There is no silver bullet to the process of developing interacting e-learning material. In all cases, the domain expert must develop the content and structure it before initiating the next stage which is largely technology-intensive. It is at this stage that tools can be used to support the process. In this paper, we discuss a tool that has been developed to support one of the critical stages in the e-content development process. The software helps content experts to quickly develop self-evaluation formative and summative test-cases that are crucial in the learning process. The background and motivation of the study are discussed and the software design presented. The system is flexible and its crucial features are modular, which makes it portable across various operating systems. Experiences from the use of the software have shown that the task of test-cases development is accomplished within less than 25% of the total time that was previously spent doing the same thing.

Introduction and background

The teaching methods present themselves in a continuum: from expository to heuristic. Starting with the lecture method on the expository end, the next in the continuum is the demonstration method then the history method, followed by discussion or questioning. Others include the assignment and supervised study method and finally discovery or inquiry, in that order. As one moves from the lecture method through to the discovery method, the learners’ participation increases as the teacher’s diminishes. Quite often a teacher may combine these methods during a teaching session (Ayot, 1992). However, some methods are better suited for teaching certain concepts or subjects than others.

In this classical education system, the delivery of course material is through interaction between the lecturer and students in the classroom environment. The process of formative testing is likewise an interactive activity in which different methods come in handy. In the continuum described, some methods are better suited than others.

Current advancements in the learning realms tend towards enabling students to take their courses privately, outside the classroom, and this makes testing assume a somewhat more prominent role. Technology-mediated learning must be supported by technology-mediated testing, hence at each stage of the continuum, a mechanism must be realised to evaluate not only the process but also the output.

Feedback and Capturing Attention. In the lecture method, the teacher is, in most cases, the only active participant in the teaching-learning process and the learners are passive listeners. This learner attribute, in fact, becomes a major disadvantage because learners’ passiveness increases chances of lack of attention and concentration. Notable
The merits of this method include easy presentation of content in a verbal and logical manner while adjusting it to the learner’s level. Note-taking, asking questions, and the use of teaching aids such as charts and diagrams tend to increase the learner’s involvement and hence attention. In e-learning the process takes the form of presentation of material in a sequential manner while ensuring that interactivity is implemented. The questioning technique helps the teacher to get or provide feedback to his students. Self-assessment questions and end-of-unit tests can be excellent substitutes during e-learning sessions. There are a variety of questions which help achieve this. Cognitive questions involve the recollection of facts and procedures, while convergent thinking questions go beyond the recollection of facts and procedures and measure learning at concept and principle levels. On the other hand, divergent thinking questions probe the area of skills learning by seeking originality in students. They involve the learner’s ability to predict, hypothesise, infer or reconstruct. Finally, evaluative questions require that a learner must have passed through all the other levels of learning and he should be somewhere at the productive skills level. These questions are very demanding and the learner should possess a wealth of information and understanding to be able to tackle them effectively.

**Software and the Instructional Process.** With software, it is possible to give instruction using a combination of teaching strategies; the success of which is dependent on the design of the software. Expository methods such as the lecture method are easily modelled as a series of Web pages, which unfortunately may not always serve the same role as lectures presented in a classroom. However, great care must be taken when modelling heuristic methods where the learner is supposed to discover the knowledge by himself. The Internet is like a deep ocean with a large pool of resources that are up for grabs, but require guidance on where to find them. Well-designed Web-based software can provide such guidance (Omwenga, 2003).

There are many different types of software that exist to support the learning process. The instructional process, generally referred to as e-learning, is a necessary approach to providing pace-and space-teaching and learning flexibility. To this end, these software systems should provide the content developer with user-friendly interfaces to support the development of pedagogically-sound materials that have all the necessary ingredients.

**Motivation for the study**

In this paper we discuss a tool that has been developed out of several years of experience in the process of e-content development. The tool is a piece of software that complements e-learning systems that support the instructional process. During the training workshops such as reported in Omwenga (2004) April 2005), it was realised that it is not always possible to enforce strict pedagogical principles when the e-content developers are not professional web designers but rather university professors who want to make their materials available to students within a minimum amount of time using affordable human resources. The development of test cases and self-assessment questions can be a difficult task for many, not only owing to the required professional competence and experience but also owing to the kind of technical demands that are
required of the developer in the use of web authoring tools or equivalents (Omwenga et al., June 2005). For fully-fledged e-learning systems, this is a rather straightforward exercise that can be accomplished within a short time.

Most of such e-learning systems provide templates for filling in the test questions of various types. But for asynchronous e-content that resides on portable media such as CDs, the approach is a lot more complex. This requires techniques that are based on text files or dynamic data structures which are later processed in order to produce run-time codes. In the study with various cohorts of trainees, it was established that although most participants were able to polish up their skills as they went about their work, it was not lost on them that the tasks were challenging. In order to reduce the complex tasks involved and bring the process within the capability of many, it was imperative that certain tasks be automated. It is clear that web design is an activity which requires mastery of technical skills and the ability to learn new techniques. This can be a complex task and needs to be replaced with an automated process. The task of automating test-cases development and hence reducing the overall time of the process was done using a software script that was implemented to run on either text files that are automatically converted into web pages or using a database-driven content repository which stores content objects that are accessed on the fly.

When this software was used in a life application environment, experiments showed that the overall process of test-cases development was accomplished with even better results within only 25% of the total original time.

System description

The software whose acronym is QuizIntegrator, aka QuizMaker, is a web-based system that is designed to run on any platform that supports any web client program. Coded in JavaScript, the system works by first capturing the details of the test cases into an array which is later demarcated into constituent parts to reconstitute an equivalent HTML code to render on the browser. The test cases could optionally be stored in temporary text files or a database that is removed from the disk storage during the garbage-collection process.

The system has been developed on the MS Windows XP operating system platform. HTML and JavaScript were the main programming tools used. To install and use QuizIntegrator effectively, one only need to have at least MS Windows 98 or above operating system and any HTML editor where the HTML code generated is pasted onto an empty page and saved as an HTML file. This is the file that is called from any point within the e-learning CD. A modular design approach was adopted in the development of the system. We briefly describe the interface and process design aspects.

Interface design

The system is web-based and is run from any web client browser. From the home page one can choose to construct any of the four different types of questions in multiples of five. There is also an online help module which gives some hints on how to develop good questions.
Figure 13.1: Overall Interface design for Developing the Test Cases.

Figure 13.1 below helps to explain the interface design approach.

Process Design

The process of developing the quiz is depicted in Figure 2 below. The user has to fill in preliminary information about the quiz: such as special instructions, duration, and name of the quiz author. After this is done, the template for creating the quiz is now ready to accept the test-case objects such as the stem, the choices and the correct answer. Upon completing all the questions, the user will then hit the generate HTML code button in order to generate the code which is pasted onto the HTML editor and saved accordingly. Figure 3 illustrates this part of the process.

Figure 13.2 Quiz development process
Figure 13.3: Upon visually developing the quizzes, one produces the HTML code that is cut and pasted in an HTML editor.

System functionality

The QuizIntegrator has been designed to provide a number of features and functionalities. Different question types are used to accomplish different levels of cognitive testing and reinforcement. The objective type of questions such as the multiple type of questions with optional distracters, true/false questions as well as short-answer questions are used to test concepts covering large portions of the syllabus. Such questions, if well designed, can test the student on simple remembrance of facts and yet also cover application, analytical and synthesising skills. This system is designed to help prepare all these types of questions. The system allows the assignment of weights to answers and offer grading statistics after the quiz has been done and submitted for online marking. The output is displayed with the examinee’s answer alongside the correct answer. The percentage mark attained and the grade are also displayed. The series of screen shots shown below help illustrate these concepts.

**Instructor Side.** The instructor side of the system allows for input of the questions using an interface such as the one shown in Figure 4 below.
The user will provide any instructions and check the correct answer on the right-hand corner of the screen.

**Client Side of the System.** Figure 5 shown below shows the screen that will appear when the student user of the system is ready to tackle the questions. The title of the quiz, the name of the instructor as well as special instructions are all displayed. The student will take the quiz and submit it for marking.

Upon marking the quiz, the student will receive a feedback screen such as the one shown in Figure 6 below advising him on the outcome of the test.

**Figure 13. 5: Learner interface for taking the Quiz**
The information that will be displayed on the window will include the correct answer for the wrong options, the percentage score and the overall grade.

**Suggested extensions of the system**

It is proposed that the QuizIntegrator be extended to provide for pointers to the places where the correct responses are found within the content. This will require careful integration of the system into the e-content for which the quizzes are developed. It will also require a modification of the design to provide for classification of the various types of questions into the five cognitive levels as advocated by Bloom (Bloom, 1956). The time taken to do the quiz might be of help in discriminating among learners. This is also proposed as an extension of this system. As a further extension to incorporate the essay type of questions, machine-learning algorithms shall be employed.

**Conclusion**

The idea of automating the process of developing test cases is not new. But the system described above has some unique features that most others tend to ignore. For instance, being able to generate the HTML code and having the opportunity to change it (although not necessary in most cases) is an important option. Moreover, it is possible to generate the quiz package on the fly without necessarily using a database. This gives the system yet another unique feature that makes it flexible, especially when there is need to produce content on portable media such as CDs.

We have described the system in detail and given the motivation for developing it. The power of this system lies in the amount of time it saves content developers in creating the test cases. It not only reduces the time taken but also makes the process much easier and bearable both for the experienced web designers and the subject professionals alike.
References


