SOFTWARE AGENT-MONITORED TUTORIALS ENABLING COLLABORATIVE LEARNING IN COMPUTER-AIDED DESIGN AND ANALYSIS

J. L. Beuth  
Department of Mechanical Engineering  
Carnegie Mellon University,  
Pittsburgh, PA 15213  
USA

C. P. Rosé  
Language Technologies Institute and Human-Computer Interaction Institute  
School of Computer Science  
Carnegie Mellon University,  
Pittsburgh, PA 15213  
USA

R. Kumar  
School of Computer Science  
Carnegie Mellon University,  
Pittsburgh, PA 15213  
USA

ABSTRACT
Internet chat-based tutorials are being developed for integrating computer modeling and design skills into mechanical engineering undergraduate and middle school outreach programs. These tutorials help students navigate complicated software interfaces while teaching fundamental concepts through dynamic dialogues between tutorial agents and student user groups. In a typical assignment, students are asked to perform a design or modeling task that includes the use of software such as a commercial finite element code or specially designed educational software. Students work in teams, but team members are distributed within a room or between remote sites, linked by a text interface. As students collaborate electronically, an intelligent agent monitors their interactions and interjects questions or comments in response to the use of key phrases, or due to other triggers. This platform is being used to help automate collaborative learning experiences and to study how students can effectively interact with each other and with the software agents. In undergraduate projects, fundamental technical skills and intuition in interpreting results are emphasized. In outreach efforts, participants are led to consider how their work relates to the broad mechanical engineering profession.

INTRODUCTION
In the past decade, the landscape in which engineering is practiced has been transformed by two trends [1]. First, there have been far-reaching technological advances in computing, information, and manufacturing. Today’s engineers, even those at the smallest companies, have access to advanced computer-aided design and analysis software, web-based communications, information databases, and numerically controlled prototyping processes that are seamlessly linked in a manner unthinkable when today’s faculty first studied engineering. This transformation has fundamentally changed how products are produced and tested. Second, there are strong economic and market pressures for shorter product development cycles and internationally distributed team-based product design and manufacturing. The goal is effective use of rapid, geometrically distributed computational engineering, allowing product developers to test less, achieve shorter times from concept to marketplace and lower costs.

Because of these two trends, industry needs engineers, and particularly mechanical engineers, who have the ability to effectively combine hands-on hardware skills with computational tools, intuition, internet-based technical communication and engineering fundamentals. Training of undergraduate students to address this new reality needs to start in the freshman year and continue as a thread throughout the undergraduate curriculum. K-12 outreach programs should also reinforce these themes. This becomes even more of a priority when recognizing that changes in the engineering profession are continuing. As noted in the NAE report...
“Educating the Engineer of 2020, Adapting Engineering Education to the New Century,” changes in the engineering profession over the past decade are just the beginning of an “ongoing evolution” of the global economy. Engineering education must adapt in ways that anticipate where the engineering profession will be in 10-20 years.

Many of the same technological advances that have transformed engineering practice also have the potential to transform engineering education. Web-based instruction coupled with on-line collaborative learning, and the real-time transfer of numerical product designs, simulations, images and product prototypes offer a wide range of possibilities for student learning. In this research, we seek to exploit a recently developed educational technology allowing machine-monitored internet dialogues between members of student teams, with dialogue agents taking part in and guiding student communications.

One key to addressing these changes is the integration of computer-aided engineering (CAE) experiences into undergraduate engineering curricula in ways that simultaneously increase intuition and understanding of engineering fundamentals. This is not an easy task, since unguided student use of simulation software can lead to an “ornamentation” effect, where the sophistication and volume of results available leads students to neglect critical analysis and interpretation of their work [2]. In our own research we have observed students more oriented toward “doing the simulation” than reflecting on and interpreting what they see. Support that leads to increased learning in connection with simulations encourages students to stop and think at key points in their exploration [3, 4, 5]. In contrast, hints that allow students to work through the simulation with greater ease and speed can actually decrease learning [4]. A second key to addressing these challenges is to create learning platforms that are automated and well-suited to internet-based communication. An important barrier to the widespread use of computer-aided engineering and collaborative learning experiences is that they require increased faculty time.

Existing engineering curricula have precious little space where additional student experiences can be added during regular classroom time, regardless of their importance or relevance to industrial practice. This problem is shared by all science and technology curricula attempting to integrate information technology skills [6]. It is also widely recognized that guided use of sophisticated simulation software can enable exploratory and inquiry-based modes of learning [7]. If these capabilities are exploited effectively, then integration of CAE projects can efficiently increase student learning and understanding.

The underlying thesis of this research is that offering a dynamic self-paced learning environment for student use outside of the lecture room is the best practical means for integrating sophisticated CAE experiences into undergraduate engineering curricula. Furthermore, the machine-monitored internet chat-based tutorial environment we use to achieve this goal offers an excellent opportunity for automating and invigorating K-12 outreach efforts and for tying them naturally to more sophisticated undergraduate-level instruction.

The foundation of our approach consists of two pillars: 1) self-paced web tutorials guiding students through software use and 2) dynamic, dialogue-based tutorial interfaces which engage students in interpreting simulation results they create. The use of self-paced web tutorials as a means of efficiently integrating complex software package use into undergraduate curricula has been the subject of a long-term effort at Carnegie Mellon [8, 9]. The integration of an agent-monitored dialogue-based interface into software instruction represents a substantial enhancement to this approach.

As we deploy dialogue-based tutorials, we are not simply using them to enhance software instruction. We are also studying how students learn and teach each other in a tutorial-guided dynamic chat environment involving students in one or multiple groups. Our initial focus has been a first-year introductory mechanical engineering course; however, we are also using our experiences as a template for a new approach to CAE instruction throughout our curriculum. Furthermore, we are also applying our agent-monitored dialogue-based tutorial platform to the task of middle school student outreach.

TUTOR ARCHITECTURE AND INTERFACE

In this research, we are adapting a prototype architecture we have developed for supporting student interactions in a broad range of activities [10, 11]. This unique, automated, collaborative learning platform naturally exploits chat room-style communications that are ubiquitous on the internet, and also students’ comfort and curiosity with that environment. In our prior work using a similar environment for middle school math instruction [12], students found the collaborative problem solving environment highly engaging. Some students commented that the collaborative environment was “way more fun” than their typical computer lab activities, and that they were disappointed when the 45 minutes lab session was over.

The idea behind its design is for a filter to process the text from an ongoing discussion as it is happening, and to build an internal model of how the conversation is progressing. Using this model, it is possible to determine where the most strategic opportunities for supporting learning exist.

Figure 1 shows an overview of the architecture used to develop our prototype infrastructure. This architecture is meant to allow context-sensitive support for collaborative learning and reflection not only to be triggered based on what is happening in the discussion, but for it to do so with awareness of how it is affecting the state of the conversation through its continuous
monitoring. Thus, if an intervention is triggered erroneously and ends up having a negative effect on the collaboration, we can detect and correct that. In this way, we minimize the risk of misdiagnosing the state of the collaboration.

As displayed in Figure 1, all interface events resulting from student contributions to the chat interface and to a shared problem solving or simulation space are sent to the Filters module. Its purpose is to identify significant events in this stream that it then reflects back to the interfaces of the students. It also uses these identified events to update its internal state. Other triggers such as timers that keep track of time elapsed since the beginning of the session or since the last significant contribution of each student are also used to manipulate the Filter module’s internal state. The internal state then is used to select strategies for selecting dialogue agents to participate in the chat session in order to offer support in the form of interactive directed lines of reasoning.

In our prior experiments we have used different kinds of triggers including topic-based filters, time-outs, interface actions, and conversational actions that are indicative of the degree of engagement of the students in the discussion. Our generic architecture is meant to be easily extended to work with other types of triggers such as cues from other modalities like speech, hand sketches, etc. We continue to improve the architecture to provide richer communication and modularization. Conversational agents meant to offer support in the midst of collaborative learning interactions can be authored with the TuTalk dialogue agent authoring system [13, 14]. As displayed in Figure 1, when the Filters module sends a notification to the Conversational Agents module to trigger a particular cognitive support agent, the scheduled TuTalk agent is appended to a queue of TuTalk Agents, which are then launched in turn at appropriate breaks in problem solving behavior.

Figure 2 illustrates a typical interface used for instruction, taken from a study in sophomore-level thermodynamics. As displayed in the figure, the interface contains two panels. The rightmost panel is a chat interface, which allows students to interact with each other as well as with the conversational agents that are triggered at different occasions during the problem solving session. In our outreach efforts, our collaboration interface also includes a sketching window, also with pen strokes by any team member viewable by the others. Currently, however, our monitoring of team member interactions is limited to text interactions.

The panel on the left is a problem-solving interface that allows students to collaboratively work on a given problem. The problem solving interface in the left panel was built using CyclePad and the Cognitive Tutor Authoring Tools (CTAT) [15]. The structured problem solving CTAT panel has a problem layout and a hint button. The problem layout can either be implemented using CTAT itself as in [12], or with a simulation environment such as CyclePad [4] as currently displayed.

The integration between CTAT and the problem solving interface allows direct feedback on problem solving to be provided to students. The hint messages that are provided by CTAT are displayed in the Chat Interface. Both panels of the interface maintain a common state across the participants at all times so that student group members are independently able to manipulate all of its interface elements. Actions performed by a student in either of the panels are immediately communicated and reflected on the interface of other students in the same group.
TUTOR APPLICATIONS

Our tutor architecture has thus far been deployed in three classroom environments: a first-year undergraduate course Fundamentals of Mechanical Engineering, a middle-school level outreach workshop, and a sophomore-level thermodynamics course.

Freshman Fundamentals of Mechanical Engineering

This course is a first-year introductory course offered to engineering students potentially interested in mechanical engineering as a major. Carnegie Mellon engineering students are required to take two discipline-specific introductory engineering courses during their freshman year. The mechanical engineering course combines an introduction to fundamentals in the areas of statics, stress analysis, dynamics, fluid mechanics, and thermodynamics with activities designed to give students a broad sense of the profession. An important project within that course is an introductory computer-aided engineering (CAE) project.

In the CAE project, students use Pro/ENGINEER, Pro/MECHANICA and Pro/MANUFACTURING software from PTC to design, analyze and plan the manufacture of a simple two-headed wrench (Fig. 3). A wrench has been chosen as the focus of this project due to its simple extruded 2-D geometry and the acquaintance nearly all students have with its use. Student wrench designs are fabricated on a CNC milling machine based on the student-generated planning file. At total of 100-130 students complete this project each semester.

An important purpose of this project is to demonstrate to students the basics of computer-aided design, analysis and manufacturing tasks routinely performed by mechanical engineers. To further reinforce fundamental concepts of stress analysis and design, an agent-monitored collaborative dialogue module was added to the analysis step of the project. In that activity, students are asked to consider changes in the wrench design, and their impact on the reliability and effectiveness of the wrench. This activity builds upon moment and bending stress concepts covered in the course lecture.

Table 1 provides an example of an actual student/tutor interaction occurring during this activity. As the text entries indicate, in this case the tutor is very explicitly leading the student team through the design activity, asking questions and providing feedback on student answers. This level of support has been found to be appropriate for this and similar projects, leading to significant learning gains and increases in conceptual depth in discussions between students [3, 10, 16, 17]. Experiments have also been carried out with tutor interactions that are more “social” than those shown in Table 1 [10, 17]. For instance, the tutor can show enthusiasm when students answer questions correctly, or it can detect which team members are not interacting as much as others, and encourage them (by name) to participate more fully in the discussion. Results from some of our evaluation studies are discussed below.

Table 1: Example tutor/student interaction from the freshman course study.

<table>
<thead>
<tr>
<th>Tutor</th>
<th>Lets talk about making the wrench easy to use.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tutor</td>
<td>Are you ready?</td>
</tr>
<tr>
<td>Student 19</td>
<td>o yeah</td>
</tr>
<tr>
<td>Tutor</td>
<td>Intuitively, if you wanted to make the wrench easier to use, would you make it longer or shorter?</td>
</tr>
<tr>
<td>Student 5</td>
<td>Longer</td>
</tr>
<tr>
<td>Tutor</td>
<td>That's right. A longer wrench is better.</td>
</tr>
<tr>
<td>Tutor</td>
<td>Why is a longer wrench easier to use though? Let's look at the concept of Moment.</td>
</tr>
<tr>
<td>Tutor</td>
<td>When you use a wrench to turn a bolt, do you want a higher or lower moment?</td>
</tr>
<tr>
<td>Student 16</td>
<td>Higher</td>
</tr>
<tr>
<td>Student 13</td>
<td>Higher</td>
</tr>
<tr>
<td>Tutor</td>
<td>That's right. A higher moment means the Wrench turns easier.</td>
</tr>
<tr>
<td>Tutor</td>
<td>We know that moment = force x length. So...</td>
</tr>
</tbody>
</table>
Middle School Outreach

A simplified version of the freshman project has also been effectively used as a 3-hour middle school outreach workshop offered on weekends. This workshop is offered as part of the Carnegie Mellon Institute for Talented Elementary and Secondary Students (C-MITES) program, targeting intellectually curious K-9th grade students with engaging activities in the arts, humanities, science and engineering.

In this workshop, students are guided step-by-step through the CAD modeling of a wrench, after an introduction to the broad subject of computer-aided engineering and the tasks of design, analysis and manufacturing (Fig. 4). Students are then given a tour of the mechanical engineering machine shop, with demonstrations of rapid prototyping and CNC machining of a wrench design (Fig. 5). In the final activity of the workshop, students are assigned to distributed groups within the classroom and asked to use an agent-monitored texting and sketching interface to collaboratively develop ideas for improving a wrench design. Students are encouraged to think broadly about what will make the wrench better.

During the collaborative design activity, student text interactions are monitored and hints are given to encourage students to think about all aspects of making a reliable, easy-to-use, and cost-effective wrench. The workshop concludes with a discussion of student group suggestions and a formal grouping of ideas into categories of 1) mechanical, 2) ergonomic and 3) cost improvements. Tutor interactions are used to lead students toward these three concepts. The ultimate conclusion of the workshop is that the task of an engineer is not to design the best product possible. It is to design a product that meets or exceeds customer expectations of reliability, is easy to use, and has the possible lowest cost. This task is difficult, but that is why engineers are paid well for their work and skills.

Figure 4: Middle school students taking part in the CAE portion of the outreach project.

Sophomore-Level Thermodynamics

The third application of this technology and approach to teaching pursued to date is in a sophomore-level thermodynamics course. In a project for that course, students study the efficiency of thermodynamic cycles using CyclePad. CyclePad is a package developed at Northwestern University that allows students to construct and analyze a wide variety of thermodynamic cycles. As noted in the discussion of Fig. 2, because of its open architecture, CyclePad enables monitoring of not only student text interactions but also student inputs into the software and software outputs. This enables more detailed monitoring and guidance of student group work.

An example of interactions between two thermodynamics students and the agent is provided in Table 2. In recent work, we have successfully used the same technology to support interactions between larger student groups, such as 3-5 person teams. As in the example of Table 1, in Table 2, Tutor is the chat agent. As we see from Table 2, there are two main interaction components working behind the tutor agent: the Hinting Actor and the Tutoring Actor. These components are supported in their task by two filtering components: the Hinting Filter and the Tutoring Filter. The role of the Hinting Filter is to look for a student contribution related to an underlying concept. When such a contribution (Line 1) is detected, the Hinting Actor generates a hint pointing the students to a relevant page in the instruction book (Line 2). Note that what is considered relevant is based on the agent’s analysis of the ongoing discussion, detecting which topics have been raised by the students so that support related to those topics can be triggered. In the example in Table 2 we see that Student 1 has raised the issue of increasing pressure to the connector, which is what is known as the maximum pressure of the cycle. This is an important parameter in the design of the power plant, and there are principles related to decisions about its configuration that students should keep in mind. Also, the Tutoring Actor invites the students to ask for help on the concept (Line 3). This is how the agent offers the student the
Table 2: Example CycleTalk/student interaction from the sophomore thermodynamics study.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Student 1</td>
<td>so i think the best thing to go with first is to increase the pressure entering the connector</td>
</tr>
<tr>
<td>2</td>
<td>Tutor</td>
<td>You might want to review what you saw earlier in the sensitivity analysis investigating the effect of changing P-max on steam quality and efficiency, from the top half of page 13 in your book.</td>
</tr>
<tr>
<td>3</td>
<td>Tutor</td>
<td>type: HELP WITH CHANGEPMAX if you want to discuss it with me.</td>
</tr>
<tr>
<td>4</td>
<td>Student 1</td>
<td>help with changepmax</td>
</tr>
<tr>
<td>5</td>
<td>Tutor</td>
<td>Remember the sensitivity analysis you have completed where you manipulated P-Max and observed it's effect on the efficiency of the cycle. You can refer to the figure that I have pasted onto your whiteboard for your convenience. Having seen how changing P-Max, the Maximum Pressure at the Heater Inlet, effects efficiency and Steam, do you think as a general principle the Maximum pressure should be increased ?</td>
</tr>
<tr>
<td>6</td>
<td>Student 1</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Tutor</td>
<td>So the Efficiency of a Cycle increases by increasing the maximum pressure of the cycle. In addition to the effect on efficiency, what happens to the steam quality at the turbine exit if you increase the maximum pressure of the cycle ?</td>
</tr>
<tr>
<td>8</td>
<td>Student 1</td>
<td>Decreases</td>
</tr>
<tr>
<td>9</td>
<td>Tutor</td>
<td>Steam quality should be maintained above 0.85 to avoid any damage to turbines blades during operation. So we need to balance these concerns by increasing the maximum pressure, but not enough that the quality of the steam drops below the minimum value (0.85). Are there other limitations on increasing the maximum pressure at the heater inlet ?</td>
</tr>
<tr>
<td>10</td>
<td>Student 1</td>
<td>no?</td>
</tr>
<tr>
<td>11</td>
<td>Tutor</td>
<td>Can the devices and pipes in the cycle stand if we keep on increasing the pressure ?</td>
</tr>
<tr>
<td>12</td>
<td>Student 1</td>
<td>20000kPa</td>
</tr>
<tr>
<td>13</td>
<td>Tutor</td>
<td>Beyond a particular pressure, the pipes and devices in a cycle become unsafe at risk for accident. The maximum pressure at heater inlet should be within safety constraints of the material used.</td>
</tr>
<tr>
<td>14</td>
<td>Student 2</td>
<td>so to what value do you want to change the pressure at s1?</td>
</tr>
<tr>
<td>15</td>
<td>Student 2</td>
<td>do we want to keep X of s3 above or below .85?</td>
</tr>
<tr>
<td>16</td>
<td>Student 1</td>
<td>Above</td>
</tr>
</tbody>
</table>

opportunity to exercise some control over the timing of the help that is offered by the conversational agents. The Tutoring Filter detects the request for help (Line 4) and initiates an instructional dialog with the students (Lines 5 through 16). Note that the dialogue that is initiated is pertinent to the conversation about manipulating the maximum pressure of the cycle.

RESULTS TO DATE
We have run a variety of pilot studies with the systems described above in undergraduate engineering courses to verify the effectiveness of the instructional support we offer. For example, in one study [3], where students worked in pairs on a power plant design task, students who worked in pairs that included the support of a conversational agent learned 1.24 standard deviations more than control condition students who worked alone in the same learning environment but without a partner or the conversational agent. Students who worked either with a partner student or with a conversational agent learned one standard deviation more than the control condition students. Note that one standard deviation is equivalent to a full letter grade. In subsequent studies [10, 16, 17] we refined our conversational agent technology, achieving even better results. For example, three different studies demonstrated statistically significant improvements in learning with the best condition from the initial study as the baseline. Improvements in agent performance included the introduction of social strategies for creating a more friendly environment for students to work together in and offering students the opportunity to delay the start of a conversation with one of the agents if they prefer. Based on questionnaire data, students also preferred the agents with enhanced social capabilities [10, 17].

CONCLUSIONS
Agent-monitored collaborative tutorials are being integrated into software-based mechanical engineering projects at Carnegie Mellon. Thus far, projects at the freshman and sophomore levels have been deployed within the undergraduate curriculum, with a spin-off of the freshman project also used as part of a middle school outreach workshop. This platform is being used to explore the effectiveness of a variety of collaborative environments, as well as combinations of collaborative learning, automated instruction, classroom-style instruction and interpersonal instruction. Tutor agents are also being used to quantify the importance of social interaction (both simulated via tutors and actual via teaching assistants) in learning. The goal is to not only more effectively teach engineering fundamentals in the context of software use, but to
also increase the efficiency of integrating software projects into courses.

**ACKNOWLEDGMENTS**

This research was supported by the National Science Foundation Division of Undergraduate Education (DUE) Course Curriculum and Laboratory Improvement (CCLI) Program and the Engineering Education and Centers (EEC) Innovations in Engineering Education, Curriculum and Infrastructure (IECII) Program under grants DUE-0837661 and EEC-0935145. This research is also supported through a software grant from Parametric Technology Corporation (PTC). The authors would also like to thank their collaborators, Ann Shoplik and Pamela Piskurich of the Carnegie Mellon C-MITES program, Jack Zhou of Drexel University, and John Stuart of PTC.

**REFERENCES**


