CHAPTER 11: APPLICATIONS

An Introduction to Multiagent Systems

http://www.csc.liv.ac.uk/˜mjw/pubs/imas/

1 Application Areas

- Agents are indicated for domains where *autonomous action* is required.
- Multiagent systems are indicated for domains where:
  - control, data, expertise are distributed;
  - centralised control is impossible or impractical;
  - processing nodes have competing/conflicting viewpoints or objectives.

Some Applications

- *Interface agents*;
- *Internet agents*;
- *E-commerce agents*;

1.1 Interface Agents

- The idea is to move away from the *direct manipulation* paradigm that has dominated for so long.
- Agents sit ‘over’ applications, watching, learning, and eventually doing things without being told — taking the initiative.
- Pioneering work at MIT Media Lab (Pattie Maes):
  - news reader;
  - web browsers;
  - mail readers.

Nicholas Negroponte's Vision

'The 'agent' answers the phone, recognises the callers, disturbs you when appropriate, and may even tell a white lie on your behalf. The same agent is well trained in timing, versed in finding opportune moments, and respectful of idiosyncracies. ' (p150)
'If you have somebody who knows you well and shares much of your information, that person can act on your behalf very effectively. If your secretary falls ill, it would make no difference if the temping agency could send you Albert Einstein. This issue is not about IQ. It is shared knowledge and the practice of using it in your best interests.' (p151)
'Like an army commander sending a scout ahead . . . you will dispatch agents to collect information on your behalf. Agents will dispatch agents. The process multiplies. But [this process] started at the interface where you delegated your desires.' (p158)

(From *Being Digital*, 1995.)

2 Email Reading Assistants

- MAXIMS (Pattie Maes, 1994) 'learns to prioritize, delete, forward, sort, and archive mail messages on behalf of a user . . . '.
- Works by 'looking over the shoulder' of a user, and learning about how they deal with email.
- Each time a new event occurs (e.g., email arrives), MAXIMS records the situation $\rightarrow$ action pairs generated.
- Situation characterised by features of event:

  sender of email; recipients; subject line; etc.

- When new situation occurs, MAXIMS matches it against previousy recorded rules.

- Predicts user action, and generates a *confidence level*.
- Confidence level compared against two threshholds: “tell me” and “do it”:

confidence $<$ “tell me”	agent gets feedback
“tell me” $<$ confidence $<$ “do it”	agent makes suggestion
confidence $>$ “do it”:	agent acts

- Rules can be “hard coded”; even get help from other users.
- MAXIMS has a simple ‘personality’, (a face icon), communicating its ‘mental state’ to the user.

3 Agents on the Internet

- It is not easy to find the right information (even with the help of search engines).
- *Systematic* searches are difficult:
  - *human factors*: we get bored by slow response times, find it difficult to read the WWW rigorously, get tired, miss things easily, misunderstand, get sidetracked;
  - *organizational factors*: structure on the net is superficial — no standards for home pages, not (yet) semantic markup to tell you what a page contains;
- The sheer *amount* of information presented to us leads to ‘information overload’.

- What we want is a kind of 'secretary': someone who understood the things we were interested in, (and the things we are not interested in), who can act as 'proxy', hiding information that we are not interested in, and bringing to our attention information that *is* of interest.
- We cannot afford *human* agents to do these kinds of tasks (and in any case, humans get suffer from the drawbacks we mentioned above).
- So we write an agent to do these tasks.

Tour guides

- The idea here is to have agents that help to answer the question 'where do I go next' when browsing the WWW.
- Such agents can learn about the user's preferences in the same way that MAXIMS does, and rather than just providing a single, uniform type of hyperlink actually indicate the likely interest of a link.

Indexing agents

- Indexing agents will provide an extra layer of abstraction on top of the services provided by search/indexing agents such as GOOGLE and LYCOS.
- The idea is to use the raw information provided by such engines, together with knowledge of the users goals, preferences, etc., to provide a *personalised* service.

FAQ-finders

- The idea here is to direct users to FAQ documents in order to answer specific questions.
- Since FAQS tend to be knowledge intensive, structured documents, there is a lot of potential for automated FAQ servers.

Expertise finders

- Suppose I want to know about experts in intelligent agents.
- Current WWW search tools would simply take the words “intelligent” “agents” and search on them.
- This is not ideal: GOOGLE has no model of what you *mean* by this search, or what you really *want*.
- Expertise finders ‘try to understand the users wants and the contents of information services’, in order to provide a better information provision service.

4 Agents for E-Commerce

- Another important rationale for internet agents is the potential for *electronic commerce*.
- Most commerce is currently done *manually*. But there is no reason to suppose that certain forms of commerce could not be safely delegated to agents.
- Examples:
  - find the cheapest copy of MS Office from online stores;
  - flight from Manchester to Dusseldorf with veggie meal, window seat, plus hotel, taxis, entertainment, restaurants.

First & Second Generation E-Commerce Systems

- First generation: *comparison shopping agents*.
- Examples:
  - 1995: Bargain Finder from Andersen;
  - 1997: Jango from NETBOT.
  - 2003: Froogle from GOOGLE
- *Second-generation*: negotiation, brokering, . . .

Jango

- Jango (Doorenbos et al, Agents 97) is good example of e-commerce agent.
- Long-term goals:
  1. Help user decide what to buy.
  2. Finding specs and reviews of products.
  3. Make recommendations.
  4. Comparison shopping for best buy.
  5. Monitoring “what’s new” lists.
  6. Watching for special offers & discounts.

- Isn't comparison shopping impossible? WWW pages all different!
- Jango/ShopBot exploits several *regularities* in merchant WWW sites:
  - *navigation regularity*:
    sites designed so that products easy to find
  - *corporate regularity*:
    sites designed so that pages have same look'n'feel;
  - *vertical separation*:
    merchants use whitespace to separate products.

- Two key components of Jango/ShopBot:
  - *learning vendor descriptions*;
  - *comparison shopping*;