Frameworks para el Desarrollo de Sistemas Multi-agentes

Martín Valacco, Analía Amandi

ISISTAN – Depto. Comp. Y Sistemas, Fac. de Cs. Exactas - UNICEN
Tandil – Bs. As., Argentina
{mvalacco, amandi}@exa.unicen.edu.ar

Resumen

En este artículo se presenta un análisis de los frameworks Brainstorm/J y Framas desarrollados en el Instituto de Sistemas ISISTAN. El análisis tiene como objetivo determinar las ventajas y las limitaciones de cada uno de ellos en el desarrollo de diversos tipos de agentes inteligentes. Entre los tipos estudiados se encuentran agentes reactivos, cognitivos, y móviles.

Introducción

Varias experiencias en el desarrollo de agentes han sido realizadas en los últimos años. Éstas han originado la necesidad del crecimiento del área de ingeniería de software, especializándola para el desarrollo de sistemas multi-agentes. En este contexto, arquitecturas de software ambientes especializados para el desarrollo de agentes fueron propuestos, juntamente con metodologías especialmente dirigidas a este tipo de sistemas.

Dentro de las herramientas de desarrollo de agentes, algunos frameworks fueron realizados contemplando generalmente características específicas tales como movilidad o reacción. Los frameworks permiten no sólo el desarrollo de agentes con estas características sino la adaptación de código, generalmente por especialización, para moldearlo a diferentes dominios. En este contexto, dos frameworks desarrollados en el Instituto de Sistemas ISISTAN de la Universidad Nacional del Centro han sido testeados y comparados en este artículo.

Este artículo está organizado de la siguiente manera. Seguidamente, dos secciones presentan los aspectos más relevantes de cada uno de los frameworks. A partir de ellos un análisis es presentado a través de comparaciones específicas.

El Framwork FraMaS

Para el desarrollo de FraMas [Avancini 00] se utilizó un enfoque bottom-up, es decir que primero se construyeron aplicaciones de agentes particulares y se fue abstrayendo comportamiento común de estas para generar las clases del framework.

La metodología de diseño seleccionada es la razón por la cual el framework no está basado en ninguna arquitectura de diseño particular, sino que consiste en un sistema orientado a objetos donde los problemas de diseño se solucionaron utilizando sólo patrones aplicables inicialmente a sistemas multi-agentes particulares.

En este framework, un agente posee un comportamiento básico al cual se le adiciona comportamiento inteligente. Es este comportamiento el que define al objeto agente como inteligente.

El framework posee una clase llamada *BasicAgentActions* la cual debe ser heredada por el objeto inicial del agente. El objeto inicial es aquel que provee todo el comportamiento no inteligente, o sea, el conjunto de acciones simples que es capaz de ejecutar cuando el componente de decisión lo indique.

En FraMas, el comportamiento inteligente corresponde a la comunicación, selección de la próxima acción y aprendizaje de las preferencias del usuario. Este comportamiento típico de agentes se adiciona por medio de decoradores (también conocidos como wrappers). Esta manera de agregar comportamiento a un agente está definida por el patrón de diseño Decorator [Gamma 95].

El patrón Decorator, también conocido como *Wrapper* permite agregar dinámicamente responsabilidades a un objeto. Los decoradores proveen una alternativa flexible a la herencia para extender funcionalidad. En la figura 1 se presenta la estructura genérica del patrón *Decorator*. A partir de este patrón de diseño, las acciones básicas que un agente puede realizar

encapsuladas en un objeto BasicAgentActions puede ser decorado dinámicamente con otros objetos que son responsables de la comunicación entre agentes, el manejo de preferencias, las reacciones, las decisiones deliberativas, etc.

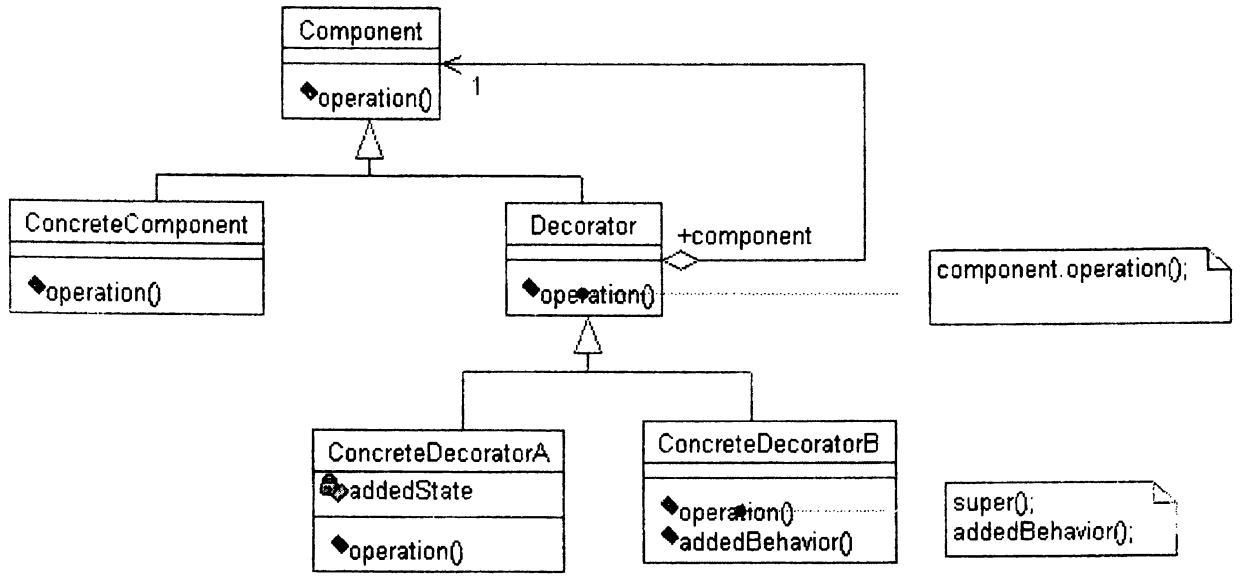

Figura 1. Bases del Framework FraMaS.

El Framework Brainstorm/J

El framework Brainstorm/J [Zunino 00] ha sido desarrollado en el lenguaje Java siguiendo los lineamientos de la arquitectura de agentes Brainstorm [Amandi 97]. Esta arquitectura prescribe agentes compuestos de un objeto base responsable de las acciones simples y determinísticas de los agentes y un conjunto de meta-objetos que interfieren en su computación haciendo que estas acciones sean combinadas convenientemente para alcanzar objetivos específicos, o para adicionar funcionalidad inherente a agentes inteligentes.

Las ventajas de este enfoque es que no son necesarias modificaciones en el código para modificar la estructura de un agente, ya que todo comportamiento es administrado a través de reflexión computacional.

La figura 2 muestra una estructura básica de la arquitectura Brainstorm, la cual fue mapeada a Java por el framework Brainstorm/J.

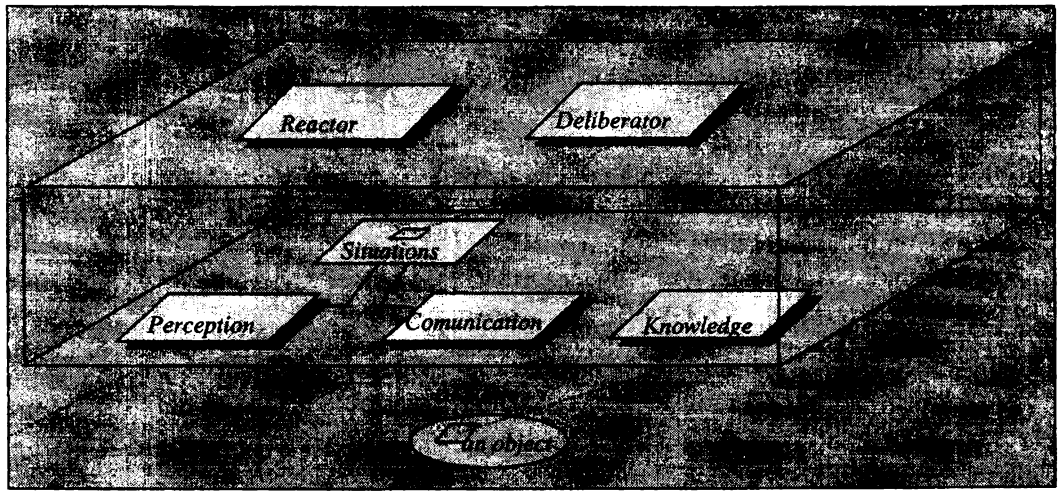

Figura 2. Esquema parcial de la arquitectura Brainstorm.

Comparación

Ambos frameworks permiten construir sistemas multi-agentes en el lenguaje Java. La primera diferencia es su concepción. Brainstorm/J mapea componentes arquitecturales de Brainstorm al lenguaje Java, teniendo que soportar para ello un sistema de meta-objetos. FraMaS es concebido partir de la abstracción de implementaciones específicas. Los resultados de las experimentaciones han mostrado que FraMaS tiene una orientación más marcadas hacia agentes de interfaz, resultado proveniente de la utilización de implementaciones en este contexto en la abstracción de clases del framework.

En cuando a su funcionalidad, la tabla presentada a continuación resume varios aspectos de estos frameworks.

Descripción	Brainstorm/J	FraMaS
Comunicación	Mensaje send esplícito	Mensajes entre objetos
Movilidad	Dévil (uso de rmi)	Dévil (uso de rmi)
Deliberación	Manejo de conversaciones Estrategias deliberativas	Soporte para incorporación de estrategias deliberativas.
Reacción	Uso de templates	Implementación esxplícita
Aprendizaje	Interferencia del meta-objeto learning en el proceso deliberativo	Implementación explícita

Tabla 1. Comparación de Brainstom/J y FraMaS.

La tabla anterior muestra algunos aspectos de funcionalidad junto con su solución. Se puede observar que FraMaS implementa una forma de comunicación más natural en ambiente orientados a objetos, pero ofrece menos soportes en aspectos deliberativos.

Otros puntos importantes considerados en el análisis son el tiempo de respuesta a un pedido determinado y la complejidad de modificación de las estructuras comportamentales de los agentes en tiempo de ejecución. El resultado del análisis es el siguiente:

Descripción	Brainstorm/J	FraMaS
Tiempo de resp.	10% más por uso de meta-objetos	4% más por delegación
Modificabilidad	Transparente	Control de wrappers

Tabla 2. Usabilidad de los frameworks.

Conclusiones

En este artículo fue presentado un análisis comparativo entre dos frameworks para sistemas multi-agentes que fueron concebidos utilizado dos metodologías diferentes: abstracción por ejemplos y dirección por una arquitectura específica. Este análisis conjuntamente con las experiencias realizadas han mostrado la utilidad de cada uno de ellos en diferentes contextos de requerimientos de sistemas de agentes.

Referencias

[Avancini 00] Avancini, H. FraMaS: Un Framework para Sistemas Multi-Agente basado en Composición. Tesis de maestrado. Universidad Nacional del Centro, Fac. de Ciencias Exactas, Departamento de Computación y Sistemas, mayo, 2000.

[Amandi 97] Amandi, A.; Price, A. Object-Oriented Agent Programming through the Brainstorm System. Proceedings of PAAM'97. London, April, 1997.

[Gamma 95] Gamma, E.; et.al. Design Patterns. Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995.

[Zunino 00] Zunino, A. Brainstorm/J: un framework para agentes inteligentes. Tesis de maestrado. Universidad Nacional del Centro, Fac. de Ciencias Exactas, Departamento de Computación y Sistemas, abril, 2000.