Softwaretechnologie für Agenten

Seminar: Methoden der KI
Themenschwerpunkt Multiagentensysteme

Jan Ahrens

Fachhochschule Wedel

2. Juni 2010

Inhaltsverzeichnis

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Agent - Definition

Ein Agent ist ein **Computerprogramm** *das sich in einer bestimmten* **Umgebung** *befindet und welches fähig ist,* **eigenständige Aktionen** *in dieser Umgebung durchzuführen, um seine* **Ziele** *zu erreichen.*

- **Aktionen** werden meistens ausgeführt, um **Ziele** eines **Auftraggebers** zu erreichen
- Es existiert keine allgemein anerkannte Definition eines Agenten

Eigenschaften von Agenten

Um einen Agenten zu konstruieren, der diesen Zielen genügt, müssen folgene Eigenschaften gegeben sein:

- **autonom**: Agent hat Kontrolle über seinen Zustand und sein Verhalten
- **proaktiv**: Ziele werden durch Initiative des Agenten erreicht
- **reaktiv**: auf Veränderung der Agenten-Umwelt reagieren

Einsatzszenarien für Agenten

Semantic Web

Um einen Urlaub zu buchen genügt es seinem **persönlichen Web-Agenten** den **Auftrag** zu geben eine Unterkunft auf Mallorca zu buchen. Der Agent tritt in **Verhandlung** mit den Agenten von unterschiedlichen Hotel-Anbietern und handelt nach den Vorlieben seines Auftraggebers das beste Angebot aus und **bucht** das Hotel.

Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Von Agent zu Multiagenten-Systemen

Um **komplexe Problemstellungen** zu lösen, reicht der Einsatz eines Agenten oft nicht aus. Ein **Multiagenten-System** (MAS) ist eine Verbindung mehrerer unabhängiger Agenten:

- Agenten können in einem System **unterschiedliche Ziele** verfolgen
- Agenten **interagieren** mit anderen Agenten
- Kooperation durch **soziale** Eigenschaften der Agenten

Zusätzliche Anforderungen in Multiagenten-Systemen

In Multiagenten-Systemen treten eine Reihe von **Anforderungen** auf, die gelöst werden müssen:

1. Agenten müssen **kommunizieren** um zu kooperieren
2. Agenten müssen sich über ihr **Wissen austauschen**
3. Agenten müssen sich gegenseitig **kennen/finden**, um zu kommunizieren

Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Grundzüge der Kommunikation

- Um Ziele des Agenten zu erreichen, besteht die Notwendigkeit ...
  - zum **Austausch** von Wissen
  - zur **Delegation** von Aufgaben
- Kommunikation zum Erreichen der eigenen Ziele
- Kommunikation zwischen Agenten erfolgt **asynchron**
  - Antworten brauchen Zeit
  - Agenten können gleichzeitig mehrere Konversationen halten

Kommunikation nach FIPA

Damit die **Kommunikation von Agenten** einheitlich erfolgt, hat die **'Foundation for Itelligent Physical Agents'** (FIPA) eine Reihe von Standards herausgegeben.

- FIPA wurde 1996 mit dem Ziel gegründet die **Interoperabilität** von Agenten-Systemen im kommerziellen und industriellen Umfeld **zu fördern**
- FIPA gehört zur **'IEEE Computer Society'**

Zu den Standards gehört die **FIPA Communicative Act Library Specification** (00037), welche eine Reihe von **Sprechakten** mit formal definierter Semantik angibt.

Sprechakttheorie

- Sprechakttheorie wurde 1962 von John Langshaw Austin begründet
  - Fragestellung: Wie verwenden wir **Sprache** um alltägliche Absichten und **Ziele** zu erreichen?
  - Beobachtung: Durch **Äußerungen** werden nicht nur **Sachverhalte** beschrieben und **Behauptungen** aufgestellt, sondern auch **Handlungen** (Sprechakte) vollzogen
  - Diese Äußerungen werden **performative Äußerungen** genannt
  - Ziel der Sprechakte ist es den Zustand der Umwelt zu verändern
- Agenten kommunizieren um ihre Ziele zu erreichen und den Zustand der Umwelt zu verändern.
- Kommunikation zwischen Agenten wird durch **performative Äußerungen** der Sprechakttheorie modelliert.

Sprechakte der FIPA

Zur FIPA konformen Agenten-Kommunikation wurden **22 performative Verben** definiert, die in der **Communicative Act Library Specification** festgelegt wurden.

- query
- inform
- request
- agree
- refuse
- propose
- reject proposal
- ...

Kommunikationsprotokolle

Aufbauend auf der Communicative Act Library, wurden von der FIPA 9 **Interaktionsprotokolle** spezifiziert, mit deren Hilfe eine **geordnete Agenten-Kommunikation** erfolgt:

- **Request** Interaction Protocol (SC00026)
- **Query** Interaction Protocol (SC00027)
- **Request When** Interaction Protocol (SC00028)
- **Brokering** Interaction Protocol (SC00033)
- **Recruiting** Interaction Protocol (SC00034)
- **Subscribe** Interaction Protocol (SC00035)
- **Propose** Interaction Protocol (SC00036)
- **Contract Net** Interaction Protocol (SC00029)
- **Iterated Contract Net** Interaction Protocol (SC00030)

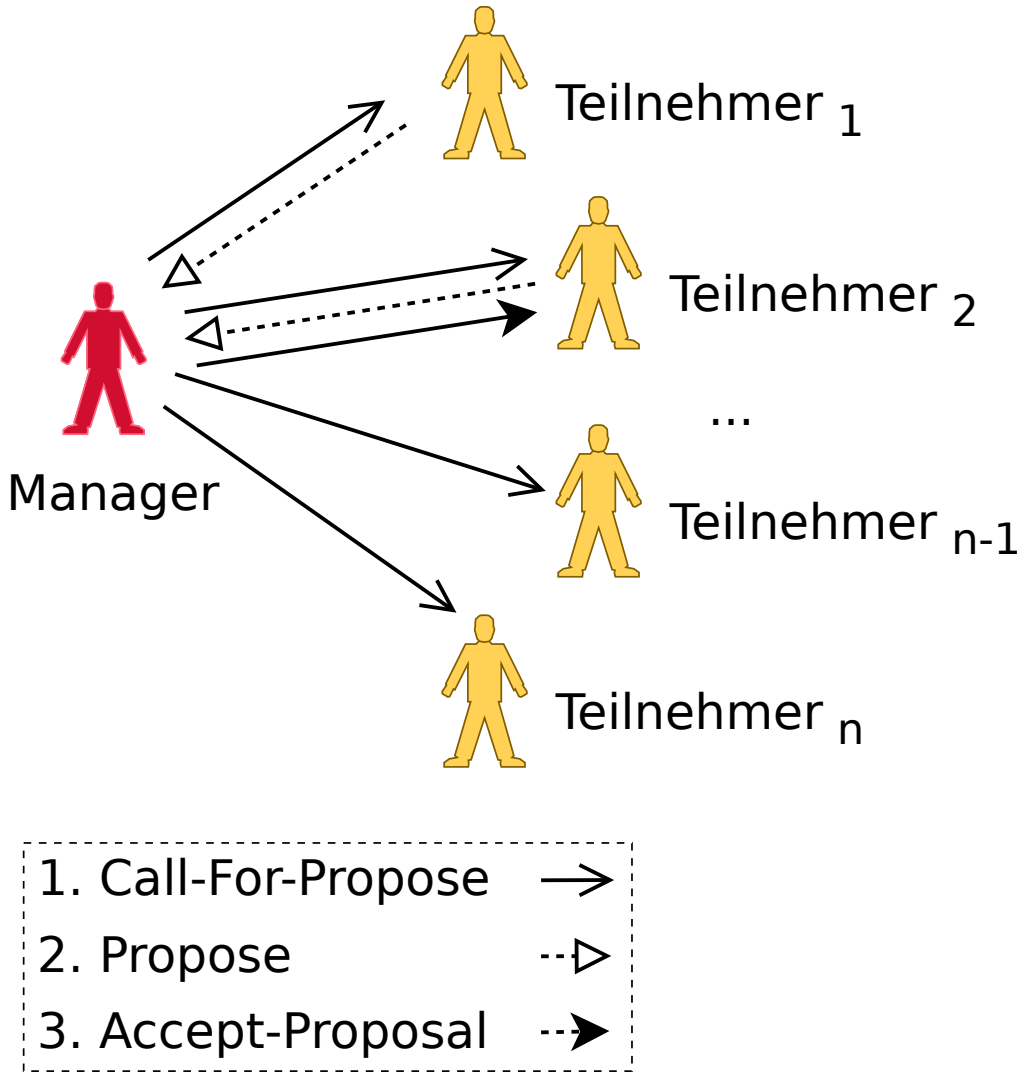
Kommunikationsprotokolle

Aufbauend auf der Communicative Act Library, wurden von der FIPA 9 **Interaktionsprotokolle** spezifiziert, mit deren Hilfe eine **geordnete Agenten-Kommunikation** erfolgt:

- **Request** Interaction Protocol (SC00026)
- **Query** Interaction Protocol (SC00027)
- **Request When** Interaction Protocol (SC00028)
- **Brokering** Interaction Protocol (SC00033)
- **Recruiting** Interaction Protocol (SC00034)
- **Subscribe** Interaction Protocol (SC00035)
- **Propose** Interaction Protocol (SC00036)
- **Contract Net** Interaction Protocol (SC00029)
- **Iterated Contract Net** Interaction Protocol (SC00030)

Das Contract Net Protokoll

Ausschnitt einer Contract Net Auktion (nicht iterativ):



Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Ontologie

Definition:

> *Ontologien [...] sind [...] formal geordnete Darstellungen einer Menge von Begrifflichkeiten und der zwischen ihnen bestehenden Beziehung in einem bestimmten Gegenstandsbereich.*

Ontologien werden verwendet um:

- Wissen mit seiner Semantik zu repräsentieren und auszutauschen
- Neues Wissen zu erschließen
  - Wissen wird durch Inferenz (logisches Folgern) erschlossen

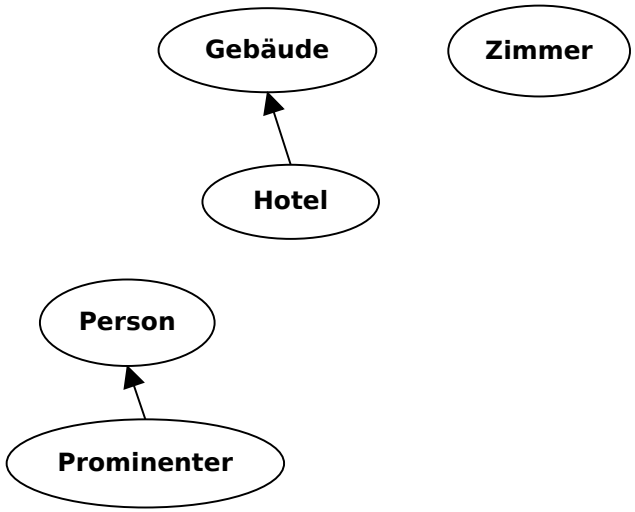
Im Agenten-Kontext werden Ontologien eingesetzt, um den **Inhalt von Sprechakten** eine Semantik zu geben.

> *Beispiel: Agent A* **bittet** *Agent B eine* **Hotelbuchung** *für* **ein Vier-Sterne-Hotel** *auf* **Mallorca** *durchzuführen.*

Eine Beispiel Ontologie

Beispiel einer einfachen Hotel-Gast-Ontologie:

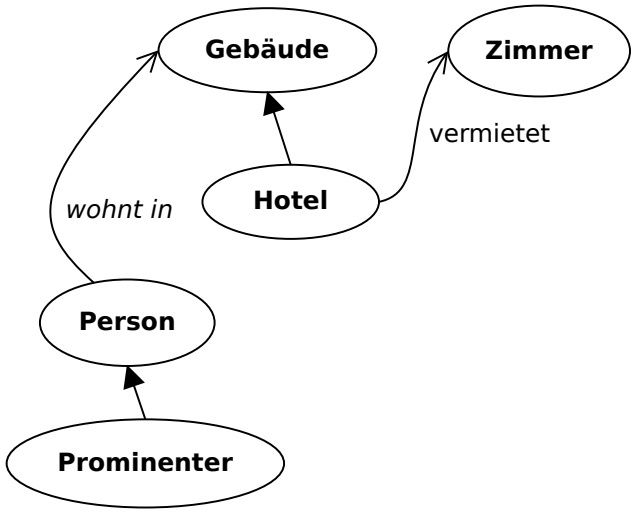
- Ein **Hotel** ist ein **Gebäude**.
- Ein **Prominenter** ist eine **Person**.
- Ein Hotel **vermietet** mehrere **Zimmern**.
- Eine **Person** kann in einem **Gebäude** **wohnen**.
- Das **Atlantic** ist ein **Hotel**.
- **Udo L.** ist ein **Prominenter**.
- **Udo L.** wohnt im **Atlantic**.



Eine Beispiel Ontologie

Beispiel einer einfachen Hotel-Gast-Ontologie:

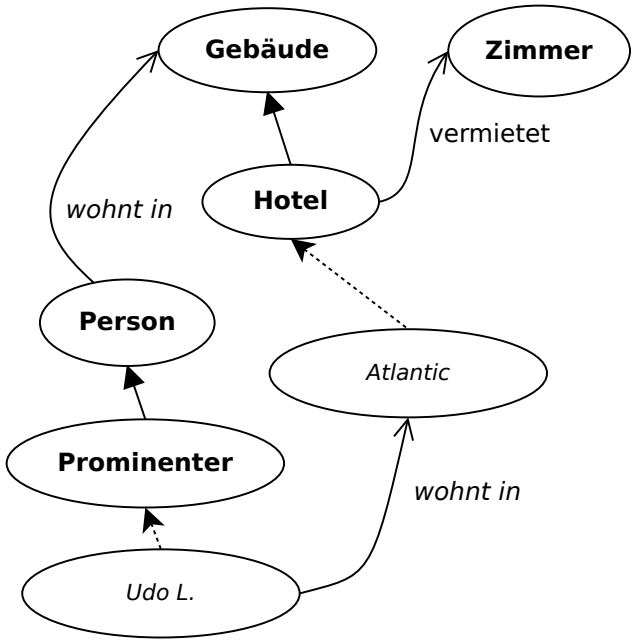
- Ein **Hotel** ist ein **Gebäude**.
- Ein **Prominenter** ist eine **Person**.
- Ein Hotel **vermietet** mehrere **Zimmern**.
- Eine **Person** kann in einem **Gebäude wohnen**.
- Das **Atlantic** ist ein **Hotel**.
- **Udo L.** ist ein **Prominenter**.
- **Udo L.** wohnt im **Atlantic**.



Eine Beispiel Ontologie

Beispiel einer einfachen Hotel-Gast-Ontologie:

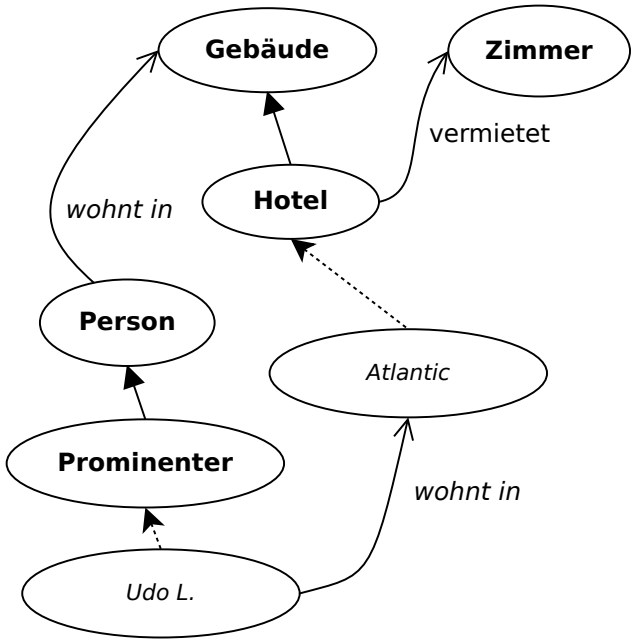
- Ein **Hotel** ist ein **Gebäude**.
- Ein **Prominenter** ist eine **Person**.
- Ein Hotel **vermietet** mehrere **Zimmern**.
- Eine **Person** kann in einem **Gebäude wohnen**.
- Das **Atlantic** ist ein **Hotel**.
- **Udo L.** ist ein **Prominenter**.
- **Udo L.** wohnt im **Atlantic**.



Eine Beispiel Ontologie

Beispiel einer einfachen Hotel-Gast-Ontologie:

- Ein **Hotel** ist ein **Gebäude**.
- Ein **Prominenter** ist eine **Person**.
- Ein Hotel **vermietet** mehrere **Zimmern**.
- Eine **Person** kann in einem **Gebäude wohnen**.
- Das **Atlantic** ist ein **Hotel**.
- **Udo L.** ist ein **Prominenter**.
- **Udo L.** wohnt im **Atlantic**.



Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen

2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien

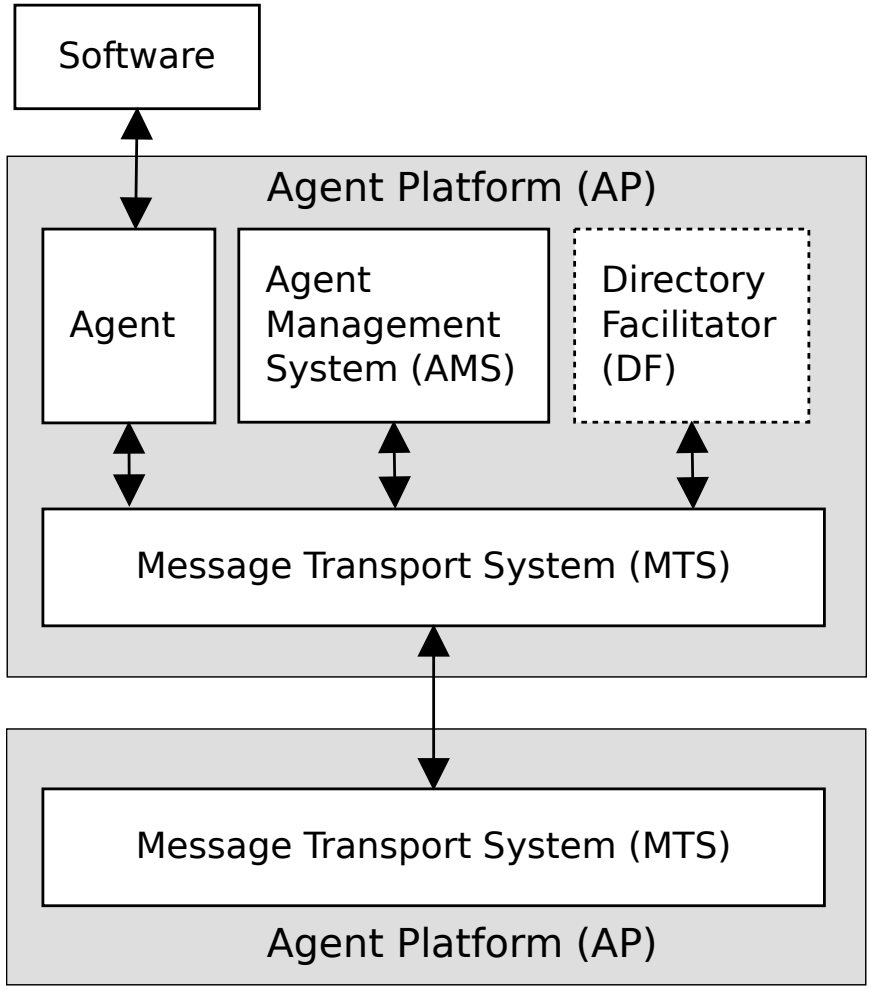
3. Einschätzung und Bewertung

Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Agent Management Reference Model

Der FIPA-Standard **FIPA Agent Management Specification** (00023) beschreibt ein Referenz-Modell für eine Agenten-Plattform:



Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

JADE Framework

Eigenschaften des Frameworks:

- **J**ava **A**gent **DE**velopment Framework
- entwickelt von der Telecom Italia
- basiert auf FIPA Spezifikationen
- aktuelle Version 4.0 vom 20.04.2010
- Lizenziert unter Open-Source Lizenz (LGPLv2)

JADE Framework

Live-Demo

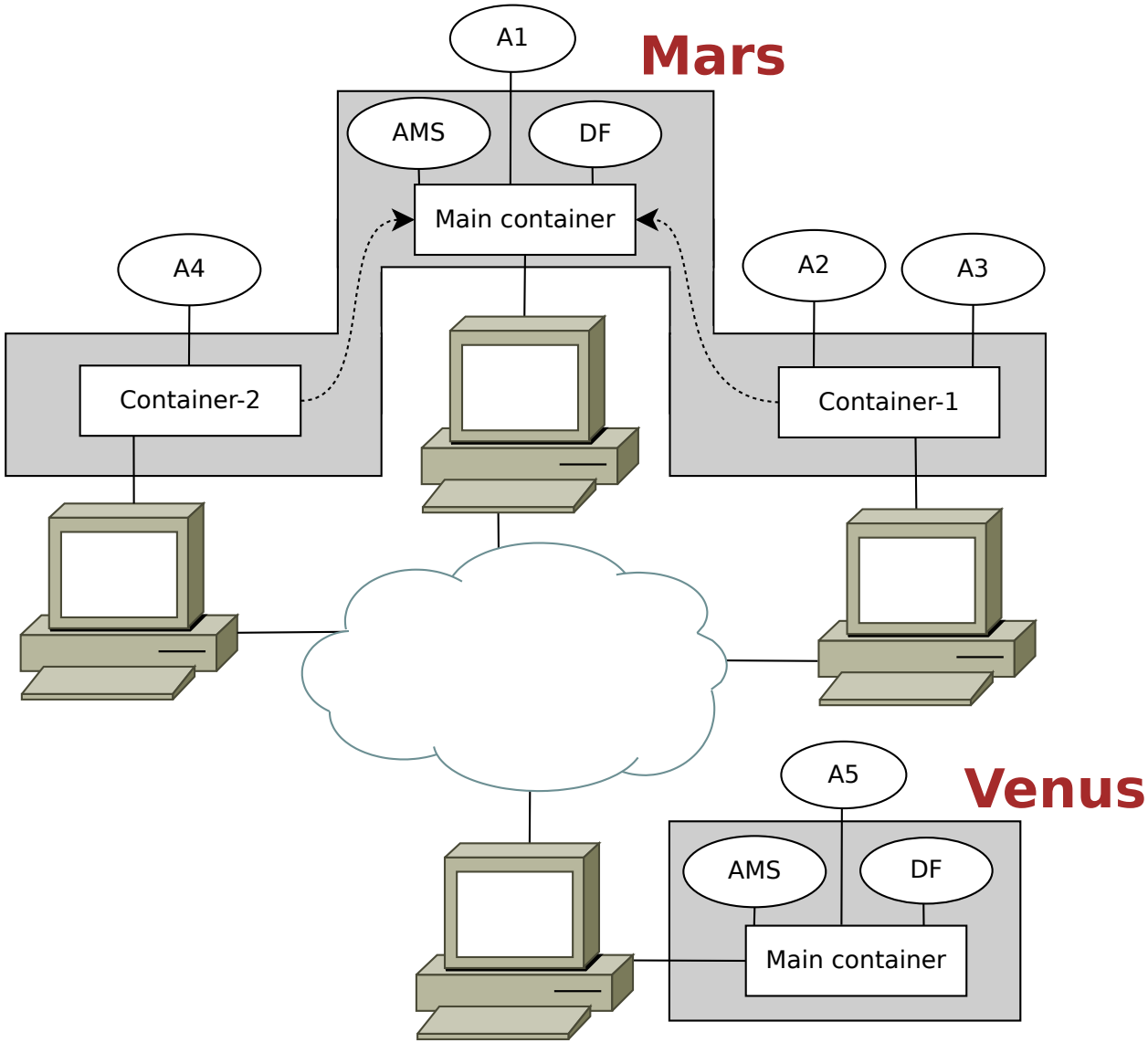
- Die JADE Plattform
- Der Remote Management Agent (RMA)
- Der Directory Facilitator (DF)

JADE Architektur Details

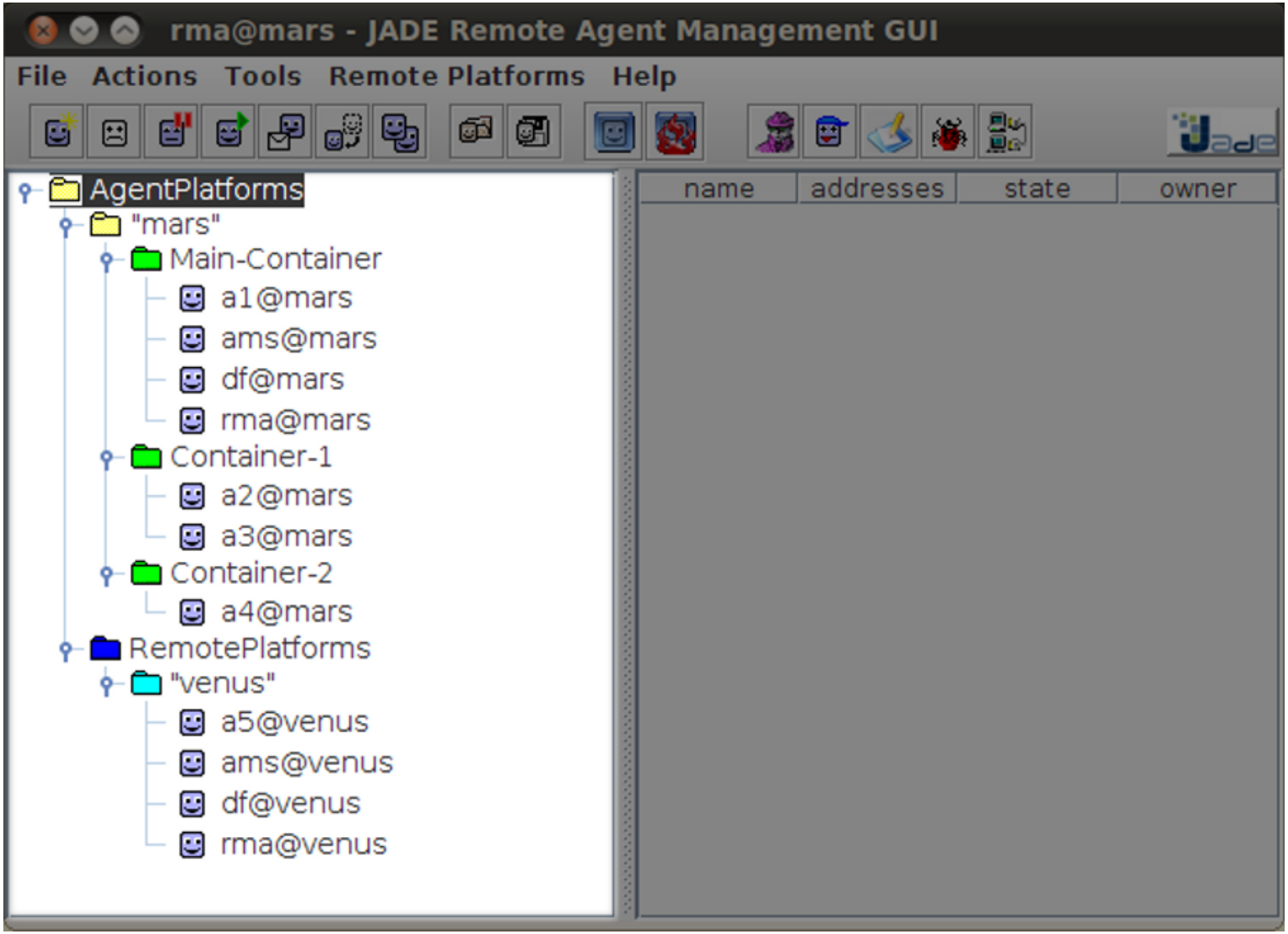
Zusätzlich zu den bereits erwähnten Plattform Bestandteilen (AMS, DF, MTS), bietet JADE ein **Container**-Konzept, zur strukturierten Verwaltung von Agent:

- eine Plattform besteht aus beliebig vielen Containern
- eine Plattform stellt immer einen **Main-Container** bereit
- innerhalb des Main-Container laufen immer die AMS und DF Agenten
- der Main-Container kann weitere Agent beinhalten, wenn diese zusammen mit der Plattform gestartet werden (z.B. RMA)
- weitere Container können auf anderen JVM und Rechnern laufen

Komplexes JADE Setup

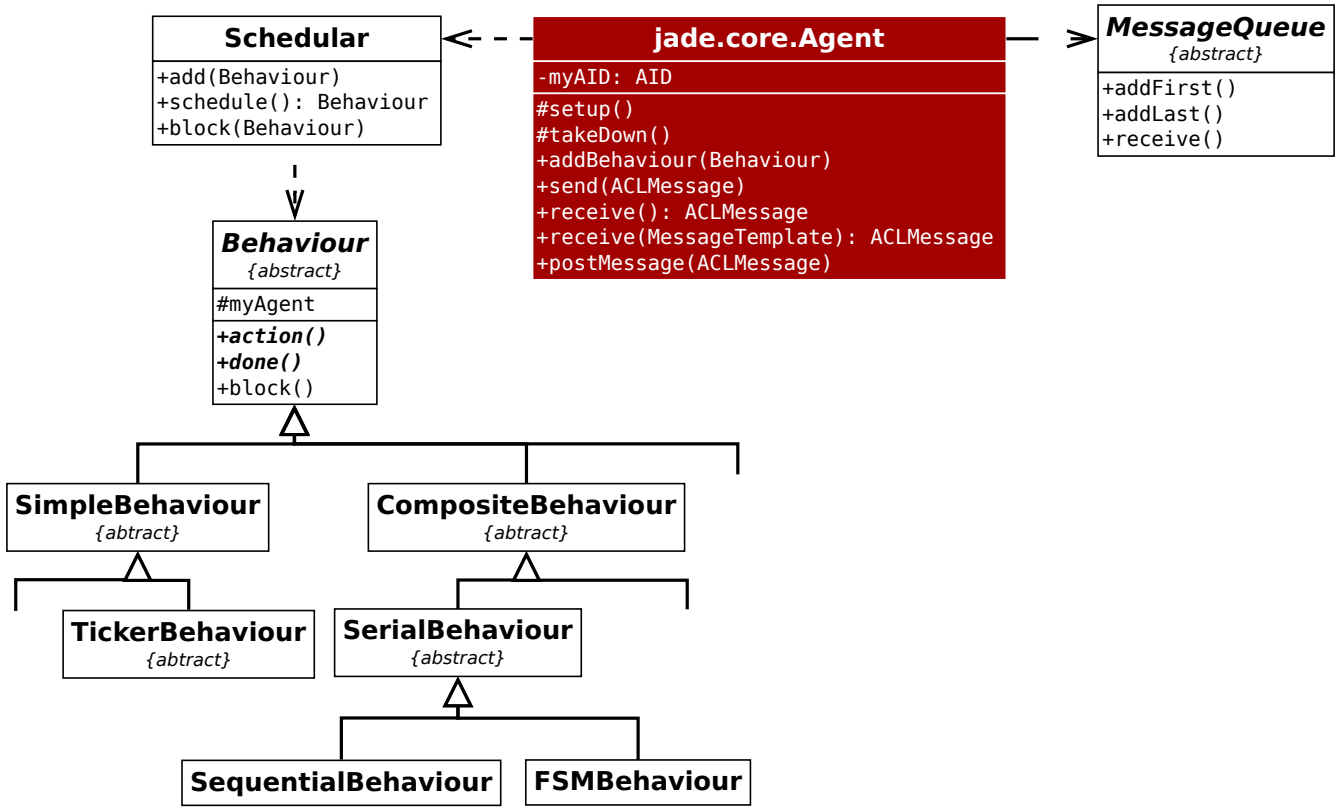


RMA des Haupt-Containers von Mars



Agenten in JADE

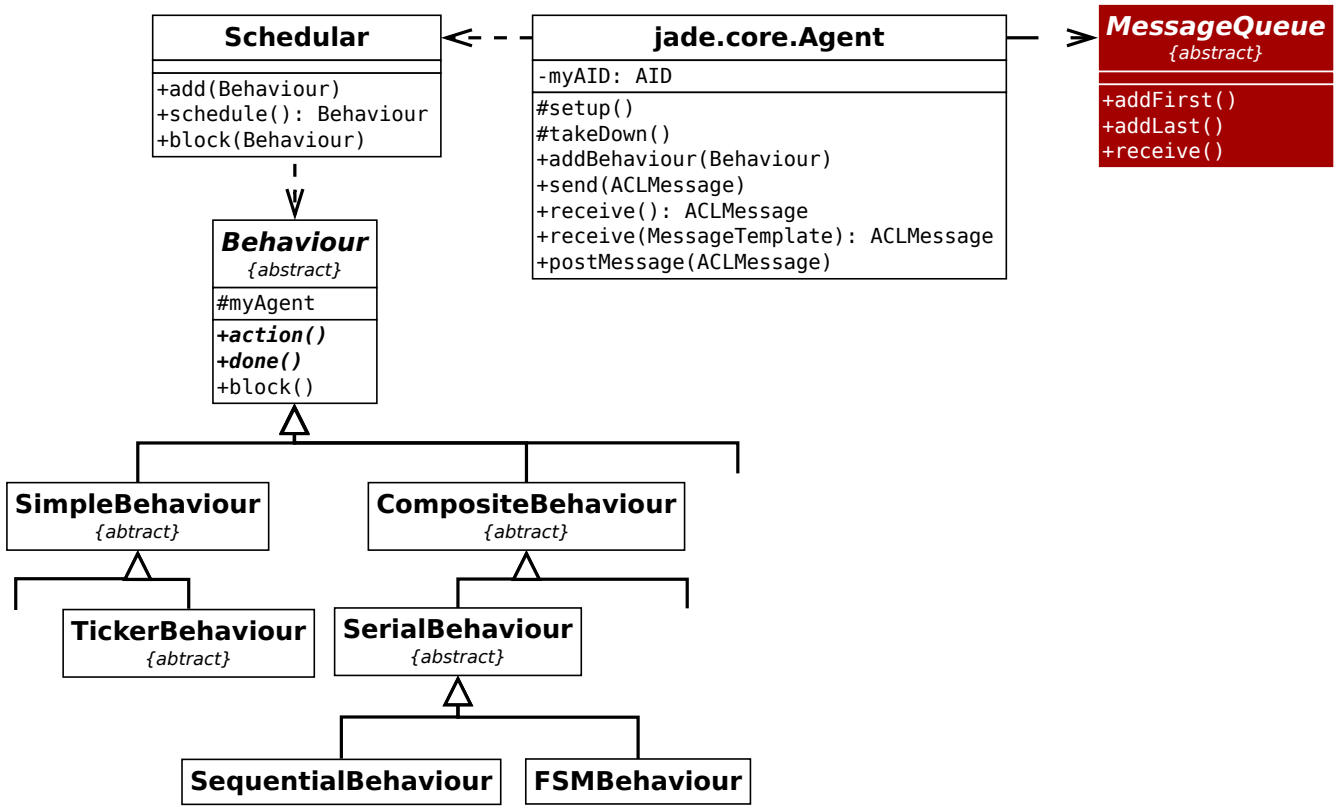
- Zur Realisierung eines Agenten, welcher auf der JADE Middleware lauffähig ist, muss von der Klasse **jade.core.Agent** geerbt werden.
- Um die Einordnung der folgenden Begriffe zu erleichtern dient dieses UML-Diagramm, welches nur einen **Ausschnitt** darstellt.



Kommunikation und Nachrichten in JADE

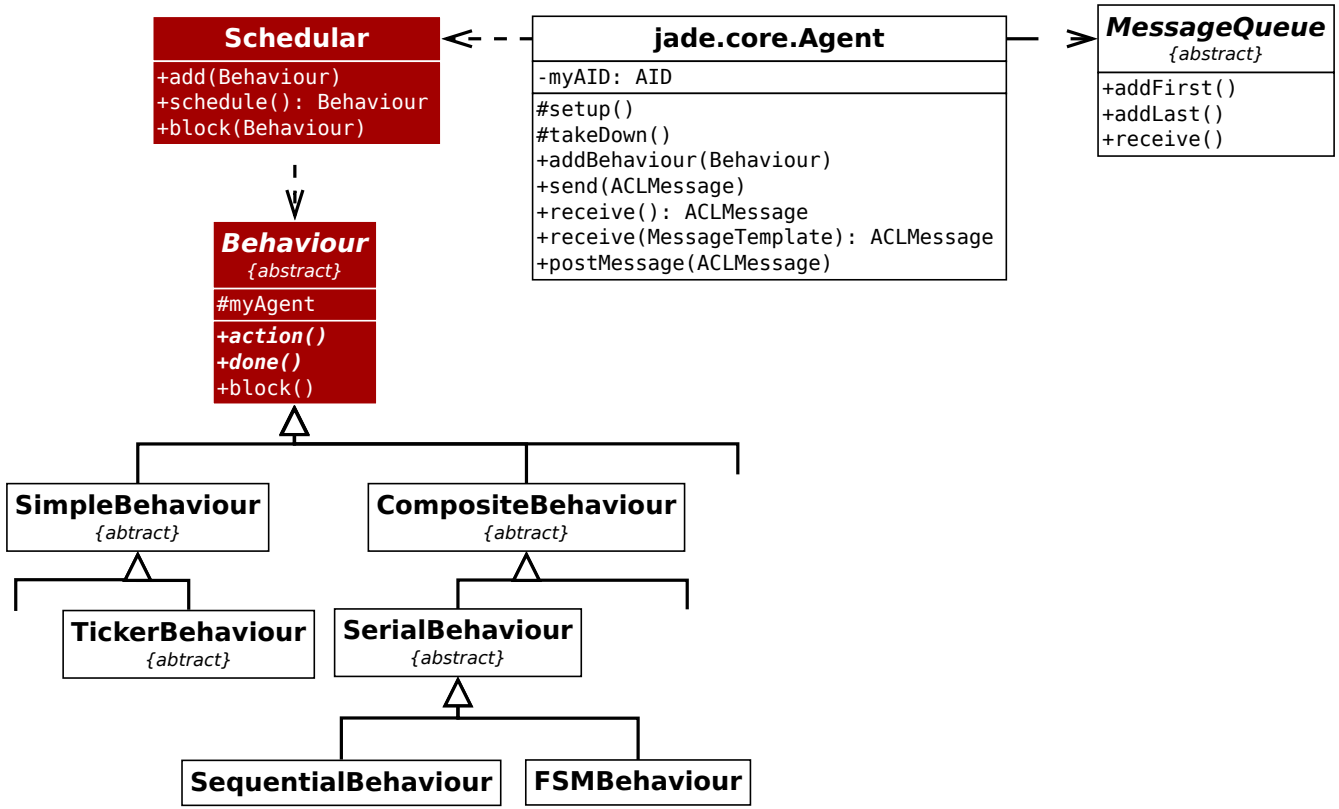
Nachrichten werden ....

- in einer Warteschlange verwaltet.
- Die Warteschlange wird von einem Objekt verwaltet, welches die MessageQueue-Klasse implementiert.

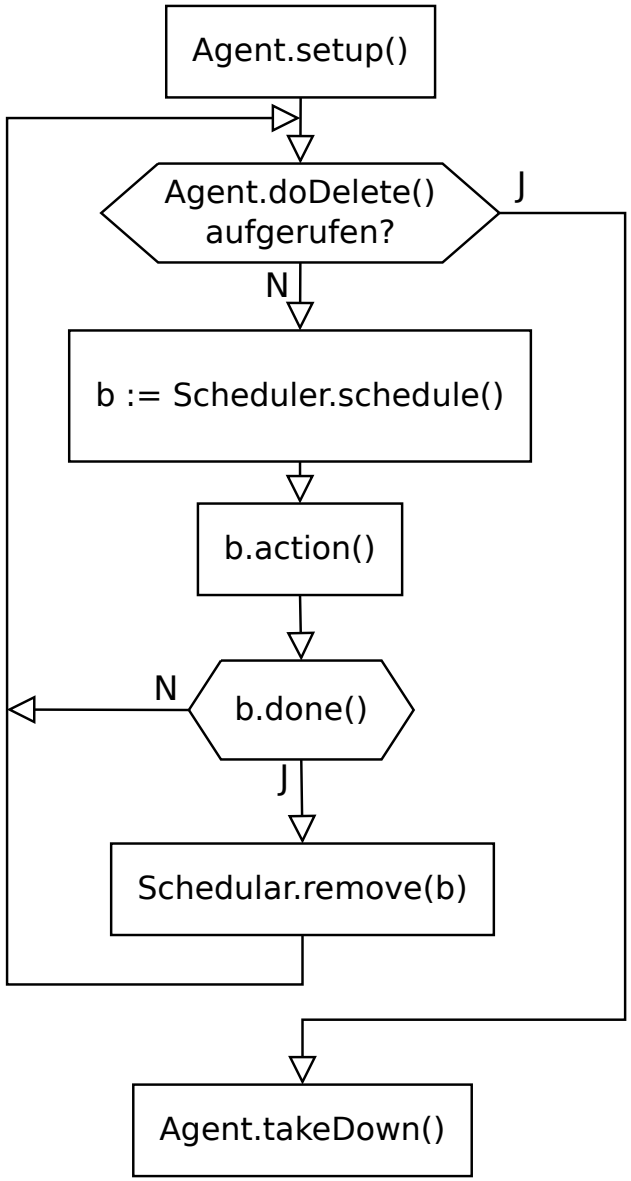


Das Verhalten eines JADE-Agenten

- Das Verhalten eines Agenten wird über beliebig viele Behaviour-Objekte gesteuert
- Ein Schedular koordiniert die Ausführung der Behaviour-Objekte
- Der Schedular arbeitet **kooperativ** und **nicht präemptiv**!

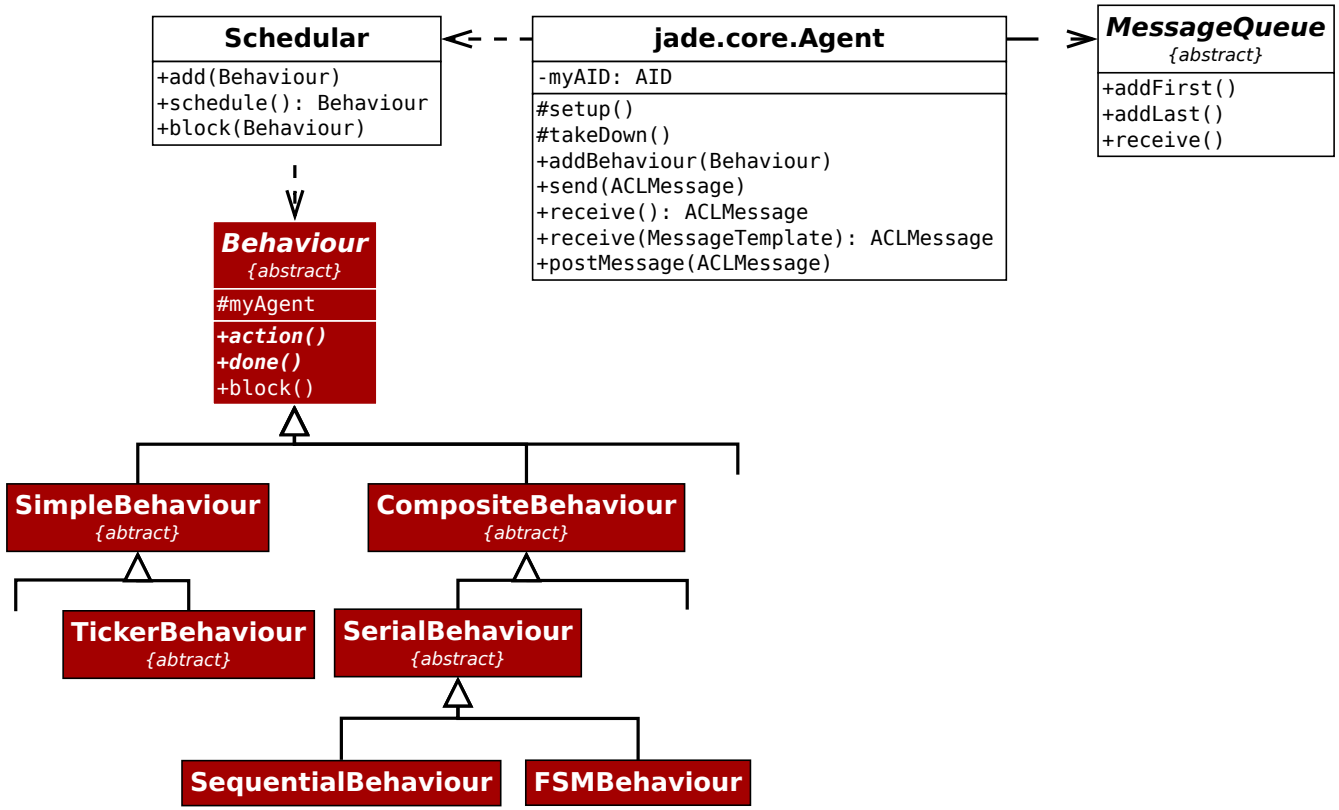


Ablauf eines Agenten mit Behaviour



Vorgefertigte Verhaltensklassen

- Um das Modellieren von Verhalten zu **vereinfachen**, wurden bereits einige **Anwendungsfälle generalisiert** und stehen dem Programmierer zur Verfügung.



Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Beispiel: Einfache Verhandlung

- Im Rahmen einer **einfachen Verhandlung** soll nun die konkrete Umsetzung von Agenten dargestellt werden.
- Als Beispiel wurde ein stark vereinfachtes Beispiel aus der E-Commerce Umgebung gewählt, welches sich am **'JADE programming for beginners'** Tutorial orientiert.
- Es gibt zwei Arten von Agenten:
  - **Händler**, die mehrere Bücher verkaufen
  - **Kunden**, die ein bestimmtes Buch kaufen möchten
- Ein Kunde wird versuchen das Buch zu dem **günstigsten Preis** von einem Händler zu erwerben. Dazu wird er das **Contract Net Protokoll** verwenden.

Der Händler-Agent

```
1  public class Haendler extends Agent {
2    protected void setup() {
3      processArguments();
4      if (registerDF()) {
5        addBehaviour(new SellOnContractNet(
6          this
7        , getContractNetMessageTemplate()));
8      } else {
9        doDelete();
10     }
11   }
12   protected void takeDown() {
13     deregisterDF();
14   }
15   // [...]
16 }
```

Der Händler-Agent - Behaviour

Der Händler reagiert über sein **SellOnContractNet**-Verhalten auf alle Nachrichten, die mit der ContractNet Auktion zusammenhängen. Damit handelt der Händler in diesem Beispiel rein **reaktiv**.

```
1 private class SellOnContractNet
2     extends ContractNetResponder {
3   // ...
4 }
```

Das Verhalten wird immer dann aufgerufen, wenn der Agent eine neue Nachricht erhält. Das Verhaltens-Objekt prüft daraufhin ob eine **relevante Nachricht** in der **MessageQueue** liegt.

Demo Händler-Agent

Live-Demo

- Überblick über Quelltext des Händler-Agent
  - Senden und Empfangen von Nachrichten
  - Registrieren von Verhaltensobjekten
  - Anmeldung am DF
- Starten von Händler-Agenten

Anmerkung

Alle Quellcode-Beispiele werden nach dem Vortrag zusammen mit der Ausarbeitung zur Verfügung gestellt.

Der Kunde-Agent

```
1  public class Kunde extends Agent {
2    protected void setup() {
3      if (!getBookArgument()) {
4        doDelete();
5      } else {
6        addBehaviour(new SearchForTraders());
7      }
8    }
9    protected void takeDown() {}
10   // [...]
11 }
```

Im Unterschied zum Händler nutzt dieser Agent:

- keine Registrierung mit dem DF, da keine Dienste bereitgestellt werden.
- Ein Behaviour-Objekt, welches weitere Behaviour-Objekte startet.

Der Kunde-Agent - Behaviour

```
1  private class SearchForTraders extends TickerBehaviour {
2    public SearchForTraders() {
3      super(Kunde.this, 5 * 1000);
4    }
5    protected void onTick() {
6      if (getTradersFromDF()) {
7        myAgent.addBehaviour(new BuyBookContractNet(
8          Kunde.this
9        , getCFPMessage()));
10     }
11   }
12   // [...]
13 }
```

- TickerBehaviour sorgt dafür das Behaviour regelmäßig von Schedular aufgerufen wird
- onTick-Methode ersetzt die action-Methode bei einer TickerBehaviour
- Wenn Händler vom DF geliefert wurden, dann wird das Kauf-Verhalten hinzugefügt

Demo Kunde-Agent und Auktionsablauf

Live-Demo

- Überblick über Quelltext des Kunden-Agent
- Starten von Kunden-Agenten
- Betrachten eines Auktionsablaufs mit dem Sniffer-Agenten

Verhaltensmuster für die Prüfung der MessageQueue

Damit ein Verhalten immer nur dann läuft, wenn eine neue Nachricht vorliegt, welche eventuell relevant sein könnte, wird folgendes **Verhaltensmuster** von den JADE Entwicklern **empfohlen**.
Ein **'Busy-Waiting'** wird damit **vermieden**.

```
1  public void action() {
2    MessageTemplate mt =
3      MessageTemplate
4        .MatchPerformative(ACLMessage.CFP);
5    ACLMessage msg = myAgent.receive(mt);
6    if (msg == null) {
7      block();
8    } else {
9      // weiterer Code
10   }
11 }
```

Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Ontologien in JADE

Über Ontologien werden Inhalte von Nachrichten **semantisch** validiert und erzeugt. Ontologien werden in JADE über Objekte abgebildet.
Eine JADE-Ontologie besteht aus folgenden Elementen:

- Konzept/Begriff (Concept)
- Agent-Aktion (AgentAction)
- Prädikat (Predicate)

Ontologien in JADE

Das jeweilige Ontologie-Element, wird durch eine Klasse repräsentiert, welche ein Element-Interface implementiert.

```
1 class Gebaude implements Concept {}
2 class Hotel extends Gebaude {}
3 class WohntIn implements Predicate {}
4 class HotelBuchen implements AgentAction {}
```

Die Ontologie-Elemente werden dann mit Hilfe der Ontologie-Klasse zu einer Ontologie verbunden.

```
1 class HotelGastOntologie extends BeanOntology {
2   public EmploymentOntology(String name)
3       throws BeanOntologyException {
4     add(Hotel.class); // [...]
5   }
6 }
```

Content-Sprachen in JADE

Um die durch Ontologien festgelegte **Semantik** in Nachrichten zu transportieren, wird ein **Syntax** gebraucht. JADE stellt zu diesem Zweck Content-Sprachen zur Verfügungen.
Eine Content-Language erfüllt dabei folgende Aufgaben:

- Umwandlung von Nachrichten-Inhalten in abstrakte Ontologie-Repräsentationen
- Umwandlung von abstrakten Ontologie-Repräsentationen in Nachrichten-Inhalte

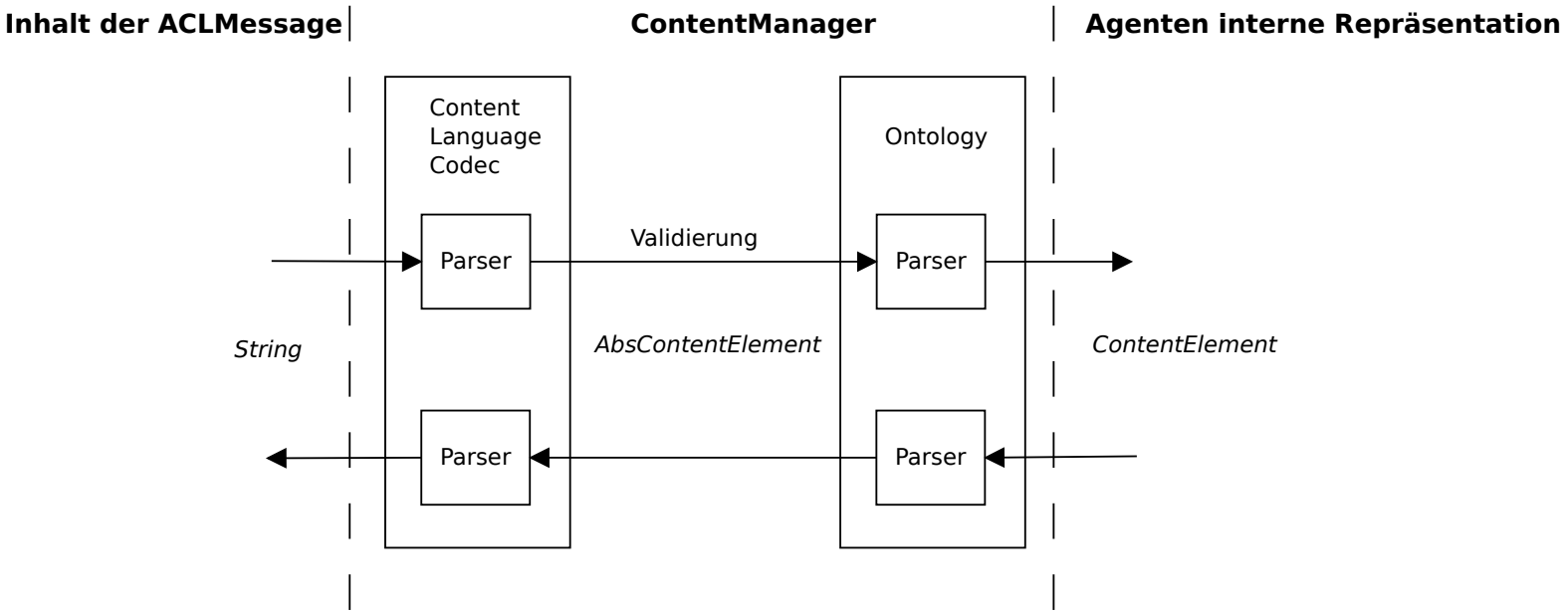
Innere Ontologien

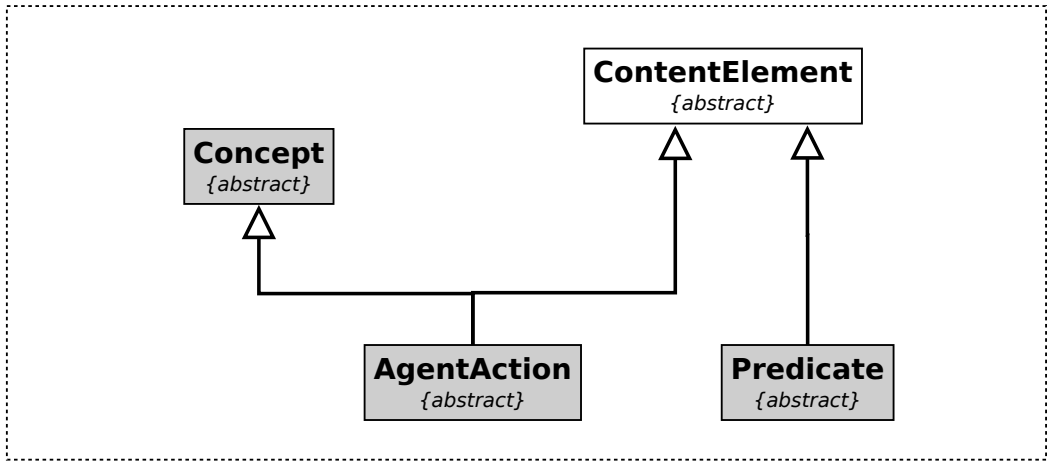
Optional kann eine Content-Sprache auch eine **innere Ontologie** bereitstellen, um eine Verknüpfung von Prädikaten und Aktionen zu **komplexen Termen** zu ermöglichen.
Der **SLCodec** stellt beispielsweise folgende Möglichkeiten zur Verfügung:

- Grundlegende logische Operatoren: AND, OR, NOT
- 'Identifying Referential Expression' (IRE): ANY, ALL, IOTA
- Angaben zum Zustand von Fakten: BELIEF, UNCERTAIN..
- Alternative-Aktionen, Sequenzen von Aktionen

SLCodec implementiert die **FIPA SL Content Language Specification** aus dem FIPA 00008 Standard (SL = Semantic Language).

Umwandlungspipeline mit dem ContentManager





Anwendung von Ontologien in JADE

```
1  // Ontologie und ContentLanguageCodec registrieren
2  Codec codec = new SLCodec();
3  getContentManager().registerLanguage(codec);
4  Ontology ontology = MyOntology().getInstance();
5  getContentManager().registerOntology(ontology);
6
7  // Nachricht versenden
8  ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
9  msg.setLanguage(codec.getName());
10 msg.setOntology(ontology.getName());
11 getContentManager().setContent(msg, myContentElement);
12
13 // Nachricht empfangen
14 ContentElement ce = getContentManager().extractContent(msg);
```

Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Weitere Technologien

- Jadex
  - Entwickelt an der **Universität Hamburg**
  - Implementiert das **Belief Desire Intention** (BDI) Modell (siehe Vortrag von Kay Hasselbach vom SS05).
  - Agenten können auf **JADE Plattform** betrieben werden (FIPA kompatibel)
  - Viele Beispielprogramm verfügbar
- NetLogo
  - Modellierung mit Agenten
  - Viele mitgelieferte Modelle (u.a. PageRank, Klimaerwärmung)
  - Gute Dokumentation

Übersicht

1. Einführung
   - Agenten
   - Multiagenten-Systeme
   - Kommunikation
   - Repräsentation von Wissen
2. Softwaretechnologie
   - Architektur
   - Das JADE Framework
   - Beispiel einer Verhandlung mit JADE
   - Ontologien und Content-Sprachen in JADE
   - Weitere Technologien
3. Einschätzung und Bewertung

Was sind die Schwächen von JADE?

- Die Code-Basis ist zwischen 2000 und 2002 entstanden.
  - Es wurde mit J2SE 1.3 gearbeitet
  - Keine Verwendung von Generics
  - Häufige upcasts von Object
- Es fehlen Sicherheitsfeatures
  - Authentifizierung von Agenten muss selbst implementiert werden
  - Verschlüsselung von Nachrichten-Inhalten zwar technisch möglich, jedoch nicht standardisiert
  - Es gibt ein Security- und Trusted Agents-Addon die von der JADE-Community entwickelt wurden und die Sicherheit verbessern sollen (wurden nicht evaluiert).
- Quellcode ist durch Integration der J2ME-Edition schwer zu lesen
  - An vielen Stellen bedingte Kompilierung notwendig

Was sind die Stärken von JADE?

- Das Framework ist stark skalierbar und lässt sich durch den Einsatz von Java auf vielen Plattformen einsetzen
- JADE kann als Agenten Plattform voll in eigene Produkte integriert werden (jade.Boot muss nicht genutzt werden)
- Durch eine Nutzung der FIPA-Standards ist eine **Kompatibilität** zu anderen Agenten-Plattformen sichergestellt
- Weiterentwicklung durch das **JADE Board** vorangetrieben, in dem sich u.a. **Telecom Italia** und **Motorola** befinden
  - Die letzte Version ist im April 2010 erschienen.
- Einige Tutorials und Beispiele helfen beim Einstieg in die Plattform
- Quellcode steht dank der Open-Source Lizenz komplett zur Verfügung

Bewertung

Wann ist es sinnvoll Agenten einzusetzen?

Tendenziell ist die Verwendung eines Multiangentensystems für Projekte mit größerem Umfang interessant, wenn:

- eine lose gekoppeltes verteiltes System realisiert werden soll
- viele unterschiedliche Organisationen beteiligt sind

Forschung

Agenten und Multiagentensysteme werden weiterhin erforscht. In Toronto fand im Mai 2010 die 9. internationale Konferenz der autonomen Agenten und Multiagentensystem (AAMAS) statt.

Vielen Dank

Vielen Dank für die Aufmerksamkeit!
Gibt es offene Fragen?