Ejercicio introductorio (Sistema distribuido sencillo)

Sergio Alvarez

salvarez@cs.upc.edu

SID2022

https://kemlg.upc.edu

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA

JADE

Definición del problema

- Tenemos un sistema distribuido formado por agentes con diferentes capacidades:
  - Agente termómetro: genera mediciones de temperatura en determinados instantes de tiempo
  - Agente termostato: depende de la presencia de un agente termómetro para obtener las mediciones y actuar en base a la temperatura
    - Por ejemplo: si la temperatura es menor que 0 o mayor que 40 debería generar una alerta
- Cada grupo debe implementar un agente de un solo tipo (o termostato o termómetro)
- Únicamente usando la JDK

Toma de decisiones

- ¿Quién es termómetro? ¿Quién termostato?
- ¿Cómo recibe el agente termostato la información?
  - ¿Ha de esperar por las actualizaciones (push) o tiene que realizar una petición periódica (pull)?
- ¿Cómo se comunican los agentes?
  - Transporte: Socket/ServerSocket (TCP), DatagramSocket (UDP), HttpServer, …?
  - Codec: Strings, bytearrays, objetos, …?
  - Significado: ¿cómo se han de interpretar?
  - Protocolo: ¿ha de haber petición y respuesta? ¿Comunicación síncrona o asíncrona?
- ¿Cómo se encuentran/descubren los demás agentes?
- ¿Qué pasa si el agente con el que contactamos no está presente o falla?

Posibles ampliaciones

- Tercer agente: gestor de alarmas
  - Recibe (¿push/pull?) las alarmas generadas por el agente termostato
- Configurar la frecuencia de las actualizaciones
- Pedir información de diferentes termómetros, por si son inexactos o mienten
- Diferentes tipos de mensaje, ejemplos:
  - Solicitar que los siguientes mensajes tengan un formato distinto
  - Solicitar dejar de recibir actualizaciones o peticiones
  - Comunicar que el agente va a desaparecer o cambia de estado
  - Mensajes en secuencia (pregunta, respuesta, contrarréplica)
- Descubrimiento automático de los demás agentes de la red y sus capacidades
- Operar a través de Internet, cruzando firewalls

Problemas típicos de los sistemas distribuidos

- Las 8 falacias de los sistemas distribuidos (Deutsch, Gosling):
  - Nos podemos fiar de la red
  - La latencia es cero
  - El ancho de banda es infinito
  - La red es segura
  - La topología de red no cambia
  - Hay un administrador
  - El coste del transporte es cero
  - La red es homogénea
- Todo lo que puede ir mal, irá mal
  - Los mensajes pueden estar mal formateados o corrompidos
  - Los demás agentes pueden no ser confiables (pueden mentir)
  - Los demás agentes pueden no estar disponibles o pueden fallar en mitad de una conversación (fallos de CPU, RAM, excepciones)
  - Los relojes pueden estar desincronizados
  - La asincronía puede llevar a estados inconsistentes (CAP theorem)
  - Puede haber *race conditions* o *deadlocks*

Tecnología actual

- Plataformas/Infraestructura (con descubrimiento de servicios)
  - Kubernetes, Mesos, Consul, Zookeeper
- Microservicios
  - Spring Boot, Akka, Axon, Erlang/OTP, websockets
- Bases de datos distribuidas
  - Redis, Riak, Cassandra, Hbase, MongoDB, ElasticSearch
- Colas de mensajes y gestión de eventos
  - Kafka, RabbitMQ, EventStore
- Monitorización
  - Grafana, Prometheus, Datadog, Kibana

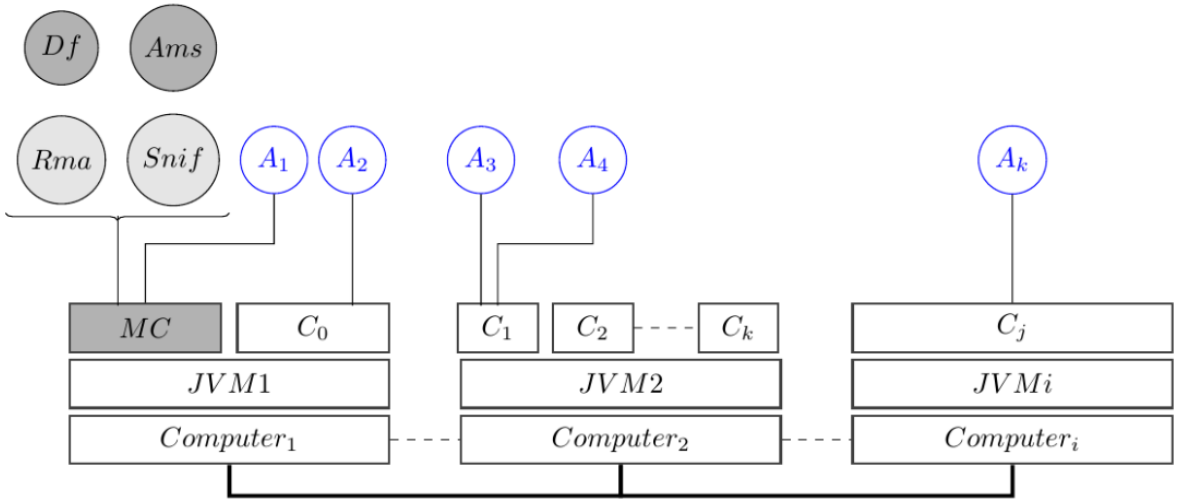
Tecnología actual (sistemas multi-agente)

- Sistemas basados en agentes (no distribuidos)
  - Agentes simples: Repast, Repast-HPC, NetLogo, MASON, Pandora, Pogamut
  - Agentes con razonamiento lógico: 2-APL, 3-APL, OperettA, GOAL
- Plataformas distribuidas de agentes
  - JADE
  - Jason (AgentSpeak)
  - Jadex
  - JACK
  - SPADE

JADE: Java Agent DEvelopment framework

- Software LGPL mantenido por Telecom Italia
  - https://jade.tilab.com/
- ¿Por qué JADE?
  - Plataforma realmente distribuida y totalmente asíncrona
  - Descubrimiento de agentes por nombre, tipo o capacidades
  - Abstracción para la comunicación entre agentes (cola de mensajes)
  - Abstracción para definir el comportamiento de los agentes
  - Compatible con estándares de la comunidad multi-agente (especificaciones FIPA)
  - Portabilidad de agentes a través de contenedores y plataformas
- Libro (copia en la biblioteca):
  - Developing Multi-Agent Systems with JADE (Bellifemine et al., 2007)

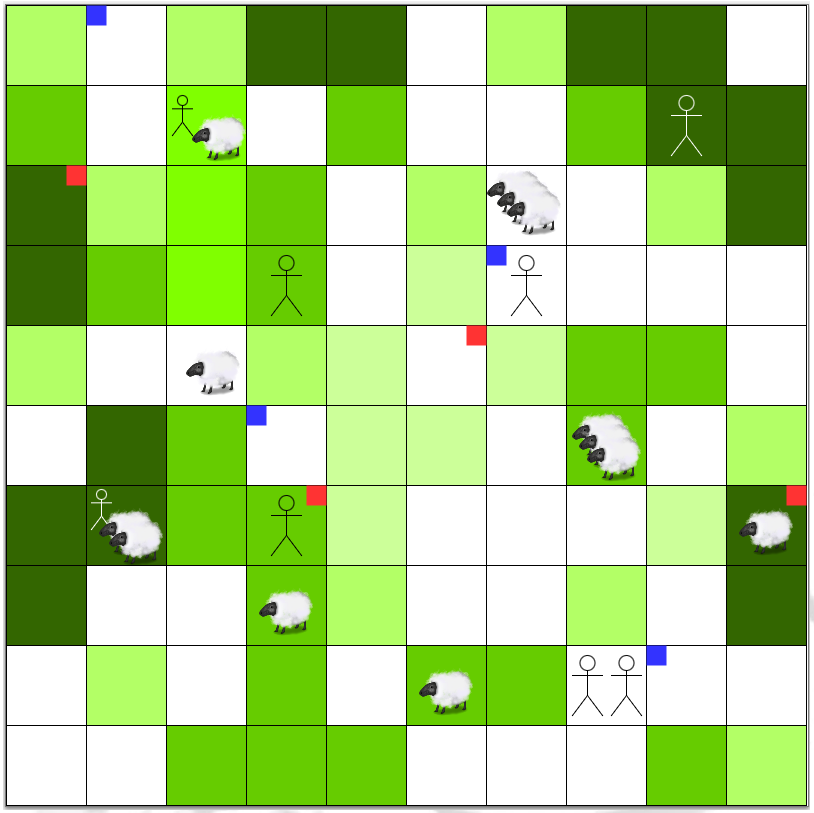
JADE: Java Agent DEvelopment framework



Mandatory ● DF : Directory Facilitator (Yellow page) ● AMS : Agent Management System (Platform control)
Optional ● RMA : Remote Management Agent (Gui for AMS) ● Sniffer : Catch messages on the fly (Debug)
Optional ● Ai : Your agents

Ejemplo de práctica con JADE (2021)

- Recursos limitados:
  - Ovejas, pasto
- Objetivo individual:
  - Maximizar beneficio (compraventa de recursos)
- Objetivo colectivo:
  - Evitar que los recursos se agoten
- 1 agente por cada grupo
- Evaluación:
  - Inyectar todos los agentes en la Plataforma
  - Ejecutar diversos entornos aleatorios (adaptación)



Knowledge Engineering and Machine Learning Group

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Sergio Alvarez

salvarez@cs.upc.edu

2022