Agent-Oriented Programming

John-Jules Ch. Meyer
Intelligent Systems Group
Universiteit Utrecht

Almende course

(Cognitive) Agents

- Software / hardware entities that display a certain degree of *autonomy / taking initiative*, are proactive/goal-directed
- Mostly described in terms of having 'mental states' ('*strong*' notion of agency) *'cognitive' agents*
- Display informational and motivational attitudes

Agent metaphor

- From an *engineering perspective*, the agent's *metaphor* (i.e. using agent concepts metaphorically) helps to *design* and *construct* complicated (distributed) systems!!
- Example: (multi) robotic systems
  - *Cognitive robotics*

Applications of MAS

- Autonomous Robots
- Softbots (Software agents)
  - Industrial applications
  - Commercial applications
  - Medical applications
  - Entertainment

(Jennings & Wooldridge 1998)

(Multi) Robot Systems

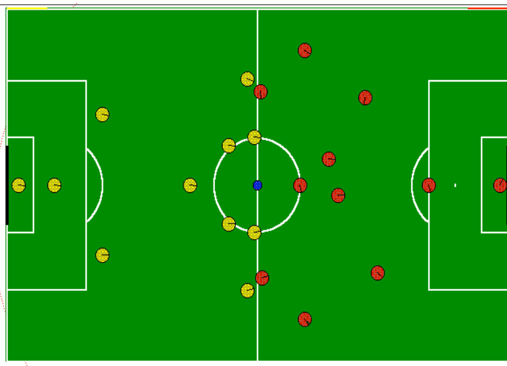
- Traffic & transport
- Space robots
- Rescue robots
- Robot soccer
- Robot companions
- …

NASA explorer robots

Autonomous vehicles

Autonomous Unmanned Aerial Vehicle - Linköping

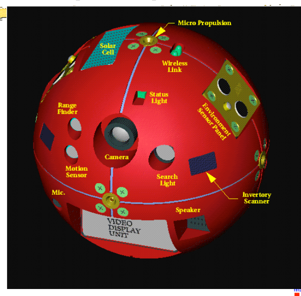
Robot soccer



Intelligent Robot Companions

- Companions of human users
  - Personal assistants
    - PSA's for ISS (NASA)
    - Intelligent user interface (Philips)
  - Playmates / Mentors / Assistants
    - Kitchen help for elderly ('boon companion')

NASA's Personal Satellite Assistant (PSA)



NASA's Personal Satellite Assistant (PSA)

Applications of MAS
(Jennings & Wooldridge 1998)

- Industrial Applications
  - Process control
  - Manufacturing
  - Air-traffic control
- Commercial Applications
  - Information management
  - E-commerce
  - Business process management

Applications of MAS
(Jennings & Wooldridge 1998)

- Medical Applications
  - Patient monitoring
  - Health care
- Entertainment
  - Games: virtual characters
  - Interactive theatre and cinema
    - *Believable* agents

Applications of (M)AS

- 'multi' aspect
  - Distributed systems
    - In which agents become processing nodes.
- individual agents
  - Personal software assistants
    - In which agents play the role of proactive assistants to users working with some application

(Wooldridge 2002)

Applications (Wooldridge 2002)

- Agents for
  - Workflow and Business Management
  - Distributed Sensing
  - Information Retrieval and Management
    - E.g. Web agents
  - Electronic Commerce
    - Comparison shopping agents
    - Auction bots

Applications (Wooldridge 2002)

- Human-Computer Interfaces
- Virtual Environments
- Social Simulation
- Industrial Systems management
- Spacecraft Control
- Air-Traffic Control
- …

Agent Applications (AAMAS'06 CfP)

- agent standardizations in industry and commerce
- agents and ambient intelligence
- agents and novel computing paradigms (e.g. autonomic, grid, P2P, ubiquitous computing)
- agents, web services and semantic web
- agent-based simulation and modeling
- agent-mediated electronic commerce and trading agents
- applications of autonomous agents and multi-agent systems
- artificial social systems
- auctions and electronic markets

Agent Applications (AAMAS'06 CfP)

- autonomous robots and robot teams
- constraint processing in agent systems
- conversation and dialog in agent systems
- cooperative distributed problem solving in agent systems
- humanoid and sociable robots
- information agents, brokering and matchmaking
- mobile agents
- negotiation and conflict handling in agent systems
- perception, action and planning in agents
- synthetic, embodied, emotional and believable agents
- task and resource allocation in agent systems

Characteristics of agent applications

What kind of applications is particularly suited for an agent-based solution?

- Hard to say in general, but includes:
- Applications where (e.g. logistical / planning) tasks may be distributed in such as way that subtasks may be (preferably / must) performed by autonomous entities (agents).
  - E.g. in situations where it is virtually impossible to perform the task centrally, due to:
    - complexity of this task
    - dynamics (changes) of the environment
    - isolated working area

Characteristics of agent applications

- Also applications where there is a *natural notion of a 'cognitive' agent*
  - Gaming
    - virtual characters need to behave in a natural / believable way and have human-like features
  - (Multi-)robotic applications
    - each robot controlled by an agent, and therefore can display autonomous but, for instance, also co-operative behaviour.
  - E-commerce / e-business and Web-applications
    - agents acting on behalf of a user

Ideas behind agents

- Stemming from philosophy
  - Practical reasoning, reasoning about actions
  - Characterization of rational decision-making
  - Balancing *desires* and *beliefs*
  - Interplay between *beliefs*, *desires*, *intentions* (Bratman)
  - Intentional stance (Dennett)

Practical Reasoning

- (Bona fide) practical syllogism

Exercise would be good for me.
Jogging is exercise.

Therefore, jogging would be good for me.

- 'Just' deductive reasoning

Practical Reasoning

- More interesting practical syllogism

Would that I exercise.
Jogging is exercise.

Therefore, I shall go jogging

- No deduction, rather specification of selection of action / decision of the agent

Dennett's intentional stance

- The *intentional stance* is the strategy of interpreting the behaviour of an entity by treating it *as if it were* a *rational agent* that governed its *choice* of *action* by a *consideration* of its *beliefs* and *desires*
  - Anthropomorphic instance of the *design (functionality) stance*, contra the *physical stance*
  - Instrumental / operational use of beliefs and desires of *human beings*: no causally active inner states of people, just calculational devices

Bratman : the role of intentions

- *Rational* behavior needs, besides *beliefs* and *desires*, also *intentions*
- Two justifications for this:
  - (Resource-bounded)agents need to settle on some desire(s) and commit themselves
  - Co-ordination of future actions after commitment(s)

Bratman

- Intentions, *unlike mere desires*, play the following functional roles:
  - Intentions normally pose *problems* for the agent; the agent needs to determine a way to achieve them *focus on solving concrete problems*
  - Intentions provide a "screen of admissibility" for *adopting other intentions*
  - Agents "track" the success of their attempts to achieve their intentions -- may give rise to *replanning*

Agents in Artificial Intelligence

- Logics for specifying intelligent/rational agents inspired by Bratman's philosophy:
  - BDI logic
  - Cohen & Levesque
  - KARO logic
- BDI model/architecture (Rao & Georgeff)

Specifying Intelligent Agents: Modal logic

- philosophical logic
- a formal treatment of intensional notions
- various 'flavours':
  - epistemic / doxastic
  - temporal / dynamic (action logic)
  - deontic
  - combinations (BDI, KARO)

Cohen & Levesque

- Achievement goals
  A-GOAL i = GOAL i (LATER) BEL i ¬
- No deferral forever assumption
  ⊨ ¬(GOAL i (LATER))
  - Agents eventually drop all achievement goals!

Cohen & Levesque

- Persistent goals
  P-GOAL i =
  GOAL i (LATER)
  BEL i ¬
  [BEFORE(BEL i BEL i ¬)
  ¬GOAL i (LATER)]

Cohen & Levesque

- Intention ('intend-to-do')
- INTEND$_1$ i = P-GOAL i [DONE i
  (BEL i (HAPPENS))?;]

KARO - Van Linder et al.

- motivational attitudes
  - PossIntend(,) <commit>Com()
  - PossIntend(,) ¬Auncommit
  - Com() <uncommit> ¬Com()
  - Com() BCom()
  - Com($_1$; $_2$) Com($_1$) B[$_1$]Com($_2$)
  - Com() ¬Can(,true)
    Can(uncommit, ¬Com())

Rao & Georgeff : BDI theory

- *"Rational agent* possesses mental attitudes of *beliefs, desires and intentions,* representing the *information, motivational, and deliberative states of an agent*, respectively"
- "These mental attitudes determine the system's behaviour and are critical for achieving *adequate* or *optimal* performance when deliberation is subject to *resource bounds*" --- *computational* perspective!

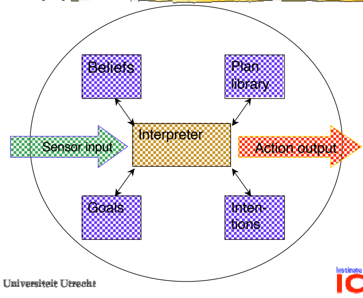
Rao & Georgeff's BDI Logic

- Commitment strategies in BDI logic
  - INTEND() inevitable(¬INTEND())
    "no infinite deferral"
  - INTEND(inevitable)
    inevitable(INTEND(inevitable) U BEL())
    "blindly committed agent"

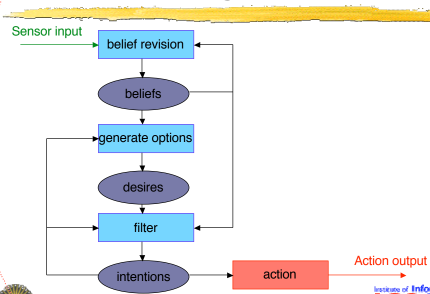
Rao & Georgeff's BDI Logic

- INTEND(inevitable□□)
  inevitable(INTEND(inevitable□□) U
  (BEL(□) ¬BEL(optional□□)))
  "single-minded committed agent"
- INTEND(inevitable□□)
  inevitable(INTEND(inevitable□□) U
  (BEL(□) ¬GOAL(optional□□)))
  "open minded committed agent"

agents: BDI Architecture



BDI architecture: 'deliberation cycle'



Agent-oriented programming (Y. Shoham)

- AOP = programming *mental states*
  - meaning of an AOP program is a *'mental state transformer'*
  - BDI agent programming languages
    - Agent-0 Shoham 1993
    - AgentSpeak(L) Rao 1996, Bordini et al.
    - 3APL, 2APL Hindriks et al. 1999, Dastani et al. 2007
    - AF-APL Collier et al. 2004
- Representation of mental attitudes
  - which mental attitudes? how represented? semantics?
    - AgentSpeak beliefs, intentions, events
    - 3APL '99 beliefs, plans

The language 3APL (Hindriks)

- An Abstract Agent Programming Language
- Attempt to get a 'true' agent language using 'mental' (BDI-like) concepts
  - So agent concepts used in *implementation*
- Supplied with *formal semantics*
- Mixture of *imperative* and *logic programming* aspects

Mental attitudes in 3APL

BDI theory	3APL (new version)
Beliefs	Beliefs
Desires	Goals (declarative)
Intentions	Plans (procedural)

3APL agent

- a complex mental state incorporating
  - beliefs about the agent's environment
  - plans, describing actions to achieve the goals
  - goals, representing the states of affairs to be achieved
- set of mechanisms working on mental state
  - to execute plans (controlling the environment)
  - for decision-making or practical reasoning (plan revision, plan generation)
- a set of capabilities, i.e. basic actions

3APL program

- a set of capabilities: basic actions:
  - e.g. `gripper_up, pickup, move_left, move_right, sense`
- an initial belief base: propositions (PROLOG):
  - e.g. `block_on_table`
- a set of initial plans: imperative programs:
  - e.g. `gripper_up ; pickup`

3APL programs (ctd)

- a set of *plan generation rules*: guarded clauses of the form □□ □|□, where
  - □ is a goal,
  - □ is a guard and
  - □ is a (generated) plan
- `on(x,y) □ clear(x) and clear(y) | put(x,y)`

3APL program (ctd)

- a set of *plan revision rules*: guarded clauses of the form □□ □|□', where
  - □ is a plan,
  - □ is a guard and
  - □' is a (revised) plan
- e.g. `gripper_up;pickup □ no_block | find_block;gripper_up;pickup`
- If the guard is implied by the agent's belief base the rule becomes applicable and *may* be applied.

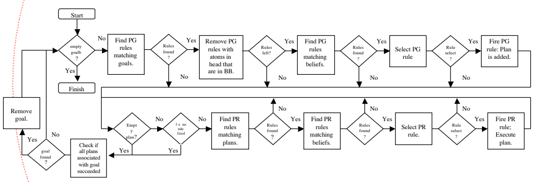
3APL control architecture

- the control architecture implements the deliberation or (Sense)-Update-Act cycle:
  - Rule application phase (plan generation / updating);
  - Execution Phase (belief updating by plan execution)

Deliberation Cycle

1. Find Plan Generation Rules that Match Goals
2. Remove Plan Generation Rules with atoms in head that exist in Belief Base
3. Find Plan Generation (PG) Rules that Match Beliefs
4. Select a Plan Generation (PG) Rule to Apply
5. Apply the Plan Generation (PG) Rule, thus adding a plan to the planbase
6. Find Plan Revision (PR) Rules that Match Plans
7. Find Plan Revision (PR) Rules that Match Beliefs
8. Select a Plan Revision (PR) Rule to Apply to a Plan
9. Apply the Plan Revision (PR) Rule to the Plan
10. Find Plans To Execute
11. Select a Plan To Execute
12. Execute the (first basic action of the) Plan

Deliberation Cycle



Example: an agent wanting to go to an event

- Belief: I live in Almere
- Goal: I want to go to the Almende course in Rotterdam
- How can I program an agent planning my travel?
- Practical reasoning:

  I want to go to event x in city y.
  I live in city z.

  I take the train from z to y;
  and next the bus in y.

Example (ctd)

- I can have rules to revise plans when things go wrong:

  I have a plan to take the train from
  z to y and next the bus in y.
  However, there are no trains today.

  I change my plan: I take the car to drive
  from z to y, and next I park the car in y.
- These reasoning rules can directly be programmed in 3APL.

Example 3APL agent

- Initial mental state
  - Beliefs: live(Almere), …
  - Goal: event(Almende'07,Rotterdam)
  - Plan: (empty)
- Rules:
  - Plan generation:
    - event(x,y) □ live(z) | take_train(z,y); take_bus(y)
  - Plan revision:
    - take_train(z,y); take_bus(y)
      □ no_trains | take_car(z,y); park_car(y)

Example 3APL agent

- Initial mental state
  - Beliefs: live(Almere), …
  - Goal: event(Almende'07,Rotterdam)
  - Plan: (empty)
- Rules:
  - Plan generation:
    - event(Almende'07,Rotterdam) □ live(Almere) | take_train(Almere,Rotterdam); take_bus(Rotterdam)
  - Plan revision:
    - take_train(Almere,Rotterdam); take_bus(Rotterdam)
      □ no_trains | take_car(Almere,Rotterdam); park_car(Rotterdam)

Making it more practical: 2APL (Dastani et al., AAMAS 2007)

- A Practical Agent Programming Language
  - So actually 2(AP)L ;-)
- Extend language with more practical features
  - PR rules applicable *only in event of plan failure*
  - *Non-interleavable plans* ('bracketed sections')
  - Explicit recursion by *abstract actions* (instead of abuse of PR rules!)
  - *Goal tests* as well as belief tests in plans (var.instant.)
  - Explicit *adopt / drop goal actions* for goal mngment
  - *External events* (interface with JAVA)
  - Built on top of JADE platform (many tools for free!)

Typical multi-agent issues

- Cooperation / competition
  - Communication & coordination
  - Cooperative distributed problem solving
  - Multi-agent planning & synchronisation
  - Trust & reliability
  - Game theory & decision theory
  - Negotiation & argumentation
  - Normative systems, institutions, agent societies
  - Mechanism design
    - Design of protocols governing multiagent interactions s.t. *desirable* properties hold

Multi-Agent 3APL: communication

- *Communication* between agents, using techniques from *communicating sequential processes (CSP)*
- Communication primitives
  - Speech acts
- Typical (FIPA/KQML-like) construct:
  - *send*(agent, performative, content)

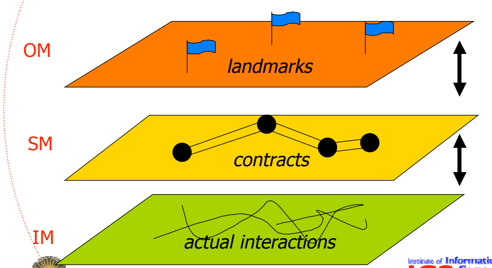
AO Methodology (Dastani, Hulstijn, Dignum, Meyer)

- Need for separate methods for AOP
  - AOP is not simply a variant/instance of OOP
  - Needed: relation between behavior of *complex MASs* and that of the *constituent individual* agents
    - *Norms*
    - Institutions
    - Protocols
    - ...

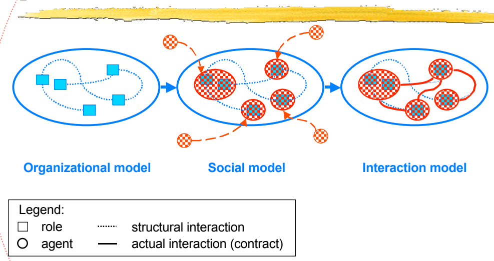
OperA (V. Dignum et al.)

- Organizational Model
  - represents *organizational aims and requirements*
  - *roles, interaction structures, scene scripts, norms*
- Social Model
  - represents *agreements concerning participation of individual agents ('hiring agents for playing roles')*
- Interaction Model
  - represents *agreements concerning interaction between the agents themselves*

OperA: 3-tiers specification



OperA



Consequence of OperA

- Separation between the individual (agent) and collective (society) level
  - For the construction of individual agents you can use what you want, e.g. *BDI model*!!
  - Link between the individual and organisation via interaction structure, roles, norms and *contracts*!

From analysis via design to implementation

- *Roles* from analysis → *agent types* in 3APL
- Agent type:
  - specification of *deliberation process* +
  - set of roles (characterized in terms of *beliefs*, *goals*, *plans*, *capabilities*, *messages*, *PR/PG/GR rules*)
- *Norms* may be implemented in various ways:
  - as *goals*
  - in *social or interaction structure*
    - obligations, protocols
  - in *environment*
    - norm enforcement

Applications

- Some ongoing agent-related projects
  - STW project "Distributed Model-Based Diagnosis and Repair" with TU Delft, NLR and U. Maastricht
  - BSIK/ICIS project "Adaptive Support Systems" with TNO
  - GATE project on virtual characters in video games (with TNO)
  - GATE project on "Explainable AI" (with TNO Soesterberg)
  - EU/ITEA project "Boon Companion" on companion robots (with DECIS, Groningen, Philips)
  - NWO project on "Coordination in Agent Societies" (with CWI)

Boon Companion Project

- aim: constructing a companion robot for elderly people, especially for help in the kitchen
- task UU:
  - (practical) reasoning
  - (natural) dialogue
- Test platform: Philips **iCat**

Philips iCat

Virtual characters in video games

- Aim: adding 'intelligence' to video game characters
- Issues:
  - Believable and natural behaviour
  - Challenges in modelling cognition:
    - Moods
    - Sensing
    - Communication
    - Group dynamics and social behaviour, roles
    - **Intention recognition / behaviour prediction *('mental state abduction')***

Conclusion

- Engaged in many agent-oriented activities:
  - Logic of agency / agents
  - Agent-oriented programming
    - Development of programming language 3APL
    - Agent-oriented software engineering
  - Applications
    - Ontologies for heterogeneous MAS
    - Multi-agent expert systems (with EB)
    - Airport Traffic Planning (with NLR, TUD and UM)
    - Adaptive Support Systems (with TNO)
    - Virtual Characters in Games (with Gaming group and TNO)
    - Intelligent companions (with DECIS, Philips)
    - …

Software & More Information

- http://www.cs.uu.nl/3apl/ and
- http://www.cs.uu.nl/2apl/

Thank you for your attention!