JADE - Java Agent Development Framework

1st Bryan Mendoza
Facultad de Ingeniería
Universidad de Cuenca
Cuenca, Ecuador
bsteven.mendoza7@ucuenca.edu.ec

2nd David Romero
Facultad de Ingeniería
Universidad de Cuenca
Cuenca, Ecuador
david.romeroa@ucuenca.edu.ec

***Abstract*—Este informe presenta una descripción detallada de JADE (Java Agent Development Framework), una plataforma para el desarrollo de sistemas multiagente conforme a las especificaciones FIPA. Se describen los requisitos de hardware y software para su ejecución, las funcionalidades que ofrece a los desarrolladores, el proceso típico de desarrollo de software utilizando esta herramienta, y se incluye un ejemplo práctico de uso.**

***Keywords*—JADE, Sistemas Multiagente, MAS, Framework, Java, FIPA**

I. Introducción

Java Agent Development Framework (JADE) es un framework de software que facilita el desarrollo de aplicaciones multiagente conforme a las especificaciones FIPA. JADE puede considerarse un middleware que implementa una plataforma de agentes eficiente y facilita el desarrollo de sistemas multiagente. La plataforma de agentes JADE busca mantener un alto rendimiento de un sistema de agentes distribuidos implementado en lenguaje Java. En particular, su arquitectura de comunicación ofrece una mensajería flexible y eficiente, seleccionando de forma transparente el mejor transporte disponible y aprovechando la tecnología de objetos distribuidos integrada en el entorno de ejecución de Java. JADE utiliza un modelo de agente y una implementación en Java que permiten una buena eficiencia en tiempo de ejecución, la reutilización de software, la movilidad de los agentes y la implementación de diferentes arquitecturas de agentes [1].

En síntesis, JADE provee una plataforma multiagente conforme a FIPA, un paquete para el desarrollo en Java de agentes y un conjunto de herramientas gráficas para administrar y monitorear la ejecución de los agentes [2].

II. Requisitos de hardware y software

Para la ejecución de JADE es necesario contar con Java Runtime Environment (JRE) versión 5 o superior [3]. En cuanto a la memoria, la ocupación en tiempo de ejecución de JADE en un entorno MIDP 1.0 es aproximadamente 120 KB, aunque mediante la técnica de ROMización —que consiste en compilar JADE junto con la JVM— este consumo puede reducirse hasta cerca de 50 KB [3], [4].

El resto de los requisitos mínimos de hardware y software necesarios para un correcto funcionamiento de la plataforma se detallan en las Tablas I y II.

III. Funcionalidades

Según [5] y [2] las funcionalidades y características que JADE ofrece al programador de agentes incluyen:

- Plataforma de agentes distribuida. La plataforma puede dividirse entre varios hosts conectados mediante RMI (Invocación de métodos remotos). Solo se ejecuta una máquina virtual Java en cada host. Los agentes se implementan como subprocesos Java y residen en contenedores que les proporcionan soporte en tiempo de ejecución.
- Interfaz gráfica para administración remota de agentes y contenedores.
- Herramientas de depuración para facilitar el desarrollo.
- Movilidad de agentes dentro de la plataforma, incluyendo transferencia de estado y código.
- Ejecución en paralelo de múltiples agentes mediante el modelo de comportamientos, programados de forma no preemptiva.
- Plataforma compatible con FIPA que incluye AMS, DF y ACC, componentes activados automáticamente al iniciar la plataforma.
- Transporte eficiente de mensajes ACL, codificados como objetos Java para evitar overhead.
- Biblioteca de protocolos de interacción FIPA lista para usar.
- Registro y baja automática de agentes en el MAS.
- Servicio de nombres compatible con FIPA, asignando GUID a los agentes.
- Soporte para lenguajes y ontologías de contenido definidos por la aplicación.
- Interfaz para que aplicaciones externas puedan iniciar agentes autónomos.

IV. Proceso de desarrollo de software usando la herramienta

El proceso para desarrollar aplicaciones multiagente con JADE consta de las siguientes etapas:

1) **Diseño del sistema multiagente:** Identificación de agentes, definición de comportamientos, protocolos de comunicación y diseño del entorno.
2) **Configuración del entorno de desarrollo:** Instalación de Java JDK y configuración del proyecto con la biblioteca JADE.
3) **Implementación de agentes:**

TABLE I
REQUISITOS DE SOFTWARE PARA JADE

Componente	Requisito
Lenguaje	Java (JADE está escrito en Java)
JDK Recomendado	Java Development Kit (JDK) 8 o superior (compatible con Java 8-17)
Sistema operativo	Cualquier SO que soporte Java
Entorno de desarrollo (opcional)	Eclipse, IntelliJ IDEA, NetBeans
Librería JADE	`jade.jar`

TABLE II
REQUISITOS MÍNIMOS DE HARDWARE PARA JADE

Recurso	Requisito mínimo
Procesador	Intel Core i3 o equivalente
Memoria RAM	4 GB (mínimo) – 8 GB o más recomendado para simulaciones grandes
Espacio en disco	200 MB disponibles (para Java, JADE y proyectos)

- Creación de clases que extienden `jade.core.Agent`, la clase base para agentes JADE.
- Esta clase define las características básicas para que el agente interactúe con la plataforma, tales como:
  - Registro en la plataforma.
  - Configuración y administración remota.
  - Envío y recepción de mensajes ACL.
  - Uso de protocolos de interacción estándar.
  - Soporte para un modelo computacional multitarea que permite definir múltiples comportamientos concurrentes.
  - Identificación y ciclo de vida del agente: inicialización, ejecución y finalización.

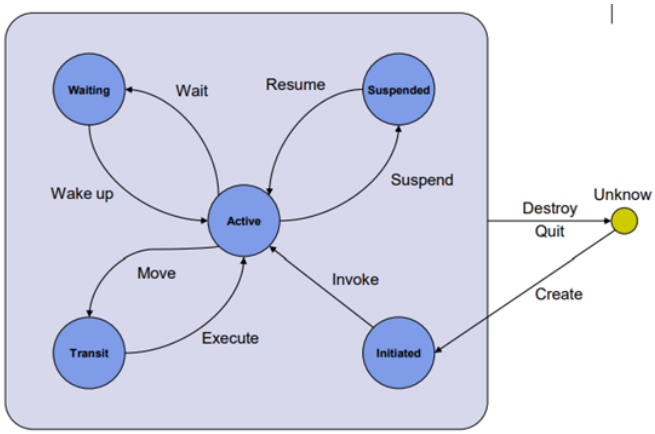

Fig. 1. Ciclo de vida de un agente

- En esta etapa se define el método `setup()` y se implementan los comportamientos del agente, utilizando clases como `OneShotBehaviour` o `CyclicBehaviour`.

4) **Comunicación entre agentes:** Implementación de mensajes ACL para el intercambio de información, utilizando filtros y plantillas para controlar los mensajes entrantes y salientes.
5) **Ejecución y monitoreo:** Lanzamiento de la plataforma JADE, despliegue de agentes y monitoreo mediante la interfaz gráfica que proporciona la plataforma.

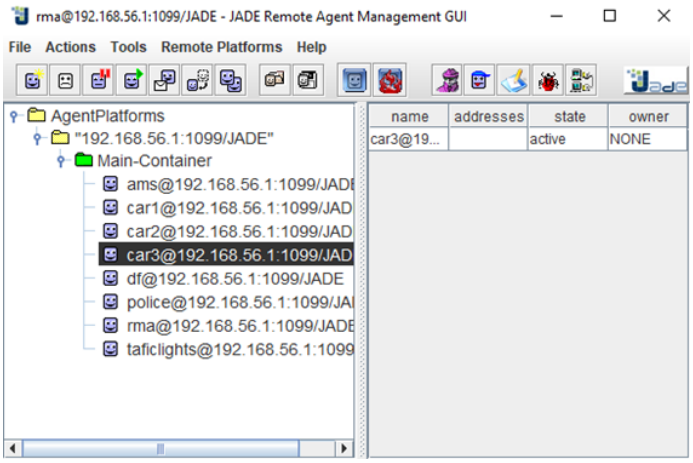

Fig. 2. Interfaz de JADE

6) **Pruebas y depuración:** Validación del comportamiento de los agentes y ajuste del sistema mediante logs y simulaciones.
7) **Empaquetado y despliegue:** Exportación de la aplicación para su ejecución en entornos distribuidos o locales.

V. EJEMPLO DE USO

Como ejemplo práctico del uso de JADE, se presenta una simulación de un sistema de control de tráfico en una intersección, implementado como una aplicación multiagente. Este proyecto fue desarrollado por un tercero y se encuentra disponible públicamente en la plataforma GitHub [6].
El sistema incluye tres tipos de agentes: *Car* (automóvil), *Police* (policía de tránsito) y *TrafficLight* (semáforo).

Cada agente automóvil es capaz de iniciar su recorrido, avanzar hacia su destino y detenerse si el semáforo está en rojo. Además, los coches pueden comunicarse con el agente policía cuando hay congestión o disputas de paso, utilizando

mensajes ACL. Por otra parte, el agente *TrafficLight* emite señales de luz verde o roja y el agente *Police* toma decisiones sobre la prioridad de paso, respondiendo a los coches según reglas predefinidas.

El comportamiento de cada agente se implementa mediante clases específicas que extienden la clase base `jade.core.Agent` y definen distintos `Behaviours`, como `GoToDestinationBehaviour`, `WaitGreenLightBehaviour` o `PriorityDecisionBehaviour`. La interacción entre los agentes se realiza de forma distribuida, simulando un entorno urbano inteligente.

Este ejemplo demuestra cómo JADE permite modelar y ejecutar un sistema distribuido, autónomo y reactivo en el que los agentes colaboran y negocian para resolver situaciones dinámicas, como las que se presentan en el tráfico urbano real.

VI. Conclusión

JADE es una herramienta robusta y flexible para el desarrollo de sistemas multiagente, que cumple con los estándares FIPA y aprovecha la plataforma Java para ofrecer portabilidad y eficiencia. Sus funcionalidades, como la movilidad de agentes, soporte para protocolos de interacción y herramientas gráficas de monitoreo, facilitan el desarrollo y la gestión de aplicaciones complejas distribuidas. Además, su arquitectura modular y escalable permite implementar desde prototipos simples hasta sistemas multiagente distribuidos en red. Por estas razones, JADE es una opción adecuada para investigadores y desarrolladores interesados en sistemas inteligentes y colaborativos.

References

[1] F. Bellifemine, A. Poggi, and G. Rimassa, "Developing multi-agent systems with jade," in *Intelligent Agents VII: Agent Theories, Architectures and Languages* (C. Castelfranchi and Y. Lespérance, eds.), pp. 89–103, Berlin, Heidelberg: Springer Berlin Heidelberg, 2001.

[2] A. Monteserin, "Jade – java agent development framework: Taller de sistemas multiagentes," tech. rep., ISISTAN – Facultad de Ciencias Exactas – UNICEN, Tandil, Argentina, 2017. Material académico. Prof. Dr. Ariel Monteserin. Email: amontese@exa.unicen.edu.ar.

[3] TILAB, "Jade – java agent development framework," s.f. Recuperado el 28 de junio de 2025.

[4] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa, "Jade: A software framework for developing multi-agent applications. lessons learned," *Information and Software Technology*, vol. 50, no. 1, pp. 10–21, 2008.

[5] F. Bellifemine, G. Caire, T. Trucco, and G. Rimassa, *JADE Programmer's Guide*. TILAB (Telecom Italia Lab), June 2007. JADE - Java Agent DEvelopment Framework. In compliance with FIPA specifications. Last update: 18-June-2007. Licensed under the GNU Lesser General Public License v2.1.

[6] mohamedzdb, "crossroad-with-jade-." https://github.com/mohamedzdb/crossroad-with-jade-, 2023. Accedido el 28 de junio de 2025.