INTE 2014

Game Programming for Complex System Development and ABET Accreditation Assessment

Cheer-Sun Yang*

Computer Science Department, West Chester University, West Chester, PA 19383, U.S.A.

Abstract

“Game Programming” has been used in the past for practicing “Worked Example” principles. In this novel attempt, it is used for teaching “Complex System Development” with the goals of covering software engineering principles and satisfying partially the ABET accreditation requirement, i.e., for students to acquire the ability to apply design and development principles in the construction of software systems of varying complexity. In this paper, the course design and the assessment are presented. The assessment results and the experiences learned are discussed.

© 2014 The Authors. Published by Elsevier Ltd.
Peer-review under responsibility of the Sakarya University.

Keywords: Game Programming; ABET accreditation assessment; complex system development.

1. Introduction

ABET is the organization for accrediting Computing Systems in the United States. This organization published the Program Criteria for Computing Systems on its website (ABET, 2013). According to the Program Criteria, a program must enable students to apply design and development principles in the construction of software systems of varying complexity.
varying complexity. Hence, the assessment must demonstrate the effectiveness in the training of software development principles for systems of varying complexity.

Teaching complex system development may be challenging for several reasons. First, students tend to avoid courses that are time consuming and programming intensified. Second, the instructor cannot simply focus on software engineering principles or game theories; actual programming is definitely required for the preparation of teaching this course. Last, the assessment could add to the complexity of teaching this course. As a result, the enrolment may suffer. One of the solutions is to use game programming for introducing software engineering principles. Game Programming has been used in the past for practicing “Worked Example” principles (Sweller, 2006, Sweller and Cooper 1985, Yang, 2003, Yang and Yu, 2011). In this novel attempt of using Game Programming for covering Software Engineering principles and ABET accreditation, many experiences have been learned.

In the remaining part of this paper, the course material is explained next. Section 2 introduces the course design. Section 3 explains an ABET accreditation requirement about making a public announcement. Section 4 presents the results of the assessment. Section 5 explains the experiences learned. Section 6 discusses the conclusion and the future work.

### Nomenclature

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABET</td>
<td>A non-profit and non-governmental accrediting agency in the USA for academic programs in the disciplines of applied science, computing, engineering, and engineering technology.</td>
</tr>
<tr>
<td>IDLE</td>
<td>An integrated computer software development tool, known as an Integrated Development Environment (IDE), for python programming.</td>
</tr>
<tr>
<td>PYTHON</td>
<td>A programming language used for developing game programs and other software development.</td>
</tr>
</tbody>
</table>

### 2. Course Design for Game Programming

#### 2.1. The IDLE Integrated Development Environment (IDE)

Throughout the course of teaching python and game programming, the IDLE IDE will be used. The IDLE IDE comes with the installation of python that can be downloaded from the official python web site at http://www.python.org/download/. While downloading python, an instructor can briefly introduce the online tutorial and library documentation available at the web site. Since there are two non-compatible versions of python, e.g., version 2 and version 3, one needs to be sensitive about the differences on syntax rules.

#### 2.2 Python Programming Language

Teaching python and game programming takes three stages: python programming, Tk interface library module for GUI, and game programming. Within each stage, incremental lab projects are presented as partial solutions; only part of the code will be available for students, and part of the code will be “carved out” and left for students to figure out initially.

In this course, focuses are on language features required later for developing 2D games. When all students did not have the experience of using python programming language, most of them had experiences of using Java programming language. Therefore, the lecture can be customized to cover python programming language features that are different from Java.

As Python is a script language supporting dynamic data typing, for students who are familiar with Java, learning python is a fun ride. If an instructor can focus on the ease of learning and use, students will be inspired.

In the following, the concept of “incremental examples” is illustrated with the list of topics to be introduced in the sequence of being introduced.
2.3 Simple Non-GUI-Based Games

The examples are used to illustrate how python can be used to develop games (Yang and Yu, 2011) including Tic-Tac-Toe and Reversi. The logic and the syntactic structure are explained. Some book examples (Sweigar 2013) are provided without using any GUI module. Students will be encouraged to convert these non-GUI-based games to GUI-based games later.

2.3.1 Tkinter Module

The Tkinter Module is the python interface to the Tk GUI library module. It supports functionalities for game programming. The incremental approach is used for teaching GUI programming with Tkinter/Pygame. Here is a list of introduced topics about Tkinter: (1) GUI Programming, (2) Recursion, and (3) Photo Image File Handling. The ultimate goal is to cover skills required for developing game projects. Several incremental projects used in teaching game programming, respectively, are described later in this paper.

2.3.2 GUI Programming

GUI Programming is one of the major features Python Game Programming covers. The software engineering principle of stepwise refinement is illustrated using GUI Programming as the goal. In the following, examples are given in an incremental fashion:

Basic Shapes & Drawing a box inside a window – As illustrated in Fig. 1(a), the drawing of a basic shape (oval, rectangle, polygon, etc.) is used to introduce the fundamental framework for GUI programming.

Timer Control: After the students become familiar with the basic GUI interfaces, the next step is to add animation to the example, e.g., moving the box without human intervention.

Event-Driven Paradigm –A variety of event handling examples is used for this stage including: adding event handling to change the speed of the moving box, the moving direction of the object, or the color of the object as shown in Figure 1(c).

2.4 Python Game Programming Examples

A stepwise refinement approach is used again for teaching python game programming. Students will follow the direction guide to complete a game with skeleton programs given to them at different stage. During the course of teaching python game programming, a simplified bubble shooting game is introduced. It is a simplified game similar to the one posted at the web page http://www.bubbleshooter.net/game1.php where the shooting is achieved with a bubble as the “bullet”. In the simplified version, a bullet is used.

Phase 1 - One row of target bubbles with the same color is used for the game as illustrated in Fig. 2(a).

Phase 2 – Students will be asked to change the colors of the target bubbles and the target will only be removed when the bullet has the same color as the target.

Phase 3 – The target bubbles will begin to drop after a while and a new row of bubbles will be added at the top of the canvas.
Phase 4 – Students can take this worked example and extend the program to support more functionality such as bouncing the bullet off the wall, removing all connected bubbles with the same color, or replace the bullet with a bubble.

![Figure 2: Bubble Shooter](image)

Phase 5 – An initial phase of a bubble spinner game is used to illustrate the strategies including rotating objects, bouncing the bullet off a wall. In this game, a cluster of bubbles is initially displayed to form a shape of a hexagon. More bubbles will be generated as the game proceeds. It is a simplified version of the bubble spinner game posted on the web page at http://www.deadwhale.com/play.php?game=774.

A worked example of a bubble spinner (Fig. 3) is used to demonstrate some of the more advanced ideas such as generating hexagon bubbles, spinning objects, and bouncing a bubble.

![Figure 3: Bubble Spinner](image)

### 3. An ABET Accreditation Requirement about Making a Public Announcement

According to ABET, a course must be announced in a public fashion that the course is used to satisfy the ABET accreditation criteria. As per the requirement, a course announcement is posted on the university course management system D2L (Desire-to-Learn). ABET does not delineate how the course information is publicized as long as the announcement that a course is used to satisfied the ABET accreditation is conveyed to the constituencies. Examples may include departmental web page, course management systems, or other means.

### 4. The Results

The assessment is incorporated as part of the Exam II to be held at the end of the class. Hence, there are three parts of the Exam II. Part I includes the Tkinter module and computer graphics; part II explores the Criterion I; and part III checks the Criteria K.

The first part of the assessment requires to show, “Upon graduation, Computer Science majors will attain an ability to use current techniques, skills, and tools necessary for computing practice” as stated in Criterion I. Five problems in Exam 3 are devised to provide a practical measurement for this criterion. Although this only indicates an instance of teaching this course with the goal of professionalism, the result indicates that over seventy percent of the students in a class of twenty students answered four or five problems correctly.
Table 1: Assessment about “Preparing Students for the Real World or Criterion I”

<table>
<thead>
<tr>
<th>#Wrong Answers</th>
<th>#Students</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6/20</td>
<td>30%</td>
</tr>
<tr>
<td>1</td>
<td>8/20</td>
<td>40%</td>
</tr>
<tr>
<td>2</td>
<td>5/20</td>
<td>25%</td>
</tr>
<tr>
<td>3</td>
<td>1/20</td>
<td>5%</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0%</td>
</tr>
</tbody>
</table>

The second part of the assessment requires to show that students can attain “an ability to apply design and development principles in the construction of software systems of varying complexity” as in Criterion K. The Criterion K requires that all computer Science majors will demonstrate proficiency in the latest, cutting-edge technology. Ten questions in Exam III are used to provide the assessment of Criterion K. Out of the twenty students participating in this assessment, over 70% of the students answer nine or ten problems correctly. All can answer more than eight problems correctly.

Table 2: Assessment about “Software Engineering Principles or Criteria K”

<table>
<thead>
<tr>
<th>#Wrong Answers</th>
<th>#Students</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6/20</td>
<td>30%</td>
</tr>
<tr>
<td>1</td>
<td>8/20</td>
<td>40%</td>
</tr>
<tr>
<td>2</td>
<td>6/20</td>
<td>30%</td>
</tr>
<tr>
<td>3-10</td>
<td>0</td>
<td>0%</td>
</tr>
</tbody>
</table>

5. Experiences Learned

This course was taught in a 5-week summer session and later in a regular semester. In the summer version, the first week is used mainly for covering python programming with some game examples. The second week and the third are used for covering graphic user interface Tkinter with more game examples. The last two weeks are used for covering advanced topics such as rotating objects using pixels for the x- and y-coordinates, respectively, drawing of hexagon bubbles, bouncing objects. In the following, some experiences learned are described briefly:

- Course preparation – It takes excessive amount of time to prepare incremental programming examples for teaching game programming. Although there are plenty python-based games available, most of them use pygame library module instead of Tkinter library module. Also, these game programs need to be “tailored” for the purpose of using them incrementally.
- Teaching material – For teaching some other courses, it may happen that only a few on-line tutorials are available. In the case of teaching python and game programming, it is just the opposite. There are many versions of on-line text material, sometimes too many; an instructor needs to customize the teaching material judgmentally with one caveat. Some require python version 2, while some others require python version 3. To be safe, an instructor needs to prepare to use both versions for demonstration and students need to be aware of the syntactic differences.
- Student evaluation criteria – Evaluation is somewhat difficult comparing with teaching without using student projects. However, assigning student projects can culminate students’ interests and conclude the course “with a high note”. A list of criteria is used including correctness, required features (a controllable component, a non-controllable component, animation, game-over, scoring, and restarting), complexity, user-interface design, and programming style, e.g., without unnecessary “hard-coded” program segments).
Follow-up course – The teaching experience for this course demonstrates the feasibility of teaching with game building. It is strongly recommend that a Computer Science curriculum should be carefully designed to be student-centered. Moreover, an advanced course based on python and game building can be used to sustain the culmination of interests created by this course. For example, Computer Graphics, Smart-phone Game Programming, and Artificial Intelligence Algorithm for Game Programming are prospective examples.

6. Conclusions and Future Work

Using the incremental worked example approach, an instructor can teach python and game programming effectively. Students demonstrate high interests while learning python as well as Tkinter and game programming. Assigning student projects culminates students’ interests and confidence about learning programming with game building. Once overcome the initial design phase, students expose tremendous innate aesthetic capabilities in a short period of time.

Despite the seemingly successful effectiveness on promoting active learning, it requires to evaluate the effectiveness systematically about teaching with incremental worked examples. Indeed, this is only another example of applying incremental worked examples for inspiring students. For example, in the future, statistical methods will be used to study the effectiveness of the approach with incremental program samples for a quantitative study of the pedagogical approach.

The assessment for the purpose of ABET accreditation has been included as one of the major activities for teaching this course. Starting from the announcement that the CS program is applying for the accreditation, to the assessment of the Criteria I and L, students all actively participate in the process cycle due to the nature of game development. Although more statistical evaluations may further prove the effectiveness of using game programming for teaching software engineering principles, the two instances of teaching with game programming can be considered a possible means for achieving high effectiveness.

References


