Sistema Multiagente para la Auto-gestión de Servicios de Red

Jorge Gea Martínez[1] y Francisco Maciá Pérez[1]

[1]Departamento de Tecnología Informática y Computación
GrupoM: Redes y Middleware
Universidad de Alicante
{jgea, pmacia}@dtic.ua.es
http://www.dtic.ua.es

Resumen. En Las TIC se han implantado de forma muy sólida en todos los sectores de nuestra sociedad. Las aplicaciones ofrecen multitud de servicios de red, al mismo tiempo que basan su funcionamiento en otro conjunto de servicios de red. El mantenimiento de este tipo de servicios es una tarea crítica y muy compleja para los administradores de red. En este trabajo se presenta un sistema multiagente (MAS) de servicios de red que trata de alcanzar la auto-gestión de estos servicios. Su diseño y funcionamiento se basa en la definición de un sistema de información expresado como una ontología de conocimiento de gestión. Se trata de un sistema distribuido con adaptabilidad a los cambios que se produzcan en la red y a la incorporación de conocimiento de nuevos servicios.

1 Introducción

La implantación de las Tecnologías de la Información (TI) en nuestra sociedad [1] es un hecho que afecta ya, casi a la totalidad de sectores y ámbitos de negocio. Según el instituto nacional de estadística [2] el uso de muchas de las aplicaciones de las TI está creciendo de una forma espectacular con el paso de los años. Existen multitud de sistemas basados en TI que controlan procesos de una importancia indudable dentro de sectores como la industria, la medicina, el control del tráfico aéreo o el sector bursátil [3]. Se puede afirmar que el fallo o mal funcionamiento de algunos de estos sistemas puede provocar grandes pérdidas económicas e incluso humanas. Estamos llegando a una situación de fuerte dependencia de las TI, por lo que un buen funcionamiento de las mismas es algo que se antoja indispensable en nuestros días.

Las aplicaciones basadas en TI ofrecen todo tipo de funcionalidades, buscando siempre la mayor sencillez posible para el usuario final. Además, los sistemas resultantes han de ser eficientes, eficaces, funcionales, portables e interoperables entre sí. Como consecuencia, la complejidad que llegan a alcanzar es muy elevada [4].

Estos sistemas se apoyan sobre una multitud de servicios de red que funcionan de forma transparente a los niveles superiores de las aplicaciones y que permiten a los desarrolladores centrarse en las funcionalidades concretas que quieren ofrecer en sus sistemas. Estos servicios son de naturaleza muy diversa: podemos encontrar servicios de correo, de bases de datos, de seguridad, que ofrecen mecanismos de comunicación,

transferencia de datos, administración de equipos, servicios de nombre, de directorio, etc. La complejidad de las operaciones que pueden llevar a cabo aplicaciones de tamaño medio que trabajan sobre una red, teniendo en cuenta los procesos implicados que ofrece esta multitud de servicios, es enorme [5]. Esto hace que sea imprescindible una administración efectiva de este tipo de servicios. La gestión de uno sólo de ellos suele resultar sencilla, sin embargo, la necesidad de integrar las funcionalidades de unos con las de otros, provoca que las tareas de administración sean muy numerosas y complejas.

Todo este mantenimiento de servicios supone un gasto muy importante en el desarrollo de proyectos basados en TI que trabajan sobre una infraestructura de red, pues además de la inversión propia en el desarrollo del software o hardware propio de las aplicaciones, se ha de invertir en la instalación y mantenimiento de los servicios de red. En este artículo, se plantea como objetivo lograr un modelo de gestión cero de servicios que trabajan a nivel de red. Para ello incorporaremos semántica en la gestión y plantearemos un sistema multiagente (MAS) que se encargue de dos objetivos fundamentales: la gestión del propio sistema de información del modelo y la ejecución de las tareas oportunas para cada servicio.

2 Antecedentes

A lo largo de los años, en el ámbito de las redes de computadores, son muchas las aproximaciones que han tratado de lidiar con la problemática que supone la existencia de tantas y tan complejas tareas de administración para llevar a buen puerto el mantenimiento de redes y servicios informáticos.

El primer paso para avanzar en la automatización de este tipo de tareas es la integración de los diferentes sistemas de gestión en un mismo modelo. En la bibliografía podemos encontrar diversos modelos de gestión de red integrada. Los más destacados han propuesto protocolos abiertos de gestión que se utilizan ampliamente en la actualidad. Entre estos sistemas, podemos destacar SNMP [6] (actualmente en su tercera versión, SNMPv3) para gestión de redes y equipos o DMI (dmi) para administrar ordenadores personales. Cada uno de estos protocolos está basado en un ámbito concreto de la gestión de redes de computadores para facilitar la labor de los administradores. Ofrecen la posibilidad de administrar elementos del mismo tipo desde un único lugar de la red basándose en un sistema de información propio. También existen propuestas en el sector que tratan de traducir los modelos de información de un sistema a otro, para así poder aplicar las mismas políticas de gestión. Ejemplos destacables de esta filosofía son la tecnología WBEM [7], o las propuestas basadas en el protocolo CORBA [8] [9]. Más reciente es el protocolo NetConf [10], en desarrollo por el IETF, que persigue la gestión integrada de redes independientemente de la infraestructura utilizada. Este protocolo, basado en XML, establece una estructura de operaciones con un nivel de abstracción muy alto para poder ser adaptada más tarde en cada ámbito concreto. Estos modelos tienen el inconveniente de ser dependientes de una implementación específica para los modelos de gestión con los cuales vayan a tratar, por lo que dependen de que exista lógica cableada o de capa intermedia (firmware) en los dispositivos como en SNMP, o de

una implementación software ad-hoc en el caso de NetConf. Por otro lado, son dependientes de un administrador humano que posea los conocimientos necesarios para poder utilizar las herramientas software que trabajan sobre estos protocolos y así gestionar la red, lo que hace que, a pesar de alcanzar un buen grado de integración en la gestión, el objetivo de una gestión automatizada quede aún lejano.

En el ámbito empresarial, encontramos diversas aplicaciones que gestionan servicios de red, también de forma integrada, que ofrecen la posibilidad de automatizar algunas tareas. De este grupo de herramientas podemos destacar Nagios para la monitorización de servicios o EBox como solución más completa para la integración de todo tipo de servicios de red como son servicios de directorio, servidores web, servicios de enrutamiento o de configuración IP. Uno de los inconvenientes más importantes de estas herramientas es la imposibilidad de incorporar nuevos servicios no ofrecidos de forma nativa por estas plataformas, lo cual hace difícil la automatización de tareas ante un escenario cambiante o escalable como puede presentarse en un entorno de gestión a gran escala. Sin embargo, ante un escenario acotado de servicios, estas herramientas se presentan como grandes soluciones para la administración efectiva de una red, siempre que estén en manos de un administrador con los conocimientos de gestión adecuados.

La barrera que siempre nos encontramos para alcanzar la automatización en la gestión es la dependencia de los conocimientos y de la experiencia de un administrador de red. En las propuestas comentadas anteriormente, se trabaja con modelos de información propios, que definen estructuras y jerarquías de conceptos, dejando la lógica de la gestión al administrador humano y al software que controla.

La comunidad científica ha indagado en los últimos años en la aplicación de modelos semánticos a la gestión de redes [11] [12].Estos modelos incorporan conocimiento en los modelos de información mediante la utilización de ontologías [13]. Una ontología permite la especificación formal [14] del conocimiento humano en un área determinada, facilitando así la automatización de procesos, la estandarización de los conceptos y el intercambio del conocimiento.

Estas propuestas basadas en ontologías, han logrado alcanzar cierto automatismo en algunos ámbitos de aplicación concretos [15] [16] [17]. A pesar de demostrar que la aplicación de estas técnicas es una vía válida para afrontar problemas de automatización, no se ha logrado presentar un modelo aplicable al entorno de la gestión de servicios de red que permita una gestión automática y la incorporación de nuevos elementos no previstos en el escenario inicial.

En estas propuestas se aborda el problema desde el punto de vista de la sustitución de un administrador humano por un sistema software que trabaje con una ontología. Este planteamiento presenta la limitación de que todo aquello que no esté definido en la ontología inicial, no puede ser contemplado por el sistema. La definición de una ontología completa e inamovible es algo inabordable por la propia naturaleza del concepto de ontología. Se trataría de conceptualizar de manera formal el conocimiento humano en la gestión de redes. El conocimiento humano, en el ámbito de la gestión de redes y en cualquier otro, es algo que está en constante evolución. Es posible, incluso afirmar, que dos administradores diferentes aplicarían estrategias de resolución diferentes a la hora de abordar un mismo problema. Un consenso total de la comunidad sobre qué se puede y cómo se debe administrar una red de computadores es una idea que se antoja, al menos a día de hoy, poco menos que

utópica. Así pues, a pesar de utilizar una ontología, es complicado adaptar estos modelos a un entorno de aplicación diferente, a un escenario escalable o lograr un grado alto de automatización.

Como consecuencia a lo expuesto, podemos extraer que una solución de automatización en un entorno no acotado, pasa por la creación de un sistema dinámico en el que el conocimiento no sea estático. Partiendo de esta premisa, el planteamiento que aquí se expone es utilizar las técnicas de conceptualización y automatización que nos ofrecen las ontologías aplicándolas en un entorno en el que, tanto los elementos a gestionar como el conocimiento de gestión, se encuentren distribuidos de forma independiente y desacoplada para conseguir la escalabilidad que necesita un entorno dinámico como la gestión de servicios de red.

El diseño e implementación de la ontología objetivo es un problema abierto que se planteará en futuros artículos, en este artículo que nos ocupa, vamos a plantear el modelo general y el sistema multiagente que se encarga de gestionar el sistema de información y de llevar a cabo lo que en él se exprese.

Un Agente Software puede describirse como un programa con capacidad de ejecución independiente y capaz de gestionar autónomamente (sin entrada directa en tiempo de ejecución de un humano) la selección de acciones cuando se producen eventos esperados o limitadamente inesperados. Los Agentes Software pueden poseer diferentes habilidades, pero generalmente poseen una pericia apropiada para tratar con su propio mundo (entorno operacional o dominio de aplicación). Cuando unimos la acción de múltiples agentes para lograr un objetivo más amplio para el que han de comunicarse y operar entre sí los agentes que forman parte del mismo entorno, podemos hablar de un sistema multiagente.

La utilización de sistemas basados en agentes puede ofrecer ventajas sobre los sistemas con control centralizados siempre que se pretenda alcanzar múltiples objetivos y gestionar múltiples entradas sensoriales. Como resultado, la administración de redes y las arquitecturas de control que explotan la tecnología de agentes adquieren un rasgo diferencial con respecto a los planteamientos clásicos. Si pretendemos crear un sistema de elementos auto-gestionables y basado en un sistema de información común, es fácil concluir que el planteamiento del sistema como un MAS resultará efectivo.

Los sistemas multiagente han supuesto un incremento en el interés sobre servicios de comunicación distribuidos en los últimos años [18]. Podemos definir un agente con la explicación ofrecida por el doctor Wooldridge en [19]: “Un agente inteligente es un proceso computacional capaz de realizar tareas de forma autónoma y que se comunica con otros agentes para resolver problemas mediante cooperación, coordinación y negociación. Habitan en un entorno complejo y dinámico con el cual interaccionan en tiempo real para conseguir un conjunto de objetivos”. Las propiedades indispensables de un agente son:

- *Autonomía*: es la capacidad de operar sin la intervención directa de los humanos, y de tener algún tipo de control sobre las propias acciones y el estado interno.
- *Sociabilidad/Cooperación*: los agentes han de ser capaces de interactuar con otros agentes a través de algún tipo de lenguaje de comunicación.
- *Reactividad*: los agentes perciben su entorno y responden en un tiempo razonable a los cambios detectados.

- *Pro-actividad o iniciativa*: deben ser capaces de mostrar que pueden tomar la iniciativa en ciertos momentos.

Otras propiedades destacables serían:

- *Movilidad*: posibilidad de moverse a otros entornos a través de una red electrónica.
- *Continuidad temporal*: los agentes están continuamente ejecutando procesos.
- *Veracidad*: un agente no comunicará información falsa premeditadamente.
- *Benevolencia*: es la propiedad que indica que un agente no tendrá objetivos conflictivos, y que cada agente intentará hacer lo que se le pide.
- *Racionalidad*: el agente ha de actuar para conseguir su objetivo.
- *Aprendizaje*: mejoran su comportamiento con el tiempo.
- *Inteligencia*: usan técnicas de IA para resolver los problemas y conseguir sus objetivos.

Llamaríamos sistema multiagente (MAS) a aquél en el que un conjunto de agentes cooperan, coordinan y se comunican para conseguir un objetivo común. Las principales ventajas de la utilización de un sistema multiagente son:

- *Modularidad***:** se reduce la complejidad de la programación al trabajar con unidades más pequeñas, que permiten una programación más estructurada.
- *Eficiencia***:** la programación distribuida permite repartir las tareas entre los agentes, consiguiendo paralelismo (agentes trabajando en diferentes máquinas).
- *Fiabilidad***:** el hecho de que un elemento del sistema deje de funcionar no tiene que significar que el resto también lo hagan; además, se puede conseguir más seguridad replicando servicios críticos y así conseguir redundancia.
- *Flexibilidad***:** se pueden añadir y eliminar agentes dinámicamente.

3 Modelo de Gestión

El modelo propuesto se ha denominado AnemuS (*Automatic Network Services Management Model, based on a Multiagent SOA System and a Semantic Information System*), acrónimo que sintetiza las bases de la propuesta: Modelo de Gestión Automática de Servicios de Red basado en un Sistema Multiagente de arquitectura SOA (Arquitectura Orientada a Servicios) y un Sistema de Información Semántico (basado en Ontologías).

En esta propuesta no se pretende modificar ninguno de los protocolos de Internet sobre los que funcionan los servicios de red que se pretende gestionar. Así pues, la capa más baja del modelo, la constituye la propia red de servicios interconectados entre sí Ésta capa, a su vez, se puede ver como una serie de dispositivos distribuidos por toda la red, sobre los cuales se ejecutan determinados servicios de red. El objetivo es plantear una solución que trabaje en la capa de aplicación del protocolo TCP/IP, la más alta. Con este planteamiento se logra un gran nivel de abstracción que nos va a permitir dotar al modelo de las capacidades de escalabilidad y adaptabilidad al entorno que planteábamos al inicio de este trabajo.

La primera capa que se establece sobre la red de servicios de red, es la que contiene las interfaces de gestión. Ésta capa ofrece, como si de un API de programación se tratara, todas las posibilidades de control sobre el servicio que se establezcan. Según las características de cada servicio, una interfaz de gestión llevará tareas tales como la ejecución de comandos de Shell, llamadas a funciones de algún lenguaje de programación, lecturas y escrituras de ficheros de configuración o quizá la ejecución de servicios web. Para que un servicio se pueda integrar en el modelo, se habrá de definir el conjunto de funcionalidades y acciones que es capaz de ofrecer y ejecutar. En capas superiores del modelo, estas acciones serán asociadas a conceptos superiores de gestión que harán transparentes al modelo de gestión estas acciones de nivel más técnico.

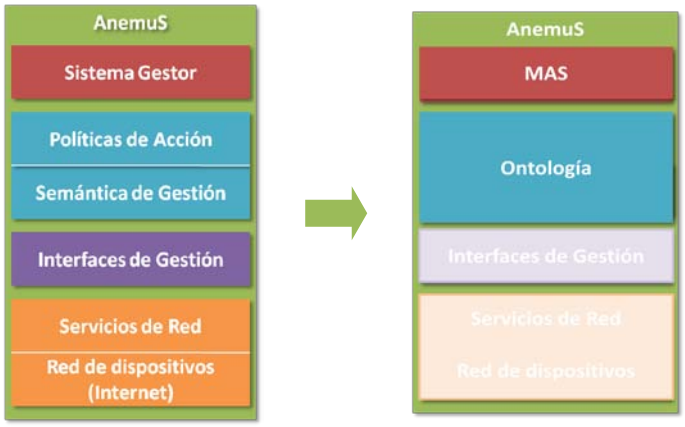

Fig. 1. Visión por capas del modelo propuesto. AnemuS.

4 Sistema Multiagente (MAS)

Ya que lograr una ontología totalmente completa que defina la totalidad de situaciones de gestión es un objetivo quizá demasiado ambicioso a día de hoy, el planteamiento que aquí se expone es el de presentar una base que se pueda incrementar de forma fácil en el futuro. Por este motivo, se ha diseñado como parte del modelo, un sistema gestor multiagente que distribuye el sistema de información entre sus elementos y que ofrece la posibilidad de incorporar nuevo conocimiento no previsto. El esquema general de este sistema lo podemos apreciar en la fig. 2.

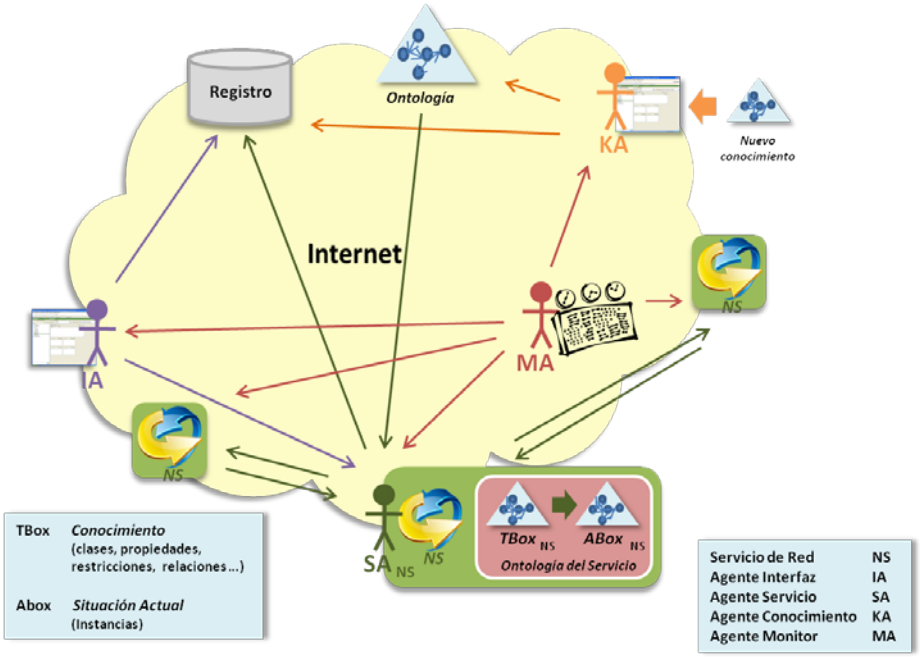

Fig. 2. Sistema gestor multiagente.

Los componentes del sistema multiagente que se define en el modelo son los siguientes:

Agente Servicio (SA)

Cada Agente gestiona de forma autónoma un servicio de red. Por cada servicio de red presente en la red, tendremos un agente de servicio. Los agentes de servicio se comunican entre sí para incorporar conocimiento de gestión unos de otros, así como para conocer la situación del entorno de aquellos con los que se relacionan. Cada agente almacena su propio sistema de información, en el que define el comportamiento del servicio (TBox) y el estado actual del mismo (ABox).

Agente Conocimiento (KA)

Se encarga de la incorporación en la ontología de nuevo conocimiento de servicios de red no definidos anteriormente en la Ontología general. Identifica las correlaciones entre los conceptos de la ontología general y las nuevas ontologías.

Agente Interfaz (IA)

Se encarga de informar a un Agente de Servicios de posibles ajustes en la configuración de un administrador humano. A pesar del enfoque automático de la propuesta, siempre es deseable contemplar los posibles cambios en el entorno que los administradores quieran realizar en los servicios.

Agente Monitor (MA)

Monitoriza el buen funcionamiento del resto de agentes. Es capaz de detectar servicios o agentes caídos y generar alarmas para su gestión. También se contempla la posibilidad de la regeneración automática de agentes.

La comunicación entre los agentes tiene un enfoque SOA. Es decir, cada uno de los agentes ofrece al resto los posibles servicios que puede ofrecer. Los eventos que se produzcan en el sistema (inferencias de los razonadores semánticos de los agentes de servicio, por incorporación de nuevo conocimiento, por caídas de servicio, etc.) serán los que provocarán que un agente realice llamadas a los servicios de otros. El Registro del sistema es el sistema de almacenamiento que se encarga de registrar los servicios de los agentes para que éstos se puedan localizar unos a otros y puedan interactuar entre sí.

La administración de agentes establece el modelo lógico para la creación, registro, comunicación, movilidad y destrucción de agentes. Un modelo de referencia para la administración de este tipo de entornos el establecido por FIPA (Foundation for Intelligent Physical Agents) al cual hemos adaptado, parcialmente, este proyecto.

Los agentes individualmente proporcionan una determinada funcionalidad, pero lo que les hace tan adecuados para ciertos sistemas es su capacidad de cooperar para resolver problemas. Para poder hacerlo, los agentes se han de comunicar entre sí, utilizando un lenguaje común: ACL (Agent Communication Language). Para garantizar la homogeneidad y compatibilidad entre los diversos agentes, la FIPA determina qué forma ha de tener un mensaje y cómo y cuándo deben utilizarse; para ello, esta organización elabora las FIPA Specifications.

El diseño del sistema multiagente expuesto se presenta en especificación de FIPA denominado AUML (Agent Unified Model Language). El uso de esta especificación nos permite crear los agentes de una forma muy precisa manteniendo un alto nivel de abstracción que permitiría la implementación de estos agentes en multitud de tecnologías.

El punto de partida para exponer el funcionamiento de este sistema multiagente es el diagrama de Casos de Uso que se presenta en la fig. 3.

En este diagrama, apreciamos de forma clara el papel central y fundamental que ejercen los agentes de servicio (SA) en el sistema. La principal característica del planteamiento expuesto es alcanzar un grado de autogestión muy alto en la administración de servicios de red. Partiendo de esta premisa, los múltiples agentes de servicio son los que han de concentrar la mayor parte de actividad y comunicación entre sí. El resto de agentes del sistema se utilizan para funciones auxiliares, no explícitamente necesarias en el caso general.

Las funciones de los agentes auxiliares se pueden resumir en:

- Permitir la acción esporádica de un administrador humano.
- Monitorizar el buen funcionamiento del resto de agentes.
- Permitir la incorporación de nuevo conocimiento (ampliar la ontología).

Los agentes de servicio, sin embargo, tienen unos objetivos algo más complejos. Un agente de servicio será capaz de:

- Interpretar la situación del entorno en cada momento.

- Interactuar con los demás agentes de servicio para enviarles información sobre su servicio y para recibir información sobre otros servicios.
- Desplegar su servicio gestionado y ejecutar acciones de configuración sobre el mismo.
- Monitorizar la actividad del servicio gestionado y ser capaz de actuar en consecuencia a los cambios que se produzcan.

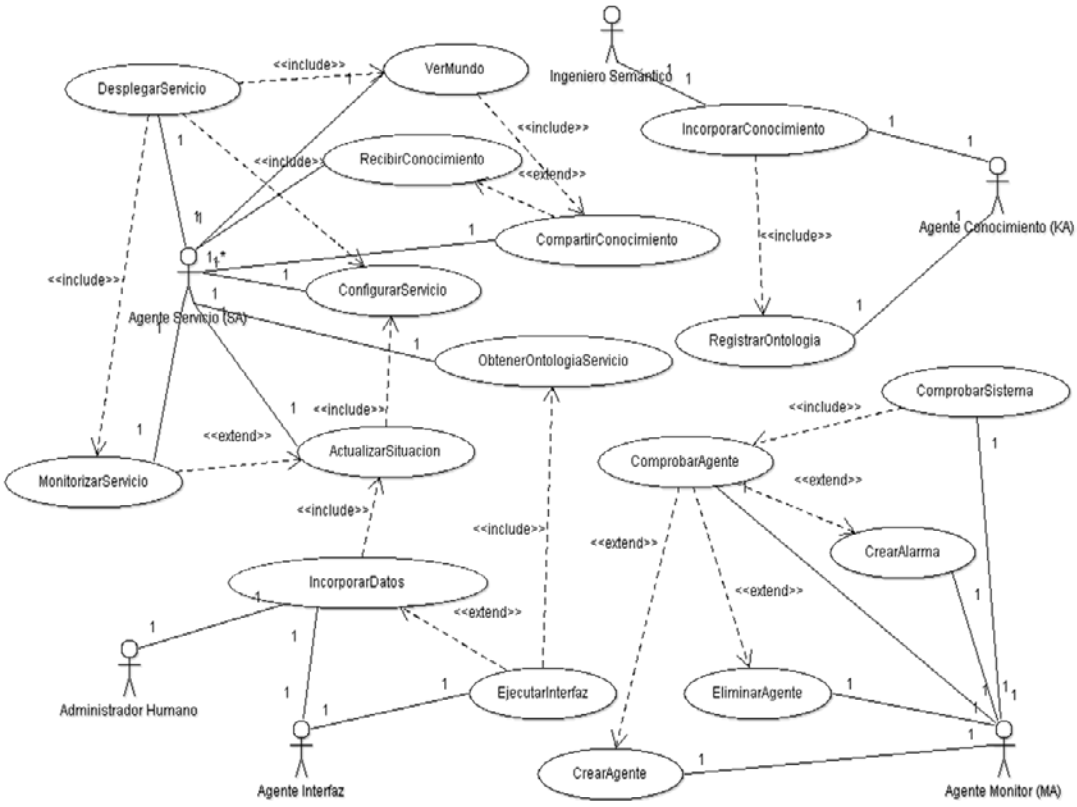

Fig. 3. Sistema Multiagente. Diagrama de Casos de uso.

El funcionamiento principal de un agente de servicio se basa en la interpretación de la ontología mediante razonamientos lógicos. Las diferencias entre el funcionamiento de un agente y otro radican en que la ontología que gestionan es diferente y en que las acciones que pueden ejecutar son las que les permite el servicio mediante la interfaz de servicio definida.

Cada servicio que se quiera gestionar ha de tener previamente conceptualizado su funcionamiento en una ontología que establezca qué conceptos maneja, cómo se relacionan entre sí y qué políticas de acción se han de seguir en cada caso. En este

punto entra en acción el papel del denominado 'Ingeniero semántico'. Un ingeniero semántico se define como a la persona, experto o conjunto de los mismos que, en un momento dado, trasladarán los conocimientos de gestión de un servicio de red a una ontología basándose en una ontología general de gestión de servicios de red ya definida. En este trabajo se propone dicha ontología general y se define un pequeño conjunto de ontologías para el correspondiente pequeño conjunto de servicios de red. En el apartado correspondiente se profundiza sobre este aspecto.

Estamos trasladando el problema del administrador de red a un problema de ingeniería semántica, ya que, en efecto, la tarea de conceptualización dista de ser sencilla. Aquí nos encontramos con un problema abierto en el que falta aún camino por recorrer, pues aún no existen herramientas lo suficientemente potentes como para ofrecer entornos de diseño de ontologías que sean intuitivos, cómodos y visuales para que cualquier persona fuese capaz de utilizarlas sin tener conocimiento alguno del área de las ontologías.

En contrapartida a esta desventaja, las ganancias que se obtienen con este traslado del problema son múltiples. La primera y más evidente consiste en que los administradores resuelven los mismos problemas en múltiples ocasiones, en diferentes entornos y lugares. Con una ontología de servicio en la que se establece cómo actuar con dicho servicio, la resolución de cada situación solamente se habría de plantear una vez para expresarla en forma de ontología. Hecho esto, es el agente de servicio quien se encarga en el futuro de resolver esos problemas por nosotros. En un entorno ideal, la ontología sería accesible desde cualquier servicio del mundo y para cualquier administrador, pudiendo ser completada, en cualquier momento y gracias al agente de conocimiento, por administradores a los que les surgieran situaciones para las que no se hubiese establecido respuesta previamente.

Frente a la complejidad habitual de los gestores software, los cuales tienen objetivos precisos y, habitualmente, inmutables, tendríamos el conocimiento totalmente globalizado, con software sencillo e idéntico para cualquier servicio de red que se plantee.

En la siguiente sección, se especifica en detalle el funcionamiento de cada caso de uso planteado en la fig. 3 utilizando los diagramas de secuencia de la especificación AUML, aunque, por cuestiones de espacio, nos centraremos solamente en el agente más importante del sistema, el Agente de Servicio

4.1 Agente de Servicio (SA)

Todos los Agentes, debido a su enfoque SOA han de publicar sus servicios en el registro. En las pruebas realizadas y, debido a la amplia aceptación actual, se han utilizado los Servicios Web como tecnología de SOA. Así pues, el registro donde se publican los agentes es un registro que utiliza el protocolo UDDI. Los agentes se describen en una hoja WSDL y la comunicación entre agentes se basa en el protocolo SOAP.

El Agente de servicios, tras registrarse, se encarga de desplegar el servicio que gestiona, para lo cual, interpretará la situación actual (VerMundo, fig 5) y aplicará las acciones de configuración necesarias a su servicio en base a lo que detecte.

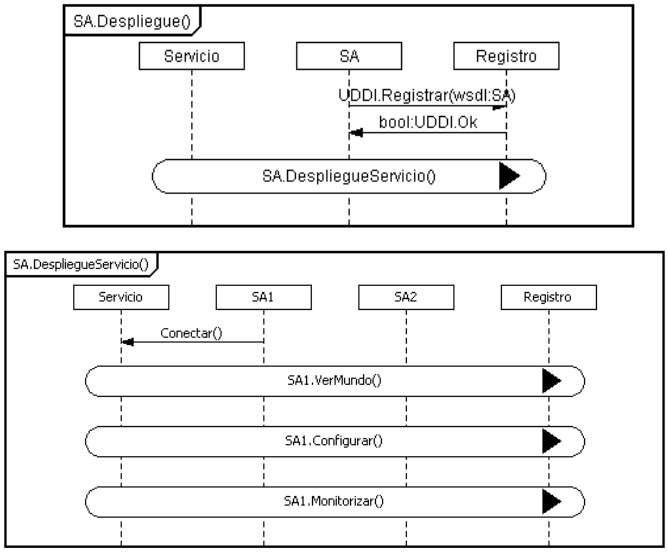

Fig. 4. Despliegue Agente de Servicio.

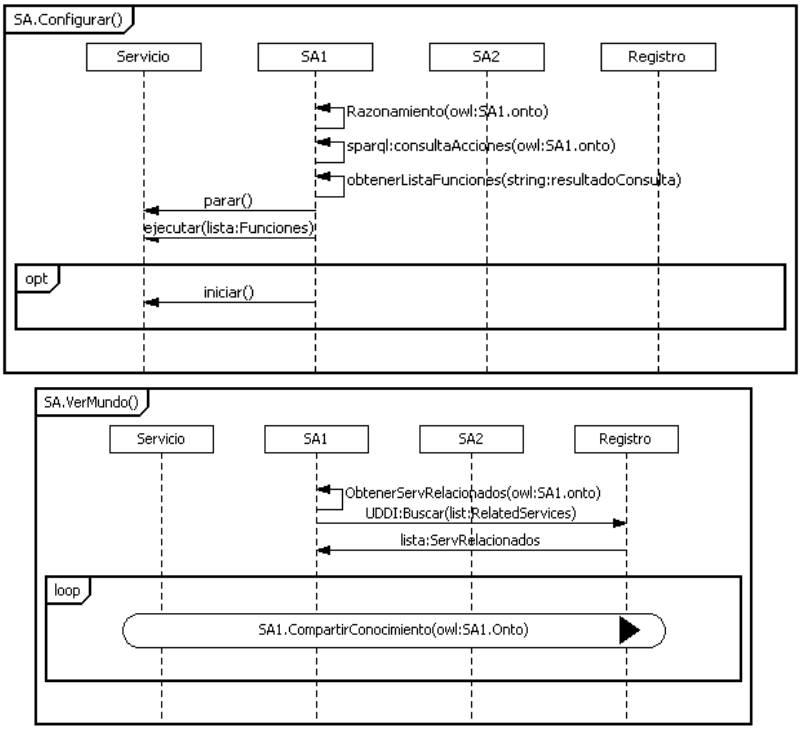

Fig. 5. Agente de Servicio (SA): Configurar y VerMundo.

La acción denominada VerMundo (fig. 5) que realizan los Agentes de Servicio no es más que una importación, en base a la ontología del servicio, de las ontologías de los servicios asociados. Una vez importadas las ontologías correspondientes, se lleva a cabo un proceso de unión de ontologías. En este proceso se extraerán los conceptos comunes para obtener la ontología final que se utilizará para el razonamiento. Así pues, cada servicio tendrá una visión particular y única del entorno, pues solamente conocerá lo que necesita conocer de otros servicios para poder funcionar (si se detectan conceptos en otras ontologías que no tienen relación con los del agente de servicio que ejecuta este proceso, esos conceptos se ignoran). En base a esta visión particular, tenemos el denominado TBox de conceptos que representábamos en el modelo general de la fig. 3 y que podemos denominar 'conocimiento del servicio'. Las instancias de estos conceptos, que se modificarán proactiva por los agentes, que reaccionarán a los cambios producidos en el entorno por otros agentes y actuarán sobre la ontología, conforman el ABox del servicio, lo que podemos llamar 'estado del mundo'.

La acción de Configurar, es la que ejecuta las acciones de interfaz que se han obtenido del resultado del razonamiento ejecutado sobre la ontología, una vez se tiene el ABox o estado del mundo. En la arquitectura que se plantea en este trabajo, se expone de manera más detallada cómo se llevarían a cabo estos procesos de forma interna por los agentes, en esencia, un razonador lógico obtiene inferencias asociadas a acciones de configuración.

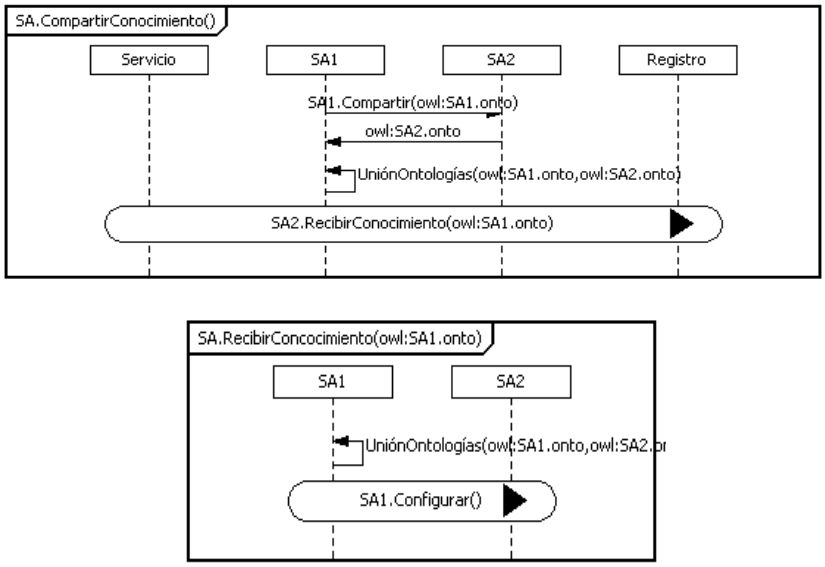

Fig. 6. Agente de Servicio (SA): Recibir Conocimiento.

En las fig. 6 y 7 podemos apreciar los procesos íntimamente relacionados de CompartirConocimiento y RecibirConocimineto que realiza el SA. Estos procesos se realizan cuando el agente de servicio trata de integrarse en un sistema enviando su conocimiento y su estado al resto de agentes. Cada agente realiza una unión de ese

conocimiento en forma de ontologías con el propio que tiene en ese momento. Se detecta qué conceptos y qué instancias son útiles y están relacionadas con el servicio que se gestiona y así se obtiene la, anteriormente comentada, visión del mundo particular de cada SA.

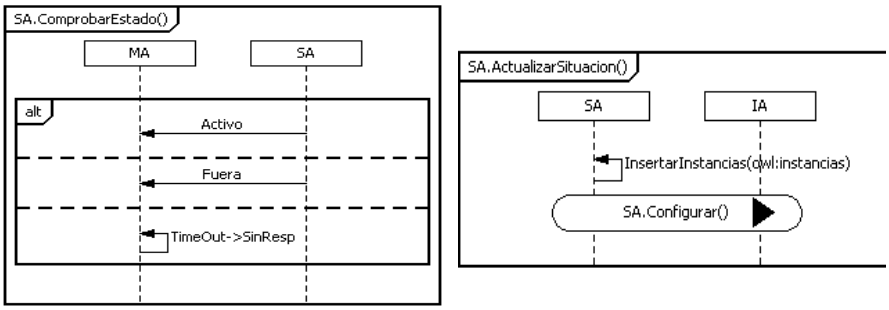

Fig. 7. Agente de Servicio (SA): ComprobarEstado y ActualizarSituación

En la fig. 7 se refleja cómo se lleva a cabo la relación con el MA y el IA. En el caso de la monitorización, el SA informará del estado de servicio. El IA invocará al SA cuando un administrador haya insertado datos de configuración de forma manual. Lo único que debe hacer el SA es actualizar su ABox con esos nuevos datos y luego ejecutar las configuraciones oportunas sobre su servicio.

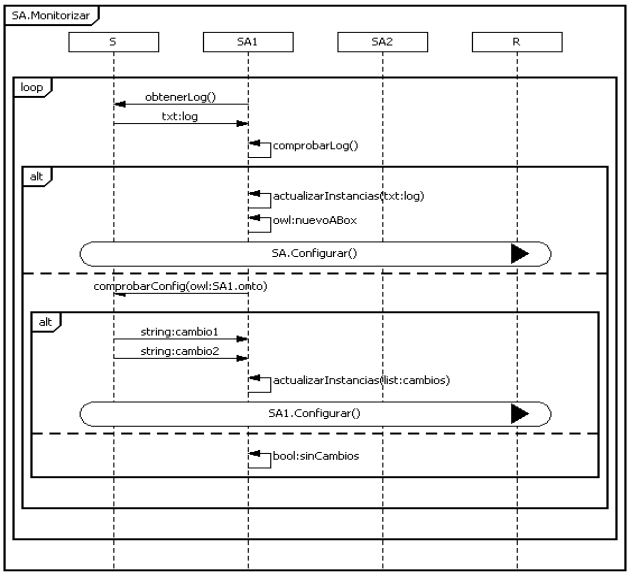

Fig. 8. Agente de Servicio (SA): Monitorizar.

Un servicio, por cualquier motivo, puede sufrir cambios en su estado. Estos cambios pueden quedar reflejados en un log de servicio o ser detectados con la ejecución de ciertos comandos según las características concretas de cada servicio de

red. El agente de servicio se encarga de la monitorización de estas cuestiones (fig 8) para actuar ante cualquier cambio con una actualización del estado (ABox) y la ejecución de las acciones de configuración pertinentes.

4.2 Ontología

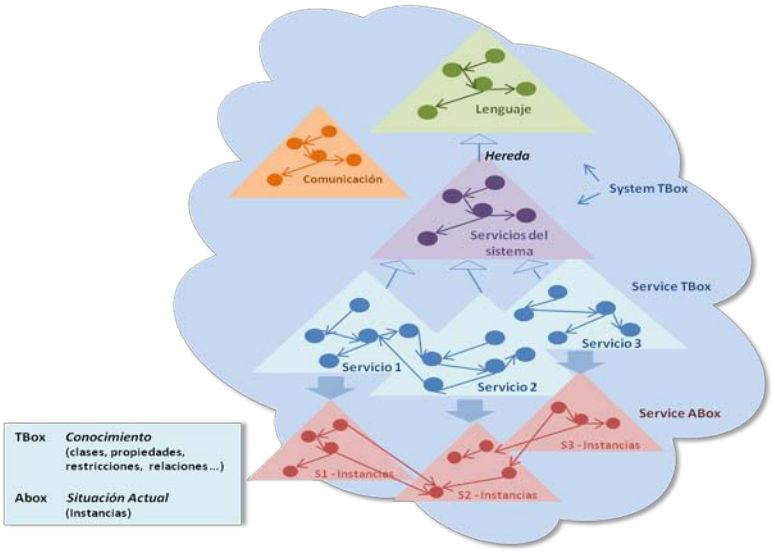

Fig. 9. Ontología

A pesar de que el objetivo de este artículo no es proponer una ontología para el modelo propuesto, es necesario señalar que para ofrecer un alto nivel de escalabilidad se debe permitir que la ontología pueda ser ampliada. Por este motivo, es fundamental establecer una estructura por niveles en la cual, basándonos en conceptos de un nivel superior, se pueda ampliar el conocimiento añadiendo conocimiento en niveles inferiores. En la fig. 9, podemos apreciar una propuesta para este tipo de estructura jerárquica: establecemos un primer nivel para definir conceptos de lenguaje de gestión, en un segundo nivel definiríamos los servicios que actualmente puede gestionar el sistema, en el siguiente definiríamos el comportamiento de cada servicio y, por último, en el último nivel tendríamos el estado de cada servicio.

5 Conclusiones

Se ha presentado un sistema multiagente basado en una ontología que pretende dotar de capacidad auto-gestión a los servicios de red. El diseño presentado del sistema, permitirá que la ontología se pueda gestionar de forma dinámica, ampliando aquellos conceptos no definidos en un primer momento cuando sea necesario. Esto dota de un alto nivel de escalabilidad al modelo presentado. La otra aportación fundamental de este trabajo, es la característica de proactividad de los agentes de servicio ante los

cambios de situación, el intercambio de conocimiento que realizan que posibilita que, en cierta manera, un agente aprenda sobre las funciones y sobre la situación de otro y reaccione en consecuencia.

Este trabajo representa un primer paso para la consecución de un sistema de servicios plenamente auto-gestionable, evidentemente, se requiere de una ontología que sea capaz de aplicar las ideas aquí plasmadas. Una ontología suficientemente general como para abarcar las funcionalidades de cualquier servicio de red, teniendo, a su vez, la capacidad de reflejar las funcionalidades concretas de de un software concreto, con sus acciones y especificaciones concretas. Por otra parte, el sistema multiagente se habrá de implementar sobre una arquitectura de ámbito general para demostrar su funcionamiento. Ambas líneas de trabajo se encuentran actualmente en desarrollo y se plantean como temas abiertos de investigación en los que se presentarán futuras propuestas.

Referencias

1. Schubert P., Leimstoll U.: The Importance of ICT: An Empirical Study in Swiss SMEs. 19th Bled eConference, Bled, Slovenia, June 5 - 7, 2006
2. Instituto Nacional de Estadística. Indicadores del sector TIC. [http://www.ine.es/jaxi/menu.do?type=pcaxis&path=%2Ft14%2Fp197%2Fe01&file=inebase&L=0]
3. Thompson M.: ICT, Power and Developmental discourse: a critical analysis. Proceedings of the Working Conference on Global and Organizational Discourse About Information Technology, Barcelona, Spain in December 2002. Cap 17, 345-375
4. Simple Network Management Protocol, IETF. [http://tools.ietf.org/html/rfc3584]
5. Desktop Management Interface, DMTF. [http://www.dmtf.org/standards/dmi/]
6. Web-Based Enterprise Management, DMTF. [http://www.dmtf.org/standards/wbem]
7. Jeong, M.S., Kim, K.H., Kwon, J.H., Park, J.T. CORBS/CMIP: Gateway Service Scheme for CORBA/TMN Integration. Knom Review, Vol.2, No. 1, 1999, pp. 55-62
8. Aschemann, G., Mohr, T., Ruppert, M.: Integration of SNMP into a CORBA and Web-Based Management Environment. Proceedings in Kommunikation in Verteilten Systemen (KiVS) 1999
9. NetConf Protocol, IETF [http://tools.ietf.org/html/rfc4741]
10. Xu H., Xiao D.: An Integration of Ontology-based and Policy-based Network Management for Automation. International Conference on Computational Intelligence for Modelling Control and Automation. 2006
11. Van der Meer, S., Jennings, B., O'Sullivan, D., Lewis, D., Agoulmine, N.: Ontology based policy mobility for pervasive computing. Proceedings of 12th Workshop of the HP Open University Association (HP-OVUA 2005) Porto, Portugal, July 10–13, 2005 pp. 211–224 (2005)
12. Berners-Lee T.: The Semantic Web. XML2000, Washington DC, diciembre 2000. [http://www.w3.org/2000/Talks/1206-xml2k-tbl/]
13. Gomez-Perez A. et al.: Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. 2004
14. López de Vergara J. E., Villagrá V. A., Berrocal J.: Applying the WebOntology Language to management information definitions. IEEE Communications Magazine, Vol. 42, Issue 7 (2004) 68-74.

15. Keeney, J., Lewis, D., O'Sullivan, D., Roelens, A., Boran, A., Richardson, R.: Runtime semantic interoperability for gathering ontology-based network context. Proceedings of IEEE/IFIP Network Operations and Management Symposium (NOMS 2006), Vancouver, Canada. 3–7 April 2006, pp. 56–65
16. Serrano, J.M., Serrat, J., Strassner, J., Cox, G., Carroll, R., ´Foghlu´ M.: Services management using context information, ontologies and the policy-based management paradigm: towards integrated management in autonomic communications. Proceedings of 1st IEEE Workshop on Autonomic Communications and Network Management (ACNM'07), Munich, Germany, May 2007
17. Jennings, N.R. "Agent-oriented software engineering". In Procedding of the 12th Internacional Conference on Industrial and Engineering Application of AI, 1999.
18. Wooldridge, M., "An introduction to multiagent systems". John Wiley Ed., 2002.
19. Especificaciones FIPA [http://www.fipa.org]