Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Entwurf einer generischen Agentenarchitektur für MABS

AW-2 / HAW-Hamburg

Agenda

- Ziel & Vision
  - Ziel
  - AW-1
  - Probleme
- Programm
  - Dynamische Zielverwaltung
  - Generische Interaktionen für MABS
  - EBDI – Emotionen für BDI-Agenten
- Ausblick
  - Aktueller Stand / Architekturüberblick
  - Weiteres Vorgehen

Ziel

- Generische Architektur für Multi-Agenten basierte Simulationen
  - soll offen für alle Anwendungsdomänen sein
    - konfigurierbare Grundlage
    - keine Restriktionen
    - erweiterbar
- Einsatz in MARS

AW-1

- gefundene Architekturen entweder sehr wage oder zu spezifisch
  - GAA: *Generic Agent Architecture*
  - GEAMAS: *Generic Architecture for Multi Agent Simulations*
- Architekturen immer nur "Mittel zum Zweck", aber nicht im Fokus
  - Eigenbau notwendig
  - [GAA]: Vererbungshierarchien nutzbringend

Probleme

- Generische Abstraktion von Sensoren und Aktionen
- Ausführung der Agenten
- Entwurf eines allgemeinen Agentenprogramms
- Wie interagieren Agenten miteinander?
- Emotionale Agenten: Repräsentation + Umsetzung von Gefühlen
  - Basisstrukturen integrieren, die der Domänenentwickler nutzen kann
- GOAP: Dynamische Repriorisierung, Ausführung & Planung

→ *Dieser Vortrag soll Lösungsansätze zu drei Problemen aufzeigen*

Dynamische Agenten

Adaptive (Neu-)Planung bestehender Ziele:

- erhöht Robustheit des Agenten
  - flexible Reaktion auf sich ändernde Umweltbedingungen
- mehr Entwurfsmöglichkeiten für den Entwickler

Situationen:

- Ressourcenkonflikte
- Synergieeffekte
- ungültiger Kontext
- kein Plan verfügbar
- Prioritätenänderung
- Befehl von anderem Agenten

Dynamische Agenten

„Suspending and Resuming Tasks in BDI Agents“

- von Thangarajah, Harland, Morley und Smith, AUS/USA, AAMAS '08

Generischer Mechanismus zum Einbau in BDI-Architekturen:

- soll die Aussetzung und Wiederaufnahme von *Tasks* ermöglichen
  - *Task*: Oberbegriff für Ziele, damit verbundene Pläne (und Aktionen)
  - transparent (falls nicht benötigt)
  - parametrisierbar andernfalls
  - Standardmechanismus integriert, bei Bedarf erweiterbar
    - Meta-Aktionen, der Planungsphase vorgeschaltet

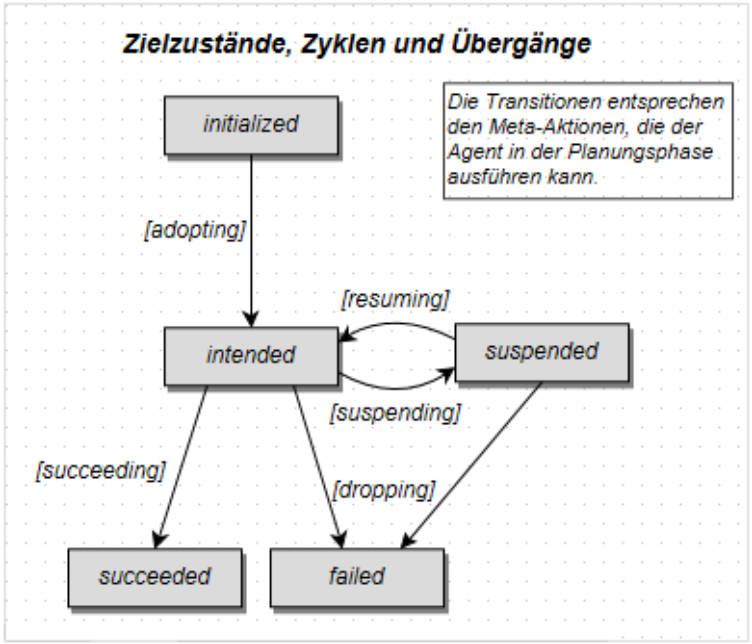
Metadaten eines Tasks

Zielverfolgung wird durch Bedingungen gesteuert:

- Vor-, In- und Nachbedingung

Ziele besitzen Zustände:

- *initialized, intended, suspended, succeeded, failed*
- Prinzip: endlicher Automat



Einflußfaktoren:

- je höher die Priorität eines Zieles, desto gravierender ist eine Absetzung
- Priorität wird dynamisch ermittelt aus:
  - Nutzwert, Deadline, Kosten, Abhängigkeiten (intern/extern), Erfolgschance, ...
- Bedingungen für Pläne analog, Aktionen atomar

Tasks: Unterbrechung

- Anhängen einer Wiederaufnahme-Bedingung
  - eventuell zusätzliche Metadaten (Restlaufzeit, …) beifügen
- *suspend()*-Methode aufrufen
- Task und Unter-Tasks rekursiv auf "*suspended*" setzen

Besonderheiten:

- Ziele:
  - wenn ein Plan mit dem Ziel verbunden ist, diesen absetzen
- Pläne:
  - falls noch nicht ausgeführt, Plan verwerfen
  - alle anstehenden Schritte auf inaktiv setzen

Tasks: Fortführung

Ziel:

- Voraussetzung: Wiederaufnahme-Bedingung wird wahr
- *resume()*-Methode aufrufen
- als aktiv markieren
- zugehörigen Plan fortführen
  - falls nicht vorhanden, normale Planerstellung

Plan:

- In-Bedingung wahr?: nächste Schritte aktivieren
  - sonst Plan abbrechen und Alternativplan suchen

Dynamische Tasks: Fazit

Konzept:

- einfach und geradlinig
- erhöht Potential und Autonomie

Implementation:

- "Operational Semantics" in CAN gegeben

Integration:

- Anreicherung der Tasks um Metadaten
- Mechanismus wird der Planungsphase vorgeschaltet

Interaktionen

Standard-MAS:

- Organisation von Agenten (Gruppierung)
- Nachrichtenaustausch
  - Protokollentwurf: Endliche Automaten, Petri-Netze, ...

MABS:

"Interaktion ist der Schlüssel zu emergentem Verhalten"

- umfaßt "physische" Aktionen
- Umgebung und/oder mehrere Agenten simultan involviert

IODA

"Interaction-Oriented Design of Agents"

- von Kubera, Mathieu und Picault, Université Lille (FR), 2011-2013
- Entwurfsmethodik, formales Modell und Umsetzungskonzept speziell für MABS

Ziele:

- klare Trennung von Modell und Implementation wahren
- Domänenspezialisten möglichst lange am Entwicklungsprozeß beteiligen
- unabhängige Interaktionsspezifikation und Agentenvererbung
  - generisches Framework, wiederverwendbare Komponenten

IODA: Repräsentation

Homogene Struktur für (normale) Aktionen und Interaktionen:

- reflexive (normale) Aktionen: "degenerate interactions"
- Interaktion ist formell ein semantischer Block
  - *I = (name, card, labels, primitives, precondition, trigger, actions)*

Polymorphismus von Interaktionen:

- abstrakte und unabhängige Definition mit generischer Semantik
  - operieren auf abstrakten Primitiven
  - interaktionsfähige Agenten müssen konkrete Implementationen liefern

Rollen:

- *Source*: Urheber, initiiert eine Interaktion
- *Target*: (Menge von) Zielagenten
- die *Kardinalität* gibt dieses Verhältnis an : $card(I) = (|agt_S|, |agt_T|)$
- Rolle legt benötigte Primitive fest

IODA: Ausführung

Interaktionen:

- Initiation erfolgt aus der Planungsphase heraus
- Teilnahme ist verpflichtend

Agent:

- Standardphasen
- Planungsphase beliebig
  - reaktiv, kognitiv, hybrid

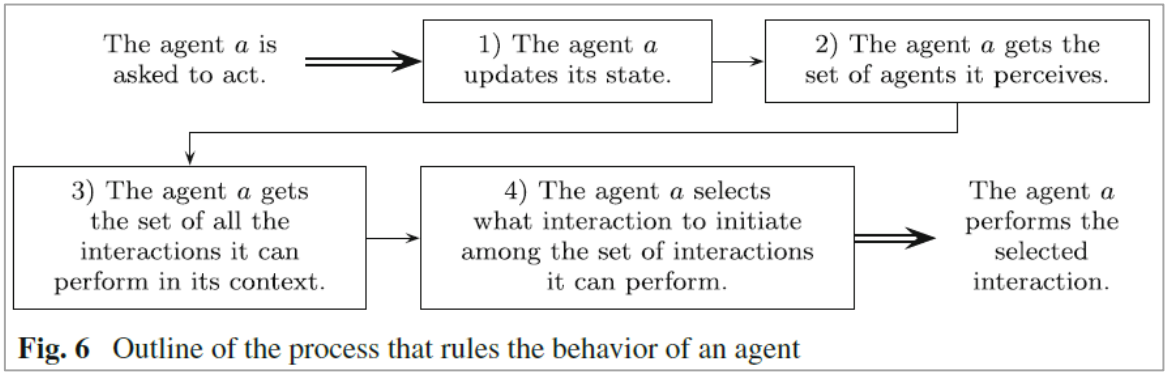


Fig. 6 Outline of the process that rules the behavior of an agent

[1]

Simulationsumgebung [IODA/JEDI]:

- sequentielle Ausführung der Agenten (Reihenfolge zufällig)
- zeitdiskret

IODA: Entwurfsphase (gekürzt)

1. Vorannahmen treffen:
   - Repräsentation von Zeit, Umwelt und Wahrnehmung
2. Erhebung der Interaktionen und Agenten:
   - Identifikation von Urheber- und Zielagenten
   - Festlegen der Interaktionsdistanz
   - (Reflexive) Aktionen ermitteln
   - Darstellung in einer Matrix (→)

Table 2 Raw Interaction Matrix of an ecosystem simulation, where wolves, sheep, goats and grass agents evolve

Source \ Target	∅	Grass	Sheep	Goat	Wolf
Grass	(Grow)				
Sheep	(Move)	(Eat, d = 0)	(Procreate, d = 1)		
Goat	(Move)	(Eat, d = 0)		(Procreate, d = 1)	
Wolf	(Move)		(Eat, d = 3)	(Eat, d = 3)	(Procreate, d = 1)

The element *(Eat, d = 0)* at the intersection of the row starting with *Sheep* and the column starting with *Grass* is read “Sheep agents are able to initiate the Eat interaction with a target Grass agent at a maximal distance of 0 from the source”

[1]

3. Spezifizierung:
   - Auslöser, Vorbedingung und Aktionshandlung festlegen
   - Kompromiß (generisch ↔ konkret) treffen, benötigte Primitive ableiten
   - Agententypen und deren Auswahlprozeß bestimmen

IODA: Fazit

Konzept:

- für Simulationen optimiert
- ausgelegt auf generische Entwicklung

Implementation:

- beschränkt sich auf Interaktionsdesign
- Synchronisation bei Parallelausführung von Agenten?

Integration:

- Erweiterung der bisherigen Aktionen
- zusätzliche Abstraktion über Interaktionsprimitive

Emotionale Agenten

Rationales vs. emotionales Verhalten:

- die meisten Agenten werden nutzenbasiert entworfen
- aber: Emotionen üben starken Einfluß auf das Verhalten aus
  - insbesondere im Simulationskontext wichtig!

"EBDI: An Architecture for Emotional Agents"

- von Jiang, Vidal und Huhns, USA, AAMAS '07
- generische BDI-Architektur um Emotionen angereichert
  - Filterprinzip, Emotionen überlagern Wahrnehmung

EBDI: Ermittlung von Wissen & Emotionen

Drei "*Belief Revision Functions*":

- Wahrnehmung: *brf-see:* $Env \rightarrow B_p$
- Kommunikation: *brf-msg:* $Cont \rightarrow B_m$
- Nachdenken: *brf-in :* $E \times I \times (B \cup B_p \cup B_m) \rightarrow B$

Zwei "*Emotion Update Functions*":

- ermitteln die primären und sekundären Emotionen
  - Primär: Unmittelbares, reflexhaftes Empfinden: *euf1:* $E \times I \times (B_p \cup B_m) \rightarrow E$
  - Sekundär: Folgeemotionen, z.B. nach Überdenken: *euf2:* $E \times I \times B \rightarrow$ E

Filter-Funktion zur Beeinflussung der Absichten:

- *filter:* $E \times B \times D \times I \rightarrow I$

Mit diesen Funktionen wird der BDI-Interpreter erweitert!

EBDI: Basisstruktur des emotionalen Agenten

EBDI-MAIN-LOOP

```
1  E ← E_0;                ▷ E_0 are initial emotions
2  B ← B_0;                ▷ B_0 are initial beliefs
3  I ← I_0;                ▷ I_0 are initial intentions
4  while true
5        do B_p ← brf-see(Env);
6           B_m ← brf-msg(Cont);
7           E ← euf1(E, I, B_p ∪ B_m);
8           B ← brf-in(E, I, B ∪ B_p ∪ B_m);
9           D ← options(B, I);
10          I ← filter(E, B, D, I);
11          E' ← E
12          E ← euf2(E, I, B);
13          if time permits and E ≠ E'
14             then B ← brf-in(E, I, B);
15                  D ← options(B, I);
16                  I ← filter(E, B, D, I);
17          π ← plan(I, Ac);
18          execute(π)
```

[2]

Figure 1: Pseudo-code of an EBDI agent's main loop.

Weitere BDI-Standardfunktionen:

- options: $B \times I \rightarrow D$
- plan: $I \times Ac \rightarrow \pi$
  - $\pi = \langle a_1, \ldots, a_n \rangle$
  - $a_i \in Ac$

EBDI: Fazit

Konzept:

- generisch, nicht einschränkend
- (laut Autoren) für alle gängigen Emotionskonzepte geeignet

Implementation:

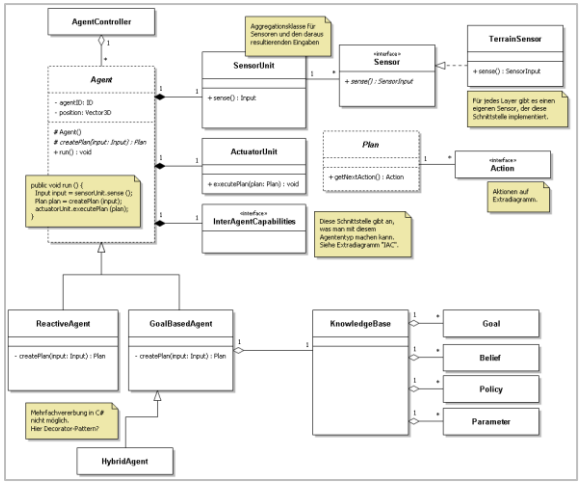
- Definition und Repräsentation nicht spezifiziert
  - eigene, abstrakte Darstellungsform finden

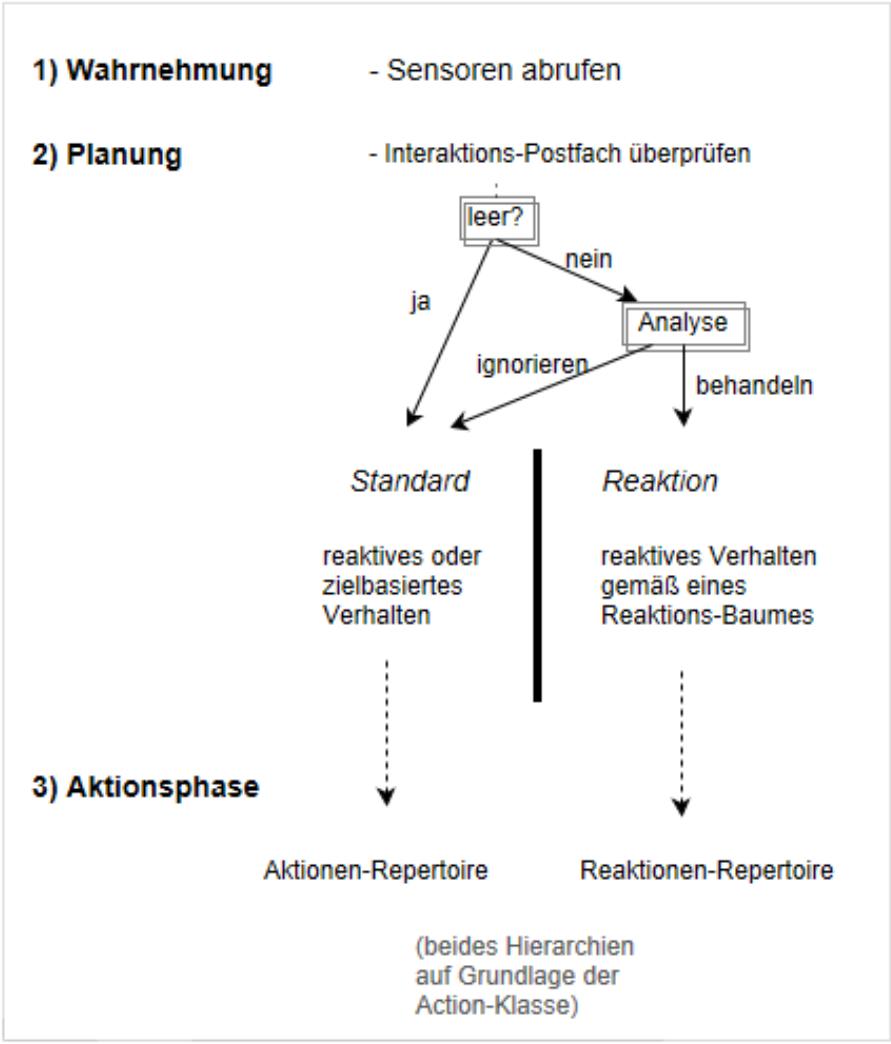
Integration:

- keine Ergänzung!
  - muß in der kompletten Planung verankert werden

Aktueller Stand

- Architekturentwurf, noch keine Umsetzung!
- Modularisierung gemäß Agentenprinzip
- Sequentielles Agentenprogramm





Planung & Ausblick

Semester 2:

- Grundarchitektur bauen
  - lauffähiges Gerüst innerhalb des MARS-Systems
- einfache, reaktive Agenten
  - keine Wissensbasis, primitive Entscheidungsbäume

Semester 3 (, 4):

- GOAP-Agenten (mit E-BDI-Funktionalität)
- dynamische Zielauswahl
- Schnittstelle für Konfigurationsmöglichkeiten (M-DSL)

Optional:

- Komposition von Bibliotheken
  - Hierarchie von (Inter-)Aktionen, Agentenprogrammen, Sensoren, ...

Quellen

Haupt-Paper:

- Suspending and resuming Tasks in BDI Agents
  – http://dl.acm.org/citation.cfm?id=1402383.1402443
- [1] IODA - Interaction-Oriented Design of Agents
  – http://hal.inria.fr/hal-00825534
- [2] EBDI - An Architecture for Emotional Agents
  – http://dl.acm.org/citation.cfm?id=1329125.1329139

Sonstige:

- Open Protocol Design for Complex Interactions in MAS
  – http://dl.acm.org/citation.cfm?id=544862.544866