The PPP Persona: Towards a Highly Personalized User Interface

Elisabeth André, Thomas Rist and Jochen Müller

German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, D-66123 Saarbrücken
Email: <name>@dfki.uni-sb.de

Abstract. Animated agents - either based on real video, cartoon-style drawings or even model-based 3D graphics - offer great promise for computer-based presentations as they make presentations more lively and appealing and allow for the emulation of conversation styles known from human-human communication. In this paper, we describe a life-like interface agent which presents multimedia material to the user following the directives of a script. The overall behavior of the presentation agent is partly determined by such a script, and partly by the agent's self-behavior.

1. MOTIVATION

A growing number of research projects both in academia and industry provide increasing evidence that a next major step in the evolution of interfaces will be a shift towards highly personalized interfaces in which communication between user and computer will be mediated by life-like agents. Due to advances in computer graphics, the realization of visually appealing agents, either based on real video, cartoon-style drawings or even model-based 3D graphics, has become manageable. To be useful, however, such agents have to behave in a reasonable and believable manner.

The focus of our work is on presentation agents as they may be used in a broad range of applications including computer-based instruction, guides through information spaces, and product advertisement over the web. There are several motivations for using an animated presentation agent in the interface. For example, it enables more natural referential acts that involve locomotive, gestural and speech behaviors (cf. [Lester et al. 97]). In virtual environments, animated agents may help users learn to perform procedural tasks by demonstrating them (cf. [Rickel & Johnson 97]). Furthermore, they can also serve as a guide through a presentation to release the user from orientation and navigation problems known from multi-window/multi-screen settings (cf. [André et al. 97]). Last but not least, there is the entertaining and emotional function of such an animated character. It may help to lower the "getting started barrier" for novice users of computers/applications, and, as Adelson notes, "... interface agents can be valuable educational aids since they can engage students without distracting or distancing them from the learning experience" (cf. [Adelson 92], pp. 355).

To illustrate this, let's have a look at some examples taken from the PPP system (cf. [André et al. 96]). The first application scenario deals with instructions for the maintenance, and repair of technical devices, such as modems. Suppose the system is requested to explain the internal parts of a modem. A strategy to accomplish this task is to generate a picture showing the modem's circuit board and to introduce the names of the depicted objects. Unlike conventional static graphics where the naming is usually done by drawing text labels onto the graphics (often together with arrows pointing from the label

to the object), the PPP Persona enables the emulation of referential acts that also occur in personal human-human communication. In the example, it points to the transformer and utters "This is the transformer" (using a speech synthesizer). The example also demonstrates how facial displays and head movements help to restrict the visual focus. By having the Persona look into the direction of the target object, the user's attention is directed to the target object.

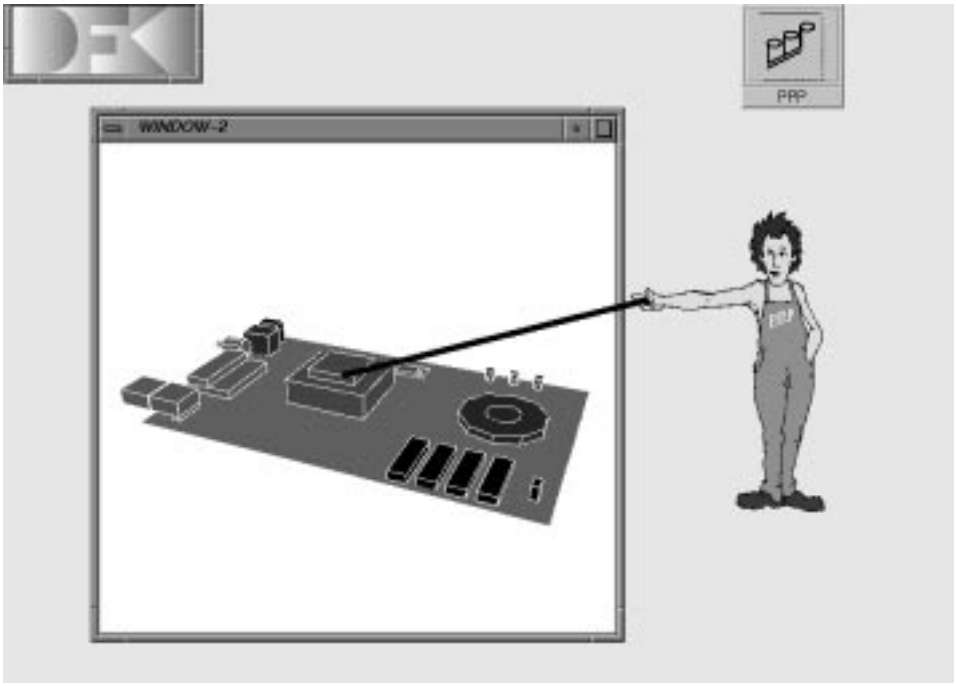

Figure 1: The Persona instructs the user in operating a technical device

In the second example, the Persona advertises accommodation offers found on the WWW. Suppose the user is planning to spend holidays in Finland and is therefore looking for a lake-side cottage. To comply with the user's request, the system retrieves a matching offer from the web and creates a presentation script for the PPP persona which is then sent to the presentation viewer (e.g. Netscape Navigator^{TM} including a Java^{TM} interpreter). When viewing the presentation PPP Persona highlights the fact that the cottage has a nice terrace by means of a *verbal annotation of a picture*; i.e., Persona points to the picture during a verbal utterance (cf. Fig. 2).

The overall behavior of the PPP persona is partly determined by a presentation script, and partly by the agent's self-behavior. In the next section, we introduce a declarative specification language for defining such behaviors. Behavior specifications are used to automatically generate a control module of a presentation display component. This display component will be referred to as the Persona Engine throughout the rest of the paper. When operated stand-alone, the Persona Engine expects a presentation script as input and displays the corresponding presentation as output. However, for an increasing number of applications the *manual* authoring of such presentation scripts (including the creation of the media material such as text, graphics, and animation clips) is no longer feasible because information has to be communicated fast and flexibly to meet the specific needs of the individual presentation consumer. Although powerful authoring tools have become available, they are not likely to solve this problem as they only facilitate presentation editing. Therefore, we also address the automatization of the whole authoring process, including the selection of appropriate information content, media allocation,

logical presentation structuring, creation of media items which are to convey selected information as well as the temporal scheduling of presentation acts.

There are several requirements an animated presentation agent has to meet. According to its functional roles in a presentation the animated character must be conversant with a broad variety of presentation gestures and rhetorical body postures. Furthermore, it should adopt a reasonable and lively behavior without being distracting. From the technical point of view, the declarative specification of agent behaviors should be supported, and the piece of software that realizes the *character player* should be highly independent of application and platform.

Figure 2: PPP Persona presents retrieval results from the web using the Netscape Navigator^{TM} and Java^{TM}

2. CONCEPTION OF THE PERSONA ENGINE

As stated above, the Persona Engine takes as input presentation scripts and displays the corresponding presentation. However, in contrast to other display components for media objects (e.g. video or audio players, graphics viewers, etc.) the output of the Persona Engine is not only determined by the directives (i.e., presentation tasks) specified in the script. Rather, the behavior of the animated character follows the equation:

Persona behavior := **directives** + **self-behavior**

Such self-behaviors are indispensible in order to increase the Persona's vividness and believability. Though it is certainly possible to include appropriate instructions directly in the presentation script, our approach has an important advantage. From a conceptual point of view, we consider it more adequate since a clear borderline is drawn between a *"what to present"- part* which is determined by the application, and a *"how to present"-part* which, to a certain extent, depends on the particular presenter. From the practical perspective, the approach considerably facilitates script production.

2.1. Defining Behaviors

The primary task of the Persona is to execute presentation acts, such as pointing, speaking or expressing emotions. These acts have to be coordinated with the application-independent self-behavior of the Persona, which includes:

- *Idle-time Actions*
  To ensure that the Persona exhibits life-like qualities, it has to stay "alive" even in an idle phase, for instance, when waiting for material to be delivered by the generators or the retrieval components. Typical idle-time actions are breathing or tipping with a foot.

- *Reactive Behaviors*
  The Persona should be able to react to external events, such as user interactions, immediately. For instance, if the user moves the window the Persona is currently pointing to, the consistency of the pointing gesture has to be restored immediately, e.g. by a prolongation of the pointing stick.

- *Navigation Acts*
  The kind of navigation act depends on the chosen metaphor. For example, human-like agents like the Persona walk or jump to an appropriate position on the screen while agents like Microsoft's parrot Peedy fly (cf. [Ball et al. 97]).

To facilitate the definition of Persona behaviors, we have developed a declarative specification language. For instance, the following definition specifies the pre- and post-conditions for the action: *bottomupjumping1*.

```
(defprimitive bottomupjumping1
    :pre ((leftarm standard)(rightarm standard)
          (icon noicon)(bodydir front)
          (bodypos stand)(stick off))
    :post ((posy -= 1))
    :gesture 42)
```

The action can only be performed if both arms are in a standard position, the Persona is not iconified, faces the user, is standing and doesn't hold a stick. If this is the case, the image sequence associated with the action (*:gesture 42*) is played and the state of the world is updated as indicated in the post conditions *:post*. Otherwise, the system tries to achieve the desired preconditions by executing further actions.

While primitive actions like *bottomupjumping1* are directly associated with an image sequence, complex actions are decomposed of several inferior actions. An example of a complex action is:

```
(defactionseq MoveUp
    :pre ((icon noicon)(bodydir front)
          (leftarm standard)(rightarm standard)
          (bodypos stand)(stick off))
    :prim startbottomupjumping
    :while ((posy ≠ target) :prim bottomupjumping)
    :prim endbottomupjumping)
```

This definition specifies a jump to a given target *target*. The preconditions of this action coincide with the preconditions of *bottomupjumping1*. If they are satisfied, the Persona starts with the jump (*startbottomupjumping*) and continues to jump until it reaches the target (*(posy ≠ target)*). After that, it finishes the jump (*endbottomupjumping*).

2.2. Compiling Behaviors

Since animations have to be performed in realtime (and our system should run on ordinary PCs/workstations as well), it's not feasible to decompose actions into animation sequences at runtime. Following [Ball et al. 97], we have developed a multi-pass compiler that enables the automated generation of a state machine from these declarations which is converted to efficient machine code (cf. Fig. 3).

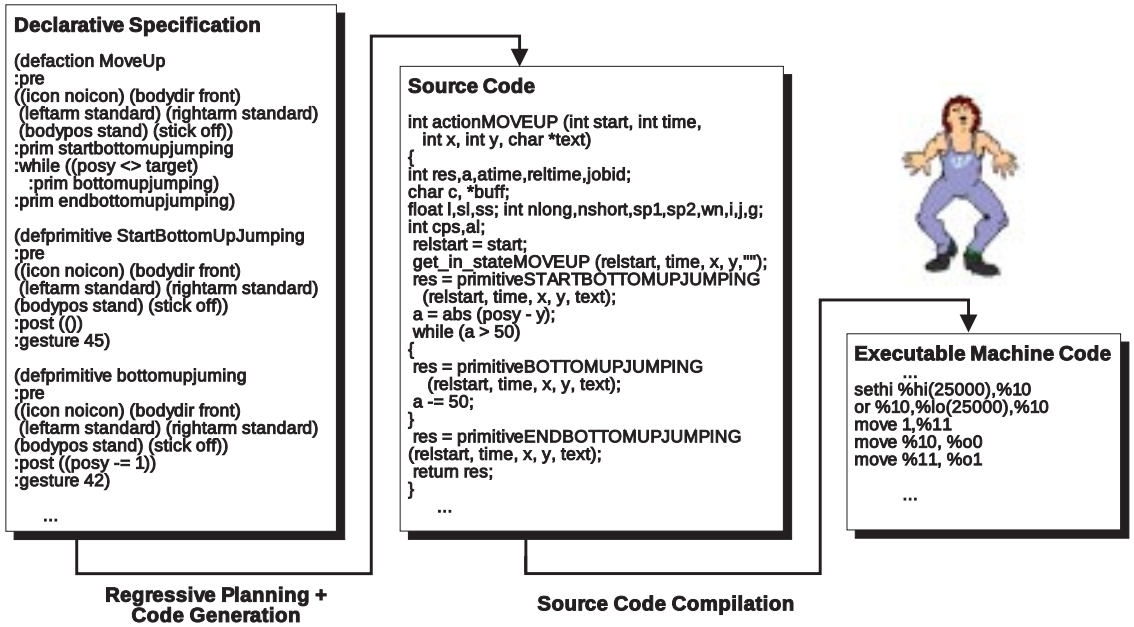

Figure 3: Compilation of Persona Actions

The compiled state machine forms the so-called *behavior monitor*. Summing up, the behavior monitor accomplishes the following tasks:

- It decomposes complex presentation tasks into elementary postures that correspond to a single Persona frame (e.g. stored as a pixmap) or to an uninterruptable image sequence.

- It ensures that necessary preconditions for the execution of primitive actions are satisfied.
- It updates the internal state of the Persona after the execution of a primitive posture.
- It augments the Persona's presentation behaviors by believability-enhancing behaviors, such as idle-time acts.

The postures determined by the behavior monitor are forwarded to a *character composer* which selects the corresponding frames (video frames or drawn images) from an indexed data-base, and forwards the display commands to the window system.

3. PLANNING PRESENTATION SCRIPTS

In order to build up multimedia presentations automatically, we have developed principles for describing the structure of coherent text-picture combinations (cf. [André & Rist 93]). Essentially, these principles are based on

- the generalization of speech act theory (see [Searle 80]) to the broader context of communication with multiple media, and
- the extension of RST (Rhetorical Structure Theory, [Mann & Thompson 87] to capture relations that occur not only between presentation parts realized within a particular medium but also those between parts conveyed by different media.

Since we consider the generation of multimedia presentations as a goal-directed activity, it seemed appropriate to implement a goal-driven, top-down planning approach. The planning component receives as input a communicative goal (for instance, *the user should be able to localize the internal parts of a modem*) and a set of generation parameters, such as target group, presentation objective, resource limitations, and target language. The task of the component is to select parts of a knowledge base and to transform them into a multimedia presentation structure (see Fig. 4). Whereas the root node of such a presentation structure corresponds to a more or less complex communicative goal, such as *describing a technical device*, the leaf nodes are elementary retrieval or generation acts, currently for text, graphics, animations and gestures. Design knowledge is represented by so-called presentation strategies which encode knowledge on: (1) how to select relevant content, (2) how to structure selected content, and finally (3) which medium to use for conveying a content.

In order to cope with the dynamic nature of presentations made by an animated agent, several extensions became necessary:

- *the specification of qualitative and quantitative temporal constraints in the presentation strategies*
  Qualitative constraints are represented in an "Allen-style" fashion which allows for the specification of thirteen temporal relationships between two named intervals, e.g. *(Speak1 (During) Point2)*. Quantitative constraints appear as metric (in)equalities, e.g. *(5 ≤ Duration Point2)*. For more details, see [André & Rist 96].

- *the development of a mechanism for building up presentation schedules*
  To temporally coordinate presentation acts, the presentation planner has been combined with a temporal reasoner which is based on MATS (Metric/Allen Time System, cf. [Kautz & Ladkin 91]). During the presentation planning process, PPP determines the transitive closure over all qualitative constraints and computes numeric ranges over interval endpoints and their difference. After that, a schedule is built up by resolving all disjunctions and computing a total temporal order (see Fig. 4). Since the temporal behavior of presentation acts may be unpredictable at design time, the schedule will be refined at runtime by adding new metric constraints to the constraint network.

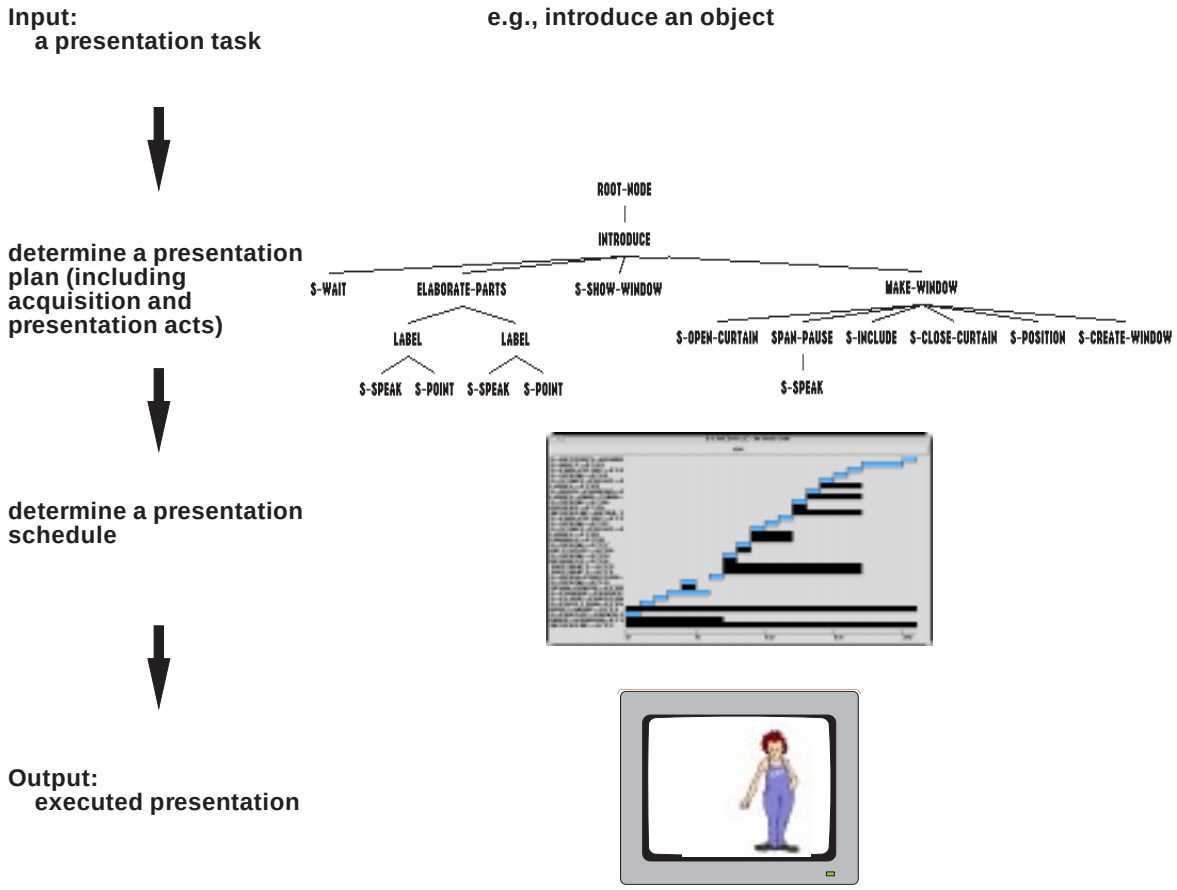

Figure 4: The Presentation Planning Process

4. TECHNICAL REALIZATION

Implementations of the PPP Persona Engine are currently available for Unix platforms running X11, and Java-enhanced WWW-browsers.

In the X-version, the Persona server builds upon the X11 library and the X11-Shape extension (cf. [Packard 89]) which allows for the definition of non-rectangular regions in an otherwise invisible window. To use this feature for the graphical realization of the Persona, we first create an invisible window that covers the whole screen. This ensures

that in case this window lies on top of the window stack, all other windows below still remain activated for mouse and keyboard input. Second, single postures of the Persona as well as the requisites such as pointing sticks are drawn into the invisible window. Since items drawn into the invisible window remain invisible unless the regions they cover were masked before, we create bitmasks of the same shape as the items. For the Persona postures (images or video frames), these masks are computed in a preprocessing phase, and are stored in the X-Server's memory. Bitmasks for other items, e.g. pointing sticks, are computed only on demand at runtime.

For WWW applications, the user downloads an instance of the Persona server in form of a Java applet. In contrast to the X-windows version, the spatial action ratio of the Persona is restricted to the Java canvas of the web-page. The animation is simply done by bitplotting the corresponding frames onto the canvas. Since Java supports transparency no additional masking is required as in the X version.

The Persona Engine has been implemented in Java and C++. It relies on about 250 frames for each Persona. Currently, we use two cartoon personas and three real personas composed of grabbed video material. To control the behavior of the personas, more than 150 different behaviors have been defined. The presentation planner, the temporal reasoner and the Persona Compiler have been implemented in Allegro Common Lisp. To plan presentation scripts, about 70 presentation strategies have been defined.

5. CONCLUSION

While animated user interface agents have been proposed by several other authors, a distinguishing new feature of our approach is that the agent not only performs tasks requested by some application clients, but also implements a basic behaviour independent of the applications it serves. This basic behaviour comprises *idle-time actions*, and *immediate reactions* to events occurring at the user interface. Both action types have to be supported, in order to obtain a lively and appealing presentation agent. Due to a built-in mechanism for action specialization and decomposition application clients can request presentation scripts at a high-level of abstraction. Since the manual scripting of agent behaviors is tedious and error-prone, we also addressed the automated generation of presentation scripts. The novelty of our system is that it not only designs the multimedia material (such as text paragraphs and graphics), but also plans how to present the material in a temporally coordinated manner. This has been achieved by combining an AI planning approach with a module for temporal reasoning.

An empirical study of our system revealed a strong affective impact of the Persona. Our subjects perceived the Persona as being helpful and entertaining. Furthermore, they rated learning tasks presented by the Persona as less difficult and demanding than presentations without a life-like character.

ACKNOWLEDGMENTS

This work has been supported by the German Federal Ministry of Education, Science, Research and Technology (BMBF) under the contracts ITW 9400 7 and 9701 0. We would like to thank Peter Rist for drawing the cartoons, H.-J. Profitlich and M. Metzler for the development of RAT and Frank Biringer for implementing the Persona Compiler.

REFERENCES

[Adelson 92] B. **Adelson**. *Evocative Agents and Multi-Media Interface Design.* In: Proc. of the UIST'92 (ACM SIGGRAPH Symp. on User Interface Software and Technology), pp. 351–356, Monterey, CA, U.S.A., 1992.

[André & Rist 93] E. **André** and T. **Rist**. *The Design of Illustrated Documents as a Planning Task.* In: M. Maybury (ed.), Intelligent Multimedia Interfaces, pp. 94–116. AAAI Press, 1993.

[André & Rist 96] E. **André** and T. **Rist**. *Coping with Temporal Constraints in Multimedia Presentation Planning.* In: Proc. of AAAI-96, volume 1, pp. 142–147, Portland, Oregon, 1996.

[André et al. 96] E. **André**, J. **Müller**, and T. **Rist**. *WIP/PPP: Automatic Generation of Personalized Multimedia Presentations.* In: ACM Multimedia 96, pp. 407–408. ACM Press, November 1996. Demo.

[André et al. 97] E. **André**, J. **Müller**, and T. **Rist**. *WebPersona: A Life-Like Presentation Agent for the World-Wide Web.* In: Proc. of the IJCAI-97 Workshop on Animated Interface Agents: Making them Intelligent, Nagoya, 1997. also available via http://www.dfki.uni-sb.de/~andre/webpersona/webpersona.html.

[Ball et al. 97] G. **Ball**, D. **Ling**, D. **Kurlander**, J. **Miller**, D. **Pugh**, T. **Skelly**, A. **Stankosky**, D. **Thiel**, M. **van Dantzich**, and T. **Wax**. *Lifelike Computer Characters: the Persona project at Microsoft.* In: J.M. Bradshaw (ed.), Software Agents. AAAI/MIT Press, Menlo Park, CA, 1997.

[Kautz & Ladkin 91] H. A. **Kautz** and P. B. **Ladkin**. *Integrating Metric and Qualitative Temporal Reasoning.* In: Proc. of AAAI-91, pp. 241–246, 1991.

[Lester et al. 97] J. **Lester**, J.L. **Voerman**, S.G. **Towns**, and C.B. **Callaway**. *Cosmo: A Life-Like Animated Pedagogical Agent with Deictic Believability.* In: E. André (ed.), Proc. of the IJCAI-97 Workshop on Animated Interface Agents: Making them Intelligent, Nagoya, 1997.

[Mann & Thompson 87] W. C. **Mann** and S. A. **Thompson**. *Rhetorical Structure Theory: Description and Construction of Text Structures.* In: G. Kempen (ed.), Natural Language Generation: New Results in Artificial Intelligence, Psychology, and Linguistics, pp. 85–95. Dordrecht: Martinus Nijhoff Publishers, 1987.

[Packard 89] K. **Packard**. *X11 Nonrectangular Window Shape Extension Version 1.0, X11 R5.* Technical report, MIT X Consortium, MIT, Cambridge, Massachusetts, 1989.

[Rickel & Johnson 97] J. **Rickel** and W.L. **Johnson**. *Integrating pedagogical capabilities in a virtual environment agent.* In: Proceedings of the First International Conference on Autonomous Agents, Marina del Rey, 1997.

[Searle 80] J.R. **Searle**. *Speech Acts: An Essay in the Philosophy of Language.* Cambridge, England: Cambridge University Press, 1980.