"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

GUÍA DE DISEÑO DE AGENTES MÓVILES EN COMPUTACIÓN UBICUA[1]

Mobile Agent Design Guide for Ubiquitous Computing

Guide de dessin d'agents mobiles dans informatique ubiquitaire

Adriana Xiomara Reyes Gamboa
Esp. en Teleinformática Universidad Distrital
MSc Ciencias Computacionales
Universidad de Los Andes ULA- Mérida Venezuela
Docente Investigador
Miembro Grupo Investigación GRINSOFT
Politécnico Colombiano Jaime Isaza Cadavid
axreyes@elpoli.edu.co

Darío Enrique Soto Durán
MSc Ciencias Computacionales
Universidad de Los Andes ULA- Mérida Venezuela
Docente Investigador
Miembro Grupo Investigación GIISTA
Tecnológico de Antioquia
dsoto@tdea.edu.co

Fanny Mojica Sepúlveda
MSc (C) Física
Universidad de Pamplona
Docente
Universidad de Pamplona
fannsepulveda@unipamplona.edu.co

[1] Artículo resultado del proyecto de investigación "Agentes Móviles en la Computación Ubicua", de la Facultad de Ingenierías de la Universidad de Pamplona, Politécnico CJIC, Tecnológico de Antioquia. Los participantes son los tres autores del presente artículo. El proyecto inició en julio de 2007 y terminó en marzo de 2009.

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

Tipo de artículo: Artículo de revisión (resultado final de investigación)
Recepción: 2009-11-28
Revisión: 2010-04-30
Aprobación: 2009-05-12

Contenido

1. Introducción
2. Metodología
3. Conceptos generales
4. Resultados
5. Conclusiones
6. Lista de referencias

Resumen. Este artículo presenta el papel relevante que los agentes, específicamente los Agentes Móviles (AM), han tenido en los sistemas computacionales, en su rápida evolución durante los últimos años. Se presta especial interés al conjunto de Agentes Móviles (móviles con los usuarios y con los dispositivos) y su aplicación en la Computación Ubicua. Como resultado de la investigación se suministra un procedimiento y una guía que abarca aspectos a tener en cuenta al momento de diseñar e implementar un agente en un ambiente ubicuo.

Palabras clave: Agente móvil, Computación ubicua, Ciclo de vida, Ingeniería del *Software*.

Abstract. This article presents the relevant roles that have had the agents, specifically the Mobile Agents (MA), in computer systems related to its rapid development during the last years. Particular emphasis is made on both mobile operator set (mobile users and devices) and its application in Ubiquitous Computing. As a result of the research procedure, we provide a guide covering aspects to consider when designing and implementing an agent in a ubiquitous environment.

Keywords: Mobile Agent, Ubiquitous Computing, Life Cycle, Software Engineering.

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

Résumé. Dans cet article nous présentons le rôle d'importance des agents, en particulier des agents mobiles sur les systèmes informatiques, dans son rapide évolution pendant les dernières années. Nous centrons spécialement notre attention sur l'ensemble de agentes mobiles (mobiles avec les utilisateurs et les dispositifs) et son application en l'informatique ubiquitaire. Comme résultat de cette recherche nous avons développé un procédé et une guide que comprendre des aspects qui doivent être considérés au moment du dessin et d'implémentation d'un agent dans un environnement ubiquitaire.

Mots-clés : Agent mobile, Informatique ubiquitaire, Cycle de vie, Génie logiciel

1. Introducción

La computación ha causado grandes impactos tecnológicos, sociales y organizacionales. Desde sus inicios, en los años 80, los grandes sistemas informáticos se crearon para dar servicio a muchos usuarios, con una sola máquina, pero de complejo funcionamiento y elevado costo. En la década siguiente ya eran computadores personales, de manejo más sencillo y menor costo. En la actualidad, la computación ubicua ***(CU)*** es penetrante u omnipresente (*Pervasive Computing*) y ofrece posibilidades de interacción[2] más simples y cercanas al usuario, mediante pequeños dispositivos dispersos en el entorno que los rodea. Aunque de sencillo manejo, no está plenamente implementada.

Gracias a la evolución tecnológica en la miniaturización de los microprocesadores, se han abierto nuevas posibilidades de servicio al usuario a través de la manipulación de la información presente en su entorno. Por ello ya no sólo se puede imaginar una industria compuesta por máquinas inteligentes, sino a muchos usuarios utilizando pequeños dispositivos inteligentes, con limitado poder de procesamiento y comunicación, que permiten la realización de muchos procesos.

Por tanto, la tecnología de agentes puede ser de gran ayuda en el desarrollo de sistemas generalizados, ya que estos agentes de *software* son entidades autónomas que pueden interactuar con su medio ambiente y, por consiguiente, se adaptan bien a esos cambios frecuentes. Sin embargo, su

[2] Todos los intercambios que suceden entre persona – computador (Baecker y Buxton, 1987)

uso plantea importantes retos que exigen conocer sus ventajas para satisfacer tanto los desafíos como las necesidades que éstos demandan y, de este modo, poder tener la información en cualquier momento y desde cualquier lugar (Weiser, 2002).

2. Metodología

La primera fase de esta investigación consistió en una búsqueda documental encaminada a recolectar, comparar y analizar información sobre la computación ubicua y los agentes móviles, con el fin de establecer un estado de arte claro y bien documentado.

Para el análisis de los documentos relacionados con el objeto de estudio del proyecto, se establecieron categorías de gran utilidad para los demás métodos de recolección de información que se utilizaron en el desarrollo de la investigación.

Así mismo, se determinaron las fuentes a consultar:
1. Bases de datos científicas.
2. Publicaciones posteriores al 2000.
3. Publicaciones científicas o académicas.

La segunda fase del proyecto planteó la integración de la computación ubicua y los agentes móviles, desde el diseño de estos últimos, con lo cual se decidió impactar el ciclo de vida del desarrollo de *software* convencional, con las premisas que se deben tener en cuenta a la hora de dar origen a un agente móvil, dando paso, con ello, a una tercera fase que consistió en generar la guía de diseño de agentes móviles en el computo ubicuo.

Este documento se construye a partir de los resultados obtenidos en la búsqueda documental.

3. Conceptos generales

En los últimos años, la información se ha constituido en un recurso indispensable para la mayoría de las actividades humanas y de vital importancia, en algunos casos. En consecuencia, es necesario preguntar:

¿Cómo el computador comienza a estar encajado en los dispositivos electrónicos de uso diario de las personas y qué efectos tiene esto en sus

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

quehaceres? ¿De qué forma los computadores empiezan a estar en todas partes, sin que las personas se den cuenta? La respuesta está en la computación ubicua ***(CU),*** la cual hace uso de dispositivos pequeños y económicos presentes en el entorno de usuario, tanto en el hogar como la oficina.

Se debe diferenciar la computación móvil o nómada de la ubicua. La primera dio paso a la segunda*,* que es la que permite la realización de tareas de cómputo mientras el usuario se traslada o está en lugares diferentes a su entorno habitual. Por tanto, se requiere de la innovación tecnológica para cubrir estas necesidades de la forma más transparente posible.

No obstante, ha de tenerse en cuenta que, si bien la computación ubicua ***(CU)*** proporciona recursos, también impone restricciones en cuanto a movilidad, autonomía, sociabilidad, seguridad, entre otras; pero éstas se pueden eliminar con las propiedades de movilidad, autonomía y cooperación que poseen los agentes móviles ***(AM)*** *y,* de esta manera, compartir no sólo recursos "*hardware* y *software*", sino también información.

Se entiende por agente móvil *(AM)* aquél que no está limitado al sistema donde se ejecutó; él mismo será capaz de moverse de una máquina a otra a través de la red y "vuelve a casa" luego de completar su tarea (Nwana, Ndumu, 2004).

Una definición más técnica consiste en que "Los *Agentes Móviles, s*on libres de migrar a través de la red, creando un ambiente de ejecución donde migran su ***código*** (el programa que define su comportamiento o descripción estática), ***estado*** (el que le permite continuar con su actividad al moverse o la descripción en un momento determinado de ejecución) y los ***datos*** (que va recopilando en su recorrido por la red a otro ambiente de ejecución en la red)" (Garcia, Solarte, 2004); a su vez son autónomos con la función de ejecutar tareas, con el fin de cumplir sus metas u objetivos, retornando al nodo inicial con sólo los datos de resultados y, a su vez, pueden rechazar peticiones de otros agentes.

4. Resultados

4.1 ¿Por qué el uso de los agentes móviles *(AM)?*

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

Los agentes móviles son procesos de *software* capaces de transitar por una red, interactuar con *host* alejados, reunir información para el usuario y volver a su origen cuando las tareas establecidas por el usuario se hayan cumplido.

Las siguientes son algunas de las ventajas más notables que se pueden obtener al usar agentes móviles (Chess, Harrison 2000):

▪ Reducen el costo de comunicación (eficiencia). Por ejemplo, si en una ubicación hay un gran volumen de información que necesita ser examinada y transmitida, ello requeriría de una gran cantidad de recursos en la red y consumiría mucho tiempo. En este caso, el agente móvil sólo encapsula la información necesaria, gracias a que el agente va a una dirección, hace la búsqueda/selección local y transfiere solamente la imagen elegida comprimida a través de la red.

▪ Computación asíncrona. Mientras que un agente móvil realiza su tarea el usuario puede ir realizando otra, de tal manera que después de un tiempo el resultado del agente móvil será enviado al usuario.

▪ Ejecución o cálculo asíncrono. Los agentes pueden operar cuando el usuario no está conectado a la red y enviarán los resultados en algún tiempo posterior.

En cuanto a sus beneficios tenemos (Lange, Oshima, 2005):

Tabla 1. Beneficios de los Agentes Móviles

Beneficios	
Reducir el tráfico en la red	- Ya que se ejecutan asíncrona y autónomamente, las tareas pueden ser embebidas en los agentes móviles, los cuales pueden ser enviados por la red; por otro lado, la independencia de los agentes hace que la comunicación con el nodo de inicio sea mínima o nula. - El código se mueve a los datos
	- En sistemas de tiempo real críticos es

Ocultar latencia de red en aplicaciones de tiempo real	importante que las instrucciones de control se ejecuten sin retraso. -Los agentes pueden resolver el problema al ser despachados desde un controlador central e instalarse en los componentes controlados.
Encapsular los protocolos de comunicación	- Se utilizan para manejar nuevos requerimientos como la autenticación, el cifrado, el control de acceso e instalación y actualización de nuevos protocolos en todos los nodos, como en el caso de las *redes activas, Active Network*[3], (donde una red activa es una red programable que permite que el código sea cargado dinámicamente en los nodos de la red en tiempo de ejecución).

4.2 Procedimiento para la integración de un *(am)* en un ambiente ubicuo

▪ El procedimiento debe permitir especificar unos sistemas *pervasivos* utilizando primitivas conceptuales y protocolos de coordinación y comunicación propios de estos entornos.
▪ El procedimiento debe generar sistemas funcionalmente operativos a partir la especificación del mismo.
▪ Los sistemas generados deberán soportar la integración de los (AM) utilizados en los sistemas *pervasivos*; para ello es útil tener presente la especificaciones ACL, *Agents Communication Languages*, definida por FIPA, *Foundation for Intelligent Physical Agents* (Bellifemine, Poggi 1999).

Como se muestra en la Figura 1, el procedimiento se puede ver desde el punto de vista del ciclo de vida del *software*, pero dándole las características de un *software* móvil. Teniendo en cuenta el modelo tradicional de ingeniería del *software*, el cual comienza por: La especificación de Requisitos -

[3] La noción *Active Network* representa la idea que los nodos de las redes de comunicación pasan a ser activos por tomar parte en el procesamiento de las aplicaciones y en el soporte de servicios personalizados.

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

>Análisis de Requerimientos->Diseño->Implementación->Pruebas y verificación->Operación y Mantenimiento.

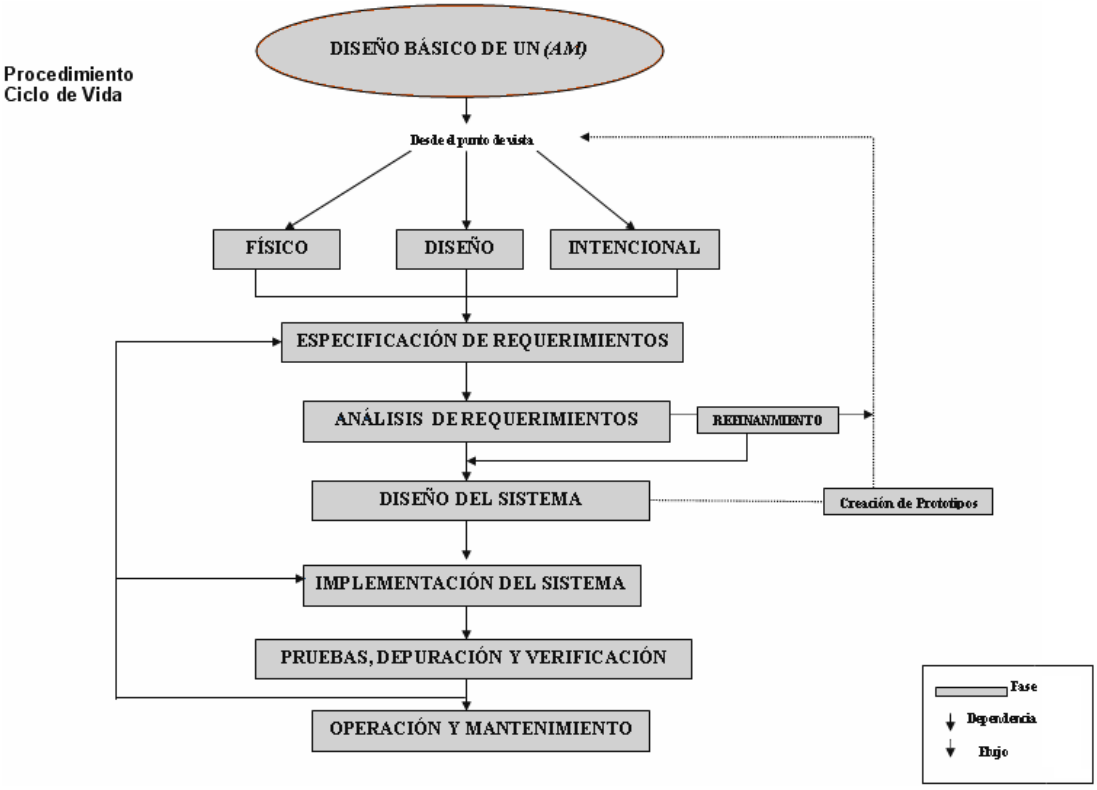


Figura 1. Ciclo de vida del SW

Descripción de cada fase:

4.2.1 Diseño básico de un agente móvil

Desde el punto de vista FÍSICO
Pensar en los constituyentes físicos de los objetos.
Desde el punto de vista del DISEÑO
Pensar en la función de los objetos (diseñarlos para una función).
Desde el punto de vista INTENCIONAL
Pensar en los objetos como agentes racionales (creencias, intenciones, deseos...).

Entorno: ambiente ubicuo (Brooks, 2003).

El agente toma la información del contexto, por tanto éste:
1. Adquiere información del contexto.
2. Procesa la información del contexto.
3. Razona sobre la información del contexto.
4. Obtiene nueva información del contexto (y vuelve al paso al paso 1).

Especificación de requisitos. ¿Qué?

En esta fase se construye un documento de manera formal, teniendo en cuenta las necesidades de un cliente con respecto al sistema nuevo para, de esta manera, establecer la forma más precisa en que los desarrolladores intentarán construir el sistema.

A partir de este documento se pueden identificar:
- Las funcionalidades del sistema.
- Los agentes, los casos de uso.
- Los diagramas de flujo de datos.
- Los diagramas de despliegue, para describir lógicamente el sistema.

Se define una arquitectura común para todos los sistemas *pervasivos* que se van a construir, ya que ésta permite interconectar componentes, tanto *software* como *hardware*, los cuales hacen que un agente se comporte como tal.

Dicha arquitectura determina cómo descomponer los agentes en módulos (o Modelos) y cómo interactúan entre sí para lograr la funcionalidad requerida; también se debe especificar el método de descomposición de las tareas o, mejor dicho, los servicios de cada agente en el ambiente.

Arquitectura interna. Existen muchos modelos para la representación del conocimiento, entre los cuales tenemos el Lógico, el Reactivo y el Híbrido.

Se han desarrollado varias alternativas para establecer la *especificación de requisitos* de los sistemas de agentes. Una de esas alternativas establece la funcionalidad a través de construcciones "mentales" como creencias, deseos e intenciones (Arquitectura BDI: *Belief, Desire, Intention*). Aunque no es fácil la construcción de especificaciones, éstas se pueden construir como una serie de "reglas" (fórmulas lógicas) como por ejemplo:

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

Si *el agente1 cree que el agente2 cree que la opción 1 está vacía*
entonces *el agente1 debería creer que el agente2 tiene información errónea y el agente1 debe intentar informar al agente2 de su error*

Las *creencias* se aplican habitualmente para referirse a la información que el agente tiene sobre su entorno. Esta información puede ser incorrecta, de la misma manera que nuestra información sobre el entorno (nuestra creencia) puede ser incorrecta.

La *intención* se utiliza para referirse a una meta que el agente seguirá hasta que tenga éxito o falle totalmente. Se admite la posibilidad de fallo ya que, en muchos sistemas complejos, se tienden a ignorar los formalismos tradicionales. Se puede considerar que el comportamiento proactivo de los agentes está representado en términos de una librería de planes, el agente selecciona un plan de la librería sobre la base de los objetivos que desea satisfacer. Un plan se instancia cuando ocurre el disparo de un evento que satisface su invocación y condiciones de contexto. Un plan instanciado es una intención. El cuerpo de un plan es un conjunto de tareas que pueden ser subobjetivos, acciones, aserciones de la base de creencias, y consultas y mensajes a otros agentes. Cuando se forma una intención, estas tareas son ejecutadas por el agente en un esfuerzo por alcanzar un objetivo dado (deseos).

Algunos requisitos a tener en cuenta para la Arquitectura Móvil son:

▪ *Extensibilidad:* se refiere a la capacidad que tiene el sistema para adoptar nuevas mejoras o cambios, por ejemplo, adaptándose a nuevas tecnologías.
▪ *Portabilidad:* las aplicaciones desarrolladas sobre la arquitectura deben estar aisladas de las peculiaridades de los diferentes servidores de aplicaciones, bases de datos y sistemas de comunicaciones a través de una serie de interfaces genéricos (capas de abstracción *software*) que ocultarán los detalles de implementación a la aplicación, de forma que el cambio de uno de estos recursos no implique un cambio en el código fuente de las aplicaciones, facilitando en consecuencia la portabilidad de la misma.

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

Análisis de requerimientos. Obtener una descripción de cuáles son los requisitos del sistema que se desea desarrollar, permitirá darle forma y orden, a la vez que posibilitará modelar los procesos para un mejor entendimiento.

Se considera que ésta es una de las fases más importantes ya que, disponer de un documento de requisitos completo y preciso, evitará propagar errores a las siguientes fases del ciclo.

Refinamiento. Una vez que se tiene una especificación de alto nivel, se debe transformar esta especificación en agentes y relacionarlos con los demás en la estructura general del sistema. Este proceso se conoce como *refinamiento* (En esta fase ya se sabe qué hace el sistema).

Diseño del sistema. ¿Cómo?

Hasta la fecha se han realizado muchos esfuerzos para que el diseño de sistemas eficientes y robustos sea un proceso metódico y riguroso; sin embargo, muy pocos de esos esfuerzos se derivan hacia la aplicación de técnicas a los sistemas de agentes. Luego del *refinamiento* deben existir o crearse:

Guías para ver cómo descomponer un problema de agentes y cómo afectará un diseño en el funcionamiento general del sistema. Para ello, ese diseño debe realizarse desde una perspectiva:
- *Top-down* –descendente- (cómo se debería descomponer el sistema idealmente), y
- *Bottom-up* –ascendente- (qué componentes *software* existen ya y no pueden ser cambiados, o *software* ya existente).

Para obtener un buen diseño será necesario:
- Descomponer el sistema o subsistemas o módulos en términos de los objetivos que ellos buscan.
- Diseñar o importar la ontología del dominio.
- Basados en la ontología, diseñar la base de conocimiento de cada agente y
- Adicionar habilidades cognitivas a cada agente (mecanismos de cooperación, coordinación y competencia).

Ontología. Para conseguir el principio de procesamiento independiente del contexto, es preciso un modelo genérico del conocimiento que independice

los servicios, los usuarios y los dispositivos. Este modelo se consigue mediante ontologías.

▪ Una ontología es una descripción (parecido a una descripción formal de un programa) de los conceptos y relaciones que pueden existir para un agente o una comunidad de agentes.
▪ Una ontología define el vocabulario de representación del conocimiento sobre un dominio y sus situaciones específicas.
▪ Cada ontología se modela mediante lenguajes de definición de ontologías (web semántica o www) como OWL[4] y RDF[5]. Estos lenguajes permiten la descripción de cada uno de los modelos de datos de cada aplicación o servicio en términos de clases, relaciones entre ellas y sus propiedades y atributos; es decir, la ontología de movilidad se proporciona por la plataforma a utilizar y el estándar FIPA (Bellifemine, Poggi, 1999).

Implementación del sistema. Para esta fase existen varias arquitecturas, entre ellas la más utilizadas son: la inteligencia artificial, la cual construye agentes basados en el conocimiento, codificando el comportamiento del agente en términos de reglas, marcos de trabajo o redes semánticas; y la otra es el lenguaje JAVA (Lange, Oshima, 2005), pero no se puede decir cuál es mejor o peor ya que ambas tienen ventajas y desventajas.

Pruebas, validación y verificación. Razonar sobre estas fases es el aspecto más importante; por lo tanto se necesitará:

▪ Una guía para ver qué está pasando. Se necesitan utilidades de validación y verificación para avanzar en la ejecución y comprobar que el comportamiento (interno y externo) del sistema cumple con los requisitos del cliente.

La visualización se puede hacer con imágenes ya que determinar qué está ocurriendo en un sistema es una tarea especialmente difícil. Esto es cierto en un agente, pero más entre agentes móviles, debido a que de esta fase depende si hay que volver a revisar la fase anterior o, incluso, la modificación de la fase de análisis donde se encuentran los requisitos del sistema.

[4] Lenguaje de ontologías w*eb (Web Ontology Language).*
[5] Marco de descripción de recurso (*Resource Description Framework).*

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

Operación y mantenimiento. En esta fase el sistema ya está en funcionamiento y, por ende, hace referencia al cambio asociado a los *errores* ya sean: detectados, fallas, mejoras solicitadas o cambios. En esta fase se puede considerar el retiro del *software*.

4.2.2 Guía de diseño de un *(AM)* en un ambiente ubicuo

Para el desarrollo de un sistema de agentes móviles, teniendo en cuenta sus características principales (movilidad, autonomía, sociabilidad, proactividad, y reactividad), éste debe contener:

1. ***Modelo de entorno***: dirigido a las entidades participantes en el entorno ubicuo.

2. ***Modelo de agente:*** describe la estructura interna del agente como parte del agente móvil; en esencia define las características de autonomía, aprendizaje y cooperatividad. Al igual que un modelo de consecución de objetivos que tratará de que el agente alcance aquellos que le hayan sido fijados.

3. ***Modelo de ciclo de vida:*** define los diferentes estados de ejecución de un agente móvil.

4. ***Modelo computacional:*** define las habilidades computacionales de un agente, como la manipulación de datos y primitivas, y control de instrucciones (hilos de ejecución).

5. ***Modelo de comunicación:*** es la implementación de un protocolo que se utilizará, tanto para solicitar a otros agentes o a humanos datos o tareas externas, como para recibir solicitudes de otros agentes y, probablemente, un modelo de control que priorice sus propias tareas y las que debe hacer para otros agentes.

6. ***Modelo de datos:*** se refiere a los datos en el que se registren, tanto las percepciones de su entorno, como su estado.

7. ***Modelo de seguridad:*** se encarga de dos protecciones: la de los agentes sobre un nodo y la de los nodos sobre los agentes, es decir, describe cómo y qué agentes pueden acceder a los recursos de la red o a los propios agentes.

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

8. ***Modelo de navegación:*** se refiere a todos los aspectos de movilidad; es el que gestiona todas las características referidas al transporte de un agente (con o sin su estado), entre dos entidades computacionales que residen en ubicaciones diferentes.

Posteriormente, se construyen los agentes que son los encargados de realizar las funciones de los modelos anteriores; éstos podrían ser:

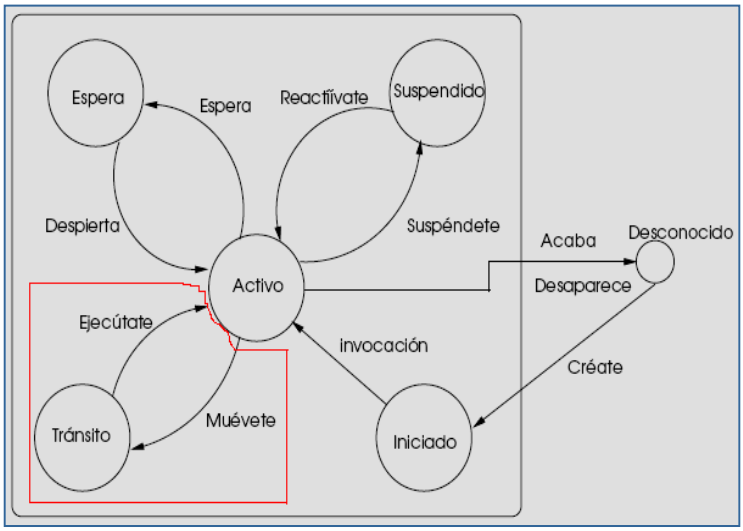
Agentes de interfaz: permiten que el usuario interactúe con el resto de los sistemas; supervisan eventos y llevan a cabo tareas en el dominio de la aplicación.

Agentes del sistema: relacionan el modelo computacional con el modelo de comunicación, estableciendo una capa de inteligencia, así como mecanismos de coordinación entre ellos.

Agentes de gestión: permiten que el proveedor de los servicios interactúe con los sistemas. De acuerdo con lo anterior, como se muestra en la *Figura 2,* para el diseño de agentes se tiene en cuenta el ciclo de vida del agente mediante estados y transiciones propuesto por la *FIPA* (Bellifemine, Poggi, 1999).

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

Figura 2. Estados y transiciones de un agente móvil



Las transiciones o acciones para pasar los agentes de un estado a otro, son:

Créate (*Create*): creación de un nuevo agente.
Invocación (*Invoke*): invocación o activación de un nuevo agente una vez inicializado.
Desaparece (*Destroy*): se fuerza la terminación de un agente. Esta acción sólo puede ser iniciada por el Sistema de Agentes Móviles (AMS) y no puede ser ignorada.
Acaba (*Quit*): se solicita la terminación de un agente. El agente puede ignorar la acción.
Suspéndete (*Suspend*): pone a un agente en el estado de suspendido. Esta acción puede ser iniciada por el propio agente o por el AMS.
Reactívate (*Resume*): saca a un agente del estado suspendido. Esta acción sólo puede ser iniciada por el AMS.
Espera (*Wait*): pone a un agente en el estado de espera. Esta acción sólo puede ser iniciada por el propio agente.
Despierta (*Wake* up): saca a un agente del estado de espera. Esta acción sólo puede ser iniciada por el AMS.

Únicamente los agentes móviles pueden entrar en el estado Tránsito "*transit*". Esto asegura que los agentes estacionarios ejecuten todas sus

instrucciones en el nodo donde fueron invocados; al igual que las acciones encerradas en color rojo, sólo son usadas por los agentes móviles:
▪ **Muévete (*Move*):** pone a un agente en el estado de transición. Esta acción sólo puede ser iniciada por el propio agente.
▪ **Ejecútate (*Execute*):** saca al agente del estado.

Plataforma

▪ La arquitectura de agentes debe estar basada en una plataforma de gestión de agentes que incorpore los recursos básicos para soportar: su ciclo de vida, el modelo de despliegue, su configuración, su supervisión, y la inclusión de nuevas funcionalidades. Entre ellas se encuentra el soporte a los diferentes protocolos de comunicación que componen el entorno ubicuo.
▪ *El framework* o plataforma, que implemente la arquitectura definida, deberá facilitar la integración de las distintas tecnologías utilizadas en el desarrollo de este tipo de sistemas.

Lenguaje de comunicación

La comunicación es el elemento representativo en este tipo de sistemas, tanto entre agentes como en el medio en el que habitan. Además, es fundamental que exista un mecanismo adecuado de comunicación entre agentes.

Para que esta interacción e interoperación entre agentes sea significativa, constructiva e inteligente, se requiere entonces de:

Protocolo de redes *Ad-Hoc*[6]

Los protocolos habilitan que dos máquinas se comuniquen. Esta comunicación permite transportar agentes. Los protocolos deben operar sobre una gran cantidad de redes de transporte, incluyendo aquellas basadas en el protocolo de internet (TCP/IP). Los protocolos operan en dos niveles. El nivel más inferior gestiona el transporte de los agentes y el más alto su codificación y decodificación.

[6] Redes cuya topología cambia constantemente debido a la ubicación de los dispositivos que a ella se conectan.

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

La computación ubicua requiere un conjunto de protocolos, disponibles y listos para ser usados, que implemente:

- Un mecanismo de serialización que transforme la llamada al método en la forma adecuada para su transmisión sobre la red (SOAP[7]).
- Una capa de transporte que lleve esos datos inherentes al método entre los sistemas remotos.
- Un mecanismo para encontrar, activar y desactivar objetos distribuidos.
- Un modelo de seguridad para proteger el sistema local y remoto.

Los lenguajes facilitan la interoperabilidad, de manera que los agentes pueden comunicarse independientemente del lenguaje en el que han sido implementados.

De acuerdo con lo anterior, existen dos tipos de lenguajes: los *declarativos* y los *procedurales.* Los segundos son los que se utilizan en el diseño de (AM), ya que son sencillos y su ejecución es eficiente y directa; por tanto, para facilitar el desarrollo de las aplicaciones de comunicación y la interacción entre lugares y agentes móviles, el lenguaje de programación debe ser:

- **Completo***:* para que cualquier algoritmo pueda ser expresado en el lenguaje.
- **Orientado a objetos***:* para obtener los beneficios de esta tecnología.
- **Dinámico***:* para que pueda transportar la clase que se requiera para crear instancias de agentes en máquinas remotas.
- **Persistente***:* para que el agente y su información sean respaldados en un medio no volátil.
- **Portable y seguro**: para que se pueda ejecutar sobre cualquier plataforma de una forma segura.
- **Versátil***:* que permita tareas complejas de transportación, autenticación y control de acceso.
-

Pero, a su vez, trae la desventaja de que requieren información de los agentes que recibirán los mensajes.

Un mensaje está estructurado como se muestra en la Tabla 2, o generalmente consta de:

Tabla 2. Estructura de un mensaje

[7] Aquitectura orientada a servicios (*Simple Object Access Protocol).*

"Revista Virtual Universidad Católica del Norte". No. 30, (mayo - septiembre de 2010, Colombia), acceso: [http://revistavirtual.ucn.edu.co/], ISSN 0124-5821 - Indexada Publindex-Colciencias, Latindex, EBSCO Information Services y Actualidad Iberoamericana. **[pp. 251-269]**

Emisor (*Sender*)	Envía el mensaje
Receptor (*Receivesr*)	Lista de receptores del mensaje
Preformativa (*Performative*)	Contenido del mensaje *REQUEST*: si se desea que el receptor realice una acción *INFORM*: si se desea que el receptor conozca una información *QUERY IF*: si el que lo envía desea conocer una información, entre otras.
Contenido (*Content*)	Contenido del mensaje
Lenguaje (*Language*)	Sintaxis del mensaje
Ontología (*Ontology*)	Vocabulario y significado utilizados en el mensaje

5. Conclusiones

Con el desarrollo de esta investigación se plantea la integración de los AM en la CU, a partir del ciclo de vida del *software*, pero con la asignación de características que se deben tener en cuenta para un *software* móvil, lo cual genera como resultado una "guía de diseño" para la definición de los agentes, en el que se muestran aspectos generales a tener en cuenta al momento de diseñar e implementar un agente móvil en un ambiente ubicuo, con el fin de poderlos entender y especificar de una manera clara y entendible.

La utilización de la tecnología de agentes permite realizar operaciones asíncronas para la ejecución de tareas, ya que el agente cuando migra, además de llevar los datos, conserva la información de estado del proceso. Gracias a esto dicha tecnología cada día tiene más estudio.

Es necesario revisar temas en cuanto a seguridad del contenido de los agentes, para que éstos no sean interceptados en el camino o suplantados por agentes con fines distintos a los originales.

Son muchos los esfuerzos que han tenido los desarrolladores para poseer técnicas que permitan reconocer actividades y adaptarlas a las necesidades de los usuarios. Sin embargo, hay un largo camino que recorrer, ya que se debe trabajar más en sistemas conscientes del contexto.

6. Lista de referencias

Abowd, G.D. & Mynatt, E.D. (2000*).* Charting Past, Present and Future Research in Ubiquitous Computing. *ACM Transaction on Computer Human Interactions, Vol. 7, 1.* P. 29-58.

Baecker, R. M., & Buxton, W. A. S. (1987). *Human computer interaction*. P. 45 -55.

Bellifemine, F., A. Poggi y G. Rimasa (1999). JADE-A FIPA-compliant agent framework. *Proceeding of PAAM'99*. London.

Brooks, K*.* (2003). *The Context Quintet: Narrative Elements Applied to Context Awareness,* presented at Human Computer Interaction International Proceedings, Crete, Greece.

Chess, C. Harrison (2000). *Mobile Agents: Are They A Good Idea*. I.B.M.

Foundation for Intelligent Physical Agents – FIPA. Recuperado: 30 de enero de 2009, desde: http://www.fipa.org

García Dávalos, A. y Solarte Astaiza, Z. M. (2004). 4-TMA Software Architecture for Developing Mobile Applications. First International Workshop MATA 2004. *Addendum to the Proceedings. P. 7–11*. Florianópolis, Brasil.

Lange Danny, Oshima Mitsuru (2005). *Programming and Deploying Java Mobile Agents with Aglets*. Addisson-Wesley Longman. Recuperado: 30 de enero de 2009, desde: http://gidis.ing.unlpam.edu.ar/downloads/pdfs/AgentesMoviles_Arturo.pdf

Nwana H., Ndumu D. (2004). *An Introduction to Agent Technology*, ISR Agent Research. Recuperado: septiembre de 2008, desde. http://more.btexact.com/projects/agents/publish/papers/intro_agents.htm

Weiser, Mark (2002). The Computer for the 21st Century. *Scientific American*, Reimpreso en *IEEE Pervasive Computing, vol1, no 1*, p. 19-25. Recuperado: octubre de 2008, desde: http://griho.udl.es/ipo/doc/03Metafo.doc