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

ICAART 2011
Autonomous Agents in Physical and Virtual Worlds
Virtual Enterprises, e-Business and Supply Chain Management
ATHENA: Structure & Domains

Business Level

Knowledge Level

Service-Oriented Architecture
Model-Driven Architecture

ICT Level

Interoperability of Business Applications
An MDA Approach to Service-Oriented Architectures (SOA)

- Business Model
- Service Model
- Platform Dependent Implementaton Models

CIM → PIM → PSM

Enterprise Model

Platform Independent Model for Service Oriented Architecture metamodel

UML Profile for SOA
- Information
- Service
- Process
- QoS

PIM Specification to PSM Specification

Platform Specific Model
- UML Profile for Web S.
- UML Profile for Agents
- UML Profile for BPEL
- UML Profile for P2P
PIM4SOA: 4 Views to System Design

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.
Organization View
Specification of Domain Specific Modeling Languages (DSML)

\[ L = <C, A, S, M_S, M_C> \]

- **Abstract Syntax** \( A \)
  - Concepts
  - Relations
  - Well formed-ness rules

- **Concrete Syntax** \( C \)
  - Notation for representing models

- **Semantics** \( M_S \)
  - Mathematical abstraction for specifying the meaning of models

- **Semantic Domain** \( S \)
  - Notation for representing models
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

$S_B$

$S_S$

$R_S$

$R_B$
Alternating Concession Protocol (Pim4Agents)
Communication Patterns
FIPA’s Contract Net Specification
Contract Net Protocol (PIM4Agents)
PIM4Agents Model to Model Transformation

Ecore

conforms to

conforms to

conforms to

PIM4Agents meta-model

ATL/Qvt

conforms to

PIM4Agents meta-model

conforms to

PIM4Agents

conforms to

PIM4Agents model

is transformed into

PIM4Agents model

JackMM

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

Diagram:

- PIM4Agents
  - JACK
    - gCode
  - Agent.java
  - Agent.java
  - Agent.java
PIM4Agents to Jadex Transformation

PIM4Agents model

QVT transformation

Application model

Jadex agent model

Jadex capability model

EMF Serialization

Application.xml

Agent.xml

Capability.xml

Information model

Xpand transformation

Plan.java

Object.java
Model-Driven Service Composition

begin parallel

Parallels start, false, null, monitor

wait for

((SimpleAgent0, getAgent0), semaphore, plan/await)

if
cursor.next0

((SimpleAgent0, getAgent0), semaphore, signal0):

end parallel

monitor.addTask(achieveSendCFR(bidder, as_string))
Integration with Semantic Services

Process Orchestrator Agent

Semantic Web services

in SAWSDL

in WSML

in OWL-S

ICAART 2011
Integration with Semantic Services

Process Orchestrator Agent

Semantic Web services

Service request
Integration with Semantic Services

Service selection

Process Orchestrator Agent

Semantic Web services

in SAWSDL

in WSML

in OWL-S

Integration with Semantic Services
Collaborative Modelling

Protocool Metamodel

conforms to

Model Repository Interaction Protocols

Model Repository Organizational Structures

conforms to

PIM4Agents Model Instance

conforms to

PIM4Agents Metamodel

Organization Metamodel
Specializing Metamodels

- General MAS Metamodel
  - BDI Agents
    - Jadex Agents
  - Believable Agents
    - Jack Agents
  - Hardware Agents
3DInternet & Real-time Raytracing
ISReal Platform: Components

Global Semantics
- Global ontology (3D scene semantics)
- Semantic services (3D object functions)

Agents
- 3D Avatar (Appearance, Movement)
- Intelligent Agent
  - Perception
  - Ontology & Services
  - Planning

Semantic World Model
- Semantic-enriched XML3D scene graphs

Graphics
- Native XML3D Implementations
- XML3D scene handler (RTSG-2)
- Rendering: OpenGL; RTFact Programmable shaders (AnySL) JScripts (Animation, Sensors, etc.)

Verification
- Hybrid automata store (3D Object, Interaction, Control)
- Automata composition
- Verifier HAVLE

ISReal Platform: Single-User Architecture

- Object semantics
- Avatar commands

**ISReal 1.1 Client**

**ISReal 1.1 Server**

- ISReal 1.1 Client
- ISReal 1.1 Server

---

**DOM + XML3D sg**

**Renderer**

**JScript Engine**
- Animations
- Obj Change
- Sensors
- C/S Com

**RTSG2**

**Graphics**

**Web Server**

**Semantics**

**Agents**

**Ont Serv**

**Plans**
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