Personalisation and Interactivity in an E-learning Tablet Application

Abstract –

Context/Background
E-learning environments are in common use at educational institutions to distribute course materials to students. These students may have different levels of background knowledge and different learning styles. Taking this into account, there have been many studies investigating adaptive e-learning methods that attempt to personalise course materials to meet the needs of individual students. However, many of these studies have required course instructors to have certain programming skills or have involved building multiple course structures.

Aims
This project aims to determine whether the use personalisation and interactivity improve the student learning experience. In order for this personalisation to be of use in the real world, the construction of courses needs to be simple; a process that does not require programming skills and is not time consuming. Therefore, a secondary aim of this project is to use and develop techniques that enable a simple course construction process for instructors to create adaptive e-learning courses.

Method
An e-learning system will be designed and developed so that adaptive e-learning courses can be created by instructors and consumed by students. The course viewing application will have an option to turn off and turn on the interactive elements and personalisation, so that the impact of these techniques can be measured with metrics and in a user study. The usability of the course authoring software will be assessed using metrics and a small user study.

Proposed Solution
A web-based application will be created to enable course instructors to build their adaptive e-learning courses and distribute them to students. An Android tablet application will be developed for students to consume the personalized courses, taking advantage of the hardware features of tablets to provide an interactive experience. An appropriate file format will be produced to store the course materials created by the course authoring application, so that the course viewing application can download a course and personalise it based on a user profile.

Keywords – Personalisation, User Profiling, Adaptive e-Learning, Interactivity, Tablet

I. Introduction

This project is investigating how e-learning can be improved by personalising the learning experience to the needs of individual students. It will also look at how tablet hardware can be utilised to improve the interactiveness of these personalised course materials. The project can be split into two main areas; how instructors can create course materials in a way that can be personalised, and how that content is consumed by end users in a personalised and interactive manner.

A. Project Domain

The use of e-learning and the ways it can be utilised to enhance student learning is becoming increasingly important. E-learning is a broad term, used to describe a technology supported approach to learning, where the learning activities that students undertake are assisted by communication and multimedia technology (Li, et al., 2008).

E-learning is widely used in educational institutions across the world. Learning Management Systems (LMS) enable students to access course materials and complete course activities from any location, and at any time, making distance learning possible. Popular examples of LMS's include Blackboard, Moodle and WebCT. These systems provide facilities for course instructors to create upload course materials and manage who can see materials and how long for. Additionally many of these systems provide online tests for students to complete, and enable the secure submission of a student's work to the instructor.

A key aspect of e-learning is interactivity; where technology facilitates student interaction with the course instructor, interactions between students, and interactions with the course material (Evans & Gibbons, 2007). LMS's currently facilitate the interaction with between students, and between students and their instructors, but they do not currently provide much interaction with the course material. The majority of course material distributed using an LMS is in office file formats, meaning that little interactivity can take place.

In recent years there has been a vast increase in the number of tablets on the market (Farago, 2013), yet the use of tablet applications to implement interactive e-learning environments has not been exploited. Standard tablet hardware can be used to improve the interactvity of a student learning experience; by making use of the touchscreen, accelerometers, microphone and speakers for example. Touchscreens are good because… Additionally, a tablets are suitable for students to use to consume courses because of their portablity and relative low-cost compared to personal computers. It is also evident that using mobile applications is a preffered way for the general public to consume content, with a billion apps being downloaded weekly worldwide (Farago, 2013).

Adaptive e-learning is a term that has several similar interpretations in this field. In this project, it is used a broad term that refers to how content is manipulated in different ways so that it meets a specified criteria. In this context, a method of adaptivity is the concept of personalisation, which is a major theme of this project. With this in mind, we can define a system that delivers learning materials personalised for individual students as an Adaptive Educational Hypermedia System (AEHS) (Watson, et al., 2010).

Personalised e-learning takes place when the content that is delivered to a user has been deemed to be suitable for them after taking into account their preferences and personal needs. Such preferences and personal needs are what makes up a user profile. A user profile may be constructed explicitly by the user specifing such details, or by examining a user's behaviour to infer the details and construct the profile implictly (Kritikou, et al., 2008). The latter is far more convenient for end-users who do not want to spend time entering lots of information into the system. However, sometimes such implicit user profiling can be inaccurate or

incomplete, and so a hybrid aproach is most appropriate, whereby the system generates a user profile which can be modified by the user. Dynamic user profiling is where a system tracks changes in a user's preferences and needs over time, and adjusts its user profile and personalisation accordingly.

A key aspect in the personalisation of e-learning is the attempt to try and provide content that matches the learning style of the end-user. (Felder & Silverman, 1988) propose a learning-style model applicable to Engineering students, which classifies students on a number of scales based on the ways they receive and process information. This model looks five key dimensions of learning style; perception, input, organization, processing and undertanding. Perception is concerned with the type of information a student prefers to receive; it is either sensory (sights, sounds, physical sensations) or intuitive (possibilities, insights, hunches). The input looks at the preffered sensory channel to be used for receiving information, which can be either visual or auditory. The organization of information can be either inductive (principles are inferred from facts) or deductive (principles are given and the consequences/applications are deduced). It also considers that a student may prefer to process information actively (through some engagement) or reflectively (through introspection), and that they may progress towards understanding in a sequential (in continual steps) or global (in large jumps) fashion.

Early work on adaptive e-learning has utilized hierarchical structures that organize course content based on its subject and level of difficuly, such as in the case of InterBook (Brusilovsky, et al., 1998). Using the hierarchical structures, such systems try to identify the couse content that is suitable for different students by matching it with their user profiles, and by collating all the matched content, a personalised course navigation is produced.

More recent work on adaptive e-learning includes sysems such as AES-CS, (Triantafillou, et al., 2002), MOT+AHA! (Stash, et al., 2004) and iWeaver (Wolf, 2003). This work has improved adaptvity by annotating the course materials in the hierarchial structure with details relating to the learning style the material is appropriate for. These annotations are then used as parameters for pre-defined rules and conditions to offer adaptive navigation and/or presentation. These systems try to adapt to different categories of learning styles based on differing theories on the recognition and cantegorisation of the ways in which students learn. AES-CS (Triantafillou, et al., 2002) looks at field-dependent (FD) and field-independent (FI) learning styles. iWeaver (Triantafillou, et al., 2002) uses the Dunn & Dunn learning styles model (Dunn, et al., 1984) that categorises auditory, visual, kinaesthetic, impulsive, reflective, global and analytical as different learning styles. (Stash, et al., 2004) noted that in many cases the choice of learning styles was limited to suitable technology and attempted to allow for the implementation of more complex learning styles in MOT+AHA!

A problem with these adaptive e-learning systems is that they require instructors to spend considerable time and have certain technical skills in order to produce adaptive courses. MOT+AHA! (Stash, et al., 2004) provides a graphical user interface (GUI) so that instructors can visually identify/construct relationships between the different learning materials within a course. Course instructors are provided with progamming scripts in order to specify the conditions under which their materials should be delivered to students. However, such programming rules are not straight-forward for instructors to apply to their courses when considering the many padeagogical needs of students. Such needs may include the student's age, overall academic background, knowledge of the individual topic, workload as well as their learning style.

Recent work has tried to simplify the course authoring process of adaptive courses, so that ordinary instructors can create adaptive courses without needed prior technical knowledge. Work by (Li, et al., 2010) developed a framework that models peices of course specific knowledge as concept nodes in a concept space, separate from the modifiers that determine how the course should be disseminated, known as concept filters. The concept

nodes each have different aspects of knowledge, which are rated on two scales that define their importance and level of abstraction. Different concept filters are applied to the concept spaces to produce a personalised e-learning experience for different users based on their learning style and previous knowledge.

B. Project Purpose

The purpose of this project is to investigate how a personalised e-learning environment can be implemented effectively. In order for the e-learning environment to be effective, it needs to improve the student learning experience yet still provide course instructors with a way to author courses without needing to learn technical skills or spend a substantial amount of time to create them.

The project therefore falls into two main implementations; the course authoring application and the tablet application for viewing the courses. However, in order to implement these two applications, there needs to be a common framework so that the course authoring application creates courses that can be personalised, and so that the tablet application knows how to personalise the courses to the specific user. Therefore one of the main purposes of this project is to investigate and implement a framework for course generation that enables the courses to be personalised by the tablet application. This framework needs to satisfy the two conflicting requirements of a highly personalised course on the tablet application, with the ability to create the course quickly and without technical skill using the course authoring application.

This project is specifically targeting the use of Android tablets as a way of delivering course content to students. Tablets are mobile devices that have screen sizes large enough to display learning materials to students effectively, and their touch interface enables a higher level of interactivity than can be achieved with a traditional keyboard and mouse. Additionally, one of the main attributes of e-learning is that it makes distance learning possible, and tablets can extend this to enable mobile distance learning because of their screen size, weight and long battery life. The use of tablets in e-learning is not an area that has been fully explored and so this project is looking to investigate how they can be used effectively in the context of delivering personalised course materials.

The implementation of the course authoring software will be evaluated using a user study to determine how easy it is for instructors to generate courses. The tablet application will be evaluated using a user study and against a set of metrics to determine whether the course personalisation and its interactive elements improved the student learning experience.

C. Deliverables

Basic Deliverables

1. Develop a simple e-learning application to present course material on an Android tablet
2. Identify and implement some basic interactive elements to aid improving student learning effectiveness in terms of enhancing content visualization and navigation
3. Based on the developed e-learning application, conduct evaluation tests to identify key interactive elements that will improve student learning progress

Intermediate Deliverables

1. Develop techniques to allow instructors to easily author course content and to introduce interactive elements into a course for students to interact with course content
2. Evaluate the effectiveness of the course authoring environment

Advanced Deliverables

1. Investigate personalisation techniques based on student learning history and their interactions to develop methods to support adaptive course dissemination
2. Enhance the e-learning environment to allow instructors to edit and deliver adaptive e-learning courses
3. Evaluate the effectiveness of the personalised e-learning environment on students

II. Design

This section outlines the functional and non-functional requirements of the e-learning environment. It then looks at the architectural design of the entire system, and then identifies the individual components and how they interact. Finally it will look at some of the finer implementation details of the different system components.

A. Hardware and Software Requirements

The demonstration e-learning tablet application will be developed to be installed on tablets running the Android operating system. The demonstration application will be run on, and tested on an 'ASUS Eee Pad Transformer TF101' running Android version 4.0.3. This will be the only tablet and operating system combination that is fully supported by the application, and is what will be used to conduct the evaluations. However, the system will be designed to run on any Android tablet with an operating system of version 3.0 of higher.

The Android platform has been chosen because of its popularity and its wide range of development tools well-supported API (Application Programming Interface). Additionally, it is an open-source and less restrictive platform compared to iOS for example, meaning that it would be easier for this project to be extended in the future.

B. Tools

The application will be developed using the Eclipse IDE running on a Windows 7 operating system. This is because there is a freely available plug-in available from the Android developer website that facilitates Android development using Eclipse. The plug-in makes it easy to develop, compile, package and deploy Android applications, and includes a powerful Android emulator. It also allows for the application to be quickly deployed and tested on a real physical tablet with real-time test data such as logs easily viewable in Eclipse as the application is being used.

C. Requirements

Requirement Number	Functional Requirements	Priority
FR1	Provide a method of storing interactive course materials	High
FR2	Provide a method of storing course materials that allows for the materials to be personalised	High
FR3	The tablet application can display course materials to users	High
FR4	The tablet application can display course materials to users as interactive elements	High
FR5	The tablet application can save a user's progress so that it can be resumed if the application is closed	High
FR6	The tablet application can personalise course materials to meet individual user needs based on a user specified information	High

FR7	The tablet application's personalisations can be disabled and re-enabled at any time	High
FR8	The tablet application can personalise course materials to meet individual user needs based on information inferred from how the user uses the application	Medium
FR9	The tablet application can download course materials from a web service	Low
FR10	There is a user management system that restricts the access to materials to authorised users	Low
FR11	The course authoring application can generate interactive courses that can be viewed on the tablet application	High
FR12	The course authoring application can generate courses in a structure that allows for the tablet application to personalise the courses	High
NFR1	The tablet application will be targeted at Android OS version 4.0.3	High
NFR2	The web-based course authoring application will function correctly in Internet Explorer 7.0+, Mozilla Firefox 13.0+ and Google Chrome 18.0+	Medium
NFR3	The tablet application will automatically save a user's progress so that unexpected events do not cause loss of progress	Medium
NFR4	The course authoring application will not require users to have any programming knowledge	High
NFR5	The user interface of the tablet application should conform to Nielsen's 'Ten Usability Heuristics' (Nielsen, 2005)	Medium
NFR6	The user interface of the course authoring application should conform to Nielsen's 'Ten Usability Heuristics' (Nielsen, 2005)	Medium

Figure 1 – The System Requirements

D. Software Development Approach

An agile approach will be used to develop the system, in which the activities of specification, development and validation are interleaved, with the system being developed in increments, adding functionality to the previous version (Sommerville, 2010).

This approach has been taken because the purpose of this project is to investigate how to implement a personalised e-learning environment. Therefore, at the start of the project, the specification of the best possible implementation is not known. By taking this incremental approach it is possible to gradually build and improve the functionality of the system, in a bottom-up fashion, so that the final system implements the requirements in a way that makes the system as effective as possible. This is particularly the case with implementing the personalisation features because it means that the personalisation of course materials can be fine-tuned without having to backtrack through a development lifecycle. This process of gradual improvement of the program will mean that the evaluation of the system will give a more accurate picture of the effects of interactivity and personalization on the student learning experience.

Additionally, this approach reflects the real-world development process in the majority of industry situations. This is particularly the case with respect to the tablet application, as most small mobile applications such as this are often developed by a single programmer, and this will be the case with the entire system in this project. Therefore rapid prototyping and constant testing and revisions will be an inevitable part of the development process.

E. System Architecture

The system can be categorised into two distinct subsystems; the tablet application and the course authoring application. These tow subsystems need to share the same data, such as the course materials and user details. As a result, the architectural style of a data-centered repository has been chosen, where all the data is stored in a central, permanent storage location. This means that the system will take the form of a client-server model whereby the course authoring application and the course delivery application are the clients of a MySQL-based server. The representation of this client-server model can be seen in figure 1.

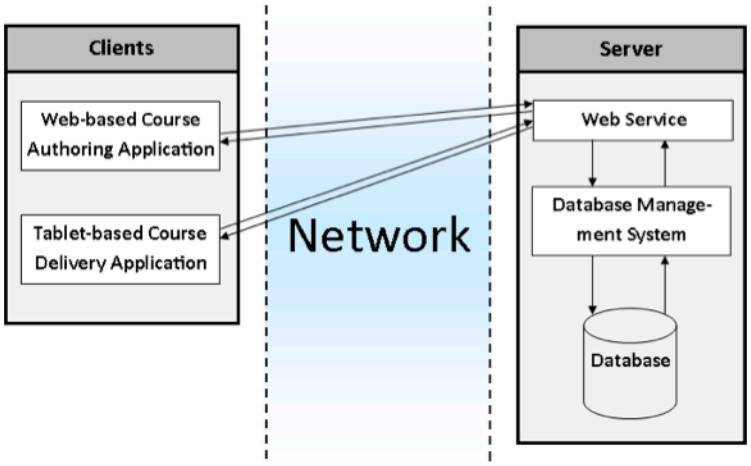

Figure 2 – The Client-Server Model

As shown in Figure 2, the web service plays a central role in the system as it is the only means by which the central data store can be accessed. All of the data created by, and read by the clients is obtained from the central server, although some of this data may temporarily be stored locally to speed up access times and to enable use of the course delivery application where there is no network connection available. The two clients will always update the central data store via the web service whenever there is a local change and a network connection is available. In order to access the database, parameters are sent to the web service in the form of specific URLs, which will return the requested data in the form of a data file or a generated XML document.

The database is the raw data store which is accessed by the DBMS, which in this case will be MySQL. By using a DBMS, a level of complication regarding physical storage details can be removed by using SQL (Structured Query Language) commands. The web service will be REST (Representational State Transfer) based, taking parameters from clients as input and constructing MySQL queries based on these parameters to access the database. A web service will be used because it helps to make the system more flexible, extensible and testable. This is especially important because of the agile approach being taken, as it means that changes made to the database or user management system will only require changes to the web service and not require multiple changes to each of the client applications.

An important component of the web service is the user management system which aims to restrict access to certain web services to appropriate users. It also influences the output of the web service so that user-specific results can be obtained, but only by a user who is logged-in and has permission to access that information. In order to speed up implementation, and because it is not the main focus of this project, the user management part of the web service will be performed by installing UserCake version 2.0.1. UserCake is a free PHP-based open-source user management system that just implements the functionality of allowing users to create accounts, log-in to accounts, and manage the permission levels of users and content.

F. The Course Framework

The course framework specifies how the structure of a course will be stored. The design of the framework takes is largely based on the ideas put forward by (Li, et al., 2010) with regards to a courses being made up of a structure of concept spaces and concept filters. We apply concept filters to the concept spaces in order to generate personalised course materials.

A concept space k is a collection of concept nodes k_n, with each concept node representing a small piece of knowledge. The concept nodes can belong to one of three concept spaces that describe its role in the course; as a pre-condition, core knowledge or advanced knowledge. An individual concept node can have several aspects $k_{n,a}$, with each aspect a representing a particular part of the information that the concept node is covering. Each aspect of a concept node has two indicators, an aspect weight and a level of abstraction that are defined on a scale between 0 and 1. The aspect weight looks at how important or relevant an aspect is, and the level of abstraction defines how detailed the content of an aspect is.

Course materials need to be saved as a data file in order to distribute the courses to students and enable the courses to be consumed offline. In order to create courses using the concept space structure outlined above, the courses will take the form of structured XML documents. The structure shown in Figure 3 can be translated into an XML document by using tags for each concept space, concept nodes and concept node aspects etc. XML tags can also be used to set the aspect indicators, as well as define the type of content and how it should be displayed. An example XML file can be seen in Figure 4.

```
<Course id="10001">
    <CourseName>Introduction to Programming</Course
    <CourseDescription>Description of the course...
    <Section id="1">
        <ConceptSpace id="Pre">
            <ConceptNode id="1">
                <ConceptNodeAspect id="1">
                    <AspectInfo>
                        <AspectWeight>0.6</AspectWe
                        <AspectAbstraction>0.3</Asp
                    </AspectInfo>
                    <AspectContent>
                        <Layout id="1">
                            <LayoutType id ="1">
                                <Text1>Text to disp
                                <Text2>More text to
                                <Image1>/IntroToPro
                            </LayoutType>
                        </Layout>
                    </AspectContent>
                </ConceptNodeAspect>
            </ConceptNode>
        </ConceptSpace>
    </Section>
</Course>
```

Figure 4 – An Example XML Document

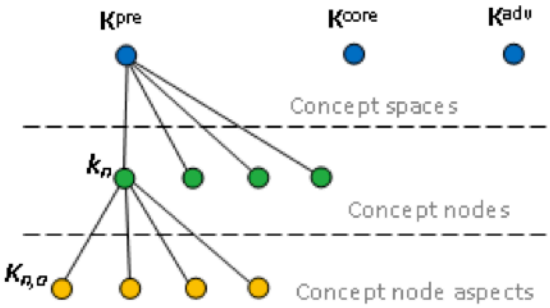

Figure 3 – The Concept Space Structure

Figure 4 shows that a course can have many sections, with each section containing three concept spaces of pre-conditional knowledge, core knowledge and advanced knowledge (the figure only shows the use of a pre-conditional concept space for brevity). Within each concept space we can have many aspects, with each aspect containing values for the two aspect indicators (weight and abstraction). In this implementation we can consider one aspect to be one or more screens of information to be displayed to the user in the tablet application. As such each aspect will utilize a layout type form a set of template layouts in order to display information. In this example we have used layout number 1, which provides us with screen space for two text sections and one image. This layout system can be extended to provide interactive elements to students by selecting a layout template that facilitates for this and specifying the parameters for it within the layout type tags.

G. Tablet-based Course Delivery Application

The tablet-based course delivery application is being implemented as an Android app, and therefore the design of the software needs to conform to the app standards required by the Android platform. The standard Android SDK uses the Java programming language and therefore that is the language that this application will be written in. Android apps make use of activities, with each activity being a type of task that the application can perform, often tied to a screen that contains the user interface for the application. Fragments are the different possible views that can be contained within an activity, and a fragment is embedded within an activity in order for it to be displayed. Different activities and fragments are implemented as subclasses of the Android API 'Activity' and 'Fragment' classes respectively.

In the class diagram shown in Figure 5, the application implements three different activities; 'AppActvitiy', 'MenuActivity' and 'ViewCourseActivity'. 'AppActivity' is the superclass of the other two activities and therefore specifies the features and functionality that are common to the entire application. For example, it will contain code to implement the ever-present action bar and any background tasks that might need to take place whilst the app is open. 'MenuActivity' will provide all of the menu functionality that is required in order to adjust settings and select the course to be viewed. This activity will use two different fragments to represent two different screens; 'HomeScreenFragment' to present the available courses for selection, and 'SettingsFragment' to adjust to present the settings screen. 'ViewCourseActivity' is the activity that will be used to present course materials to the users. It fetches the course materials and the user profile from the 'CourseManager' and 'UserProfile' classes respectively, and uses methods in the 'AdaptationManager' class to produce the course materials that are to be displayed in order to meet the needs of the user. The activity embeds a number of different fragments in order to present this information. As mentioned in the previous section, each aspect of a course from the course XML document can be presented using a different layout, some of which will include interactive elements such as games or animations. These layouts will be stored as fragments, and when a course aspect is to be presented to the user, it will be presented within the fragment that corresponds with the layout selected by the course designer.

The 'UserProfileManager' and 'CourseManager' are subclasses of the Android API 'AsyncTask' class because these classes need to run in the background when the activities are being used. The 'UserProfileManager' deals with logging-in users, it stores and updates the profile of the user as they are using the application, and syncs any changes to the user profile with the server via the 'Networking' class. The 'CourseManager' checks which courses are available for a user and downloads the courses using the 'Networking' class. It also opens the course files by parsing the data and storing it into memory so that it can be accessed by the activities.

The 'AdaptationManager' class personalises the course materials to meet the needs of the learning styles of individual students. The main way it will achieve this goal will be by applying a set of concept filters to the aspects of the concept nodes, as proposed by (Li, et al., 2010). The concept filters will determine the way that the course content from course structure should be disseminated to meet the needs of different students. These filters will be implemented as lists of (r_i, d_i)-pairs, where each (r_i, d_i)-pair defines the weight, on a scale between 0 and 1, for adjusting the importance and abstraction level respectively of an individual concept. The 'AdaptationManager' should formulate these filters based on the user profile, and apply them to the concept node aspects from the course structure. The different formulations of concept filters will change the content that is viewed by the user in the 'ViewCourseActivity' activity. It will allow for the courses to generated differently based factors such as a student's background, or whether they have a sequential or global learning style.

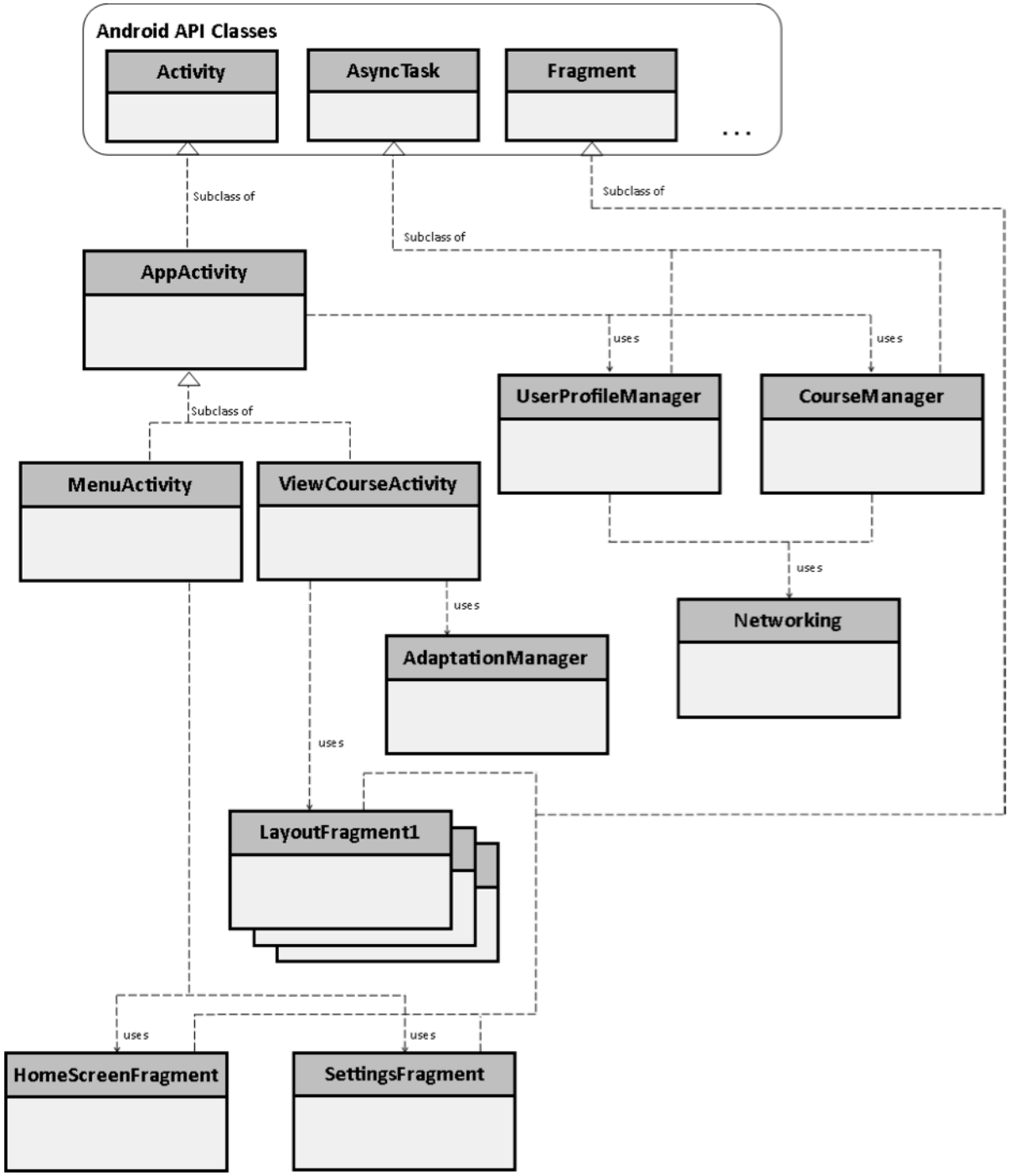

Figure 5 – Class Diagram of the Tablet-Based Course Delivery Application

Figure 6 shows the simple structure of the user interface, designed so that the hierarchy is not very deep, ensuring that the user does not get lost when using the application. On every screen in the application, there will be an action bar which will provide users with the option to change some of the settings in the application, such as turning personalisation on/off. It will also always have a 'return' button on that takes the user up the hierarchy by one level, unless they are already on the homescreen, in which case the return button closes the application. It is

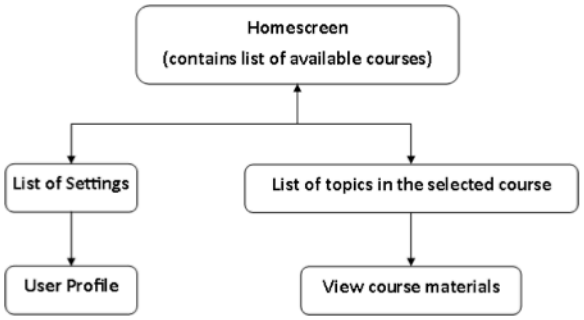

Figure 6 – The Hierarchical Structure of the GUI

important that this 'return' button only ever goes up the hierarchy by one level and does not perform any other function. This is a key principle in Nielsen's 'Ten Usability Heuristics' (Nielsen, 2005); that users need control and freedom. Additionally, when the return button is pressed whilst a user is viewing course materials, the progress of the student is automatically saved, meaning that the user will always return to the correct part of the course when they next load the application. When the users are viewing the different screens of course content, the user can simply swipe the screen left-to-right to go back a screen, or right-to-left to go forward a screen. Additionally, throughout the application, when there is more information to fit on the screen than there is space for, a vertical scroll will be introduced; where users swipe the screen bottom-to-top to move down the page, and top-to-bottom to move up the page. These 'swipes' are examples of how the principle of 'consistency and standards' will be implemented to help improve the usability of the application.

H. Web-based Course Authoring Application

The web-based course authoring application will be made up of a set of PHP (Hypertext Preprocessor) web pages and associated CSS (Cascading Style Sheet) files. Figure 7 shows the proposed structure of the application, without the CSS files for brevity. As shown, the course authoring functionality of the application is provided on a single page, which has a container that is updated by AJAX (Asynchronous JavaScript and XML) calls based on user input. The container can be updated with five different PHP files, which provide five different views; the first prompting the user to select a course, and the other four providing a way to navigate and edit different levels of the course hierarchy.

The first four views are concerned with navigating and editing the hierarchy of the course, with the final view of the 'Aspect Editor' being the view where content is added by course instructors. In this view, course instructors add the individual aspects of a course topic (referred to as a concept node in the framework). Each aspect has two indicators associated with it, which are specified on a scale between 0 and 1 by the course instructor. These indicators specify the importance/relevance and the level of abstraction of the aspect. When creating an aspect, course instructors can specify how the content should be delivered by specifying the layout template, which is effectively how the content will be displayed on screen to the students. The course instructors will have the choice of a range of layouts, some of which allow for interactive elements (such as games) to be created. The instructors then add the information that is requested and an aspect is created.

Whenever a change is saved on the course authoring page, no matter which view is being used, this change is passed as a parameter to the 'XML Processor'. The 'XML Processor' is a PHP function (it is not displayed to the user) that edits the XML document associated with the course that is being edited. When a change is made, the details of that change are passed to the 'XML Processor' as a set of parameters, which specify the change and the exact location of the change. The 'XML Processor' obtains the relevant XML document from the database via the web service, opens it and makes the required change using pre-defined functions. The revised XML document is then created and uploaded to the database in place of the existing one via the web service.

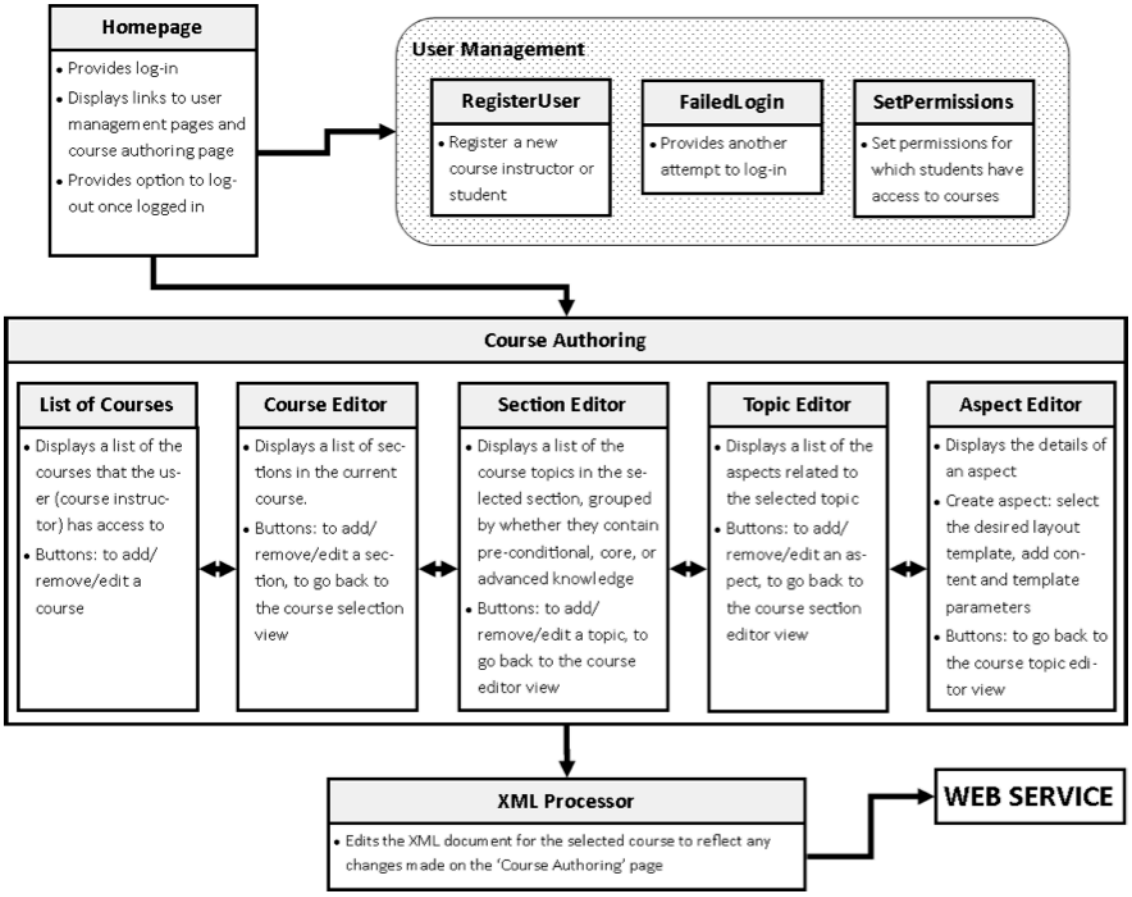

Figure 7 – Structure of the Course Authoring Application

REFERENCES

Brusilovsky, P., Eklund, J. & Schwarz, E., 1998. Web-based education for all: a tool for development adaptive courseware. *Computer Networks and ISDN Systems,* 30(1-7), pp. 291-300.

Dunn, R., Dunn, K. & Freeley, M. E., 1984. Practical Applications of the Research: Responding to Student's Learning styles - Step One. *Illinois State Research and Development Journal,* 21(1), pp. 1-21.

Evans, C. & Gibbons, N. J., 2007. The interactivity effect in multimedia learning. *Computers & Education,* 49(4), pp. 1147-1160.

Farago, P., 2013. *Flurry Blog: Holiday 2012 Delivers Historical Worldwide App Downloads*. [Online]
Available at: http://blog.flurry.com/bid/92809/Holiday-2012-Delivers-Historical-Worldwide-App-Downloads
[Accessed 04 01 2013].

Felder, M. R. & Silverman, L. K., 1988. Learning and Teaching Styles: In Engineering Education. *Engineering Education,* 78(7), pp. 674-681.

Kritikou, Y. et al., 2008. User Profile Modeling in the context of web-based learning management systems. *Journal of Network and Computer Applications,* 31(4), pp. 603-627.

Li, F. W. B., Lau, R. W. H. & Dharmendran, P., 2010. An Adaptive Course Generation Framework. *International Journal of Distance Education Technologies,* Jul-Sep, 8(3), pp. 47-64.

Li, Q., Lau, R. W. H., Shih, T. K. & Li, F. W. B., 2008. Technology Supports for Distributed and Collaborative Learning over the Internet. *ACM Transactions on Internet Technology,* 8(2, Article No. 5).

Nielsen, J., 2005. *Ten Usability Heuristics*. [Online]
Available at: http://www.nngroup.com/articles/ten-usability-heuristics/ [Accessed 23 01 2012].

Sommerville, I., 2010. *Software Engineering*. 9th (International Edition) ed. s.l.:Pearson.

Stash, N., Cristea, A. & Be Bra, P., 2004. Authoring of Learning Styles in Adaptive Hypermedia: Problems and Solutions. *Proceedings of the 13th International World Wide Web Conference,* pp. 114-123.

Triantafillou, E., Pomportsis, A. & Georgiadou, E., 2002. AES-CS: Adaptive Educational System based on Cognitive Styles. *Proceedings of AH2002 Workshop, Second International Conference on Adaptive Hypermedia and Adaptive Web-based Systems,* pp. 10-20.

Watson, C., Li, F. W. B. & Lau, R. W. H., 2010. A Pedagogical Interface for Adaptive e-Learning Course Authoring. *Proceedings of ACM MM Workshop on Multimedia Technologies for Distance Learning,* Oct.

Wolf, C., 2003. iWeaver: Towards 'Learning Style'-based e-Learning in Computer Science Education. *Proceedings of the Fifth Australasian Conference on Computing Education,* Volume 20, pp. 273-279.