Technologies for Semantic Interoperability in SOA Systems: Agent Technologies

Dr. Klaus Fischer

Multiagent Systems Group
DFKI GmbH
Saarbrücken, Germany

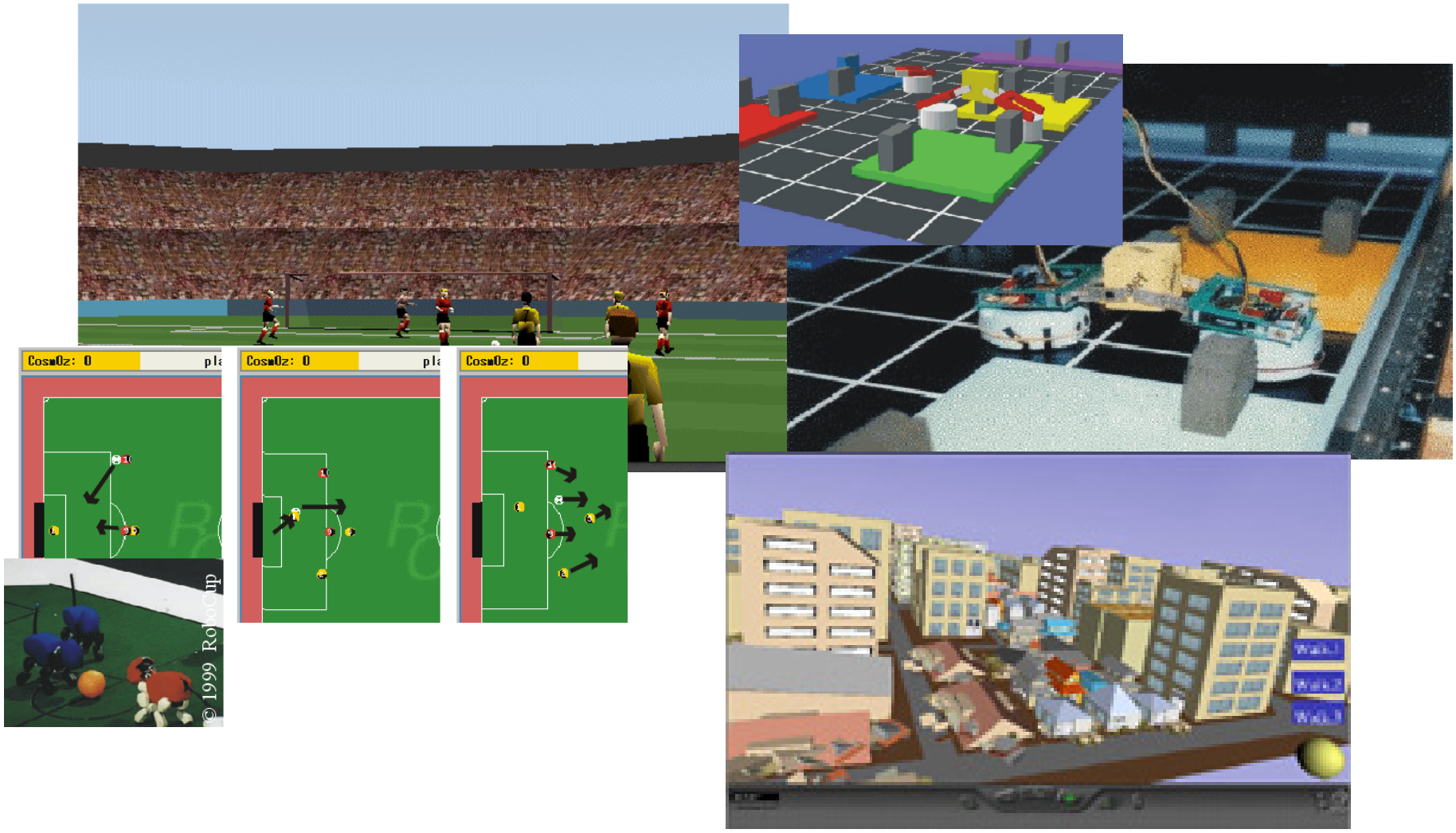
Overview

- Context at DFKI
- An MDA Approach to Agent Design
  - Origins in the ATHENA Project
  - Platform Independent Models for SOA
  - A Platform-Independent Metamodel for Agents and Multiagent Systems
- Agent Interaction
- Model Transformation
- Semantic Services
- Agents in Virtual Worlds
- Conclusion

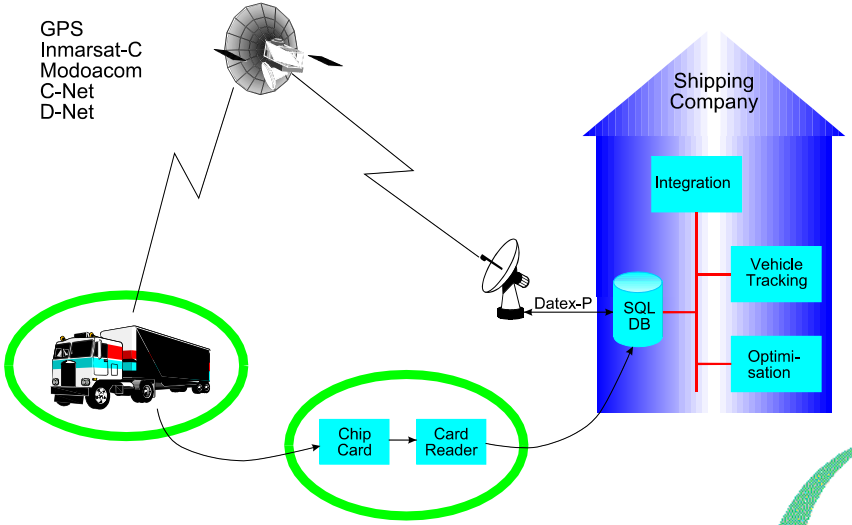
DFKI is Situated at Three Sites

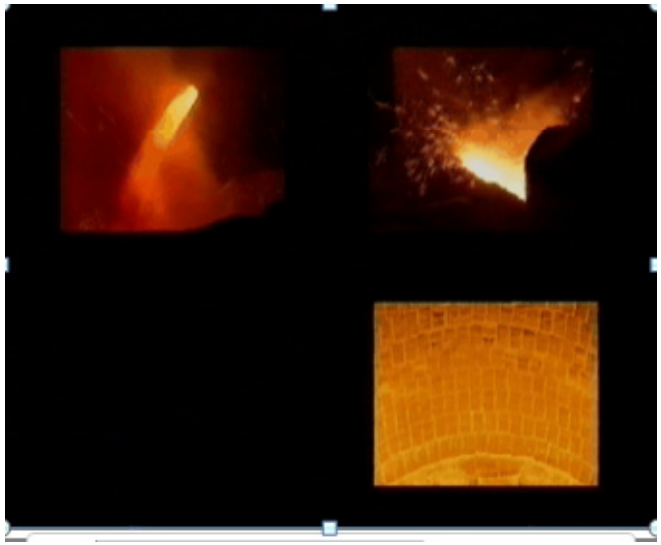
Bremen

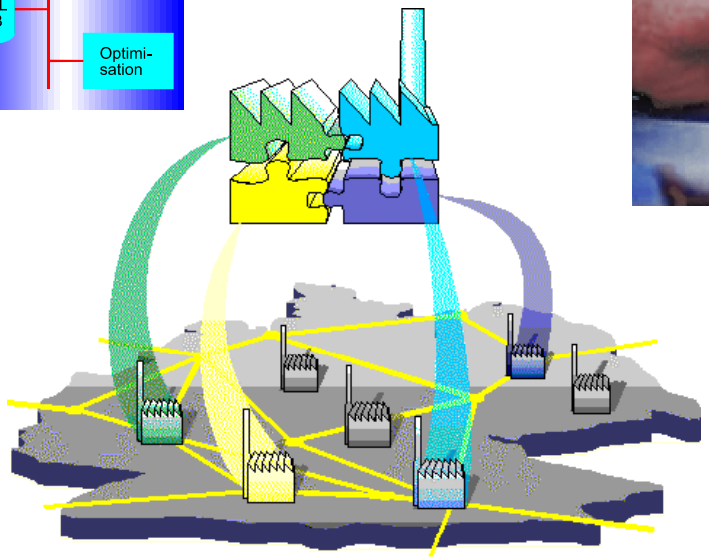
Autonomous Agents in Physical and Virtual Worlds



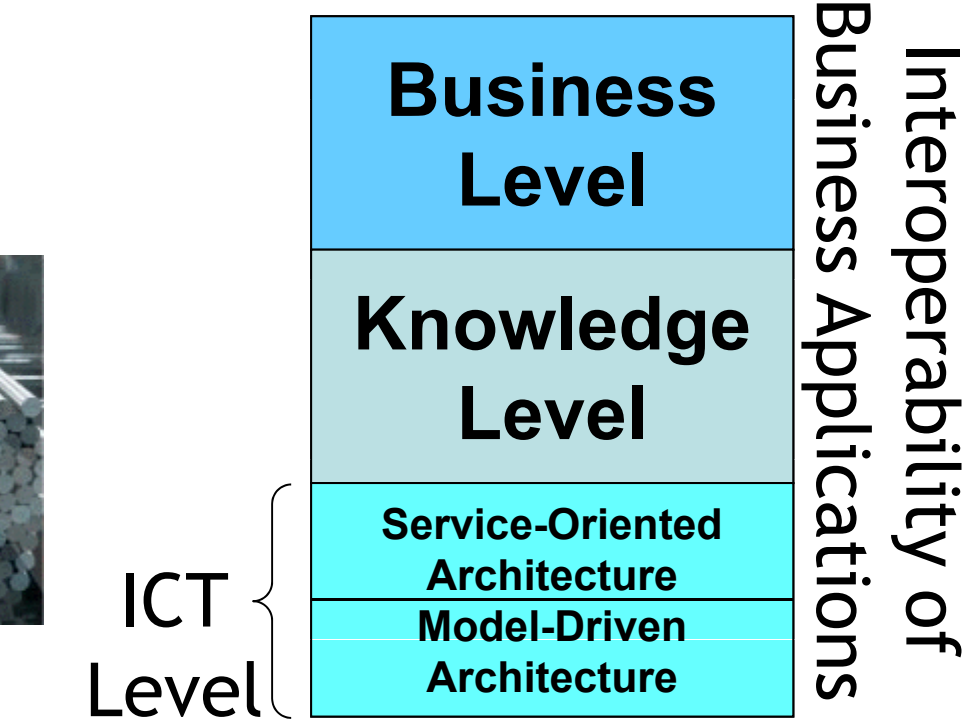
Virtual Enterprises, e-Business and Supply Chain Management



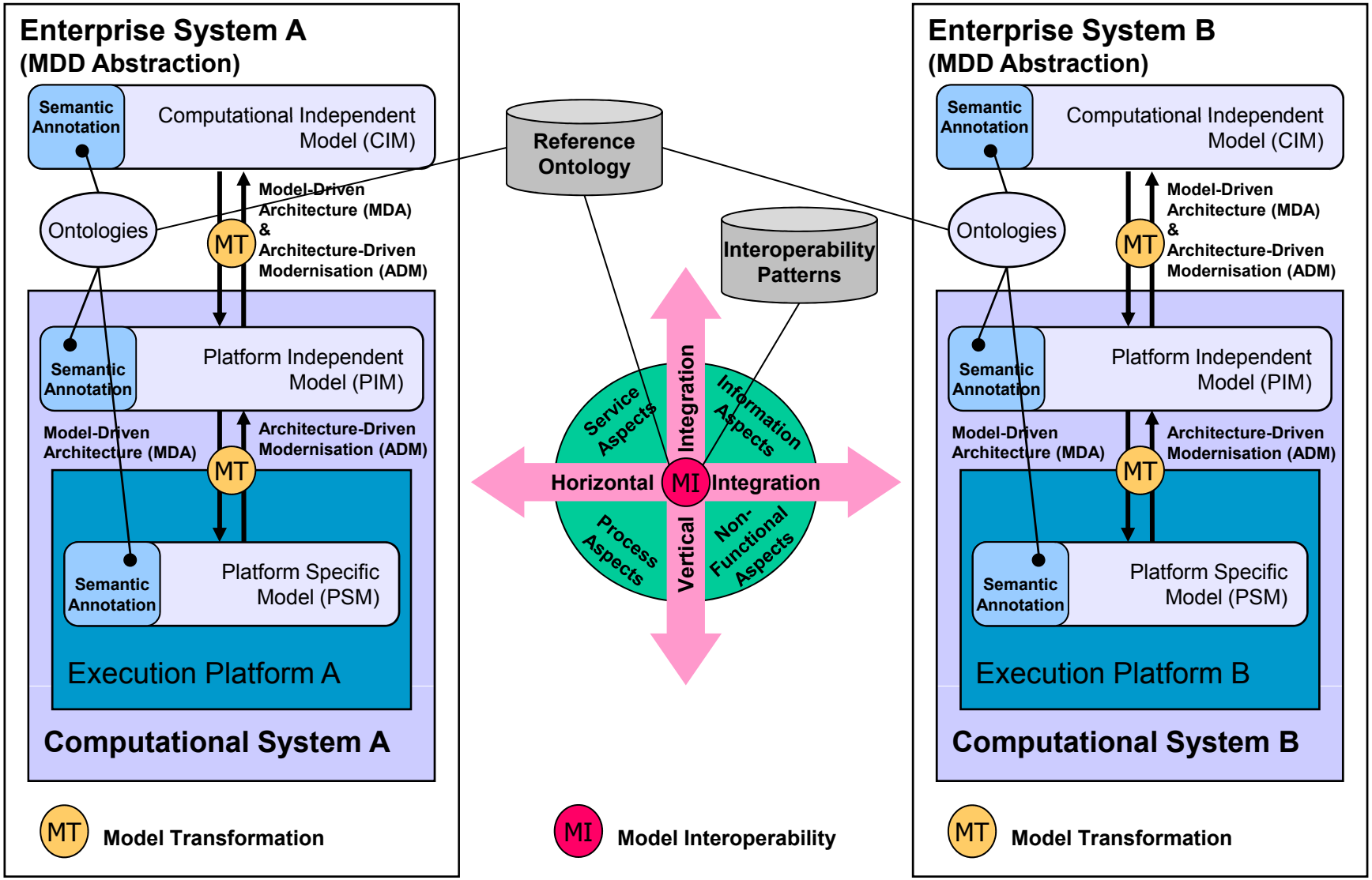




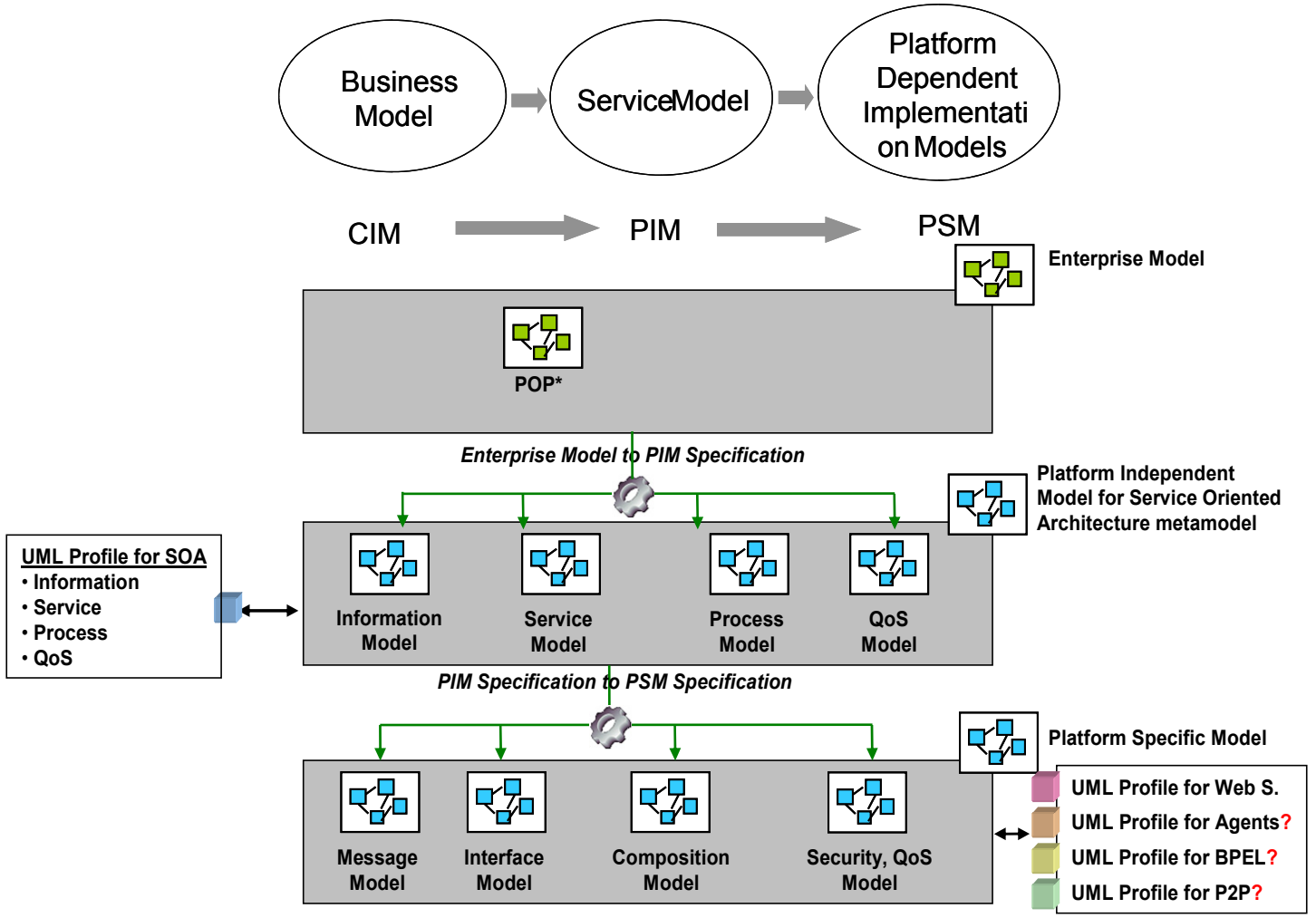
ATHENA: Structure & Domains



ATHENA Reference Model



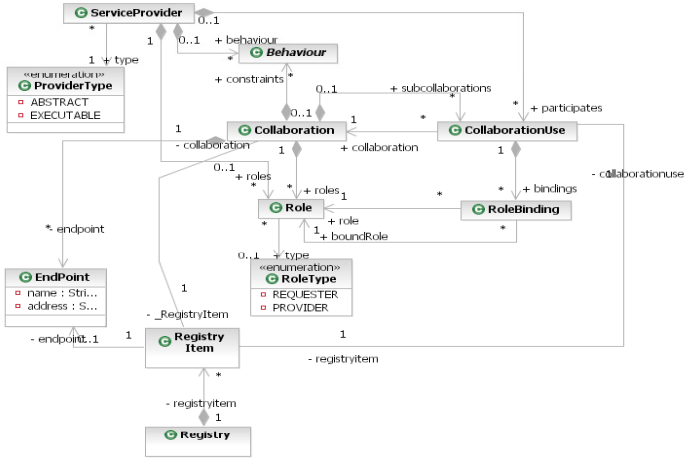
An MDA Aproach to Service-Oriented Architectures (SOA)



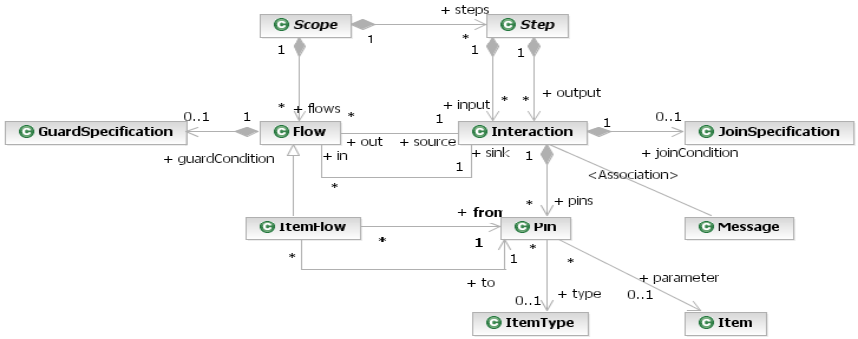
PIM4SOA: 4 Views to System Design

ATHENA
European Integrated Project

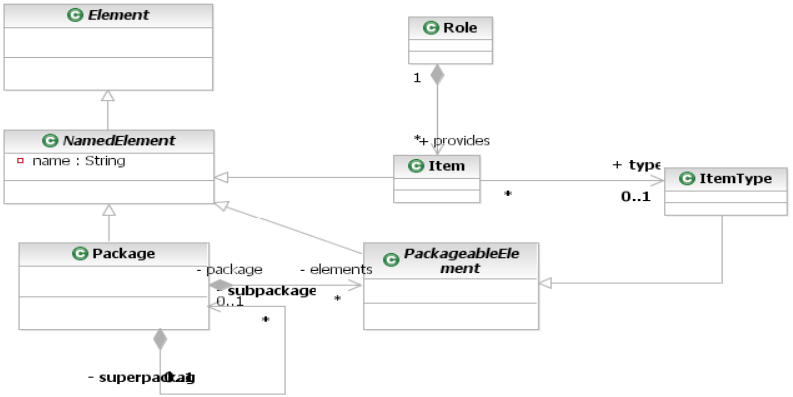
Metamodel for (software) services



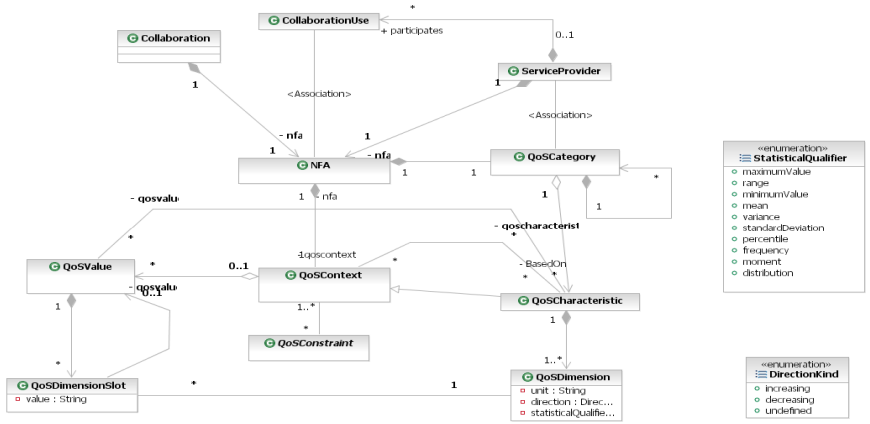
Metamodel for (automated software) processes



Metamodel for information



Metamodel for quality of service (QoS)

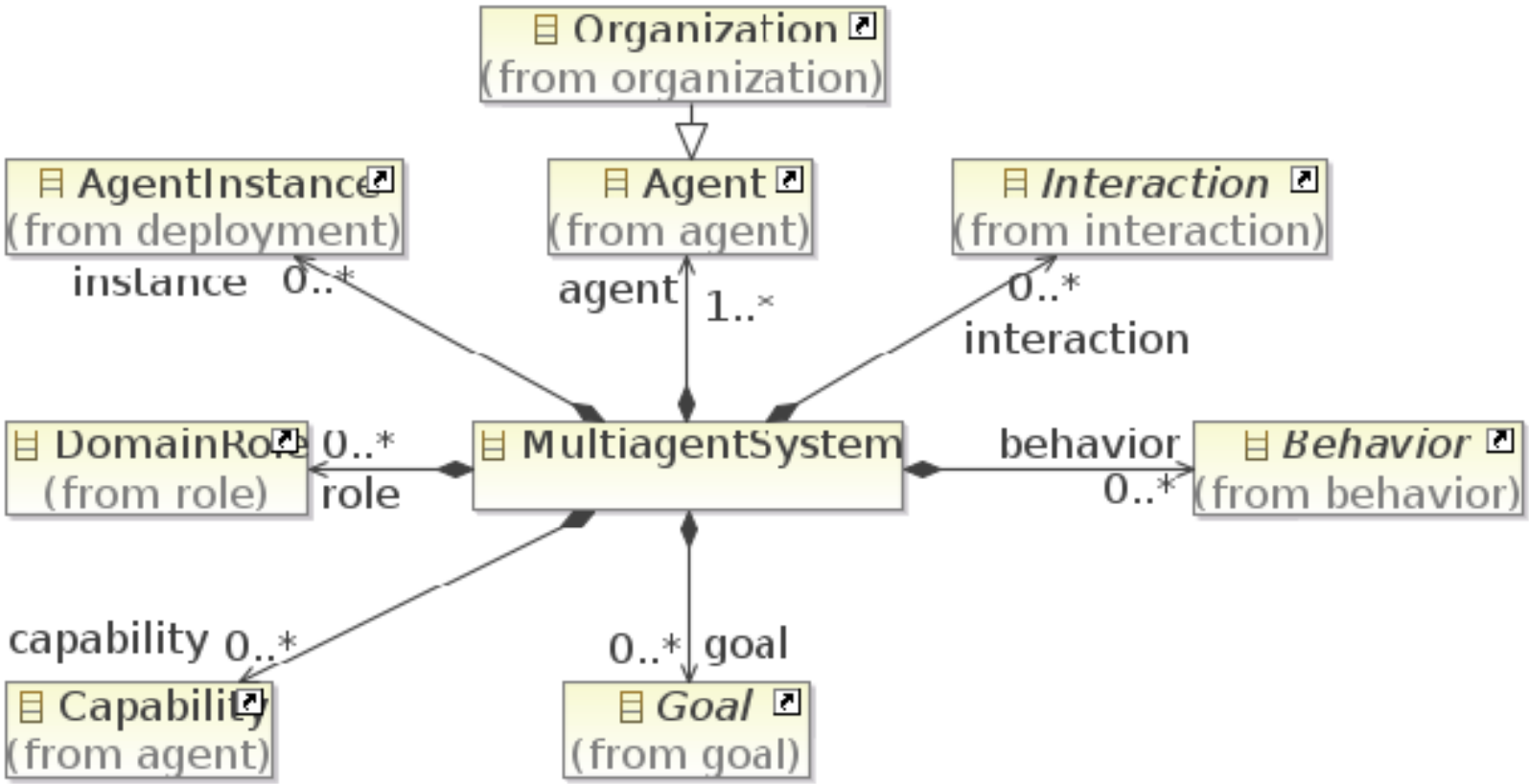


Platform Independent Model for Agents (PIM4Agents)

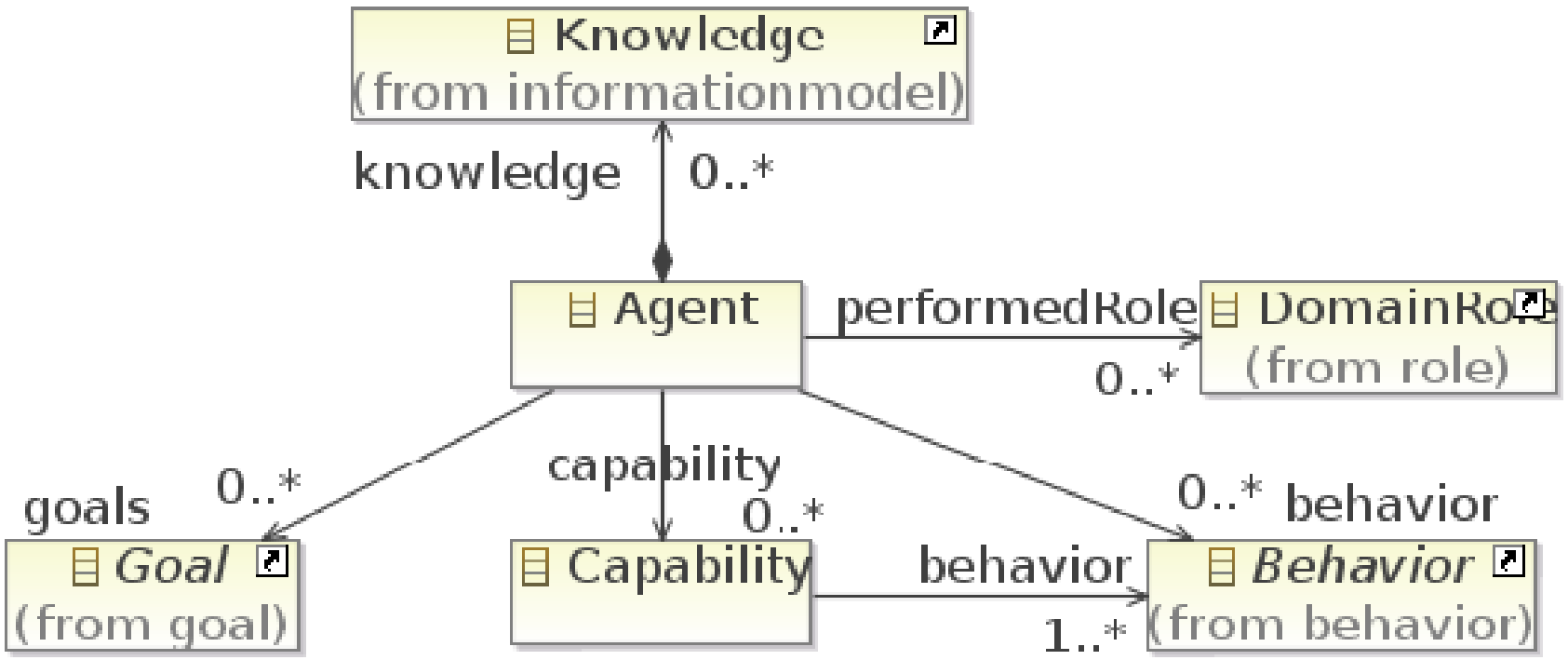
PIM4Agents is structured into several aspects each focusing on a specific viewpoint of a MAS:

- Agent aspect describes single autonomous entities, the capabilities they have to solve tasks and their roles they play within the MAS
- Organization aspect describes how single autonomous entities cooperate within the MAS and how complex organizational structures can be defined
- Interaction aspect describes how the interaction between autonomous entities or organizations takes place
- Behavioral aspect describes how plans are composed by complex control structures and simple atomic tasks
- Role aspect covers feasible specializations and how they could be related to each other
- Environment aspect contains any kind of resource that is dynamically created, shared, or used by agents or organizations

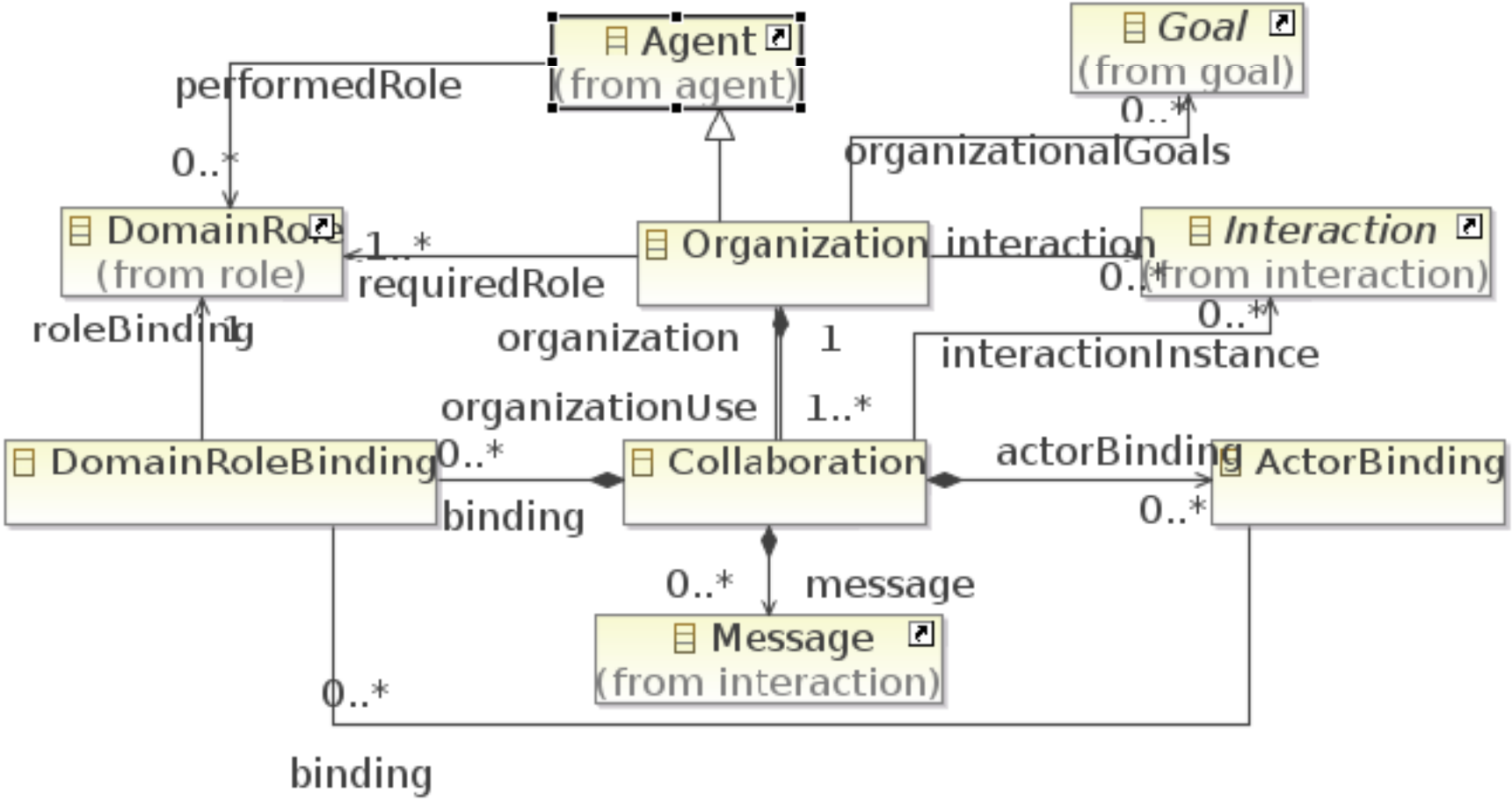
Multiagent System View



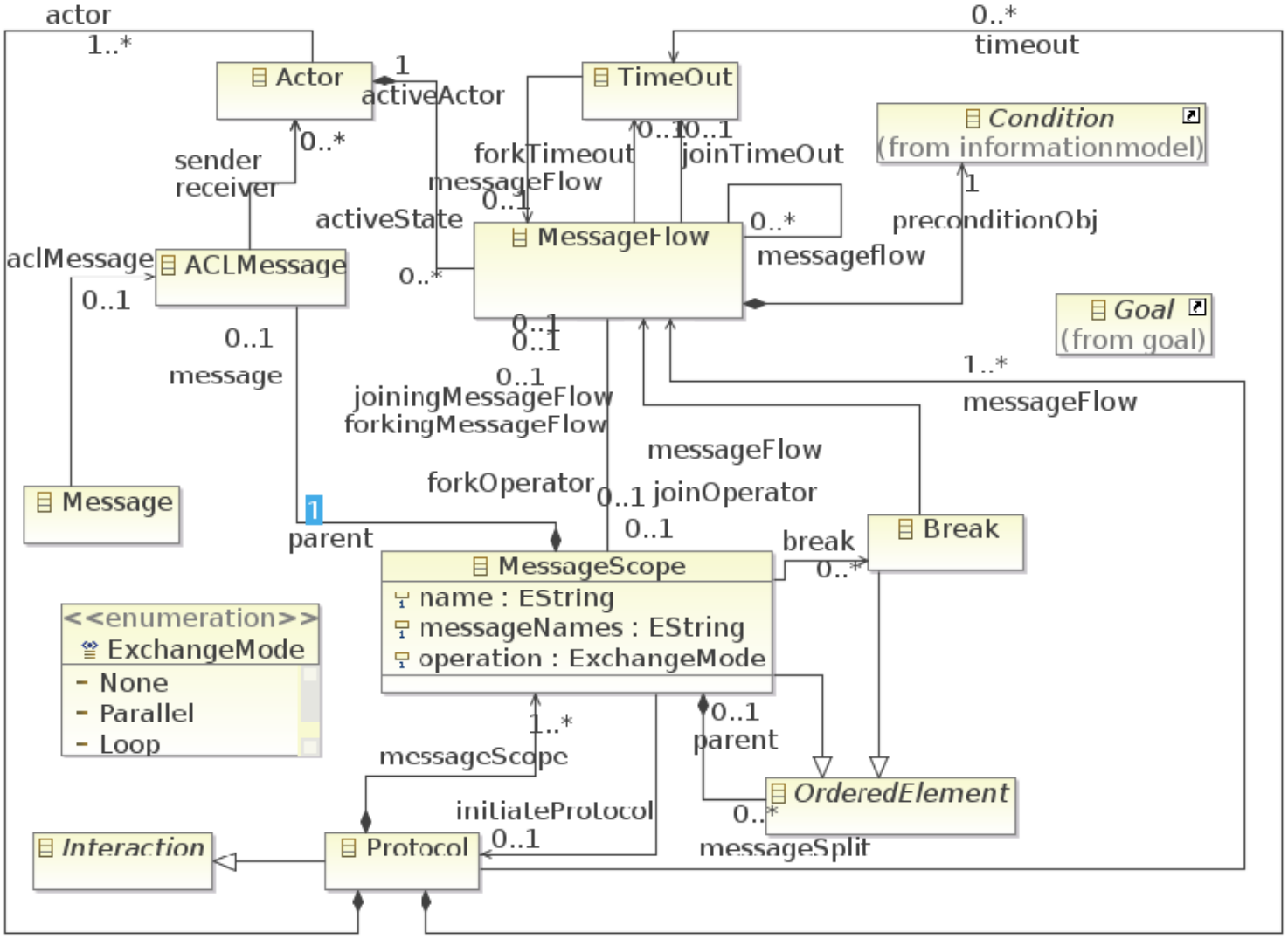
Agent View


Knowledge
(from informationmodel)
knowledge
0..*
Agent
performedRole
0..*
DomainRole
(from role)
goals
0..*
capability
0..*
0..*
behavior
Goal
(from goal)
Capability
behavior
1..*
Behavior
(from behavior)

Organization View

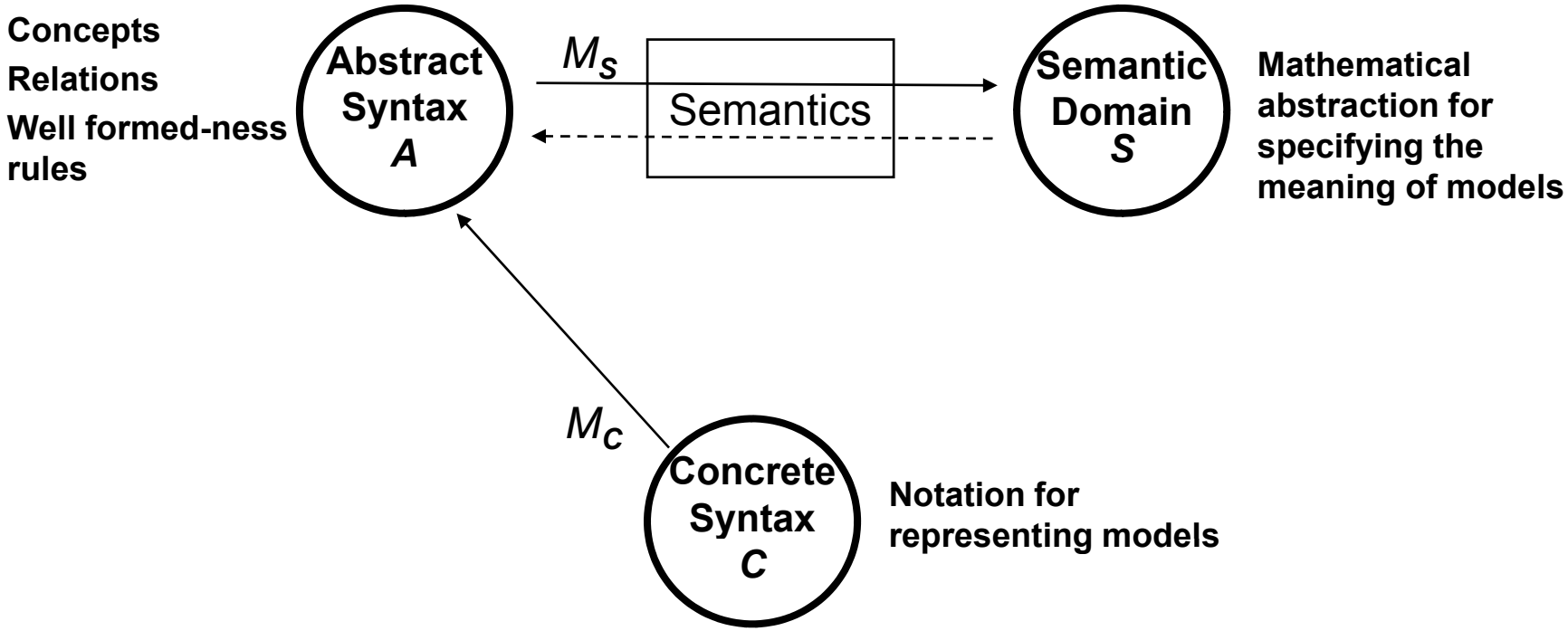


Interaction View



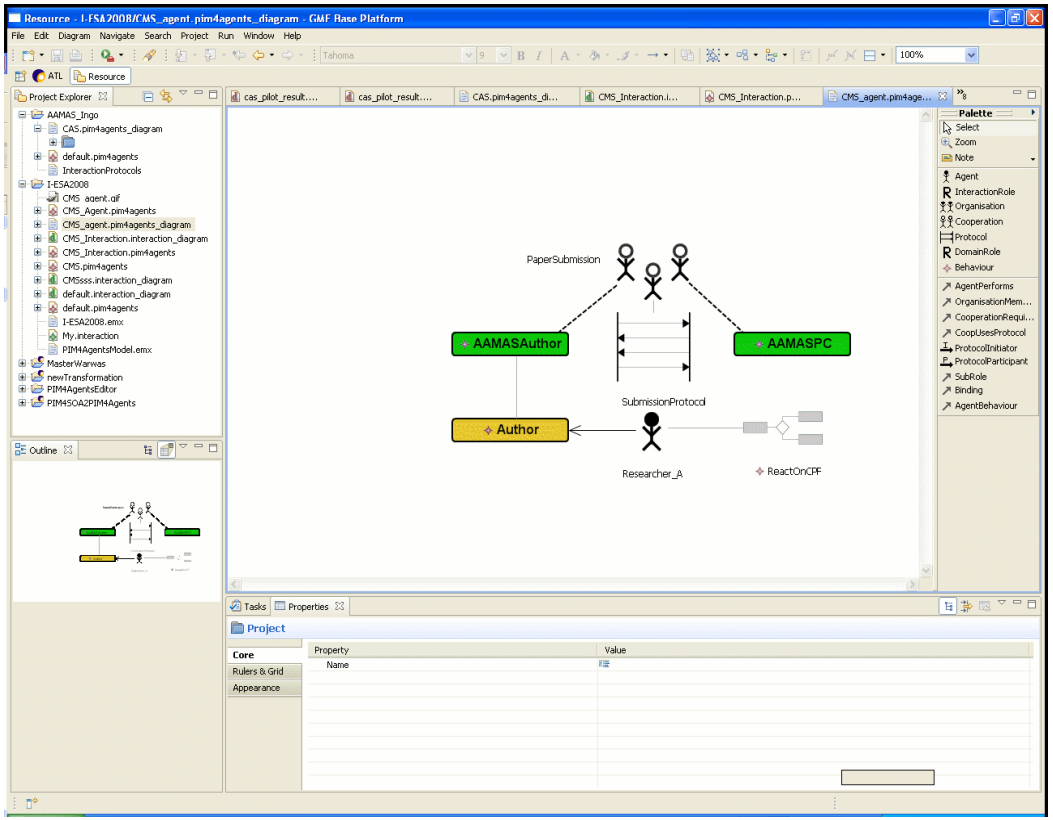
Specification of Domain Specific Modeling Languages (DSML)

$$L = < C, A, S, M_S, M_C >$$

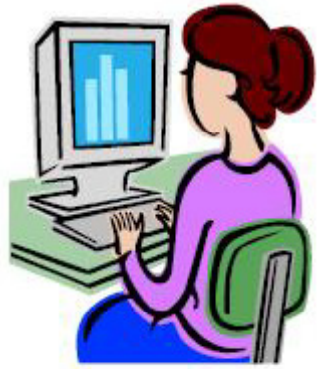


Concrete Syntax – Graphical Editor

- Graphical editor is based on the Graphical Modeling Framework (GMF) of Eclipse
- GMF is based on Ecore
- Abstracts syntax is mapped to concrete syntax
- Semantics defined with Object-Z is transformed to OCL constraints

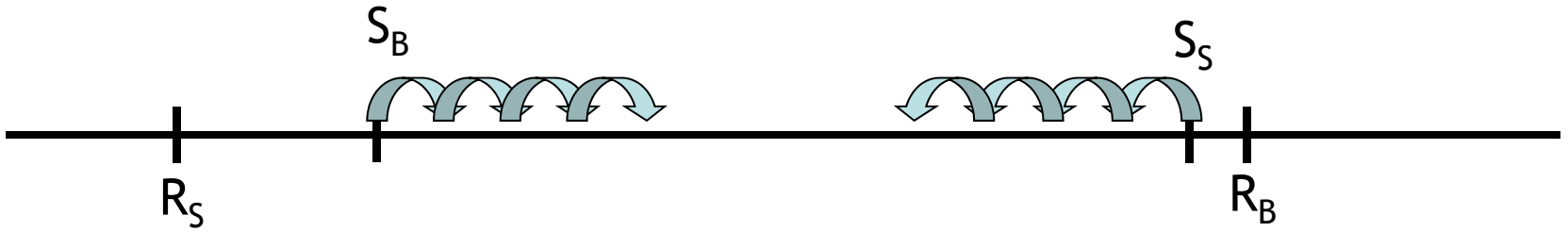


Alternating Concession Protocol

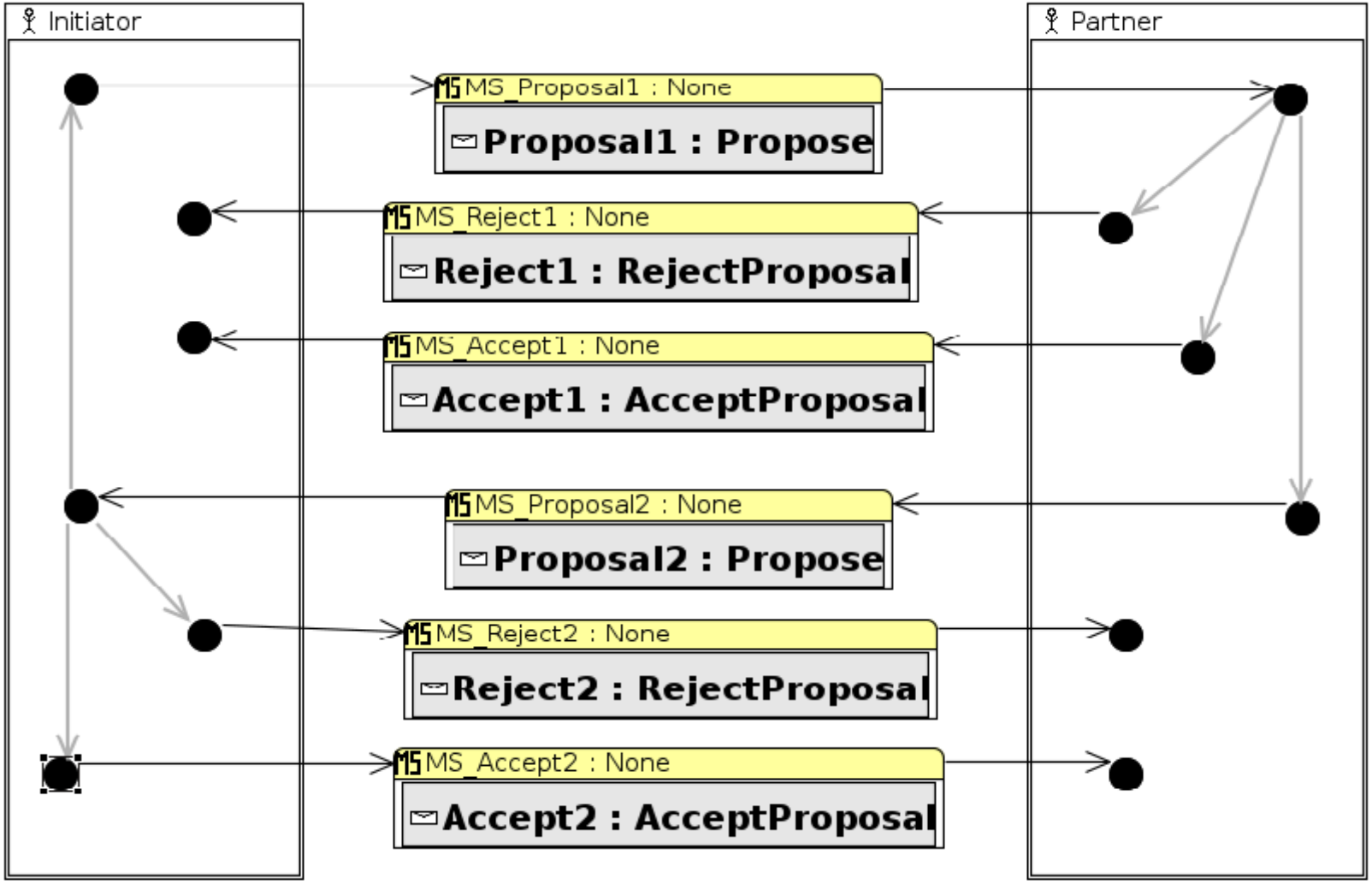


Seller

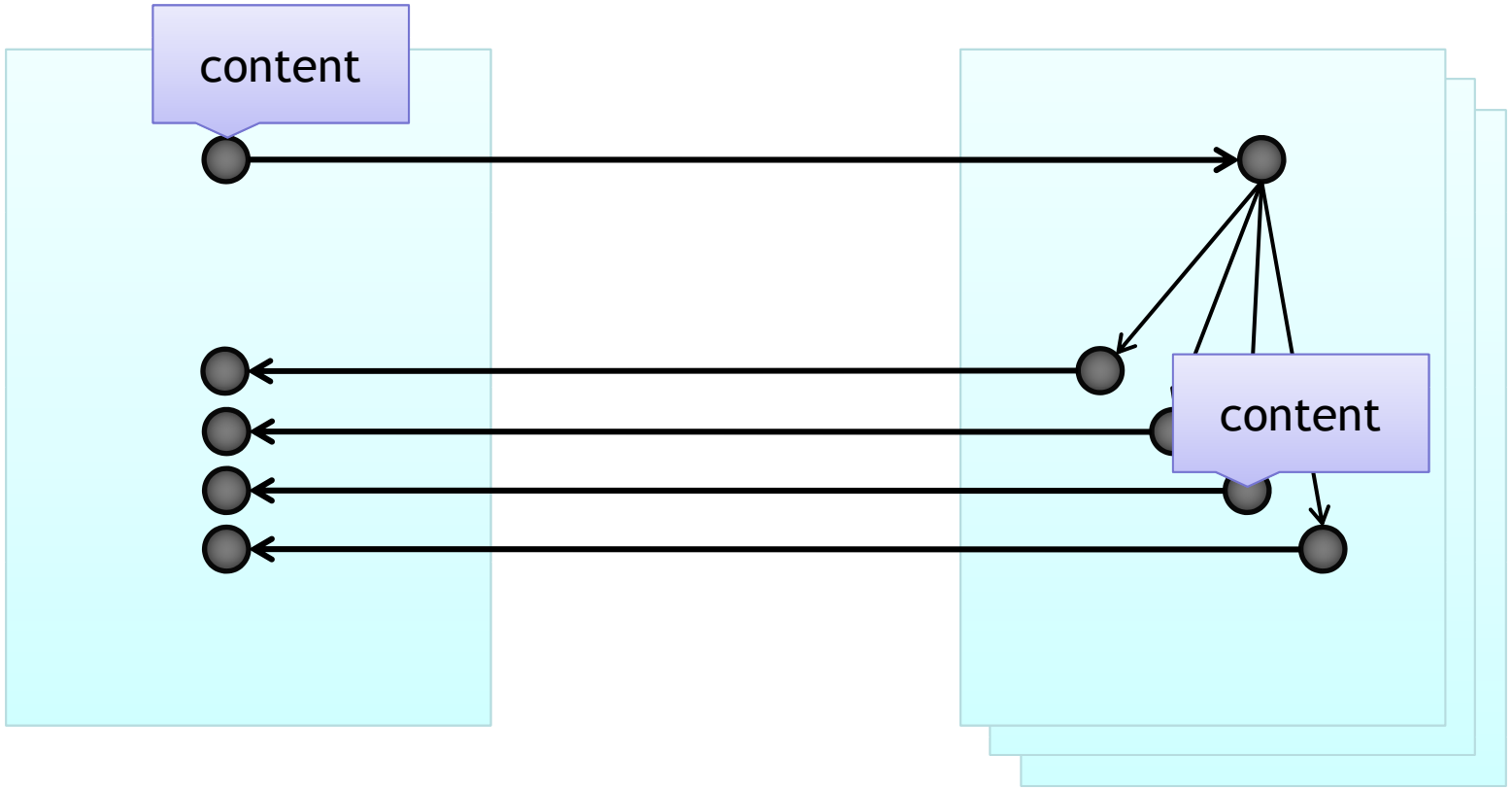
Buyer



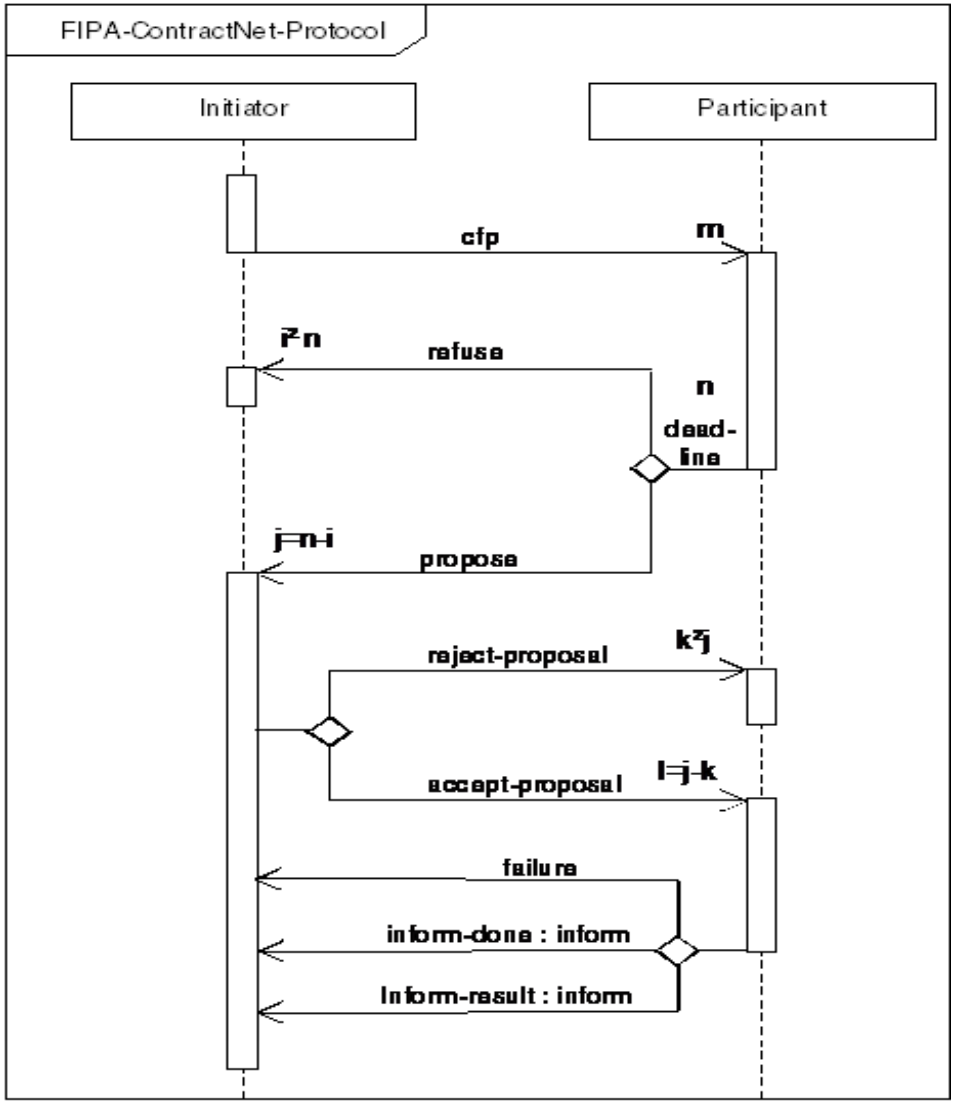
Alternating Concession Protocol (Pim4Agents)



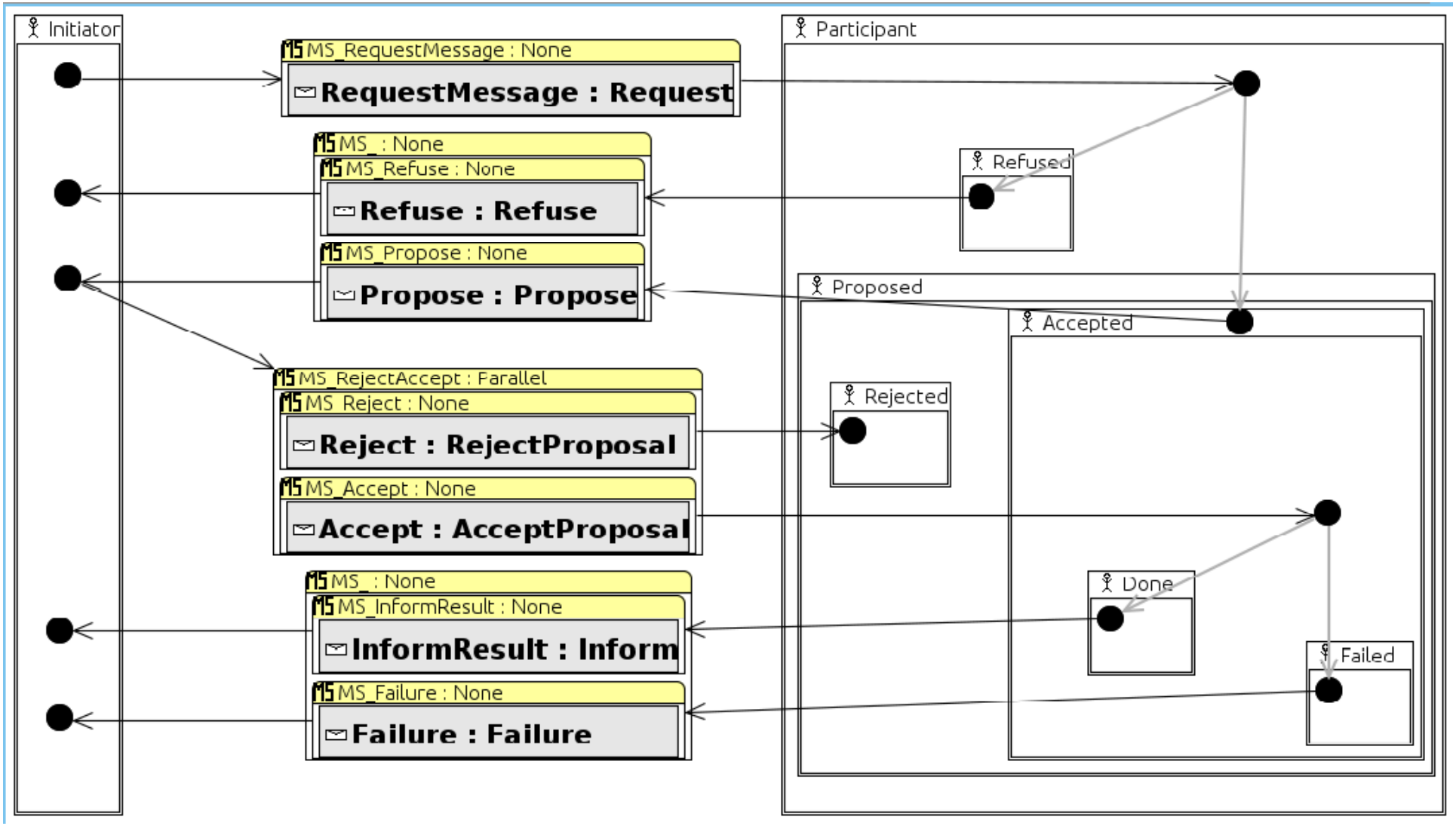
Communication Patterns



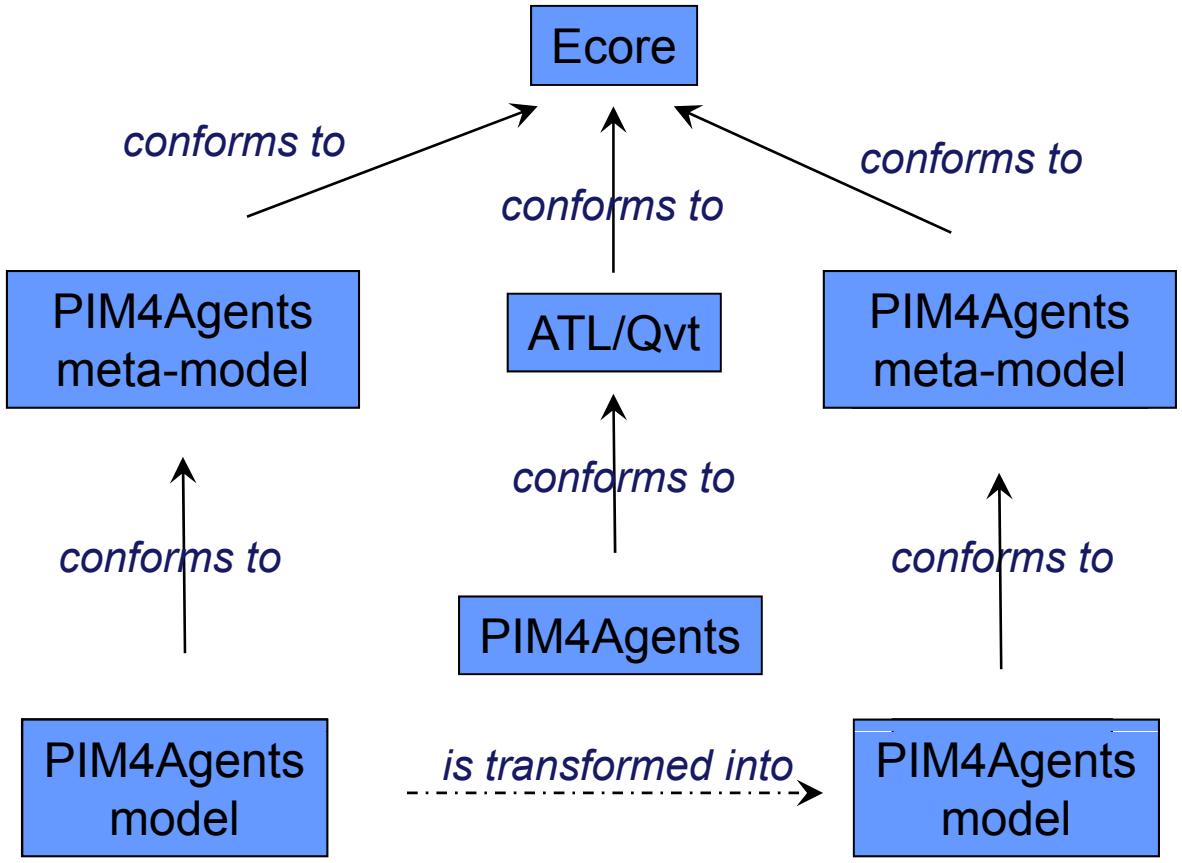
FIPA's Contract Net Specification



Contract Net Protocol (PIM4Agents)



PIM4Agents Model to Model Transformation

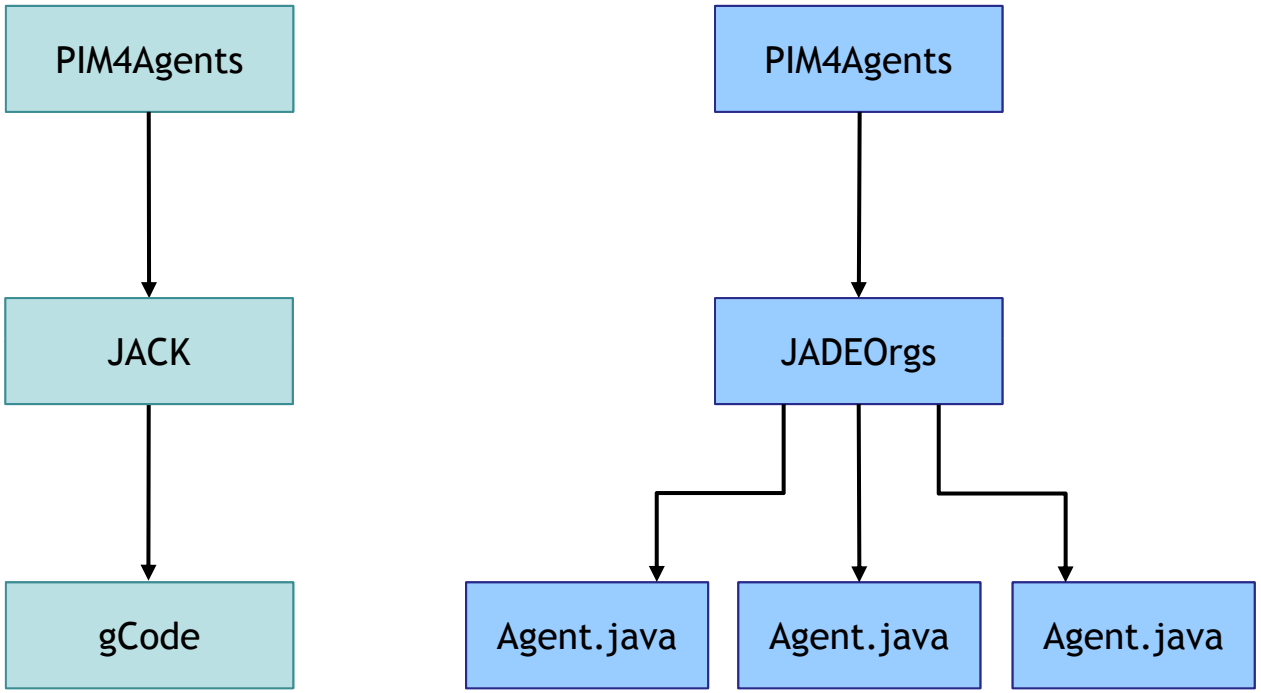


Model Transformations with Qvt

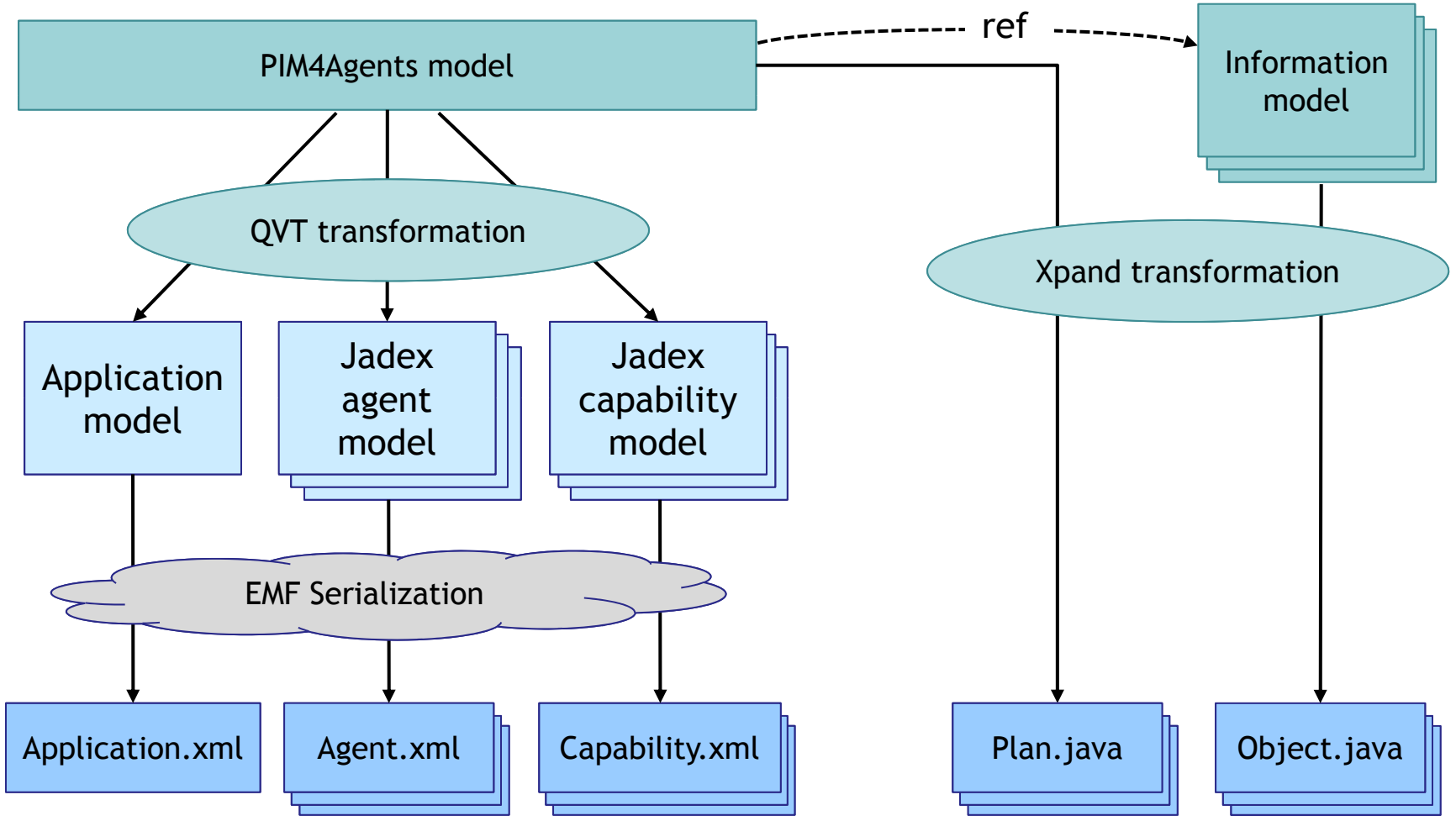
```
helper pim4agents::interaction::Actor::collectMsfs () : Set(pim4agents::interaction::MessageFlow) {
        var res : Set(pim4agents::interaction::MessageFlow);
        res := self.activeState;
        self.subactor->forEach(a) {
                res := res->union(a.collectMsfs());
        };
        return res
}
```

```
mapping P4A::interaction::Actor::toDomainRole(p : pim4agents::interaction::Protocol) : P4A::role::DomainRole
{
        var msf : Set(pim4agents::interaction::MessageFlow) := self.collectMsfs();
        var rmsf : Set(pim4agents::interaction::MessageFlow) :=
                msf -> select(d|d.isInitialMessageFlow or
                        ((d.forkOperator <> null) and (d.MsfSuccessors(msf)->size() > 0)));
        name := 'Role' + self.name;
        providesCapability := rmsf.map toCapability(msf,rmsf);
}
```

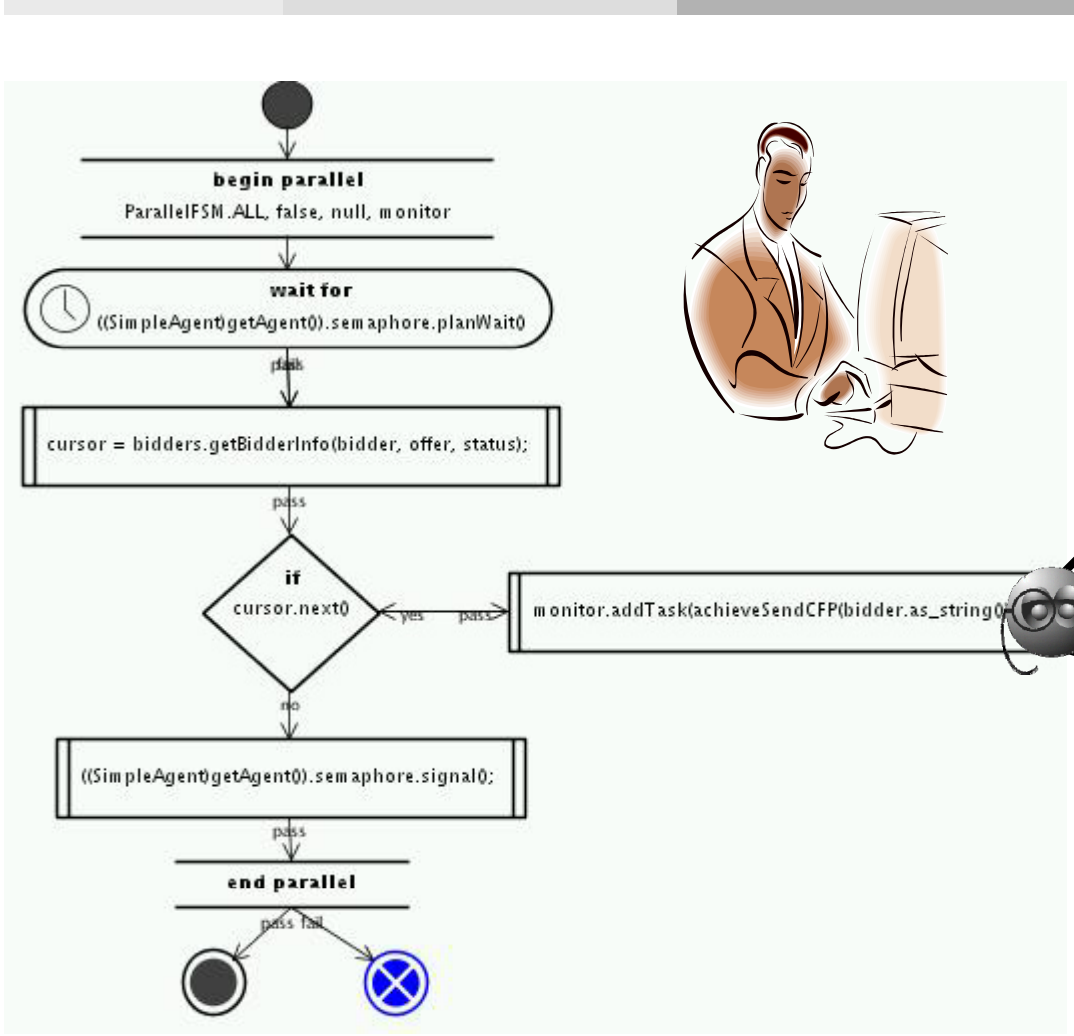
PIM4Agents to Jack/Jade Transformation

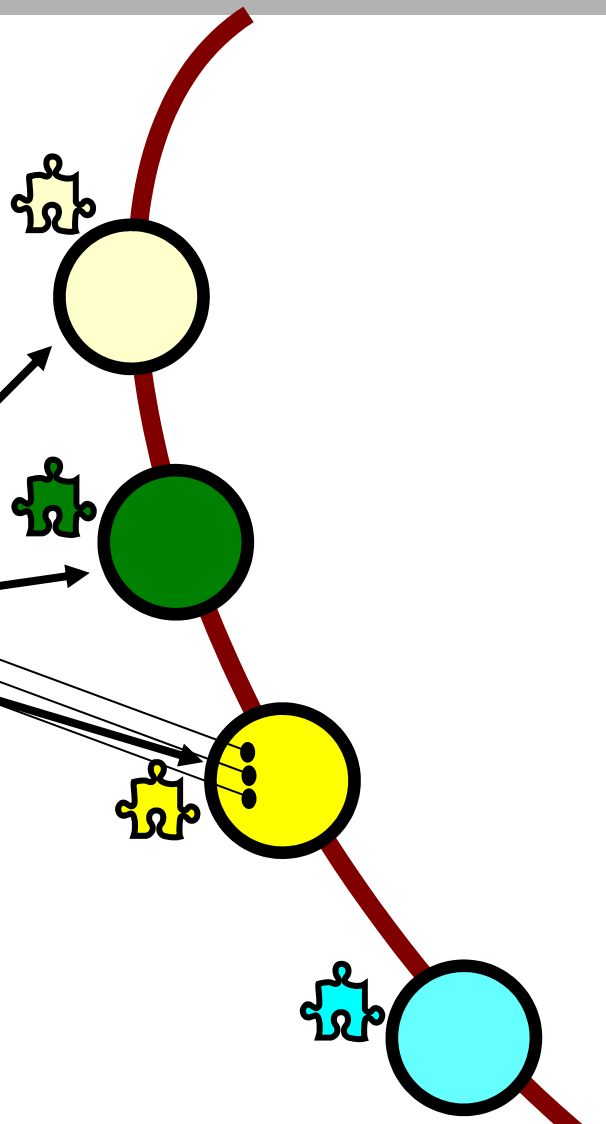


PIM4Agents to Jadex Transformation

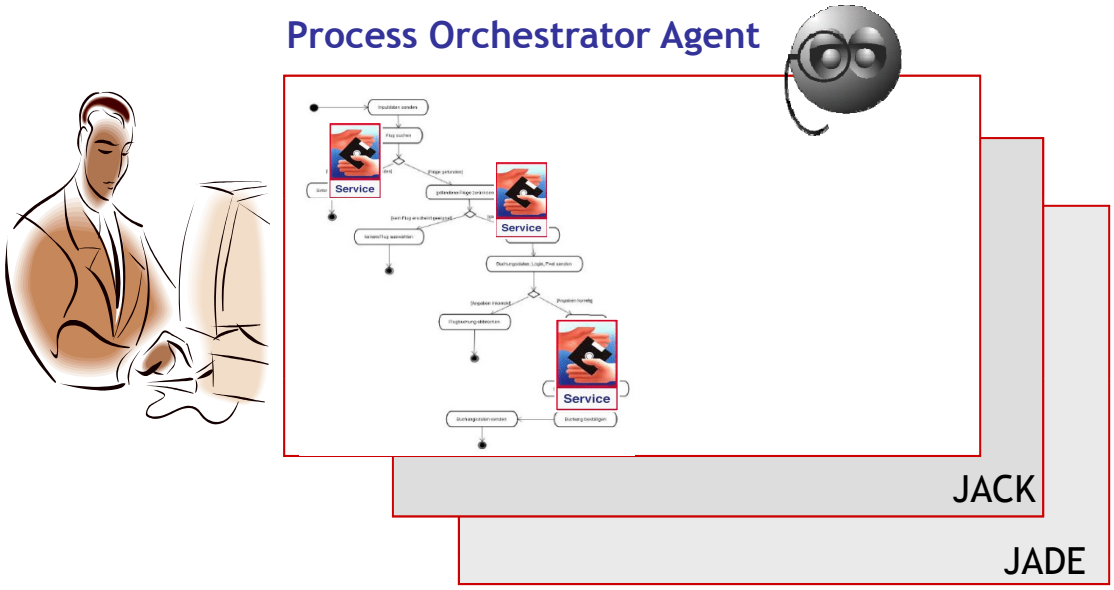


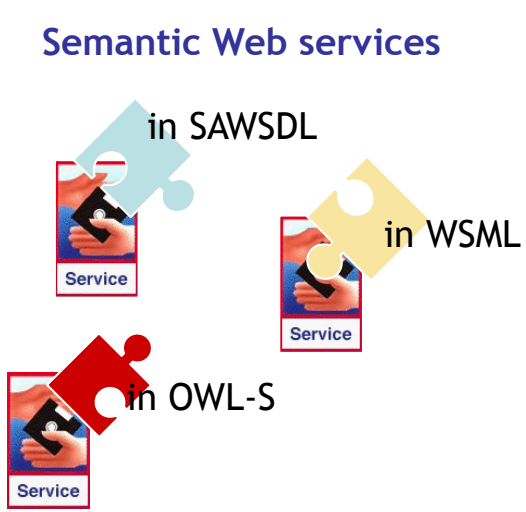
Model-Driven Service Composition



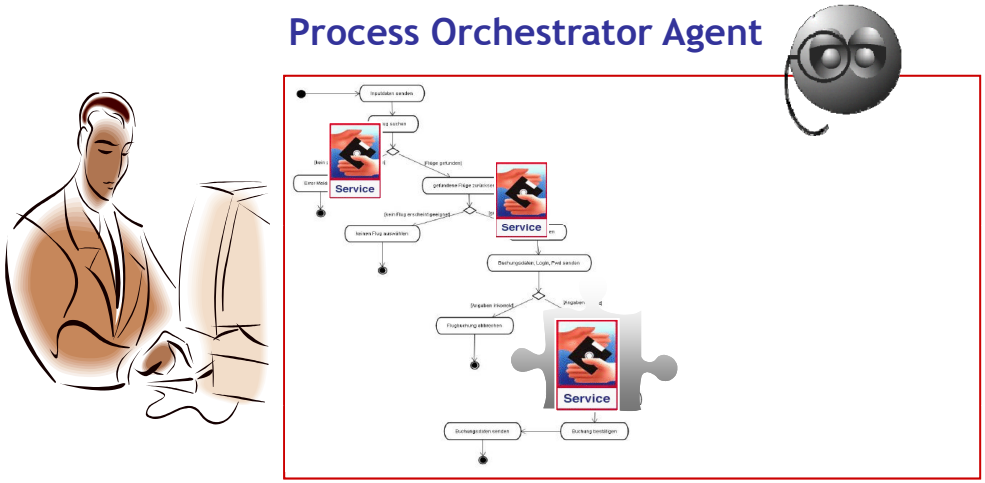


Integration with Semantic Services



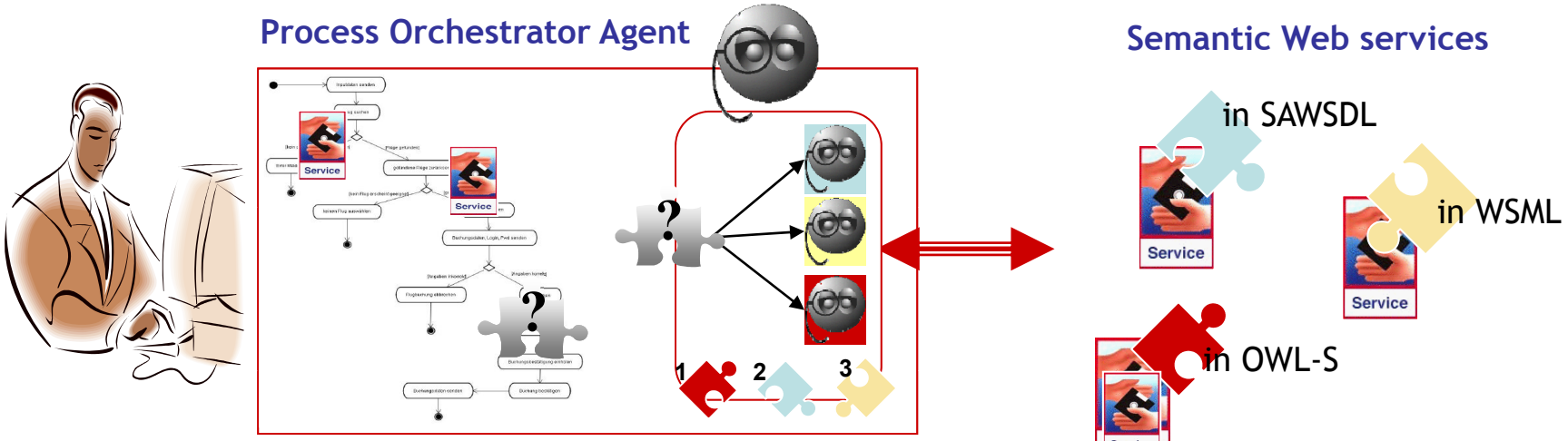


Integration with Semantic Services



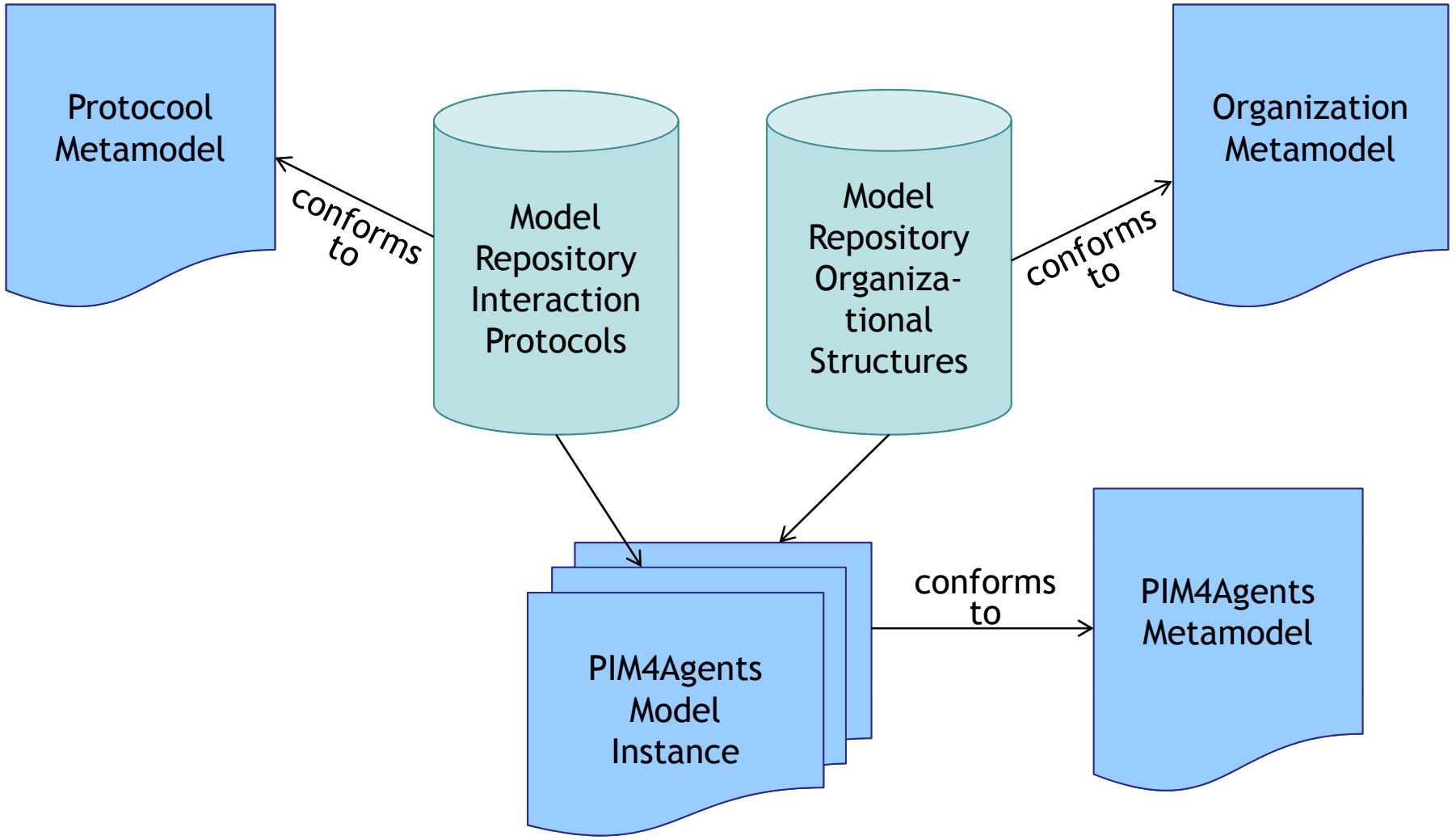
Service request

Integration with Semantic Services

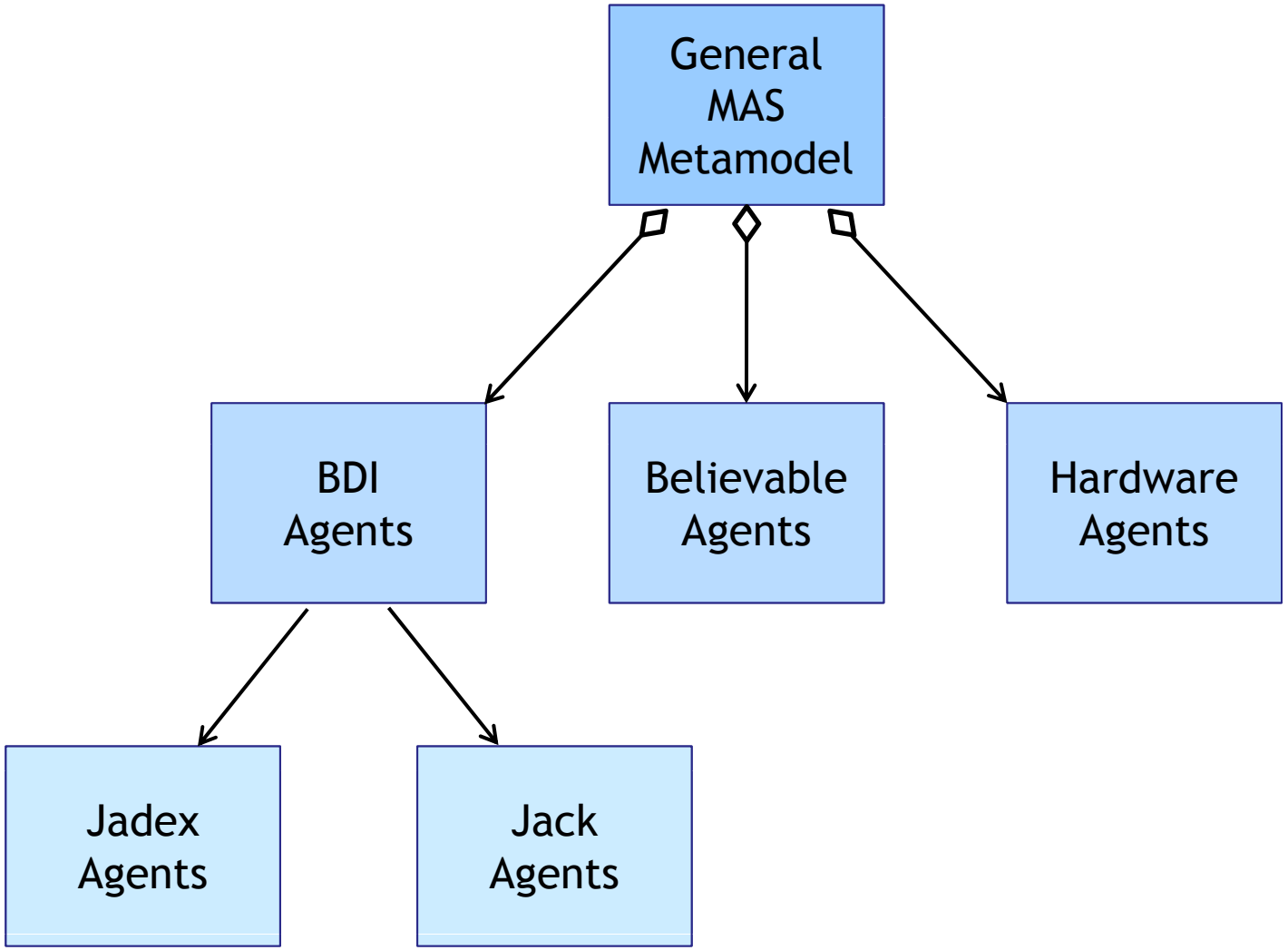


Service selection

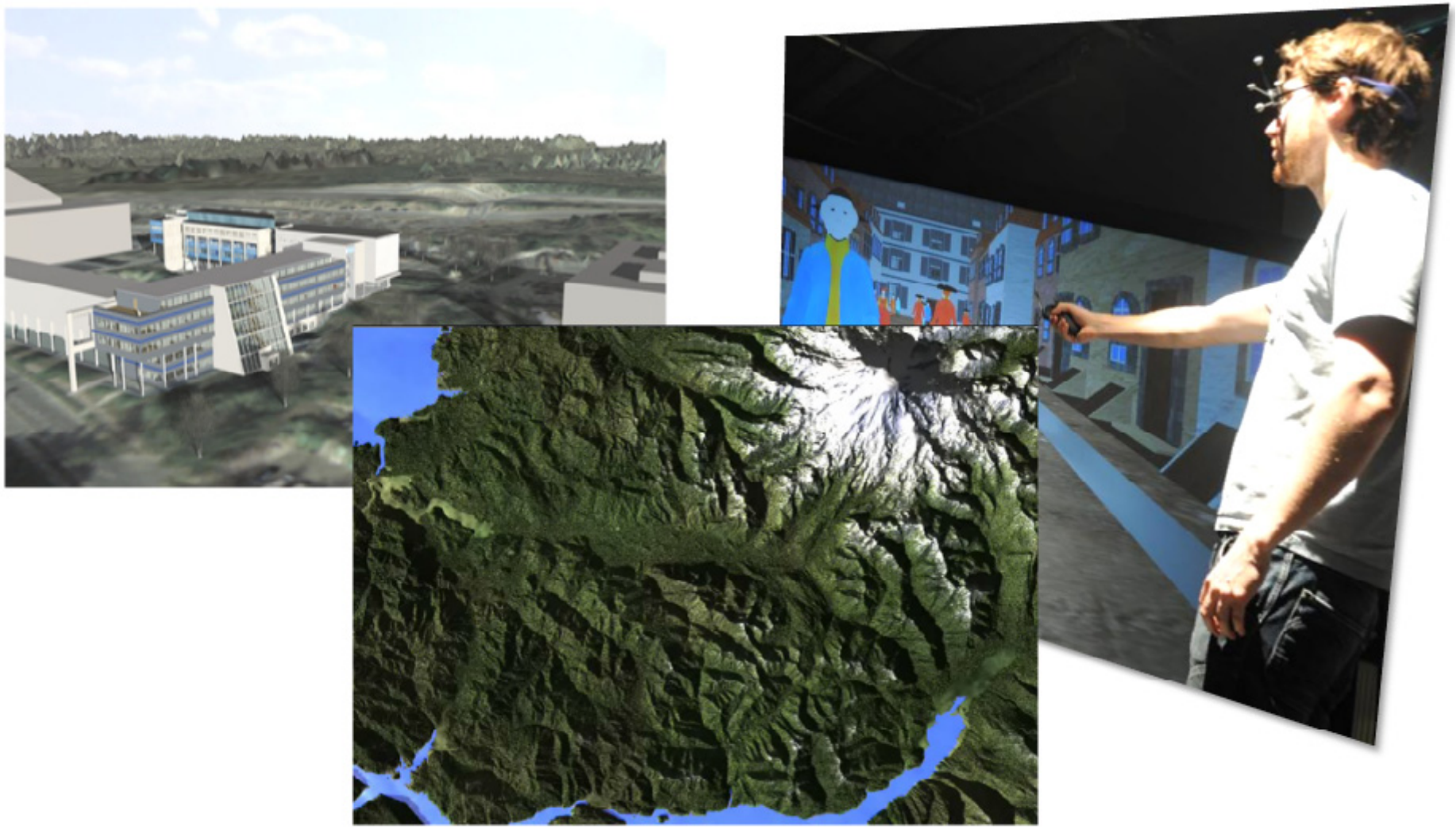
Collaborative Modelling



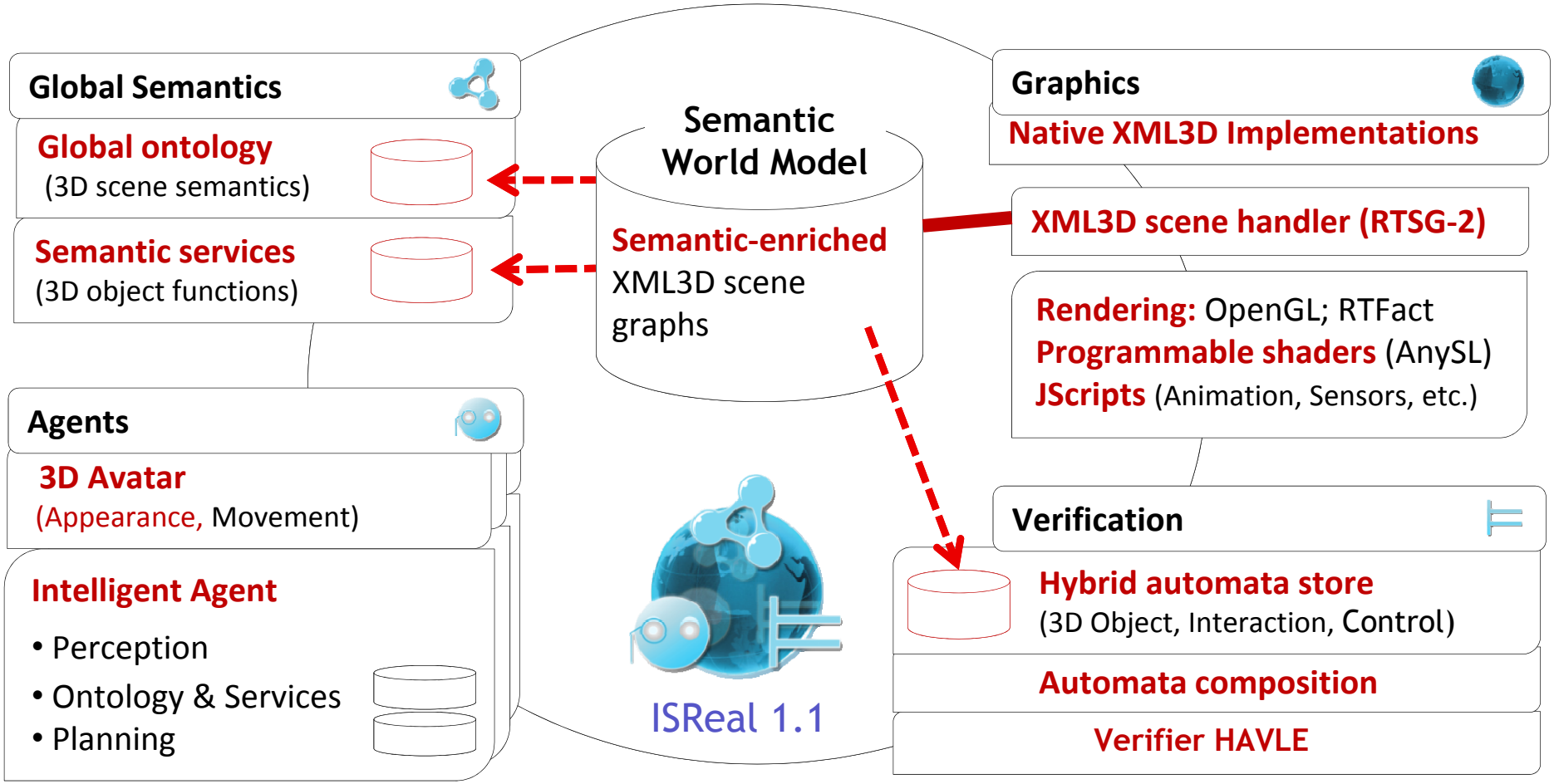
Specializing Metamodels



3DInternet & Real-time Raytracing



ISReal Platform: Components



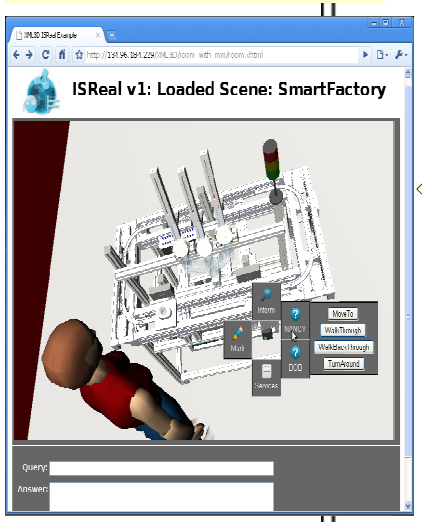
Nesbigall, Kapahnke, Warwas, Liedtke, Klusch: Proc. International Semantic Web Conference, 2010

ISReal Platform: Single-User Architecture

ISReal 1.1 Client

ISReal 1.1 Server

- Object semantics
- Avatar commands



Google Chrome

Renderer

DOM + XML3D sg

RTSG2

.xml3d

JScript Engine

- Animations
- Obj Change
- Sensors
- C/S Com

Graphics

Web Server

.xml3d

Semantics

Ont

Serv

Agents

Plans

Conclusion

Modeling is a hot topic for the near future

- in general for software engineering
- but especially for agent-based systems

Collaborative modeling should be supported

Accepted metamodels form the basis for establishing model repositories

Definition of semantics is a challenge!

How do logic formalisms like (OWL, OWL-S, WSMO, …) and UML-style model descriptions relate to each other?

Acknowledgements to the MAS group

Heads: Dr. Klaus Fischer, Prof. Dr. Matthias Klusch

Esteban Leon

Ingo Zinnikus

Martin Vasileski

Stefan Warwas

Patrick Kapahnke

Xiaoqi Cao

Former Group Members:

Christian Hahn

Cristián Madrigal

Stefan Nesbigall

Sven Jacobi