Agent-Oriented Programming

John-Jules Ch. Meyer
Intelligent Systems Group
Universiteit Utrecht

(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 (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

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

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
  \[ \text{A-GOAL } i \equiv \text{GOAL } i \land \text{BEL } i \land \neg \neg \]

- No deferral forever assumption
  \[ \equiv \neg \neg \neg \text{GOAL } i \land \text{LATER } j \]

- Agents eventually drop all achievement goals!

KARO - Van Linder et al.

- Motivational attitudes
  \[ \text{PossIntend} i, j \equiv \text{P-GOAL } i \land \text{DONE } i \land \text{BEL } i \land \text{HAPPENS } j \land \neg \]

- Commitment strategies in BDI logic
  \[ \text{INTEND}(\text{inevitable}) \equiv \neg \neg \neg \text{INTEND}(\text{inevitable}) \]

  - "no infinite deferral"

  \[ \text{INTEND}(\text{inevitable}) \equiv \text{inevitable}(\text{INTEND}(\text{inevitable}) \lor \text{BEL}()) \]

  - "blindly committed agent"
Rao & Georgeff’s BDI Logic

1. INTEND(inevitable) \( \dot{\Box} \)
2. inevitable(INTEND(inevitable) \( \dot{\Box} \)) U (BEL(\( \dot{\Box} \)) \( \dot{\Box} \) BEL(optional) \( \dot{\Box} \))

"single-minded committed agent"

1. INTEND(inevitable) \( \dot{\Box} \)
2. inevitable(INTEND(inevitable) \( \dot{\Box} \)) U (BEL(\( \dot{\Box} \)) \( \dot{\Box} \) GOAL(optional) \( \dot{\Box} \))

"open minded committed agent"

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
  - AgentSpeak
  - Bordini et al.
  - AgentSpeak(L)
  - Rao 1996
  - Collier et al.
  - Hindriks et al. 1999
  - Dastani et al. 2007
- Representation of mental attitudes
  - which mental attitudes? how represented? semantics?
  - AgentSpeak
  - 3APL
  - 2APL
  - Collier et al.
  - Hindriks et al.

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
  - Beliefs
  - Desires
  - Intentions

- 3APL (new version)
  - Beliefs
  - Goals (declarative)
  - 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
- A 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 \( \Box \ C \land \Phi \), where
  - \( \Box \) is a goal,
  - \( \Phi \) is a guard and
  - \( C \) is a (generated) plan
  - Must satisfy the belief base, and each clause will become applicable if the guard becomes true.

3APL program (ctd)

- A set of plan revision rules: guarded clauses of the form \( \Box \ C \land \Phi \), where
  - \( \Box \) is a plan,
  - \( \Phi \) is a guard and
  - \( C \) is a (revised) plan
  - E.g., gripper_up; pickup \( \land \) no_block \( \land \) 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
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 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(x,y) & take_bus(y)
  - Plan revision:
    - take_train(x,y) & take_bus(y)

- Example 3APL agent

  - 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 (ctd)

- Initial mental state
- Beliefs: live(Almere), …
- Goal: event(Almende'07,Rotterdam)
- Plan: (empty)

- Rules:
  - Plan generation:
    - event(x,y) & live(z) & take_train(x,y) & take_bus(y)
  - Plan revision:
    - take_train(x,y) & take_bus(y)

- 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(x,y) & take_bus(y)
    - Plan revision:
      - take_train(x,y) & take_bus(y)

- Making it more practical: 2APL
  - (Dastani et al., AAMAS 2007)
  - A Practical Agent Programming Language
  - So actually 2(APL) :-)
  - Extend language with more practical features
  - PR rules applicable only in event of plan failure
  - Non-interleave 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 mgmt
  - 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
- Trust & reliability
- Game theory & decision theory
- Negotiation & argumentation
- Declarative 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, scenes, 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**

Legend:
- role → structural interaction
- agent → actual interaction (contract)
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/ICS project “Adaptive Support Systems” with TNO
  - GATE project on virtual characters in video games (with TNO)
  - EU/ITEA project “Explainable AI” (with TNO Soesterberg)
  - ELITEA 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

Virtual characters in video games

- Aim: adding ‘intelligence’ to video game characters
- Issues:
  - Believable and natural behaviour
  - Challenges in modeling 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/
- http://www.cs.uu.nl/2apl/

Thank you for your attention!