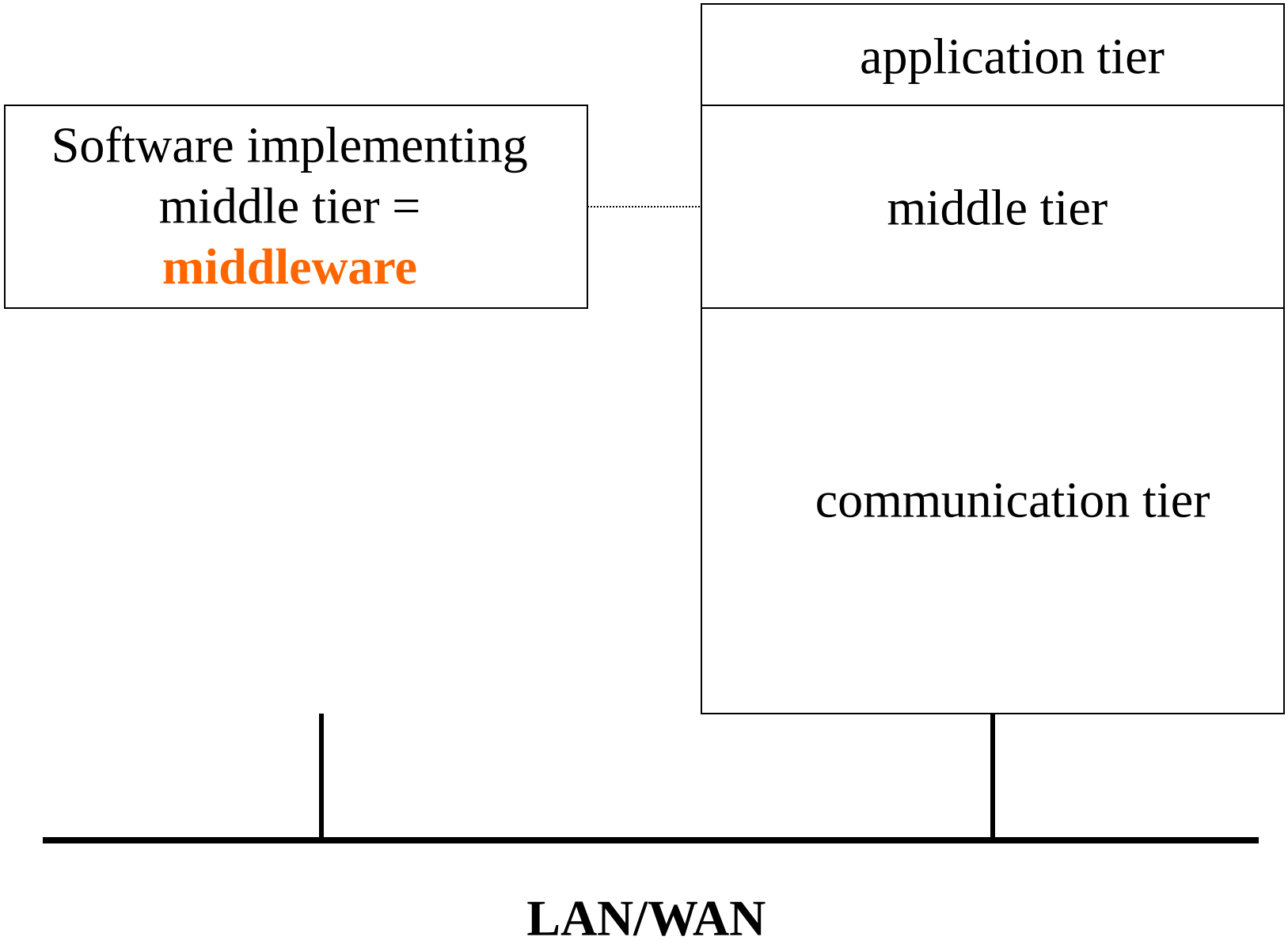
Multi-agent systems

Andrej Lúčny

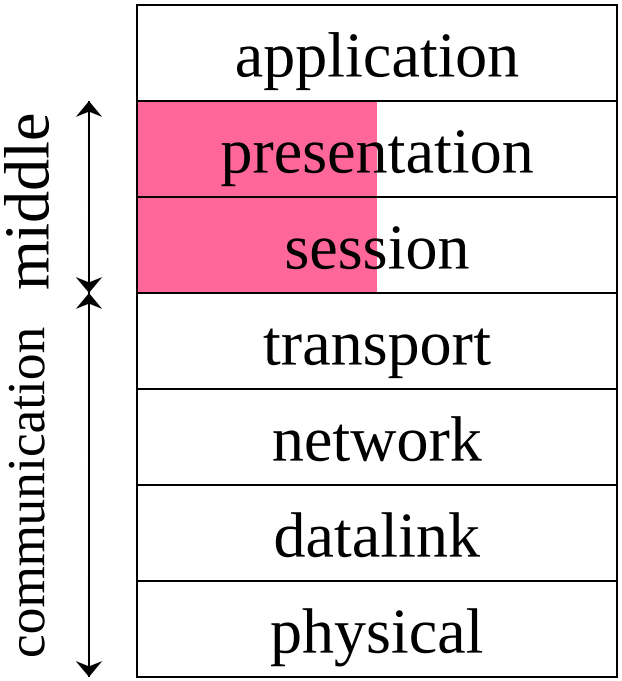
KAI FMFI UK

lucny@fmph.uniba.sk

http://www.agentspace.org/mas


application tier
Software implementing middle tier = middleware
middle tier
communication tier
LAN/WAN

MAS as middleware



- **MAS implementation is based on some existing communication base**
- **MAS is a special kinf od middleware, mostly based on message passing**
- **it can be based on another middleware** (typically based on distributed object)

MAS API

- What interface is provided to the application layer by a multi-agent system implementation?

There are two choices

- Prevailing direct communication (peer-to-peer) (e.g. JADE)
- Prevailing indirect communication (stigmergic communication) (e.g. Cougaar)

MAS with prevailing direct communication

Services of the direct communication

The basic service is

- asynchronous transmission of messages, i.e. calling a SEND method with convenient parameters

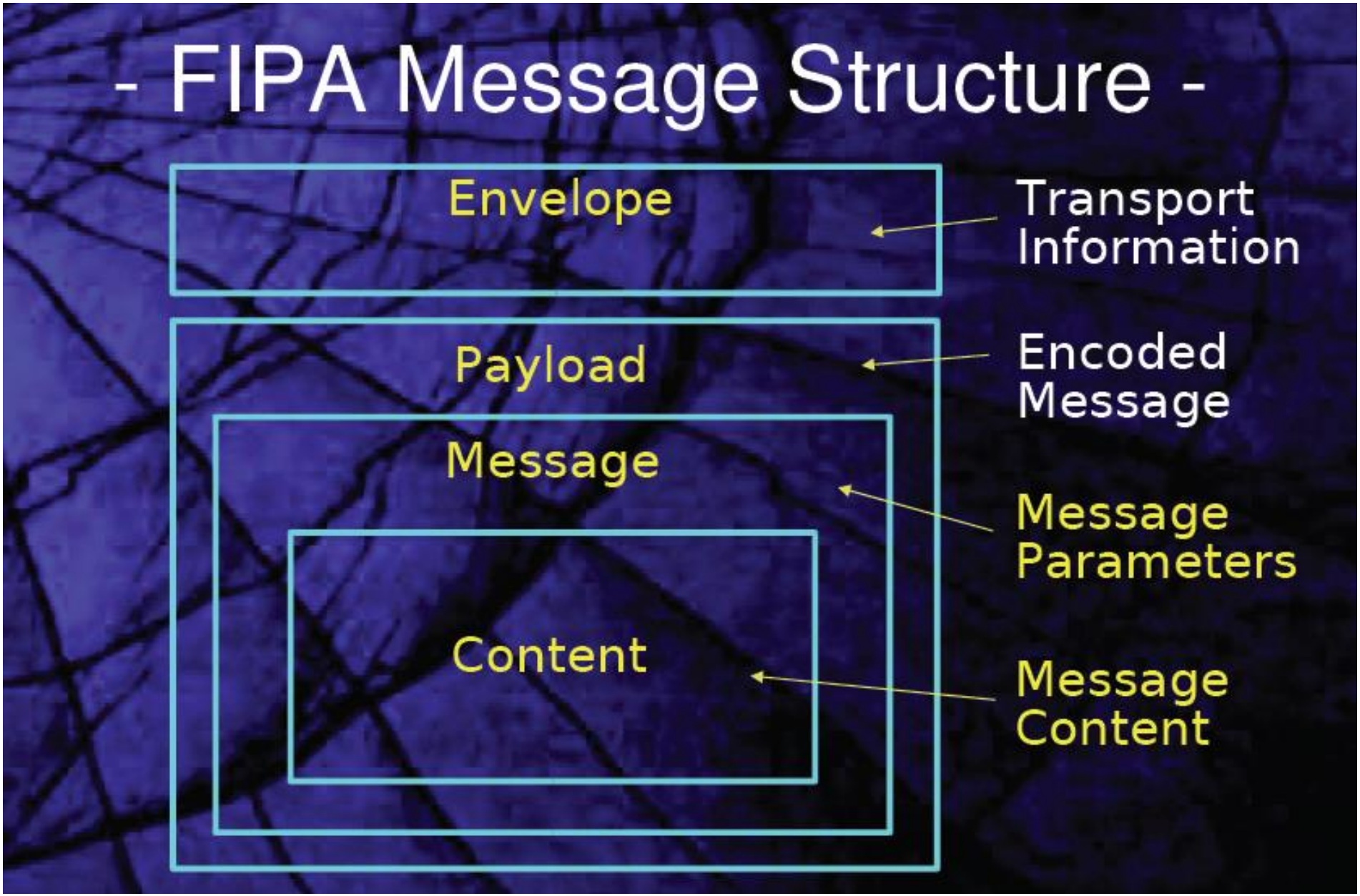
  in generall, this method can call directly a code of the receiver – so called callback (typical for actors) or push the message into a shared memory from which it is popped by the thread of the receiver (typical for agents) so the agent need to call RECEIVE method

- Java Agent DEvelopment Framework
- Middleware implemented via RMI
- It provides
  - communication among agents (ACL) and its initialization (AMS, DF)
  - paralel course of agents (Jade container)
- Implemention of FIPA 97 in Java

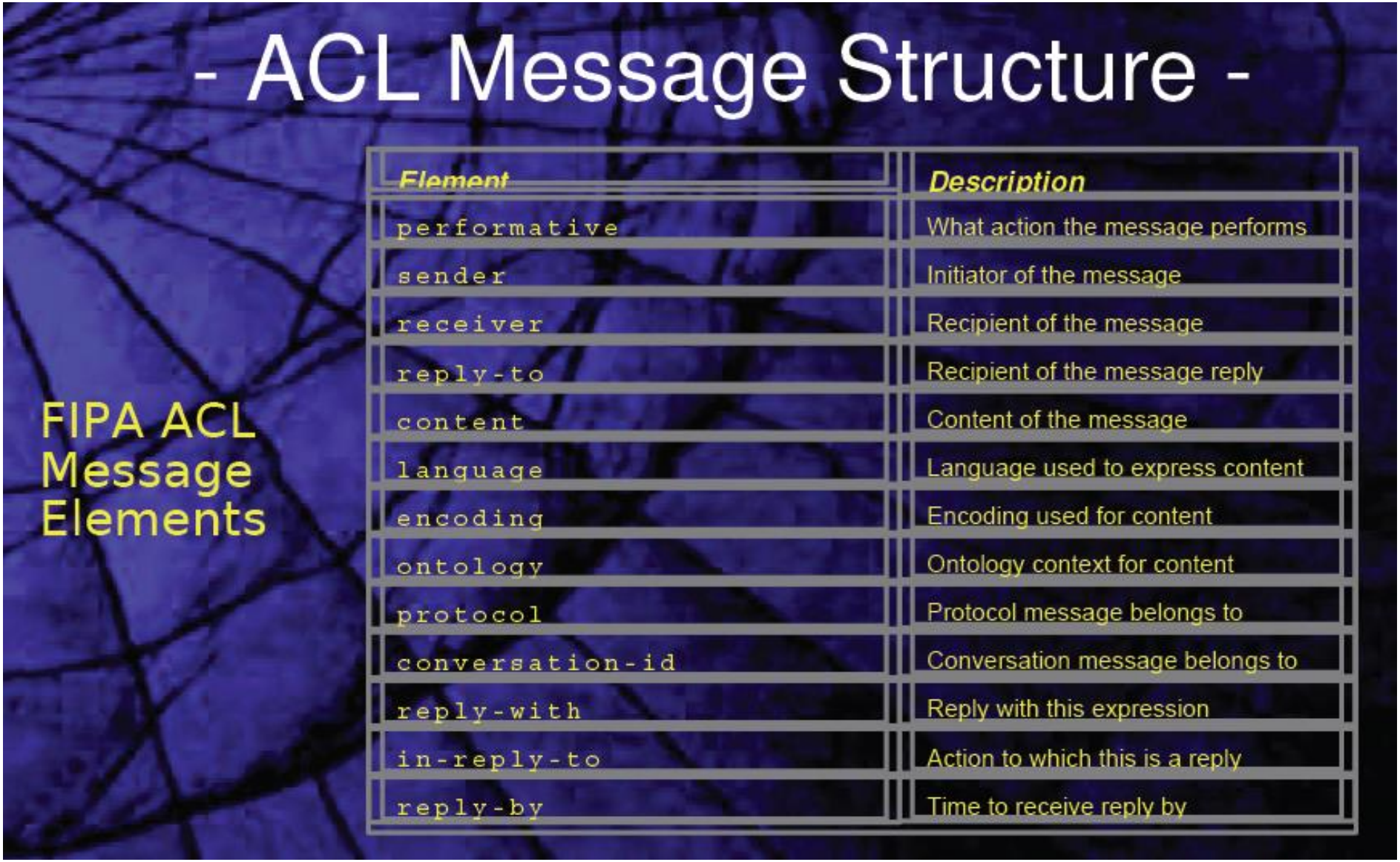
FIPA 97

- Started work in 1997
- At peak comprised 60 members
- Primary specifications became standard in 2002
- Work ongoing in Modeling, Methodology, Semantics & Services

ACL = Agent Communication Language



Attributes of ACL performatives



Example of message in ACL (& FIPA-SL)

Message sent via Java Serialization

(INFORM
 :sender (agent-identifier :name Writer@10.102.101.216:1099/JADE)
 :receiver (set (agent-identifier :name Reader@10.102.101.216:1099/JADE))
 :X-JADE-Encoding Base64
 :content
"rO0ABXNyAAlNeU1lc3NhZ2X3XJJaURjcRQIAAUwAB2NvbnRlbnR0ABJMamF2
YS9sYW5nL1N0cmluZzt4cHQACkFob2ogSmVsa2E="
 :language JavaSerialization)

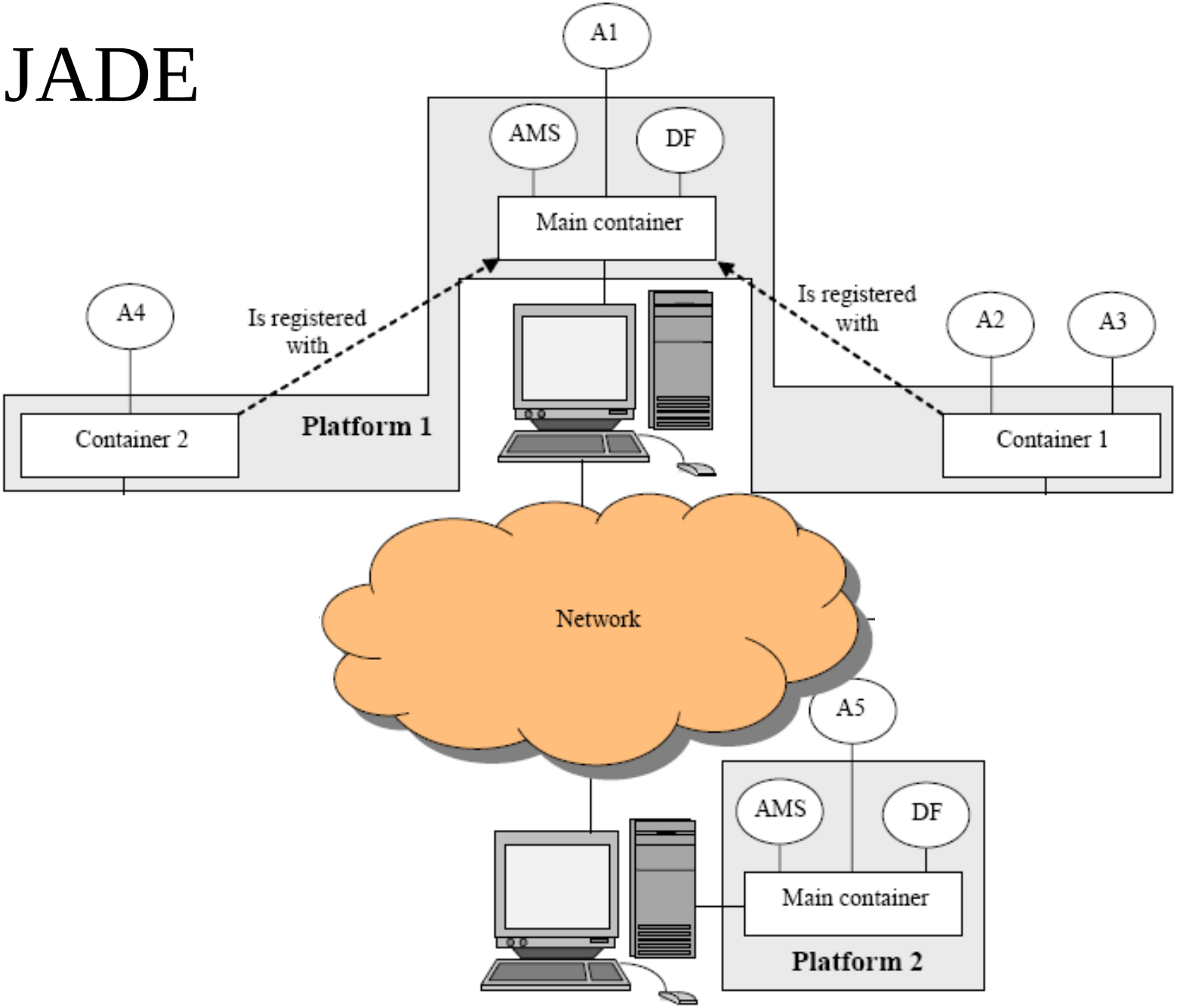
Message sent via XML

(INFORM
 :sender (agent-identifier :name Writer@192.168.1.15:1099/JADE)
 :receiver (set (agent-identifier :name Reader@192.168.1.15:1099/JADE))
 :content "<primitive type=\"FLOAT\" value=\"3.1415927F\"/>"
 :language XML :ontology FIPA-Agent-Management)

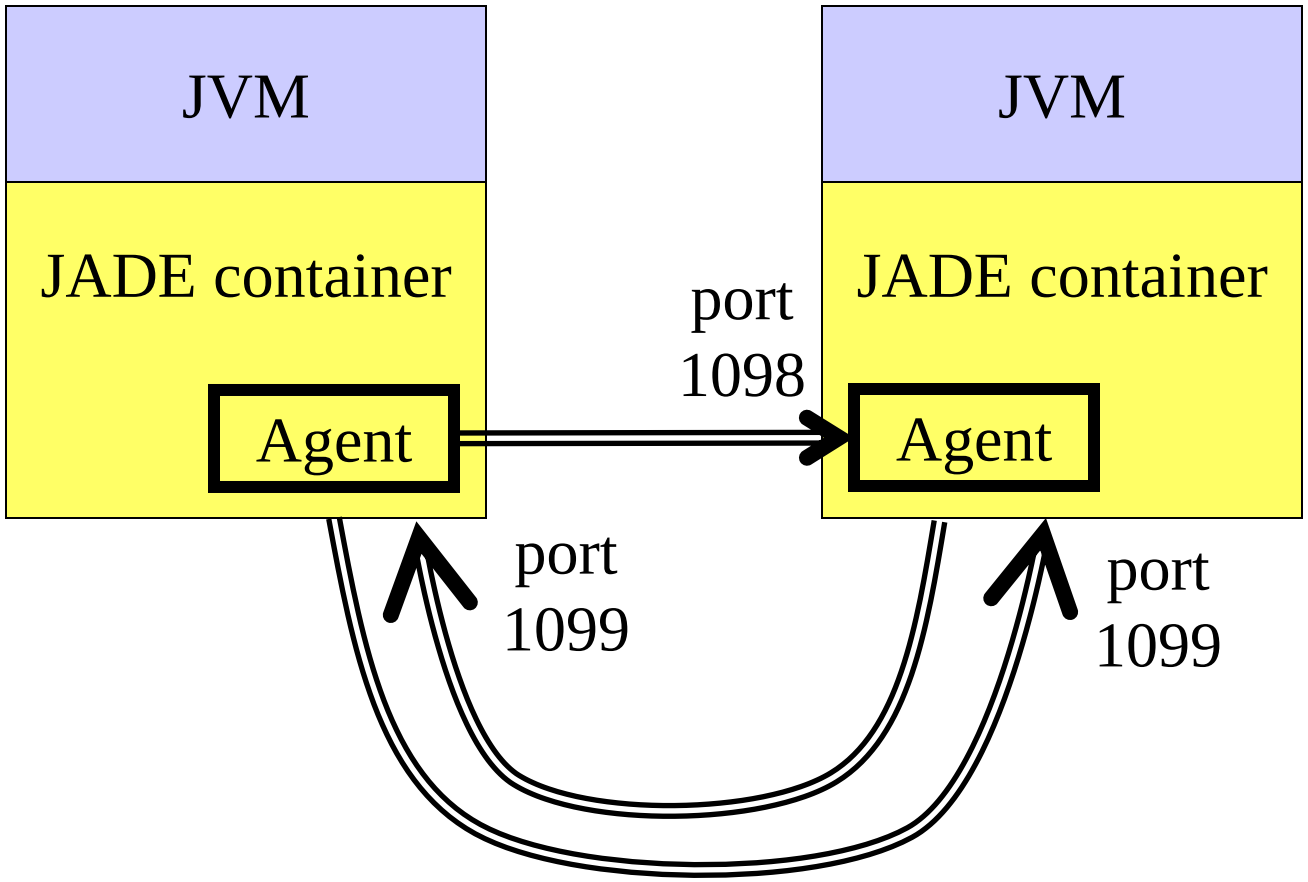
Content type in the ACL (Marshalling)

- FIPA-SL, FIPA-KIF
- Java Serialization (RMI)
- ACL/XML
- ACL/Bit-efficient

JADE


A1
AMS
DF
Main container
A4
Is registered with
Is registered with
A2
A3
Container 2
Platform 1
Container 1
Network
A5
AMS
DF
Main container
Platform 2

TCP sockets used by JADE



Jade container

- Asynchronous model of agents
- Agent has own thread, even more threads
- Container hides difference between local and network communication (it contains and hides all neccessary stubs)

Jade container

- Each container occupies one JVM
- There can be more containter on one node but just one of them can be the main container (and this one occupies the port 1099)
- Container can connect to a main containter on another node and from that moment agent in the both containers can communicate

Jade: Agent

- Agent beží len v containeri, ktorý realizuje jeho rozhranie
- Agent má metódu **`setup()`**
- V nej môže vo svojom vlákne realizovať svoju činnosť, ale častejšie je, že inicializuje rôzne správania metódou **`addBehaviour()`**

Jade: Behaviour

- SimpleBehavior
- CyclicBehaviour
- TickerBehaviour
- OneShotBehaviour
- ReceiveBehaviour
- WakerBehaviour
- ParallelBehavior

Jade: Behaviour – examples

```
protected void setup() {
  …
  addBehaviour(
    new CyclicBehaviour () {
      public void action () {
        …
    }
  );
}
```

```
 protected void setup() {
   …
   addBehaviour(
     new MsgReceiver (this, MessageTemplate.MatchAll(), Long.MAX_VALUE,
         new DataStore(), "ObjectReaderAgent") {
       protected void handleMessage (ACLMessage msg) {
         …
     }
   );
 }
```

Jade Agent Management System

- Each agent has unique AID and name and it is located in certain container
- For being visible by the other agents, it can register one or more services
- It can find AID of other agents by name of their services (yellow pages)
- It can find AID of other agents by their location (white pages)
- It can use ACL for mutual communication with those agents which AID it knows

Jade Directory Facilitator

- DF serves for registration of services
- It is able to organize the services in so called directories which are global, i.e. same for all containers.

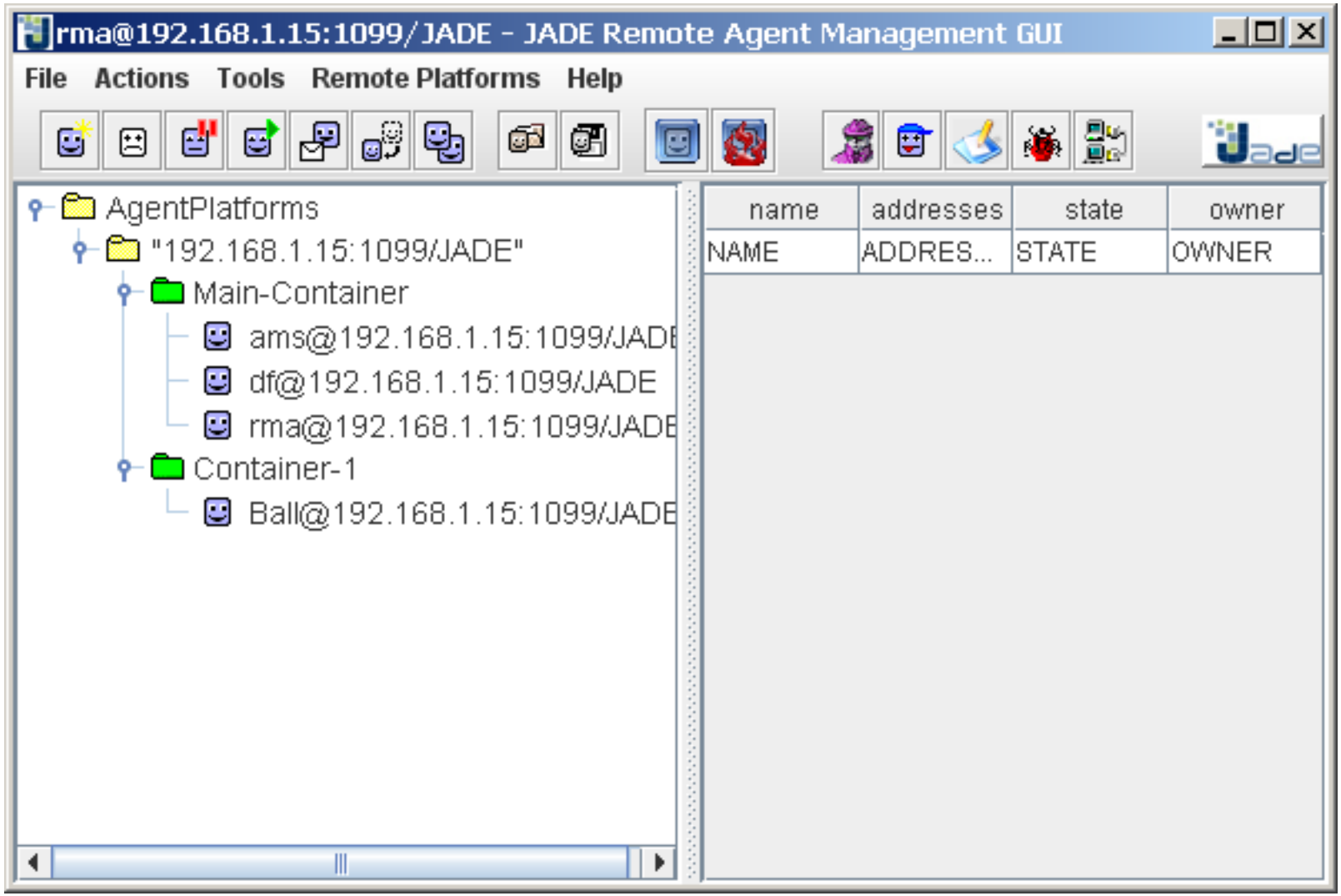
Registration

```
DFAgentDescription dfd = new DFAgentDescription();
ServiceDescription sd = new ServiceDescription();
sd.setType(„ServiceName");
sd.setName(getName());
dfd.addServices(sd);
dfd.setName(getAID());
try {
  DFService.register(this,dfd);
} catch (FIPAException e) {
}
```

Search for agent by service

```
AID reader;
DFAgentDescription dfd = new DFAgentDescription();
ServiceDescription sd = new ServiceDescription();
sd.setType("ServiceName");
dfd.addServices(sd);
try {
  for (;;) {
    DFAgentDescription[] result = DFService.search(this,dfd);
    if ((result != null) && (result.length > 0)) {
      dfd = result[0];
      reader = dfd.getName();
      break;
    }
    Thread.sleep(1000);
  }
} catch (Exception e) {
}
```

Jade GUI


rma@192.168.1.15:1099/JADE - JADE Remote Agent Management GUI
File
Actions
Tools
Remote Platforms
Help
AgentPlatforms
"192.168.1.15:1099/JADE"
Main-Container
ams@192.168.1.15:1099/JADE
df@192.168.1.15:1099/JADE
rma@192.168.1.15:1099/JADE
Container-1
Ball@192.168.1.15:1099/JADE
name
addresses
state
owner
NAME
ADDRES...
STATE
OWNER

More reading

- http://jade.tilab.com/
- http://jade.tilab.com/download/add-ons/
- http://www.iro.umontreal.ca/~vaucher/Agents/Jade/JadePrimer.html

MAS with indirect communication
is a distributed system of type:

Peer-to-peer

- The central parts of the system are minimized to a naming service
- Each node is a server for all other peers and a client of that peers