Software agent computing

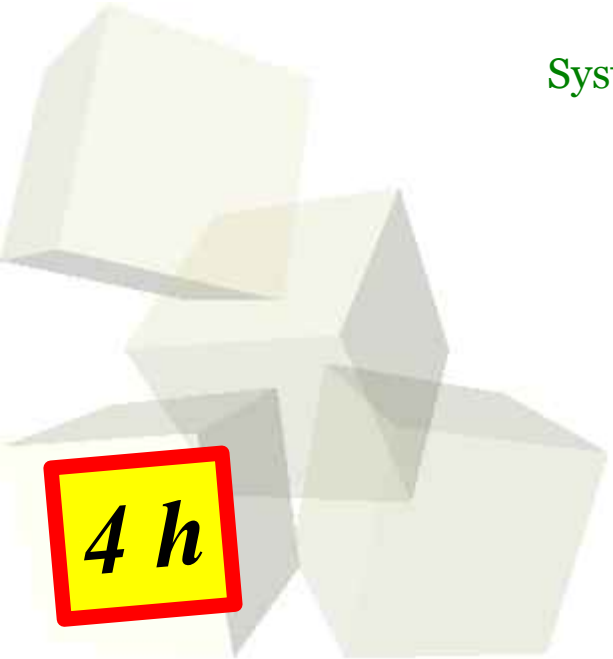
2nd laboratory activities
at Warsaw University of Technology

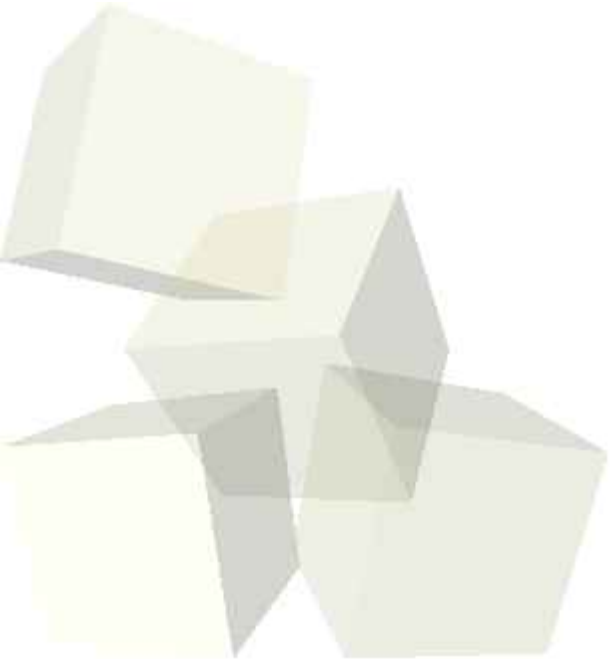
Maciej Gawinecki

Systems Research Institute, Polish Academy of Sciences

maciej.gawinecki@ibspan.waw.pl

http://www.ibspan.waw.pl/~gawinec





Agent-oriented methodology

Technological challenge

Methodology for software development is intended to:

- ✔ discipline software development by defining set of **guidelines** for covering the whole **lifecycle of system development**
- ✔ define the **abstraction for modelling** software (*object-oriented, agent-oriented, data-oriented, knowledge-based* etc.)

Chris Preist
HP Laboratories Bristol, UK

et al. say*

*"One of the most fundamental obstacles to large-scale take-up of agent technology is the **lack of mature software development methodologies** for agent-based systems."*

* M. Luck, P. McBurney, Ch. Preist (2003). *Agent technology: Enabling next generation computing: A roadmap for agent-based computing*. AgentLink report., http://www.ecs.soton.ac.uk/~mml/papers/al2roadmap.pdf

Exisiting methodologies

- Exisiting methodologies
  - MaSE

    Scott A. DeLoach, Mark F. Wood and Clint H. Sparkman, *"Multiagent Systems Engineering"*, *The International Journal of Software Engineering and Knowledge Engineering*, Volume 11 no. 3, June 2001, http://www.cis.ksu.edu/~sdeloach/publications/Journal/MaSE%20-%20IJSEKE.pdf
  - Tropos

    Perini, P. Bresciani, F. Giunchiglia, P. Giorgini, and J. Mylopoulos. "*A knowledge level software engineering methodology for agent oriented programming*". *Autonomous Agents*, Montreal CA, May 2001, http://www.auml.org/auml/supplements/Bresciani-Agents2001.pdf
  - JADE-oriented

    Magid Nikraz1a, Giovanni Caireb, Parisa A. Bahri (2006), "A Methodology for the Analysis and Design of Multi-Agent Systems using JADE", http://jade.tilab.com/doc/JADE_methodology_website_version.pdf
  - Other: Gaia, Prometheus – *described on 1st lecture*!

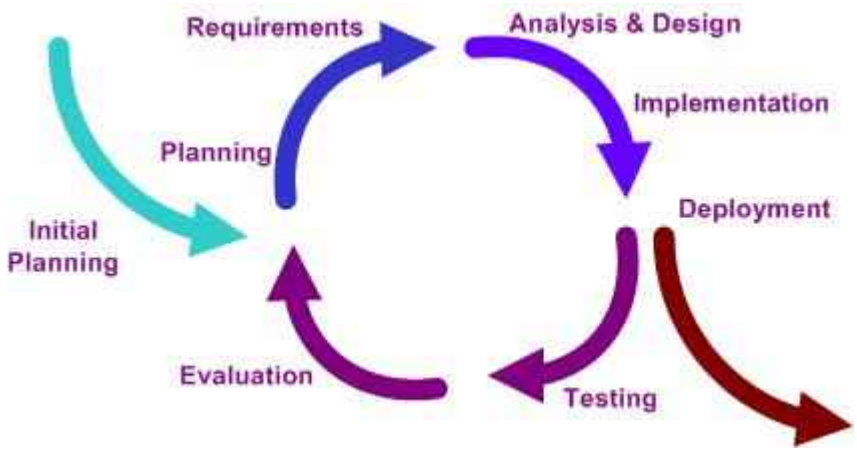
See 1st Lecture for details.

- Comparision:which one is better ?
  - Onn Shehory, Arnon Sturm, *Methodologies for Agent-Oriented Software Engineering*, Presentation from EASSS 2006, Annecy, France.
    - ➔ I've got copy for you :-)
    - ➔ See links there

Ask **Kate Slezavina,**
Project **#3**: *Designing eLearning environment with three different agent-oriented methodologies*

Software development lifecycle

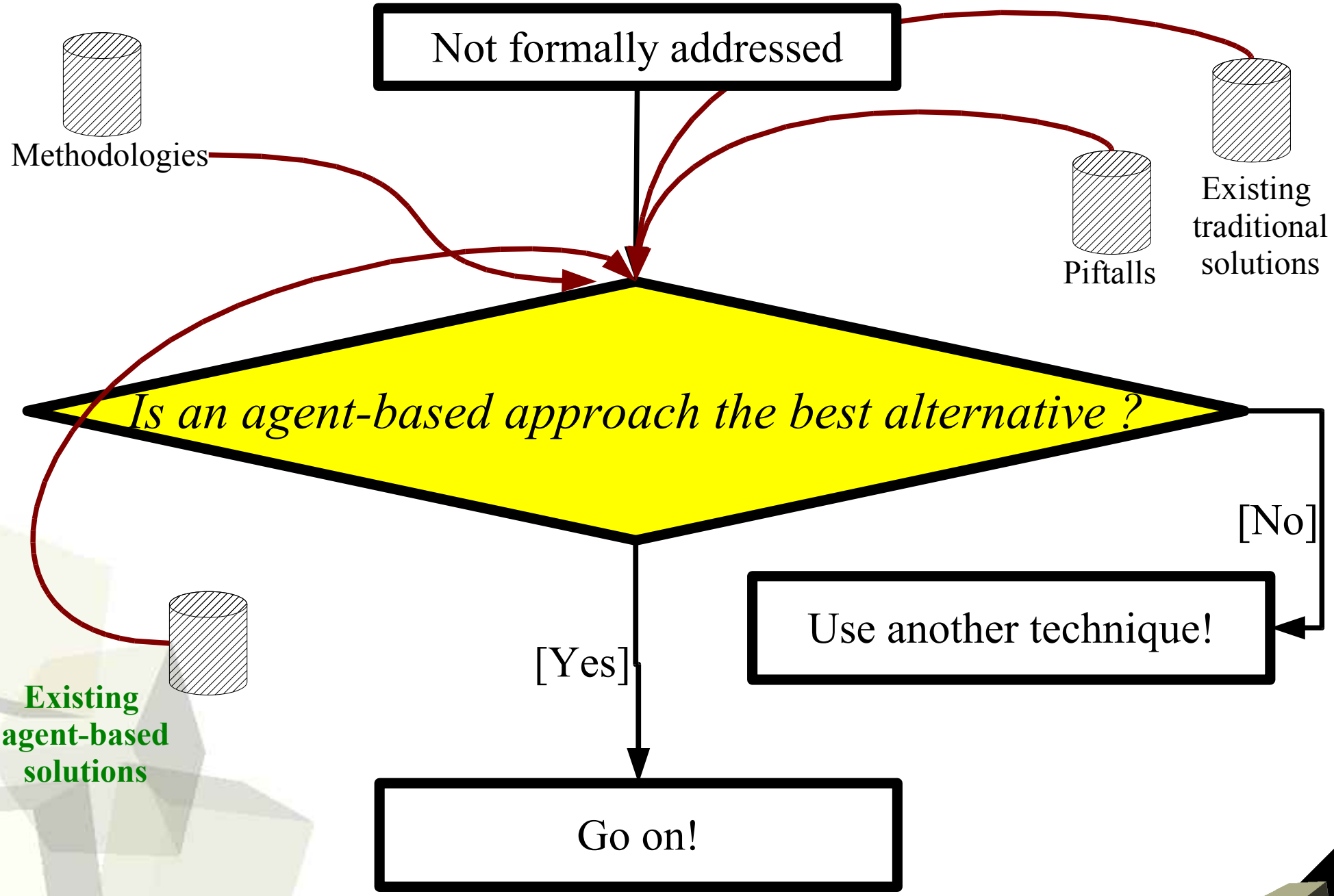
- Planning
- Analysis
- Design
- Implementation
- Testing



I propose methodology based on
✔ *JADE-oriented* methodology
✔ *Prometheus* methodology
✔ own experience

Planning phase

Planning phase



Philosophy of agent modelling

Prof. Nick Jennings
School of Electronics & Computer Science
University of Southampton, UK

says*

- Agents mainly should be used for modelling

  ✔ Decentralized nature of a problem
  ✔ Many points of control
  ✔ Various perspectives
  ✔ Competitive tasks

*Nicholas R. Jennings. *An agent-based approach for building complex software systems*. Commun. ACM, 44(4):35–41, 2001.

PLANNING PHASE

Examples and contr-example

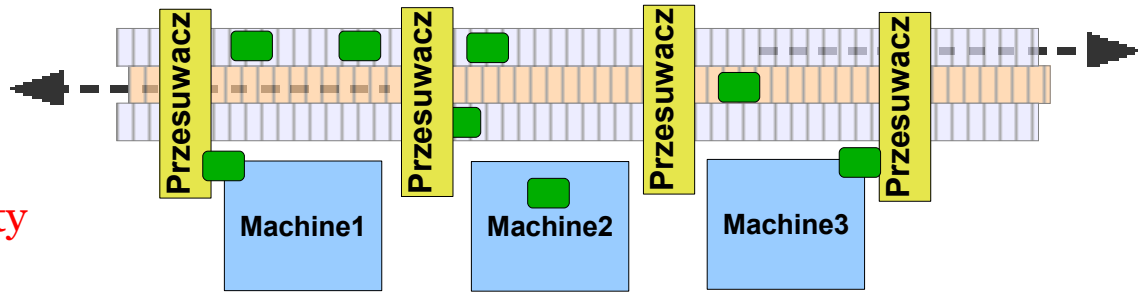
- ✔ **Air Traffic Manament System**,
  - tested at Sydney airport
- ✔ **Flexible Manufacturing System**
  - developed and used by *DaimlerChrysler*
- ✗ **Travel Support System**,
  -

Example 1: Flexible Manufacturing System

Traditional manufacturing line

Sytuacja

- ✔ różne maszyny realizują różne etapy produkcyjne
- ✔ przesuwacze dostarczają maszynom produkty z taśmy

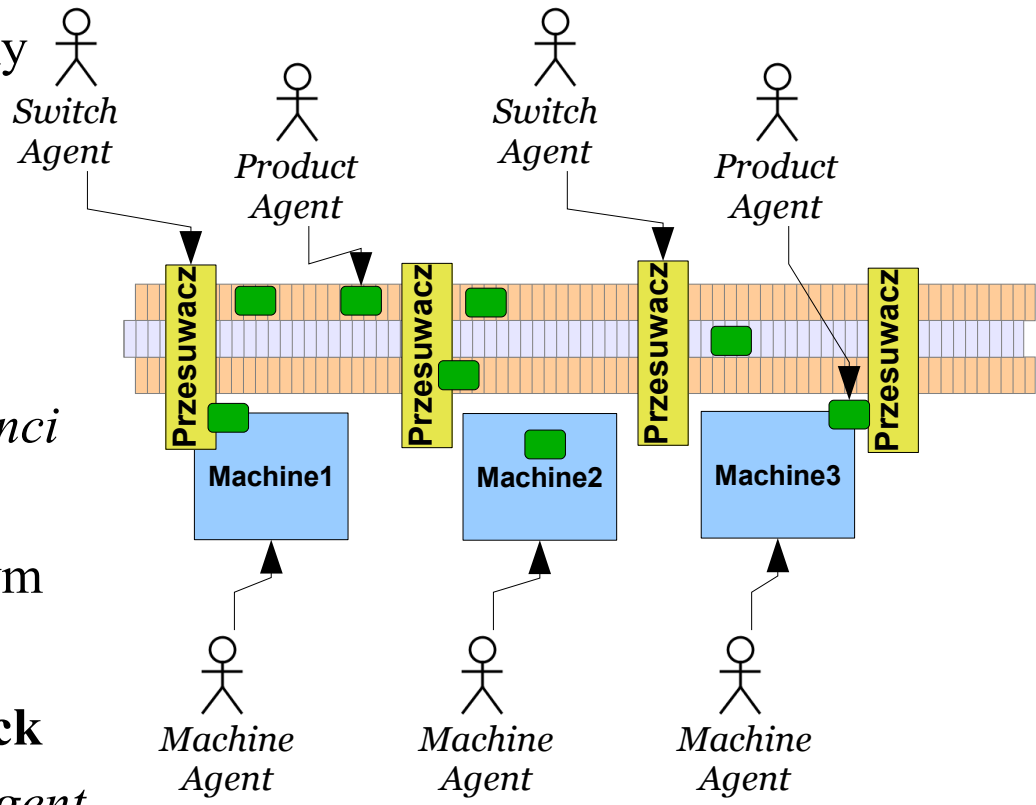


Problem

- ✗ **Mała zdolność do adaptacji**
  każda zmiana produktu to kosztowna rekonstrukcja linii produkcyjnej
- ✗ **Brak elastyczności**
  uszkodzenie pojedynczej maszyny to blokada całej linii produkcyjnej

Making the system flexible

- ✔ **Lokalni agenci**
  każdy element systemu kontroluje oddzielny agent
- ✔ **Koordynacja**
  agenci koordynuują działania między sobą
- ✔ **Negocjacje zmniejszają obciążenie**
  ProductAgent ogłasza aukcję, *MachineAgenci* składają propozycje – wygrywa maszyna o (przede wszystkim) najmniejszym aktualnym obciążeniu
- ✔ **Rozmowy bilateralne rozwiązują deadlock**
  rozmowy między *ProductAgent* a *SwitchAgent* pozwalają przemieszczać produkty w kierunku pożądanych maszyn z pomięciem zablokowanych maszyn



DaimlerChrysler: Agent-based Manufacturing Control System*

Zysk

- ✔ **Zwiększona wydajność** (10%)
- ✔ **Elastyczność:** szybka reakcja na lokalne problemy typu bottleneck

* S. Bussmann and K. Schild (2004). *An Agent-based Approach to the Control of Flexible Production Systems*

Questions ??

Example 2: "OASIS" Air Traffic Manament System

Aim of project

PLANNING PHASE

- Aim: ***Design an air traffic manament system*** which:
  - ✔ Calculates expected time of arrival (ETA) of aircraft
  - ✔ Sequences them in respect to optimality criteria
  - ✔ Issuing control directives to the pilots to achieve assigned ETAs
  - ✔ Monitor conformance

Domain characteristics

- **Environment** can evolve in **nondeterministic** way:
  - wind field ***can change***
  - operating conditions ***can change***
  - runway conditions ***can change***
  - presence of other aircraft ***can change***, etc.

- **System** can act in **nondeterministic** way:
  - system can take a ***number of different actions***:
    - ➔ Requesting an aircraft change speed
    - ➔ Strech / shorten / hold a flight path, etc.

Questions ??

Other examples

PLANNING PHASE

Ask **Paweł Olesiuk,**
Project **#8:** Modelling logistic company with holonic multi-agent system

Ask **Andrzej Borowczyk,**
Project **#9:** BDI agents in erderly people hospitalization

Contr-example: Travel Support System

Inspiration for system

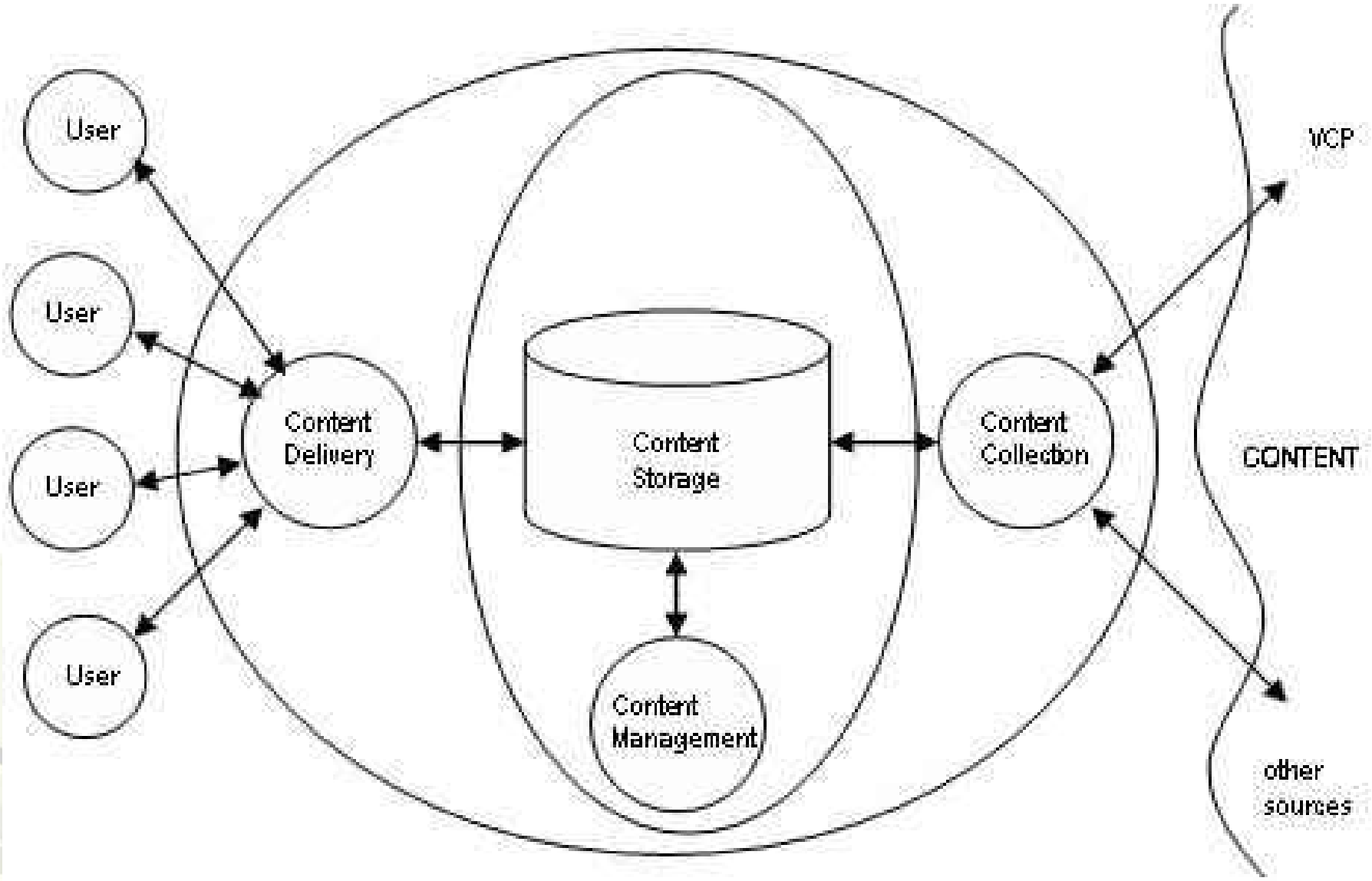
"Hungry foreign tourist arrives to an unknown city and seeks a nice restaurant serving cuisine that she likes. Internet, contacted for advice about restaurants in the neighborhood, recommends mainly establishments serving steaks, not knowing that the tourist is a fanatic vegetarian."

- Paradigmatic case of agent system design and implementation

See 1st Lecture for details.

Bird-flight perspective

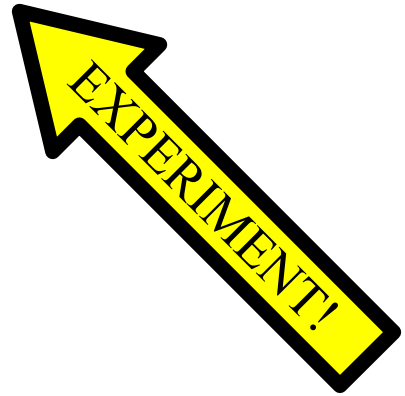
PLANNING PHASE



Pitfall of agentifying all

Prof. Marcin Paprzycki
Systems Research Institute
Polish Academy of Sciences
Warsaw, Poland

*"Decompose all functionalities as agents! Agentify all! If something is not an agent (e.g. expert system, database, etc.) it will be wrapped in an agent or interfaced via an agent."**



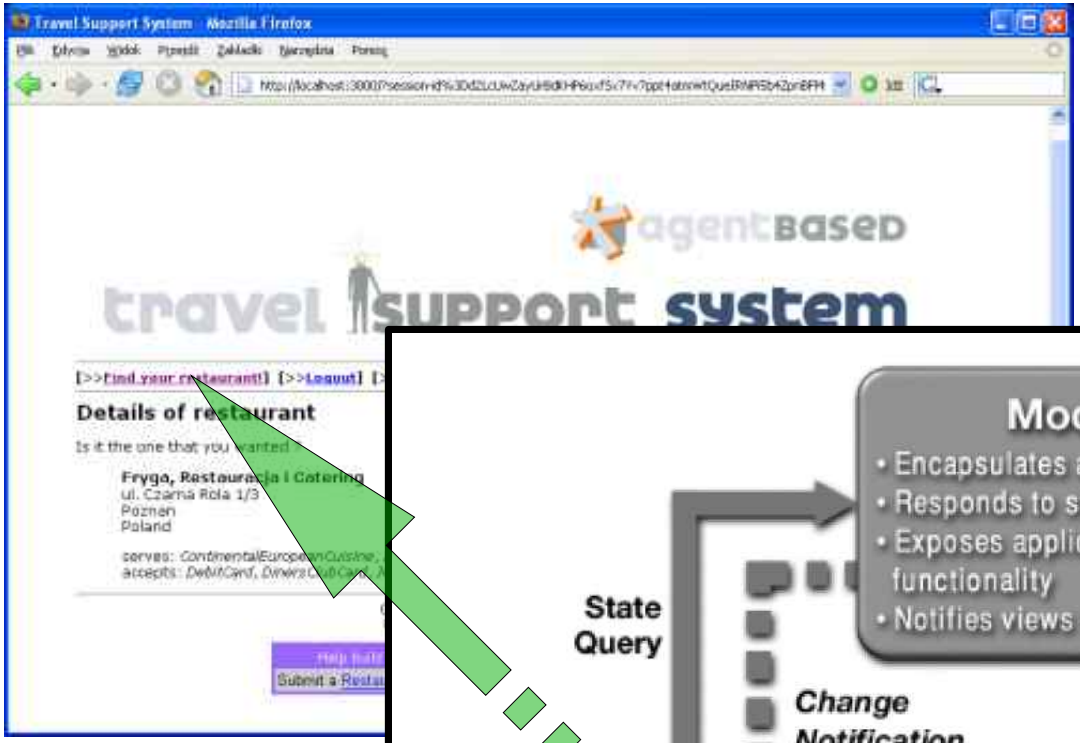
* Maciej Gawinecki, Mateusz Kruszyk, Marcin Paprzycki, Maria Ganzha (2007) *"Pitfall of agent system development o the basis of a Travel Support System". Proceedings of the BIS Conference* (to appear)

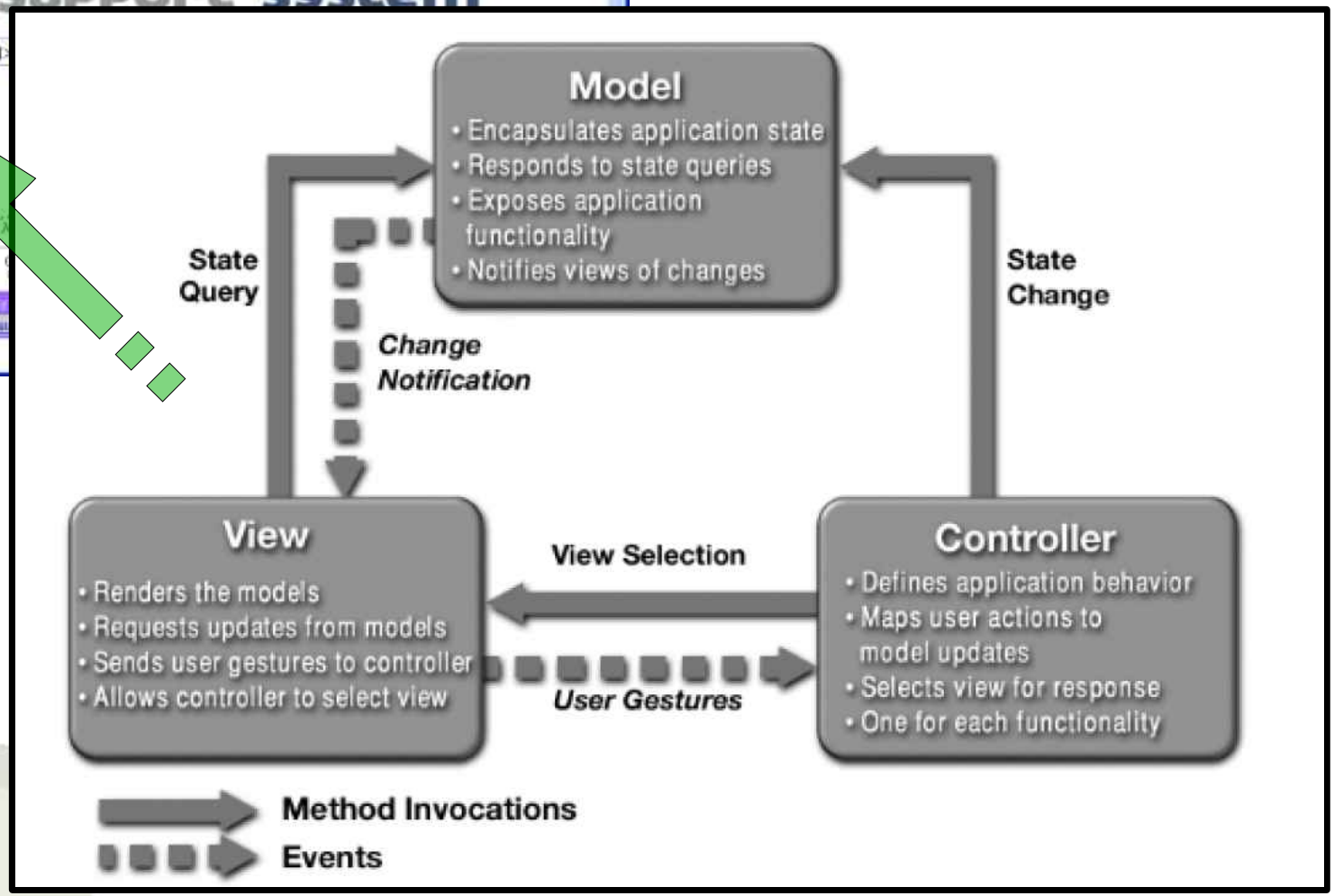
Prof. Michael Wooldridge
Department of Computer Science
University of Liverpool
Liverpool, UK

*"You see agents everywhere! It'ss a pitfall."**

* Wooldridge, M., Jennings, N.R.: "*Pitfalls of Agent-Oriented Development*". In Sycara, K.P., Wooldridge, M., eds.: *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, New York, ACM Press (1998) 385–391

MVC pattern – theory





MVC pattern – adaptation

SHA – *Session Handling Agent*, PrA – *Proxy Agent*,
VTA – *View Transforming Agent*, PA – *Personal Agent*

Approach	Model	View	Controller	HTTP server
Agent-based TSS	**SHA/PA** prepares models with pure data .	**VTA** wrapping Raccoon server generates view: HTML representation for model	**SHA** receives requests from PrA, forwards model to VTA, and returns view to PrA	**PrA** wraps home-made Java-based server
Traditional approach (Content Management System – CMS)	Data classes + DAO objects + data sources (e.g. database)	Templates processor, e.g. Velocity, Freemaker	J2EE application framework, e.g. Spring using container (Tomcat, Jetty, Resin)	Efficient HTTP server e.g. Apache

Ask **Edilbek Slanor,**
Project **#6:** *Agent as data provider in Content Management System*

Why not agents ?

- Characteristics of MVC + HTTP:
  - **stateless** – each user request is independent to others,
  - **reactive** – MVC components react only to external requests,
  - **synchronous** – process of realizing a single user request is a sequence of steps, where each next step cannot be realized until the previous one has been finished,
  - **parallel, but not concurrent** – parallelism is utilized to decrease interleaving in I/O operations.

- But... agents are:
  - **proactive**,
  - **statefull**,
  - **concurrent**,
  - use **asynchronous** communication.

General guideline

Use simpler traditional approaches if possible!

- *Modelling a part of the system with higher abstraction than naturally necessary, results in difficulties of verifying and reasoning about such solution (i.e. the simpler the model the easier it is to think about it, to verify its correctness and to remove errors).**

* Maciej Gawinecki, Mateusz Kruszyk, Marcin Paprzycki, Maria Ganzha (2007) *"Pitfall of agent system development o the basis of a Travel Support System". Proceedings of the BIS Conference* (to appear)

Questions ??

Example for modelling E-Commerce Agent Platform

E-CAP team

PLANNING PHASE

IBS

Pracownia Systemów Inteligętnych

dr Marcin PAPRZYCKI

PAN

Ośrodek Komputerowy

dr Maria GANZHA
mgr Maciej GAWINECKI
mgr Paweł KOBZDEJ

Costin Badica

Analysis Phase

Analysis aim and steps

- Aim
  - Clarify the problem
  - Avoid concerning about the solution

- Steps
  - Defining system scenarios and use cases
  - Identifying roles and their responsibilies
  - Identifying roles acquaintance
  - Matching agents with roles
  - Deploying agents over platforms and hosts

Main scenario in E-CAP

1. The Client chooses a Shop to negotiate at.

2. The Shop registers the Client for negotiation.

3. Negotiations.

4. The Shop reserves a product for the Client.

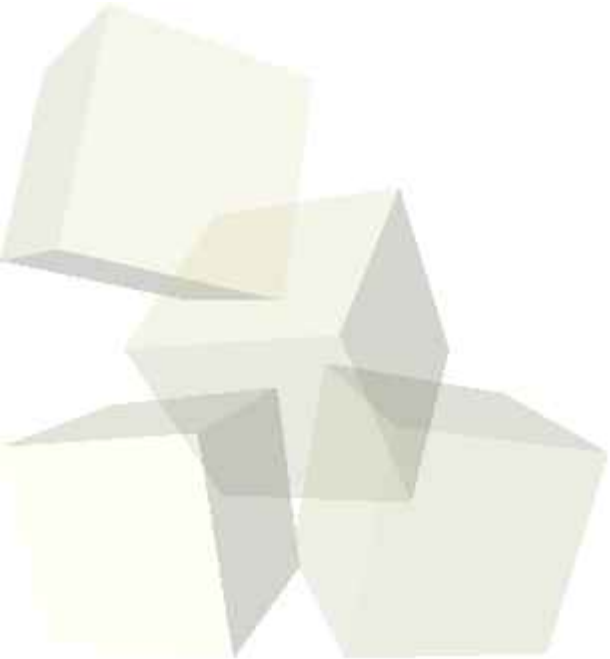
5. The Client confirmes/cancels the reservation.

6. Sale finilization:

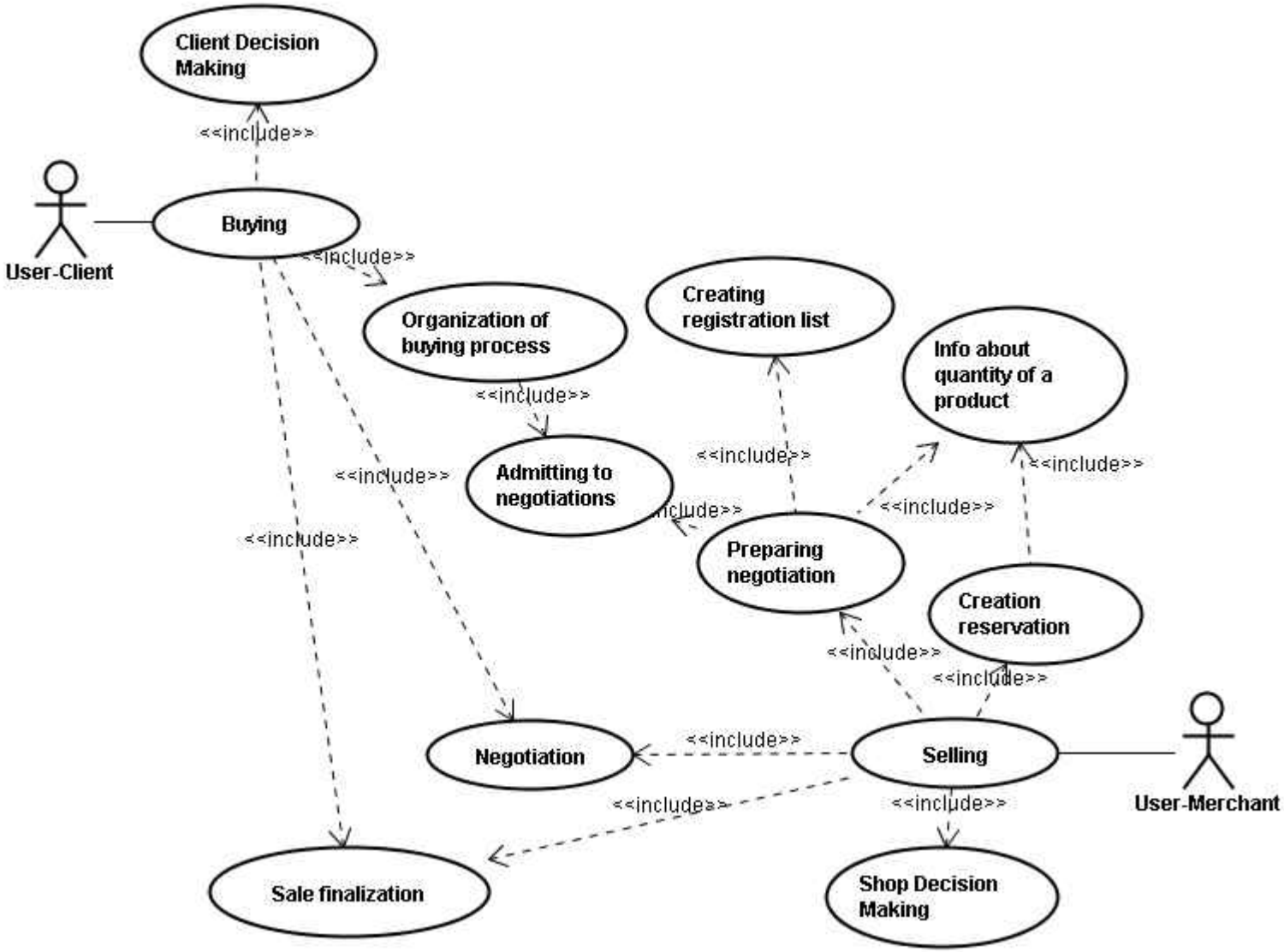
(a) the Client pays for the product to the Shop,

(b) the Shop delivers the product to the Client.

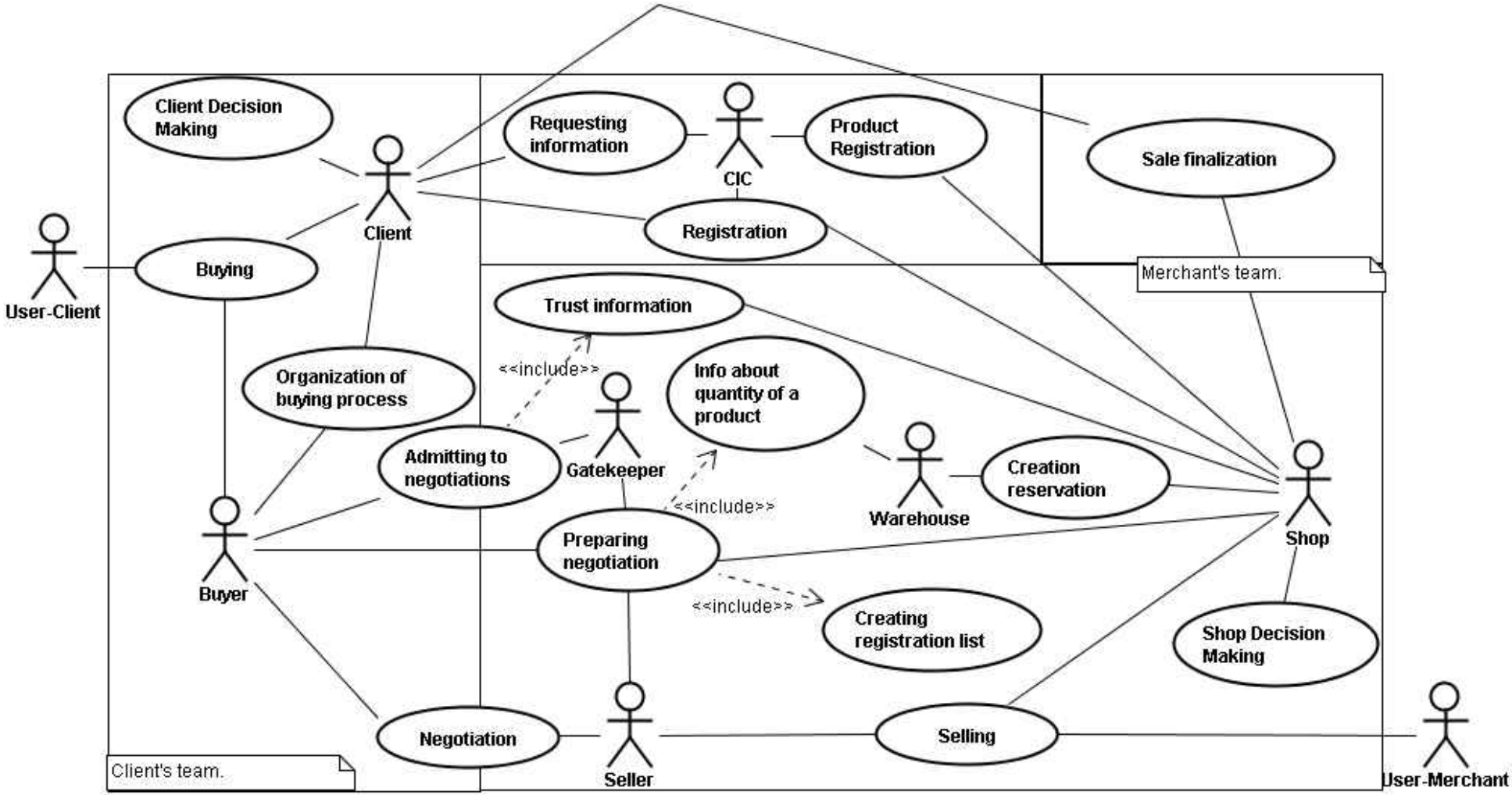
Identifying roles and responsibilies



E-CAP general use-case diagram



E-CAP detailed use-case diagram

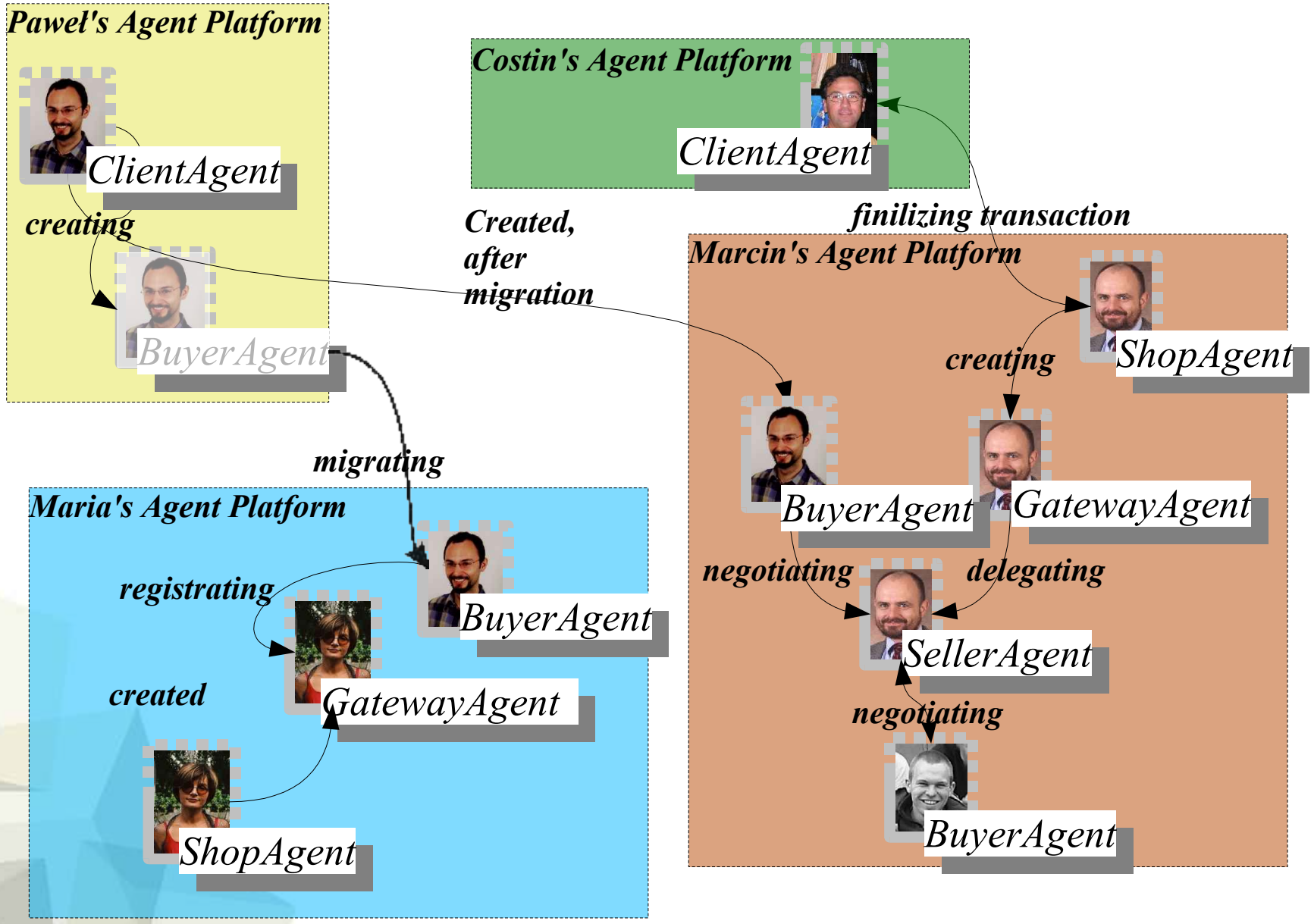


Deploying agents over platforms

- Physical hosts/platforms are indicated for particular agents

- Factors to consider:
  - agents belonging to competitive or politically conflicted owners go to different platforms (security reasons)
  - communication efficiency
  - inter-platform communication
  - inter-platform mobility

Deployment diagram



E-Commerce Agent Platform*

Design Phase

Analysis aim and steps

- Aim
  - Clarify the problem
  - Avoid concerning the about solution

- Steps
  - Defining system scenarios and use cases
  - Identifying roles and their responsibilies
  - Identifying roles acquaintance
  - Matching agents with roles
  - Deploying agents over platforms and hosts

Implementation Phase

Analysis aim and steps

- Aim
  - Clarify the problem
  - Avoid concerning the about solution

- Steps
  - Defining system scenarios and use cases
  - Identifying roles and their responsibilies
  - Identifying roles acquaintance
  - Matching agents with roles
  - Deploying agents over platforms and hosts

Perspectives in MAS

- External (inter-agent) perspective
  - Cooperation and competitiveness
  - Communication
  - Interaction protocols
  - Communication ontologies
- Internal (intra-agent) perspective
  - Behaviours
  - Knowledge
  - Handling with messages

Translating Interaction Protocols into FSMBehaviour

Interaction protocols

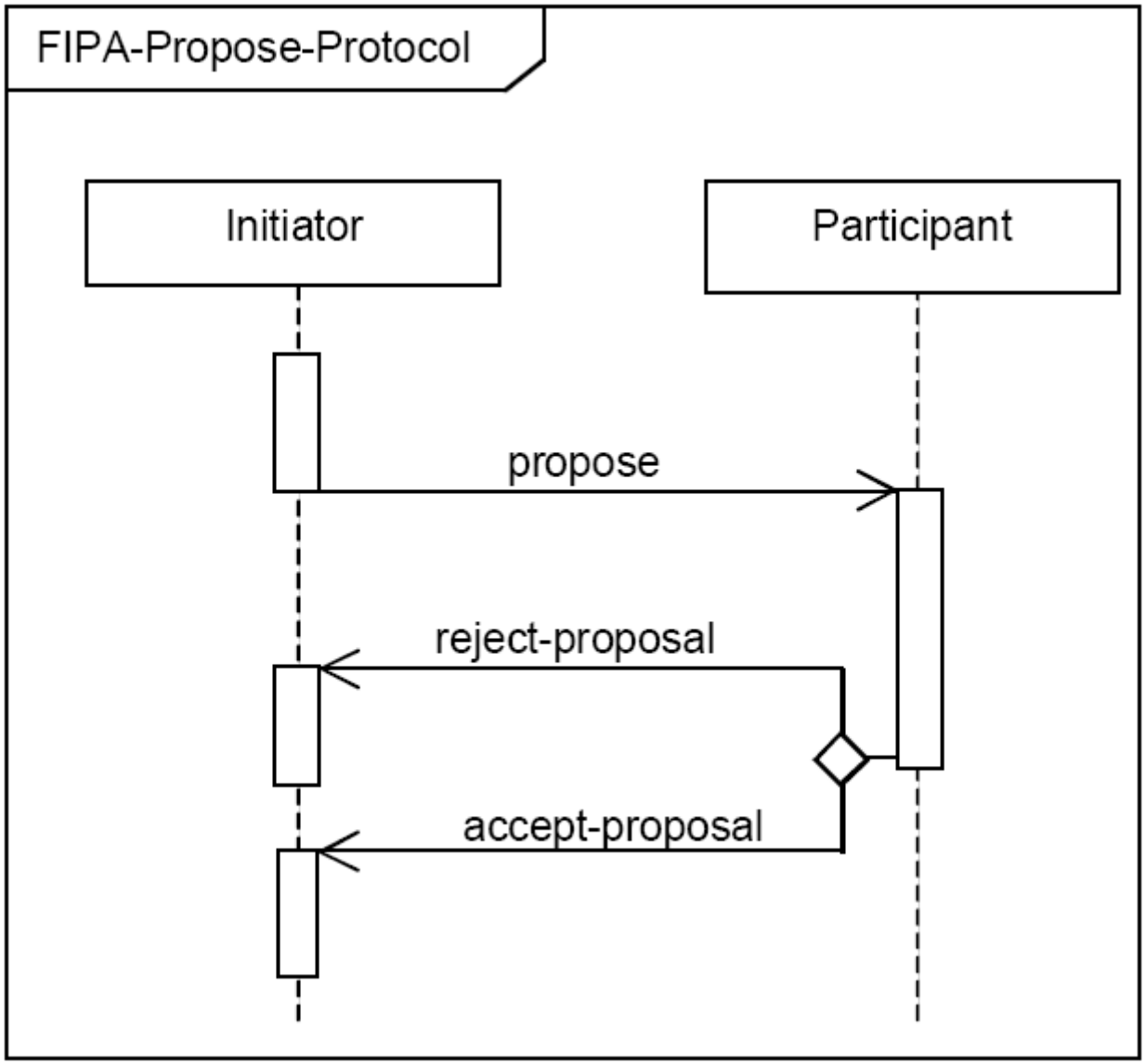
- Having a standard set of types of messages (INFORM, REQUEST, PROPOSE ....) allows specifying **predefined sequences of messages** exchanged by agents during a conversations.

- These are known as **Interaction Protocols**

FIPA Propose Interaction Protocol

- ***Description:***
  Initiator propose to do an action, if *Participant* will accept the proposal.

- ***Exception:***
  Participant can inform *Initiator* that it did not understand what was communicated.

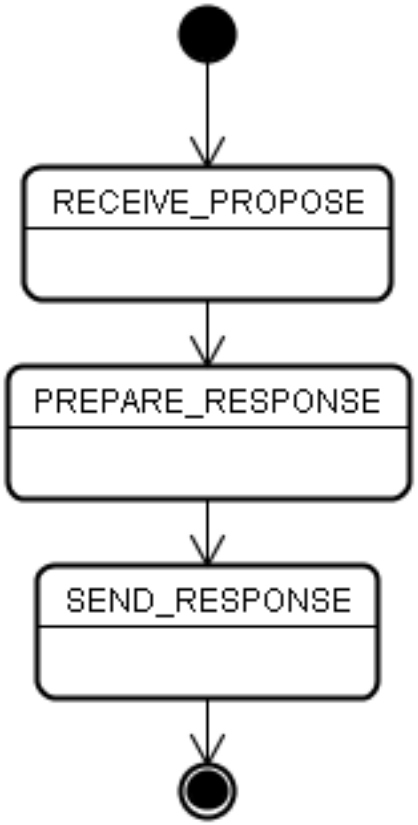


FIPA Propose Interaction Protocol Specification, http://www.fipa.org/specs/fipa00036/

Support for interaction protocols

- The **`jade.proto`** package contains behaviours for both the *Initiator* and *Responder* role in the most common interaction protocols:
  - *FIPA-Request* (**`AchieveREInitiator/Responder`**)
  - *FIPA-Contract-Net* (**`ContractNetInitiator/Responder`**)
  - *FIPA-Subscribe* (**`SubscriptionInitiator/Responder`**)
  - *FIPA-Propose* (**`ProposeInitiator/Responder`**)
- All these classes automatically handle
  - the **flow of messages** checking that it is compliant to the protocol
  - The **timeouts** (if any)
- They provide callback methods that should be redefined to take the necessary actions when e.g. a message is received or a timeout expires.

Propose Participant states

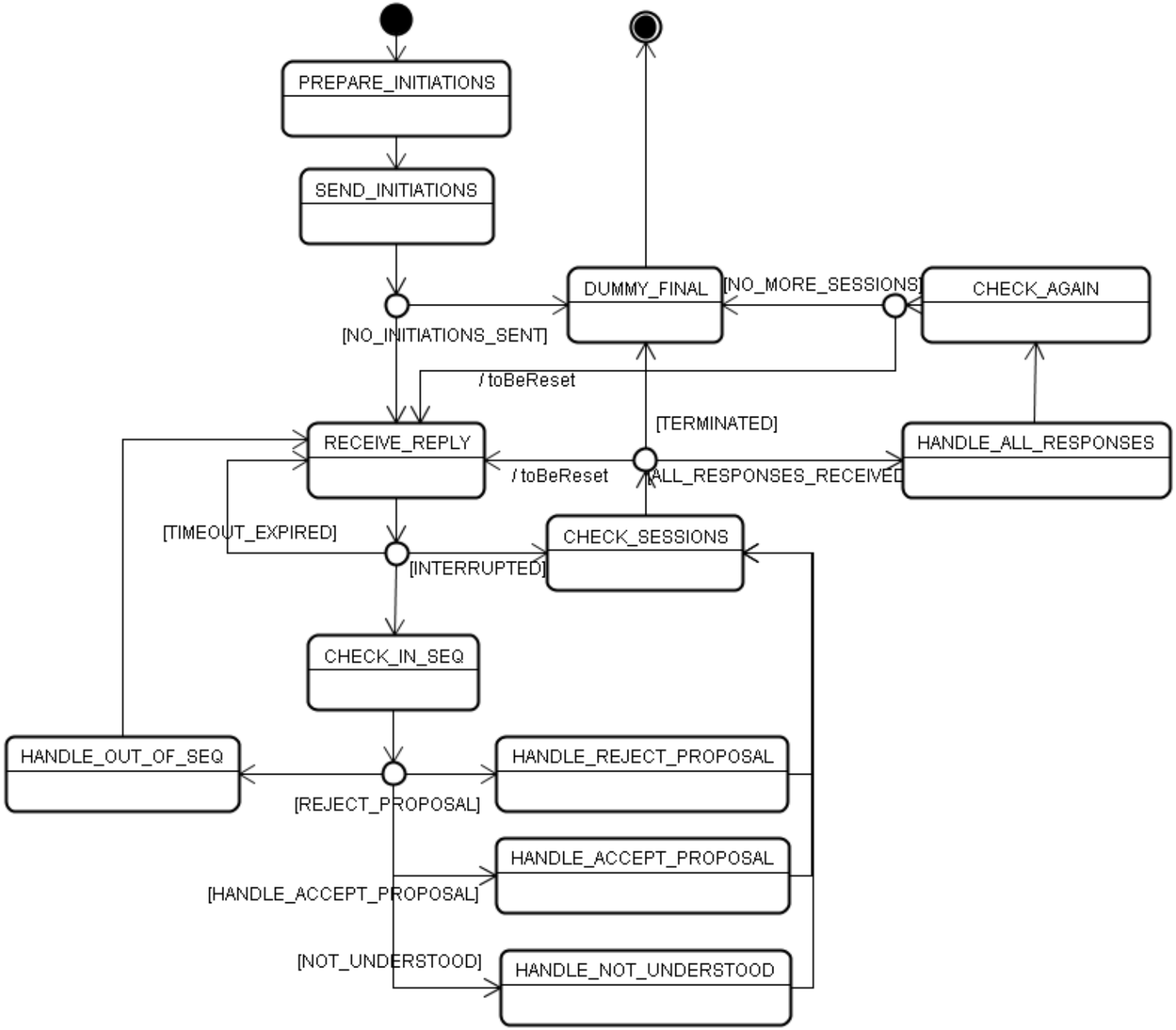


Propose Participant implementation

- *Code:* jade.proto.ProposeResponder

```
public class ProposeResponder extends FSMBehaviour implements
    FIPANames.InteractionProtocol {
  ...
  // Register the FSM transitions
  registerDefaultTransition(RECEIVE_PROPOSE, PREPARE_RESPONSE);
  registerDefaultTransition(PREPARE_RESPONSE, SEND_RESPONSE);
  registerDefaultTransition(SEND_RESPONSE, RECEIVE_PROPOSE);

  // Create and register the states that make up the FSM
  Behaviour b = null;

  // RECEIVE_PROPOSE
  rec = new MsgReceiver(myAgent, mt, -1, getDataStore(), PROPOSE_KEY);
  registerFirstState(rec, RECEIVE_PROPOSE);

  // PREPARE_RESPONSE
  b = new PrepareResponse(myAgent);
  b.setDataStore(getDataStore());
  registerState(b, PREPARE_RESPONSE);

  // SEND_RESPONSE
  b = new ReplySender(myAgent, RESPONSE_KEY, PROPOSE_KEY);
  b.setDataStore(getDataStore());
  registerState(b, SEND_RESPONSE);
```

Propose Initiator states



Propose Initiator implementation

- *Code:* jade.proto.ProposeIntiator

```
public class ProposeInitiator extends FSMBehaviour {
  ...
  // Register the FSM transitions
  registerDefaultTransition(PREPARE_INITIATIONS, SEND_INITIATIONS);
  registerTransition(SEND_INITIATIONS, DUMMY_FINAL, 0);
  registerDefaultTransition(SEND_INITIATIONS, RECEIVE_REPLY);
  registerTransition(RECEIVE_REPLY, CHECK_SESSIONS,
    MsgReceiver.TIMEOUT_EXPIRED);
  registerTransition(RECEIVE_REPLY, CHECK_SESSIONS,
    MsgReceiver.INTERRUPTED);
  registerDefaultTransition(RECEIVE_REPLY, CHECK_IN_SEQ);
  registerTransition(CHECK_IN_SEQ, HANDLE_NOT_UNDERSTOOD,
    ACLMessage.NOT_UNDERSTOOD);
  registerTransition(CHECK_IN_SEQ, HANDLE_REJECT_PROPOSAL,
    ACLMessage.REJECT_PROPOSAL);
  registerTransition(CHECK_IN_SEQ, HANDLE_ACCEPT_PROPOSAL,
    ACLMessage.ACCEPT_PROPOSAL);
  registerDefaultTransition(CHECK_IN_SEQ, HANDLE_OUT_OF_SEQ);
  registerDefaultTransition(HANDLE_NOT_UNDERSTOOD, CHECK_SESSIONS);
  registerDefaultTransition(HANDLE_REJECT_PROPOSAL, CHECK_SESSIONS);
  registerDefaultTransition(HANDLE_ACCEPT_PROPOSAL, CHECK_SESSIONS);
  registerDefaultTransition(HANDLE_OUT_OF_SEQ, RECEIVE_REPLY);
  registerDefaultTransition(CHECK_SESSIONS, RECEIVE_REPLY, toBeReset);
  registerTransition(CHECK_SESSIONS, HANDLE_ALL_RESPONSES,
    ALL_RESPONSES_RECEIVED);
  registerTransition(CHECK_SESSIONS, DUMMY_FINAL, TERMINATED);
  registerDefaultTransition(HANDLE_ALL_RESPONSES, CHECK_AGAIN);
  registerTransition(CHECK_AGAIN, DUMMY_FINAL, 0);
  registerDefaultTransition(CHECK_AGAIN, RECEIVE_REPLY, toBeReset);
  ...
  registerFirstState(b, PREPARE_INITIATIONS);
   ...
  registerLastState(b, DUMMY_FINAL);
```

Documentation

- Chapter 3.5 in the *Programmers guide* included in the JADE distribution provides a detailed explanation of the interaction protocol support
- API documentation (*javadoc*): **`jade.proto`** package
- Sample code: **`examples.protocols`** package in the examples included in the JADE distribution.

Questions ??

Tools for testing phase

Debugging communication

- **DummyAgent**
  - interacting with JADE agents
  - sending ACL messages
  - maintains a list of ACL messages sent and received

See tutorial from 1st Laboratory for details.

- **Introspector**
  - monitoring and controlling the *life-cycle* of agent
  - monitoring agent's exchanged messages
  - monitoring the queue of behaviours (*step-by-step* execution)

- **SnifferAgent**
  - tracking and displaying messages from/to sniffed an agents
  - saving tracked messages

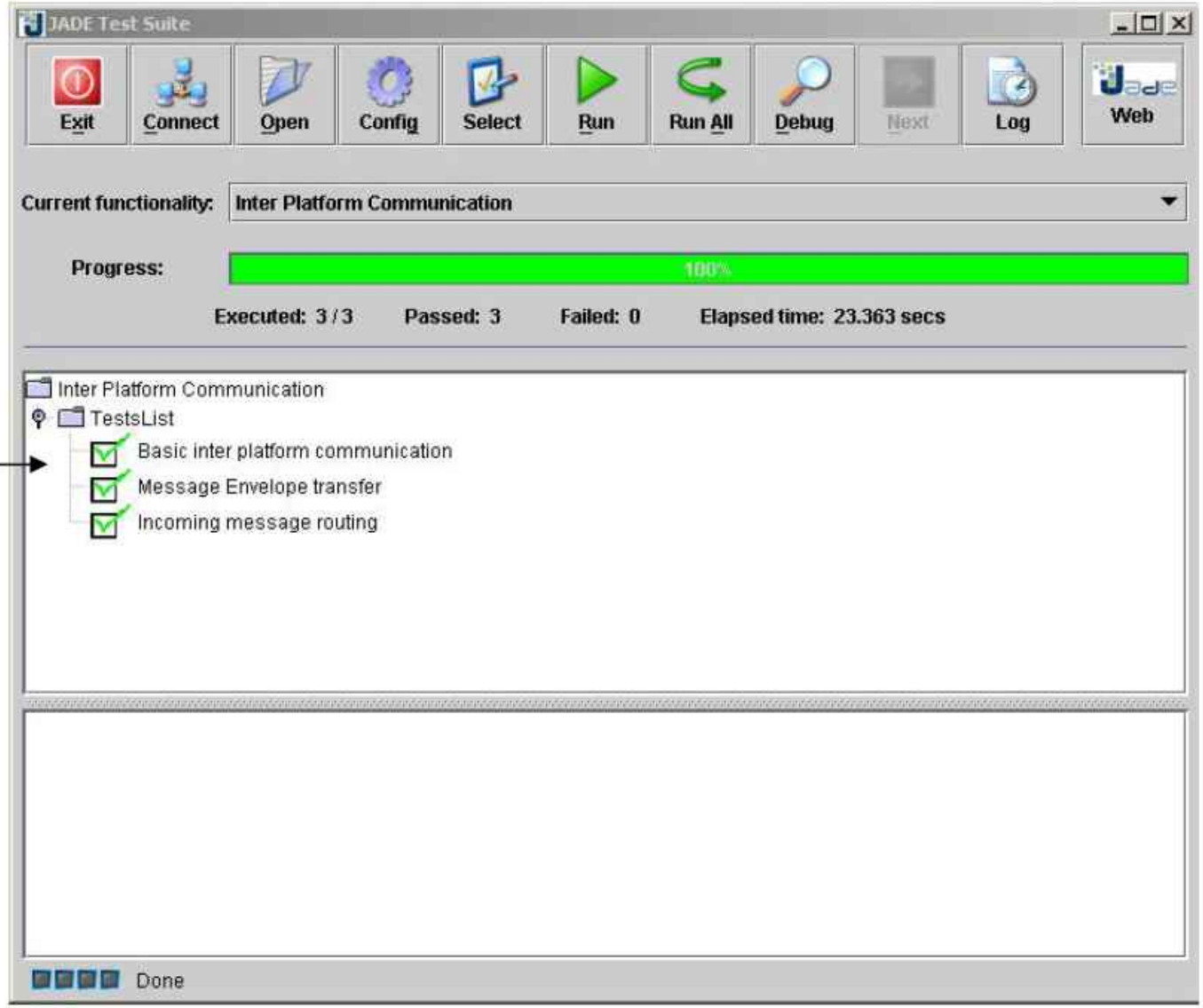
Logging

- Type of error information and its narrow context are usually sufficient to find the reason of the error and eliminate it in the source code.
- Traditional solutions for Java:
  - **Java Logging API** (JSR47), comes with the JRE, http://java.sun.com/
  - **log4j**, http://logging.apache.org/log4j/
- Agent-oriented solutions for JADE:
  - **JADE Logging service**, http://jade.tilab.com
    - → JSR47-based
  - **LoggerAgent environment**, http://jadex.sourcefoge.net/
    - → Part of JADEX project,
    - → JSR47-based
  - **Log4JADE** *(experimental)*, http://log4jade.sourceforge.net/
    - → log4j-based
- Comparision of different approaches can be found in
  - Maciej Gawinecki, *"Agent-based logging system,"* in: *Proceedings of the 18th Mountain Summer School of Polish Information Processing Society*. Szczyrk, Poland. 2006.

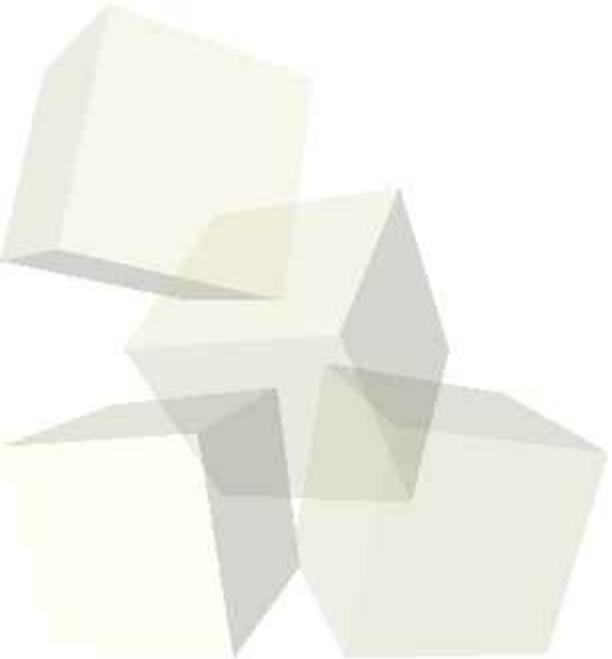
Testing suites

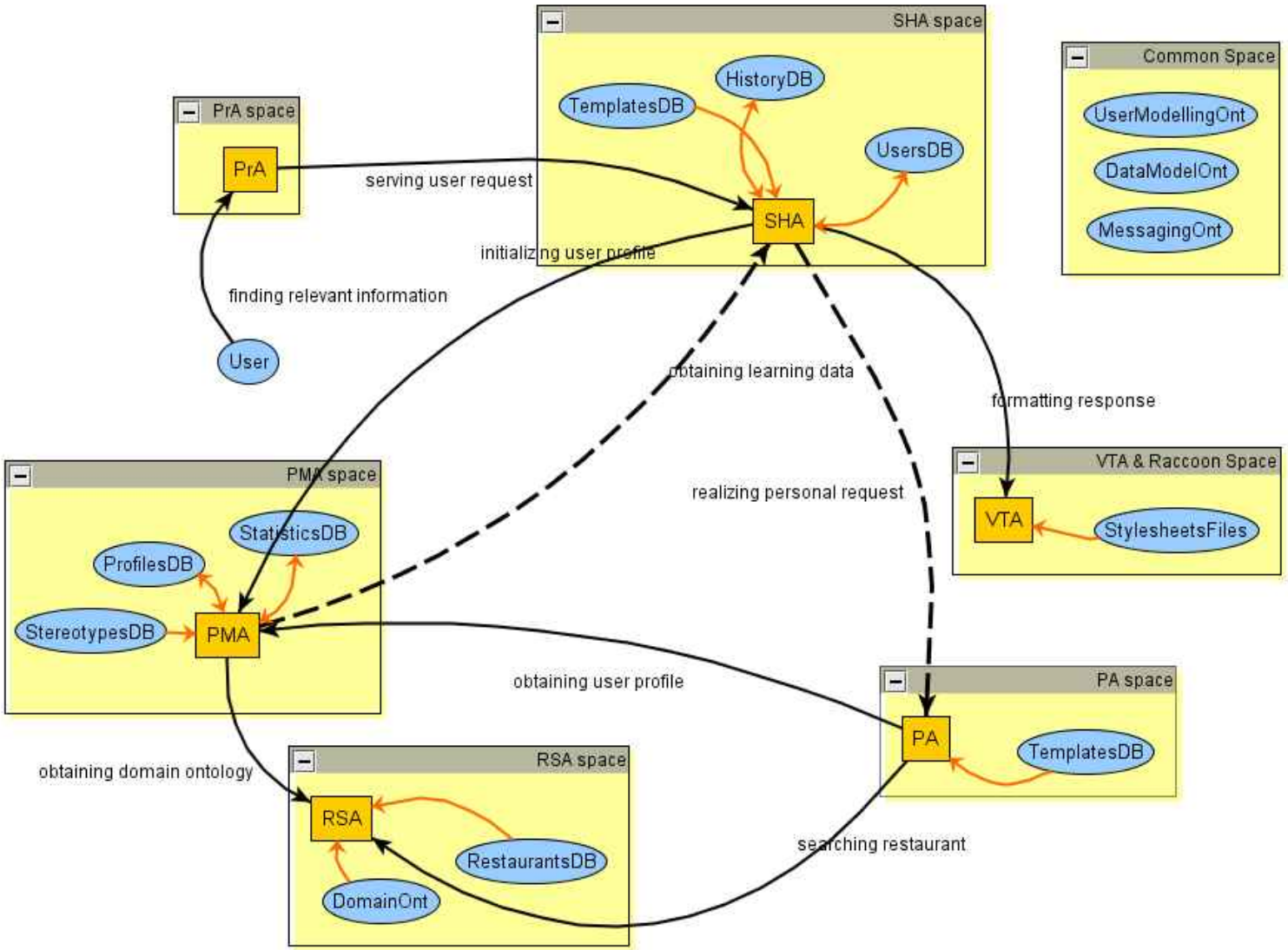
- Traditional solution

Tests in execution



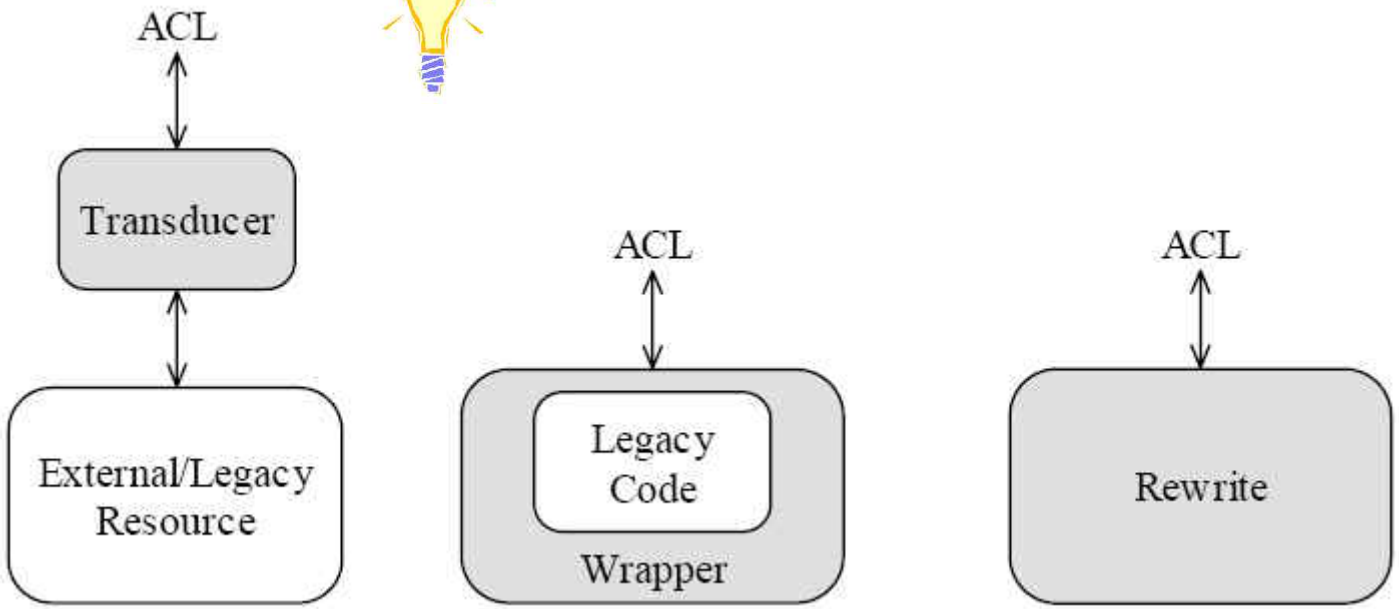
Questions ??

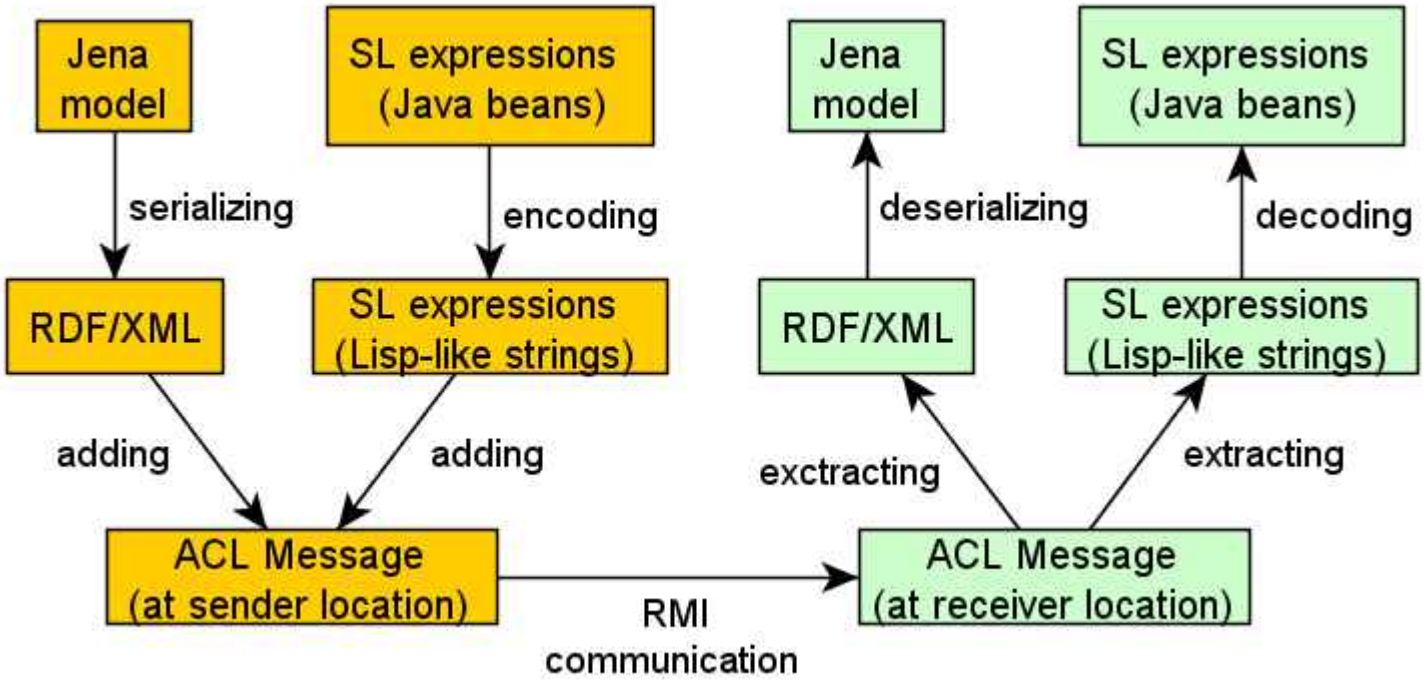



SHA space
HistoryDB
TemplatesDB
UsersDB
SHA
Common Space
UserModellingOnt
DataModelOnt
MessagingOnt
PrA space
PrA
serving user request
initializing user profile
finding relevant information
User
obtaining learning data
formatting response
VTA & Raccoon Space
VTA
StylesheetsFiles
realizing personal request
PMA space
StatisticsDB
ProfilesDB
StereotypesDB
PMA
obtaining user profile
PA space
PA
TemplatesDB
obtaining domain ontology
RSA space
RSA
RestaurantsDB
DomainOnt
searching restaurant

Agents and legacy software

Ask **Michał Oglodek,**
Project **#5**: *Utilization of an agent and Web Service as wrappers for existing legacy software*




Jena model
SL expressions (Java beans)
Jena model
SL expressions (Java beans)
serializing
encoding
deserializing
decoding
RDF/XML
SL expressions (Lisp-like strings)
RDF/XML
SL expressions (Lisp-like strings)
adding
adding
exctracting
extracting
ACL Message (at sender location)
RMI communication
ACL Message (at receiver location)

Homework

- Re-implement presented application by use of any of complex behaviours (**`SequentialBehaviour`**, **`FSMBehaviour`**)
  - Documentation
    - **`examples.behaviours.ComplexBehaviourAgent`**, **`examples.behaviours.FSMAgent`** classes in JADE package
    - *JADE Programmer's Guide, http://jade.tilab.com*

- Think about agents modelling some phenomen from real world

Questions ??

Acknowledgements

- Mateusz Kruszyk for veryfing part of tutorial