Creating English Language Learning iOS Applications from Paper-based Texts

Gary V. Ireland* ・Joshua Rappeneker**

Abstract

This paper describes how a series of textbooks (The English Course) is being dismantled and recreated for use as applications (iApps) for Apple's iOS devices: iPhone, iPad and iPod Touch. The paper describes the concept behind the creation of the module format for these Apps, the difficulty in creating such modules, and also an explanation of the programming development used in their creation.

Introduction

The English Course is a series of English language textbooks that have been created over the past seven years. The series currently comprises, two listening and speaking books, one writing book, two reading books and a business English book that is being prepared for publication. The overriding principle behind the development of the series was to produce textbooks and complimentary materials that form a marriage (or balance) between using the communicative approach (CA) to language learning in the classroom and utilizing the best techniques and materials available to incorporate computer assisted language learning (CALL). The authors have always tried to stay up to date not only with developments in the CALL field, but with all technological developments in language learning methodology, and to incorporate this technology as extensively and as imaginatively as possible into the course.

During their research, the authors have monitored the movement by other publishers and materials creators away from traditional paper-based texts. This research confirms that through the development firstly of computer technology and then the introduction of smart phones and more recently, the release of tablet computers, for example, the iPad, that education – both the way that we learn, and the way that we teach – are now rapidly changing, and will be changed forever.

People who produce educational materials, from the humblest teacher, perhaps a mother

＊准教授／CALL
＊＊非常勤講師／CALL

teaching her own children, right up to the largest educational materials publishers are coming to terms with these changes. The authors of The English Course have always sought to be at the very forefront of all new technological developments in English language learning and teaching.

The new project

So what could authors and publishers now create from their existing materials and textbooks to move them into the future?

DRM

Digital Rights Management is a set of technologies used to control access to digital content and devices. In essence, DRM can act as a 'lock' on digital content, only allowing users with the corresponding key to access it. DRM is designed to protect copyright holders from the unauthorised copying and distribution of their works. It is important to note that no DRM is unbreakable and experienced users determined to circumvent its protections are likely to be able to do so.

ePub

ePub is perhaps the most common ebook format. It is available on a wide variety of devices including Apple's iOS (operating system) devices, Google's Android devices, as well as Windows and Mac OS X. ePub features a 'reflowable' format, which means that the text of an ePub can adapt to changes in text size, and typeface. Other formats, such as PDF, have static formats which do not allow for any changes to pagination, even when text size or typeface are changed.

The ePub standard allows for some rich content such as images, audio, and movies to be embedded into the text. The most current version of ePub, 3.0, may allow for greater interactivity using JavaScript, though the tools for creating such interactive content are still in the early phases of development. Finally, ePub works with DRM, which makes it an attractive option for many large content distributors.

Digital Download

Perhaps the simplest and most common method of digital distribution is via download from a central website. Users can purchase documents through an on-site store, and are then given links to download ebooks and other media. The media is usually protected by DRM which limits the user's

ability to share the content with others.

The major hurdles involved in this approach lie in the development of an online store, managing a large, expensive website for downloads, as well as a persistent user database. The complications of setting up and maintaining secure payment systems may very well outweigh the advantages of complete control over distribution. Even with such systems in place, users may be uncomfortable sharing their financial information with a new, untrusted site.

iBook

iBooks is an Apple ebook application produced by Apple. Authors can create an iBook using Apple's "iBooks Author" and publish it to the iBookstore. Books published using Author use an Apple extension to ePub that can only be viewed and sold on iOS devices.

Whilst the iBook format is limited to a smaller set of devices than ePub, distribution through Apple alleviates the complications of managing financial transactions and maintaining a website for downloads. Books produced using iBooks Author can also incorporate a limited amount of interaction, such as short quizzes and interactive graphs.

Online books

Online book libraries grant access to documents online through subscriptions. A user pays a regular fee and is able to view, search, and print a certain number of documents in a given time period. The documents are not downloaded to the user's computer, rather they remain hosted on the content provider's server. Generally, the documents on offer are simple PDFs with little or no interactive capabilities. The drawbacks to implementing this system oneself are the same as for a digital download platform.

Web-hosted applications

Instead of offering traditional books online, it is possible to write applications for the web which run equally well on mobile devices and personal computers. Modern day browsers are capable of presenting web applications almost as seamlessly as those running directly on the desktop. Access to these apps can be charged for on a subscription basis or as a flat rate.

Similarly to digital downloads and online books, the costs associated with maintaining the systems required to host these online applications may be higher than the benefits offered. Additionally, unlike applications written for a small set of devices (e.g. iOS devices), web applications are required to run on a very large number of operating systems, and in a large number of different browsers and browser versions.

Android devices

Android devices globally have a larger share of the smartphone and tablet market than iOS devices, and offer most of the advantages that developing for iOS does. There are a few drawbacks, however, to developing for the Android operating system.

The first problem a developer faces is platform fragmentation. Unlike Apple which is the sole manufacturer of its iOS devices, Google does not produce the majority of Android devices. Rather, it licenses the Android operating system to other companies, such as Samsung, who produce the hardware and integrate it with Google's OS. This means that a developer must create an application for potentially hundreds of different devices running different versions of the operating system. It is very difficult to test an application on every possible hardware/software permutation, and so software is produced that is not guaranteed to work on every device it is sold on.

Another issue for Android developers is the prospect of making less money than their iOS counterparts. The same application released on both the iOS and Android markets makes significantly less money on the Android. One report showed Android apps making 24 cents to every dollar the same iOS apps made. This is due in part, no doubt, to the different user cultures. Whereas the majority of iOS users have an Apple iTunes account with an associated credit card, a much smaller minority of Android users are similarly registered with the Android marketplace. The vast majority of Android apps are 'free' ad-supported applications.

After experimenting with several of the electronic formats explained above, particular ePub and iBook, the authors decided to pursue the creation of iOS iApps. One of the main reasons for making this choice was that the modules would be made compatible for all Apple iOS devices. This fulfils the aim of making the materials suitable for both self-study (on all iOS devices) and for classroom usage (particularly on the iPad). The other main consideration was the capability to make the material as interactive as possible.

It was decided to begin by creating prototype modules specifically for the iPhone. However,

though all of the digital audio and video media chosen to build the modules can be used across all devices, it must be remembered that a great deal of additional engineering is needed to recreate those modules to be usable for both the iPod Touch and also the iPad.

Creating applications for iOS devices

Philosophy rationale

One of the first and most critical considerations in creating iOS Apps from existing textbooks is the obvious difference in how these different materials are designed to be used. As with most modern-day language learning texts, The English Course series of listening and speaking books were primarily designed to be used in classrooms with other students and a teacher present. Therefore, the speaking exercises within are (predominantly) pair work and group work speaking tasks. Naturally, when considering utilizing the existing materials for iPhone/iPad use, we had to consider by whom, where and how the iAPPs would be used. We do believe that as more and more educational organizations come to use electronic materials, instead of paper-based texts, that these materials may be selected by teachers for classroom usage. In some of the more forward-thinking institutions, all freshman students are given an iPad on entry. There are also cases where a certain number of these tablets are made available for particular classes. However, at this time, we assumed that many learners will be using the material for self-study, and therefore, alone. When we understand this basic principle, it is obvious to see that simply repacking the existing material in its original format would not necessarily be the most appropriate or suitable for the iOS devices.

Thus, though we are able to reuse much of the audio and video material that has already been recorded for the two speaking books, a great deal of rewriting of new materials and a completely new design for how to make each unit suitable for use by individual users on an iOS device was still necessary. As we were planning to deconstruct textbook units and rebuild smaller chunks of target study areas, we decided to call these new items modules.

Designing the new module layout

Designing the new module layout and its exercises required many hours of discussion and experimentation. Each module also requires several days of rewriting and editing in order to make the material suitable. Additional audio and video recordings were also added in order to bolster the self-study usability. In addition, programming the actual engines for each different activity in the module also takes an enormous amount of time and experimentation. To date, it has taken two years to complete the design layout and programming for a prototype module and to have all the necessary texts, audio and video clips ready for the first five modules. The prototype modules are now usable and have been tested on the authors' own iOS devices.

In designing these new iApp modules, three main considerations were taken into account.

1. Each individual module must be able to stand alone as a complete and pedagogically suitable unit.
2. The modules would remain designed to teach listening and speaking skills, but must be suitable for self-study.
3. The modules should incorporate a variety of different tasks and activities and should be as interactive as possible. In addition these tasks should be designed to be as interesting and as much fun as possible.

The authors feel that perfecting point 3 on this list would be the key to whether or not this project will become successful or not.

In trying to make the modules appealing to learners, we had to consider how to use the technology to make each module as interesting as possible. One of the first decisions made was to include original video clips in each module. Even though using video files requires much more memory space than audio files, we felt that being able to actually see the conversation interactions is far more beneficial than simply listening to distended voices only. This principle was also applied to the creation of the original textbook DVDs.

In deciding to make each module as interactive as could be possibly engineered, we knew that we would be attempting to create materials that we have never seen available elsewhere, and that we would have to push the programming boundaries. By interactive, we mean that students need to undertake a series of different tasks (actions) as they move through each module. One of the main points was to avoid having students just reading and listening to the materials. Having looked at many of the English language learning applications available at this time, for example, *Conversation English*, by Red River Press Inc, we are quite sure that there is nothing as complicated or interactive on the market at this time.

In developing this interactivity, two main points were focused upon.

1) To imagine and create as many different activities as possible.
2) Where possible, to make the activities into a kind of game or challenge format.

Users of Apple iOS devices such as the iPad and iPhone/iPod touch, particularly younger users, are very familiar with playing games on their devices. Just as with usage in the classroom, games are fun and encourage learning. The same principle was adhered to for the DVDs for the original textbooks in the form of interactive quizzes, which were very popular with students, and

with a variety of video-based challenges and games in the Finale section of the discs. Naturally, adding interesting and fun quizzes and games encourages learners to redo the activities many times.

Creating the modules

The materials used for creating these iApp modules were to be taken from two different listening and speaking based English language textbooks; *The* English Course – Speaking Book 1, and *The* English Course – Speaking Starter.

The English Course – Speaking Book 1 is the first book of the series and is aimed at lower-intermediate to intermediate level students.

The English Course – Speaking Starter is aimed at beginner to lower-intermediate level students.

Each book in *The* English Course series consists of the following:

I) A textbook for classroom usage, which contains numerous listening and speaking activities, as well as structure and vocabulary material.

II) A set of DVDs for classroom and self-study usage, which contains approximately 50 audio and video clips including conversations, pronunciation tasks, quizzes and tests.

III) An online self-study centre for out-of-classroom usage, which contains additional structure, vocabulary and listening exercises, as well as quizzes and tests.

Unlike many textbooks available, the wealth of digital material already created makes *The* English Course series ideal for recreating and adapting to the iOS App platform.

The initial issue to be resolved was to select parts of both of the original textbooks from which video and audio content could be taken to form the base of stand-alone self-study modules. Both of the existing textbooks are functional/situational based, and therefore the modules selected would, where possible, represent the language needed for a particular language function or situation. It was decided to create the initial prototype modules from materials from *The* English Course – Speaking Book 1.

The English Course – Speaking Book 1 consists of only eight units. However, these units are divided into two distinct parts, each of which represents a separate, but connected language function. Each part is then subdivided and contains two separate video conversations. Thus, each unit consists of four interconnected video conversations. To explain how this is designed, we can look at Unit 1 of the book for an example. The unit title is introductions. The first part of the unit deals with the language needed for self-introductions. Video conversation 1 is a formal self-introduction and video conversation 2 is an informal self-introduction. The second part of the unit

deals with the language needed for introducing other people to each other. Video conversation 3 is a formal third-party introduction, and video conversation 4 is a third-party informal introduction. Thus, the units are designed to deal with two connected functions using four interconnected but different video conversations. These conversations are also supported by additional audio and video conversations and a series of speaking and role-playing exercises to back up the target language content.

Though, in this case, each of these individual video conversations can conceivably be recreated into stand-alone modules for self-study, in other units, the materials did not lend themselves so easily for recreation into this format. In addition, due to the size of the video files, only one video clip per module is practical. In some cases only one or two of the original video conversations from a particular unit was deemed suitable for use.

The design of the iOS App

After all of the discussion and experimentation, the following format was created.

Activity 1 – Video	(watch video of target language)
Activity 2 – Video Task	(Comprehension activity)
Activity 3 – Language and Vocabulary	(vocabulary/structure explanations)
Activity 4 – Audio Game	(listening game activity)
Activity 5 – Pronunciation Practice	(pronunciation practice)
Activity 6 – Questions and Answers	(listening activity)
Activity 7 – Dictionary and Phrasebook	(vocabulary/structure reference and explanation)
Activity 8 –Test	(end-of-module test)

The contents page of the application opens like this.

Plate 1

Activity 1 – Video conversation

Students begin the module by watching the video clip of the target language conversation. After a great deal of trial and error in programming, we were able to give this activity some of the same flexibility and functionality as the original DVD version. Therefore, students can stop, pause, play forward and rewind as much as they wish. They can watch the clip as many times as they like and even press a button to add English subtitles if they want to. For those who have apple TV, this conversation can easily be played on a TV (Plate 2).

Plate 2

When students have finished watching the video clip, they can move to the Video Task.

Activity 2 – Video Task

This activity is a comprehension exercise consisting of 10 statements related to the video conversation. Students decide if the statements are correct ☑ or incorrect ☒ (Plate 4). The box begins in an empty state ☐. When students click the box once, the correct box ☑ appears. A second click changes this to the incorrect box ☒. One more click returns the box to the empty state ☐. When students are happy with their answers, they press the 'Answer' button to see their result. Rather than simply have the correct or incorrect answers appear on screen, we decided to add feedback for students to consider. After pressing the 'Answer' button, students will see their incorrect answers on the next screen. However, they will also see an audio image icon, which allows them to click and listen to each relevant line in the conversation (Plate 5). Thus, allowing students to hear where they were wrong. Adding this feedback feature meant undertaking extensive additional programming.

Plate 3

Plate 4

Plate 5

When students have completed the tasks, they can move to the Language and Vocabulary task.

Activity 3 – Language and Vocabulary

When students open this section, they will see the entire video conversation transcribed. Vocabulary and expressions that students are expected to have difficulty with appear in red ink (Plate 6). Simply touching any of these words or expressions causes a drop down box to appear on the screen with the explanation for the respective word or expression (Plate 7). From a programming point of view, this engine was the simplest to design and is similar to those used in the dictionary type applications that are already widely available.

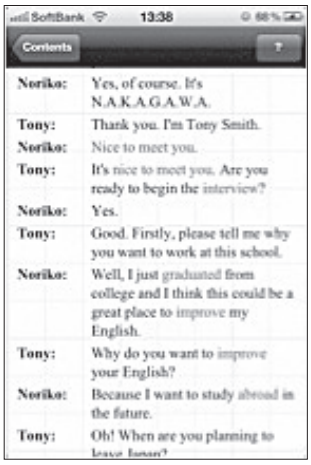

Plate 6

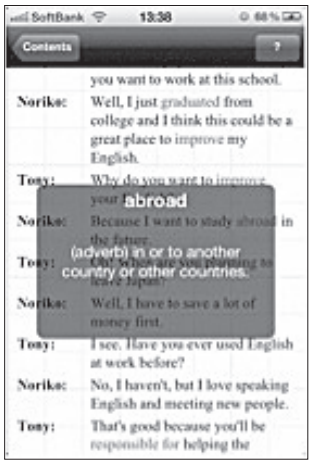

Plate 7

When students have checked the vocabulary and expressions in the conversation, they can move to the Audio Game task.

Activity 4 – Audio Game

As stated above, one of the main criteria targeted in the design of these Apps was to make them interesting and fun to use. This activity poses a very interesting and fun challenge, and is, we believe, unique at this time in language learning Apps. The idea was taken from an activity that we had already created for use on the DVD of the first textbook.

When students open this activity, they will see a set of square symbols on the screen (Plate 8). These appear in two different colours, which represent the two different speakers in the conversation. Each box (or button) represents one line of the conversation. Students must listen to the different lines of the conversation and then build a completed conversation by dragging the boxes into the correct place on the screen grid (Plate 9). There are three levels of difficulty (Easy, Medium, Hard), which can easily be selected from the 'Difficulty' menu (Plate 10). In the 'Easy' mode, several of the audio clips are in the correct location, however, in the 'Hard' mode, none of them are in the correct location. If students drag the wrong clip into a location, it will simply spring back to where it came from. This activity can be done numerous times, as the grid will recreate itself into a different pattern each time.

Plate 8

Plate 9

Plate 10

When students have completed the game, they can move to the Pronunciation Practice task.

Activity 5 – Pronunciation Practice

This activity is another idea which we believe to be unique in language learning applications available for the iOS platform to date. As many students will be studying alone and do not receive feedback from either a teacher or other students, we felt it important for them to be able to analyze and measure the quality of their pronunciation and intonation in some way. Therefore, each line of the conversation is broken down into separate audio clips (Plate 11). Students are able to select and listen to each item as often as they want and then record their voice into the application as they repeat each line (Plate 12). Students press ‘play’ to listen to each line of the dialogue. They then press the microphone image to start and stop the recording of their voices. There are two different ways that they can listen and compare their efforts with the original voices. They can be heard either simultaneously with one voice in each ear (from headphones), or one after the other. This is not only very useful feedback for students, but also a unique and fun activity for them to enjoy. The idea was taken from a similar idea that was developed for the DVD of the second book in The English Course series. Once again, to create something as sophisticated as this took a great deal of original programming.

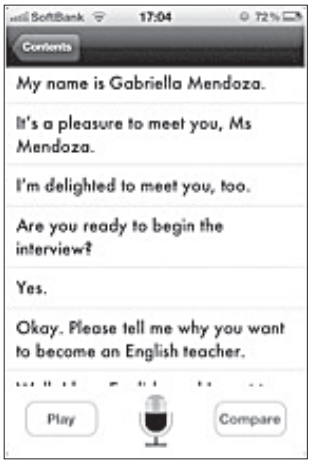

Plate 11

Plate 12

When students have completed the pronunciation practice, they can move to the Questions and Answers task.

Activity 6 – Questions and Answers

The principle for this audio activity was taken from an idea developed for the student online self-study area of the second book of *The* English Course series. This is another fun listening activity that students can enjoy trying several times. In this activity, students first press the question button on the left side of the screen and listen to a question (Plate 13). They then listen to the three different possible answers under buttons A, B, and C. They leave the button lit for the answer they believe to be correct (Plate 14). Students are then given feedback on which of their answers are correct (Plate 15). To make the activity more difficult and capable of being attempted several times, the answers are automatically shuffled after each attempt.

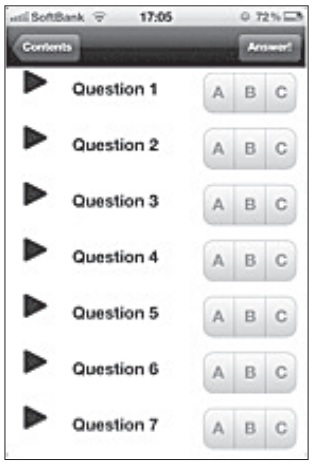

Plate 13

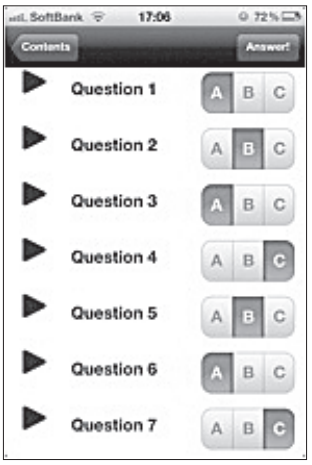

Plate14

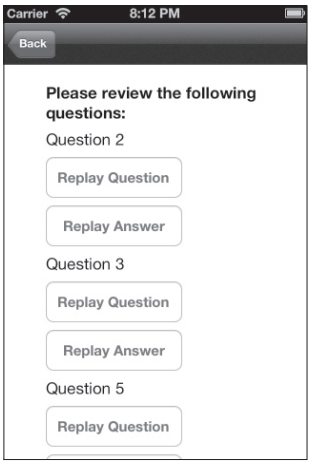

Plate 15

When students have completed the listening activity, they can move to the Dictionary and Phrasebook task.

Activity 7 – Dictionary and Phrasebook

This section of the module has been through several changes and updates since its original inception. In the original textbook format, each unit of both books had two Language Focus sections. These sections introduced the target vocabulary and structure for the unit. In addition, each unit had a vocabulary section with simple English explanations at the back of the book. In the end, we decided to combine both of these aspects of the textbook into the iApp format. In the text areas of each module, dropdown boxes appear when students click on a word or phrase that is highlighted. All of these explanations of words and phrases are repeated in the "Dictionary and Phrasebook" section of the module (Plates 16 & 17). In addition, one or two Language Focus boxes have been added to each module, depending on their necessity. Generally, these will coincide with those connected to the target language of the original textbook unit. In module 1 (which teaches students about self-introductions), for example, the first Language Focus explains different expressions that are used for self-introductions (Plate 18), and the second Language Focus deals with the different ways in which we talk about names in English (Plate 19). As students move through the course, this section can be used as a reference source and also a library of language studied

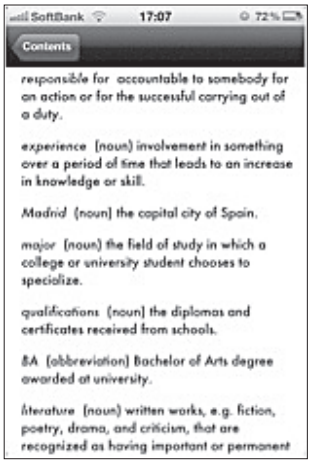

Plate 16

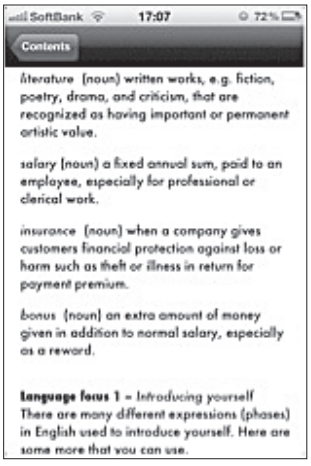

Plate 17

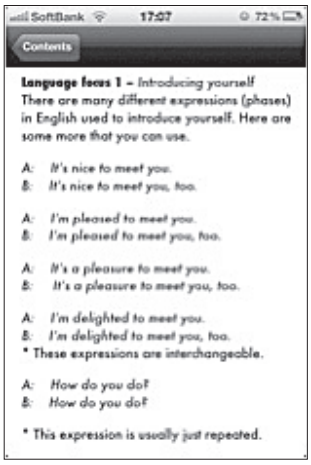

Plate 18

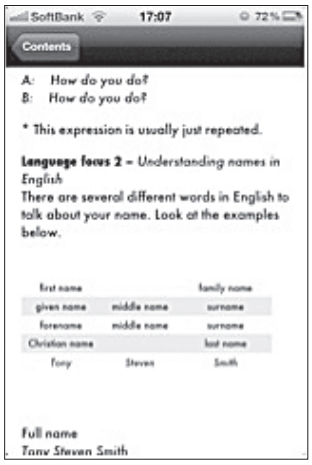

Plate 19

When students have checked the dictionary and phrasebook section, they can move to the end-of-module Test activity.

Activity 8 – Test

Each module ends with a 20-question test, which reviews all of the important vocabulary and language structure learned in the module (Plates 20 & 21). Having originally planned to have only 20 items in total, we have now decided to actually create 30 items per unit. The App will automatically shuffle a different and random 20 items to be selected each time the test is undertaken. At this time, the items are formatted to be correct/incorrect answers. Students decide if the statements or questions are grammatically correct ☑ or incorrect ☒ (Plate 4). The box begins in an empty state ☐. When students click the box once, the correct box ☑ appears. A second click changes this to the incorrect box ☒. One more click returns the box to the empty state ☐. When students are happy with their answers, they press the 'Answer' button to see their result. However, we may introduce a different variety of answers, for example, multiple-choice. In addition, we may be able to create a mixed format test.

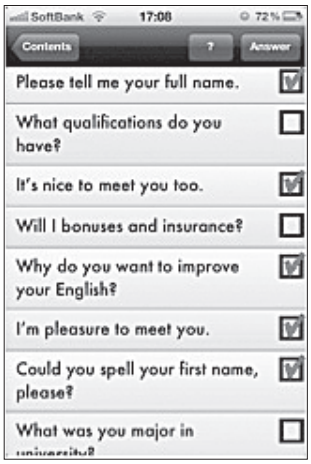

Plate 20

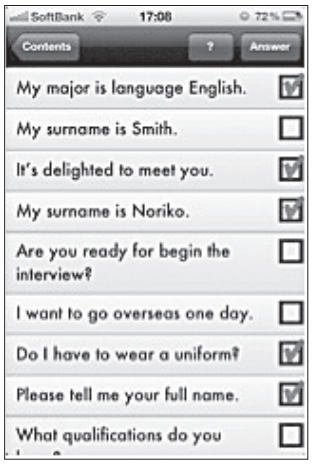

Plate 21

The technical aspects of the development of the application

After designing a prototype module from the existing textbook format, the next thing to consider was how the user was expected to interact with each of the different aspects of that content. And then to develop this into an application that would run on iPhone, iPad, or iPod touch devices, collectively known as iOS devices. Whilst the application is not yet finished (indeed, the introduction of a new iPhone with a different screen aspect ratio will mean some significant redevelopment), it has progressed to the stage of a usable beta, meaning that the final version will very much resemble our current stage of development.

The application has three main modes of interaction with the user. The first, 'information mode', presents the user with information such as a video clip or a text-based explanation of a grammar point. In this mode the application is most like its textbook forebear, there is very little – if any – user input. The second, 'quiz mode', presents the user with questions pertaining to previously presented information. In this mode, the user is more actively engaged with the application, though it does to a large extent resemble a textbook quiz. The defining difference is that the application provides the user with immediate feedback, along with links to the correct information. The third, least 'textbook-like' mode, 'game mode', requires constant interaction between the user and the application. An example of game mode would be section 4 – "Audio game" – where a user drags various randomized parts of a conversation into the correct order with the application giving constant real-time feedback. Unsurprisingly, 'information mode' provided the fewest development challenges, and 'game mode' the most.

The difficulties of developing an iOS textbook application can be divided into five broad categories:

- Limitations and changes in the provided application programming interface (API)
- Balancing brevity and explicitness in the user interface (UI)
- Developing a suitably flexible data storage mechanism
- Requirements elicitation, especially in the context of a book-to-application adaptation
- General programming difficulties

As this paper is written with a non-technical audience in mind, technical details will be kept to a minimum, and sufficiently explained when necessity demands they be included.

Limitations and changes in the API

An API is the set of specifications and libraries provided to allow different parts of software to communicate with each other. The API that Apple has provided for iOS devices allows for relatively rapid and straightforward application development. It is, however, not without its shortcomings.

The API, at its best, allows the programmer to ignore low-level tasks, and deal with higher level abstractions in the development of his or her applications. The Gesture Recognizer in iOS, for example, allows programmers to deal with a user's touch-based input as a 'tap', or a 'swipe' instead of a large and rapidly updating set of coordinates. This kind of high-level abstraction reduces bugs by reducing the complexity of the code produced by the programmer, and makes the code base ultimately more maintainable. The programmer is still free to access the lower, more flexible levels of the API if he/she wishes to do something not afforded by the higher level interface.

The problem with this system arises when the high level APIs are too inflexible for a given task, and the low level APIs are too complex to consider. This issue was encountered at two points during development.

The first issue was in the presentation of the video to the user – perhaps the most fundamental task of the entire application. The initial idea for the video presentation was to have the video presented to the user with several options. The user could choose to turn subtitles on or off, to view the subtitles for a particular speaker, or to select to hear either of the speakers individually. Unfortunately, the high level APIs did not allow for any manipulation of either audio streams or subtitles, and the low level interfaces were simply too complex to use given our development

time-frame. Instead, it was decided that a single audio track and subtitle track would be included. Therefore, users can watch the video with either subtitles (in English) on or off, but cannot make any other adjustments except, of course, to pause, rewind, fast-forward, etc, the video. Users with Apple TV capability can also watch the video on their TVs or monitors.

The second, more surprising deficiency encountered in the API was in the display of text. Simple text is very easy to display in iOS through a group of high-level 'classes', such as UITextView (read 'U.I. Text View' or 'Yui Text View'). UITextView allows the programmer to display scrollable text on multiple lines in an area on the screen. This seemed ideal to present the "Language and vocabulary" and "Dictionary and phrasebook" sections of the program. However, none of these high-level text display classes had the ability to display text in more than one style. This meant the choice of a single font, size, and colour for the entire block of text. Clearly in the case of "Language and vocabulary", with its highlighted touchable vocabulary, this was unsustainable.

Early attempts to fix this involved the use of the low-level "Core Text" libraries, which provide the programmer with multi-styled text, at the cost of significantly greater complexity. A working version of the Core Text solution was tested. However it required too much CPU (Central Processing Unit) power to implement on older devices and was simply too complex to maintain. In its place, UIWebView, a "class" (an abstract entity that describes certain behaviours and data structures that can be instantiated in its 'class objects') that displays web content, was used. This solution meant that text could be styled using HTML (HyperTest Markup Language) and CSS (Cascading Style Sheets) in the same manner in which websites are styled, and each touch to the webpage's links could be intercepted by the application and dealt with accordingly. This was certainly not an ideal solution because on slower devices the screen is blank for a fraction of a second as the webpage loads. However, it was the only reasonable solution.

At the time of writing this paper, Apple has just released the latest version of its operating system, iOS 6.0. Briefly perusing the new API documents has revealed the existence of 'attributed strings' available in UITextView. These text-strings seem to offer a middle ground between the simplicity of single-styled text and the complexity of Core Text - they may solve the issues related to the use of UIWebView. Unfortunately, the subtitles now no longer display in the video presentation - which will mean a further reworking of that particular section.

Balancing brevity and explicitness in the user interface (UI)

The iPhone display is small. It measures only 8.9cm across diagonally. With the addition of the

status bar at the top of the display and any other toolbars or navigational bars, the space available to deliver content to the user is even smaller. This lack of screen 'real estate' drives the design towards small controls (buttons, switches, etc.) and minimalistic interfaces.

However, small controls present their own problems. Any control smaller than 44 by 44 pixels can be difficult to interact with, leading to missed touches, and a frustrated user. Even with smaller controls, when presenting a question to the user with a checkbox ☑ (true) and ☒ (false) for each possible answer, the controls dominate the screen and significantly reduce the space available to the content. The solution to this problem was to utilise multi-function controls.

Instead of one checkbox for true ☑ and one for false ☒, a single checkbox ☐ could be used. One single tap changes the checkbox to true ☑, the next tap to false ☒, and the next tap resets the checkbox to unanswered ☐. These multifunction checkboxes allow for individual controls to be large enough to use, whilst still increasing the amount of screen real estate devoted to content.

The main drawback of these controls, and others like them is that they make the interface slightly less clear to new users. Throughout the development process we have discussed the balance between a clean, content focused interface, and an interface that is intuitively understandable to new users. The final solution (still in development) is the use of transparent help overlays, which guide the user throughout the interface graphically.

Developing a suitably flexible data storage mechanism

One of the ongoing difficulties with the project is the question of how to store the actual content of the application. As the long-term goal of the project is to electronically publish 'modules' of content semi-regularly, it is important that the content size be kept to a minimum. Pushing very large amounts of data to mobile users is impractical, especially if those users are in areas with lower speed Internet connections. In addition to the networking concerns, iOS devices have a limited amount of storage space. With this in mind, it was decided that each module 'bundle' be kept to a minimum size, using video and audio compression, along with the compression of all text data.

In creating each bundle, a compromise must also be made between the human and machine readability of the contents. Some of the data, such as the definitions of new vocabulary items, are stored in XML format. An example item from the definition file looks like this:

<key>nice to meet you</key>
<string>(expression) a greeting used when meeting someone for the first time.</string>

XML is a popular choice for data files as it is editable by hand in any text editor, relatively human readable, and very easy to parse in software. The problem, as with all files meant to be processed by computer is that a simple human mistake (such as a missing '>') often makes the entire file unreadable. Considering the large amount of written content that needs to be added to each bundle, hand editing XML files did not seem the most viable option. With this in mind, a graphical tool was developed to let the computer take care of the syntax, and the user to simply cut and paste the content (plate 22).

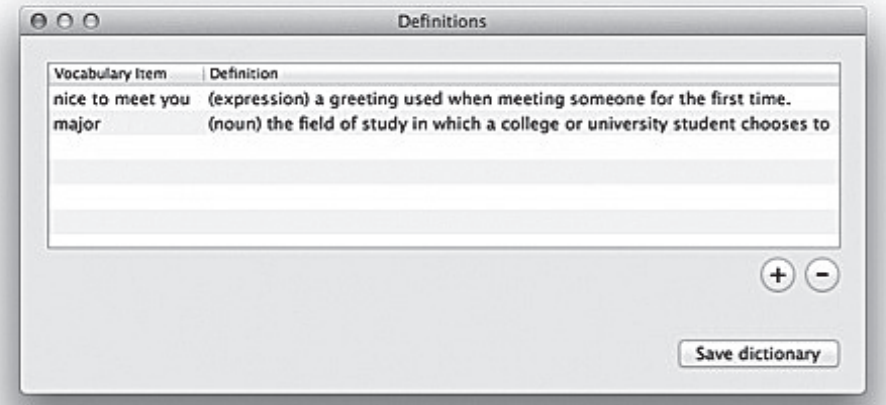

Plate 22

Above is the 'definitions' screen in the graphical bundle editor.

Using this tool, we are able to cut and paste and drag and drop all the textual content and media into a new bundle, save it, and have the details of compression and syntax dealt with programmatically.

Requirements elicitation, especially in the context of a book-to-application adaptation

Requirements elicitation, and requirements engineering is the process by which we begin to tie the design of the target module to the development of an actual software product. The engineering process involved constant experimentation and trial and error. This is where it becomes clear which of the module requirements are reasonable, and which may be impossible. It is also important to recognise how some of the requirements might interact within the target system and to make

adjustments or to come up with alternative ideas.

In this particular project, the requirements were developed and documented before the programming was undertaken. This made the initial stage of elicitation quite straightforward, and more a matter of clarification than elicitation.

As The English Course already had a great deal of content, both textual and electronic media, this meant a very solid, though perhaps somewhat book-like and linear application design. One of the first mistakes made in the initial engineering was to understate the non-linear, interactive potential of the application up-front. As it is, the first module of the program is designed to be used by the user in a step-by-step fashion: first viewing the video, then answering related questions, then learning vocabulary, structure, et cetera. As an introductory module, this is a perfectly reasonable and relatively 'gentle' lead into the application. However, with subsequent modules, we will attempt to make the user experience less linear by adding randomisation and mixed-media sections to the application.

General programming difficulties

Adapting the user interface to the iPad

The project was first conceived of and developed with the iPhone in mind. This has made several parts of the iPad adaptation complicated and difficult to programme. Adapting an iPhone application to the iPad is not simply a matter of enlarging all the content and user interface elements to fit the new, larger screen. As the iPad is many times larger than the iPhone screen, this can lead to the interface looking like an absurd 'blow up' version of the iPhone's.

Instead, the interface needs to be redesigned to take advantage of the different user interface elements available on the iPad. The iPad's larger screen can present the user with much more information at one time than the iPhone without being visually confusing.

Plate 23

Above is an enlarged interface on the iPad. Most of the screen real estate is wasted white space.

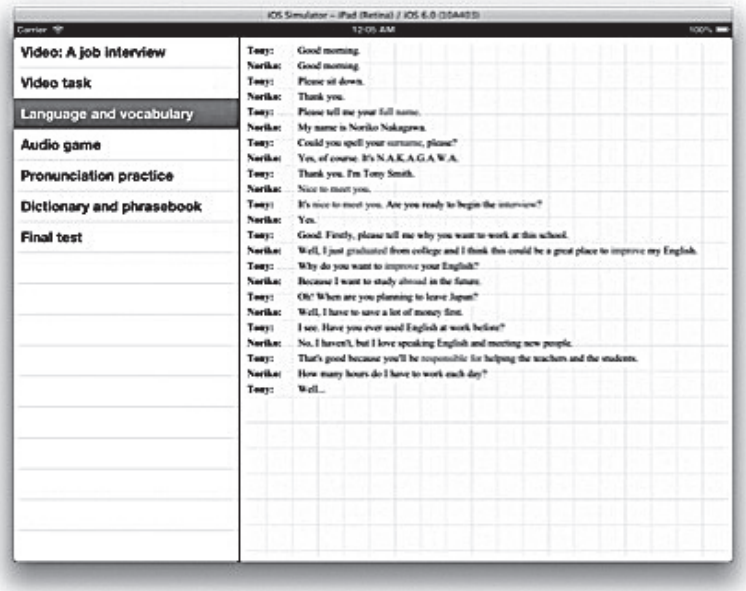

Plate 24

Above is an example of an interface which makes use of the greater screen real estate of the iPad.

Of course, modifications to the interface also require modifications to the underlying code. Wherever possible, the interface design of both the iPhone and iPad versions of applications should be done at the same time to avoid possible conflicts further into the development cycle.

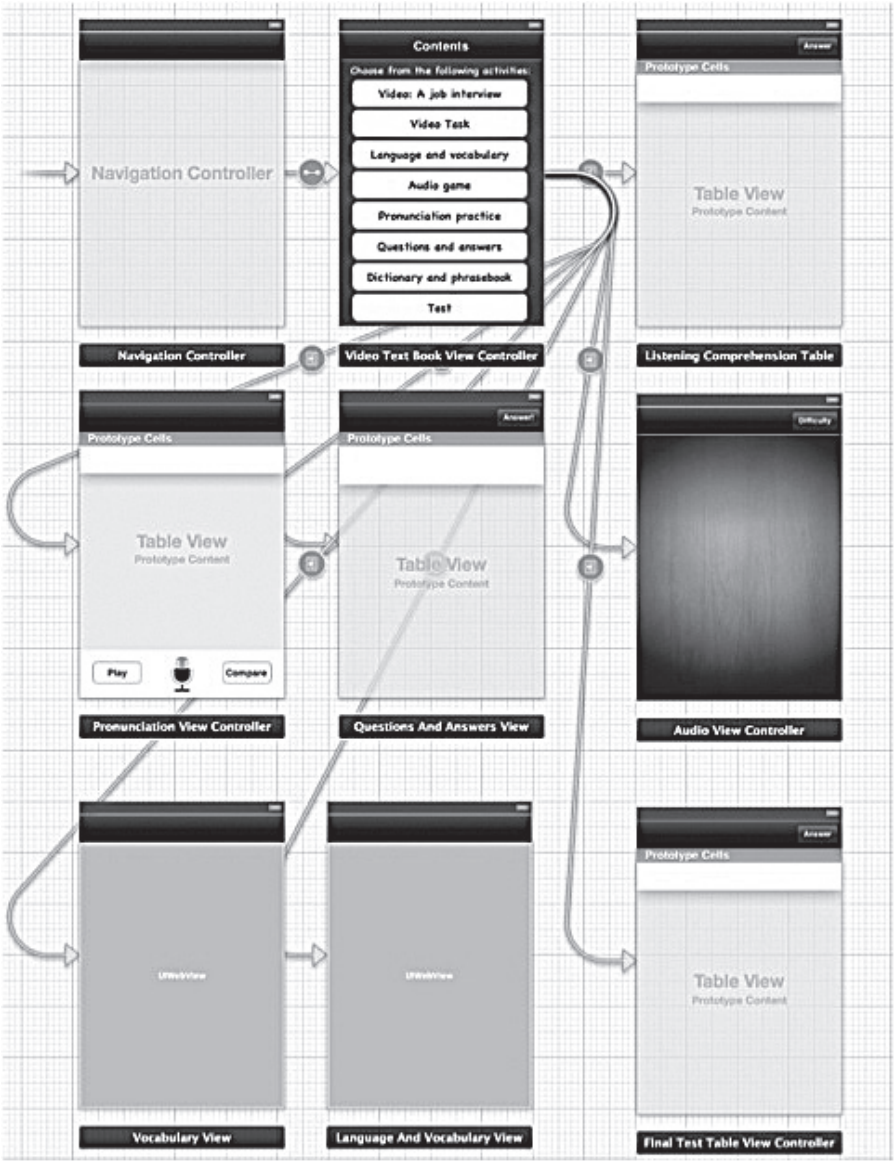

Plate 25

Above is an overview of the user interface design. Notice the two UIWebViews in the bottom left corner. Each of these rectangles represent a ‘screen’ the user interacts with. Each different screen is controlled by its own logic. The ‘Audio View Controller’ screen shows the background for the game, as well as the difficulty button.

Plate 26

Above is a listing of some of the files that make up the application. In the “Audio Game” directory is the file AudioViewController.m, which is responsible in part for managing the user’s interaction with the audio game.

```
// TODO: add circles to answerRects for guidelines
- (void)viewDidLoad
{
    /* set difficulty to easy = 0, medium = 1, hard = 2 */
    // FIXME: use enum

    [self.view setAutoresizesSubviews:NO];

    /* set up view */
    [super viewDidLoad];
    self.view.backgroundColor = [UIColor whiteColor];

    /* initialize variables*/
    clipViews = [NSMutableArray new];
    gestureRecognizers = [NSMutableArray new];
    answerRects = [NSMutableArray new];
    questionRects = [NSMutableArray new];
    playing = NO;
    icons = [NSMutableArray new];

    NSURL *imageListURL = [[NSBundle mainBundle] URLForResource:@"audioTaskIcons" withExtension:@"plist"];
    icons = [NSMutableArray arrayWithContentsOfURL:imageListURL];


    [self loadIcons];

    /* For the moment, two fifths of the screen will be for answers, two fifths for questions, and one fifth buffer zone */
    yOffset = (self.view.bounds.size.height-20)/5;
    answerSpace = CGRectMake(0, 0, self.view.bounds.size.width, yOffset*2);
    questionSpace = CGRectMake(0, yOffset*3, self.view.bounds.size.width, yOffset*2);


    /* I'm too tired to do this properly...brute force for now */
    /* work out about what size our squares should be */
    numberOfViews = 16, viewsPerRow = 2;
    width = questionSpace.size.width/viewsPerRow;
    while (ceil(numberOfViews/viewsPerRow)*width > questionSpace.size.height) {
        width = questionSpace.size.width/++viewsPerRow;
    }

    [self createQuestionSquares];
    [self createAnswerSquares];

    /* randomize squares */
    [self randomizeSquares];

    /* here we should move some of the squares to the answered position, depending on the difficulty */
    /* initial difficulty of 0->easy */
    [self setDifficulty:0];
    UIBarButtonItem *difficultyBarButton = self.navigationItem.rightBarButtonItem;
    UIBarButtonItem *helpBarButton = [[UIBarButtonItem alloc] initWithTitle:@"?" style:UIBarButtonItemStylePlain target:self action:@selector(helpTapped)];
    NSArray *barButtons = [NSArray arrayWithObjects:difficultyBarButton, helpBarButton, nil];
    self.navigationItem.rightBarButtonItems = barButtons;

    /* present help view if the user hasn't asked for it to be forever dismissed */
    NSUserDefaults *userDefaults = [NSUserDefaults standardUserDefaults];
    if ![[userDefaults boolForKey:[NSString stringWithFormat:@"Don't Show %@ Help",kTaskName]]]
    {
        [self presentHelpView];
    }
}
```

Plate 27

Above is a ‘method’ in AudioViewController.m which runs once the game screen has loaded. The text preceded by ‘//’, or surrounded by ‘/*’ and ‘*/’ represents developer comments. This method represents about 10% of the AudioViewController.m file, which is one of many files in the project.

Many of the technical difficulties encountered in creating an application, as outlined above, are shared by any iOS application. However, there are problems particular to the adaptation of a textbook. They include how to elegantly present the user with a significant amount of textual content, making the interface clear enough to be used in English by a second-language speaker, and breaking out of the linear “book-like” mentality, which inevitably comes with the adaptation process. Tackling these problems, with special care to avoid the pitfalls of the latter, will help in the successful development of a textbook adaptation.

Conclusion

As creators of CALL related English Language learning textbooks and materials, the authors of The English Course series set out initially to investigate how such language learning course materials could be developed into purely technology-based formats. The authors considered many of the possible formats for the recreation of their existing materials, and experimented with several, including the ePub and iBook formats. The authors feel that both of these formats would be much easier formats to work with in creating materials, but that in both cases, they lack the amount of interactivity capability that we were looking for at this time. Having said that, it is easy to see that these platforms will be developed and that programming more complex materials will soon be possible. During the research it was concluded that attempting to create iOS applications for Apple devices would be the best project to attempt in recreating paper-based materials into digital, electronic only formats. There were two reasons for this choice. Firstly, the modules would be made compatible for all Apple iOS devices (iPhone, iPod Touch and iPad). This fulfils the aim of making the materials suitable for both self-study (on all devices) and for classroom usage (particularly on the iPad). The other main consideration was the capability to make the material as interactive as possible.

The project has now been underway for two years and at the present time has reached the stage where a prototype module for the iPhone platform has been completed and tested successfully. To reach this stage of development two main challenges have had to be constantly addressed, adjusted and improved upon. The first task was to design interesting, interactive and pedagogically suitable modules from the existing materials, and the second issue was to be able to bring this concept into reality through the creation of appropriate programming engines. The researchers have spent countless hours addressing both of these issues, and as with any new idea, there have been many changes and improvements to the original design of both aspects.

The authors strongly believe that the age of paper-based learning materials, not only for language learning, is rapidly coming to end and that in the near future most or all educational materials will be designed for one or several of the electronic formats described. This project has proved that it is possible to take components of existing materials and to produce appropriate and exciting new materials for the iOS device platform. It has also shown how complicated a task this is and how time consuming it must inevitably become. Another important consideration that has become obvious through this project is the future need for educational materials makers to work hand in hand with computer programming companies. The programming for the project is of an extremely high level and could not be undertaken by someone without a programming background. This is certainly a trend that will develop amongst all publishers of educational materials. The authors of this paper and The English Course series have created everything connected to this

project completely by themselves; all texts, audio/video media, the design of the module structure and all of the programming has been achieved without any external help or service. It is clear that not many independent materials producers or teachers would have the necessary knowledge or skills sets to be able to undertake such a project.

The authors believe that the module created has no equivalent available at this time. However, we also believe that, as the course becomes well known, aspects of the design will be copied and that other materials developers will attempt to create something as complex and interesting. However, it will take a great deal of time, energy and money to do so.

References

http://www.ipadinschools.com/
http://www.ipadinschools.com/157/ipad-apps-for-elementary-school/
http://www.appcelerator.com/thinkmobile/surveys
http://www.digitaltrends.com/mobile/why-many-developers-look-to-ios-first-money/
http://blog.flurry.com/bid/79061/App-Developers-Bet-on-iOS-over-Android-this-Holiday-Season
The Impact of the iPad and iPhone on Education, by Gary V. Ireland and Maxim Woollerton, Bunkyo Journal 2010
https://www.eff.org/issues/drm “DRM” (accurate, but strongly anti-drm)
http://dashes.com/anil/stuff/doctorow-drm-ms.html “Microsoft Research DRM talk” (describes why DRM is breakable)
http://news.bbc.co.uk/2/hi/technology/6337781.stm “Q&A: What is DRM?”
http://idpf.org/epub/30/spec/epub30-contentdocs.html “EPUB Content Documents 3.0”
http://www.myce.com/news/amazon-kindle-to-invade-all-target-retail-stores-30431/
Ken Auletta, Annals of Communications, “Paper Trail,” The New Yorker, June 25, 2012, pp. 36-41
http://opensignal.com/reports/fragmentation.php “Android Fragmentation Visualized”
http://blog.flurry.com/bid/85911/App-Developers-Signal-Apple-Allegiance-Ahead-of-WWDC-and-Google-I-O
http://developer.apple.com/library/ios/#documentation/uikit/reference/UIWebView_Class/Reference/Reference.html “UIWebView Class Reference”

（2012.10.1 受稿 , 2012.11.30 受理）