Cost of Migrating Large-Scale Computer Assisted Learning (CAL) Software to Web Delivery

Simon Price[1] and Ian M Marshall[2]

[1]University of Bristol, [2]University of Abertay Dundee

© EURODL 2001

Abstract

This paper presents an initial analysis of the duration and effort data collected during the migration of WinEcon, a large-scale CAL package to WWW-based delivery. The paper reviews the data collected during the initial development of WinEcon before presenting the preliminary data collected during the recent migration. The data presented represents a snapshot of the data collection process towards the end of the migration project including an overview of the raw project data in terms of cost, duration and effort. Initial lessons learned from the project will be presented along with the potentially controversial view that conversion of large-scale projects, while feasible, may not represent the most efficient use of resources. The authors will argue that in many cases it may be more efficient to start again rather than migrate and reuse exiting tools and technology.

Key words
Migration, Cost, Effort, CAL, WWW

1   Introduction

In 1992 the original Microsoft Windows PC/CD-ROM/LAN-based WinEcon courseware project was funded under the United Kingdom Higher Education Funding Council's (HEFC) Teaching and Learning Technologies Programme (TLTP) [1,2]. This involved a Consortium of 8 universities who collaborated on the project which was co-ordinated by the Centre for Computing in Economics at the University of Bristol. The Consortium's aim was to produce courseware to replace tutorials and augment lectures for approximately 12,000 students per academic year studying 'Introductory Economics' courses at UK universities [3]. Each of WinEcon's 25 chapters of courseware made extensive use of text, graphics, simulation and student interaction to deliver on average between three and five hours of student activity. In addition, the finished courseware was customisable to meet the requirements of individual lecturers. Although the WinEcon courseware was supplied at nominal cost to HEFC-supported institutions in the UK, it has been sold to commercial organisations and overseas higher education establishments and has also won a number of prizes for its educational quality.
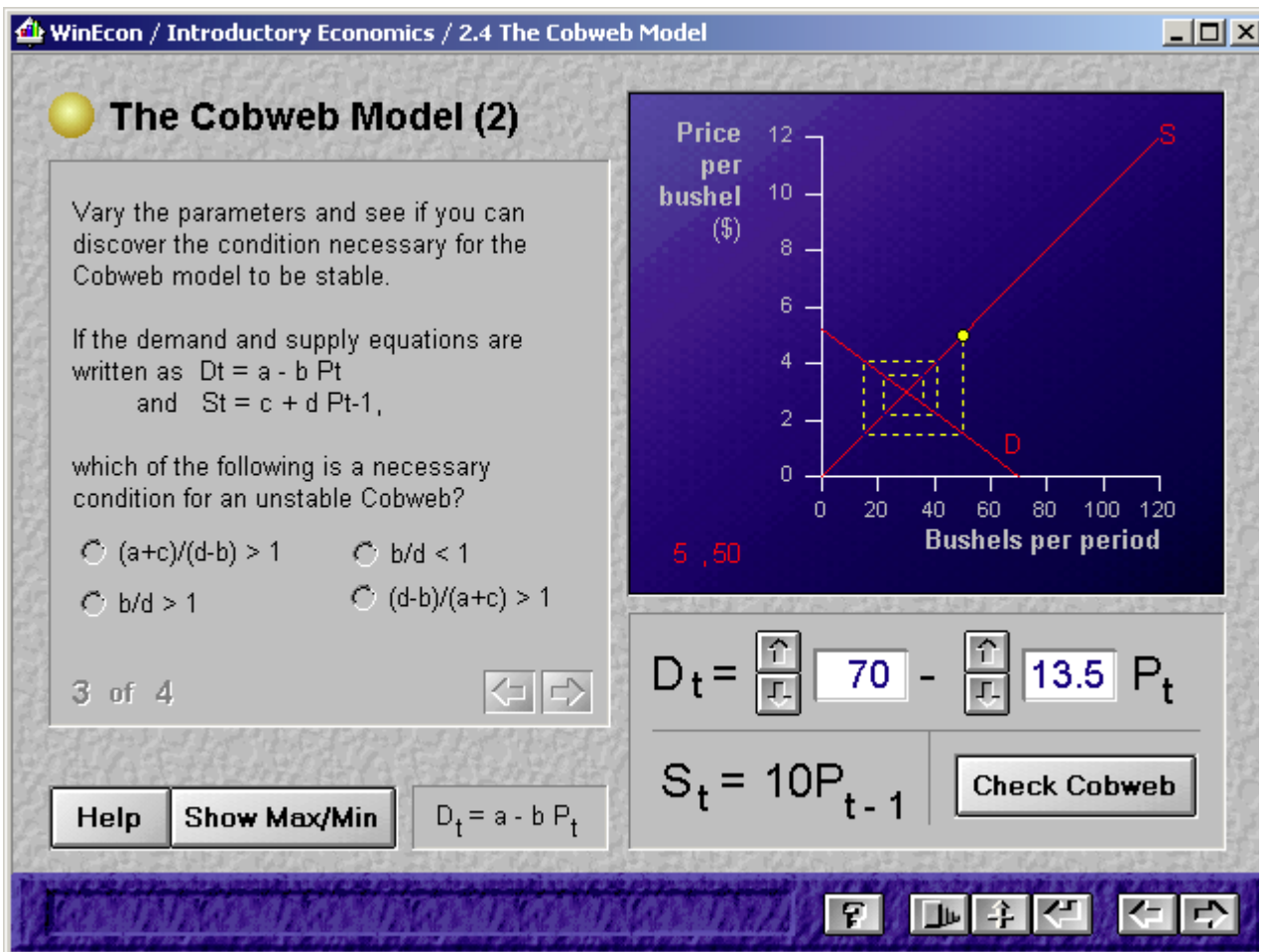
Figure 1 – Screenshot of the Original Asymetrix ToolBook version of WinEcon

By the Autumn of 1997 it became clear that a Web-based version of WinEcon would be required sooner rather than later. It became increasingly obvious that the headlong dash by universities and colleges for WWW-based products could make WinEcon obsolete. WinEcon's Support Desk was beginning to receive growing numbers of requests similar to the following "... it's very good but does it work on the Web?" or " ...sorry, but we now only use products that run in WebCT. Do you have a WWW-based version?". This highlights a problem with all forms of technology-based educational products. As Marshall et al [4] suggested "... in the modern world technologies come and go, leaving very little except store cupboards full of obsolete hardware, and a small selection of inadequate and non-transferable courseware". It was now the turn of many good multimedia and CAL packages developed in the early 1990s to suffer the same fate. The decision was therefore taken to update WinEcon to allow WWW-based delivery using funding from the successor to TLTP [5]. Once funding was received, the project to migrate WinEcon to a WWW-based format was begun in 1998.

There appears to be many advantages in reusing or migrating existing CAL or multimedia products to alternative platforms, but there is very little hard data or information available to help developers to make decisions about the viability of this or any other strategy. This paper will present an overview of the process and the lessons learnt which may be of general use to other developers migrating their CAL products to the WWW. The original WinEcon project was used to collect and analysis data about courseware development effort estimation [6]. This paper presents some initial findings about the migration project as a prelude to a more detailed analysis.

2   Data Collection and Analysis

2.1   Overview of the Original WinEcon

The project was originally intended to start in October 1992 with the completed courseware to be delivered in September 1994 for the first non-pilot study student use in October 1994. During this twenty-four month period the Consortium members were expected to design and develop commercial quality courseware from an outline specification which they had submitted to the HEFC to win the funding bid. The core development team, based at Bristol, were responsible for project co-ordination, overall design and the provision of development tools. To ensure commitment to the project, each of the Consortium members was allocated individual courseware chapters to develop along with the financial resources to employ courseware development teams.

WinEcon courseware was developed in Asymetrix ToolBook, the object-oriented authoring language based

around a book metaphor [7,8]. Pages form the basis of the high level structure while objects (including pages) can have OpenScript programs attached. OpenScript is a fully functional object-oriented programming language. The finished courseware used the ToolBook Runtime to deliver the courseware. This runs the courseware but prevents students from editing pages or changing OpenScript programs. During the project ToolBook was upgraded from version 1.5 to version 3.0. This required considerable reworking of the original code to take advantage of built-in features which had previously required specially developed functions. The book metaphor was extended to describe the modules which made up the courseware with a related collection of pages being described as a chapter. The twenty-six chapters planned in the original WinEcon submission to HEFC were reduced to twenty-five following a re-analysis of the content. Chapter 9 was dropped but the original chapter numberings were retained internally.

The most obvious problems encountered in the actual delivery of the original WinEcon project was that it overran the planned end of the contract by twelve months and that the courseware released in October 1995 had reduced functionality in comparison to the original specification. For example, the student record keeping system was missing although this was included in a maintenance upgrade launched in December 1995. The original bid documentation had also suggested the inclusion of extensive video resources. This was dropped quite quickly due to the cost and the perceived difficulty in guaranteeing adequate delivery platforms for student use in client universities.

The WinEcon project was used as the basis of research to capture metrics data in an attempt to develop accurate courseware development effort prediction methods. Project records, documentation and questionnaires were used to collect development data for each of the chapters produced by the Consortium. The finished WinEcon source code, executables and project documentation were also analysed to count the number of objectives, screens, interactions and media objects. An automated measurement tool was used to collect 53 measures from the WinEcon program source code [4]. Table 1 describes a summary of the data collected against the original WinEcon project.

Table 1 - Summary of WinEcon courseware results

Heading	Description	Total course
Sizing factors	Estimated *learner time (Lh)*	69.7
	Screens	4113
	Interactions	456
	Lessons	25
	Objectives	150
	Media objects	620
	Source lines of code *(SLOC) – No comments or padding*	134,844
Measures	*Elapsed time (days)*	1170
	Development time (h)	26,045
	Effort *(Dh)*	35,651
	Team size	111
	FTE developers	34.05
	Cost (£)	1,000,000
	Productivity (Lh Dh $^{-1}$)	0.0020
	Effort *to learner time ratio (Dh Lh^{-1})*	511
	Developer effort *charge (£ Dh^{-1})*	16.83

The data collected from the WinEcon courseware development project was analysed to produce various courseware development effort estimation models [4].

2.2   Migration of WinEcon to WWW-based delivery

Initial analysis of the problem suggested two alternative methods of migrating WinEcon to a WWW format. The project team had to choose between:

- Re-originating the entire application in a WWW-orientated environment such as DHTML or JAVA Applets.
- Migrating the existing code-base to a browser plug-in version of ToolBook, called Neuron.

The funding available for the migration represented only 25% (£250,000) of the original WinEcon development budget. It was therefore decided to adopt the plug-in route which appeared to be the far more cost effective and low risk option than re-coding in either DHTML or JAVA. According to the Asymetrix's product specification Neuron was essentially a runtime version of ToolBook 5.0 which could be embedded in a WWW page. As the original WinEcon product was built using ToolBook 3.1, the migration appeared straightforward using the Asymetrix's upgrade tool to update the existing code to ToolBook 5.0 specification and then create suitable WWW pages to act as containers for Neuron.

Lecturer customisation features enhancement

Rather than do a straight port to WWW the project team took the opportunity to enhance the features available within WinEcon. Part of the rationale behind converting WinEcon to WWW format was to enable academics to pick and mix pages of the courseware with their own lecture notes, PowerPoint slides, quizzes and other materials. The project team described this process as the 'componentisation' and it involved breaking the existing single 20MByte WinEcon ToolBook file into approximately 1000 separate WWW pages that could be used individually or in any lecturer-defined combination. This required the existing ToolBook book to be split up into hundreds of smaller books, each consisting of one or two ToolBook pages. A ToolBook page represents the smallest atomistic, inseparable learning object possible in the WinEcon pedagogical model. It also required the exporting of all the learning object metadata from a bespoke database, known as WinEcon Lecturer, into a format which could be incorporated into the Web pages. This metadata included items such as navigation links, glossary references, references to textbooks and introductory or summary text. Both the decomposition of the large ToolBook book and the exporting from WinEcon Lecturer required the writing of custom utilities as neither function are supported by ToolBook itself.

Fortunately WinEcon had been initially designed and developed with a complex central core library that encapsulated all the system-wide features and functionality. All of the pages in the courseware shared this central ToolBook System Book. Consequently, when changes needed to be made across the entire application, these were restricted to the 7,344 lines of the System Book code rather than the 134,844 lines of code spread over the 697 ToolBook pages of the courseware content.

Use of CALnet as a conversion tool

To facilitate the conversion process a pre-existing in-house WWW authoring tool called CALnet [9] was used to generate the pages that contained the componentised WinEcon pages from the metadata exported from WinEcon Lecturer. CALnet is a simple WWW-orientated macro processor written in JAVA which greatly simplified the migration process even though custom macro definitions and transformations had to be defined for the task. However, CALnet had not been originally designed to cope with generating more than 10-20 Web pages at a time and so had to be enhanced to cope with outputting more than 1000-2000 HTML, and later, XML files.

CALnet also had a role to play in the migration of WinEcon's self-assessment quizzes and formal exam tests to the WWW. WinEcon uses its own bespoke Computer Assisted Assessment engine built in ToolBook which is designed to run on a LAN. The architecture of this was fundamentally incompatible with browser delivery under Neuron for security, performance and other technical reasons. It was therefore decided to completely re-originate the questions in DHTML using another set of CALnet macros and transformations in conjunction with data exported from WinEcon Lecturer. This was highly successful, taking relatively little time to do and the finished product proving more popular than the original which had began to look outdated in comparison to newer WWW-based interfaces.

Adoption of IMS standards

The original WinEcon had a bespoke customisation system that allowed lecturers to re-arrange the courseware into arbitrary sections, chapters and courses and to associate glossary terms, references and questions with any part of a course. This customisation feature was dropped in favour of a move towards offering support for third-party Virtual Learning Environments (VLE) and the emerging international Instructional Management System (IMS) standards. Simply having the courseware available in WWW format gave instant compatibility with all the major VLEs and creating IMS manifest XML descriptors of each learning object gave standards compliance for the courseware. The questions, however, required more work as, at the time of developing, the IMS question item interchange format was neither fully defined nor fully supported by the mainstream VLEs. Consequently, the questions were transformed into the appropriate import formats for WebCT and Blackboard, once again using CALnet. This process resulted in the loss of some interactivity because the target VLEs had lower capabilities than the bespoke WinEcon assessment engine in terms of user feedback and question types.

Outputs of migration of WinEcon to WebEcon

The migration of WinEcon to a WWW version, WebEcon, was completed successfully and Blackwell Publishers are now distributing the software. Initial user feedback is positive and institutional uptake is looking promising. However, the estimated elapsed time for the migration project was 9 months to beta and a further 3 months to final release giving 12 months in total. This estimated elapsed time for the migration project was based on an experimental conversion of the WinEcon from ToolBook 3.1 to 5.0 plus a technical feasibility test that showed that the Neuron plug-in worked as promised. The technical feasibility test conversion took only two days and worked almost first time. However, the test simply ran the entire WinEcon application inside a single WWW page and, importantly as it later turned out, did not break WinEcon up into multiple smaller components, as the full migration would need to. In reality, the elapsed time for the delivery was 9 months to beta and a further 27 months to final release, 33 months in total i.e. 2 years late. The actual development effort was in fact only 3 developer months over budget, but this additional effort was spread over a 2-year period.

The output of the WinEcon to WebEcon migration involved the:

- Conversion of 697 tutorial screens from ToolBook 3.1 to ToolBook 5.0, then 8.0 and finally 8.1. This required the automatic or manual updating of:
    - 134,844 lines of ToolBook OpenScript programming code
    - 215 interactive graphs
    - 1,077 information pop-ups
    - 2,965 illustrative graphics

- Creation of 937 HTML pages to present and organise the ToolBook tutorial screens
- Conversion of 23 exam and 24 self assessment quizzes (866 questions in total) to HTML
- Conversion of a database of 617 glossary terms to HTML
- Conversion of a database of 2,010 textbook page references (i.e. additional reading) to HTML
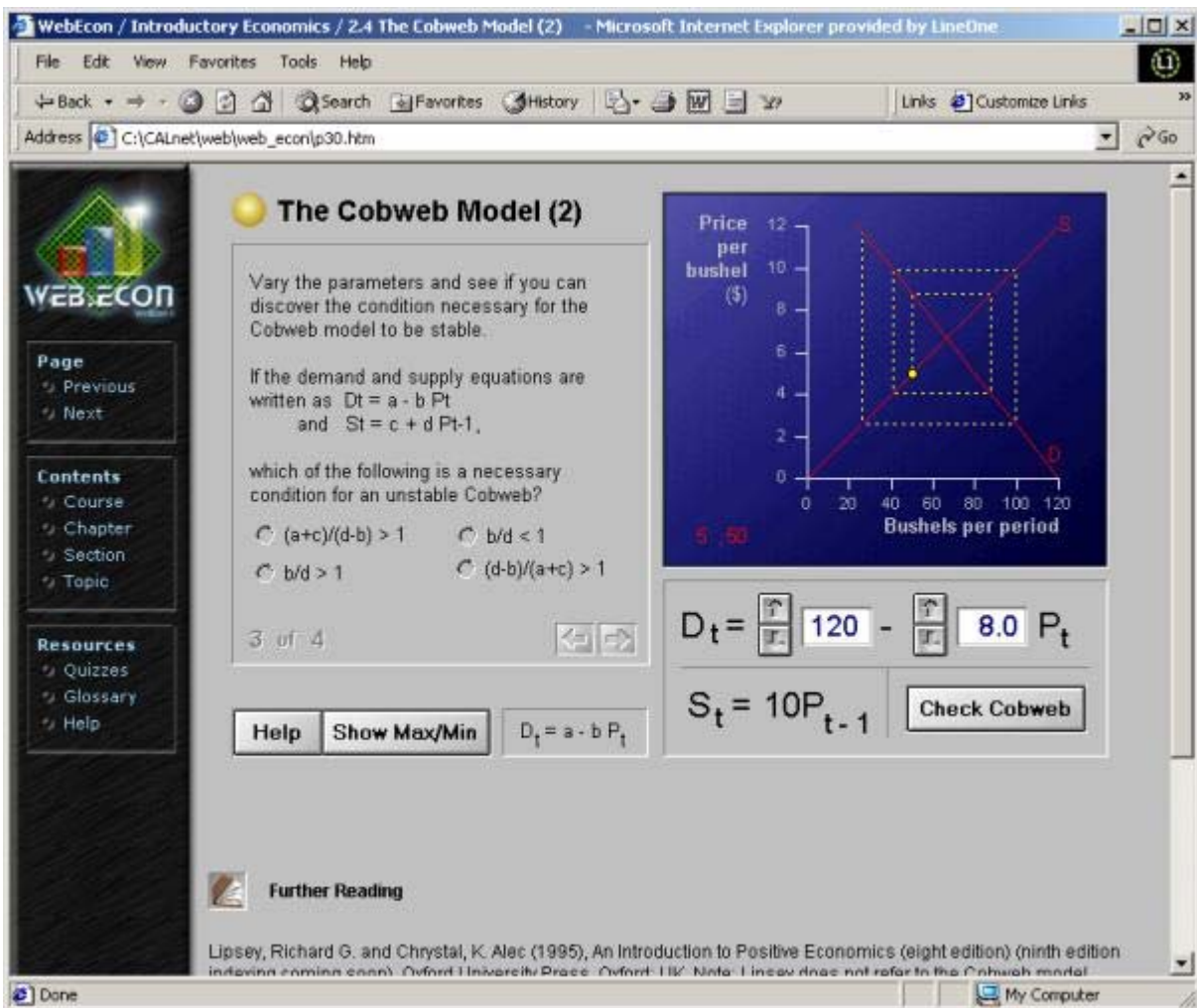- Adaption of 7,344 lines of core system programming code to run in WWW browser

Figure 2 – Screenshot of the WWW-based Version of WinEcon using Asymetrix's Neuron

3   Analysis and Conclusion

3.1   Analysis

The migration of an existing body of courseware for browser delivery under essentially the same runtime environment proved to be far from the smooth transition that was envisaged. The reasons for this range from non-technical user interface issues through to highly technical details relating to the constant change in technology over time. The following highlights some of the major issues that resulted in delays in delivering WebEcon on time.

Adapting a WWW browser interface

At the simplest level, the user interface and WWW browsing paradigm was sometimes at odds with the proprietary, home-grown user interface conventions of WinEcon. In terms of overall navigation, WinEcon adopted ToolBook's native page metaphor which mapped particularly well onto the Web. However, WinEcon used its own navigation control panel down the bottom of the screen, paralleling the browser's own tool bar and so this, plus any references to this features, had to be removed from the software. The control panel also included dynamic help text which appeared as the mouse pointer moved over any object on the WinEcon page. This had to be replaced by introducing Windows standard tooltips that popped up at the mouse pointer if it rested over an object. More complicated problems arose when cross referencing between WinEcon pages where the result of clicking on a cross reference button was that the browser navigated to another Web page rather than the plug-in displaying another ToolBook page. A lookup table was required to determine the relevant container for any given ToolBook page. Similar problems occurred with the glossary where a jump to one of 26 Web pages was required instead of simply popping up a definition in a separate window as previously occurred in WinEcon. References to further reading, which used to appear in a popup window, were simply tacked onto the bottom of the HTML page to make use of a small amount of spare space at the bottom of the Web page.

The courseware itself occasionally required specific one-off edits to correct user interface "errors" caused by different conventions in a browser as compared to a stand-alone program. The most abundant of these was the use of highlighted blue text on some WinEcon pages which in a browser was interpreted by users as a hyperlink but in reality was merely a visual highlight. These were changed to bold black text. Figure 3 shows an unedited WinEcon screen showing blue text with underlines which users frequently interpreted as clickable hyperlinks.
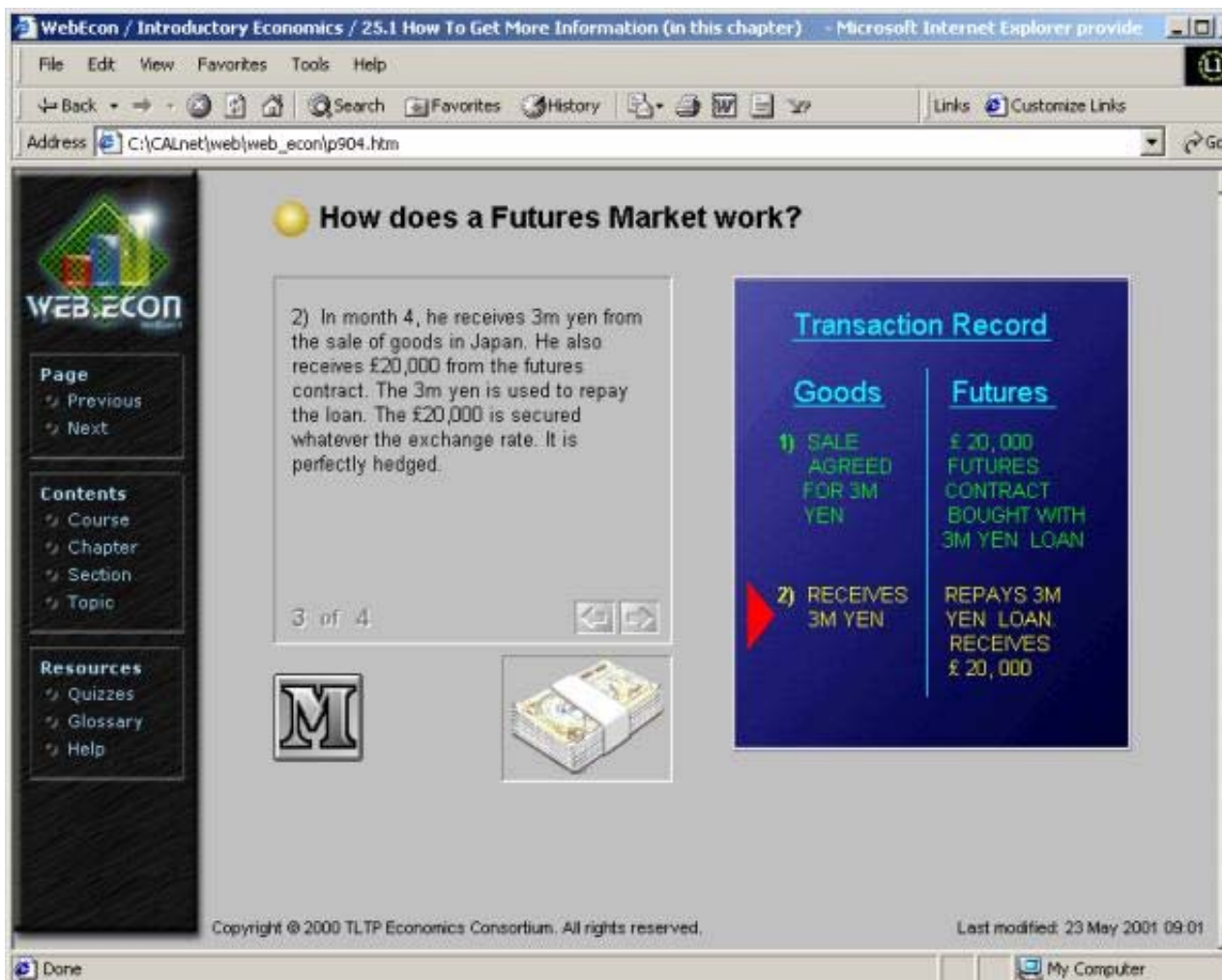

Figure 3 – An example in the Transaction Record of blue underlined text

Browser-based screen real estate

One of the more surprising user interface issues is that of screen real estate. WinEcon was developed when the normal screen resolution was 640 x 480. A safe assumption in UK HE nowadays is a resolution of at

least 800 x 600 and so on the face of it, one would think that there would be a lot of spare space on a WebEcon HTML page because the Neuron plug-in only occupied around 640 x 480 pixels. However, when the typical array of browser toolbars, location bar and status bar are taken off the vertical height then the available vertical space is barely more than the 480 height of old. This influenced the HTML graphic design and positioning of the navigation bar used in WebEcon which was located down the left side rather than across the top of the page because there was more horizontal space than vertical.

Legacy code and Windows conventions

Moving beyond the user interface to more technical issues, the change in running environment and operating systems caused a number of more serious problems for WebEcon. Seven pages out of the 697 used an old 16-bit spreadsheet DLL that pre-dated COM, ActiveX and OCX and which crashed the browser if run within Neuron. This feature had to be dropped because there was no way to cheaply convert the 16-bit legacy code. Less seriously, but with an ability to cause a spectacular crash, were WinEcon's attempts to manipulate (e.g to minimise or maximise) the Windows containing the plug-in. This fault required the core of WinEcon to be re-coded because in the plug-in version the topmost Window was no longer ToolBook but Internet Explorer or Netscape.

Deployment and support issues

When the beta version of WebEcon was released it became apparent that there were a number of deployment and support issues relating to the use of the Neuron plug-in. Many of these are general issues relating to any browser plug-in and centre around the move towards the managed desktop. Institutions and organisations establish highly managed client configurations to help control the costs of supporting large numbers of workstations and under such regimes a strong case has to be made before the system administrators will consent to the addition of a new piece of software which they have to support. Browser plug-ins are treated as completely new pieces of software requiring their own support budget, skills and expertise. At best this can result in extended lead times for getting Neuron installed on a system and at worst it can be turned down.

An often-cited reason for not allowing plug-ins is the security risk they (potentially) pose. WebEcon for instance must run in Neuron's "Non-Secure Mode" in order to access DLLs and manipulate the browser. Neuron is also 16-bit and as such runs it as a NT virtual device manager. This has potential security holes and thus is now disallowed for use by some large organisations e.g. the USA Army. Another valid reason for turning down an installation is a notorious versioning problem with Neuron whereby different versions must be installed in ascending release order number sequence (i.e. version 5 before version 6 before version 7 etc.) or else some versions will fail to function. It is common in education for more than one version of Neuron to be installed on a system and to install an earlier version; all later versions must first be uninstalled, the older version installed and then all the newer versions installed in version number order.

Other people's bugs

The remaining issues can be summarised as being the result of using immature technology which has not been fully debugged or widely configuration tested in the field. Both Internet Explorer and Netscape have a policy of not unloading a plug-in on one page until after the next page has loaded. This inevitably results in two instances of a plug-in running simultaneously when you move from one page with a plug-in to another page with a plug-in. With Neuron this depleted all the available system memory and the browser locked up. The only solution was to route all plug-in page to plug-in page navigation via a blank dummy page that waits one second before then continuing on to the true destination page. The one second delay allows the browser time to unload the first plug-in before it proceeds to the next page with a plug-in.

The final technical problem occurred when the project hit a show stopping bug (actually a set of bugs) in Windows 2000 and Neuron that meant that the browser locked up when navigating from page to page on a Windows 2000 PC. The project team reported the problem to Asymetrix, which was called Click2learn.com by this time, and had to postpone the release of the software until a solution was forthcoming. Click2learn blamed Microsoft and said that they had to wait for a fix from Microsoft before they could fix their end of the problem. Each time a new version of ToolBook came out the project re-upgraded WinEcon and tested to see if the problem went away. Each upgrade and test took around two days to a week to perform and this went on over a two-year period with WebEcon being put back on the shelf each time. Finally, with ToolBook 8.1 and Windows 2000 SP2 plus a Microsoft hotfix the problem was resolved and it became possible to release WebEcon.

3.2   Conclusions

McDonough, Strivens and Rada [10] stated that the "... development of CBT is often expensive in terms of both time and money and all researchers agree that to be viable it must achieve high student usage ... High student usage should still be possible if the courseware is easy to customise and reuse elsewhere". This suggests that to develop a sustainable individualised interactive learning revolution, courseware has to be designed to be transferable to other platforms. Without this ability, developers are doomed to continuously re-develop courseware with each change in hardware and software platform or consumer or educational taste. The current thrust for the establishment of international standards for courseware

authoring and delivery are intended to make the process simpler. However, there exists a wealth of legacy courseware adequately fulfilling a role in the education of students which will require to be migrated or replaced. So what lessons can be learned from the project to migrate WinEcon to a WWW-based platform?

Clearly, the migration of WinEcon to the WWW had problems and yet the most worrying aspect of this is that WinEcon is, in many respects, a best case scenario. WinEcon is a large-scale CAL program developed by a distributed team of developers using a tightly defined set of standards in conjunction with a custom-written core system. The distributed nature of the original development meant that the courseware was developed in a compenentised fashion in the first place and then aggregated together into a single ToolBook book at the end. The original courseware architect and project manager was in charge of the migration project as were some of the original developers. There also existed a wealth of tools developed for other projects to aid the migration process.

Splitting WinEcon into its learning objects should therefore have been a simple case of dividing the components up with no changes necessary to the program code because each learning object was originally designed to be entirely self contained. Almost all the programming changes that were required were confined to the complex, but small core system code rather than being distributed across the courseware pages. WinEcon also had the benefit of already having externally stored metadata describing each learning object stored in a format from which it was possible to export into the required format. Smaller CAL projects or ones which had not been developed with a highly modular architecture may well prove far more expensive to migrate to the WWW using plug-ins because they would not benefit from the economies of scale enjoyed by WinEcon or because their changes are less localised and repetitive.

However, the final product was delivered 2 years late primarily because of minor bugs in other people's software which were eventually resolved as part of the general upgrading of ToolBook and Windows 2000. In the end WebEcon certainly works but it has slightly fewer interactions because of the loss of the 16-bit spreadsheet tool and loss of some types of self-assessment questions because they are not supported by target VLEs. In addition, the deployment and support issues caused by the plug-in may restrict the adoption of WinEcon by some institutions. At the end of the day WebEcon is only partially WWW-based and a further project would be required to migrate from the plug-in to a fully JAVA, DHTML or XML system.   With hindsight, the decision taken to use the plug-in will probably result in further expenses in the future to complete the transition to the Web. Controversially, we feel that it may have been quicker and more cost effective to freeze the development of the ToolBook-based version of WinEcon and build the new version on a clean code base.

The data collected in the original WinEcon project clearly indicated that the major development effort and cost was in the establishment of courseware standards and the original authoring and development of the content materials. The development of the ToolBook delivery system was a minor element of the cost with the major expense being the establishing how the courseware was going to deliver the content. Given that the "how" is very well defined in WinEcon it would have been relatively easy to create a new version either using an existing VLE or by developing new code using a more open standard language platform than provided by ToolBook and the Neuron plug-in. This would have had the advantage of allowing the limitations of the hybrid user interface to be addressed at the same time.

Projects to migrate courseware products are not well documented nor are the effects on the overall development effort. This paper presents an overview of the project to migrate CAL to WWW-based delivery. Just as with the original WinEcon development project, qualitative and quantitative data collected during this project will be fully analysed with the aim of developing metric-based effort estimation tools which will be presented in future papers. However, the conclusion from this overview is that the decision to migrate rather than completely redevelop WinEcon has resulted in delays and a marginally less interactive finished product. This along with the need to migrate to fully WWW-based delivery in a future project is the "cost" of a decision not to redevelop 4 years ago.

References

[1]   HEFC, (1992). *TLTP - Phase I.*, Higher Education Funding Council.

[2]   HEFC, (1993). *TLTP - Phase II.*, Higher Education Funding Council.

[3]   Sloman, J., (1995). *WinEcon Software Review.* Economic Journal, 5, 429.

[4]   Marshall, I.M., Samson, W.B., Dugard, P.I and Sutherland, J.N. (1997). Multimedia Courseware Engineering – Lessons Learned form the Failure of Technology-Based Learning. In Bell, C., Bowden, M and Trott, A. *Implementing Flexible Learning – Aspects of Educational and Training Technology XXIX*. Kokan Page: London, 227-236.

[5]   HEFC, (1997). *TLTP - Phase III.*, Higher Education Funding Council.

[6]   Marshall, I.M., Price, S., Dugard, P.I, Hobbs, P. and Samson W.B. (1996). *Code-based Analysis of the Development Effort of a Large Scale Courseware Project*. Information and Software Technology. 39, 541-549.

[7]   Price, S. and P. Hobbs, (1994). *ToolBook 1.5.* The Economic Journal, 4, 425.

[8]   Price, S. and P. Hobbs, (1995). *Asymetric ToolBook 3.0 - Implications for Developers*, Centre for Computing in Economics: University of Bristol, Bristol.

[9]   Price, S., (1998). The Making of *CALnet*. Interact, ILRT University of Bristol, 15,

[10]   McDonough, D., Strivens, J. and Rada, R. (1994) Current development and use of computer-based teaching at the University of Liverpool. *Computers in Education*, 22, 4, 335-343.

Author(s)

Simon, Price, Mr
University of Bristol, Institute for Learning and Research Technology
University of Bristol, 8-10 Berkeley Square, Bristol BS8 1HH
simon.price@bristol.ac.uk

Marshall, Ian M., Dr
University of Abertay, School of Computer Science
Bell Street, Dundee DD1 1HG
i.marshall@abertay.ac.uk