Agent-based Simulation of Reactive, Pro-active and Social Animal Behaviour*

Catholijn M. Jonker, Jan Treur

Vrije Universiteit Amsterdam
Department of Mathematics and Computer Science, Artificial Intelligence Group
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
URL: http://www.cs.vu.nl/~{jonker,treur} Email: {jonker,treur}@cs.vu.nl

Abstract. In this paper it is shown how animal behaviour can be simulated in an agent-based manner. Different models are shown for different types of behaviour, varying from purely reactive behaviour to pro-active and social behaviour. The compositional development method for multi-agent systems DESIRE and its software environment supports the conceptual and detailed design, and execution of these models. Experiments reported in the literature on animal behaviour have been simulated for a number of agent models.

1 Introduction

One of the most important aspects of agents (cf. [25]) is their behaviour. In the past, behaviour has been studied in different disciplines. In Cognitive Psychology the analysis of human behaviour is a major topic. In Biology, animal behaviour has been and is being studied extensively. Around 1900 a discussion took place about the manner in which observed animal behaviour can be interpreted in order to obtain an objective and testable description; for an overview, see [2], [23]. A risk of taking the intentional stance (e.g., [10]) as a perspective to explain behaviour, is that explanations are generated that make use of (a large number of) mental concepts that cannot be tested empirically. Therefore the *principle of parsimony* was introduced, stating that 'in no case may we interpret an action as the outcome of the exercise of a higher psychical faculty, if it can be interpreted as the outcome of the exercise of one which stands lower in the psychological scale'; see [18].

Building further on this perspective *behaviourism* was developed, e.g., [12], [21], [24]. In this approach animal behaviour is explained only in terms of a black box that for each pattern of *stimuli* (input of the black box) from the environment generates a *response* (output of the black box), that functionally depends on the input pattern of stimuli; i.e., if two patterns of stimuli are offered, then the same behaviour occurs if the two patterns of stimuli are equal. This view was also extended to human behaviour. Because of the underlying black box view, behaviourism discouraged reference to internal (mental) activities of organisms: any speculation about the

* In: Proceedings of the 11th International Conference on Industrial and Engineering Applications of AI and Expert Systems, IEA/AIE'98, Lecture Notes in AI, Springer Verlag. In press, 1998

internal functioning of the black box (i.e., the processes that might mediate between sensory inputs and behavioural outputs) was forbidden; cf. [23], p. 4.

In this paper, the compositional development method for multi-agent systems DESIRE (see [4]) is used to design, implement and experiment with agent-based simulation models for animal behaviour. In Section 3 a generic model of a *purely reactive agent* is introduced which is an adequate agent model to describe the (immediate) functional character of stimulus-response behaviour. The black box is represented by the agent component. The stimuli form the input (observation results), and the response is formed by the actions generated as output.

Viewed from a Software Engineering perspective, modelling behaviour by a functional relation between input and output provides a system that can be described as a (mathematical) *function* $F : \text{Input\_states} \rightarrow \text{Output\_states}$ of the set of possible input states to the set of possible output states. Such a system is transparent and predictable. For the same input always the same behaviour is repeated: its behaviour does not depend on earlier processes; for example, no information on previous experiences is stored in memory so that it can be remembered and affect behaviour. Well-known traditional programming methods are based on this paradigm; for example, program specification and refinement based on preconditions and postconditions as developed in, e.g., [11].

As opposed to behaviour defined by a purely functional dependency between input and output, an agent's behaviour often takes previous processes in which it was involved into account. These previous processes may have led to internal storage of information in a *memory* so that the same input pattern of stimuli can lead to different behaviour a next time it is encountered; the agent may be able to deliberate about it. Again viewed from a Software Engineering perspective, this makes that agents do not fit strictly in the paradigm based on a functional relation: to keep the functional relation, not only the actual input, but also the complete history of input should be taken into account, or the internal information in memory should be considered to be additional input.

In Section 4 a generic agent model is presented that can be used to model more complex behaviour. It includes not only components that represent the agent's memory (the agent's beliefs on the world and on other agents), but also components that represent the agent's goals and the agent's communication with other agents. This generic agent model has been used to obtain different agent models for different types of animal behaviour that go beyond purely reactive behaviour: *delayed response behaviour*, *deliberate pro-active behaviour*, and *social behaviour*. In Section 2 a problem description (a description of a pseudo-experiment) is presented; in Section 5 the behaviours of the different agent models introduced in Section 3 and 4 are compared for each of the situations defined in Section 2.

2 Problem Description

The deliberations put forward in the introduction can be illustrated by a very concrete example, taken from the discipline that studies animal behaviour; e.g., [23]. Animals, for example dogs, sometimes show a *delayed response*: they look for food in places where they have seen food before. This suggests that these animals might have some internal representation or memory of stimuli received earlier. More systematic experiments on this delayed response issue, for example those reported in [13] and [22], support this suggestion.

2.1 The Domain

The type of experiment reported in [22] is set up as follows (see Figure 1). Separated by a transparent screen (a window, at position p0), at each of two positions p1 and p2 a cup (upside down) and/or a piece of food can be placed. At some moment (with variable delay) the screen is raised, and the animal is free to go to any position. Consider the following three possible situations:

Situation 1 At both positions p1 and p2 an empty cup is placed.

Situation 2 At position p1 an empty cup is placed, and at position p2 a piece of food, which is (and remains) visible for the animal.

Situation 3 At position p1 an empty cup is placed and at position p2 a piece of food is placed, after which a cup is placed at the same position, covering the food. After the food disappears under the cup it cannot be sensed anymore by the animal.

In situation 1 the animal will not show a preference for either position p1 or p2; it may even go elsewhere or stay where it is. In situation 2 the animal will go to position p2, which can be explained as pure stimulus-response behaviour. In situation 3 the immediate stimuli are the same as in situation 1. Animals that react in a strictly functional stimulus-response manner will respond to this situation as in situation 1. Animals that show delayed response behaviour will go to p2, where food can be found.

In the literature, many reports can be found of observed delayed response behaviour in experiments of the type described above: [23], p. 4-5. The animal species used in these experiments vary from rats and dogs to macaques, chimpanzees and human infants. Therefore, it is assumed that animals of the type studied maintain internal (mental) representations on the basis of their sensor input, and that they make use of these representations (in addition to the actual sensory input that is used) to determine their behaviour. In a way it can be said that they may act as deliberate agents.

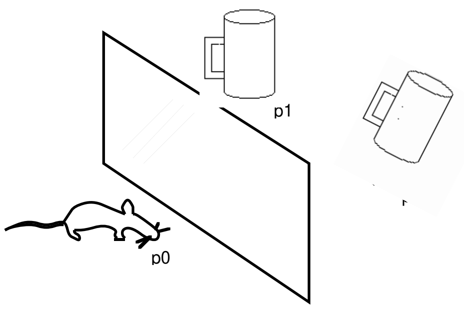

Fig. 1. Situation 3 of the experiment

2.2 The Requirements

In this paper four agent models A, B, C and D for the experiment are described. The following requirements on their behaviour express possible hypotheses that can be made about the behaviour of animals in the experiments:

A. An agent with purely reactive behaviour should behave the same for the two situations 1 and 3 described above: doing nothing, as if no food is present. Only in situation 2 should it go to the position of the food.

B. An agent with delayed response behaviour should behave the same in the situations 2 and 3: it should go to the position of the food. In situation 1 it should do nothing.

C. A deliberate pro-active agent's behaviour in the situations 1, 2 and 3 depends on whether the agent has a motivation or goal to do so. E.g., the agent may start acting in a pro-active manner (without any specific stimulus) in situation 1.

D. A social agent is able to take into account communication with other agents. If another animal is present that communicates that it wants to have the food (e.g., by growling), and the agent believes that this other agent is higher in the hierarchy, then the agent will not try to get the food.

3 An Agent Model for Purely Reactive Behaviour

An agent is *purely reactive* if it immediately responds to stimuli from its environment. Such agents are also called behaviour-based or situated agents; e.g., see [19]. These agents make their decisions based on a very limited amount of information, and simple situation-action rules. The stimuli can either consist of perceived changes in the external world or received communications from other agents. Changes in the external world are perceived by the agent by observation. The response behaviour of the agent affects its environment. Several archictures have been developed for reactive agents, see [1], [8], [16], [17]. In [19] an extensive overview of these architectures and the motivations behind them can be found.

3.1 Process Composition

For the design and implementation of the different models the compositional development method for multi-agent systems DESIRE has been used; see [4] for more details. A generic agent model for purely reactive behaviour developed earlier within the DESIRE environment (and applied in chemical process control) was reused. The (rather simple) agent system in this model consists of two components, one for the agent (of type A) and one for the external world with which it interacts (see Figure 2).

In the current domain, the observation information that plays a role describes that certain *objects* (cup1, cup2, food, screen, self) are at certain *positions* (i.e., p0, p1, p2). This is modelled by two sorts OBJECT and POSITION and a relation at position between these two sorts. Moreover, two types of *actions* can be distinguished: eat and goto some position. The latter type of actions is parameterized by positions; this can be modelled by a function goto from POSITION to ACTION. E.g., goto(p1) is the action to go to position p1. The action eat that is specified assumes that if the animal is at the position of the food, it can have the food: if a cup is covering the food, as part of the action eat the animal can throw the cup aside to get the food. Variables over a sort, e.g., POSITION, are denoted by a string, e.g., P, followed by : POSITION, i.e, P : POSITION is a variable over the sort POSITION. The unary relation to_be_performed is used to express the information that the agent has decided to *perform an action*; for example, to_be_perfomed(goto(p1)) expresses that the agent has decided to go to position p1. The relation observation_result is used to express the information that certain information has been acquired by *observation*; e.g., observation_result(at_position(food, p1), pos) expresses that the agent has observed that there is food at position p1, whereas the statement observation_result(at_position(food, p1), neg) expresses that the agent has observed that there is *no* food at position p1.

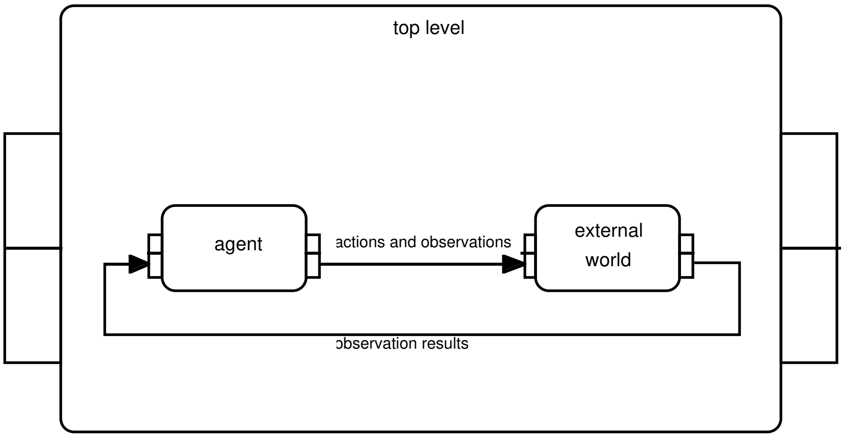

Fig. 2. A generic agent model for purely reactive behaviour

3.2 The Domain Knowledge

Assuming that food is offered at at most one position (for example, position p2), the stimulus-response behaviour of agent model A expresses that if the agent observes that there is food at any position and that no screen at position p0 separates the agent from this position, then it goes to this position. This knowledge has been modelled in the following form:

if observation_result(at_position(food, P:POSITION), pos)
and observation_result(at_position(screen, p0), neg)
and observation_result(at_position(self, P:POSITION), neg) **then** to_be_performed(goto(P:POSITION))

if observation_result(at_position(self, P:POSITION), pos)
and observation_result(at_position(food, P:POSITION), pos) **then** to_be_performed(eat)

3.3 The Behaviour of the Purely Reactive Agent

The requirement imposed on agent A was that it shows the same behaviour for situations 1 and 3 in the problem description: do nothing. Moreover, in situation 2 the agent is required to go to the position of the food. The agent of type A indeed shows behaviour as expressed by the requirements.

4 Agent Models with More Complex Behaviour

To design an agent model that will show delayed response behaviour, the internal structure of the agent is made more complex. Within the agent a component maintenance of world information to maintain the observation results as beliefs (a memory) is distinguished from a component world interaction management that manages the interaction with the world. Moreover, if the agent has to generate its own goals in order to show pro-active behaviour, a component own process control is added, and if the agent has to show social behaviour, components are added to manage communication (agent interaction management) and to maintain beliefs on other agents (maintenance of agent information). The generic agent model depicted in Figure 3 (see also [7]) is composed of all of these components.

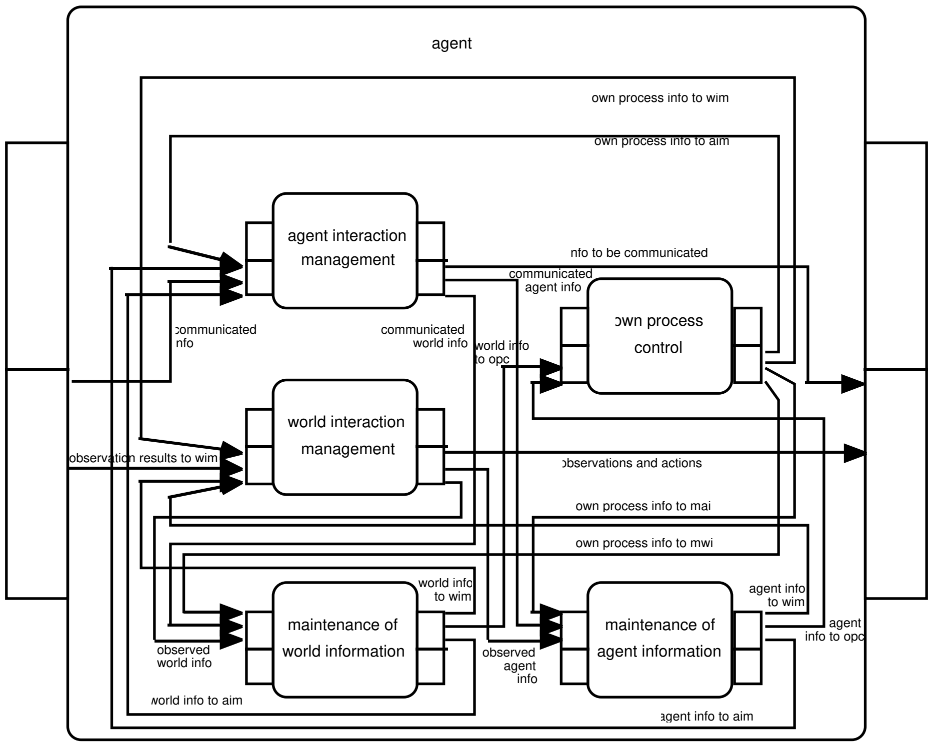

Fig. 3. A generic agent model for deliberate reactive, pro-active and social behaviour

4.1 An Agent Model with Delayed Response Behaviour

For an agent with *delayed response behaviour* (type B), the component maintenance of world information is used, in addition to the component world interaction management. The only task performed by the component maintenance of world information is storage of observation information. No further knowledge is used within this component.

The part of the knowledge of the component world interaction management that determines the actions is a variant of the knowledge used in agent model A. An additional part determines that the world information that was acquired by observation has *to be maintained*, expressed by the relation new_world_info.

```
if    observation_result(I:INFO_ELEMENT, S:SIGN) then new_world_info(I:INFO_ELEMENT, S:SIGN)

if    belief(at_position(food, P:POSITION), pos)
 and  belief(at_position(screen, p0), neg)
 and  belief(at_position(self, P:POSITION), neg)    then    to_be_performed(goto(P:POSITION))

if    belief(at_position(food, P:POSITION), pos)
 and  belief(at_position(self, P:POSITION), pos)    then    to_be_performed(eat)
```

An essential difference with the knowledge in agent model A is that in the knowledge above the relation observation result is replaced by the relation belief. Not only can information from direct observation be used, but also information retrieved from memory: all input information gets the status of belief, in contrast to observation. The behaviour of this agent model in comparison to the behaviours of the other models is discussed in Section 5.

4.2 An Agent Model with Pro-active Behaviour

The third agent model to be discussed is a model for a *pro-active agent* (type C). A pro-active agent does not simply respond to stimuli, neither immediately, nor delayed. In addition to the observation information, its so-called *motivational attitudes* (such as goals and plans) play an important role (e.g., see [25]). These motivational attitudes can be based on the agent's own character (for example, an agent's character may be that it always wants to eat, or that it is totally apathic), but also on specific aspects of the agent's own state, such as being hungry, or being depressed. To determine the motivational attitudes of the agent, the component own process control is used; additional knowledge structures are introduced for this new component. One action is added to the information type domain actions: the action explore position. In addition to the existing information types of agent model B, information types are required for knowledge on own process control; these information types express information on:

- the agent's beliefs
- aspects of the agent's own state, such as being hungry, or being depressed, and specific characteristics of the agent, such as always eager to eat, or totally apathic
- the agent's goals, such as be fed, just hang around, find food, or get food inside.

Information on the agent's *own state* can be expressed using the unary relation own_state; for example, the statement own_state(hungry) expresses that the agent is hungry. The agent's *own characteristics* can be expressed using the unary relation own_characteristic; e.g., the statement own_characteristic(totally_apathic) expresses the information that the agent is totally apathic. The *goal* that has been selected by the agent is expressed using the unary relation selected_goal. The knowledge to be used in the component own process control models:

- an agent that is always eager to eat, always selects the goal be fed
- any not apathic agent that is hungry or depressed selects the goal be fed
- a totally apathic agent always selects the goal just hang around
- an agent which has be fed as a goal, selects the goal get food inside if it has a belief that food is present at a specific position; in the other case it selects the goal find food

The knowledge used in own process control knowledge can formulated in a concise form as follows:

```
if        own_characteristic(always_eager_to_eat)      then      selected_goal(be_fed)

if        own_state(hungry)
  and not own_characteristic(totally_apathic)          then      selected_goal(be_fed)

if        own_state(depressed)
  and not own_characteristic(totally_apathic)          then      selected_goal(be_fed)

if        own_characteristic(totally_apathic)          then      selected_goal(just_hang_around)

if        selected_goal(be_fed)
  and belief(at_position(food, P:POSITION), pos)       then      selected_goal(get_food_inside)

if        selected_goal(be_fed)
  and not  belief(at_position(food, p1), pos)
  and not  belief(at_position(food, p2), pos)          then      selected_goal(find_food)
```

Depending on the type of agent modelled, some facts can be added to this knowledge base, for example in the agent of type C:

```
own_characteristic(always_eager_to_eat)
```

(alternatively, for example, own_state(hungry), **not** own_characteristic(totally_apathic) could be specified, or own_characteristic(totally_apathic))

Depending on the agent characteristics specified, the agent determines one or more goals. To actually show certain pro-active behaviour, also suitable knowledge has to be specified on which actions are to be performed for a given goal. This knowledge is used in the component world interaction management.

To determine actions related to the goal get food inside, two possible cases are considered:

- the agent believes that food is present at its own position; in this case it simply can start eating
- the agent believes that no food is present at its own position, but it believes that food is present at another position; in this case the agent can go to such a position (and if it arrives there it can start eating, according to the previous item)

This knowledge is expressed in a concise form as follows:

```
if        selected_goal(get_food_inside)
  and belief(at_position(food, P:POSITION), pos)
  and belief(at_position(self, P:POSITION), pos)       then      to_be_performed(eat)
```

```
if       selected_goal(get_food_inside)
  and belief(at_position(self, P1:POSITION), pos)
  and belief(at_position(food, P1:POSITION), neg)
  and belief(at_position(food, P2:POSITION), pos)
  and belief(at_position(screen, p0), neg)          then        to_be_performed(goto(P2:POSITION))
```

The goal get food inside assumes that the agent already knows at least one position where food is present. If this is not the case, the goal find food may be selected by the agent. To determine the actions for the goal find food, the following cases are considered:

- the agent does not know whether food is present at its own position; then the action explore position is selected (which determines whether food is present at the agent's own position)
- the agent believes that no food is present at its own position, and it does not know whether food is present at positions p1 and p2; in this case the action go to p1 is selected (and if it arrives there it can start exploring it, according to the previous item)
- the agent believes that no food is present at its own position and at position p1; it does not know whether food is present at p2; in this case the agent goto p2 is selected (and if it arrives there it can start exploring the position, according to the first item)

This knowledge is expressed in a concise form as follows:

```
if       selected_goal(find_food)
  and belief(at_position(self, P:POSITION), pos)
  and not  belief(at_position(food, P:POSITION), pos)
  and not  belief(at_position(food, P:POSITION), neg)    then    to_be_performed(explore_position)

if       selected_goal(find_food)
  and belief(at_position(self, P:POSITION), pos)
  and belief(at_position(food, P:POSITION), neg)
  and not  belief(at_position(food, p1), neg)
  and not  belief(at_position(food, p1), pos)
  and not  belief(at_position(food, p2), neg)
  and not  belief(at_position(food, p2), pos)          then        to_be_performed(goto(p1))

if       selected_goal(find_food)
  and belief(at_position(self, P:POSITION), pos)
  and belief(at_position(food, P:POSITION), neg)
  and belief(at_position(food, p1), neg)
  and not  belief(at_position(food, p2), neg)
  and not  belief(at_position(food, p2), pos)          then        to_be_performed(goto(p2))
```

4.3 An Agent Model with Social Behaviour

To obtain *social behaviour* (an agent model of type D), also the components agent interaction management and maintenance of agent information are used in the model. In the component agent interaction management knowledge is specified that identifies new communicated knowledge about other agents:

```
if communicated_by(I_want_food, pos, A:AGENT)  then        new_agent_info(wants_food(A:AGENT))
```

Here, the statement communicated_by(I_want_food, pos, A:AGENT) expresses that the information I_want_food *has been communicated* (positively) by the agent A:AGENT. This *new agent information* (expressed using the relation new_agent_info) is stored in the component maintenance of agent information: the knowledge used in maintenance of agent information specifies the hierarchy between different animals, and whether another animal that is present wants the food. For example, if an1 is an animal which is present and is *higher in the hierarchy*, then this can be specified within the knowledge base used in maintenance of agent information of the agent self as higher_than(an1, self). It is also possible to model this information in a dynamic form, as an outcome of earlier experiences (fights). If the other animal wants the food, within the component maintenance of world information of the agent self it is derived that the food is protected, using the knowledge

if wants_food(A:AGENT) **and** higher_than(A:AGENT, self) **then** food_protected

Within the knowledge elements used in the component world interaction management an additional condition **not** belief(food_protected, pos) is specified.

5 Overview of the Behaviour of the Different Agent Models

In the following table the differences in behaviour of the agent models are summarised. Here D2 denotes the agent model D in the presence of a higher competitor, whereas D1 denotes the same agent model without such a competitor.

	agent A	*agent B*	*agent C*	*agent D1*	*agent D2*
situation 1 (no food)	do nothing	do nothing	look for food	look for food	do nothing
situation 2 (visible food)	go to food	go to food	go to food	go to food	do nothing
situation 3 (invisible food)	do nothing	go to food	go to food	go to food	do nothing

The different variants of behaviour depicted in this table indeed satisfiy the requirements expressed in Section 2.

6 Conclusions

In this paper it is shown how different types of animal behaviour can be modelled and simulated at a conceptual level on the basis of the compositional multi-agent development method DESIRE (cf. [4]). Different (variants of) reusable compositional agent models were used to model the different required behaviours. The advantage of this approach is that the models are designed at a high conceptual level, in terms of the processes, information and knowledge that is needed, and abstracting from implementation details. Nevertheless they can be executed by the DESIRE software environment. Besides the simulation of animal behaviour discussed in this paper, a variety of other applications have been developed using DESIRE. Some recent multi-agent applications can be found in [3] (negotiation between agents), [5] (simulation of a society of agents), [6] (distributed work flow and agenda scheduling), and [15] (agents in brokering processes).

Simulation of animal behaviour is an interesting type of application for multi-agent systems. Both areas can benefit from a more extensive study of this type of application. The area of multi-agent systems can benefit from the more detailed analyses and distinctions that have been made for different types of animal behaviour (see the introduction). The study of animal behaviour can benefit from software tools for agent modelling at a conceptual level that support simulation. Moreover, formal techniques in the area of verification can be used to analyse and formalise behaviour properties of animals and their logical relations; e.g., [9], [14]. These formalisations can be a basis for the development of a theory of animal behaviour.

Acknowledgements
Parts of an earlier version of this paper were read by Frances Brazier. Based on her comments a number of improvements have been made in the text.

References

1. Agre, P.E., and D. Chapman, Pengi: an Implementation of a Theory of Activity. In: Proceedings of the sixth National Conference of the American Association for Artificial Intelligence (AAAI-87), Morgan Kaufmann, 1987, pp. 268-272.
2. Allen, C., and Bekoff, M., Species of Mind: the philosophy and biology of cognitive ethology. MIT Press, 1997
3. Brazier, F.M.T., F. Cornelissen, R. Gustavsson, C.M. Jonker, O. Lindeberg, B. Polak, J. Treur, Agents Negotiating for Load Balancing of Electricity Use. In: Proceedings of the Third International Conference on Multi-Agent Systems, ICMAS-98, IEEE Society Press, 1998
4. Brazier, F.M.T., Dunin-Keplicz, B., Jennings, N.R. and Treur, J., Formal specification of Multi-Agent Systems: a real-world case. In: V. Lesser (Ed.), Proceedings of the First International Conference on Multi-Agent Systems, ICMAS-95, MIT Press, Cambridge, MA, 1995, pp. 25-32. Extended version in: International Journal of Cooperative Information Systems, M. Huhns, M. Singh, (Eds.), special issue on Formal Methods in Cooperative Information Systems: Multi-Agent Systems, vol. 6, 1997, pp. 67-94.
5. Brazier, F.M.T., Eck, P.A.T. van, and Treur, J. Modelling a society of simple agents: from conceptual specification to experimentation. In: Conte, R., Hegselmann, R. and Terna, P. (eds.), Simulating Social Phenomena, Proceedings of the International Conference on Computer Simulation and Social Sciences, ICCS&SS'97, Lecture Notes in Economics and Mathematical Systems, vol. 456, Springer Verlag, 1997, pp. 103-107.
6. Brazier, F.M.T., C.M. Jonker, F.J.Jüngen, J. Treur, Distributed Scheduling to Support a Call Centre: a Co-operative Multi-Agent Approach. In: Proceedings of the Third International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAM-98, Practical Application Company, 1998
7. Brazier, F.M.T., Jonker, C.M., Treur, J. , Formalisation of a cooperation model based on joint intentions. In: J.P. Müller, M.J. Wooldridge, N.R. Jennings (eds.), Intelligent Agents III (Proc. of the Third International Workshop on Agent Theories, Architectures and Languages, ATAL'96), Lecture Notes in AI, volume 1193, Springer Verlag, 1997, pp. 141-155.

8. Brooks, R.A., A robust layered control system for a mobile robot. In: IEEE Journal of Robotics and Automation, Vol. RA-2 (1), 1986, pp. 14-23.
9. Cornelissen, F., Jonker, C.M., Treur, J., Compositional verification of knowledge-based systems: a case study in diagnostic reasoning. In: E. Plaza, R. Benjamins (eds.), Knowledge Acquisition, Modelling and Management, Proceedings of the 10th EKAW'97, Lecture Notes in AI, vol. 1319, Springer Verlag, 1997, pp. 65-80.
10. Dennett, D.C., The Intentional Stance. MIT Press, Cambridge, 1987.
11. Dijkstra, E.W., A discipline of programming. Prentice Hall, 1976.
12. Gibson, J.J., The concept of the stimulus in psychology. In: American Psychology 15, 1960, pp. 694-703.
13. Hunter, W.S., The delayed reaction in animals. Behavioral Monographs, 2, 1912, pp. 1-85
14. Jonker, C.M., Treur, J., Compositional Verification of Multi-Agent Systems: a Formal Analysis of Pro-activeness and Reactiveness. In: W.P. de Roever, A. Pnueli et al. (eds.), Proceedings of the International Workshop on Compositionality, COMPOS'97, Springer Verlag, 1998.
15. Jonker, C.M., J. Treur, A Generic Architecture for Broker Agents. In: Proceedings of the Third International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAM-98, Practical Application Company, 1998
16. Kaelbling, L.P., An archictecture for intelligent reactive systems. In: Allen, J., and J. Hendler, and A. Tate (eds.), Readings in Planning, Morgan Kaufmann, 1990, pp. 713-728.
17. Maes, P. (ed.), Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back. MIT / Elsevier, 1990.
18. Morgan, C.L., An introduction to comparative psychology. London: Scott, 1894.
19. Müller, J. P., The design of intelligent agents: a layered approach. Lecture Notes in Artificial Intelligence, Vol. 1177, 1996.
20. Pavlov, I.P., Conditioned reflexes. London: Oxford, 1927.
21. Skinner, B.F., The generic nature of the concepts of stimulus and response. Journal of gen. Psychology 12, 1935, pp. 40-65.
22. Tinklepaugh, O.L., Multiple delayed reaction with chimpanzees and monkeys. Journal of Comparative Psychology, 13, 1932, pp. 207-243
23. Vauclair, J., Animal Cognition. Harvard Univerity Press, Cambridge, Massachusetts, 1996.
24. Watson, J.P., Psychology from the standpoint of a behaviourist. Philadelphia: Lippincott, 1919.
25. Wooldridge, M.J., and N.R. Jennings, Intelligent Agents: Theory and practice. In: Knowledge Engineering Review, 10(2), 1995, pp. 115-152.