Diseño de un Sistema Multiagente para un Software de Simulación de Celdas de Producción Robóticas

Carlos Álvarez G.
Universidad de Santiago de Chile
Doctorando en Ciencias de la Ingeniería, Mención Automática
Departamento de Ingeniería Eléctrica
Ecuador 3769, Estación Central, Santiago de Chile.
e-mail: carlos.alvarez@technologies.cl

Andrés F. Soto P.
Universidad Tecnológica Metropolitana
Doctorando en Ciencias de la Ingeniería, Mención Automática, USACH
Departamento de Ingeniería Eléctrica
Av. José Pedro Alessandri 1242, Ñuñoa. Santiago, Chile
e-mail: andres.soto@utem.cl

Patricio Olavarrieta S.
Ingeniero Civil Electricista, Universidad Tecnológica Metropolitana
U de Ch., Master DpI U. Jaume I Castellón España
Departamento de Ingeniería Eléctrica
Av. José Pedro Alessandri 1242, Ñuñoa, Santiago de Chile
e-mail: polavarr@utem.cl

Francisco Herrera G.
Magister en Telecomunicaciones
Universidad Tecnológica Metropolitana
Departamento de Ingeniería Electrónica
Av. José Pedro Alessandri 1242, Ñuñoa, Santiago de Chile
e-mail: fherrera@utem.cl

Resumen

Este trabajo presenta una investigación en curso con los aspectos más importantes para el diseño de un software de simulación de celdas de producción robótica, basado en el paradigma de multiagentes. El objetivo central es fundar las bases para la construcción de un sistema que permita a través de una interfaz gráfica modelar celdas de producción. Para ésto, se definen los conceptos necesarios de agentes y celdas de producción, un marco de trabajo que incluyen estándares, metodologías, herramientas orientadas al desarrollo de sistemas multiagentes y finalmente una propuesta inicial del modelo de agentes para el desarrollo del software.

Palabras Clave: Sistemas Multiagentes, Simulación, Celdas de Producción, Robótica Industrial, Manufactura Flexible.

Abstract

This paper presents an ongoing investigation with the most important aspects for the design of a software simulation of robotic production cells based on the multiagent paradigm. The main objective is to lay the foundation for building a system that allows, through a graphical interface to model production cells. For this, define the necessary concepts of agents and production cells, a framework that includes standards, methodologies, tools aimed at developing multi-agent systems and finally a model initially proposed by the agents for the development of software.

Keywords: Multiagent Systems, Simulation, Production Cells, Industrial Robotics and Flexible Manufacturing.

▸ Carlos Álvarez G.
▸ Andrés F. Soto P.
▸ Patricio Olavarrieta S.
▸ Francisco Herrera G.

1 Introducción

El concepto de agente y sistemas de multiagentes (MAS – Multi-Agents Systems) se ha convertido en un tópico importante, tanto en inteligencia artificial como en informática, específicamente en tópicos relacionados al diseño, construcción de arquitecturas distribuidas y colaborativas, con aplicaciones de gran importancia en numerosos campos como el manejo y control de la información en redes computacionales, redes de sensores, sistemas de tiempo real, robótica, manejo de recursos y aplicaciones de tráfico aéreo entre muchas otras.

Múltiples definiciones de agentes se pueden encontrar en la literatura técnica desde hace algunos años, de todas ellas nos quedaremos con la siguiente definición: "Los agentes de software son programas que establecen diálogos, negocian y coordinan transferencia de información. Los agentes necesitan ser abstraídos de los protocolos de comunicación de las redes, de las prestaciones del sistema y de los asuntos computacionales de los niveles inferiores. Un lenguaje de programación de agentes debe basarse en tareas; solamente se quiere especificarse el comportamiento del agente en forma abstracta y que el lenguaje brinde un soporte apropiado". Esta definición nos impone la necesidad de tener un marco de trabajo que nos permita la integración de agentes de distinta índole, definir una metodología orientada a agentes y elegir herramientas adecuadas para su implementación.

Un sistema multiagente será entonces, un conjunto de agentes interactuando entre si, de manera exitosa y cada uno con objetivos diferentes capaces de cooperar, negociar y coordinarse para resolver un problema o cumplir con un objetivo más grande.

Por otra parte si consideramos una definición simple de celdas de producción como: "Un sistema compuesto por una o varias estaciones de trabajo compuestas por una o más máquinas, donde cada una de ella ejecuta una acción sobre la pieza o parte", las que, a su vez forman parte de un sistema más complejo, que incluye máquinas de producción, computadoras y software, dando como resultado lo que se conoce como sistemas de fabricación por computadora (CIM - Computer Integrated Manufacturing) y que dio paso a sistemas de manufactura flexible (FMS – Flexible Manufacturing Systems) donde se incluyen robots industriales, un sistema automático de manejo de materiales y la flexibilidad suficiente para que sus componentes pueden ser dispuestos y configurados de la mejor forma para alcanzar la eficiencia productiva deseada, es aquí donde encontramos un entorno fértil para aplicar el paradigma de multiagentes.

En el diseño de un software de simulación es importante considerar el aspecto relacionado con los grados de flexibilidad de una celda de producción bajo ambientes inciertos y dinámicos, esto grados de flexibilidad pueden ser del producto o del proceso, lo cual hace necesario revisar y analizar diferentes modelos [1], como situaciones que se deben tener en cuenta en el software de simulación.

En una primera aproximación podemos visualizar un sistema de producción robótica como un conjunto de agentes que se interrelacionan entre si, en la figura 1, se muestra un esquema simple de una celda de inspección, donde se observan los elementos clásicos tales como: cámaras, sensores, cintas transportadoras, manipuladores, etc.

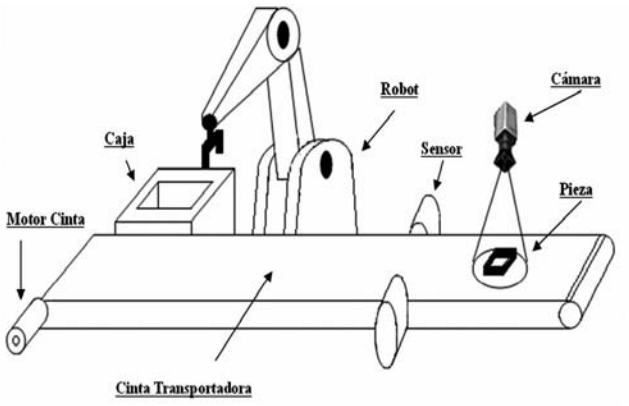

Figura 1. Celda de Inspección.

Si efectuamos una revisión de las implementaciones típicas de celdas de producción, que incluyen uno o más manipuladores, en ambientes académicos e industriales, estas terminan siendo soluciones cerradas, donde el robot manipulador es el corazón del sistema y condiciona o jerarquiza el sistema por completo. Físicamente, el robot trae incorporado puertos de entrada y salida

para sensores y periféricos, a los que posteriormente se hacen referencia en el programa del robot, quedando de manifiesto el control jerárquico que ejerce este sobre algunas de las maquinas de la celda, es decir se implementan esquemas del tipo maestro-esclavo, que carecen de toda flexibilidad.

En las siguientes secciones se desarrollan conceptos importantes a tener en cuenta en el diseño de un sistema multiagente.

2 Conceptos

2.1 Agente

En la figura 2, se muestra el modelo general de un agente, que posee sensores para determinar mediante un "algoritmo inteligente", la mejor salida o acción del medio ambiente.

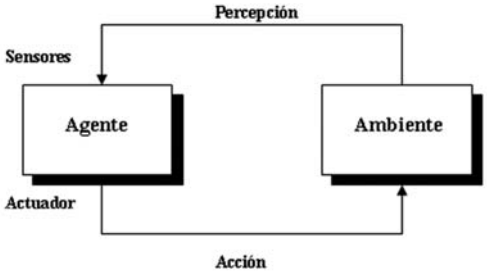

Fig. 2. Modelo General de un Agente.

Estos algoritmos se construyen utilizando en general arquitecturas cognitivas[2] que se traducen en un conjunto de acciones posibles sobre el medio ambiente, con ésto podemos usar el concepto para describir distintos sistemas con complejidades diferentes, por ejemplo: un sistema de control PID, un demonio en un sistema operativo, un sensor de incendio en una ambiente de RED, un robot móvil o una red se sensores de seguridad. Por lo tanto la capacidades del agente dependerán de la aplicación particular y como el desarrollador a través de un lenguaje le entrega cierta inteligencia.

Es evidente que el medio ambiente sobre el cual el agente trabaja puede facilitar o dificultar su cometido. Estos se pueden clasificar de acuerdo a: accesible versus no accesible, determinístico versus no determinístico, estático versus no estático, discreto versus continuo. El medio más complejo para implementar un agente es en un ambiente: inaccesible, no determinístico, dinámico y continuo, referenciado habitualmente como ambiente abierto. Un robot móvil está más propenso a encontrar éstos ambientes abiertos, en contraposición en la robótica industrial donde el ambiente es conocido, cerrado o controlado de acuerdo a requerimientos.

Existen diferentes tipos de agentes: móviles, colaborativos, de información, reactivos, híbridos y todos ellos con capacidad de conformar un sistema hibrido de agentes, que llamaremos un sistema de agentes múltiples (MAS). Un agente será inteligente si es capaz de mostrar las siguientes características [3]:

- **Reactividad**: Percibir el ambiente y responder de manera oportuna a los cambios del medio para satisfacer sus objetivos.
- **Proactividad**: Capacidad para exhibir comportamiento basado en metas tomando la iniciativa para satisfacer sus objetivos.
- **Habilidad Social**: capacidad para interactuar con otros agentes para satisfacer sus objetivos.

Estas características ciertamente dejan al control PID y al demonio como agentes no inteligentes, lo importante en este punto es que el agente esta inserto en un entorno donde interactúa con el medio y otros agentes, es altamente deseable que muestre autonomía para conseguir sus objetivos y que sea capaz de comunicarse con otros agentes. Aun cuando esto no es un requisito para todos los agentes.

Algunas características complementarias[3] de los agentes son: movilidad, veracidad, benevolencia, adaptación, aprendizaje y el de un sistema múltiagente es que sean Autónomos, Distribuidos, Heterogéneos, Cooperativos, multiobjetivo, en este contexto la comunicación

entre ellos es de vital importancia dado que permite sincronizar acciones, enviar y recibir conocimiento, resolver conflictos en la resolución de tareas etc.

2.2 Arquitecturas De Un Agente

Los agentes tienen un conjunto de acciones posibles a ejecutar produciendo una transformación de estados sobre el ambiente, que es un conjunto finito de estados discretos. El modelo básico es que el ambiente parte en un estado inicial y el agente escoge que acción ejecutar sobre el ambiente, como resultado de esto el ambiente responde con un número posible de estados.

Un sistema es un par que contiene un agente y un ambiente, un sistema tendrá asociado a un conjunto posible de ejecuciones, que por simplicidad solo contendrá ejecuciones que tienen término. El efecto que un agente tiene sobre el medio ambiente, se puede describir con una función de transformación de estado.

Se debe considerar que el ambiente es dependiente de la historia y permite ambientes no deterministas, formalmente un ambiente es una tripleta $E_{nv}(E,e_0,\tau)$, donde E es un conjunto de estados del ambiente, e_0 E que es el estado inicial, y τ la función de transferencia de estado.

En la figura 3, se puede ver el esquema de percepción del agente:

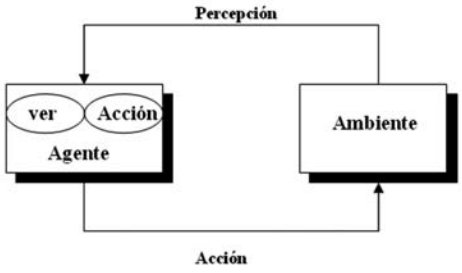

Figura 3: Subsistemas Percepción y Acción

Con la función ver el agente captura el ambiente y la función accion representa el proceso de toma de decisión, la salida de la función ver es una percepción y, la acción es una función acción nos lleva de una percepción a una acción, de esta forma un agente es un par Agente(ver , accion), en la figura 4, se muestra un agente que mantiene un registro de sus estados de tal forma de al tomar una decisión considera esta información guardada.

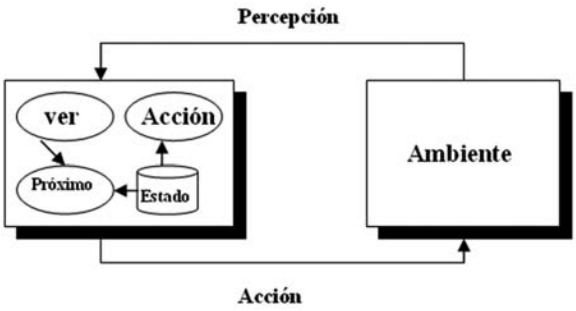

Figura 4: Mantención de Estados

El comportamiento puede ser resumido como: el agente comienza en un estado inicial eo, ver el ambiente en un estado e y genera una percepción ver(e). El estado interno del agente es actualizado por la función proximo(io, ver(e)), donde io, es el estado interno inicial del agente, la acción seleccionada por el agente es accion(proximo(io, ver(e))) y así sucesivamente.
Según como se implementa la función acción, se definen diferentes arquitecturas [2], [3] tales como: Basadas en lógica, Deliberativas, Reactiva, en Capas, BDI, Biológicas e Hibridas.

2.3 Comunicaciones

La comunicación entre agente es un tema importante y como se señalo la comunicación entre ellos permite sincronizar acciones, enviar y recibir conocimiento, resolver conflictos en la resolución de tareas, etc, los componentes fundamentales para poder establecer la comunicación entre agentes son:

Protocolo de transporte: Mecanismo de transporte o protocolo de bajo nivel usado (TCP/IP, SMTP, HTTP,etc).

Lenguaje Común: para indicar el contenido de la comunicación pudiendo ser una pregunta, aceptación, de petición, solicitud de ejecución o cualquier otro contenido.

Protocolo de Interacción: Patrones que suele seguir la secuencia de mensajes intercambiados entre agentes.

Respecto a la estrategia de comunicación se tienen, comunicación restringida a un número fijo de coman-

dos y con una interpretación predefinida, otro enfoque es la arquitectura de pizarra, donde los agentes dejan información o mensaje con resultados totales o parciales y toman información o mensajes útiles para su interés, también usando RPC(Remote Procedure Call) es un mecanismo de comunicación y finalmente usando lenguajes de comunicación de alto nivel o interacción entre agentes a nivel de conocimiento.

3 Arquitectura De Sistemas Multiagente (Mas)

En este punto lo que queremos es distribuir en muchos agentes diversas tareas para cumplir con el objetivo global de resolver un problemas, esto no es un tema simple de resolver, se tienen los mismos problemas que en la construcción de sistemas distribuidos y la complejidad que agrega la comunicación y la integración de agentes.

Para el desarrollo del software de simulación se ha definido la especificación de la arquitectura FIPA (the foundation for intelligent, physical agents), que es un estándar bastante conocido al alero de una organización sin fines de lucro, cuyo propósito es promover el desarrollo de estándares o especificaciones genéricas para que un MAS opere en forma interna o relacionada con otro MAS, de la mejor manera y constituye un marco normativo dentro del cual los agentes pueden existir, funcionar y ser gestionados, Los principios de FIPA son [4]:

- La tecnología de Agentes proporciona un nuevo paradigma para resolver problemas.
- Algunas technologies de Agentes han alcanzo un considerable grado de madurez.
- Para hacer uso de la tecnología de agentes es necesario estandarizar.
- La estandarización de tecnologías genéricas es posible y provee un buen resultado.
- La estandarización interna de un agente no es lo principal, más bien importa la infraestructura y lenguajes abiertos para la operación entre agentes y MAS.

Estas especificaciones consisten es una plataforma de agentes de software y el objetivo principal es el intercambio de mensajes con contenido semántico entre agentes, pudiendo utilizar diferentes mecanismos de transporte, lenguajes de comunicación o lenguajes de contenido, apuntando a la interoperabilidad entre agentes, e incluye mecanismos para la creación , registro, localización y transferencia de mensajes entre agentes. En la figura 5 se muestra el modelo de referencia de agentes FIPA.

El estándar define solo el comportamiento externo, esto es la interfaz y deja las decisiones de diseño interno del agente a cada equipo de desarrollo.

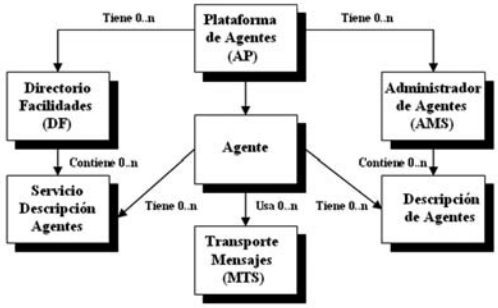

Figura 5: Plataformas y Contenedores

Dentro de las definiciones más importantes de este modelo [4] están:

Plataforma de Agentes (AP): Proveen la infraestructura física donde los agentes de desenvuelven.

Agente: Agente será un proceso computacional.

Sistema Administrador de Agentes (AMS): Es el responsable de la gestión de la operación de un AP, realiza tareas como la creación y supresión de agentes, la supervisión de agentes, la migración hacia y desde una plataforma. Cada agente debe registrarse con en AMS con el fin de obtener un AID que luego es registrado por la AMS en un directorio con todos los agentes presentes y su estado actual

Servicio de Transporte de Mensajes (MTS)

Servicio proporcionado por un AP para el transporte de mensajes entre agentes y entre AP., la comunicación entre agentes es mediante ACL (Agent Comunication Lenguaje)

Directorio de Facilidades (DF)

Provee un directorio de servicios que proporcionan los agentes, se definen dos tipos de servicios, Servicio de Directorio o Pizarra: Lugar donde los agentes registran información y Servicio de Transporte de Mensaje: Se establece un estándar de codificación, semántica y se transmiten sobre un medio de transporte.

La especificación tiene un centenar de documentos donde se incluyen especificaciones entre otras: Guía del Desarrollador, Servicio de Ontología, Gestión de Soporte de la movilidad y seguridad de agentes, interfaz hombre-agente, etc.

Otros estándares son: KQML (Knowledge Query Meta-Language). MASIF (Mobile Agent System Interoperability Facility).

4 Metodología De Desarrollo

El desarrollo de un sistema requiere la adopción de una metodología, que incluya soporte y cobertura de todas las fases del ciclo vida. La aplicación de una metodología supone estructuración, método, modelos, simplificación de lo complejo entre muchas otras características, metodologías tales como GAIA[5], Análisis y Diseño basado en UML [6], MAS CommonKADS[7] y MAD-SMART[8], son algunos de las revisadas para este trabajo y en general proporcionan un marco adecuado para su aplicación, como toda herramienta, su correcto uso queda en manos del equipo de trabajo. En la tabla 1, se puede ver algunas de las metodologías orientadas a agentes [9], las cuales se dividen en 2 categorías: Ingeniería de Conocimiento (IC) y Orientado a Objeto (OO).

Metodología	*Autor*	*Categoría*	*Referencia*
AAII	Kinny, Georgeff &Rao	IC	[Kinny 1996, Rao 1995]
ADEPT	Jennings et al.	IC	[Jennings 2000b]
AO methodology for	Kendall et al.	OO	[Kendall 1995]
Enterprise modeling AOR	Wagner	OO	[Wagner 2000, 2003] [Odell 2000]
AUML	Odell et al.	OO	[Collinot 1996]
Cassiopeia	Collinot et al.	OO	[Brazier 1997]
DESIRE	Brazier et al.	OO	[Wooldridge 2000, Zambonelli 2003]
GAIA	Wooldridge et al.	OO	[Debenham 2002]
OPEN Agents	Debenham & Henderson	OO	[DeLoach 2001, DeLoach 2003, Wood 2001]
MaSE	DeLoach & Wood	OO	[Iglesias 1996]
MAS-CommonKADS	Iglesias et al.	IC	[Lind 2000]
MASSIVE	Lind	OO	[Caire 2001]
MESSAGE	Caire et al.	OO	[Chung 2000]
NFR	Chung	IC	[Burrafato 2002]
PASSI	Cossentino et al.	OO	[Padgham 2002a, 2002b, 2002c, 2005]
Prometheus	Padgham & Winikoff	OO	[Bush 2001]
Styx	Bush et al.	OO	[Bresciani 2002, Bresciani 2004, Castro 2002, Giunchiglia 2002, Mylopoulos 2002, Perini 2001, Sannicolo 2002].
Tropos	Mylopoulos et al.	IC	

Tabla 1: Metodología Orientada a Agentes [9].

Para partir hemos adoptado la metodología MAS CommonKADS[8], que consta una fase de conceptualización utilizando la técnica de casos de uso basado en el usuario y de los modelos de: agente, tarea, coordinación, comunicación y organización, adicionalmente se define un modelo de diseño que decide que arquitectura es la más adecuada para cada agente y los requisitos de la infraestructura.

- Modelo de Agente: Identificación del agente, su descomposición funcional, el tipo de agente, objetivos que debe cumplir y otras características relevantes.
- Modelo de Tarea: Para cada agente se definen las tareas y un detallado de actividades
- Modelo de Coordinación: Especificación de las interacciones entre agentes.
- Modelo de Comunicación: Interacciones hombre-máquina, definición de transacciones e interacciones entre agentes.
- Modelo de Organización: Define las relaciones estáticas entre los distintos agentes, comunicaciones permitidas.

- Modelo de Experiencia: Identificación de tareas genéricas y su identificación en el modelo de tareas.

- Modelo de Diseño: Este modelo no es parte de la metodología pero es necesaria dado que proporciona los requisitos de la plataforma de desarrollo del agente, que en nuestro caso será JADE.

De forma práctica se estará abierto a utilizar diferentes metodologías o parte de ellas e incluso revisar otras. El objetivo aquí es poder modelar de la mejor manera nuestro sistema y en este sentido se procederá de la manera más flexible.

5 Herramienta De Implementación

Las herramienta para sistemas multiagentes que implementan el estándar FIPA encontradas en este trabajo y que están disponibles son: FIPA-OS, AGLETS, AJANTA, JADE, ZEUS y Grasshopper, entre otros y cada uno con sus consideraciones particulares, aún cuando se observa en general ambientes open-source, académicos y entornos JAVA.

Hemos elegido JADE que es una implementación del modelo FIPA en Java con licencia open-source, entrega una API que facilita el sistema de comunicación en forma transparente al usuario implementando todos los protocolos FIPA, dando soporte a ontologías, dentro de las características principales están:

- **Concurrente**: Funciona mediante hebras.

- **Distribuido**: Soporta múltiples servidores de diferentes arquitecturas.

Utilizando JADE podemos modelar el software requerido en cada componente, desarrollar sus interacciones sin necesidad de tener el hardware físicamente.

También podemos utilizar JADE como la capa de control que contenga todo el software para que cada componente ejecute sincronizadamente un conjunto de tareas. Básicamente la arquitectura dispone de plataformas, contenedores y agentes, como se muestra en la figura 6.

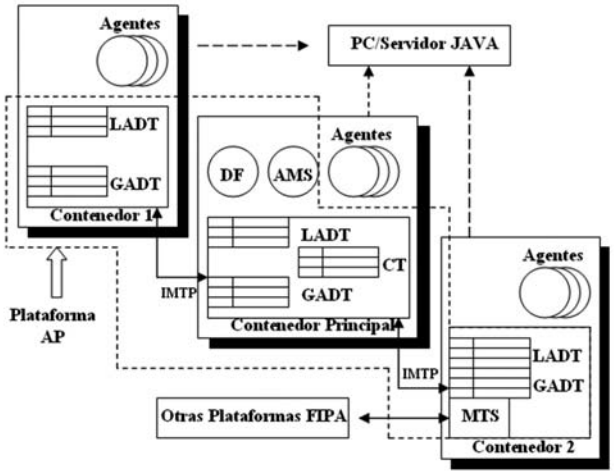

Figura 6: Plataformas y Contenedores JADE

Managing the Container Table (CT): Donde está el registro de las referencias de objetos y direcciones de todos los nodos que componen la plataforma.

Managing the global agent descriptor table (GADT): Es el registro de todos los agentes presentes en la plataforma, incluido su estado actual y ubicación;

AMS y el DF: Dos agentes especiales que proporcionan el agente de gestión y el directorio de servicios.

LADT (local agent descriptor table): Tabla local de descripciones de agentes presentes.

6 Modelo De Agentes

De acuerdo a la metodología seleccionada un primer paso en el diseño del sistema de simulación es construir el modelo de agentes. Como hemos visto podemos construir un ambiente de simulación de robótica industrial donde cada componente es modelado como un agente, se agregan otros para cumplir tareas de administración e interfaz del sistema, de acuerdo a los estándares FIPA y su implementación JADE.

En la figura 7, se puede ver la arquitectura de agentes del sistema, no se indican aquellos aportados por el estándar.

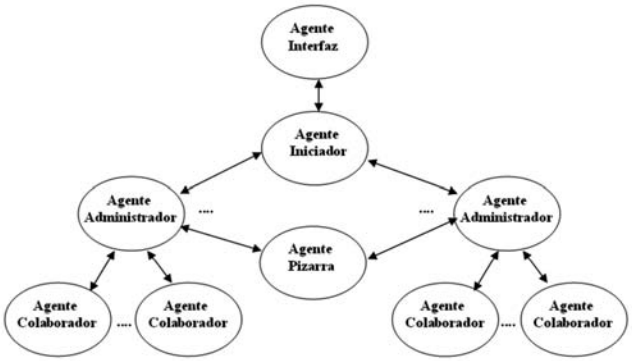

Figura 7: Plataformas y Contenedores

Agente Interfaz: Permite la interacción entre el usuario y el sistema de simulación, permite seleccionar los agentes colaborativos, indicando su topología y con la posibilidad de modificar sus atributos. Una vez iniciada la simulación este agente mostrará la información entregada por el agente pizarra.

Agente Iniciador: Este agente permitirá revisar y validar el escenario definido por el usuario e iniciara el registro en las tablas de definiciones CT, LADT y GADT con el agente administrador.

Agente Administrador: Este agente realizará las tareas que realiza el AMS del estándar FIPA.

Agente Pizarra: Este agente permitirá la comunicación entre los agentes colaborativos relacionados en la topología definida, será quien enviará los datos obtenidos al agente interfaz.

Agente Colaborativo: Corresponde a los agentes que implementa o simulan el comportamiento de las máquinas o artefactos que conforman el sistema, estos agentes están pre-codificados y son activados por el agente iniciador.

Metodológicamente con el modelo de agentes se debe continuar con la definición detallada de tareas, actividades y las demás etapas de CommonKADS.

7 Conclusiones

Multiagente nos da un marco de trabajo importante para el desarrollo de un sistema de simulación de celdas robóticas productivas, existen estándares, entornos de desarrollo disponibles y un conjunto de conceptos que nos permiten sistematizar el desarrollo utilizando este paradigma.

Disponer de modelos de manufactura flexible es de vital importancia para dotar al sistema de potencia y robustez técnica.

El estándar FIPA solo define aspectos externos y de comunicación de los agentes. Como son desarrollados en forma interna sigue siendo tema de los desarrolladores y la búsqueda de comportamiento inteligente sigue siendo un problema abierto, donde las arquitecturas cognitivas son un aporte a la forma en que se pueden desarrollar agentes inteligentes

Referencias

1. Modelo y Análisis de sistemas Flexibles de Manufactura en condiciones de alto rendimiento y flexibilidad, Manuel Sánchez y Bernal, Trabajo de Tesis presentada al programa de doctorado en Ciencias de la Ingeniería de con mención en Automática, Universidad de Santiago de Chile, 2004.

2. Jerarquía Dinámica de esquemas para la Generación de Comportamiento Autónomo, Tesis Doctoral, José María Cañas Plaza, Universidad Politécnica de Madrid, 2003.

3. An Introducction to MultiAgent Systems, Michael Wooldridge, John Wiley & Sons, Ltda.

4. Developing multi-agent systems with Jade, Fabio Bellifemine, Giovanni Caire y Dominic Greenwood, Wiley Series in Agent Technology.

5. Meta-Model Source Gaia, A. Garro, P.Turci, versión agosto 11, 2003, FIPA.

6. CommonKADS y el lenguaje de modelado unificado –UML, Pedro Salcedo Lagos, Universidad de Concepción.

7. Definición de una Metodología para el Desarrollo de Sistemas Multiagente, Tesis Doctoral, Carlos Iglesias, Universidad Politécnica de Madrid, 1998.

8. Metododogía para el Análisis y Diseño de Sistemas Multiagente Robóticas: MAD-Smart, J.Jiménez, Marcela Vallejo, John Ochoa, Revista Avances en Sistemas e Informática, II congreso Colombiano de Computación, 2007.

9. Uma Metodología Unificada para o Desenvolvimiento de Sistemas Orientados a Agentes, Tesis, Claudia Filomena Bratficher Dário, Universidade Estadual de Campinas. Facultad de Engenharia Electrica e de Computacao, 2005