Case-Based Techniques Used for Dialogue Understanding and Planning in a Human-Robot Dialogue System

Karolina Eliasson
Linköping University
Department of Computer and Information Science
SE - 581 83 Linköping
karel@ida.liu.se

Abstract

We describe an approach to the use of case-based techniques for natural language understanding and for action planning in a system for dialogue between a human and a robot, which in our case is a UAV (unmanned aerial vehicle). A single case base and case-based reasoning engine is used both for understanding and for planning actions by the UAV. This approach has been developed through the work on an experimental dialogue system, called CEDERIC. Dialogue experiments where a number of users have solved tasks by dialogue with this system showed very adequate success rates, while at the same time they indicated a few weak points in the system that could then easily be corrected.

1 Introduction

Artificial intelligence has always been associated with natural language understanding, generation, and dialogue. However, traditional dialogue systems do not incorporate any learning abilities and very few memory abilities. The memories used are often dedicated only to discourse modeling. Instead of learning, most systems rely on a repertoire of manually written phrases that are used to mimic intelligence and react correctly on the user's input. There are two major disadvantages with this design:

- It is very difficult to anticipate all the different phrases the user may use to communicate with the system. This approach requires substantial knowledge of dialogue system development and domain knowledge and it is tedious and time consuming.
- The dialogue system can not adapt to a new situation nor function well together with a learning artificial intelligence back end system that evolves over time.

The first item can partly be solved by using Wizard-of-Oz techniques. On the other hand, Wizard-of-Oz tests are time consuming and it is difficult to know if the entire spectrum of phrases is covered. Further more, even if the system is well written and covers the current aspects of use, it is still static and can not adapt to a new situation.

In this article, an approach to use machine learning techniques for natural language understanding and for action planning in a human-robot dialogue system is presented. This approach has been developed through the work on an experimental dialogue system called CEDERIC, which is an abbreviation for Case-base Enabled Dialogue Extension for Robotic Interaction Control.

The machine learning algorithm selected is case-based reasoning (CBR) [Aamodt and Plaza, 1994]. CBR is chosen partly because it mimics human learning [Schank, 1982] and partly because it learns through its entire lifetime, unlike many other machine learning algorithms that only learn during an initial learning phase. CBR is used in our approach to interpret utterances from the operator and messages from the robot, and to find suitable responses to them, hence the dialogue system learns from experience and adapts to its current operator's use of language and the domain at hand. Further more, CEDERIC uses case-based planning (CBP) [Spalazzi, 2001] to make effective use of the experiences gained and combine it in different ways and in various contexts to solve new, unseen problems.

2 An Overview of CEDERIC

CEDERIC is developed within the WITAS project whose goal is to develop an autonomous unmanned aerial vehicle (UAV) system that is able to make rational decisions about the continued operation of the aircraft, based on various sources of knowledge including pre-stored geographical knowledge, knowledge obtained from vision sensors, and knowledge communicated to it by data link [Doherty *et al.*, 2000]. The UAV used in the project is a Yamaha RMAX helicopter which can be controlled by high level voice commands or by written commands.

In a dialogue with such a robot, the dialogue manager must be able both to interpret the utterance from the operator, receive additional information to be able to react correctly to the utterance, compute data internally, and send commands to the agent and results to the operator. To do this, it has to distinguish between *dialogue acts*, *physical acts*, and *internal acts*. A dialogue act can be to ask a clarifying question to the operator, a physical act can be to ask the agent to perform a high-level command such as `land`, and an internal act can be to look up information in a database. These acts must be executed in a correct order to solve the problem at hand. If the information is not sufficient for the dialogue manager to complete the task then it has to ask for complementary infor-

Operator: Show me the front facade of the main building.
CEDERIC: I am taking off.
CEDERIC: Is this view ok?
Operator: Climb five meters.
CEDERIC: OK I have ascended five meters.
Operator: Fly to the red car.
CEDERIC: OK I am starting to fly now.
CEDERIC: I am at the car now.
Operator: Now please go to the school.
CEDERIC: I have several schools to choose between. Which one do you mean?
Operator: The red one.
CEDERIC: OK I am starting to fly now.
CEDERIC: I am at the red school now.

Figure 1: An example dialog performed by CEDERIC.

mation from the operator. In such a case a discourse model is needed to, e.g., resolve anaphoric references and to manage sub-dialogue.

The various kinds of acts that are needed to perform a dialogue are stored as plans in CEDERIC. For example, for the user utterance `fly to the hospital`, a plan can be to look up the position of the hospital, make sure that the helicopter is in the air, create a command to be sent to the robot that commands it to fly to the position of the hospital, wait for the robot to announce that it has reached the position and finally report to the operator that the robot has reached its destination.

2.1 A Dialogue Example

Because of CBR, the operator is free to use synonyms in the utterances and words can be added or deleted without reducing the performance of the system, hence the operator does not have to learn and remember exact formulations.

Planning adds the functionality to combine cases to solve new, unseen problems, hence the information already known to the system can be used in new, creative ways. Planning also makes CEDERIC understand implicit commands and solve problems on its own. For example it is very useful when the system realizes some of the preconditions for performing a command from the operator, are missing. Using planning, CEDERIC can automatically perform some actions to be in a state that satisfies the preconditions.

Figure 1 gives an implemented and working example dialogue between an operator and CEDERIC. It is nontrivial in several aspects and illustrates the benefits of the CBR/CBP approach. The first command from the operator activates a case in the case base that moves the on-board camera in the right direction to be able to view the front facade of the mentioned building. As the helicopter is still on the ground at the start of the dialogue, this command can not be directly performed, hence the system has to search all its knowledge and experiences to solve the problem and fulfill the preconditions of being in a position where the front facade of the main building can be viewed in the camera. The solution of taking off before executing the command is found automatically by the system using CBP.

The command `Fly to the red car` is not previously stored in the case base, but the rather similar command `Fly to the hospital` is stored. However, the solution to the latter command does not solve the problem of the former command because a vehicle and a building are not dealt with uniformly. By using information from the command `What is the position of the red car`, which is stored in the case base, a plan for solving the original command can be found using CBP.

The fourth operator command in the dialogue is an example where CBR is used but CBP is not. The command `Now please go to the school` is not previously stored in the case base but the similar command `Fly to the hospital` is. By using similarity measures and adaptation techniques the solution of the latter command can be transformed to solve the former command. This command also exemplifies the use of sub-dialogue and anaphoric references.

2.2 System Architecture

The CEDERIC architecture is shown in Figure 2. The dialogue manager is connected to a Speech and Graphics User Interface (SGUI). SGUI contains a map of the area where the UAV is situated and functionality for the operator to choose to use either speech or text for the input to the dialogue system. The system Nuance[1] is used as the speech recognizer, and the systems Festival[2] and BrightSpeech[3] are used for speech generation. Nuance and BrightSpeech are commercial products. CEDERIC is also connected to the robotic control system of the actual helicopter or to a simulator that simulates the robot and the environment.

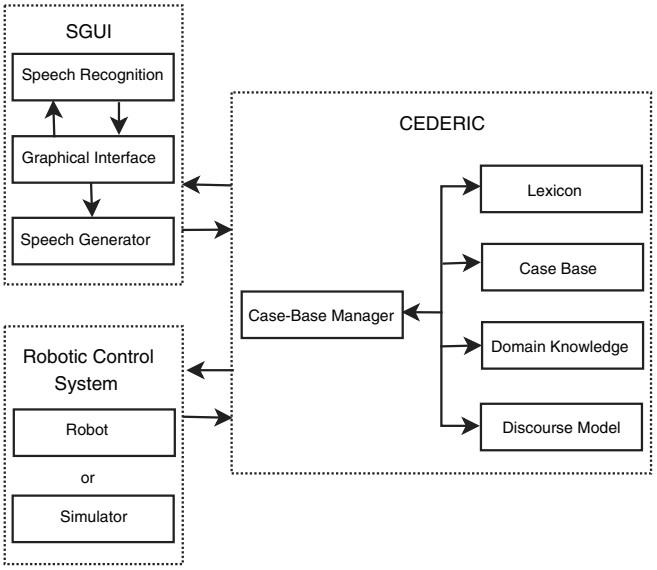

Figure 2: Architecture of CEDERIC.

[1] http://www.nuance.com/
[2] http://www.cstr.ed.ac.uk/projects/festival/
[3] http://www.brightspeech.com/

CEDERIC consists of a *lexicon*, a *case base*, *domain knowledge*, a *discourse module* and a *case-base manager*.

The lexicon is used to classify the words in the user utterance according to some predetermined word classes. The classified utterance is matched against cases in the case base by the case-base manager. The discourse module is responsible for maintaining a discourse model of the dialogue in order to be able to interpret the operator's sentences. The discourse model helps the system to interpret references that may refer to sentences earlier in the dialogue, to keep track of different ongoing dialogues, and to decide if it is a good moment to send a phrase to the operator or not. The domain knowledge contains an ontology of the world as the robot knows it and a categorization of the world items. The purpose is twofold. It serves as a world representation which gives CEDERIC knowledge about which buildings there are in the known world, what kinds of buildings they are, where they are located, and their attributes such as color and facade material. It also gives CEDERIC fundamental knowledge about categorization, e.g., which items can be called buildings in the dialogue and which can not.

Regardless of whether a speech recognizer is used or not, a sentence from the operator arrives to CEDERIC in plain text format. It is classified and processed in the CBR engine. CEDERIC returns either a new phrase in text format to be sent to the speech generator, or a request to the robotic control system.

3 The Case Base

The actual dialogues are saved in the case base. The initial functionality is hand crafted, but more functionality is added automatically when the system is used. There are two basic case structures in the case base: *dialogue cases* and *act cases*. Figure 3 shows an example of a dialogue case for the utterance `Fly to the hospital` and an example of the first act case in the plan.

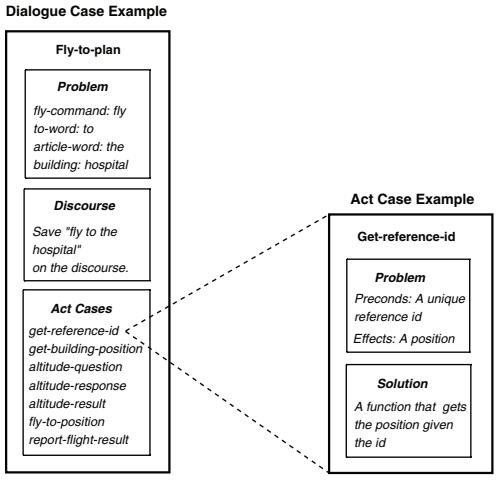

Figure 3: An example of a dialogue case and a close up on the first act case in the plan.

3.1 Dialogue Case

A dialogue case contains the plan for solving a particular kind of problem. It consists of three parts:

- A *Problem Part* that describes what problem the case is solving. The problem is usually described by the words in the input phrase used by the operator.
- A *Discourse Part* that describes how the discourse model should be updated according to the problem.
- A *Plan Part* containing a plan that solves the problem. The plan consists of references to act cases that themselves are cases in the case base.

3.2 Act Case

An act case is either a dialogue act, a speech act or an internal act. Arranged in plans, they build up the functionality of the dialogue manager.

Act cases consist of the following parts:

- A *Problem Part* that describes which preconditions that have to be met to be able to execute the act case, and the effects of the act case.
- A *Solution Part* containing a solution to the problem as a higher order function and the parameter values used when the solution was executed.

The preconditions and the effects of an act case are used for planning and validation purposes. To be able to construct a plan in advance that probably will hold when executed, some knowledge of each act is necessary. The precondition part holds information of what has to be known before the act can be executed, and the effect part holds information of what result the act can produce. The effects may then serve as preconditions for other acts further on in the plan. Each act may produce several different types of results depending on circumstances that are only known at execution time, therefore the effects part must contain information about each possible result type so that the planner can take different possible outcomes into account.

The solution function constructs the actual result, such as a message to be sent to the helicopter control system, or performs some internal computations.

4 Case-Base Manager

The case-base manager is the engine of CEDERIC. It manages different ongoing dialogues and executes the best fitting act case. If there are open and ongoing dialogues, the new input is first seen as a continuation of one of the open dialogues. If it does not fit as a continuation, it is seen as the start of a new dialogue, and a suitable, similar dialogue case is searched for, and the plan is executed. The dialogue handling allows dialogue topic switches, where the operator may start a new dialogue before ending an old one, or return to an old dialogue, without confusing the system.

4.1 Syntactic Categorization of Words

The utterance from the operator enters the system as a sentence in text format. Each word in the utterance are cate-

gorized using a lexicon. An entry may for example look as follows, in the KM[4] notation [Clark and Porter, 2004]:

```
(school has
  (instance-of (building))
  (plural (schools))
  (weight (2)))
```

where `school` is modeled to be an instance of the category `building`. It also has the attribute `plural` which is set to `schools`, and a weight which is set to 2. The weight is used in the similarity function and is an indication of how important the word is for the comprehension of an utterance. A word may be ambiguous and belong to several different categories depending on the semantics of the utterance. In that case, the word is categorized with all the matching categories, and the similarity function decides which meaning to use. A word that is not present in the lexicon obtains the category `no_category`.

4.2 Case Retrieval

In the retrieval phase, the utterance is compared with the problem part of each dialogue case, and a similarity value is computed. The cases are then priority-ordered using the similarity value. The similarity value is based on how well each word in the utterance matches the words in the problem formulation and the weight of each word according to the lexicon. The case retrieval phase is only executed if the input utterance is a start of a new dialogue, i.e., there are no old or ongoing dialogues where the utterance fits as a continuation.

To be more specific, the similarity value for each dialogue case is computed using the following formula, where w_c means the sequence of words in the problem part of the case and w_i means the sequence of words in the input from the operator:

$$Similarity = points(w_c, w_i) - uncovering(w_c, w_i)$$

where

$$points(w_c, w_i) =$$

$$\sum_{w_x \in w_c} \sum_{w_y \in w_i} class - value(w_x, w_y) \times word - weight(w_y)$$

$$class - value(w_x, w_y) = \begin{cases} 3 & \text{if the words are equal} \\ 2 & \text{if the words are classification similar} \\ 0 & \text{otherwise} \end{cases}$$

Two words, w_1 and w_2, are *classification similar* if either w_1 is equal to the classification of w_2, the classification of w_1 is equal to w_2, or the classification of w_1 is equal to the classification of w_2.

When two cases have the same similarity-value they are subprioritized using the *uncovering* function which indicates how much the sequences differ in length. A low value on the *uncovering* function is prioritized.

When the cases are ranked in order of priority, the case with the highest priority is chosen unless it does not have lower similarity value than a threshold. In that case, no similar case was found.

If a similar case is found, the plan is validated before execution. In the validation phase, the preconditions of each act case are checked. They can be fulfilled by words from the input phrase or by effects from previously checked acts in the plan.

5 Planning

Planning is used in two different phases in CEDERIC:

- When there is a plan fault in the plan validation process of the most similar dialogue case, i.e., when the preconditions of the act cases in the plan are not met.
- When there is a plan fault in the execution of a plan.

The same replanning routine, based on case-based planning (CBP) methods, is used in both cases. In CBP, different plans are combined to form a new plan that can serve as a solution to a new problem. In CEDERIC, two plans can be combined to form a new plan, where one of the original plans works as the foundation on which the new plan is constructed. The preconditions and effects in the act cases are used in the replanning process to make sure the newly constructed plan is valid. Figure 4 shows a schematic illustration of a planning problem and its solution. When a plan fault is encountered,

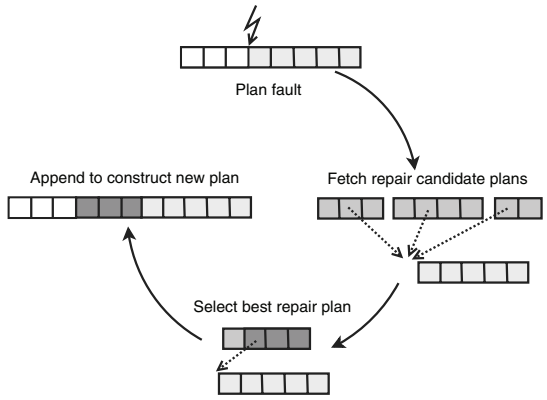

Figure 4: A schematic view of replanning in CEDERIC.

the effects of the previous act cases in the plan, or the results if it was an execution plan fault, are sent to the replanning routine together with the rest of the plan, i.e., the plan where the act case that caused the fault is the first item. Other plans in the case base are searched to find a plan or part of a plan that is valid to execute given the preconditions and results already obtained. All such repair candidate plans are examined to find a plan sequence that can be merged with the original plan to create a new valid plan, that is, to find a plan sequence to execute that can produce the information needed to be able to execute the item that caused the plan fault. If several such plans are found, the shortest one is chosen.

[4]The language KM (the Knowledge Machine) is a frame-based knowledge representation language used in CEDERIC both for representing the case base, the lexicon, the domain knowledge, and the discourse model.

If no valid plan is found, the first act in the original plan is removed and the algorithm tries again to find new, partly matching plans to be used to repair the original plan, but this time the goal it to be able to satisfy the preconditions of the next act case in turn in the original plan. In this manner, parts of the original plan are replaced by parts of another plan. The newly found plan sequence is then linked back to the original plan. Together with the already checked or executed act cases, the newly created plan consists of a sequence from the foundation plan, a sequence from a repair candidate plan, and then again a sequence from the foundation plan. If it was impossible to link back to the original plan, the planning routine halts with a failure message. In this way, the goal of the planning routine is exchanged during the planning attempts and the routine explores more ways of solving the problem.

The revised plan may again obtain a plan fault when executed, and it is then repaired using the same replanning routine, but the system has to make sure it does not repair the plan with a solution that has been tested and failed previously. To avoid such cycles, the new plan is compared with the execution history, and the new plan has to be unique in some sense to be a promising repair plan.

6 Case Retention

When a new problem is not identical to the problem part of the plan case that was used for solving it, or the plan was not identical with the plan of the similar case, and the execution of the problem was successful, the problem and solution are saved as a new case in the case base. The new cases increase the case base and the new experiences can be used to solve other problems in the future. A new problem and solution can differ from the cases used to solve it if one or more of the following has happened:

- One or more of the words in the input are not identical to the words in the problem part of the dialogue case. They are only classification similar.
- One or more of the words in the input are added or deleted from the words in the dialogue case, but the plans are identical to each other.
- The plan has been replanned in the plan check phase.
- The plan has been replanned in the plan execution phase.

In those cases, a new plan case with corresponding act cases are created and stored in the case base.

7 Tests and Results

Using CBR, it is possible to solve similar problems as the ones described in the case base, by adapting the solution to the most similar problem stored in the case base. This approach is mainly used in CEDERIC to understand and react correctly to utterances where some of the words in the utterance are missing, where additional words are added, or where some of the words are exchanged to synonyms that may be unfamiliar to the system. In those cases where CBR methods alone fail, CBP is used to combine experiences from several different cases to find a solution to a new problem.

In a test performed manually, the performance of the system has been tested with a case base that initially contained 22 different problems. Using only CBR, the system could solve at least another 30 similar but not identical problems. Using CBP, the system could solve 8 additional problems that could not be solved with CBR techniques only.

7.1 User Tests

CEDERIC has been tested in user tests, together with the SGUI user interface and simulator. In one test, five persons without previous experience of CEDERIC were used. A simple scenario including five missions and a model of the world where the helicopter is situated was given to each test person before the test began. The missions were described in such a manner that they did not reveal the actual phrases that should be used to solve them. The goals of the missions were for example to move the helicopter to a given position indicated by a building or to move it to a certain altitude.

A reference test was also performed where five persons, different from the ones that tested the original system, were testing a stripped baseline version of CEDERIC. The test procedure, including the order in which the missions was performed, was identical to the test procedure used in the test of CEDERIC. In the baseline version, the similarity function only accepted utterances where the word categories, the order of the words in the utterance, and the length of the utterance were identical to the matching case. The new experience was not saved in the case base and the planning algorithm was removed. To be able to use the same test scenario, some additional cases that in the original version demanded planning (e.g. sub-dialogue due to the need of clarifying questions) had to be added to the case base.

Every test person that tested CEDERIC completed the missions successfully. No one out of the five test persons that tested the baseline system succeeded with mission 4. The results of the tests are shown in Figure 5. The large difference

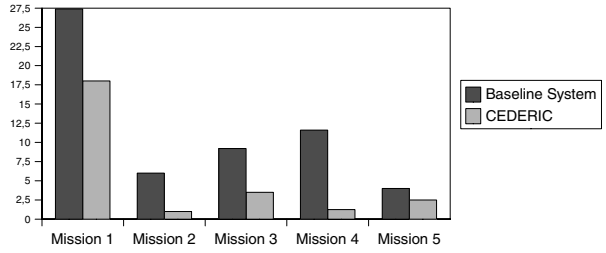

Figure 5: The result of the user test where the y-axis is the average number of turns used to complete each mission.

between the number of turns needed to complete each of the five missions using CEDERIC compared to the number of turns needed to complete (or fail due to the test person giving up trying) the missions using the baseline system indicates that CBR and CBP significantly improves the usability of a dialogue system.

The suggested formulations, that was not understood by CEDERIC at the time of testing, was easily implemented after the entire test for all test persons were performed. Hence, the system could be improved using the dialogue collected from the tests.

7.2 Related Work

Systems for dialogue with a UAV have been implemented earlier, e.g., in the WITAS project [Doherty *et al.*, 2000]. Several dialogue systems have been developed within the project. The first WITAS Dialogue System [Lemon *et al.*, 2002] was a system for multi-threaded robot dialogue using spoken I/O. The DOSAR-1 system [Sandewall *et al.*, 2003] was a new implementation using another architecture and a logic base. Our work takes a rather different approach than their systems due to the use of machine learning and planning.

Machine learning techniques in combination with dialogue systems has mostly been used to learn dialogue strategies, that is, to formalize a dialogue as an optimization problem [Levin *et al.*, 2000], [Singh *et al.*, 2002]. The method is useful in a dialogue system where the information needed can be asked for in several different ways and in different order. Learning dialogue strategies differs from the approach in CEDERIC both in the performance and in the achieved goal, as CEDERIC attempts to learn entirely new problems and their solutions.

Murao et al. present a dialogue system that uses CBR to learn new dialogues from an example corpus [Murao *et al.*, 2003]. The cases are made up by an input from the user in natural language and the solution to the input is a reply, also in natural language. Longer dialogues are not captured in the case base in the same manner as CEDERIC does, and no planning is performed to solve new unseen problems. In combination with the lack of discourse information, this makes the dialogue rather strict and simple, compared to the dialogues in CEDERIC.

CEDERIC has much in common with the interactive planning system TRIPS [Ferguson and Allen, 1998]. The goal in both projects is to give an approach to integrated AI systems, where several AI components are integrated so solve end-to-end problems. However, TRIPS makes a more precise distinction between dialogue and action planning than CEDERIC. Our approach integrates planning with dialogue understanding and generation more tightly. In addition it also integrates machine learning.

7.3 Conclusion

We present a combined CBR and CBP approach to natural language understanding. Both the CBR/CBP application to dialogue management, and features related to CBR/CBP techniques, such as the similarity function, the construction of the case base, and the planning algorithm, are novel contributions. The approach has turned out to be fruitful. The user of CEDERIC can be more relaxed in the dialogue with the system, and she does not have to learn the exact phrases to be used since the system can make a qualified guess out of the information at hand. CEDERIC also makes effective use of the knowledge gained from previous experiences, and is able to solve problems that it has not encountered before, without any help from a human operator. This functionality gives a flexible dialogue manager that is able to understand a larger number of phrases than are initially given, without the need of a human developer adding the phrases by hand. User tests shows that CEDERIC has a good success rate and that the machine learning approach actually facilitates the use of the system.

Acknowledgments

This research work was funded by the Wallenberg Foundation and the Swedish National Graduate School of Computer Science (CUGS).

References

[Aamodt and Plaza, 1994] Agnar Aamodt and Enric Plaza. Case-based reasoning; Foundational Issues, Methodological Variations, and System Approaches. *AI Communications*, 7(1):39–59, 1994.

[Clark and Porter, 2004] Peter Clark and Bruce Porter. *KM - The Knowledge Machine 2.0: Users Manual*, 2004.

[Doherty *et al.*, 2000] Patrick Doherty, Gösta Granlund, Krzysztof Kuchinski, Erik Sandewall, Klas Nordberg, Erik Skarman, and Johan Wiklund. The WITAS Unmanned Aerial Vehicle Project. In *Proceedings of the 12th European Conference on Artificial Intelligence*, pages 747–755, 2000.

[Ferguson and Allen, 1998] George Ferguson and James F. Allen. TRIPS: An Integrated Intelligent Problem-Solving Assistant. In *AAAI '98: Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 567–572, 1998.

[Lemon *et al.*, 2002] Oliver Lemon, Alexander Gruenstein, and Stanley Peters. Collaborative Activities and Multitasking in Dialogue Systems. *Traitement Automatique des Langues (TAL), special issue in dialogue*, 43(2):131–154, 2002.

[Levin *et al.*, 2000] Ester Levin, Roberto Pieraccini, and Wieland Eckert. A Stochastic Model of Human-Machine Interaction for Learning Dialog Strategies. *Journal of Artificial Intelligence Research*, 8(1):105–133, 2000.

[Murao *et al.*, 2003] Hiroya Murao, Nobuo Kawaguchi, Shigeki Matsubara, Yukiko Yamaguchi, and Yasuyoshi Inagaki. Example-based Spoken Dialogue System using WOZ System Log. In *SIGdial Workshop on Discourse and Dialogue*, pages 140–148, 2003.

[Sandewall *et al.*, 2003] Erik Sandewall, Patrick Doherty, Oliver Lemon, and Stanley Peters. Words at the Right Time: Real-Time Dialogues with the WITAS Unmanned Aerial Vehicle. In *Proceedings of the 26th Annual German Conference in AI*, pages 52–63, 2003.

[Schank, 1982] Roger C. Schank. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*. Cambridge University Press, 1982.

[Singh *et al.*, 2002] Satinder Singh, Diane Litman, Michael Kearns, and Marilyn Walker. Optimizing Dialogue Management with Reinforcement Learning: Experiments with the NJFun System. *Journal of Artificial Intelligence Research*, 16(1):105–133, 2002.

[Spalazzi, 2001] Luca Spalazzi. A Survey on Case-Based Planning. *Artificial Intelligence Review*, 16(1):3–36, 2001.