UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES

ARAGON

ESTUDIO DE METODOLOGIAS Y DESARROLLO DE PROTOTIPO DE UN SISTEMA MULTIAGENTE PARA CONFIGURACION DE HARDWARE

TESIS PROFESIONAL

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A :

JORGE MARTINEZ RENDON

DIRECTOR DE TESIS: M. EN C. JESUS DIAZ BARRIGA ARCEO

SAN JUAN DE ARAGON 2004

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ARAGÓN

JEFATURA DE CARRERA DE INGENIERÍA EN COMPUTACIÓN

OFICIO:ENAR/JACO/0357/04.

ASUNTO: Asignación de Jurado

LIC. ALBERTO IBARRA ROSAS
SECRETARIO ACADÉMICO
P r e s e n t e .

Por este conducto me permito presentar a usted el nombre de los profesores que sugiero integren el Sínodo del Examen Profesional del alumno **JORGE MARTÍNEZ RENDÓN** con el trabajo de tesis "ESTUDIO DE METODOLOGÍAS Y DESARROLLO DE PROTOTIPO DE UN SISTEMA MULTIGENTE PARA LA CONFIGURACIÓN DE HARDWARE".

PRESIDENTE:	**M. EN C. JESÚS DÍAZ BARRIGA ARCEO**
VOCAL:	**ING. SILVIA VEGA MUYTOY**
SECRETARIO:	**ING. MA. GABRIELA GONZÁLEZ HERNÁNDEZ**
SUPLENTE :	**ING. IMELDA DE LA LUZ FLORES DÍAZ**
SUPLENTE:	**ING. VÍCTOR RAMÓN AGUILAR OCAMPO**

Quiero subrayar que el director de tesis es el **M. en C. Jesús Díaz Barriga Arceo,** el cual está incluido con base en lo que reza el reglamento de Exámenes Profesionales de esta Escuela.

Sin otro en particular, me es grato enviarle un cordial saludo.

A T E N T A M E N T E
"POR MI RAZA HABLARÁ EL ESPÍRITU"
San Juan de Aragón, Edo. de México, junio 25 del 2004.
EL JEFE DE CARRERA

M. EN C. JESÚS DÍAZ BARRIGA ARCEO

U.N.A.M. E.N.E.P. ARAGÓN INGENIERÍA EN COMPUTACION

c.c.p. Lic. Ma. Teresa Luna Sánchez.- Jefa del Departamento de Servicios Escolares.
M. en C. Jesús Díaz Barriga Arceo. Asesor.
Interesado.

JDA*vjd.

Agradezco al Instituto Mexicano del Petróleo
por permitirme desarrollar mi tesis
dentro del programa de seguridad y a el
M. en C. José Manuel Cervantes Martínez
por brindarme su apoyo y tiempo

Agradecimientos

Quiero dedicar este trabajo a las personas
que me ayudaron directa
o indirectamente a la
realización de este sueño:

En especial a mi mamá
Alicia Rendón Larrea
por apoyarme y
alentarme en los momentos
mas difíciles.

A mi amigo Juan Jesús Mateos Chavez
por sus sabios consejos
en los momentos de confusión.

A mi compañero y amigo
Francisco Javier González Blancas
por apoyarme durante la carrera
y en el desarrollo de este trabajo.

A mi director de tesis el M. en C.
Jesús Díaz Barriga Arceo
por ayudarme en darle forma
a este trabajo.

A mis revisores que con sus
sus comentarios me ayudaron
a enriquecer el trabajo.

Resumen

En este trabajo se presenta una propuesta de un sistema inteligente basado en la tecnología de agentes para la configuración del hardware de una computadora personal para apoyar las actividades de soporte técnico en el Instituto Mexicano del Petróleo. Además de mostrar una breve descripción de las metodologías comúnmente usadas en el análisis y diseño de sistemas multiagente.

También se desarrolló un prototipo que incluye dos agentes: el agente que apoya las actividades de la secretaria y el agente que ayuda al usuario a configurar su conexión de red.

Abstract

This paper shows a proposal of intelligent system by agent technology for hardware configuration of personal computer (PC). This system will help to the activities of technical support at Mexican Institute of Petroleum. Besides shows a brief description of the methodologies generally used in analysis and designed of multiagent system.

Also a prototype was developed with two agents: the agent helps the activities of secretary and the other agent helps to user to configure the connection of net.

Índice de contenidos

Lista de Tablas y Figuras

Introducción

El documento que aquí se presenta, describe uno de los paradigmas de programación que más desarrollo tiene, el paradigma de agentes inteligentes.

En un principio se presenta un pequeño panorama general acerca de los agentes inteligentes dando alguna de las definiciones mas usadas. Después se describe la forma en como se comunican, y los elementos que utilizan para tal objetivo. En el siguiente punto se encuentra una definición de sistema multiagente.

En el siguiente capítulo se encuentra una breve descripción de algunas de las metodologías más utilizadas en el área de los sistemas multiagente; presentando sus principales características y un comentario de las mismas. En un siguiente punto se comparan para obtener la que mejores ventajas nos presenta y poderla utilizar posteriormente.

En el siguiente capítulo se encuentra el desarrollo de un sistema multiagente que configura el hardware de una computadora personal. Este sistema tienen como objetivo apoyar en las actividades del área de soporte técnico del Instituto Mexicano del Petróleo.

En el último capítulo está el desarrollo de un prototipo que sirve como base para el desarrollo completo del sistema propuesto. El prototipo realiza la configuración de la tarjeta de red, esto es, revisará la configuración y si ésta se encuentra con alteraciones procederá a recuperar los valores perdidos (dirección IP, DNS, puerta de enlace entre otros).

La arquitectura que se utilizó en el desarrollo del prototipo fue la de un agente reactivo. Los agentes implementados para el prototipo fueron: el agente del usuario, y el agente de la secretaria.

Objetivos

Objetivo general

Estudiar y analizar las metodologías para el desarrollo de sistemas multiagente mas utilizadas, para aplicar la que sea adecuada en el análisis, diseño y construcción de un prototipo inteligente basado en agentes, que apoye las labores realizadas en el área de Soporte al hardware en el Instituto Mexicano del Petróleo, con el propósito de brindar un servicio de mayor calidad a los usuarios de esta área.

Objetivos específicos

- Elegir la metodología más adecuado para el desarrollo del sistema
- Analizar y diseñar un sistema que permita dar solución al problema de configuración de hardware
- Construir un prototipo que sea susceptible de ser utilizado en la configuración del hardware

Entorno del problema

Panorama general

El Instituto Mexicano del Petróleo se encuentra estructurado por diversos programas conformados por Direcciones Ejecutivas. La Dirección de Planeación y Desarrollo Institucional y sus requerimientos son el punto de partida.

Como parte de esta Dirección, se encuentra la Gerencia de Tecnología Informática, la cual está dividida en diferentes áreas. El área de interés para el estudio en este trabajo es la de Soporte Técnico, específicamente el mantenimiento al hardware.

Los objetivos que tiene planteados el área son los siguientes:

- Atender en forma rápida y oportunamente a los usuarios/clientes.
- Proporcionar con calidad, eficiencia y oportunidad la atención de los requerimientos de los usuarios.
- Mantener los equipos de cómputo en óptimas condiciones de funcionamiento.

Situación actual

El personal con el que cuenta actualmente el área de mantenimiento al hardware es de dos técnicos, una secretaria y el supervisor del área, estas son personas contratadas por el Instituto; y además de uno a cinco becarios que prestan servicio social.

Proceso actual

En la actualidad el proceso de atención a los usuarios tiene la siguiente secuencia: Cuando el usuario detecta un problema en su equipo hace el reporte respectivo vía telefónica al área correspondiente, dependiendo del tipo de problema que tenga.

Al recibir el reporte de falla, se le coloca en una cola, por lo que la atención se puede realizar en un tiempo variable que depende del número de solicitudes anteriormente recibidas.

Problemática detectada

De lo anterior se pueden listar los siguientes problemas detectados:

- La cantidad de personas asignadas no son las suficientes para lograr los objetivos planteados por el área de mantenimiento al hardware. La cantidad de usuarios es mayor al número de personas que dan soporte a dichos usuarios; por lo consiguiente el servicio prestado en ocasiones se muestra ineficiente.

- El tiempo de repuesta es grande. En ocasiones se presentan casos en los cuales se recibe un número alto de solicitudes de servicio, y como consecuencia el tiempo de respuesta crece.

- El usuario no sabe a que área dirigirse, por no identificar el problema con exactitud. Cuando un usuario tiene un problema con su equipo, no conoce la causa del mismo, por consiguiente no sabe a que área dirigir el reporte para que se le atienda. En este caso el usuario hace el reporte al área que considera adecuada. Si el problema no corresponde al área que recibió el reporte, el usuario tendrá que volver a hacer la solicitud a otra área y por consiguiente tendrá que esperar más tiempo para que se le atienda de nuevo.
- Los problemas que se presentan, en su mayoría son porque los usuarios borran archivos de configuración. En ocasiones los problemas que se tienen con los equipos de cómputo, son ocasionados porque los usuarios borran accidentalmente archivos que son necesarios para que funcionen correctamente los dispositivos; y no por una falla física del equipo.

Cabe mencionarse que en lo referente a los recursos humanos, dado que el IMP es un organismo descentralizado regulado por las disposiciones federales, en este momento las contrataciones de personal están cerradas debido a recortes presupuestales. La única forma en que la institución puede obtener más recursos humanos es a través de convenios con las Instituciones de Educación Superior (IES) que permiten a los estudiantes de los últimos semestres realizar su servicio social en el instituto a cambio de una ayuda económica. El IMP en este momento

no tiene considerado contratar más personal para el área de mantenimiento del hardware.

Solución propuesta

Considerando los problemas que se especifican en el punto anterior, se propone la siguiente solución:

Dado que los problemas que se presenta, por lo general en una computadora, relacionados con el hardware son de configuración, y muy rara vez se presentan problemas físicos; además de que el personal resulta ser insuficiente para atender a los usuarios, y el tiempo de traslado al lugar de los usuarios es alto, se propone utilizar un sistema automatizado que revise periódicamente la configuración de los equipos de cómputo que pertenecen a los usuarios para evitar que se pierdan los archivos de configuración respectivos. En el caso de que suceda, el sistema debe ser capaz de detectar el problema y resolverlo, de lo contrario el sistema dará aviso al usuario para que lo reporte, además de darle la opción de mandar el reporte al área correspondiente.

El sistema deberá ayudar a la secretaria a elaborar los documentos necesarios; al técnico a detectar problemas en las computadoras; y deberá contener un módulo para la administración del mismo.

Otras de las características que debe de tener el sistema es la interacción con el sistema encargado de instalar software para dar un mejor servicio.

Para lograr lo anterior el sistema será desarrollado con un nuevo paradigma denominado Agentes, pues estos tienen la capacidad de comunicarse con otros agentes para lograr sus objetivos, además de poseer la característica de poderse desplazar por la red.

Justificación

El área de Soporte técnico es un área importante dentro de cualquier empresa o institución, puesto que es la encargada de mantener los equipos de cómputo con un funcionamiento correcto. Los servicios que presta deben de proporcionarse en un tiempo mínimo, para lograr los objetivos planteados por la empresa.

Para desarrollar sus actividades de una manera más fácil y oportuna se puede crear software inteligente que resuelva problemas de configuración en los equipos sin la intervención del humano. Lo anterior proporciona un beneficio, puesto que los técnicos a cargo de un servicio puedan atender a otros usuarios que también tengan problemas que requieran de su intervención.

Cuando disminuye la carga de trabajo en el área, el servicio hacia los usuarios es más rápido, el personal no tiene detenido su trabajo y por consiguiente los proyectos se terminan a tiempo.

Capítulo 1
Generalidades de agentes

Introducción

Al igual que en la programación Orientada a Objetos tenemos elementos básicos con los que se trabaja de una forma esencial como son las clases, mensajes, objetos; en la programación de sistemas de agentes se tienen varios elementos clave.

En este capítulo se abordan los términos básicos que se encuentran en el área de los agentes inteligentes; tales como: definiciones sobre agentes, características, arquitecturas y lenguaje de comunicación. Además de proporcionar información relacionado con las aplicaciones que se han realizado.

1.1 Agentes Inteligentes

Aunque los investigadores en agentes vienen de una gran variedad de campos, las comunidades de la Inteligencia Artificial Distribuida (DAI) y la Computación Distribuida (DC) sobresalen como las áreas tradicionales en la investigación de agentes [1].

Una de las áreas de la Inteligencia Artificial Distribuida que mayor importancia ha tomado es la de los Agentes Inteligentes. También se puede decir que es el campo con mayor potencial de desarrollo en la computación.

Se dice que el estudio de este campo se inicia en 1987 con el modelo de actor concurrente de C. Hewitt al proponer el concepto de un objeto con estado interno, siendo capaz de comunicarse vía mensajes con otro objeto parecido. En este

modelo se propone el concepto de autocontenido, interactivo, y de objeto concurrentemente ejecutable que él nombró como 'actor' [2].

Shoham propone llamarle a esta nueva teoría "Paradigma de programación orientado a Agentes (POA)", está basada en lógica cognitiva y en la teoría de actos del habla de Searle [8].

1.1.1 ¿Qué es un Agente?

El término agente es difícil de definir. Los agentes son normalmente definidos como entidades con atributos considerados útiles en un dominio particular. Éste es el caso de los agentes inteligentes, donde los agentes son vistos como entidades que emulan procesos mentales o simulan un comportamiento racional; tales como asistentes personales, donde los agentes son entidades que ayudan a los usuarios a realizar una tarea; agentes móviles, donde las entidades son capaces de vagar por una red para conseguir sus objetivos; agentes de información, donde los agentes filtran y organizan de forma coherente datos dispersos y no relacionados; y agentes autónomos, donde los agentes son capaces de cumplir acciones no supervisadas [3].

En la actualidad no existe una definición académica ampliamente aceptada sino varias ideas intuitivas.

Veamos algunas definiciones:
- Shoham
"Un agente es una entidad que tiene un estado interno formado de componentes mentales, tales como creencias, capacidades, elecciones y compromisos" [8].

- Maes
"Es un sistema comprometido en alcanzar metas en un medio ambiente complejo y cambiante" [9].

-Wooldridge y Jennings
"Es aquel sistema de cómputo que tiene las propiedades de ser autónomo, auto-contenido, reactivo, proactivo, típicamente controlado de manera central y que es capaz de comunicarse con otro agente a través de un lenguaje" [10].

-FIPA
"Un agente es el actor fundamental en una plataforma de agentes que combina una o mas capacidades de servicios en un modelo de ejecución unificado e íntegro que puede incluir acceso al software externo, usuarios humanos y medios de comunicación" [11].

-I.B.M. Aglets [12]
"Los agentes inteligentes son entidades programadas que llevan a cabo una serie de operaciones en nombre de un usuario o de otro programa, con algún grado de

independencia o autonomía, empleando algún conocimiento o representación de los objetivos o deseos del usuario".

-Smith et al (el agente KidSim)
"Definamos un agente como una entidad software persistente dedicada a un propósito específico. 'Persistente' distingue a un agente de las subrutinas; los agentes tienen sus propias ideas acerca de cómo cumplir las tareas, su propia agenda. 'Propósito específico' los distingue de todas las aplicaciones multifunción; los agentes son típicamente más pequeños".

En esta tesis la definición que se adoptó es la de I.B.M Aglets por adecuarse al tipo de sistema que se va a desarrollar.

1.1.2 Características de los agentes

Algunos investigadores dan alguna clasificación comprensible de los atributos que un agente puede tener.

Una lista de los atributos comunes en los agentes se muestra a continuación [3,4]:

- Adaptación: la habilidad de aprender y mejorar con la experiencia.
- Autonomía: dirigidos por el objetivo, preactivos y con un comportamiento propio.
- Comportamiento colaborador: la habilidad de trabajar con otros agentes para conseguir un objetivo común.
- Capacidad de inferencia: la habilidad de actuar con especificaciones abstractas.
- Habilidad de comunicación a nivel de conocimiento: la habilidad de comunicarse con otros agentes con un lenguaje más parecido a los actos de comunicación humanos que a la típica comunicación a nivel de símbolo de los protocolos entre programas.
- Movilidad: la habilidad de migrar de la plataforma que lo contiene a otra por su propia decisión.
- Personalidad: la habilidad de manifestar atributos de un comportamiento humano creíble.
- Reactividad: la habilidad de "sentir" y actuar de una forma selectiva.
- Continuidad temporal: persistencia de la identidad y del estado durante los largos periodos de tiempo.
- Actúan continuamente.
- Capacidad para resolver problemas.
- Racionalidad.
- Habilidad social

Una de las propiedades más importantes es la adaptación. Para lograr esto se requieren otras propiedades tales como: percepción, interacción, conocimiento, aprendizaje, coordinación, comportamiento secuencial y orientado a metas y experiencia.

1.2 Arquitecturas de Agentes [5]

Por supuesto, hay multitud de clasificaciones que dependen del punto de vista del investigador.

Las dos grandes familias clásicas de agentes son las de los deliberativos y la de los reactivos. Un agente deliberativo es aquel que "contiene un modelo simbólico del mundo representado internamente de forma explícita y que toma decisiones mediante razonamiento lógico" [2]. Por otro lado, un agente reactivo es aquel que "no incluye un modelo simbólico del mundo ni usa razonamiento simbólico complejo de ningún tipo" [2].

1.2.1 Agentes deliberativos [5]

Los agentes deliberativos usan un modelo del mundo, explícitamente representado y funcionan siguiendo el paradigma de los sistemas clásicos de planificación de la Inteligencia Artificial (IA), basado en el ciclo percepción-planificación-acción. Se muestra la arquitectura genérica representada en el diagrama estructural de la Figura 1.1.

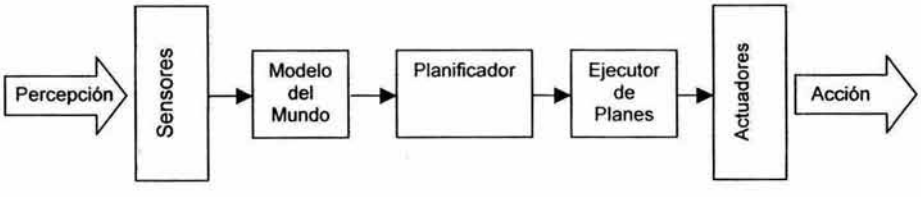

Figura 1.1. Arquitectura básica de un agente deliberativo

El modelo de agente más representativo del tipo de los deliberativos es el BDI (Belief-Desire-Intention). Estos se caracterizan por tener "ciertas actitudes mentales" [6] tales como las creencias, deseos e intenciones.

Las creencias corresponden al conocimiento que el agente posee sobre el resto del mundo. Desde el punto de vista de la implementación, las creencias pueden ser desde un conjunto de variables hasta una base de datos, pasando por un conjunto de expresiones lógicas. En definitiva, cualquier estructura de datos.

Por otro lado, los deseos se refieren a cómo se ordenan, en términos de prioridades o coste, los múltiples objetivos que tiene actualmente el agente. Así podemos decir que guían la motivación de éste.

La parte de las intenciones es la que representa la estrategia de acción que el agente está siguiendo actualmente. Por ello, es posible que las intenciones actuales estén sujetas a reconsideración. Desde el punto de vista de la programación, se trata de representar la última acción o secuencia de acciones llevadas a cabo en el entorno.

1.2.2 Agentes reactivos

Los agentes reactivos no incluyen representación explícita de conocimiento simbólico complejo. Se trata de ofrecer respuestas inmediatas a estímulos del entorno. A estos agentes Nilsson los denomina también agentes estímulo-respuesta [7]. Se tratan de agentes que tienen una percepción concreta a partir de la cual se disparan las acciones pertinentes, dependiendo de las condiciones activadas en el proceso perceptivo. Si se representa la arquitectura típica de un agente reactivo, se tiene el diagrama estructural de la Figura 1.2.

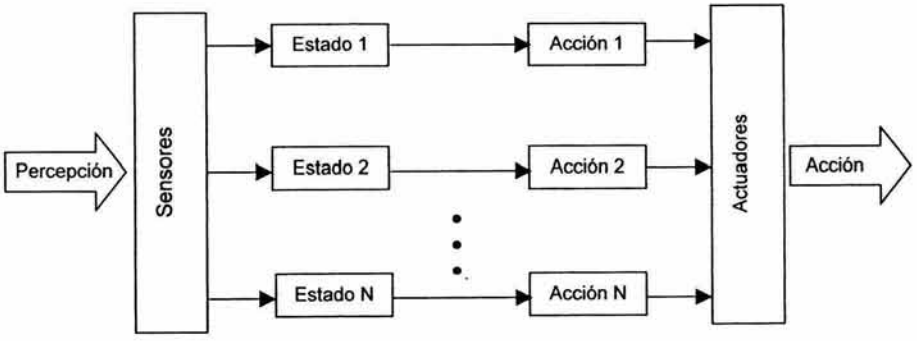

Figura 1.2. Diagrama estructural de la arquitectura de un agente reactivo

1.2.3 Agentes Híbridos [13]

Las arquitecturas deliberativas y reactivas presentan diferencias sustanciales. Aún así, en la literatura podemos encontrar un enfoque híbrido que combina aspectos de las dos arquitecturas para dar lugar a una arquitectura híbrida como se presenta en la Figura 1.3. Como puede probarse, un agente híbrido típico dispone de componentes deliberativos que permiten realizar razonamientos complejos, realizar planes y tomar decisiones. Todo esto en combinación con componentes reactivos que permiten la reacción inmediata ante eventos para los cuales es necesario ese tipo de reacción.

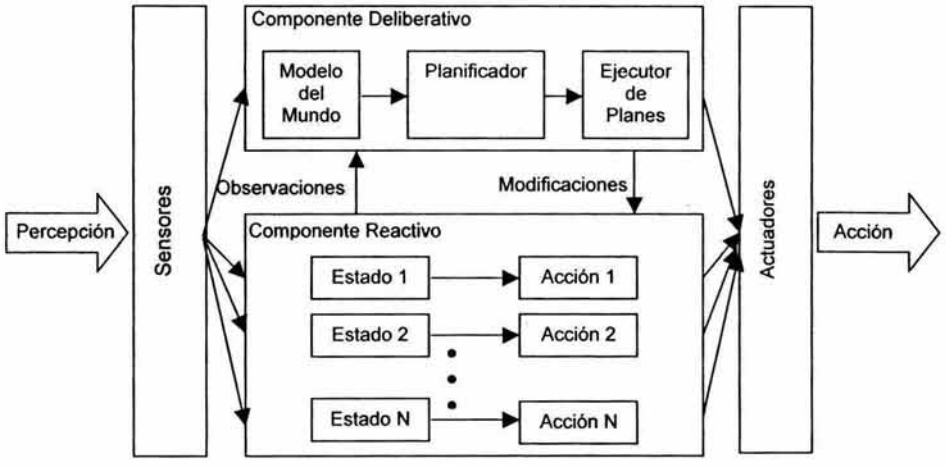

Figura 1.3. Diagrama estructural de la arquitectura de un agente híbrido

1.3 Sistemas Multiagente

Normalmente un agente no actúa aislado, sino que se localizan en entornos o plataformas con varios agentes, donde cada uno de ellos tiene sus propios objetivos, toma sus propias decisiones y puede tener la capacidad de comunicarse con otros agentes. Dichos entornos se conocen con el nombre de sistemas multiagente o agencias.

Como características principales de los sistemas multiagente se destacan las siguientes:

- Cada agente del sistema tiene un punto de vista limitado (no tiene la información completa).
- No existe un control global para todo el sistema.
- Los datos están descentralizados.
- La computación es asíncrona.
- Permite interoperación con sistemas existentes.

Sin embargo, este tipo de diseños suele plantear una serie de problemas, como la elección el tipo de comunicación entre los agentes, el tipo de plataforma o del método de desarrollo, de los criterios para la seguridad del sistema.

1.4 Lenguaje de comunicación entre agentes [14]

Muchas de las dificultades de la comunicación entre agentes inteligentes podrían atacarse al darles un lenguaje común. En términos lingüísticos, esto significa que deben compartir una sintaxis común, una semántica y pragmática.

Hay dos aproximaciones principales para diseñar un lenguaje de comunicación entre agentes. La primera aproximación es procedural, donde la comunicación se basa en un contenido ejecutable. Esto podría hacerse usando lenguajes de programación como Java o TCL. La segunda aproximación es declarativa, donde la comunicación está basada en sentencias declarativas, como definiciones, asunciones, entre otras.

El método procedural se basa en la idea de que la comunicación se puede modelar más adecuadamente como un intercambio de directivas. De esta manera se pueden transmitir no sólo comandos sino también programas enteros. Usualmente su ejecución es eficiente y directa. Las desventajas principales con éste método son el requerimiento de información sobre el receptor del mensaje, la cual no siempre está disponible, y el carácter unidireccional del procedimiento, cuando la mayoría de las veces es conveniente que los agentes compartan información.

El método declarativo por el contrario, establece que la comunicación se modela de forma más adecuada utilizando enunciados declarativos. Este método para ser útil requiere que el lenguaje sea lo suficientemente expresivo como para comunicar diferentes clases de información incluyendo procedimientos, que sean compactos y que no se requiera que el vocabulario crezca demasiado para seguir manteniendo la comunicación. Uno de los leguajes declarativos más populares es KQML (Knowledge Query Manipulation Language).

1.4.1 ACL [14]

Como resultado del esfuerzo por crear un lenguaje que permitiera la interoperación entre agentes autónomos distribuidos surgió el lenguaje llamado ACL (Agent Communication Language). ACL tiene tres componentes: un vocabulario, un lenguaje de contenido llamado KIF (Knowledge Interchange Format) y un lenguaje de comunicación llamado KQML. Un mensaje de ACL es un mensaje en KQML que consiste de una directiva de comunicación y un contenido semántico en KIF expresado en términos del vocabulario [3].

1.4.2 Vocabulario [14]

El vocabulario de ACL es un diccionario de palabras apropiado para áreas comunes de aplicación. Cada palabra en el diccionario tiene una descripción que es usada por los agentes para entender su significado y una anotación formal (escrita en KIF) utilizada por los programas. El diccionario es abierto, es decir, es posible añadir nuevas palabras en áreas ya definidas y en nuevas áreas de aplicación.

La existencia de este diccionario no significa que solamente hay una manera de describir un área de aplicación. Un diccionario puede contener múltiples ontologías para un área dada y un agente puede utilizar la ontología que le sea más

conveniente. Las definiciones formales asociadas con cualquiera de estas ontologías pueden ser usadas por los agentes para traducir mensajes que emplean una ontología en específico a mensajes que usan otras ontologías [14].

1.4.3 KIF [14]

KIF es una versión en prefijo del cálculo de predicados de primer orden con varias extensiones para incrementar su expresividad.

Este formato para intercambio de conocimiento permite primeramente expresar datos simples. Por ejemplo, las siguientes oraciones codifican dos tuplas en una base de datos personal. El primer argumento es el número de seguro social, el segundo es el departamento en el que la persona trabaja y el tercero es el salario de la persona

(salario 016-46-3946 compras 7200)
(salario 415-32-4707 desarrollo técnico 4800)

Se puede expresar información más complicada mediante el uso de términos complejos. Por ejemplo, la siguiente oración afirma que una pieza es más grande que la otra:

(>(*(ancho pieza1)(largo pieza1))(*(ancho pieza2)(largo pieza2)))

KIF incluye una variedad de operadores lógicos para ayudar a codificar información lógica (esto es, negaciones, disyunciones, reglas, cuantificadores, entre otros). La siguiente es una oración compleja en KIF que quiere decir que cualquier número X elevado a la n es positivo si X es real y n es par.

(<=(>(exp. ?x ?n) 0)(número-real ?x)(número-par ?n))

KIF también puede ser usado para describir procedimientos, esto es, escribir programas para agentes. Dada la sintaxis en prefijo de KIF, tales programas se parecen a LISP o SCHEME. El siguiente es un ejemplo de un procedimiento de tres instrucciones escrito en KIF.

(secuencia (fresh-line t) (print "?Hola!")(fresh-line t))

La semántica del KIF básico (KIF sin reglas ni definiciones) es similar a la lógica de primer orden. Existen extensiones para manejar operadores que no son estándar y una restricción a modelos que satisfacen algunos axiomas. A pesar de estas extensiones y restricciones, el lenguaje conserva las características fundamentales de la lógica de primer orden [3].

1.4.4 KQML [14]

KQML, acrónimo de Knowledge Query and Manipulation Language, es un lenguaje y protocolo que soporta la programación en red específicamente de sistemas basados en conocimiento o agentes inteligentes. Fue desarrollado por ARPA apoyado por el Knowledge Sharing Effort, e implementado de manera separada por varios grupos de investigación.

KQML fue concebido tanto como un formato de mensajes como un protocolo de manejo de mensajes para el soporte de comunicación en tiempo real para el intercambio de conocimiento entre agentes. Este lenguaje puede verse dividido en tres capas: una capa de comunicaciones (describe los parámetros de comunicación a bajo nivel, el emisor, el receptor, y los identificadores de la comunicación); la capa de mensajes (contiene una performativa y que indica el protocolo de la interpretación); y la capa de contenidos (contiene información relativa a la preformativa enviada). El formato de un mensaje en KQML se muestra en la Figura 1.4.

```
(register
        :sender      agentA
        :receiver    agenteB
        :reply-with  message2
        :language    common_language
        :ontology    common_ontology
        :content     "(ServiceProvision Manufacturing:TaskDecomposition)"
)
```

Figura 1.4. Ejemplo de un mensaje en KQML

Tres características importantes de KQML son [14]:

1. Los mensajes de KQML son opacos al contenido de lo que transportan, esto es, los mensajes en KQML no comunican únicamente oraciones en un lenguaje, sino que comunican una actitud acerca del contenido (por ejemplo, afirmación, solicitud, pregunta).
2. Las primitivas del lenguaje se llaman preformativas, las cuales indican las acciones u operaciones permitidas que los agentes pueden utilizar cuando se comunican.
3. Un ambiente en el que los agentes se comunican con KQML puede ser enriquecido con un tipo de agentes especiales llamados facilitadores.

1.5 Formas de comunicación en los sistemas multiagente

La forma en que se comunican los agentes depende en gran medida de la manera de organizarlos. Dentro de las más importantes podemos mencionar dos: una es mediante comunicación directa y la otra es por coordinación asistida.

1.5.1 Comunicación directa [14]

Una organización de agentes por comunicación directa se caracteriza por agentes que manejan su propia coordinación, con la ventaja de que no depende de otros programas y la desventaja de que aumenta el grado de complejidad en la implementación de cada agente. Dentro de esta forma de organización existen dos arquitecturas que son la red de contratos y la de especificación compartida.

En la arquitectura de red de contratos los agentes cuando necesitan algún servicio distribuyen solicitudes a diferentes agentes. Los receptores de estas solicitudes las evalúan y lanzan ofertas, las cuales son usadas por los agentes solicitantes para decidir con que agente realizar un contrato. Esta arquitectura es evidentemente costosa por la cantidad de mensajes que se requieren enviar.

En la arquitectura de especificación compartida los agentes no hacen solicitudes de servicio sino que proveen de información a otros agentes acerca de sus capacidades y necesidades. Cuando surge la necesidad de un servicio esta información es utilizada por los agentes para coordinar sus actividades. El número de mensajes que se intercambian se reduce considerablemente en comparación con la arquitectura anterior, lo que hace una arquitectura más eficiente.

1.5.2 Coordinación asistida

Otra forma de organizar a los agentes es mediante la coordinación asistida y un ejemplo bastante popular de este tipo de organización es la arquitectura conocida como sistema federado. En este sistema los agentes no se comunican directamente, lo hacen por medio de facilitadores. En esta arquitectura los agentes generalmente utilizan ACL para comunicar sus necesidades y habilidades a su facilitador local, el cual se encarga de encontrar la ruta correcta por la cual hacer llegar solicitudes a otros facilitadores, quienes a su vez pasan las solicitudes a alguno de los agentes de su dominio para satisfacer la solicitud [14].

Una característica importante de la arquitectura de federación es que soporta la interacción anónima entre los agentes a cambio de que estos cedan algo de su autonomía al facilitador y de que se apeguen a condiciones adicionales. Para cada agente debe parecer que existe sólo un agente que maneja todas sus solicitudes directamente [14].

Los servicios que el facilitador provee son los siguientes:

- Encontrar la identidad de agentes por su nombre.
- Encontrar la identidad de los agentes capaces de desempeñar una tarea.
- Enviar un mensaje a un agente específico.
- Hacer uso de las especificaciones para manejar las solicitudes y dar la impresión de que él provee todos los servicios.
- Descomponer solicitudes complejas en subprogramas, obtener las soluciones de cada subproblema y combinarlas para obtener las respuestas a la solicitud original.
- Traducir del vocabulario de un agente al vocabulario de otro.
- Monitorear su base de conocimiento para determinar si una solicitud puede ser satisfecha.

1.6 Aplicaciones de Agentes[15]

1.6.1 Aplicaciones industriales

1.6.1.1 Procesos de control

El proceso de control es una aplicación natural para los sistemas de agentes y multiagente, ya que los sistemas de control de procesos son autónomos y reactivos. No es sorprendente, por consiguiente, que un número de aplicaciones de procesos de control basados en agentes debieran haber sido desarrollados. El mejor conocido de esos es el ARCHON, un software para construir sistemas multiagente, y una metodología asociada para la construcción de aplicaciones con esta plataforma [17]. ARCHON ha sido usado en varias aplicaciones de procesos de control, incluyendo la administración de la transportación de la electricidad (la aplicación está en uso en el norte de España), y en el control del Acelerador de Partículas. ARCHON también tiene la distinción de ser uno de los primeros sistemas de multiagentes probados en el mundo. Los agentes en ARCHON son sistemas computacionales bastante pesados, con cuatro principales componentes. Un módulo de alto nivel de comunicación (HLCM), el cual administra la comunicación entre agentes. Un módulo de planeación y coordinación (PCM), el cual es esencialmente responsable de decidir que hará el agente. Un módulo de administración de información (AIM) que es responsable de mantener el modelo del mundo del agente, y finalmente un sistema fundamentalmente inteligente (IS), que representa la experiencia del agente. El HLCM, PCM y AIM juntos constituyen un tipo de 'envoltura de agente', que puede ser usado para encapsular un sistema inteligente existente.

1.6.1.2 Uso en el control del tráfico aéreo [15]

Sistema basado en agentes para el control del tráfico aéreo conocido como OASIS. Este sistema se encuentra en el campo experimental en el aeropuerto de Sydney en Australia, los agentes son usados para representar 2 aviones y varios sistemas de control de tráfico en operación. El agente provee un natural modo de modelado del mundo real con componentes autónomos. Cuando un avión entra al espacio aéreo de Sydney, un agente es reservado para él, y el agente se inicia con la información y la meta correspondiente para el avión en el mundo real.

1.6.2 Aplicaciones comerciales

1.6.2.1 Administración de información

Como la rica y diversa información disponible en el entorno que rodea al ser humano en la vida diaria ha aumentado a pasos agigantados, también la necesidad de administrar esa información creció. La carencia de herramientas para la administración efectiva de la información ha dado un alza a lo que es coloquialmente conocido como problema global de información. La gran cantidad de información disponible vía Internet y la World Wide Web (WWW) representa un problema real. Es posible caracterizar el problema global de información de dos formas:

- Filtrado de información: Todos los días, se encuentran enormes cantidades de información (vía e-mail y noticias, por ejemplo) y sólo una mínima proporción es relevante o importante. Se requiere clasificar y ordenar la información que sea de utilidad a las personas según sus intereses.
- Reunión de información: Se debe ser hábil para obtener información que sea útil a los requerimientos solicitados, si esta información puede ser sólo colectada de un determinado número de sitios.

A continuación se describen tres proyectos que se pueden utilizar para este problema:

MAXIMS
Es un agente que filtra los mails llamado MAXIMS. El programa "aprende" a priorizar, borrar, ordenar y archivar mails relacionados con el usuario. MAXIMS hace constantemente predicciones internas acerca de que hará el usuario con el mensaje.

NEWT
Programa que filtra noticias llamado NEWT. Este programa, implementado en C++ sobre UNIX, tiene como entrada una cadena de artículos de la red, y como salida da un subconjunto de esos artículos que recomienda al usuario que lea. El agente NEWT es programado por medio de ejemplos de entrenamiento.

Librería digital ZUNO
Una librería digital organiza y administra la colección de datos, junto con los servicios para asistir al usuario que hace uso de estos datos. La librería digital ZUNO (ZDL) es un sistema multiagente que ayuda al usuario a obtener información acorde a lo que busca.

1.6.2.2 Aplicación en telefonía móvil [16]

Los sistemas de telefonía móvil se caracterizan por una serie de restricciones: de ancho de banda ilimitado, alta probabilidad de error en la interfaz de radio, cobertura discontinua y limitada, baja capacidad de procesamiento en los sistemas finales, interfaz de usuario limitada, por mencionar algunos.

La utilización de la tecnología de agentes en estos sistemas permite adaptarse a estas limitaciones para proporcionar mejores servicios a los usuarios finales y mejorar las prestaciones de la red, debido a que:

- Los agentes que proporcionan el servicio pueden enviarse dinámicamente y bajo demanda a los propios usuarios.
- Los agentes permiten realizar distribución de tareas para realizar actividades de gestión, siendo los propios agentes quienes recopilen datos y los procesen localmente en la parte del terminal móvil.
- La autonomía de los agentes permite que se realicen tareas de forma asíncrona.
- Los agentes pueden realizar gran parte del procesamiento de forma local, por lo que se conseguirá una reducción importante del tráfico en la red.
- Los agentes permiten una mayor independencia de la disponibilidad de la red, ya que su capacidad de movilidad les permite migrar a otros nodos de la red.

1.6.3 Aplicaciones Médicas [15]

La informática médica es un área que mejor crece en la ciencia de la computación. Nuevas aplicaciones están siendo encontradas para computadoras todos los días en la industria del cuidado. No es sorprendente, por lo tanto, que los agentes deberían ser aplicados en este campo. Dos de las primeras aplicaciones están en el área de cuidado intensivo y monitoreo de pacientes.

1.6.3.1 Monitoreo de pacientes

El sistema guardián descrito en (Hayes-Roth et al., 1989) está pensado para ayudar en la administración del cuidado del paciente en cuidados intensivos de la Unidad Quirúrgica (SICU). El sistema estuvo motivado por dos cosas: primero, que el modelo de cuidado del paciente en un SICU es esencialmente de un equipo, donde un grupo de expertos en distintas áreas cooperan para el cuidado de la salud del paciente; y segundo, y mas importante de los factores en el buen

cuidado de la salud es el compartir adecuadamente la información entre miembros del equipo de cuidado crítico. El sistema guardián de monitoreo de pacientes funciona entre un número de agentes, de tres tipos diferentes:

- Agentes de Percepción/Acción: responsables de la interfaz entre el sistema y el mundo, trazando las entradas del sensor a una forma simbólica adecuada, y trasladando las acciones de petición al interior del sistema de control de comandos del efector.
- Agentes Razonadores: responsables de organizar las decisiones hechas por el sistema, y
- Agentes de Control: que solo será uno, con un control global del sistema.

1.6.3.2 Cuidado de la salud [15]

En 1996 se describió un prototipo basado en agentes un sistema para el cuidado médico. El sistema esta diseñado para integrar el proceso de administración del paciente, que típicamente involucra varios pacientes. Por ejemplo, un practicante general puede sospechar que un paciente tiene cáncer en el pecho, pero esta suposición no puede ser confirmada o rechazada sin la asistencia de un especialista del hospital. Si el especialista confirma la hipótesis, entonces un programa de cuidado puede ser diseñado para tratar al paciente, involucrando los recursos de otros individuos. El sistema prototipo permite una representación natural de este proceso. Los agentes en el prototipo contiene una base de conocimientos, contiene la experiencia del agente, una interfaz computadora-humano, permitiendo al usuario agregar, quitar, o ver las metas del sistema, y un administrador de comunicaciones, que realiza la funcionalidad de pasar los mensajes de los agentes. El componente del sistema inteligente esta basado sobre el modelo de experiencia KADS, y toda la arquitectura de agente está implementado en PROLOG.

1.6.4 Entretenimiento

En la industria del ocio se pueden encontrar varios juegos basados en agentes por ejemplo el desarrollo de una versión del popular Tetris para computadora [15].

Capítulo 2
Metodologías para el desarrollo de agentes

Introducción

Para el desarrollo de sistemas multiagente existen diversas metodologías y combinaciones de las mismas. En este capítulo se describen las más usadas para el modelado de sistemas multiagente, además de efectuar un análisis, que permita seleccionar la que mejor se adecue mejor para el análisis y diseño del sistema en estudio.

2.1 Metodologías orientadas a agentes

Una metodología de ingeniería de software provee métodos, guías, descripciones y herramientas para cada fase en el ciclo de vida de un sistema.

Aunque en los últimos años se ha prestado gran atención a la tecnología de agentes, aún no existe consenso sobre cual metodología es la óptima para el desarrollo de sistemas basados en agentes.

Una vez establecida la tecnología de agentes, y se han desarrollado diversas plataformas y lenguajes para emplear sistemas multiagente en variadas aplicaciones, han surgido metodologías que tratan de asistir en el ciclo de vida de la construcción de los sistemas multiagente para obtener las ventajas de reciclaje de los sistemas y mantenimiento, entre otras.

Podemos distinguir los siguientes enfoques:

- Metodologías orientadas a agente basadas en metodologías orientadas a objetos: parten de las metodologías orientadas a objeto añadiendo las peculiaridades de los agentes, tales como creencias, objetivos, planes, cómo identificar agentes, relaciones e interacciones entre agentes. [18].

- Metodologías orientadas a agente basadas en metodologías de ingeniería del conocimiento: extienden metodologías de ingeniería del conocimiento, añadiendo principalmente el modelado de las interacciones y la conducta preactiva de los agentes [19].

2.1.1 Extensiones de metodologías orientadas a objetos [27]

2.1.1.1 Ventajas del enfoque

En primer lugar, se pueden observar similitudes entre el paradigma orientado a objetos y el paradigma de agentes [18]. Desde el inicio de la Inteligencia Artificial Distribuida, quedó establecida la estrecha relación entre la tecnología multiagente y la propagación orientada a objetos recurrentes [20]. Tal y como enunció Shoham en su propuesta de programación orientada a agentes (AOP) [21], los agentes pueden considerarse como objetos activos, esto es, objetos con un estado mental. Ambas tecnologías comparten el paso de mensajes para comunicarse, y el empleo de herencia y agregación para definir su arquitectura. Las principales diferencias estriban en que los mensajes de los agentes tienen un tipo predeterminado (su acto comunicativo) y en que el estado mental del agente se basa en el conocimiento que tienen del ambiente en el que se encuentran, sus intenciones, además de los deseos (objetivos) y acuerdos.

Otra ventaja es el empleo habitual de los lenguajes orientados a objetos para desarrollar sistemas basados en agentes porque se consideran un entorno natural de desarrollo.

2.1.1.2 Desventajas del enfoque

Aunque existen similitudes entre las tecnologías de la orientación a objetos y la de agentes, evidentemente, los agentes no son nada más objetos. Por lo consiguiente, las metodologías orientadas a objetos no abordan aspectos que los diferencien [18].

En principio, los objetos tienen una estructura simple (atributos y métodos) mientras que los agentes tienen una estructura compleja. Una arquitectura de agente puede ser comparada a un módulo de las metodologías orientadas a objetos, y puede ser analizada e implementada siguiendo una metodología orientada a objetos.

Aunque tanto los objetos como los agentes emplean paso de mensajes para comunicarse, mientras que en los objetos el paso de mensaje es simplemente invocación de métodos, en los agentes este paso de mensaje se suele modelar como el intercambio de un conjunto predeterminado de actos de habla, con protocolos asociados para negociar o responder a cada acto comunicativo. Además, los agentes realizan un procesado de los mensajes, y tienen la

característica de decidir ejecutar, o no, la acción correspondiente al mensaje recibido.

Otro aspecto a considerar es que los agentes se pueden caracterizar por su estado mental, y las tecnologías orientadas a objetos no proporcionan técnicas para modelar cómo los agentes llevan a cabo sus inferencias y su proceso de planificación.

2.1.2 Metodologías existentes [27]

2.1.2.1 Técnicas de modelado para sistemas de agentes BDI [22]

La arquitectura BDI se inspira en un modelo cognitivo del ser humano. Este tipo de agentes se caracterizan por tener "ciertas actitudes mentales" como son las creencias, deseos e intenciones. Los agentes utilizan un modelo del mundo, una representación de cómo se le muestra el entorno, y funcionan siguiendo el paradigma de los sistemas clásicos de planificación de la Inteligencia Artificial (IA). El agente recibe estímulos a través de sensores ubicados en el mundo. Estos estímulos modifican el modelo del mundo que tiene el agente (representado por un conjunto de creencias). Para guiar sus acciones, el agente tiene deseos. Un deseo es un estado que el agente quiere alcanzar a través de intenciones. Las creencias corresponden al conocimiento que el agente posee sobre el resto del mundo. Estas son acciones especiales que pueden abortarse debido a cambios en el modelo del mundo. Aunque la formulación inicial es de Bratman, fueron Georgeff, Rao y Kinny quienes formalizaron este modelo y le dieron forma de metodología.

Esta metodología adopta un conjunto de modelos que operan a dos distintos niveles de abstracción: externo e interno para el modelado de agentes BDI.

"La descripción de un sistema agente desde el punto de vista externo, es capturado en dos modelos, que son altamente independientes de la arquitectura BDI.

- Modelo de agentes, para describir la jerarquía de relaciones entre agentes y las relaciones entre agentes concretos; y
- Modelo de interacción, para describir las responsabilidades, servicios e interacciones entre agentes y sistemas externos.

Desde el punto de vista interno, cada clase de agente está especificado por tres modelos, específico de la arquitectura BDI.

- Modelo de Creencias, que describe las creencias acerca del ambiente;
- Modelo de objetivos, que describe los objetivos y eventos que un agente puede adoptar o responder; y
- Modelo de planes, que describe los planes que un agente puede utilizar para lograr sus objetivos." [22]

El proceso de desarrollo del nivel externo, se inicia con la identificación de los roles (funcionales, organizacionales, etc) del dominio de la aplicación, a fin de identificar los agentes y organizarlos en una jerarquía de clases de agentes descrito utilizando una notación parecida a la OMT. Luego las responsabilidades asociadas a cada rol son identificadas, junto con los procesos proveídos y utilizados para cumplir con las responsabilidades. El siguiente paso es la identificación de las interacciones necesarias para cada servicio. Finalmente, se colecta la información en un modelo de instancias de agente.

El desarrollo del nivel interno, se inicia con el análisis de las diferentes formas (planes) de lograr un objetivo. Los planes para responder a un evento o lograr un objetivo.

En esta metodología, la integración con el ciclo de vida de software es reducida. Los autores proponen una serie concreta de modelos. Estos pasos se repiten haciendo que los modelos, que capturan el resultado del análisis, sean progresivamente elaborados, revisados y refinados. Además, el refinamiento de los modelos internos conllevan la realimentación de los modelos externos.

Comentarios

Como se observó esta es una metodología que solo se puede aplicar a los agentes de tipo BDI. Contiene una parte que es general, pero si se aplica solo esta parte se necesita utilizar una metodología adicional.

Otro desventaja que tiene es que no considera todas las etapas del ciclo de vida de los sistemas, solo toma en cuenta la parte del análisis. Además no considera el análisis de sistemas multiagente solo aplica a agentes individuales.

La ventaja que tiene es la amplia utilización de la metodología Orientada a Objetos, que le facilita el trabajo al analista, pues utiliza en la representación gráfica diagramas de la tecnología de objetos.

2.1.2.2 Metodología MaSE [30]

MaSE (Multi-agent Systems Software Engineering) fue desarrollada en el Air Force Institute of Technology. Se concibe como una abstracción del paradigma orientado a objetos donde los agentes son especializaciones de objetos. En lugar de simples objetos, con métodos que pueden invocarse desde otros objetos, los agentes se coordinan unos con otros vía conversaciones y actúan proactivamente para alcanzar metas individuales y del sistema.

En MaSE los agentes son simplemente una abstracción conveniente, que puede o no poseer inteligencia. En este sentido, los componentes inteligentes y no inteligentes se gestionan igualmente dentro del mismo armazón. Dado el enfoque inicial, los agentes se ven como especializaciones de objetos.

El principal objetivo de MaSE es guiar a un desarrollador a través de un ciclo de vida del software, desde una especificación escrita en prosa hasta un sistema de agentes implementado.

Los pasos en el diseño a nivel del Dominio son:

1. Identificar los tipos de agentes
2. Identificar las posibles interacciones entre los tipos de agentes
3. Definir protocolos de coordinación para cada tipo de coordinación

En esta primera fase de la metodología se toman las especificaciones iniciales del sistema y las transformamos en un conjunto de metas del sistema.

En MaSE una meta siempre es definida a nivel del sistema como un objetivo. Como consecuencia, cada acción dentro del sistema debe soportar una meta determinada. Una meta es una sentencia que generalmente se expresa como una posible funcionalidad del sistema.

Las metas que fueron estructuradas se transforman en una forma más útil para la construcción de sistemas multiagentes: los roles y sus tareas asociadas.

Los roles son los bloques de construcción utilizados para definir clases de agentes durante la fase de diseño. Los roles están definidos por cuatro atributos: responsabilidades, permisos, actividades y protocolos.

Luego de que los roles fueron creados, se asocian tareas para cada rol. Cada meta asociada con un rol puede tener tareas que definen los detalles de cómo se logra o alcanza una de las metas planteada. Esto se debe de hacer luego de la creación de los roles ya que las tareas se comunican con otras tareas en varios roles.

Las tareas se representan a través de rectángulos con vértices circulares, mientras que el paso de mensajes entre tareas se representan mediante una flecha con el nombre del mensaje por encima de la misma.

El diagrama de secuencia[1] se utiliza para determinar la cantidad mínima de mensajes que se debe de pasar entre roles.

1."Diagrama que muestra interacciones entre objetos organizado en una secuencia de tiempo. En particular muestra, los objetos participando en la interacción y la secuencia de mensajes que se intercambian". OMG Unified Modeling Languaje Specification (DRAFT), Febrero 2001

Los pasos específicos de diseño a nivel de agentes son:

1. Mapear las acciones identificadas en la conversación de los agentes a componentes internos
2. Definir estructuras de datos identificadas en la conversación de agentes. Esas estructuras de datos representan input u outputs de los agentes
3. Definir estructuras de datos adicionales, internas al agente. Estas estructuras de datos representan los flujos de datos entre los componentes en la arquitectura.

Las clases agentes se identifican a partir de los roles. El producto de esta etapa de la metodología es el diagrama de clases de agentes, que muestra las clases agente existentes junto con las conversaciones que mantienen dichos agentes. "Una clase agente es una plantilla para un determinado tipo de agente que habrá en el sistema, de la misma manera que una clase objeto es una plantilla para el paradigma de objetos." Wood [29].

Después se construyen las conversaciones. Una conversación bajo MaSE define el protocolo coordinado entre dos agentes. La metodología define específicamente una conversación como dos diagramas de comunicación de clases, uno para el iniciador de la conversación y otro para el que responde. Este diagrama es un par de máquinas de estados que definen los estados de la conversación de las dos clases agentes que participan en la misma.

La etapa de ensamblado de agentes es la siguiente fase. En esta parte del modelo se define la arquitectura de que habrán de tener los agentes del sistema.

Diseño de los componentes: una vez que la arquitectura del agente ha sido definida, los componentes especificados deben ser diseñados.

Los pasos específicos del diseño del sistema incluyen:

1. Seleccionar los tipos de agentes que son necesarios
2. Determinar el número de agentes requeridos para cada tipo y definir:

    - La ubicación física o dirección de los agentes
    - Los tipos de conversaciones que los agentes serán capaces de mantener
    - Cualquier otro parámetro definido en el dominio.

Desarrollo del sistema

La fase final de la metodología MaSE toma las clases agentes y las instancia como agentes reales. Se utiliza un diagrama de desarrollo para mostrar números, tipos y localidades de los agentes dentro del sistema. Este diagrama es el mas simple de la metodología, ya que la mayor parte del trabajo se hizo en fases anteriores. El objetivo del Diagrama de Desarrollo es definir un sistema basado en clases agentes en las fases previas de MaSE.

El proceso de desarrollo de MaSE se puede realizar con una herramienta llamada AgentTool [30].

Comentarios

MaSE usa intensivamente máquinas de estados para especificar el comportamiento de diversos elementos de la especificación. Aunque las máquinas de estados sean ampliamente aplicados en la industria, no es claro si son útiles para la especificación de los elementos de un sistema MultiAgente.

Otro de los problemas que tiene esta metodología, que no cubre todos los aspectos de la ingeniería de software, en particular en la parte de la implementación, la menciona pero no la desarrolla.

En cuanto a AgentTool, la herramienta que soporta esta metodología, si bien es una ventaja que genere el código, también es un inconveniente pues lo mezcla con el código de la herramienta, con lo cual a la hora de hacer cambios es necesario recompilar la herramienta. Además de que AgentTool no incorpora la totalidad de las etapas.

2.1.2.3 Metodología MESSAGE [23]

MESSAGE (Methodology for Enginnering Systems of Software Agents), es una versión inicial de una metodología para el desarrollo de software orientado a agentes, en particular sistemas multiagente (SMA) para aplicaciones de Telecomunicaciones. La metodología provee un lenguaje, un método y unas guías de cómo aplicar la metodología centrándose en las fases de análisis y diseño y lanzando ideas sobre el resto de las etapas como implementación y pruebas.

Se presenta un conjunto de 5 modelos de análisis, que pueden ser usados por el analista para capturar diferentes aspectos de un sistema agente. Los modelos son descritos en términos de un conjunto de conceptos interrelacionados. A continuación se describen brevemente los modelos:

Modelo de Organización: Este describe la estructura general del sistema. Especifica el número y el tipo de agentes y como están interrelacionados en términos de "relaciones". Contempla dos vistas: social, donde se describen los agentes en forma global; e individual donde los agentes son descritos como si solo existiera uno.

Modelo metas/tareas: este captura lo que hace el sistema agente en términos de las metas que deben alcanzar y las tareas que deben realizar para alcanzarlas. El modelo también captura la manera que se relacionan las metas y las tareas del sistema en conjunto y las metas y tareas asignadas a los agentes en específico dentro del sistema multiagente (SMA).

Modelo de agente: este modelo contiene una detallada descripción de cada agente y el papel que desempeña dentro del SMA. Esta descripción provee una vista interna incluyendo las metas de los agentes y los servicios que ellos proveen. Esto contrasta con la perspectiva externa proveída por el modelo de Organización.

Modelo del Dominio (información): este modelo actúa como un repositorio de información (entidades y relaciones) concernientes al dominio del problema.

Modelo de interacción: este modelo captura la manera en que los agentes se comunican con otros agentes. Especifica las interacciones de las perspectivas desde el alto nivel hasta el bajo nivel. Aquí se definen el protocolo y la asociación que hay entre los agentes del sistema. Esto se realiza en términos de los actos comunicativos: Semántica bien definida, lenguaje de contenido y ontología.

Comentarios

Esta metodología no da una definición de agente, solo lista una serie de características que deben de tener los agentes. Lo que se traduce en una desventaja, porque si los agentes que se van a utilizar no cubre esas características no se puede utilizar la metodología. Otra de las desventajas es que no considera agentes móviles.

Una ventaja que se le puede observar es la de utilizar los diagramas de UML para la representación gráfica. Es una metodología mejor desarrollada.

Para poder utilizar esta metodología se necesita de un permiso especial por parte de los autores, lo cual se convierte en una desventaja.

2.1.2.4 Zeus [24]

La metodología ZEUS propone un desarrollo en cuatro etapas [31]: el análisis del dominio, el diseño de los agentes, la realización de los agentes y la de soporte en tiempo de ejecución. Las etapas anteriores se basan en el uso de los roles para analizar el dominio y en su asignación de agentes.

Los resultados a producir a lo largo del proceso de desarrollo son los siguientes:

- Análisis del dominio. El propósito de esta parte inicial del análisis es modelar y entender el problema. La metodología Zeus no prescribe ninguna técnica en particular para analizar el problema dejando a los desarrolladores utilizar cualquiera de sus técnicas favoritos, tales como los casos de uso; la técnica recomendado en el documento es el "Modelado de roles". Esta técnica se orienta a obtener el dominio de roles. El modelado de roles se compone de diagramas UML de clases para representar roles, diagramas UML de colaboración para indicar qué mensajes se intercambian

y fichas para describir los roles. Zeus no ofrece actualmente ningún software de soporte para esta etapa.

- Diseño del agente. Al inicio de esta etapa el desarrollador debería de conocer que agentes estarán presentes y que responsabilidades deberán cumplir. Desde esta etapa se incluye la traducción de los roles de responsabilidades dentro del nivel de agentes los problemas que ellos representan, y derivando en soluciones apropiadas. El proceso de diseño involucra pericia, conociendo cuando y como reusar y adaptar soluciones probadas existentes. Como este conocimiento es difícil de acumular, cada uno de los modelos de rol viene con un caso de estudio asociado que describe el razonamiento detrás de un diseño para una aplicación de ejemplo. Esta etapa consiste en determinar qué necesita cada agente para poder desempeñar su cometido. Esto incluye revisar las tecnologías y disciplinas relacionadas con el diseño de agentes. No hay software de Zeus que soporte este estado.

- Implementación de los agentes. En esta etapa se realiza el trabajo de implementación de agentes tomando en cuenta el diseño conceptual creado durante las etapas anteriores. Este proceso consiste de varios estados que son acoplados por niveles de abstracción que existen dentro de un agente Zeus. Para este punto es donde Zeus empieza a ofrecer software, proveyendo una herramienta de generación de agentes a través del cual los diseños pueden ser ingresados, y entonces ser usados para generar código Java para los agentes.

Comentarios

La metodología Zeus es muy pobre metodológicamente pues no tiene técnicas propias; la metodología que sugiere para utilizar es la de "Modelado de roles", la cual no profundiza mucho y deja algunas cosas ambiguas. La herramienta que ofrece es útil en las ultimas dos etapas, pero la herramienta depende de la metodología y esto no es bueno para una metodología pues las herramientas y las metodologías deben de ser independientes.

2.1.2.5 GAIA^2 [25]

GAIA pretende llevar al analista sistemáticamente desde una declaración de requerimientos a un diseño que es suficientemente detallado que pueda ser implementado directamente. Es una metodología para el diseño de sistemas basados en agentes.

2. El nombre proviene de la hipótesis formulada por James Lovelock, al efecto de que todos los organismos en la biosfera de la tierra pueden ser vistos como actuando en conjunto para regular el ambiente.

En GAIA se entiende que el objetivo del análisis es conseguir comprender el sistema y su estructura sin referenciar ningún aspecto de implementación. Esto se consigue a través de la idea de *organización*. Una organización en GAIA es una colección de roles, los cuales mantienen ciertas relaciones con otros y toman parte en patrones institucionalizados de interacción con otros roles. Los roles agrupan cuatro aspectos: responsabilidades del agente, los recursos que se le permite utilizar, las tareas asociadas e interacciones. GAIA propone trabajar inicialmente con un análisis a alto nivel. En este análisis se usan dos modelos, *el modelo de roles* para identificar los roles clave en el sistema junto con sus propiedades definitorias y el *modelo de interacciones* que define las interacciones mediante una referencia a un modelo institucionalizado de intercambio de mensajes, como el FIPA-Request. Tras esta etapa, se entraría en lo que GAIA considera diseño a alto nivel. El objetivo de este diseño es generar tres modelos: el *modelo de agentes* que define los tipos de agente que existen, cuántas instancias de cada tipo y qué papeles juega cada agente, el m*odelo de servicios* que identifica los *servicios* (funciones del agente) asociados a cada rol, y un *Modelo de conocidos*, que define los enlaces de comunicaciones que existen entre los agentes.

Los conceptos que utiliza para el análisis del sistema los divide en dos: Abstractos y concretos. Los conceptos abstractos se utilizan para la conceptualización del problema; estos no necesariamente tienen una implementación directa en el sistema. Y los conceptos concretos que son aquellos que se utilizan en la etapa de diseño y se implementan directamente.

A partir de aquí se aplicarían técnicas clásicas de diseño orientado a objetos. Sin embargo, esto queda fuera del ámbito de GAIA. Esta metodología sólo busca especificar cómo una sociedad de agentes colabora para alcanzar los objetivos del sistema, y qué se requiere de cada uno para lograr esto último.

Comentarios

Esta metodología se queda en un nivel de abstracción alto, según los autores esto se debe a que se busca que el análisis y diseño sea independiente del tipo de agente y de la herramienta utilizada. Tomando en consideración lo anterior no es una metodología que se pueda utilizar en el análisis y diseño de un sistema multiagente.

Es uno de los primeros intentos de estandarizar el modelado de sistemas muliagente por lo consiguiente tiene algunas deficiencias, y no ha llegado a ser una metodología en forma.

2.1.2.6 Metodología INGENIAS [26]

El método de desarrollo de SMA propuesto en INGENIAS concibe el SMA como la representación computacional de un conjunto de modelos. Cada uno de estos modelos muestra una visión parcial del SMA: los agentes que lo componen, las

interacciones que existen entre ellos, cómo se organizan para proporcionar la funcionalidad del sistema, qué información es relevante en el dominio y cómo es el entorno en el que se ubica el sistema a desarrollar.

Para especificar cómo tienen que ser estos modelos se definen meta-modelos. Un meta-modelo es una representación de los tipos de entidades que pueden existir en un modelo, sus relaciones y restricciones de aplicación. Los meta-modelos que se describen aquí son una evolución del trabajo realizado en MESSAGE. En MESSAGE se propusieron meta-modelos para representar agentes, organizaciones, el dominio, interacciones, tareas y objetivos.

INGENIA propone que el trabajo de MESSAGE es mejorable en cuatro aspectos: la integración de los meta-modelos con las prácticas de ingeniería, un mayor nivel de detalle en los meta-modelos, una mayor cohesión entre los meta-modelos y representación del entorno del sistema.

Para lograr estas mejoras, ha sido necesario rediseñar todos los meta-modelos para resaltar aspectos comunes entre los diferentes aspectos modelados, incluir el nivel de detalle requerido en la etapa de diseño del sistema y probar los resultados en casos de estudio para determinar cómo debía ser la integración con un proceso de desarrollo. El motivo de este cambio es que, según la experiencia de MESSAGE, la información proporcionada por la vista del meta-modelo del dominio podía ser expresada dentro de otros sin perjuicio aparente. Sin embargo, no era posible expresar cuál era la naturaleza del entorno ni cómo se interactuaba con él.

A continuación se hace una breve descripción de los Meta-modelos

- **Meta-modelo del agente**. Describe agentes particulares y los estados mentales en que se encontrarán a lo largo de su vida, excluyendo las interacciones con otros agentes. En general este modelo se encarga de la funcionalidad del agente y el diseño de su control.
- **Meta-modelo de tareas y objetivos**. Se usa para asociar el estado mental del agente con las tareas que ejecuta. Define las acciones identificadas en los modelos de organización, interacciones o de agentes y como afectan estas acciones a sus responsables.
- **Meta-modelo de organización**. Define como se agrupan los agentes, la funcionalidad del sistema y qué restricciones hay que imponer sobre el comportamiento de los agentes. El modelo de organización también contribuye al modelo de tareas y objetivos identificando las tareas relevantes para la organización así como los objetivos que se persiguen globalmente.
- **Meta-modelo de interacción**. Detallar como se coordina y comunican los agentes. Además de analizar y detallar las interacciones con usuarios y otros sistemas.

- **Meta-modelo de entorno**. El propósito no es generar representaciones del mundo en el que se ubica el SMA, se trata de discretizar el entorno utilizando un conjunto finito de variables observables.

Comentarios

Esta metodología es una de las más completas para el desarrollo de sistemas multiagente; además de que cuenta con soporte de una herramienta para el análisis y diseño.

Uno de los inconveniente que tiene esta metodología es la de no ser muy flexible y se aplica mejor a sistemas muy grandes. El otro inconveniente es que no tiene un trabajo formal.

2.1.3 Extensiones de metodologías de ingeniería del conocimiento

Uno de los intentos de las metodologías de ingeniería de agentes, es la de desarrollar una nueva metodología que se pueda aplicar en el área de agentes, a partir de metodologías utilizadas en Inteligencia Artificial (IA) o de Sistemas Basados en conocimiento (KBS).

2.1.3.1 Ventajas del enfoque [27]

Las metodologías de ingeniería del conocimiento pueden proporcionar una buena base para modelar sistemas multiagente ya que estas metodologías tratan del desarrollo de sistemas basados en conocimiento. Dado que los agentes tienen características cognitivas, estas metodologías pueden proporcionar las técnicas de modelado necesaria para el modelado del conocimiento de los agentes.

La extensión de metodologías de conocimiento puede aprovechar la experiencia adquirida en dichas metodologías. Además se pueden reutilizar las bibliotecas de métodos de resolución de problemas y ontologías así como las herramientas desarrolladas en estas metodologías.

Aunque estas metodologías han sido empleadas en ámbitos más restringidos que las orientadas a objetos, también han sido aplicadas con éxito en la industria.

2.1.3.2 Desventajas del enfoque[27]

La mayor parte de los problemas planteados en las metodologías de ingeniería del conocimiento también se dan, obviamente, en el diseño de sistemas multiagente: adquisición del conocimiento, modelado del conocimiento, representación y reutilización. Sin embargo, estas metodologías conciben un sistema basado en conocimiento como un sistema centralizado.

2.1.4 Metodologías existentes

2.1.4.1 Metodología MAS-CommonKADS [27]

Esta metodología extiende CommonKADS aplicando metodologías orientadas a objetos y metodologías de diseño de protocolos para su aplicación a la producción de SMA. La metodología CommonKADS gira en torno del modelo de experiencias y está pensada para desarrollar sistemas expertos que interactúan con el usuario.

La metodología inicia con una fase de conceptualización que es una fase informal destinada a colectar los requerimientos del usuario y obtener una primera descripción del sistema desde el punto de vista del usuario. Para este propósito se utilizan técnicas de OOSE[3].

La metodología incluye los siguientes modelos:

- **Modelo de Agentes:** describe las características principales de los agentes, incluyendo las capacidades de razonamiento, habilidades (sensores/efectores), servicios, objetivos, entre otras.
- **Modelo de Tarea:** describe las tareas llevadas a cabo por el agente, y la descomposición de tareas.
- **Modelo de Capacidades:** describe el conocimiento que necesita el agente para llevar a cabo las tareas.
- **Modelo de Coordinación:** describe las conversaciones entre los agentes, esto es, sus interacciones, protocolos y capacidades requeridas.
- **Modelo de Organización:** describe la organización en la cual el sistema Multiagente será introducido y la organización de la sociedad de agentes.
- **Modelo de Comunicación:** detalla las interacciones humano-agente, y los factores humanos para desarrollar estas interfaces.
- **Modelo de Diseño:** reúne los modelos previos y se subdivide en tres submodelos:
    - **Diseño de la aplicación:** composición o descomposición de los agentes del análisis
    - **Diseño de la arquitectura:** diseño de los aspectos relevantes de la red de agentes.
    - **Diseño de la plataforma:** selección de la plataforma de desarrollo de los agentes para cada arquitectura de agente.

3." Object Oriented Software Engineering (OOSE). Metodología que proporciona un soporte para el diseño creativo de productos de software". Ivar Jacobson

Comentarios

El principal inconveniente de esta metodología es el nivel de detalle que se debe de alcanzar, el cual es complicado de lograr sin una herramienta de apoyo. En los documentos encontrados se menciona una herramienta para el apoyo de esta metodología pero no se puede descargar. En la página www.gsi.dit.upm.es solo están disponibles algunas arquitecturas y *frame works* para el desarrollo de agentes.

De las ventajas que se pueden observar es que es la primera metodología que se integra al ciclo de vida del software, además de que es una metodología con un amplio nivel de detalle y es independiente de la arquitectura del agente. Esta metodología es una de las mas utilizadas por los desarrolladores de sistemas multiagente.

Además es producto de un trabajo formal, pues fue desarrollada y discutida en una tesis doctoral.

2.1.4.2 CoMoMAS[32]

CoMoMAS (Contribution to Knowledge Acquisition and Modelling in a Multi-Agent Framework) es una extensión de la metodología CommonKADS. Propone una extensión de CommonKADS para modelar sistemas multiagente, a través del desarrollo de los siguientes modelos:

Modelo de Agente: es el modelo central de la metodología y define la arquitectura del agente y el conocimiento del mismo, que se clasifica en social, cooperativo, cognitivo, reactivo y de control.

Modelo de la experiencia: describe las competencias cognitivas y reactivas del agente. Distingue entre conocimiento de tareas, de resolución de problemas y reactivo. El conocimiento de las tareas contiene la descomposición de las tareas, descrita en el modelo de tareas. El conocimiento de resolución de problemas describe los métodos de resolución de problemas (Problem Solving Methods, PSM) y las estrategias para seleccionarlos. El conocimiento reactivo describe los procedimientos para responder a un estímulo.

Modelo de tareas: describe la descomposición de las tareas indicando si son resolubles por el usuario o por un agente.

Modelo de cooperación: describe la cooperación entre varios agentes. Se descompone en métodos de resolución de conflictos (Métodos de negociación y cooperación) y conocimiento de cooperación (primitivas, protocolos y terminología de interacción).

Modelo del sistema: representa la organización de los agentes y describe la arquitectura del sistema multiagente y de sus a gentes.

Modelo de diseño: define cómo pasar de los modelos previos a código ejecutable. Recoge los requisitos funcionales y no funcionales de la aplicación, así como las guías de diseño (Plataforma y lenguaje de implementación) y la historia del diseño.

Comentarios

CoMoMAS es una metodología que no profundiza demasiado en el análisis de los sistemas multiagente. Se centra en el modelo de la experiencia mientras que las demás (organización, interacciones, identificación de agentes, etcétera) apenas se desarrollan. En cuanto al modelo de experiencia se distingue explícitamente el conocimiento reactivo.

2.2 Comparación de metodologías

A nivel general de metodología se pueden mencionar los siguientes aspectos importantes.

De las metodologías aquí presentadas sólo dos son el resultado de trabajos de tesis: CoMoMAS y MAS-CommonKADS; las demás han sido desarrolladas en artículos esporádicos y en algunos casos no tienen continuidad.

GAIA, MaSE y MAS-CommonKADS optan por una cierta independencia y la posibilidad de selección de la arquitectura de agente utilizada.

Conceptualmente las metodologías, introducen términos en los cuales es uniforme la definición que se da. Conceptos tales como Agente, tarea, rol, organización interacción, metas u objetivos. Estos conceptos son unos de los mas importantes dentro del paradigma de agentes.

En la metodología BDI un objetivo es un estado al cual debe de llegar un agente; un agente es una entidad que tiene deseos y creencias; una tarea es una acción a realizar para llegar a un estado determinado (objetivo); un plan es el conjunto de pasos a seguir para lograr un objetivo. Una meta es una sentencia que generalmente se expresa como una posible funcionalidad del sistema. Las tareas definen los detalles de como se logra o alcanza cada una de las metas planteadas.

En MaSE los agentes son tratados como una abstracción. Según esto un agente tiene las siguientes características: son entidades que están distribuidas, comparten información, conformando sistemas multiagente. Un rol en esta metodología es una función de una entidad que debe cumplir obligatoriamente.

MaSE tiene una herramienta de soporte denominada AgentTool, la cual ayuda al analista a utilizar la metodología eficazmente.

MaSE Utiliza diagramas de UML para la representación gráfica de las interacciones entre agentes.

La definición de agente de INGENIAS esta influenciada por la inteligencia artificial. Los define en dos partes: en responsabilidades y comportamientos.

INGENIAS mejora cuatro aspectos de la metodología Message: la integración de los meta-modelos con las prácticas de ingeniería, un mayor nivel de detalle en los meta-modelos, una mayor cohesión de los meta-modelos y representación del entorno.

En INGENIAS se utilizan los meta-modelos para la representación computacional de los sistemas multiagente.

INGENIAS no tiene una herramienta de soporte específica, pues se puede utilizar cualquier herramienta con la cual se puedan generar meta-modelos.

En Message un agente es definido como una unidad atómica autónoma capaz de desempeñar alguna función. Una organización la definen como un conjunto de agentes trabajando juntos para un fin común. Un rol es el papel desempeñado por el agente separado lógicamente del concepto de agente. Una tarea es definida como una unidad de nivel de conocimiento de una actividad con un solo ejecutante principal [29].

Message utiliza los diagramas de UML donde sea apropiado, a los cuales le va agregando conceptos, entidades y relaciones necesarias para el modelado orientado a agentes.

En GAIA un agente se define como un sistema computacional de grano grueso que hace uso de recursos computacionales, es responsable de alcanzar y mantener ciertos objetivos que caracterizan su conducta; además un agente puede ejecutar acciones que afecten a los objetos del entorno.

Un rol es definido como una descripción abstracta de una función esperada de la entidad, en este caso un agente.

Para la metodología CoMoMAS los agentes son entidades inteligentes y autónomos que actúan sobre si mismos usando sus conocimientos individuales.

Como se puede observar las metodologías, definen los conceptos fundamentales de la tecnología de agentes desde diferentes puntos de vista, lo cual hace que las metodologías propuestas no sean de carácter general.

Con respecto a las características con que cuentan los agentes a continuación se muestra en la Tabla 2.1 donde se resumen aquellas que abarca cada metodología

Características	Zeus	MaSE	Message	Gaia	Ingenias	BDI	CoMoMAS	MAS-CommonKADS
Autonomía	√	√	√	√	√	√	√	√
Racionalidad	√	√	√		√		√	
Conocimiento		√	√	√	√	√	√	√
Movilidad								
Pro-actividad	√				√	√		√
Sociabilidad	√	√	√	√	√	√	√	√
Reactividad	√	√	√	√		√	√	√

Tabla 2.1. Características de las metodologías

Como se puede observar en la tabla anterior, ninguna de las metodologías para el desarrollo de agentes aquí mencionadas toman en cuenta la movilidad.

La movilidad es una de las características necesarias para la solución del problema mencionado al inicio de este trabajo.

Etapas del desarrollo de sistemas consideradas

De las metodologías aquí estudiadas la mayoría se quedan en las etapas de análisis y diseño; algunas dan guías de cómo realizar las siguientes fases del ciclo de vida de un sistema, pero sin definir una técnica o guía en concreto; esto se ve reflejado en la Tabla 2.2.

GAIA en la descripción de su metodología da unas guías para la etapa de implementación; al igual que MaSE, MESSAGE, e INGENIAS.

Metodología	Vistas en el análisis	Vistas en el diseño
Zeus	Análisis del dominio	Diseño de los agentes
MaSE	Captura de objetivos Aplicación casos de uso Transformación de roles	Identificar los tipos de agentes Identificar las interacciones entre agentes Definición de protocolos de coordinación. Diseño del sistema
Message	Modelo de organización Modelo de objetivos/tareas Modelo de agente Modelo de interacción Modelo de dominio	Refinamiento de modelos Definición de la arquitectura
Gaia	Modelo de roles Modelo de interacción	Modelo de agente Modelo de servicios Modelo de conocimiento
Ingenias	Meta-modelo del agente Meta-modelo de tareas y objetivos Meta-modelo de organización	Meta-modelo de interacción
BDI	Modelo de agentes Modelo de interacción Modelo de creencias Modelo de objetivos Modelo de planes	
CoMoMAS	Modelo de agente Modelo de la experiencia Modelo de cooperación Modelo del sistema	Modelo de diseño
MAS-CommonKADS	Fase de conceptuación Modelo de organización Modelo de agente Modelo de tareas Modelo de la experiencia Modelo de coordinación	Diseño de red Diseño de agentes Diseño de plataforma

Tabla 2.2. Fases que comprenden las metodologías

2.3 Elección de la metodología

De acuerdo con el panorama anterior la metodología que se decidió utilizar es la MAS-CommonKADS por las siguientes razones:

1. Es una de las metodologías más utilizadas por los desarrolladores de sistemas multiagente.
2. Ha sido aplicada con éxito en varios proyectos de investigación, en diferentes campos, tales como la gestión inteligente de la red (proyecto CICYT TIC94-0139 PROTEGER, Sistema Multiagente para gestión de Red y Servicios) y el desarrollo de aplicaciones con sistemas híbridos (proyecto ESPRIT-9119 MIX, Integración Modular de Sistemas Basados en Conocimiento Simbólicos y Conexiones).

3. Es una metodología flexible, ya que tiene diferentes guías dependiendo del tamaño del sistema. Tiene guías para sistemas pequeños, medianos y sistemas multiagente. Lo cual es útil para el desarrollo del prototipo y la continuación del sistema.
4. Permite elegir la herramienta de desarrollo y la plataforma multiagente a utilizar. Esta característica es importante pues se puede utilizar la herramienta selecciona en otro trabajo de investigación desarrollado.
5. Es independiente de la arquitectura que se va a utilizar en la solución del problema. Es de gran importancia este punto puesto que se van a utilizar diferentes arquitecturas de agentes para la solución del problema.
6. Es una metodología resultado de una tesis de doctorado. Además de ser el resultado de una investigación bien definida.
7. Da completa libertad de elección del método para la representación gráfica de los modelos. También se considero esta característica porque a nivel institucional en el Instituto Mexicano del Petróleo se tiene planeado utilizar el UML como herramienta de modelado de sistemas computacionales. Y en este proyecto se utilizará una extensión denominada AUML.
8. El nivel de detalle puede ser tan profundo como sea necesario para el desarrollo del sistema.
9. La metodología es una de las mas recientes por lo que contempla y corrige los problemas que tienen los primeros trabajos.

Capítulo 3
Análisis y diseño del sistema multiagente

Introducción

En este capítulo se muestra el modelado y diseño del sistema utilizando la metodología MAS-CommonKADS. Primero se muestran los casos de uso identificados, a continuación se presentan los diferentes modelos desarrollados que establece la metodología; tales como: Modelo de tareas, Modelo de agentes, Modelo de coordinación, Modelo de organización, Modelo de experiencia y Modelo de diseño.

3.1 Modelo de la organización

La parte correspondiente al modelo de la organización se describió en la sección Entorno del problema.

En las Figuras 3.1 y 3.2 se muestra como se lleva a cabo actualmente el proceso en forma gráfica.

Figura 3.1. Proceso actual del servicio de soporte técnico

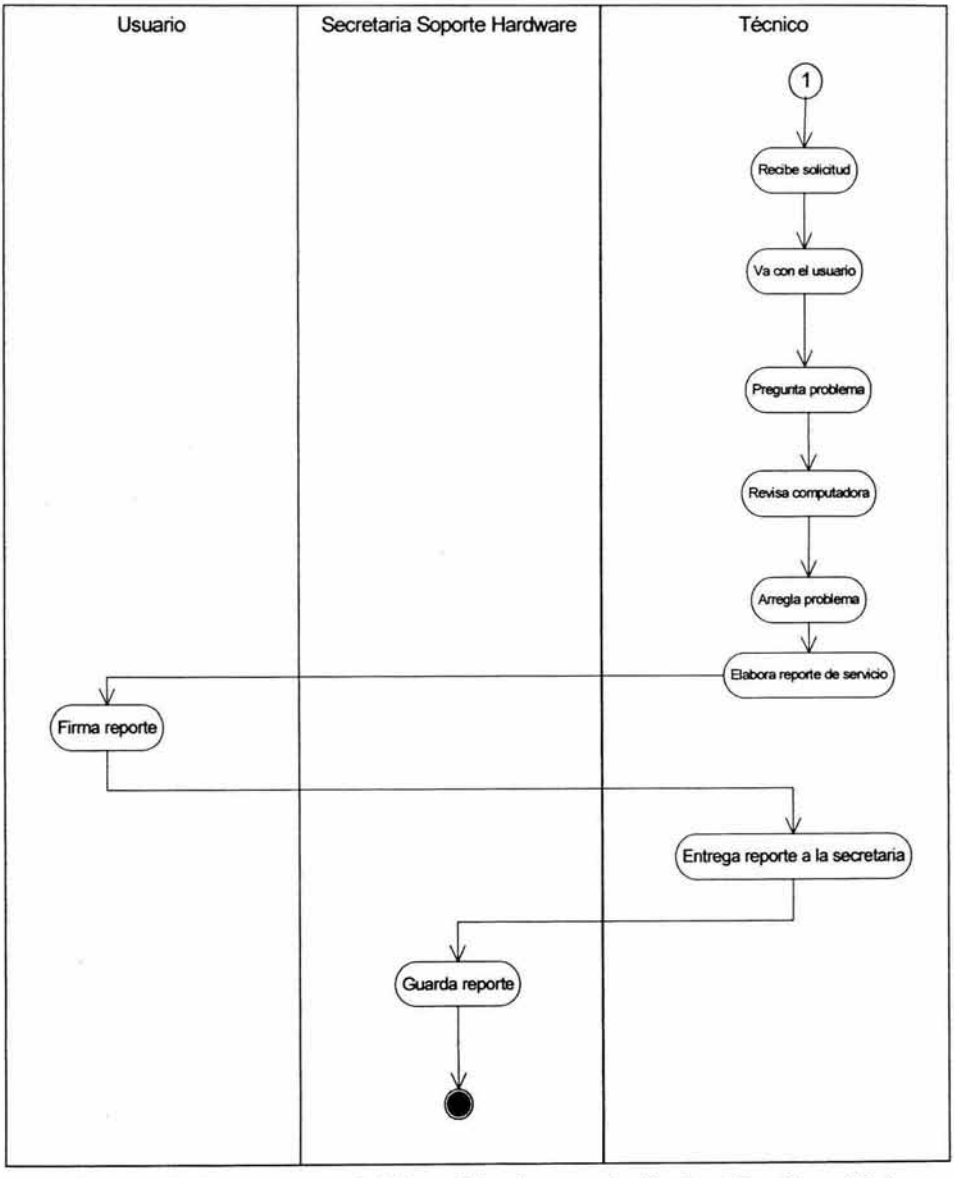

Figura 3.2. Proceso actual del servicio de soporte técnico (Continuación)

3.1.1 Solución sugerida

Los problemas que se presenta comúnmente en una computadora, relacionados con el hardware son de configuración; y muy rara vez se presentan problemas físicos; además de que el personal resulta ser insuficiente para atender a los usuarios y el tiempo de traslado a la ubicación de los usuarios es alto.

El proceso es repetitivo y sencillo; basta con detectar el dispositivo con problemas, revisar sus archivos de configuración y en el caso de que falte alguno copiarlo. Si la configuración es correcta se puede deducir que el problema es causado por algún daño físico del dispositivo.

Ayudándonos de las características mencionadas en el párrafo anterior, la solución sugerida es la de utilizar un sistema automatizado que revise periódicamente la configuración de los equipos de cómputo de los usuarios para evitar que se pierdan los archivos de configuración del hardware; y en dado caso de que se pierda el sistema debe ser capaz de detectar el problema y resolverlo, con la condición de que el problema sea de configuración.

Cuando el sistema detecte un problema de configuración del hardware, lo tratará de resolver, y si no logra su objetivo; el sistema dará aviso al técnico para que revise el sistema físicamente.

Además el sistema ayudará a la secretaria a elaborar los documentos necesarios; al técnico a detectar problemas en las computadoras; al usuario a realizar una solicitud de servicio sin importar a que área deba ser dirigido; y contendrá un módulo para la administración del mismo.

Otra de las características que debe de tener el sistema es la de interactuar con el sistema encargado de instalar software para dar un mejor servicio.

Para lograr lo anterior el sistema será desarrollado bajo el paradigma denominado Agentes, los cuales tienen las características de poder realizar actividades repetitivas, además pueden aprender a resolver nuevos problemas, también tienen la capacidad de comunicarse con otros agentes para lograr sus objetivos.

Además el paradigma de agentes. es adecuado para el análisis y diseño de sistemas distribuidos, como el que se plantea en este trabajo.

Los diagramas de las Figuras 3.3 y 3.4 muestran cómo será el desarrollo de las actividades con la implantación del sistema

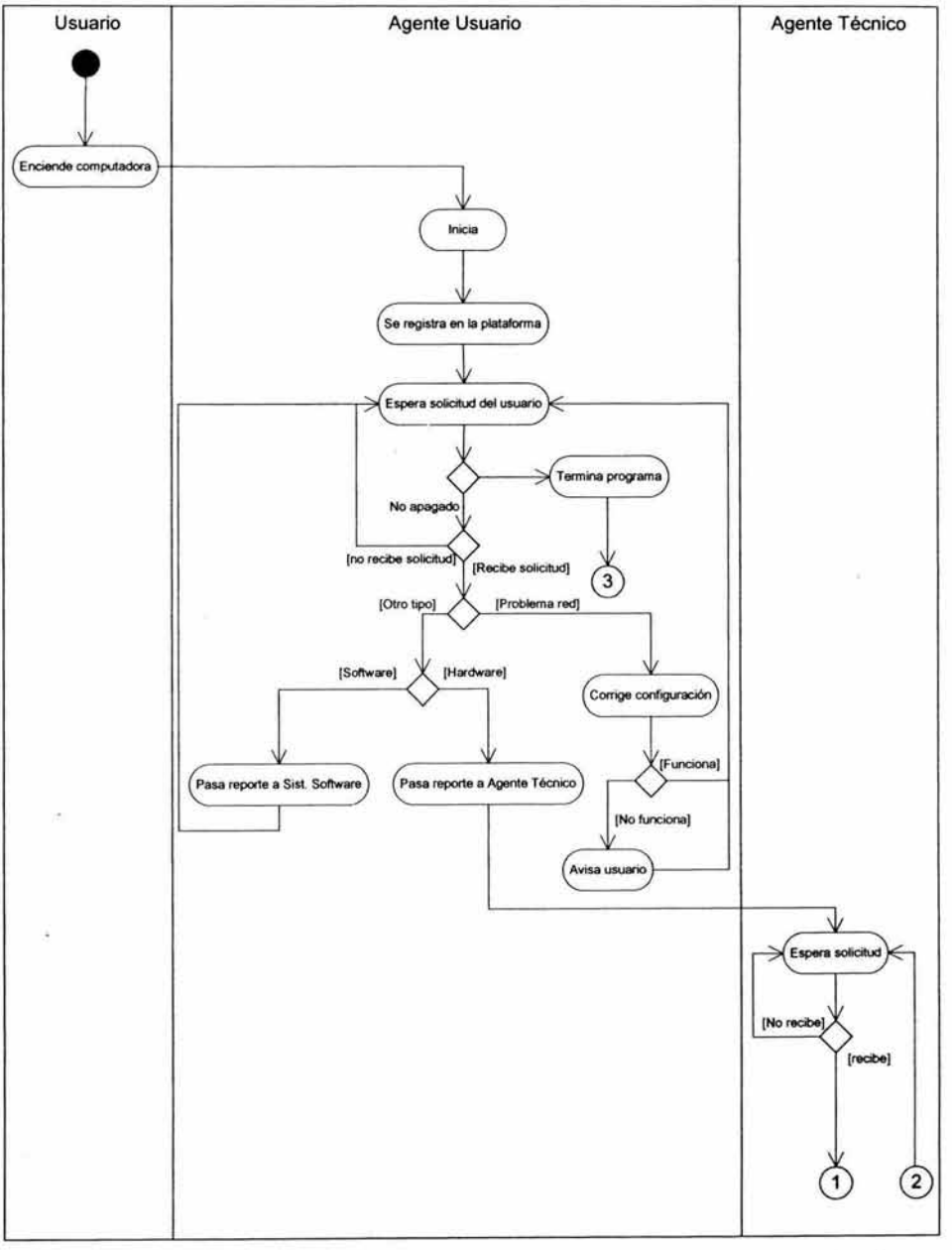

Figura 3.3. Proceso con el sistema implantado del servicio de soporte técnico

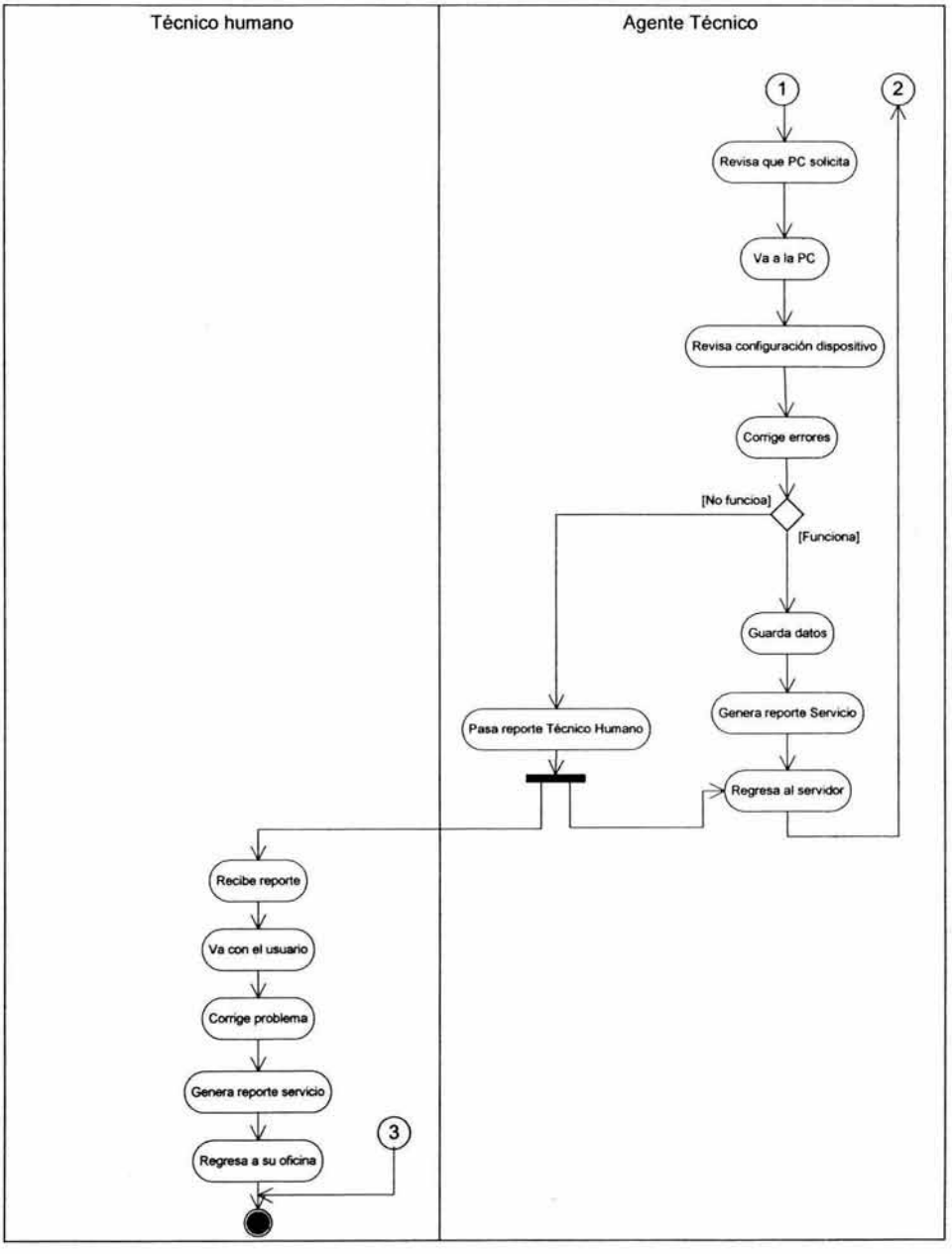

Figura 3.4. Proceso con el sistema implantado del servicio de soporte técnico (continuación)

En las figuras se observa que la intervención de los técnicos se reduce a resolver problemas físicos de los equipos, además de que la carga de trabajo también se reduce para ellos.

Las mejoras esperadas con la implementación del sistema se mencionan a continuación:

- El usuario se encargará de realizar su reporte sin importar que área sea la encargada de prestar el servicio. El sistema va a tratar de resolver el problema y en el caso de no poder hacerlo, será capaz de dar el reporte al área correspondiente.
- El técnico solo se desplazará al área donde se encuentra el usuario, en el caso de que el sistema no pueda resolver el problema.
- El tiempo de respuesta será inmediato.
- El número de personas necesarias para la atención de los usuarios también se reducirá.

3.1.2 Conceptuación

La fase de conceptuación consiste en obtener un panorama general del problema que se va a resolver y elaborar una primera aproximación del sistema.

3.1.2.1 Identificación de los actores

En este problema, podemos identificar varios actores que interactuarán con el sistema: los usuarios que solicita el servicio de revisión de su equipo, ya sea directamente con la interfaz de usuario o por medio de un correo electrónico; el técnico encargado de dar el mantenimiento, el cual puede solicitar un reporte del estado actual de un equipo; la secretaria, encargada de elaborar los reportes y estadísticas de los servicios que se realizaran durante un determinado periodo de tiempo; la base de datos donde se almacenan los reportes y estadísticas de los servicios; administrador encargado de mantener el sistema funcionando.

3.1.2.2 Descripción de los actores

A continuación se describen los actores identificados

Actor Usuario
descripción
Persona que tiene una computadora, la cual necesita que se le de servicio de mantenimiento; puede solicitar el servicio por medio de la interfaz de usuario, enviando un e-mail al técnico o por vía telefónica.

Actor Técnico

descripción

Es el encargado de dar mantenimiento a los equipos y de corregir los problemas identificados.

Actor Secretaria

descripción

Es la persona encargada de elaborar los reportes generales y estadísticas de los servicios realizados durante un periodo establecido previamente; además de archivarlos.

Actor Administrador

descripción

Persona encargada de darle mantenimiento al sistema, como por ejemplo cuando la base de datos no funciona correctamente.

3.1.3 Identificación de los casos de uso

A continuación se muestran los casos de uso identificados para cada actor

- Usuario: Solicita servicio, llena formulario de opinión, recibir formulario, envía formulario contestado.
- Técnico: Configura hardware, recibe solicitud de servicio, elabora reporte de servicio, revisa un equipo, consulta base de datos, carga nuevos controladores, envía formulario.
- Secretaria: Almacena reportes en la base de datos, elabora reportes generales, elaborar estadísticas, recibe formulario contestado.
- Administrador: Revisa el funcionamiento del sistema y de las bases de datos, corrige problemas del sistema.

Casos de uso del Usuario

Caso de uso Solicita servicio

resumen

El usuario solicita el servicio de mantenimiento o de corrección de la configuración de algún dispositivo; y recibe la confirmación de solicitud recibida.

actores

Usuario, Técnico.

precondiciones

El usuario ha detectado una falla en su equipo o ha conectado algún dispositivo nuevo que necesite de su configuración.

descripción

El usuario solicita el servicio de mantenimiento, esto lo puede hacer de diferentes maneras: enviando un correo electrónico, haciendo la

petición directamente mediante el sistema o hablando por teléfono al área de soporte técnico.

excepciones

Ninguna.

poscondiciones

Una vez realizado este caso de uso, al usuario se le manda una confirmación; y dependiendo de la situación se lleva a cabo la acción solicitada.

Caso de uso Enviar formulario contestado

resumen

El usuario envía contestado el formulario de opinión.

actores

Usuario, secretaria.

precondiciones

El usuario llenó el formulario.

descripción

Después de dar el servicio al usuario, él tiene la opción de dar su opinión acerca del servicio recibido.

excepciones

La secretaria no está disponible. Cuando se presenta esta situación, se mandará posteriormente el formulario.

poscondiciones

El usuario recibe una confirmación de recibido.

Caso de uso Llenar formulario de opinión

resumen

Dar a conocer la opinión que se tiene a cerca del servicio que se le está prestando.

actores

Usuario.

precondiciones

El usuario ha recibido el formulario de opinión.

descripción

El área de soporte le envía un formulario al usuario para que lo llene con la opinión que tiene sobre el servicio prestado.

excepciones

La computadora está apagada: para este caso la acción llevada a cabo es la de enviar después el formulario.

poscondiciones

El formulario fue llenado correctamente.

Caso de uso Recibir formulario

resumen

La secretaria envía formulario para que el usuario exprese su opinión.

actores

Secretaria, Usuario.

precondiciones

El técnico terminó de dar mantenimiento a la computadora del usuario.

descripción

Una vez que el técnico terminó de darle mantenimiento a la computadora, la secretaria envía el formulario al usuario para conocer su opinión del servicio, y poder mejorarlo.

excepciones

La secretaria no está disponible.

El servicio de red no está funcionando correctamente.

poscondiciones

El usuario recibió el formulario de opinión.

En la Figura 3.5 se muestran los casos de uso del Usuario en forma gráfica.

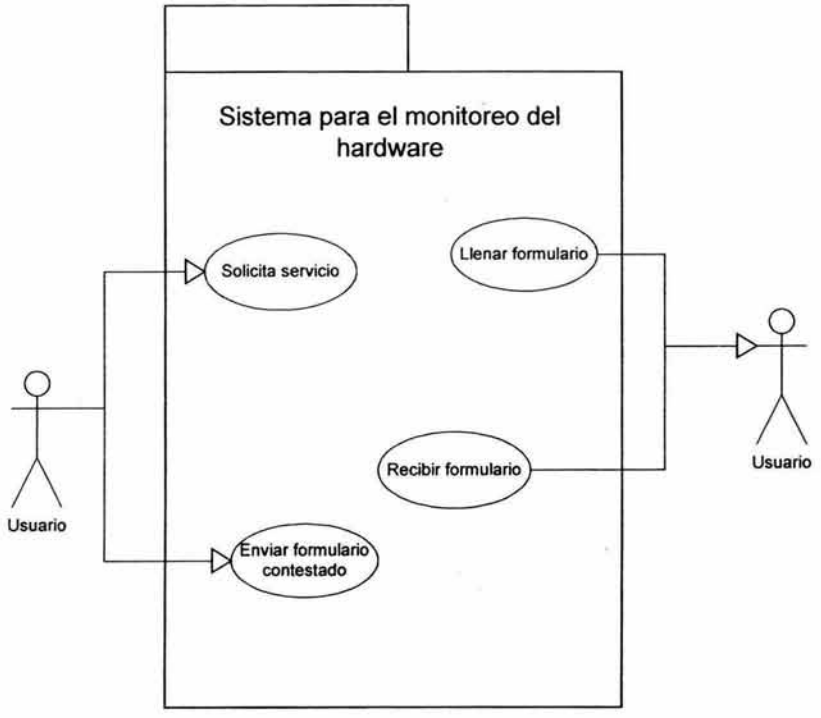

Figura 3.5 Casos de uso del usuario

Casos de uso del Técnico

Caso de uso Configurar Hardware

resumen

El técnico revisa y/o configura los dispositivos conectados al equipo para que funcionen correctamente.

actores

Técnico.

precondiciones

Hay una solicitud de servicio.

descripción

El técnico puede realizar revisiones del estado de la computadora para detectar algún problema y solucionarlo. Esta acción la lleva a cabo cuando el sistema no puede corregir el problema presentado por el equipo debido a que no es un problema de configuración sino un problema físico.

excepciones

Ninguna.

poscondiciones

Los dispositivos conectados a la computadora funcionan correctamente.

Caso de uso Recibir solicitud de servicio

resumen

El técnico recibe una solicitud de servicio cuando el sistema no puede corregir el problema presentado por la computadora.

actores

Usuario, Técnico.

precondiciones

El problema no puede ser solucionado por el sistema sin la intervención del técnico humano.

descripción

Cuando algún dispositivo conectado a la computadora no funciona correctamente el agente técnico tratará de corregir el problema; si no lo logra entonces solicitará al técnico humano que revise el equipo para que solucione el problema.

excepciones

El técnico humano está atendiendo otro problema.

poscondiciones

El técnico corrige el problema.

Caso de uso Elaborar reporte de servicio

resumen

Elaborar reporte donde se describa el servicio prestado

actores

Técnico.

precondiciones
El técnico terminó de dar mantenimiento a la computadora del usuario.
descripción
Este reporte se realiza con el fin de saber que problema tenía la computadora y la solución dada para que el agente adquiera nuevos conocimientos; además de los datos del usuario y del equipo.
excepciones
Ninguna.
poscondiciones
Reporte de servicio elaborado.

Caso de uso Revisar un equipo
resumen
Revisar el equipo para corregir la falla detectada por el sistema.
actores
Técnico.
precondiciones
El técnico recibió una solicitud de servicio.
descripción
El técnico corrige las fallas que no pueden ser solucionadas por el sistema.
excepciones
El sistema operativo no funciona: en este caso el sistema no podrá ser ejecutado y el técnico tendrá que utilizar otra medida para corregir el problema.
poscondiciones
Los dispositivos que tenía problemas funcionan correctamente.

Caso de uso Consultar la base de datos
resumen
Consultar la base de datos para revisar los controladores almacenados en ella.
actores
BDConfiguración, Técnico.
precondiciones
No se encuentra un controlador.
descripción
En ocasiones se pude presentar la situación de no encontrar un determinado controlador y el técnico puede consultar la base de datos para verificar si se encuentre o volverlo a cargar.
excepciones
No hay conexión con la base de datos.
poscondiciones
Están cargados todos los controladores necesarios.

Caso de uso Cargar nuevos controladores

resumen

Cargar a la base de datos los controladores de nuevos dispositivos.

actores

Técnico.

precondiciones

El Instituto Mexicano del Petróleo adquirió nuevos dispositivos para las computadoras .

descripción

Cuando el instituto compra nuevos dispositivos para las computadoras, es necesario cargar los controladores en las base de datos para que el sistema pueda utilizarlos.

excepciones

Ninguna.

poscondiciones

Controladores almacenados en la base de datos.

Caso de uso Enviar formulario

resumen

Enviar formulario de opinión al usuario.

actores

Técnico, Usuario.

precondiciones

El técnico realizó un servicio.

descripción

Después de haber realizado un servicio, el técnico envía un formulario que el usuario tiene que llenar; en el cual se recoge la opinión que tiene el usuario acerca del servicio prestado por el técnico.

excepciones

No funciona el servicio de red.

poscondiciones

El usuario recibe el formulario.

A continuación se muestra un resumen de los casos de uso del técnico. Figura 3.6.

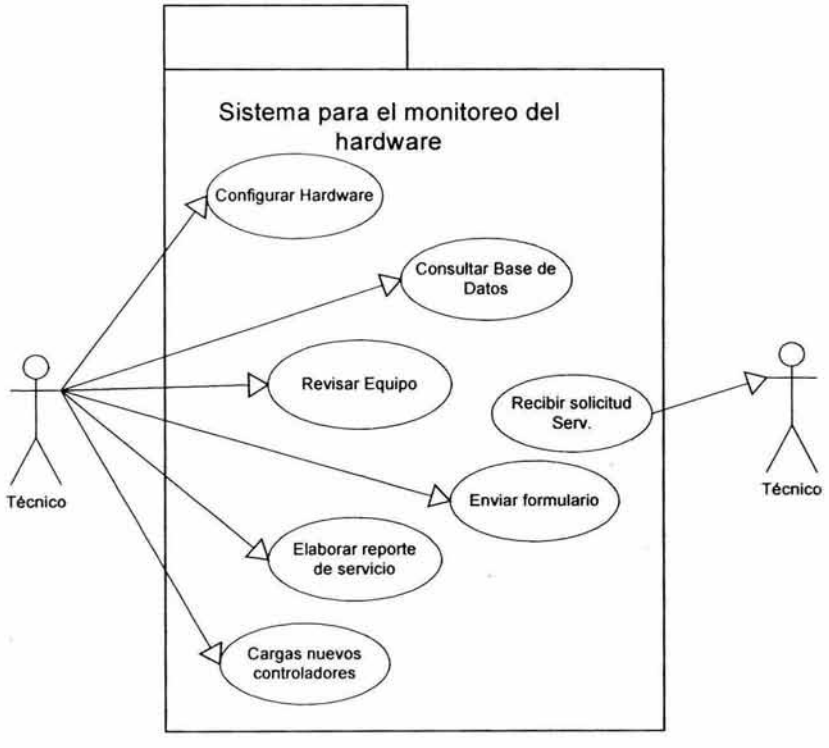

Figura 3.6 Casos de uso del técnico

Casos de uso de la secretaria

Caso de uso Almacenar reportes en la base de datos.

resumen

Guardar los reportes de servicio en la base de datos.

actores

Secretaria, BDAdmon.

precondiciones

La secretaria recibió un reporte de servicio.

descripción

Cuando el técnico realiza un servicio de corrección de configuración, envía un reporte del problema presentado y la solución dada. Este reporte además contiene los datos del usuario y la configuración actual del sistema.

excepciones

La base de datos no está disponible. En este caso se almacenarán temporalmente los reportes en la computadora de la secretaria.

poscondiciones
El reporte está almacenado en la base de datos.

Caso de uso Elaborar reportes Generales
resumen
Hacer un resumen de los servicios realizados a una determinada fecha.
actores
Secretaria, BDAdmon.
precondiciones
Ha transcurrido un determinado periodo de tiempo y existen reportes almacenados en la base de datos con fecha que comprende ese periodo.
descripción
Después de haber transcurrido un lapso de tiempo se deben entregar reportes generales de todos los servicios realizados por los técnicos para poder elaborar estadísticas y llevar un control.
excepciones
No se realizó ningún servicio en ese periodo.
poscondiciones
Reporte general terminado.

Caso de uso Elaborar estadísticas
resumen
Elaborar estadísticas cada determinado periodo.
actores
Secretaria, BDAdmon.
precondiciones
Transcurre un lapso de tiempo.
descripción
Para llevar un control de las actividades del área, se realizan estadísticas; además de que también se pueden tomar decisiones con ellas.
excepciones
Ninguna.
poscondiciones
Estadísticas elaboradas.

Caso de uso Recibir formulario ya contestado del usuario
resumen
El usuario envía su opinión a la secretaria.
actores
Usuario, Secretaria.
precondiciones
El usuario ya llenó el formulario.

descripción

Cuando el usuario termina de llenar el formulario, lo envía a la secretaria para que lo almacene en la base de datos.

excepciones

La computadora de la secretaria no está disponible.

poscondiciones

La secretaria tiene la opinión del usuario.

A continuación se observan los casos de uso de la secretaria en forma gráfica(Figura 3.7)

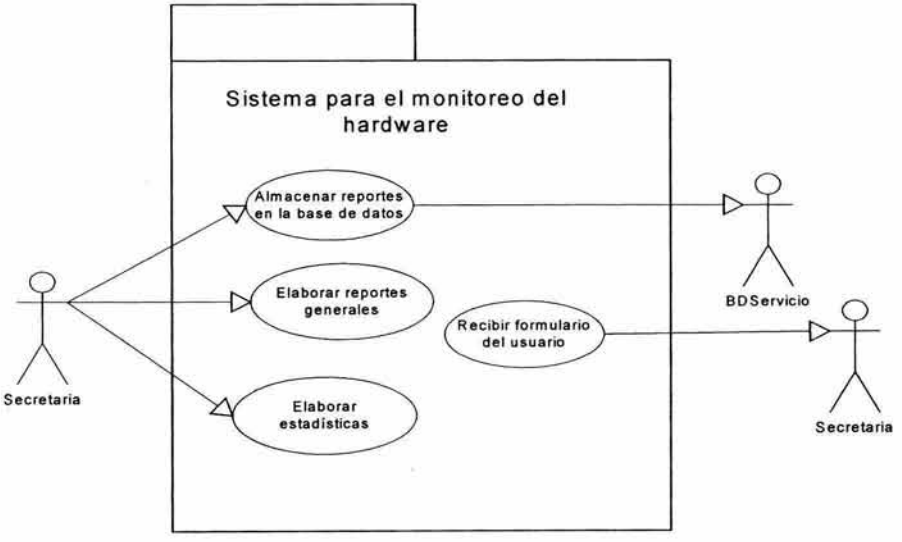

Figura 3.7 Casos de uso de la secretaria

Casos de uso del Administrador

Caso de uso Revisar funcionamiento del sistema

resumen

Detectar posibles fallas que pueda presentar el sistema.

actores

Administrador.

precondiciones

El sistema no funciona debidamente.

descripción

El administrador supervisará el sistema para detectar posibles fallas en el sistema.

excepciones
El sistema no funciona totalmente.
poscondiciones
Sistema funcionando correctamente.

Caso de uso Revisar Bases de datos
resumen
Verificar que las base de datos funcionen correctamente.
actores
Administrador.
precondiciones
Ha transcurrido el tiempo establecido para revisar el funcionamiento de la base de datos.
descripción
Para que la información que contiene la base de datos sea íntegra el administrador hará revisiones periódicas a las bases de datos.
excepciones
El sistema no funciona correctamente.
poscondiciones
Bases de datos funcionando correctamente.

Caso de uso Corregir problemas del sistema
resumen
Corregir las fallas que tenga el sistema.
actores
Administrador.
precondiciones
El sistema no funciona correctamente.
descripción
Cuando el sistema tiene fallas, el administrador será la persona encargada de corregir los problemas que presente.
excepciones
No es posible corregir el problema.
poscondiciones
Funcionamiento del sistema correcto.

La Figura 3.8 muestra los casos de uso del administrador en forma resumida.

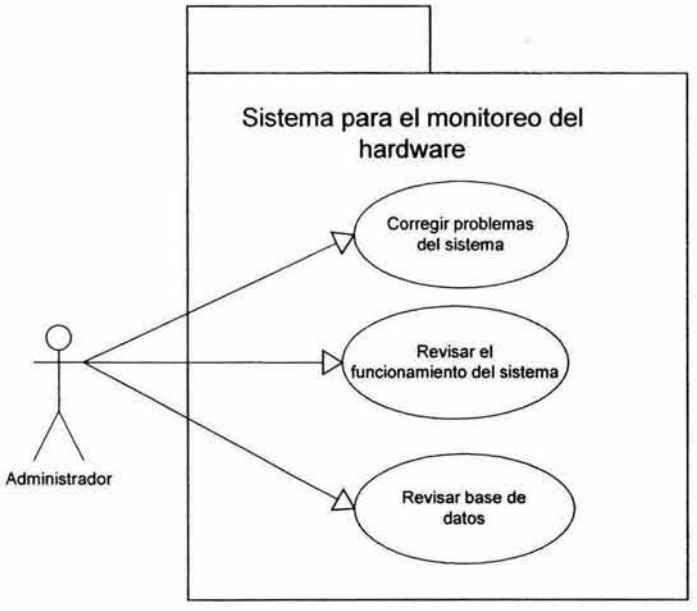

Figura 3.8 Casos de uso del administrador

3.2 Modelo de tareas

Este modelo describe las tareas (objetivos) realizados por los agentes y su descomposición, utilizando plantillas textuales.

Tarea Elaborar reporte general

objetivo

Elaborar resumen de todos los servicios que se prestaron durante un determinado periodo.

descripción

Esta tarea es para elaborar los reportes mensuales, donde se muestra un resumen de los servicios que se prestaron durante un periodo determinado.

entrada

Reporte de los servicios realizados durante el periodo que comprende el reporte general.

salida

Reporte general.

precondición

Ha transcurrido el periodo de tiempo previamente establecido.

supertarea

Ninguna.

subtareas
Imprimir reporte, guardar en la base de datos.
tipo de descomposición
Ninguna.
ingrediente
Reportes de servicio.
descripción
Contiene la información necesaria para elaborar el reporte general; tales como datos del usuario, e información del problema presentado.

Tarea Elaborar estadísticas
objetivo
Elaborar un documento que muestre el comportamiento de las actividades realizadas por el sistema a un determinado tiempo.
descripción
Se elaboran las estadísticas para ver el comportamiento de los servicios. Los datos son obtenidos de los reportes de servicio: fecha, identificador de la computadora.
entrada
Datos de la computadora, fecha en que se prestó el servicio, problema presentado.
salida
Datos de los servicios mostrados en forma tabular.
precondición
Solicitud por parte de la secretaria.
supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninuguna.
ingrediente
Datos de los reportes de servicio.
descripción
Se obtienen los datos más importantes de los reportes de los servicios.

Tarea Mostrar resultado de la revisión de la computadora
objetivo
Mostrar al usuario el estado de su computadora con los resultados enviados por el agente técnico.
descripción
Después de que se le revise su equipo al usuario, es necesario mostrarle el resultado de dicha acción.
entrada
Datos enviados por el agente técnico para ser mostrados al usuario.

salida
Datos en la pantalla en un formato entendible para el usuario.
precondición
Se ha realizado una revisión del equipo.
supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.

Tarea Corregir problemas de configuración
objetivo
Hacer que los dispositivos conectados a la computadora funcionen correctamente.
descripción
Para llevar a cabo la corrección de los problemas de configuración, se revisará y corregirá el registro de Windows.
entrada
Tipo de problema, datos de configuración, ubicación en el registro, ubicación de los controladores.
salida
Archivo de configuración escrito correctamente.
precondición
Detección del problema.
supertarea
Revisar equipos.
subtareas
Ninguna.
tipo de descomposición
Ninguna.
ingrediente
Datos de configuración del dispositivo.
descripción
Los dispositivos conectados a la computadora necesitan de parámetros para su buen funcionamiento, los cuales vienen especificados con el dispositivo.

Tarea Revisar configuración de tarjeta de red
objetivo
Mantener la comunicación por medio de la red para poder interactuar con el sistema.
descripción
El agente debe mantener la comunicación por medio de la red para. Revisando con regularidad que no se modifique la dirección IP, los DNS, etc.

entrada
Datos de configuración.
salida
Datos correctos.
precondición
El tiempo de revisión se ha cumplido.
supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.
ingrediente
Dirección IP, direcciones de los DNS's, de la puerta de enlace, identificación de la computadora en la red.
descripción
Son los datos necesarios para poder comunicarnos por medio de la Intranet.

Tarea Corregir configuración de la tarjeta de red
objetivo
Tener comunicación con los demás agentes.
descripción
El agente restaurará la configuración de la red .
entrada
Datos de configuración.
salida
Archivo de configuración escrito correctamente.
precondición
Detección del problema.
supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.
ingrediente
IP, DNS, Getway, Nombre computadora.
descripción
Direcciones de los servidores de nombre, de la puerta de enlace y de la computadora.

Tarea Leer el registro de Windows
objetivo
Leer las cadenas que se encuentran en el registro para detectar posibles cambios en las configuraciones de los dispositivos conectados a la computadora.

descripción
Los agentes van a utilizar los valores almacenados en el registro de Windows para detectar los problemas de configuración y poderlos corregir.
entrada
Nombre de la cadena a buscar.
salida
Valores encontrados en esa cadena.
precondición
El agente está corrigiendo un problema de configuración.
supertarea
Revisar equipo.
subtareas
Ninguna.
tipo de descomposición
Funcional.
ingrediente
Valores posibles de encontrar en la cadena.
descripción
Obtiene los datos de configuración de un respaldo que se encuentra almacenada en una base de datos del servidor.

Tarea Leer respaldo de configuración de la computadora
objetivo
Obtener los valores originales de la configuración de la computadora para corregir los problemas presentados por la misma.
descripción
El agente tiene que leer el respaldo de la configuración de la computadora que está revisando para poder ver si existe algún problema de configuración y corregirlo.
entrada
Nombre de la cadena a buscar.
salida
Valores encontrados en esa cadena.
precondición
El agente está corrigiendo un problema de configuración.
supertarea
Revisar equipo.
subtareas
Ninguna.
tipo de descomposición
Funcional.
ingrediente
Nombre de la cadena a buscar.

descripción

Es la cadena que contiene la configuración del dispositivo que tiene problemas.

Tarea Comparar valores de las cadenas encontradas en el registro de Windows

objetivo

Detectar posibles modificaciones en los valores almacenados en el registro de Windows para detectar fallas.

descripción

El agente comparará los valores que tenga en el respaldo de la configuración de la computadora con los valores encontrados en el registro de Windows para encontrar el problema que hace que un dispositivo no funcione correctamente.

entrada

Valores obtenidos del respaldo de la configuración.
Valores leídos del registro de Windows.

salida

Resultados de la comparación.

precondición

El agente está corrigiendo un problema de configuración.

supertarea

Revisar equipo.

subtareas

Ninguna.

tipo de descomposición

Funcional.

ingrediente

Valores encontrados en el registro de Windows.
Valores leídos del respaldo de la configuración.

descripción

Obtiene los datos de configuración de un respaldo que se encuentra almacenada en una base de datos del servidor.

Tarea Revisar el funcionamiento de los agentes

objetivo

Revisar el buen funcionamiento de los agentes.

descripción

Revisar el comportamiento de los agentes para detectar una posible falla.

entrada

Datos de comportamiento, ubicación del agente.

salida

Informe de funcionamiento.

precondición

Ha transcurrido el periodo determinado por el administrador para el monitoreo de los agentes; o el administrador ha detectado alguna falla en los agentes.

supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.
ingrediente
Datos de comportamiento.
descripción
El agente conoce como se debe de comportar un agente técnico y así poder determinar cual agente funciona correctamente y cual no.

Tarea Crear y configurar otro agente.
objetivo
Mantener los agentes funcionando correctamente.
descripción
Crear agentes con características iguales al agente que dejo de funcionar para remplazarlo y haga las mismas actividades que la anterior.
entrada
Características del agente anterior.
salida
Agente funcionando correctamente.
precondición
El agente no funciona correctamente.
supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.
ingrediente
Datos del agente anterior.
descripción
Características con que contaba el agente.

Tarea Notificar al técnico de una falla no corregida
objetivo
Avisar al ingeniero encargado del soporte, de un problema que el agente técnico no puede corregir.
descripción
Cuando el agente no puede corregir un problema, avisa al ingeniero para que lo revise y corrija.
entrada
Falla no corregida.

salida
Informe de la falla.
precondición
Identificación de una falla que no puede corregir el agente.
supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.
ingrediente
Tipo de falla.
descripción
Contiene el tipo de problema que tiene el equipo y la posible causa.

Tarea Enviar solicitud de servicio
objetivo
Mandar una solicitud de servicio a un agente técnico para que realice el servicio solicitado.
descripción
Cuando se recibe una solicitud de servicio por correo o por otro agente, la secretaria envía la solicitud a un agente que pueda atenderla.
entrada
Datos del usuario.
salida
Solicitud elaborada.
precondición
Se recibe una solicitud de servicio.
supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.

Tarea Elaborar reporte de servicio
objetivo
Elaborar un documento donde se especifiquen los datos del usuario, problema que presentó su equipo y la solución que se le dio; para llevar un control de los servicios realizados y de las fallas más comunes.
descripción
Para llevar un control de los servicios realizados y los problemas más comunes se elabora un documento que se denomina "reporte de

servicio" donde se especifican los datos del usuario, problema que presentó su equipo y la solución que se le dio.

entrada

Datos usuario, problema, solución.

salida

Reporte de servicio.

precondición

Servicio realizado.

supertarea

Corregir problemas de configuración.

subtareas

Ninguna.

tipo de descomposición

Funcional.

Tarea Revisar equipo

objetivo

Revisar periódicamente el equipo para detectar posibles fallas que se presenten.

descripción

El agente encargado de mantener la configuración correcta del equipo, revisa periódicamente los archivos de configuración para detectar posibles cambios en ellos; o cuando el usuario lo solicite expresamente.

entrada

Configuración correcta.

salida

Resultados de la revisión.

precondición

Ha transcurrido un determinado lapso de tiempo o el usuario lo solicita expresamente.

supertarea

Ninguna.

subtareas

Corregir problemas de configuración, Leer registro de Windows, Leer respaldo de configuración de la computadora, Comparar valores encontrados en el registro de Windows.

tipo de descomposición

Funcional.

ingrediente

Respaldo de los archivos de configuración.

descripción

Contienen la última configuración correcta del sistema, con la cual se va a comparar la configuración actual.

Tarea Actualizar respaldo de configuración de la computadora

objetivo

Tener actualizado el respaldo de la configuración de una computadora.

descripción

Una vez que se han corregido los problemas del hardware o configurado un nuevo dispositivo se actualiza el respaldo de la configuración de la computadora a la que se le brindo el servicio.

entrada

Valores actualizados durante el servicio.

salida

Ninguna.

precondición

El agente terminó de configurar la computadora.

supertarea

Ninguna.

subtareas

Ninguna.

tipo de descomposición

Funcional.

ingrediente

Valores modificados del registro de Windows.

descripción

Durante la configuración de una computadora los valores del registro de Windows son modificados y por lo tanto hay que actualizar los valores del respaldo de la configuración con los nuevos valores.

Tarea Enviar formulario de opinión

objetivo

Proporcionar al usuario una forma por medio de la cual pueda expresar su opinión acerca del servicio del sistema.

descripción

Cuando un agente técnico realiza un servicio, se le envía al usuario un formulario para que exprese su opinión acerca del servicio.

entrada

Solicitud de envío.

salida

Formulario recibido por el usuario.

precondición

Se realizó un servicio.

supertarea

Corregir problemas de configuración.

subtareas

Ninguna.

tipo de descomposición

Funcional.

Tarea Leer el contenido del correo

objetivo

Conocer la petición del usuario para enviarla a la secretaria.

descripción

Extraer el contenido del correo para saber quien hizo la petición y pasarlo a la secretaria.

entrada

Correo.

salida

Solicitud de servicio.

precondición

Hay un mensaje nuevo en el buzón.

supertarea

Ninguna.

subtareas

Ninguna.

tipo de descomposición

Ninguna.

ingrediente

Contenido del correo.

descripción

El contenido del correo tiene un formato previamente definido para que el agente lo pueda leer e interpretar.

Tarea Consultar Base de datos

objetivo

Obtener información almacenada en una base de datos para alcanzar un objetivo.

descripción

Cuando un agente tiene que realizar una tarea en la cual necesite información adicional a la que conoce, consultará la base de datos para obtener la información necesitada.

entrada

Datos desconocidos.

salida

Datos conocidos.

precondición

Necesita información para realizar una tarea.

supertarea

Ninguna.

subtareas

Ninguna.

tipo de descomposición

Ninguna.

Tarea Almacenar reportes de servicio y/o generales

objetivo

Tener guardados los documentos generados por el sistema en una base datos para su posible consulta en un tiempo posterior.

descripción

Cuando el sistema genere documentos el agente secretaria se encargará de guardarlos en una base de datos, de acuerdo con el tipo de documento.

entrada

Documento elaborado.

salida

Documento en la base de datos.

precondición

Un documento fue generado por el sistema.

supertarea

Ninguna.

subtareas

Ninguna.

tipo de descomposición

Ninguna.

Tarea Ayudar al usuario a comunicarse con el sistema

objetivo

Facilitar la comunicación de los usuarios con el sistema.

descripción

Para que los diferentes usuarios (técnico, secretaria, usuarios, técnico, administrador) se puedan comunicar con el sistema fácilmente con el sistema hay un conjunto de agentes que los asesora.

entrada

Peticiones de los usuarios.

salida

Respuesta a los usuarios.

precondición

El usuario solicita una acción.

supertarea

Ninguna.

subtareas

Ninguna.

tipo de descomposición

Ninguna.

Tarea Mostrar reportes

objetivo

Dar los reportes solicitados por el agente secretaria.

descripción
El agente secretaria solicita al BDAdmon le proporcione reportes que se encuentran almacenados en la base de datos.
entrada
Fecha, Tipo.
salida
Reportes que coinciden con la fecha y tipo.
precondición
Solicitud por parte de la secretaria.
supertarea
Ninguna.
subtareas
Elaborar estadísticas.
tipo de descomposición
Funcional.

Tarea Guardar nuevos controladores
objetivo
Actualizar la base de datos de los controladores.
descripción
La base de datos de los controladores se debe actualizar cuando se adquieren nuevos dispositivos para las computadoras.
entrada
Controladores.
salida
Controladores guardados en la base de datos.
precondición
Solicitud por parte del agente Asistécnico.
supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.

Tarea Proporcionar controladores
objetivo
Proporcionar los controladores que sean solicitados por los agentes.
descripción
Los agentes pueden solicitar que se les proporcionen algunos controladores, ya sea para configurar un dispositivo o para revisarlos.
entrada
Tipo y marca del dispositivo.
salida
Controladores solicitados.
precondición
Solicitud de algún agente.

supertarea
Ninguna.
subtareas
Ninguna.
tipo de descomposición
Ninguna.

3.3 Modelo de agentes

Con este modelo se describen las características de los agentes tales como: nombre, capacidades de razonamiento, servicios, objetivos, entre otras características.

3.3.1 Identificación de agentes

Como resultado del análisis de tareas realizado, se pueden identificar los agentes del sistema.

Para cada uno de los actores humanos del sistema se necesita crear los agentes de interfaz, que van a servir para que se comunique con el sistema. A continuación se listan:

- Administrador
- AUsuario
- AsisTécnico

Los agentes detectados del sistema son:

- ATécnico
- ASecretaria
- Cartero
- BDAdmon

3.3.2 Descripción de los agentes

A continuación se pueden observar las características de cada uno de los agentes identificados:

Agente AUsuario
tipo
Agente software de interfaz
papel
Interfaz de comunicación usuario-sistema
capacidades de razonamiento
Conoce la configuración original de la tarjeta de red y los datos del usuario.

experiencia

Conocimiento del agente encargado de proporcionar el servicio de mantenimiento.

Conoce la ontología de servicio.

descripción

Este agente ayuda al usuario a interactuar con el sistema para lograr que su equipo este bien configurado y funcionando correctamente. Además de corregir la configuración de la red para poder tener comunicación con el sistema.

Objetivo Comunicar al usuario con el sistema.

tipo

Persistente.

parámetros-entrada

Datos del usuario.

parámetros-salida

Agente usuario ejecutándose.

condición-activación

El usuario activa al agente.

condición-finalización

El usuario desactiva al agente.

condición de éxito

Comunicación correcta con el sistema.

condición de fracaso

No se reciben mensajes por parte del sistema

descripción

El agente-usuario ayuda al usuario a interactuar con el sistema para lograr que su equipo esté bien configurado y funcionando correctamente.

Objetivo Mantener la configuración de la red

tipo

Persistente.

parámetros-entrada

Falla la comunicación con los demás equipos.

parámetros-salida

Configuración correcta.

condición-activación

Solicitud por parte del usuario.

condición-finalización

Comunicación por medio de la red.

condición de éxito

Recepción de un mensaje de un agente externo.

condición de fracaso

No recibe respuesta por parte de un agente externo

descripción

Este agente es el encargado de mantener la configuración de la tarjeta de red para que el usuario tenga comunicación con los demás agentes.

En la Tabla 3.1 se puede ver un resumen de las características del agente que va a interactuar directamente con el usuario

Agente: Usuario		**Clase**: Agente software Interfaz		
Objetivos	**Planes**	**Conocimiento**	**Colaborador**	**Servicio**
Comunicar al usuario con el sistema	Registrarse con el DF	Ubicación del agente facilitador	Facilitador	Registro de agentes en la plataforma
	Buscar el agente que presta el servicio en el DF		Facilitador	Proporciona la dirección del agente que presta el servicio buscado
Mantener la configuración de la red	Revisar periódicamente los datos de configuración de la red	Conoce las direcciones de los DNS's, de la puerta de enlace, la IP, el nombre de la PC en la red		
	Avisar al usuario en el caso de que no se pueda corregir algún problema de la configuración de red			

Tabla 3.1 Agente usuario

Agente AsisTécnico

tipo

Agente software Interfaz

papel

Agente de interfaz de comunicación Técnico-Sistema

capacidades de razonamiento

Conoce la ubicación de los controladores

experiencia

Ubicación de los controladores
Como instalar los controladores
Controladores necesarios para un determinado dispositivo

descripción

Agente encargado de ayudar al técnico a comunicarse con el sistema para poder tener un diagnóstico de la computadora que tiene

problemas en la configuración de los dispositivos conectados a la computadora.

Objetivo Ayudar al técnico a corregir problemas de configuración

tipo
Intermedio.

parámetros-entrada
Datos de configuración del equipo.

parámetros-salida
Configuración corregida.

condición-activación
Solicitud expresa del Técnico.

condición-finalización
Solicitud expresa del Técnico.
Corrección de la configuración terminada.

condición de éxito
Equipo funcionando correctamente.

condición fracaso
Desactivación del agente.

descripción
El agente Asistécnico mostrará un reporte de las acciones que llevo a cabo en el intento de corregir el problema y las posibles causas que pueden estar provocando dicho problema.

Objetivo Ayudar a cargar controladores a la base de datos

tipo
Reactivo

parámetros-entrada
Ubicación de donde obtener los controladores.
Dispositivos al que pertenecen dichos controladores.

parámetros-salida
Controladores disponibles para su uso.

condición-activación
Solicitud expresa del Técnico

condición-finalización
Controladores almacenados en la base de datos.

condición de éxito
Controladores almacenados correctamente

condición fracaso
controladores no cargados

descripción
El agente AsisTécnico actualiza la base de datos de los controladores cuando se adquieren nuevos dispositivos, esta acción la realiza el agente cuando el Técnico lo solicita.

Servicio Cargar controladores

tipo

Funcional

objetivo

Actualizar la base de datos de los controladores

parámetros-entrada

Nombre y marca del dispositivo

parámetros-salida

Ubicación de los controladores

ontología

Controladores

La Tabla 3.2 muestra una descripción resumida del agente AsisTecnico

Agente: AsisTecnico			**Clase**: Agente de Interfaz	
Objetivos	**Planes**	**Conocimiento**	**Colaborador**	**Servicio**
Ayudar al técnico a corregir problemas	Revisar registro	Controladores necesarios	BDAdmon	Consulta de controladores
	Instalar controladores	Forma de instalar los controladores		
	Corregir los valores del registro de Windows	Valores de las cadenas de configuración		
Ayudar a cargar controladores en la base de datos	Obtener los controladores		Técnico	Proporcionar ubicación de los controladores
	Guardar controladores de la base de datos	Ubicación de la base de datos	BDAdmon	Guardar controladores

Tabla 3.2 Características del agente que asiste al técnico

Agente ATecnico

tipo

Agente Software reactivo.

papel

Técnico.

capacidades de razonamiento

Conocer los problemas de configuración mas comunes.

experiencia

Conocimiento de la configuración de la computadora.

Conocimiento de los requisitos mínimos de los dispositivos conectados a la computadora.

Ubicación de los controladores.

descripción

Agente encargado de revisar las computadoras, y de corregir los problemas de configuración que encuentre. Cuando el agente no

pueda corregir algún problema avisará al técnico para que lo solucione.

Objetivo Mantener la configuración correcta de los equipos

tipo

objetivo persistente.

parámetros-entrada

datos de configuración de la computadora.

parámetros-salida

la configuración de la computadora coincide con la de la base de datos del agente.

condición-activación

Ha pasado un lapso de tiempo determinado por el usuario.

condición-finalización

orden expresa del usuario.

condición éxito

equipos funcionando correctamente

condición fracaso

se reciben solicitudes de servicios

descripción

el objetivo principal del agente Técnico es la de mantener funcionando correctamente los dispositivos conectados a la computadora.

objetivo Corregir problemas de configuración

tipo

Persistente.

parámetros-entrada

tipo de problema.

parámetros-salida

configuración correcta del sistema.

condición-activación

Detección de un problema de configuración.

Solicitud expresa por parte del usuario

condición-finalización

El sistema funciona correctamente.

condición de éxito

Equipo funcionando correctamente

condición de fracaso

El equipo no funciona correctamente

descripción

cuando el agente técnico detecta un problema de configuración trata de resolverlo; si el problema es de tipo físico, avisa al usuario para que este a su vez llame al técnico encargado.

objetivo Revisar equipos

tipo

Final.

parámetros-entrada

Datos de configuración de la computadora a la que le va a dar el servicio.

parámetros-salida

reporte del estado de la computadora.

condición-activación

Recibe solicitud de servicio por parte de algún agente usuario.

condición-finalización

El agente técnico termina de configurar el equipo.

condición de éxito

El equipo funciona correctamente.

condición de fracaso

No se puede resolver el problema.

descripción

Cuando el usuario detecta un problema en su equipo, hará la solicitud de revisar su equipo por medio del agente usuario; el cual enviará la solicitud al agente técnico para corregir los problemas .

Servicio Configurar dispositivo

tipo

Funcional

objetivo

Mantener la configuración de los equipos sin errores

parámetros-entrada

Datos del equipo

ontología

Configuración

Servicio Revisar equipo

tipo

Funcional

objetivo

Detectar problemas de configuración

parámetros-entrada

Datos del equipo

ontología

Configuración

En la tabla 3.3 se observa un resumen de las características principales del agente ATecnico

Agente: ATecnico			**Clase**: Agente de sistema	
Objetivos	**Planes**	**Conocimiento**	**Colaborador**	**Servicio**
Mantener la configuración correcta de los equipos	Revisar periódicamente el estado del sistema	Conoce la configuración correcta del sistema	BDAdmon	Proporcionar configuración
	Revisar los registros del sistema	Valores correctos		
Corregir problemas de configuración	Comparar la configuración actual con la última configuración correcta	Conocer la última configuración correcta del sistema		Proporcionar configuración
	Actualizar la configuración			
	Instalar controladores	Controladores necesarios	BDAdmon	Proporcionar controladores
	Elaborar reporte de servicio			
	Enviar formulario de opinión			
Revisar equipos	Desplazarse al otro equipo	Dirección IP	Secretaria	
	Consultar la base de configuraciones	Ubicación de la base de configuraciones		

Tabla 3.3 Descripción del agente técnico

Agente Cartero

tipo

Agente Software Reactivo

papel

Administrador del correo

capacidades de razonamiento

Identificar los mensajes nuevos.

descripción

Agente encargado de leer los correos electrónicos que llegan solicitando el servicio de mantenimiento, y de pasarlos al agente AsisTécnico.

objetivo Administrar el buzón de correos

tipo

objetivo reactivo.

parámetros-entrada

correo electrónico.

parámetros-salida

petición de servicio.

condición-activación

llega un correo al buzón.

condición-finalización
El agente Asistécnico recibió la petición de servicio.
condición de éxito
No hay mensajes sin revisar
condición de fracaso
Existen e-mail sin revisar
descripción
Cuando el agente deja de funcionar, y el usuario quiere que se le revise su equipo, puede solicitar el servio enviando un correo al buzón de servicio.

Servicio Revisar correo
tipo
Funcional
objetivo
Revisar que no haya e-mail sin revisar
parámetros-entrada
Solicitud por parte del AsisTécnico
ontología
Configuración

En la Tabla 3.4 se muestran las características que tiene el agente Cartero

Agente: Cartero		**Clase**: Agente Software		
Objetivos	**Planes**	**Conocimiento**	**Colaborador**	**Servicio**
Administrar el buzón de correos	Leer los mensajes	Interpretación del contenido del mensaje		
	Obtener la información necesaria para dar el servicio			
	Enviar la solicitud al AsisTécnico			

Tabla 3.4 Resumen de las características del agente cartero

Agente ASecretaria
tipo
Agente Software.
papel
Secretaria
capacidades de razonamiento
Conocimiento de los documentos necesarios para el sistema
experiencia
Elaborar reportes generales, estadísticas además de almacenarlos en sus respectivas bases de datos.

descripción
Este agente es el encargado de elaborar los reportes generales, almacenar reportes recibidos, elaborar estadísticas; en general de administrar los documentos generados pos el sistema. Además de servir como interfaz de comunicación de la secretaria

objetivo Comunicar a la secretaria con el sistema
tipo
Intermedio
parámetros-entrada
Solicitudes de la secretaria.
parámetros-salida
Secretaria comunicada con el sistema.
condición-activación
La secretaria inicia sesión en el sistema.
condición-finalización
La secretaria termina la sesión en el sistema.
condición de éxito
Comunicación con el sistema establecida.
condición de fracaso
No se reciben mensajes por parte del sistema.
descripción
El agente secretaria, tiene como objetivo principal a ayudar a la secretaria a comunicarse con el sistema para realizar su trabajo.

objetivo Administrar los documentos generados por el sistema
tipo
Persistente.
parámetros-entrada
Periodo que comprenden los reportes.
parámetros-salida
Reportes realizados satisfactoriamente.
condición-activación
Solicitud de la secretaria de elaborar reportes.
condición-finalización
La secretaria acepta los reportes realizados.
condición de éxito
Documentos ordenados y completos.
condición de fracaso
Documentos faltantes.
descripción
El agente secretaria, ayuda a la secretaria a elaborar los reportes generales, estadísticas y además le ayuda a administrar los documentos generados por el sistema.

Servicio Almacenar reportes
tipo
Funcional
objetivo
Mantener un historial de las actividades realizadas por el área.
parámetros-entrada
Tipo de reporte.
ontología
Formularios

Servicio Mostrar reportes
tipo
Funcional
objetivo
Consultar la base de datos de los reportes
parámetros-entrada
Fecha de los reportes que solicita
ontología
Formularios

La Tabla 3.5 muestra la descripción del agente Asecretaria

Agente: ASecretaria			**Clase**: Agente Software	
Objetivos	**Planes**	**Conocimiento**	**Colaborador**	**Servicio**
Administrar los documentos generados por el sistema	Elaborar reportes generales	Ubicación de la base de datos. Formato de los reportes	BDAdmon	Proporcionar reportes de servicio
	Elaborar estadísticas	Formato de las estadísticas		
	Obtener la información necesaria para elaborarlas			
	Mostrar reportes en pantalla			
	Imprimir reportes			
Comunicar a la secretaria con el sistema	Realizar las actividades solicitadas por la secretaria			
	Enviar mensajes solicitando un servicio que pide la secretaria	Ubicación del agente que presta el servicio	Facilitador	Servicio de páginas amarillas

Tabla 3.5 Características del agente secretaria

Agente AAdministrador

tipo
Agente Software

papel
Asistente del administrador

capacidades de razonamiento
Habilidad para detectar problemas.

experiencia
conocimiento de los agentes asignados a cada computadora.

descripción
Agente encargado de ayudar al administrador del sistema a localizar y corregir fallas; además de ayudar a administrar los controladores de los dispositivos de la computadora.

Objetivo Ayudar al administrador a localizar fallas en el sistema

tipo
Persistente.

parámetros-entrada
Datos correctos de configuración.

parámetros-salida
Coincidencia entre los datos de configuración correctos y los datos que contiene el sistema.

condición-activación
Solicitud expresa por parte del administrador.

condición-finalización
Solicitud expresa por parte del administrador.

condición de éxito
Sistema funcionando correctamente.

condición de fracaso
No hay comunicación entre agentes.

descripción
Uno de los objetivos del agente administrador es el de localizar fallas y corregirlas con la ayuda del administrador del sistema.

Objetivo Comunicar al Administrador con el sistema

tipo
Intermedio.

parámetros-entrada
Datos del Administrador.

parámetros-salida
Agente administrador ejecutándose.

condición-activación
El Administrador inicia sesión en el sistema.

condición-finalización
El Administrador termina sesión en el sistema.

condición de éxito
Comunicación establecida.

condición de fracaso

Comunicación no establecida.

descripción

El agente administrador tiene como objetivo el ayudar al Administrador a comunicarse con el sistema, además de ayudarlo a realizar su trabajo dentro del mismo.

La tabla 3.6 describe al agente Administrador

Agente: Administrador			**Clase**: Agente Software	
Objetivos	**Planes**	**Conocimiento**	**Colaborador**	**Servicio**
Ayudar al administrador a localizar fallas	Revisar los agentes	Características de los agentes	Facilitador	Proporcionar la dirección de los agentes
	Revisar base de datos	Ubicación de la base de datos	BDAdmon	Mostrar contenido
	Mostrar contenido de la base de datos al administrador			
	Corregir problemas	Configuraciones correctas		
	Almacenar nuevos controladores en la base de datos	Ubicación de la base de datos	BDAdmon	Almacenar controladores
Comunicar al administrador con el sistema	Realizar las actividades solicitadas por el administrador			
	Enviar mensajes a los agentes del sistema	Ubicación de los agentes	Facilitador	Proporcionar la dirección de los agentes
	Recibir los mensajes por parte de los agentes	Tipo de mensaje		

Tabla 3.6 Descripción del agente administrador

Agente BDAdmon

tipo

Agente Software

papel

Administrador de la base de datos que contiene controladores y documentos

capacidades de razonamiento

Clasifica los controladores.

Ordena los reportes.

experiencia
Conoce la organización de la base de datos de los reportes.
Conoce la organización de la base de datos de los controladores.
descripción
Este agente es el encargado de administrar las bases de datos además de proporcionar los datos contenidos en estas, a los agentes que lo soliciten.

Objetivo Administrar las bases de datos
tipo
Reactivo
parámetros-entrada
Tipo de solicitud (reportes o controladores).
Fecha de reporte ó periodo.
Tipo de dispositivo, marca.
parámetros-salida
Controladores
Reportes
condición-activación
Solicitud expresa de algún agente
condición-finalización
El agente solicitante obtiene lo que necesita.
condición de éxito
La información almacenada en la base de datos es correcta.
condición de fracaso
La información en la base de datos tiene errores.
descripción
Para tener el control y ordenar adecuadamente las bases de datos es necesario que algún agente se encargue de administrarlas y así no tener datos erróneos.

Servicio Mostrar controladores
tipo
Funcional
objetivo
Permitir la consulta de las bases de datos
parámetros-entrada
Nombre del dispositivo
ontología
Controladores

Servicio Proporcionar controladores
tipo
Funcional
objetivo
Obtener los controladores necesarios para configurar un dispositivo

parámetros-entrada
Nombre y marca del dispositivo
ontología
Controladores

En la Tabla 3.7 se muestra el resumen de la descripción del agente encargado de administrar la base de datos.

Agente: BDAdmon			**Clase**: Agente Software	
Objetivos	**Planes**	**Conocimiento**	**Colaborador**	**Servicio**
Administrar las bases de datos	Guardar reportes	Ubicación de las bases de datos		
		Clasificar por fechas		
	Proporcionar reportes			
	Actualizar controladores	Tipo de dispositivo		
		Marca de dispositivo		
	Proporcionar controladores			

Tabla 3.7 agente administrador de la base de datos

Una vez descritos los agentes con los que va a contar el sistema se les asignan a cada uno las tareas a realizar. A continuación se muestran las tareas descritas anteriormente distribuidas entre los agentes.

Tareas / Agentes	Leer el contenido del correo	Crear y configurar otro agente	Revisar el funcionamiento de los agentes	Revisar equipos	Corregir problemas de configuración	Elaborar estadísticas
ATécnico				X	X	
AAdministrador		X	X			
Cartero	X					
ASecretaria						X

Tabla 3.8. Distribución de Tareas-Agentes

Tareas / Agentes	Elaborar reportes generales	Comparar los valores de las cadenas del registro	Corregir configuración de tarjeta de red	Guardar nuevos controladores	Mostrar reportes	Almacenar reportes
AUsuario			X			
AAdministrador						
BDAdmon				X	X	X
Asecretaria	X					
Atécnico		X				

Tabla 3.9. Distribución de Tareas-Agentes (continuación)

Tareas / Agentes	Enviar solicitud de servicio	Consultar base de datos	Leer respaldo de configuración de la computadora	Ayudar a los usuarios a comunicarse con el sistema	Enviar formulario de opinión	Revisar configuración de tarjeta de red
ATécnico		X	X		X	X
AAdministrador		X		X		
Cartero	X					
ASecretaria		X		X		
AUsuario	X			X		X
Asistécnico				X		

Tabla 3.10. Distribución de Tareas-Agentes (continuación)

Tareas / Agentes	Proporcionar controladores	Leer registro de Windows	Actualizar respaldo de configuración	Mostrar resultado de la revisión de la computadora	Notificar al técnico de problemas no corregidos	Elaborar reporte de servicio
ATécnico		X			X	X
BDAdmon	X					
AUsuario				X		

Tabla 3.11. Distribución de Tareas-Agentes (continuación)

3.4 Modelo de Coordinación

Con este modelo se describen las interacciones de los agentes, los protocolos empleados y las capacidades necesarias para realizar estas interacciones.

3.4.1 Descripción de las conversaciones entre agentes

Conversación solicitar servicio de mantenimiento

tipo
Solicitar un servicio

objetivo
Corregir problema de configuración detectado.

agentes
AUsuario, ATécnico.

iniciador
AUsuario.

servicio
Revisar equipo.

descripción
El usuario puede solicitar al agente ATécnico que revise su computadora. Para detectar o corregir alguna falla. Esta solicitud la puede realizar mediante el agente AUsuario
precondición
Solicitud expresa por parte del usuario.
poscondición
El agente ATécnico manda mensaje al agente AUsuario (de aceptación o de rechazo).
condición-terminación
Ejecución de la acción exitosamente o mensaje de error.
tiempo-ejecución
Depende de la configuración del equipo.

Conversación Enviar formulario de opinión
tipo
Envío de archivo.
objetivo
Obtener la opinión del usuario acerca del servicio prestado.
agentes
ATécnico, AUsuario.
iniciador
ATécnico.
servicio
Ninguno.
descripción
El agente ATécnico envía al usuario un formulario de opinión del servicio, después de haberle dado el servicio.
precondición
El agente ATécnico termino de revisar el equipo.
poscondición
El agente AUsuario recibió el formulario.
condición-terminación
El agente Atécnico recibe la confirmación de parte del agente AUsuario indicando que recibió el formulario.
tiempo-ejecución
No definido.

Conversación Regresar formulario contestado
tipo
Envío de archivo.
objetivo
Enviar el formulario de opinión contestado a el agente ASecretaria.
agentes
AUsuario, ASecretaria.

iniciador
AUsuario.
servicio
Ninguno.
descripción
Cuando el usuario termina de llenar el formulario, lo envía a la secretaria para que lo almacene. Este proceso es llevado a cabo por el agente AUsuario.
precondición
El usuario solicita enviar el formulario de opinión.
poscondición
Formulario enviado.
condición-terminación
El agente Ausuario recibe confirmación de que fue recibido el formulario.
tiempo-ejecución
No definido.

Conversación Registrarse con el DF
tipo
Obligatorio.
objetivo
Darse de alta en la plataforma para poder ser localizado por los demás agentes.
agentes
DF, Cartero, AsisTécnico, Atécnico, Asecretaria, BDAdmon, Ausuario, AAdministrador.
iniciador
Cartero, AsisTécnico, Atécnico, Asecretaria, BDAdmon, Ausuario, AAdministrador.
servicio
Registro en el DF.
descripción
Para localizar a los agentes que se encuentran dados de alta en la plataforma, es necesario que los agentes se registren en el DF. Aquí queda registrado el nombre del agente su dirección y el servicio que presta.
precondición
Inicia el agente.
poscondición
El agente esta registrado en el DF.
condición-terminación
Recibe mensaje de confirmación de registro.
tiempo-ejecución
Sin definir.

Conversación Enviar resultados de la revisión del equipo

tipo
Información.

objetivo
Proporcionar la información obtenida de la revisión del equipo al agente AUsuario para que la muestre al usuario.

agentes
ATécnico, AUsuario.

iniciador
ATécnico.

servicio
Revisar equipo.

descripción
El ATécnico envía los resultados obtenidos durante la revisión del equipo al agente AUsuario. El agente AUsuario a su vez la mostrará al usuario.

precondición
Revisión del equipo terminado.

poscondición
Se tiene el informe de los resultados de la revisión.

condición-terminación
El agente AUsuario envía la confirmación de que recibió los datos.

tiempo-ejecución
No definido.

Conversación Actualizar controladores

tipo
Actualización.

objetivo
Guardar los controladores de los nuevos dispositivos adquiridos.

agentes
AsisTécnico, BDAdmon.

iniciador
AsisTécnico.

servicio
Cargar nuevos controladores.

descripción
Cuando se actualizan los equipos, es necesario actualizar la base de datos con los nuevos controladores para configurarlos. El técnico realiza esto mediante el agente de interfaz AsisTécnico el cual establece la una conversación con el agente encargado de la base de datos BDAdmon

precondición
Solicitud expresa por parte del técnico.

poscondición
Los controladores están almacenados en la base de datos.

condición-terminación
El agente DBAdmon manda un mensaje de terminación de la actualización.
tiempo-ejecución
No definido.

Conversación Solicitar controladores
tipo
Envío de archivo.
objetivo
Obtener los controladores necesarios para poder configurar un dispositivo.
agentes
ATécnico, BDAdmon.
iniciador
ATécnico.
servicio
Proporcionar controladores.
descripción
Para desarrollar su trabajo el agente Atécnico necesita de los controladores del dispositivo que tiene el problema. Estos se encuentran almacenados en una base de datos la cual a su vez la administra un agente BDAdmon al que se le deben de pedir los controladores necesarios.
precondición
El problema que se esta atendiendo necesita utilizar controladores.
poscondición
El agente ATécnico tiene los controladores solicitados.
condición-terminación
El agente ATécnico manda una confirmación de recibido.
tiempo-ejecución
No definido.

3.4.2 Descripción de las intervenciones

En esta parte del análisis se muestra la forma en que intervienen los agentes en las conversaciones entre ellos.

Intervención Solicitar servicio de mantenimiento
lenguaje comunicación Agente
FIPA-KQML
acto-de-habla
Petición
ontología
Servicios
emisor
AUsuario
receptor
Atécnico

En la Figura 3.9 se observa en forma gráfica de que forma intervienen los agentes en el momento de solicitar el servicio de mantenimiento.

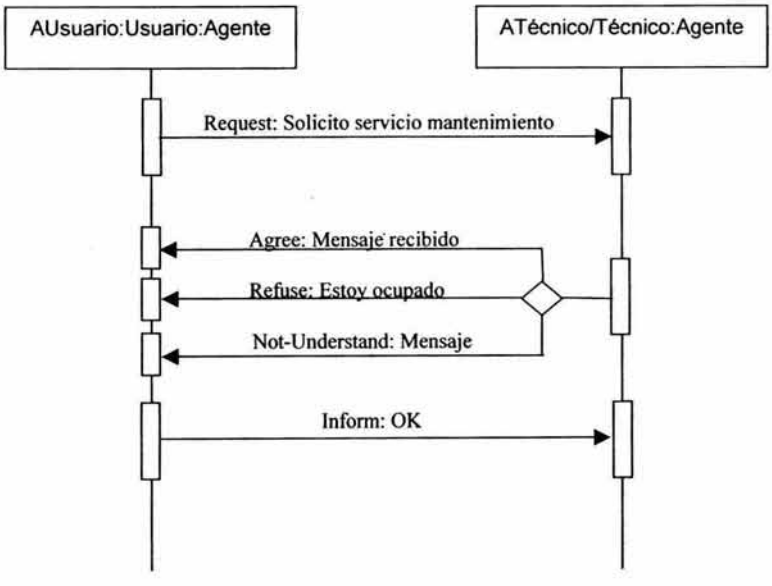

Figura 3.9 solicitar servicio de mantenimiento.

Intervención Pide dirección de ATécnico
lenguaje comunicación Agente
FIPA-KQML
acto-de-habla
Petición
ontología
Servicios
emisor
AUsuario
receptor
DF

En la Figura 3.10 se observa la intervención de los agentes con el facilitador para encontrar a un agente.

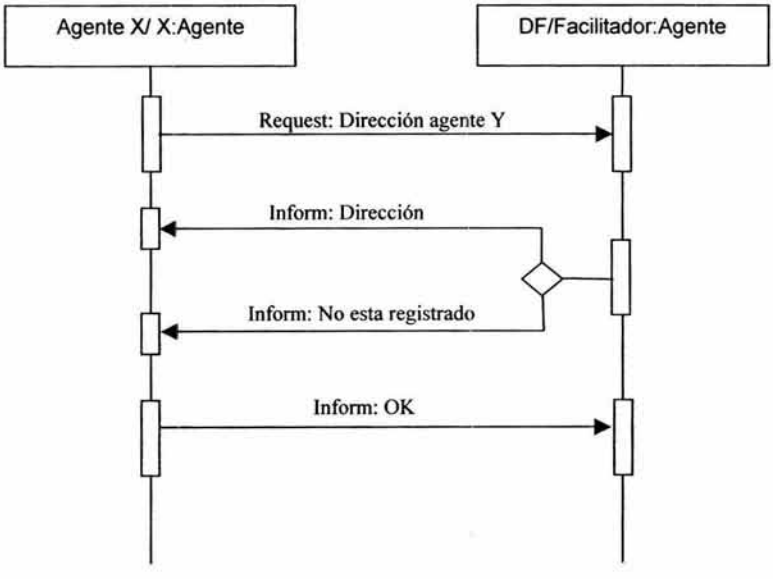

Figura 3.10 Solicita dirección de un agente

Intervención: Registrar con el DF un servicio
lenguaje comunicación Agente
FIPA-KQML
acto-de-habla
Petición
ontología
Servicios
emisor
Cualquier
receptor
DF

A continuación en la Figura 3.11 se muestran los mensajes enviados por los agentes en el momento de registrar un servicio con el facilitador.

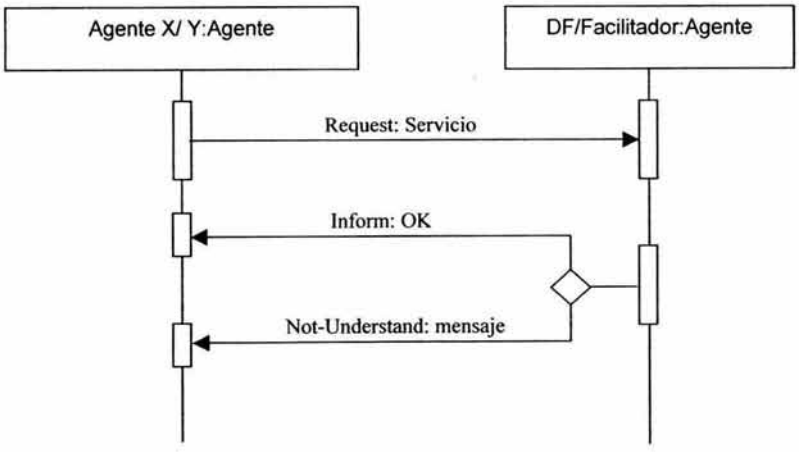

Figura 3.11. Registrar un servicio con el DF

Intervención: Necesito controlador X
lenguaje comunicación Agente
FIPA-KQML
acto-de-habla
Petición
ontología
Servicios
emisor
ATécnico
receptor
BDAdmon

En la Figura 3.12 se observan los mensajes enviados por los agentes ATecnico y Administrador cuando se necesita de un controlador.

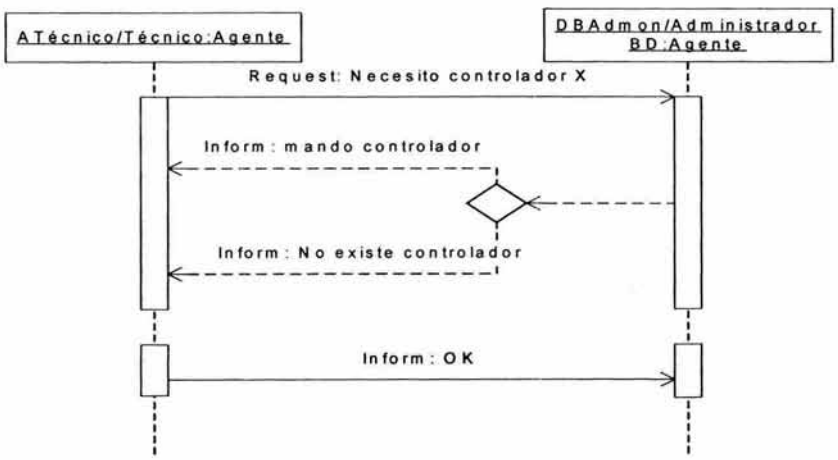

Figura 3.12 Solicitud de un controlador

Intervención: Actualizar controladores
lenguaje comunicación Agente
FIPA-KQML
acto-de-habla
Petición
ontología
Servicios
emisor
AsisTécnico
receptor
BDAdmon

Para la actualización de los controladores los agentes envían los mensajes mostrados en la Figura 3.14

Intervención: Diagnóstico computadora
lenguaje comunicación Agente
FIPA-KQML
acto-de-habla
Informe
ontología
Servicios
emisor
ATécnico
receptor
AUsuario

La Figura 3.13 muestra las intervenciones de los agentes después de haber hecho algún diagnóstico en una computadora.

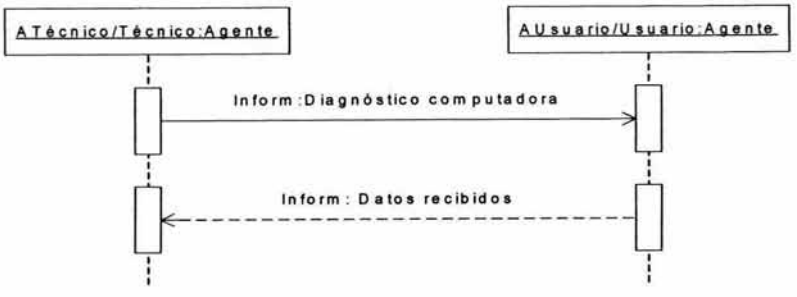

Figura 3.13 Diagnóstico computadora

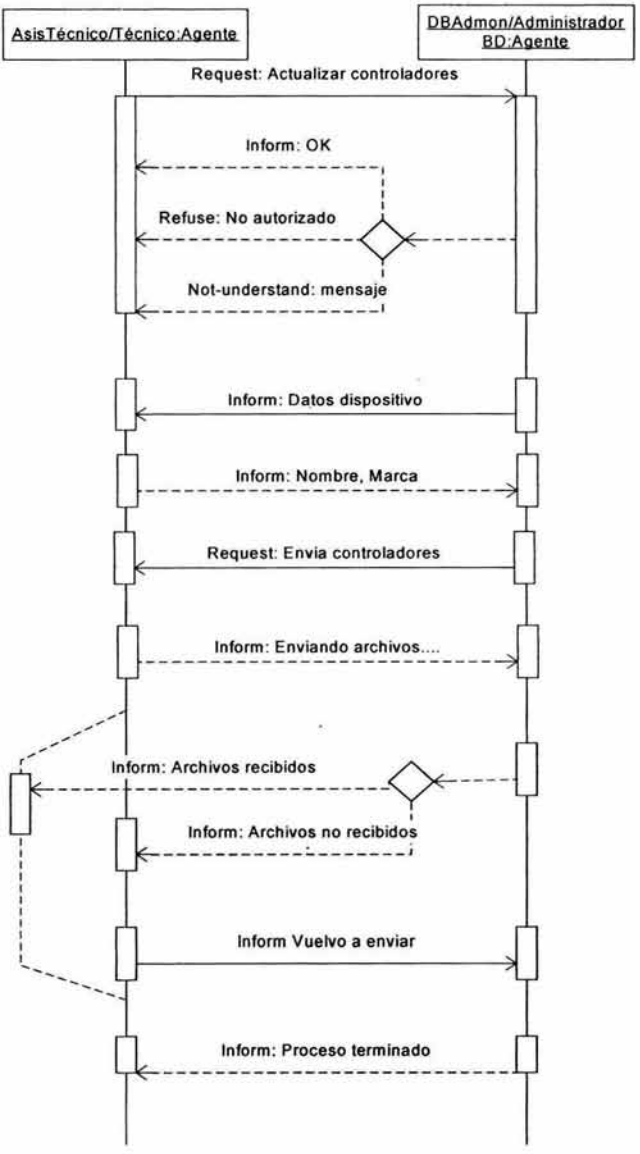

Figura 3.14 Actualizar controladores

Intervención: Formulario contestado
lenguaje comunicación Agente
FIPA-KQML
acto-de-habla
Informe
ontología
Servicios
emisor
AUsuario
receptor
Asecretaria

En la Figura 3.15 se muestra la intervención de los agentes Ausuario y ASecretaria

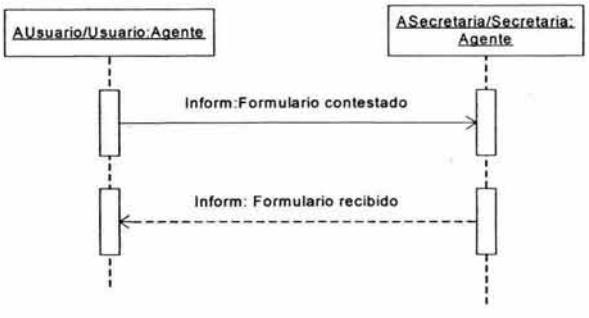

Figura 3.15. Formulario contestado

Intervención: Formulario opinión
lenguaje comunicación Agente
FIPA-KQML
acto-de-habla
Petición
ontología
Servicios
emisor
ATécnico
receptor
AUsuario

Los mensajes enviados por los agentes en el momento de enviar el formulario de opinión se muestran en la Figura 3.16.

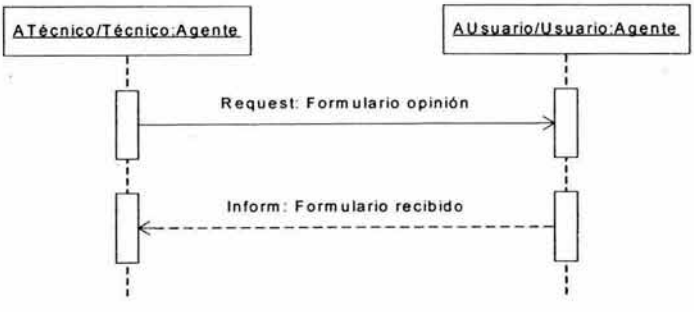

Figura 3.16 Enviar formulario

3.5 Modelo de la organización de Agentes

Describe la forma en que se encuentran organizados los agentes dentro del sistema: la jerarquía de agentes, su relación con el entorno y su estructura.

Los agentes estarán organizados como se muestra en la Figura 3.17.

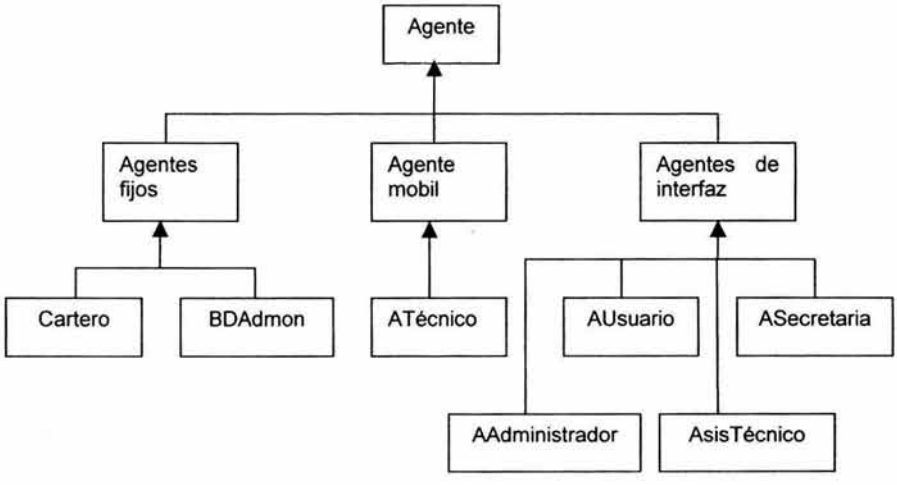

Figura 3.17 Jerarquía de Agentes

AGENTE es el agente base, de la cual todos los agentes que se van a desarrollar heredan características comunes a todos.

En el siguiente nivel se observan los agentes fijos que se encuentran ubicados en una sola computadora. Los agentes móviles, los cuales tienen la capacidad de moverse por la red; y los agentes de interfaz encargados de ayudar a los usuarios del sistema a comunicarse con el mismo.

La relación que encontramos entre agentes se puede observar en el diagrama mostrado en la Figura 3.18.

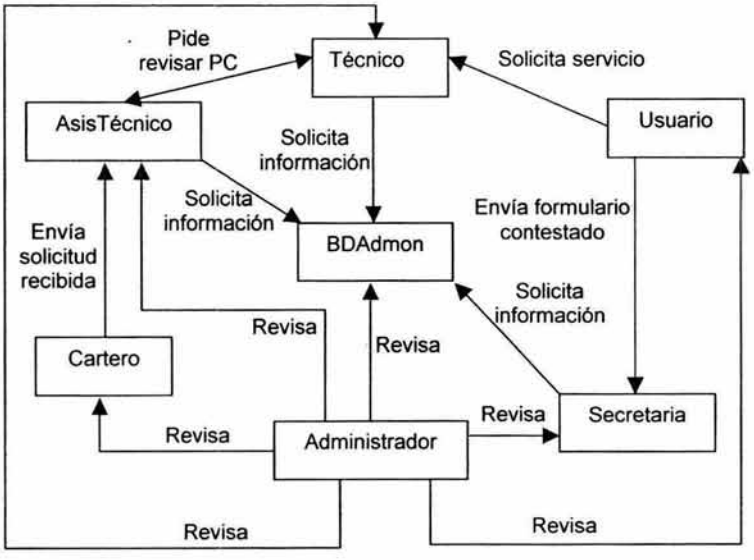

Figura 3.18. Relación de los Agentes

3.5.1 Identificación de los objetos del entorno

Los objetos identificados dentro del sistema con los cuales van a interactuar los agentes son: Archivo de configuración, formulario de opinión, reporte de servicio, reporte general, estadísticas, base de datos de configuraciones, base de datos de reportes, solicitud de servicio, impresora. La Figura 3.19 muestra la relación de los agentes con los objetos identificados

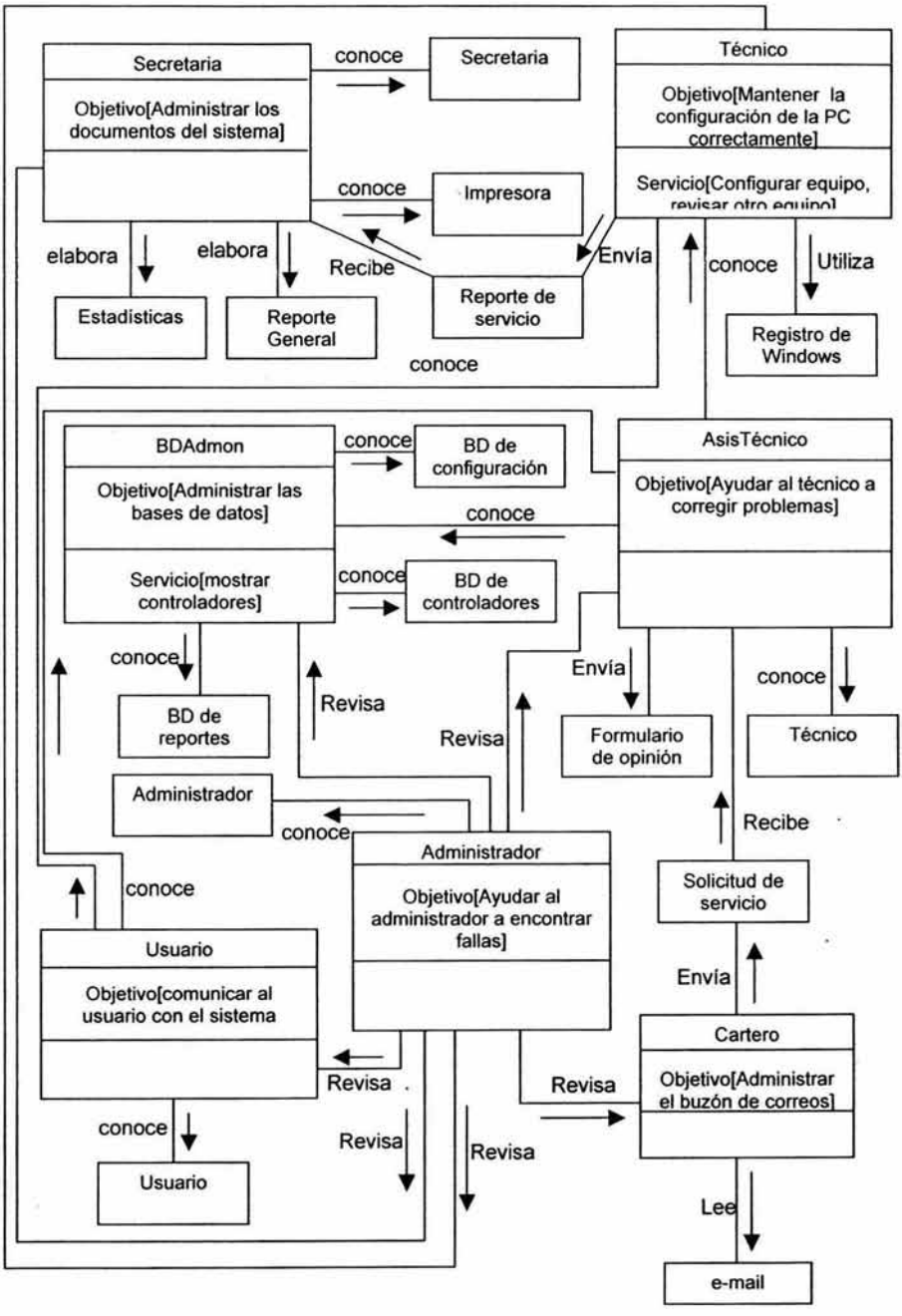

Figura 3.19. Relación con los objetos de entorno

3.6 Modelo de la Experiencia

Describe el conocimiento que necesitan los agentes para poder conseguir sus objetivos. Se distingue entre métodos de resolución de problemas autónomos que descomponen un objetivo en subobjetivos hasta llegar a objetivos directamente realizables y métodos de resolución de problemas cooperativos que descomponen un objetivo en objetivos que realiza el agente en cooperación con otros agentes. El primer método es el utilizado en este trabajo.

3.6.1 Identificación de las tareas genéricas

En el dominio de aplicación se identificaron las siguientes tareas como genéricas (tareas que requieren conocimiento para su ejecución):

- Leer registro de Windows
- Revisar el funcionamiento de los agentes
- Comparar los valores de las cadenas
- Leer respaldo de configuración de la computadora
- Ayudar al usuario a comunicarse con el sistema
- Consultar base de datos

3.6.2 Identificación y descripción del esquema del modelo

En esta parte se describe la ontología o esquema del modelo, es decir, los principales conceptos del dominio y sus relaciones.

Concepto BDServicios
descripción
Conjunto de carpetas donde se almacenan los reportes de servicio, los reportes generales, las estadísticas y los controladores de los dispositivos.
propiedades
Nombre_carpeta: cadena;
ubicación: cadena;

Concepto Usuario
descripción
Persona que utiliza una computadora y que puede comunicarse con el sistema.
propiedades
nombre: cadena;
clave_empleado: numérico;
extensión: numérico;
PC_asignada: cadena;

Concepto Secretaria
descripción
Persona que se encarga de administrar los documentos generados por el sistema.
propiedades
nombre: cadena;
password: cadena;

Concepto Técnico
descripción
Persona encargada de corregir problemas en el equipo cuando un agente no puede corregirlo.
propiedades
nombre: cadena;
password: cadena;

Concepto Archivo de configuración
descripción
Archivo donde se guarda la configuración de la computadora.
propiedades
ubicación: ruta de acceso;
fecha_actualización: fecha;
nombre_archivo: cadena;

Concepto Buzón
descripción
Dirección electrónica donde se reciben las solicitudes de servicio por medio de la red.
propiedades
ubicación: cadena;
nombre_usuario: cadena;
password: cadena.

Concepto Reporte de servicio
descripción
Documento que contiene la información relacionada con el servicio que se le presto a un determinado usuario.
propiedades
fecha_elaboración: fecha;
nombre_usuario: cadena;
nombre_PC: cadena;
dirección_IP: alfanumérico;
problema: cadena;
solución: cadena;
observaciones: cadena.

Concepto solicitud de servicio
descripción
Documento con el cual se solicita un servicio a un agente técnico.
propiedades
fecha: fecha;
nombre_usuario: cadena;
dirección_IP: alfanumérico;
servicio: cadena.

Concepto Reporte General
descripción
Documento en el que se hace un resumen de los servicios realizados durante un determinado periodo de tiempo.
propiedades
periodo: alfanumérico;
fecha_servicio: fecha;
tipo_servicio: cadena;
problema: cadena;
usuario: cadena.

Concepto Impresora
descripción
Dispositivo físico que se utiliza para imprimir documentos en papel.
propiedades
nombre: cadena;
ubicación: cadena;
estado: cadena.

Concepto Diagnóstico de la PC
descripción
Contiene la información relacionada con el estado de la computadora después de haber la revisado el agente.
propiedades
fecha: fecha;
nombre_PC: cadena;
dispositivos: cadena;
estado_dispositivos: cadena;

Concepto Parámetros de configuración
descripción
Características de los dispositivos que se necesitan configurar para que funcione correctamente el sistema.
propiedades
Nombre: cadena;
tipo: alfanumérico;

Concepto Requerimientos
descripción
Recursos de la computadora que necesita un dispositivo para que funcione correctamente.
propiedades
nombre_dispositivo: cadena;
características: cadena.

Concepto Hardware
descripción
Partes físicas de la computadora.
propiedades
controlador: cadena;
fecha: fecha;
proveedor: cadena.

Concepto Configurar
descripción
Acción que realiza el agente para mantener el equipo funcionando correctamente.
propiedades
datos_disp: cadena;
requerimientos: cadena.

Concepto Computadora
descripción
Objeto el cual se va a monitorear y configurar para que funcione correctamente.
propiedades
dirección_IP: cadena;
marca: cadena;
modelo: cadena;
nombre: cadena;
dispositivos: cadena;

Concepto Estadísticas
descripción
Documento que informa de la cantidad de servicio que se tuvieron en un determinado periodo de tiempo.
propiedades
nombre_doc: cadena;
fecha: fecha;
tipo_servicio: cadena;
fecuencia_serv: numérico;

3.6.3 Correspondencia entre el esquema del modelo y tareas genéricas

Las relaciones que se identificaron se listan a continuación:

- Leer registro de Windows **RELACIÓN**: ATécnico, Valores de configuración, AUsuario.
- Comparar valores de las cadenas del registro de Windows **RELACIÓN**: ATécnico, AUsuario, respaldo configuración de la computadora.
- Leer respaldo de configuración de la computadora **RELACIÓN**: ATécnico, AUsuario, BDAdmon, BDServicios.
- Ayudar al usuario a comunicarse con el sistema **RELACIÓN**: AUsuario, Asecretaria, AsisTécnico, diagnóstiico de la computadora.
- Enviar solicitud de servicio **RELACIÓN**: AUsuario, cartero, Solicitud de servicio, ATécnico.
- Consultar base de datos **RELACIÓN**: ASecretaria, BDServicios, Reporte general, ATécnico, BDAdmon, Parámetros de configuración Computadora, Hardware, controladores.

3.7 Modelo de diseño

Con este modelo se define el tipo de agentes que se va a utilizar.

3.7.1 Diseño de los agentes

Sistema-Agente AUsuario
arquitectura
Reactivo
tiene-subsistema
No

Sistema-Agente AsisTécnico
arquitectura
Reactivo
tiene-subsistema
No

Sistema-Agente ATécnico
arquitectura
Híbrido
tiene-subsistema
No

Sistema-Agente AAdministrador
arquitectura
Reactivo
tiene-subsistema
No

Sistema-Agente Cartero
arquitectura
Reactivo
tiene-subsistema
No

Sistema-Agente ASecretaria
arquitectura
Híbrido
tiene-subsistema
No

Sistema-Agente BDAdmon
arquitectura
Reactivo
tiene-subsistema
No

3.7.2 Diseño de la plataforma

El objetivo de este modelo es la de describir el sistema en forma general para conocer los requerimientos del mismo.

Plataforma Sistema Basado en agentes para el monitoreo del hardware
descripción
Sistema que detecta y corrige problemas de configuración del hardware. Este sistema se implementará sobre la plataforma multiagentes llamada JADE.
usa-lenguaje
Java
hardware requerido
PC, con Windows 95 ó mayor
software requerido
j2sdk1.4.0, JADE 2.6
usuario
Técnico, Usuario de PC, Secretaria, Administrador.

Capítulo 4
Desarrollo del prototipo

Introducción

En el prototipo solo se desarrollaron las partes mas representativas del sistema tales como: el agente secretaria, el agente usuario y el diseño de los mensajes mediante los cuales se comunican dichos agentes. En el presente capítulo se describe la forma en que se desarrollo el prototipo, además de explicar algunos detalles de la programación de agentes.

4.1 Herramientas de desarrollo

Las herramientas que se contemplaron para desarrollar el prototipo fueron las siguientes:

- TextPad
- Jcreator
- Gel
- JBuilder

A continuación se describen brevemente cada una de las herramientas

TextPad está diseñado para ofrecer la potencia y la funcionalidad que satisfagan los requisitos más exigentes para edición de texto. La edición de 32 bits puede editar archivos hasta los límites de la memoria virtual, y funciona con MS Windows 9x, NT 4, 2000 y XP.

Textpad está implementado siguiendo las directrices de Microsoft Windows para el diseño de software accesible.

Las principales características con que cuenta son:

- Puede utilizar archivos grandes, hasta el tamaño que permitan los límites de la memoria virtual.
- Compatible con los nombres que utilizan la Convención Universal de Nomenclatura, así como con los nombres de archivos largo que incluyen espacios.
- Resaltado de sintaxis en colores personalizables

JCreator es un entorno para la edición y compilación de programas Java. Este entorno puede utilizar cualquier versión de JDK que se tenga instalada. Además se pueden ejecutar los programas desde el ambiente.

Gel es un IDE para Java; tiene pequeñas diferencias que los IDEs estándar de Java, en concreto este es un ambiente de desarrollo creado para la plataforma Windows, y un bajo consumo de memoria. Tiene un gran número de poderosas características para el desarrollo de programas incluyendo:

- Administrador inteligente de proyectos
- Complementación de código
- Sugerencia de parámetros
- Complementación de etiquetas de html y personalización de etiquetas JSP
- JavaDoc integrado

La desventaja que tiene esta herramienta es que se tiene que descargara y configurar el JDK; y posteriormente configurarlo en la herramienta.

JBuilder también es un editor de programas Java. En esta herramienta es posible agregar mas librerías de java, además de que ya trae integrado el JDK, otra de las características es que genera el código mínimo necesario de las clases que se están programando.

La herramienta que se utilizó para desarrollar el prototipo fue JBuilder por ser un entorno de desarrollo completo y fácil de utilizar; además de que se puede ejecutar la plataforma de agentes directamente desde el entorno de desarrollo.

4.2 Jbuilder

Para el desarrollo del prototipo se utilizó JBuilder. Esta herramienta es para escribir programas con java, además de que se pueden agregar librerías; esto es de gran utilidad pues es posible cargar las librerías necesarias para la plataforma multeagentes (JADE en este caso).

JBuilder permite crear el código mínimo necesario para cada clase utilizada, además de generar los métodos utilizados por la clase. Con esta característica el programador solo se debe preocupar por llenar dichos métodos con el código necesario para que realice la tarea a programar.

Otra de las características es la de ejecutar la plataforma de agentes (JADE) sin tener que salir del ambiente de programación.

4.3 Plataforma multiagente

JADE (Java Agent Development Framework) es un conjunto de herramientas que tiene como objetivo simplificar el desarrollo de sistemas multiagente apegado a las especificaciones de FIPA, tales como: servicio de páginas amarillas, transporte de mensajes y analizadores de servicios, y una biblioteca de los protocolos de interacción FIPA lista para ser utilizada.

JADE contiene todos los componentes obligatorios del control de la plataforma; estos son el acumulador; el AMS, y el DF. Toda la comunicación entre agentes se realiza a través de los mensajes que se envían, todos ellos escritos mediante FIPA ACL.

Básicamente, los agentes se ejecutan mediante un hilo de ejecución cíclico. Siguiendo la solución multi-thread, ofrecida directamente por el lenguaje Java, JADE favorece también la programación de los comportamientos cooperativos.

4.3.1Creación de agentes en JADE

La clase *Agent* representa la clase base común para los agentes definidos por el usuario. Desde el punto de vista del programador, un agente JADE es simplemente una instancia de una clase de Java definida por el usuario que extiende de la clase *Agent*. Esto implica la herencia de características para cumplir interacciones básicas con la plataforma de agentes (registro, configuración, conexión remota, etc.) y un conjunto básico de métodos que pueden ser llamados para implementar el comportamiento particular del agente (por ejemplo enviar/recibir mensajes, usar protocolos de interacción estándar, registrarse con varios dominios, etc.).

4.4 Desarrollo de comportamientos

Los comportamientos son las actividades que los agentes pueden realizar. Para desarrollarlos fue necesario referirse a las tareas que les fueron asignadas a cada uno de los agentes (ver modelo de tareas sección 3.2). Para agregarlos a los agentes simplemente se revisó la tabla de distribución de tareas (ver sección 3.3.8).

En el código que se muestra en seguida se puede observar la estructura de un comportamiento. Los métodos principales que debe de tener un comportamiento son: done() del tipo de regreso booleano y Action().

```
package Comportamientos;

import jade.core.behaviours.*;
import jade.core.Agent;

public class ObtenerConfiguracion extends SimpleBehaviour {

  public boolean done() {
    Método utilizado cuando el agente termina de utilizar el
comportamiento
   return terminar;
  }
  public void action() {
    En esta parte de la clase se coloca el código que le indica al
agente lo que debe de hacer.

  }

}
```

El método *public boolean done()* es el encargado de verificar si la tarea se realizó correctamente. En el caso de que no se ejecute satisfactoriamente, se volverá a ejecutar el comportamiento hasta que no se obtengan errores o hasta el numero de veces que se le haya especificado que la ejecute. Cuando la acción no tiene errores el método regresa un verdadero, de lo contrario regresa un falso.

El método *public void action()* que es el cuerpo de la clase, es la parte en la que se definen las acciones a realizar al invocar la clase.

Una vez declarado el comportamiento es necesario indicarle al agente cuales son los comportamientos que puede utilizar. En el siguiente bloque de código se muestra la forma de cómo se realiza lo anteriormente mencionado.

```
public class Agent1 extends Agent
    {
        protected void setup()
        {
            addBehaviour( new Looper( this, 300 ) );
            addBehaviour( new Looper( this, 500 ) );
        }
    }
```

La parte del código que se encarga de agregarle el comportamiento al agente es *addBehaviour(new Looper(this, 300))*, *Looper()* es el nombre de la clase que define el comportamiento a del agente.

4.5 Desarrollo de la comunicación entre agentes

Los agentes se comunican por medio de mensajes, los cuales siguen una cierta estructura dependiendo del lenguaje utilizado. En este trabajo el lenguaje que se utilizó fue el SL (Semantic Language) el cual es definido por FIPA. Existen tres subconjuntos del lenguaje, el más simple es SL0 que es con el que se trabajo en el prototipo. Para ver cual es la estructura de un mensaje revise la sección 1.4 (Lenguaje de comunicación entre agentes).

La parte que se desarrolló primeramente fue la ontología. Una ontología se define como la forma en que deben de interpretar los agentes el contenido de los mensajes recibidos; en otras palabras es la manera de representar el conocimiento que tienen los agentes. Para desarrollarla se utilizaron los conceptos identificados en el modelo de la experiencia (ver sección 3.6).

Los conceptos son los elementos que forman el vocabulario manejado por los agentes. Para que los agentes puedan manipular dichos conceptos es necesario definir los métodos setXXX y getXXX en las clases donde se definen los conceptos. El método setXXX sirve para establecer los valores asociados a un concepto; y el método getXXX es utilizado por los agentes para obtener los datos asociados a cada uno de los elementos en la ontología.

Con las ontologías también se pueden definir predicados, que están compuestos por conceptos y se utilizan normalmente para referirse a las acciones que debe de ejecutar un agente.

Se debe de poner un nombre a la ontología. Para este trabajo se le puso "OntologíaSISMA" y se agregó a la ontología general, que se encuentra definida dentro de la plataforma multiagente JADE

A continuación se exponen los pasos seguidos para la definición de la ontología utilizada por los agentes del prototipo.

Primeramente se definen los conceptos que pueden manejar los agentes del prototipo. Esto se muestra en el siguiente fragmento de código.

```
public class Usuario implements Concept{

  public void setExtension(String ext) {
    Extension = ext;
  }

  public String getNombre() {
    return nombre;
  }

}
```

Para cada elemento definido dentro de la ontología se necesita una clase. En este fragmento se define el concepto "Usuario", la cual es una clase derivada de la superclase "Concept". Dentro de la clase se encuentran dos métodos "setExtension()" y "getNombre()"; con los cuales el agente asocia una extensión con el usuario y obtiene el nombre del usuario respectivamente.

Después se define el vocabulario que conoce el agente, esto se muestra en el siguiente bloque de código.

```
public interface Vocabulario {
  public static final String USUARIO = "USUARIO";
  public static final String NOMBRE_PERSONA = "Nombre";
  public static final String GERENCIA = "Gerencia";
  public static final String EXTENSION = "Extension";
}
```

Por último se rellena el esquema de la ontología con los conceptos definidos en los pasos anteriores. Esto se ejemplifica con el siguiente código

```
  public OntologiaHardware(Ontology base) {
    super(Nombre_ontologia, base);

add(new ConceptSchema(USUARIO),Usuario.class);

cs = (ConceptSchema) getSchema(USUARIO);
cs.add(EXTENSION, (PrimitiveSchema)getSchema(BasicOntology.STRING));
cs.add(NOMBRE_PERSONA,(PrimitiveSchema)getSchema(BasicOntology.STRING)
);

cs.add(GERENCIA, (PrimitiveSchema) getSchema(BasicOntology.STRING));

}
```

El objetivo de este paso es asociar las clases que definen a los conceptos con las palabras que se refieren a dichos conceptos.

Una vez desarrollada la ontología, se procedió a diseñar los mensajes. Para realizar esta actividad se tomaron en cuenta las conversaciones identificadas y definidas en el modelo de coordinación (ver sección 3.4).

También se escogió el tipo de mensaje a utilizar. Esto se hace dependiendo de si es una petición para realizar una acción, una solicitud de información o una confirmación.

Otro aspecto que se debe de tomar en cuenta es el protocolo de comunicación que se va a utilizar. El protocolo que se utilizó para este prototipo fue el

establecido por FIPA. A continuación se muestran algunos de los elementos que utiliza:

- FIPA-request: para solicitar que se realice una acción.
- FIPA-query: utilizado para solicitar información.
- FIPA_contract-net: con este protocolo se solicitan propuestas a varios agentes.

Para terminar es necesario definirle al agente en que momento debe utilizar un determinado mensaje; esto se hizo apoyándose en la descripción de las intervenciones (ver sección 3.4.2).

En la siguiente Figura se muestra uno de los mensajes utilizado por los agentes

```
(REQUEST
 :sender  ( agent-identifier  :name jorge@cliente:1099/JADE
 :addresses (sequence
ior:000000000000001149444c3a464950412f4d54533a312e3000000000000000000100
000000000006000010200000000a3132372e302e302e3100041800000019afabcb00
00000002d8eadd6d00000008000000000000000a0000000000001000000001000000
2000000000001000100000002050100010001002000010109000000100010100 ))

 :receiver(set(agent-identifier :name secretaria@servidor:1099/JADE))

 :content "((ReporteServicio \"Jorge Martínez Rendón\" GTI \"19 Jul
4\" \"9090\" \"Configuracion de IP\" \"7171\" Mcastor
\"192.168.132.117\" \"No hay conexion con la red\"))"
 :language  FIPA-SL
 :ontology  Ontologia_SISMA
)
```

El mensaje mostrado sigue el especificado en la sección 1.4.4 KQML. Tiene una preformativa "REQUEST" en este caso; el nombre del agente que lo envía "jorge@cliente:1099/JADE"; una dirección la cual esta representado por la secuencia de números de los renglones del 4 al 7; el nombre del agente que recibe el mensaje secretaria@servidor:1099/JADE; el contenido del mensaje definido por la palabra "content"; el lenguaje utilizado "FIPA-SL"; y la ontología "Ontología_SISMA".

Este mensaje es el que se utiliza para enviar el reporte de servicio al agente secretaria. Contiene los datos del usuario, el tipo de problema resuelto, la solución dada al problema, así como la fecha. El mensaje es enviado por el agente técnico.

4.6 Desarrollo de los agentes

Los agentes desarrollados para el prototipo fueron el agente usuario (AUsuario) y el agente secretaria (ASecretaria) los cuales son los agentes mas representativos descritos en el capítulo 3. A continuación se describen los pasos realizados en el desarrollo del prototipo.

4.6.1 Agente usuario (AUsuario)

Para utilizar el JBuilder se cargaron las librerías necesarias para programar los agentes. Se creó la clase inicial, tomando en cuenta lo expuesto en la sección 4.1. JBuilder crea la parte medular del programa extendida (heredada) de la clase Agent ver sección 4.2.1.

En el siguiente bloque de código se muestra como JBuilder entrega la clase inicial.

```
package sisma;
import jade.gui.*;

/**
* <p>Title: </p>
* <p>Description: </p>
* <p>Copyright: Copyright (c) 2002</p>
* <p>Company: </p>
* @author not attributable
* @version 1.0
*/

public class AUsuario extends GuiAgent {

public AUsuario() {

  }

 protected void onGuiEvent(GuiEvent parm1) {

  /**@todo Implement this jade.gui.GuiAgent abstract method*/

 }

}
```

Aquí se observa que JBuilder genera el paquete en donde se va a almacenar nuestra clase (línea 1); importa la clase requerida (línea 2); crea un encabezado para generar la documentación y coloca los métodos necesarios para el agente.

Como se está creando un agente de tipo interfaz la clase se extiende (hereda) de la superclase GuiEvent(); y crea también el método onGuiEvent(), el cual se utiliza par recibir los datos y acciones obtenidas de la interfaz gráfica del usuario.

Una vez que se obtuvo el "esqueleto" del agente, se agregaron los comportamientos necesarios los cuales tienen las actividades descritas en el modelo de tareas (ver sección 3.2), que el agente debe realizar para lograr los objetivos planteados en el modelo de agentes de la sección 3.3.

Para cada una de las tareas asignadas al agente se realizó un comportamiento como se muestra en la Tabla 4.1.

Tarea	Comportamiento
Corregir configuración de tarjeta de red	RevisarEquipo
Enviar solicitud de servicio	SolicitudServicio
Ayudar al usuario a comunicarse con el sistema	IntefazU
Enviar reporte de servicio	EnviarReporte

Tabla 4.1 Comportamientos del agente usuario

Una vez agregados los comportamientos a la clase anteriormente creada el código se ve como sigue:

```
// Método que recoge los valores capturados en la GUI
protected void onGuiEvent(GuiEvent ev){

  if ( ev.getType() == NEW_ACCOUNT){

// Obtiene el dispositivo seleccionado en la GUI
      Seleccion = (String) ev.getParameter(4);

//Comportamiento que corrige la configuración de la red
      addBehaviour( new RevisarEquipo(this, Seleccion));

// Envia reporte de servicio después de terminar la configuración

      addBehaviour( new EnviarReporte(this,(String)
      ev.getParameter(4),(String) ev.getParameter(0),(String)
      ev.getParameter(1),(String) ev.getParameter(3),(String)
      ev.getParameter(2), "Configuracion de IP", "No hay conexión de
      red"));

  }

  else if( ev.getType() == OCULTAR)
      //Oculta la GUI
}
```

El resultado de la ejecución del código anterior es un mensaje que contiene los elementos del reporte de servicio que realiza el agente técnico para enviarlo al agente secretaria.

Como el agente es del tipo interfaz, también se necesito desarrollar la interfaz gráfica de usuario (GUI), la cual captura los datos generales del usuario, así como los requerimientos y el dispositivo que va a ser configurado por el agente (para el prototipo el agente solo configura la tarjeta de red).

En resumen el agente genera un respaldo de los valores de configuración cada determinado periodo de tiempo (determinado por el administrador del sistema).

Cuando el agente realiza alguna corrección de configuración genera un "reporte de servicio", el cual es enviado a la secretaria para almacenarlo y llevar el control de lo servicios que se llevan a cabo mensualmente.

4.6.2 Agente secretaria (ASecretaria)

El segundo agente realizado fue el agente secretaria. Este agente como se describió en el capitulo 3 (ver modelo de agentes sección 3.3), es el encargado de ayudar a la secretaria a realizar sus actividades.

Al igual que el agente anterior para programar la clase, se crearon los métodos principales y posteriormente se fueron rellenando con el código necesario. Se le agregaron los comportamientos (Tabla 4.2) correspondiente a las tareas asignadas; además de su interfaz correspondiente.

Tarea	Comportamiento
Almacenar reportes de servicio	ReporteServicio
Mostrar reportes	MostrarReportes

Tabla 4.2 Comportamientos del agente secretaria

Los demás agentes descritos en la sección 3.3 modelo de agentes, no fueron implementados porque solo se planteo desarrollar un prototipo.

4.7 Comunicación de los agentes con los humanos

Los agentes utilizan, al igual que cualquier otra aplicación, interfaces para comunicarse con los usuarios del sistema.

Para desarrollar las interfaz fue necesario tomar en cuenta los casos de uso identificados para cada uno de los actores (usuarios del sistema) en la parte del análisis (ver sección 3.1.5 identificación de los casos de uso). Además se tienen que tomar en cuenta los datos que necesita el agente para poder realizar su trabajo; y los datos que devolverá para mostrar al usuario.

Otro de los aspectos que se tomaron en cuenta fue que las interfaces deben de tener los elementos necesarios sin que, al usuario le parezca tedioso y difícil de usar.

Las interfaces para poderse comunicar con los agentes necesitan del método GuiEvent ( ); en este método se envían los datos y los eventos recogidos por la interfaz. En la parte del agente, el método que los recibe es onGuiEvent( ), de donde los toma el agente para poder manipularlos y realizar su trabajo.

En la Figura 4.1 se muestra la ventana que presentará el agente cuando se inicia por primera vez.

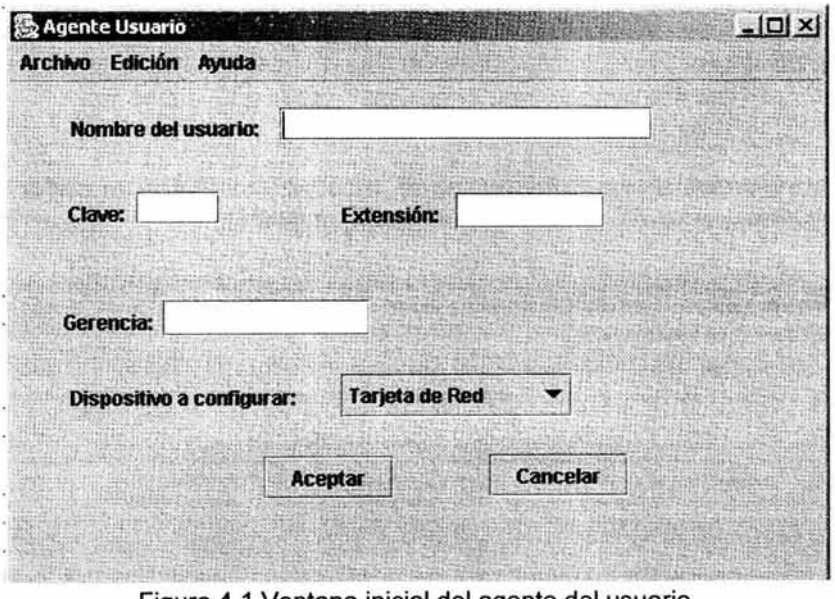

Figura 4.1 Ventana inicial del agente del usuario

En esta venta se solicitan los datos generales del usuario, después el agente los guarda en un archivo para su posterior utilización.

En las siguientes ocasiones que el usuario solicite un servicio, el agente tomará los datos desde el archivo sin necesidad de que el usuario los teclee otra vez. La ventana que se mostrará será la siguiente, en la cual solo se solicitará el dispositivo que necesita revisión.

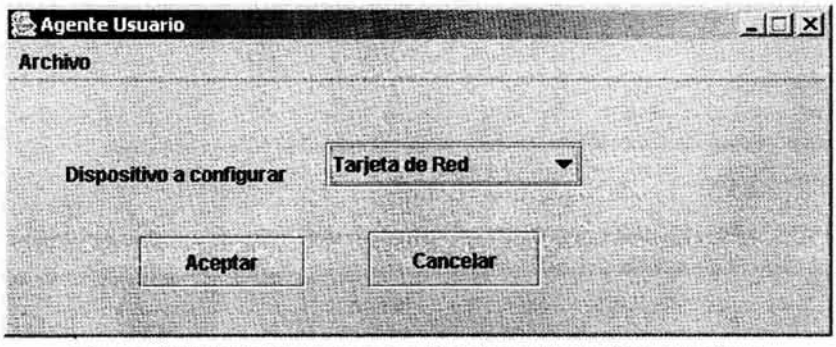

Figura 4.2 Ventana para seleccionar un dispositivo a revisar

Cuando el usuario tenga la necesidad de modificar sus datos, lo podrá hacer desde el menú "Archivo" en la opción de "Modificar datos".

La ventana que va a servir de interfaz entre el agente "Asecretari" y la secretaria se muestra en la Figura 4.3

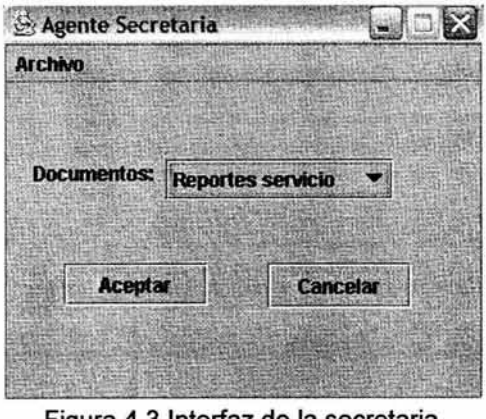

Figura 4.3 Interfaz de la secretaria

Por medio de esta ventana la secretaria puede comunicarse con el agente "Asecretaria". Aquí se pueden consultar los reportes recibidos durante el mes y los reportes generales.

La venta que se muestra en la Figura 4.4 es la que aparece cuando se selecciona desde el menú "Archivo", "Mostrar Reportes"

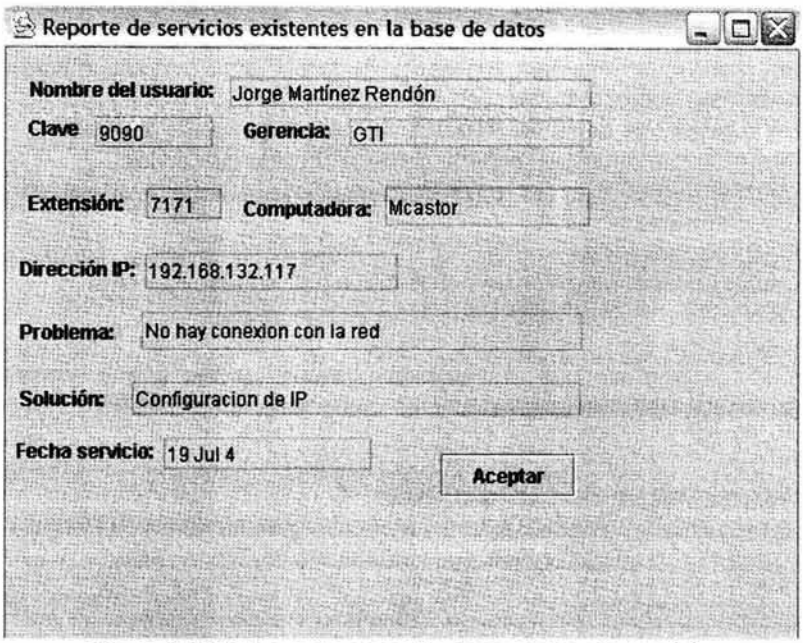

Figura 4.4 Ventana de consulta de reportes de servicio

Esta ventana muestra el contenido de los reportes de servicio recibidos, tales como: nombre del usuario, clave de empleado, Gerencia a la que pertenece, extensión, nombre de la computadora, dirección de red, problema presentado y solución que se le dio; además de la fecha.

4.8 Clases "accesorios" para apoyo en las tareas

Para apoyar las actividades de los agentes se crearon funciones que son comunes para los agentes, las cuales se desarrollaron en clases separadas para que pudieran ser utilizadas por los agentes y así tener menos líneas de código. A estas clases se le denominaron clases "Accesorios".

Una de las clases accesorio es la que obtiene la fecha del sistema. Esta clase la utiliza el agente usuario y el agente secretaria para colocarla en los documentos elaborados por ellos.

4.9 Pruebas aplicadas

Las pruebas aplicadas al sistema fueron las siguientes:

El agente del usuario (AUsuario) fue el primero al que se le hicieron pruebas, tales como:

- Se probó que obtuviera los datos necesario para realizar su trabajo en diferentes computadoras.
- En seguida que detectara el problema de configuración de la red.
- Después que enviara el reporte correspondiente de servicio.
- La interfaz realizara las acciones especificadas cuando el usuario hiciera una determinada acción.
- Mostrara una de las dos interfaz dependiendo de si era la primera vez que se ejecutaba el agente o ya había sido ejecutado previamente.
- Enviara los reportes generados al agente secretaria.

Las siguientes pruebas que se realizaron fueron las del agente secretaria, y constaron en los siguiente puntos:

- Recibiera los mensajes enviados por el agente usuario correctamente.
- Guardara los reportes en un archivo.
- Leyera el archivo de los reportes recibidos y los mostrara en pantalla.
- Enviara un mensaje confirmación cuando recibe un reporte.

Las pruebas anteriores se realizaron teniendo los dos agentes en una misma computadora. Posteriormente las mismas pruebas se realizaron en maquinas diferentes para probar la compatibilidad con las configuraciones realizadas por el agente.

Por ultimo las pruebas se realizaron con el agente secretaria en una computadora y el agente del usuario en otra para poder probar que la configuración realizada por el agente usuario se haya llevado a cabo correctamente, y exista comunicación con la red. Además de que también se puede probar el envío de mensajes y su recepción correcta.

En el CD-ROM que acompaña este trabajo se encuentra el prototipo desarrollado en este capítulo. También se encuentra un manual para su instalación y ejecución.

Capítulo 5
Conclusiones y Trabajos futuros

Conclusiones

La tecnología de agentes es la adecuada para el desarrollar una solución eficiente al problema planteado al principio de este trabajo por las siguientes razones:

- La tecnología de agentes hace diferencias entre humanos y objetos. Mientras que en la programación orientada a objetos todo es modelado como objetos.
- El paso de mensajes entre los agentes se denomina acto del habla, en la tecnología de agentes, mientras que en la programación orientada a objetos la comunicación se realiza entre objetos.
- En la tecnología de agentes las funciones de los objetos son vistos como la forma en que pueden ser manipulados por los agentes.
- Un agente tiene la capacidad de decidir si ejecuta una acción o no.

Como se apreció durante el desarrollo del presente trabajo, las técnicas de modelado de agentes son muy variadas, van desde modificaciones de técnicas de modelado de objetos hasta la ingeniería del conocimiento.

Cada una de la metodologías aquí mencionadas abarcan aspectos diferentes de los agentes; dependiendo del tipo de agente que se desea programar es la metodología que se utiliza.

La metodología que mejores ventajas presenta es la MAS-CommonKADS porque se puede aplicar a diferentes arquitecturas de agentes; además tiene tres diferentes guías para el desarrollo de agentes: para sistemas pequeños, sistemas

medianos, y sistemas grandes (multiagente). La metodología es el resultado de un trabajo de investigación de doctorado (Tesis), que es otra característica importante de la metodología.
Otro aspecto que se puede concluir es que la tecnología de agentes con todas sus ventajas, no es posible utilizarla sin tomar en cuenta la tecnología de objetos, pues se debe valer de esta para tener un modelo de la realidad apegada a ella. En otras palabras, se necesitan definir los objetos con los que van a trabajar los agentes como tales no como agentes.

Una de desventaja que tiene esta tecnología es que no existe un consenso sobre cual es la definición exacta de agente, cada quien emite su propia definición en base a las características deseadas de un agente. Tomando en cuenta su concepto dado, los diferentes investigadores de los agentes inteligentes diseñan y desarrollan sus metodologías y estas varían dependiendo de las necesidades del problema que se va a resolver

Otra desventaja de la mayoría de metodologías es la dependencia de estas con las herramientas. Esto es, al escoger una metodología para el desarrollo de un sistema de agentes, se tiene que conseguir la herramienta que soporta a la misma.

En cuanto al prototipo desarrollado se concluye lo siguiente:

- La herramienta JADE es una de las más completas, y fácil de aprender pues esta desarrollada en Java y es independiente de la plataforma en la que se coloque el sistema.
- La elección de la tecnología de agentes fue la correcta, puesto que:
    - Los sistemas multiagente son intrínsicamente altamente modulares. Lo que ayuda a dar mantenimiento al sistema sin que se afecte el funcionamiento general del sistema.
    - La orientación de agentes tienen la característica intrínseca de que son excelentes para el análisis y diseño de sistemas complejos y distribuidos.
    - Los agentes pueden actuar autónomamente sin la intervención directa del usuario.

Con el desarrollo del prototipo se logra el decremento de la carga de trabajo para el personal se soporte técnico, porque los agentes se encargan de configurar el hardware de la computadora sin la intervención de personas. Este decremento permite a los técnicos dedicar su esfuerzo en la realización de otras actividades que también son importantes para el crecimiento del Instituto.

El prototipo tiene la desventaja de que necesita que el sistema en el que se instale debe de estar configurado correctamente para que el sistema realice su trabajo satisfactoriamente.

Trabajos futuros

1. Desarrollar e implementar los agentes supervisor y cartero para obtener el sistema completo.
2. Desarrollar e implementar las tareas que se diseñaron en la parte del análisis para completar el sistema.
3. Integrar este sistema de configuración de hardware con el sistema de instalación remota de software.
4. Seguir con el estudio de las metodologías para el desarrollo de agentes y proponer una metodología estándar a las existentes.
5. Desarrollar un comportamiento que permita al agente técnico configurar el sistema cuando ha sido formateado el equipo.
6. Implementar un mecanismo de seguridad para evitar que agentes ajenos al sistema obtengan información confidencial del Instituto Mexicano del Petróleo.

Apéndice A
Metodología MAS-CommonKADS

A.1 MAS-CommonKADS

La metodología MAS-CommonKADS extiende los modelos definidos en CommonKADS, añadiendo técnicas de las metodologías orientadas a objetos (OOSE, OMT) y de ingeniería de protocolos (SDL y MSC96) para describir los protocolos entre agentes. La metodología comienza con una fase de conceptuación que emplea la técnica de casos de uso descritos con la notación de OOSE y formalizados con MSC (Message Sequence Charts). Define los modelos descritos a continuación para el análisis y el diseño, que son desarrollados siguiendo un ciclo de vida en espiral. Para cada modelo se definen sus constituyentes (entidades que son modeladas) y las relaciones entre los constituyentes. Asociado a cada constituyente se define una plantilla textual para describirlo. El estado de desarrollo de cada constituyente (vacío, identificado, descrito o validado) facilita la gestión del proyecto.

A.2 Modelos que contempla MAS-CommonKADS

MAS-CommonKADS contempla siete modelos que son:

- Modelo de Organización
- Modelo de Tareas
- Modelo de Agente
- Modelo de Comunicación
- Modelo de la Experiencia
- Modelo de Diseño
- Modelo de coordinación

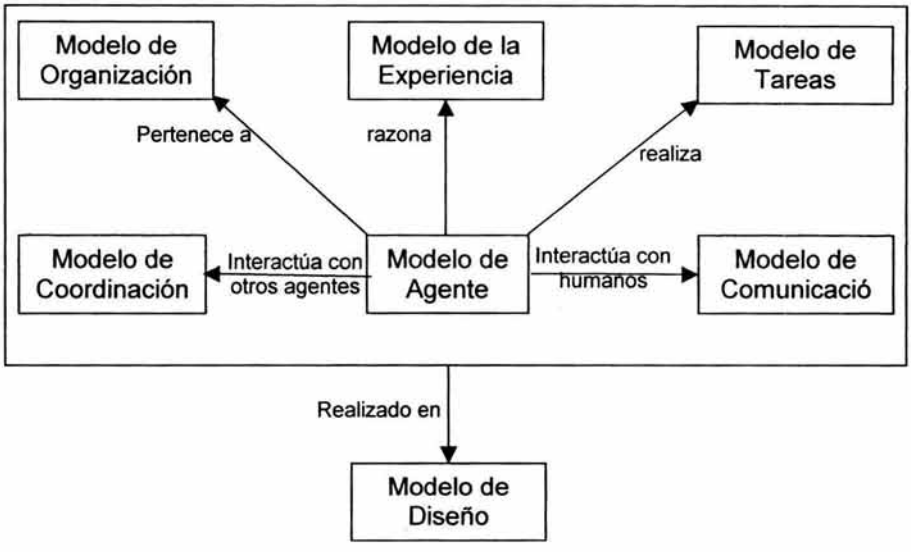

Figura A1 Los modelos de MAS-CommonKADS

A.2.1 Modelo de agente

Describe las características de los agentes: su nombre, capacidades de razonamiento, sensores y actuadores, servicios, objetivos, etc. Este modelo incluye una plantilla textual para describir los agentes y varias técnicas para identificar los agentes tales como el análisis de los actores de la fase de conceptuación, análisis sintáctico del enunciado del problema, heurísticos para identificar agentes, reutilización de componentes (agentes) desarrollos previamente o empleo de tarjetas CRC, que han sido adoptadas al desarrollo orientado a agentes.

A.2.2 Modelo de tareas

Describe las tareas (objetivos) realizados por los agentes y su descomposición, empleando plantillas textuales y diagramas para su desarrollo.

A.2.3 Modelo de la experiencia

Describe el conocimiento que necesitan los agentes para poder conseguir sus objetivos. Se desarrollan varios ejemplares de este modelo para modelar inferencias sobre el dominio, razonamientos sobre el propio agente y sobre otros agentes. Se distingue entre métodos de resolución de problemas cooperativos que

descomponen un objetivo en objetivos que realiza el agente en cooperación con otros agentes.

A.2.4 Modelo de organización

Describe tanto la organización en que va a ser introducido el sistema multiagente para estudiar la viabilidad del mismo como la organización de la sociedad multiagente. La descripción de la organización multiagente emplea una extensión de la notación gráfica del modelo de objetos de OMT, y describe la jerarquía de agentes, su relación con el entorno y su estructura. Esta extensión define un nuevo símbolo para diferenciar los agentes de los objetos del entorno, que consta de una zona para describir los atributos internos del agente (creencias, objetivos, etc.) y otra para describir los atributos externos (sensores, servicios, etc.). La relación de herencia entre agentes para un atributo se define como la unión de los valores las clases precedentes, pudiendo indicar que se desee no heredar los valores de un atributo.

A.2.5 Modelo de Coordinación

Describe las interacciones de los agentes, los protocolos empleados y las capacidades necesarias para realizar estas interacciones. El desarrollo de este modelo consta de dos hitos: la definición de los canales de comunicación entre los agentes, que permite el desarrollo de un primer prototipo, y el análisis de las interacciones para determinar qué interacciones son más complejas y requieren un protocolo de coordinación. Su desarrollo parte de la descripción de los escenarios prototípicos entre los agentes con notación MSC, en cuyos mensajes se especifica el acto de habla realizado y los datos intercambiados.

El conjunto de interacciones permitidas entre agentes se recoge en un diagrama de flujo de eventos. El modelado del procesado de los mensajes se realiza con diagramas de estados de SDL.

A.2.6 Modelo de comunicación

Describe las interacciones hombre-máquina y los aspectos relevantes para desarrollar las interfaces de usuario.

A.2.7 Modelo de diseño

Recoge los requisitos de los modelos previos y determina cómo se llevan a cabo. Se subdivide en tres submodelos:

Diseño de la red

Define los requisitos de la infraestructura de la red de los agentes que son proporcionados por agentes de red. Se distinguen tres tipos de facilidades: facilidades de red, tales como servicio de nombrado de agentes, páginas

blancas y amarillas, protocolos de envío de mensajes disponibles, seguridad de red, contabilidad, etc.; facilidades de conocimiento, tales como servidores de ontologías y métodos de resolución de problemas, traductores entre lenguajes de representación del conocimiento, etc.; y facilidades de coordinación, que indican los protocolos y primitivas de comunicación consensuados en la red, servidores de protocolos, facilidades de gestión de grupos, detección de malas conductas en el uso de recursos comunes, etc.

Diseño de los agentes
Determina la arquitectura de agente mas adecuada para cada agente y descompone cada agente en módulos, que recogen las funciones desempeñadas por cada agente de los modelos de análisis. Por último se establece una correspondencia entre los módulos y la arquitectura seleccionada.

Diseño de la plataforma
Se selecciona y justifica el software y hardware empleado en el desarrollo del sistema multiagente.

A.3 Pasos para la utilización de la metodología

A continuación se muestra un resumen de los pasos de la metodología. En cada modelo se ofrecen diversas técnicas y la notación adecuada para llevarlos a cabo.

A.3.1 Conceptuación

El objetivo de esta fase es comprender mejor cuál es el sistema que desea el cliente. Los principales resultados de esta fase serán una identificación de los objetos que debe satisfacer el sistema desde el punto de vista del usuario, junto con la identificación de los actores que interactúan con en el sistema.

A.3.2 Análisis

El resultado de esta fase es la especificación del sistema compuesto a través del desarrollo de los modelos descritos anteriormente. Solo las extensiones a CommonKADS serán desarrolladas en esta sección. Los pasos de esta fase son:

- Estudio de viabilidad: el modelo de organización permite modelar la organización en que va a ser introducido el sistema multiagente, para estudiar las áreas posibles de aplicación de sistemas inteligentes, su viabilidad y su posible impacto.
- Delimitación: delimita el sistema multiagente de los sistemas externos. El desarrollo del modelo de organización habrá delimitado la interacción del sistema multiagente con el resto de los sistemas de la organización.

- Descomposición: el sistema se analiza mediante el desarrollo solapado de varios puntos de vista:
  - Descomposición funcional: se descompone el sistema en funciones (tareas u objetivos) que deben ser realizados, mediante el desarrollo de un modelo de tareas global.
  - Descomposición en ejecutores: el sistema se descompone en agentes que realizan las funciones anteriormente desarrolladas, mediante el desarrollo del modelo de agente.
- Descripción de las interfaces: tomando como punto de partida la fase de conceptuación y el modelo de agente, se describen las relaciones estáticas que determinan los canales de comunicación de los agentes
- Identificación y descripción de interacciones: la identificación y descripción de las relaciones dinámicas entre los agentes se realiza de la siguiente manera:
  - Las interacciones dinámicas con otros agentes software se describen en el modelo de coordinación.
  - Las interacciones dinámicas con agentes humanos se describen en el modelo de comunicación.
- Descripción del razonamiento de los agentes: para cada agente, debemos modelar el conocimiento que necesita para llevar a cabo sus objetivos, mediante el desarrollo del modelo de la experiencia.
- Validación: cada vez que un agente es descompuesto en nuevos agentes, estos deben ser consistentes lógicamente con la definición previa:
  - Los subagentes son responsables de los objetivos del agente.
  - Los subagentes deben ser consistentes con el modelo de coordinación y mantener las mismas interacciones externas.
  - Validación cruzada con los otros modelos.
  - Los conflictos detectados en los escenarios deben de tener determinado al menos un método de resolución de conflictos.

A.3.3 Diseño

Como resultado de la fase de análisis, un conjunto inicial de agentes ha sido determinado, así como sus objetivos, capacidades e interacciones. Durante esta fase se desarrolla el Modelo de Diseño, que se basa en extender el modelo homónimo de *CommonKADS* para diseñar sistemas multiagente. El Modelo de Diseño recopila los modelos desarrollados en el análisis y se subdivide en tres actividades:

- Diseño de la aplicación. El sistema es descompuesto en subsistemas. Para una arquitectura multiagente, se determina la arquitectura de más adecuada para cada agente.
- Diseño de la arquitectura. En nuestro caso, se selecciona una arquitectura multiagente (en vez de, por ejemplo, una pizarra o una descomposición orientada a objetos). Proponemos que en este punto se determine la infraestructura del sistema multiagente (el denominado modelo de red

[179]). Durante el diseño de la arquitectura se definen los agentes (denominados agentes de red) que mantienen dicha infraestructura.

- Diseño de la plataforma. Se especifica el software y hardware que se necesita (o está disponible) para el sistema.

A.3.4 Codificación y prueba

Las dos tendencias principales son el uso de un lenguaje de propósito general y la utilización de un lenguaje formal de agentes, que puede ser directamente ejecutable o traducible a una forma ejecutable.

A.3.5 Integración y prueba global

Se puede comprobar parcialmente la corrección de la conducta global del sistema utilizando los escenarios típicos que tratan los posibles conflictos y los métodos de resolución de los mismos. Pero, dado que la conducta global del sistema no puede ser determinada durante la fase de análisis o diseño, porque depende de los acuerdos y compromisos concretos realizados entre los agentes, normalmente se necesitará recurrir a la simulación.

A.3.6 Operación y mantenimiento

Una vez probado el sistema, puede ponerse en operación. La filosofía de los agentes facilita el mantenimiento del sistema dada su naturaleza modular.

Apéndice B
Creación de agentes inteligentes en JADE

B.1 JADE

JADE (Java Agent Development Framework) es un conjunto de herramientas que tiene como objetivo simplificar el desarrollo de sistemas multiagente apegado a las especificaciones de FIPA: servicio de páginas amarillas, transporte de mensajes y analizadores de servicios, y una biblioteca de los protocolos de interacción FIPA lista para ser utilizada.

JADE contiene todos los componentes obligatorios del control de la plataforma; estos son el acumulador; el AMS, y el DF. Toda la comunicación entre agentes se realiza a través de los mensajes que se envían, todos ellos escritos mediante FIPA ACL.

Básicamente, los agentes se ejecutan mediante un hilo de ejecución cíclico. Siguiendo la solución multi-thread, ofrecida directamente por el lenguaje JAVA, JADE favorece también la programación de los comportamientos cooperativos.

B.2 Herramientas

JADE tiene algunas herramientas que simplifican el desarrollo de aplicaciones y la administración de la plataforma. A continuación las listamos:

- Agente de Administración Remota, (RMA, Remote Management Agent). Actúa como una consola gráfica para administrar y controlar la plataforma.
- El Agente dummy, es una herramienta de monitoreo y depuración, hecho de una interfaz gráfica de usuario y un agente JADE subyacente en la interfaz. Usándolo es posible componer mensajes ACL y enviarlos a otros agentes y también desplegar la lista de todos los mensajes ACL enviados o recibidos, completados con información de hora para permitir grabar la conversación entre agentes y ensayar.
- El Sniffer es un agente que puede interceptar mensajes ACL mientras están en camino, y los despliega gráficamente usando una notación similar a los diagramas de secuencia de UML. Es útil para depurar las sociedades de agentes observando la manera en que ellos intercambian mensajes.
- El agente Introspector es una herramienta muy útil que permite monitorear el ciclo de vida de un agente y sus mensajes ACL intercambiados.
- El agente SocketProxyAgent es un agente simple, actúa como una compuerta bidireccional entre una plataforma JADE y una conexión TCP/IP ordinaria. Los mensajes ACL, viajando sobre un servicio de transporte propietario de JADE son convertidos a cadenas ASCII simples y enviados sobre una conexión por socket y viceversa.

Hay una interfaz gráfica de usuario que es usada por el Facilitador de Directorios default de JADE y que puede también ser usado por cualquier otro DF que el usuario necesite. Esta interfaz permite, en una forma simple e intuitiva, controlar la base de conocimiento de un DF para federar un DF con otros DFs.

B.3 Creación de agentes en JADE

La clase *Agent* representa la clase base común para los agentes definidos por el usuario. Desde el punto de vista del programador, un agente JADE es simplemente una instancia de una clase de Java definida por el usuario que extiende de la clase *Agent*. Esto implica la herencia de características para cumplir interacciones básicas con la plataforma de agentes (registro, configuración, conexión remota, etc.) y un conjunto básico de métodos que pueden ser llamados para implementar el comportamiento particular del agente (por ejemplo enviar / recibir mensajes, usar protocolos de interacción estándar, registrarse con varios dominios, etc.).

El modelo computacional de un agente es multitarea, donde las tareas (o comportamientos) son ejecutadas concurrentemente. Cada funcionalidad o servicio dado por un agente debería ser implementado como uno o más comportamientos. Un calendario interno de la clase *Agent* y oculto al programador, automáticamente maneja los comportamientos.

JADE controla el nacimiento de un nuevo agente de acuerdo con varios pasos: el constructor del agente es ejecutado, este se registra con el AMS, pone su estado

en activo y finalmente ejecuta el método *setup*(); este método es el punto por donde cualquier agente definido en la aplicación inicia su actividad. El programador tiene que implementar este método para inicializar el agente. Por lo general este método es usado para varias cosas: si es necesario, modificar el registro del agente en el AMS, registrarse con uno o más dominios, y añadir tareas a la cola de tareas listas usando el método *addBehaviour*().

Hasta aquí, podríamos pensar que el modelo de agente que existe realmente en la plataforma JADE es simplista, pero no es así. El comportamiento del agente está determinado por el AFD que aparece en la figura 4 y que aparece también en el documento FIPA.

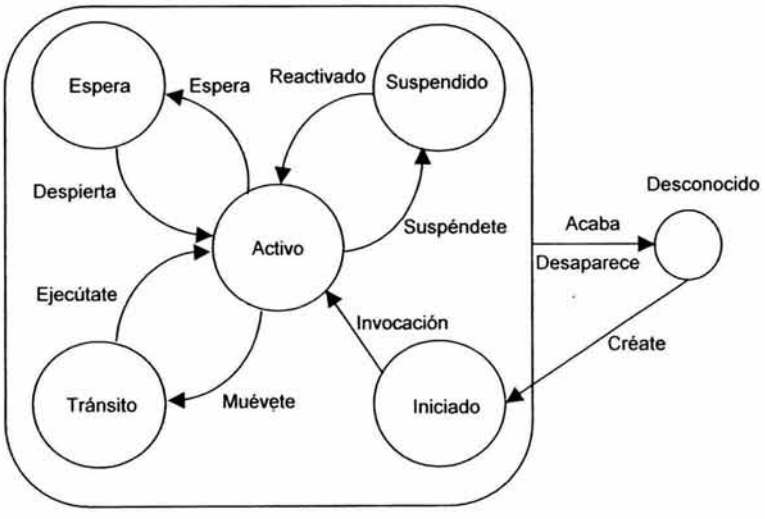

Figura B.1. Autómata que define el comportamiento de un Agente JADE

B.4 Mensajes

Un agente JADE puede mandar mensajes del tipo jade.lang.acl.ACLMessage. Un mensaje perteneciente a esta clase puede ser transformado en un flujo de bits que representa la estructura de mensajes FIPA. Básicamente, un mensaje FIPA contiene los siguientes campos:[6]

- performative: indica el tipo de acto comunicativo del mensaje.
- sender: el emisor del mensaje.
- receiver: el receptor del mensaje.

- reply-to: indica que el mensaje subsiguiente en esta conversación será dirigido al agente nombrado en este campo, en lugar de al agente nombrado en el parámetro del remitente
- content: denota el contenido del mensaje
- language: el lenguaje en el que va representado
- encoding: la codificación
- ontology: la ontología a la que pertenecen los datos del mismo
- protocol: el protocolo de conversación subyacente entre los agentes implicados.
- conversation-id: identificador de la conversación particular a la que pertenece el mensaje.
- reply-with: identificador de mensaje que utiliza el agente que responde.
- in-reply-to: identifica la respuesta con respecto al mensaje original.
- reply-by: dato de tiempo o fecha, dentro de la cual el agente emisor desearía recibir la contestación.

Glosario

ACL
Lenguaje de comunicación entre agentes

AgenTool
Herramienta de desarrollo de agentes creado por IBM

AMS
Agent Management Service. Es el agente responsable de administrar la plataforma y proveer el servicio de páginas blancas.

ARPA
Agencia de proyectos de investigación avanzada

AUML
Extensión del Lenguaje Unificado para el Modelado (UML) que se utiliza para el modelado de agentes.

Concepto
Clase de objetos del dominio de estudio. Se corresponde con el término "entidad" del modelo entidad-relación y "clase" en la programación orientada a objetos.

DF
Directorio Facilitador. De acuerdo con la arquitectura FIPA, este es el agente que provee el servicio de páginas amarillas.

Facilitadores
Agentes que se encargan de prestar el servicio de "páginas amarillas". Esto es, de dar la ubicación de un agente que se está buscando.

FIPA
(Foundation for Intelligent Physical Agents). Fundación para los agentes físicos inteligentes.

GAIA
Metodología para el análisis y diseño de sistemas basados en agentes.

JADE
(Java Agent DEvelopment Framework). Marco de desarrollo de agentes basado en Java.

Jbuilder
Software de desarrollo de programas basados en Java.

JDK o SDK
Es el ambiente en el cual es posible desarrollar cualquier aplicación Java. Este ambiente o paquete incluye: el API del Java, el compilador así como la máquina virtual de la plataforma correspondiente.

KQML
(Knowledge Query Manipulation Language). Lenguaje que se utiliza para representar la intensión del mensaje de un agente.

LISP
Es una familia de lenguajes funcionales no estrictos.

Máquinas de estados
Es un objeto que tiene un estado en cada momento y responde a sucesos.

MAS-commonKADS
Metodología de modelado de sistemas multiagente que utiliza como base la metodología commonKADS.

Meta-modelo
Es una representación de los tipos de entidades que pueden existir dentro en un modelo, sus relaciones y restricciones de aplicación.

Modelo de roles
Es el conjunto de estructuras de papeles a desempeñar dentro del sistema.

Objetos
Es cualquier cosa abstracta, acerca de la cual almacenamos datos y los métodos que controlan dichos datos.

Ontología
Una ontología es una base de datos que describe conceptos dentro de un dominio, algunas de sus propiedades y como se relacionan estos con otros.

OOSE
Metodología para el desarrollo de sistemas orientados a objetos. Este método proporciona un soporte para el diseño creativo de productos de software.

Rol
Es la descripción de un determinado papel que desempeña un agente dentro de una interacción.

Sistema Federado
En este tipo de sistemas los agentes no se comunican directamente, lo hacen por medio de facilitadores.

SMA
Sistema multiagente.

UML
Lenguaje unificado para modelado. Es un leguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software.

Referencias

[1] Cadena Sandoval Carlos y Núñez Gustavo, **Arquitecturas de Agentes Adaptativos: Un Estado del Arte**, No. 15, serie: verde, Centro de Investigación en Computación, IPN. México 1999

[2] Wooldridge M. and N. R. Jennings. Intelligent agents: **Theory and practice. The Knowledge Engineering Review**, 1995.

[3] Wooldridge M., **An introduction to Multiagent Systems**. Ed. Wiley, England 2001

[4] Mosquera Montan, **Análisis y estudio experimental de herramientas basadas en agentes**. Facultad de Informática, Universidad de Coruña, 2001.

[5] F. Gómez Sharmeta and A. Botía Blaya. **Tecnologías y plataformas de agentes**, 2002.

[6] A. S. Rao and M. P. George. **BDI-agents: from theory to practice. In Proceedings of the First Intl. Conference on Multiagent Systems**, San Francisco, 1995.

[7] Nils J. Nilsson. **Artificial Intelligence: A New Synthesis**. Morgan Kaufmann, 1998.

[8] Cadena Sandoval Carlos A., Núñez Esquer G. **Arquitecturas de Agentes Adaptativos: Un Estado del Arte**. Centro de Investigación en Computación, IPN. Informe Técnico, No. 15, Serie: VERDE, Mayo 1999.

[9] Maes, P., 1995, **Artificial Life Metes Entertainment: Life like Autonomous Agents**. Communications of the ACM, Vol. 38, No. 11.

[10] Wooldridge, Michael y Jennings, Nicholas R. **Intelligent Agents: Theory and Practice**. 1995.

[11] **FIPA Agent Management Specification**, pp 2, FIPA, 2002.

[12] IBM Aglets. 1998. Web page: http://www.trl.ibm.co.jp/aglets/.

[13] M. Wooldridge and N. R. Jennings. **Intelligent agents: Theory and practice. The knowledge Engineering Review**, 1995.

[14] Barceinas Guevara, A, **Un marco de comunicación Inter-Agentes en una biblioteca digital**. Tesis de licenciatura. Universidad de las Américas-Puebla.

[15] N. R. Jennings y M. Wooldridge. **Applications of Intelligent Agents. University of London**, Springer 1998.

[16] Campo Vázquez María et al. **Tecnología de Agentes en los Sistemas de Telefonía Móvil**. Departamento de Ingeniería Telemática, Universidad Carlos III de Madrid.

[17] Jennings, N. R. Corera, J. M., Laresgoiti. **Developing industrial multi-agent systems**. Proceedings of the First International Conference on Multi-Agent Systems, (ICMAS-95), 423-430.

[18] B. Burmeister. **Models and methodology for agent-oriented analysis and design. In K. Fischer, editor, Working Notes of the KI'96 Workshop on Agent-Oriented Programming and Distributed Systems**, 1996. DFKI Document D-96-06.

[19] N. Glaser. **Contribution to Knowledge Modelling in a Multi-Agent Framework (the CoMoMAS Approach)**. PhD thesis, L'Universtité Henri Poincaré, Nancy I, France, November 1996.

[20] L. Gasser and J.-P. Briot. **Object-based concurrent processing and distributed artificial intelligence**. In N. M. Avouris and L. Gasser, editors, Distributed Artificial Intelligence: Theory and Praxis, pages 81–108. Kluwer Academic Publishers: Boston, MA, 1992.

[21] Y. Shoham. **Agent-oriented programming. Artificial Intelligence**, 60(1):51–92, Mar. 1993.

[22] Muller, Jorg P., Wooldridge, Michael J. et al. **INTELLIGENT AGENTS III: AGENT THEORIES, ARCHITECTURES, AND LANGUAGES**. ECAI496 WORKSHOP (ATAL) Budapest, Hungary, Agosto 12-13, 1996. Proceedings

[23] **MESSAGE: Methodology for Engineering Systems of software Agents. Initial Methodology**. Julio del 2000. http://www.eurescom.de

[24] grasia.fdi.ucm.es/ingenias/estado/

[25] Wooldridge Michael. **An introduction to MuliAgent Systems**. Ed. JOHN WILEY & SONS, LTD. England, 2001

[26] http://grasia.fdi.ucm.es/ingenias/Aquí se explica el lenguaje.htm

[27] Iglesias Fernández Carlos, **Definición de una metodología para el desarrollo de sistemas multiagente**. Universidad Politécnica de Madrid , 1998 www.gsi.dit.upm.es/tesis/pdf/

[28] Nicholas R. Jennings; Michael J. Wooldridge.**Agent technology: foundations, applications, and markets**. Springer, 1998

[29] http://www.upv.es/sma/teoria/metodologias/articulos/D3finalReviewed.pdf

[30] DeLoach, S., **Analysis and Design using MaSE and AgentTool**. Actas de conferencia. Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference (MAICS). 2001.

[31] Collis, J. C. Y Ndumu, D. T. **The Role Modeling Guide**. Informe. Applied Research and Technology, BT Labs. 1999.

[32] N. Glaser. **Contribution to knowledge Modelling in a Multi-Agent Framework (the CoMoMAS Approach)**. PhD thesis, L' Universtité, Nancy I, Francia, Noviembre 1996.