiones

La investigación entorno al paradigma de agentes móviles se ha centrado fundamentalmente en el desarrollo de plataformas de agentes, y en los retos que plantea en sí la movilidad de código. Han sido pocas las investigaciones que se han centrado en demostrar la validez del desarrollo de servicios empleando agentes móviles y la mejora que supone su utilización respecto al paradigma clásico cliente/servidor. Además, su implantación en productos comerciales se ha visto frenada fundamentalmente debido a los problemas de seguridad que plantea la movilidad de código.

Los agentes se caracterizan por estar orientados a realizar tareas, por ser autónomos, por su capacidad de cooperar con otros sistemas y si poseen la característica de movilidad, por ser capaces de moverse a los sistemas remotos para realizar sus tareas y de esta forma minimizar el coste de las transmisiones. Estas características se adaptan a las limitaciones indicadas anteriormente y por las que se caracterizan los entornos ubicuos: entornos cambiantes, coste de las comunicaciones y limitaciones de los dispositivos.

Los desarrollos llevados a cabo a nivel agentes se realizaron para redes como Internet, las plataformas se ejecutaban en PCs sin limitaciones en cuanto a su capacidad de proceso y comunicación, y aunque en su definición se consideraba que se adaptaban a entornos cambiantes, con conectividad intermitente y calidad cambiante, la realidad es que estos estados eran tratados más como excepciones a las que el sistema sabía adaptarse, que como las características habituales del entorno en el que se ejecutaban. Por ello, no es viable migrar los desarrollos realizados hasta la actualidad a los sistemas limitados que operan en entornos ubicuos, sino que es necesario volver a diseñarlos.

En esta tesis doctoral se propone emplear el paradigma de agentes móviles como tecnología middleware para el desarrollo de servicios en entornos ubicuos, para ello es necesario realizar un diseño de plataforma de agentes que se adapte a las limitaciones que presentan estos entornos. En el diseño tomaremos como punto de partida el estándar FIPA e intentaremos con el mínimo impacto en la especificación adaptarlo a los nuevos requisitos. Como tecnología para apoyar la viabilidad de implantación de nuestro diseño, utilizaremos la versión de Java para dispositivos limitados, J2ME.

4. Trabajos relacionados

La investigación en computación móvil y ubicua a nivel de aplicación es bastante reciente, ya que los principales retos hasta el momento estaban fundamentalmente en el campo de la microlectrónica y de los nuevos protocolos de red, principalmente inalámbricos. Además de proyectos precursores como ParcTab de Xerox Park [Schilit et al., 1993], en estos últimos años han aparecido los primeros resultados de algunos otros. En este apartado describimos brevemente aquellos proyectos que están relacionados directamente con la tesis doctoral propuesta, y cuyo resultados han de ser tenidos en cuenta para contrastarlos con nuestras contribuciones. En el Cuadro 7 se presenta de forma resumida los objetivos y retos que se abordan en cada uno de los proyectos.

Recientemente, también han aparecido algunas plataformas de agentes que se ejecutan en dispositivos limitados basados en J2ME, aunque ninguna de ellas tiene como objetivo operar en entornos ubicuos, las describimos brevemente porque tienen un reto común con el que nosotros nos planteamos, que es la implementación de plataformas de agentes en la versión limitada de Java, J2ME.

Proyecto	**Visión/Objetivos**	**Retos direccionados**
Aura (CMU) www.cs.cmu.edu/ aura	Computación centrada en el usuario: la tecnología debe ser "transparente" y debe proporcionarle servicios personalizados y adaptados al contexto/localización en el que se encuentra.	Infraestructura software. Construcción y composición de servicios. Adaptación a la localización. Captura y gestión de las intenciones de los usuarios.
DEAPspace (IBM)	Computación distribuida peer-to-peer en redes inalámbricas ad-hoc de un solo salto, formadas por dispositivos limitados. Su objetivo es minimizar las transmisiones y la cantidad de información a transmitir para realizar un consumo mínimo de las baterías.	Infraestructura software. Formato de descripción de servicios y codificación eficiente. Protocolo peer-to-peer para el descubrimiento y anuncio de servicios.
PIMA (IBM) www.research.ibm.com/PIMA	Modelo en el que los dispositivos son portales, las aplicaciones realizan tareas para el usuario, y el entorno de computación añade información y nueva funcionalidad al espacio físico que rodea al usuario.	Herramientas para el desarrollo de aplicaciones según el modelo definido abarcando el diseño, descarga y ejecución. Adaptación de las aplicaciones al contexto del usuario.
Oxygen (MIT) oxygen.lcs.mit.edu/	Computación centrada en el usuario. Desarrollo de espacios inteligentes que interaccionan con el usuario de forma "invisible" mediante interfaces visuales y lenguaje natural. Adaptación de las aplicaciones al dinamismo y características de las nuevas redes.	Nuevos dispositivos embebidos para crear espacios inteligentes. Infraestructura software para el desarrollo y composición de servicios. Nuevos protocolos de routing en redes ad-hoc. Protocolos de descubrimiento y anuncio de servicios. Soporte a la movilidad.
Portolano (University Washington) portolano.cs.washington.edu	Computación centrada en el usuario. Tres líneas de investigación: interfaces de usuario, computación distribuida basada en código móvil y solventar las deficiencias a nivel infraestructura de red.	Infraestructura software. Descubrimiento de servicios. Construcción y composición de servicios. Arquitectura para el desarrollo de aplicaciones basadas en localización.

Cuadro 7: Trabajos relacionados

4.1. Aura of Carnegie Mellon University

El proyecto Aura comenzó en el año 2000 y tiene como objetivo crear una **arquitectura software de computación ubicua que se adapte al contexto y a las necesidades del usuario de forma invisible**, es decir, que su interacción con la tecnología embebida en el entorno sea mínima, de manera que no lo distraiga de sus tareas habituales. Aura se ha desarrollado para un entorno ubicuo en el que interaccionan dispositivos personales, dispositivos embebidos y dispositivos "vestibles" empleando protocolos de comunicación inalámbricos.

En Aura [Garlan et al., 2002] se trabaja con dos conceptos fundamentales: el primero se denomina *proactivity*, que es la capacidad que tienen las capas del sistema para anticiparse a las peticiones realizadas a alto nivel, el segundo se denomina *self-tuning*, que consiste en que las capas adaptan su consumo de recursos y su rendimiento a las peticiones y uso que se están haciendo de ellas. Teniendo como base estos conceptos han definido un arquitectura para dispositivos limitados con las siguientes capas:

- Kernel basado en Linux.
- Odyssey, que permite controlar recursos y da soporte a la adaptación de las aplicaciones al contexto.
- Coda, que proporciona acceso a ficheros adaptado a las diferentes condiciones de ancho de banda de la conexión inalámbrica, a la movilidad de los dispositivos y que permite operación desconectada.
- Spectra, que proporciona un mecanismo de ejecución remoto que emplea el contexto del usuario para decidir cuál es la mejor forma de ejecutar la petición remota realizada.
- Prism, que permite capturar y gestionar las intenciones de los usuarios y es una capa por encima de la de aplicación.

Tanto Odissey como Coda [Satyanarayanan, 1996] eran sistemas existentes desarrollados para entornos de computación móvil, que se han modificado para operar bajo las restricciones que impone los entornos de computación ubicua. La capa Prism es la capa en la que están centrando sus últimas investigaciones.

En Aura para aumentar las capacidades de los dispositivos limitados se emplea lo que se denomina *cyber foraign* que son típicamente PCs que se embeben en los entornos por los que se mueve el usuario y que los dispositivos limitados emplean para delegar ciertas tareas en ellos y a través de los que se conectan a Internet.

Dentro del proyecto Aura se han realizado también desarrollos orientados a lo que se denominan *advisor devices* para proporcionar información de las condiciones de la red, y a lo que se denomina *people locator* que permiten ofrecer servicios teniendo en cuenta la localización y el contexto del usuario. Ambos trabajos se han realizado para el protocolo inalámbrico IEEE 802.11.

4.2. IBM Research

IBM es una de las empresas que más está apoyando la investigación en computación ubicua, y así lo demuestran los diferentes proyectos que se están llevando a cabo en sus principales centros de investigación. A continuación describimos dos de los más destacados.

4.2.1. DEAPspace project of IBM Research Zurich

El proyecto DEAPspace se centra en dar soporte a la computación distribuida peer-to-peer entre dispositivos limitados cuando se aproximan unos a otros y forman una red ad-hoc.

Uno de los principales resultados del proyecto ha sido la definición de un **protocolo de descubrimiento y anuncio de servicios** denominado DEAPspace Algorithm [Nidd, 2001], a través de este algoritmo un dispositivo puede detectar la presencia de dispositivos próximos, compartir información de los servicios que están disponibles en la red y detectar cuando un dispositivo deja de estar disponible. El protocolo ha sido diseñado para operar de forma eficiente en redes inalámbricas ad-hoc de un solo salto, y su objetivo principal es dar respuesta a los cambios frecuentes que se producen en este tipo de entornos, teniendo en cuenta las restricciones de potencia y limitaciones de los dispositivos. El algoritmo se basa en un método "*push*" puro, en el que todos los dispositivos mantienen un "*world view*" que transmiten cada cierto periodo de tiempo a sus vecinos mediantes mensajes de broadcast y que actualizan escuchando el "*world view*" que tienen los demás.

En el proyecto se ha definido un **nuevo modelo de servicios** basado en roles [Hermann et al., 2001], el rol que realiza la entidad que proporciona el servicio se denomina *provider* y el que realiza la entidad que accede al servicio se denomina *requester*. Los mecanismos de comunicación entre estas entidades son independientes de la localización de las mismas, siguiendo las bases de los modelos tradicionales de computación distribuida. Independientemente del rol que realice una entidad, ésta proporciona al menos una de dos siguientes interfaces: *input interface* y *output interface*, estas interfaces aceptan datos en un formato determinado y para que un *requester* pueda acceder a un *provider*, sus *input interface* y *output interface* respectivamente, deben poseer algún formato de datos común.

Otra de las contribuciones ha sido la definición del **formato de descripción de los servicios** y de su **mecanismo de codificación** para minimizar la cantidad de datos a transmitir. En DEAPspace se realiza una distinción entre el protocolo de comunicación a través del que se accede a un servicio y el protocolo que se emplea para el descubrimiento, en general puede no ser el mismo y por ello se tiene en cuenta a la hora de definir el formato de descripción de un servicio.

En la actualidad este proyecto no está activo, aunque sus investigaciones se iban a centrar en solventar los problemas de seguridad, de configuración y gestión existentes en estos entornos, así como introducir aplicaciones sensibles a la localización y multidispositivo.

4.2.2. PIMA project of IBM T. J. Watson Research Center

En el proyecto PIMA [Banavar et al., 2000] se pretende dar soporte tecnológico a un **nuevo modelo de aplicaciones** que han definido para entornos de computación ubicua. Este modelo está basado en tres principios:

- Un dispositivo es un portal en el espacio de datos y aplicaciones, no es un repositorio de software instalado por el usuario.
- Una aplicación realiza una tarea que precisa el usuario, no es una pieza de software que está escrita para explotar las capacidades de un determinado dispositivo.
- Un entorno de computación proporciona información sobre el espacio físico en el que se encuentra el usuario, no es un espacio que nos permite almacenar y ejecutar software.

El modelo de aplicación se divide en tres partes: diseño, descarga y ejecución. En la parte de **diseño** se necesita tener un modelo de programación que permita al programador abstraerse del dispositivo concreto en el que se va a ejecutar la aplicación, y estructurar el programa en términos de tareas y subtareas.

En la parte de **descarga** es necesario abordar importantes retos: descubrimiento dinámico de las aplicaciones y servicios que se proporcionan a nuestro alrededor, negociación para adaptar los requisitos de las aplicaciones a las capacidades del dispositivo en el que se va a ejecutar, selección del interfaz de usuario más adecuado al dispositivo, e integración y composición de la aplicación descargada con las aplicaciones residentes en el dispositivo.

Por último, en tiempo de **ejecución** es necesario que las aplicaciones se adapten al entorno cambiante y limitado en el que se ejecutan, por lo tanto, deben implementar mecanismos de detección y recuperación de errores, mecanismos de monitorización y redistribución de recursos para adaptarse a las limitaciones de los dispositivos en los que se ejecutan y ser capaces de adaptarse a la conectividad discontinua, empleando para ello mecanismos como la migración de código.

4.3. Oxygen project of MIT

Este proyecto está siendo desarrollado en el *Artificial Intelligence Laboratory* del MIT y tiene como principal objetivo hacer de la computación ubicua una realidad, desde la prespectiva de que ésta debe proporcionarle al usuario lo que necesita. Oxygen aborda este reto considerando tres elementos: los dipositivos personales del usario, *Handy21s* (H21s), que serán el interfaz entre el usario y el entorno ubicuo; los dipositivos embebidos, *Enviro21s* (E21), que permitirán introducir capacidad de comunicación y computación en el entorno que rodea al usuario; y por último las nuevas redes, *Networks21s* (N21s), que permitirán la comunicación espontánea entre usuarios y dispositivos, y entre dispositivos.

En H21 y E21 la investigación se ha realizado fundamentalmente a nivel hardware. En cuanto a las redes, N21, el objetivo de Oxygen es proporcionar redes descentralizadas en las que los dispositivos entren y salgan de forma dinámica sin necesidad de configuraciones automáticas. En N21 se pretenden integrar todo tipo de redes tanto inalámbricas, como terrestres, como satélites y para ello se están definiendo una serie de algoritmos, protocolos y *middleware* para:

- Configuración automática de regiones formadas por un conjunto de dispositivos conectados entre ellos, entre los que existe una relación de confianza.
- Descubrimiento y localización de servicios de forma automática.
- Implementación de mecanismos de seguridad para el acceso a los servicios.
- Adaptación a los cambios constantes en la red, incluyendo congestión, control de errores, balance de carga, latencia, restricciones de potencia y requisitos de las aplicaciones.

En la actualidad existen ya algunas propuestas realizadas para cumplir estos objetivos, la más significativa desde el punto de esta propuesta de tesis son las relacionados con el **descubrimiento y localización de servicios** de forma automática en redes dinámicas: **Intentional Naming System (INS)** [Adjie-Winoto et al., 1999]. Las principales diferencias de INS, respecto a otros protocolos de descubrimiento de servicios, es que trata el descubrimiento de servicios de manera similar a la resolución de nombres en Internet (DNS). INS integra esta resolución de

nombres con el encaminamiento, además, los elementos que realizan la resolución incorporan balanceo de tráfico y se soporta replicación mantiendo la consistencia mediante el intercambio de mensajes.

4.4. Portolano project of University of Washington

La Universidad de Washigton es uno de los centros más activos en cuanto a la investigación en computación móvil y ubicua. Uno de los proyectos más importantes que se están desarrollando es el proyecto Portolano.

El proyecto Portolano [Esler et al., 1999] aborda los retos que plantea la computación móvil y ubicua centrándose fundamentalmente en tres elementos. El primero de ellos y más importante son las **interfaces de usuario**, los objetivos principales son: por una parte, permitir que múltiples interfaces diseñadas para distintos tipos de dispositivos interaccionen con la misma aplicación final, es decir, independizar la presentación de la semántica del interfaz; y por otra parte, conseguir interfaces "invisibles" desde el punto de vista del usuario.

El segundo de ellos se centra en los **servicios distribuidos**, en un entorno ubicuo no se deben proporcionar soluciones centralizadas dependientes de dispositivos concretos, y es necesario construir arquitecturas horizontales, de manera que las aplicaciones finales se apoyen en una serie de servicios comunes. Además, las aplicaciones deben estar orientadas a realizar las tareas que desea el usuario y en este sentido en Portolano se pretende aplicar el paradigma de código móvil. Otro reto que es necesario abordar a este nivel es la integración de servicios de manera fácil y flexible para que el usuario pueda acceder a nuevos servicios de forma dinámica y espontánea, sin necesidad de configuraciones o actualizaciones previas.

Por último, plantean la necesidad de realizar un estudio de los retos que aparecen a nivel **infraestructura de red** para alcanzar los objetivos anteriores, para ello van a analizar la viabilidad de utilizar soluciones ya existentes en el campo del descubrimiento dinámico de servicios de foma segura, de la compartición y acceso ubicuo a información almacenada en la red y de la tecnología para computación distribuida.

El proyecto Portolano a su vez se ha dividido en una serie de subproyectos que se han centrado en aportar soluciones en temas concretos de la visión global planteada:

- **Contact: Intrabody Signaling** [Partridge et al., 2000], este proyecto tiene como objetivo permitir la comunicación entre dispositivos ubicuos utilizando como medio de transmisión el cuerpo humano, de esta forma a través de un dispositivo personal del usuario se configurarán otros dispositivos con los que el usuario se ponga en contacto físico.
- **Hydra: Embedded Web Server**, en este proyecto, que ya no está activo, se desarrolló un dispositivo hardware que servía como gateway entre dispositivos con interfaz RS-232C e Internet. El software del dispositivo está basado en Linux y sobre él se ha desarrollado un servidor web embebido, un daemon telnet y un servidor de ficheros propietario denominado "my-FTP".
- **The Location Stack** [Hightower et al., 2002, Hightower et al., 2001], en este proyecto se aborda el desarrollo de aplicaciones sensibles con la localización, de tal forma que se tengan en cuenta los movimientos y posición del usuario a la hora de proporcionarle servicios. En el proyecto se ha definido un stack de siete capas (similar al modelo OSI) como abstracción para modelar los diferentes aspectos a tener en cuenta al desarrollar una aplicación sensible con la localización.

- **one.world** [Grimm et al., 2002], este proyecto se centra en proporcionar una arquitectura para computación ubicua. Esta arquitectura se basa en una serie de abstracciones para separar los datos de la funcionalidad, siguiendo la visión global del proyecto Portolano. Así una aplicación almacena y intercambia datos utilizando *tuples* y está compuestas por *components* que implementa la funcionalidad asociada. Las aplicaciones tienen al menos un *environment* asociado en el que almacenan *tuples* y en el que sus *components* se instancian. La arquitectura además de dar soporte para desarrollar aplicaciones según este modelo, incluye una serie de servicios básicos: *operations* que ayudan a la gestión de operaciones asíncronas, *migration* que permite mover o copiar un *environment* de un dispositivo a otro, *checkpointing* que permite almacenar el estado de ejecución de una tarea en una *tuple* para poder recuperarlo después de una migración, *remote event passing* (REP) que permite enviar eventos a servicios remotos, que es el proceso de comunicación básico que se emplea en one.world y por último, *discovery* que permite encaminar eventos a servicios de los que no se conoce su localización.

  El desarrollo llevado a cabo en este proyecto, es similar al diseño e implementación de una plataforma de agentes móviles, pero basándose en sus propias abstracciones.

4.5. Plataformas de agentes en dispositivos limitados

El hecho de que la mayoría de plataformas de agentes se hayan realizado en Java, ha llevado a que, cuando aparecen las primeras implementaciones de referencia de J2ME, algunos grupos de investigación intentaran migrar sus plataformas a esta versión y de esta forma se pudiesen ejecutar agentes en dispositivos limitados.

En este apartado realizamos una breve descripción de las dos plataformas existentes en la actualidad que se ejecutan sobre la versión CLDC de J2ME, la primera de ellas, LEAP, es el resultado de un proyecto europeo, y la segunda, MAE, de los trabajos realizados en Monash University en Australia.

También describimos el trabajo del grupo FIPA Ad Hoc, formado a principios de este año con el objetivo de adaptar las especificaciones FIPA, para que se puedan implementar en dispositivos limitados que se comunican mediante protocolos inalámbricos y forman de forma espontánea una red ad-hoc.

4.5.1. LEAP

El proyecto LEAP [LEAP Project, 2000] se engloba dentro de los proyectos europeos IST, y está siendo desarrollado por un consorcio de compañías entre las que se encuentran, entre otras, Motorola, British Telecom y Siemens. El objetivo del proyecto es el desarrollo de una plataforma de agentes móviles conforme al estándar FIPA que pueda operar tanto en dispositivos móviles (teléfonos móviles, PDA, pagers) como en PCs. El proyecto comenzó en Enero de 2000 y tiene dos fases, la primera de ellas, ya finalizada, consistió en la revisión de los estándares FIPA y WAP y el diseño e implementación de una plataforma de agentes para dispositivos móviles. La segunda fase consiste en evaluar la plataforma en sistemas reales y analizar las prestaciones obtenidas en dos aplicaciones: asistencia en carretera y tareas de gestión de red. Para que la plataforma de agentes desarrollada en este proyecto opere tanto en sistemas PCs como en dispositivos móviles con capacidades limitadas, han diseñado una arquitectura modular estructurada en dos partes:

- Parte obligatoria, compuesta por varios módulos, uno denominado *kernel* independiente del dispositivo y otros dedicados a las comunicaciones y dependientes del dispositivo en el que se ejecute la plataforma.
- Parte opcional, compuesta por varios módulos para interfaces gráficas, *parsers*, modelado de usuario y datos.

La plataforma de agentes se configurará así mediante un instalador que compondrá los módulos necesarios para el dispositivo concreto en el que se instale.

Para dispositivos móviles emplean las extensión correspondiente de Java, J2ME/CLDC, y para entornos PCs J2SE. El kernel de la plataforma solamente emplea las APIs comunes a ambas especificaciones de Java.

El proyecto LEAP fue pionero a la hora de demostrar la viabilidad de construir plataformas de agentes en dispositivos limitados, aunque su aplicabilidad está centrada en redes con infraestructura y por lo tanto, en los terminales móviles no existía una plataforma completa de agentes como tal y su operación dependía siempre de un sistema intermedio al que estuviese conectado.

Aunque era uno de los objetivos marcados en el proyecto, la movilidad a nivel agente no está soportada, esto se debe fundamentalmente a las propias limitaciones de J2ME, que no permite la definición de cargadores de clase definidos por el usuario.

El desarrollo de LEAP se basa en la plataforma sobre J2SE conforme con FIPA desarrollada en la Universidad de Parma, denominada JADE.

Posteriormente a esta iniciativa han aparecido algunas otras plataformas de agentes conformes con FIPA como son [Micro FIPA-OS, 2001] y Grasshopper MicroEdition [Grasshopper MicroEdition, 2001], para dispositivos limitados que soportan Personal Java. En la actualidad, Personal Java es una versión obsoleta cuyo equivalente en la arquitectura J2ME será el Personal Profile sobre la configuración CDC.

4.5.2. MAE of Monash University

En Monash University [Mihailescu and Kendall, 2002] se ha diseñado e implementado una plataforma de agentes para dispositivos personales móviles basados en PalmOS.

La plataforma desarrollada se denomina MAE y en la actualidad tienen dos implementaciones utilizando dos versiones reducidas de Java: la configuración CLDC de J2ME, y SuperWaba, que es una versión limitada de Java con algunas funcionalidades no soportadas por la versión oficial de Sun, como JNI.

MAE está compuesta por un *device agent execution environment* (DAEE) que proporciona un conjunto de servicios a los agentes que se ejecutan en la plataforma. Estos servicios se agrupan en tres tipos: *presentation*, que proporciona los servicios que permiten a los agentes interaccionar con el usuario; *agent*, que proporciona los servicios que permiten a los agentes interactuar con los servicios de red, entre ellos se encuentra el que permite descubrir y anunciar servicios y el que permite conocer el estado de la red; *network*, que permite a los agentes comunicarse con otros dispositivos.

Esta plataforma soporta movilidad a nivel agente, para solventar el problema de la carga dinámica de clases se han empleando mecanismos específicos de los sistemas PalmOS, por lo que esta característica no es migrable a otro tipo de dispositivos J2ME.

4.5.3. Working Group FIPA Ad Hoc

El *Working Group FIPA Ad Hoc*[17] surge a principios de 2002 con el objetivo de definir una plataforma de agentes conforme con FIPA, que pueda operar en redes ad-hoc. El principal problema al que se enfrentan en este tipo de redes, es que son redes sin infraestructura y entonces, la plataforma existente en cada dispositivo debe ser autónoma, y por lo tanto poseer todos los componentes definidos en el modelo de referencia de FIPA.

En la actualidad el tema más activo del grupo se centra en determinar cómo debe realizarse el descubrimiento de servicios/agentes y por lo tanto, analizan si la definición de *Directory Facilitator* y los mecanismos de federación para búsquedas remotas en FIPA es válido y posible en redes ad-hoc. El grupo fundamenta sus trabajos en tres requisitos:

- Añadir los mínimos cambios a las especificaciones existentes de FIPA.
- Utilizar los mecanismos de descubrimiento dinámico de servicios existentes en la actualidad (Jini, JXTA, SLP, SSDP, etc), de manera que la solución propuesta pueda interoperar con cualquiera de ellos.
- Adaptarse a entornos ad-hoc, pero que la solución propuesta también sea válida para redes con infraestructura.

El grupo está escribiendo un White Paper [WG-AdHoc, 2002] en el que se analizan detalladamente los requisitos anteriores y los retos que éstos plantean, realizando además un estudio de las diversas tecnologías de descubrimiento dinámico de servicios existentes en la actualidad. La autora de esta propuesta de tesis doctoral ha aportado contribuciones al grupo de trabajo, que han sido incluidas recientemente en el White Paper.

4.6. Conclusiones

Los proyectos que hemos descrito a lo largo de este apartado tienen una visión común, que coincide con la que nosotros nos hemos planteado: la tecnología desarrollada en entornos de computación ubicua debe adaptarse al usuario, de manera que para él, ésta sea "invisible". Lo que diferencia unos proyectos de otros es la temática concreta que abordan para contribuir a que la tecnología haga que ésto sea una realidad. Algunos de ellos como Oxygen centran gran parte de su investigación a nivel hardware, construyendo dispositivos que se pueden embeber en el entorno físico del usuario, otros como PIMA se centran en proporcionar entornos de desarrollo que faciliten el diseño e implementación de aplicaciones según el nuevo modelo que impone la computación ubicua.

La característica común de todos ellos es que abordan la necesidad de proporcionar una serie de tecnologías middleware para facilitar el desarrollo de servicios y aplicaciones en entornos de computación ubicua, en concreto:

- Soporte para la ejecución de aplicaciones en dispositivos limitados: arquitectura Aura y one.world de Portolano.
- Soporte para el descubrimiento y anuncio dinámico de servicios: DEAPspace algorithm, INS de Oxygen y el protocolo de descubrimiento dentro de one.world de Portolano.

[17] En Octubre de 2002 pasó de *Technical Commitee* a *Working Group*

- Soporte para desarrollar servicios teniendo en cuenta el contexto y localización del usuario: Aura, Oxygen y Portolano.

En esta tesis doctoral planteamos realizar contribuciones a las dos primeras. En concreto, en el caso del soporte para la ejecución de aplicaciones, las propuestas realizadas difieren de la que aquí se plantea en que, en la propuesta de Aura los dispositivos limitados precisan un dispositivo tipo PC para integrarse en el sistema, y en la propuesta de one.world, aunque se basa como nosotros proponemos en conceptos de agentes móviles, define de nuevo una plataforma basada en sus propias abstracciones sin tener en cuenta las investigaciones y estándares que se han realizado hasta la actualidad.

En el caso del descubrimiento dinámico, las contribuciones difieren respecto a la que nosotros nos proponemos en cuanto que el protocolo de one.world utiliza en su algoritmo elementos centrales; el protocolo INS va más allá del descubrimiento de servicios y en él se introducen mecanismos de routing, que desde nuestro punto de vista deben abordarse de manera independiente; por último, el protocolo DEAPspace se aproxima más a los requisitos que nosotros nos planteamos, aunque realiza ciertas consideraciones sobre la propagación de anuncios en los que no se tienen en cuenta las distancias.

En cuanto a los trabajos relacionados para llevar la tecnología de agentes, aunque las propuestas analizadas han realizado importantes contribuciones en cuanto a adaptarlas a las limitaciones de los dispositivos, ninguna de ellas se ha realizado teniendo en mente las restricciones que impone el entorno. En el caso de LEAP, la plataforma se realizó para operar en redes inalámbricas con infraestructura, por lo que la plataforma que se ejecuta en el dispositivo limitado no es autónoma y depende de un sistema no limitado situado en la red. En cuanto a MAE, su implementación se ha realizado para utilizarla en comercio móvil, y aunque es bastante completa ya que incluye movilidad, se ha realizado utilizando funcionalidades concretas de dispositivos PalmOS y no se ha tenido en cuenta el estándar FIPA para realizar su diseño.

5. Objetivos

En esta tesis nos hemos planteado realizar contribuciones en el campo de la definición de tecnologías middleware, en las que se apoyarán los servicios y aplicaciones que se desarrollen en entornos de computación ubicua. Como hemos ido describiendo y justificando a lo largo de los apartados anteriores, los objetivos concretos que nos hemos planteado en la realización de esta tesis doctoral son los que enumeramos a continuación:

- Estudio y análisis de la tecnologías relacionadas con los entornos de computación ubicua para determinar:
  - las necesidades a nivel tecnologías middleware que es necesario cubrir,
  - los requisitos que se deben imponer a estas tecnologías teniendo en cuenta las características de la computación ubicua:
    - entornos cambiantes,
    - heterogeneidad y limitaciones hardware/software de los nuevos dispositivos, y
    - heterogeneidad y diferentes prestaciones en cuanto ancho de banda, calidad de la transmisión y conectividad de las nuevas redes.

- Contribución en la definición de una tecnología middleware para proporcionar descubrimiento y anuncio dinámico de servicios.

  El alcance de este objetivo lo hemos dividido en una serie de subobjetivos:

  - Estudio y análisis de los protocolos de descubrimiento y anuncio dinámico de servicios existentes.
  - Enumeración detallada y justificada de las necesidades y requisitos en los que se debe fundamentar la definición de nuestro nuevo protocolo.
  - Definición de un nuevo protocolo de descubrimiento y anuncio dinámico de servicios.
  - Análisis de las prestaciones del nuevo protocolo y comparativa con otros protocolos existentes.

- Contribución a la implantación del paradigma de agentes móviles como tecnología middleware para el desarrollo de servicios y aplicaciones en entornos ubicuos.

  El alcance de este objetivo lo hemos definido en una serie de subojetivos:

  - Estudio y análisis del paradigma de agentes móviles, desarrollos y estándares relacionados.
  - Enumeración detallada y justificada de las necesidades y requisitos concretos que debe cubrir una plataforma de agentes móviles en entornos de computación ubicua.
  - Definición de los servicios básicos que debe proporcionar una plataforma de agentes en estos entornos.
  - Adaptación y compatibilidad de nuestra propuesta con el modelo de referencia FIPA, realizando contribuciones en los foros de discusión del Working Group FIPA AdHoc.
  - Análisis de la viabilidad de su implantación actual basándose en la tecnología J2ME.

A lo largo de este documento hemos ido reflejando la necesidad de abordar otras tecnologías middleware básicas para el desarrollo de servicios en entornos de computación ubicua, como la seguridad, y la composición y personalización de servicios según las preferencias del usuario, su localización y su contexto. Aunque en esta tesis doctoral no se van a abordar estos temas, las conclusiones obtenidas del estudio de las necesidades y requisitos en los entornos de computación ubicua pueden servir como base para realizar propuestas en esta temática.

Además, en las aportaciones concretas que realicemos tendremos en cuenta estos aspectos para que estos trabajos puedan apoyarse y extender al que nosotros realicemos. Así, en el protocolo de descubrimiento y anuncio de servicios somos conscientes de que se deberán introducir mecanismos de seguridad y además, deberá ser lo suficientemente flexible, para que pueda ser utilizado por cualquier descripción y definición de servicios que se realice a más alto nivel. En cuanto a la implantación del paradigma de agentes en entornos ubicuos, esta propuesta está motivada en gran parte, por la flexibilidad y facilidad que aporta este paradigma para desarrollar servicios personalizados, aunque es cierto que en este campo todavía existen muchos retos por alcanzar y a los que comienza a dar respuesta la inteligencia artificial.

6. Plan de desarrollo

El plan de desarrollo que vamos a llevar a cabo para la realización de esta tesis doctoral se detalla a continuación:

- Estudiar las características de los entornos de computación ubicua.
- Analizar las necesidades y restricciones que imponen estos entornos para el desarrollo de aplicaciones.
- Contribuir con la definición de un protocolo de descubrimiento y anuncio dinámico de servicios. Para ello se han planificado las siguientes subtareas:
  - Estudiar y analizar los protocolos de descubrimiento y anuncio dinámico de servicios propuestos en la literatura.
  - Definir un nuevo protocolo de descubrimiento y anuncio dinámico de servicios adaptado a las restricciones de entornos ubicuos.
  - Analizar las prestaciones del protocolo definido, comparándolo con otros protocolos existentes.
  - Conclusiones.
- Contribuir en la adaptación del paradigma de agentes móviles para ofrecer y componer servicios en entornos de computación ubicua. Para ello se han planificado las siguientes subtareas:
  - Estudiar y analizar el paradigma de agentes móviles, así como de las plataformas y estándares existentes.
  - Definir los elementos básicos que debe proporcionar una plataforma de agentes móviles para operar en estos entornos.
  - Estudiar la compatibilidad de la propuesta realizada con la especificación actual de FIPA.
  - Integrar la propuesta realizada, respecto al descubrimiento y anuncio de servicios, en la plataforma de agentes móviles propuesta.
  - Conclusiones.
- Redacción de la tesis.

Referencias

[3G TS 21.101, 2000] 3G TS 21.101 (2000). Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); 3rd Generation mobile system Release 1999 Specifications.

[3G TS 21.102, 2001] 3G TS 21.102 (2001). Universal Mobile Telecommunications System (UMTS); 3rd Generation mobile system Release 4 Specifications.

[3G TS 21.103, 2002] 3G TS 21.103 (2002). Universal Mobile Telecommunications System (UMTS); 3rd Generation mobile system Release 5 Specifications.

[3G TS 43.051, 2002] 3G TS 43.051 (2002). Digital cellular telecommunications system (Phase 2+); GSM/EDGE Radio Access Network (GERAN) overall description; Stage 2.

[Adjie-Winoto et al., 1999] Adjie-Winoto, W., Schwartz, E., Balakrishnan, and Lilley, J. (1999). The design and implementation of an intentional naming system. In *17th ACM Symposium on Operating Systems Principles (SOSP'99)*, pages 186–201.

[Andersson, 2001] Andersson, C. (2001). *GPRS and 3G wireless applications*. John Wiley & Sons.

[Baldi and Picco, 1998] Baldi, M. and Picco, G. (1998). Evaluating the tradeoffs of mobile code design paradigms in network management applications. In Kemmerer, R., editor, *Proceedings of the 20th International Conference on Software Engineering*, IEEE CS Press, pages 146–155.

[Banavar et al., 2000] Banavar, G., Beck, J., Gluzberg, E., Munson, J., Sussman, J., and Zukowshi, D. (2000). Challenges: An Application Model for Pervasive Computing. In *Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 2000)*.

[Bellifemine et al., 1999] Bellifemine, F., Poggi, A., and Rimassa, G. a. (1999). JADE: A FIPA-compliant agent framework. In *Proceedings of PAAM'99*, pages 97–108, London.

[Bettstetter et al., 1999] Bettstetter, C., Vogel, H.-J., and Eberpacher, J. (1999). GSM Phase 2+ General Packet Radio Service GPRS: Architecture, Protocols, and Air Interface. *IEEE Communications Surveys*, 2(3).

[Bisdikian, 2001] Bisdikian, C. (2001). An Overview of the Bluetooth Wireless Technology. *IEEE Communications Magazine*, pages 86–94.

[Bluetooth v1.1, 2001] Bluetooth v1.1 (2001). Specification of the Bluetooth System v1.1. http://www.bluetooth.com/dev/specifications.asp.

[Borenstein, 1994] Borenstein, N. S. (1994). EMail With A Mind of Its Own: The Safe-Tcl Language for Enabled Mail. In *IFIP International Conference*.

[Bray and Sturman, 2001] Bray, J. and Sturman, C. F. (2001). *Bluetooth : connect without cables*. Prentice Hall Professional Technical Reference.

[Cable Home 1.0, 2002] Cable Home 1.0 (2002). CableHome 1.0 Specification.

[Callaway et al., 2002] Callaway, E., Gorday, P., Hester, L., Gutierrez, J. A., Naeve, M., Heile, B., and Bahl, V. (2002). Home Networking with IEEE 802.15.4: A Developing Standard for Low-Rate Wireless Personal Area Networks. *IEEE Communications Magazine*, pages 70–77.

[Campo, 2002a] Campo, C. (2002a). Agentes móviles en computación ubicua. In Aedo Cuevas, I., Díaz Pérez, P., and Fernández Llamas, C., editors, *Actas del Tercer Congreso Interacción Persona-Ordenador Interacción 2002*, pages 215–219, Leganés, Spain.

[Campo, 2002b] Campo, C. (2002b). Directory Facilitator and Service Discovery Agent. Technical report, FIPA Ad-hoc Technical Committee.

[Campo, 2002c] Campo, C. (2002c). Service Discovery in Pervasive Multi-Agent Systems. In Finin, T. and Maamar, Z., editors, *AAMAS Workshop on Ubiquitous Agents on embedded, wearable, and mobile agents*, Bologna, Italy.

[Campo et al., 2002a] Campo, C., Carcía, C., Almenares, F., Marín, A., and Delgado, C. (2002a). TAgentsP y PDP: propuestas para una plataforma de agentes en computación ubicua. In *Segundo Congreso Iberoamericano de Telemática CITA 2001*, Mérida, Venezuela.

[Campo et al., 2001a] Campo, C., García, C., Marín, A., and Delgado, C. (2001a). Plataformas de Agentes en Terminales de Telefonía Móvil. In *XI Jornadas de I+D en Telecomunicaciones*, Madrid, Spain.

[Campo et al., 2001b] Campo, C., García, C., Marín, A., and Delgado, C. (2001b). Tecnología de agentes en los sistemas de telefonía móvil. In *III Jornadas de Ingeniería Telmática. JITEL 2001*, Barcelona, Spain. ISBN 84-7653-783-2.

[Campo et al., 2001c] Campo, C., Marín, A., García, A., Díaz, I., Breuer, P., Delgado, C., and García, C. (2001c). JCCM: Flexible Certificates for Smartcards with Java Card. In Attali, I. and Jensen, T., editors, *Proceedings of the International Conference on Research in Smart Cards*, volume 2140 of *Lecture Notes in Computer Science*, pages 34–42, Cannes, France. ISBN 3-540-42610-8.

[Campo et al., 2002b] Campo, C., Marín, A., García, C., and Breuer, P. (2002b). Distributed Directory Facilitator: A proposal for the FIPA Ad-hoc First CFT. Technical report, FIPA Ad-hoc Technical Committee.

[CDC, 2002] CDC (2002). *Connected Device Configuration (JSR-36)*. SUN Microsystems.

[Chen, 2000] Chen, Z. (2000). *Java Card technolgy for smart cards : architecture and programmer's guide*. Addison-Wesley.

[CLDC, 2000] CLDC (2000). *Connected, Limited Device Configuration (JSR-30)*. SUN Microsystems.

[Cugola et al., 1997] Cugola, G., Ghezzi, C., Pietro Picco, G., and Vigna, G. (1997). *Analyzing Mobile Code Languages. In Mobile Object Systems: Towards the Programmable Internet*, volume 1222 of *Lecture Notes in Computer Science*. Springer-Verlag.

[Dayem, 1997] Dayem, R. A. (1997). *Mobile data and wireless LAN technologies*. Prentice Hall.

[Diego Bartolomé et al., 2002] Diego Bartolomé, M. V. d., Gallego Pérez, D., López Mora, J. A., and Gómez Vicente, A. (2002). UMTS: hacia una red todo IP. *Comunicaciones de Telefónica I+D*, (4):85–106.

[Dobrev et al., 2002] Dobrev, P., Famolari, D., Kurzke, C., and Miller, B. A. (2002). Device and Service Discovery in Home Networks with OSGi. *IEEE Communications Magazine*, pages 86–92.

[Doufexi et al., 2002] Doufexi, A., Armour, S., Butler, M., Nix, A., Bull, D., and McGeehan, J. (2002). A Comparison of the HIPERLAN/2 and IEEE 802.11a wireless LAN Standards. *IEEE Communications Magazine*, pages 172–179.

[Dutta-Roy, 1999] Dutta-Roy, A. (1999). Networks for Homes. *IEEE Spectrum*, pages 26–33.

[Echelon, 1999] Echelon (1999). Introduction to the LonWorks System. Technical report, Echelon Corporation.

[Edens, 2001] Edens, G. T. (2001). Home Networking and the CableHome Project at CableLabs. *IEEE Communications Magazine*, pages 112–121.

[Esler et al., 1999] Esler, M., Hightower, J., Anderson, T., and Borriello, G. (1999). Next Century Challenges: Data-Centric Networking for Invisible Computing The Portolano Project at the University of Washington. In *Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 1999)*.

[Frank and Hollaway, 2000] Frank, E. H. and Hollaway, J. (2000). Connecting the Home with a Phone Line Network Chip Set. *IEEE MICRO*, pages 1–12.

[Frengle, 2001] Frengle, N. (2001). *I-Mode: A Primer*. John Wiley & Sons.

[Frodigh et al., 2000] Frodigh, M., Johansson, P., and Larsson, P. (2000). Wireless ad hoc networking-The art of networking without a network. *Ericsson Review*, (4):248–293.

[Gardner et al., 2000] Gardner, S., Markwalter, B., and Yonge, L. (2000). HomePlug Standard Brings Networking to the Home. Technical report, CommsDesign.com.

[Garlan et al., 2002] Garlan, D., Siewiorek, D. P., Smailagic, A., and Steenkiste, P. (2002). Project Aura: Toward Distraction-Free Pervasive Computing. *IEEE Personal Communications*, pages 22–31.

[Glass, 1999] Glass, G. (1999). *Mobility: Processes, Computers, and Agents*, chapter ObjectSpace Voyager Core Package Technical Overview, pages 612–627. Addison-Wesley.

[Glitho and Magedanz, 2002] Glitho, R. H. and Magedanz, T. (2002). Applicability of Mobile Agents to Telecommunications. *IEEE Network*, page 6. Es una referencia a la editorial, realmente la revista entera está dedicada a la aplicación de agentes móviles para tareas de gestión de red, fundamentalmente.

[Glitho and Pierre, 2002] Glitho, R. H. and Pierre, S. (2002). Mobile Agents and Their Use for Information Retrieval: A Brief Overview and an Elaborate Case Study. *IEEE Network*.

[Goland et al., 1999] Goland, Y. Y., Cai, T., Leach, P., and Gu, Y. (1999). Simple service discovery protocol/1.0. Technical report.

[Gong, 2001] Gong, L. (2001). JXTA: A Network Programming Environment. *IEEE Internet Computing*, pages 88–95.

[Grasshopper, 1998] Grasshopper (1998). Grasshopper Technical Overview.

[Grasshopper MicroEdition, 2001] Grasshopper MicroEdition (2001). http://www.grasshopper.de/.

[Grimm et al., 2002] Grimm, R., Davis, J., Lemar, E., Macbeth, A., Swanson, S., Anderson, T., Bershad, B., Borriello, G., Gribble, S., and Wetherall, D. (2002). Programming for Pervasive Computing Enviroments.

[GSM 02.60, 1997] GSM 02.60 (1997). Digital cellular telecommunications system (Phase 2+); General Packet Radio Service (GPRS); Service description; Stage 1.

[Guttman and Kempf, 1999] Guttman, E. and Kempf, J. (1999). Automatic Discovery of Thin Servers: SLP, Jini and the SLP-Jini Bridge. In Press, I., editor, *25th Ann. Conf. IEEE Industrial Electronics Soc. (IECON 99)*, Piscataway, N. J.

[Hansmann et al., 2001] Hansmann, U., Merk, L., Nicklous, M. S., and Stober, T. (2001). *Pervasive Computing Handbook*. Springer-Verlag.

[Hansmann et al., 2002] Hansmann, U., Mettala, R., Purakayastha, A., and Thompson, P. (2002). *SyncML: Synchronizing and Managing Your Mobile Data*. Prentice Hall.

[Helal, 2002] Helal, S. (2002). Standards for Service Discovery and Delivery. *IEEE Pervasive Computing*, pages 95–100.

[Hermann et al., 2001] Hermann, R., Husemann, D., Moser, M., Nidd, M., Rohner, C., and Schade, A. (2001). DEAPspace – Transient ad hoc networking of pervasive devices. *Computer Networks*, pages 411–428.

[Hightower et al., 2002] Hightower, J., Brumitt, B., and Borriello, G. (2002). The Location Stack: A Layered Model for Location in Ubiquitous Computing. In *4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2002)*, Callicoon, NY. IEEE Computer Society Press.

[Hightower et al., 2001] Hightower, J., Vakili, C., Borriello, G., and Want, R. (2001). Design and Calibration of the SpotON Ad-Hoc Location Sensing System.

[HIPERLAN/2, 1999] HIPERLAN/2 (1999). ETSI Broadband radio access networks (BRAN); HIPERLAN type 2 technical specification; Physical layer.

[IBM Research Zurich, 1999] IBM Research Zurich (1999). Advanced Infrared. http://www.zurich.ibm.com/cs/wireless/ircommunication.html.

[IEEE 802.11, 1999] IEEE 802.11 (1999). IEEE 802.11 Local and Metropolitan Area Networks: Wireless LAN Medium Acess Control (MAC) and Physical (PHY) Specifications.

[IEEE 802.15.1, 2002] IEEE 802.15.1 (2002). IEEE Standard for information technology - Telecommunication and information exchange between systems - LAN/MAN - Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPANs).

[IEEE Std 1394-1995, 1996] IEEE Std 1394-1995 (1996). IEEE Standard for a High Performance Serial Bus.

[Java 2 Micro Edition, 1999] Java 2 Micro Edition (1999). Java 2 Platform Micro Edition. http://java.sun.com/j2me.

[Jini, 1999] Jini (1999). Jini Architectural Overview. White Paper. Technical report.

[Johansen et al., 1999] Johansen, D., Renesse, R. v., and Schneider, F. B. (1999). *Mobility: Processes, Computers, and Agents*, chapter Operating System Support for Mobile Agents, pages 557–566. Addison-Wesley.

[Kaaranen et al., 2001] Kaaranen, H., Ahtiainen, A., Laitinen, L., Naghian, S., and Niemi, V. (2001). *UMTS networks : architecture, mobility, and services*. John Wiley & Sons.

[Kapp, 2002] Kapp, S. (2002). 802.11: Leaving the Wire Behind. *IEEE Internet Computing*, pages 82–85.

[Karaoğz, 2001] Karaoğz, J. (2001). High-Rate Wireless Personal Area Networks. *IEEE Communications Magazine*, pages 96–102.

[Khun-Jush et al., 2000] Khun-Jush, J., Malmgren, G., Schramm, P., and Torsner, J. (2000). Hiperlan type 2 for broadband wireless communication. *Ericsson Review*, (2):108–119.

[Khun-Jush et al., 2002] Khun-Jush, J., Schramm, P., Malmgren, G., and Torsner, J. (2002). HiperLAN2: Broadband Wireless Communications at 5 GHz. *IEEE Communications Magazine*, pages 130–136.

[Kotz et al., 1999] Kotz, D., Gray, R., Nog, S., Rus, D., Chawla, S., and Cybenko, G. (1999). *Mobility: Processes, Computers, and Agents*, chapter AGENT TCL: Targeting the Needs of Mobile Computers, pages 513–523. Addison-Wesley.

[KVM, 2000] KVM (2000). *Java 2 Platform Micro Edition (J2ME) Technology for Creating Mobile Devices*. SUN Microsystems.

[Lange and Oshima, 1998] Lange, D. B. and Oshima, M. (1998). *Programming and Developing Java Mobile Agents with Aglets*. Addison-Wesley.

[Lea et al., 2000] Lea, R., Gibbs, S., Dara-Abrams, A., and Eytchison, E. (2000). Networking home entertainment devices with HAVi. *IEEE Computer*, pages 35–43.

[LEAP Project, 2000] LEAP Project (2000). Deliverable D31: Specifications of the LEAP Architecture.

[Margherita2000.com, 2000] Margherita2000.com (2000).

[Marples and Kriens, 2001] Marples, D. and Kriens, P. (2001). The Open Services Gateway Initiative: An Introduction Overview. *IEEE Communications Magazine*, pages 110–114.

[Matthes et al., 1994] Matthes, F., Müssig, S., and Schmidt, J. (1994). Persistent Polymorphic Programming in Tycoon: An Introduction. Technical report, FIDE Project Coordinator, Dept. of Computing Sciences, University of Glasgow, Glasgow.

[Micro FIPA-OS, 2001] Micro FIPA-OS (2001). http://fipa-os.sourceforge.net.

[MIDP, 2000] MIDP (2000). *Mobile Information Device Profile (JSR-37)*. SUN Microsystems.

[Mihailescu and Kendall, 2002] Mihailescu, P. and Kendall, E. A. (2002). Development of an agent platform for mobile devices using J2ME. In *Proceedings of the Evolve 2001 conference*.

[Miller and Pascoe, 1999] Miller, B. and Pascoe, R. (1999). Mapping Salutation Architecture APIs to Bluetooth Service Discovery Layer. http://www.salutation.org/whitepaper/BtoothMapping.PDF.

[Miller and Pascoe, 2000] Miller, B. A. and Pascoe, R. A. (2000). Salutation service discovery in pervasive computing environments. Technical report, IBM White Paper.

[Muller et al., 2001] Muller, F., Sorelius, J., and Turina, D. (2001). Further evolution of the gsm/edge radio access network. *Ericsson Review*, (3):116–123.

[Negus et al., 2000] Negus, K. J., Stephens, A. P., and Lansforb, J. (2000). HomeRF: Wireless Networking for the Connected Home. *IEEE Personal Communications*, pages 20–27.

[Nidd, 2001] Nidd, M. (2001). Service Discovery in DEAPspace. *IEEE Personal Communications*.

[Nwana, 1996] Nwana, H. S. (1996). Software Agents: an overview. *Knowledge Engineering Review*, 1(3):205–244.

[Ortiz, 2002] Ortiz, E. C. (2002). A Survey of J2ME Today. Technical report, Wireless Java.

[Papastravrou et al., 1999] Papastravrou, S., Samaras, G., and Pitoura, E. (1999). Mobile agents for WWW distributed database access. In *Proceedings of the International Conference on Data Engineering (ICDE99)*.

[Partridge et al., 2000] Partridge, K., Arnstein, L., Borriello, G., and Whitted, T. (2000). Fast Intrabody Signaling (Demostration). In *3rd Workshop on Mobile Computer Systems and Applications (WMCSA 2000)*, Monterey, California.

[Peine and Stolpmann, 1999] Peine, H. and Stolpmann, T. (1999). *Mobility: Processes, Computers, and Agents*, chapter The Architecture of the Ara Platform for Mobile Agents, pages 583–595. Addison-Wesley.

[RFC 2608, 1999] RFC 2608 (1999). Service location protocol, version 2.

[Richard III, 2000] Richard III, G. G. (2000). Service Advertisement and Discovery: Enabling Universal Device Cooperation. *IEEE Internet Computing*, pages 18–27.

[Rose, 2001] Rose, B. (2001). Home Networks: A Standards Perpective. *IEEE Communications Magazine*, pages 78–85.

[Satyanarayanan, 1996] Satyanarayanan, M. (1996). Mobile Information Access. *IEEE Personal Communications*, pages 26–33.

[Satyanarayanan, 2001] Satyanarayanan, M. (2001). Pervasive computing: Vision and challenges. *IEEE Personal Communications*, 8(4):10–17.

[SC00023J, 2002] SC00023J (2002). *FIPA Agent Management Specification*. FIPA.

[Schilit et al., 1993] Schilit, B., Adams, N., Gold, R., Tso, M., and Want, R. (1993). The PARCTAB Mobile Computing System. In *Fourth Workshop on Workstation Operating Systems*, pages 34–39.

[SDP, 2001] SDP (2001). *Bluetooth Specification v1.1, Part E: Service Discovery Protocol (SDP)*. http://www.bluetooth.com/dev/specifications.asp.

[Singhal, 2001] Singhal, S. (2001). *WAP The Wireless Application Protocol : Writing Applications for the Mobile Internet*. Addison-Wesley.

[Suvak, 2000] Suvak, D. (2000). IrDA and Bluetooth: A Complementary Comparison. Technical report, Extended Systems, Inc.

[Tschudin, 1994] Tschudin, C. F. (1994). An Introduction to the MO Messenger Language. Technical report, University of Geneva, Switzerland.

[Wacks, 2002] Wacks, K. (2002). Home Systems Standards: Achievements and Challenges. *IEEE Communications Magazine*, pages 152–159.

[Weiser, 1991] Weiser, M. (1991). The Computer for the 21st Century. *Scientific American*.

[WG-AdHoc, 2002] WG-AdHoc (2002). Agents in Ad Hoc Environments. A Whitepaper.

[White, 1995] White, J. E. (1995). Telescript Technology: Mobile Agents. Technical report, General Magic.

[Williams, 2000] Williams, S. (2000). IrDA: Past, Present and Future. *IEEE Personal Communications*, pages 11–19.

[X10, 1997] X10 (1997). http://www.x10.com.