Metacognition Learning
DOI 10.1007/s11409-007-9010-0

Designing for metacognition—applying cognitive tutor principles to the tutoring of help seeking

Ido Roll · Vincent Aleven · Bruce M. McLaren · Kenneth R. Koedinger

Received: 23 December 2006 / Accepted: 1 May 2007

Abstract Intelligent Tutoring Systems have been shown to be very effective in supporting learning in domains such as mathematics, physics, computer programming, etc. However, they are yet to achieve similar success in tutoring metacognition. While an increasing number of educational technology systems support productive metacognitive behavior within the scope of the system, few attempt to teach skills students need to become better future learners. To that end, we offer a set of empirically-based design principles for metacognitive tutoring. Our starting point is a set of design principles put forward by Anderson et al. (*Journal of the Learning Sciences*, 4:167–207, 1995) regarding Cognitive Tutors, a family of Intelligent Tutoring Systems. We evaluate the relevance of these principles to the tutoring of help-seeking skills, based on our ongoing empirical work with the Help Tutor. This auxiliary tutor agent is designed to help students learn to make effective use of the help facilities offered by a Cognitive Tutor. While most of Anderson's principles are relevant to the tutoring of help seeking, a number of differences emerge as a result of the nature of metacognitive knowledge and of the need to combine metacognitive and domain-level tutoring. We compare our approach to other metacognitive tutoring systems, and, where appropriate, propose new guidelines to promote the discussion regarding the nature and design of metacognitive tutoring within scaffolded problem-solving environments.

Keywords Meta-cognition · Help seeking · Intelligent tutoring systems · Cognitive tutors · Instructional design principles

One of three key instructional design principles proposed in "How People Learn" (Bransford et al. 2000) is to support metacognition. Having better metacognitive skills can help students learn better within a learning environment (such as a tutoring system or traditional classroom), and furthermore, can help them self-regulate their learning across domains and contexts. Indeed, research shows that well-designed metacognitive instruction

I. Roll (✉) · V. Aleven · B. M. McLaren · K. R. Koedinger
Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: idoroll@cmu.edu

has a positive impact on metacognitive behavior and subsequently on domain learning, for example, instruction on the use of debugging skills (Carver and Mayer 1998) or self-regulatory strategies (Bielaczyc et al. 1995).

Supporting metacognition is a challenge that has been tackled by an increasing number of tutoring systems. Several researchers have developed systems focused on self-explanation (e.g., Aleven and Koedinger 2002; Conati and VanLehn 1999); others teach their students to monitor learning of computer agents (e.g., Biswas et al. 2004; Reif and Scott 1999); Baker et al. (2006a, b) encourage students to not "game the system" (i.e., to not try to guess answers or abuse hints repeatedly); and Gama (2004) focuses on coaching a reflection process. Azevedo describes several characteristics of metacognitive support by tutoring systems, including student control over setting subgoals and the use of learning resources; having a model to support the use of both metacognitive and domain-level skills; human or virtual assistance; and the use of metacognitive skills by the student within the learning context (Azevedo 2005b).

However, not every system that supports metacognition also teaches it. Merely channeling students toward more productive metacognitive behavior may lead to increased learning at the domain level, but is not likely to improve the students' general learning skills. The improved metacognitive behavior in such cases is not likely to persist beyond the scope of the tutored environment; one might say that it is the result of a metacognitive "crutch," as opposed to a metacognitive "scaffold." For example, following research that showed that students lack efficient help-seeking skills (Aleven and Koedinger 2000), a 2 seconds time threshold between consecutive hint requests was added to the Cognitive Tutor. This type of scaffold improved the way students ask for hints within the system, but did not improve students' general hint-reading skills, as discussed later. In this paper we focus on guidelines for teaching metacognition, and not merely supporting it.

Relatively few tutoring systems can be considered *metacognitive* tutoring systems under the requirements stated above. An example of such system is the SE-Coach (Conati and VanLehn 1999), which attempts to teach, and not only require, the use of self-explanations in learning Physics. In our work with the Help Tutor discussed in the current paper, we attempt to improve students' help-seeking behavior in a cross-domain, cross-environment fashion. Other systems have been shown to be successful in tutoring metacognition using the support of a human tutor (e.g., Azevedo et al. 2004; White and Frederiksen 1998).

Metacognitive vs. cognitive tutoring

It is commonly agreed that metacognitive knowledge includes two main types of skills (Brown 1987): knowledge of knowledge ("What is my knowledge gap?") and regulation of knowledge ("What should I do to overcome it?"). Metacognitive tutoring has a number of characteristics that make it fundamentally different from domain-level tutoring. First, metacognition is somewhat domain independent in nature; ideally, the metacognitive knowledge that students acquire should be flexible enough to be applied while learning new domains, in a variety of different learning environments. Second, metacognitive tutoring is usually situated within the context of domain learning, and thus imposes extra cognitive load. Third, metacognitive learning goals are often perceived by students as secondary to the domain learning goal, or even as insignificant. Finally, while for most domains the targeted knowledge is independent of the student, this is not the case in metacognitive tutoring, in which the "correct" answer depends on students' knowledge, performance, motivation, and

goals at the domain level, as well as on characteristics of the task to be completed. These unique characteristics impose special challenges for the design of metacognitive tutoring.

Within the traditional classroom, a number of programs for teaching students specific metacognitive processes have been successful, in the sense that they led to better domain learning (Carver and Mayer 1998; Schoenfeld 1992). However, by and large, these successes are hard to achieve (Resnick 1987).

Considerable progress has been made in designing for metacognition within open learning environments, such as inquiry (Luckin and Hammerton 2002; Quintana et al. 2005; White and Frederiksen 1998), discovery (de Jong and van Joolingen 1998), and hypermedia (Azevedo 2005a) environments. However, not much is known yet regarding design guidelines for metacognitive tutoring in Intelligent Tutoring Systems (ITS). Typically, these systems guide students in problem-solving activities. They offer a rich problem-solving environment, and use a cognitive model of the domain to adapt instruction to individual learners. By tracing students' actions and knowledge relative to a cognitive model, the ITS can tailor the curriculum (Corbett and Anderson 1995; Koedinger et al. 1997), the scaffold (Razzaq et al. 2005), the feedback (Corbett and Anderson 2001), or any combination of these to the students' needs. These systems have been shown to approach the effectiveness of a good one-on-one human tutor (Koedinger and Corbett 2006).

Recently, an increasing number of ITS attempt to support metacognitive learning. Various researchers describe methods for modeling metacognitive knowledge (Aleven et al. 2006; Baker et al. 2006b; Bunt and Conati 2003). Yet, guidelines concerning the *pedagogical* and *interactive* aspects of metacognitive tutoring are currently less available. One exception is Gama (2004) who presents two such guidelines: (1) Not to add cognitive load and (2) to help students recognize the importance of metacognitive learning goals.

In this paper we formulate and discuss a set of empirically-based design guidelines for metacognitive tutoring in ITS. We use Anderson et al. (1995) as the basis for our work. They offer a comprehensive set of design principles on which Cognitive Tutors, a prominent type of ITS, are based. Since we attempt to tutor metacognition using Cognitive Tutors, their guidelines offer an interesting framework for us to explore. Given the proven success of Cognitive Tutors at domain-level tutoring (e.g., Koedinger et al. 1997) we expect these principles to be applicable also at the metacognitive level. Yet, the different challenges imposed by metacognitive tutoring require, at least, some adaptation. We focus on a specific aspect of metacognition, namely, help seeking, and study it within a context of "tutored problem solving," that is, problem-solving practice with the help of a Cognitive Tutor.

The help tutor—a metacognitive tutor to teach help seeking

Students' help-seeking behavior while working with ITS is known to be suboptimal (see Aleven, Stahl, Schworm, Fischer, and Wallace, 2003 for an extensive review). By help-seeking skills we mean the strategies involved in identifying the need for help, selecting appropriate sources of help, eliciting the needed information, and applying the help that was received. Productive help seeking requires the two main aspects of metacognitive knowledge mentioned earlier: knowledge of knowledge ("Do I know enough to succeed on my own?") and regulation of knowledge ("How can I obtain additional information I may need?").

Students perform different types of help-seeking errors. In particular, students often avoid help when needed (e.g., by repeatedly guessing instead of asking for help) or abuse help (e.g., by asking for the most elaborated hint that conveys the answer immediately, without attempting to read or reflect upon hints that explain the reasons behind the answer).

In our research we have attempted to teach students better help-seeking skills while working with Cognitive Tutors. Cognitive Tutors (Koedinger et al. 1997; see Fig. 1) are a leading family of ITS, currently used by students in over 2,000 middle- and high-schools in the United States. Cognitive Tutors are available in several full-year math curricula (such as Geometry and Algebra) and combine two class periods per week of Cognitive Tutor work with three class periods per week of classroom instruction, including many small-group activities. The Algebra Cognitive Tutor was recently recognized by the What Works Clearinghouse as one of only two curricula in Algebra proven to improve learning (Morgan and Ritter 2002) according to the highest standard of scientific rigor.

The Help Tutor, which we use as a case study throughout this paper, is an add-on tutoring agent that can be integrated with different Cognitive Tutors. In line with the Cognitive Tutor approach, the Help Tutor traces students' actions relative to a (meta) cognitive model of help-seeking (Aleven et al. 2006; see Fig. 2). The help-seeking model is a prescriptive model of effective help-seeking behavior. The model uses several parameters (such as estimated mastery of the domain knowledge involved in the step and previous interaction on the same step) to predict what actions would be most useful at each point of the learning process: A solution attempt, hint request, glossary search, or asking the teacher

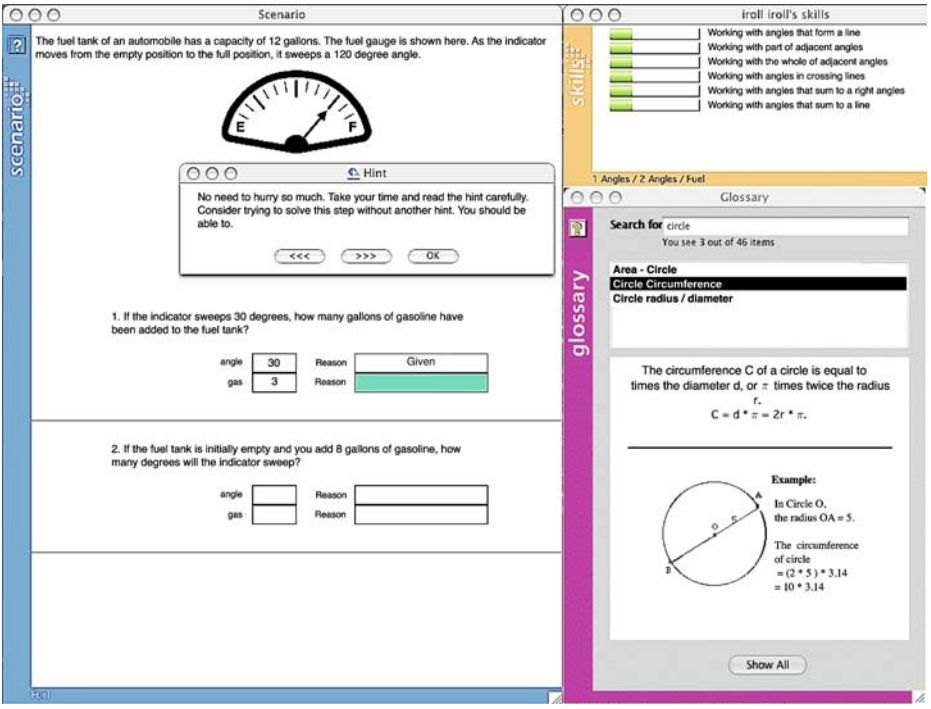

Fig. 1 The Help Tutor is integrated with a Cognitive Tutor for geometry. It traces students' actions within the Scenario window (*left*) against a model of effective help-seeking behavior. The Help Tutor uses several parameters such as estimated skill level (*top right*) and previous interactions (e.g., whether the student has already made multiple unsuccessful attempts at solving the given step) in order to arrive at a set of "acceptable" metacognitive actions. Deviations from these actions result in metacognitive error messages (*pop-up window in the center*), specifying the error, the appropriate action to take, and the general applicable metacognitive rule

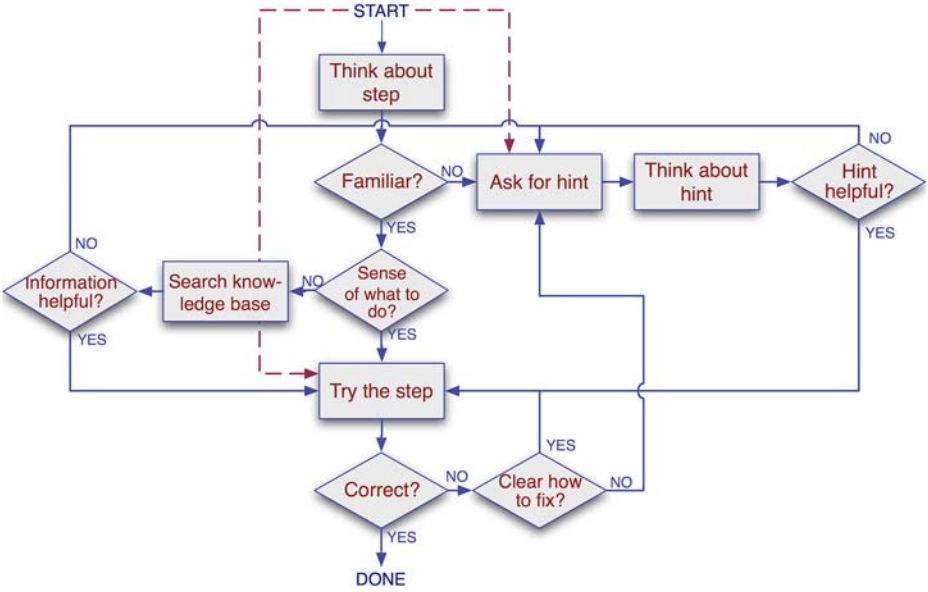

Fig. 2 The help-seeking model describes ideal help-seeking behavior with the Cognitive Tutor. When actions of students deviate form the model they are being classified as one of several types of metacognitive errors, and immediate and tailored feedback is given to the student. For example, the *dotted lines* demonstrate errors in which the student asks for hint, or attempts to solve, too quickly

a question. When the student's action deviates from the model, it is classified as one of several types of help-seeking errors, which triggers an immediate and tailored feedback message. The error feedback includes an explanation about the nature of the error and a recommendation for a better action, for example, "Even though you have missed this step, you probably know enough to solve it without a hint." The Help Tutor is integrated with the Geometry Cognitive Tutor so that students practice help seeking in the actual learning context, as shown in Fig. 1. The system traces each student action both with respect to the help-seeking model and with respect to the domain-level model. When feedback on the same action is generated by both models, the Help Tutor uses a simple conflict resolution strategy to determine which message will be displayed, so that students do not receive more than a single message at any point in time (Aleven et al. 2005).

The Help Tutor has been designed iteratively, using offline analysis, pilot studies, and in vivo classroom studies (see Table 1). We began by studying the help-seeking errors students make, as well as reviewing the relevant literature about help seeking in ITS and in traditional classrooms (Aleven et al. 2003; Aleven and Koedinger 2000). Based on that data we designed and evaluated the help-seeking model (referred to in this paper as study 1), and then evaluated it using data sets from different domains (referred to as study 2). At this point we designed the help-seeking feedback in the form of help-seeking error messages. The Help Tutor was evaluated first in a small-scale pilot study within a school (study 3), and following that, with 60 students in two high schools who used it for 3 weeks (study 4). After each of these steps the Help Tutor was tuned and improved based on previous findings. The most recent study evaluated the Help Tutor and classroom instruction with 80 students working with the system for 2 months (study 5).

Table 1 Evaluation studies of the Help Tutor

Study	Goal	Methodology	Main findings	Further details
1	Design the help-seeking model	Log-file analysis	73% of students' actions were classified as different types of help-seeking errors. These errors were significantly negatively correlated with learning (p=−0.65, p<0.0005)	(Aleven et al. 2006)
2	Evaluate the model across domains and cohorts	Log-file analysis	Students' errors in two different Cognitive Tutors were highly correlated (r=0.89, p<0.01)	(Roll et al. 2005)
3	Implement and pilot the Help Tutor	Pilot	Students improved the help-seeking behavior while working with the tutor	(Aleven et al. 2005)
4	Evaluate the Help Tutor	Randomized experiment with 60 students	Students improved several aspects of their help-seeking behavior. No improved learning at the domain level was observed	(Roll et al. 2006)
5	Evaluate the combination of the Help Tutor, preparatory Self-assessment sessions, and help-seeking classroom instruction	Experiment with 80 students	Under analysis	(Roll et al. 2007)

So far the Help Tutor has achieved mixed results. On the one hand, it has improved students' help-seeking behavior within the Cognitive Tutor. For example, in study 4 we found that students working with the Help Tutor asked to see the bottom-out hint significantly less often than the Control group students (46 vs. 72% respectively, p<0.001). On the other hand, the improved help-seeking behavior did not transfer to a paper-and-pencil evaluation of students' help-seeking strategies, and we did not yet see improved learning at the domain level.

Instructional principles for metacognitive ITS

Based on our experiences with the Help Tutor we examine the relevance of the principles stated in Anderson et al. (1995) to metacognitive tutoring. We sort the principles into three groups, as suggested by Carver (2001):

- Goals, which describe the design of metacognitively appropriate learning objectives for ITS
- Instruction, which discusses the design of the instructional means, interaction style, and pedagogy to be used; and
- Assessment, which discusses the evaluation of the metacognitive tutoring

Metacognitive goals: what should be taught?

Principle 1: Represent student competence as a production set (Anderson #1)

We find Anderson's first principle to be fully applicable at the metacognitive level. In the Cognitive Tutors, a domain-level cognitive model is used to describe the procedural knowledge students should acquire. Similarly, a metacognitive model can be built, comprised of production rules, to encompass the desired metacognitive learning goals. Figure 2 shows the model we implemented to describe desired help-seeking behavior. This model is the basis for the production rule set used to trace students' actions in the tutor.

Such models can be designed using a variety of methods. In addition to traditional cognitive modeling methods such as think-aloud protocols and log file analysis, students' learning gains at the domain level can inform the design of the metacognitive model. In developing the model, we ran it off-line against existing log data of students' interactions with the Geometry Cognitive Tutor. By correlating the metacognitive actions generated by the model to students' learning outcomes, we were able to identify (metacognitive) actions that are associated with productive learning (at the domain level), and (metacognitive) actions that are not. In subsequently refining the model, we tried to maximize the correlation between the modeled metacognitive behavior and students' domain-level learning. Aleven et al. (2006) describe the process by which we attempted to improve the help-seeking model, changing the way it handles repeated errors by students. Further changes based on log-file analysis were made following study 3, in order to reduce the proportion of student actions that do not conform to the model (and hence will be designated as metacognitive errors by the Help Tutor) from 73 to 17% while maintaining the same correlation with learning. We made the model focus on errors that were highly negatively correlated with learning (such as rapidly repeating hint requests) while changing it so that it no longer "outlawed" other actions previously considered to be metacognitive errors (such as fast attempts at solving steps by skilled students).

Principle 2: Set explicit declarative, procedural, and dispositional learning goals of the desired metacognitive skill (new principle)

While most teachers and curriculum designers would agree that instruction should address appropriate metacognitive goals, it is rare that such goals are stated explicitly. In our work with the Help Tutor, we found out that we should specify clearly, simply, and accurately what help-seeking behavior we want students to acquire. Following Anderson's principle #1 (described above), our original thought was to focus on the procedural help-seeking skills to be taught. At the time, this approach seemed right, since the main goal of the Help Tutor is to improve students' help-seeking *behavior*. However, since then we have found that it is important that the instruction focuses on declarative knowledge of help seeking as well. In study 5, a declarative help-seeking assessment revealed that many students lack an adequate conceptual understanding of help seeking. For instance, in response to one of the questions in this assessment, 42% of the students reported that when the tutor poses a tough problem on which their friends have made progress, the appropriate action to perform would be to ask immediately for the bottom out hint (that gives away the answer) without attempting to solve it first. This and other examples demonstrated that declarative learning goals should be supported in addition to procedural goals.

Having the appropriate declarative and procedural knowledge in place may not be sufficient for successful application of metacognitive skills, however, as was illustrated in one of our pilot studies (study 3), in which we observed how students use the Help Tutor.

One student repeatedly avoided help even though he clearly needed it. When asked whether the system was right to suggest that he ask for hints, he replied: "Yes, I needed them." However, when asked why he had not followed these recommendations, he replied: "Real men never ask for help."

Quite often, the metacognitively correct approach takes more time. This is the case with self-explanation, as well as with help seeking. When classroom culture at times encourages performance goals and quick progress within the curriculum, students may choose suboptimal learning strategies, such as repeated guessing or clicking through hints (termed "Gaming the System"; Baker et al. 2004). It is important to offer students motivational support that will encourage the use of desired metacognitive skills. For example, del Soldato suggests a system that adapts to students' motivational, as well as cognitive, state (Del Solato and du Boulay 1995).

To summarize, in the Help Tutor project, we have three sets of goals:

- Declarative: Improve students' knowledge of the source of help that should be used in the various situations.
- Procedural: Improve students' behavior in the online learning system.
- Dispositional: Improve students' understanding of the importance of appropriate help-seeking behavior.

Instruction: how should these goals be achieved?

Principle 3: Promote an abstract understanding of the problem-solving knowledge (Anderson #4)

Anderson et al. (1995, p. 180) describe the way Cognitive Tutors reinforce abstraction through "the language of our help and error messages." Recently, Koedinger and Corbett (2006) rephrased this principle: *Promote a correct and general understanding of the problem-solving knowledge*. We believe that this principle can be applied in a very similar manner in the metacognitive domain. The instruction and practice of metacognitive skills should emphasize, and link, the three types of learning goals: declarative, procedural, and dispositional. For example, the help-seeking error messages specify the error made by the student, the general rule that should be learned, and provide appropriate motivational support (e.g., "by clicking through hints you may solve the problem faster, but you will not learn, and you may have similar problems the next time you encounter a similar problem").

On top of phrasing the instructional messages appropriately, another aspect of metacognition can be used to help students acquire properly-abstracted metacognitive knowledge: its domain-independent nature. While students are accustomed to "putting aside" the skills acquired at the domain level once they are done with an instructional unit, they should not do so with metacognitive skills. One way to promote such abstraction is to provide metacognitive instruction in several domains. In study 2 we studied two different groups of students working with two different Cognitive Tutors in different areas of mathematics (high-school geometry and middle-school data analysis). The overall pattern of metacognitive errors students made was remarkably similar ($r=0.89$, $p<0.01$). The high correlation suggests that student apply similar metacognitive strategies (and thus, also errors) across tutoring systems. Therefore, it may be beneficial to apply the same (or similar) metacognitive tutoring techniques in different instructional units. To evaluate this hypothesis, in study 5 students worked with the Help Tutor over two instructional units—Angles and Quadrilaterals (A good next step would be to investigate whether such tutoring

is even more beneficial when done across learning environments, not just across instructional units.) In study 4 we found a significant correlation between the quality of students' help-seeking behavior in the tutor (as evaluated by the help-seeking model) and in the paper-and-pencil test (as evaluated by their use of the embedded hints in the paper test; $r=0.5$, $p<0.01$). Therefore, in the next study (study 5), we added complementary declarative instruction in a traditional classroom setting, in order to promote abstraction of the principles across learning environments.

Principle 4: Provide immediate feedback on errors (Anderson #6)

As Corbett and Anderson (2001) showed, immediate feedback contributes to learning. Koedinger and Corbett (2006) later restated this principle as follows: *Provide immediate feedback on errors relative to the model of desired performance*. Cognitive Tutors apply this principle by giving immediate feedback to students when they make errors. This principle is applicable also in tutoring metacognition. Mathan and Koedinger (2005) evaluated a Cognitive Tutor that teaches skills related to the coding of formulas in Microsoft Excel, and compared two versions of feedback based on different cognitive models. One version of the tutor used a domain-level model of Excel coding skills to trace students' performance. This tutor gave feedback when the student made a mistake in coding an Excel formula (i.e., a domain-level error). In the other version, the feedback was based on a model that included, in addition to the Excel coding skills, a metacognitive component, namely, self monitoring. According to this model, students needed to notice their own domain-level mistakes before moving on to the following step. This tutor provided the feedback once the student failed to notice and correct the error (instead of after the error itself). The latter kind of feedback led to increased learning.

We apply the principle of direct feedback within the Help Tutor by giving students immediate feedback on their help-seeking errors. Yet, in order to better fit the metacognitive nature of the instruction, a number of exceptions are made. First, when students commit a metacognitive error (such as attempting to solve a step too quickly) but get the answer right at the domain level, no metacognitive feedback is given. It was thought that students are not likely to pay attention to (negative) metacognitive feedback immediately following a successful attempt (e.g., feedback saying "even though you got this step right, you should have spent more time on it," however well-intentioned, might not be very effective). Another exception is made when both a domain-level and a metacognitive-level feedback message are available following a student action. In order to reduce cognitive load, we implemented a prioritizing algorithm, which chooses which content to display at each situation (Aleven et al. 2005). When the student commits an error, feedback messages with domain-level content (e.g., messages pointing out why the step is wrong, or what the students should be doing instead) receive priority over those focused on improving students' metacognitive behavior. When no domain-level feedback message is available (as happens often in Cognitive Tutors—most of the time, they provide only implicit feedback on domain-level errors), any applicable metacognitive message is displayed. The opposite occurs when the student asks for a hint – the Help Tutor messages (which provide feedback on the appropriateness of asking for a hint in the given situation, taking into account the student's current estimated knowledge level) receive priority over domain-level hint messages, in order to encourage students to view only as many hints as needed. For example, a domain-level hint may be pre-empted by a message from the Help Tutor emphasizing the value of reading hints deliberately, or suggesting that the student knows enough to solve the step without further hints from the tutor.

We emphasize that metacognitive tutoring systems should *let students err* (i.e., they should not prevent metacognitive errors). This point may seem obvious at the domain level—students should be allowed to make errors when solving problems, so as to learn to avoid them, and the system should not take over the hard parts of the task until the student has mastered them. Scaffolding metacognition may prevent students from making errors at the metacognitive level. For example, the geometry Cognitive Tutor displays a hint after three consecutive errors were made. This mechanism does not require the student to recognize her need for help. Since most ITS estimate the student's knowledge level, learning decisions can be made by the system for the students. This type of scaffold may be effective at the domain level. For example, Wood and Wood (1999) describe a contingent tutor that automatically adapts the hint level to the students' needs. However, empirical results suggest that metacognitive scaffolding may not yield metacognitive learning. For example, following earlier findings about help misuse, a 2-seconds delay was added to Cognitive Tutors in between hints, to prevent rapid repeated hint requests (which typically represent an attempt to get the system to reveal the answer to a problem step). When this scaffold was removed in study 4, students reverted back to quickly "clicking through" hints: 30% of the requested hints were viewed for less than 2 s by students who did not work with the Help Tutor.

Principle 5: Support metacognition before, during, and after the problem-solving process (an adaptation of Anderson's principle #3: Provide instruction in the problem-solving context)

At the domain level, students are often given instruction on a given skill or topic before they are asked to solve problems. Further instruction is given *during* problem solving, via hints and error messages. The successful completion of a problem is rarely followed by instruction on the same topic. However, the timing of metacognitive tutoring presents an interesting challenge from the viewpoint of instructional design. On the one hand, there are good reasons for giving instruction on metacognitive skills in the context of (and *during*) domain problem solving. What better opportunity to help students learn to seek help at appropriate times than at the very moment that they need it. At the same time, these challenging situations are the least suitable context for adding metacognitive instruction, due to the high cognitive load that they impose even without an explicit focus on metacognition. First, the student is likely to pay little attention to metacognitive content since her attention is focused on solving the problem. Second, metacognitive instruction may distract the student and thus hinder learning at the domain level. Third, metacognitive feedback at this time may be somewhat discouraging.

Several approaches for timing the metacognitive instruction can be used. Gama (2004) describes a system in which all metacognitive interaction precedes or follows the domain-level problem solving, which therefore remains unchanged. However, students may not pay attention to delayed feedback on their actions once they have solved a problem correctly. Also, as noted earlier, delayed feedback is often less effective than immediate feedback for the acquisition of cognitive skills (Corbett and Anderson 2001). Reif and Scott (1999) present an interesting approach: The metacognitive tutoring system and the student repeatedly switch roles: At times, the student is engaged in problem solving, while the ITS monitors her performance. At other times, it is the student who monitors the ITS's performance.

We believe that metacognition should be practiced within a learning context similar to the one in which it should later be applied. Therefore, we chose to give metacognitive feedback during the problem solving itself, directly applying Anderson's principle #5 at the metacognitive level. In addition, we added preparatory help-seeking sessions. In these sessions, students are introduced to the (domain-level) skills to be practiced in the

subsequent Cognitive Tutor unit, and self-assess their proficiency with respect to these skills (see Fig. 3). They work in an environment, low on cognitive demand with respect to domain-level skills, in which the focus is on help-seeking skills and their links to relevant domain knowledge. Later, when students solve problems with the Cognitive Tutor, the immediate feedback from the Help Tutor serves more as a reminder to what was learned during these preparatory sessions than as novel instruction. An ideal solution should probably balance all components: preparatory sessions with a strong metacognitive focus, immediate tailored feedback on metacognition during problem solving, and reflection exercises following it.

Principle 6: Minimize working memory load (Anderson #5)

Gama (2004) argues cogently that it is important not to unduly increase cognitive load when adding metacognitive instruction to an environment for domain-level learning. Increased load is especially likely to occur when metacognitive instruction is given during domain problem solving, as is done by the Help Tutor.

A possible solution to that problem is to embed the metacognitive content within the cognitive instruction in a manner that does not draw attention to another layer of instruction. However, seamless integration of the cognitive and metacognitive content does not stress the metacognitive goals. Students are likely to pay attention only to the cognitive content, given that it is immediately useful with respect to their problem-solving goals. They are likely to ignore the metacognitive content, the benefits of which may seem remote and uncertain. For example, when asking for a hint within the Angles unit of the Geometry Cognitive Tutor, one of the first hints on each step used to convey metacognitive content—it recommended that students search the system's glossary of geometry knowledge. Yet, while students asked for a hint on 29% of their answer steps, they hardly ever followed the tutor's metacognitive recommendation: students used the glossary on only 2.7% of the answer steps (Aleven and Koedinger 2000).

We chose to give the metacognitive tutor a "different voice" than the domain-level tutor, by displaying its messages in a different font type and color. This simple distinction was expected to help students extract the metacognitive content and link it to other similar messages they receive. Anecdotally, it seems to work. Students refer to the help-seeking feedback as the "blue messages," named after their font color. They compare how often they receive them and discuss their value (and occasionally, the degree to which they are annoying). Giving a different voice to cognitive and metacognitive content can be done in

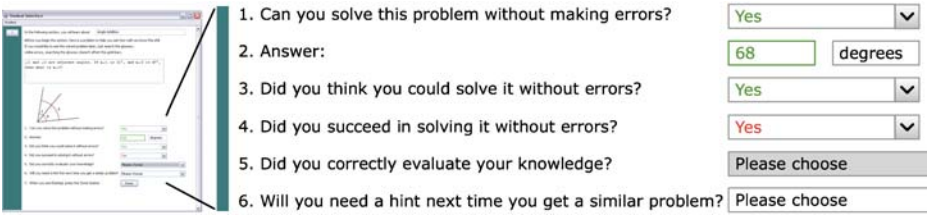

Fig. 3 During study 5, students first engage in preparatory self-assessment activities within an environment focused on metacognition (in the context of solving relatively simple problems that involve skills that are likely to be new to the students). Following these sessions, students are asked to apply, within the more demanding Cognitive Tutor environment, the metacognitive skills initially acquired in the self-assessment environment. The preparatory sessions help reduce the cognitive load during subsequent problem solving

various ways, based on the interaction style. For example, different animated agents could be used to show a metacognitive mentor side-by-side with a domain level tutor.

Principle 7: Communicate the goal structure underlying the problem solving (Anderson #2)

Cognitive Tutors often communicate a goal structure by using an interface that breaks up a problem into smaller steps. By doing so the tutor helps students to set (and achieve) appropriate subgoals. Also, more information about students' thinking is made visible, so that it becomes easier for the tutor to trace students' reasoning (and perhaps for students to reflect on their own reasoning). This principle can also be applied to metacognitive instruction. In our work, the environment used for the preparatory self-assessment sessions (Fig. 3) communicates a goal structure for desired help-seeking behavior. The interface makes explicit four subgoals related to self-assessment: predicting one's own ability, attempting to solve the problem, reflecting on the experience, and planning future interaction. While we do not communicate the goal structure of the help-seeking skills during domain-level problem solving, Conati and VanLehn (1999) offer a metacognitive scaffold within the main tutoring environment. Their SE-Coach scaffolds self-explanation of solved examples, so no domain subgoaling is necessary. The self-explanation process is scaffolded in a way that makes the process clear and apparent to the student. In another example, Gama (2004) requires students to self-evaluate their ability, choose appropriate strategies to be applied, and reflect on their experiences once the problem is solved. In that manner students go through the subgoals of a metacognitively-correct problem-solving process.

Principle 8: Communicate the metacognitive learning goals (new principle)

Metacognitive tutoring should make students aware of their metacognitive learning goals. The corresponding need usually does not occur with respect to domain-related learning goals, since the skills to be learned tend to be quite apparent in the problem-solving process. For instance, while attempting to solve a geometry problem, students consciously apply their domain knowledge. They are less likely, however, to be aware of the learning strategies they use. As Gama writes, metacognitive tutoring should "get students to recognize the importance of the metacognitive activities to the learning process" (Gama 2004, p. 670).

During our work on the Help Tutor, we learned that students do not internalize the three different aspects of the help-seeking knowledge merely by practicing it and receiving feedback. For example, in study 4, students' help-seeking declarative knowledge did not improve as a result of their work with the Help Tutor, as evaluated using an assessment of declarative knowledge of good help-seeking strategies (pretest score: 60%, posttest score: 64%, $p=0.5$). We therefore decided to incorporate help-seeking instruction in the form of a video and a classroom discussion as part of a help-seeking support environment. The instruction included examples of good vs. bad help seeking behavior, in addition to stating the principles and dispositional values of productive help-seeking behavior. Results from the recent study 5 suggest that this form of declarative instruction helps: Students who received declarative instruction and worked with the Help Tutor scored 73% on a help-seeking declarative knowledge assessment, compared to 48% of the control group who worked with the conventional Cognitive Tutor and received no help-seeking instruction or support ($p<0.02$; Roll et al. 2007). Another example of successful application of this process is given in White and Frederiksen (1998), in which a scientific inquiry cycle is taught explicitly to students, in addition to self-review and peer-review activities in the classroom.

Principle 9: Attach a price tag to metacognitive errors (new principle)

Unfortunately, students often do not see the value of productive metacognitive behavior. In our studies, the distinctive voice of the metacognitive instruction (i.e., the "blue messages" of the Help Tutor, mentioned above) often is seen by students as a signal that these messages can be skipped, since they convey no domain knowledge of immediate benefit. In order to reduce that kind of behavior, besides emphasizing the value of the metacognitive messages, it may be helpful to "attach a price tag" to committing metacognitive errors. For example, after we saw that students dismiss the Help Tutor messages rapidly, we added a 2-seconds delay during which messages cannot be dismissed. Another option is to add mastery learning to metacognitive content.

The need for a price-tag does not necessarily exist at the domain level tutoring, since progress in the curriculum (and classroom grade) depends on successful problem solving (and thus a price tag for cognitive errors is embedded in the learning process). Currently, Cognitive Tutors apply a mastery learning criterion to domain-level content only, with one exception: recently, an experimental version of the Geometry Cognitive Tutor has been altered to apply its mastery learning approach to the quality of students' (tutored) self-explanations.

Other principles

Anderson et al. specify two more design principles. While these may well be applicable at the metacognitive level, at the time of this writing we have not yet experimented with them.

One of these principles suggests to *facilitate successive approximations of the target skill* (Anderson #8). According to this principle, support that is given initially should be removed as the student progresses. Anderson describes a division of labor between the tutor and the student, in which the tutor's role is reduced with practice. This principle can be applied to the metacognitive level as well. For example, the student-system interaction could be structured so that students first learn *when* to ask for hints, with the system controlling the level of the hint. Once proficient in that regard, the student could take on the responsibility of choosing the right level of help, or the right resource to use when seeking help.

The other principle we have not experimented with is to *adjust the grain size of instruction with learning* (Anderson #7). According to this principle, students have control over the grain size at which they seek feedback from the tutor, for example, by skipping intermediate steps (once mastered the associated skills). In order to apply this principle at the metacognitive level, a hierarchical structure of metacognitive skills should be used, which we have yet to do. Our current attempt is to teach the building blocks of metacognitive knowledge, such as help usage, self-explanation, etc. Once we know how to do so effectively, we can proceed to scaffold their combination. For example, the Help Tutor could give feedback on observed *patterns* of faulty help-seeking behavior—in order to respond to a student's observed tendencies rather than their second-by-second metacognitive behavior.

Evaluation

Principle 10: Assess metacognitive knowledge and application directly (new principle)

While not the focus of the current paper, it is important to note that progress toward metacognitive learning goals should be evaluated directly (in addition to measuring the

effect that metacognitive instruction has on domain-specific learning). With respect to metacognitive knowledge, not only should students' (declarative) understanding be evaluated, but also its spontaneous application. When appropriate, the evaluation should be done in a novel learning context: the litmus test of instruction on help-seeking surely should be whether students have become better help seekers and therefore better learners of novel subject matter and skills. Studies 4 and 5 included several direct help-seeking measures: Log file analysis is used to evaluate students' procedural help-seeking knowledge; embedded hints in the paper test evaluate the way students transfer their help-seeking knowledge to a paper-and-pencil environment; and hypothetical help-seeking dilemmas are used to evaluate students' declarative knowledge of good help-seeking strategies.

Summary

In this paper we investigate the applicability of the design principles put forward by (Anderson et al. 1995) to metacognitive tutoring and more specifically, the tutoring of effective help seeking. While most principles apply to metacognitive tutoring in a similar fashion as they do to domain level tutoring, several additional challenges arise. These challenges are due mainly to the need to combine tutoring at metacognitive and domain levels, as well as from the nature of metacognitive knowledge. In our work on the Help Tutor, we have made progress in addressing these challenges. As a result, we have extended and modified some of Anderson's principles.

The list of principles we considered in this paper is not complete, and there is much room for evaluation and improvement. We studied them within the context of tutoring help seeking with an intelligent computer tutor. We expect that these principles will vary somewhat as they are applied to other metacognitive learning goals in other types of learning environments. However, we believe that framing our experience as a set of principles can make it easier for researchers and developers to apply the lessons learned in our work to other metacognitive ITS. It would be of much interest to examine these principles in other types of learning environments, such as environments focused on inquiry, games, and collaboration, and then to compare how they hold up in these different contexts. Evaluating these principles in a variety of ITS teaching different metacognitive skills will help improve them and lead to a greater understanding of the requirements for and characteristics of metacognitive tutoring systems.

Acknowledgments We would like to thank Ryan deBaker, Eunjeong Ryu, Jo Bodnar, Ido Jamar, Brett Leber, Jonathan Sewall, Mike Konieczki, Kathy Dickensheets, Grant McKinney, Terri Murphy, Sabine Lynn, Dale Walters, Kris Hobaugh and Christy McGuire for their help carrying out these studies. This research is sponsored by NSF Award IIS-0308200, the Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education (# R305B040063), and NSF Award SBE-0354420 to the Pittsburgh Sciences of Learning Center. The contents of the paper are solely the responsibility of the authors and do not necessarily represent the official views of the NSF.

References

Aleven, V., & Koedinger, K. R. (2000). Limitations of student control: Do students know when they need help? In G. Gauthier, C. Frasson, & K. VanLehn (Eds.), *Proceedings of the 5th International Conference on Intelligent Tutoring Systems* (pp. 292–303). Berlin: Springer.

Aleven, V., & Koedinger, K. R. (2002). An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science, 26*, 147–179.

Aleven, V., McLaren, B. M., Roll, I., & Koedinger, K. R. (2006). Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education, 16*, 101–128.

Aleven, V., Roll, I., McLaren, B. M., Ryu, E. J., & Koedinger, K. R. (2005). An architecture to combine meta-cognitive and cognitive tutoring: Pilot testing the Help Tutor. *In Proceedings of 12th International Conference on Artificial Intelligence in Education* (pp. 17–24). Amsterdam, The Netherlands: IOS.

Aleven, V., Stahl, E., Schworm, S., Fischer, F., & Wallace, R. M. (2003). Help seeking and help design in interactive learning environments. *Review of Educational Research, 73*(2), 277–320.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences, 4*(2), 167–207.

Azevedo, R. (2005a). Using hypermedia as a metacognitive tool for enhancing student learning? The role of self-regulated learning. *Educational Psychologist, 40*(4), 199–209.

Azevedo, R. (2005b). Computer environments as metacognitive tools for enhancing learning. *Educational Psychologist, 40*(4), 193–197.

Azevedo, R., Cromley, J. G., & Seibert, D. (2004). Does adaptive scaffolding facilitate students ability to regulate their learning with hypermedia? *Contemporary Educational Psychology, 29*, 344–370.

Baker, R. S., Corbett, A. T., Koedinger, K. R., & Wagner, A. Z. (2004). Off-task behavior in the Cognitive Tutor classroom: When students "game the system." *In ACM CHI 2004: Computer–Human Interaction* (pp. 383–90).

Baker, R. S. J. d., Corbett, A. T., Koedinger, K. R., Evenson, E., Roll, I., Wagner, A. Z., et al. (2006a). Adapting to when students game an intelligent tutoring system. *In 8th International Conference on Intelligent Tutoring Systems* (pp. 392–401). Berlin: Springer.

Baker, R. S. J. d., Corbett, A. T., Koedinger, K. R., & Roll, I. (2006b). Generalizing detection of Gaming the System across a tutoring curriculum. *In 8th International Conference on Intelligent Tutoring Systems* (pp. 402–11). Berlin: Springer.

Bielaczyc, K., Pirolli, P. L., & Brown, A. L. (1995). Training in self-explanation and self-regulation strategies: Investigating the effects of knowledge acquisition activities on problem solving. *Cognition and Instruction, 13*(2), 221–252.

Biswas, G., Leelawong, K., Kadira, B., Viswanath, K., Vye, N., Schwartz, D. L., et al. (2004). Incorporating self regulated learning techniques into learning by teaching environments. *In The Twenty Sixth Annual Meeting of the Cognitive Science Society* (pp. 120–125).

Bransford, J. D., Brown, A. L., & Cocking, R. R. (2000). *How people learn: Brain, mind, experience, and school*. Washington, DC: National Academy Press.

Brown, A. (1987). Metacognition, executive control, self-regulation, and other more mysterious mechanisms. In F. Reiner, & R. Kluwe (Eds.), *Metacognition, motivation, and understanding* (pp. 65–116). Hillsdale, NJ: Erlbaum.

Bunt, A., & Conati, C. (2003). Probabilistic student modelling to improve exploratory behavior. *User Modeling and User-Adapted Interaction, 13*(3), 269–309.

Carver, S. M. (2001). Cognition and instruction: Enriching the laboratory school experience of children, teachers, parents, and undergraduates. In S. M. Carver, & D. Klahr (Eds.), *Cognition and instruction: Twenty-five years of progress* (pp. 385–426). Mahwah, NJ: Erlbaum.

Carver, S. M., & Mayer, R. E. (1998). Learning and transfer of debugging skills: Applying task analysis to curriculum design and assessment. *In Teaching and Learning Computer Programming: Multiple Research Perspectives* (pp. 259–97). Hillsdale, NJ: Erlbaum.

Conati, C., & VanLehn, K. (1999). Teaching meta-cognitive skills: Implementation and evaluation of a tutoring system to guide self-explanation while learning from examples. *In Artificial Intelligence in Education* (pp. 297–304). Amsterdam, The Netherlands: IOS.

Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction, 4*, 253–278.

Corbett, A. T., & Anderson, J. R. (2001). Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement and attitudes. In J. Jacko, A. Sears, M. Beaudouin-Lafon, & R. Jacob (Eds.), *CHI'2001 Conference on Human Factors in Computing Systems* (pp. 245–52). New York: ACM.

de Jong, T., & van Joolingen, W. R. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research, 68*, 179–201.

Del Solato, T., & du Boulay, B. (1995). Formalization and implementation of motivational tactics in tutoring systems. *Journal of Artificial Intelligence in Education, 6*(4), 337–378.

Gama, C. (2004). Metacognition in interactive learning environments: The reflection assistant model. *In 7th Conference on Intelligent Tutoring Systems* (pp. 668–677). Berlin: Springer.

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education, 8*, 30–43.

Koedinger, K. R., & Corbett, A. T. (2006). Cognitive tutors: Technology bringing learning science to the classroom. In K. Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences* (pp. 61–78). Cambridge University Press.

Luckin, R., & Hammerton, L. (2002). *6th International Conference on Intelligent Tutoring Systems. Getting to know me: Helping learners understand their own learning needs through metacognitive scaffolding*. Berlin: Springer.

Mathan, S. A., & Koedinger, K. R. (2005). Fostering the intelligent novice: Learning from errors with metacognitive tutoring. *Educational Psychologist, 40*(4), 257–265.

Morgan, P., & Ritter, S. (2002). *An experimental study of the effects of cognitive tutor algebra I on student knowledge and attitude*. Pittsburgh, PA: Carnegie Learning, Inc.

Quintana, C., Zhang, M., & Krajcik, J. (2005). A framework for supporting metacognitive aspects of online inquiry through software-based scaffolding. *Educational Psychologist, 40*(4), 235–244.

Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N. T., & Koedinger, K. R. (2005). The Assistment project: Blending assessment and assisting. *In The 12th International Conference on Artificial Intelligence In Education* (pp. 555–62). Amsterdam: IOS.

Reif, F., & Scott, L. A. (1999). Teaching scientific thinking skills: Students and computers coaching each other. *American Journal of Physics, 67*(9), 819–831.

Resnick, L. B. (1987). *Education and learning to think.* Washington: National Academy Press.

Roll, I., Aleven, V., McLaren, B. M., & Koedinger, K. R. (2007). Can help seeking be tutored? Searching for the secret sauce of metacognitive tutoring. *In Proceedings of the 13th International Conference on Artificial Intelligence in Education* (pp. 203–210). Amsterdam, The Netherlands: IOS.

Roll, I., Aleven, V., McLaren, B. M., Ryu, E., Baker, R. S., & Koedinger, K. R. (2006). The Help Tutor: Does metacognitive feedback improve students' help-seeking actions, skills and learning? *In 8th International Conference in Intelligent Tutoring Systems* (pp. 360–369). Berlin: Springer.

Roll, I., Baker, R. S., Aleven, V., McLaren, B. M., & Koedinger, K. R. (2005). Modeling students' metacognitive errors in two intelligent tutoring systems. *In user modeling 2005* (pp. 379–388). Berlin: Springer.

Schoenfeld, A. H. (1992). Learning to think mathematically: Problem solving, metacognition, and sense-making in mathematics. In D. Grouws (Ed), *Handbook of research on mathematics teaching and learning* (pp. 334–370). New York: MacMillan.

White, B. Y., & Frederiksen, J. R. (1998). Inquiry, modeling, and metacognition: Making science accessible to all students. *Cognition and Instruction, 16*(1), 3–118.

Wood, H. A., & Wood, D. J. (1999). Help seeking, learning, and contingent tutoring. *Computers and Education, 33*(2), 153–169.