UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf
Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Coordinación y comunicación entre agentes físicos basados en la plataforma SPADE

Proyecto Final de Carrera

Ingeniería Informática

Autor: Bellver González, Miguel

Director: Carrascosa Casamayor, Carlos

Co-director: Palanca Camara, Javier

30 de septiembre de 2015

Resumen

El objetivo del presente proyecto es desarrollar un sistema multi-agente sobre la plataforma SPADE, basada en python, que incorpore tanto agentes software como agentes físicos. La parte física del sistema estará soportada principalmente por dos robots NAO. Estos son uno de los robots humanoides de Aldebaran robotics, con 25 grados de libertad y gran variedad de sensores diferentes y comportamientos predefinidos, que lo convierten en una herramienta muy potente. Además, este robot esta diseñado con un aspecto estéticamente agradable, especialmente pensado para la interacción con niños y personas mayores.

Como caso de estudio a desarrollar, proponemos el mapeado de un espacio físico desconocido por parte de dos robots NAO. Este mapeado se llevará a cabo de manera colaborativa entre los dos robots, de manera que cuando uno de ellos explore un área notifique los resultados al otro, manteniendo así un mapa global y las correspondientes copias locales de cada robot actualizado con la información conjunta. La elección de SPADE para el desarrollo se vio muy condicionada por el hecho de que NAO ejecute python de manera nativa.

Palabras clave: NAO, Sistema Multi-agente, Colaboración, SPADE, Coordinación.

Índice general

Capítulo 1

Introducción

1.1. Motivación

Este proyecto está englobado dentro de la línea de trabajo con agentes físicos del Grupo de Tecnología Informática Inteligencia Artificial (GTI-IA), en la rama de Sistemas Multi-Agente (SMA) y más específicamente en el estudio de la interacción entre humanos y agentes software.

En un mundo cada día más tecnológico, las máquinas nos rodean y es inevitable tener que tratar día a día en mayor o menor medida con ellas, tanto es así que muchas están ya totalmente integradas en nuestra vida. En este escenario se busca hacer estas interacciones cada vez más fluidas, cómodas y naturales. Dentro de esta corriente se encuentran los robots humanoides, máquinas de aspecto amigable y familiar para nosotros que ayudan a eliminar barreras y a integrarlas mejor en nuestro entorno, ampliando además las funciones que pueden desempeñar. Una vez el primer paso a este acercamiento entre el humano y el ordenador está dado por medio de la apariencia, se nos presenta un segundo paso importante, poder comunicarnos e interactuar con ellos de forma natural. Para ello se debe buscar la manera de que adopten comportamiento y capacidades que nos sean afines y los acerquen más a nosotros. Esta es una línea de avance deseable. En el desarrollo de este proyecto nos centraremos en el campo de la navegación autónoma, consistente en dar capacidad a la maquina para orientarse y desplazarse por un entorno físico si necesidad de supervisión por parte de un ser humano.

1.2. Objetivos

Con este proyecto se pretende conseguir un sistema multi-agente robusto que, independientemente del número de agentes físicos disponibles, sea capaz de realizar el mapa de una habitación cerrada de extensión indeterminada que puede contener obstáculos de tipo caja de dimensiones mínimas predeterminadas y dimensiones máximas indeterminadas. Además deben estar separados una distancia mínima entre ellos. El mapa será construido a modo de plano en dos dimensiones de la habitación representando una vista cenital de la misma. Para el desarrollo del proyecto hemos utilizado el robot humanoide bípedo de la empresa Aldebaran robotics NAO como agente físico. Esta decisión ha estado motivada por la elección de SPADE como nuestra plataforma de agentes, puesto que esta plataforma se ejecuta sobre python y los robots NAO lo pueden ejecutar como código nativo. A su vez el hecho de elegir esta plataforma y no otra reside en su versatilidad, que sea libre y que su protocolo de comunicación sea XMPP. Esto hace que diferentes tipos de agentes físicos puedan colaborar en la resolución de las tareas con pequeñas modificaciones en el código del agente debidas a las diferencias en el soporte físico de cada uno. Así, aunque la colaboración durante este desarrollo ha sido entre robots NAO, no esta limitada a ellos siendo capaces de comunicarse entre si y colaborar diferentes tipos de robots y agentes software.

Capítulo 2

Estado del arte

El presente proyecto consiste en la realización de un sistema multi-agente, basado en SPADE (Plataforma de agentes basa en mensajería instantánea y en su protocolo XMPP) utilizando como agentes físicos los robots humanoides NAO. Por consiguiente, seria conveniente profundizar más en estos conceptos.

2.1. Sistemas multi-agente

Un sistema multi-agente (SMA) es un sistema compuesto por varios agentes inteligentes que interactúan entre ellos en un determinado entorno, siendo capaces de alcanzar colectivamente metas que serian difíciles de realizar por un agente individual o por sistemas monolíticos.

El bloque fundamental de construcción de un sistema multi-agente son los agentes. Un agente inteligente, es una entidad capaz de percibir su entorno, procesar tales percepciones y responder o actuar en su entorno de manera racional. También son capaces de comunicarse a través de un mecanismo de comunicación interproceso, utilizando protocolos de comunicación.

Dependiendo de la complejidad del agente, podemos distinguir tres tipos básicos:

- Agentes pasivos: Son agentes muy simples que no disponen de un objetivo. Puede tratarse de un obstáculo o cualquier objeto del entorno.

- Agentes activos: Son agentes que disponen de un objetivo simple, por ejemplo esperar un mensaje y reenviarlo.

- Agentes complejos: Son agentes activos cuyo objetivo no es tan simple.

Son capaces de realizar cálculos complejos, así como de llevar a cabo un amplio abanico de funciones.

Los agentes en un sistema multi-agente tienen varias características importantes:

- Autonomía: los agentes son al menos parcialmente autónomos.
- Visión local: ningún agente tiene una visión global del sistema, o el sistema es demasiado complejo para un agente para hacer un uso práctico de esos conocimientos.
- Descentralización: no hay un agente de control designado.

Aunque cuando hablamos de agentes inteligentes nos solemos referir a agentes software, también podría tratarse de robots o incluso de seres humanos.

2.1.1. FIPA

La Foundation for Intelligent Physical Agents (FIPA) es un organismo para el desarrollo y establecimiento de estándares de software para agentes heterogéneos que interactúan y sistemas basados en agentes.

FIPA fue fundada como una organización sin animo de lucro en 1996 con el objetivo de definir un conjunto completo de normas para la implementación de sistemas en los que se puedan ejecutar agentes y especificación de cómo los propios agentes se deben comunicar e interactuar.

La organización se disolvió en 2005 y se creo un comité de estándares IEEE en su lugar.

Una plataforma de agentes es un framework para la creación y gestión del ciclo de vida de un sistema multi-agente. Para ser compatibles con FIPA las plataformas de agentes requieren cuatro requisitos mínimos:

- Un canal de comunicación de los agentes o ACC, que permita a los agentes y al plataforma comunicarse entre ellos.
- Un sistema de gestión de agentes o AMS. Un mecanismo que permite a los agentes registrarse en la plataforma y ser alcanzables para el contacto. Constituye una especie de paginas blancas de la plataforma.
- Un facilitador de directorios o DF. Es un tipo de servicio publico en el cual los agentes publican los servicios que ofrecen. El DF seria pues como las paginas amarillas de la plataforma.

- Soporte para el Lenguaje de comunicación de agentes (ACL) de FIPA, que es un lenguaje común de comunicación para todos los agentes, cuya sintaxis se basa en XML.

FIPA-ACL

Un mensaje FIPA-ACL contiene un conjunto de uno o más parámetros. Precisar que parámetros son necesarios para una comunicación entre agentes efectiva varia de acuerdo a la situación; el único parámetro que es mandatorio en todos los mensajes ACL es la performativa, además se espera que la mayoría de mensajes contengan los campos emisor, receptor y contenido.

Si un agente no reconoce o es incapaz de procesar uno o más parámetros o valores de estos, puede responder con el apropiado mensaje de no-comprension.

Implementaciones específicas son libres de incluir parámetros de mensajería definidos por el usuario que no se encuentren entre los especificados por FIPA-ACL. La semántica de estos parámetros no esta definida por FIPA y la compatibilidad con FIPA no requiere una interpretación particular de esos para metros. El antecedente "X-"debe ser usado para los nombres de esos parámetros adicionales.

2.2. SPADE

SPADE es una plataforma de agentes basada en la tecnología XMPP/Jabber de mensajería instantánea. Esta tecnología ofrece herramientas y facilidades a la hora de construir un sistema multi-agente como son: un canal de comunicación, el concepto de usuarios (agentes) y de servidor (plataforma) y un protocolo de comunicación extensible basado en XML como es FIPA-ACL. Existen otras plataformas de agentes, pero SPADE es la primera en basarse en XMPP.

Las principales características de la plataforma son:

- Soporte del estándar FIPA mediante el protocolo de mensajería instantánea XMPP (Agentes AMS y DF incluidos)

- Notificación de presencia entre agentes.

- Organizaciones Virtuales basadas en el protocolo de multi-conferencia MUC

- Comunicación P2P entre agentes.
- Invocación remota de servicios entre agentes usando el estándar XML-RPC.
- Procesadores de lenguajes de contenido en SL0 y RDF
- Modelo de agente BDI basado en Conocimiento, Deseos e Intenciones.
- Modelo de comportamientos: Cíclicos, Periódicos, Timeout, una ejecución, máquina de estados finita y basado en eventos.
- Soporte de comunicación con otras plataformas mediante diversos protocolos de transporte: JADE (via HTTP o XMPP) y SIMBA.
- Publicación y suscripción de eventos mediante el estándar PubSub.
- Interfaz gráfica basada en web.

2.2.1. Mensajería instantánea

La mensajería instantánea es una forma de comunicación en tiempo real entre dos o más personas basada en texto. El texto es enviado por medio de dispositivos conectados por una red.

La mayoría de sistemas de mensajería instantánea incluyen una serie de funciones básicas:

- Mostrar los usuarios conectados.
- Mostrar un estado de usuario de entre una serie de estados.
- Mostrar un mensaje de estado personalizado.
- Registrar y borrar usuarios de una lista de contactos propia.
- Agrupar los contactos según diferentes criterios.
- Usar una imagen identificativa o avatar.
- Mandar ficheros.

Algunos clientes de mensajería instantánea ofrecen, además de charlas entre dos usuarios, la posibilidad de mantener chats grupales.

Estas herramientas son de gran utilidad a la hora de diseñar un SMA, dadas las posibilidades de coordinación que ofrecen para los agentes.

XMPP

"Extensible Messaging and Presence Protocol" (XMPP), en sus inicios conocido como Jabber, es un protocolo abierto inspirado en XML para mensajería instantánea y información de presencia en tiempo real.

Con el protocolo XMPP queda establecida una plataforma para el intercambio de datos XML que puede ser usada en aplicaciones de mensajería instantánea. Las características en cuanto a adaptabilidad y sencillez del XML son heredadas de este modo por el protocolo XMPP.

Creado para ser extensible, XMPP ha sido usado para sistemas de PubSub, señalización para VoIP, transmisión de archivos y muchas más aplicaciones.

Entre las ventajas de XMPP se encuentran:

- Descentralización: La arquitectura de las redes XMPP es similar a la del correo electrónico; cualquiera puede poner en marcha su propio servidor XMPP, sin que haya ningún servidor central.

- Estándares abiertos: La Internet Engineering Force ha formalizado el protocolo XMPP como una tecnología de mensajería instantánea estándar. El desarrollo de esta tecnología no está ligado a ninguna empresa en concreto y no requiere el pago de regalías.

- Seguridad: Los servidores XMPP pueden estar aislados de la red pública XMPP, y poseen robustos sistemas de seguridad (como SASL y TLS).

- Flexibilidad: Se pueden hacer funcionalidades a medida sobre XMPP; para mantener la interoperabilidad, las extensiones más comunes son gestionadas por la XMPP Software Foundation.

- Historia: Las tecnologías XMPP llevan usándose desde 1998. Existen múltiples implementaciones de los estándares XMPP para clientes, servidores, componentes y bibliotecas, con el apoyo de importantes compañías.

Por otra parte, cuenta también con alguna desventaja:

- Sobrecarga de datos de presencia: Típicamente cerca de un 70 % del tráfico entre servidores son datos de presencia, y cerca de un 60 % de estos son transmisiones redundantes. Se están estudiando nuevos protocolos para aliviar este problema.

- Escalabilidad: XMPP también sufre el mismo problema de redundancia en los servicios de chatroom y de suscripción.
- Sin datos binarios: XMPP es codificado como un único y largo documento XML, lo que hace imposible entregar datos binarios sin modificar. De todas formas, las transferencias de archivos se han solucionado usando otros protocolos como HTTP.

Tras varios años de su existencia, XMPP ha sido adoptado por empresas como Facebook, Tuenti, WhatsApp Messenger y Nimbuzz, entre otras, para su servicio de chat.

Jabber ID's

Cada usuario en la red XMPP tiene un único identificador (Jabber ID, normalmente abreviado como JID). Para evitar la necesidad de un servidor central con una lista exhaustiva de identificadores, el Jabber ID está estructurado como una dirección de correo electrónico, con un nombre de usuario y una dirección DNS para el servidor en el que reside el usuario, separado por un signo @. Un identificador Jabber sería algo como nombredeusuario@dominio.com.

Como un usuario puede querer identificarse desde distintos lugares, el servidor permite al cliente especificar una cadena de referencia conocida como recurso, que identifica el cliente que está utilizando el usuario (por ejemplo: casa, trabajo, portátil, etc.). Esto será incluido en el JID añadiendo un carácter '/' seguido del nombre del recurso. Cada recurso debe tener especificado un valor numérico de prioridad. Por ejemplo el JID completo de la cuenta del trabajo del usuario sería: nombredeusuario@dominio.com/trabajo. Los mensajes de la forma nombredeusuario@dominio.com serán dirigidos al cliente con mayor prioridad, y los de la forma nombredeusuario@dominio.com/trabajo serán dirigidos al cliente del trabajo. Los JID sin la parte del nombre de usuario también son válidos y se utilizan para enviar mensajes de sistema y control.

En el caso de SPADE cada agente tendría su propio JID que sería su identificador dentro de la plataforma.

2.3. NAO

NAO es un robot humanoide programable y autónomo de 58cm de alto, desarrollado por Aldebaran Robotics con el cual pretenden conseguir un

compañero encantador e interactivo en casa. El desarrollo del robot comenzó en 2004. En 2007, NAO sustituyó al perro robot Aibo de Sony como la plataforma estándar para la Robocup (Robot Soccer World Cup"), un concurso internacional de robótica.

La Edición Académica de NAO fue desarrollada para las universidades y laboratorios con fines de investigación y educación. Fue lanzado a las instituciones en 2008, y se puso a disposición del público antes de 2011. El robot ha entrado ya su uso en numerosas instituciones académicas de todo el mundo, incluyendo la Universidad de Tokio, el IIT Kanpur de la India y la Universidad del Rey Fahd de Petróleo y Minerales de Arabia Saudita. En 2014 se lanzó una versión mejorada del robot, que ofrece una mayor síntesis de voz multilingüe.

NAO ve usando dos cámaras de 1,22 mp, que pueden capturar 30 imágenes por segundo. La primera cámara, situada en la frente de NAO, escanea el horizonte, mientras la segunda, situada a la altura de la boca, escanea los alrededores inmediatos. El software te permite recuperar imágenes y vídeos

de lo que NAO ve. Pero los ojos solo son útiles si puedes interpretar lo que ves. Por eso NAO contiene una serie de algoritmos para detectar y reconocer caras y formas. NAO puede reconocer quien está hablándole o encontrar una pelota u objetos mas complejos. Estos algoritmos están especialmente desarrollados pensando constantemente en usar el menor número de recursos de procesador. Además el SDK de NAO permite desarrollar tus propios módulos para interactuar con OpenCV (La libreria Open Source de visión por computador, originalmente desarrollada por Intel)

NAO usa cuatro micrófonos direccionales para detectar sonidos. Uno de los principales propósitos de los robots humanoides es interactuar con personas. La localización de sonido permite al robot identificar la dirección de los sonidos. Cuando una fuente cercana emite un sonido, cada uno de los cuatro micrófonos de NAO recibe la onda de sonido en tiempos ligeramente diferentes.

Utilizando las capacidades de audio de NAO, un amplio rango de experimentos e investigaciones pueden tener lugar en los campos de la comunicación y la interacción humano-robot.

Además de cámaras y micrófonos, NAO esta equipado con sensores capacitativos posicionados en tres secciones en lo alto de su cabeza y en sus manos.

Se puede por tanto dar información a NAO mediante tacto: presionando una vez para decirle que se apague, por ejemplo, o usando los sensores como una serie de botones para activar una acción asociada.

NAO esta equipado con dos canales de sonar: dos transmisores y dos receptores. Estos permiten a NAO estimar la distancia de los obstáculos de su entorno. El rango de detección es de 1cm a 3 m. A menos de 15 cm, no devuelve información de distancia, solo sabe que hay un obstáculo presente.

NAO actualmente soporta conexión Wi-fi y Ethernet, que son los dos protocolos de comunicación de red más extendidos.

Esta combinación de tecnologías dota a NAO de la habilidad de detectar su entorno. ¿Puede interpretar lo que detecta? Aquí es donde el software integrado en NAO entra en acción. Aldebaran ha creado un sistema operativo propio, NAOqi, permitiendo al pequeño humanoide entender e interpretar la información que recibe de sus sensores. Este sistema operativo de ejecuta sobre la CPU principal de NAO, un Intel ATOM 1,6GHz situado en la cabeza,

junto con una CPU secundaria localizada en el torso.

Además el robot viene con un paquete de software que incluye una herramienta gráfica de programación (Aldebarán Chorégraphe), un software de simulación y un kit de desarrollo de software. NAO también es compatible con el Microsoft Robotics Studio, Cyberbotics Webots y el Gostai Urbi Studio.

Capítulo 3

Trabajo desarrollado

Se pretende desarrollar un sistema multi-agente compuesto tanto de agentes software como de agentes físicos. Estos últimos ejecutaran su código onboard sobre robots humanoide NAO. El cometido del sistema es realizar el mapa de una habitación cerrada de extensión indeterminada que puede contener obstáculos de tipo caja. Esta tarea sera llevada a cabo por los agentes de manera colaborativa. El mapa será construido a modo de plano en dos dimensiones de la habitación representando una vista cenital de la misma. Para el desarrollo del proyecto se utilizará la plataforma de agentes SPADE. Esta decisión esta motivada por el hecho de que la plataforma este construida en python, el único lenguaje que pueden ejecutar los NAO como código nativo.

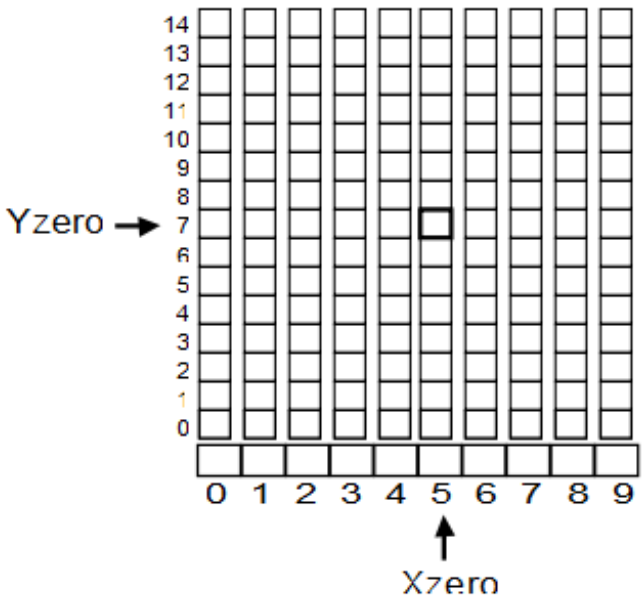
3.1. Escenario

El escenario esta compuesto por tres agentes SPADE. Los agentes deben poder manejar y almacenar un mapa de la habitación a reconocer, además este mapa debe ser fácilmente actualizable y comunicable al resto de agentes, así como ser escalable, dado el tamaño indeterminado de la habitación.

Los dos primeros agentes se ejecutarán en un robot NAO cada uno y serán los encargados de controlarlos a la hora de recorrer la habitación y ejecutar las funciones que modifiquen el mapa. El tercer agente, denominado agente de control, sera un agente software que se ejecutara junto con el servidor. Su función principal sera la de servir de enlace entre los agentes físicos y el usuario por medio de un chat de texto. Será también el encargado de traducir la información del mapa a un formato visual comprensible por el usuario y actualizarlo en tiempo real conforme los agentes físicos vayan explorando.

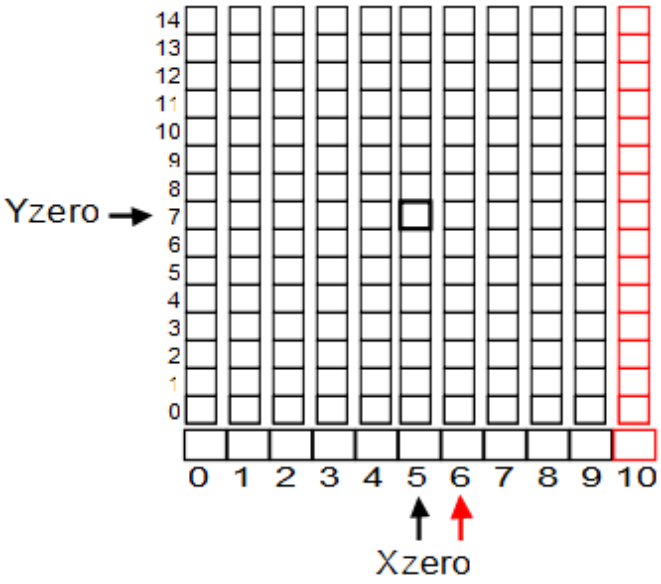
3.2. Mapa

Haciendo referencia al mapa, contamos con una estructura de datos basada en una lista de listas, donde el indice de la primera lista indicaría la coordenada en el eje x y el indice de la segunda lista indicaría la coordenada en el eje y. Como nos encontramos en una habitación desconocida y las condiciones iniciales del experimento no dependen de nuestra localización dentro de la misma, la estructura debe ser extensible en ambas direcciones de ambos ejes. Esto supone que no podemos asignar de manera definitiva la coordenada 0,0 de nuestro mapa a la primera posición de la lista en 'y' situada en la primera posición de la lista en 'x'. Para solucionar este problema, se decidió mantener dos variables extras que representarían la posición de la lista 'x' a partir de la cual representaríamos las coordenadas positivas para este eje y anterior a la cual se encontrarían las coordenadas negativas. Lo mismo se hizo para las listas del eje y, se mantiene una variable común a todas las listas que guarde el indice a partir del cual representaremos las 'y' positivas y bajo el cual estarán representadas las 'y' negativas. Este diseño nos garantiza poder expandir el mapa de una forma computacionalmente poco costosa debido a y con intención de, mantener siempre una forma cuadrada.

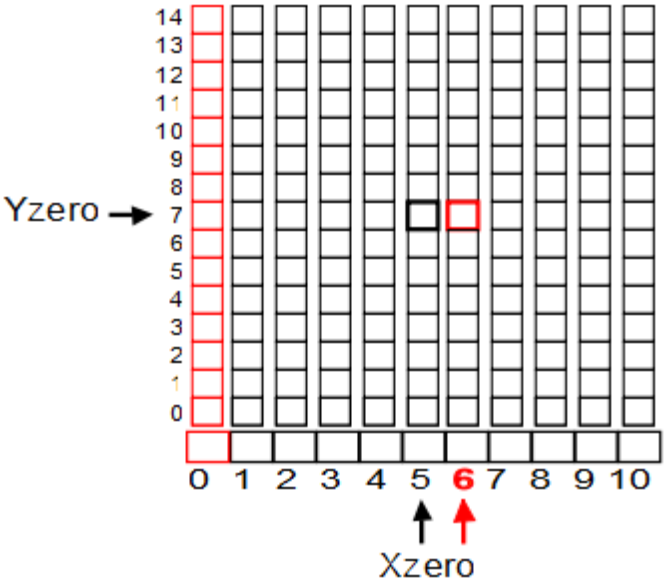


A continuación explicaremos las implicaciones de una expansión del mapa en cada una de las 4 direcciones disponibles:

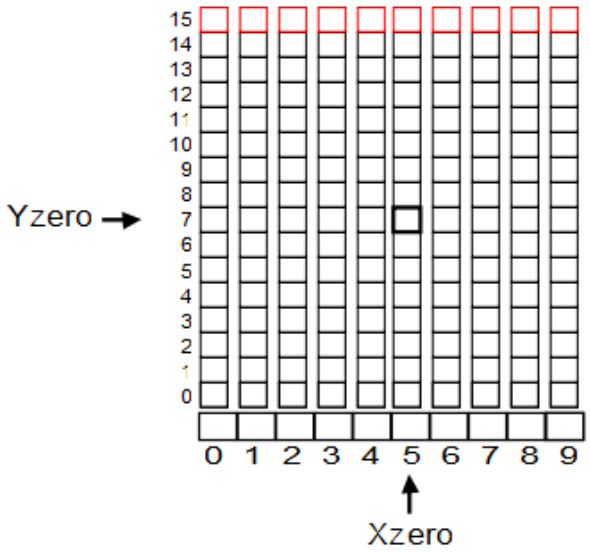
Expansión hacia x positivas: Una expansión en esta dirección implicaría insertar en cola de la lista 'x' tantas listas de tamaño igual al resto de listas de 'y', como unidades quisiéramos expandir, no resultaría en ninguna variación de la variable que indica el 0 en 'x'.



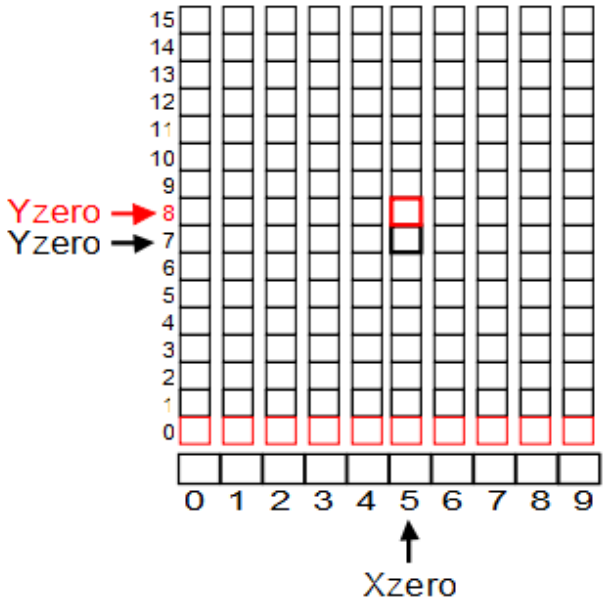
Expansión hacia x negativas: Una expansión en esta dirección implicaría insertar en cabeza de la lista 'x' tantas listas de tamaño igual al resto de listas de 'y', como unidades quisiéramos expandir, resultaría además en un aumento de la variable que indica el 0 en 'x' en el numero de unidades a expandir.



Expansión hacia y positivas: Una expansión en esta dirección implicaría insertar en cola de cada lista 'y' tantas posiciones como unidades quisiéramos expandir, no resultaría en ninguna variación de la variable que indica el 0 en 'y'.



Expansión hacia y negativas: Una expansión en esta dirección implicaría insertar en cabeza de la lista 'y' tantas posiciones como unidades quisiéramos expandir, resultaría además en un aumento de la variable que indica el 0 en 'y' en el numero de unidades a expandir.



Las celdas del mapa estarían marcadas con 5 posibles valores diferentes:

X: Representa una celda de contenido desconocido, es el estado inicial de todas las celdas del mapa. Se marca en color gris medio.

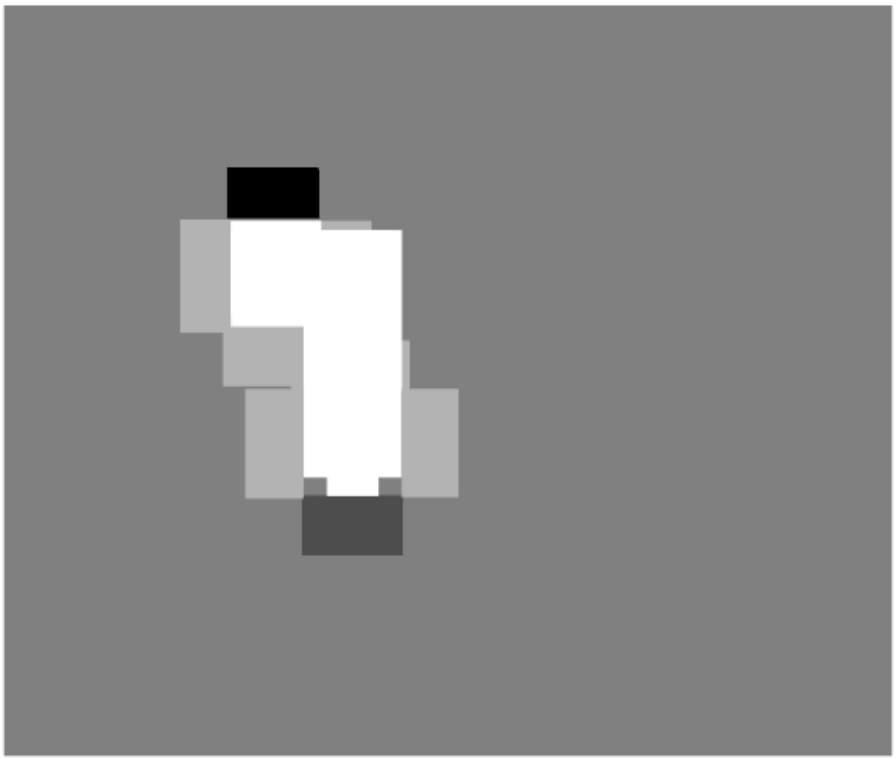
H: Representa las celdas del mapa donde hemos detectado un obstáculo y hemos confirmado que se encuentra allí. Se marca en color negro.

O: Representa las celdas que conocemos con seguridad que se encuentran libres de cualquier obstáculo. Se marca en blanco.

XH: Representa las celdas que creemos pueden estar ocupadas por un obstáculo, sin embargo aun no hemos podido verificarlo. Se marca en color gris oscuro.

XO: Representa las celdas que podemos suponer están libres, sin embargo aún no hemos tenido la oportunidad de comprobarlo. Se marca en color gris claro.

Se puede ver un ejemplo de mapa donde aparece toda la gama de colores a continuación.



La granularidad del mapa sera el cm2. Esta decisión viene motivada por el hecho de que el 1 cm es la mínima unidad de movimiento del NAO y queremos mantener la menor granularidad posible para representa la habitación de la manera más precisa posible.

3.2.1. Mapa local vs. Mapa global.

Como ya comentamos con anterioridad cada NAO dispone de un sistema de coordenadas interno que nos permite saber su posición relativa a su origen interno de coordenadas. Esta herramienta nos permite consultar con facilidad en que punto se encuentra el robot en los momentos que nos interese para marcar un obstáculo frente a el, una zona libre, o para calcular un camino, ahora bien, como también se ha comentado, cada robot dispone de su propio sistema interno. Así pues el mundo que pueden percibir los robots es distinto, o más bien esta desplazado en uno con respecto al otro y dado que deben coordinarse para construir un mapa común, es necesario constituir un sistema de coordenadas externo a los dos, sobre el cual puedan trabajar de manera colaborativa y puedan ver con las mismas referencias. Para conseguir esto debemos implementar alguna forma de que cada robot pueda conocer cual es su posición en el mapa compartido, basándonos unicamente en su coordenada interna y en el hecho de que sus posiciones relativas son conocidas al inicio, como también hemos comentado, la posición inicial de uno de los robots sera tomada como el 0,0 y se notificara al otro su posición en el mapa. Dado que aun con este proceso, al menos uno de los robots no corresponderá sus coordenadas internas con las del mapa compartido, se ha implementado un algoritmo que a partir de la posición inicial del robot en el mapa compartido y en sus coordenadas internas, pueda en todo momento traducir su visión local a el sistema de coordenadas global. Para ello lo primero que debemos hacer es desplazar al nao al origen de coordenadas, y posteriormente, corregir su angulo.

3.3. Agente de control

3.3.1. Coordinación

La coordinación básica entre los tres agentes de este escenario consiste en un intercambio de mensajes referentes a las distintas actualizaciones en la información del mapa, generadas por los agentes físicos que realizan el cambio y recibida por todo el resto de agentes. Fue, por tanto, necesario establecer un estándar de contenido de los mensajes, para que los agentes

pudieran interpretar claramente la información que intercambian entre ellos. Estos mensajes quedan recogidos dentro de la ontología COORDINATION.

Según el inicio del contenido del mensaje distinguiremos cuatro tipos de modificaciones en el mapa:

FREE Bajo este comienzo se encontrarán los mensajes que indiquen el descubrimiento de zonas libres de obstáculos. Con el contenido FREE position X, nos encontraríamos los mensajes que indican una posición del robot, indicada por el parámetro X, la cual esta libre porque el NAO es capaz de estar sobre ella. Con el contenido FREE from X1 to X2, nos encontraríamos los mensajes que indican una ruta en linea recta comprendida entre X1 y X2, por la cual el NAO ha sido capaz de transitar y por tanto esta libre de obstáculos.

OBS Bajo este comienzo se encontrarán los mensajes que indiquen el descubrimiento de un obstáculo frente a un agente. El contenido OBS from X1 to X2 dist D1 to D2, que describen un obstáculo definido entre la posición X1 y la posición X2 y situado a una distancia D1 por la izquierda del agente y una distancia D2 por la derecha del agente.

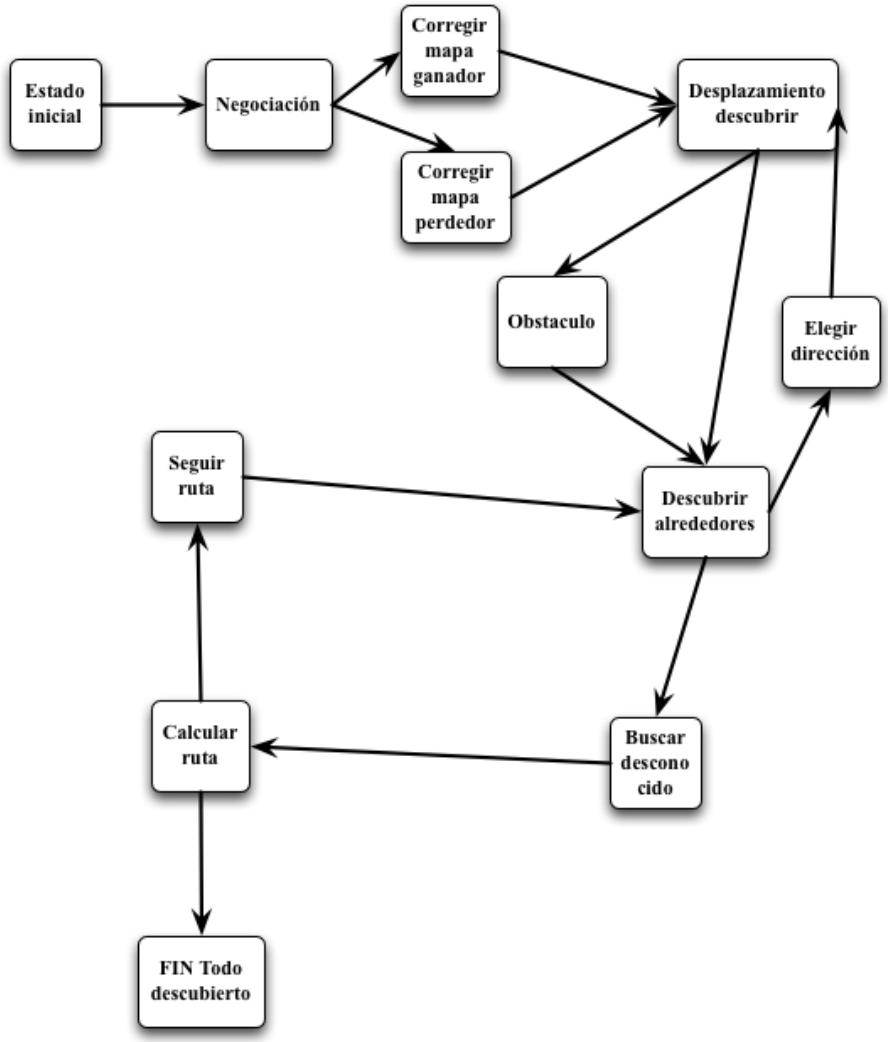
XO Bajo este comienzo se encontrarán los mensajes que indiquen la sospecha obstáculo frente a un agente. El contenido XO POS indica que esta zona se encuentra frente al agente situado en POS.

XH Bajo este comienzo se encontrarán los mensajes que indiquen la sospecha de una zona libre de obstáculos frente a un agente. El contenido XH POS indica que esta zona se encuentra frente al agente situado en POS.

3.4. Agentes físicos

3.4.1. Ciclo de vida

A lo largo de la tarea de mapeado se darán una serie de situaciones que deberemos reconocer y ante las cuales deberemos actuar, a continuación enumeraremos y desarrollaremos estas situaciones, así como también estableceremos las relaciones pertinentes entre ellas.



Estado inicial. Este es el punto de partida del experimento. Nos encontramos en esta situación en el momento en que ambos robots NAO están operativos y con sus respectivos agentes SPADE a la espera de la orden de inicio. Los robots estarán situados en un lugar cualquiera de la habitación objetivo, espalda con espalda uno del otro. Una vez se reciba la orden de inicio, pasaremos a encontrarnos en la situación de inicio del experimento.

Negociación. Nos encontraremos en este punto cuando, partiendo de la situación inicial, emitamos la orden de inicio del experimento. Cuando esto ocurra, los NAO deberán negociar el origen de coordenadas del

mapa. Esta negociación se basara en la consulta de la posición interna de cada robot, tomándose como ganador de la negociación aquel que registre una distancia absoluta menor con respecto a su origen de coordenadas. Llegados a este punto, pasaremos a determinar la relación existente entre las coordenadas internas del agente y las externas. Se explicara el proceso de negociación con más detalle posteriormente.

Corregir mapa ganador. Al finalizar la negociación inicial se tomará la posición del agente ganador como origen de coordenadas del mapa compartido. Sin embargo poseerá una posición interna diferente, que sera la que actualice de manera automática con sus desplazamientos y a la que podremos acceder consultando la posición de memoria asociada. Para conocer la posición real en el entorno en todo momento, basta con obtener en este punto de partida una transformación entre las coordenadas del mundo real y las del mundo interno del agente, que aplicaremos sobre las consultas posteriormente. Una vez establecidas estas transformaciones daremos paso al inicio de la exploración, pasando a la situación de desplazamiento.

Corregir mapa perdedor. Al finalizar la negociación inicial se establecerá la posición del agente perdedor en relación al ganador (Desplazado una distancia en el sentido negativo del eje X y con un angulo de 180º). Sin embargo poseerá una posición interna diferente, que sera la que actualice de manera automática con sus desplazamientos y a la que podremos acceder consultando la posición de memoria asociada. Para conocer la posición real en el entorno en todo momento, basta con obtener en este punto de partida una transformación entre las coordenadas del mundo real y las del mundo interno del agente, que aplicaremos sobre las consultas posteriormente. Una vez establecidas estas transformaciones daremos paso al inicio de la exploración, pasando a la situación de desplazamiento.

Desplazamiento descubrir. Nos encontraremos en esta situación al inicio de la exploración o siempre que establezcamos un nuevo objetivo. Durante esta fase el agente se desplazara en linea recta una distancia prefijada o hasta que encuentre un obstáculo en su camino. En caso de encontrar un obstáculo pasara a gestionar su inclusión en el mapa. En caso de desplazarse la totalidad de la distancia, procederá a descubrir el entorno que le rodea. En ambos casos marcara en el mapa el recorrido efectuado como terreno libre de obstáculo, ya que ha podido transitar por el, y notificara esta transición al resto de agentes para

que la registren también como libre en sus respectivas copias locales del mapa.

Obstáculo. Se dará esta situación siempre que un desplazamiento sea interrumpido por la detección de un obstáculo inminente en la trayectoria. En este caso el agente procederá a marcar en el mapa un obstáculo de tamaño similar al suyo a la distancia a la que lo detecte por medio del uso del sonar y como espacio transitable la distancia entre el y el obstáculo. A continuación notificara al resto de agentes la posición de este obstáculo para que lo marquen en sus respectivos mapas locales. Pasara posteriormente a descubrir el entorno que le rodea. Es posible que pretenda marcar como obstáculo al otro agente, sin embargo hacemos prioritario el estado libre de una casilla con respecto al de marcada como ocupada. Esto es así porque resulta más fiable el hecho de que este libre si hemos podido transitar por ella que la detección de un obstáculo mediante el sonar, sujeto a más variabilidad e interferencias. Así pues en caso de tratarse del otro agente podríamos encontrarnos en dos casos. Caso A: Hemos marcado como obstáculo la posición antes de que el otro agente la marque como libre, cuando el agente la marque como libre al validar su transición por ella sobrescribirá nuestros datos. Caso B: el otro agente y ha marcado la zona como libre, no podremos sobrescribir esa información.

Descubrir alrededores. Esta situación se dará cuando el agente acabe acabe de efectuar un desplazamiento, bien sea exitoso o interrumpido por la detección de un obstáculo. Llegados a este punto el robot girara 360º en etapas de 90º y comprobara sus posiciones adyacentes, marcando si detecta un posible obstáculo o si en una posición hasta ahora desconocida se detecta libre, pudiendo generar dos situaciones: Elegir una dirección adyacente con casillas desconocidas o buscar una zona desconocida más allá de sus adyacentes.

Elegir dirección. Esta situación se plantea cuando después de haber analizado las casillas adyacentes, encuentra alguna zona aún desconocida hacia la cual poder seguir explorando. En caso de disponer de una zona donde sospeche que se encuentra un obstáculo se dirigirá hacia ella. En caso de que todas las zonas adyacentes con un valor desconocido estén detectadas como libres, seleccionara una de ellas tomando como criterio elegir la zona hacia la cual represente un giro menor dirigirse. Ambas opciones generan el paso al estado Desplazamiento descubrir.

Buscar desconocido. Nos encontraremos en esta situación cuando todas

las casillas a nuestro alrededor tengan un valor conocido. En este punto revisaríamos el mapa a nuestro alrededor siguiendo una trayectoria en espiral desde nuestro centro hasta encontrar un puto de valor incierto. Una vez encontrado ese punto, pasaríamos a intentar calcular una ruta hacia él.

Calcular ruta. Siempre que estemos rodeados totalmente por zonas conocidas y tengamos una zona lejana objetivo a la que queramos desplazarnos acabaremos llegando a esta situación. Intentaremos calcular un camino entre nuestra posición y la zona desconocida dada como explicaremos más adelante. En caso de encontrar una ruta factible, pasaremos a seguirla. Si, por otra parte, no encontramos una ruta, podemos asumir que hemos llegado a la condición de final.

Seguir ruta. Nos encontraremos en esta situación siempre que la búsqueda de un camino sea exitosa. Seguiremos el camino devuelto y una vez en la zona comenzaremos a reconocer nuestro alrededor.

FIN Todo descubierto. Llegaremos a este estado siempre que al intentar encontrar una ruta, recorramos todo el mapa y no encontremos ninguna posible. Esta condición marca el final del experimento como explicaremos más adelante.

3.4.2. Negociación

Como ya comentamos anteriormente la decisión de en que lugar de nuestro espacio físico se establecerá el origen de coordenadas se tomará por medio de un proceso de negociación entre los dos agentes físicos presentes en el momento de inicio. Pasaremos a describir este proceso a continuación.

Para distinguir los mensajes propios de la negociación de los otros muchos que puede recibir el agente, estos mensajes serán tratados en un comportamiento propio y serán clasificados dentro d en una ontología NEGOTIATION.

Al recibir la orden de inicio de experimento cada agente consultara su posición en su sistema de coordenadas interno, una vez hecho esto enviaran un mensaje de contenido INIT_NEGO X bajo la ontología anteriormente citada al otro agente, donde X corresponderá a su posición interna. Con el envío de este mensaje dará comienzo el proceso de negociación. En este punto el agente esperará recibir un mensaje similar por parte de su compañero, al recibirlo calculará las distancias absolutas de ambas coordenadas al origen, reconociendo ya como ganador a aquel que presente esta distancia menor.

En caso de que ambos NAO registren idénticas distancias absolutas con respecto a su origen local, se procederá a una segunda fase de negociación, siguiendo el mismo protocolo de envío y espera de mensaje, cada agente generará un número aleatorio y lo transmitirá como STEP2 NUM, donde NUM es el citado número, bajo la ontología antes descrita. Se considerará ganador al que genere el número más alto, en caso de empate se repite este proceso de generación aleatoria hasta conocer un ganador.

Una vez terminado el proceso de negociación ambos robots mandarán al otro un mensaje READY para indicar que son conocedores del resultado y están listos para comenzar el mapeado.

3.5. Condición de final

En los momentos en que alguno de los robots se encuentre rodeado de áreas conocidas, intentará buscar una zona cercana que aún esté por explorar para dirigirse a ella. Si no encuentra ningún área alcanzable desde su posición que sea susceptible de ser explorada, podemos garantizar que el mapa esta ya totalmente explorado. Pero ahora bien, como llega nuestro agente a esa conclusión. El primer paso cuando se encuentra bloqueado en una zona conocida, es buscar el punto desconocido más cercano, esto lo consigue consultando las posiciones del mapa en espiral desde su centro hasta encontrar una coordenada desconocida o indeterminada. Una vez se ha hecho con esa coordenada, intentara trazar un camino hacia ella. Para llevar esto a cabo, el primer paso es realizar una simplificación del mapa, de manera que las casillas sean transitables de manera unitaria por el robot. Cada casilla del nuevo mapa representa una cuadricula de 30x30 del mapa original. Se marcaran como libres si todas sus casillas están libres, como obstáculo si todas sus casillas son libres u obstáculos, como potencialmente libre si todas sus casillas son libres o potencialmente libres, como potencialmente obstáculo si todas sus casillas son obstáculos o potencialmente obstáculos y como desconocido en cualquier otro caso. Situaremos al robot y el punto objetivo en el mapa simplificado y lanzaremos un algoritmo de encaminamiento, si durante el trayecto encontrásemos otro punto desconocido u indeterminado, nos quedaríamos ese camino y se convertiría en el nuevo objetivo, se analizara el camino obtenido y se procesaran los diferentes nodos que representen movimientos en linea recta para reconstruir el camino que deberá realizar el NAO. Posteriormente el robot llevara a cabo ese camino, procesado como una serie de desplazamientos y giros. En caso de no poder encontrar un camino valido a ningún punto desconocido o indeterminado, nos contrariamos en la situa-

ción de una habitación delimitada por obstáculos y rellena de obstáculos o libres. En otras palabras una habitación completamente explorada.

Capítulo 4

Conclusiones

Durante el presente proyecto se ha desarrollado un sistema multi-agente sobre la plataforma SPADE que incorpora tanto agentes software como agentes físicos. La parte física del sistema ha estado soportada principalmente por dos robots NAO.

Se ha conseguido desarrollar el caso de estudio propuesto consistente en el mapeado de un espacio físico desconocido por parte de los agentes. Se ha llevado a cabo el diseño de un protocolo de contenido de mensajes que permita la comunicación y la resolución del problema de manera colaborativa entre los robots. Se ha desarrollado a su vez una estructura de mapa de tamaño ampliable en tiempo de ejecución y de contenido modificable basado en una cola de prioridad de actualizaciones. Además este mapa se mantiene almacenado como copias locales en cada agente. También se han desarrollado los comportamientos encargados de gestionar la coordinación entre agentes, posibilitar la comunicación con el usuario, gestionar negociaciones y transitar entre las diferentes fases de la exploración.

Derivado de la elección de la plataforma SPADE, y dada su versatilidad, este sistema es ampliable en cuanto al numero de agentes físicos, y su tipo, que pueden colaborar en la resolución de las tareas con pequeñas modificaciones en el código del agente. Así, aunque la colaboración durante este desarrollo ha sido entre robots NAO, no está limitada a ellos siendo capaces de comunicarse entre si y colaborar diferentes tipos de robots y agentes software.

En el desarrollo de este proyecto nos hemos encontrado con dificultades a la hora de gestionar las concordancias entre el mundo físico y la representación que pretendíamos mantener de él. Esta problemática derivada del trabajo el mundo físico. Lejos de la precisión y el control del entorno al que solemos

estar acostumbrados cuando trabajamos con simulaciones, en un entorno real hay que lidiar con los errores de precisión en sensores y actuadores e intentar corregir y minimizar este error constantemente.

Se ha validado el desarrollo del sistema mediante pruebas individuales de cada caso, y enfrentando a los agentes a una serie de habitaciones simples. Los agentes han satisfecho los requerimientos y han llevado la tarea a buen termino en todos los casos.

Capítulo 5

Bibliografía

https://www.aldebaran.com/en

http://doc.aldebaran.com/1-14/index.html

https://es.wikipedia.org/wiki/SPADE

https://github.com/javipalanca/spade

https://es.wikipedia.org/wiki/Foundation_for_Intelligent_Physical_Agents

http://www.fipa.org/specs/fipa00061/SC00061G.html

https://es.wikipedia.org/wiki/Mensajería_instantánea

https://en.wikipedia.org/wiki/XMPP

https://es.wikipedia.org/wiki/Sistema_multi-agente

https://en.wikipedia.org/wiki/Multi-agent_system

https://es.wikipedia.org/wiki/Nao_(robot)