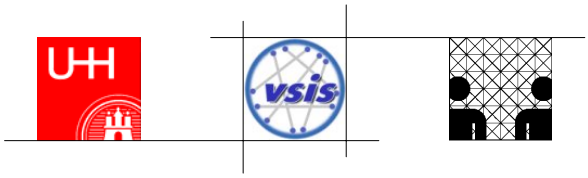


JADE Einführung

Gliederung

- Einführung in die Multi-Agenten Plattform JADE
  - Beschreibung der Eigenschaften
  - Beschreibung des Programmierparadigmas
  - Beschreibung der Tools

Die JADE Plattform

- Opensource-Projekt der Tilab aus Italien
  http://sharon.cselt.it/projects/jade/
- Firmenbeteiligungen über JADE Board
  - Telecom Italia
  - Motorola
  - Whitestein
  - France Telecom
  - Profactor
- Middleware Agentenplattform für FIPA-Standards

Was bedeuten FIPA-Standards

- FIPA (Foundation for Intelligent Physical Agents) ist ein internationales non-profit Standardisierungsgremium
  - seit 1996, 60 Mitglieder, FIPA97/98/2000
  - www.fipa.org
- Hauptziel: Interoperabilität für offene Agentensysteme
- Anwendungs- und Middlewarestandards
  - Anwendungsstandards: Systematisch analysierte Beispieldomänen mit Ontologie- und Servicebeschreibungen
  - Middlewarestandards: Plattformarchitektur, Agentenkommunikation, Agentenmanagement

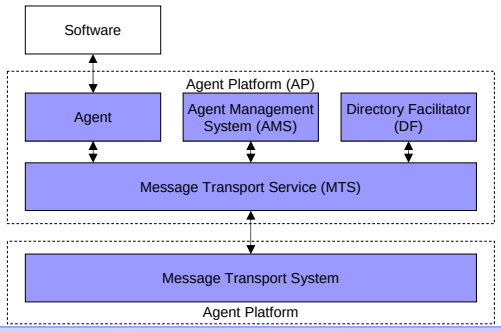
Die JADE Plattform

Bestandteile:

- Eine Laufzeitumgebung für Agenten
- Ein Klassenbibliothek zur Konstruktion von Agenten
- Graphische Werkzeuge zur Administration und zum Debugging

Die Laufzeitumgebung

- Setzt die FIPA-Plattformarchitektur um



JADE Laufzeiterweiterungen

- Container als Bestandteile einer Plattform
- Container können über das Netz verteilt werden
- Container erlauben Migration von Agenten
- Pro Plattform muss ein "Main-Container" vorhanden sein

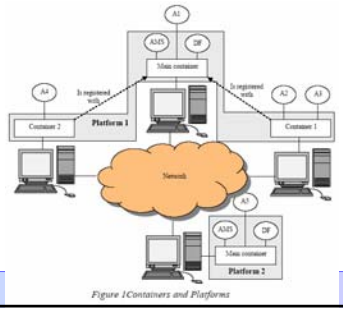


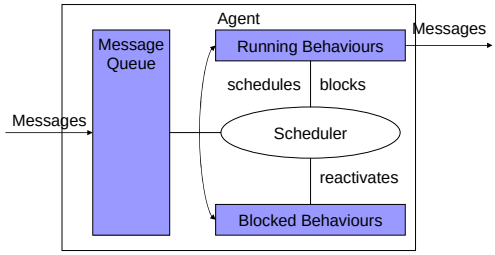
Figure 1Containers and Platforms

JADE Agenten Konzept (1)

- Zerlegung der Aufgaben in kleine Teileinheiten (Behaviours) = Taskmodell
- Diese können dem Agenten hinzugefügt werden
- Der Agenten arbeitet die Behaviours nach einem Schedulingverfahren ab
  - ***Jeder Agent arbeitet nur in einem Thread***

JADE Agenten Konzept (2)

Agent / Behaviours



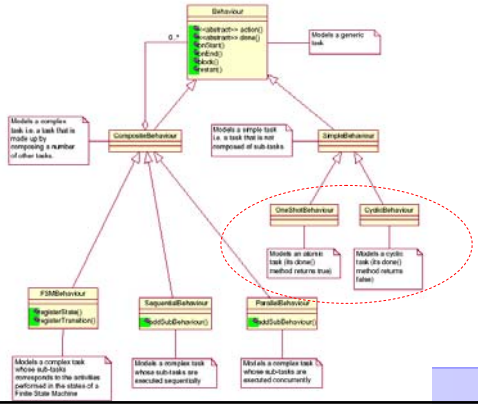
Schedulingeinzelheiten

Der Scheduler entscheidet welches Behaviour zur Ausführung gelangt
- Ruft dann die **action()** / **done()** Methoden

Arbeitet nicht-preemptiv / round-robin, d.h. die action-Methode wird immer ganz ausgeführt
- sauberes Behaviour Design nötig (keine langen Operationen / Endlosschleifen)
- > Zerlegung / Automaten

Es wird keine Zustandssicherung durchgeführt
- > Objektvariablen müssen Behaviourzustand sichern

Behaviours - Ein Überblick



Methoden eines Behaviours

Innerhalb eines Behaviours

- **action()** Enthält den Code für die zu erledigende Aufgabe.
- **done()** Darf ein Behaviour weiter geschedult werden
- **onStart()** Startup Code
- **onEnd()** Cleanup Code
- **block([timeout])** Blockiert ein Behaviour
  - resume wenn ACL Message ankommt, timeout entsteht oder die restart() Methode aufgerufen wird
- **restart()** Startet ein blockiertes Behaviour erneut
- **reset()** Startet ein blockiertes Behaviour von vorne

Methoden eines Agenten

- Konstruktorersatz **setup()**
- Behaviourhandling mittels add / removeBehaviour()
- Nachrichtenkommunikation mittels
  - **send(message)**
  - **receive([pattern])**
    - Durchsuche die Messagequeue
  - **blockingReceive([timeout])**
    - VORSICHT!!! Legt den gesamten Agenten schlafen, bis eine Nachricht eintrifft
- Cleanup über **takeDown()**

Agenten - Lifecycle

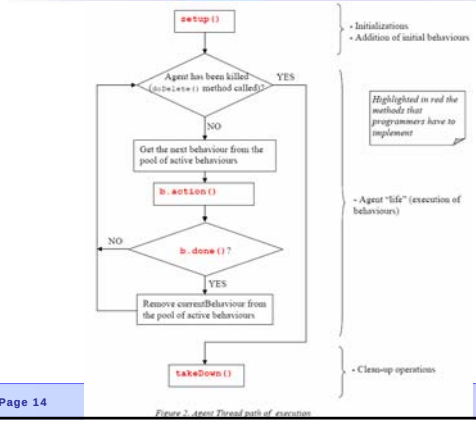


Figure 2. Agent Thread path of execution

Ein kleines Beispiel (1)

Erstellung eines PingBehaviors - 1

```
public class PingAgent extends Agent {
  class WaitPingAndReplyBehaviour extends CyclicBehaviour {

    public WaitPingAndReplyBehaviour(Agent a) {
      super(a);
    }

    public void action() {
      ACLMessage  msg = blockingReceive();
      if(msg != null){
        if(msg.getPerformative() == ACLMessage.NOT_UNDERSTOOD){
            //received a NOT-UNDERSTOOD message
            ...
      } else if(msg.getPerformative() == ACLMessage.QUERY_REF){
        //received a QUERY_REF with correct content.
      }
      else{
        //received a QUERY_REF with incorrect content.
      }
    } // end class WaitPingAndReplyBehaviour
```


Ein kleines Beispiel (2)

Erstellung eines PingBehaviors - 2

```
  protected void setup() {
    /** Registration with the DF */
    try {
      DFService.register(this,dfd);
    } catch(FIPAException e) {
      System.err.println(getLocalName() +
      " registration with DF unsucceeded. Reason: "+e.getMessage());
      doDelete();
    }

    WaitPingAndReplyBehaviour PingBehaviour = new WaitPingAndReplyBehaviour(this);
    this.addBehaviour(PingBehaviour);
  }
} //end class PingAgent
```


Agentenkommunikation in JADE

- Setzt die FIPA-Kommunikationsstandards um
- asynchone Nachrichten
- jeder Agent besitzt eine Nachrichten-mailbox

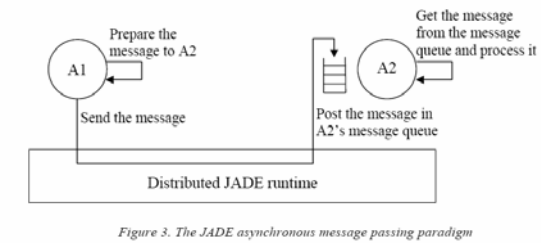


Figure 3. The JADE asynchronous message passing paradigm

Bestandteile einer ACL-Nachricht

In JADE implementiert in jade.lang.acl.ACLMessage

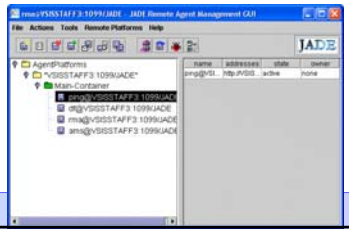
- Sender
- Liste von Empfängern
- Performativ (kommunikative Intention)
- Inhaltssprache (Syntax)
- Ontologie (Semantik der verwendeten Symbole)
- Inhalt
- weitere Felder zur Steuerung von Konversationen (conversation-id, reply-with, in-reply-to, reply-by)

```
ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
msg.addReceiver(new AID("Peter", AID.ISLOCALNAME));
msg.setLanguage("English");
msg.setOntology("Weather-forecast-ontology");
msg.setContent("Today it's raining");
send(msg);
```

Jade Tools (1) RMA

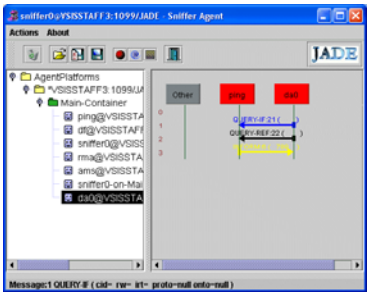
RMA (Remote Monitoring Agent)

- Kontrollzentrum
- Agent zur Darstellung und Admistration der Plattform, Container und Agenten
  - Agenten starten, beenden, anhalten, …



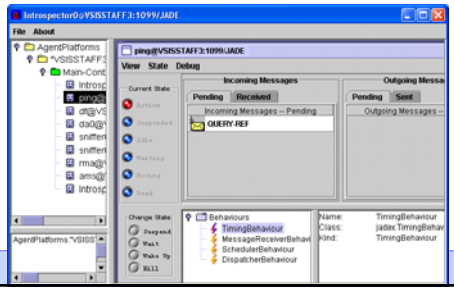
Jade Tools (2) Sniffer

Nachrichtenüberwachung



Jade Tools (3) Introspector

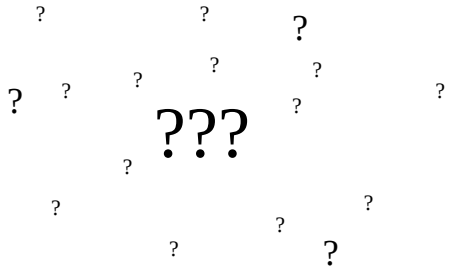
- Überwachen der internen Statusinformationen eines Agenten
- Debugging durch Einzelschritt/Zeitlupe

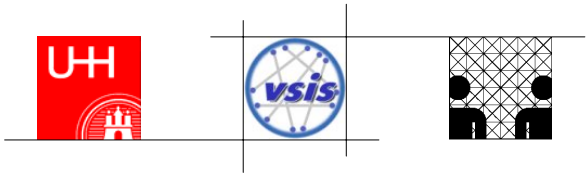


Jade Tools (4) DummyAgent

Zum einfachen Versenden von Nachrichten

Fragen





Ontologien
mit Protégé und Jade

Alexander Pokahr

Gliederung

- **Was ist eine Ontologie**
- Ontologien in JADE
- Ontologiekonstruktion mit Protégé
- Aufgaben

Was ist eine Ontologie

Eine Ontologie ist (aus unserer Sicht)
- Eine Dokumentation einer bestimmten Weltsicht.
- Eine Konzeptualisierung.
- Definiert eine Menge von Termen - das Vokabular.
- Definiert die Interpretation der Terme.
- Definiert die Struktur der Statements in dem Modell.

Was ist mit Ontologie (für uns) nicht gemeint?
- Die philosophische Interpretation: „Lehre vom Seienden“, ...

Definition (Gruber 1993)
- *An ontology is a **formal**, **explicit** specification of a **shared conceptualization***

Ontologien - Zweck

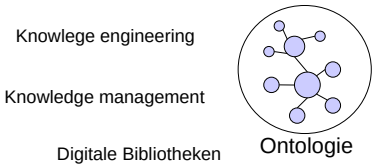
Das Fehlen eines gemeinsamen Verständnisses führt zu Kommunikationsproblemen:
- Menschen, Organisationen und Software Systeme müssen mit- und untereinander kommunizieren können

Unterschiedliche Modellierungsparadigmen, Sprachen und Softwaretools beschränken:
- Interoperabilität
- Knowledge sharing & Wiederverwendbarkeit

Wozu wollen wir Ontologien verwenden?
- Um Softwaresystemen das gleiche Verstehen einer Darstellung/eines Konzeptes zu ermöglichen.
- Um die Interoperabilität und die Wiederverwendbarkeit von Softwarekomponenten, Modellen und Spezifikationen zu verbessern.
- Um das Domänenwissen von dem Anwendungs-/Lösungswissen zu trennen.

Ontologien - Anwendungsfelder

Ontologien sind ein populäres Forschungsziel in verschiedenen Fachgebieten



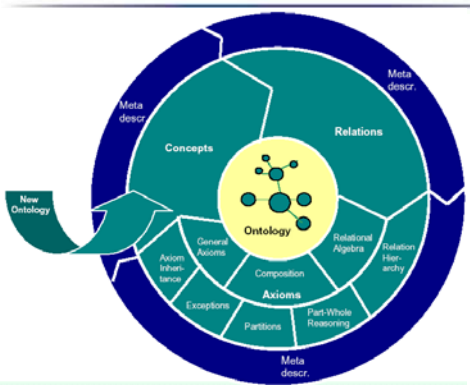
Ontologien - Typen von Ontologien

Domänenspezifische Ontologien
- Stellen Konzeptualisierungen und Gerüst für Wissensbasen bestimmter Domänen dar (z.B. Medizin, Öl-Produktion, Flugzeugbau usw.).
- Entsprechen dem Standard ISO 10303 - Standard of Product Data Modelling, Series 101-105. (Schiffsbau, Elektrotechnische Verkabelung und andere Bereiche)
- Besteht aus Objekt Modell und Axiomen

Task-Ontologien
- Beschreiben Konzepte des Vorgehens zu einer Aufgabenlösung.
- Sind für die Lösung einer bestimmten Aufgabe vorgesehen (z.B. propose-and-revise - Methode für Konfigurationsaufgaben).
- Stellt keine Struktur einer Wissensbasis dar!

Top-level Ontologien
- Allgemeine Ontologien als generische Grundlage für spezielle Ontologien
- Unabhängig von konkreten Domänen
- Definieren allgemeine Konzepte, die in mehreren Domänen vorkommen (z.B. Ereignisse, Zustände, Zeit, geometrische und topologisch Darstellungen)
- “Upper Ontology“ thema internationaler Standardisierung (z.B. IEEE SUO, P1600.1)

Ontologie - Bestandteile



Ontologien - Konstruktion (1)

Inspirative Methode
- Basiert auf den Antworten zur Frage, warum die Ontologie benötigt wird.
- Führt zu einer Ontologie, die eine individuelle Sicht einer Domäne enthält.

Induktive Methode
- Baut auf der Analyse von Szenarien aus der entsprechenden Domäne auf.
- Führt zu einer Ontologie, die für die meisten Szenarien in einer Domäne verwendet werden kann.

Deduktive Methode
- Baut auf allgemeinen Prinzipien für ein bestimmtes Szenario auf.
- Anpassung der Prinzipien zur Integration in die Ontologie.

Ontologien - Konstruktion (2)

Synthetische Methode
- Es werden mehrere „kleine“ Ontologien erstellt, so dass keine eine Teilmenge einer anderen ist (Überschneidungen sind möglich).
- Aus den „kleinen“ Ontologien wird eine „große“ Ontologie erstellt, in welche alle vorhandenen Konzepte einfließen.

Kollaborative Methode
- Die Sichtweise von allen Entwicklern der Ontologie werden berücksichtigt.
- Die Ontologie wird schrittweise verfeinert, wobei z.B. in jedem Schritt die Änderung eines Entwicklers durch die restlichen Entwickler angepasst wird.
- Führt meistens zu einer Ontologie mit größerer Akzeptanz.

Gliederung

- Was ist eine Ontologie
- **Ontologien in JADE**
- Ontologiekonstruktion mit Protégé
- Aufgaben

Ontologien in FIPA

Nachrichteninhalt in FIPA definiert durch
- Content (eigentlicher Nachrichteninhalt, z.B. String)
- Encoding (verwendeter Zeichensatz, z.B. UniCode)
- Language (Syntax)
- Ontologie (Semantik)

→ Ontologien zur Definition der Bedeutung der in ACL Nachrichten verwendeten Begriffe

Außerdem: Inhaltssprache (content language) zur Festlegung der Syntax (z.B. XML, FIPA-SL)

JADE Content Reference Model

Legt fest was Inhalt einer Nachricht sein kann
- **Concept**: Beschreibt einen realen oder abstrakten Gegenstand mit Eigenschaften (z.B. Person mit Name und Alter, Datum mit Tag, Monat und Jahr)
- **AgentAction**: Beschreibt eine Handlung eines Agenten (z.B. Kaufen eines Gegenstands)
- **Predicate**: Beschreiben logische Aussagen über Gegenstände (z.B. ist-Vater-von)
- Weitere Elemente die nicht Teil einer Ontologie sind (Variablen, Listen, Primitive)

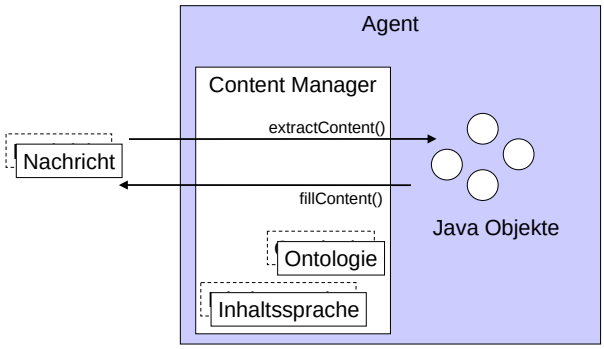
JADE Basic Ontology

Standardontologie mit häufig benötigten Konzepten nach FIPA
- Package jade.content.onto.basic
- **Action**: Behälter für AgentActions
- **Done**: Prädikat, das Beendigung einer Action anzeigt
- **Result**: Prädikat, das Ergebnis einer Action angibt
- **Equals**: Prädikat für Gleichheit zweier Elemente
- **True/FalseProposition**: Konstante Prädikate für “wahr” und “falsch”

Weitere JADE Ontologien

- FIPAManagementOntology: Standardontologie nach FIPA zur Kommunikation mit AMS und DF
- JADEManagementOntology: Für JADE-spezifische Managementfunktionalität (z.B. bezgl. Container)
- ExceptionOntology: Ontologie die Java Exceptions in Fehler-Prädikate abbildet
- MobilityOntology: Ontologie für mobile Agenten (enthält u.a. Location Konzept und Move Aktion)

JADE Content Manager



Benutzung des Content Managers

Registrieren der Sprache und Ontologie

```
myAgent.getContentManager().registerLanguage(new
  jade.content.lang.sl.SLCodec(0));
myAgent.getContentManager().registerOntology(MyOntolo
  gy.getInstance());
```

Setzen des Nachrichteninhalts

```
myAgent.getContentManager().fillContent(msg, object);
```

Auslesen des Nachrichteninhalts

```
object= yAgent.getContentManager().extractContent(msg);
```


Gliederung

- Was ist eine Ontologie
- Ontologien in JADE
- **Ontologiekonstruktion mit Protégé**
- Aufgaben

Ontologien - Werkzeuge

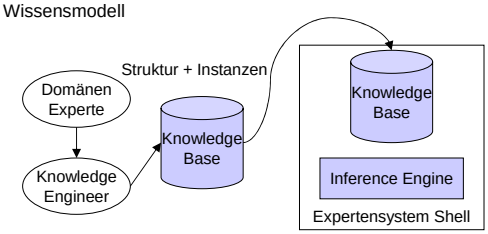
Existierende Werkzeuge
- Ontolingua, Chimæra, OILEd, OntoEdit, *Protégé 2000*

Funktionalität der Werkzeuge
- Unterstützung beim Design von Ontologien: Modellierung, Implementierung, Test.
- Unterstützung bei der Pflege von Ontologien: Zusammenführung, Extraktion, Versionsverwaltung
- Schnittstellen zu anderen Systemen: Agentensysteme, Expertensysteme, logische Programmiersprachen.
  Export in XML oder in relationale Datenbanken mit SQL-Schnittstelle.

Vorteile
- Schnelleres Design von Ontologien möglich.

Ontologie Design - Protegé (1)

- Protégé Projektstart 1987!, Uni Stanford, Oncocin + Opal
  - http://protege.stanford.edu
- Zunächst für Expertensystementwicklung
- Heute domänenunabhängiges Werkzeug zur Erstellung von Ontologien
- Open Knowledge Base Connectivity Protocol (OKBC) basiertes Wissensmodell



Ontologie Design - Protégé (2)

Vorgehen:

- Default Ontologie laden und unter anderem Namen speichern
- Konzepte, Prädikate und Aktionen erstellen
- Mit dem Converter Plug-in Java Klassen aus der Ontology generieren

Verwendung in Jade:

- Ontologiekonzepte als Objekte verwenden
- ContentManager zum Setzen und Auslesen des Nachrichteninhalts