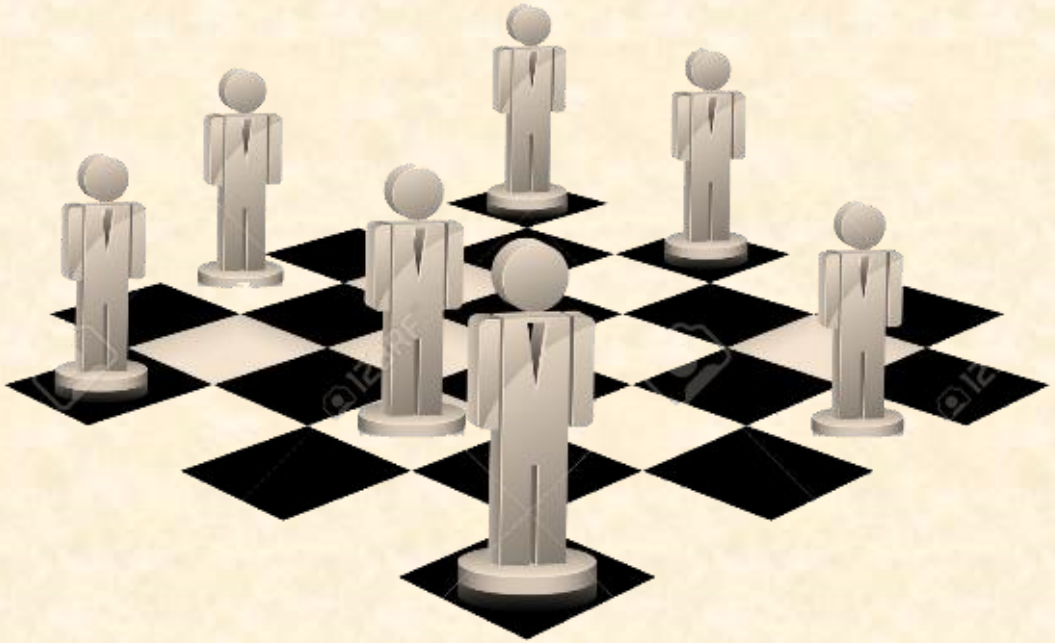
JADE

Java Agent Development Framework

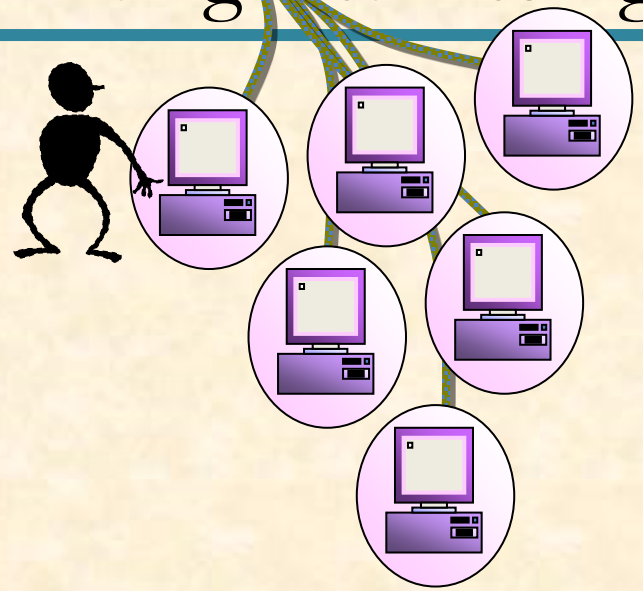


Plan

- Definition JADE
- Specification FIPA
- Composants de base
- JADE architecture et environnement
- Agents JADE
- Exemples

1. C’est quoi JADE?

- Il est basé sur une architecture distribuée, Compatible pour la communication avec des agents non Jade. Les agents Jade peuvent se déplacer d’une machine à une autre.
- JADE est un middleware qui se compiles avec les spécifications FIPA destiné au développement d’applications pair à pair d’agents intelligents.

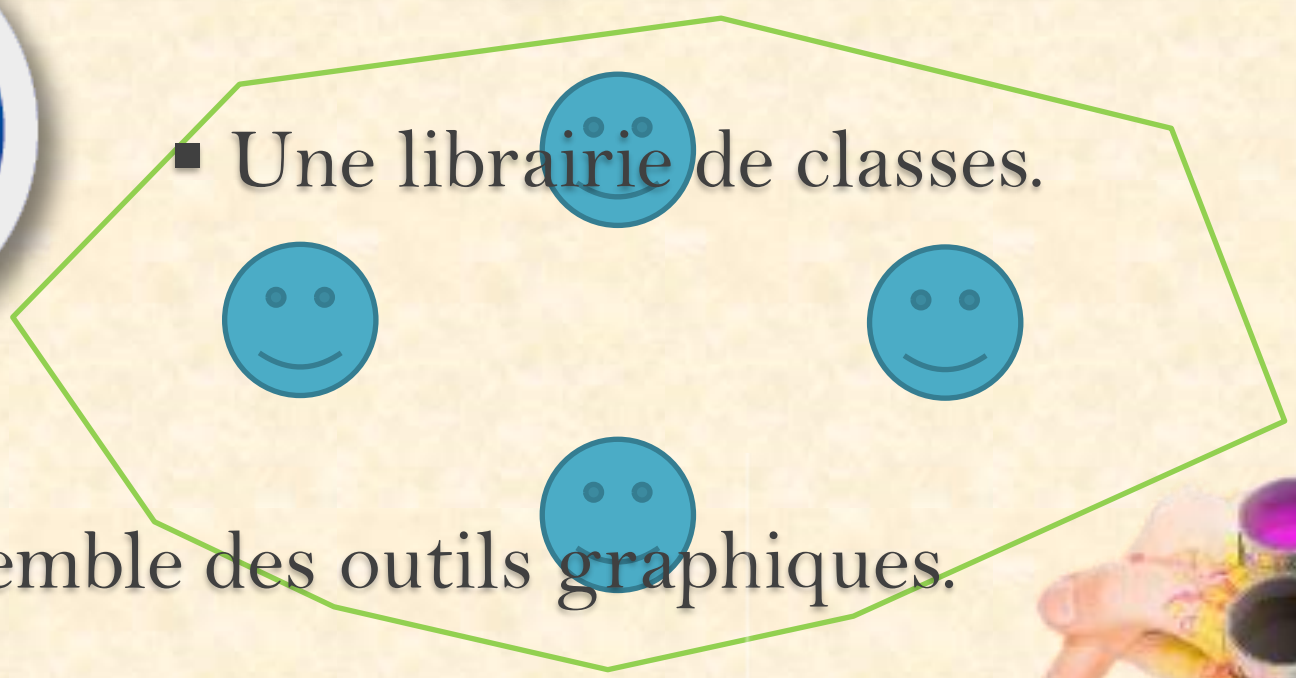


La norme FIPA
(Foundation for Intelligents Physical Agent)

- Formé en 1996 afin de construire des standards plateforme pour les agents hétérogènes, en interaction et les SMA.
- Les document FIPA établissent les règles **normatives** qui permettent à une société d'agents d'inter-opérer.
- Design des spécifications afin de faciliter l'intéropérabilité entre les différents SMA développés par différentes sociétés et organisations.
- Relations fortes avec d'autres standard et organisations.

Composantes de bases 2/2
API pour développer des agents en JAVA

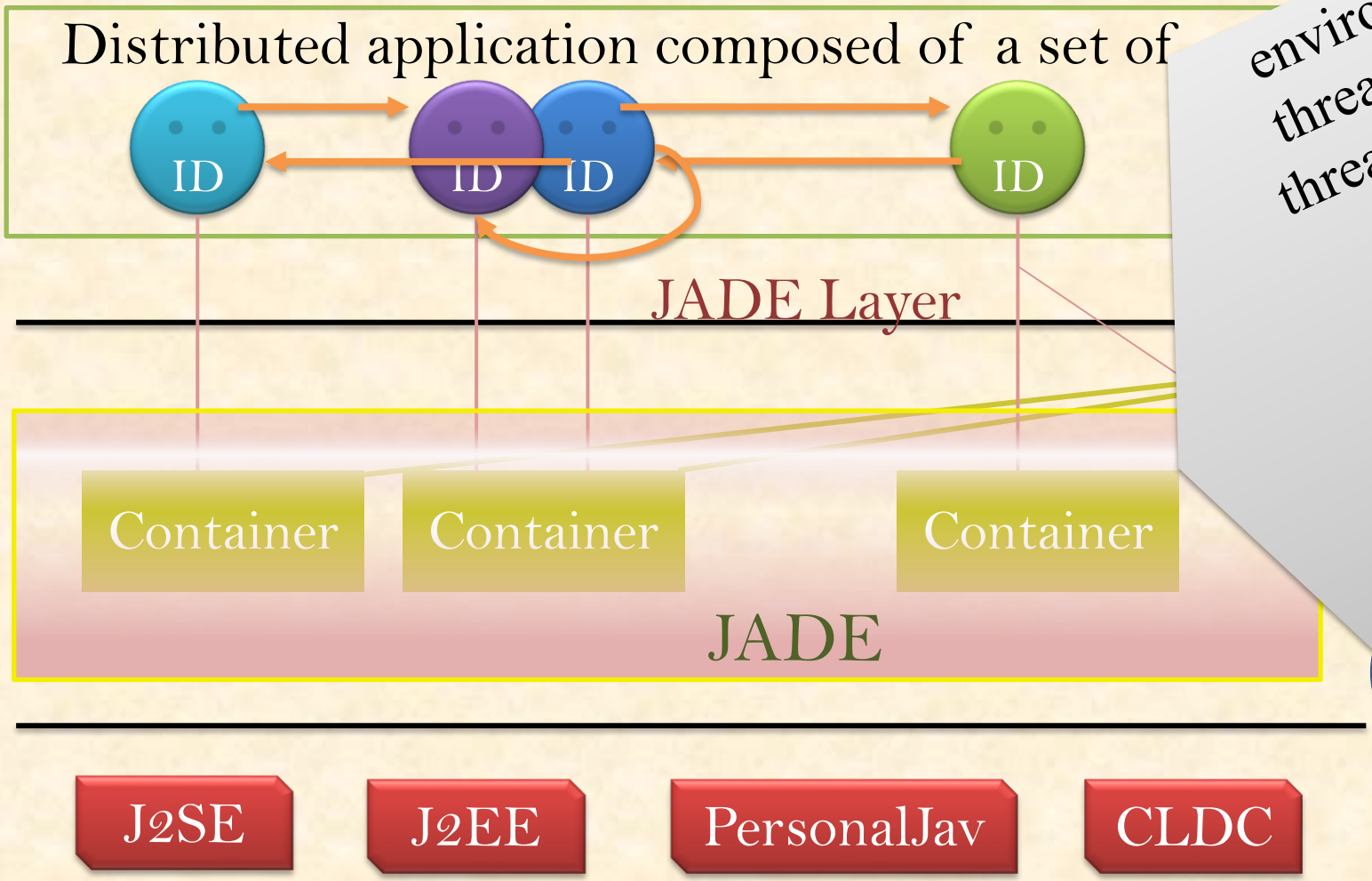
Environnement de développement et exécution Multi-Agents



- Une librairie de classes.
- Un ensemble des outils graphiques.

implémenté complètement en Java dis r JVM
à J2ME MIDP

JADE environnement et architecture



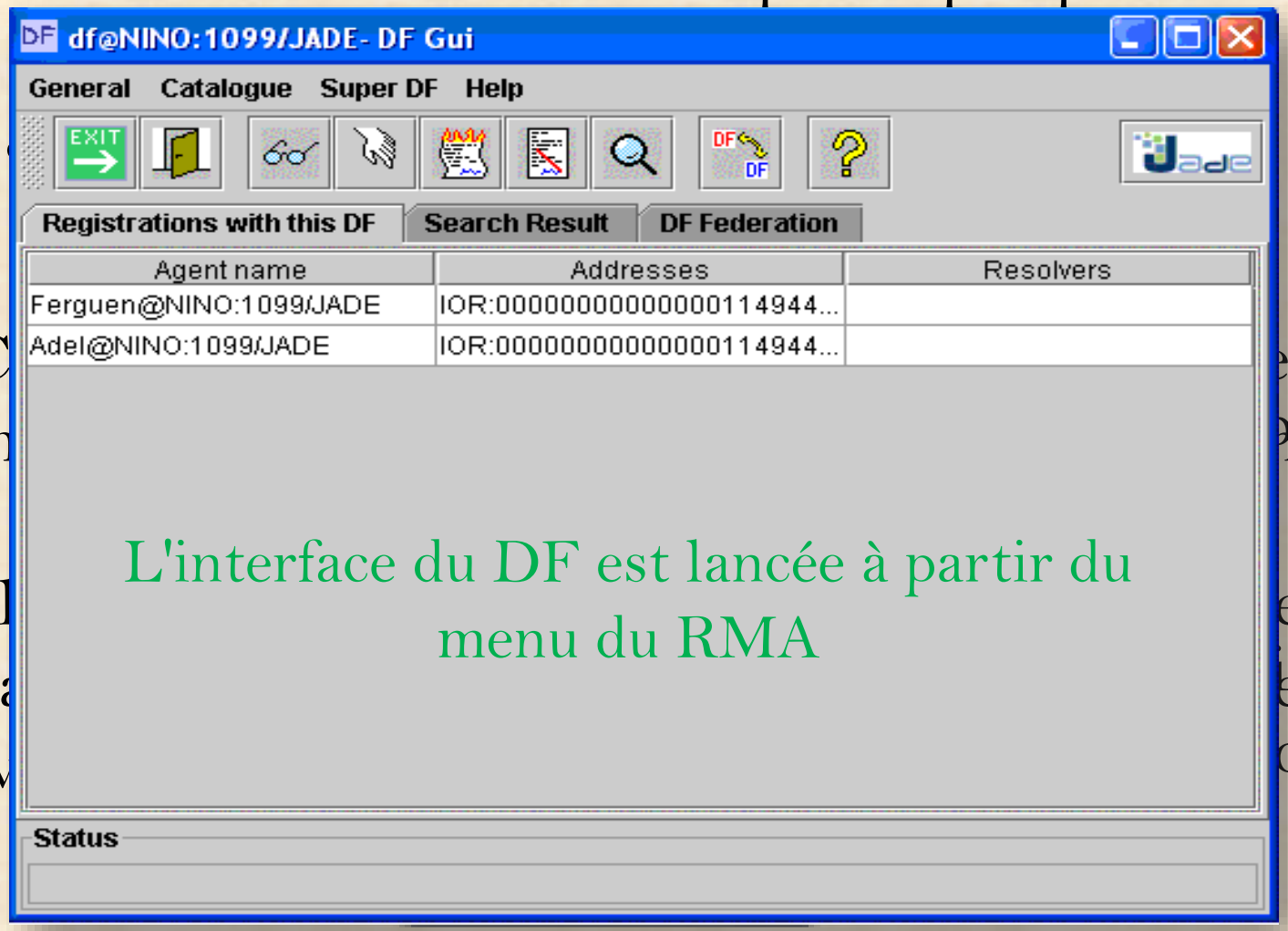
environnement multi-threads composé d'un thread d'exécution pour chaque agent

- gère localement un ensemble d'agents
- règle le cycle de vie des agents (création, attente et destruction)
- assure le traitement des communications

qui
un conteneur
autres
teneurs
ner) qui lui fournit
son environnement
d'exécution;

JADE environnement et architecture

Le conteneur principal possède 2 agents spéciaux :



- DF Directory Facilitator

est un composant qui fait office d'annu rvice de « ages jau t de mettre en rela vec leurs comp nt peut interrog nnaître les ompéten r les autres

Service de Pages Jaunes référence à leur demande les agents suivant leur(s) service(s).

Outils de débogage

Pour supporter la tâche difficile du débogage des applications multi-agents, des outils ont été développés dans la plate-forme JADE.
Chaque outil est empaqueté comme un agent

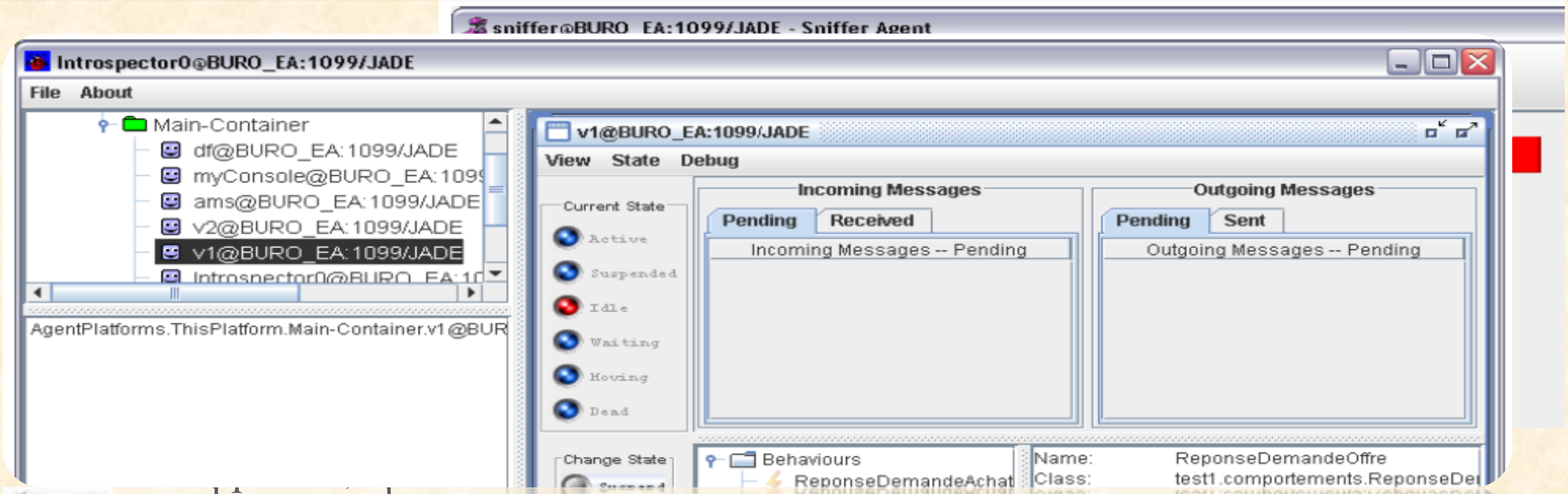
RMA (Remote Monitoring Agent)

- Visualisation et gestion (ajout, suppression ..) de :
  - l'ensemble des conteneurs déployés au sein d'une plateforme JADE
  - des agents présents au sein de la plateforme (inscrits ou non dans le DF), par accès à l'AMS.
- Possibilité d'avoir plusieurs RMA au sein d'une plateforme mais un seul par conteneur.

Lancement :
- par l'option **-gui** en ligne de commande

Ou
- java jade.Boot monInterface:jade.tools.rma.rma

Outils de débogage



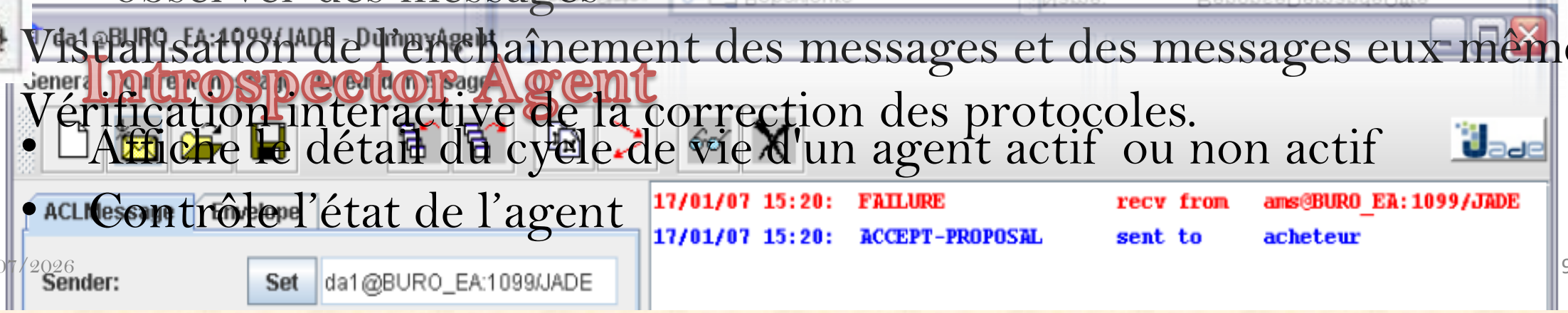
observer des messages

Visualisation de l'enchaînement des messages et des messages eux même.

Introspector Agent

Vérification interactive de la correction des protocoles.

- Affiche le détail du cycle de vie d'un agent actif ou non actif
- Contrôle l'état de l'agent



Agents JADE

- 1 agent = 1 thread implémenté en JAVA selon API JADE

hérite de la classe Agent *jade.core.Agent*

- Exécute un ensemble d'actions. Les actions sont regroupées en comportements (*behaviour*); Différents types de comportements : parallèle, composite, cyclique
- Possibilité de s'inscrire et Rechercher/Offrir un service
- Méthode *setup()* invoquée dès la création de l'agent
  - ajouter des comportements à l'agent *addBehaviour*()
  - l'inscrire auprès du DF *DFService.register*()
- Méthode *takedown()* invoquée avant qu'un agent ne quitte la plateforme
  - Demander au DF de supprimer les services qui ont été inscrits par l'agent
  - Finir de traiter les messages reçus…

Agents et communication

- Le langage de communication est FIPA-ACL (Agent Communication Langage), et en mode

Types

- Inte
- Exte P, JMS

Agent
Asma

Un message ACL dispose obligatoirement des champs suivants :

Performative :	type de l'acte de communication
Sender :	expéditeur du message
Receiver :	destinataire du message
reply-to :	participant de la communication
content :	contenu du message
language :	description du contenu
encoding :	description du contenu
ontoloy :	description du contenu
protocol :	contrôle de la communication
conversation-id :	contrôle de la communication
reply-with :	contrôle de la communication
in-reply-to :	contrôle de la communication
reply-by :	contrôle de la communication

Exemples : création d'un agent

```
import jade.core.Agent;

public class DécideurAgent extends Agent {
  protected void setup() {
    // Afficher un message d'accueil
    System.out.println("Hello! Décideur"+getAID().getName()+" est prét.");
  }
}
```

La méthode getAID() de class Agent permet de récupérer l'identifiant d'agent. Inclue un nom unique

Exemples : terminaison d'un agent

```
 Object[] args = getArguments();
    if (args != null && args.length > 0) {
      targetBookTitle = (String) args[0];
      System.out.println("Trying to buy
"+targetBookTitle); }
 else {
      System.out.println("No book title specified");
      doDelete();   }   }

protected void takeDown() {
        System.out.println("Buyer-agent
"+getAID().getName()+" terminating.");
  }
```

```
public int onEnd() {
myAgent.doDelete();
return
super.onEnd(); }
```