In Proceedings of the AI in Education (AIEd-2003)

Proactive Acquisition from Tutoring and Learning Principles

Jihie Kim and Yolanda Gil
Information Sciences Institute, University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292, U.S.A.
(310) 448-8769
{jihie, gil}@isi.edu

Abstract. Knowledge acquisition interfaces can be seen as students learning new knowledge from the user (teacher), and we believe that they should be able to use some of the strategies that good learners pursue during a tutoring dialogue. This will help acquisition interfaces become proactive learners, able to reason about learning activities and with initiative in participating in the process accordingly.

In this paper, we present our design and implementation of a new acquisition dialogue tool called *SLICK* that is built based on the tutoring and learning principles we have compiled. SLICK makes use of these principles through 1) goals that represent what remains to be learned, 2) strategies to achieve these goals and acquire further knowledge, and 3) awareness of the current status of the body of knowledge learned. The tool has been used for acquiring two very different types of knowledge: biological process models and military plans. The resulting interactions show that the system is aware of its progress towards acquiring the new knowledge, and moves forward by understanding what acquisition goals and strategies to pursue.

1 Introduction

In instructional systems (both educational software and intelligent tutoring systems), the tutor's role is to help the user (student) achieve some degree of proficiency in a certain topic (the lesson). In interactive acquisition interfaces, these roles are reversed. Acquisition tools can be seen as students learning new knowledge from the user (teacher) and they should be able to use some of the strategies that good learners pursue during a tutoring dialogue. Ideally, it should also be able to supplement the user's skills as a teacher by helping the user pursue effective tutoring techniques. This would help the user teach the material better and faster to the system, as well as delegate some of the tutor functions over to the system.

In essence, we are trying to investigate what it takes to create a good student, while most ITS work has focused on creating good teachers. We believe that the work in educational systems and acquisition systems share a lot of issues and they may be able to contribute to each other in many ways[1]. For example, student's role and the ability to assess one's own progress in learning is emphasized as an important factor in cognitive development [3, 16], which may be closely related to our work. In fact there has been work that bridges the two communities. For example, there have been recent interests in acquiring knowledge for

[1] There are some issues that interactive acquisition interfaces will not face. For example, human students in need of tutoring often have a lack of motivation that the instructional system has to address [11].

Teaching/Learning principle	Tutoring literature
Introduce lesson topics and goals	Atlas-Andes, Meno-Tutor, Human tutorial dialog, human learning
Use topics of the lesson as a guide	BE&E, UMFE
Subsumption to existing cognitive structure	human learning, WHY, Atlas-Andes
Immediate feedback	SOPHIE, Auto-Tutor, LISP tutor, Human tutorial dialog, human learning
Generate educated guesses	Human tutorial dialog, QUADRATIC, PACT
Keep on track	GUIDON, SCHOLAR, TRAIN-Tutor
Indicate lack of understanding	Human tutorial dialog, WHY
Detect and fix "buggy" knowledge	SCHOLAR, Meno-Tutor, WHY, Buggy, CIRCSIM, human learning
Learn deep models	PACT, Atlas-Andes
Learn domain language	Atlas-Andes, Meno-Tutor
Keep track of correct answers	Atlas-Andes
Prioritize learning tasks	WHY
Limit the nesting of the lesson to a handful	Atlas
Summarize what was learned	EXCHECK, TRAIN-Tutor, Meno-Tutor
Assess learned knowledge	WEST, Human tutorial dialog

Table 1: Some Tutoring and Learning Principles [Kim and Gil, ITS-2002]

intelligent tutoring systems [12]. We think that technology built by the knowledge acquisition community will be useful for building tools to help users develop the knowledge and models used in ITS.

The goal of our work is to develop acquisition interfaces that are proactive learners, able to reason about learning activities and with initiative in participating in the process accordingly. Our approach is to having tutoring and learning principles represented explicitly and declaratively and be aware of the level of competence and confidence of the knowledge they are acquiring. This would enable acquisition tools to reason in terms of the teaching and learning process and to make interaction with the user dynamically generated given the situation at hand. We present our design and implementation of a new acquisition dialogue tool called *SLICK* (Skills for Learning to Interactively Capture Knowledge) that is built based on the tutoring and learning principles we have compiled so far[10]. SLICK makes acquisition tools more proactive by maintaining 1) goals that represent what remains to be learned, 2) strategies to achieve these goals and acquire further knowledge, and 3) awareness of the current status of the body of knowledge learned. The tool has been used for acquiring two very different types of knowledge: biological process models and military plans. The resulting interactions show that the system is aware of its progress towards acquiring the new knowledge, and moves forward by understanding what acquisition goals and strategies to pursue.

The paper begins discussing the rationale behind our approach as we explored relevant literature in learning and tutoring. We then introduce our system we developed in the context of supporting interactive knowledge acquisition. Finally we present our lessons learned from user feedback.

2 Tutoring and Learning Principles for Proactive Learning

We have investigated various tutoring principles used by human tutors and instructional software that are relevant to acquisition tasks (Figure 1). They helped us understand the nature of good teacher-student interactions. We noticed that many useful learning principles could be seen as learning goals and teaching goals that students and teachers seem to pursue at different points throughout a lesson[10]. For example, the topic of the lesson is sometimes presented to the student at the beginning, followed by the content of the lesson, then test questions, and then a summary of the lesson[17, 7, 13]. Setting up the topic of the lesson at the beginning helps draw on prior knowledge (subsumption to existing cognitive structure) and

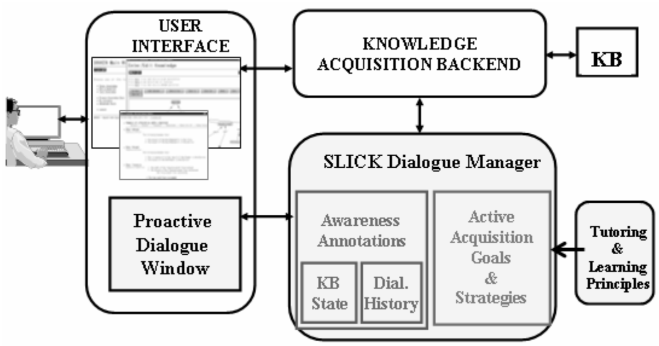

Figure 1: SLICK architecture

helps the teacher detect missing prior knowledge that needs to be provided before carrying on with the lesson[18, 7, 1]. A reasonable expectation in a tutoring situation is that all new items defined must have a connection to the topic of the lesson[14]. Testing the student is also a major tutoring activity[4, 18, 2]. Some questions will test the new knowledge with respect to existing knowledge to ensure it fits adequately[13, 1]. Students should not only be expected to give the right answer but to do so for the right reasons[15]. The tutor should be notified if the answer to a question asked previously changes in light of additional material taught[5]. Another interesting aspect of a lesson is learning to describe the new knowledge in terms that are appropriate in the domain at hand[15]. Finally, it is useful to limit the nesting of lessons to a handful[15].

We also have investigated how these principles would benefit the current acquisition tools (including our own work) that use various techniques to support users. We have found that the principles have only been used in some aspects of the functionality of acquisition tools, and are exhibited by some but not all the tools[8]. For the principles they use, they rather implicit in the design of the tool, and their influence is limited to the degree that they are implemented in the underlying code.

3 Declarative Representation of Tutoring and Learning Principles

We have used the above principles in the context of building proactive acquisition interfaces. Acquisition tools can be seen as students learning new knowledge from the user (teacher) and they should be able to use some of the strategies that good learners pursue during a tutoring dialogue. The following presents the capabilities we provide to acquisition interfaces based on these principles.

- Acquisition interfaces should be able to represent *acquisition goals* explicitly. Many of the tutoring principles suggest a more goal-oriented behavior for acquisition interfaces. Having acquisition goals explicitly and declaratively is key to making a tool truly proactive because it could then steer the dialogue with the user to work towards those goals. The goals that are achieved at each point during the dialogue represent the progress made towards acquiring the desired body of knowledge.

- Acquisition tools should have *awareness* of what they have learned already and what they do not know about yet, so that they can better assess their competence and confidence in specific topics, and steer the dialogue with the user in directions that improve their body of knowledge on both counts.

- Acquisition interfaces should have *acquisition strategies* in order to understand and actively pursue what is involved in learning about a new topic. Acquisition strategies outline how to achieve acquisition goals. Because so many things are unknown to the system during the lesson, these strategies can only be pursued under the user's guidance and in a mixed-initiative interaction.

SLICK is developed as a front-end to existing basic acquisition tools by embodying these capabilities. Figure 1 shows the architecture of the system. The boxes in gray represent the SLICK components that extends basic acquisition tools. The arrow between "Tutoring & Learning Principles" and "SLICK Dialogue Manager" means that the general tutoring and learning principles are operationalized based on the target knowledge to be acquired and the features of the given acquisition tool. For example, tools that acquire different forms of knowledge (such as problem-solving knowledge vs. concepts) may need different operational goals because they have different subcomponents and functions to build up knowledge bases. The details are described below. Actions done by the user through the basic acquisition tool are intercepted by our system. While the backend tool will update the backend knowledge base and its own user interface, SLICK will update its own structures and user interface.

Acquisition Goals: Figure 2-(a) shows the general acquisition goals that we use. The tutoring and learning principles in Section 2 are mapped into these goals according to the main activities in acquisition interfaces. (The goals and principles not used currently can be incrementally added in the future.) We found it useful to group acquisition goals into six themes, each with a different emphasis on what is being learned. For example, Goal 1.1 (Get the overall topic and purpose of lesson) can be adopted in acquisition interface in order to make the lesson more coherent. There is no notion in acquisition tools that there is a lesson being started or ended, since at any point users can choose to enter knowledge about any topic. Current acquisition tools do not have any basis to evaluate or pursue depth in their knowledge base. One thing acquisition tools can do is to provide a way of enforcing users to check how the answers were generated to check that the system provides the right answer for the right reasons (Goal 3.3).

These high level acquisition goals are mapped to more specific goals to accommodate different acquisition tools and representations. For example, in some cases the purpose of the lesson can be specified as a suite of types of test questions that the system should be able to answer correctly after the lesson. In other cases it could be given as an exhaustive list of new terms to be defined during the lesson.

Learning Awareness: We represent awareness with two kinds of annotations: annotations to the new body of knowledge acquired and annotations to the interaction history.

A new body of knowledge based is associated with the lesson/purpose/topic of the session(s) where it is acquired. We consider a new body of knowledge as a collection of *knowledge items* (e.g., concepts, problem solving methods or rules, instances, examples), each with an associated set of *axioms* (e.g., range constraints, subclass relations) that embody the knowledge about that item. We record this structure (axioms associated with items, items associated with lessons) and extend it as the user goes through the session. This basic structure is annotated with meta-level information about its status, where we aim to capture how much is known about that lesson/item/axiom and how confident the system is about it. Figure 2-(b) shows the annotations that we use.

1) SET UP LESSON AND CHECK BACKGROUND:
- 1.1. Get the overall topic and purpose of the lesson.
- 1.2. Acquire any assumed prior knowledge before pursuing the lesson.

2) ACCEPT AND RELATE NEW DEFINITIONS:
- 2.1. Accept new definitions.
- 2.2. Ensure that new knowledge is specific as possible.
- 2.3. Ask the user to be complete when enumerating items in terms of the elements and in terms of the significance of the order given.
- 2.4. Get all the information required when existing knowledge indicates it must be provided.
- 2.5. Make all new definitions consistent with existing knowledge.
- 2.6. Connect all new items with the topic of the lesson.

3) TEST AND FIX:
- 3.1. Test the new body of knowledge and generate tests for the aspects that have not been thoroughly tested.
- 3.2. Fix problems that result from self-checks or from user's indications.
- 3.3. Ensure user checks the reason for the answers, not just the answers .
- 3.4. Confirm new answers that change in light of new knowledge.

4) FIT WITH EXISTING KNOWLEDGE STRUCTURES:
- 4.1. Establish identity of new objects by checking if existing objects appear to be the same.
- 4.2. Generalize definitions if analogous things exist and there could be plausible generalizations.

5) ACHIEVE PROFICIENCY:
- 5.1. Acquire domain terms to describe new knowledge.
- 5.2. Learn to reason/generate answers efficiently and with shorter explanations.

6) REACH CLOSURE ON LESSON:
- 6.1. Ensure that the purpose/topics of the lesson were covered and the test questions appropriately answered.

(a) Acquisition Goals

- Annotations to the new body of knowledge:
  - For each lesson: purpose, assumed background, sub-lessons, overall competence and confidence (based on tests)
  - For each k item: connection to lesson, relation to other items, identity wrt other items, possible analogies and generalizations, domain terminology details, competence, confidence
  - For each axiom of a k item: required information, generality, completeness, confidence
- Annotations to the dialogue history:
  - For each user action: changes to the annotations to the new knowledge, acquisition goals achieved and/or activated, possible future KA strategies

(b) Awareness Annotations

Figure 2: Acquisition Goals and Awareness Annotations.

A novel feature here is the focus on keeping track of what is known, not just on what is not known. Traditionally, the focus of acquisition tools has been on errors and gaps in the knowledge base. In some sense, a knowledge base is never complete, so these annotations should ideally become part of the knowledge base or at least in an accessible record of how a body of knowledge was acquired by the system in certain sessions with certain users.

Annotations to the interaction history record what action the user took at each point in time (e.g., define a concept as a subclass of another one, define a new role for that concept, test the knowledge with a question), and what progress resulted from that action in terms of the lesson at hand. The system notes the changes to the annotations of the body of knowledge that resulted from the user's action. In addition, the system records what learning goals have been achieved and what learning goals become active, as well as what strategies seem to make sense in order to achieve those goals. These annotations of the interaction history allow the system to share with the user its understanding of what it is learning as the lesson progresses.

Acquisition Strategies: Acquisition strategies can be also formulated based on the principles shown in Table 1. For example, the system can attempt to be a good learner by making educated guesses when possible, and by noting surprise if its guesses are wrong.

If the system can use some heuristics to determine that an instantiation of an acquisition strategy is more likely than others (for example, by drawing an analogy with existing knowledge), then that more concrete strategy would be shown to the user. Strategies that achieve more than one acquisition goal are considered more likely. For example, a goal to fill in required information of an item and a goal to connect a new item to the lesson can be both solved if the two items are connected (assuming that the first item is already connected to the lesson).

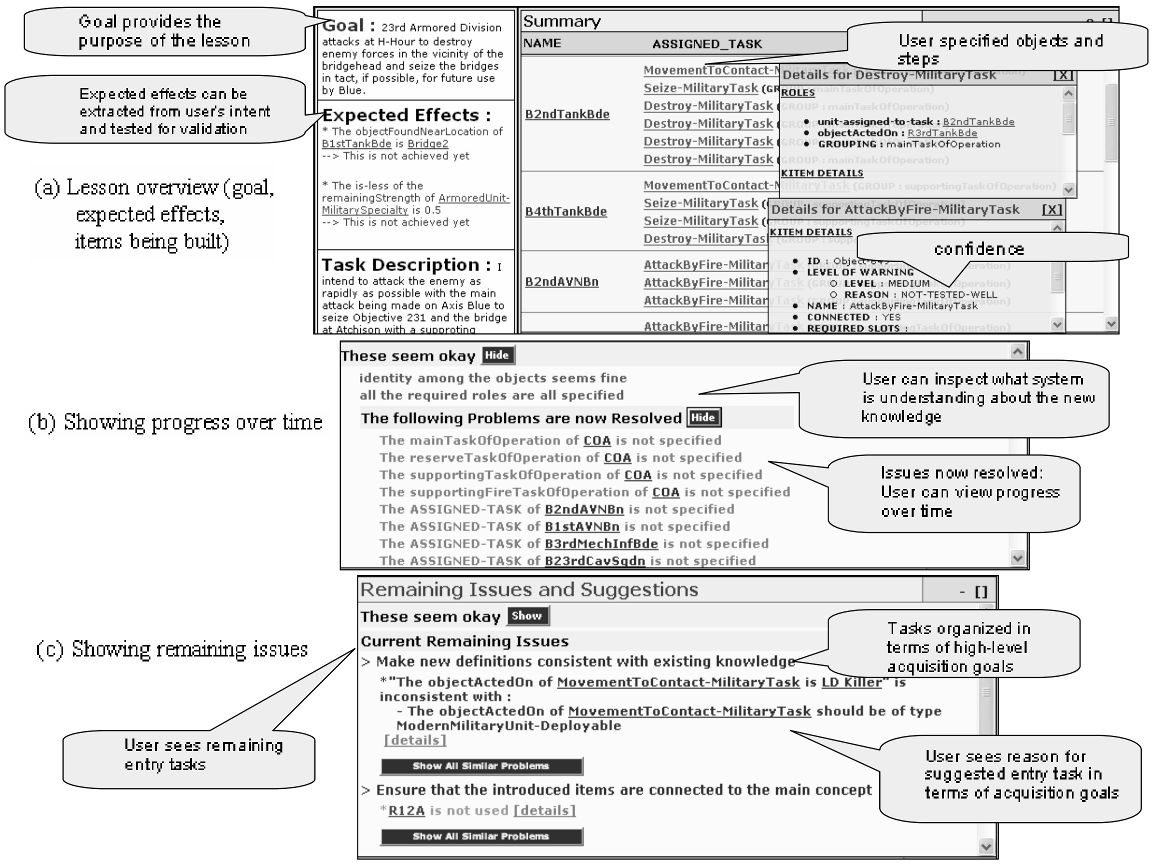

Figure 3: Acquiring military plans with SLICK.

Because acquisition strategies drive the interaction with the user, acquisition interfaces need to strike a balance between exploring and covering all possible strategies that users can follow and not overwhelming them with options that they are unlikely to choose in the first place. This is a very challenging problem and an area of future work.

4 Using SLICK: Preliminary User Feedback

SLICK has been used for acquiring two very different types of knowledge: biological process models and military plans. The basic acquisition tools were developed as a part of the DARPA RKF (Rapid Knowledge Formation) that aims to help end users, i.e., people without formal training in computer science, develop knowledge bases. Although the acquisition tools had various support for the users, the tools were rather passive in organizing various acquisition tasks and they were not able to actively participate in the learning process. SLICK was built as a front-end to these acquisition tools in order to make them more proactive, able to efficiently reason about learning activities with initiative in its dialogue with the user.

Figure 3 shows the SLICK interface for acquiring military plans (army courses of actions). With the basic entry tool, users describe their plans in terms of the steps (such as attack, seize, destroy, etc.) and the objects involved (military units, terrain features, etc.). Here SLICK is presenting a report on a plan being entered by a military officer, pointing out how the system is understanding the plan. As shown in Figure 3-(a), SLICK keeps track of the lesson goal and the user's intention (e.g., expected effect), which can be used to guide the user as well as to check if the plan is valid (e.g., intended effects are achieved). The summary window shows how the plan is being built, illustrating the essential elements of the plan: the list of involved objects and their tasks. It highlights the objects with potential problems (such as unassigned units) in red and confident subtopics are shown in blue. The user can check details of each item by clicking the interested items, as shown in Figure 3-(a). For example, SLICK presents confidence on knowledge items based on the number of times they were involved in testing. In past work, in user evaluations with other acquisition tools, we have found that subjects often had difficulty in understanding how they are making progress [9], and here the officers commented that the SLICK functionalities are very helpful for it. One of the officers said that the status report from SLICK is not only useful for the plan builder (the commander) but also can be sent out to other people (military units) who participate in the plan.

Towards the end of the lesson (i.e., building a plan), SLICK confirms that all the required roles (such as the information that existing knowledge indicates it must be provided) are specified, and identity among the objects are fine (none of the existing objects appear to be the same) (Figure 3-(b)). The user can view the progress by checking the issues resolved over time. When SLICK notices remaining issues, it also collects the sources of the problems so that it can help users understand the problems better. For example, Figure 3-(c) shows that there is an inconsistency between the plan and existing definitions in the KB because in the existing definitions, the 'objectActedOn' should be a military unit (ModernMilitaryUnitDeployable), but currently the user has assigned a phase line (a terrain feature) for it.

Note how SLICK's learning principles have derived these output. For example, in the figure, SLICK reports its understanding of the lesson and the remaining issues in terms of its goals such as "Make new definitions consistent with existing knowledge", "Ensure that the introduced items are connected to the main concept", "Ensure that the required roles are all specified", "Establish identity among the items", etc.

SLICK has been also applied to acquiring biological process models where learning goals that are active given the state of the lesson are shown to the users.

From these exercises, we have found that SLICK may help users more when the acquisition tasks have many steps involving various sub-tasks (searching, editing, testing, fixing, etc.) and the tasks require keeping track of the context of what needs to be achieved. For example, SLICK may be effectively used in building instructional systems. However, if the given acquisition task is very simple, with a small number of steps, then SLICK may not provide much help.

5 Conclusion

We have presented a new approach for interactive knowledge capture that can be used to extend existing tools with acquisition goals, learning strategies, and awareness annotations over the current state of the knowledge base in terms of its completeness and competence. Our system presents users with useful information regarding the progress made throughout the dialogue, current status of the new body of knowledge, goals that remain to be addressed, and suggested strategies to accomplish those goals. We believe that the information that the system is capturing about its current knowledge and its progress during the acquisition dialogue gives the user a crucial tool for externalization, i.e., an external record of the teacher/student interaction that helps the user visualize where the lesson is at, relieving users of a significant burden during the acquisition process. SLICK presented highlight problems in actual knowledge bases that their creators had neither noticed nor fixed.

We plan to extend the work on dialogue plans for acquisition tasks, and incorporate a plan recognition module that relates user commands with multi-step plans. We also would like to incorporate in our system other useful principles of student/teacher interactions. For example, tracking the history to limit the subnesting of lessons and to detect thrashing (defining something, then changing it to fix a problem, then changing it back and getting the problem again). Finally, we would like to use SLICK for building instructional systems. SLICK's learning goals may be useful for developing student models as well as building domain models.

References

[1] D. Ausubel, Educational psychology: A cognitive approach, New York: Holt, Rinehart and Winston (1968).

[2] J. Brown, and R. Burton, Diagnostic models for procedural bugs in basic mathematical skills. Cognitive Science, 2:155–191 (1978).

[3] M. Chi, N. deLeeuw, M. Chiu, and C. Lavancher, Eliciting self-explanations improves understanding, Cognitive Science, 18 (1994).

[4] A. Collins and A. Stevens, Goals and strategies of inquiry teachers. Advances in Instructional Psychology, 2:65–119 (1982).

[5] M. Core, J. Moore, C. and Zinn, Supporting constructive learning with a feedback planner. Proceedings of the AAAI Fall Symposium on Building Dialogue Systems for Tutorial Applications (2000).

[6] K. Forbus, and P. Feltovich, eds., Smart Machines in Education, AAAI press (2001).

[7] B. Fox, The Human Tutorial Dialog Project. Lawrence Erlbaum (1993).

[8] Y. Gil and J. Kim, Interactive Knowledge Acquisition Tools: A Tutoring Perspective, Proceedings of the 24th Annual Meeting of the Cognitive Science Society (2002).

[9] J. Kim and Y. Gil, Acquiring Problem-Solving Knowledge from End Users: Putting Interdependency Models to the Test, Proceedings of AAAI-2000 (2000).

[10] J. Kim and Y. Gil, Deriving Acquisition Principles from Tutoring Principles, Proceedings of Intelligent Tutoring Systems Conference (2002).

[11] M. Lepper, M. Woolverton, D. Mumme, and J. Gurtner, Motivational techniques of expert human tutors: Lesson for the design of computer-based tutors. In Lajoie, S. and Derry, S., (Eds.), Computers as Cognitive Tools (1993).

[12] T. Murray, Authoring intelligent tutoring systems: An analysis of the state of the art, International Journal of Artificial Intelligence in Education, 10:98–129 (1999).

[13] C. Rose, P. Jordan, M. Ringenberg, S. Siler, K. VanLehn, and A. Weinstein, Interactive conceptual tutoring in Atlas-Andes, Proceedings of AI in Education (2001).

[14] D. Sleeman, Inferring student models for intelligent computer-aided instruction. In R. Michalski, J. Carbonell, and T. Mitchell, eds., Machine Learning: An Artificial Intelligence Approach, Springer (1984).

[15] K. VanLehn, R. Freedman, J. Pamela, C. Murray, R. Osan, M. Ringenberg, C. Rose, K. Schulze, R. Shelby, D. Treacy, A. Weinstein, and M. Wintersgill, Fading and deepening: The next steps for Andes and other model-tracing tutors. Proceedings of Intelligent Tutoring Systems Conference, (2000).

[16] B. White and J. Frederiksen, Metacognitive facilitation: An approach to making scientific inquiry accessible to all. J. Minstrell and E. van Zee (Eds.), Inquiring into Inquiry Learning and Teaching in Science. (pp. 331-370) (2000).

[17] B. Woolf, and J. Allen, Spoken language tutorial dialogue. In Proceedings of the AAAI Fall Symposium on Building Dialogue Systems for Tutorial Applications (2000).

[18] B. Woolf, and D. McDonald, Building a computer tutor: Design issues. IEEE Computer, 17(9) (1984).