Classroom Assessment Tools as Discussion Frameworks in CS 1

Tammy VanDeGrift and Richard Anderson
Computer Science & Engineering Department
University of Washington
Seattle, WA 98195-2350
{tammy, anderson}@cs.washington.edu

Abstract

In this paper we describe Classroom Assessment Tools (CATs) designed to supplement lecture material in an introductory programming course (CS 1). CATs allow instructors to collect feedback from the students regarding their understanding of course material [2]. With frequent use of CATs, instructors can monitor student progress throughout a course. CATs are ungraded, learner-centered activities, usually submitted anonymously by students encouraging honest feedback from students about what they do and do not understand in the course [2].

We developed over 60 Classroom Assessment Tools for use by CS 1 instructors. The CATs highlight concepts from the recorded lectures that are used in CS 1 courses employing the Tutored Video Instruction (TVI) model. The TVI teaching model utilizes a set of recorded lectures to stimulate discussion among a small set of students and a local instructor (tutor). The CATs described in this paper can be utilized in all teaching styles, allowing instructors and students to monitor the learning process.

1 Introduction

Introductory computer science is taught in several styles through various media, ranging from large lecture classrooms to online courses. In many teaching models students learn introductory programming through a delivery phase and an interactive phase. MIT teaches “Structure and Interpretation of Computer Programs” using audio recordings and lecture slides that students view during or prior to class meetings.1 Georgia Institute of Technology offers “Lecture Anytime” sections for CS 1 where students learn through online lectures and contact time with teaching assistants and instructors [5]. We have experience teaching CS 1 and CS 2 at the University of Washington (UW) in a large lecture hall setting combined with weekly quiz sections led by teaching assistants. In any teaching model knowing what and how students learn can greatly improve the learning experience. Instructors and students can then be more efficient and effective in teaching and learning the material. How might instructors continually assess students’ learning in CS 1?

We designed over 60 Classroom Assessment Tools (CATs) to aid in the learning and teaching of computer programming. CATs are flexible assessment tools for instructors to use during the interactive phase of teaching that accompanies a variety of classroom formats. We developed these tools to encourage students to actively engage in an introductory programming course, discussing concepts with fellow students and their instructors. These tools provide a mechanism for instructors to monitor students’ learning throughout the course. Typically completed anonymously, CATs give students an opportunity to provide honest feedback to the instructor [2]. We will evaluate the impact of using CATs in introductory programming courses during the 2001 - 2002 academic year.

This paper provides a description of Classroom Assessment Tools, specific examples of CATs developed for CS 1, a brief overview of the Tutored Video Instruction (TVI) project, conclusions made from observations of the TVI courses, and future evaluation methods to assess the Classroom Assessment Tools.

2 Classroom Assessment Tools

Classroom Assessment Tools (CATs) allow instructors to collect feedback from their students about what and how they are learning the material [2]. With frequent

1http://sicp.ai.mit.edu/Fall-2001/
use of CATs, instructors can monitor student progress throughout the quarter or semester. By using a variety of CATs, instructors can learn how students respond to various teaching approaches. Instructors have the opportunity to revise teaching strategies based on feedback from the students.

CATs are learner-centered activities that are ungraded and usually submitted anonymously by students. This type of activity encourages students to be honest in what they do and do not understand in a course. Instructors, after evaluating the responses submitted by students, can address student concerns and questions in a timely manner. Therefore, the confusion experienced by students can be resolved before an exam or project.

Angelo and Cross outline several Classroom Assessment Techniques, some of which appear below [2].

**Minute Paper:** Ask students to respond briefly on paper to the questions “What was the most important thing you learned during this class?” and “What important questions remain unanswered?”.

**Muddiest Point:** Ask students to respond in writing to the question “What was the muddiest point in ______?”.

**One-Sentence Summary:** Ask the students to respond in writing to the question “Who does what to whom and why?” to summarize the most important topic of the day’s lecture.

**Directed Paraphrasing:** Students are directed to paraphrase part of a lesson for a specific audience and purpose.

**Applications Cards:** After studying an important principle or theory, have students write down at least one real-world application of that principle or theory.

### 2.1 Benefits of CATs

CATs benefit both instructors and students. These tools provide a continual feedback loop for the instructor and could potentially prepare students better for exams and programming projects. Some students learn material in a lecture/video environment while others need concrete activities; providing lecture material combined with CATs targets both learning styles.

In a study conducted by Steadman, she found that students benefit from the use of CATs in the classroom [7]. Students feel that they have more control and voice in the classroom. The CATs also encourage students to be involved in their own learning and monitor their progress in a course.

These findings are promising for the success of CATs in an introductory programming course. Giving students more control and ownership in the course might encourage participation in initiating discussions with other students and the instructor during and after lecture presentation.

### 2.2 Methodology

Angelo and Cross provide a teaching goals inventory survey to help direct professors in developing CATs to achieve appropriate goals [2]. We administered this survey to three college instructors to discover their goals when teaching introductory programming courses.

We used the results of these surveys to create appropriate CATs for CS 1 lectures. Most of the CATs are content specific, testing knowledge related to the main concepts in each lecture. There are a variety of tools ranging from role-playing activities to code-writing exercises. Many CATs encourage small group coordination while some are individual exercises or whole class activities. Over 60 CATs developed for the CS 1 Torted Video Instruction (TVI) lectures can be found on the course web site. The CATs supplement an introductory programming course taught in the C programming language; however, many CATs are language independent teaching tools for general introductory computer science courses.

Each CAT comes with a list of goals associated with the tool, the content tested, the activity as the instructor would give it to the students, possible solutions to the activity, and a list of suggestions for extending and adapting the CAT.

### 2.3 Examples

We provide some exemplary CATs for CS 1 in this section. The first example illustrates the organization of the material we provide to college instructors. The remaining examples describe only the activity.

---

**Example 1:** (From Spring 2001 offering\(^2\))

**Lecture 2:** Problems, Algorithms, and Programs

**Content Tested:** Syntax and Semantic Errors

**Lecture Content:**

- Problems, Algorithms, Programs
- Problem Solving
- Compiling and Running a C Program
- Errors and Debugging

---

\(^2\) [www.cs.washington.edu/education/online/cse142/](http://www.cs.washington.edu/education/online/cse142/)

\(^3\) [www.cs.washington.edu/education/courses/cse142/01sp/](http://www.cs.washington.edu/education/courses/cse142/01sp/)
Goals:

- Ability to synthesize and integrate information
- Learn concepts and theories
- Develop capacity to think for oneself
- Improve memory skills
- Develop ability to think creatively

Assessment Technique: Approximate Analogies

Purpose: To allow instructors to find out if students understand the differences between a syntax and a semantic error and to discover relationships between these concepts in another domain.

Activity: Write the following on the blackboard, overhead, or a piece of paper:

Imagine a game or sport you like to play. (Or, if you’re a musician, use musical scores for this analogy). Now, in that new domain (games, sports, or music) describe what the difference between a syntax error and a semantic error would be. Give concrete examples of a syntax error and a semantic error.

Sample Responses:

Let’s look at a sport. Syntax errors in a sport would be violations of the rules of the game. For example, if the sport is soccer then the players (other than the goalie) catching the ball with their hands would be a syntax error. A semantic error would be poor execution of a play or a poor strategy to win the game. For example, the soccer team might have a play to kick the ball to a certain person to score and a semantic error would be the team failing to get this person the ball at the appropriate location.

Let’s look at music. If a musician is composing a piece, a syntax error would be an error in writing down the appropriate notes or dynamics. A semantic error would correspond to someone playing the wrong notes according to the written music.

Possible Uses of Activity:

- Have students write responses individually and submit these to the instructor anonymously. The instructor reads the responses and the class discusses why the analogies fit or don’t fit.
- Break students into small groups (2-4 people) and have each group create two or more responses. Have each group explain why their responses fit or don’t fit the analogy.

Example 2:

Lecture 8: Function Parameters

Assessment Technique: Mini-Drama

Activity:

We have just seen the purpose of using and the syntax for creating functions in a program. Today we will learn more about the execution of a function call which you saw in today’s lecture. In a group of 3-4 people, design a mini-drama that demonstrates the function call execution for the following code:

```c
int main (void)
{
    double x, y, z;
    y = 6.0;
    x = area(y/3.0);
    z = 3.4 * area(7.88);
    return 0;
}

/* Find area of circle with radius r */
double area (double r)
{
    return (3.14 * r * r);
}
```

You may want to split up the roles as the “main” person, the “area” person, and the “controller” who commands the execution. Props such as cards or pieces of paper with variable names and values might also be helpful.

After spending about 5 minutes creating your drama, we will share these with the rest of the class.

Example 3:

Lecture 10: Loops

Assessment Technique: Application Cards

Activity:

Say the following to the students:

We have seen how to write “for” and “while” loops in lecture. State one or more applications where you see loops in gadgets, appliances, software, or a task. Also, state the application of the tool and what type of loop you think this tool uses. Why did you decide on a “for” loop or a “while” loop?

Students may need to see some examples before thinking of an application on their own. Here are examples of possible applications:

Cooking/kitchen timer: counts down to zero from a time
given by user and beeps/buzzes when time has expired. 

**type of loop**: “for”, given a time, there are a set number of times the loop which counts down the time will execute.

**Search engine results list**: prints the “hits” found that match key words.

**type of loop**: “for” if the engine looks through entire database or web and then prints out a list of “hits”. This printing is the action executed each time through the loop; “while” if the engine prints results as they are found since the engine doesn’t know in advance how many listings should be printed.

**A parent packing lunches for his/her family**: Each lunch gets a sandwich, fruit, and beverage.

**type of loop**: “for” since the parent packs a lunch for each member of the family. The number of members in the family is known. The body of the loop contains the actions of packing a sandwich, fruit, and beverage in each lunch.

---

### 3 Tutored Video Instruction

Classroom Assessment Tools can be used in a wide variety of pedagogical approaches. The University of Washington provides Tutored Video Instruction (TVI) materials for introductory programming courses at local community colleges. Seven sites have participated in this teaching approach since Autumn 1998 [1]. We will assess the Classroom Assessment Tools in two community colleges using the TVI materials during the 2001 - 2002 academic year.

#### 3.1 Tutored Videotape Instruction

Jim Gibbons and his colleagues developed Tutored Videotape Instruction [4] in the late 1970’s [3]. Their original proposal for TVI utilizes unrehearsed videotapes of classroom courses for the viewing by a small number of students (3 to 10 students). A tutor assists the students when they interrupt the tape to ask a question. They employed the TVI model to allow participation in a course by a group of graduate engineering students at a remote site.

In the study conducted by Gibbons et al., students using the TVI model out-performed the on-campus students by measure of GPA. Their work also shows that having a tutor at the remote site is a key component to achieving success with this model.

---

#### 3.2 TVI Model in CS 1 and CS 2

Our TVI project takes advantage of the instructional model first proposed by Gibbons. We are interested in learning how well this teaching model and Classroom Assessment Tools suit introductory programming courses at local community colleges.

**Course Content and Structure** The introductory courses at the University of Washington are taught in a two-quarter sequence (CSE142/143). The content for the TVI courses matches the on-campus versions from the 2000 - 2001 academic year. The Classroom Assessment Tools support the lecture videos for CS 1. The 27 lectures cover basic constructs and data structures of the C language as well as some language independent topics such as searching and sorting.

The CS 1 TVI lectures, recorded in a studio without a live classroom in attendance, lack the administrivia details related specifically to an on-campus version of the course. Removing the live class component cuts the total length of the lectures from 23 hours (lecture time for one quarter) to 13.2 hours, leaving much class-time for students to discuss concepts with the local facilitators. On the other hand, removing the live class component also eliminates the questions that on-campus students ask during lecture and the interactions between the professor and the students.

**Previous Experiences** Our goal was to provide an alternative teaching model and course materials to community college instructors teaching CS 1 and CS 2. Several motivations led to the development of the TVI materials. By offering UW course materials to community colleges, course credits are easily transferable. Additionally, community colleges face the challenge of hiring and retaining qualified faculty in computer science; community college instructors with less experience in teaching and programming can facilitate courses using the TVI model. The results of the assessment of the TVI materials and Classroom Assessment Tools can contribute to curricula at other schools as demands for distance education options increase.

Kors, Postner and Shelton conducted observations of local community college instructors teaching CS 1 and CS 2 with the TVI materials [4, 6]. During the time of these observations, the studio-recorded lectures were not yet finished; instead, the instructors used video recorded during on-campus lectures. Student and instructor interaction was less frequent than the experiences described by Gibbons [3]. Often the instructor initiated the discussions by pausing the video. Each discussion lasted, on average, under two minutes [4]. We are interested in studying computer science as a suitable domain for discussion in a TVI course and how we...
might enhance and encourage discussion through the use of Classroom Assessment Tools.

Some community college students taking CS 1 in the TVI model had negative experiences in the course since a live instructor did not deliver the lectures [1]. The students had difficulty identifying with the local instructor and felt that the TVI course was a “University of Washington” course not designed for them. The CATs might give students a feeling of ownership and local control, helping to improve students’ perceptions of and experiences in a TVI course.

3.3 CATs in the Context of TVI

We want to learn if CATs enhance and encourage discussion in a TVI course. The CATs provide concrete exercises for the students to practice skills relating to the concepts discussed in lecture. These CATs can provide a springboard for student discussion and provide students concrete examples and evidence of their own (mis)understanding. Additionally, CATs might encourage local interaction, improving students’ perceptions of a TVI course.

4 Evaluation Plan

We are interested in studying how teachers and learners can use Classroom Assessment Tools effectively. Evaluations will take place at the University of Washington as well as local community colleges. During the 2001-2002 academic year, two instructors will use the Classroom Assessment Tools to supplement the TVI videos. Other instructors will use the assessment tools in traditional, lecture-style classrooms. We will evaluate the effectiveness of these CATs through discussions with the instructors, surveys designed for the instructors, surveys designed for the students, conversations with the students, and classroom observations. We will have preliminary results of these studies after fall semester/quarter has commenced.

5 Conclusions

Several teaching paradigms are employed to teach introductory programming concepts to college students. One such teaching strategy that might benefit from Classroom Assessment Tools is Tutored Video Instruction. Tutored Video Instruction offers a teaching model involving archived lecture videos and a tutor at the local site facilitating discussion about the concepts described in the videos. The Computer Science & Engineering Department at the University of Washington has experience in distributing TVI materials to community colleges in the state. The experiences and observations of the TVI courses led to the development of Classroom Assessment Tools to supplement the lecture material.

The CATs have the potential to create an active learning environment for students introduced to computer programming in a TVI course. CATs can also enhance the learning and teaching of computer programming in more traditional classrooms. It is our hope that the CATs in the context of an introductory programming course will create positive and successful experiences for instructors and students alike.

6 Acknowledgments

This work was funded by the Program for Educational Transformation through Technology, a university initiative fund at the University of Washington. We would like to express our thanks to the college instructors and students who are using and providing feedback on the Classroom Assessment Tools. We thank Lori Postner, a graduate student in the College of Education, for proofreading the CATs.

References


