Designing Questioning Strategies for Information Technology Courses

Dr Elozor Shneider
School of Information and Social Sciences, The Open Polytechnic of New Zealand
elozor.shneider@openpolytechnic.ac.nz

Dr Olga Gladkikh
Department of Informational and Mathematical Sciences, New Zealand International Campus, University of Ballarat

Abstract

Testing students within an education system represents a practice that ensures a degree of accomplishment through the educational process and is thus a vital element in the development of high quality education programs. The main objective of the research project described in this paper was to improve and simplify the process of assessment design. The suggested generic approach to designing questioning strategies is based on the application of Bloom's cognitive taxonomy to a range of different subjects and developing problem templates useful at different levels of student assessment. The paper discusses the design of questioning strategies for diagnostic assessments, formative assessments, and summative assessments for information technology courses.

Keywords: Questioning strategies, students' assessment, information technology, Bloom's taxonomy.

1 Introduction

Testing students in an educational system is a practice that ensures a degree of accomplishment through the educational process. It reveals the level of knowledge and skills that students have acquired, their strengths and weaknesses, as well as the effectiveness of academic curricula and teaching methods. Thus, testing is a vital element in the development of high quality education programs.

Design of testing tools can be greatly simplified using information systems capable of generating the required types of test, tailored to the level of the students. The essential step in creating such a system is to develop a rational approach to testing design based on generic patterns derived from modern pedagogical ideas. This is the subject of our research.

Through considerable involvement in the design of various types of assessment for information technology, mathematics and science courses, we have found that, although the particular testing activities are discipline-focused, there is always a common underlying pattern that can be subject to rationalization and "algorithmisation". This finding has reaffirmed our prior belief that design and implementation of generic testing systems is quite achievable. The next step was to develop a strategic approach to test design. The development process, in turn, involves several phases: selecting the conceptual model, deriving generic questioning patterns and building templates based on the selected model, applying the derived patterns to specific domains, testing and evaluating the developed strategies using empirical data. We intend to use an iterative approach that implies attempting each of the development phases several times in order to refine the results and satisfy the extended system requirements so that the information system will evolve and grow with each iteration.

2 Objective of this paper

This paper discusses the design of questioning strategies for diagnostic assessments, formative assessments, and summative assessments for information technology courses. The questioning strategies have been applied in the courses 'Programming with Visual Basic', 'Systems Analysis', and 'Systems Design and Development', taught by the authors at The Open Polytechnic of New Zealand.

Later stages of this project, to be conducted in the near future, will involve the development of questioning strategies for other kinds of learning activities and then development of testing software applications.

3 Selection of a conceptual model

In 1956 a committee of college and university examiners described a hierarchical model for the cognitive domain, commonly known to educators as "Bloom's taxonomy" or "Bloom's model of critical thinking" (Bloom, Englehart, Furst, Hill, and Krathwohl,1956). In its simplest form the model involves a pyramidal layered structure containing the following elements in ascending order: Knowledge, Comprehension, Application, Analysis, Synthesis, and Evaluation. However, further development of a model proposed by Anderson and Krathwohl (2001), and related work of Huitt (1998, 2004), suggested a revision of the relative positions of the upper layers (synthesis and evaluation). These authors also proposed certain changes to the naming of the layers. The original and revised forms of the model are shown in Table 1.

In our project, we have used the modified taxonomy (MT taxonomy in Table 1), which is based on Anderson's interpretation of the cognitive thinking model (Anderson and Krathwohl, 2001). However, we retained the terminology of Bloom's original taxonomy (with the exception of the Knowledge level) since, from our

This quality assured paper appeared at the *19th Annual Conference of the National Advisory Committee on Computing Qualifications* (NACCQ 2006), Wellington, New Zealand. Samuel Mann and Noel Bridgeman (Eds).

Table 1: Bloom's taxonomy and its modifications

Bloom's taxonomy	Revised taxonomy (Anderson & Krathwohl)	Modified Taxonomy used in this paper (MT)
Knowledge	Remembering	Recall (Re)
Comprehension	Understanding	Comprehension (Co)
Application	Applying	Application (Ap)
Analysis	Analysing	Analysis (An)
Synthesis	Evaluating	Evaluation (Ev)
Evaluation	Creating	Synthesis (Sy)

perspective, this terminology represents the activities performed in IT and science disciplines very closely.

4 Developing questioning templates

The diagnostic, formative, and summative assessments can be set at different levels of the Modified Taxonomy. The following is a discussion of the major questioning techniques and methods that can be applied at each of these levels.

4.1 Diagnostic assessment

The Quality Assurance Agency for Higher Education (QAA) Code of Practice states: "*Diagnostic Assessment* provides an indicator of a learner's aptitude and preparedness for a unit or programme of study and identifies possible learning problems."

Although some authors consider diagnostic assessment as a common form of formative assessment (Dictionary, Answers.com, 2006), we believe that together with knowledge-based testing it should include a component that reveals some psychological aspects of students readiness to undertake the selected courses. Consequently, a diagnostic assessment should pursue at least three independent objectives:

1. Testing the level of students' learning skills and their attitude towards learning
2. Determining learning styles most appropriate and beneficial for each student
3. Confirming students' level of competence in terms of the required course prerequisites.

These objectives are reflected in the following parts of the assessment. The first part reveals if the standard of students' learning skills is high enough to undertake the course at the required level. This part can include questionnaires related to students' attitude towards learning, their motivation, and their familiarity with different methods of learning. Among those questions we suggest evaluation and ranking the behaviour patterns for a given scenario; completing the given statements about students' approach to the solution of complex problems, dealing with stress, time management, student's ethics, plagiarism, communication with the teacher and the fellow students. The next group of questions will test students' ability for information management that include mind mapping, cause and effect diagramming, using appropriate citation methods, performing analysis and evaluation exercises that include short Internet searches.

The second part consists of a standard test that determines the students' learning style and suggests appropriate ways of individual learning. It also gives some practical recommendations related to the learning environment that facilitates more efficient studies.

The third part contains knowledge-based questions that cover general prerequisites for a chosen course and can be categorised according to the MT model. The prerequisites include familiarity with basics concepts of discrete mathematics, simple logical operations and students' ability to apply those concepts to solution of simple problems. We suggest splitting part three into two sections.

The first section covers the Recall, Comprehension and Application levels in the MT model. It will include such question types as yes/no – simple and compound statements, multiple choice, and short answer questions for calculation problems. The students will be tested for their ability to perform basic arithmetical operations upon whole numbers (positive and negative), fractions, decimals and percentages, solve linear equations, resolve simple logical problems using distributive and associative rules, and perform reasoning with negations. This section could be offered as either a web-based or a paper-based test. As a result, the students can be given recommendations on relevant readings as well as suggestions to take short-term intensive courses to refresh their knowledge base and polish their skills in problem solving at the Application level.

The second section will probe for a higher cognitive potential. It will include problems and short case studies that require analytical investigation, synthesis and/or evaluation. In tertiary level information technology courses students have to be able to compare concepts and strategies, build conceptual models and perform critical analysis of algorithms. Therefore, students that are not confident in synthesis and evaluation techniques would be significantly disadvantaged. To test analysis skills we suggest including problems that require pattern recognition based on numbers, graphics, lingual sequences and their combinations. The technique of combining two patterns in order to create a third one could be used to reveal the level of synthetic thinking. Similar skills can be applied to the problems that require the synthesis of two or more statements to produce a more general one that combines and reinforces the given statements.

The result of diagnostic assessment should be given in the form of general feedback upon the students' achievement as well as a list of recommendations that could be either a reference to particular texts and articles, sets of problems to practice any skills in need of reinforcement, or a suggestion of one or more short bridging courses.

4.2 Formative assessment

The QAA Code of Practice states: "Formative assessment is designed to provide learners with feedback on progress and inform development, but does not contribute to the overall assessment."

Formative assessment aims to evaluate students' knowledge and skills as well as to make appropriate changes in teaching and learning. Well-designed formative assessment can significantly improve student learning. Nevertheless, there is evidence that "high-quality formative assessment is relatively rare in classrooms and that teachers do not know well how to engage in such assessment" (The Value of Formative Assessment, 2006).

We believe that good formative assessment needs to be clearly structured with respect to the cognitive level of the assessment questions for a given topic. The first step in template development is a thorough analysis of learning outcomes for a topic or a section of the course to be assessed. We found it to be good practice to arrange the learning outcomes in an order corresponding to the levels of the MT model. It is possible that some outcomes would relate to more than one level of MT. Then the question templates are constructed in such a way that each learning outcome is addressed at a corresponding MT level.

To illustrate this approach, we will use a topic "Arrays" from the "Programming with Visual Basic" course with the following set of learning outcomes:

- Define the terms and concepts related to the array structures (Re)
- Establish an array and refer to individual elements in the array (Co, Ap)
- Use repetition structures to initialise an array and traverse its elements (Ap, Sy)
- Perform sorting of array elements (Ap, An, Sy)
- Perform calculations and data manipulations using arrays (Ap, An)
- Merge two ordered arrays (Ap, Sy)
- Bind arrays to GUI objects (Sy, Ev)
- Store and search data in multidimensional arrays (An, Sy, Ev).

The main objective of the formative assessment activities at the Recall cognitive level is to reinforce the terminology and concept definitions. The cognitive taxonomy model (Bloom et al, 1956) associates this level with such actions as "locate", "select", "identify", "define", "describe", and "name". These verbs suggest that students need to make use of their memory; thus, assessment activities at the Recall level should evaluate students' ability to recall and understand key subject terms. The learning outcome at this level is initialising and maintaining a glossary of items that enables students to read and understand technical, course-related literature. At this level, most suitable are selection-type questions, such as True/False or multiple choice, as well as the questions where students have to write a definition of a term or a brief description of a concept with examples. Most of the technical subjects require learning the domain-specific language, therefore the types of questions at this level are most generic and subject-independent. The examples of the questioning template for the selected topic are presented in Table 2 (Recall level).

At the Comprehension level, a deeper understanding of basic terms and concepts should be tested. The key actions suggested in the relevant taxonomy models at this level include "compare", "explain", "summarise", "illustrate" and "generalize". We found that among listed deliverables at the Comprehension level, "analogy" is the most appropriate for building sets of questions for the information technology courses. The idea of analogy and contrast can be successfully used to emphasise the importance of understanding both structural and functional aspects of programming. We have included the questions that reinforce the structural approach and usefulness of arrays in the Comprehension level template (Table 2, Comprehension level).

Unlike the previous two levels, assessment activities at the Application level are designed to test students' abilities to apply the concepts to a particular problem and find the solution. We have found that, among the actions suggested within the MT model at this level, the most appropriate for information technology courses are "apply", "change", "write", and "create", while the most general deliverable is the "solution". Thus, most questions at this level are focused on building code-writing skills: arrays declaration and initialization, manipulation with array data (like sorting arrays), passing array data to functions and performing calculations with arrays (Table 2, Application level).

It is hard to imagine a problem in the information technology area that does not include Analysis as an essential part of the problem solution. That is why developing an ability to think analytically is one of the most important tasks of information technology courses. Such actions as "find", "select", "analyse", "compare", and "determine" can be used in many assessment activities. The important issue in the IT area is "identifying the problem". In programming, "identifying the problem" includes the processes of debugging and refactoring that aim to eliminate errors in code as well as improve the structure of the software application and its efficiency. To illustrate this issue, we have used several examples that relate to analysis of a given erroneous block of code in order to identify the mistakes and omissions. Also, the questions include analysis of a working fragment to establish a clear understanding of the code functionality and outputs (Table 2, Analysis level).

Although Evaluation tasks in information technology courses include different actions of comparison, weighing, assessment and critique, they mostly revolve around such deliverables as choice and recommendations

based on analytical consideration of a particular case (Table 2, Evaluation level).

Assessment activities at the Synthesis level are most challenging and creative. They often include tasks such as "assemble", "design", "develop", "combine", "modify", "rearrange", and "reconstruct" (Table 2, Synthesis level).

Table 2: The example of the questioning template for the formative assessment

MT level	Problem type	Provided input	Expected form of solution
Re	***Define*** the given terms	Concept or term	Description
	Identify the key-terms matching the following descriptions	Two pools containing the list of key-terms and the list of descriptors	A matching arrow-diagram
	Select True or False for given statements and give an explanation for each choice	Statements containing key-terms of the "Array" topic	Label T/F at each statement. A line of explanation for each choice
Co	***Compare and contrast*** a two-dimensional table structure and an array	Not required	Detailed explanation of the two structures accompanied with the list of differences and common features
	Explain giving examples, when should arrays be used to hold data?	Not required	A list of examples with the short explanation notes
	Illustrate passing an array to a function	Not required	An example with a short explanation
Ap	***Write*** code to declare an array with given specifications	Array specifications: array name, size and the data type	Code snippet
	Change the given code so that the elements of an array are printed out in a required format	The code with array declaration and initialization	Code snippet
	Apply sorting procedure to a given array containing N random numbers to arrange the elements in an ascending order	Array definition	Code snippet
	Write a procedure to place the numbers from a given array into the other arrays	Array definition	Code snippet
	Create an application that displays an average, median, maximum and minimum exam scores based on the user input	Exam scores (list)	Working application
An	***Determine*** the output of the given program segment	A program segment containing an array	Output listing
	Find the error in the given block of code	A block of code containing an array	A clear indication of an error with an explanation
	With a single loop, ***determine*** if an array is in ascending order, descending order, both or neither	Not required	Code snippet
	Analyse a given block of code and ***find out*** which lines of code are missing. Fill in the missing lines	A block of code containing an array and aiming to perform a given task. One or two lines in the block are missing.	Lines of code
Ev	Discuss, giving reasons, which type of search would be best for the given array	Definition of array	Discussion with reasons
	Discuss the types of problems where two-dimensional arrays are superior to arrays of structures. Give examples	Not required	Discussion with examples

	For a given scenario, suggest a possible application of an array structure. Discuss advantages and disadvantages of the suggested approach	Short scenario that requires one- or two-dimensional array or an array structure	Suggestion with discussion
	A given function is dealing with a large number of similar objects, introducing them in a sequential manner (one by one). Discuss if there is a better solution that allows representing the function in more compact form	The code for a function to be transformed	Discussion and a code snippet
Sy	***Reconstruct*** the Select Case block in a given fragment of code so that all elements of two arrays are merged into the third array	A program segment	Code snippet
	Bind the content of the two-dimensional array containing user responses to a given question to the GUI objects (Labels)	A block of code containing an array, the sample of output	A program with the GUI objects populated from the array
	Design a decision structure that calls a given function only if a specific condition is valid. The function contains an array as a parameter	Function definition and a condition to be met	Code snippet
	Assume that a GUI contains N Label objects. ***Develop*** the program that creates an array of the Label objects and then sets each Label to a random number between 1 and N	Not required	A program

4.3 Summative assessment

The QAA Code of Practice states: "Summative Assessment provides a measure of achievement or failure made in respect of a learner's performance in relation to the intended learning outcomes of the unit or programme of study".

Although the goals of summative and formative assessment are different, the key actions that can be used in designing the questioning strategies for these assessments are similar. To create templates for summative tests we have used the approach introduced in the section "Formative assessment". Table 3 illustrates the templates that contain examples of the summative assessment questions used in the "Systems Analysis" course.

5 Conclusion

This paper illustrates a layered approach to the design of questioning strategies applicable to different kinds of student assessment. We believe that careful targeting of the assessment questions to the various levels of cognitive thinking will give teachers a powerful tool to identify students' major problem areas and customise the set of methods addressing those problems. We believe that such approaches will significantly enhance the efficiency of both teaching and learning.

The strategies suggested in this paper have been tested in a number of information technology courses taught at The Open Polytechnic of New Zealand. Feedback received from both external moderators and students was highly positive.

Table 3: The example of the questioning template for the summative assessment

MT level	Problem type	Provided input	Expected form of solution	Notes on problem type
Re	Describe a concept	Concept or term	Description	
Co	Identify items/issues with reason	Case study	Items names, identifiers. Reasoning	
Ap	Draw a Use Case diagram	Case Study and extended system requirements	Use case diagram	
	Insert relevant attributes into a class diagram	Case Study, extended system requirements, and a UML class diagram	Updated UML class diagram with attributes	

	Add associations to the class model	Case Study, extended system requirements, and a list of classes in UML notation	UML class diagram with associations	
	Draw a system sequence diagram for the given scenario, using alternate notation for repetitive messages	Case Study, extended system requirements, a relevant scenario or a SSD with repetitive message exchange in it	SSD diagram with the loop structure that reflects the repetitive actions	
An	Assign appropriate actors and Use Cases for a given scenario	Case Study and extended system requirements	A table linking system requirements, actors and use cases	
	Identify and discuss input and output messages between an actor and a system	Case Study, extended system requirements, and a relevant scenario	A list of input and output messages with required parameters between the actor and the system	Additional challenge: distinguish between the source and the actor
	Identify the actions that are best described with the use of a loop. State the loop condition. Suggest a set of input and output messages and parameters for the loop	Case Study, extended system requirements, and a relevant scenario	A list of input and output messages to be included in the loop. Loop termination conditions.	Option: draw a segment of SSD with the described loop
Ev	Discuss case-study related tasks that can be considered as out-of-scope functions	Case Study and extended system requirements	Task names. Reasoning	Step-by-step hints might be provided
	Give the explanation why a class(s) has been removed from the UML class diagram	Case Study, extended system requirements, and a list of omitted classes or a UML class diagram	Reasoning	Increase the challenge by giving a final UML class diagram instead of a list of omitted classes
Sy	Create a Use Case document for a selected use case	Case Study, extended system requirements, and a Use case diagram	Use case descriptor represented in a formatted text form or a table	Hinted version: provide a template for a full Use Case description
	Create a superclass(es) for an appropriate set of classes. Add generalisation(s) to the class diagram. State assumptions	Case Study, extended system requirements, and UML class diagram	Modified UML class diagram, a list of assumptions	

6 References

Anderson, L.W. and Krathwohl, D. (Eds.). (2001): *Taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. New York, Longman.

Bloom, B., Englehart, M., Furst, E., Hill, W., and Krathwohl, D. (1956): *Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive Domain*. New York, Longmans Green.

Code of practice for the assurance of academic quality and standards in higher education. *Section 6: Assessment of students.* Accessed March 9, 2006. (http://www.qaa.ac.uk/academicinfrastructure/codeOfPractice/section6/default.asp)

Dictionary, Answers.com. Accessed March 9, 2006. (http://www.answers.com/topic/assessment)

Huitt, W. (1998): *Critical thinking: An overview.* Educational Psychology Interactive. Valdosta, GA, Valdosta State University.

Huitt, W. (2004): *Bloom et al.'s taxonomy of the cognitive domain*. Educational Psychology Interactive. Valdosta, GA, Valdosta State University. Accessed March 1, 2005. (http://chiron.valdosta.edu/whuitt/col/cogsys/bloom.html)

The Value of Formative Assessment. FairTest Examiner. Accessed March 9, 2006. (http://www.fairtest.org/examarts/winter99/k-forma3.html)