Robótica y Aprendizaje Bioinspirado :: Robótica multi-agente

Miguel Angel Astor Romero

Semestre 2-2015 :: Versión 1 - 2016-01-19 mar

1. Agentes inteligentes

1.1. Definición

Existen múltiples definiciones para el término "Agente Inteligente" en informática y computación. Dos de las definiciones analizadas por Ana Mas en [1] son las siguientes:

Wooldridge Un agente es un sistema informático situado en un entorno y que es capaz de realizar acciones de forma autónoma para conseguir sus objetivos de diseño [5].

Garijo Una entidad de software robusta y adaptable que puede funcionar en distintos entornos o plataformas computacionales y es capaz de realizar de forma "inteligente" y autónoma distintos objetivos intercambiando información con el entorno, o con otros agentes humanos o computacionales [4].

Como se puede ver ambas definiciones coinciden en el hecho de que un agente es un ente de software caracterizado por poseer autonomía y capacidad de comunicación. La autonomía se refiere principalmente al hecho de que un agente debería ser capaz de tomar decisiones para lograr sus objetivos, sin necesidad de intervención de otros agentes externos, sean humanos o de software. Por otra parte, la capacidad de comunicarse con el entorno le da al agente la posibilidad de obtener información del mismo, así como también el poder notificar a otros agentes de los resultados que obtenga durante su ejecución. Esta comunicación puede llevarse a cabo a travéz de una red de computadoras mediante algún protocolo de comunicación, o mediante el uso de sensores y efectores físicos.

Aparte de las ya mencionadas autonomía y capacidad de comunicación, Garijo enuncia las siguientes como las características fundamentales de un agente de software:

Funcionamiento continuo Un agente no debe detenerse al producir un resultado.

Robustéz Un agente debe poseer tolerancia a fallos y debe ser capaz de tomar decisiones con información incompleta.

Adaptabilidad Capacidad de realizar objetivos y tareas en distintos dominios de forma incremental y flexible.

Otras caracterísitcas deseables para un agente son:

Capacidad de aprendizaje El agente puede ser capaz de aprender de su entorno para mejorar sus procesos de toma de deciciones.

Movilidad El agente debe ser capaz de desplazarse por su entorno.

Sobre el tema de la movilidad, esta se refiere a la capacidad del agente de migrar de forma autónoma y transparente entre distintos nodos de un sistema distribuido, o como su desplazamiento físico en un entorno real en el caso de agentes robóticos.

1.1.1. Tipos de agentes

Los agentes inteligentes se pueden clasificar de múltiples formas según varios criterios. Una clasificación particular es el llamado "modelo de las vocales" de Demazeau [6], el cual organiza a los distintos tipos de agentes inteligentes según los siguientes 5 grupos:

A (Agente) Según características individuales de cada agente.

Se distinguen dos tipos de agentes en este grupo: los agentes reactivos y los agentes cognitivos. Los agentes reactivos son agentes orientados a eventos los cuales consisten en un ciclo de captura de eventos y ejecución de métodos espécificos para reaccionar a los distintos tipos de eventos reconocidos por el agente. Estos métodos de reacción pueden estar condicionados por el estado interno del agente y por lo general son capaces de modificar dicho estado, convirtiendo al agente en una máquina de estados finitos que procesa eventos externos, generalmente en tiempo real. Estos agentes no suelen poseer capacidades de aprendizaje, aunque pueden apoyarse en mecanisnmos como las redes neuronales en lugar de en máquinas de estados.

Por su parte los agentes cognitivos utilizan mecanismos simbólicos para representar su conocimiento y se basan fundamentalmente en técnicas de razonamiento y aprendizaje para tomar decisiones. Estos agentes utilizan un ciclo de percepción, asimilación, razonamiento y acción. En las fases de percepción y asimilación el agente captura información de su entorno y la transforma a una representación simbólica para integrar dicha percepción a su base de conocimiento actual. Luego el agente determina que acción debe tomar para cumplir su objetivo en la fase de razonamiento, para finalmente llevar a cabo una acción o comunicar un resultado.

E (Entorno) Según el entorno en el que funcionan los agentes.

Se entiende por entorno del agente a toda la plataforma computacional que da soporte al agente, incluyendo el hardware, sistema operativo, arquitectura de red y software de utilidad que coexiste con el agente. En este sentido se pueden distinguir dos tipos de agente según su entorno, el primero siendo los agentes de entornos específicos (por ejemplo los agentes robóticos) y el segundo siendo los agentes para entornos generales. Estos últimos son diseñados y programados para funcionar en sistemas computacionales de propósito general, usualmente con herramientas independientes de la plataforma.

I (Interacción) Según el modo de interacción del agente.

Existen tres modos de interacción para un agente; estos son las interacciones agente-agente, agente-entorno y agente-usuario. Cada modo de interacción define y utiliza distintos protocolos o dispositivos de comunicación. Por ejemplo, las interacciones agente-agente utilizan herramientas como RMI, CORBA o servicios web, así como protocolos *ad hoc* de capa de aplicación; por otra parte las interacciones agente-usuario hacen uso de sensores, efectores y otros dispositivos de entrada/salida para recibir y comunicar información a agentes humanos.

O (Organización) Según el modo de organización individual o grupal de los agentes.

En primera instancia se distinguen dos tipos de agentes dentro de esta categorización. Estos son los agentes individualistas, aquellos que no poseen capacidades de cooperación, y su contraparte, los agentes cooperantes. Estos últimos son capaces de resolver problemas de manera individual, o pueden delegar la solución de un problema en todo o en parte a otros agentes. Un conjunto de agentes cooperantes que trabajan juntos para resolver un problema o alcanzar un objetivo común se conoce como un sistema multi-agente.

Así mismo, existen otras clasificaciones de agentes según su organización. Algunas de estas clasificaciones son:

- Según el modelo de cooperación, esta puede originar organizaciones de agentes implícitas, es decir establecidas según los algoritmos de resolución de problemas que ejecutan los agentes; o explícita, donde los agentes tienen roles, responsabilidades y establecen protocolos de resolución de problemas en grupo.

- Según el rol de los agentes dentro de un sistema multi-agente. Estos pueden ser estáticos si dicho rol es invariante durante la duración del sistema multi-agente, o evolutivo en caso contrario.
- Según el modo de resolución de conflictos en una organización, los agentes pueden ser agentes especialistas, encargados de detectar y resolver conflictos entre otros agentes; o pueden ser agentes de negociación, los cuales resuelven conflictos inter-agentes mediante protocolos de comunicación especializados.

Evidentemente, estas tres clasificaciones aplican únicamente al caso de los agentes cooperantes.

U (Utilidad) Según el dominio de aplicación y la finalidad de los agentes en el mismo.

A diferencia de las categorizaciones anteriores, esta categorización es de naturaleza variable y distingue a los agentes según sus dominios de aplicación y según las tareas que llevan a cabo dentro de dichos dominios.

1.2. Sistemas multi-agente

Como se mencionó anteriormente, un sistema multi-agente es una organización de agentes cooperantes, los cuales tienen la facilidad de comunicarse entre ellos para coordinar tareas comunes en caso de ser necesario. Los sistemas multi-agentes se utilizan principalmente cuando la naturaleza de los objetivos del sistema es físicamente distribuida, cuando se necesita la cooperación entre múltiples sistemas expertos de dominios heterogeneos, o si el entorno computacional de los agentes está compuesto por múltiples sistemas de red o incluso por un sistema distribuido.

Un sistema multi-agente presenta las siguientes características [7]:

1. Cada agente tiene un conjunto completo de habilidades de adquisición de datos, comunicación, planificación y actuación.
2. El sistema posee un objetivo común, el cual puede descomponerse en tareas independientes. Así mismo, el sistema de alguna manera es capaz de distribuir dichas tareas entre los distintos agentes que lo componen.
3. Los distintos agentes del sistema poseen un conocimiento limitado sobre el estado del entorno, del estado de completitud del objetivo común, y del estado interno e intenciones de los demás agentes.
4. Cada agente posee un cierto grado de especialización en cuanto a las tareas que puede realizar.

Estas características implican que en un sistema multi-agente no existe (o no debería existir) una entidad centralizada para la toma de decisiones y los agentes deben ser capaces de funcionar de manera autónoma a pesar de la incertidumbre producida por la falta de conocimiento global del sistema.

1.2.1. Arquitecturas de sistemas multi-agente

Para organizar un sistema multi-agente se han propuesto múltiples arquitecturas, las cuales toman en cuenta distintos criterios estructurales y funcionales [1]. Dos arquitecturas concretas son las siguientes:

FIPA *Foundation for Intelligent Physical Agents*

Esta arquitectura fué planteada a partir de 1996 por la organización homónima FIPA (*Foundation for Intelligent Physical Agents* - Fundación para Agentes Físicos Inteligentes)[1], la cual se encarga de producir y actualizar un amplio conjunto de estándares que definen a la arquitectura. Curiosamente, a pesar de que su nombre indica "agentes físicos", esta arquitectura esta diseñada especificamente para agentes de software en entornos distribuidos.

[1] http://fipa.org

El núcleo de la arquitectura FIPA es un elemento llamado *Agent Platform* (AP), la cual contiene todos los recursos de hardware y software necesarios para que el sistema funcione. Los estándares FIPA definen únicamente que interfaces y comportamientos deben poseer los distintos elementos de la AP, dejando que cada implementación concreta decida que recursos tecnológicos específicos utilizar.

La AP se compone por los siguientes elementos:

1. El sistema de gestión de agentes, el cual es el encargado de gestionar la AP y a los distintos agentes que pertenecen a ella. Este sistema es un ente centralizado que se encarga de crear, movilizar, cambiar el estado de, y destruir a los agentes. Adicionalmente, este sistema proporciona un servicio de nombres global que permite a los agentes encontrarse entre si dentro del sistema.

2. El facilitador de directorio, que permite ubicar agentes dentro del sistema en base a sus capacidades, y actua como complemento al mecanismo de nombres del sistema de gestión de agentes.

3. El sistema de transporte de mensajes, encargado de transmitir mensajes entre distintos agentes dentro de un mismo sistema FIPA, o incluso entre múltiples sistemas compatibles con FIPA. El sistema de transporte de mensajes de un protocolo de comunicación llamado ACL (*Agent Communication Language* - Lenguaje de Comunicación entre Agentes).

La arquitectura FIPA puede verse en la Figura 1.

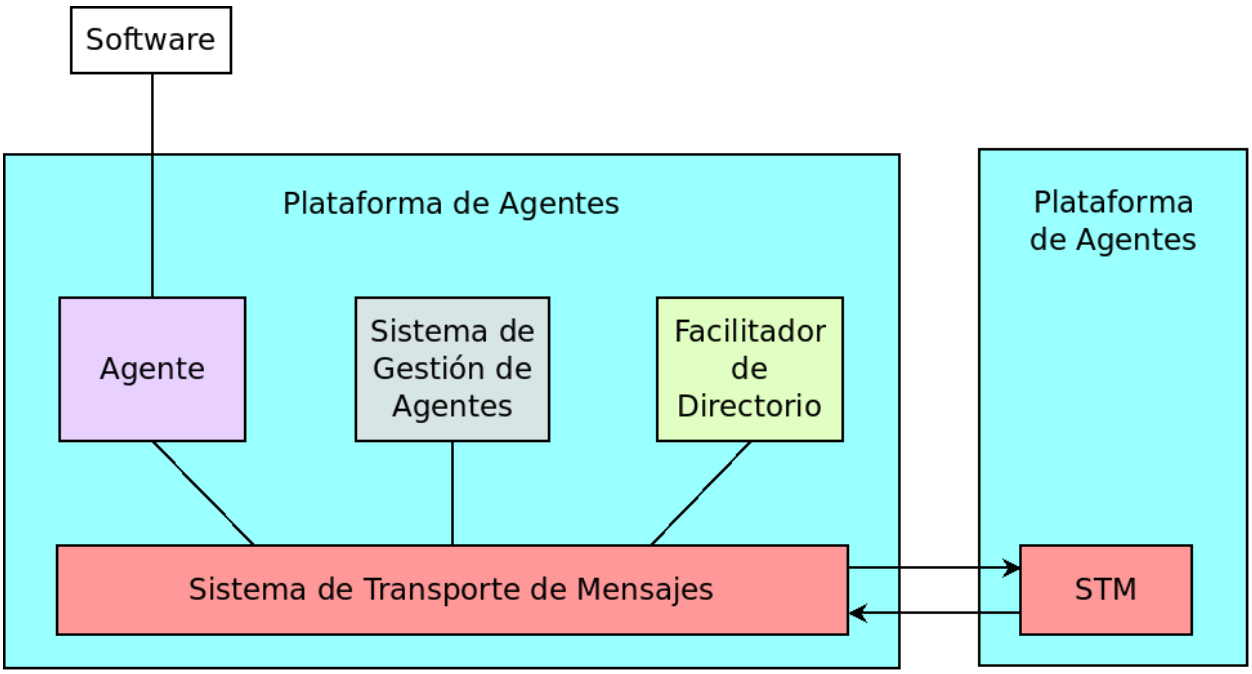

Figura 1: La arquitectura FIPA. Figura recuperada de [1].

RETSINA *Reusable Environment for Task-Structured Intelligent Network Agents.*

Al igual que FIPA, esta es una arquitectura independiente del dominio. RETSINA fue desarrollada por el *Robotics Institute* de la *Carnegie Mellon University*. La arquitectura RETSINA se basa en la premisa de que los agentes incorporados al sistema son entes asíncronos y colaborativos, los cuales poseen roles específicos dentro del funcionamiento del sistema, aunque no se impone una jerarquía entre ellos.

En esta arquitectura se distinguen cuatro tipos de agentes diferentes:

1. Los agentes de interfaz son los únicos que interactúan directamente con los usuarios humanos. Estos agentes se encargan de recibir e incluso anticiparse a las peticiones de los usuarios con respecto al sistema, así como de mostrar los resultados generados.

2. Los agentes de tarea se encargan de descomponer las solicitudes de los usuarios en objetivos ejecutables, y luego coordinar la ejecución de dichos objetivos con los otros agentes de tarea y con los agentes de información.

3. Los agentes de información son capaces de buscar información en fuentes heterogeneas y presentarla a los agentes de tarea con alguna representación uniforme de manera que estos puedan interpretarla.

4. Los agentes intermediarios tienen la responsabilidad de interconectar los distintos agentes entre si utilizando mecanismos de negociación y busqueda de agentes en base a sus capacidades. Se distinguen los siguientes tipos de agentes intermediarios:

   a) **Agent Name Service** Estos buscan la ubicación de un agente según su nombre.

   b) **Matchmakers** Estos buscan agentes según sus capacidades.

   c) **Blackboards** Estos agentes se encargan de recolectar peticiones de servicio.

   d) **Brokers** Estos se encargan de repartir las peticiones recolectadas por los agentes *blackboard* y luego de enviar los resultados a los agentes que originaron dichas peticiones.

   e) **MAS Interoperator** Estos agentes se encargan de intercambiar mensajes con otros sistemas multi-agente.

La arquitectura RETSINA puede verse en la Figura 2.

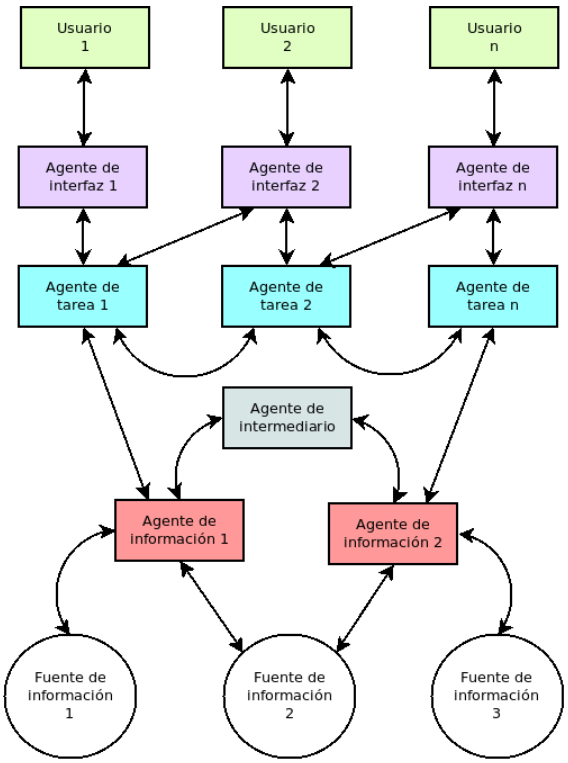

Figura 2: La arquitectura RETSINA. Figura recuperada de [1].

1.2.2. Comunicación y coordinación en sistemas multi-agente

Para realizar la comunicación entre varios agentes de software las arquitecturas mencionadas en la Sección 1.2.1 definen sus propios protocolos para ubicar agentes según varios parámetros, y para la definición y transmisión de mensajes entre ellos. Estos protocolos funcionan en la capa de aplicación en la pila de protocolos TCP/IP, y se caracterizan por permitir la comunicación entre agentes utilizando ontologías e incluso lógica de predicados de primer orden [1].

KQML y FIPA ACL

Estos elementos representan los lenguajes de definión de mensajes utilizados por las arquitecturas RETSINA y FIPA respectivamente. KQML (*Knowledge Query and Manipulation Languaje* - Lenguaje de Consulta y Manipulación de Conocimiento) fue definido por el grupo de *Knowledge Sharing* (Compartición de Conocimiento) de la agencia DARPA del gobierno de los Estados Unidos en el año 1993 [8] con la finalidad de producir un lenguaje de formato de mensajes y su correspondiente protocolo que permitiera el intercambio de conocimiento entre agentes de software de manera automática. Posteriormente la organización FIPA extendió el lenguaje KQML en el antes mencionado lenguaje ACL. Ambos lenguajes permiten definir el contenido de un mensaje utilizando algún lenguaje especializado para ontologías, y en el caso particular de KQML, este fue especialmente para poder ser traducido al lenguaje Prolog [1].

FIPA IPL

IPL (*Interaction Protocol Library* - Biblioteca de Protocolos de Interacción) es una lista de protocolos para comunicación Agente-Agente utilizados en la arquitectura FIPA. Estos protocolos están definidos en una extensión del lenguaje UML llamada diagrama de protocolos AUML. IPL comprende los protocolos mencionados en la Tabla 1, aunque los estándares FIPA permiten que dicha lista sea expandida según las necesidades de cada implementación.

Tabla 1: Protocolos FIPA IPL. Tabla recuperada de [1].

Protocolo	**Descripción**
Request	Solicita a un agente que realice cierta acción.
Request when	Similar a *Request* pero permite incorporar una. precondición a la solicitud.
Query	Se pide información a un agente.
Contract net	Un agente solicita la realización de un trabajo a otro conjunto de agentes.
Brokering	Un *broker* ofrece funcionalidades de otros agentes o reenvía peticiones.
English auction	Inicia una petición de realización de trabajo a modo de subasta inglesa, con el iniciador del mensaje actuando de subastador.
Dutch auction	Inicia una petición de realización de trabajo a modo de subasta holandesa.
Recruiting	Similar al *Brokering* pero las propuestas de servicio van directamente al iniciador, en lugar de ser acumuladas en un *broker*.
Propose	El iniciador propone la ejecución de un trabajo a otro agente, que tiene la potestad de rechazarla.
Subscribe	Solicita una notificación en caso de que se cumpla alguna determinada condición.

1.3. Agentes robóticos

Un robot es un agente físico que puede realizar tareas manipulando elementos en el mundo real [3]. Para lograr esto, los robots están equipados con tres clases de componentes que les permiten obtener información del entorno, y poder interactuar con el mismo: los sensores, actuadores y efectores. Un sensor es un dispositivo que permite al robot percibir una variable física específica del ambiente; por ejemplo, temperatura, humedad, sonido, distancia, e incluso imágenes y otras clases de señales. Un efector es otra clase de dispositivo físico que permite al robot el aplicar una fuerza sobre algún elemento de su entorno. Los efectores suelen tomar la forma de brazos, gruas, piernas, ruedas, helices, entre otros. Por su parte los actuadores son lineas de control que conectan a los sensores con los efectores.

Se distinguen dos clases de robots. La primera son los llamados manipuladores industrales, los cuales permanecen sobre una base fija y realizan acciones determísticas preprogramadas. Estos robots son muy utilizados en la industria del ensamblado. La otra clase son los robots móviles, los cuales se distinguen de los manipuladores por tener la capacidad de desplazarse físicamente en el ambiente utilizando ruedas, piernas, hélices o propulsores. Los robots móviles suelen tomar forma de vehículos según su entorno, aunque se han desarrollado robots móviles que poseen formas humanoides e incluso animales.

Los robots tienen que programarse para lidiar con entornos caracterizados por ser:

Estocásticos El estado del entorno es producto de su estado en un instante de tiempo anterior, modificado por elementos impredecibles que pueden ser de naturaleza real o aparentemente aleatoria.

Continuos Las percepciones del robot, incluyendo la percepción del tiempo son funciones continuas.

Parcialmente observables Los sensores del robot no son capaces de percibir el estado completo de todo el entorno en un momento dado. Adicionalmente, estas percepciones pueden estar afectadas por ruido cuya naturaleza varía según los distintos tipos de sensores.

Dinámicos El estado del entorno puede cambiar mientras el robot delibera su próxima acción a ejecutar.

Dado que los agentes robóticos pertenecen a un espacio físico y, en el caso de los robots móviles, pueden desplazarse dentro de este, se hace necesario que el robot mantenga información sobre su ubicación en el entorno y la disposición física del mismo. Este problema es conocido como SLAM (*Simultaneous Location and Mapping* - Cartografía y Localización Simultanea). Para resolver este problema existen varios algoritmos como el algoritmo EKF (*Extended Kalman Filter* - Filtro de Kalman Extendido) o el algoritmo de localización Monte-Carlo [3].

De la misma forma que los agentes de software pueden cooperar para formar sistemas multi-agente, los agentes robóticos pueden formar sistemas o entornos multi-robot. Esto agrega la necesidad de conocer la posición de otros robots en el entorno, además de la ubicación propia, de forma que los robots del sistema puedan cooperar e incluso evitar colisiones físicas entre ellos.

2. Categorías de sistemas robóticos multi-agente

Los sistemas multi-robot nacen por la necesidad de resolver problemas de manera robusta, flexible y confiable en entornos potencialmente desconocidos. Comparados con los sistemas mono-robot, los sistemas multi-robot aportan una serie de ventajas, dado que estos son proporcionan la capacidad de resolver problemas físicamente distribuidos en paralelo y son más tolerantes a fallos. Así mismo, es más sencillo diseñar robots simples para resolver tareas complejas usando esquemas tipo divide y vencerás, que diseñar robots monolíticos complejos capaces de resolver por si solos problemas complejos y los subproblemas que estos implican.

Los sistemas multi-robot se pueden organizar de distintas formas, utilizando diseños bio-inspirados o arquitecturas basadas en las utilizadas para la implementación de sistemas multi-agente en software. Sin embargo, todas estas organizaciones se engloban dentro de una categorización mayor llamada robótica colectiva.

2.1. Robótica colectiva

Robótica colectiva es un término general que se utiliza para referise al estudio de problemas que involucran múltiples robots realizando tareas simultaneamente en un mismo entorno. Vito Trianni identifica en [2] a los siguentes como los primeros y más comunes problemas de estudio en robótica colectiva:

Busqueda y recolección de objetos Este problema consiste en coordinar un conjunto de robots para que recolecten objetos de cierta naturaleza repartidos en el ambiente para luego acumularlos en una ubicación específica denominada hogar o nido (en asociación con colonias de hormigas).

Empujado de cajas Este problema, inspirado en el llamdo *piano movers problem* (problema de los movedores de pianos), consiste en desplazar una serie de contenedores o cajas bajo ciertos criterios, tomando en cuenta consideraciones con respecto al tamaño y peso de las mismas, y la prescencia de obstaculos en el ambiente.

Movimiento coordinado Este problema consiste en movilizar un conjunto de robots según algún criterio específico, de forma que estos mantengan una determinada formación durante su desplazamiento; esto sin provocar colisiones entre los robots y minimizando el tiempo que estos pasen fuera de la formación de ser necesario.

Las distintas organizaciones de sistemas multi-robot mencionadas a continuación tratan de resolver estos tres problemas y otros problemas derivados de estos.

2.2. Robótica de segundo orden

Según este paradigma, los agentes robóticos mencionados hasta el momento son conocidos como robots de primer orden; es decir, agentes robóticos completos y autonomos. Se define entonces a los robots de segundo orden como agentes robóticos conformados por una combinación física de robots de primer orden. Entre las distintas organizaciones de sistemas multi-robot esta es la más antigua de todas [2], propuesta en 1987 por el investigador T. Fukuda y colaboradores entre 1987 y 1988 con su robot CEBOT *Cellular Robotic System* (Sistema Robótico Celular) [10, 11].

En la actualidad se distinguen dos clases de sistemas robóticos de segundo orden, los sistemas auto-reconfigurables y los sistemas auto-ensamblables.

2.2.1. Robots auto-reconfigurables

Estos sistemas se componen por pequeños módulos sensoriales que por si solos tienen pocas capacidades de movilidad. Estos módulos tienen la capacidad de interconectarse físicamente para ensamblar un sistema robótico más complejo, así como también tienen la capacidad de modificar dichas interconecciones de forma autónoma para modificar al robot de segundo orden formado por ellos. Estos robots comienzan su funcionamiento en alguna configuración específica ensamblada manualmente por un operador humano.

La reconfiguración puede realizarse en una, dos o tres dimensiones. Esta puede consistir en el desplazamiento o rotación de los módulos según los grados de libertad que posean los mismos, e incluso puede consistir en la separación del robot de segundo orden en varios robots de segundo orden de menor tamaño o viceversa.

2.2.2. Robots auto-ensamblables

Los robots auto-ensamblables son una extensión de los robots auto-reconfigurables, con la distinción de que los módulos robóticos que se conbinan para ensamblar las unidades de segundo orden poseen completa autonomía funcional. Adicionalmente, estos sistemas auto-ensamblables no necesitan poseer una configuración de segundo orden inicial.

Una arquitectura particular de robot auto-ensamblable es la *Super Mechano Colony* de R. Damoto y colaboradores [12]. En esta arquitectura un robot llamado nave madre posee un conjunto de robots adjuntos

llamados unidades hijas. Estas unidades hijas se componen por una rueda y un manipulador sencillo con el cual pueden acoplarse a la nave madre para funcionar como ruedas de esta, o pueden acoplarse entre si para formar cadenas para superar obstaculos que una sola unidad no pueda sortear.

Otro enfoque consiste en modulos sensoriales pasivos suspendidos en algún fluido que permita movimiento browniano. Este enfoque, conocido como Robots Celulares Estocásticos, consiste en la combinación aleatoria de los modulos sensoriales cuando estos colisionan en el medio, de forma que puedan compartir recursos energéticos, e incluso formar sistemas multi-procesador para el análisis de sus medidas sensadas.

2.3. Enjambres de robots

La robótica basada en enjambres es una continuación de las investigaciones en IA bioinspirada, particularmente la referente a *Swarm Intelligence* (Inteligencia de Colmena) la cual estudia y desarrolla algoritmos de resolución de problemas inspirandose en el comportamiento de insectos sociales. Sin embargo no basta con utilizar algoritmos de *Swarm Intelligence* para controlar robots, o sencillamente utilizar multiples robots colaborativos para que un sistema multi-robot sea considerado un enjambre. Los autores Dorigo y Şahin definen en [13] cuatro condiciones que deberían cumplir los enjambres de robots:

1. El sistema debe ser descentralizado y compuesto por robots con capacidades comunicación y sensado limitadas por su localidad. Se excluyen los sistemas con controles centralizados o mecanismos de sensado global del ambiente porque estos no escalan bien con el número de robots.

2. El sistema debería considerar la coordinación y control de grandes números de robots. Esto incluye sistemas que contemple escalabilidad en el número de robots, pero excluye aquellos sistemas de tamaño invariable compuestos por pocos robots.

3. El sistema debería estar compuesto por robots homogeneos, o en su defecto por pocos grupos heterogeneos de muchos robots homogeneos. Se busca la homogeneidad en el tipo de robots porque los enjambres deben ser altamente redundantes.

4. Debe ser demostrable que las tareas que realiza el sistema no pueden ser resueltas de manera eficiente por robots individuales o por pequeños grupos de robots.

El cumplir con las condiciones mencionadas le proporciona a los enjambres de robots ciertas propiedades características revisadas a continuación.

2.3.1. Descentralización

Como se mencionó anteriormente, los mecanismos de control centralizados no funcionan en enjambres de robots dado que estos últimos implican grandes cantidades de robots que deben operar de forma simultanea y cooperativa. Para lograr un mecanismo de control centralizado en un ejambre es necesario poseer una infraestructura de comunicación que sea capaz de conectar a cada robot con el sistema de control de manera rápida y confiable, además de que este debe poseer la capacidad de cómputo necesaria para poder calcular el curso de acción de todos los robots del sistema. Adicionalmente, el sistema de control centralizado se convierte inevitablemente en un punto de falla crítico sobre el cual descansa la estabilidad del sistema completo.

Al contrario, los enjambres de robots deben permitir que cada robot sea capaz de tomar decisiones autónomas, distribuyendo así el proceso de toma de decisiones entre todos los robots del sistema. Idealmente, estos robots tendrán un sistmea de control propio relativamente sencillo capaz de transmitir intenciones a otros robots del sistema dentro de su localidad. El objetivo global y la complejidad del sistema emerge como resultado de la combinación de los objetivos y el comportamiento de todos los robots que conforman el mismo.

Sin embargo, el mecanismo de control descentralizado de los enjambres puede sufrir de un problema de paralización o estancamiento. Este problema surge cuando los objetivos personales de los robots del sistema son mutuamente excluyentes, lo que produce una situación de interbloqueo que, en principio, solo puede resolverse por algún cambio aleatoreo en el ambiente o por interferencia de agentes externos al sistema que permitan que algunos de los robots pueda continuar su ejecución.

2.3.2. Localidad y estimergia

Los robots pertenecientes a un ejambre solo son capaces de comunicarse con los robots que están en su vecindad, presumiblemente mediante enlaces de comunicación inalámbricos. Comunicarse de esta manera resuelve el problema de poseer un ente central que actúa como cuello de botella para todas las comunicaciones del sistema. Sin embargo, incluso estas comunicaciones locales implican un costo computacional y están sujetas a todos los problemas que afectan a los sistemas de comunicación por red.

Una alternativa bio-inspirada para lograr la comunicación entre los robots de un enjambre es el uso de comunicaciones implícitas, originadas como resultado natural de las acciones que llevan a cabo los robots, o mediante el uso de marcas que los robots pueden dejar conscientemente en el ambiente y que pueden ser percibidas por los demás robots del sistema. Este mecanismo es conocido como estimergia y se basa en el comportamiento de distintas especies de insectos sociales como las hormigas y las termitas, especies de insectos caracterizados por utilizar rastros de feromonas y otros mecanismos para comunicar información entre individuos de una colmena. Un ejemplo de robots que se comunican con mecanismos implícitos mencionado por Trianni [2] es un enjambre de robots diseñado para podar cesped. En este sistema particular no es necesario que los robots se comuniquen explicitamente las rutas que van a tomar de forma que dos o más robots poden la misma zona, desperdiciando el trabajo de los robots que sigan al primero por dicha zona. En este caso, los robots podrían sensar de alguna manera que zonas han sido podadas y evitarlas, siendo el trabajo mismo de los robots el mecanismo mediante el cual estos indican por donde han pasado.

2.3.3. Flexibilidad y robustez

Obtener robustez es la razón principal por la cual se establece que un enjambre de robots debe estar compuesto por robots mayormente homogeneos, de forma que el sistema cuente con un alto grado de redundancia. De esta forma, si uno o varios robots del enambre sufren alguna falla, sus responsabilidades pueden ser absorvidas de manera natural y transparente por el resto de los robots del sistema. En el caso contrario, si el sistema está compuesto por muchas clases diferentes de robots, en particular si existen robots no redundates especializados en realizar ciertas tareas críticas para el correcto cumplimiento de los objetivos del sistema, estos robots se convierten en puntos críticos importantes cuyo fallo no puede ser tolerado por el sistema en conjunto.

La flexibilidad se refiere a la capacidad del sistema de adaptarse rápidamente a cambios en el ambiente. Esta característica es una consecuencia de la descentralización del sistema, y está estrechamente ligada al mecanismo de comunicación utilizado por el mismo. En el caso de los insectos sociales, la flexibilidad de los sistemas sociales se obtiene principalmente como consecuencia de la estimergia, de manera que cambios ambientales son considerados de alguna manera como producto del trabajo de los entes de dicha sociedad, independientemente de si esto es cierto o no.

2.3.4. Comportamientos emergentes

Idealmente, el comportamiento global de un enjambre de robots debe surgir como consecuencia natural del comportamiento de los robots que lo componen, sin necesidad de que este se encuentre programado o especificado directamente sobre los robots. Sin embargo, un sistema multi-robot no necesita poseer un comportamiento global emergente para ser considerado un enjambre. El principal reto al implementar un sistema con comportamiento global emergente consiste en diseñar el comportamiento individual de los robots de manera que estos puedan manipular propiedades globales del entorno del sistema de una manera implícita que contribuya al cumplimiento de los objetivos del sistema.

Este problema de diseño consiste en descomponer el comportamiento global del sistema en mecanismos e ineracciones simples, además de que estos mecanismos e interacciones deben codificarse de alguna manera para que puedan ser incorporados a los sistemas de control de los robots del enjambre. Como puede observarse, el diseño de sistemas con comportamiento emergente suele realizarse como un proceso de diseño descendente.

2.4. Auto-organización en sistemas multi-robot.

La auto-organización se refiere a la capacidad de un sistema de tender a un estado de equilibrio de manera natural como producto de las interacciones de los componentes del mismo. Formalmente, se define como "un proceso mediante el cual un patrón en el nivel global de un sistema emerge únicamente como resultado de las numerosas interacciones entre los componentes de menor nivel del sistema. Más aún, las reglas que especifican las interacciones entre los componentes del sistema se ejecutan únicamente utilizando información local, sin hacer referencia al patrón global." [14].

Los estados de equilibrio en los sistemas auto-organizables son conocidos como *attractors* (atractores) y emergen como el resultado de las interacciones locales de los componentes del sistema en respuesta a dos mecanismos fundamentales conocidos como retroalimentación positiva y negativa. La retroalimentación positiva consiste en la amplificación de las fluctuaciones aleatorias del sistema las cuales guian al sistema hacia un estado de equilibrio. Por su parte la retroalimentación negativa funciona como un mecanismo auto-regulador, el cual usualmente se produce por la reducción de recursos disponibles al sistema como producto de la retroalimentación positiva.

El principal beneficio que se espera obtener por utilizar diseños auto-organizativos en sistemas multi-robot es la simplificación del problema de diseño de sistemas que muestren comportamiento emergente. Utilizando auto-organización, se espera que el sistema pueda encontrar de forma autónoma un estado de equilibrio que satisfaga los objetivos del mismo, partiendo de un estado inicial ruidoso.

Asi mismo, el diseño de sistemas auto-organizativos se plantea como una alternativa de diseño ascendente, contrapuesta al diseño descendente de sistemas con comportamientos emergentes descrita en la sección anterior.

3. Una arquitectura de control para sistemas multi-robot con comportamiento emergente.

En [9] A. Gil y colaboradores plantean la arquitectura de control para sistemas multi-robot que puede observarse en la Figura 3. Esta arquitectura se compone de los siguientes tres niveles:

Nivel colectivo Este nivel se encarga de apoyar las interacciones entre los robots del sistema entre sí, así como entre los robots y otros elementos del ambiente con los cuales puedan comunicarse.

Nivel individual En este nivel se administran las percepciones individuales de cada robot, así como la toma de decisiones locales y el manejo de las *emociones* del robot.

Nivel de administración del conocimiento y procesos de aprendizaje Este nivel administra todo lo referente a la adquisición y representación del conocimiento tanto individual como colectivo, así como los procesos que aprenden de dicho conocimiento, también de manera individual y colectiva.

Es a travez de la interacción de los robots por medio de estos componentes que se hace posible el surgimiento de comportamientos cooperativos entre los robots. Sin embargo, los robots poseen en su sistema de control interno únicamente los mecanismos necesarios para controlar sus sitemas de locomoción y comunicación, así como ciertos comportamientos básicos fundamentales. Los demás elementos de la arquitectura se ubican en un sistema de control centralizado al cual tienen acceso todos los robots del sistema.

3.1. Nivel colectivo

En el nivel colectivo se compone de un módulo de coordinación el cual se encarga de administrar las interacciones entre los distintos elementos del sistema desde la perspectiva personal de cada robot involucrado.

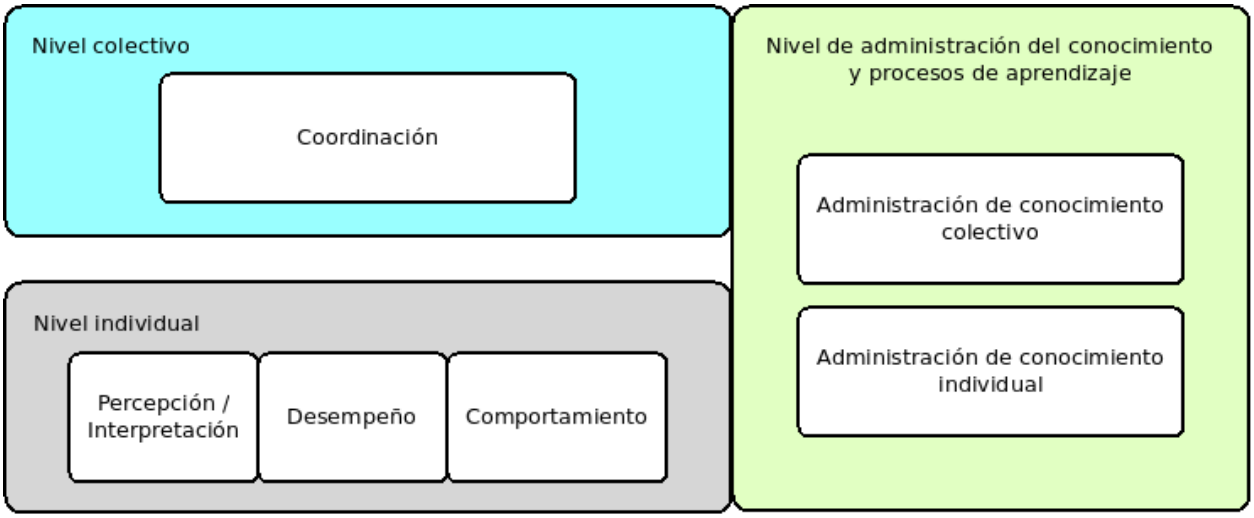

Figura 3: Arquitectura de control para sistemas multi robot. Figura recuperada de [9].

3.2. Nivel individual

En este nivel se gestionan los procesos de control de cada robot. Este nivel se compone por los siguientes tres módulos:

Módulo de de percecpión/interpretación Este módulo se encarga de obtener información de los sensores del robot. Estas percepciones son preprocesadas directamente en el módulo y pueden provocar la ejecución de comportamientos reactivos de ser necesario. Si no se producen comportamientos reactivos, entonces las percepciones son enviadas a un submódulo de interpretación que puede activar comportamientos deliberativos.

Módulo de desempeño Este módulo administra los actuadores del robot; en particular se encarga de gestionar el sistema de locomoción. Este módulo se implementa directamente en cada robot.

Módulo de comportamiento Este módulo se encarga de la ejecución de comportamientos de alto nivel y se divide en varias subcapas.

3.2.1. El módulo de comportamiento

Este módulo se divide en cuatro subcapas las cuales se encargan de generar comportamientos según las percepciones obtenidas por el robot en un momento dado. Dichas subcapas son las siguientes:

Capa reactiva Esta capa se encarga de generar comportamientos reactivos mediante un mecanismo de estimulo-respuesta, con el objetivo de garantizar la supervivencia del robot. Esta capa tiene prioridad sobre las demás capas del módulo de comportamiento.

Capa cognitiva Esta capa se encarga de deliberar sobre las percepciones del robot utilizando su conocimiento local del ambiente y los demás robots en el sistema. Estos comportamientos deliberados se componen de comportamientos más sencillos que pueden ser ejecutados directamente por el robot.

Capa social Esta capa utiliza el conocimiento colectivo obtenido por el sistema para generar comportamientos colaborativos que luego puedan ser ejecutados por el módulo colectivo de la arquitectura. Esta capa es la responsable de administrar las interacciones entre los robots.

Capa afectiva Esta capa permite activar o vetar los comportamientos generados por las capas cognitiva y social mediante un modelo emocional que permite al robot poseer un grado de comodidad con respecto a la realización de dichos comportamientos. Este modelo emocional consiste en un conjunto de emociones positivas y negativas que se mantienen como resultado del estado interno de cada robot en un momento dado.

Las tres últimas capas se ejecutan de manera transversal. Esta organización del módulo de comportamiento puede observarse en la Figura 4.

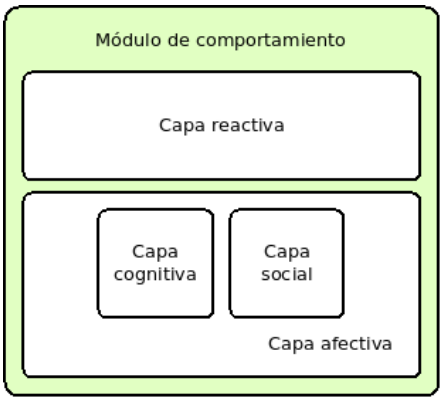

Figura 4: Módulo de comportamiento. Figura recuperada de [9].

3.3. Nivel de administración del conocimiento y procesos de aprendizaje

Este nivel se encarga manejar los procesos individuales y colectivos referentes a la manipulación y compartición de conocimiento en el sistema. Para lograr esto este nivel define una base de conocimiento colectiva centralizada la cual se construye en base una serie de procesos aplicados sobre múltiples bases de conocimiento individuales a cada robot en el sistema. Los procesos que se pueden realizar entre las bases de conocimiento son los siguientes:

Socialización Permite a los robots compartir su conocimiento individual con el resto del sistema.

Agregación Permite ordenar, filtrar y combinar conocimiento a nivel colectivo.

Apropiación Permite a los robots convertir el conocimiento colectivo en conocimiento individual.

Estos procesos se realizan en dos módulos:

Módulo de manejo de conocimiento individual Este módulo implementa los procesos de socialización y apropiación, así como los procesos de aprendizaje que pueden llevar a cabo los robots de manera individual.

Módulo de manejo de conocimiento colectivo Este módulo implementa el proceso de agregación, y su responsabilidad es el mantener la base de conocimiento colectivo a disposición de todos los robots del sistema.

Los tres procesos de socialización, agregación y apropiación definen un ciclo de retroalimentación que permite a los robots construir conocimiento de manera grupal, el cual luego puede ser utilizado para enriquecer el conocimiento individual de cada robot.

Referencias

[1] A. Mas, *Agentes Software y Sistemas Multi-Agente: Conceptos, Arquitecturas y Aplicaciones*, Pearson, 2005.

[2] V. Trianni, *Evolutionary Swarm Robotics*, Springer, 2008.

[3] S. Russel y Norvig, P., *Inteligencia Artificial: un enfoque moderno*, Pearson, 2010.

[4] F. J. Garijo, *Tecnología de agentes: Experiencias y perspectivas para el desarrollo de nuevos servicios y aplicaciones*, Bole.tic N^o 24, 2002, pp. 1-9.

[5] M. Wooldridge, *Agent-based software engineering*, IEEE Proc. on Software Engineering, 144(1), 1997, pp. 26-37.

[6] Y. Demazeau, *La méthode VOYELLES, dans Systèmes Mutli-Agents: Des Théories Organisationnelles aux applications Industrielles*, Eds. Hermès, 2001.

[7] R. Wesson, et al., *Network Structures for Distributed Situation Assessment*, Readings in Distributed Artificial Intelligence, Morgan Kaufmann, 1988.

[8] T. Finin, et al., *Specification of the KQML Agent-communication Language*, ARPA Knowledge Sharing Initiative, External Interfaces Working Group.

[9] A. Gil, et al., *Architecture for Multi-robot Systems with Emergent Behavior*, Proceedings on the International Conference on Artificial Intelligence (ICAI), WorldComp, 2015, pp. 41

[10] T. Fukuda y Nakagawa, S., *A dynamically reconfigurable robotic system*, Proceeding of the 1987 IEEE International Conference on Industrial Electronics, Control and Instrumentation, 1987, pp. 588-595.

[11] T. Fukuda, et al., *Self organizing robots based on cell structures - CEBOT*, Proceedings of the 1988 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '88), 1988, pp. 145-150.

[12] R. Damoto, Kawakami A. y Hirose, S., *Study of super-mechano colony: concept and basic experimental set-up*, Advanced Robotics, 15(4), 2001, pp. 391-408.

[13] M. Dorigo y Şahin, E., *Swarm Robotics*, Autonomous Robots, 17(2-3), 2004, pp. 111-113.

[14] S. Camazine, et al., *Self-Organization in Biological Systems*, Princeton University Press, 2001.