The design and architecture of Research Methods Tutor, a second generation dialog-based tutor

Peter Wiemer-Hastings

School of Computer Science,
Telecommunications, and Information Systems
DePaul University
Chicago, Illinois

Corresponding author:
Peter Wiemer-Hastings
DePaul University CTI
243 South Wabash Avenue
Chicago, Illinois 60604, USA
peterwh@cti.depaul.edu
+1 312 362 5736 (voice)
+1 312 362 6116 (fax)

Abstract:
RMT (Research Methods Tutor) is a dialog-based tutoring system that is designed to be used via the internet in order to augment traditional classroom experience. RMT was designed with a modular architecture to enable us to interchange and evaluate different tools and techniques for improving tutoring. The focus of this paper is to describe the architecture of the system and the current components that allow it to provide an effective complement to classroom teaching.

1 Introduction

Research on human-to-human tutoring has identified one primary factor that influences learning: the cooperative solving of example problems (Graesser, Person, & Magliano, 1995). Typically, a tutor poses a problem (selected from a relatively small set of problems that they frequently use), and gives it to the student. The student attempts to solve the problem, one piece at a time. The tutor gives feedback, but rarely gives direct negative feedback. The tutor uses pumps (e.g. "Go on."), hints, and prompts (e.g. "The groups would be chosen ...") to keep the interaction going. The student and tutor incrementally piece together a solution for the problem. Then the tutor often offers a summary of the final solution (Graesser et al., 1995).

Over the last 10 years, the field of dialog-based intelligent tutoring systems (DBITS's) has evolved and produced some quite successful systems that mimic this behavior. DBITS's have been implemented and have been demonstrated to be help learning in such areas as medical diagnosis (Hume, Michael, Rovick, & Evens, 1996; Freedman, Zhou, Glass, Kim, & Evens, 1998; Glass, 2001), geometry (Aleven, Popescu, & Koedinger, 2001), electronics (Zinn, Moore, Core, Varges, & Porayska-Pomsta, 2003), computer literacy (Wiemer-Hastings, Graesser, Harter, & the Tutoring Research Group, 1998), and physics (Graesser, Jackson, Mathews, Mitchell, Olney, Ventura, Chipman, Franceschetti, Hu, Louwerse, Person, & TRG, 2003) to name just a few. This paper describes a the architecture and the components of a DBITS called Research Methods Tutor (RMT). RMT was developed with two goals in mind: to provide a internet-based tutoring system that can augment classroom teaching, and to provide that in the form of a domain-independent, modular architecture that supports research in a variety of different tutoring techniques.

The paper begins with a high-level overview of the major components of the system. Then we focus on the individual components. As is the case for most

DBITS's, a critical aspect is its ability to understand student responses, that is, its natural language processing mechanism. RMT uses a technique called Latent Semantic Analysis (LSA) which learns a semantic representation from a corpus, and which provides a simple similarity metric between texts. RMT augments LSA with spelling correction and a keyword-matching mechanism for special circumstances. In addition to the natural language understanding techniques, RMT uses a modular dialog manager to control the interaction with the student, and a transition network to determine appropriate responses. These primary modules and the other support modules are described in the rest of this paper.

2 Overall architecture

RMT is a close descendant of the AutoTutor system, which was explicitly modeled after the behavior of human tutors. While AutoTutor incorporates a wide variety of artificial intelligence techniques, RMT was designed as a lightweight, modular system that would incorporate only those techniques required to provide educationally beneficial tutoring to the student. This section gives an overview of RMT's critical components, and these modules are described in detail in the following sections.

Figure 1 graphically depicts the relationships between RMT's major components. The dialog manager (described in the next section) is the hub of activity in the system. It controls the interactions with the user, presenting tutor responses, and requesting evaluation of the students' replies. On the left side of the graphic are the language understanding components, LSA with the help of a spellchecker. These are further described in section 4. On the right are the components which determine the type of the next tutor move (section 5). The Curriculum Script (section 9) provides all of the content material in the tutoring domain in a structure that combines questions, expected answers, and follow-up dialog moves. The data logger

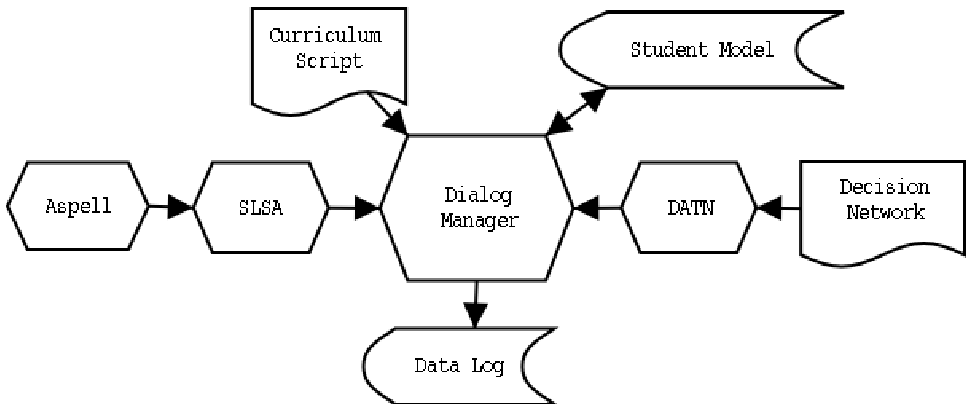

Figure 1: RMT Architecture

(section 7) is critical for research purposes and pedagogical purposes because it allows us to save information at a fine-grained level, and to flexibly retrieve it.

3 Dialog Manager

As shown in figure 1, the dialog manager (DM) is the central controller of the system. Because RMT is a web-based system, each tutoring session has its own dialog manager, and the DM maintains information about the parameters of the tutoring session and, crucially, the current state of the dialog. Traditional (non-internet-based) interactive programs can rely on the internal state of variables associated with some processing loop. But an internet server program must act as an agent which is responding to user requests, and thus, it must maintain enough state information to allow it to recreate the context of the dialog whenever it is invoked.

The DM receives student responses as posts from a web page, and then asks the Dialog Advancer Transition Network (DATN) to compute an appropriate tutor response. RMT was designed to also work in a text-only console mode. RMT's design includes an object-oriented parallel representation to provide the same functionality in both operating environments.

4 Understanding student contributions

RMT uses Latent Semantic Analysis (LSA) to evaluate student contributions. LSA was first developed for information retrieval — selecting query-relevant texts from a database (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990). Landauer and Dumais (1997) later showed that it accurately models human acquisition of word knowledge. LSA has also been shown to perform well at finding synonyms (Landauer, Laham, Rehder, & Schreiner, 1997), suggesting appropriate texts for students to read (Landauer et al., 1997), and even grading student essays (Foltz, 1996). AutoTutor was the first system to use LSA to "understand" student responses in an interactive tutoring system (Wiemer-Hastings et al., 1998), and it has subsequently been incorporated or evaluated for use by several other systems (Aleven et al., 2001; Rosé, Jordan, Ringenberg, S. Siler and, & Weinstein, 2001; Kanejiya, Kumar, & Prasad, 2003; Glass, 2001).

LSA evaluates a student response by comparing it to a set of expected answers. This works well in the tutoring setting because the tutor asks most of the questions and knows what types of answers (good and bad) the student is likely to produce. LSA calculates the similarity between two sentences (or texts of any size) using a vector-based representation that is learned from a large corpus of domain-relevant texts (Wiemer-Hastings, Wiemer-Hastings, & Graesser, 1999). The pattern of co-occurrence of words in the paragraphs of the corpus causes similar words to be represented with similar vectors. Word vectors are combined to represent entire texts, and the similarity between texts is determined by the cosine between the vectors (Landauer & Dumais, 1997). In the tutoring setting, LSA has been shown to evaluate student responses nearly as well as human raters with intermediate domain knowledge (Wiemer-Hastings et al., 1999). For tutoring, this is good news, because even peer tutors have been shown to produce significant learning gains (Bloom, 1984).

A strength of LSA is that it is robust to grammatical errors in the student inputs. RMT has additionally included an automatic spell-checking mechanism based on GNU Aspell, the most effective available (Atkinson, 2002). For words that are not recognized by LSA, RMT compares Aspell's suggested respellings to find the first one that is defined in LSA. This allows RMT to avoid asking the student to either re-enter the response, or to confirm which word was intended.

5 Dialog Act Transition Network

Each tutor "turn" can perform three different functions: evaluate the student's previous utterance (e.g. "Good!"), confirm or add some additional information (e.g. "The dependent variable is test score."), and produce an utterance that keeps the dialog moving. Like AutoTutor, RMT uses pumps (e.g. "What else?"), prompts (fill-in-the-blank sentences), and hints (follow-up questions) to try to get the student to add information about the current topic. RMT also asks questions, summarizes topics, and answers questions.

The DATN determines which type of response the tutor will give using a decision network which graphically depicts the conditions, actions and system outputs. Figure 2 shows a segment of RMT's DATN. When a student response is received, the diaman invokes the DATN and processing begins at the Start state. The paths through the network eventually join back up at the Finish state, not shown here.[1] On the arcs, the items marked `C` are the conditions for that arc to be chosen. The items labeled `A` are actions that will be performed. For example, on the arc from the

[1] If a traversal of the DATN does not end at the Finish state, then the DATN is incomplete.

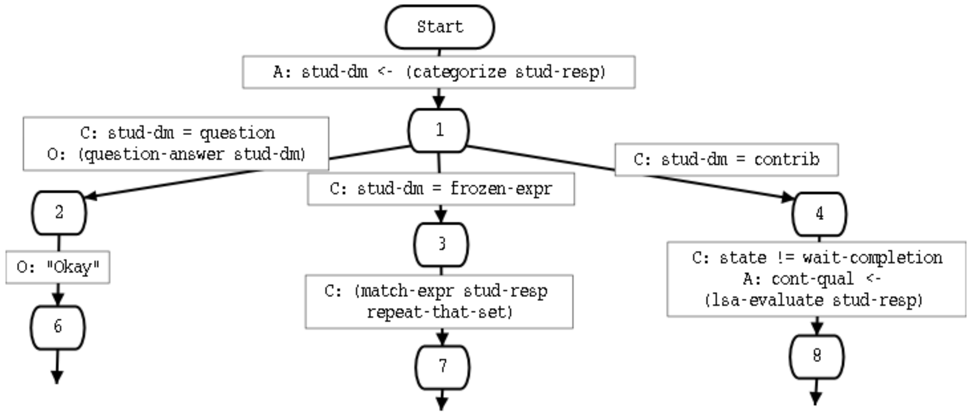

Figure 2: A partial decision network

start state, the DATN categorizes the student response. The items marked `O` are outputs — what the tutor will say next. Because this graph-based representation controls utterance selection, the tutor's behavior can be modified by simply modifying the graph. Although this technique has long been used for parsing natural language sentences, RMT is the only dialog-based ITS that uses it for determining the tutor's next utterance. This is another example of the modularity and extensibility of RMT.

In fact, the DATN is implemented on two levels. The basic level describes the structure of the network, i.e. the nodes and links, and provides the functionality for navigating around the networks and for verifying that they are syntactically well-formed. The semantic level of the network (the ueber-DATN) contains a description of the meanings of the registers that store information, the predicates that are used for conditions, the functions that are used for actions, and the types of outputs that the tutor can produce. This separation makes the framework more flexible and allows it to process different types of data.

6 Student modeling

Within the field of intelligent tutoring systems, a large amount of effort has gone into student modeling (VanLehn, Niu, Siler, & Gertner, 1998; Katz, Lesgold, Eggan, & Gordin, 1992, for example). A standard technique is to create a structured representation of the desired knowledge in the tutoring area and then layer on top of that the concepts that the student has demonstrated (Goldstein, 1982). But human tutors appear to have only very shallow knowledge of what their students know (Person, Graesser, Magliano, & Kreuz, 1994). For this reason, RMT follows AutoTutor's lead in tracking only the basic information about student performance: average response quality, topics covered, and a measure of student initiative (how much they drive the conversation). It is possible that more knowledge would help the tutor help the student better, but that has not yet been demonstrated.

7 Logging

For data collection purposes, RMT borrows a piece of wisdom from a very successful reading tutor called Project LISTEN, "Log everything" (Mostow & Aist, 2001). As it interacts with a student, RMT stores in a database information about each interaction. The database collects and relates the individual utterances and a variety of other variables, for example, the type and quality of a student response. The database also contains information about the students and the tutoring conditions that they are assigned to. Thus, in addition to providing data for the experiments described below, we will be able to perform *post hoc* analyses by selecting relevant tutoring topics. (For example, "Is there a difference in student response quality on Mondays and Fridays?")

As mentioned above, this information could also be used pedagogically. RMT could use information about the student's previous use of the system to guide it when selecting new topics. And the tutor could also use the student's pretest scores to focus on problem areas (although it currently does not do this).

8 Talking heads

As AutoTutor does, RMT uses an animated agent with synthesized speech to present the tutor's utterances to the student. In principle, this allows the system to use multiple modes of communication to deliver a richer message. For example, the tutor can avoid face-threatening direct negative feedback, but still communicate doubt about an answer with a general word like "Well" with the proper intonation. Furthermore, in relation to text-only tutoring, the student is more likely to "get the whole message" because they can not simply skim over the text.

There are also potential disadvantages to using talking heads. The synthesized speech may not be comprehensible to the students. It can be augmented with a "speech bubble" that simultaneously shows the words, but then the student might not notice any facial expressions. Furthermore, the rate at which the talking head speaks the text may be either too slow or too fast to accommodate the students patience and/or speed of comprehension.

RMT can also run in text-only mode via the internet (in addition to the console version mentioned above). This allows us to explore exactly what (if anything) the animated agents add to the tutoring situation.

9 Curriculum Script

A number of studies have shown that human tutors use a "curriculum script", or a rich set of topics which they plan to cover during a tutoring session (Graesser et al., 1995; McArthur, Stasz, & Zmuidzinas, 1990; Putnam, 1987). RMT's curriculum script serves the same function. It is the repository of the system's knowledge about the tutoring domain. In particular, it contains the topics that can be covered, the questions that the tutor can ask, the answers that it expects it might get from the students, and a variety of dialog moves to keep the discourse going. RMT's curriculum script currently contains approximately 2500 items in 5 topics. We believe that this gives us a reasonable starting point for using the tutoring system throughout a significant portion of a quarter-long class.

A portion of the (xml-formatted) curriculum script is shown in figure 3. The main questions are named, and can include ordering information to control the relative order of presentation. We have included three types of questions:

1. **Conceptual**: geared at getting descriptions of concepts or simple relationships in the domain,

2. **Analytic**: critical analysis of example scenarios, and

3. **Synthetic**: design-oriented tasks for testing given hypotheses.

This item includes an information delivery item which is presented before the question to "ground" it. It also includes a "picture" which can either be a graphic, or a textual description that supports the question. The `PGOOD` items contain the different aspects of an ideal answer that the tutor is trying to evoke from the student. There are 4 – 7 `PGOOD`s per question. Each `PGOOD` contains one or more `TARGET`s that the student answer is compared to. If the LSA match is sufficiently high (above the current empirically determined threshold of 0.5), that aspect is marked as covered, and is not discussed (although it may come up in the summary). The highest sub-threshold match is chosen by the tutor to discuss. The tutor will choose one of the associated dialog moves (marked as `PELAB`, `PHINT`, and `PPELAB`) to try to elicit the information from the student. Hints and prompts have associated completions which are specific strings that are matched against the ensuing student answer with both LSA and a keyword matching technique. If the match is sufficiently high, the aspect is marked as covered, and the tutor chooses a new one. If not, the tutor chooses a different dialog move from the same aspect.

```
<PICTURE-QUESTION-ANSWER NAME="Lighting1" TYPE="ANALYTIC">
  <INFO>Now I'll give you a description of an experiment, and ask
    you some questions related to the validity of the experiment.  The
    description is in this scenario. </INFO>
  <PICTURE TYPE="HTML">An experimenter wants to test whether an
    increase in lighting can help test-taking ability.  It is the
    researcher's hypothesis ...  </PICTURE>
  <QUESTION>The concept of internal validity in research ...   What
    is one way that other things besides the lighting may have
    influenced the test scores?</QUESTION>
  <PGOOD>
    <TARGET>Students took the exam at different times of the
     day.</TARGET>
    <PELAB>The 7th grade students took the exam in the morning, and
     the 8th grade students took the exam after lunch.</PELAB>
    <PHINT>
      <CONTENTS>When did each group take the exam?  </CONTENTS>
      <PHINTC>The 7th graders took it in the morning, and the 8th
       grades took it after lunch.</PHINTC>
    </PHINT>
    <PPROMPT>
      <CONTENTS>The two different grounps took the test at
       different</CONTENTS>
      <PPROMPTC>times of the day.</PPROMPTC>
    </PPROMPT>
  </PGOOD>
  ...
  <SUMMARY>Confounds are outside factors that vary along with the
  independent variable that keep us from knowing if the independent
  variable ...</SUMMARY>
</PICTURE-QUESTION-ANSWER>
```

Figure 3: A segment of the curriculum script

10 Testing

As a complement to the online tutor, we have implemented a dynamic testing system which presents multiple choice and short-answer essay questions and logs the student answers. Test questions are stored in XML format, so they are easily created and modified by instructors. For multiple-choice items, the correct answer is also encoded, so we can give immediate feedback to the students on their results. The testing component is also linked to the logger, so we can ensure that the student has completed enough of the discussion topics before they get a post-test, for example.

11 Conclusions

Because RMT is designed to be used in conjunction with classes on an everyday basis, there are obviously significant technical issues to overcome. In addition to the issues mentioned in the previous section, we plan on focusing on the natural language understanding mechanism to incorporate a variety of syntactic and discourse mechanisms in order to improve the system's understanding of the student replies.

We feel that in the long run, this type of system will be shown to be a valuable adjunct to classroom instruction. With a dialog-based tutoring system, the student can interact in a natural way using their own words. The process of constructing responses to the tutor's questions can in itself help the students "firm up the ideas" in their heads.

While the results of our current experiment indicate that the use of an animated agent "talking head" does not increase learning (and in fact, appeared to lead to degradation of the students' knowledge), we feel that further research is warranted on this topic. The limitations of our current agent may have interfered with the student's attention to the material under discussion.

In any case, RMT has been shown to help students learn the rather difficult material covered in Psychology research methods classes. As we continue to develop and refine the system, we hope that it can eventually become another standard mechanism for augmenting the students' education.

References

Aleven, V., Popescu, O., & Koedinger, K. R. (2001). Towards tutorial dialog to support self-explanation: Adding natural language understanding to a cognitive tutor. In *Proceedings of the 10th International Conference on Artificial Intelligence in Education*.

Atkinson, K. (2002). Spell Checker Test Kernel Results. Web page. url: http://aspell.net/test/, accessed Dec 7, 2003.

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher*, *13*, 4–16.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, *41*, 391–407.

Foltz, P. (1996). Latent semantic analysis for text-based research. *Behavior Research Methods, Instruments, and Computers*, *28*, 197–202.

Freedman, R., Zhou, Y., Glass, M., Kim, J., & Evens, M. (1998). Using rule induction to assist in rule construction for a natural-language based intelligent tutoring system. In *Proceedings of the 20th Annual Conference of the Cognitive Science Society*, pp. 362–367 Hillsdale, NJ. Erlbaum.

Glass, M. (2001). Processing language input in the CIRCSIM-tutor intelligent tutoring system. In Moore, J., Redfield, C., & Johnson, W. (Eds.), *Artificial Intelligence in Education*, pp. 210–221 Amsterdam. IOS Press.

Goldstein, I. (1982). The genetic graph: A representation for the evolution of procedural knowledge. In Sleeman, D. H., & Brown, J. S. (Eds.), *Intelligent Tutoring Systems*, pp. 51–77. Academic Press.

Graesser, A. C., Person, N. K., & Magliano, J. P. (1995). Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology*, *9*, 359–387.

Graesser, A., Jackson, G., Mathews, E., Mitchell, H., Olney, A., Ventura, M., Chipman, P., Franceschetti, D., Hu, X., Louwerse, M., Person, N., & TRG (2003). Why/AutoTutor: A test of learning gains from a physics tutor with natural language dialog. In *Proceedings of the 25rd Annual Conference of the Cognitive Science Society* Mahwah, NJ. Erlbaum.

Hume, G. D., Michael, J., Rovick, A., & Evens, M. W. (1996). Hinting as a tactic in one-on-one tutoring. *The Journal of the Learning Sciences*, *5*, 23–47.

Kanejiya, D., Kumar, A., & Prasad, S. (2003). Automatic Evaluation of Students' Answers using Syntactically Enhanced LSA. In *Proceedings of the Human Language Technology Conference (HLT-NAACL 2003) Workshop on Building Educational Applications using NLP*. available at: http://www.cse.iitd.ernet.in/ eer99010/pub/hlt-naacl03.pdf.

Katz, S., Lesgold, A., Eggan, G., & Gordin, M. (1992). Modelling the student in Sherlock II. *International Journal of Artificial Intelligence in Education*, *3*, 495–518.

Landauer, T. K., Laham, D., Rehder, R., & Schreiner, M. E. (1997). How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. In *Proceedings of the 19th Annual Conference of the Cognitive Science Society*, pp. 412–417 Mahwah, NJ. Erlbaum.

Landauer, T., & Dumais, S. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, *104*, 211–240.

McArthur, D., Stasz, C., & Zmuidzinas, M. (1990). Tutoring techniques in algebra. *Cognition and Instruction*, *7*, 197–244.

Mostow, J., & Aist, G. (2001). Evaluating Tutors that listen. In Forbus, K., & Feltovich, P. (Eds.), *Smart Machines in Education*, pp. 169–234. AAAI Press, Menlo Park, CA.

Person, N. K., Graesser, A. C., Magliano, J. P., & Kreuz, R. J. (1994). Inferring what the student knows in one-to-one tutoring: The role of student questions and answers. *Learning and Individual Differences*, *6*, 205–229.

Putnam, R. T. (1987). Structuring and adjusting content for students: A study of live and simulated tutoring of addition. *American Educational Research Journal*, *24*, 13–48.

Rosé, C., Jordan, P., Ringenberg, M., S. Siler and, K. V., & Weinstein, A. (2001). Interactive Conceptual Tutoring in Atlas-Andes. In *Proceedings of AI in Education 2001 Conference*.

VanLehn, K., Niu, Z., Siler, S., & Gertner, A. (1998). Student modeling from conventional test data: A Bayesian approach without priors. In Goettl, B., Halff, H., Redfield, C., & Shute, V. (Eds.), *Intelligent Tutoring Systems, Proceedings of the 4th international conference*, pp. 434–443 Berlin. Springer.

Wiemer-Hastings, P., Graesser, A., Harter, D., & the Tutoring Research Group (1998). The foundations and architecture of AutoTutor. In Goettl, B., Halff, H., Redfield, C., & Shute, V. (Eds.), *Intelligent Tutoring Systems, Proceedings of the 4th International Conference*, pp. 334–343 Berlin. Springer.

Wiemer-Hastings, P., Wiemer-Hastings, K., & Graesser, A. (1999). Improving an intelligent tutor's comprehension of students with Latent Semantic Analysis. In Lajoie, S., & Vivet, M. (Eds.), *Artificial Intelligence in Education*, pp. 535–542 Amsterdam. IOS Press.

Zinn, C., Moore, J. D., Core, M. G., Varges, S., & Porayska-Pomsta, K. (2003). The BE&E Tutorial Learning Environment (BEETLE). In *Proceedings of the Seventh Workshop on the Semantics and Pragmatics of Dialogue (DiaBruck 2003)*. Available at http://www.coli.uni-sb.de/diabruck/.