Arquitectura de Agentes de Control con Soporte a la Calidad de Servicio

José Luis Poza Luján, Juan Luis Posadas Yagüe y José E. Simó Ten

***Resumen*—En las arquitecturas distribuidas de control inteligente basadas en agentes, los sistemas de comunicaciones pueden realizar tareas más importantes que la de simples transmisores de información. Para poder enriquecer las conexiones del sistema distribuido, el sistema de comunicaciones debe proporcionar una calidad de servicio que los agentes puedan emplear para tomas algunas decisiones correspondientes a aspectos como la distribución de la información o la movilidad por el sistema. Basándose en las ventajas que puede proporcionar la calidad de servicio, se decidió extender una arquitectura ya desarrollada con el soporte a la calidad de servicio. En la arquitectura expuesta en éste artículo, la responsabilidad de la gestión de las políticas de calidad de servicio, reside en el sistema de comunicaciones y está basada en el estándar DDS propuesto por la OMG.**

***Palabras Clave*— Agentes distribuidos, Arquitectura de control, calidad de servicio.**

I. Introducción

Las arquitecturas de control inteligentes basadas en agentes distribuidos precisan de un soporte de comunicaciones que proporcione, además de los datos propios de las conexiones, unos meta datos a aquellos agentes que los precisen para realizar sus tareas. De entre los meta datos que se puede proporcionar a los agentes se encuentran los parámetros de calidad de servicio de las comunicaciones y son especialmente importantes ya que determinan la calidad de la capacidad de comunicación que puede tener un agente desde un determinado nodo dentro del sistema.

En los sistemas de comunicaciones que dan soporte a las arquitecturas de control distribuido, la calidad de servicio no ha sido uno de los aspectos más importantes. Esta carencia es comprensible en los sistemas de comunicaciones basados en paso de mensajes, ya que son sistemas muy básicos donde no existe una capa intermediaria que pueda llevar un control temporal o del flujo de mensajes. Sin embargo, el empleo de colas de mensajes o el uso de servidores y servicios en los nodos, permite llevar un control de diversos aspectos de la comunicación orientados a obtener unos parámetros de rendimiento que puedan ser empleados para aumentar la eficiencia del sistema.

Jose Luis Poza Luján. Instituto de Automática e Informática Industrial. Universidad Politécnica de Valencia.
E-mail: jopolu@disca.upv.es

Juan Luis Posadas Yagüe, José E. Simó Ten. Instituto de Automática e Informática Industrial. Universidad Politécnica de Valencia.

Existen gran cantidad de sistemas de comunicaciones que dan soporte a diversos sistemas de control, para los sistemas complejos los modelos más adecuados son los basados en el paradigma de colas de mensajes con publicación y suscripción [1].

Para cuantificar las prestaciones, tanto de los nodos en los que se ubican los agentes, como las de los medios de comunicación que emplean, suele emplearse una serie de parámetros en los que basar las mediciones, y consecuente mente las decisiones que tomar. Generalmente se habla de calidad de servicio (QoS) al conjunto de parámetros que gestionan las comunicaciones entre los componentes del sistema distribuido.

En la actualidad uno de los desafíos tecnológicos más relevantes es la gestión de la calidad de servicio punto a punto para la ejecución de sistemas de control. Este aspecto de los sistemas distribuidos va más allá de la distribución de información con requerimientos de tiempo real ya que involucra consideraciones sobre la disponibilidad de recursos computacionales, seguridad, distribución y cooperación de los algoritmos de control, estabilidad, y rendimiento de las tareas de control.

Actualmente la tendencia en las arquitecturas de control consiste en aislar la tecnología: objetos, componentes, agentes del middleware. Uno de los ejemplos de implementación más recientes es el modelo SWE (Sensor Web Enablement) propuesto por la OGC (Open Geospatial Consortium), aplicable a los sistemas que se basen en el uso de sensores para la obtención de información [2].

En las comunicaciones la tendencia más actual es el empleo de sistemas basados en el paradigma de publicación-suscripción. Entre estos sistemas, la OMG (Object Management Group) ha propuesto el modelo DDS (Data Distribution Service). Este modelo cubre desde los sistemas de tiempo real estricto, hasta los sistemas sin necesidades de tiempo real [3]. Además ofrece una gestión muy interesante de la calidad de servicio.

El trabajo presentado en este artículo, se ha centrado en el uso de la calidad de servicio de los sistemas distribuidos de agentes para optimizar la gestión de la redundancia en la información. El diseño de la arquitectura está basado en los modelos SWE y DDS. El siguiente apartado realiza un breve análisis de requisitos de las arquitecturas. A continuación se realiza una revisión de los sistemas DDS y SWE. Seguidamente se describe la arquitectura desarrollada y un sencillo ejemplo de la gestión de la redundancia por medio de la calidad de servicio. Se finaliza con unas breves conclusiones.

II. ARQUITECTURAS DE AGENTES, COMUNICACIONES Y CALIDAD DE SERVICIO

A. *Arquitecturas*

Existen un gran número de arquitecturas de control basadas en agentes [4]. Generalmente se basan en el paradigma reactivo o deliberativo, aunque el paradigma más empleado en las arquitecturas actuales es el híbrido, ya que aprovecha las ventajas de velocidad del paradigma reactivo, con el soporte a la inteligencia que ofrece el paradigma deliberativo.

La organización básica del control en cada capa suele hacerse a partir de sistemas multiagente, donde los agentes se transfieren la información a través de algún sistema de comunicaciones con una mayor o menor adaptación a la arquitectura en el que se aplica [5]. La comunicación entre las capas suele ser responsabilidad de sistemas dedicados y genéricos como JMS [6], o MSMQ [7]. Sin embargo, la especificidad en los objetivos de cada sistema hace que, en ocasiones se deba requerir del sistema de comunicaciones unas capacidades que los sistemas genéricos no ofrecen.

B. *Comunicaciones*

En la mayor parte de los sistemas de comunicaciones que dan soporte a una arquitectura multiagente distribuida, debe existir un módulo de conectividad que haga transparente la interconexión entre los agentes. Cuando éste módulo no pertenece a las aplicaciones a las que ofrece sus servicios de comunicación, se le conoce como "middleware", traducido en ocasiones por "soporte lógico de intermediación". No obstante suele aceptarse el término "software de conectividad" aunque está asimilado el término en inglés. La capa intermedia suele ser un software situado entre los servicios de la red y las aplicaciones; esta capa es la responsable de de proporcionar los servicios, necesarios para aumentar la efectividad de la comunicación, entre ellos están la identificación, autenticación, autorización, la estructuración jerárquica o la movilidad de los agentes.

Existe una gran cantidad de entornos de herramientas que dan soporte a las comunicaciones en los sistemas de control inteligente. Algunos de los entornos son genéricos y se emplean para la comunicación de una gran cantidad de sistemas distribuidos, mientras que otros son específicos de una arquitectura de control. Cuando se desarrolla un sistema multiagente distribuido, es habitual que la interfaz de comunicaciones de los agentes se adapte a los sistemas de comunicaciones. Si los agentes de control emplean un paradigma orientado a objetos suele emplearse CORBA [8]. En el caso de tener que comunicar sistemas basados en agentes, es habitual emplear FIPA [9] o KQML [10].

C. *Calidad de servicio*

La calidad de servicio es un concepto que trata de definir los parámetros por los que evaluar un servicio ofrecido. Desde el punto de vista del procesamiento, la calidad de servicio representa el conjunto de las características tanto cuantitativas como cualitativas de un sistema distribuido necesarias para alcanzar las funcionalidades requeridas por una aplicación [11].

Desde el punto de vista de los mensajes, se define como el conjunto de requisitos del servicio que debe cumplir la red en el transporte de un flujo [12]. Es habitual que los parámetros, por medio de los cuales se definen las calidades de servicio y las políticas que las gestionan, sean específicos a los sistemas en los que se aplica la calidad de servicio. Por ejemplo, en el ámbito de las redes de comunicaciones, la calidad de servicio se entiende como la capacidad de asegurar una tasa de datos en la red, también conocida como ancho de banda, un retardo o una variación del retardo, mientras que en otros ámbitos, como el procesamiento distribuido, conceptos como el ancho de banda, son más difíciles de ubicar.

III. FUNDAMENTO DE LA ARQUITECTURA: SWE Y DDS

A. *Control: SWE*

En todas las arquitecturas de control, se hace necesario disponer de los datos de los sensores, así como de tener la capacidad de procesarlos y de tomar las decisiones de control que enviar a los actuadores. Para lograr realizar todo, desde el punto de vista de la red, se desarrolló Sensor Web Enablement (SWE) como parte de una iniciativa del OpenGis Consortium (OGC) con el propósito de disponer de los datos de sensores vía Servicios Web. El modelo SWE fue propuesto por la OGC en 2004. El modelo propuesto está siendo actualmente empleado por muchas organizaciones, especialmente la NASA y sistemas de meteorología. En la actualidad está tomando una gran importancia en cuanto a su uso para la monitorización y gestión de redes de sensores. El objetivo fundamental de SWE es disponer del marco de referencia para el conjunto de estándares que posibilitan la explotación de sistemas y/o sensores conectados en Internet [13]. SWE divide las funcionalidades en diversos elementos que se relacionan entre sí. Estas especificaciones van desde los componentes de proceso hasta las especificaciones de la información que se debe transmitir entre componentes, todas en XML. Los componentes de SWE se dividen en dos grupos: modelos de información y de servicios. Los modelos de información son especificaciones XML con las que se intercambia información en el sistema, mientras que los servicios son los componentes de control responsables de trabajar con los modelos.

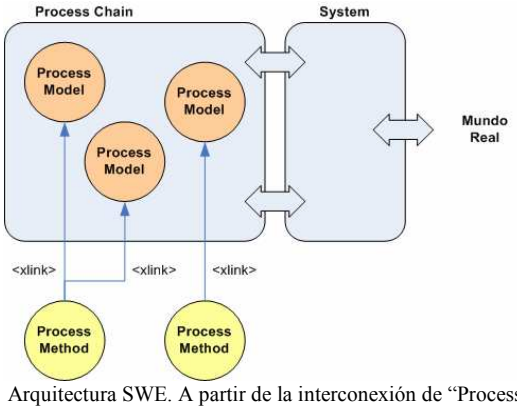

Fig. 1. Arquitectura SWE. A partir de la interconexión de "Process Model" en uno o varios "Process Chain" se forman los componentes que reciben la información del mundo real, la procesan y generan la acción de control.

En el caso de la arquitectura desarrollada, se utiliza el modelo de servicios para adaptarlo al control. Aunque el ámbito original de aplicación de SWE es el de las redes de sensores, el soporte a procesamiento que ofrecen se acerca mucho a la funcionalidad requerida por los agentes en una arquitectura de control inteligente. El procesamiento se basa en unos componentes interconectados [14] que por medio de funciones proporcionan un procesado de la información entrante y ofrecen el resultado. Desde el punto de vista de SWE un componente es un proceso físico atómico que transforma información. Ejemplos sencillos de componentes son sensores, actuadotes o filtros de procesos físicos. Ejemplos complejos de componentes, son núcleos de control que fusionan diversas fuentes para tomar decisiones que trasmitir a otros núcleos de control, para formar un sistema más complejo, y así componer el control básico de un agente.

Tal como se ve en la figura 1, un "Process Model" es un bloque atómico de proceso, normalmente empleado dentro de una composición llamada "Process Chain" que es más compleja. Además un "Process Model" está asociado a un "Process Method" que define el interfaz y la forma de ejecución del "Process Model", además de definir sus entradas, salidas y parámetros de funcionamiento.

El modelo planteado por SWE es especialmente interesante, si se tiene en cuenta que permite especificar patrones de proceso, por medio de los "ProcessMethod" que actúan a modo de plantilla de control. Este "ProcessMethod" constituye el núcleo de control que actúa como unidad básica de procesamiento dentro de un agente. Además la utilización de "ProcessChain" compuestos por diferentes "ProcessModel" interconectados entre sí, pero realizados a partir de los mismos "PrrocessMethod" permite una reutilización de éstos, y además una escalabilidad muy interesante cara a implementar agentes de control complejos a partir de piezas muy básicas. En esta implementación compleja, la capacidad de interconexión entre los diferentes "Process" será muy importante ya que determinará el potencial de intercambio de información, aunque podrá traer algunos problemas como la redundancia en la información, la repetición de acciones de control o la formación de ciclos no deseados de conexiones dentro de una agente control. El sistema que se base en SWE deberá contemplar estos aspectos

B. Comunicaciones: DDS

Data Distribution Service (DDS) es una especificación para sistemas de distribución de datos según el modelo de publicación-suscripción. Este modelo conecta a productores de información (publicadores) con consumidores (suscriptores) desacoplados en tiempo, espacio y flujo [15]. DDS Proporciona un modelo independiente de la plataforma orientado a sistemas distribuidos de tiempo real. La configuración de las comunicaciones se realiza mediante calidades de servicio (QoS). El concepto de QoS se emplea como la forma de describir el comportamiento de un servicio en función de unos parámetros.

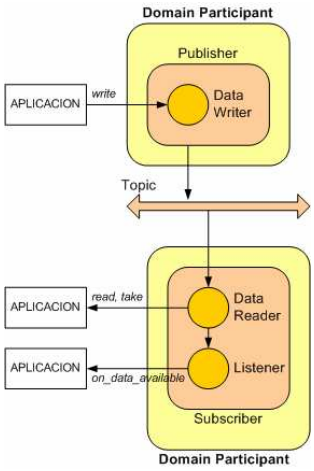

Fig. 2. Arquitectura DCPS. Las aplicaciones se comunican por medio de "DataWriters", "DataReaders" y "Listeners", gestionados por un "Publisher" o un "Subscriberr" y con un "Topic" como referente de canal común.

DDS se estructura formalmente en dos componentes el Data-Centric Publish-Subscribe (DCPS) que es responsable de la distribución de los datos, y el Data Local Recostruction Layer (DLRL) que es la capa que adapta los datos a las aplicaciones locales. Ésta última capa no es necesaria, especialmente en el caso de que las aplicaciones estén implementadas a partir de los objetos de la capa DCPS. DCPS tiene una gran cantidad de componentes en su especificación formal, sin embargo, los básicos son los que se observan en la figura 2. Cuando una aplicación desea publicar una información, lo hace escribiendo en un "Topic", a través de un objeto "Publisher", éste objeto reside en un "DomainParticipant" que gestiona tanto los objetos "Publisher" como los "Subscriber" o "Listener", que son los responsables de recibir los mensajes Los objetos "Publisher" proporcionan los mensajes a petición de la aplicación, mientras que los objetos "Listener" avisan a la aplicación de la llegada de un mensaje. En este último caso la iniciativa de la comunicación la toma el sistema de comunicaciones y no la aplicación.

A todos los objetos del modelo DCPS se les puede asociar una serie de políticas de calidades de servicio que gestionarán las comunicaciones en diversas áreas. Actualmente la especificación contempla 22 políticas de calidad de servicio. Estas políticas se agrupan en áreas que cubren la gestión temporal de los mensajes, gestión del flujo de mensajes, gestión de los componentes de comunicación y gestión de los meta datos. Para obtener más detalles de las políticas de gestión de calidad de servicio contempladas se puede consultar [16].

Un asunto importante es que las calidades de servicio que se solicitan por parte de un "Subscriber", deben ser cumplidas por un "Publisher". Para esto se sigue el patrón "Subscriber" solicita y "Publisher" ofrece. Bajo este tipo de patrón, desde el punto de vista del subscriptor, se puede solicitar un valor, y el publicador también podrá ofrecer un valor. El servicio será quien determinará si los requerimientos son compatibles. Los servicios ya negociados pueden ser cambiados aun ya establecida la conexión, pero solo algunos de ellos.

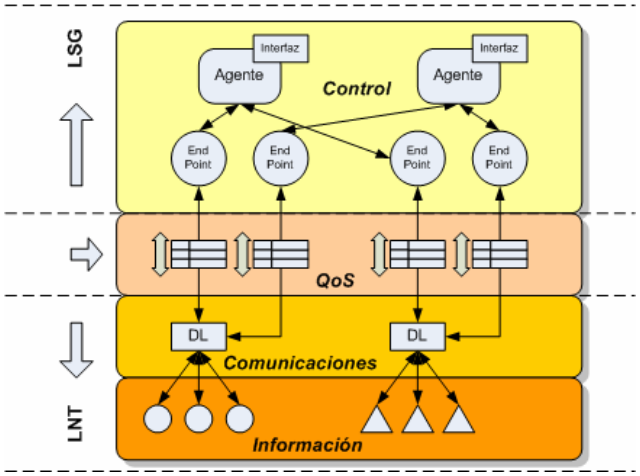

Fig. 3. Arquitectura FSA-Ctrl. En este modelo se distingue la gestión de las comunicaciones (LNT) y la gestión del control del sistema (LSG) con las políticas de calidad de servicio como puntos de conexión entre ambos componentes.

El modelo de comunicaciones basado en DDS está actualmente empleándose en muchos ámbitos, especialmente en sistemas distribuidos de control en tiempo real [17] muy estricto por parte de la US-Navy y son muchas empresas como RTI o PrismTech.

IV. Arquitectura FSA-Ctrl

A. *Arquitectura*

La arquitectura del sistema presentado, se compone de dos áreas diferenciadas, pero no por ello pertenecientes a sistemas distintos: el área de comunicaciones y el área de control, los detalles pueden verse en la figura 3. Los agentes residen en el área de control, y por medio de la calidad de servicio que proporciona el área de comunicaciones, toman las decisiones acerca de su ubicación en el sistema, o de los datos que consideran o no relevantes.

El componente de comunicaciones entre agentes de la arquitectura es el modelo jerárquico con estructura de árbol, en el que los canales de comunicación de cualquier tipo (tanto buses, como conexiones punto a punto) son organizados en una estructura simbólica llamada Logical Namespace Tree (LNT) o árbol de espacio de nombres lógicos [18]. Los componentes de comunicaciones están basados en un modelo desarrollado en el grupo de investigación, cuyos componentes son similares a los componentes de comunicación de DDS, el modelo se conoce como FSA [19].

Para el control se emplea un modelo de agentes interconectados llamado LSG (Logical Sensor Graph) [20]. Este modelo, está basado en el modelo de componentes de proceso de SWE citado anteriormente, y se basa en el uso de unos componentes básicos de control, llamados "Sensores Lógicos" que son los responsables de recibir datos, procesarlos y retransmitirlos, bien al sistema de comunicaciones, bien a otros sensores lógicos. Los sensores lógicos se agrupan, y se interconectan entre ellos, de manera similar a los "Process Chain" de SWE. A continuación se describen con detalle cada uno de estos componentes.

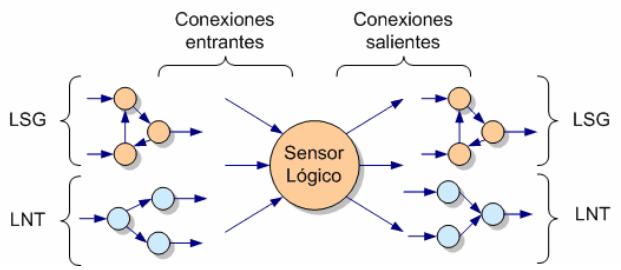

Fig. 4. Arquitectura FSA-Ctrl con las políticas de calidad de servicio como puntos de conexión entre las comunicaciones y el control.

B. *Control*

Los componentes que implementan el control del sistema, se conocen como sensores lógicos, y pueden implementar desde sencillas operaciones atómicas como podrían ser sumas aritméticas o comparaciones lógicas, hasta operaciones complejas como el seguimiento de una trayectoria en la navegación de un robot móvil. Una de las claves del sistema de control implementado, es que se pueden incluir diversos sensores lógicos unos dentro de otros para formar sensores lógicos más complejos que implementen comportamientos más avanzados dentro del sistema de control que se desee implementar. Los agentes del sistema están implementados a partir de la composición e interconexión de los diferentes componentes de control.

La comunicación entre los sensores lógicos puede ser de dos tipos, dependiendo de si se encuentran en localizaciones distintas dentro del sistema distribuido, o si se encuentran en el mismo nodo (figura 4). La primera de ellas se realiza por medio del sistema de comunicación LNT, y la segunda por medio del paso de mensajes internos entre los componentes del LSG. En ambos casos la interfaz de comunicaciones es la misma, lo que hace que la frontera entre las comunicaciones y el control pueda variarse y se facilite en gran medida el movimiento de componentes, o partes de los mismos.

Los mensajes de los sensores de comunicaciones y de procesamiento son similares lo que oculta la ubicación de cada componente en el sistema. Los mensajes están implementados en XML, para acercarse en la medida de lo posible a la especificación de SWE.

C. *Comunicaciones*

Las plataformas que dan soporte a los sistemas de comunicación en las arquitecturas de control, suelen emplear el paradigma de envío de mensajes entre componentes o el de memoria compartida [21]. En ambos casos existen métodos para localizar los componentes dentro del sistema distribuido, pero no se suele realizar una abstracción de los canales de comunicación que facilite el procesamiento de la información, o que incluso oculte por completo los componentes que tratan con la información. En el caso de la arquitectura FSA-Ctrl, se añade una capa que encapsula los detalles de los mensajes entre los agentes o entre los componentes de un agente proporcionando una meta-información adicional acerca del tipo de contenido, o del tipo de mensaje con el que se está comunicando un componente.

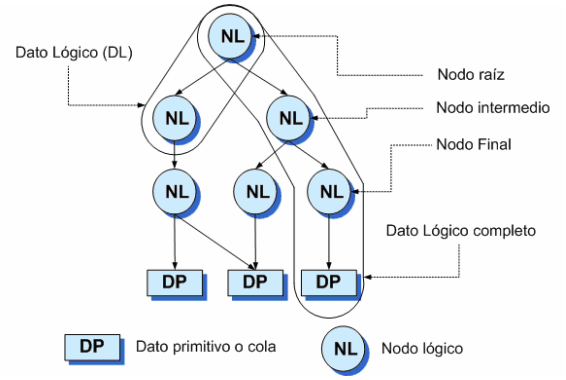

Fig. 5. Componentes y denominación en los árboles de nombres lógicos que representan la denominación y el ámbito de los datos con los que se comunican los componentes

Esta capa se ha denominado LNT o árbol de espacio de nombres lógicos y consiste en cambiar el concepto de dirección, o nombre, del componente en otro concepto llamado dato lógico (DL). Un dato lógico se compone de una secuencia de nombres separados por el símbolo '/', cada nombre se conoce como nodo lógico. Por medio del símbolo '*' se puede referenciar al subárbol que deriva de un dato lógico. Los detalles de la estructura se pueden observar en la figura 5, y las posibles combinaciones de datos lógicos se pueden ver en la figura 6. Por medio de los datos lógicos es posible referenciar una o varias colas de mensajes directamente. El uso de datos lógicos permite obtener una abstracción jerárquica y estructurada de los productores y los consumidores de información.

D. Especificación formal

Una descripción formal de los componentes anteriores puede verse en la figura 7. La clase base es "Entity", a partir de la cual se derivan todas las clases de los componentes de la arquitectura. Cada componente puede tener asociada varias políticas de calidad de servicio, esta relación se realiza en el nivel de la clase base para uniformizar las políticas de calidad de servicio sin que éstas dependan del componente sobre el que se esté aplicando. A partir de la clase "Entity" se derivan las clases contenedoras de los elementos de la arquitectura, estas clases son "Frame" y "FrameEntity", la primera de ellas es la que representa el marco donde residen los adaptadores de comunicación, que gestionan los datos lógicos, y los sensores lógicos. "FrameEntity" es la clase base para los componentes de comunicación: "Adapter" y "LogicalSensor", que son los componentes que contienen los LNT, en el caso del Adaptadores y que forman los LSG, en el caso de los Sensores Lógicos.

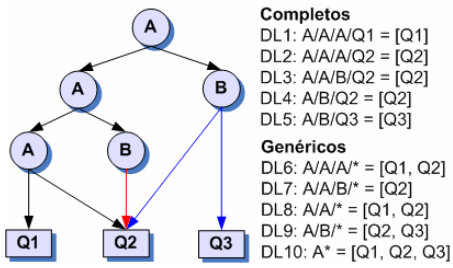

Fig. 6. Ejemplos de datos lógicos.

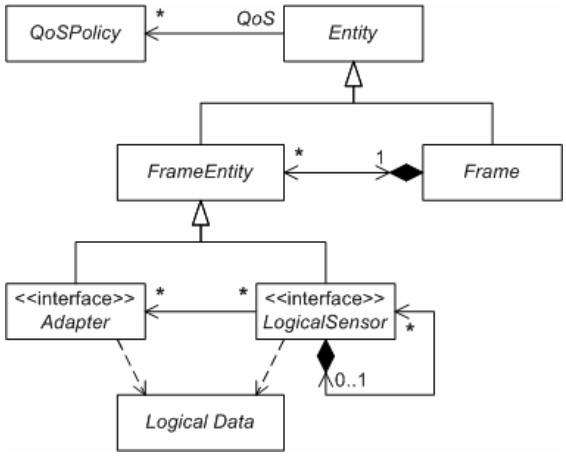

Fig. 7. Diagrama UML de clases de los componentes de la arquitectura FSACtrl.

Para hacer una similitud con el modelo DCPS, los adaptadores hacen el papel de "Publisher" y "Subscriber", mientras que los sensores toman el rol de "DataWriters", "DataReaders" y "Listener", dependiendo de la función que realicen en cada momento. Además de las funciones de comunicación, los sensores también realizan las funciones de control, conteniendo el código que procesan los datos recibidos de los sensores lógicos de comunicación.

En la arquitectura presentada, la negociación de los parámetros de las políticas de la calidad de servicio es responsabilidad de los sensores lógicos, mientras que los componentes de comunicación son los responsables de gestionar dicha calidad. En la figura 8, se observa cómo la calidad de servicio "recubre" los componentes, tanto los de control como los de comunicaciones. Con ello, la calidad de servicio, se convierte en un interfaz común con el que se comunican y gestionan las restricciones del sistema de control. Las políticas de gestión de calidad de servicio empleadas están basadas en las propuestas por DCPS.

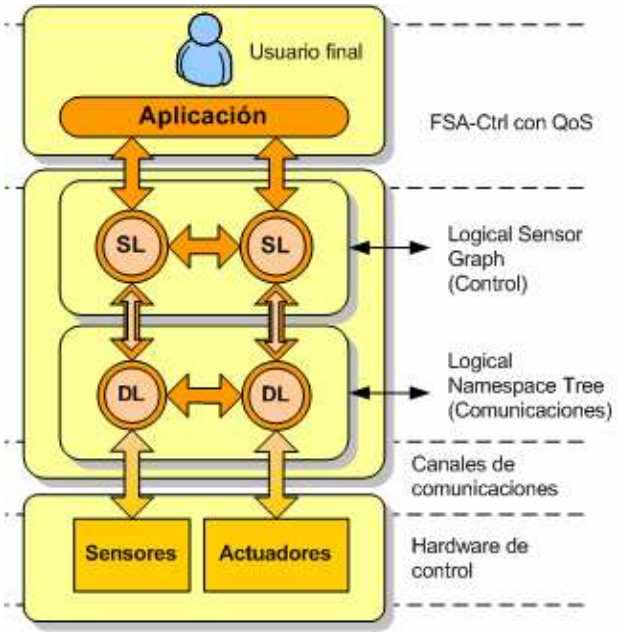

Fig. 8. Ejemplo de cómo la calidad de servicio "envuelve" las conexiones entre los componentes de la arquitectura de control y la representación de las comunicaciones entre los componentes. En este caso, el diálogo entre componentes se realiza siempre en el lenguaje de la calidad de servicio.

V. CASO DE USO

A. *Redundancia en los sistemas distribuidos*

En un sistema distribuido, existen una serie de problemas que se deben tener en cuenta en el momento de configurar el intercambio de información en un sistema d e control. Entre los problemas, uno de los más habituales es la redundancia de información [22]. La cantidad de información que manejan los sensores complejos exige, por parte del sistema de comunicaciones, unas características, tanto de su arquitectura física como lógica que no son habituales en la mayor parte de sistemas. Entre las características está la variabilidad de las características de la información, especialmente en lo que se refiere al volumen, a los requisitos temporales y el tipo de flujo de mensajes que se requiere. El sistema de comunicaciones empleado debe ser capaz de gestionar las características anteriores, lo que por medio de las calidades de servicio puede reducir la dificultad de la implementación.

La arquitectura propuesta, al estar basada en el modelo DCPS permite gestionar los requisitos de tiempo real, tanto estricto como blando, el volumen de información que se debe transmitir y el flujo de mensajes, por lo que es un modelo muy adecuado para la gestión de los sistemas complejos de agentes especialmente en función de las características y requerimientos de información.

La arquitectura propuesta tiene en la calidad de servicio uno de sus pilares básicos. Las políticas de calidad de servicio pueden ofrecerse a otros agentes, o pueden ser requeridas por parte de los agentes, lo que da lugar a la posibilidad de ser negociadas entre ellos para lograr cumplir los requerimientos de control del sistema.

Por medio de la configuración de la calidad de servicio de los agentes involucrados en el sistema, se busca la forma de gestionar la redundancia en la información. Esta gestión consistirá inicialmente en definir y restringir el concepto de información redundante, obtener la forma de detectarla y determinar cómo gestionar la redundancia, tanto la información detectada como las fuentes que la generan.

B. *El papel de la calidad de servicio*

La redundancia en la información tiene múltiples causas, aunque la redundancia temporal es la más habitual. La redundancia temporal consiste en la reiteración de la misma información en diversos instantes de tiempo. La detección de la repetición de la información puede hacerse en cualquiera de los elementos que la transmiten, aunque cuanto más cercana al origen sea la detección de la redundancia, menor impacto tendrá en el sistema.

A modo de ejemplo, en la arquitectura desarrollada, se emplea la combinación de las políticas de calidad de servicio "TimeBasedFilterQoSPolicy", que define un margen temporal en la que se descartan los mensajes intermedios, ya que éstos se consideran redundantes aunque pudiesen contener una información diferente. Esta redundancia es la que se considera redundancia temporal del sistema.

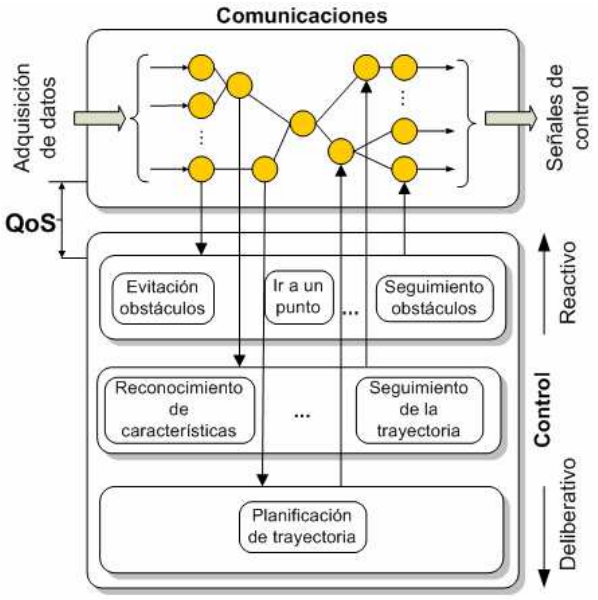

Fig. 9. Sistema de navegación de un robot autónomo desarrollado con la arquitectura presentada. El control se realiza por medio de sensores lógicos que implementan los comportamientos básicos.

C. *Sistema de navegación*

En la figura 9, se observa la arquitectura de navegación de un robot, donde los agentes de control, tanto los más reactivos como los deliberativos, están implementados por medio de sensores lógicos. Por ejemplo, los agentes de evitación de obstáculos, están compuestos por sensores lógicos que se conectan a los datos lógicos que proporcionan los valores de la velocidad de los motores. Estos sensores lógicos se conectan a otros sensores lógicos que fusionan los datos recibidos, los datos fusionados son enviados a sensores lógicos que los comparan con valores de referencia para decidir si se puede navegar a una u otra velocidad. A su vez, el control deliberativo también interviene enviando los datos correspondientes a los agentes de control de evitación de obstáculos.

En la figura 10 puede observarse un ejemplo de la aplicación de diseño de control desarrollada, en el que se ve a la izquierda la jerarquía de comunicaciones y a la derecha los sensores lógicos que componen el agente de evitación de obstáculos.

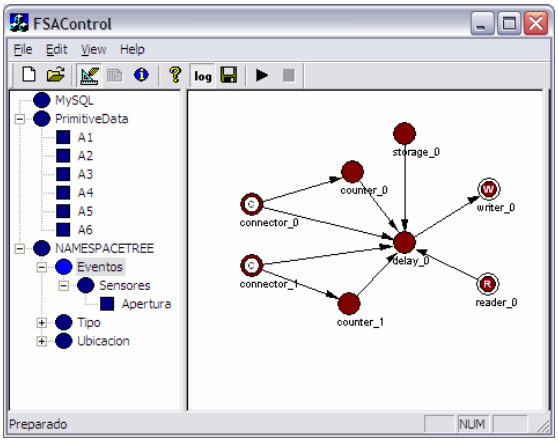

Fig. 10. Aplicación de control que implementa el agente de evitación de obstáculos.

D. *Redundancia temporal*

En la arquitectura propuesta, un sensor lógico puede determinar un valor de "TimeBasedFilterQoSPolicy" para evitar que desde dos datos lógicos diferentes, del LNT o del LSG que representan procesos similares, puedan llegarle datos durante cierto tiempo, con lo que los mensajes descartados son considerados redundantes, aunque contengan información diferente. Por ejemplo, si un sensor de distancia de un robot, proporciona datos con un periodo T_s, pero la duración de la tarea de procesamiento es T_{p1}, cuando $T_{p1} > T_s$, se establece T_{p1} como valor de "TimeBasedFilterQoSPolicy" para la política de calidad de servicio de el sensor lógico responsable del procesado de la información. Sin embargo, si la velocidad del robot móvil aumenta, es posible que el control pase a ser reactivo y en ese caso, la tarea de control cambie. En caso de ser otro agente de control el que toma el mando del movimiento del robot, el periodo de ejecución de su acción de control (T_{p2}) puede ser mucho menor que el anterior, en este caso $T_{p2} < T_s$ y por tanto el valor de "TimeBasedFilterQoSPolicy" para el sensor lógico que procesa la información pasa a ser T_{p2}. En la figura 11, se puede ver el diagrama correspondiente a éste ejemplo.

En el ejemplo anterior, los agentes de control, independientemente de la tarea asignada, sólo actúan cuando se cumple la condición de la calidad de servicio negociada, lo que lleva a que no deban ser responsables de la toma de decisiones en función de la velocidad del vehículo, aspecto que desconocen. Además un cambio en el comportamiento no supone un cambio en un umbral de velocidad sino un cambio en la política de calidad de servicio a la que aceptarán, o no, los mensajes provenientes del sistema de comunicaciones. La combinación de diversas políticas de calidad de servicio proporciona muchas más posibilidades de gestión de agentes para realizar las acciones de control.

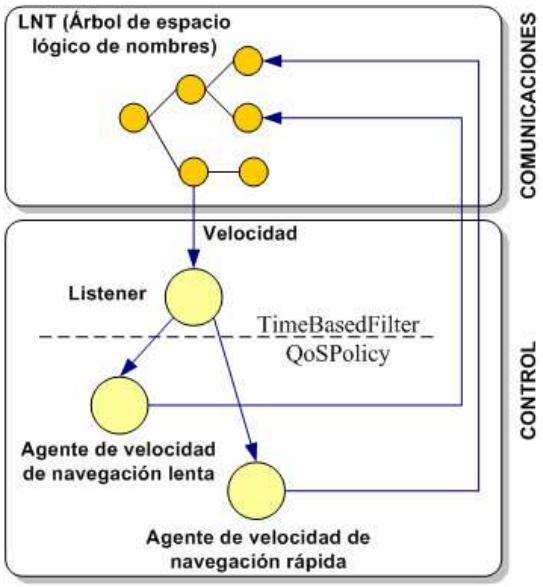

Fig. 11. Ejemplo de cómo la calidad de servicio "decide" el agente que determinará la acción de control a ejecutar, en función de un parámetro de calidad de servicio dependiente de un dato lógico del LNT

VI. CONCLUSIONES

En este artículo se ha presentado la implementación una arquitectura de control inteligente de sistemas distribuidos con soporte a la calidad de servicio. El hecho de que las comunicaciones se basen en el paradigma de publicación-suscripción, y en el estándar DCPS, avalado por OMG, ofrece una base de comunicaciones estandarizada que permite adaptar fácilmente el sistema a otros similares. El control está basado en el estándar SWE, que permite una distribución muy alta de los algoritmos de control.

El uso de políticas de gestión de calidad de servicio, proporciona unos parámetros, por medio de los cuales, poder tomar decisiones en aspectos como la discriminación de la información en función de la calidad de la misma o la toma de decisiones acerca del movimiento de agentes.

El sistema proporciona diversas ventajas, entre ellas una visión estructurada y jerárquica del sistema a controlar. Además la gran capacidad de distribución de los componentes, hace que el sistema pueda ser empleado para la evaluación de la correcta disposición en el sistema distribuido de los componentes. Esto último es importante, ya que el sistema puede llegar a realizar una distribución automática de los componentes en función de unos parámetros de calidad de servicio. Es importante destacar que como método de análisis del control del sistema, esta arquitectura es muy adecuada.

Actualmente se está trabajando en diversas líneas que emplean la arquitectura presentada. En un intento básico de adaptación, se está implementando un sistema domótico donde los parámetros de calidad de servicio se usan para discriminar reenviar información y así evitar sobrecargar los buses internos y la comunicación con los agentes de control que se encuentran en dispositivos conectados en red, como teléfonos móviles y PDAs. Por medio de la implementación en un sistema con bajas restricciones temporales, como es un sistema domótico, se está comprobando la viabilidad de la arquitectura.

En una segunda fase, se pretende dar soporte a un sistema de control de la navegación de un robot móvil. Por medio de la implementación en un sistema con altas restricciones temporales, como es un sistema de robots móviles, se comprobará el comportamiento bajo condiciones de tiempo real, donde la detección de la redundancia en los datos, por parte de los agentes, puede suponer la ejecución de tareas en determinadas condiciones de restricción temporal. Finalmente, el empleo de varias políticas de calidad de servicio para determinar la ubicación de los agentes, puede dar lugar a un sistema que se ajuste automáticamente en función de unos requisitos de calidad de servicio.

AGRADECIMIENTOS

La arquitectura desarrollada es parte del proyecto coordinado KERTROL: Núcleo de Control en los Sistemas Empotrados Fuertemente Interconectados. Ministerio de Educación y Ciencia. CICYT: DPI2005-09327-C02-01/02.

REFERENCIAS

[1] Matteucci., M. "Publish/Subscribe Middleware for Robotics: Requirements and State of the Art", Technical Report N 2003.3, Dipartimento di Elettronica e Informazione - Politecnico di Milano, Milano I, 2003.

[2] OGC Sensor Web Enablement: Overview and High Level Architecture. OGC White Paper. OGC 06-050r2. Edited by Mike Botts, George Percivall, Carl Reed, and John Davidson. 2006.

[3] OMG. "Data Distribution Service for Real-Time Systems, v1.1." Document formal/2005-12-04. (Dec. 2005).

[4] Muller, J. P. Architectures and applications of intelligent agents: a survey. The Knowledge Engineering Review, 13(4), (1998) 353-380.

[5] Gaddah A., and Kunz, T. A survey of middleware paradigms for mobile computing. Technical Report SCE-03-16. Carleton University Systems and Computing Engineering. (2003)

[6] M. Hapner, R. Sharma, J. Fialli, and K. Stout, JMS specification, Sun Microsystems Inc., 4150 Network Circle, Santa Clara, CA 95054 USA, 1.1 edition, April 2002.

[7] Lewis, R.: Advanced Messaging Applications with MSMQ and MQ Series. Que Publishing, 1999.

[8] OMG. Real-Time Corba Specification version 1.1 formal/02-08-02, (2002).

[9] Foundation for Intelligent Physical Agents: FIPA 97 Specfication. Part 2, Agent Communication Language. (1997)

[10] T. Finin, Y. Labrou, and J. Mayfeld, "KQML as an agent communication language," in Software Agents (J. M. Bradshaw, ed.), ch. 14, pp. 291{316, AAAI Press. The MIT Press, 1997.

[11] Andreas Vogel, Brigitte Kerherve, Gregor von Bochmann, Jan Gecsei. Distributed Multimedia and QoS: A Survey. Vol.2,. No. 2, 1995, pp.10-19.

[12] Crawley, E.; Nair, R; Rajagopalan, B. "RFC 2386: A Framework for QoS-based Routing in the Internet". August. 1998, pp. 1-37, XP002219363

[13] Botts M, Percivall G, Reed C, Davidson J, (2006) OGC®. Sensor Web Enablement: Overview and High Level Architecture, OpenGIS Consortium Inc.

[14] Alexandre Robin and Michael E. Botts. Creation of Specific SensorML Process Models. White Paper Earth System Science Center (NSSTC). University of Alabama in Huntsville (UAH). 2006.

[15] P. Houston. Building distributed applications with message queuing middleware – white paper. Technical report, Microsoft Corporation, 1998.

[16] Pardo-Castellote, G. OMG Data-Distribution Service: architectural overview. Distributed Computing Systems Workshops, 2003. Proceedings. 23rd International Conference. Volume , Issue , 19-22 May 2003 Page(s): 200 - 206

[17] Pardo-Castellote, G., Schneider, S., and Hamilton, M. NDDS: The real-time publish subscribe network. In IEEE Workshop on Middleware for Distributed Real-Time Systems and Services (San Francisco, CA, 1997), pp. 222–232.

[18] Poza, J. L. Posadas, J. L. Simó, J.E. and Benet, G. Hierarchical communication system to manage maps in mobile robot navigation. International Conference on Automation, Control and Instrumentation. (2006)

[19] Posadas, J. L. Pérez, P. Simó, J. E. Benet, G. and Blanes, F. Communications structure for sensory data in mobile robots. Engineering Applications of Artificial Intelligence. Volume 15, Issues 3-4, June-August 2002, Pages 341-350.

[20] José Luis Poza Luján, Juan Luis Posadas Yagüe, José E. Simó Ten y Ginés Benet Gilabert. Especificación de Agentes Distribuidos para una Arquitectura de Control Inteligente. 6th International Workshop on Practical Applications of Agents and Multiagent Systems (IWPAAMS07). pp. In Press. Salamanca, Spain (2007)

[21] José Luis Poza, Juan Luis Posadas, José Enrique Simó, Pascual Pérez, Miguel Albero. Modelo de arquitectura de comunicaciones Frame-Sensor-Adapter (FSA). XXV Jornadas de Automática (JA2004) (ISBN 84-688-7460-4). Ciudad Real, 2004

[22] Coulouris, G., Dollimore, J., Kindberg, T. Sistemas Distribuidos, Conceptos y diseño. Tercera Edición. Addison Wesley. 2001.