

Chapter 28

FLoReS: A Forward Looking, Reward Seeking, Dialogue Manager

Fabrizio Morbini, David DeVault, Kenji Sagae, Jillian Gerten, Angela Nazarian, and David Traum

Abstract We present FLoReS, a new information-state-based dialogue manager, making use of forward inference, local dialogue structure, and plan operators representing subdialogue structure. The aim is to support both advanced, flexible, mixed initiative interaction and efficient policy creation by domain experts. The dialogue manager has been used for two characters in the SimCoach project and is currently being used in several related projects. We present the design of the dialogue manager and preliminary comparative evaluation with a previous system that uses a more conventional state chart dialogue manager.

28.1 Introduction

In this paper we present a new information-state-based dialogue manager called FLoReS (Forward Looking, Reward Seeking). FLoReS has been designed to provide flexible, mixed initiative interaction with users, while at the same time supporting the efficient creation of dialogue policies by domain experts.

The development of new frameworks and techniques that can streamline the creation of effective dialogue policies is an important issue for spoken dialogue systems. One common approach in practical applications is to adopt a strong system initiative design, in which the system steers the conversation and generally prompts the user for what to say at each point. System initiative policies can be relatively easy for authors to create and can also simplify the system's language understanding problem at each point. For authoring, a system that will behave as desired can be specified with simple structures, such as a call flow graph [12] and branching

F. Morbini (✉) • D. DeVault • K. Sagae • J. Gerten • A. Nazarian • D. Traum
Institute for Creative Technologies, University of Southern California,
Los Angeles, CA 90094, USA
e-mail: morbini@ict.usc.edu; devault@ict.usc.edu; sagae@ict.usc.edu; gerten@ict.usc.edu;
nazarian@ict.usc.edu; traum@ict.usc.edu

narrative for interactive games [15]. A strong system initiative system also reduces the action state space and generally reduces the perplexity of the understanding process, since user actions are only allowed at certain points in the dialogue and usually are limited to a reduced set of options.

Strong system initiative systems can work well if the limited options available to the user are what the user wants to do, but can be problematic otherwise, especially if the user has a choice of whether or not to use the system. In particular, this approach may not be well suited to a virtual human application like SimCoach [13]. In the SimCoach system, which we describe in Sect. 28.2, a virtual human is designed to be freely available to converse with a user population on the web, but its users may choose not to use the system at all if it does not respond to their needs and alleviate their concerns in the dialogue in a direct and efficient way.

At the other extreme, pure user initiative systems allow the user to say anything at any time, but have fairly simple dialogue policies, essentially just reacting to each user utterance individually, in a way that is not very sensitive to context, e.g., [8]. These systems can be easy for authors to design and can work well when the user is naturally in charge, such as in a command and control system, or interviewing a character [9], but may not be suitable for situations like SimCoach, in which the character should sometimes (but not always) take the initiative and ask the user questions, or generally, when mixed initiative is desired. The Tactical Questioning system architecture and authoring environment [4] provides an alternative for systems in which the user is generally in charge and asking the system questions, but policy rules can be easily authored to provide some context sensitivity to the system's responses.

True mixed initiative is notoriously difficult for a manually constructed call flow graph, in which the same sorts of options may appear in many places, but the system might want to take different actions, depending on local utilities. An alternative approach is for authors to develop complex hand-authored rules to achieve a dialogue policy with the desired mixed initiative behavior. These rules will govern system initiative decisions and information state updates [7]; however, in practice, this development often involves a number of formal modeling tasks that require substantial dialogue modeling expertise as well as programming skills, which many potential dialogue system authors and domain experts do not possess [1]. Reinforcement learning approaches [2, 18] can be very useful at learning local policy optimizations from data rather than hand-authored rules, but they require large amounts of training data, possibly using simulated users [5]. While this approach can take some of the burden off the author, it also removes some of the control, which can be undesirable [11]. Moreover, it is very difficult to apply for large state-spaces.

Our approach, embodied in the FLoReS dialogue manager, is to create a forward looking, reward seeking agent with support for complex, mixed initiative dialogue interaction and rich dialogue policy authoring. FLoReS combines several methods of dialogue reasoning, to promote the twin goals of flexible, mixed initiative interaction and tractable authoring by domain experts and creative authors. Authoring involves design of local subdialogue networks (called *operators*) for

specific conversation topics. Within a subdialogue network, authors can craft the specific structure of interaction. The higher-level structure of the dialogue, which determines the flow of initiative and topics as the dialogue progresses, is determined at run-time by the dialogue manager. This is realized in a reward-sensitive plan-based paradigm.¹ The subdialogues are given preconditions and effects, to decide where they may be applicable, and also qualitative reward categories (goals), which can be assigned to quantitative reward values. The dialogue manager locally optimizes its policy decisions by calculating the highest overall expected reward for the best sequence of subdialogues from a given point.

The rest of this paper is structured as follows. Section 28.2 describes two of the systems currently using FLoReS. Section 28.3 describes the dialogue manager components, while Sect. 28.4 describes the dialogue manager execution. Section 28.5 describes the evaluation (currently still in progress), comparing the FLoReS version of Bill Ford with a previous version of the character, using a more traditional state chart and directed dialogue, and current limitations of FLoReS. We conclude in Sect. 28.6, describing current work.

28.2 Virtual Human Applications Using FLoReS

SimCoach [13] is the first system to use FLoReS for dialogue management. One example of a SimCoach character, named Bill Ford, is shown in Fig. 28.1. SimCoach is motivated by the challenge of empowering troops and their significant

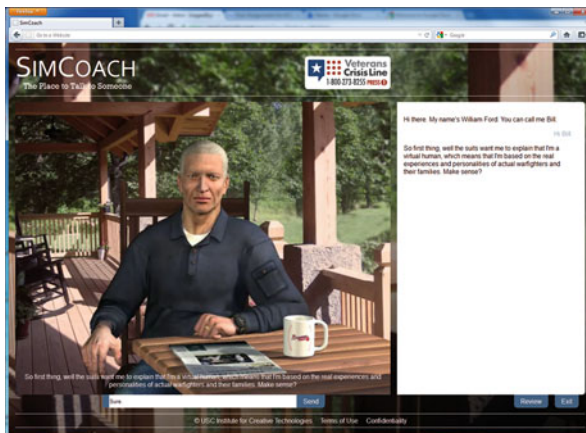


Fig. 28.1 Bill Ford, a SimCoach character. SimCoach virtual humans are accessible through a web browser. The user enters natural language input in the text field on the bottom of the screen. The SimCoach responds with text, speech, and character animation. The text area to the *right* shows a transcript of the dialogue

¹This approach is similar at a high level to [10], though they do not address its use in dialogue planning specifically.

others in regard to their healthcare, especially with respect to issues related to the psychological toll of military deployment. The SimCoach project is developing virtual human dialogue systems to engage soldiers and their families who might otherwise not seek help for possible mental and behavioral health issues. The reluctance to seek help may be due to stigma or lack of awareness. SimCoach virtual humans are not designed to act as therapists or dispense medical advice, but rather to encourage users to explore available options and seek treatment when needed. To achieve this goal, we aim to provide a safe and anonymous environment, where users can express their concerns to an artificial conversational partner without fear of judgment or possible repercussions.

SimCoach virtual humans present a rich test case for the FLoReS dialogue manager. To be convincing to a military user population, aspects of the system’s content need to be authored by someone familiar with the military domain. At the same time, psychological expertise is needed in order to offer appropriate suggestions and helpful information. To do this, the system assesses the user’s needs and potential issues using variants of standard psychometric questionnaires such as PCL, the PTSD checklist [17]; the use of such standard instruments requires psychological expertise for authoring of both policies and utterances. Authoring therefore needs to be feasible by military and psychological experts, who cannot be expected to have detailed knowledge of dialogue modeling and natural language processing.

Finally, the dialogue system must be able to take initiative when trying to collect the information it needs to help the user, such as responses to questionnaire questions. But it also must respond appropriately when the user takes initiative. Given the sensitive nature of the topics covered in the system, responding poorly to user initiative is likely to severely compromise the system’s ability to accomplish its goal to help the user. Mixed initiative is therefore key to the design goals of SimCoach.

We provide an excerpt of a conversation with Bill Ford in the following table. The user’s utterances are in italics, and the subdialogue networks are highlighted in gray boxes. The side notes in the figure summarize FLoReS’ decision making for each received user input and will be discussed further as we present FLoReS’s design, below. An example of the user taking the initiative is the user’s utterance *Is this confidential?* In this domain, due to stigma, users often worry that their conversation could be revealed to their military chain of command, and they might refuse to answer questions such as the system’s question, *Have you been deployed?*, until they are assured of the confidentiality of their conversation.

Dialogue transcript	Notes
Greetings and Introduction	This network is activated by the initial event sent when a new conversation is started. No other networks are available to handle that event.
...What do you think?	This is a possible symptom and is stored in the information state
<i>User: I'm always tired.</i>	

Continued on next page

continued from previous page

Dialogue transcript	Notes
<p>Feedback</p> <p>This must be hard for you.</p>	<p>Here the system could pick the prequestionnaire network you see later in the conversation or give empathetic feedback to the symptom just mentioned. The DM selects to give feedback because the reward is available only if the network is used immediately after a symptom and that path in the search graph achieves a higher expected reward.</p>
<p>Deployed QA</p> <p>Have you been deployed?</p>	<p>In the SimCoach domain, being tired could be an indicator of two conditions; to disambiguate them, the system needs to know whether the user has been deployed. Even though asking that kind of personal question to the user has a negative reward, the system selects this action because it enables future networks with high reward.</p>
<p><i>User: Is this confidential?</i></p> <p>Confidentiality QA</p> <p>All your information will be kept extremely confidential. I'm not interested in your name, just in how I can help. Did that answer your questions?</p> <p><i>User: Yes it did.</i></p> <p>Great.</p>	<p>Here the user doesn't respond to the system question, but asks his own question. The system computes the expected rewards of answering the user and that of ignoring the question. In this case, answering the user comes out on top.</p>
<p>Deployed QA</p> <p>Have you been deployed?</p> <p><i>User: Yes.</i></p>	<p>After the user completes the confidentiality network, the system recomputes the rewards and decides to ask one more time the deployment question to the user.</p>
<p>Prequestionnaire</p> <p>Can you tell me a bit more about that?</p> <p><i>User: Not really.</i></p> <p>Okay, you may not be real excited about it, but I've got some questions to ask you so I can help you better. First off, has your sleeping been troubled lately?</p> <p><i>User: Sure I have nightmares</i></p> <p>I know you mentioned it before, but I want to make sure I got it right. You're having nightmares?</p> <p><i>User: Yes</i></p> <p>• • •</p>	<p>Answering the deployment question disambiguates the initial symptom and that enables the prequestionnaire network. Here you can see the event listeners in action to update the information state when the user replies to the question about sleeping problems with having nightmares. That reply modifies the way in which the system asks the next question about nightmares to a confirmation.</p>



Fig. 28.2 Ellie is a virtual character similar in purpose to Bill Ford but equipped with multimodal sensing to improve both the interaction with the user and the detection of possible health problems

Another virtual human using the FLoReS DM is Ellie (see Fig. 28.2). Ellie is in an early stage of development, and is designed to address similar problems as those targeted by the SimCoach project. However, Ellie will use speech input and multimodal sensing using a Kinect sensor and cameras in order to have a more compelling conversation with the user, and also to more effectively detect signs of possible health problems (e.g., signs of depression and post-traumatic stress disorder). Information extracted both from the audio (e.g., recognized speech) and visual signals (e.g., movements, facial expression, gaze) will be integrated and used by FLoReS to select what the virtual human will do.

28.3 FLoReS Dialogue Manager Components

In this section, we describe the main components in the FLoReS dialogue manager (DM), which are:

- An **information state** [16], including information about what has happened in the dialogue so far, and information relevant for continuing the dialogue.
- A set of **inference rules** that allows the system to add new knowledge to its information state, based on logical reasoning.
- An **event handling** system, that allows the information state to be updated based on user input, system action, or other classes of author-defined events (such as system timeouts).
- A set of **operators**, that allow the system to model local and global dialogue structure and plan the best continuation.

We will discuss dialogue policy execution in Sect. 28.4.

28.3.1 *Inference Rules*

FLoReS allows the dialogue system author to specify a set of implications that can be used to infer new knowledge given the current information state. For example, toward the end of the above dialogue excerpt, the user says *Sure i have nightmares*. In the information state, there is a variable that captures whether the user has nightmares, and another variable that captures whether the user has sleeping problems more generally. An author can define an implication that if the user has nightmares, then they also have sleeping problems. These implications are generally expressed in the form of if-then-else conditionals that can test and set the values of information state variables. The implications are evaluated, and the information state is updated, each time the information state is changed.

28.3.2 *Event Handling and Dialogue Acts*

Most updates to the information state are ultimately triggered by *events*. There are several different kinds of events, including those corresponding to user input (produced by the system's NLU module), system generated events (corresponding to decisions and actions made by the system), and external events (such as timeouts or other perceptual information).

Events received by FLoReS have a name and can have a content. For example, for user events, the name is a dialogue act, and the content is a set of key-value pairs representing the semantic content of the utterance.

Event listeners have a matching expression and a resulting action. The matching expression defines the events being listened for, which can include full event names or a regular expression, focusing on one or more fields of the name. The resulting action is an assertion of a variable value or an if-then-else conditional. For example, when the user says *I'm always tired*. in the example above, an event handler matches the recognized NLU dialogue act (which is represented as `answer.observable.tired`), and increments a counter representing the number of times the user has mentioned being tired.

28.3.3 *Operators*

The main part of a FLoReS dialogue policy is a set of subdialogue networks, which are formalized as *operators*. The gray boxes in the above dialogue excerpt indicate the different operators that are active during the dialogue. Our definition of operators is motivated by the desire to encourage re-usability of subdialogues across multiple policies, allow authors to craft short portions of conversations, and maintain local coherence. Operators represent local dialogue structure, and can also be thought of

as reusable subdialogues. Each operator contains a subdialogue structure, composed of a tree of system and/or user actions and resulting states.² A system action can be (1) an update to the information state, (2) a generated utterance to perform back to the user (using speech and animation), or (3) a command to send a particular event to the DM itself (which is later handled by an event handler).

Each state within the subdialogue can be associated with a *goal* and thus a *reward* for reaching that state.³ The system's goals, such as providing the user with relevant information, are the main factor used to decide what to do when there is more than one applicable operator. Each goal is associated in the information state with a specific numeric reward value. Goals are attached to states within the subdialogue, and their corresponding rewards are thus associated with those states.

Like AI planning operators,⁴ operators can have preconditions and effects. Effects specify changes to the information state and can occur at any state within the operator. The preconditions define when an operator can be activated. Preconditions are divided into three types (an operator can have any number of each type):

1. **System initiative** preconditions specify that the system may elect to activate this operator on system initiative, regardless of what the user has recently said.
2. **User initiative** preconditions specify when an operator can be used as part of the processing of a user utterance.⁵
3. **Re-entrance** preconditions enable the DM to resume an operator that was previously interrupted (e.g., by user initiative).

Because an operator can contain a complex subdialogue, each precondition, in addition to specifying when the operator can be initiated (or continued), also defines the state at which it enters the operator's subdialogue tree.

Operators can also be associated with topics, and preconditions can test the current topic (for example) in order to decide whether an operator can be used.

28.4 Dialogue Policy Execution

When each event is received, the event listeners are checked and matching listeners execute their resulting actions, leading to updates to the information state. For each change to the information state, the inference rules are evaluated repeatedly until the information state is stable.

Then the dialogue manager decides which operator to use to best deal with the received event. At any point, the set of available operators is divided in three:

²The tree structure excludes cycles within subdialogue networks.

³Currently each operator must include at least one reward, somewhere in the subdialogue.

⁴Our operators are nonparametric like propositional STRIPS operators [3].

⁵If an operator has a satisfied user initiative precondition for a particular event *e*, the operator is said to *handle e*.

(1) the currently active operator (if any), (2) a set of paused operators; these operators were once active but have been interrupted before completion and put in a paused state, and (3) the set of inactive operators: the remaining operators that are neither active nor paused.

If the current active operator can handle the received event, the DM just continues its execution, traversing the subdialogue of the operator to the next state. An example of this is in the **Deployed QA** operator in the above dialogue excerpt, where the user's answer *Yes.* is handled by the active operator.

Otherwise, the dialogue manager computes the expected rewards for the following cases:

- *Ignore the received event:* here the dialogue manager searches for the most promising system initiative operators. Two sub-searches are executed: one considers keeping the current active operator as it is, and the other considers switching to any of the other system initiative operators.
- *Handle the received event:* here the dialogue manager searches for the most promising operator among those that are paused or inactive and that handle the received event.

An example in the above dialogue excerpt arises with the user's utterance of *Is this confidential?*, which cannot be handled by the active **Deployed QA** operator. The dialogue manager then switches to the inactive **Confidentiality QA** operator, which can handle it.

The expected reward of a given operator, O , starting from the current information state I , is computed by simulating future dialogues that can happen if O is activated. The simulation is executed breadth first and builds a graph where the nodes are possible information states and the edges are operators.

Because an operator represents a subdialogue, it can produce multiple resulting information states depending on the subdialogue branch that is traversed. The multiple possible information states are weighted by the probability with which each state can be reached.⁶

To compute the possible information states that an operator can produce when executed, the DM also considers the updates executed by event listeners. The simulation constructs a graph instead of a tree because the nodes corresponding to information states with the same content are merged into a single node. Arcs that would cause loops are not added. The simulation that builds this graph of possible information states continues until a termination criterion is satisfied. Currently this criterion is based on a maximum depth of the graph and a timeout. The expected reward is computed for all operators in this graph whose head is the current information state (i.e., the root node) and the operator with the highest expected reward is made active and executed. The currently active operator is made paused (if possible, otherwise is made inactive).

⁶Because multiple branches can produce the same final information state, this weight is not simply the uniform weight obtained by $1/|leaves|$.

The formula to compute the expected utility of operator O_i in information state I is

$$E[O_i, I] = \sum_{I_i \in I_r} (\alpha \cdot P(I_i) \cdot R(O_i, I_i) + \operatorname{argmax}_O (E[O, I_i]))$$

where α is a discount factor for future rewards; I_r is the set of possible information states that can be reached from I by executing the subdialogue contained in O_i . $P(I_i)$ is the probability of reaching the information state I_i from I when executing O_i . Currently this probability is just based on uniformly distributing the probability across all paths of the subdialogue of a given operator and merging the paths that produce the same final information state. $R(O, I_i)$ is the reward realized by traversing the path in the subdialogue associated with the operator O that produces the final information state I_i . The final term in the formula calculates the maximum expected reward from operators that could be activated after operator O_i . This allows the immediate selection of operators that have low immediate reward, but whose effects enable higher reward operators in the future. As noted in the above dialogue excerpt, an example of this is when the system asks *Have you been deployed?* in the excerpt.

28.4.1 Mixed Initiative

This architecture allows for mixed initiative and opportunistic action selection based on reward. When the user takes the initiative, as in asking *Is this confidential?* in the above dialogue excerpt, the system can choose whether to follow the user's initiative or not according to the expected rewards associated with each of these options. The system can also take the initiative, as when the system asks the user *Have you been deployed?* in the above dialogue excerpt. Which speaker will have the initiative at each point does not need to be hard coded, as in a call flow graph, but rather is determined by the system's expected reward calculation at each point.

28.5 Evaluation and Discussion

We have explored the various features and design goals for this dialogue manager during the creation of the Bill Ford policy and testing of the system.

We explored the feasibility of nonexperts to author nontrivial dialogue policies by working with a creative writer to define Bill Ford's dialogue policy. The resulting Bill Ford policy is composed of 285 operators, of which about 150 were automatically generated (they are operators that answer simple factual questions). This policy was built in about 3 months by one creative writer, with help from the first author.

To evaluate the FLoReS dialogue manager, we are currently conducting a comparative study between the current version of the SimCoach character Bill Ford,

using FLoReS, and an older version of Bill Ford based on a dialogue policy encoded as a finite-state machine. The policy for the older version was encoded in SCXML⁷, and focused strongly on system initiative.

To find appropriate users for our study, we selected ROTC (Reserve Officer Training Corps) students in four college campuses in Southern California. Although ROTC members are not the target user population for the SimCoach dialogue system, they serve as a suitable approximation due to their military background and familiarity with the issues faced by our target users. We have so far collected data from over 30 users in these ROTC programs. Each user interacted with one version of Bill Ford (either the one using FLoReS or the one using a simpler finite-state machine), and rated several aspects of the interaction with Bill Ford. Following the methodology in [14], users rated Bill on these items using values from 1 (strongly disagree) to 7 (strongly agree).

Although we have not yet collected data from enough users to determine statistically significant differences between the two systems, we note trends that for the FLoReS version, users had higher ratings for “Bill understood what I said” and “Bill let me talk about what I wanted to talk about” (addressing a user sense that they can take the initiative), but also, unexpectedly “Bill asked too many questions for no reason”. Further analysis will examine the role of aspects of the FLoReS dialogue manager versus the specific subdialogues and reward structures used.

In terms of limitations, while we were successful in enabling the new Bill Ford dialogue policy to be almost completely designed by non-programmers, it remains a hard task that requires multiple iterations, and occasional intervention from dialogue system experts, to get a dialogue policy to work as desired. One of the main problems we noticed is that even though FLoReS allows authors to relax preconditions and allow ordering to be determined at run-time based on rewards, they still often preferred to use logical preconditions. This seems to be caused by the fact that predicting the behavior based on rewards is challenging, whereas logical conditions, although hard to write clearly, make it easier to foresee when an operator will be executed. We also observed that deciding on the specific reward value to associate to certain goals was a nontrivial task for our author.

Another limitation is that the simulation currently can make some inaccurate predictions, due to time and resource limits in the search process.

28.6 Conclusion

We have described FLoReS, a mixed initiative, information state, and plan-based DM with opportunistic action selection based on expected rewards that supports non-expert authoring. We presented examples to demonstrate its features, a summary

⁷ <http://www.w3.org/TR/scxml/>

of our experience enabling a creative writer to develop a FLoReS dialogue policy, and encouraging preliminary results from a user evaluation.

Next we plan to work on ways to simplify or eliminate the need to assign numeric rewards by possibly using partial ordering constraints on operators, or inverse reinforcement learning to set the reward values. We also plan to further develop and study the efficacy of a dialogue policy visual editor, including developing a library of templates to quickly build common operators.

Finally, the FLoReS dialogue manager will be made available for other researchers through the ICT Virtual Human Toolkit [6]⁸.

References

1. DeVault, D., Leuski, A., Sagae, K.: Toward learning and evaluation of dialogue policies with text examples. In: 12th annual SIGdial Meeting on Discourse and Dialogue (2011)
2. English, M., Heeman, P.: Learning mixed initiative dialogue strategies by using reinforcement learning on both conversants. In: HLT-EMNLP (2005)
3. Fikes, R.E., Nilsson, N.J.: Strips: A new approach to the application of theorem proving to problemwilliams-young:2007 solving. *Artif. Intell.* **2**(3–4), 189–208 (1971). DOI 10.1016/0004-3702(71)90010-5. URL <http://www.sciencedirect.com/science/article/pii/0004370271900105>
4. Gandhe, S., Whitman, N., Traum, D.R., Artstein, R.: An integrated authoring tool for tactical questioning dialogue systems. In: 6th Workshop on Knowledge and Reasoning in Practical Dialogue Systems. Pasadena (2009). URL <http://people.ict.usc.edu/%7ETraum/Papers/kprd09authoring.pdf>
5. Georgila, K., Henderson, J., Lemon, O.: User simulation for spoken dialogue systems: Learning and evaluation. In: Interspeech, Pittsburgh (2006)
6. Kenny, P.G., Hartholt, A., Gratch, J., Swartout, W., Traum, D., Marsella, S.C., Piepol, D.: Building interactive virtual humans for training environments. In: Proceedings of Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC). Orlando (2007)
7. Larsson, S., Traum, D.: Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Nat. Lang. Eng.* **6**, 323–340 (2000)
8. Leuski, A., Patel, R., Traum, D., Kennedy, B.: Building effective question answering characters. In: Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue, pp. 18–27 (2006)
9. Leuski, A., Traum, D.R.: Practical language processing for virtual humans. In: Twenty-Second Annual Conference on Innovative Applications of Artificial Intelligence (IAAI-10) (2010). URL <http://www.ict.usc.edu/%7ELeuski/publications/papers/iaai-10-npceditor.pdf>
10. Liu, D., Schubert, L.K.: Combining self-motivation with logical planning and inference in a reward-seeking agent. In: Filipe, J., Fred, A.L.N., Sharp, B. (eds.) ICAART (2), pp. 257–263. INSTICC Press (2010)
11. Paek, T., Pieraccini, R.: Automating spoken dialogue management design using machine learning: An industry perspective. *Speech Comm.* **50**(8–9), 716–729 (2008). DOI 10.1016/j.specom.2008.03.010. Evaluating new methods and models for advanced speech-based interactive systems
12. Pieraccini, R., Huerta, J.: Where do we go from here? Research and commercial spoken dialog systems. In: Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue. Lisbon (2005). URL http://www.sigdial.org/workshops/workshop6/proceedings/pdf/65-SigDial2005_8.pdf

⁸<http://vhtoolkit.ict.usc.edu>

13. Rizzo, A.A., Lange, B., Buckwalter, J.G., Forbell, E., Kim, J., Sagae, K., Williams, J., Rothbaum, B.O., Difiede, J., Reger, G., Parsons, T., Kenny, P.: An intelligent virtual human system for providing healthcare information and support. In: *Studies in Health Technology and Informatics*, Victoria (2011)
14. Silvervarg, A., Jönsson, A.: Subjective and objective evaluation of conversational agents in learning environments for young teenagers. In: *7th IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, Barcelona (2011)
15. Tavinor, G.: *The art of videogames. New Directions in Aesthetics*. Wiley-Blackwell, Oxford (2009)
16. Traum, D., Larsson, S.: The information state approach to dialogue management. In: van Kuppevelt, J., Smith, R. (eds.) *Current and New Directions in Discourse and Dialogue*, pp. 325–353. Kluwer (2003)
17. Weathers, F., Litz, B., Herman, D., Huska, J., Keane, T.: The PTSD checklist (PCL): Reliability, validity, and diagnostic utility. In: *the Annual Convention of the International Society for Traumatic Stress Studies* (1993)
18. Williams, J., Young, S.: Scaling POMDPs for spoken dialog management. *IEEE Trans. Audio, Speech, Lang. Process.* **15**(7), 2116–2129 (2007)