USING *TOOLBOOK* FOR ENGINEERING COURSEWARE DEVELOPMENT
A case-study of online courses for hurricane and earthquake resistant design

Richard P. Clarke

The University of the West Indies, Department of Civil and Environmental Engineering, St. Augustine, Trinidad, rclarke@eng.uwi.tt

Abstract

Given the dependence of the economies of the developing countries of the Caribbean on its human resource capital in turn dependant on the resilience of its housing infrastructure, it is important that design professionals are thoroughly equipped to mitigate the effects of these hazards. Hurricane and earthquake resistant design is taught at the Department of Civil and Environmental Engineering of The University of the West Indies at both the undergraduate and postgraduate levels. A decision was made to develop two courseware programs - one each that covers the effects of hurricanes and earthquakes on Caribbean houses, which is to be used either over the internet within the student's browser, or as a standalone Windows application. Other requirements are that use be made of computer graphics, especially animation, to convey information otherwise difficult or tedious to assimilate. Toolbook Instructor (TI) by SumTotal Systems Inc. was selected as the authoring tool for developing the programs. This paper describes how TI was used for the development of each program after presenting a brief review of several major possible authoring tools that could have been used. The particular tasks involved are described and information provided on the way each task was handled with TI including critical challenges encountered.

Keywords: Courseware development; engineering education; Toolbook Instructor.

INTRODUCTION

The Caribbean region is prone to significant hurricane and earthquake activity. Caribbean housing is typically built without suitable input from professional engineers and uses forms of construction known to be inadequate for providing acceptable levels of resistance to the effects induced within a house by these forces. Even with professional engineering input, recent advances in our understanding of the proper engineering requirements imply a need for substantial re-education of the engineers and technicians. Given the dependence of the economies of the developing countries of the Caribbean on its human resource capital in turn dependant on the resilience of its housing infrastructure, it is of vital importance that awareness of the requirements of proper hurricane and earthquake resistant housing be raised to a level, among both homeowners and the technical community, so as to enable the development of appropriate programmes to address the situation. With the global development of ICT in the form of web-based applications and readily available personal computers with graphical operating systems, and the characteristics of these systems for widespread efficient dissemination of information, it is natural to seek an ICT-based solution to the aforesaid problem of inadequate awareness.

Hurricane and earthquake resistant design is taught at the Department of Civil and Environmental Engineering of The University of the West Indies at both the undergraduate and postgraduate levels. A decision was made in 2002 to develop two courseware programs - one each that covers the effects of hurricanes and earthquakes on Caribbean houses, which is to be used either over the internet within the student's browser, or as a standalone Windows application. Other requirements are that use be made of computer graphics, especially animation, to convey information otherwise difficult or tedious to assimilate.

Engineering courses frequently deal with abstract concepts for which comprehensive ICT-based teaching technologies offer a means of enhancing student learning. Yet the overwhelming majority of instructors do not use such technologies since the unjustified impression is that the exercise is too complex. This is not the case and a wide range of tools exists to facilitate the creation of engineering courseware. Collectively, ICT-technologies form a spectrum of approaches where at one end you have tools such as *MS-Power Point* and *Adobe Acrobat PDF Reader*. Though at this time most tertiary level instructors refer to these products as courseware development tools, to the eLearning professional they are not. According to Hutchinson (2008) "Power Point is not the answer…a Power Point presentation is designed to display a linear presentation of bullet points. Power Point has no bearing on effective….training." Due to its linearity, this criticism also applies to pdf-based courseware. On the other hand, at the other end of the spectrum and representing the state-of-the-art, is the most favored type of presentation called Immersive Learning Simulation (ILS). The eLearning Guild (eLearning Guild 2009) defines ILS, also known as a "Serious Game" as –

> .. an optimized blend of simulation, game element, and pedagogy that leads to the learner being motivated by, and immersed into, the purpose and goals of a learning interaction. Serious games use meaningful contextualization, and optimized experience, to successfully integrate the engagement of well-designed games with serious learning goals.

Hence the tertiary level instructor wishing to utilize the best practices for courseware development is required to design the courseware for highly nonlinear user-interaction. There are dozens of software tools available to the courseware developer and they can be grouped into 6 classes (Hutchinson 2008): (1) Rapid Development – these either require no programming experience or simple drag-and-drop, for example *Articulate*, *Captivate*, and *Toolbook Assistant*. (2) Component Creators - these tools specialize in the creation of dynamic elements and typically are not sufficient to provide all the requirements for courseware development, but their products can be incorporated into the other types of tools. Examples are *Receptivity* and *Flash*. (3) Interactive Tutorial Creators - these can be further subdivided into Assessment Production tools, Games, Software Simulators, and Scenario Creators and examples of these are *Question Mark*, *Thinking Worlds*, *Camtasia*, and *RapideL Discover*, respectively. (4) LMS or LCMS-Based – these are tools that come bundled with the system, for example, the tools provided by *Moodle*, *Blackboard*, and *Saba*. (5) Programming-Based - these tools require a specific programming language to create the courseware and can be sub-divided into tools that use their own built-in controls, frameworks that allow you to import third-party controls, and tools for developing the courseware in a programming language "from scratch". As examples there are respectively, *Toolbook Instructor*, *VBTrain.NET*, and the usual web programming tools of XML/HTML/DHTML, JavaScript, etc, for courseware to be deployed over the internet, or C/C^{++} or any member of the .NET suite etc, for CD-based courseware. Lastly, there are tools that though not requiring programming, nevertheless require sufficient input from the user that they cannot be classed as "Rapid Development" though they are otherwise similar. A typical example of this class of tool is *Authorware*.

In this paper the process whereby *Toolbook Instructor* (TI) was selected is described, as well as how this tool was used to develop the courseware.

TOOL SELECTION PROCESS

Tools Capability

Given the 6 classes of courseware development tools presented earlier it is possible to approximately describe key properties of these classes. Thereafter, knowing the performance requirements for the specific cases of the earthquake and hurricane-resistant design courses in terms of these properties, and other necessary considerations, it is possible to select an appropriate tool. Table 1 shows the main properties of the 6 classes of tools.

Property	Rapid Development	Component Creator	Tutorial Creator	LMS/LCMS-Based	Programming-Based	Other
Simple Animation	✓	✓	✓		✓	✓
Complex Animation		✓			✓	
Built-in Templates	✓	✓	✓			✓
Built-in Controls	✓	✓	✓		✓	✓
Cut-and Paste Content	✓		✓	✓		✓
Imported Controls					✓	
Programming Required				✓	✓	
SCORM Compliant	✓				✓	✓
Very Flexible Creation					✓	
Not Very Flexible Creation	✓	✓	✓			✓
Quick Creation	✓					

Table 1. Properties of Classes of Courseware Development Tools

To select an appropriate courseware development tool requires information in addition to the properties of the various classes of courseware development tools. The following, though not exhaustive, must also at least be considered – browser-independence; platform-independence; accessibility; conformance to web standards, and portability.

Performance Requirements

In this specific instance the requirement was for courseware for teaching the structural design essentials for hurricane and earthquake-resistant Caribbean housing. As is frequently the case for engineering content, critical concepts are described mathematically and this abstraction presents difficulties for many students. To address this challenge, fortunately the mathematics is describing physical phenomena that can be graphically simulated. Hence it is the author's belief that the aforesaid philosophy of Immersive Learning Simulation is critical for engineering education using eLearning technologies. This translates to the need for relatively complex animations to implement the graphical simulations. For example, with respect to the hurricane-resistant design there must be animations of the different ways a house can fail during a hurricane. Likewise, the simulation of the collapse of a typical Caribbean house under an earthquake is required. Other aspects of the specification are quite simple in that the final product must be deployable both over the internet, and via CD-ROM on a PC offline, and for MS-Internet Explorer and Windows, respectively, given the present infrastructure into which the courseware is intended to function. As a secondary item, the courseware products must be written in such a manner as to enable future expansion in the form of student assessment exercises.

Selection

With the aforesaid properties of the various classes of courseware development tools and other important considerations, and the specifications defining the performance requirements, it is clear that what is needed is a tool that has the properties of both "Rapid Development" and "Programming-Based" types of

tools. The former is especially required for quick creation, and the latter for complex animation, and very flexible creation. Of the very many tools available *Toolbook Instructor* was selected because it met the criteria, and additionally contains templates for providing the future student assessment ability. Additionally, and for the same reasons of speed and flexibility, the author already had considerable experience with the tool since 1991. The following endorsement from the eLearning Guild is also important, as reported at the Street Directory website (Street Directory 2009) based on events at the 2007 international conference and exposition of the American Society of Training and Development:

> "When the eLearning Guild published its 360-degree report in Immersive Learning Simulations, we presented SumTotal ToolBook with a Member's Choice Market Share Gold Award," said Steve Wexler, director of Research and Emerging Technologies for the Santa Rosa, Calif.-based eLearning Guild, a global network of more than 25,000 e-learning professionals.

TI 8.5 was the version used to develop the courseware since this was the latest version in 2002. The current version is 9.0. *TI* is owned by SumTotal Systems Inc. (though previously by Click-2-Learn Ltd, and before that by Asymmetrix Ltd). A more lightweight version is called *Toolbook Assistant* but it does not have the programming capabilities. The programming language is called *OpenScript* which is an object-oriented language. The language is an interpreted language hence requires an interpreter. The interpreter, called the *Toolbook* reader must be packaged along with the binary script file (extension .tbk) and the interpreter's DLLs for CD-ROM delivery. For internet-based delivery the *Neuron* browser plug-in is required. This is delivered to the client who runs the installer and from that point onwards the browser can run any courseware developed with *TI*. To do this the browser must request an html file from the web server where the files are stored. The web server MIME must be set to recognize .tbk files. The html file has the appropriate object id for enabling the browser to open a window for the courseware .tbk file to run in. The courseware products are available at the following website: http://ideascaribbean.com/hurri/. They have been used by over 400 students so far with very favorable and encouraging feedback.

USING TOOLBOOK

How to use Toolbook is obviously covered in the manufacturer's documentation. However, in this section a brief review of the steps used to develop the courseware is presented with particular emphasis on challenges encountered and workarounds employed. The same process was used for each courseware item so only the general process is described:

Step 1 – Define the screen layouts. This is done using hand sketches on a scratchpad such that each possible screen occupies a page. The order is irrelevant except for the first sheet since what is displayed next depends on the user's choice. For each sheet a sketch is done of exactly what the user will see or can see depending on his choices in which case certain objects initially can be invisible. Each sheet is marked-up for colors, fonts types and style.

Step 2 – Define the experience. In this step the developer creates a notepad and details exactly what he wants the user of the courseware to experience when she interacts with any object on the screen.

Step 3 – Select a template for the book and for each sheet or page select the controls from *TI*. *TI* has templates for many different types of courseware. From the sketchpad, the developer selects the template that matches the design. In this case it was not necessary to use a book template so the "blank document" selection was made. *TI* uses a hierarchy of objects in the order of system->book->page->background->controls. The controls are mainly selected from a drawing palette and include, for example, text boxes, buttons, geometric shapes, multimedia viewers, etc. These are drawn based on how they are to look as designed in the sketchpad. This is achieved by selecting, for any particular control, its set of properties. Powerfully, *TI* can also create objects and set their values from its programming language *Openscript*, but it is rare that this is necessary and for this courseware the controls are created by using the drawing palette. *TI* works using event-driven programming where an event is whatever can happen to an object. The critical thing is that an event generates a number of messages and the user determines how the courseware is to react to these messages using what are called "message handlers". For example, clicking on a button is an event that causes the message "ButtonDown". If the developer wants something to happen when this message is triggered she can

program the response using *Openscript*. The messages are passed to the other objects in reverse order in the hierarchy presented earlier. In earlier versions of *TI* this was the only way to handle messages and the language was designed to be English-like to make the process easier. *Openscript* is so flexible that it also allows access to the Windows API at the system level, and the creation of new classes at the other extreme, and the ability to run other executables from within its environment. However, market response indicated that it was necessary to enable message-handling without programming and subsequent versions of *TI* also use a catalogue of pre-defined message-handlers. Since a *TI* program is actually a binary script file, in effect the catalogue is a code-generator and upon its use, *TI* automatically creates the code in the file. The ability to use *TI*, or any courseware tool, without programming is at the expense of flexibility in terms of getting exactly what you want as the developer, and in many instances, execution speed is also reduced. Typically, for really interesting projects it is rarely the case that absolutely no programming is required. Since the author's *TI* experience goes back to version 1.5 when programming was mandatory, this programming-based approach was used for this project.

With these simple steps, the *TI* book was created page by page as dictated by the scratchpad and notes of what you want the user to experience, and using *TI*'s many features. However, *TI* was selected because of a mission-critical requirement that in the author's opinion will be typical for most engineering education courseware, and science or math courseware as well. As discussed earlier, this is being described in the current jargon as Immersive Learning Simulation. In the specific case, this is the need to simulate failure processes in Caribbean houses under hurricanes and earthquakes using animation.

From a courseware development perspective animation is implemented by the following techniques: (1) for each object mathematical transform matrices are applied to the data representing the vector form of the object and this may also require the use of clipping algorithms for 3D representations. (2) The show-and-hide method in which many objects are drawn representing how the object looks as it moves along the intended path, with each successive object made visible then invisible to that it seems as if there is only one object moving. (3) Moving the object from point-to-point on the screen by exploiting the screen's coordinate system. Each of these methods can be implemented in *TI* 8.5 using *Openscript* and the commands "show" and "hide" for method 2, and the command "move" for method 3. However, *TI* 8.5 has also made provisions for a non-programming way to implement method 3. In this approach, *TI* 8.5 has the user draw a path for the object and it generates the data "under the hood" so the user does not need to be concerned with the details. Unfortunately, in the author's view, *TI* 8.5's implementation of this method is terrible. This is because the way it works is clunky and you have to make several attempts for one object so it is practically impossible to use this approach to method 3 for this project given the number of objects that must move simultaneously and in a coordinated fashion. Again due to the number of objects involved, methods 1 and 2 are also not feasible. Hence a workaround is required and it was only because this was known beforehand that the project was attempted with the degree of animation envisaged.

The workaround is to use a recorder in *TI*. A recorder is a software utility that you switch on, then using the mouse, move the object or objects bit-by-bit until their paths are travelled then, you switch off the recorder. The record of what was done is stored textually as the equivalent set of programming code lines required to achieve what was done. In other words, the recorder works as a text-based code generator. You then assign this code to the handler of the appropriate message. The reason this was not mentioned earlier is that this feature is not available in *TI* 8.5. However, it is available in *TI* 1.53 so the author merely moved the project file to this version, used the recorder there and cut-and-paste the generated code to the *TI* 8.5 project file. The *TI* 8.5 was still the major tool since the *TI* 1.53 requires a cumbersome translation step if a .tbk file developed using it, is to be deployed over the internet. The author is unaware of why the recorder was removed from later versions of *TI*.

CONCLUSION AND RECOMMENDATIONS

Practical eLearning for engineering education requires Immersive Learning Simulation which in this context means complex animations of physical phenomena usually represented by complex mathematical constructs. The feature set of *Toolbook Instructor* by SumTotal Systems Inc. is sufficiently rich that it can be used effectively as a courseware authoring system to achieve this aim, as was demonstrated in this paper for the cases of the structural design of hurricane and earthquake-resistant Caribbean houses.

The author therefore highly recommends the tool and hopes that this encourages engineering educators and eLearning professionals to make use of it as we seek to improve the learning experience of our future engineers, scientists and mathematicians in general.

REFERENCES

Hutchinson, P. 2008. http://pipwerks.com/journal/2008/01/20/how-i-build-my-elearning-courses/

eLearning Guild. 2009. http://www.elearningguild.com/research/archives/index.cfm?id=128&action=viewonly

Street Directory. 2009. http://www.streetdirectory.com/travel_guide/137117/software/toolbook__sumtotal__software_features__benefits.html