Software Agents

Tony White
arpwhite@scs.carleton.ca
Overview

• Motivation and definitions
  – Why do we need agents?
  – What is an agent?

• Agent architectures
  – technologies, issues, advantages, disadvantages

• Collaboration
  – blackboard, KQML, etc.

• Examples
  – e-commerce, network management
  – enabling technologies
Motivations

• Why do we need agents?
  – Increasingly networked, temporary connectivity increasing (wireless).
  – Data overload (e-mail, web pages, fax, …).
  – Greater exchange of digital information
  – Increasingly dependent upon electronic sources of information.
  – Desire to be ‘better informed’.
Tools

• Inadequacy of current tools
  – Browsers are user driven, Pull technology marginally better.
  – ‘Friendly’ software becoming more difficult to use (e.g. MS Word!)
  – WWW too polluted for casual browsing, intelligent search tools required; even search engines beginning to fail us!
• Coverage, web pages exploiting indexing algorithms of engines, broken links.
Solution!

• Need software solution (agents) that can act in our place:
  – can interact with (say) Internet data sources
  – can process e-mail, voice, fax and other electronic message sources
  – can communicate with other agents
  – can accurately represent our needs and preferences in the networked information environment
  – can negotiate
And the solution is... **Agents**

- So, what is a software agent? *No generally agreed definition.* Has characteristics:
  - Something that acts on behalf of another
  - Is sociable, capable of meaningful interaction with other agents (and humans)
  - Can make decisions on our behalf
  - Is capable of adapting to changing environments and learning from user interaction
  - Is mobile
A Basic Definition

“Intelligent software agents are defined as being a software program that can perform specific tasks for a user and possessing a degree of intelligence that permits it to perform parts of its tasks autonomously and to interact with its environment in a useful manner.”

*From Intelligent Software Agents*

*Brenner, Zarnekow and Wittig.*
Potential agent rewards

• In the Internet:
  – **efficiency**: agent is given goal and returns the result;
  – **effectiveness**: agent can terminate search when acceptable solution found. Has a higher degree of multi-threading;
  – **transparency and optimization**: correlation between multiple data sources possible => higher quality results.
Taxonomy of Agents

- **Intelligent Agents**
  - **Human Agents** (e.g. travel agents)
  - **Hardware Agents** (e.g. robot)
  - **Software Agents**
    - **Interface Agent**
    - **Information Agents**
    - **Cooperation Agents**
    - **Transaction Agents**

- **Intelligent**
- **Interactive**
- **Social**
- **Mobile**
- **Adaptable**
Intelligent Agents' Characteristics

Agent
- Learning
- Proactivity
- Goal-oriented

Character
- Autonomy
- Mobility
- Reactivity

Environment

Communication
- Cooperation
- Coordination
Information Agent

Number of agents

Degree of Intelligence

Mobility

Multi-agent system

Single agent

stationary

mobile

simple complex
Cooperation Agent

Number of agents

Degree of Intelligence

Mobility

Multi-agent system

Stationary

Mobile

Single agent

Simple complex
Transaction Agent

Number of agents

Mobility

Degree of Intelligence

Multi-agent system
Single agent
stationary
mobile

simple complex
Areas of Influence

Characteristics

- Autonomy
- Decision Theory
- Artificial Intelligence
- Learning Capability
- Proactivity
- Reactivity
- Character
- Psychology

Mobility

Communication

Cooperation

Network Communication

Distributed Artificial Intelligence

Artificial Intelligence

Communication

Distributed Artificial Intelligence

Network Communication
Subareas of D.A.I.

- Parallel A.I.
- Distributed Problem Solving
- Multi-agent Systems
Agent as a black box

Intelligent Agent Processing

Input (perception)  Output (action)

Reactive vs Deliberative
The work of an Intelligent Agent

Input (perception)

Interaction

Information fusion

Information processing

Action

Output (action)
BDI Architecture

Rao/Georgeff ‘95
Architectural of deliberative agents

Input (perception)

Executor

Scheduler

Planner

Reasoner

Knowledge base

Symbolic environment model

Manager

Information receiver

Output (action)
Architecture of reactive agents

Brooks ‘86
## Existing Agent Architectures

<table>
<thead>
<tr>
<th>Deliberative Agents</th>
<th>Existing System Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRATE (Jennings), BDI (Rao, Georgeff), MECCA (Steiner et al)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reactive Agents</th>
<th>Subsumption (Brooks), Pengi (Agre, Chapman), Dynamic Action Section (Maes), SynthECA (White)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Hybrid Agents</th>
<th>RAP (Firby), <strong>Interrap</strong> (Muller), AIS (Hayes-Roth), TouringMachine (Ferguson)</th>
</tr>
</thead>
</table>
BDI

```
Initialize-state();
repeat
  options=option-generator(event-queue);
  selected-options=deliberate(options);
  update-intentions(selected-options);
  execute();
  get-new-external-events();
  drop-successful-intentions();
  drop-impossible-intentions();
end repeat
```

BDI has formal logic, partially implemented in algorithm, dMars, PRS also BDI implementations.
Subsumption

• Brooks ‘86, Hayzelden ‘98, White ‘98
• No explicit knowledge ("connectionist")
• Distributed behaviour architecture
• Intelligence is “emergent”
• No reasoner, planner or centralized “manager”
• pure activity-oriented task division rather than functional decomposition.
Suppressor and Inhibitor Nodes

Competence module 1: Move around

Competence module 0: Avoid contact

Suppressor node

Inhibitor node

Suppressor node: modifies input signal for period of time
Inhibitor node: inhibit output for period of time
Spreading Activation Model

Mathematical Model
Reactive Systems

Pengi explained...

More Pengi
Interrap Hybrid Architecture

Social Model
Mental Model
World Model

SG → PS
SG → PS
SG → PS

Cooperative planning layer (CPL)
Local planning layer (LPL)
Behavior-based layer (BBL)

Sensors → Communication → Actuators
Touring Architecture

Layer connectivity in Touring Machines

- Modelling Layer (M)
- Planning Layer (P)
- Reactive Layer (R)

Context activated Control Rules

Sensors → Modelling Layer (M) → Planning Layer (P) → Reactive Layer (R) → Action Effectors

Clock
# Communication and Cooperation

<table>
<thead>
<tr>
<th>Communication</th>
<th>Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blackboard</td>
<td>Dialogs</td>
</tr>
<tr>
<td></td>
<td>Messages</td>
</tr>
</tbody>
</table>

- Strategies
- Protocols
Distributed Problem Solving

Overall Problem

Subproblem 1
Subsolution 1

Subproblem 2
Subsolution 2

...

Subproblem n
Subsolution n

Overall Solution
Blackboard

Domain Blackboard

Control Blackboard

Agent

Agent

Agent

Management module

Enumerate KSAR

Choose KSAR

Execute KSAR

Domain KSAR

Control KSAR
• Messages based upon ‘speech acts’ [Austin, 62]
• A speech act designates a message that contains not only a true/false statement but also exercises a direct influence on the environment by causing changes within the environment.

Can you give me certain information?
Knowledge Query and Manipulation Language

• KQML based upon speech act theory
  – result of American Knowledge Sharing Effort (KSE) [Finin ‘93].

• KQML differentiates between three layers: communication, messages and content
  – communication: protocol
  – messages: speech acts
  – content: content or meaning of message

• KQML deals with speech acts.
Dialog: a sequence of agent message interactions with some common thread.
KQML format

(<Performative>
  :content <statement/speechact>
  :sender <name>
  :receive <name>
  :language <text>
  :ontology <text>
)

Performative corresponds to speech act types.
## Important KQML speech act types

<table>
<thead>
<tr>
<th>Speech act type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>achieve</td>
<td>S wants E to make true some statement in his environment</td>
</tr>
<tr>
<td>advertise</td>
<td>S is particularly suitable to perform some particular speech act type</td>
</tr>
<tr>
<td>ask-all</td>
<td>S wants all answers in E's knowledge base</td>
</tr>
<tr>
<td>ask-one</td>
<td>S wants an answer in E's knowledge base</td>
</tr>
<tr>
<td>broker-one</td>
<td>S wants E to find help for answering of his speech act</td>
</tr>
<tr>
<td>deny</td>
<td>The speech act no longer applies for S</td>
</tr>
<tr>
<td>delete</td>
<td>S wants E to remove specific facts from his knowledge base.</td>
</tr>
</tbody>
</table>
# Important KQML speech act types

<table>
<thead>
<tr>
<th>Speech act type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>recommend-one</td>
<td>S wants the name of an agent that can answer a speech act</td>
</tr>
<tr>
<td>recruit-on</td>
<td>S wants E to request an agent to perform a speech act</td>
</tr>
<tr>
<td>sorry</td>
<td>S does not possess the required knowledge or information</td>
</tr>
<tr>
<td>subscribe</td>
<td>S wants continuously information of E's answers for a speech act</td>
</tr>
<tr>
<td>tell</td>
<td>S transfers an information item.</td>
</tr>
</tbody>
</table>
Example

(ask-one
 :content (PRICE IBM ?price)
 :receiver stock-server
 :language LPROLOG
 :ontology NYSE-TICKS
 )

Query formulated using LPROLOG. Ontology is ‘computer systems’.
Using a Facilitator

1. Advertise (ask(x))

2. Broker (ask(x))

3. Ask(x)

4. Tell(x)

5. Tell(x)

agent A

agent B

facilitator

ask

tell

reply

Cooperation typology

Multi-agent systems

Independent
- Discrete
- Emergent Cooperation
  - Stigmergic
  - Similarity
    - Spatial
    - Temporal

Cooperative
- Communicative
- Non-communicative
  - Deliberative
  - Negotiating

Doran et al ‘97
Contract-Net Protocol

• Desire for efficient coordination in multi-agent systems.
  – Subtasks are openly offered as bids
  – Nodes reply, if interested

• Requires a commonly understood inter-node language [Smith, 80].
  – Common message format.
Contract Net Systems

- Contract net system engaged after problem division phase.
- Manager node undertakes the assignment of subproblems via the contract net protocol.
Example Protocol

1. TO: all nodes
   FROM: manager
   TYPE: task bid announcement
   ContractID: xx-yy-zz
   Task Abstraction: <subproblem description>
   Eligability Specification: <list of minimum requirements>
   Bid specification: <description of required application information>
   Expiration time: <latest possible application time>

2. TO: manager
   FROM: node X
   TYPE: application
   ContractID: xx-yy-zz
   Node Abstraction: <description of the node’s capabilities>

3. TO: node X
   FROM: manager
   TYPE: contract
   ContractID: xx-yy-zz
   Task Specification: <description of the subproblem>
Mobility: Remote Procedure Call

Interaction with remote object using ‘well known’ interface
Mobility: Remote Programming

Physical movement of agent is implied
<table>
<thead>
<tr>
<th>Communication</th>
<th>Properties</th>
<th>Agents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote Programming</td>
<td>High intelligence, flexible</td>
<td>mobile</td>
</tr>
<tr>
<td>Remote procedure call</td>
<td>low intelligence, proprietary</td>
<td>stationary</td>
</tr>
</tbody>
</table>
Collaboration

• Division of work amongst many agents of the same type in achieving goal