AN ARCHITECTURE FOR AUGMENTING THE SCORM RUN-TIME ENVIRONMENT WITH A PERSONALISED LINK SERVICE

By

Nor Aniza Abdullah

A thesis was submitted for the degree of Doctor of Philosophy

In the

Faculty of Engineering and Applied Science
Department of Electronics and Computer Science
University of Southampton
United Kingdom

June 2006

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

AN ARCHITECTURE FOR AUGMENTING THE SCORM RUN-TIME ENVIRONMENT WITH A PERSONALISED LINK SERVICE

by Nor Aniza Abdullah

As the result of recent advances in the business of e-learning there has been a growing interest in e-learning standards, particularly SCORM (Sharable Content Object Reference Model). SCORM is a reference model that integrates a collection of e-learning resource standards and specifications. In a SCORM compliant courseware, content and the pedagogic approach to be taken are predefined by the course author. As a consequence, users are unable to learn according to their preferences, and implicitly they will all encounter the same learning experience. Recent attempts to personalise learning in SCORM often resulted in either modifying SCORM or substituting its course sequencing mechanism with Adaptive Hypermedia techniques. Nonetheless, SCORM is a widely-used solution to interoperability problems with e-learning resources and can realise not only content sharing and reusability but also a consistent sequencing of course content across different systems and tools. For these reasons, this thesis focuses on *supplementing* SCORM sequencing rather than redefining it. The novelty of this work is that it builds an adaptive environment around the existing SCORM, without extending or modifying SCORM itself. The work integrates Adaptive Hypermedia principles into the SCORM Run-Time Environment as an independent service to support the environment with learning materials that are preferable to both teachers (primary materials of their choice) and students (supplementary materials that are pre-selected to suit some aspects of their user models to assist their understanding of the primary materials). This SCORM-complementing approach can also enable the SCORM Run-Time Environment to include on-demand external resources into the environment in order to address SCORM's limitation of static pool of learning resources.

The novel contribution of this work is the design of an authoring architecture which enables the automatic generation of a concept map and consequently links to alternative learning resources, and a run-time service oriented architecture which delivers these alternative resources, alongside the SCORM defined resources, according to a dynamic user model.

Table of Content

List of Figures

List of Tables

Acknowledgements

I would like to thank Dr Hugh C. Davis for his excellent supervision and support in this research. Also thanks to Dr Christopher Bailey, Dr Yvonne Howard, and Dr Sanjay Vivekanandan, who have helped me proof-reading this thesis. Special thanks go to my family, friends and all the LTG and IAM members that have helped and supported me over the last three and a half years. Above all, I want to thank God for His grace that makes it possible for me to produce this work.

Definitions and Abbreviations Used

AICC	Aviation Industry Computer Based Training Committee
ADL	Advanced Distributed Learning
AEHS	Adaptive Educational Hypermedia System
ARIADNE	Alliance of Remote Instructional Authoring and Distribution Networks for Europe
FOHM	Fundamental Open Hypertext Model
IEEE	The Institute of Electrical and Electronic Engineers
LOM	Learning Object Metadata
LTSC	Learning Technology Standard Committee
MERLOT	Multimedia Educational Resource for Learning and Online Training.
OHP	Open Hypermedia Protocol
SCORM	Sharable Content Object Reference Model
PLS	Personalised Link Service

In the memory of my beautiful and beloved mother whose smile will always last forever.

1 Introduction

1.1 Research Overview

Today's phenomenon of e-learning environments reveal a growing number of disparate tools and applications that are incompatible with each other, making interoperability impossible to achieve (Conlan et al., 2002; Anido-Rifón et al., 2001). This is often the case whereby developers and publishers of the learning resources must design their content for a particular learning environment due to the specific requirements imposed by the local repository, institution, or by a specific content management service offered by the learning environment (Conlan et al., 2002). As a result, content has become application-dependent and cannot be shared or reused by other applications and/or by other organisations.

Nonetheless, the ability to share and reuse existing content can alleviate the time and costs involved in preparing high quality multimedia learning resources. This need has given rise to the notion of Learning Objects and the development of standards and specifications to support their use across different learning environments and products (Mohan and Greer, 2003; Clark and Rossett, 2002). As a Learning Object, each resource should be labelled with metadata that basically provides bibliographic information about the resource. To enable Learning Objects to be reused from one system to another, it is crucial to have standards for labelling them so that they can be located by

potential users in a consistent manner regardless their development or delivery platform. Standards are also required for packaging Learning Objects so that they can be moved and used across any computer platform and learning management system without modification (Mohan and Greer, 2003).

Currently, there are several learning resource standards and specifications that have been developed by initiatives bodies such as the IMS (IMS, 2003), ARIADNE (Alliance of Remote Instructional Authoring and Distribution Networks for Europe) (ARIADNE, 2004), AICC (Aviation Industry Computer Based Training Committee) (AICC, 2000), and ADL (Advanced Distributed Learning) (ADL, 2004). Among the most widely adopted set of standards and specifications are Learning Object Metadata (LOM) accredited by the IEEE (The Institute of Electrical and Electronic Engineers) (IEEE LOM, 2002), several IMS learning specifications (e.g. IMS Content Packaging and IMS Simple Sequencing) and SCORM (Sharable Content Object Reference Model) by ADL (SCORM, 2002).

Today, SCORM appears to be the most popular technical specification among e-learning developers (DigitalThink, 2003). The reason is because SCORM adopts most of the above mentioned e-learning industrial standards and specifications to provide a comprehensive suite of e-learning capabilities that enable interoperability, accessibility, and reusability of Web-based learning content (DigitalThink, 2003; Slosser, 2002). Another attribute of SCORM that highly influenced its widespread use among e-learning practitioners is that it provides practical implementation of its functional model by offering an open java-based run-time environment.

However, SCORM learning environment has several limitations, among which it currently lacks user personalisation (Power et al., 2005; Blackmon et al., 2004) and user's learning experience are constrained to a static pool of predefined learning resources (Abdullah and Davis, 2005).

To provide personalisation and adaptive learning in SCORM, potential solutions can be derived from the following technologies: intelligence tutoring system (Sleeman and Brown, 1982) and adaptive hypermedia system (Brusilovsky, 2001). While an intelligence tutoring system is usually instructional-centred, an adaptive hypermedia system attempts to make the presentation of content or links of hypermedia content adapt to individual students (Brusilovsky, 2001). However, despite its ability to accommodate personalised and adaptive learning, adaptive hypermedia learning applications are not as popular among e-learning developers (Duval, 2004; Cristea et al., 2003; Abdullah and Davis, 2003).

Duval states that the unpopularity of the adaptive hypermedia systems is due to the lack of standards and specifications used in designing and developing the systems (Duval, 2004). Other possible reasons for its unpopularity are also described in (Cristea et al., 2003), among which a lack of standards or common practices in authoring adaptive techniques and behaviours is pointed out. To address this issue, an European Community project called ADAPT has begun to develop a common design pattern for adaptive educational hypermedia system, and to use it in authoring adaptive courseware (Cristea, 2003; Brown et al., 2005). Other Adaptive Hypermedia researchers, for instance, Dagger and Dolog have incorporated e-learning standards and specifications for learning resources in the design and development of their adaptive systems (Dagger et al., 2003; Dolog et al., 2003). Modritscher, Baldoni and Power, on the other hand, have attempted to extend the SCORM learning environment with Adaptive Hypermedia techniques (Modritscher et al., 2004; Baldoni et al., 2004; Power et al., 2005). However, their efforts to personalise SCORM's learning often resulted in either modifying the SCORM's structure or substituting its existing course sequencing mechanism with an application-dependent adaptive sequencing mechanism. Omitting the SCORM sequencing mechanism will result in the inability to achieve a consistent course sequencing behaviour across different systems. This is because SCORM sequencing mechanism which is based on IMS Simple Sequencing specification (IMS SS, 2002a) can enable any SCORM-compliant learning management

system to deliver a consistent sequencing of course content regardless its delivery platform (Bouras et al., 2003). The IMS Simple Sequencing specification defines the required sequencing behaviours and functionality that conforming learning technology systems must implement in order to sequence discrete learning activities in a consistent way.

For these reasons, this thesis focuses on supplementing SCORM sequencing rather than completely disregarding it. The novelty of this work is it builds an adaptive environment around the existing SCORM, without extending or modifying SCORM itself. Therefore, this thesis suggests that instead of personalising the sequence of learning activities in SCORM, it would be useful to populate SCORM with supplementary materials, and personalise the selection of these materials according to each student's preference. This SCORM-complementing approach can address both SCORM's limitations mentioned earlier – lack of user's personalisation and the inability to dynamically include alternative learning resources at run-time. Another aspect that influences this approach to augment SCORM with personalised links to supplementary materials is the ADL vision of SCORM. As reported by Slosser, the lead software engineer at the Joint ADL Co-Lab, ADL's long-term plan for SCORM is to make it adaptive (Slosser, 2002). The work by Blackmon has demonstrated the possibility of enhancing the capability of the IMS Simple Sequencing (IMS SS, 2002a) elements to sequence course content adaptively (Blackmon et al., 2004).

Therefore, the author believes that by providing Adaptive Hypermedia advantages as a support system to SCORM rather than as an alternative to its existing delivery mechanism, it will increase the likelihood of the Adaptive Hypermedia application to be embraced by the e-learning community. By doing so, the author hopes to bring Adaptive Hypermedia technology into the realm of the e-learning community and offer a solution that can be easily adapted to future versions of SCORM specification.

To implement this recommendation, an architecture has been developed that can populate a service to augment SCORM learning environment with personalised

supplementary resources. This architecture also demonstrates a way for a structural approach of adaptive hypermedia system to work in an open learning environment, which is accomplished from the integration of the following technologies: Adaptive Hypermedia, Open Hypermedia and Learning Object Technology.

Throughout this thesis, terms 'personalisation' and 'adaptation' have been frequently used and sometimes they are even used interchangeably. As according to the Collins Dictionary, both terms can be defined as follows:

a) Personalisation : To provide with personal or individual qualities

b) Adaptation :
- i- The act or process of adapting or the state of being adapted.
- ii- Something that is changed or modified to suit new conditions

Based on these definitions, we can derive that personalisation is the outcome of an adaptation process towards users. For instance, users of the PLS system receive links personalised to their attribute values (Learning Style). However, the system also undergoes the process of adaptation in which the criteria for selecting potential links will change to accommodate new users' requirements (e.g. changes in Learning Style) that may occur as the result of users interactions with the system.

Adaptation can also be specifically defined as a process of changing the availability or presentation of links or content based on some criteria, for example user model, devices or location. In this regards, the PLS system can also be seen as providing an adaptation service because it does not only personalised to user's attribute value (Learning Style) but also adapts to the SCORM content and changes in user's attribute value while interacting with the system.

1.2 Research Objectives

The first objective of this work is to develop an architecture that permits the incorporation of Adaptive Hypermedia technology into SCORM to augment its predefined course content with dynamic links to alternatives learning resources chosen in accordance to a user model.

The second objective is to develop an architecture that can automate the process of preparing the domain model, and the linkbase (a digital repository that contains link to the recommended supplementary resources).

1.3 Research Contributions

This thesis provides an architecture for a Personalised Link Service to dynamically populate SCORM with external resources personalised to a user model. With this, the author believes that her work has contributed in the following aspects:

i. A critical analysis of the differences between Adaptive Hypermedia and IMS Simple Sequencing techniques

ii. An architecture for a Personalised Link Service to SCORM which demonstrates that it is possible, to populate a service to supplement SCORM learning materials according to some preferences in a user model, without any mandatory authoring intervention.

iii. An approach to automatically construct a concept map from the SCORM package which is dictionary and human independent.

iv. A table-driven approach that provides a familiar environment for an author to map resource types against some user model values (e.g. Preferred Learning Style). The input will later form a basis for generating flexible implementation rules for defining the required adaptation.

1.4 Thesis Structure

Chapter 2 introduces readers to the Adaptive Hypermedia technology – the history of its origins, its characteristics and techniques, the benefits it offer especially to facilitate learning, as well as its current limitations. This chapter ends with a brief literature on some learning style models as the prototype implemented in this system relies on the awareness of these models in order to deliver dynamic learning style adaptation.

Chapter 3 provides the background to the Open Hypermedia technology with emphasise on a standardised data model for hypermedia structure, FOHM (Fundamental Open Hypertext Model). This is followed by a description of Auld Linky, a contextual link server that implements and maintains FOHM, which is used in the implementation to provide adaptive links. The chapter ends with a brief review of a web-based adaptive hypermedia application which demonstrates how FOHM can be used to deliver Adaptive Hypermedia techniques.

Chapter 4 introduces readers to Learning Object technology. The chapter begins by establishing some understanding about what is Learning Objects and why there is an enormous need for them. Related existing e-learning standards and specifications are then described with emphasise on SCORM (Sharable Content Object Reference Model). This is then followed by reviewing SCORM's sequencing mechanism which is based on the IMS Simple Sequencing specification. The chapter ends by reporting the differences between Simple Sequencing and Adaptive Hypermedia.

Chapter 5 introduces readers to the Personalised Link Service architecture. The chapter begins by addressing certain aspects of SCORM that motivate this research, followed by the overview of the architecture and related work. Detailed elaboration of the design phases of the architecture is spread over the subsequent chapters (Chapter 6 and 7).

Chapter 6 describes the process of designing and developing the pre run-time authoring environment for the Personalised Link Service (PLS). The chapter begins with the design issues pertaining to the preparation of a concept map, followed by the development of a FOHM-structured linkbase. Both the concept map and the linkbase will serve as the input components for the next design phase (PLS Run-Time Environment). This chapter also includes a description of the UtilityTool used in the production of these input components. The chapter ends by comparing PLS's approach in automatic extraction of learning concept from the SCORM's resources with the one mentioned in the literature.

Chapter 7 describes the process of designing and developing the PLS Run-Time Environment. The processes involved are primarily described according to the following implementation layers: PLS Client and PLS Web Service. The PLS Client layer contains the functional model of an adaptive engine which contains the following components: Concept Tracker, Preference Tracker, Mapper, Query, and Renderer. The PLS Web Service is primarily composed of a contextual link server i.e. Auld Linky that is responsible to perform adaptive query to linkbases based on a requested context from the Client. The chapter also includes the following descriptions: the mechanism used in delivering dynamic learning style adaptation, the correlation between types of learning resources and learning styles, the related work found in the literature in regards to dynamic learning style adaptation, and finally the implementation of the PLS Service-Oriented Architecture using Apache Axis.

Chapter 8 evaluates the feasibility of the proposed architecture to deliver personalised and adaptive links to a SCORM open learning environment. Two experiments have been performed to test two hypotheses. The first hypothesis is that different user model will receive different set of links appropriate to his/her user model value (e.g. Preferred Learning Style) at any point of time. Whilst, the second hypothesis is that the returned links are indeed relevant to the requested learning concept. The chapter ends with some justifications on the experiments' results.

Chapter 9 discusses issues arises from the development of the Personalised Link Service to SCORM followed by the conclusion of this thesis. The chapter ends with some recommendations of the possible direction for future work as well as future research emanating from this work.

2 Adaptive Hypermedia

Chapter 2 begins with a brief history to hypertext and hypermedia before proceeding to the main concepts required to understand adaptive hypermedia technology. The concepts include the basic components of an adaptive hypermedia system, its primary techniques in delivering adaptation to users as well as issues that hindered its advancement into the mainstream of e-learning. Some awareness about this technology is necessary in order to reason the design approach implemented in this thesis.

This chapter also provides a piece of information regarding the theory of learning style and its associated learning style models. This information is important as the prototype implemented in this work delivered adaptation based on user's Preferred Learning Styles. Readers' awareness of the individual preferences towards learning as portrayed in these learning style models can therefore facilitate their understanding of the outcome of the prototype.

2.1 History: Hypertext and Hypermedia

The term hypertext term was introduced by Ted Nelson in 1965 as a part of his Xanadu project. Hypertext refers to a document in which information is stored in nodes connected associatively by links (Nelson, 1965). According to Smith and Weiss each node can be thought of as being analogous to a short section of an encyclopaedia article or perhaps a graphic image with some brief explanation (Smith and Weiss, 1988). These

sections are then joined together with associative links represented via link anchors to form the article as a whole and the articles are joined in the same way to form the encyclopaedia. In this hypertext environment, the link anchors usually become user's navigational tool to explore the whole encyclopaedia (Smith and Weiss, 1988).

The idea of linking one text to another via a form of association (hypertext) was first introduced in the year 1945 by Vannevar Bush via his conceptual machine called MEMEX, 'Memory Extender' (Bush, 1945). MEMEX tried to bring forward the idea of augmenting human memory, exploiting its power to store and retrieve data by following association trails of thought. The envisioned device consists of a desk with slanting translucent screen, on which materials can be projected for reading, a keyboard to enable users to input codes of material they wish to search, and sets of buttons and levers to control the speed and sequence (i.e. forward or backward) of the displayed materials. With MEMEX an individual can store any kind of information such as books, periodicals, newspapers, and pictures. These data items, which are stored on microfilm, could be projected onto MEMEX's desk, and several projectors would enable the user to view more than one document at a time.

The MEMEX illustrates Bush's idea of associative indexing whereby "*any item may be caused at will to select immediately and automatically another*" (Bush, 1945). The process of building an associative trail began with a user giving a name to a new trail that he wanted to build in his code book. He then tapped his keyboard to project two documents to be linked, adjacently onto the desk. After that, he joined the two items by another tap on the keyboard again. Subsequently, a code is built on the code space for each item to designate the index number of the item it is linked to. The process continued to form a trail. Thereafter, when one of these items is in view, the other can be instantly recalled and reviewed in turn (Bush, 1945).

While hypertext usually deals with the associative linking between texts, hypermedia extends the ability to also include other types of media such as image, graphics, audio and simulation. Today the terms hypertext and hypermedia are used interchangeably

(Bailey, 2002; Ng, 2003). In 1990 Tim-Berners-Lee demonstrated the concept of hypertext client-server approach in a remote and distributed environment, producing what today is known as the World Wide Web or the Web (Berners-Lee and Cailliau, 1990). Today the Web has emerged as the dominant hypertext technology because it is highly accessible to anyone, anywhere in the world (Ng, 2003).

2.2 Hypermedia and Learning

Like any other hypertext system, the Web uses the concept of documents, nodes or pages that are interrelated by a set of navigational links. Users explore the Web by activating the links to navigate from one page to another. While the underlying hypertext structure of the Web easily "*permits users to freely explore and follow links in whenever and whatever sequence they please*" (Ng, 2003), its free-browsing behaviour often leads to two shortcomings: 'Disorientation' and 'Cognitive Overload' (Conklin, 1987).

The disorientation effect is actually due to the nature of the hypermedia itself. Since the Web is growing rapidly with trillions of linked nodes, users might easily lose their learning directions. This occurs when each new encounter interests users, and thus gradually drifts them away from their initial goals. As highlighted by Jonassen "*simply browsing hypertext is not engaging enough to result in more meaningful learning*" (Jonassen, 1993). Bajraktarevic confirms it by saying that "*simple browsing of Web documents does not necessarily lead to successful learning*" (Bajraktarevic, 2003).

The cognitive overload occurs when user is overwhelmed with too much unguided information and options while interacting with the Web. The consequence of too much information can result in low efficiency of the human mind to absorb and process useful information which may lead to unsuccessful learning (Ng, 2003).

Adding to these shortcomings is the fact that each user is different and has unique preferences towards learning. One solution is to control the availability of information to individual users by personalising the users' learning environment (Ng, 2003).

Fischer states that "*the challenge in an information-rich world is not only to make information available to people at any time, at any place, and in any form, but specifically to say the 'right' thing at the 'right' time in the 'right' way*" (Fischer, 2001). Therefore Web-based applications should cater for heterogeneous learners by applying a specific adaptation according to the characteristic of an individual. Adaptive Hypermedia research which emerged in the early nineties is aimed to serve this purpose.

Adaptive Hypermedia techniques enhance a hypertext and hypermedia system by providing directional and cognitive support to users while browsing. This is accomplished by storing some personal features about the user in a user model and then applies this model in order to adapt the presentation of links and content of hypermedia pages to the current need of the user (Brusilovsky, 2001).

2.3 Adaptive Hypermedia System Components

A typical Adaptive Hypermedia System is composed of a domain model, a user model, an adaptation model, and an adaptive engine (Wu et al., 1998). The following subsections elaborate each of these components.

2.3.1 Domain Model

Usually a domain model contains a set of domain concepts along with their relationships. There are several commonly used methods to structure a domain model. These include linear, concept graph, semantic network, hierarchical tree, combined structure, and teaching task and rule-based structure (Carro, 2002).

In a linear structure, a linear relationship is established among a set of identified concepts or information units, allowing only a sequential type exploration of the hyperspace.

A concept graph defines a domain structure in terms of nodes (represent the information units, concepts, or tasks) and arcs (represent the relationships among the nodes). The arcs of a common concept graph usually represent an 'is-related-to' relationship. A specific type of a concept graph called a prerequisite graph is normally used to represent 'is-prerequisite-of' relationship.

Similar to a concept graph, domain structuring based on a semantic network is also composed of a set of nodes and arcs. The only distinctive feature is that the arcs can represent different types of relationships such as 'is-similar-to', 'is-opposite-to', 'is-part-of', 'is-prerequisite-of', 'is-example-of' and etc.

The next structure is a hierarchical tree. This structure usually consists of a set of nodes, which represent basic units of knowledge, and a set of arcs that represent the decomposition relations among them ('is-part-of' relationship). In this structure, each node represents a concept with only one ancestor and its direct descendents represent its sub-concepts.

In a combined structure, a combination of several structures described above will be used to represent a domain model.

Finally, in teaching task and rule based structure, an atomic task is defined as the basic unit that represents the concepts, topics or procedures to be learned. A composed task represents ways of grouping those tasks. A rule is then assigned to the composed task. The rule contains an activation condition that is associated with a user's characteristic or behaviour. By defining several rules for the same composed task using a different set of activation conditions, different structures for different kinds of users that access the same set of topics can be achieved.

2.3.2 User Model

User model captures individuals' characteristics that encompass each specific user. Therefore, in an adaptive hypermedia system, a user model is crucial in determining the success of the adaptation process.

According to Kavcic, there are three important aspects that have to be considered when designing a user model (Kavcic, 2000): 1) the types of user's information that needs to be captured and how to obtain it; 2) how to represent the information in the system; 3) how to construct and update the model.

Information that is normally captured in a user model can be divided into two categories: static and dynamic information. Static information conveys users' personal data such as users' identification and background, for instance user's background knowledge or career. Whilst, dynamic information referring to user's information that requires update as a result of their interactions with the system such as knowledge levels and learning goals. This information can be obtained by requesting users to fill in the required information in a form from a dialogue window. The initial value for the dynamic information can also be obtained using the same approach. However, the value should be updated at the end of a session or throughout users' interactions with the system.

In representing user's information in an adaptive educational hypermedia system, several types of user models are normally used. The most commonly employed are the overlay model, stereotype model or a combination of both. In the overlay model, user's knowledge is regarded as a subset of expert knowledge. Therefore the user model usually contains a list of concepts from the domain model with the corresponding values that indicate the system's belief of how much a student has mastered a given concept. Among the adaptive hypermedia systems that uses the overlay model are MetaLinks (Murray et al., 1998), Dynamic Course Generator (Vassileva, 1997) and AHA! (DeBra and Calvi, 1998). In a stereotype model, an individual user is assigned to

one or more stereotypes after responding to a series of questions or other form of user input. Each stereotype has its predefined properties and users that belong to that stereotype also inherit its properties. For example, HyperTutor (Perez et al., 1995) uses stereotypes where users can belong to one of the following three groups: novice, medium or expert. The implementation of both models can be found in WHURLE (Brailsford et al., 2002).

There are also several ways to create and update a user model. According to Bajraktarevic, in some systems, the user model is created at the start of the learning process and continuously updates stored information as the learner interacts with the system such as in AHA! (DeBra and Calvi, 1998); while in other systems it is created at the end of a learning session in which user's performance or interest is tracked over a longer period of time (Bajraktarevic, 2003). In certain systems, it's a mixture of both.

2.3.3 Adaptation Model

An adaptation model usually contains rules that define how the domain model relates to the user model in order to specify adaptation. The rule usually takes the form of "*if <condition> then <action>*" (Wu et al., 1998). By interpreting this rule, an adaptive engine will generate the adaptation outcomes by either manipulating the presentation of the link anchors or the fragments of the hypermedia content pages. The adapted page will then be delivered to the users.

2.4 User Model Features for Adaptation

There is a variety of adaptations, including those that adapt to users' knowledge levels (Brailsford et al., 2002 ; DeBra and Calvi, 1998; Vassileva, 1997), learning goals (Murray et al., 1998; Vassileva, 1997), users' interests (DeBra and Calvi, 1998), users' backgrounds (Brailsford et al. 2002), users' experiences (Vassileva, 1997), learning styles (Stash et al., 2004; Kinshuk and Lin, 2004; Bajraktarevic, 2003; Paredes and

Rodriguez, 2004; Carver et al., 1999), reading speed (Ng, 2003) and navigational history (Murray et al., 1998, Ng, 2003; DeBra and Calvi, 1998).

2.5 Adaptive Hypermedia Techniques

Adaptive hypermedia techniques can be segmented into two different categories of adaptation, labelled as an adaptive presentation (content- based adaptation) and an adaptive navigational support (link-based adaptation) (Brusilovsky, 2001), as depicted in Figure 2.1.

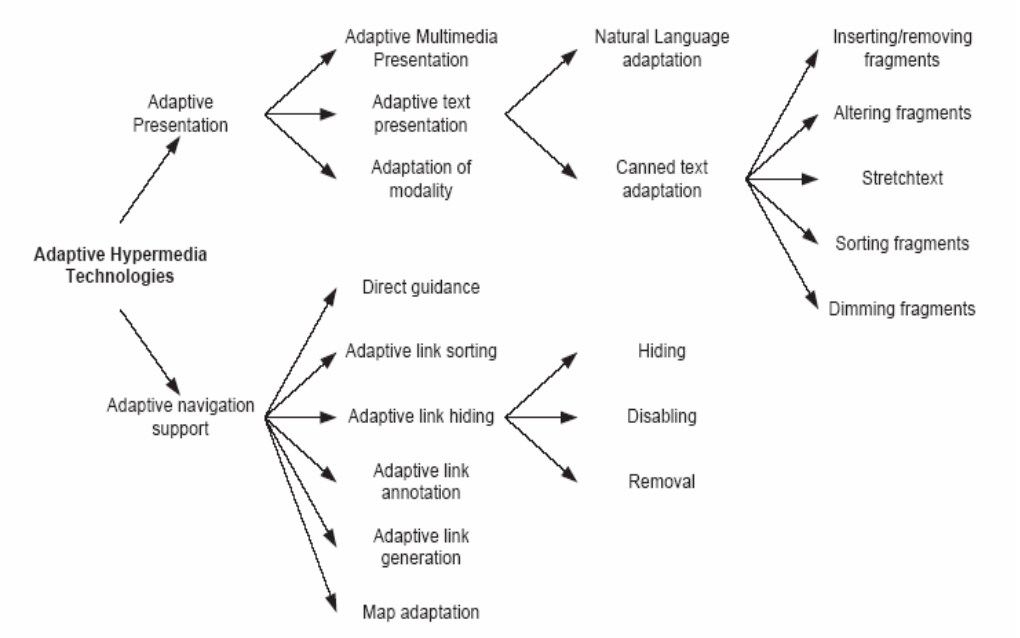

Figure 2. 1: Updated taxonomy of adaptive hypermedia technologies (Brusilovsky, 2001)

2.5.1 Adaptive Presentation

Adaptive presentation comprises of a collection of techniques for altering the content of page accessed according to the needs of a particular user or a group of users. Under this banner, there are adaptive multimedia presentation, adaptive text presentation as well as adaptation of modality. At present the majority of the work is in the area of canned text adaptation that is under the adaptive text presentation category. Canned

text adaptation deals with text and fragment processing like inserting or removing fragments, altering fragments, stretch-text, sorting fragments and dimming fragments (Brusilovsky, 2001).

2.5.2 Adaptive Navigational Support

Adaptive navigational support guides users' orientations and navigations in the hyperspace. This is accomplished by modifying the appearance of hyperlinks shown to users in order to facilitate them in finding their paths and relevant information (Brusilovsky, 2001; Bailey, 2002). Adaptive techniques that provide adaptive navigational support can be classified in six groups: direct guidance, adaptive link sorting, adaptive link hiding, adaptive link annotation, adaptive link generation and map adaptation (Brusilovsky, 2001). Table 2.1 summarises each of the aforementioned techniques.

Technique	**Description**
Direct guidance	The system visually points out the next 'best' links on the current page for the users to traverse based on their user models' values.
Adaptive sorting	Hyperlinks are sorted with the topmost items having the most relevance to users.
Adaptive hiding	Links to 'irrelevant page' is hided from a particular user to minimise user's information or cognitive overload.
Adaptive link annotation	Links to 'relevant pages' is annotated with some visual cues, for example using a specific colours, icons or font sizes.
Adaptive link generation	Additional links are inserted to an existing page to lead user to some additional useful or related information.
Map annotation	Altering the form of local or global hyperspace map presented to the user by using the aforementioned adaptive navigational support techniques.

Table 2. 1: Adaptive hypermedia techniques under the category of adaptive navigational support (Brusilovsky, 2001).

2.6 Adaptive Educational Hypermedia Systems: the Primary Problem

Brusilovsky described six kinds of hypermedia systems which are commonly used as application areas in most research projects on adaptive hypermedia, with educational hypermedia being the most popular area for adaptive hypermedia research (Brusilovsky, 1996).

When compared to a regular hypermedia system, an Adaptive Educational Hypermedia System (AEHS) is more complex to develop and maintain as it involves more specific stages as enumerated below (Brusilovsky, 2003):

i. Design and structure the knowledge space;
ii. Design a generic user model;
iii. Design a set of learning goals;
iv. Design and structure the hyperspace of educational materials;
v. Design connections between the knowledge space and the hyperspace of educational materials.

The complexity of authoring an AEHS is mainly influenced by the fact that a course author is often required to have a complete knowledge about all the possible output of the system before rules that define the alternative routes can be designed (Cristea et al., 2003). This is partly because most AEHSs deploy a deterministic form of adaptive hypertext design in a sense that an author always understands the navigational paths available to their user at any one time (Millard et al., 2003).

In addition to that, as there is yet no standardisation approach in authoring adaptive techniques and behaviours, "*everybody has to invent everything from scratch, and reuse is almost non-existence*" (Cristea et al., 2003). To address this issue, an European project called ADAPT has attempted to develop a common practice in defining adaptive behaviour by extracting adaptive patterns from several adaptive hypermedia systems

and then develop adaptation strategies that can be reused in authoring adaptive courseware (Cristea, 2003; Brown et al., 2005).

The inability to reuse existing authored adaptive content also leads to the laborious effort to develop and maintain an AEHS. The intertwined of adaptive logic with learning resources in some adaptive hypermedia systems make the reusability of the adaptive content problematic (Dagger et al., 2003). Simple insertion of new materials or deletion of the old ones might lead to the re-applying of adaptive rules for every page. However, the separation of logic from content is prevalent in many new generations of AEHSs such as JointZone (Ng, 2003), WHURLE (Web-based Hierarchical Universal Learning Environment) (Brailsford et al., 2002) and KOD (Knowledge-on-Demand) (Sampson et al., 2002). For instance, in WHURLE (Brailsford et al., 2002), the sequencing logic among the atomic units of learning resources is stored in an XML-file known as a 'Lesson Plan'. (Dagger et al., 2003) encapsulates the expert's knowledge of a domain and the adaptive logic in a 'narrative' model, separate from the content. On the other hand, in KOD (Knowledge on Demand), the IMS Content Packaging (IMS CP, 2003) manifest file is extended with the adaptive rules' definitions (Sampson et al., 2002). The encapsulation of adaptive rules' definitions within the common package permits reusability and interoperability of adaptive material between different KOD-compliant learning management systems.

2.7 Learning Style Adaptation: The Theory of Learning Style

When implementing learning style adaptation, it is necessary to take into account the research outcomes in cognitive sciences such as the theory of learning style. According to the theory, learners have their own preferable approach to perceive and process information while interacting with and responding to their learning environment (Kolb, 1984; Felder, 1996). Based on this theory, several learning style models have been developed. Among the most popular learning style models used in an adaptive hypermedia systems are Kolb's Learning Style Model (Kolb, 1984), Honey and Mumford Learning Style Model (Honey and Mumford, 1992), and Felder-Slivermann

Learning Style Model (Felder and Silvermann, 1988). These learning style models will be described in greater detail in the following sections

2.7.1 Kolb Learning Style Model

Kolb Learning Style Model (Kolb, 1984) classifies every learner as having a preference for 1) concrete experience or abstract conceptualisation (how they perceive information), and 2) active experimentation or reflective observation (how they process information). Four types of learners emerged from this classification: *Diverger* (concrete experience and reflective observation); *Assimilator* (abstract conceptualisation and reflective observation); *Converger* (abstract conceptualisation and active experimentation) and *Accommodator* (concrete experience and active experimentation). Table 2.2 briefly describes common characteristics portrayed by each type along with the recommended requirements of how to tackle their learning needs (Kolb, 1984; Felder, 1996).

Learner Type	**Learner Characteristics**
Diverger	A typical question of this learning type is "Why"; Respond well to explanations on how course material relate to their experience, their interests and their future careers; Typically a quiet thinker and stops periodically to review what has been learned before they proceed to new material; Regard their instructor as their motivator; Favour lectures with plenty of reflection time and likes to write short notes; Prefer to be assessed by external criteria; Their strength is their imaginative abilities.
Assimilator	A typical question of this learning type is "What"; Respond well to lectures where information is presented in an organised, and logical fashion; Like to discover the relationship between what has been learned with existing theories or concepts; Regard their instructor as their expert interpreter;

	Favour case studies, theories readings, analogies and lectures with some reflection time (thinking alone); Their strength is their ability to create theoretical models.
Converger	A typical question of this learning type is "How"; Respond well to having opportunities to practice the methods learned, works actively on well-defined tasks, and learns by trial-an-error in an environment that permits them to fail safely; Favour activities that apply skills (laboratories, field work), and provide peer feedbacks; Regard their instructor as a coach providing guided practice and feedback; Their strength lies in their ability to practically apply ideas.
Accommodator	A typical question of this learning type is "What if"; Like to apply course material in new situations to solve real problems; Favour activities that encourage exploration of applications like simulations, case studies/design projects, problem-solving in groups, and peers feedback; Regard their instructor as a model professional, leaving the learners to determine their own criteria for relevance materials; Their strength is their ability to do things and get involved in new experiences.

Table 2. 2: Kolb learner types and characteristics (Clark, 2000)

2.7.2 Felder-Silvermann Learning Style Model

Felder-Silvermann Learning Style Model (Felder and Silvermann, 1988) emphasises on finding a balance between the teaching strategies and the learners' learning styles, in order to accomplish effective learning. The model classifies an individual's preferred learning style by a sliding scale of four dimensions: *sensing-intuitive, visual-verbal, active-reflective* and *sequential-global*. Table 2.3 describes the characteristics of the learners for each dimension along with some suggestions of the requirements to accommodate them (Felder, 1996; Kinshuk and Lin, 2004).

Dimension	Learners Characteristics and Requirements
Active	Learn by trying things out, like to work with others; Provide discussion area; features for simulated experiments.
Reflector	Learn by thinking things through, stop periodically to review what has been learned, like to work alone; Provide note taking functions.
Sensing	Prefer concrete information, practical, oriented toward facts and procedures; Provide descriptions of physical phenomena, results from real and simulated experiments, demonstrations and problem-solving algorithms.
Intuitive	Learners can often handle conceptual information, innovative, oriented towards theories and meanings; Provide theories, mathematical models, and materials that emphasises fundamental understanding.
Visual	Learners prefer visual information like sketches, diagrams, pictures, flow chart, demonstration, concept maps, colour notes, slides with multimedia elements; Provide visual information and highlight functions in addition to oral and written explanations in lectures.
Verbal	Learners get more out of words heard and written; Provide written and spoken explanations (lectures/readings).
Sequential	Learners like to learn in small incremental steps that follow logically; Provide step-by-step learning format, related by simple and limited links and demonstrate the logical flow of an individual course topics.
Global	Learners like to jump in, absorb material nearly at random and then get a big picture; Provide course's overview and all the possible links relevant to the subject learned; Use graphics, illustrations, and analogies can help learner to map out new information.

Table 2. 3: Felder-Silvermann learning style dimensions and their requirements (Kinshuk and Lin, 2004)

2.7.3 Honey and Mumford Learning Style Model

While the previous two models are used effectively in engineering education (Felder, 1996) , Honey and Mumford Learning Style Model has been developed in an attempt to apply learning style theory in the context of business and management studies (Zwanenberg et al., 2000). Based around Kolb's theory, this model identifies four types of learners: *Reflector, Theorist, Pragmatist,* and *Activist*, as briefly described in Table 2.4. In fact, each type is often associated with Kolb's learner's types (i.e. Reflector with Diverger, Theorist with Assimilator, Pragmatist with Converger, and Activist with Accommodator).

Learner Type	**Learner Characteristics**
Reflector	Observe and describe processes, try to predict outcomes and focuses on reflecting and trying to understand meaning.
Theorist	Focus on ideas, logic and systematic planning, but are mistrustful of intuition and emotional involvement.
Pragmatist	Like practicality, down-to-earth approaches, group work, debate and risk-taking, but tend to avoid reflection and deep level of understanding.
Activist	Individuals who enjoy new experiences, are active, tend to make decisions intuitively, but dislike structured procedures.

Table 2. 4: Honey and Mumford learner types and characteristics (Zwanenberg et al., 2000)

2.8 Summary

This chapter gives the reader an overview of Adaptive Hypermedia technology and the main reasons that restrict its practical use in e-learning. The following chapter will discuss the fundamentals of Open Hypermedia technology, highlighting a common hypermedia data model called FOHM (Fundamental Open Hypertext Model). The

chapter will also introduce the pioneering effort in delivering Adaptive Hypermedia techniques from a structural perspective using Auld Linky.

3 Open Hypermedia

Chapter 3 begins with an overview to the background of Open Hypermedia research. This is then followed by the efforts from the Open Hypermedia System Working Research Group to develop a protocol to enable interoperability between different open hypermedia systems. As a result, an Open Hypermedia Protocol (OHP) was developed. This protocol is later called as OHP-Nav to better reflect its functions which covered mainly the requirements of a navigational hypertext model.

In the attempt to address the requirements of other hypertext domains such as spatial and taxonomy hypertext, a research group from the Southampton University has proposed an alternative data model to represent hypermedia structure found in these three hypertext domains. This model is known as Fundamental Open Hypertext Model or FOHM. A contextual link server named Auld Linky was then developed to implement FOHM. Auld Linky is used in this work as the adaptive engine for selecting appropriate links from the linkbase. This chapter, therefore, aims to provide reader with some understanding about FOHM and Auld Linky, especially their roles in implementing adaptive hypermedia techniques as demonstrated by a web-based adaptive hypermedia learning application named HA^{3}L (Hypermedia Adaptation using Agents and Auld Linky).

3.1 Overview

When Engelbart and Nelson had successfully demonstrated the implementation of hypertext envisioned by Vannevar Bush in 1945, many approaches had emerged to create a hypertext based system, and some of these approaches have resulted in the development of an open hypermedia system and the World Wide Web in the early nineties (Millard, 2000).

The objective of Open Hypermedia research is to make hypermedia rich structuring, navigation and annotation features (Vitali and Bieber, 1999) accessible across all applications on a user's desktop, and this meant a complete separation of link information from content (Davis et al., 1992). Link structures are then treated as 'first class' objects where they can be dynamically created, edited or removed from one or more link databases (better known as linkbases). A link service is later deployed to apply the updated links to the appropriate documents during run-time delivery.

The separation of link information from their documents offers several advantages to hypertext systems. First, the same document can be easily reused in different contexts without being tightly-coupled with the application, allowing the same document to have a different set of links for different types of users and/or for different learning contexts (Millard, 2000). Secondly, links can be followed on top of any media (e.g. text, sound, video, etc) and dynamic links can also be authored for a read-only medium such as a CD-ROM (Davis et al., 1992). Thirdly, hypertext systems can be easily maintained because link structures can be processed separately. This improves the system's ability to cope with a large numbers of documents, thus promotes system extensibility. Fourthly, authoring effort for a hypertext system can be reduced because existing content can be shared, reused or repurposed. Finally, hypermedia services can be provided to the client application via an open distributed architecture (Anderson, 1997) as demonstrated in the Distributed Link Service (Carr et al., 1998).

However, the aforementioned advantages offered by open hypermedia systems have been overshadowed by the popularity of the Web (Millard, 2000). The main reason the Web is more popular than open hypermedia systems is because it is globally distributed and accessible via any web browser. Its ability to offer easy remote access to Web documents using any browser lies on the facts that is applies standards in both its file format (Hypertext Transfer Markup Language) and networking protocol (Hypertext Transfer Protocol) (Millard, 2000). This interoperable capability cannot be offered by open hypermedia systems as the literature denotes that each open hypermedia system, including Microcosm (Hall et al., 1996), Chimera (Anderson et al., 1994) and DeVise Hypermedia (DHM) (Grønbæk and Trigg, 1994), have their own hypermedia data model, architecture framework and linking protocol (Østerbye and Wiil, 1996). A system independent framework named Flag taxonomy was purposely developed to compare, describe and classify different open hypermedia systems (Østerbye and Wiil, 1996).

In 1996, Open Hypermedia System Working Research Group has attempted to develop a standard open hypermedia protocol to enable different open hypermedia systems to communicate with each other (Davis et al., 1996). However, the proposed standard has been criticised for not taking into account other models of hypertext besides the navigational hypertext (Anderson et al., 1997, Millard et al., 2000).

In the attempt to address the requirements of other hypertext domains such as spatial and taxonomy hypertext, a research group from the Southampton University has proposed an alternative data model to represent hypermedia structure found in these three hypertext domains. This model is known as Fundamental Open Hypertext Model or FOHM (Millard, 2000; Millard et al., 2000). A lightweight contextual link server named Auld Linky was then developed by (Michaelides et al., 2001) in order to store and serve the hypermedia structures expressed in FOHM. Auld Linky enables a client application to query FOHM structured linkbase for the appropriate structure. Since its implementation, Auld Linky has been used to support a wide variety of hypermedia systems ranging from an augmented reality interaction system (Sinclair et al., 2002) to

an adaptive hypermedia application within an agent-based framework (Bailey et al., 2003). In the development of a Personalised Link Service, Auld Linky is used as the engine for retrieving adaptive links.

3.2 Hypertext Domains

In the hypertext environment, nodes of documents are interconnected using links. The way that nodes are arranged categorises hypertext systems into several domains. The most common hypertext domains are navigational, spatial and taxonomy hypertexts. Navigational hypertext is the most traditional domain of hypertext (Bush, 1945). In navigational hypertext, the authors create links between parts of documents that are related, and the users navigate the space by following these links. In spatial hypertext (Marshall and Shipman, 1995), visual symbols are used to create hyper-textual meaning wherein information nodes are clustered, coloured and sized according to their relationships with existing nodes, creating a sense of logical spaces. Users navigate this hypertext domain by exploring and traversing the spaces themselves. Finally, a taxonomy hypertext (Van Dyke Parunak, 1993) organises information objects into hierarchies forming taxonomies based on their characteristics and relationships with other objects. By specifying a particular perspective context onto the hierarchical structure, users can quickly view the requested set of information.

The common feature among these hypertext domains are the notions of data (node in navigational, visual in spatial, or artefact in taxonomic hypertext) and association (link in navigational, composites in spatial or category in taxonomic hypertext) (Millard et al., 2000).

There are also features that are unique to a particular domain. For instance, the navigational domain contains anchors that can point into documents rather than the entire documents. The spatial domain introduces type structures (e.g., a list, or a set) to the relationships. Finally, the taxonomic domain contains perspective objects that allow the relationship structures to diverge according to different views (Millard, 2000).

3.3 Open Hypermedia Protocol

Interoperability between hypertext systems has always been a concern in the Open Hypermedia community. In 1996, the first draft proposal of an Open Hypermedia Protocol (OHP) was developed to provide a lightweight communication mechanism between hypertext client programs and link servers (Davis et al., 1996). However, the draft did not address any other hypertext domain than navigational hypertext which makes interoperability among the domains seem impossible to achieve. As a result, the original OHP became OHP-NAV (OHP-Navigational Hypertext Interface) which better reflects its function to accommodate navigational hypertext domain.

OHP-NAV is then referred to as a simple protocol that expresses a navigational structure where the concepts of a *Node*, *DataRef*, *EndPoint* and *Link* are defined as follows (Bailey, 2002). A *Node* contains an information item of any media format. A *DataRef* is a pointer to a unique identifier of a data item, or a region within the data item. An *EndPoint* is a container for specifying the link's end attribute, and it usually contains direction attributes (source, destination, or bi-directional). Finally, a *Link* is a conceptual representation of association between information which is implicitly represented by several *DataRef*.

In order to successfully create a standard protocol for interoperability, the OHP-NAV model was re-examined so that a new model that would cover both taxonomic and spatial hypertext system could be built. This effort resulted in the development of FOHM, a data model for establishing a common language to consistently express all the hyper-structures found in these hypertext domains (Millard, 2000).

3.4 The Fundamental Open Hypertext Model (FOHM)

There are four core objects that constitute the FOHM model (Millard, 2000). Association objects are defined as structures that represent relationships between data objects or associations. Data objects become the wrappers for any piece of data that lies outside the scope of the model. Reference objects are used to point at (or into) data objects and associations. Finally, the binding objects are used to attach references to the association objects.

In addition to the core, context and behaviour objects are the additional objects which rationalize the implementation of the taxonomy hypertext domain. Context objects can be attached to any core objects and are usually followed by a behaviour object. Figure 3.1 illustrates the relationships among these objects.

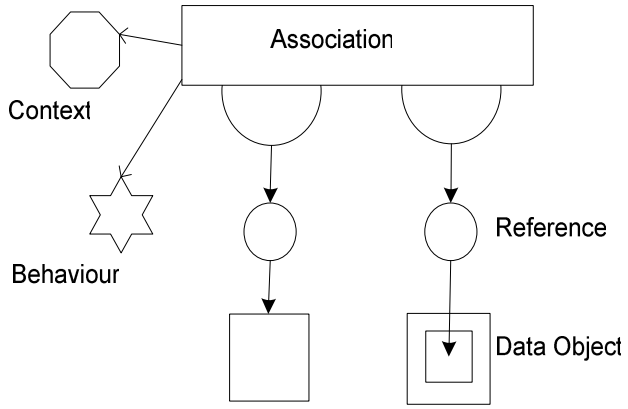

Figure 3. 1: The FOHM model

The fundamental element in FOHM structure is the association object (Millard, 2000). In fact it is designed to support any type of associational structure (e.g., Link or List). To represent a navigational link structure with FOHM, an association is assigned with a link structure and a set of endpoints as illustrated in Figure 3.2. Each binding must specify whether the attached reference is a 'source', 'destination', or 'bi-directional'

member. The attached context object will then determine the condition that permits the visibility of the association object or its elements.

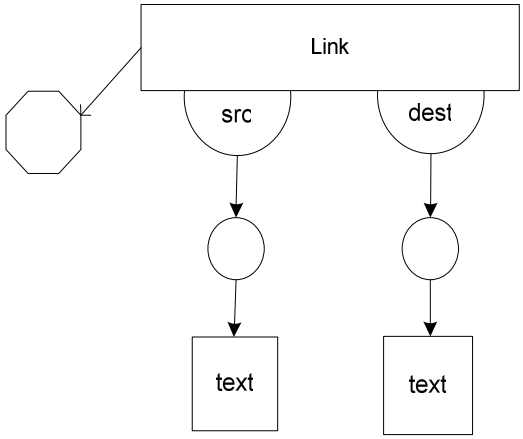

Figure 3. 2: A Navigational Link Structure in FOHM model

Bailey has extended the association structure to include *tour*, *Level-Of-Detail* and *concept* structures (Bailey, 2002). The extension is necessary in order to express other hypermedia structures (besides navigational link) that are often used in an adaptive hypermedia system. In (Bailey, 2002), the extended structures have been demonstrated to be capable of implementing almost all adaptive techniques present in the Brusilovsky's taxonomy (Brusilovsky, 2001).

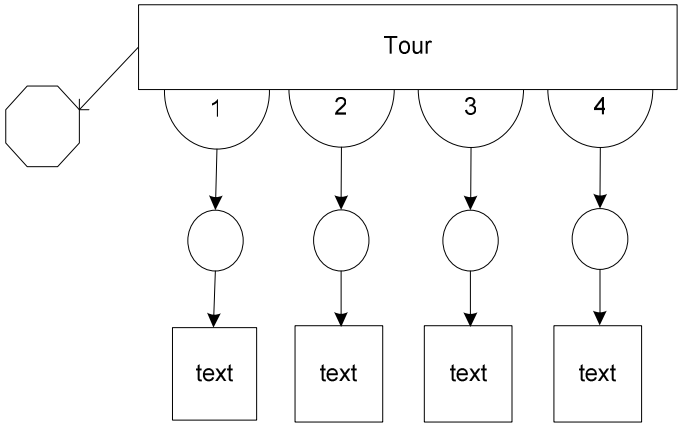

Figure 3. 3: A Tour structure in FOHM (Bailey, 2002)

An association with type *tour* represents a data structure that comprises a set of objects designed to be viewed in sequence. These objects might be data items representing a sequence of pages, association objects representing a sequence of substructures or a mixture of both (Bailey, 2002). The *tour* structure can enable the implementation of both adaptive navigational support (e.g. direct guidance) and adaptive content presentation (e.g. inserting or removing fragment). The former usually requires each data object in an association to represent a document or an URL of a web page. The latter will need every data object of an association to contain a specific fragment of a document, as depicted in Figure 3.3. In this case, the association will represent a document that is composed by several segments adaptively selected based on a user model (i.e. users' values matched the associated context objects).

An association with type *concept* is used to bundle a collection of data items composed of different media representations but carrying the same conceptual content, as illustrated in Figure 3.4. A specific data item can be selected adaptively by querying the concept using a particular user profile. The concept structure permits the implementation of adaptation modalities and fragment alteration techniques for an adaptive hypermedia system, in which, based on the associated context object, a particular media format represented by the data object will be selected to express a taught concept.

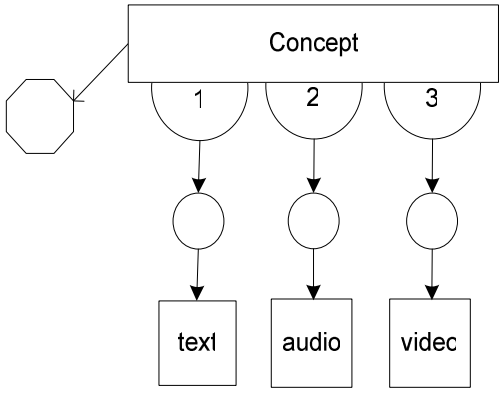

Figure 3. 4: A Concept structure in FOHM (Bailey, 2002)

The *Level-Of-Detail* structure is a list of ordered objects in which each object presents the same conceptual information with increasing detail or complexity (Bailey, 2002), as depicted in Figure 3.5. When this structure is queried in context, the members of the *Level-Of-Detail* will be filtered to display only the objects preferred by the current user. A *Level-Of-Detail* serves adaptive hypermedia systems by implementing techniques such as stretch-text whereby the information detail will be displayed depending on a user model.

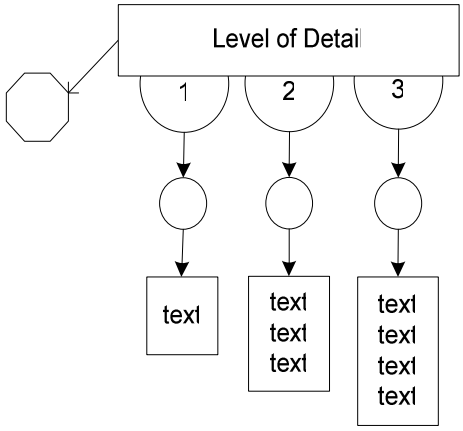

Figure 3. 5: A Level of Detail Structure in FOHM model (Bailey, 2002)

FOHM model aims to provide a common data model to represent hypermedia information, taking into account the navigational, spatial and taxonomic hypertext domains. Therefore, the FOHM model includes both the common and unique features of these hypertext domains.

To support the unique features of the navigation hypertext, FOHM provides a concept of linking to a reference object where it can point either at a data item or into the data item.

To incorporate the classification method used by the spatial hypertext systems, FOHM introduces the notion of a structure type and feature space for each association object. The structure type indicates whether an association represents a collection of links or other spatial values such as a List, Map or Matrix (Millard, 2000). Feature space, on the other hand, composed of '*featurevalue*' attributes that represent the spatial aspects of a

reference object. For instance, if a feature space is associated with direction of links, the '*featurevalue*' will indicate the direction type of the data object it referred to whether it is a '*source*', a '*destination*', or a '*bi-directional*' type.

Finally, for accommodating the requirement of the taxonomic hypertext domain, FOHM introduces the notion of context objects. This context object can be attached to any FOHM objects to provide different views on the structure according to a chosen perspective (Bailey, 2002). In FOHM, context objects are usually paired with behaviour objects. The context objects specify the conditions for the visibility of certain objects; whilst, the behaviour objects inform the client applications handling the FOHM structures of actions to perform given certain event conditions (e.g. on display) (Bailey, 2002).

3.5 Auld Linky

Auld Linky (Michaelides et al., 2001) is a contextual open hypermedia FOHM server that was developed to implement FOHM structures expressed in XML. Auld Linky provides APIs for storing, looking up and querying the FOHM structures in the linkbase. Auld Linky link service was designed to satisfy the following criteria: simple to install and execute (lightweight); operates over a simple HTTP connection (e.g. GET, POST, PUT, DELETE); and based upon the XML.

In the POST operation, requests for querying the linkbase are sent with a FOHM structure in the content of the querying message (XML-message) (Michaelides et al., 2001). When Auld Linky receives the XML message, it will convert it into an internal FOHM structure. The FOHM structure are then pattern-matched against each FOHM structure that are stored in the linkbase. The pattern matching process will produce a set of matches that later will be converted to XML and returned to the client. These querying processes can span over more than one linkbase

Auld Linky also permits querying in context (Michaelides et al., 2001). There are two ways to use context objects for querying Auld Linky. First is to attach the context object to the FOHM structures sent in the queries; these structures then only match if their contexts match as well. Secondly, only context objects are attached to the queries themselves. In this case the context object acts like a filter on the query results. The FOHM objects that are returned from the query matching are compared to the supplied query context, any association and data item with attached context that do not match the query context will be removed from the result set.

3.6 Implementing Both Open Hypermedia and Adaptive Hypermedia Technologies in HA³L

HA³L (Hypermedia Adaptation Using Agents and Auld Linky) system is developed to incorporate Auld Linky into an adaptation agent environment which is based on the SoFAR (Southampton Framework Architecture) agent framework (Bailey, 2002). By using Auld Linky, HA³L demonstrates the use of FOHM structures and the first use of Auld Linky in supporting adaptive hypermedia techniques for delivering a course on 'Rheumatology' to degree-level medical students.

In HA³L, context and behaviour objects play a crucial role. The context object determined which structure should be returned to a user, and the behaviour object was used to modify the user profile when a particular event was accomplished. For instance, when a page is displayed to a user, the '*on display*' event (i.e. an event of a behaviour object) will be activated and a flag is set in a user profile stating that the information in the data item has been read (Bailey et al., 2002). Table 3.1 provides a summary of some adaptive hypermedia techniques implemented in HA³L using the FOHM structures described earlier.

FOHM Structure	**Adaptive Hypermedia Techniques**
Navigational Link	Link Hiding A context object is attached to the structure to specify that the link will only visible to a particular user profile, e.g. an expert. Hence, when a novice queries the system for links, the link will be hidden.
Tour	Inserting or Removing Fragments Data objects within a tour structure are assigned to individual fragments. When a client queries Auld Linky to extract this structure, any sub-structures which context does not match the user profile will be removed before the structure is returned to the client. The client inserts the remaining sub-structures, in the order that they appear in the tour structure, into the document.
Concept	Adaptation Modality Data objects within a concept structure are assigned to different media representations of the same information. When a client queries Auld Linky to extract this structure, only those data objects that fit the query profile (or user model) will be returned to the user.
Level of Detail	Stretch-text Data objects within this structure must represent text at increasingly advanced levels. Initially the lowest positioned data object in the Level-Of-Detail would be displayed. When a user has completed a text it could be replaced with the next available data object in the Level-Of-Detail.

Table 3. 1: Description on methods for implementing adaptive techniques using FOHM structure, derived from (Bailey et. al. 2002).

3.7 Summary

This chapter provides the background knowledge to Open Hypermedia technology emphasising FOHM as a data model that represents a fundamental structure of hypermedia information. The chapter also highlights the use of FOHM and Auld Linky in delivering adaptive hypermedia techniques via an agent-based adaptive hypermedia system named HA^{3}L.

The following chapter will bring forward another technology involved in this work i.e. Learning Object Technology. The main focus of the chapter is to introduce reader to SCORM (Sharable Content Object Reference Model) (SCORM, 2002) which provides the reference model for standardising the development and use of learning objects across several learning systems.

4 Learning Object Technology

Chapter 4 introduces readers to Learning Object technology and to the world of e-learning standards and specifications. The introduction begins by establishing some understanding about what Learning Objects are and why there is an enormous need for them. Current related e-learning standards and specifications are described with emphasis on SCORM (Sharable Content Object Reference Model) (SCORM, 2002). This is then followed by reviewing SCORM's sequencing mechanism which is based on the IMS Simple Sequencing specification. The chapter ends by reporting the differences between Simple Sequencing and Adaptive Hypermedia.

4.1 Learning Objects

Generally Learning Objects can take many electronic forms ranging from files containing static descriptions of content (e.g. HTML pages, PDF, files containing text and graphics, Power Point presentations, etc.) to more sophisticated interactive displays (e.g. HTML pages with JavaScript, Java applets, etc). Between this explicit categorisation there exist other kinds of Learning Objects such as audio files, video clips, Flash animations, etc (Mohan and Greer, 2003).

However, the exact definition of a Learning Object still varies in the literature. IEEE broadly defines a Learning Object as being "*any entity digital or non-digital that may be used for learning, education and training*" (IEEE LOM, 2002). Likewise Wiley defines a Learning Object as "*any entity resource that can be used to support learning*" (Wiley, 2000). A more specific definition was drawn by Hodgins, who believes that a Learning Object should be "*a collection of information objects assembled using metadata to match the personality and needs of the individual learner*" (Hodgins, 2000). Whereas Mohan and Greer prefer to define "*a Learning Object to be a digital learning resource that facilitates a single Learning Objective and which may be reused in a different context*" (Mohan and Greer, 2003). Finally Heng pointed out that e-learning vendors usually define Learning Objects more specifically for more useful implementation (Heng, 2003). For instance, Macromedia's concept of e-Learning Object is a unit of instructionally sound content centred on a Learning Objective or outcome intended to teach a focused concept (Hines and Himes, 2002).

While there are many interpretations, most people would agree that a Learning Object should have the following characteristics (Millar, 2002):

- Should be smaller than a course.
- Should be self contained i.e. it can be used independently of other Learning Objects.
- Should be tagged with standard metadata for easy retrieval.
- Could be reused and repurposed i.e. the same Learning Objects can be used in multiple contexts or multiple purposes.
- Could be aggregated to compose a higher-level unit of instruction.

Rehak and Mason further add that in spite of so much variation about what constitutes a Learning Object, there is a substantial agreement about its attributes i.e. they must be (Rehak and Mason, 2003):

- Reusable – they can be modified and versioned for different courses.
- Portable – they can be operated across different hardware and software.
- Accessible – they can be indexed for easy retrieval using metadata standards.

- Durable – they can remain intact through upgrades to the hardware and software.

In fact, Mohan and Greer believe that the driving force behind the development of Learning Object technologies is the promise of content reusability (Mohan and Greer, 2003). This is because when a Learning Object encapsulates a chunk of reusable content, a course or a lesson can easily be generated by just assembling smaller granular Learning Objects. Hence, the ability to reuse can reduce the amount of time and cost in preparing course content whereby high quality materials can be prepared once and used many times by many people in different contexts. Clark and Rossett quote the following people by saying (Clark and Rossett, 2002):

- Cisco's Chunk Barritt said, Learning Objects "*result in shortened development time when updating existing objects or modifying them for a new audience*."
- Joe Jurzyeck of LOBJ.org as saying "*the primary benefit coming from different instructors using the same materials in several contexts*."

The benefits of using a Learning Object approach in course development have been listed by (Longmire, 2000). Longmire believes that Learning Objects provide:

- content flexibility if they are designed to be used in multiple contexts;
- easy and quick content management because metadata tags can facilitate rapid searching, and allow selection of only the relevant content for the purpose;
- a 'Just-In-Time' approach for content customisation to suit individual and organisational requirements;
- content interoperability across disparate systems when Learning Objects are designed based on the recommended specifications and standards;
- competency-based learning as each Learning Object can be specifically tagged to permit adaptive matching between object metadata with individual competency level;
- incremental value of content as the value of content is increased every time it is reused. This is reflected not only in costs saved by circumventing new design

and development time, but also in the possibility of selling content objects or providing them to partners in more than one context.

However, for Learning Objects to become more widely used in educational practice, two things must occur according to (Millar, 2002). First, a wide variety of Learning Objects must become readily available. Secondly, educators need to feel comfortable about using Learning Objects and they also need to become adept at searching and creating Learning Objects.

4.2 Learning Objects Taxonomy

According to Duval and Hodgins, the IEEE LOM broad definition on Learning Objects permits content components of all sizes to be defined as Learning Objects, either it be a picture of Mona Lisa, a web document on Mona Lisa, a course module on da Vinci, or even a full course on art history (Duval and Hodgins, 2003). Nonetheless, the broad and vague definition is also problematic especially when the process of authoring, deploying and repurposing are affected by the granularity of the Learning Objects. For instance, a large sized Learning Object needs less description and can easily be mapped to a pre-designed curriculum, yet it is difficult to fine tune or reuse it in other contexts. Conversely, a small Learning Object is easy to repurpose, reuse and re-contextualise but difficult to describe and defined.

In order to provide a clear concept of what a Learning Object is, Duval and Hodgins have proposed a Learning Object taxonomy that describes different kinds of Learning Objects and their components (Duval and Hodgins, 2003).

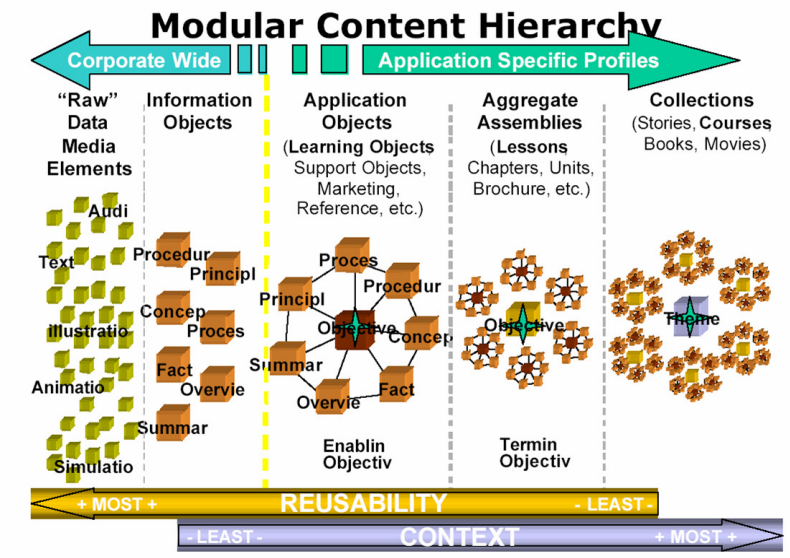

Figure 4. 1: Learning Objects Taxonomy (Duval and Hodgins, 2003)

As illustrated in Figure 4.1, the taxonomy denotes raw media elements as the smallest entities in the model, for instance, a picture, a single sentence or paragraph, a video etc (*1st level*). A collection of several raw elements will compose information objects like a concept, summary, procedure, process, overview etc (*2nd level*). Based on an objective, information objects are selected and assembled into application objects (*3rd level*). Then based on a larger objective, the application objects are assembled to form lessons, chapters and so forth (*4th level*). The assembling processes continue to assemble aggregation results into larger collections, like courses, or whole curricula.

4.3 The Need for a Common Object Model

A Learning Object that is developed based on a common object model can be easily reused and assembled with other Learning Objects to generate another new learning experience (Slosser, 2002). Nonetheless, reusability occurs only when content developed in one context is transferable to a different context which may involve different repositories or learning systems, making interoperability the key to reusability (Heng, 2003).

Therefore, one focus of Learning Object technology is to provide the infrastructure to commonly describe Learning Objects so that different Learning Objects created by different authors in different organisations can be found, extracted and aggregated to accommodate a new context or need (Rehak and Mason, 2003).

To accomplish this aim, the Learning Object community has initiated the effort to develop common models and methods for defining and using Learning Objects via the development of standards and specifications in e-learning. The standards and specifications developed will be described in detail in the following section.

4.4 Standards and Specifications

Learning technology standards and specifications are developed to facilitate the description, packaging, sequencing and delivery of educational content, learning activities and learner information (Campbell, 2002). Campbell further reported that only through standards and specifications, can content be prevented from being 'locked in' to proprietary systems. As a result, educational content and learner information can be shared and interoperability can be achieved between different systems. This can be accomplished because learning standards bring an agreement on types of data models that should be used in e-learning to allow communication between many systems. Also by using standards or common specifications, effort required to prepare high quality learning materials can be minimised. This is because the learning materials can be developed once by an expert and used many times by others regardless of their learning contexts or environments.

Learning standards evolve through a series of well-documented specifications developed by several consortia like IEEE LTSC (Learning Technology Standard Committee), AICC (AICC, 2000), ARIADNE (ARIADNE, 2004), and IMS Global Learning Consortium (IMS, 2003). These technical specifications later evolved into reference models and application profiles. Only after a thorough testing and validation processes, the specification can be advanced to the accredited organisations like ISO

(International Standards Organisation) that will turn them into standards. Is it worth mentioning that until today, only IEEE LTSC LOM (IEEE LOM, 2002) specification has been accredited as e-learning resource standard by the ISO.

The IEEE LOM defines a common way to describe, characterize and classify a Learning Object so it can be searched and retrieved in a consistent way. IEEE LOM schema defines a structure that is divided into nine categories (Slosser, 2002): *General* (describes the resource as a whole); *Life Cycle* (describes the history and current state of the resource); *Meta-metadata* (information about the metadata record itself); *Technical* (technical requirements and characteristics); *Educational* (educational and pedagogic characteristics); *Rights* (intellectual property rights and conditions of use for the recourse); *Relation* (relationship between the resource and others); *Annotation* (comments on educational use of the resources and on when they were made and by whom); and *Classification* (labels the resource in a classification system). Each category comprises of a group of data elements which contains sub-elements.

There are also several specifications to described a user profile, for example, IMS Learning Information Package (IMS LIP, 2000) and PAPI (IEEE Personal and Private Information) (PAPI, 2004). These specifications allow information about learners to be shared and reused among different systems.

Next is the IMS Content Packaging specification (IMS CP, 2003). This specification enables Learning Objects for an individual course or collection of courses to be packaged into an interoperable package (known as a 'Package Interchanged File'). By having a common packaging method, content can be transported across different applications and learning management systems. An IMS Content Package can contain Learning Objects at different levels of granularity for instance it can contain part of a course, an entire course, or several courses that are part of a curriculum (Mohan and Greer, 2003).

In order to deliver meaningful learning, Learning Objects need to be sequenced according to some instructional strategies. A specification named IMS Simple Sequencing (IMS SS, 2002a) can accommodate this requirement. In addition to that, it can ensure a consistent sequencing of Learning Objects across different learning management systems.

In mid 1999, ADL emerged with a unified reference model known as SCORM that adopts most of the aforementioned standard and specifications to produce a comprehensive suite of e-learning capability. Therefore, by adopting SCORM, developers are guided in their efforts to apply a collection of e-learning industry standard and specifications for describing, organising and delivering e-learning contents across different learning management systems.

4.5 The Sharable Content Object Reference Model (SCORM)

The primary aim of SCORM is to define a learning environment that enables interoperability, accessibility, reusability and durability of Web-based learning materials with any learning tools (SCORM, 2002). SCORM is probably an important step towards the development of interoperable e-learning systems as it provides the means to address compatibility issues in the current e-learning environment (Arapi et al., 2003). SCORM comprises of the Content Aggregation Model (CAM), and the SCORM Run-Time Environment (RTE) (SCORM, 2002).

4.5.1 The Content Aggregation Model

SCORM CAM defines how learning resources can be described with metadata, how they are organised and packaged, and moved between different learning management systems or content repositories (Qu and Nejdl, 2002). In the SCORM specification version 1.3 (SCORM, 2002), the CAM is made up of four parts: a Content Model (from AICC), Meta-data (from IEEE), Content Packaging and Sequencing (from the IMS).

The SCORM content model allows content components to be built based on a common structure so that they can be reassembled, reused and repurposed more easily. In the current SCORM content model, four models of Learning Objects have been defined: Assets, Sharable Content Assets (SCA), Sharable Content Objects (SCO), and Content Aggregations.

Assets are the basic form of digital learning content such as a web page, an image file, an audio file, XML document, a Flash object and etc.

The aggregation of Assets constructs a SCA or a SCO. Even though both units can be launched by a learning management system, it is only SCO that can communicate with the learning management system using the SCORM RTE Communications API. This permits the learning management system to track learner's interactions with a SCO using the SCORM RTE Data Model (SCORM, 2002). Therefore, every SCO must adhere to the SCORM RTE's requirement. That is, a SCO must contain at least two API calls: *LMSInitialize* and *LMSFinish* (SCORM RTE, 2004). *LMSInitialize* allows the learning management system to initialize a SCO so that the SCO can call other API functions, whereas *LMSFinish* terminates any communication between SCO and the learning management system.

The final component in the SCORM Content Model is a Content Aggregation. Content Aggregation is a process of pooling resources (Assets/SCAs/SCOs) into a defined content structure in order to build a particular learning experience (SCORM, 2002). Within this structure, sequencing rules are used to define the sequencing of the learning resources. The learning management system is then responsible for interpreting the intended sequence described in the content structure and controlling the actual sequencing of the learning resources at run-time (SCORM, 2002). This content structure is described in a XML-based file called a '*manifest*' in a SCORM Content Package.

Besides a manifest file, a SCORM Content Package which is based on the IMS Content Packaging (IMS CP, 2003) also contains the related physical resources and IMS schema

files. By applying a common structure for packaging course content, content can be easily moved between any SCORM-compliant learning environments (Slosser, 2002). In SCORM version 1.3, the static content structure is enhanced with sequencing and navigation behaviour by applying elements from the IMS Simple Sequencing specification. The sequencing is based on some instructional design strategies. The integration of the IMS Simple Sequencing enables SCORM Content Aggregation Model to support consistent sequencing of learning activities (Bouras et al., 2003).

The SCORM Meta-data components represent a mapping and recommended usage of the IEEE LTSC LOM elements for each of the SCORM Content Model elements (SCORM, 2002). Content Aggregation Meta-data describes the content aggregation. Activity Meta-data describes the individual activities found in a content package. SCO/SCA Meta-data is applied to SCOs/SCAs to provide the descriptive information about the content represented in the SCO/SCA. Finally Asset Meta-data is applied to "*raw media*". The mechanism used for binding each resource to their respective metadata is the SCORM Content Package (IMS CP, 2003).

4.5.2 The SCORM Run-Time Environment (SCORM RTE)

The SCORM approach leverages the e-learning commercial standards and practices towards Web-like architectures by specifying a Web-based run-time environment model for a learning management system (Slosser, 2002). The goal of SCORM RTE model is to detail the requirements for launching content objects (SCOs, SCAs or Assets), establishing communication between learning management systems and SCOs, and managing the tracking information that can be communicated between SCOs and learning management systems (SCORM RTE, 2004). When e-learning developers adopt this model, a common method for a learning management system to launch and manage content objects and for the content to communicate with a learning management system can therefore be achieved (SCORM RTE, 2004; Slosser, 2002). As a

result, content objects can be reused and be interoperable across multiple learning management systems (SCORM RTE, 2004).

There are three important aspects of SCORM RTE i.e. a Launch mechanism, a Data Model and an Application Program Interface (API) (SCORM RTE, 2004).

The Launch mechanism defines a common way for learning management systems to start Web-based content object (SCOs, SCAs and Assets), and for the content (SCOs) to establish communications with the initiating learning management system (Slosser, 2002; SCORM RTE, 2004). As such, it determines which content object will be delivered next to the learner by the learning management system (Bouras et al., 2003). The decision is made by referring to the current value or status of the SCORM Data Model elements, as well as the result from the sequencing engine.

SCORM Data Model is derived from the AICC data model (AICC, 2000) with some extensions in order to accommodate IMS Simple Sequencing specification. The role of the SCORM Data Model is to provide a defined set of data elements about SCOs, the students and the run-time session so that they can be tracked by learning management systems to enable information exchange between the systems and the SCOs (Slosser, 2002).

SCORM Data Model elements are: *Launch Data* (information used by SCO during the launch), *Suspend Data* (stored SCO data from previous run), *Student Preference* (SCO options that can be selected by the user), *Student Data* (SCO customisation information based on student performance), *Objectives* (information on SCOs' objectives), *Interactions* (recognised student input to the SCO), *Core* (basic information about the students and the lesson), and *Comments* (information exchange and feedback between the SCO and the learning management system) (SCORM RTE, 2004; Slosser, 2002). These data elements are usually optional for SCOs but some are mandatory for the learning management systems. Additionally, some of the Data Model elements relate to the Tracking Status Data Model for the respective activity to permit consistent

sequencing of contents across different learning management systems (Bouras et al., 2003).

The SCORM API (a set of defined functions) defines a common communication mechanism that allows a SCO to communicate with the learning management systems (SCORM RTE, 2004). A SCORM API is used to permit the exchange of SCORM Data Model elements between SCOs and the learning management system, for instance, by using *GetValue* or *SetValue* functions. The API relieves developers from being concerned with the details of communicating with the learning management systems. "*It is like a steering wheel of a car where you can drive a car without the need to know about how the engine works*" (Slosser, 2002). It is noteworthy that, all communication between the SCO and learning management system is initiated by the SCO (SCORM RTE, 2004). SCO initiates and terminates a communication session by using *LMSInitialized* and *LMSFinish* functions, making both functions the mandatory parts of the API. Other related functions such as *LMSSetValue*, *LMSGetValue*, and *LMSCommit* allow data about a particular SCO to be retrieved from a learning management system or recorded into the system (SCORM RTE, 2004; Slosser, 2002).

4.6 The SCORM Sequencing Mechanism

The SCORM course sequencing mechanism is based on the IMS Simple Sequencing specification (IMS SS, 2002a). This specification provides a model of learning activities organised as a tree of clusters of learning activities (activity tree). A cluster is a node with its immediate child nodes. Each node in a cluster can be identified using an item identifier and only the leaf nodes contain references to the learning resources. Also attached to each node is a set of tracking data elements (Objective Progress, Activity and Attempt Progress Information) to monitor a user's progress in the activity tree.

The spatial arrangement of the activity tree determines a default sequencing path of the resources which is based on the pre-order traversal method. However, this path can be altered by applying sequencing rules at each parent node in order to control the

availability of the learning activities within its cluster, hence determining the sequencing outcome of the learning activities to a user. This sequencing information is kept in the organisation element of a manifest file, separate from the content files.

By integrating the IMS SS specification, SCORM enables course designers to translate learning strategies into sequencing rules (Bouras et al., 2003). The overall sequencing process will be controlled by a sequencing engine based on the sequencing information from the manifest file and the other related data models. The sequencing engine is provided by learning management system (based on a SCORM RTE) to ensure that the sequencing of the learning activities conforms to the instructional strategy and consistent across different learning management systems (Bouras et al., 2003).

4.6.1 Sequencing Rules

A sequencing rule determines the availability of the learning activities a learner is allowed to experience and the relative sequence of the activities (IMS SS, 2002b). The rule is defined using the following syntax: *if <condition> then <action>*. That is, if a condition is satisfied, the associated action will be executed.

The condition parameter will check a learner's progress and performance for a current activity. For instance, if the learners do not succeed in the learning objectives that are associated with the current activity, they will be prompted to experience additional learning activities that will help them to succeed (Bouras et al., 2003).

The action parameter provides the intended action for the learning management system during the sequencing process. Usually the action is divided into three steps that must be evaluated sequentially, which are *pre-condition* (applied when traversing the activity tree to select an activity for delivery), *post-condition* (applied when an attempt on an activity terminates) and *exit action* (applied after a descendent activity's attempt terminates) (IMS SS, 2002b).

Normally the sequencing rules are evaluated at the parent nodes for each cluster of the activity tree. Within the cluster, a set of rollup rules specifies how the values of children's activities influence the condition status of their parent. The sequencing rule then evaluates those conditions to determine whether the system should or should not deliver the learning resources available within that cluster to a user.

4.6.2 Rollup Rules

The rollup rules define the process of evaluating the tracking data values for the children's activities within a particular cluster to determine the corresponding values for its parent, as illustrated in Figure 4.2 (SCORM, 2002). The evaluation of the rollup rules may alter the default sequencing path, thus affect the availability of the activity and/or its sub-activities for delivery.

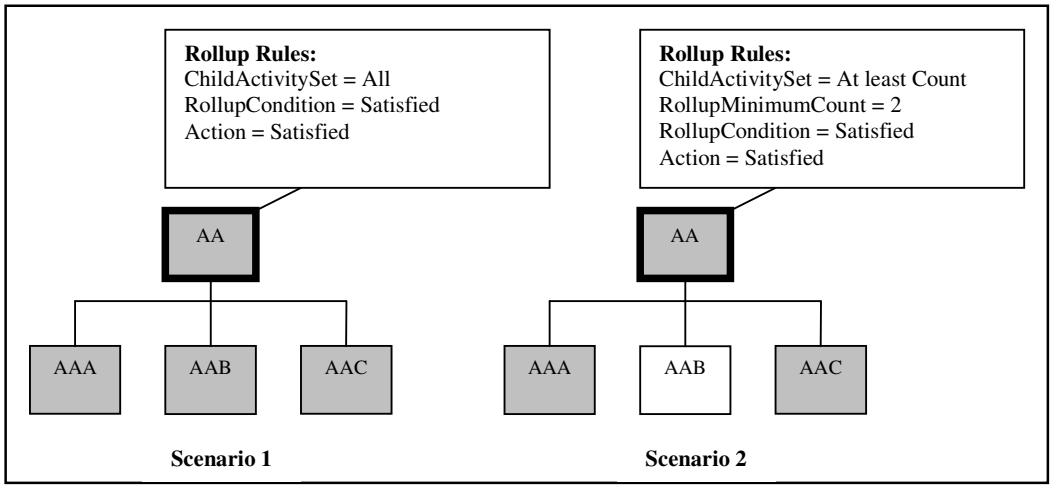

Figure 4. 2: Rollup rule illustration (SCORM, 2002)

As referred to the above diagram, notice that in the first scenario, the rule defines a condition that states if "*All*" of AA's children are "*Satisfied*" then AA should be considered "*Satisfied*". The second scenario states that if at least two of activity AA's children are "*Satisfied*" than AA should be considered "*Satisfied*" (SCORM, 2002).

Outcomes from the rollup will determine the current rule condition of the parent so that appropriate action will be applied to it by the sequencing rule.

4.6.3 Global and Local Objectives

In the IMS SS specification, Learning Objectives are defined separately from the Learning Object. They can either be declared as local or global variables, each with a satisfaction status and measure. While a local variable is accessible within its cluster, a global variable allows the learning management system to share status values between each cluster.

The objective usually represents a user's performance in learning a concept. Normally its value is obtained either by completing a test or by attempting an activity. In some instructional strategies, for example, in a competency-based strategy, both the objectives are used to represent user's knowledge level for a particular module within a course. Consider the following scenario. John has successfully obtained 80% in a pre-test. Assume that this result conveys information that John has satisfied Learning Objectives associated with all modules except the advanced module. Therefore he is able to skip modules catering for a beginner and intermediate user, as demonstrated in the following segment code.

```
<item identifier="BeginnerModule" identifierref="RESOURCE2">
  <title>Lesson 1 - Programming Principle</title>
</item>
<imsss:sequencing>
  <imsss:controlMode choice="false" flow="true"/>
    <imsss:sequencingRules>
       <imsss:preConditionRule>
         <imsss:ruleCondition condition ="objectiveMeasureGreaterThan"
      referencedObjective="PRIMARYOBJ"/>
             measureThreshold="0.75"
         </imsss:ruleCondition>
        <imsss:ruleAction action="skip"/>
       </imsss:preConditionRule>
   </imsss:sequencingRules>
   <imsss:objectives>
     <imsss:primaryObjective objectiveID = "PRIMARYOBJ">
       <imsss:mapInfo targetObjectiveID="obj_module_1"
```

```
                    readNormalizedMeasure = "true"/>
      </imsss:primaryObjective>
    </imsss:objective>
</imsss:sequencing>
```

The execution of the 'global and local objectives' concept can be shown by referring to the above example. *objectiveMeasureGreaterThan* is the rule condition and *PRIMARYOBJ* is the referenced local objective. Since *readNormalizedMeasure* parameter is assigned to *true*, value stored in the global objective i.e. *obj_module_1* (which is obtained from the pretest i.e. 80%) is copied into the local objective (i.e. *PRIMARYOBJ*). As a result *PRIMARYOBJ*'s value is now equivalent to 0.8 which is more than the stated *measureThreshold* (0.75). Therefore the evaluation of the rule condition (*PRIMARYOBJ* > *measureThreshold*) becomes *true*. Consequently, the associated rule action (*skip*) is executed. Meaning that, this *BeginnerModule* will be excluded from John's navigation sequence.

4.7 IMS Simple Sequencing and Adaptive Hypermedia: The Differences

Adaptive Hypermedia and IMS Simple Sequencing (IMS SS, 2002a) are two distinct pedagogical approaches for selecting what materials will be presented to learners. In the Hypertext Conference 2003, the author has published a study on these differences. The remainder of this chapter will represent a version of this publication (Abdullah and Davis, 2003).

An Adaptive Hypermedia System employs a user model, a model of the concepts in the learning domain and an adaptation model in order to decide what content and navigational features to display, and how to present that content. The user model may be instantiated by some prior understanding of the user (e.g. obtained by a pre-test) and will then be dynamically updated as the system monitors the user's behaviour.

IMS Simple Sequencing provides a learner with a sequence of learning activities to guide them through a learning space in order to achieve a specific Learning Objective. The sequencing is based on instructional design strategies selected by the course's instructor and are expressed in XML in the 'manifest file'. The learning management system maintains a simple user model based on the Learning Objects the user has visited and on intermediate test results. Based on this information the learning management system will decide what Learning Object should be next in the sequence.

On the face of it these two approaches appear similar; both aim to deliver the appropriate material to enable the user to achieve their Learning Objective as quickly as possible. This section explores the differences and concludes by considering what each community might learn from the other.

4.7.1 The Objective of the Approach

In general the objective of IMS Simple Sequencing is to ensure that a learner completes all the activities that an instructor deems important, while avoiding those that may be unnecessary, as demonstrated by prior knowledge or by swift grasp of the material; the objective is instructor centred. In contrast the objective of an adaptive hypermedia system is to assist the learner in navigating a complex information space in order to achieve whatever goal they choose; Adaptive Hypermedia is user centred.

4.7.2 Constituent Components

According to the AHAM reference model (Wu et al., 1998), an Adaptive Hypermedia System should consist of a domain model, a user model, an adaptation model, and an adaptive engine. The domain model describes how the information is structured and linked together using concept relationships. The user model describes a set of user's characteristics and preferences in browsing the hypermedia space and the adaptation model consists of pedagogical rules which define how the domain model relates to the user model to specify adaptation. The adaptive engine is responsible for generating the

actual adaptation outcomes by manipulating link anchors or fragments of the pages' content before sending the adapted pages to a browser.

On the other hand, IMS Simple Sequencing does not have either an explicit domain model or an explicit user model. Instead the IMS Simple Sequencing specification describes three data models: the Sequencing Definition Model which states the rules for progressing through the Learning Objects, the Tracking Model which holds the results of the learners' interactions with the content, and the Activity State Model which records the current activity or activity to return to if the learner has temporarily quit. Clearly there is a user model implicit in the tracking model, but it is simplistic when compared to the models employed by Adaptive Hypermedia Systems, recording only progress and having no interest in user preferences or learning styles. Instead of a domain model, there are sequencing rules, which express the pedagogical strategy of the instructor/author, rather than any generalised concept map of the subject domain.

4.7.3 Conceptual Structures

In many Adaptive Hypermedia Systems each page is normally associated with a concept. Since the concepts are usually represented in a form of a hierarchy, values attributed to sub-concepts may contribute to the same attribute value in their parent concept via propagation rules. For example, interest in Manchester United may suggest an interest in football. For each page adaptive rules will decide, based on the information in the user model, whether links to a page should be shown as desirable.

In IMS SS, learning activities are managed and arranged in the form of an activity tree and each node represents a learning activity. The root of the tree is the main activity, while the branches represent the sub-activities. Navigation through the activity space is based on a pre-order traversal method. The sequencing rules will decide whether a particular node will be traversed or skipped in the learning process based on the status or values obtained from the Tracking Model and Activity State Model.

4.7.4 Techniques Employed

Adaptive Hypermedia Systems use several adaptation techniques to provide adaptation. Currently, there are three major categories for adaptation techniques: adaptive navigation support, adaptive content presentation and adaptive content selection. Under the adaptive navigation support category, there are several adaptive techniques such as adaptive link sorting, adaptive link hiding, adaptive link annotation, map adaptation and direct guidance. In adaptive content presentation, adaptive techniques such as adaptive text presentation and adaptive multimedia presentation are used. Techniques like conditional-inclusion of fragments are employed by AHA! (DeBra and Calvi, 1998) to provide additional explanation when requested.

On the contrary, IMS Simple Sequencing controls the sequencing processes by using sequencing and rollup rules. Sequencing rules provide a means to describe conditional sequencing behaviour for an individual activity in the activity tree; rollup rules specify how the values of a child's activities influence the objective of the parent.

4.7.5 Tracking Mechanisms

In many Adaptive Hypermedia Systems, learners' browsing behaviour is continuously tracked by the adaptive engine in the form of log entries in the user model. For each user page access, a log entry is created which records the fact that the page has been visited and for how long. The adaptive engine will also monitor which concept's attributes are updated by rule actions, and will trigger the execution of their associated rules.

On the other hand, the tracking mechanism in IMS Simple Sequencing is controlled by its Tracking Model and Activity State Model. Results of learner's interactions with the learning activity are tracked, kept and updated in the Tracking Model in order to control the selection and sequencing of other activities. The sequencing and delivery

process use the tracking data to determine which activity should be sequenced next and which associated contents need to be delivered to the learner.

4.7.6 Re-usability

There are two main factors involved in ensuring reusability and interoperability of content across any Web-based learning management system: first, adaptive-related rules and functions must be separated from their contents; secondly, content should be represented by Learning Objects that conform to agreed specifications. However, many Adaptive Hypermedia Systems have adaptive rule definitions and links embedded within the learning content, hindering re-usability.

With IMS Simple Sequencing instructional designers can specify meaningful sequencing behaviour externally from the learning resources. The sequencing behaviour is encoded using XML in a manifest file, which references the learning resources.

4.7.7 Conclusion

Albeit Adaptive Hypermedia Systems and IMS Simple Sequencing are superficially similar, there are some fundamental and important differences. Adaptive Hypermedia aims to use intelligence and knowledge of each individual user to assist the learner in achieving their chosen Learning Objective. Whilst, the IMS Simple Sequencing has no intelligence and makes no distinction between users, but simply applies a set of rules decided by an author to sequence a learner towards a pre-determined Learning Objective.

Even though the goals of Adaptive Hypermedia appear laudable, the concept has yet to catch on in either industry or education partly due to the lack of standards and the difficulty in reusing and authoring materials. SCORM 1.3 with Simple Sequencing can provide some useful tips on working in the mainstream: it makes use of re-usable

content, it has a specified language for expressing its sequencing rules, the sequencing rules are held independently from the content and there is a standard API for communicating the tracking information with the learning management system.

4.8 Summary

This chapter gives readers the overall idea of what Learning Object technology is and the benefits it offers to e-learning. It also emphasises the SCORM specification and its sequencing mechanism adopted from the IMS Simple Sequencing specification.

The following chapter will discuss some limitations of SCORM and how this thesis aims to address these limitations by incorporating an adaptive support system around the existing SCORM environment. So the next chapter will introduce readers to the proposed architecture of a Personalised Link Service, an adaptive support system for SCORM.

5 Personalised Link Service Architecture: An Overview

Chapter 5 introduces readers to the architecture of a Personalised Link Service (PLS). PLS aims to deliver on-demand external learning resources to the SCORM open learning environment based on a student model. The chapter begins by addressing some limitations of the SCORM learning environment that have motivated this research. This is then followed by a description on how these limitations are addressed in the PLS using the incorporation of the Adaptive Hypermedia principles and Auld Linky into SCORM. Readers are then presented with a walkthrough into the PLS architecture, covering its primary system components. The presentation of the overall description of the architecture is intended to prepare readers for the detailed elaboration of its design process which spans over the next two chapters (Chapter 6 and 7). This chapter ends by discussing the latest work found in the literature in regards to the attempt to personalise learning within SCORM.

5.1 Motivation

As described earlier in Chapter 4, SCORM promotes the concept of an open learning environment through a set of e-learning resource standard and specifications. By promoting common ways of defining and packaging learning content into a commonly identifiable course structure, content can be shared, reused and interoperate across

many systems and products (Mohan and Greer, 2003; DigitalThink, 2003; Slosser, 2002). These promises have made SCORM the most widely adopted e-learning technical specification today, as evident in the growing number of participants in every Plugfest[1] event (ADL , 2004).

SCORM courseware consists of a 'manifest file' describing both the resources to be used for a particular unit of learning material (html, Flash, PDFs etc.) along with one or more 'organisations' that shows the order in which these resources will be presented or sequenced. Both the manifest and the resources may then be packaged in a Package Interchange File (SCORM, 2002) in order to facilitate interchange between learning management systems. The most recent version of SCORM includes IMS Simple Sequencing which allows a course author to include assessment points and use the results to dictate the conditions under which a learner might progress to further material or be routed to remedial material (Abdullah et al., 2004). Also by adopting the specification, a consistent way of sequencing the SCORM course content can be accomplished regardless of the delivery platform. Additionally, since the IMS Simple Sequencing technique keeps the sequencing logic of the learning resources in a SCORM's manifest file, externally from their content, it increases the degree of content granularity and reusability.

The implementation of the Simple Sequencing technique, however, has resulted in a navigation that is less student-friendly. The sequencing is pre-crafted by the course author based on several instructional design strategies, and students' navigational path is normally determined at run-time, based on their performance in the given assessment. For instance, if a competency-based strategy is employed, students' knowledge of a course will first be assessed before assigning them to any particular module. So, even though SCORM can adapt learning activities to the current ability of a student, it actually occurs at a very basic personalisation level i.e. user's performance

[1] Plugfest is a small convention-like gathering of SCORM adopters from government, industry and academia, which aims to test the interoperability of the specifications and resolve inconsistencies and omissions

is evaluated in a test at run-time, and then provides the course sequence accordingly. When a course is completed, the only information the system has for a particular student is whether they pass or fail the course. In this essence, it appears that the principle of ADL SCORM is simple, that is, once the student passes a course then the skill delivered by the course is accepted as being successfully accomplished.

Recently, the issue of lack of student personalisation in an open learning environment, such as in the SCORM environment, has triggered much interest, particularly among the Adaptive Hypermedia research community, as demonstrated in (Modritscher et al., 2004; Baldoni et al., 2004; and Power et al. 2005). However, their work often resulted in either modifying the SCORM's structure or substituting its existing course sequencing mechanism with their own in order to sequence course content adaptively. As a result, a consistent sequencing behaviour across different learning management systems would not be accomplished. It is also important to note that in the work carried out by Blackmon, some recommendations have been proposed to extend the sequencing model adopted by SCORM in order to deliver adaptive course sequencing in the environment (Blackmon et al., 2004).

As an alternative to these approaches the idea of PLS is to accommodate students' differences without proposing any changes to the existing SCORM learning environment. Therefore in its approach, the existing SCORM's course sequencing mechanism is retained, whilst, the SCORM RTE is augmented with personalised links to on-demand additional learning resources based on a student model.

This approach of complementing the existing system also manages to tackle another limitation of the SCORM learning environment i.e. SCORM's students are confined to a static pool of learning resources organised in a predefined narrative structure. As a consequence, even though students' learning paths may differ, they will all encounter the same learning experiences. This limitation is resolved in the PLS system by dynamically including external learning resources at run-time to support the primary

material delivered by the environment. Since the additional resources delivered by the PLS system are retrieved from multiple remote locations as well as from the local linkbases (databases of links) within the local institutions (i.e. they are not kept within the SCORM package), each student can have the opportunity to further explore the taught concept using their preferred resources.

The innovation of the PLS design is that it can personalise students' experiences within the SCORM open learning environment without subverting the teacher's narrative. The benefits of this approach are twofold. First is the augmented SCORM learning environment can deliver learning materials that are preferable to both teachers (primary materials of their choice) and students (supplementary materials that are pre-selected to suit some aspects of their user models to assist their understanding of the primary materials). Secondly, it allows students to explore the primary taught concept using their preferred learning materials without adding extra load on the teacher to prepare variations of materials in order to satisfy individual needs.

Therefore the primary contribution of this thesis is to demonstrate that it is possible to populate a service to dynamically supply the SCORM environment with some additional learning resources chosen according to some preferences in a user model, without any mandatory intervention from a course author.

The feasibility of the PLS architecture is then demonstrated by the successful implementation of a prototype for a personalisation service based on students' preferred Learning Styles. The choice of Learning Styles as the user factor on which to select alternative materials is largely arbitrary, and it would have been possible to demonstrate other personalisation sources such as accessibility needs (e.g. for learners with impaired vision or hearing).

The implementation of the PLS system is based on a Web-service architecture for the following reasons. First this permits the available linkbase to be used by both humans and electronic software systems. Secondly, by having a loosely-coupled system, it provides more flexibility to introduce new functionality into the system such as implementing adaptive content presentation or populating the linkbase with resources from learning object repositories.

5.2 Personalised Link Service's Architecture: An Overview

The PLS architecture is designed with the objective of incorporating Adaptive Hypermedia principles and Auld Linky into the existing SCORM Run-Time Environment (SCORM RTE, 2004), so that the environment can be augmented with a collection of personalised links which can adapt to changes in a user model.

The major step in realising this incorporation was to hook into the SCORM RTE's sequencing engine, in order to attain real-time data about the following:

- a user id,
- the activity id for the next learning resource to be launched,
- the navigational control data.

This process takes place at the point where the SCORM RTE sends a navigation request from a user to an ADL Sequencer, as depicted in Figure 5.1. The ADL Sequencer encapsulates the functionality of all four conceptual processes (navigation interpreter, sequencing, rollup, and content delivery) required for sequencing the SCORM course content based on the IMS Simple Sequencing specification. It provides an interface for the SCORM RTE to implements the overall sequencing process of the SCORM courseware according to the specification. The navigation request is created when a user clicks any of the navigational buttons (e.g. next, previous, quit, suspend) or the option from the navigational tree map.

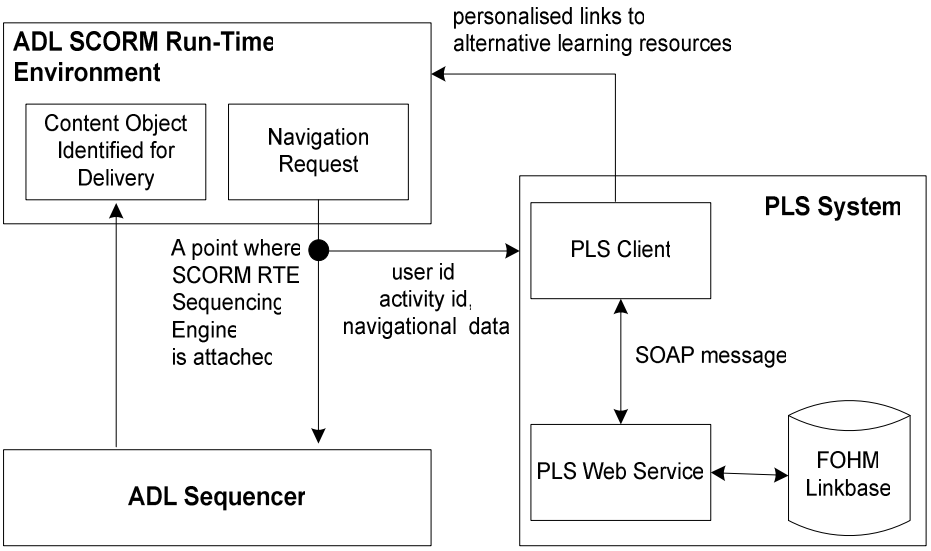

Figure 5. 1: High level conceptual view of the system integration between the PLS and ADL SCORM Run-Time Environment

The data obtained from the SCORM RTE will be processed accordingly by the PLS Client using related information derived from the following models:

- a domain model,
- a user model, and
- an adaptation model.

Once the process is completed, a query will be sent to the PLS Service using a SOAP message. Upon receiving the SOAP request, the PLS Service will interpret the message, then build a query object for Auld Linky to query the designated linkbases for the corresponding links. The PLS system uses a FOHM structure linkbase to store its links to alternatives resources.

The choice of FOHM as the model for storing links permits the links in a linkbase to be organised into meaningful groups. For instance, links to resources that deliver the concept 'layer' in Photoshop can be collectively put under the same 'association' object.

Moreover, the categorisation process can easily be refined by adding additional context values (e.g. resource type) or another context object. This representation enables Auld Linky to query links based on several context values. For example, it can dynamically query for additional links that match the current taught concept delivered by the SCORM RTE (e.g. 'layer') and fulfil a user's preference for learning resources of type 'exercise', 'experiment' and 'case study'. In other words, FOHM enables several context dimensions to be attached to a link so that different contextual views can be used to pull out only the links relevant to the current users.

As the PLS offers a dynamic querying mechanism at the application-to-application level (SCORM RTE querying PLS Client and PLS Client querying PLS Service), it can accommodate each individual's needs automatically. That is, the PLS can dynamically deliver a set of adaptive and personalised links without requesting any additional input from a user. The component-based conceptual diagram in Figure 5.2 illustrates the data flows among the major components of the PLS Run-Time Environment.

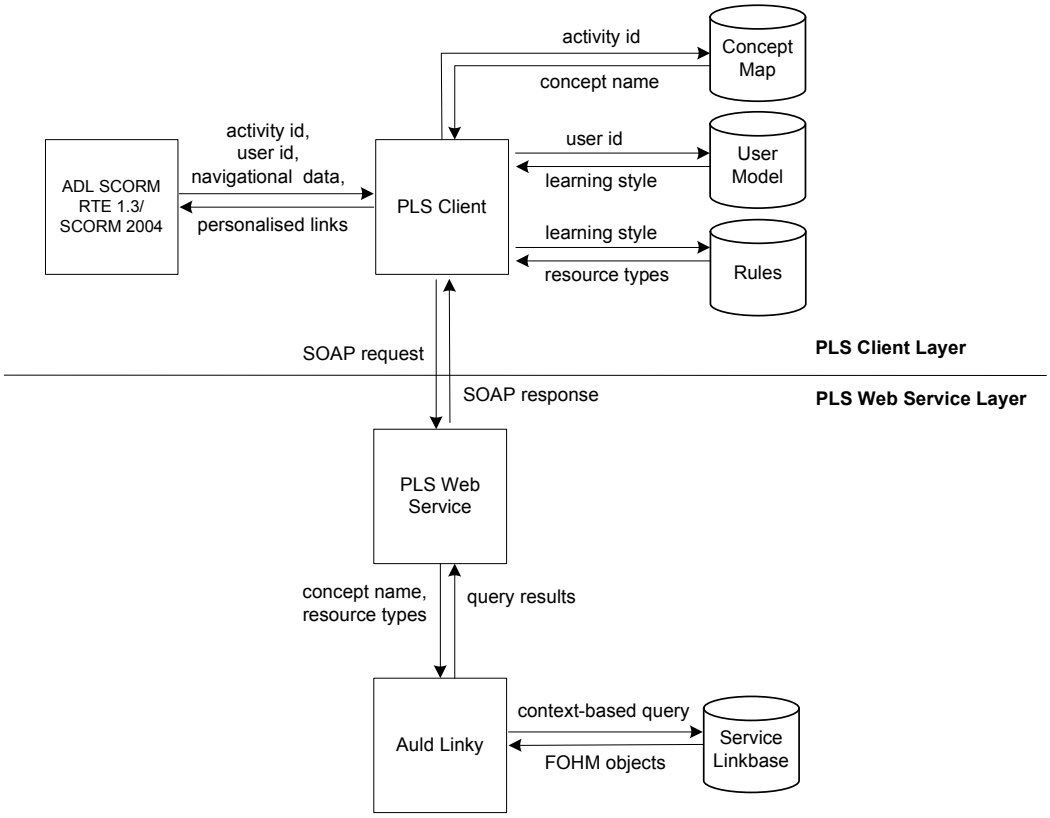

Figure 5. 2: Component-based conceptual diagram of the PLS Run-Time Environment

The top layer (PLS Client Layer) of the diagram represents the interaction among the components in the PLS Client (which consists mainly of an adaptive engine) as well as the interaction between the PLS Client and the existing SCORM RTE. When a student activates the PLS system via the SCORM RTE, information about his/her user id will be derived so that his/her user model which is based on the IMS Learner Information Packaging (IMS LIP, 2000) will be retrieved. Information about user's preferences (e.g. Preferred Learning Style) is later used by the adaptive engine of the PLS Client to derive suitable types of learning resources. In the PLS system, each link to an additional resource is marked according to the learning concept and resource type (based on the instructional function of the learning resource such as example, experiment).

The possible mapping between resource types and learning style is determined by the course author or course designer using a table-driven interface (refer Section 7.4.1). Once the mapping choices have been made, each entry is transformed into an XML data representation, forming some rules known as dynamic rules (see Section 7.4.1). Unlike most common adaptive logic, these rules are editable by the course author or designer. This flexibility permits the adaptive engine to accommodate many perceptions or interpretations arising from modelling human psychological factors (e.g. learning style).

The bottom layer (PLS Web Service Layer) of the diagram in Figure 5.2 represents the PLS Service. The PLS Service embodies Auld Linky, an engine for querying additional links from the linkbase based on the required contexts. When the Service receives a SOAP message from the PLS Client, it will de-serialize the message to derive the following parameters: page concept (e.g. 'layer'), a context label (e.g. 'resource type'), and a set of context values (e.g. 'experiment', 'example', and 'homework'). A FOHM query object is then built using the attained parameters values as its context values. Given this query object, Auld Linky will search one or more FOHM linkbases and retrieve appropriate links. The query results (i.e a list of URLs together with their relevant properties) will be returned to the PLS Client using a SOAP response. At the client, the SOAP message will be interpreted and the appropriate URLs elements are retrieved and processed accordingly. Finally, the PLS Client will display an adaptively sorted list of links to a user's browser along with other information such as an indicator of the links' priorities.

While a student is interacting with the PLS, their selections of the presented links will be monitored. Based on the evaluation of both dynamic and static rules (rules that are hard-coded into the adaptive engine), the system will decide whether the learning style currently deployed should be sustained or changed to the next recommended learning style. When a student has completed or suspended the course, the latest style incurred will be updated to his/her user model. As the system keeps a user model for each

individual student, it can directly use the recorded learning style to personalise the delivery of additional links to the SCORM RTE each time a student begins any course with the SCORM RTE.

5.3 Previous Work in Personalising Learning with SCORM

There have been several efforts to provide personalised learning within SCORM learning environment. The commonality feature among these attempts is that they either proposed some changes to the SCORM Content Aggregation Model or disregarded its course sequencing mechanisms which is based on the IMS Simple Sequencing specification. In this section, some of these attempts are reviewed to highlight the mechanism used to deliver personalised learning with SCORM.

In (Mödritscher et al., 2004), changes have been proposed to the concept of the SCORM Content Aggregation Model (CAM). According to the SCORM CAM, a SCO is built from a collection of Assets, and there should be no context-dependent material among SCOs in order to permit reusability and interoperability of the SCOs. However, their proposal suggested that a SCO should also be permitted to contain other SCOs besides a collection of Assets. They claimed that the extension is necessary in order to fulfil the following requirements: supporting different types of learning objects (e.g. content, exercises); providing different levels of detail for a learning object (novice, expert); and mapping a learning object to a learner's characteristics (language, accessibility, learning style). Their next suggestion is in regards to the concept of an Asset in the SCORM Content Model. According to SCORM, an Asset represents a basic form of digital content. For instance, an Asset can either be an image file, a web page, a video file or an audio file. However, they have suggested that an Asset should be able to contain more than one resource. For example, an Asset can contain an XML file as well as several XSLT files. They claimed that this enhancement can enable SCORM to deliver different content presentations to suit individual preferences.

When compared to this work, PLS builds an adaptive learning environment around the existing structure of SCORM. Unlike (Mödritscher et al., 2004), PLS does not propose changes to the core elements of SCORM which may disturb SCORM's integrity and its purpose of existence. For instance, allowing Assets to contain more than one resource may result in the inability for the Asset to be reused. Also, by suggesting that SCO should be allowed to contain another SCO may introduce conflicts in the existing communication process between SCOs and the learning management system.

On the other hand, (Power et al., 2005) aimed to apply curriculum sequencing techniques from the field of adaptive hypermedia to the problem of generating personalised SCORM-based courses. In their approach, an adaptive course is pre-authored using a Web-based authoring tool named MOT (Cristea and de Mooij, 2003) taking into account all the possible routes required to accomplish a particular learning goal. Subsequently, a MOT-to-SCORM converter is used to transform the adaptive course into a SCORM manifest. Then the pre-adapted course is packaged using a SCORM content package, and delivered to the student via any SCORM-compliant viewer in a SCORM-compliant learning management system.

When compared to the work done by (Power et al., 2005), PLS mainly differs in the way it integrates the adaptation feature into SCORM. Instead of sequencing SCORM courseware adaptively, PLS incorporates personalisation features by offering an independent adaptive link support service into SCORM Run-Time Environment alongside its existing course sequencing mechanism. As a result, SCORM can offer personalised links to supplementary learning materials according to students' preferences, yet still able to sustain the consistent sequencing of primary materials (selected by a course author) based on the well-designed instructional strategies developed by instructional experts (a particular teaching strategy is described using IMS Simple Sequencing specification). Additionally, the independent nature of the PLS makes it easier to adapt to future versions of the SCORM specification. As evidence,

the PLS prototype can work well without any alteration in both SCORM RTE Version 1.3 and SCORM2004 (ADL, 2004).

Another aspect that distinguishes the PLS approach from the approach deployed by (Power et al., 2005) is that the personalised links delivered by the PLS system can be dynamically adapted to changes in a student model (e.g. Preferred Learning Style) while they are interacting with the system. Instead of this dynamic approach, (Power et al., 2005) designed a course that is pre-adapted to a student initial learning goal, and does not accommodate dynamic adaptation.

There was also an attempt to deliver adaptive course sequencing from the Learning Object research community as demonstrated by (Blackmon et al., 2004). In their work, requirements to extend the sequencing model adopted in SCORM have been proposed. One of the requirements is to build a new sequencing rule that can identify learning objective in each lesson and aggregate the performance data of the lessons into one measure of performance. In the existing IMS Simple Sequencing specification, the score data for a learning objective is stored in a global variable. Unfortunately, the specification imposes a restriction that no two local objectives for a given learning activity can write to a same global objective (IMS SS, 2002a). Hence it cannot maintain a running total of scores computed after each lesson. However, the ability to accumulate the performance data for every lesson is essential to enable a learning management system to rank the remaining lessons before suggesting the next suitable lessons for a student to choose from. So, with this new recommendation, adaptive sequencing of a SCORM courseware can be established.

Finally, it is worth mentioning the approach taken by the Knowledge-On-Demand (KOD) project (Sampson et al., 2002) in order to bring adaptive learning into an open learning environment. In their approach, they proposed an extension to the IMS Content Packaging (IMS CP, 2003) in order to accommodate an adaptation model. This extended package is called a Knowledge Package. A Knowledge Package requires a

special adapter to be built into a learning management system to enable the Knowledge Package to be disaggregated, and for the adaptive rules to be interpreted. The adapter which comprises of several dedicated intelligent agents will then deliver the course content according to the associated adaptive rules and a user model. The concept of a Knowledge Package will enable adaptive courseware to be easily moved across several KOD e-learning systems or any KOD-compliant learning management system (Karagiannidis and Sampson, 2004).

When a comparison is made between the PLS system and KOD, the main differences arises from the objectives of both systems. The primary objective of KOD is to provide a means to easily transport and deliver adaptive educational e-content across several learning management systems (i.e. Knowledge Package compliant). Whereas the PLS aims to support SCORM-compliant learning management systems with personalised links to on-demand additional learning resources, in order to supplement its existing learning resources. Therefore, in the PLS, the structure of the SCORM package which is based on the IMS Content Package is preserved.

	PLS	Mödritscher et al., 2004	Power et al., 2005	Blackmon et al. 2004	Sampson et al., 2002 **
Changes to SCORM Content Aggregation Model	No	Yes	No	No	-
Changes to SCORM Sequencing Model	No	No	No	Yes	-
Changes to IMS Content Package Structure	No	No	No	No	Yes
Omitting IMS SS	No	Yes	Yes	No	-
Pre-authored adaptive SCORM content package	-	-	Yes	-	-
Use dedicated software agents to deliver adaptation	-	-	-	-	Yes
Deliver alternatives resources dynamically into SCORM at run-time	Yes	No	No	No	-
Adaptive link generation for SCORM	Yes	No	No	No	-
Adaptive course sequencing for SCORM	No	Yes	Yes	Yes	-

***KOD is not designed specifically for SCORM, rather for general open learning environment*

Table 5. 1: A summary of the comparisons between PLS approach and other approaches found in the literature.

5.4 Summary

This chapter highlights the reasons that motivate the work of this thesis, provides an overview of the Personalised Link Service (PLS) architecture and describes related work. The next chapter will deal with the first design phase of the architecture i.e. the PLS Pre Run-Time Authoring Environment.

6 Personalised Link Service: Pre Run-Time Authoring Environment

Chapter 6 reports the designing and implementation process involved in the PLS Pre Run-Time Authoring Environment. The chapter begins by describing the architecture of the environment which comprises of two important phases in the design: 1) generating a concept map from a SCORM course content package; 2) constructing a linkbase based on the understanding of the knowledge items embodied in the package and the instructional function of the Web learning resources.

The innovation of the design is the automation of the process of creating both concept map and linkbase, so that involvement from a course author or subject expert is not mandatory. Both the concept map and the linkbase will provide the essential input data for the next design phase i.e. the PLS Run-Time Environment.

Also included in this chapter are the algorithms used in both phases. First is the algorithm to identify a dynamic threshold value for including terms in a lexicon[2] entry. Throughout this thesis the term lexicon is used to represent a list of terms that have the potential to represent learning concept of a learning resource. The second algorithm is for deducing concept names for every learning resource in a SCORM package. Finally, is the algorithm for auto-constructing PLS's linkbase according to the FOHM data model.

[2] According to Oxford English Dictionary a lexicon refers to a list of terms that representing a subject domain

6.1 An Overview of the Architecture

This section provides the overall architecture of the PLS Pre Run-Time Authoring Environment. The architecture serves two purposes; to automatically deduce a concept map from a SCORM package and construct a linkbase.

Before we proceed, it is worthwhile to recall that SCORM package contains the SCORM resources (physical files) required in delivering the intended course alongside with a manifest file and other IMS schema files. The manifest is an XML-file that describes how the physical files should be organised to enable the delivery of a course that meets a particular instructional strategy. It contains a set of metadata, an organisation element, resources elements, as well as sequencing rule elements. The resources element contains a set of learning resources. The organisation element is composed of a collection of activity nodes (arranged in a hierarchical tree structure), in which each leaf node is usually attached to a corresponding learning resource and enclosed a title element. The title element usually reflects the content of the learning resource associated to the learning activity. This title element plays an important role in the process of inferring learning concept from the SCORM resources.

Figure 6.1 shows that the generation of a concept map and a linkbase begins from a SCORM Content Package. The left side of the diagram shows how the SCORM package is processed to generate a concept map, whereas the right side of the diagram indicates how the identified concept names for every learning resource in the package are used to locate the potential Web resources and consecutively construct a linkbase of alternative resources.

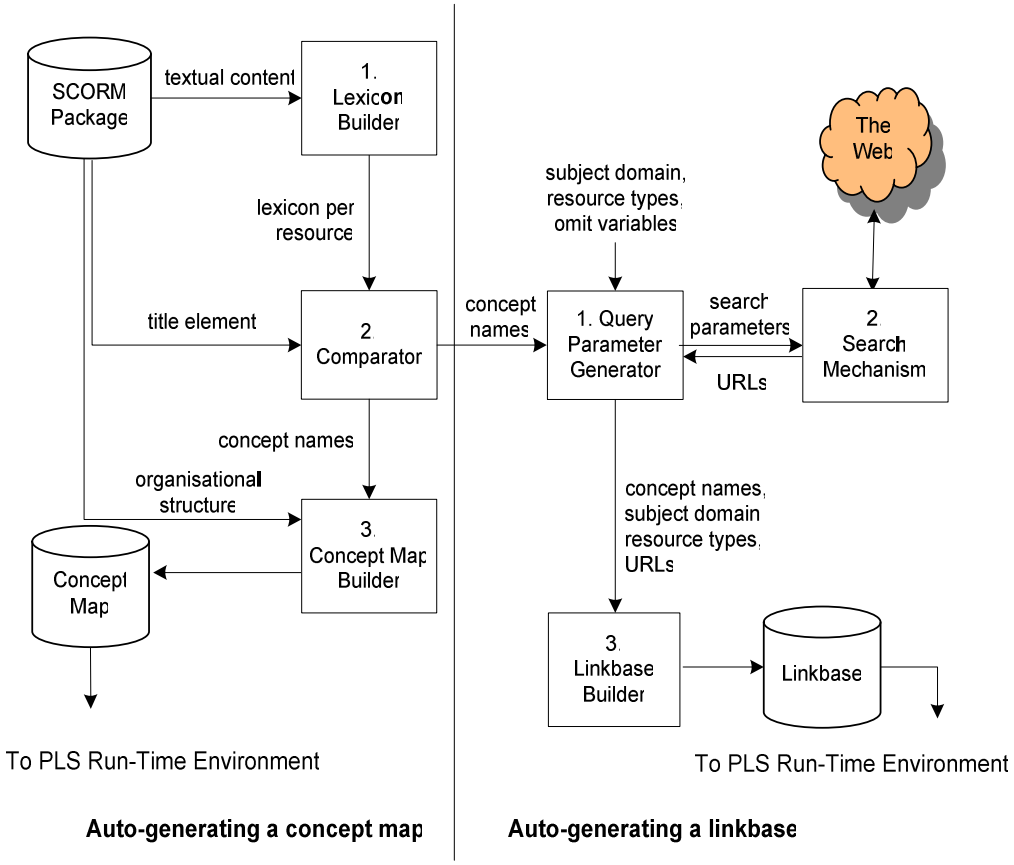

Figure 6. 1: Process flow diagram for the architecture of the PLS Pre Run-Time Authoring Environment

The left side of Figure 6.1 shows that when a SCORM package is downloaded, the textual content of its every learning resource (html pages) is analysed by the *Lexicon Builder*. The analysis process resulted in a list of potential terms for the learning concept (referred to as *lexicon*) for each resource. In a package which aims to teach about Photoshop, the lexicon entry might be '*color*', '*balance*', and '*image*'. In order to determine the concept name for the resource, the *Comparator* will compare this lexicon entry with the title element (e.g. "*Lesson 9 – Color Balance*") of the associated learning activity. This title information is obtained from a manifest file inside the SCORM package. Any matches found in the comparison process will be the concept name (e.g. Color Balance) for that resource. The process continues for every resource in the SCORM package.

The final stage in generating the concept map would be the process of identifying the relationships among the generated concept names. The organisational structure of the learning activities outlined in the manifest file provides the foundation to infer these relationships. *Concept Map Builder* will extract these relationships from the manifest file and obtain the concept names from the *Comparator* prior to constructing a concept map for the intended SCORM courseware package.

The right side of the diagram in Figure 6.1 deals with the tasks of locating alternative resources from the Web and generating the linkbase. Each concept name derived earlier by the *Comparator* will be used by the *Query Parameter Generator* as one of the input required in formulating a query object for a search mechanism. Other inputs are as follows: subject domain (e.g. adobe photoshop), a list of resource types (e.g. example, assignment), and omit-variables (e.g. price, bookstore). Once the query object (or statement) is formulated, the *Search Mechanism* such as Google (Google API, 2002) will search the Web for the resources that match the concept name as well as other aforementioned input parameters. A successful search will return a list of corresponding links. These links along with other related information will be processed by the *Linkbase Builder* prior to constructing a FOHM-structured linkbase.

Note that the concept map provides the system with the knowledge model of the downloaded SCORM package, whereas the linkbase accommodates the system with a repertoire of links to the alternative learning resources. The PLS linkbase is an XML-file that is composed of several associations of links expressed in FOHM (Fundamental Open Hypertext Model) (Millard, 2000). Each association is marked up with the concepts that they respond to, and the resource types they represent.

6.2 Auto-Constructing the Concept Map

This section describes in detail the PLS approach in auto-constructing a concept map from a downloaded SCORM package. The approach comprises of two stages. The first stage aims to deduce concept names from each learning resources in the SCORM package. In the second stage, the relationships among the generated concept names will be derived from the manifest file so that concept map's elements and attributes can be inferred. The process involved in each of these stages will be elaborated in the following sub sections.

Note that in the attempt to describe the implementation of the auto-generation of the concept map, a SCORM package example on the subject of '*Adobe Photoshop*' is deployed. This example package (.zip file) is purposely developed by the ADL technical team to demonstrate the use of IMS Simple Sequencing in sequencing and delivering of the SCORM course content via any SCORM-compliant learning management systems (ADL Technical Team, 2004). The package contains a collection of learning resources about Adobe Photoshop, a manifest file (named as "*imsmanifest.xml*") and some relevant IMS schema files.

6.2.1 Deducing the Concept Names

This process requires each learning resource in a SCORM package to be analysed in order to extract a relevant set of keywords for its lexicon entry. Among these keywords, a single or multiple words will be chosen to represent the learning concept of the resource.

Prior to extracting a lexicon from each resource, its textual content must first be extracted into a cache. Then by using a Java Lucene text processing tool (Lucene, 2001), the content will be processed to reduce the set of representative keywords. In this process every term in a document will be tokenized, and then filtered to eliminate stop-words such as articles (a, the), prepositions (in, of), conjunctions (and, either), pronouns (he, they), symbols (punctuation, coma), auxiliary verbs (be, might, will, I've,

hasn't), alphabets (a to z), and numbers (0 to 9). Tokens that represent the intended subject name (e.g. 'adobe' and 'photoshop') will also be removed. A stemming process is not performed onto the tokens in order to sustain the original state of some important nouns such as 'selection' and 'saturation'. Instead, any 's' suffix will be stripped off from each filtered token based on the following rule:

> Remove the 's' suffix from the analysed token only if its second last character is not 's' but its last character is 's'.

The execution of this rule will result in a term 'layers' will become 'layer', whereas term 'brightness' will sustain in its original form. However, the algorithm will also lead to a scenario wherein the term 'serious' (an adjective) will be truncated to 'seriou', but this normally happens to terms which are not representing nouns. As most concepts are nouns as pointed out by (Lin et al., 2004) and further supported by the Oxford Advanced Learner's Dictionary which stated that "*a concept is a noun that defines an idea or a principal*", there will be a low probability for this sort of terms (non nouns) to be included in the lexicon entry (because their frequency of occurrences will be lower than the nouns).

When the stop-words and s-suffix have been eliminated from the term tokens, each token will be assigned a weight based on their frequency of occurrences in the document. The higher the frequency is, the greater the chance for a term token to be the candidate for the lexicon. However, for the candidate to be accepted as a lexicon entry, its frequency must be above a certain threshold value. A method of determining the appropriate threshold value for each processed document is described in Section 6.2.3.

The process of generating the lexicon will be repeated for every learning resource in the SCORM package. The third column of Table 6.1 denotes the lexicon entry for each referred SCORM resources in which each element in the lexicon per resource is sorted in decreasing order of importance based on their frequencies.

Subject domain: Adobe Photoshop		
Resource Id	**Activity Title**	**Lexicon Per Resource**
Intro.htm	Introduction	tutorial, image, web, book, site
Lesson 1.htm	Lesson 1 – Interface	image, option, tool, interface, bar, component
Lesson 2.htm	Lesson 2 – Toolbox	tool, image, color, selection, toolbox, area
Lesson 3.htm	Lesson 3 – Palettes	palette
Lesson 4.htm	Lesson 4 – Layers	layer, image, circle
Lesson 5.htm	Lesson 5 – Color Balance	color, balance, image, layer, original, adjustment
Lesson 6.htm	Lesson 6 – Brightness and Contrast	contrast, brightness, image, layer, adjust, adjustment
Lesson 7.htm	Lesson 7 – Hue and Saturation	hue, saturation, image, layer, color, adjustment
Lesson 8.htm	Lesson 8 – Selection Tools	tool, selection, lasso, select, option,
Lesson 9.htm	Lesson 9 – Transform	image, transformation, selection, bounding, box, transform

Table 6. 1: An auto-generated lexicon obtained from each learning resource in a SCORM package

Up to this stage, a lexicon list is available for every lesson page of the SCORM package. The next task is to determine which terms within the lexicon should be chosen as the concept name to represent the resource. This task is addressed by using the existing information from the SCORM manifest file that is the title element for the related learning activity (attached with the associated resource). It is worthwhile to recall that every resource in the SCORM package is referred by an item element in the manifest file and each item element encloses a sub element named title as depicted in Figure 6.2.

```
<item identifier="MODULE2">
  <title>Module 2 -- Enhancing Images</title>
     <item identifier="LESSON5" identifierref="RESOURCE_LESSON5">
         <title>Lesson 5 – Color Balance</title>
     </item>
  …
</item>

<resource identifier="RESOURCE_LESSON5" adlcp:scormType="asset"
type="webcontent" href="Lesson5.htm">
  <file href="Lesson5.htm" />
  <file href="images/EndOfLesson.gif" />
  <file href="images/LessonTitle5.gif" />
  <file href="images/colorbalancebox.gif" />
  <file href="images/headerside.gif" />
  <file href="images/headertop.gif" />
…
</resource>
```

Figure 6. 2: A fragment of the item and resource element in a SCORM manifest file

In SCORM, the title element is mandatory and it often conveys meaningful information about the associated learning resource as depicted in the second column in Table 6.1. Therefore by comparing the lexicon (per resource) against the corresponding title element, a possible match can be found. Any terms that match will be regarded as the concept name for the resource. In the example shown in Figure 6.2, the concept name for a resource *Lesson5.htm* (identified by the resource id 'RESOURCE_LESSON5') will be represented by the following keywords: *color*, *balance*. Otherwise the keyword with the highest frequency found in the lexicon entry would be chosen as the concept name.

The following section will explain the process of deducing the relationships among the generated concept names in order to infer elements and attributes of a concept map.

6.2.2 Deducing the Concepts Relationships and the Concept Map's Elements and Attributes

After the concept names have been confirmed for every learning resource of the identified SCORM package, the relationships among the concepts must be determined prior to constructing a concept map. The tree structure of the organisational element of the manifest file permits '*has-a*' and '*is-part-of*' relationship to be established between

the activity nodes. For instance, if an item element A contains two child nodes named A1 and A2, we can infer that the relationship between A and its children is a '*has-a*' relationship whereas the children will have the '*is-part-of*' relationships with their parent which is A. From this information, the following elements can be inferred for the concept map: the activity node id, and its parent node id.

Next, the arrangement of the activity nodes in the tree structure will determine the prerequisite element. Finally, the rollup rule element (see Section 4.6.2) of the sequencing element in the manifest file can provide useful information in indicating the percentage contribution the concept at every child node makes to its parent node (Abdullah et al., 2004). The rule can contain statements such as '*any*', '*all*' or '*at least N*' of the child nodes are needed to complete this node. This information can be used to specify the value of the weight attribute of a concept in the concept map as demonstrated in the following scenario.

Assume a tree cluster is composed of four children nodes. In a case where '*all*' statement is deployed, a value of 25 (i.e. 100/4) is assigned to all four nodes. When this rule is evaluated, the parent node will only be considered as '*has been satisfied*' if all child nodes have been completed (i.e. value of 100 has been obtained). Likewise if '*at least 2*' is used, each node will have a weight of 50 because it only requires two nodes to be completed in order for the parent node to be satisfied.

As the weight element conveys the information of the importance of a particular concept to its parent's concept, it can be used to serve knowledge-based adaptation in the future. The process of assigning weights to each concept element is inspired by the principle of 'concept-weighting' deployed in AHA! (DeBra et al. 2002a; Brusilovsky, 2003).

When the relationships among the concept names have been identified and the relevant concept map's elements and attributes have been deduced, a concept map can now be generated. Figure 6.3 displays a fragment of a generated concept map. The generated concept map will later served the PLS Run-Time Environment in deducing

the learning concept from each learning resource delivered by the SCORM player. It is also important to note that based on the generated learning concepts provided by the concept map, search for alternative learning resources can then be established.

```
<conceptmap>
    <concept name="layer " nodeid="LESSON4">
      <prerequisite>LESSON3</prerequisite>
      <parent pnodeid="MODULE1" />
      <weightpropagate>25</weightpropagate>
    </concept>
    <concept name="balance color " nodeid="LESSON5">
      <prerequisite>MODULE1</prerequisite>
      <parent pnodeid="MODULE2" />
      <weightpropagate>33</weightpropagate>
    </concept>
    …
</conceptmap>
```

Figure 6. 3: A fragment of the auto-generated concept map

The structure of the attained concept map is relatively simple as compared to the one used in other Adaptive Hypermedia System such as AHA! (DeBra and Calvi, 1998). This simplicity is due to the limited information that can be derived from the SCORM package. Nevertheless, the success of the PLS prototype (see Chapter 8) has confirmed that the generated concept map can facilitate the system to understand the knowledge domain of a SCORM package. Moreover, as the concept map is constructed automatically using only the knowledge deduced from a SCORM package, it can overcome application dependency on either a subject expert or a dictionary.

6.2.3 Determining a Dynamic Threshold Value

It is worthwhile to recall that only the term tokens that have the frequency above a threshold value will be accepted as a lexicon entry. In the PLS system, this threshold value is determined dynamically based on the terms' frequencies distribution within a document. This section will describe the PLS's approach in determining a dynamic threshold value. The approach involves plotting a graph of a term frequency against the total number of term tokens for each stated frequency based on the data presented

in Table 6.2. By analysing the points distributions in the graph a dynamic threshold value can be deduced for each document.

Concept Name	Layer	Image	Circle	Icon, Arrow	Others		
Number of Element	1	1	1	2	18	30	29
Frequency of Term	0.1502	0.0750	0.0346	0.0231	0.0173	0.0115	0.0057
Caption/Legend			Z0	Z1	Z2		

Table 6. 2: Term frequency data analysis for a document that delivered concept 'layer'

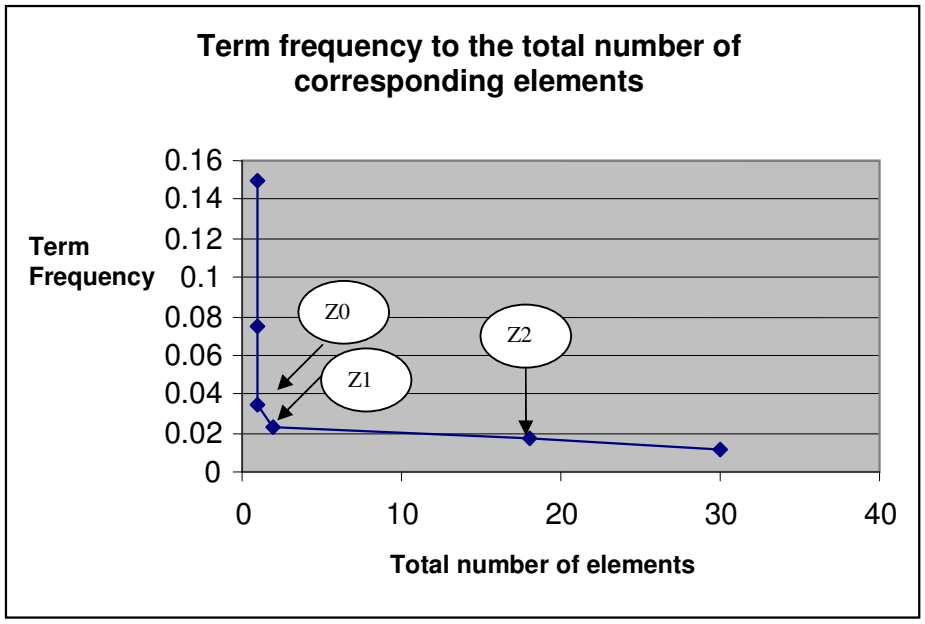

Figure 6. 4: Represent the number of term element with a particular term frequency value

The graph shown in Figure 6.4 denotes that as the value along the y-axis gets nearer to zero there is a huge leap along the x-axis. For instance, it goes from one unit (i.e. from Z0 to Z1) to sixteen unit gaps (i.e. from Z1 to Z2). A same pattern was attained when this algorithm is applied to another document (with another learning concept). This observation signified that the distributions of terms in a document can be divided into

two priority sets. That is, all points that preceding point Z1 (0.0231) appeared to be more important than those succeeding it. Therefore, terms with the frequency below 0.0231 will automatically be excluded from the lexicon entry.

In order to determine the probability for the terms at point Z1 to be included in the entry, a '*differentiation rule*' must be applied. According to this rule, the frequency differences between a target frequency (Z1=0.0231) with a preceding frequency (Z0=0.0346) and between the target to a succeeding frequency (Z2=0.0173) must be calculated and compared. A smaller differentiation value denotes higher correlation values. In the case of the above graph, the frequency difference between Z0 and Z1 is 0.0115, whereas the frequency difference between Z1 and Z2 is 0.0058. Since the later pair has higher correlation than the former, we can conclude that Z1 belong to the Z2's group. In other words, Z1 will be excluded from the lexicon entry. Therefore the derived lexicon for this document will only be comprised of the following terms: *layer*, *image*, and *circle*.

This dynamic approach in identifying a threshold value eliminates the requirement for a course designer to manually specify the default value or for a system designer to anticipate the most appropriate value for it.

The idea for this approach is solely the work of the author which originated from her observation of the repetitive patterns received while analysing the products of the term frequency algorithm for every document in the SCORM package. This pattern was then discovered useful in deducing a relevant set of lexicon for this work. The more relevant the lexicon is, the higher the possibility of finding the match when these lexicons are compared with the title elements of the associated learning activities. The author, however, does not claim her finding make any novel contribution to the field of information retrieval which is beyond the intention of this research.

6.3 Auto-Constructing the FOHM Linkbase

In the PLS system, links to the potential learning resources are stored in a linkbase, based on a FOHM representational model. In (Abdullah and Davis, 2005), the linkbase is generated manually; now it is constructed automatically using the UtilityTool (see Section 6.4).

The ability to auto-construct a linkbase is important because it allows the PLS system to offer students with many adaptive choices of supplementary learning resources without putting extra load on the course author in preparing them. As in the PLS system, links to some potential supplementary resources are automatically searched and retrieved from the Web, using a list of contextually formed query object to Google API (Google API, 2002).

The following subsections will further elaboration the process required in auto-constructing the linkbase.

6.3.1 Determining Potential Learning Resources

Prior to constructing a linkbase, sources for the potential learning materials must first be identified. The PLS system requires that each resource must convey both a learning concept and a required instructional function (e.g. example, or experiment). Throughout this thesis, a metadata called resource type is used to represent the value of the instructional function of a learning resource. Both concept and resource type represent the context values in the linkbase. These values will later be used by Auld Linky (Michaelides et al., 2001) to query linkbase for alternative resources.

Initially there is a consideration to obtain the potential resources from the learning object repository such as MERLOT (Multimedia Educational Resource for Learning and Online Training) (MERLOT, 2005) and ARIADNE knowledge base (ARIADNE, 2004). However, attempts with their search mechanisms are less successful to provide the required resource due to the following reasons. Firstly, the query result often led to

an index page which listed the potential Web sites. In this case, it is not feasible to directly access a specific Web page as there is no metadata description at the page level (as in the case of many MERLOT resources). On the contrary, PLS requires links to point directly to a specific Web page that conveys a specific learning concept and embodies a pedagogic value. This requirement is necessary because PLS aims to deliver as fine a granularity of the learning concepts as possible within a targeted subject domain. For instance, it intends to deliver learning resources on how to apply layering techniques in Adobe Photoshop as opposed to how to use Adobe Photoshop. Secondly, when a metadata description is not available at the page level, links to a particular Web page cannot be marked up with the required metadata. Nonetheless, this requirement is essential for labelling links in the PLS linkbase.

Apart from that, there are related issues pertaining to metadata representation in learning object repositories that demand an intricate algorithm for querying several repositories based on a specific metadata element such as resource type. This complication arises because of the following two factors. Firstly, there are different terms used to represent a resource type metadata across learning object repositories, according to the survey carried out by (Bernd, 2002). For instance, ARIADNE (ARIADNE, 2004) uses the term '*document format*' while both MERLOT (MERLOT, 2005) and IEEE LOM (IEEE LOM, 2002) use the same term '*resource type*'. Secondly, there are indeed many variants exist in the vocabulary representation for this metadata element. For instance, MERLOT's representation is based on the instructional function of the material (e.g. lecture, tutorial, simulation, example, case study), ARIADNE's uses the function of the document which is either passive (e.g. essays, video clips, graphical materials) or active (e.g. simulations, exercises), while EducaNext (EducaNext, 2001) prefers to use the content type (e.g. text, image, sound, data set, software) (Bernd, 2002).

These findings imply that an enormous amount of work and specific research is needed before the PLS system can utilise resources from the learning object

repositories. As this demand is beyond the scope of this thesis, other option that can promise a large number of potential and '*ready-to-use*' links is the Web. The availability of a formidable search engine like Google further supports the Web as the source for the PLS linkbase. With its Google API (Google API, 2002), Google offers a convenient platform for its search mechanism to be integrated into the PLS system. The following section describes a specific querying method to Google in order to increase the relevancy of its result and to enable links to be labelled with the required metadata (e.g. concept, resource type).

6.3.2 Searching for the Potential Learning Resources

As PLS should provide alternative learning resources for every taught concept in a chosen SCORM package, it would be a tremendous task for a course designer or course author to articulate the suitable search query parameters and manually enter the querying statement in Google for each SCORM resource. Moreover, the process might lead to an inconsistent query statement.

To alleviate these issues, the system provides an auto generated query statement which is based on the following syntax {+*subject_domain*, +*concept_name*, +*resource_type*, -*omit_variable*}. Each parameter plays an essential role in scoping and refining query results. The plus sign is a feature offered by Google to make it a compulsory for Google search engine to include the associated word in the search result. The *subject_domain* represents the targeted subject domain which in this case will be '*adobe photoshop*'. The *concept_name* refers to the concept names generated from the process of comparing the auto-generated lexicon with the associated title element from the manifest file (see Section 6.2.1). The *resource_type* represents the instructional function required for the alternative resources such as example, and experiment. The minus sign in front of the *omit_variable* is also a feature offered by Google to exclude a word from the search result. Both the *resource_type* and the *-omit_variable* can be edited by a course author using the UtilityTool (see Section 6.4).

For the prototype, the *omit_variable* list comprises the following elements: *price*, *fee*, *bookstore*, and *bookshop*. By deploying this list the tendency for the query result to include links that are related to marketing the Adobe Photoshop products can be minimised. The disadvantage is it may also lead to some useful links being excluded from the query result. However, concern over the issue of irrelevant links especially links for marketing purpose is more dominating than not having all the useful links. Moreover, too many useful links can create the problem of cognitive overload to a user.

A successful search based on this auto-generated query statement will produce a list of URLs for each query statement. The list will then be parsed to form a nested vector that categorises results into a more meaningful structure for later analysis. The nested vector has the following syntax: [*subject_domain* [*concept_name*, [*resource_type*, [*url0*, *url1*, *url2*,..]]]]. Based on this nested vector, a FOHM-structured linkbase will be built.

The auto generated query statement have two advantages. First it requires little effort from the course author to derive supplementary resources for every lesson page within a SCORM package. With a click of a button, a list of formulated queries will be sent to Google API to get potential resources to supplement each lesson page found in the SCORM package (see UtilityTool in Section 6.4). Secondly, the designated parameters can facilitate refining the query result to the most significant scope as required. By having this semantic combination of keywords, the querying result can deliver a significant Web page with a relevant learning concept and resource type (e.g. experiment, homework), as depicted in Figure 6.5.

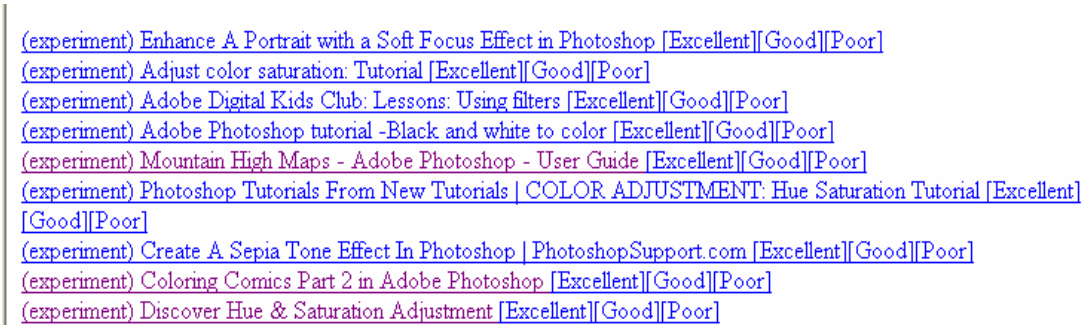
(experiment) Enhance A Portrait with a Soft Focus Effect in Photoshop [Excellent][Good][Poor]
(experiment) Adjust color saturation: Tutorial [Excellent][Good][Poor]
(experiment) Adobe Digital Kids Club: Lessons: Using filters [Excellent][Good][Poor]
(experiment) Adobe Photoshop tutorial -Black and white to color [Excellent][Good][Poor]
(experiment) Mountain High Maps - Adobe Photoshop - User Guide [Excellent][Good][Poor]
(experiment) Photoshop Tutorials From New Tutorials | COLOR ADJUSTMENT: Hue Saturation Tutorial [Excellent] [Good][Poor]
(experiment) Create A Sepia Tone Effect In Photoshop | PhotoshopSupport.com [Excellent][Good][Poor]
(experiment) Coloring Comics Part 2 in Adobe Photoshop [Excellent][Good][Poor]
(experiment) Discover Hue & Saturation Adjustment [Excellent][Good][Poor]

Figure 6. 5: Query result returned by the PLS system pertaining to learning resource 'hue and saturation' and resource type 'experiment'. Each entry will link to alternative resources.

Oppositely, the author's observation when querying MERLOT for learning resources pertaining to learning concept '*hue and saturation*' does not obtain any result. However, when learning concept '*photoshop*' is used instead, the result of the querying is as displayed in Figure 6.6. Note that each item in the MERLOT result links to Web sites instead of a Web page. Comparisons cannot be established with other learning object repositories because at the moment only MERLOT contains learning materials for the domain (Adobe Photoshop).

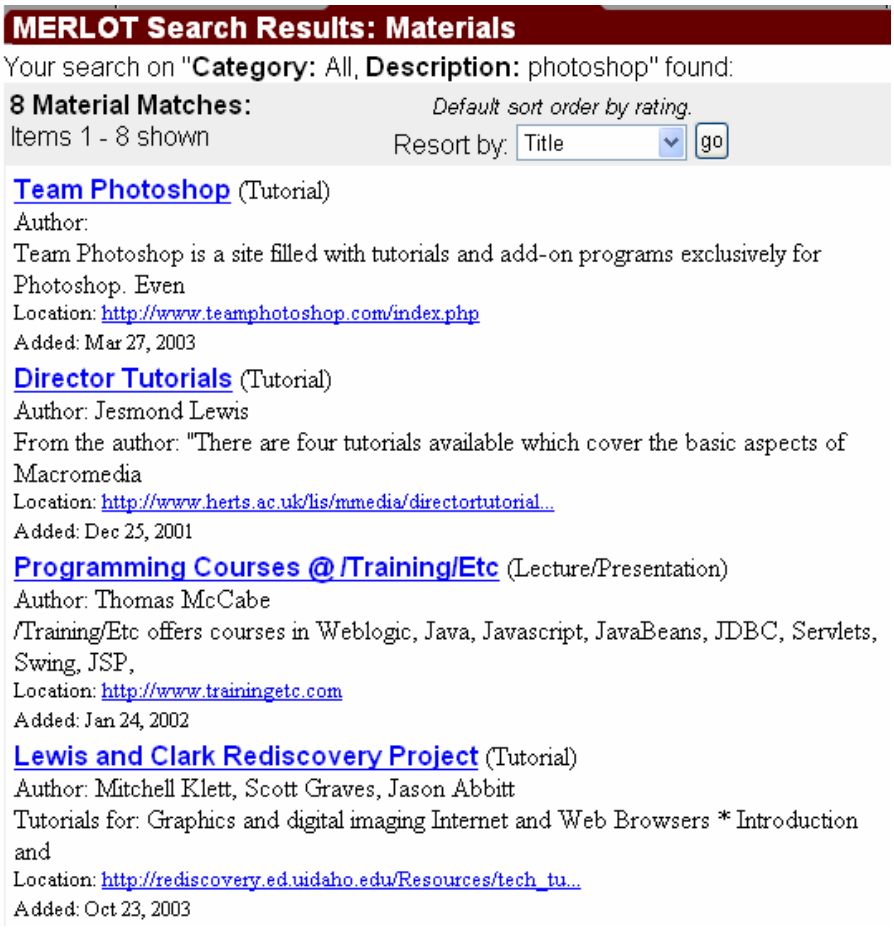

Figure 6. 6: Query result returned by MERLOT for 'photoshop'

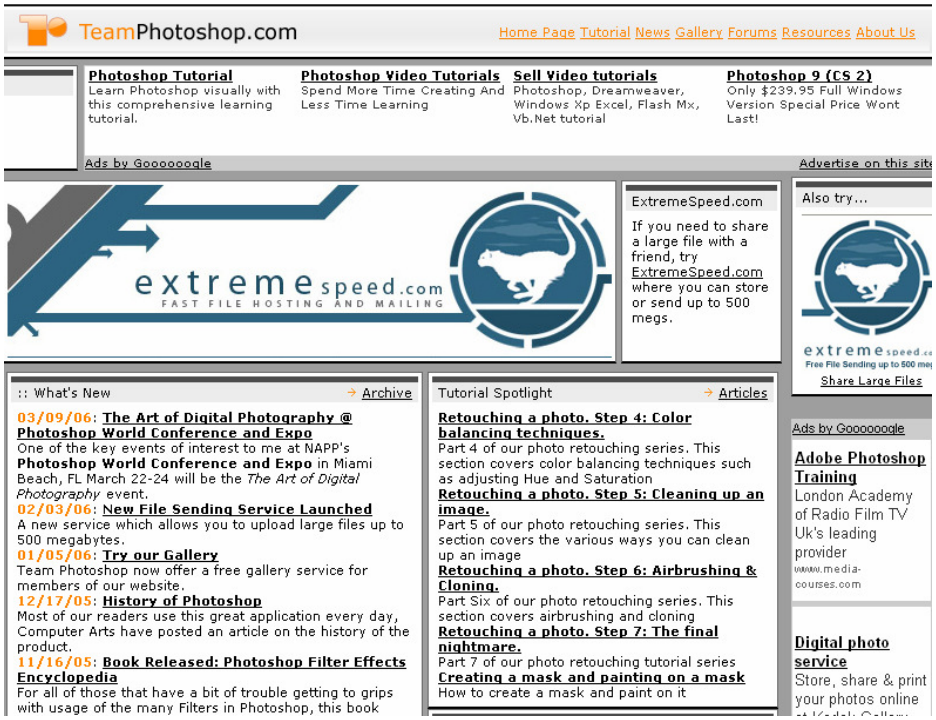

Figure 6. 7: Page delivered by the first item listed in the query result comprises of an index page

Figure 6.7 displays a typical index page that a user will get when any of the items presented in Figure 6.6 is selected.

6.3.3 Populating the Linkbase

This section describes the algorithm for populating the linkbase with links to the identified Web resources. The PLS linkbase is an XML-file that is composed of associations of links expressed as hyper-structures in FOHM, and marked up with the concepts that they respond to, and the resource type (e.g. example, experiment, exercise, homework) they represent. Hence, the identified links can be semantically organised according to these required metadata. In our earlier work (Abdullah and Davis 2005), the associations are marked up with concepts and learning styles. However the adjustment (i.e. to have resource types instead of learning styles) was necessary in order to alleviate linkbase dependency on a particular type of learning styles so that it can have a more generalized function. The algorithm used for constructing the linkbase automatically is as described in Figure 6.8.

1. Create a xml linkbase root document for the subject domain
2. Retrieve the *build* vector = {*concept_name*, [*resource_type*, [*url0*, *url1*, *url2*,..]].}
3. Get all elements with the same *concept_name* value
   - 3.1. Get all the elements with the same *resource_type* value
   - 3.2. Extract all the url elements
   - 3.3. Create a FOHM association object to bind all the *url* elements in step (3.2)
   - 3.4. Create a context object with the *resource_type* and *concept_name* values
   - 3.5. Attach the context object in 3.4 to the association in 3.3
   - 3.6. Repeat steps 3.1 to 3.5 until all *resource_type* for every *concept_name* are completed
4. Repeat step 3 until there are no more *concept_name* elements in the *build* vector

Figure 6. 8: An algorithm for auto-constructing the FOHM-structured linkbase

In summary, the linkbase construction process begins by creating a root element of an XML document. Then a nested vector that contains the search result will be retrieved and parsed accordingly, resulted in the following outcomes. Firstly, an association is created to bind all the '*url*' vector elements. This is followed by generating a context object and assigning it with the corresponding *resource_type* and *concept_name*. The completed context object will then be attached to the association object. Both processes will be repeated for each *resource_type* in every *concept_name* element that exists in the vector. Then all the associations will be bound to the root association of the linkbase. This root association will be attached to another context object that contains the targeted *subject_domain* which is in this case '*adobe photoshop*'. Figure 6.9 shows the structure of the linkbase resulting from this process.

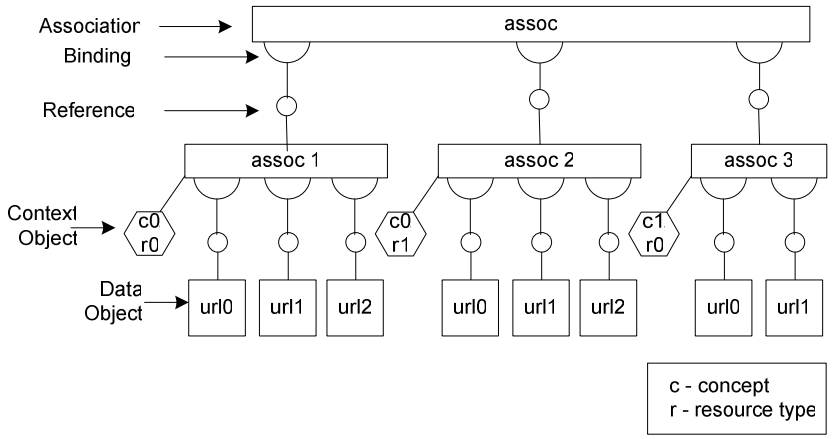

Figure 6. 9: The resulting structure of the FOHM linkbase

However, there is an issue pertaining to the representation of a learning concept as context value if the concept comprises of more than one word. For instance, how can a system conclude that '*color balance*' and '*balance color*' are conveying the same learning concept? In the PLS design, this issue is addressed by representing this sort of concept in an alphabetical order. The standardisation in the representation can alleviate the probability of concept redundancy when each time the linkbase is populated with a new batch of information.

Another question that will arise is how the system can decide that '*color*' and '*colour*' convey the same concept. One of the ways to tackle this issue is by deploying a digital dictionary that contains all potential word variants that can be found in both UK and US English. Additionally, a spelling checker that has the ability to recognise UK English words from US English words is also required. However, this option will not be considered at this implementation stage of the PLS.

As a conclusion, the PLS approach in automating the process of populating and constructing a FOHM-structured linkbase has introduced an alternative method to automate the process of categorising existing Web materials according to the required adaptation metadata within an educational context. This method enables a list of potential additional resources for every learning resource in the SCORM package to be automatically generated with no mandatory involvement from a subject expert or a course author.

6.4 UtilityTool

The UtilityTool (Figure 6.10) allows a course author to automatically derive a lexicon for a downloaded SCORM course package, build a concept map, trigger related search query to Google API, and subsequently construct a FOHM-structured linkbase. It also provides an interface to generate dynamic rules as further described in Section 7.4.1.

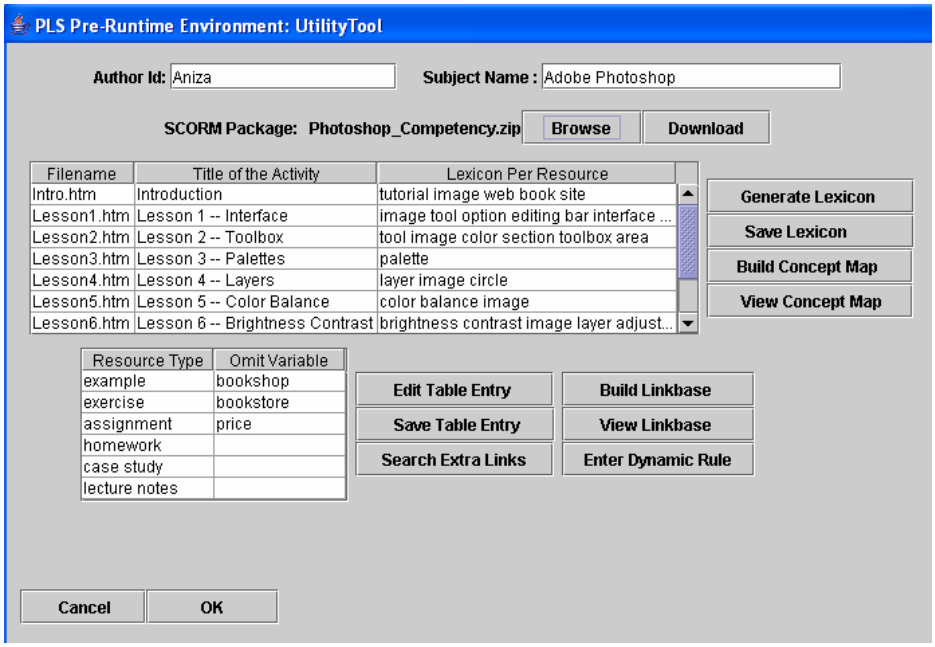

Figure 6. 10: UtilityTool interface

A list of lexicon for every SCORM's resource is generated by clicking the '*Generate Lexicon*' button after a SCORM package has been downloaded. By clicking on the '*Build Concept Map*' button, elements in every lexicon per resource will be compared with the title element of the activity node defined in the manifest file. As a result a concept name will be deduced for every learning resource in the SCORM package. After the relationships among the concepts have been automatically established, a concept map will be constructed.

Prior to build a linkbase, a course author or a course designer must first need to click on the '*Search Extra Links*' to allow the system to search the Web for potential links

based on the following information: subject name, concept name, resource types and omit variables. Once the user has been notified that the search operation is completed, they can then click on the '*Build Linkbase*' button to start the process of building up a linkbase based on a FOHM data model.

6.5 Previous Work in Automatic Concept Extraction

The automatic extraction of concepts from the resources is not a new approach. For instance, (Ramirez and Mattman, 2004) have proposed an approach to increase the potential of the current search engines by automatically extract page concept from the Web pages returned by heterogeneous search engines. The concepts obtained are then presented to the end user in the results page returned by the search engine along with the other information such as page's title, and page's description.

However, what is new in the PLS approach is that after the inference of the potential concept names, a matching process is performed whereby the output is matched with the title element of the SCORM manifest to deduce the most significant concept name to represent the resource. By considering the title element in deriving the primary concept from the auto generated page lexicon, the chosen concept name can be confirmed as the most significant one. It is worth recalling that the main purpose of generating a concept map in the PLS system is to enable the extraction of learning concepts from every resources in the SCORM package so that a knowledge model for the package can be built.

6.6 Summary

This chapter deals with the process of authoring the concept map and the linkbase for the PLS system. Both concept map and the linkbase are the most important elements required in the next PLS designing phase which will be covered in the following chapter.

7 Personalised Link Service: Run-Time Environment

Chapter 7 reports the process of designing and developing the run-time environment for the Personalised Link Service (PLS). The PLS Run-Time Environment aims to dynamically deliver a list of personalised alternative resources to the SCORM Run-Time Environment (SCORM RTE, 2004) based on the inputs received from the PLS Pre Run-Time Authoring Environment as described in Chapter 6.

The processes involved are primarily described according to the following PLS implementation layers: PLS Client and PLS Web Service. The PLS Client layer mainly represents the functional model of the PLS adaptive engine. The PLS Web Service is primarily composed of a contextual link server i.e. Auld Linky (Michaelides et al., 2001) that is responsible for performing adaptive queries to linkbases, based on a requested context from the Client.

This chapter also highlights the method used in permitting PLS to dynamically adapt the additional links to the possible changes in user's browsing behaviours (e.g. selection patterns of the user's preferred links' types) while interacting with the system. The method introduces two sets of implementation rules for generating adaptive links: static and dynamic rules. The process of modelling and implementing these rules will be covered at the end of this chapter.

7.1 High Level Conceptual View

In an augmented SCORM environment, SCORM Run-Time Environment (SCORM RTE) (SCORM RTE, 2004) becomes the client to the PLS Client whose becomes the client to the PLS Web Service. Therefore, in this environment, there are three applications involved: SCORM RTE, PLS Client and PLS Web Service. Both the PLS Client and the PLS Web Service constitute the PLS system which is the main product of this thesis. This section describes in greater detail about the development process of both applications.

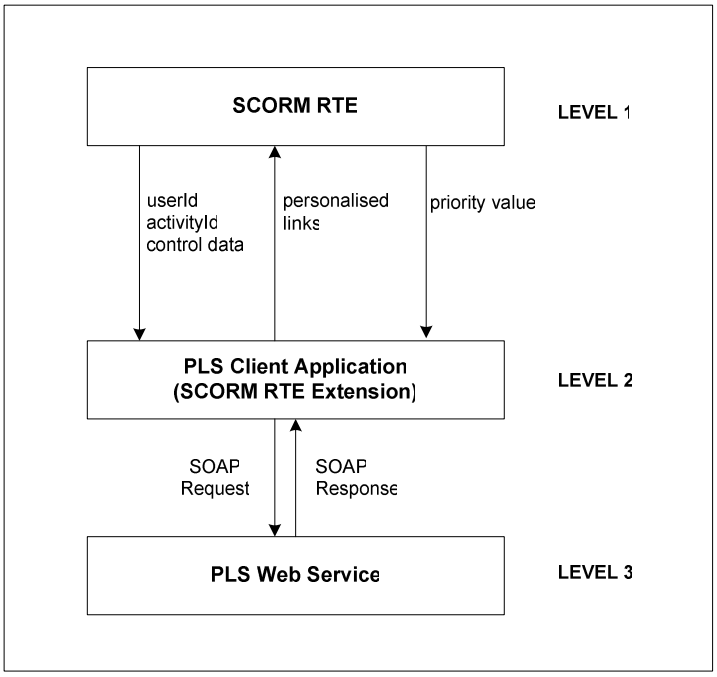

Figure 7. 1: Conceptual view of the PLS implementation layers

Figure 7.1 provides the high level conceptual view of the data flow between the implementation layers of the PLS system. The topmost layer (*level 1*) is the SCORM RTE (SCORM RTE, 2004). SCORM RTE allows a learning management system to deliver SCORM courseware according to the SCORM specification. This layer contributes to the PLS Client by supplying data pertaining to the SCORM user,

learning activity and navigational control data, and providing a medium for the delivery of the personalised links to the user.

Rather than describing the complexity of the SCORM RTE, this work only emphasises on the parameters that are required from this layer for the execution of the PLS Client. These parameters are a user id, a navigational control data and an activity id. A user id parameter represents the id of the SCORM RTE's user. Its value is obtained from the client database which is maintained by a learning management system. A navigational control data provides information regarding user's navigation action be it selecting an item in the navigational tree menu or clicking a navigational button (*continue*, *previous*, *suspend*, or *quit*). This information will inform the PLS Client of the user's intention, whether a user would like to suspend or quit the application, or just continue learning.

Finally, an activity id parameter represents the id of the learning activity currently experienced by a user within the SCORM RTE. It is worthwhile to recall that SCORM's learning space is composed of several learning activities, and a related learning activity will be associated with a learning resource. This hierarchical tree structure represents the course structure for the SCORM package which is described in a manifest file. Based on the manifest file and the related learning resources in the package, a concept map is generated (see Section 6.2). The concept map contains the learning concepts for every learning resource in the SCORM package, and each concept is indexed with an associated learning activity id. Therefore, when a learning activity that is associated with a particular resource is invoked by a user of the SCORM RTE, PLS Client can infer the learning concept of the resource to be launched by the SCORM RTE.

The implementation layer at *level 2* represents the PLS Client. The responsibility of the PLS Client is to provide the PLS Web Service with a relevant query object so that the SCORM learning environment can be augmented with an appropriate set of links to alternative learning resources. Note that these resources are situated in the Web or a local server, external to the SCORM package. The generated links are adaptively selected to match the learning concept of a resource being delivered by the SCORM

RTE and user model values (e.g. Preferred Learning Style). To fulfil this responsibility, the PLS Client needs to conduct the following tasks:

- Generates and updates a user model automatically
- Tracks user's browsing behaviour within the SCORM RTE
- Tracks user's selection of links within the PLS RTE
- Builds an appropriate query object for the PLS Web Service
- Renders the query result to the user

The final implementation layer (level 3) contains the PLS Web Service. The responsibility of the Service is to feed the SCORM RTE with an adaptive links to alternative resources based on an adaptation criteria requested from the PLS Client. The communication of data between the Service and the Client is based on the Simple Object Access Protocol (SOAP).

The following sections will explain in greater detail about the process of designing and implementing both the PLS Client and the PLS Web Service.

7.2 The PLS Client

The PLS Client is composed of a concept map, an adaptation model, a user model and an adaptive engine. The concept map represents a domain model of a SCORM package. The PLS's adaptation model, on the other hand, consists of a set of rules that defines the associations between user's attributes (e.g. Preferred Learning Styles) and a collection of a predefined types (e.g. determined by the instructional functions of the content) of the learning resources (see Section 7.4.1 for details). These rules, by default, are provided by the PLS system based on the associations deduced from related literature (see Section 7.5). However, a course author is equipped with a simple tool (table-driven interface) if he/she decides to amend these default rules (refer to Section 7.4.1). Both the rules and the concept map are in the form of the XML files as the outcomes from the PLS authoring process. PLS's user model, on the other hand, is auto-generated by the PLS Run-Time Environment based on a recommended

specification i.e. the IMS Learner Information Packaging (IMS LIP, 2000). The generated user model will then be used to serve PLS adaptation process. The following section explains the process of auto-generating and updating a user model for each SCORM-PLS's user.

7.2.1 Auto-Generating and Updating A User Model

User model for the PLS system is developed based on the IMS Learner Information Packaging specification (IMS LIP, 2000). The IMS LIP package is structured in eleven groupings including: Identification, Goal, QCL (Qualification, Certification and License), Accessibility, Activity, Competence, Interest, Affliction, Security Key and Relationship.

However, only three of the elements (Identification, Accessibility and Security Key) are currently being used in the prototype as they are sufficient to demonstrate the feasibility of the PLS architecture for delivering alternative learning resources based on a related user model value (Preferred Learning Style). The *identification* element communicates the user's profile such as user id and name. The *security key* element holds user's security information like a password. Finally the *accessibility* element conveys information about user's preferences such as learning style. An example of a user model produced by the PLS system is depicted in Appendix 3.

Figure 7.2 visually describes the process of generating this user model. The process begins with the PLS system checks on the status of a SCORM-PLS's user. If the user is an existing user, then his/her user model (XML file) will be retrieved from the PLS Client server. Otherwise, the PLS RTE will query the database of a learning management system to retrieve the information about the user such as user name and password. Once the information is obtained, a user model will be created for the user. To facilitate the creation of the user model based on the IMS LIP specification, a template is used. This template contains the following elements: *Identification*, *Security*

Key and *Accessibility*. The attained user's information will then used to fill up the corresponding attributes of the template.

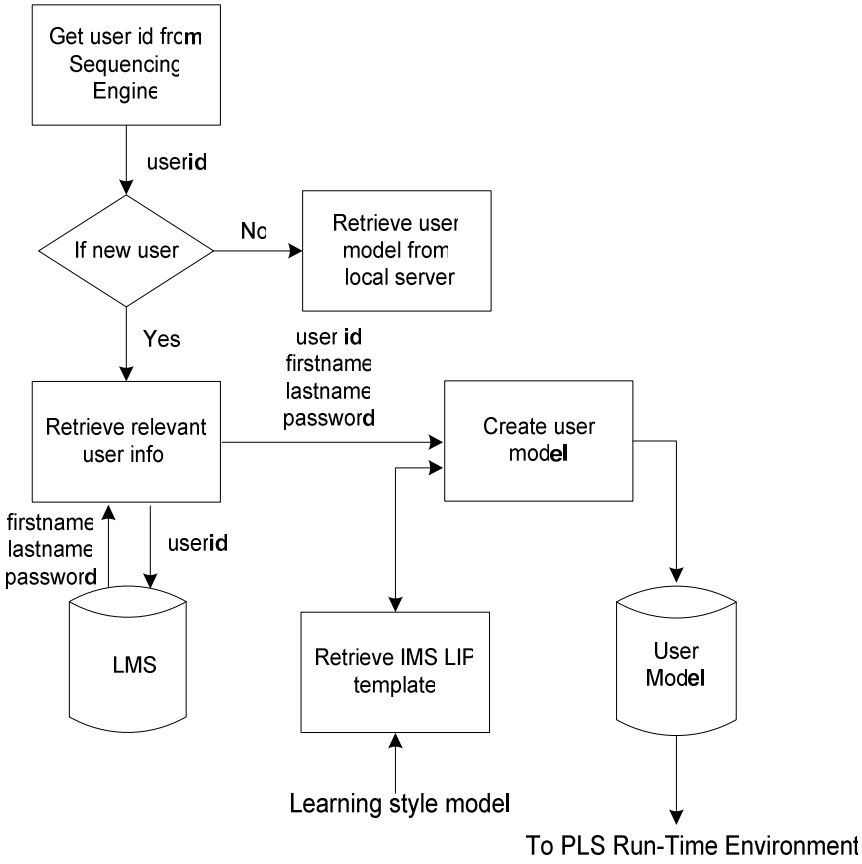

Figure 7. 2: The process of auto-generating a user model for a new user

The client database, however, does not provide information about user's Preferred Learning Style. To address this problem, a URL (Uniform Resource Locator) that points to an XML file that contains the related adaptive rules is manually entered in the template (into its <preference> element). As mentioned earlier, these adaptive rules provide the associations between the required adaptation's attributes (e.g. learning styles) and the content's attributes (e.g. resource types). The associations are established based on the assumption that users with different learning styles preferred different types of learning materials (see Section 7.5 for further elaboration). In the example shown in Figure 7.3, an XML file (*honeymumford.xml*) is referred from the <preference> element of the user model. is used. By traversing this file, a relevant

learning style attribute can be inferred to initialise user's learning style attribute. In the case of this example, the first learning style attribute that appeared in the *honeymumford.xml* (e.g. '*reflector*') which is based on the Honey and Mumford Learning Style Model (Honey and Mumford, 1992) will be used for initialising the user model as shown by its <preference> element in Figure 7.3. The system will later update this attribute based on the result of user's interaction with the system and the associated adaptation rules contained in the *honeymumford.xml*.

```
<preference>
    <typename>
       <tysource sourcetype = "proprietary">
      http://localhost:8080/adlextra/honeymumford.xml</tysource>
        <tyvalue>reflector</tyvalue>
    </typename>
    <comment>learning style preferences</comment>
        <contentype>
            <referential>
                <indexid>learningstylemodel_02</indexid>
            </referential>
        </contentype>
    <description>Honey and Mumford Learning Style Model</description>
</preference>
```

Figure 7. 3: A preference element in the IMS LIP template for the PLS system

Technically, there will be two stages wherein the user model needs to be updated. First is during the user's continuous interactions with the system (e.g. browsing and selecting links), and secondly is when the user chooses to suspend or terminate the application. However, the first stage may require frequent updates to the user model as it occurs during user-system interactions throughout the application. This process can generate a processing overhead. In order to minimise this overhead, a cache is used to maintain the required information such as a user id and user's preferred learning style attribute throughout user-system interactions process. Therefore, by using a cache, the process of updating the user model can only occur once i.e. when a user decides to either suspend or terminate his/her learning session with the SCORM RTE. When the system receives this request from a user, the system will retrieve this cached information and update the relevant attributes of his/her user model.

In the PLS system, the user model is kept within the user's learning management system server. This is mainly because of the following reasons. First is to reassure users that their information is under their own control and that their privacy is protected. The second reason is to be able to allocate the same user model to any other service as wished for by the user.

The following section will explain how the input component (a user model, an adaptation model, a concept map and the SCORM sequencing data) is processed by the adaptive engine in order to build a relevant query for the PLS Web Service in the attempt to deliver personalised links to users.

7.2.2 Building Query Object

Prior to constructing a query object to the PLS Web Service for additional links, PLS Client must first determine the learning concept of the page delivered by the SCORM RTE and related user model values (e.g. Preferred Learning Style). To accomplish this task, PLS adaptive engine must interpret the input deduced from a user model, an adaptation model, a concept map and the SCORM sequencing data.

Figure 7.4 conceptually describes the data flows involved in the adaptive engine. Note that when a requested set of data from the SCORM RTE layer (see *level 1* in Figure 7.1) enters the adaptive engine, it will be processed by either a Concept Tracker or a Preference Tracker.

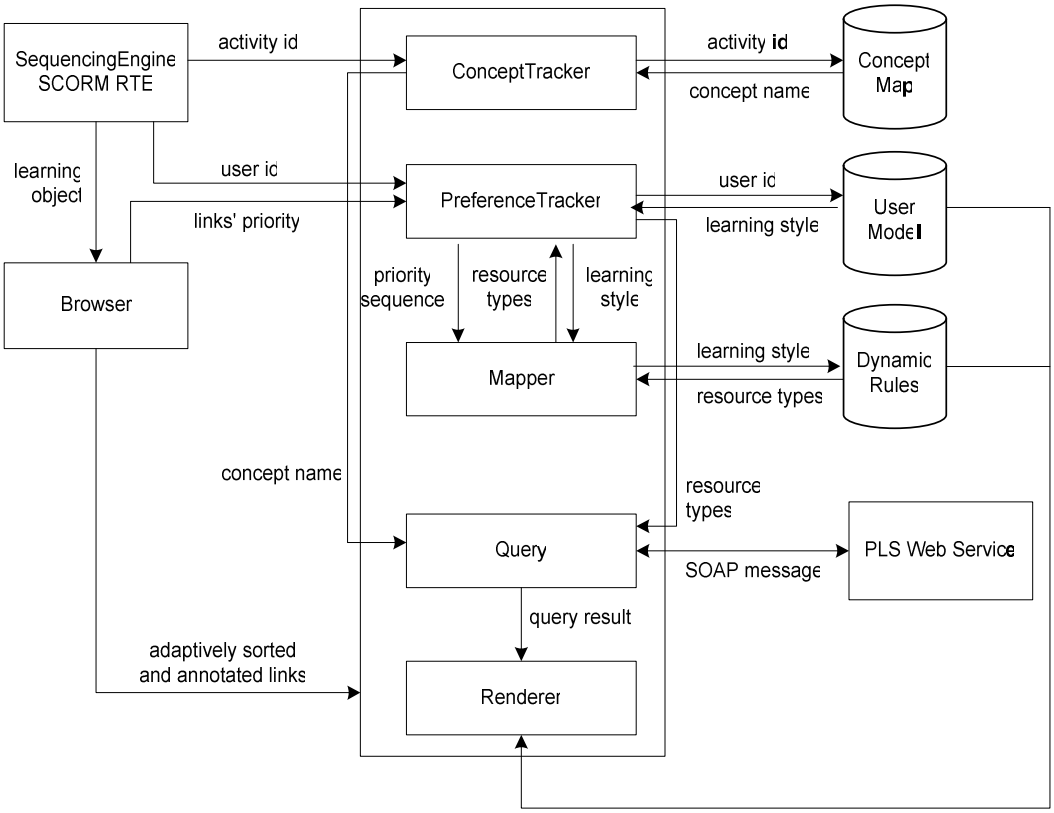

Figure 7. 4: A conceptual view of the data flows circulating the adaptive engine in the PLS Client Application implementation layer.

The Concept Tracker tracks which learning activity (activity id) will be delivered next by the SCORM delivery system. Based on the activity id and a concept map, a learning concept of a resource that is associated to that learning activity can be inferred. The attained concept name will then be forwarded to the Query component.

On the other hand, a user id and navigational control data will be captured by a Preference Tracker. Information on the user id is used to get his/her user model. When a relevant user model is identified, information such as user's Preferred Learning Style is retrieved. This information is then channelled to a Mapper component which will identify the matching resource types for a Preferred Learning Style. To accomplish this task, the Mapper relies on the input of the adaptation rules (dynamic rules) (see Section 7.4 for details). The identified resource types will then be delivered to the Preference Tracker which later forwards it to the Query component.

When the Query component receives the learning concept and resource types from both Concept Tracker and Preference Tracker respectively, it will build a query object based on these parameters and then deliver it to the PLS Web Service in a SOAP message. The Service will process the request and return the query's result to the Query component of the PLS Client.

7.2.3 Rendering the Adaptation Effects

The query result receives by a *Renderer* component from the *Query* component contains a nested vector object which comprises of a list of links (URLs) for every related resource types required by a specific user model. In other words, the result contains a list of links to resources marked as matching the concept taught by the SCORM environment and the required resources types to support a required user model (e.g. Preferred Learning Style). By analysing the user model and the priority value (*high* or *low*) of every resource type associated with a particular learning style, as specified in the dynamic adaptation rules (e.g. *honeymumford.xml*) (see Section 7.4.1 and Appendix 2), the Renderer can adaptively sort the URLs in a decreasing order of importance and annotate the links accordingly.

The associated title or description metadata for each referred page will then be extracted and displayed as a list of personalised links into the extended interface of the SCORM RTE in a browser, as demonstrated in Figure 7.5.

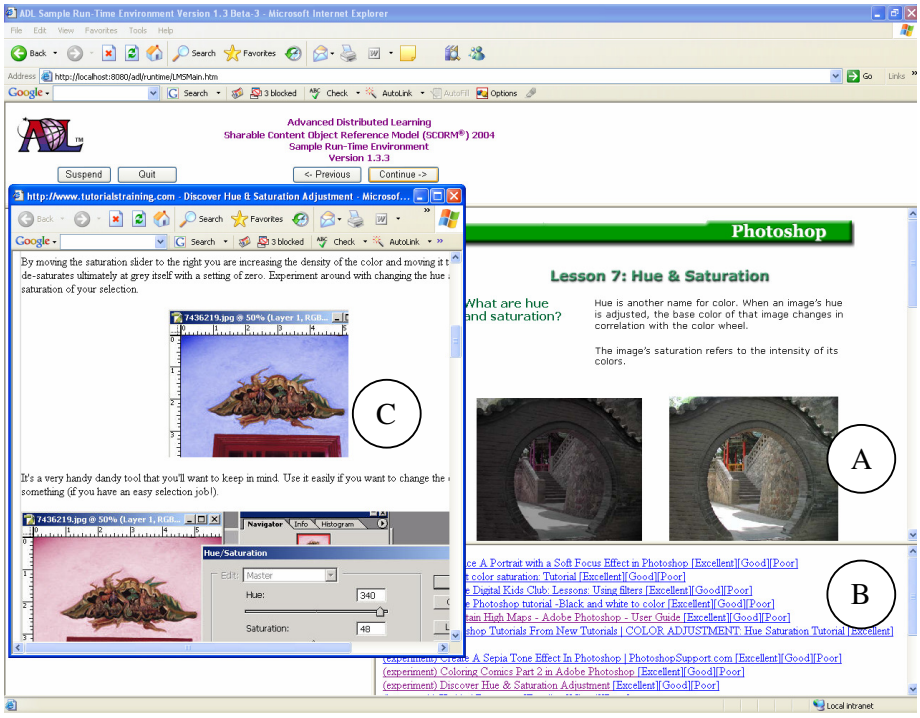

Figure 7. 5: The interface of the augmented SCORM RTE with the recommended links to alternative resources

The above screen shot demonstrates the outcome of the PLS's algorithm whereby the additional links are personalised to the type of learner who prefers experimenting the technique of applying learning concept '*hue and saturation*' in Adobe Photoshop. The list of links to supplementary resources is provided in frame (B) matching the concept being displayed in the primary resources in frame (A). Once a link from frame (B) is selected a pop up window displaying the content of the URL appears in (C), so that the additional resources does not interfere in any way with a course author's intended resources and his/her instructional sequence. Frame (B) is the component that has been integrated with the existing SCORM environment.

7.2.4 Tracking User's Selection of Presented Links

The main task of the PLS Client is to build an appropriate query to the PLS Web Service so that the SCORM environment can be augmented with personalised links that can adapt to current content delivered by the SCORM RTE and changes in user

model values (e.g. Preferred Learning Style). This adaptive environment is achieved through continuous monitoring of user's interaction with both SCORM and PLS run-time environments.

This monitoring task is performed by a Preference Tracker. The tasks include monitoring user's browsing behaviour within the SCORM RTE as well as his/her selection of the presented links within the PLS RTE. The system uses the navigational control data obtained from the SCORM RTE for monitoring user's movement within the SCORM environment. Monitoring user's links selections within the PLS environment, on the other hand, requires the system to retrieve the priority values associated to the links that have been confirmed by a user. It is worthwhile to recall that these priority values will inform the system about a set of resource types preferred by a user. Information about the preferred resource types will allow the system to decide whether a currently deployed learning style should be sustained or changed to the next recommended learning style. The attribute of the next recommended learning style can be derived from the dynamic adaptation rules (see Section 7.4.1. and Appendix 2).

A user can confirm the relevancy status of a selected link by evaluating the learning resource associated to the link according to three levels of feedbacks: *Good*, *Fair* and *Poor*. Every presented links will be associated with these types of feedbacks. The use of these feedbacks is to ascertain that the visited link provides the learning resource that a user perceives as relevant to his/her learning needs. Only the *Good* and *Fair* feedbacks will trigger the system to capture the priority values (high or low) associated with the selected links. Therefore, each time a user has confirmed a selected link as relevant (via *Good* and *Fair* feedbacks), the system will record the priority value associated to that link. When the user moved to a new learning resource (launched by the SCORM RTE), the system will calculate the recorded number of selected links with high and low priority values respectively. If the quantity of high priority link is higher than the low priority link, then the system will infer that for that learning concept, user prefers high

priority link (implies a specific set of resource types) and vice versa. This inference data will then be aggregated into a '*prioritysequence*' variable.

The whole process of monitoring user's links selection for the visited learning concepts continues until a predefined number of concept pages are reached. In the current PLS system, a total of three concept pages are required before the aggregated priority data inside the '*prioritysequence*' variable will be evaluated. During the evaluation process, the sequence of the captured priority values (the content of the '*prioritysequence*' variable) will be continuously compared with every static rule's <*condition*> until a match is found. Note that a static rule is a conditional rule which controls the execution of the dynamic rules (see Section 7.4.2).

When the match <*condition*> is found, the associated rule <*action*> will be executed. If the <*action*> does not require changes in user's current learning style, then the system will proceed with the same learning style which implies continuing using the same set of learning resource types. Otherwise, the next recommended learning style specified in the dynamic rules (XML file) must be derived. As a result, a new set of resource types will be used as one of the parameter to query PLS Web Service for another set of personalised links.

Note that *Good*, *Fair* and *Poor* rating approach is chose because the PLS system only requires a simple and discrete rating method for ensuring that the selected link contains the learning material that a user actually prefer. The rationale of having *Good* and *Fair* feedbacks is that to differentiate useful links from the less useful ones from the selections of links preferred by a user with a specific user model. This is necessary to serve future maintenance of the linkbase, as described in Section 9.3.1.3.

7.3 PLS Web Service Layer

The Service technically embodies a contextual link server named Auld Linky (Michaelides et al., 2001). Auld Linky can dynamically derive links from the linkbase based on the context values delivered by the Client. The data structure of the linkbase is compliant to the FOHM data model (Millard, 2000). The '*tour*' association structure is used to organise links in the linkbase. Each association object is identified by an id and is attached with a context object with the following values: learning concept and resources type. There can be more than one linkbase because Auld Linky allows more than one 'service linkbase' to be dynamically queried at run-time to retrieve additional links according to any given context. In the PLS system, each linkbase is designed to represent a specific subject domain.

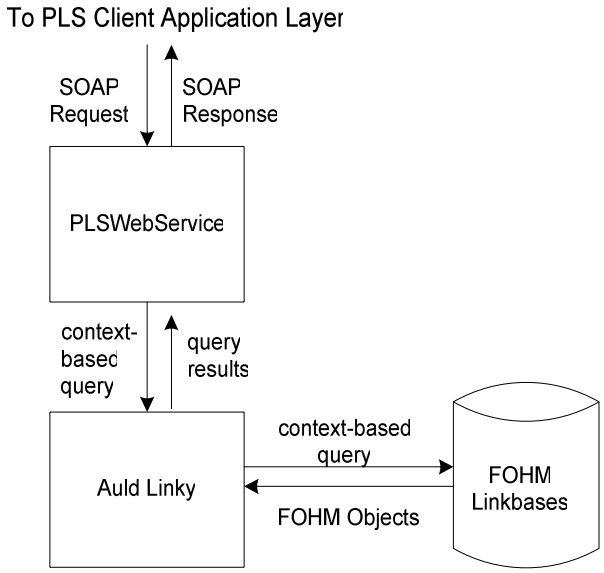

Figure 7. 6: The data flow among the system components of the PLS Web Service layer.

Figure 7.6 illustrates the data flows within the system components in this implementation layer. When the Service receives a SOAP request from the PLS Client application, the SOAP messages will be de-serialised into Java String objects. Each object represents a learning concept, resource type and a '*resourcetype*' attribute respectively. Subsequently, a FOHM context-based query object will be built based on these parameters. Once completed, the query object will be sent to Auld Linky to query

the designated linkbase. In response to the querying process, any FOHM structure that meets the requested context values will be retrieved by Auld Linky. Auld Linky then forwards the result (FOHM objects) to the waiting PLSWebService. At the PLSWebService, the result will be processed and serialised into a nested Java Vector ([[*resourcetypes*,[*url0*,*url1*,*url2*,…]]]). Before returning this result to the calling PLS Client, this vector object will be encapsulated into a SOAP message. When this SOAP message is received by the *Query* component of the PLS Client, it will be interpreted and consequently, the vector object is derived. This vector object is then delivered to a *Renderer* component to be processed and rendered accordingly.

7.4 Implementation Rules for Dynamic Adaptation

In the PLS system, PLS Client queries the Service for adaptive and personalised links based on a user model value (e.g. user's preferred learning style), a learning concept of a resource that will be launched by the SCORM RTE, and dynamic rules. The dynamic rules define which types of learning resources (based on their instructional functions) should be assigned to each learning style identified in a particular learning style model. They are called dynamic because they are editable by course authors or designers. However, to make this adaptation model capable to adapt to changes in user preferences (e.g. Preferred Learning Style) while interacting with the system (selecting the presented links), the PLS system requires another set of rules to control the execution of the dynamic rules. These rules are called static rules because they are less-volatile, so they are embedded in the system. In the following sub-sections, the development and implementation of both rules are further explained.

Before proceeding, it is worthwhile mentioning that the prototype implemented in this thesis works on the assumption that users with different learning style preferences will prefer different resource types. The assumption is based on the literature mentioned in Section 7.5 and has been further supported by (Karagiannidis and Sampson, 2004). However, it is important to understand that the contribution of this work is in the

implementation of an architecture for personalisation depending on some features of a user model. For this prototype the simple example of learning style has been chosen, but this work makes no claims for validating whether this approach is educationally valid, beyond quoting the works of others in this domain.

7.4.1 Dynamic Rules

Dynamic rules define the data input for the adaptation rules. This input can be provided by a course author or determined by some theoretical model such as a learning style model. In the latter case, different models may require a different set of data input as there are many perceptions and interpretations arising from modelling users' learning styles. In order to accommodate these differences the system provides a tool that permits a course author or course designer to update the initial input recommended by the system.

This tool has an interface consisting of a two dimensional table in which rows represent the resource types and columns represent the required user model attribute (e.g. learning styles of a deployed learning style model) as shown in Figure 7.7.

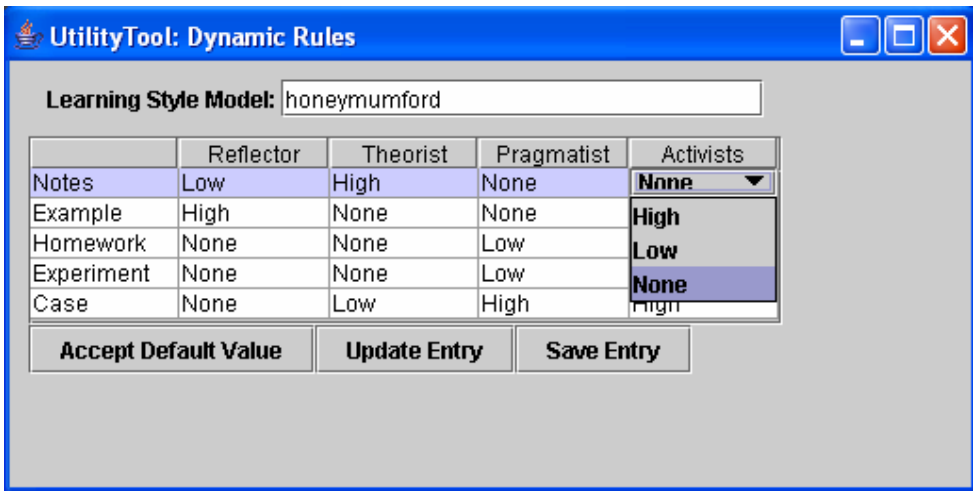

Figure 7. 7: Utility tool interface for dynamic rules entry

The sequence of the attributes determines their correlation factor. Each of its cells is editable with three priority values (high, low, or none) that a user can choose from. These priority values represent user preference level of a corresponding resource type in accommodating a specified learning style.

Once the table entry is completed, the tool will automatically transform all entries into an XML data structure as outlined in a following DTD (Data Type Description) (see Figure 7.8).

```
<!ELEMENT learningmodel (id, learningstyle*)>
<!ELEMENT learningstyle (resource*, changestyle>
<!ATTLIST resource (type, priority)>
<!ATTLIST changestyle (type)>
```

Figure 7. 8: The Data Type Description for the PLS dynamic rules

The *learningmodel* element contains an id attribute that represents the name of a deployed learning style model and a *learningstyle* element. The *learningstyle* element contains the following elements: *resource,* and *changestyle*. There can be more than one resource element for each learning style. Each resource contains two attributes i.e. *type* and *priority*. The *type* attribute represents the type of a learning resource. In the PLS prototype, this attribute indicates the instructional function of a particular resource such as an example or an assignment. On the other hand, a *type* attribute in the *changestyle* element indicates the next recommended learning style. The *priority* attribute signifies the degree of importance of a specific resource type in accommodating a particular learning style. These *priority* values can be assigned by the system using a default entry. This default entry is the outcome of interpreting the requirements of the respective learning style models in regards to a specific type of learning resources. A simple tool is provided to allow course author to edit this default entry.

The generated XML file will become the basis for the adaptive engine to establish association between resource types and learning styles. As a result, the returned links

can be personally generated, sorted and annotated to accommodate each individual's preferences.

7.4.2 Static Rules

Static rules control the execution of the dynamic rules by monitoring user's links selections as previously described in Section 7.4.2. The rules have a form of a conditional rule (*if* <*condition*> *then* <*action*>). The condition is the priority sequence, whilst the action is either to sustain a currently deployed user model feature (e.g. learning style) or change it to another style specified by the dynamic rules.

Static rules use two values to indicate which type of the additional links a user would preferred the most. In Table 7.1, value 0 and 1 displayed in columns C1, C2 and C3 represent low and high priority values respectively. These values are obtained from *Equation (1)* and *(2)*. Currently in PLS system, only three concept pages are required to invoke the process of re-evaluating the deployed learning style (see Section 7.4.2). In this case, there will be eight possible combinations of static rules as displayed in Table 7.1.

C 1	C 2	C 3	Output
1	1	1	1
1	1	0	1
1	0	0	0
1	0	1	1
0	1	0	0
0	1	1	1
0	0	1	1
0	0	0	0

Table 7. 1: Eight possible combinations of static rules. Column *C1*, *C2* and *C3* represent the possible conditions for the rule. Whereas column '*Output*' represents the possible action for the rule based on the evaluation of the associated three conditions.

The column signified by *C1*, *C2*, and *C3* represent a learning concept of the learning resource delivered by the SCORM RTE. Each table entry represents the result of which link's type a user preferred the most for every visited taught concept. This result is obtained by the following equations:

$$Cn = (\sum low\_links > \sum high\_links) = 0 \quad (1)$$
$$Cn = (\sum high\_links > \sum low\_links) = 1 \quad (2)$$

In the above equations, the *low_links* indicates the number of links with low priority being selected by a student when a list of additional links are delivered by the PLS system into the SCORM RTE. On the other hand, *high_links* denotes the number of selected links with high priority value. *Equation (1)* conveys that for a particular concept page most of the selected links are those marked as low priority links. *Equation (2)* indicates that the most of the selected links are those marked as high priority links.

The value for the '*Output*' is obtained by attending to the following rule: *IF ((C3=0) AND (((C1 OR C2)=0) OR((C1 AND C2)=0))) THEN Output=0, ELSE Output = 1*. Only when the '*Output*' is equivalent to 0, the adaptive engine will substitute the current learning style with the next recommended learning style as specified in the dynamic rules, otherwise the current learning style will be sustained. Then based on a newly deployed learning style, a new list of recommended resources types can be deduced from the dynamic rules and a new query for the corresponding links can be sent to the PLS Web Service. However, prior to the substitution process, a learner will be first notified whether he/she would like to proceed with the process of changing the current resource types of the presented links or wish to sustain with the current ones.

7.5 Deriving the Correlation between Resource Types and Learning Styles

This section provides the related findings from the literature which shows that different types of resources are required to facilitate different learning styles. Table 7.2 displays the matrix view of the Kolb's learning style model (Kolb, 1984) along with the suggestions of the relevant type of learning resources for each targeted dimension and learning style. A thorough review of this learning style model can be found in (Coffield et al., 2004).

	Doing (Active Experimentation) (Eftekhar and Strong, 1998): simulation, case study, laboratory/experiment, field work. project, homework (Hartmann, 1995): simulation, case study, homework	***Watching (Reflective Observation)*** (Eftekhar and Strong, 1998): log, journal, discussion, brainstorming, thought question, rhetorical question (Hartmann,1995): log, journal, brainstorming
Feeling (Concrete Experience) (Eftekhar and Strong, 1998): laboratory/experiment, observation (example/demonstration), primary text reading, simulation/game, field work, trigger film, reading, problem set example (Hartmann, 1995): laboratories/experiment, field work, observations/example, trigger film	***Accommodator / Activist*** (Clark, 2000): simulation, case study, homework <u>recommendation:</u> simulations (h), case study (h), homework (h), example (l), experiment (l), fieldwork (l), trigger film (l)	***Diverger / Reflector*** (Clark, 2000): log, journal, brainstorming, observation/example <u>recommendation</u>: example (h), log (h), journal (h), brainstorming (h), experiment (l), fieldwork (l), trigger film (l)
Thinking (Abstract Conceptualisation) (Eftekhar and Strong, 1998): lecture, academic paper, model building project, analogy (Hartmann, 1995): lecture, academic paper, analogy	***Converger / Pragmatist*** (Clark, 2000): laboratory/experiment, field work, observation/example, case study <u>recommendation</u>: case study (h) experiment (l), homework (l), field work (l), simulation (l)	***Assimilator / Theorist*** (Clark, 2000): lecture, academic paper, analogy, theory reading <u>recommendation</u>: lecture (h), academic paper (h), analogy (h), log (h), journal (h), brainstorming (h)

Table 7. 2: Inferring resource types recommendation for every learning style (on Kolb/Honey and Mumford Learning Style Model) based on the related findings in the literature

The objective of this matrix (Table 7.2) is to facilitate the process of analysing findings from the literature pertaining to the correlation between resource types and learning styles. The analysis will enable the inference of the appropriate recommendations of which resource types can better accommodate a particular learning style. Note that (Eftekhar and Strong, 1998) and (Hartmann, 1995) have suggested the relevant resource types (based on their instructional values) for each horizontal (*Active Experimentation* and *Reflective Observation*) and vertical dimensions (*Concrete Experience* and *Abstract Conceptualisation*) of the Kolb Learning Style Model (Kolb, 1984). At the intersection of each dimension, four types of learners (or learning styles) have been established (*Accommodator*, *Diverger*, *Converger* and *Assimilator*). For each learner type, (Clark, 2000) has recommended a set of resource types as shown in the matrix (Table 7.2).

Since Honey and Mumford Learning Style Model (Honey and Mumford, 1992) was basically derived from Kolb's theory (Kolb, 1984), each of its proposed learner types (*Reflector*, *Theorist*, *Pragmatist*, and *Activist*) are often associated with Kolb's learner's types (i.e. *Reflector* with *Diverger, Theorist* with *Assimilator, Pragmatist* with *Converger,* and *Activist* with *Accommodator*, as depicted in Table 7.2).

A list of resource types under the '*recommendation*' field shown in Table 7.2., are obtained by comparing resource types from these three sources: those suggested by (Eftekhar and Strong, 1998) and (Hartmann, 1995) for each Kolb learning style dimension, and those recommended by (Clark, 2000) for each type of learners (or learning style). A particular resource type will be chosen if it occurs in at least two of the recommended sources. Also note that symbol '*h*' will be used for a resource type that appears in all three suggestions. It conveys that the represented resource type is given a high priority for serving the associated learning style. Otherwise, if a particular resource type only appeared in two suggestions, a symbol '*l*' is used to indicate that the resource type has low priority in accommodating the associated learning style.

For the prototype implementation, the system only considered the following resource types: *example*, *experiment*, *lecture notes*, *case study*, and *homework*. The decision was made after taking into account the availability of only these types in the Web for our intended subject domain. Table 7.3, therefore, contains the recommended mapping between the aforementioned resource types and learning styles (based on Honey and Mumford Learning Style Model).

Note that the sequence of the learning style categories (*Reflector*, *Theorist*, *Pragmatist* and *Activist*) in Table 7.3 indicates the transition cycle from one category to another as a result from user's interactions with the presented links. For example, if a user model is first initialised with a '*reflector*' category, and his/her interactions' results suggested for changing the currently deployed learning style, the next recommended style will be the next category in the sequence i.e. '*theorist'*.

It is important to recall that user's high preference towards low priority links will trigger the possible changes to his/her current learning style (refer to Section 7.4). Therefore, it is a must to have at least one low priority entry for every learning style category. Otherwise, dynamic adaptation based on user's Preferred Learning Style could not be implemented.

As the initial table entry in Table 7.3 does not satisfy this requirement, therefore, some additional 'low' entry is required. To differentiate the additional low entry from the initial ones recommended from the analysis in Table 7.2, an italic '*low*' is used as depicted in Table 7.3.

Resource Types	Honey and Mumford Learning Styles Model			
	Reflector	**Theorist**	**Pragmatist**	**Activist**
Lecture	*low*	high		
Example	high			low
Homework	-	-	low	high
Experiment			low	low
Case study		*low*	high	high

Table 7. 3: Resource types required to accommodate a particular learning style. This table entry becomes the adaptation input for dynamic rules

This recommended mapping provides the basis for the adaptive engine to decide which resources types should be used to support a particular learning style. However, the thesis makes no in-depth pedagogic claims for the justification of this particular mapping and the teaching expert is free to change the suggested mapping.

7.6 Related Work in Modelling Dynamic Learning Styles Adaptation

Recently there have been a growing number of systems that aim to provide adaptation based on user's learning styles (Carver et al., 1999; Bajraktarevic, 2003; Paredes and Rodriguez, 2004; Karagiannidis and Sampson, 2004; Stash and DeBra, 2003), and this subsection reviews the evidence from the literature as the basis for choosing to model adaptation based on learning style as the example for this prototype.

However, many of the developed systems only decide to re-evaluate the currently deployed learning style depending on the user's knowledge obtained from completing an assessment. Those that offer dynamic learning styles adaptation by just monitoring a user's browsing behaviour are merely a few, for example AHA! (Stash and DeBra, 2003) and TANGOW (Parades and Rodriguez, 2004). These systems model dynamic learning styles adaptation by first initialising a user model with a learning style. Then

during the course, the system will dynamically monitor user's navigational behaviour before it can decide whether to sustain or change the current learning style.

As for the process of generating a user model, the AHA! system can be mentioned where its user model is declared as a concept with attributes values (Stash et al., 2004). These values are initialised through a registration form. From the form, a user can choose his/her learning style attribute. However, if they are not certain about their learning style, the system can make assumption about their preferences by monitoring their browsing behaviours. TANGOW, on the other hand, initialises its user model with the result obtained by a user from completing a set of questionnaires based on Index of Learning Styles of Felder-Silvermann Learning Style Model (Parades and Rodriguez, 2004).

The main difference with PLS approach is that its user model is based on a standard-based specification for a learner profile i.e. IMS LIP. The model is automatically created for every new PLS user within the SCORM open learning environment. During its initialisation process, information about a user's profile is obtained from the database of the learning management system that hosted the SCORM RTE. Since the information about user's learning style is not available from the database, this attribute is initialised with a default value depending on a particular learning style model. Like AHA!, this value will be updated by the system based on the result obtained from monitoring user's browsing behaviour and links' selections.

There also exist different approaches in monitoring user's behaviour. Parades and Rodriguez have proposed the following rules (Parades and Rodriguez, 2004):

- A sequential learner status will change to global learner if the system detects that the student goes systematically back and forward without staying enough time in intermediate pages.
- A global status will change to sequential learner if the system observes that the student always performs tasks (i.e. use sorted links) in the order presented to them.

- A sensing learner status will change to intuitive learner if the system detects that the student goes systematically a step back when they were presented with explanations before an example.
- An intuitive learner status will change to sensing learner if the system detects that the student goes systematically a step back when they were presented with an example before explanations.

However, the mechanism used in detecting changes in users' actions is not explicitly described. In PLS, the mechanism used to deliver dynamic adaptation is explicitly presented through the execution of the dynamic and static rules. Dynamic rules provide the necessary input to the adaptation mechanism in a form of an XML file. It is called dynamic because it permits a course author to re-define system's recommendation pertaining to the suitability of a particular resource type to a particular learning style via a familiar environment (table-driven interface) (see Section 7.4.1). On the other hand, static rules control the execution of this mechanism based on the user's selection of preferable links (see Section 7.4.2). A clear separation of both implementation rules allows more flexibility in the development of an adaptive system. For instance, it alleviates system dependency on any particular learning style model. Moreover, as the rules are delivered in an XML format file, they can be shared and reused by any user model.

AHA!'s monitoring mechanism, on the other hand, is designed to cater for the following two scenarios (Stash et al., 2004):

If the learner specified their learning styles through the registration form:

- If the learner accesses the recommended concept, then the system's confidence that the learner has defined their learning style correctly increases.
- If the learner accesses non-desirable concepts, this confidence decreases. If it becomes lower than some threshold value the system may ask the learner if they want to change to an instructional strategy which corresponds to another learning style.

If the learner did not specify any learning style through the registration form:

- The system may trace the order in which concepts' representations are accessed, which resulted in increasing or decreasing the confidence that the learner has in a particular preference.
- When the system reaches some threshold value the system may inform the learner that their browsing behaviour indicates a preference which corresponds to a particular learning style and they may switch to an instructional strategy which corresponds to that style.

In order to implement AHA!'s user monitoring mechanism, a course author is expected to define the probability value for each page in contributing to a particular learning concept. This value is later used to contribute to the system's confidence in the currently deployed learning style. A course author also needs to specify the threshold value that will enable the system to begin re-evaluating the current deployed learning style, whether to sustain or change the style. This differs in PLS in which it does not require course author to specify any value to control the adaptation process as this is provided by the static rules. Also, if in AHA the confidence value may either increase or decrease each time a user *click* a link, in PLS the system only captures the priority value associated to a link after a user has provided his/her feedback (good, fair, or poor) on the visited link.

7.7 Implementation

PLS uses the Web Service-based service oriented architecture to supplement SCORM Run-Time Environment with appropriate personalised links. Web Service is a new generation of a cross-platform and cross-language distributed computing application. It is about a client and a server exchanging data in XML messages using a standard format or communication protocol known as SOAP (Simple Object Accessible Protocol). A SOAP message consists of SOAP headers and one or more SOAP Body elements inside an envelope. Usually, SOAP exchanges messages over HTTP: the client

POSTs a SOAP request, and receives either a HTTP success code and a SOAP Response or a HTTP error code.

As PLS Web Service is implemented using Apache Axis (Apache Extensible Interaction System), it is easier to implement a Java based PLS Client to enable the Service to be deployed easily. Apache Axis is a third generation of Apache SOAP (Axis Guide, 2005). Axis is a tool that simplifies the deployment of Web services. It implements SOAP in which it handles the process of converting Java objects to and from SOAP message before the message is send and received over the HTTP connection.

In addition to that Axis can automatically generate WSDL (Web Service Description Language) description of the Web Service based on the information on Java classes (i.e. Service) provided by the developer. WSDL is an XML-document that publishes a Web Service provided by a service provider. It describes the operations (services) and messages available to the client application.

It is also important to note that AXIS uses the JAX-RPC method to build Java-based Web Services. The RPC method is a synchronous technique using a client-server model to execute remote SOAP services. The RPC model can be defined using the following steps (SOAP Tomcat, 2002):

- A client application builds an XML document containing the URI (Universal Resource Identifier) of the server that will service the request, the name of the remote method to execute, and the parameters associated with that method.
- The target service receives and unwinds the XML document. It then executes the named method.
- After the named method has returned its results, the results are encapsulated into a response XML document, which is sent back to the calling client.
- The client application receives the response and unwinds the results, which contains the response of the invocated method.

Since PLS uses Axis for its Web Service development, these steps have become the foundation for developing the PLS Web Service's client.

7.7.1 Java-based Web Service Client

The primary step in building a java-based Web Service client using Apache Axis is for the PLS Client to construct its deployment descriptor (i.e. Axis-specific XML file), named as *deploy.wsdd*. The structure of the *deploy.wsdd* file is as shown in Figure 7.9.

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
    xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">
  <service name="PLSWebService" provider="java:RPC">
    <parameter name="className" value="pls.Query" />
    <parameter name="allowedMethods" value="get" />
  </service>
</deployment>
```

Figure 7. 9: The content of a deploy.wsdd file

The deployment descriptor describes how PLS Client can deploy the PLS Web Service Service. The *service name* indicates the Web Service to be invoked. The value for the *provider* attribute signifies that the service is built based upon java RPC communication mechanism. The values of the '*allowedMethods*' and the '*className*' parameters denote that the PLS Client would like to call the '*get*' method of the *Query* class of the *PLSWebService*.

Once a *deploy.wsdd* has been created and the relevant classes are available in the classpath, the Axis Administration Web Service i.e. *AdminClient* will be executed to interpret the *deploy.wsdd* file. As a result, the Axis 'engine' will update its configuration and normally by default saves its state into a global configuration file named *server-config.wsdd*. *Server-config.wsdd* file is auto-generated by Axis after the deployment. Axis will then automatically generate WSDL file based on the information available in the *server-config.wsdd* (Axis Guide, 2005).

When the process of deploying the PLS Web Service is completed and the required query parameters have been obtained from both Concept Trackers and Preference Trackers, PLS Client (via its Query mechanism) will invoke a call to the PLS Service using the Service's WSDL (*http://localhost:8080/MyService/services/PLSWebService?wsdl*). Once the connection has been established the query parameters will be serialised into SOAP-format XML messages, and delivered to the PLS Service. PLS Service will then process the query and the query's result is returned to the PLS Client as SOAP response.

7.7.2 Java-based Web Service

The PLS Web Service is implemented using Apache Axis. Its development involves creating Java server classes using a standard Java codes in which its public class (Query class) is exposed to permit Service's invocation by the Client.

7.8 Summary

This chapter reports the designing and developing processes involved in the PLS Run-Time Environment. It has demonstrated how the SCORM RTE can be extended to include a Personalised Link Service for supplying SCORM with links to alternative resources based on the current concept being taught by the environment and related user model feature. In the next chapter, experiments will be delivered to justify the feasibility of the PLS architecture in augmenting SCORM with personalised alternatives resources.

8 System Experiments and Justification

This chapter delivers the experiments that were carried out to test the feasibility of the PLS architecture in augmenting SCORM Run-Time Environment (SCORM RTE, 2004) with adaptive and personalised links to alternative learning resources.

For each experiment, methods and analysis of the experiments' results are described. The chapter ends with some justification of the experiments' results.

8.1 Aims of the Experiments

The aims of the experiments reported in this chapter are to prove the following concepts (contributions) forwarded in this thesis:

1. The architectural design of the PLS Run-Time Environment is feasible to augment SCORM environment with personalised links to alternative resources at run-time using the relevant information extracted from the SCORM environment (SCORM content package, its run-time sequencing data and its user data).

   To prove this concept, the Web resource referred by the links returned by the PLS system need to be examined in order to measure the relevancy of the returned links. Links are considered as relevant if the referred content matches the current

learning concept being delivered by the SCORM environment and the possible user model value (e.g. Preferred Learning Style). As the thesis holds on the assumption that different user model prefers different types of learning resources, hence showing that the referred Web resource does deliver the intended resource type will consequently imply that different user model will receive appropriate links sets.

2. The architectural design of the PLS Pre Run-Time Authoring Environment is feasible to automatically populate a linkbase with the Web resources using the learning concepts deduced from a SCORM package and a predefined set of resource types, and this auto-generated linkbase is indeed contained many significant links to alternative resources.

   To prove this concept, links in the linkbase have to be examined in order to determine that the links which are relevant to the intended learning concepts and resource types are indeed more in quantity than the irrelevant ones.

To accomplish both aims, the following two hypotheses were inferred so that experiments can be designed to test them:

- Hypothesis 1: Different users' models will receive appropriate links set. In other words, returned links are indeed relevant to the requested resource type.
- Hypothesis 2: Returned links are indeed relevant to the requested learning concept.

It is important to note that this work has been concerned with the design of an architecture for adaptation and personalisation. It is not within the scope of this thesis to evaluate the educational validity of this approach; rather the educational approach has been justified by reference to existing research literature. However, it is appropriate to evaluate whether the system developed does indeed recommend an appropriate set of links, given the assumptions that have been made in the algorithm

and the use of the Google API (Google API, 2002) for automatically populating the linkbase.

8.2 Experiments

The first experiment is to confirm that different user models (e.g. Preferred Learning Style) receive appropriate links sets. The second experiment is to confirm that the returned links are relevant to the intended concept.

Note that in both experiments, the relevancy of the returned links is manually evaluated only by the author for the following reasons. First is the author has a concrete experience in the subject domain used in these experiments (Adobe Photoshop), thus, the author represents a view of a subject expert. The next reason is that a consistent perception of the content relevancy to the targeted learning concept and resource type is crucial to maintain a consistent content rating in the evaluation process. Furthermore, it would be a complex task and consume a longer period of times if many users are used to evaluate the content of 200 links as research from cognitive science denotes that different individual perceive information differently. The author, however, suggested that this kind of evaluation should be conducted in the future by another research work related to educational field so that the usefulness of the alternative learning resources delivered by the PLS system in improving one's learning performance can be evaluated. This requirement is however, beyond the scope of this thesis.

8.2.1 Hypothesis 1: Different User Models Receive Appropriate Link Sets.

In this experiment, concept 'layer' in Photoshop is used as the intended learning concept. 50 potential links i.e. the 10 links for each resource type that were chosen by the link generation algorithm and the adaptation input shown in Table 8.1 is used for defining its dynamic rules. For this experiment, the PLS system is executed using

different user models and Table 8.2 confirms the expected number of high and low priority links, and the corresponding resource types that are sorted in decreasing order of importance.

Resource Types	Honey and Mumford Learning Style Model			
	Reflector	**Theorist**	**Pragmatist**	**Activist**
Lecture notes	low	high		
Example	high			low
Homework	-	-	low	high
Experiment			low	low
Case study		low	high	high

Table 8. 1: Adaptation input for defining dynamic rules. Here, the learning styles are defined by the Honey and Mumford Learning Style Model (Honey and Mumford, 1992).

	Honey and Mumford Learning Style Model			
	Reflector	**Theorist**	**Pragmatist**	**Activist**
High priority links	10	10	10	20
Low priority links	10	10	20	20
Associated resource types	Example, Lecture notes	Lecture notes, Case study	Case study, Experiment , Homework	Case study, Homework, Example, Experiment

Table 8. 2: Number of links delivered to user based on four distinctive user models along with their associated resource types

The resources referred by the returned links were then manually inspected and analysed. Table 8.3 shows the extent to which the returned links were actually judged to be of the resource type with which the algorithm had labelled them. The links returned by the PLS system were evaluated and categorised into five categories. The first category which is 'All Relevant' signifies that 99% of the content of the referred page represent the named resource type. The second category which is 'Mostly

Relevant' denotes that 80% of the content of the referred page represents the named resource type and 20% is another resource type. The third category which is 'Some Relevant' conveys that only 50% of the referred content page is of the correct resource type, while the rest is some other resource type. The forth category which is 'A Bit Relevant' denotes that the intended resource type constitutes between 10% to 30% of the content page. Finally, 'Not Relevant' represents irrelevant links for resource type.

	Category				
Resource Types	**All relevant**	**Mostly relevant**	**Some relevant**	**A bit relevant**	**Not relevant**
- lecture notes	3	2	2	-	3
- example	4	3	2	1	-
- homework	3	2	3	-	2
- experiment	3	3	3	1	-
- case study	3	2	1	2	2
Total	16	12	11	4	7

Table 8. 3: The relevancy of the returned links to the requested resource types for concept 'layer.

This experiment assumes that relevant links are represented by the first four categories (from 'All Relevant' to 'A Bit Relevant') whereas only the last category represents the irrelevant links. With this assumption, both relevant and irrelevant links are calculated as shown in Table 8.3. Results derived from Table 8.3 confirm that 86% of the returned links produced by the automatic link generation algorithm are relevant to the requested resource types. On the contrary, only 14% of the returned links are considered irrelevant. The reasons for their irrelevancy to the intended resource type vary: they may represent a course outline, index pages, or mismatching of the resource types e.g. a page specified as 'example' turns out to be an 'experiment' page. In this case the same resource might have been linked twice. The ability to personalise and adapt links to a current user model makes PLS distinctive from any common search engine such as Google.

A similar experiment was repeated to test the relevancy of the returned links to the resource types for another three learning concepts i.e. selection tool, contrast brightness and hue saturation. The results are displayed in the subsequent tables below (Table 8.4, 8.5, 8.6).

Resource Types	Category				
	All relevant	**Mostly relevant**	**Some relevant**	**A bit relevant**	**Not relevant**
- lecture notes	5	1	3	-	1
- example	3	2	2	-	3
- homework	-	1	2	4	3
- experiment	2	1	3	3	1
- case study	2	-	1	-	7
Total	12	5	11	7	15

Table 8. 4: The relevancy of the returned links to the requested resource types for concept 'selection tool'.

Resource Types	Category				
	All relevant	**Mostly relevant**	**Some relevant**	**A bit relevant**	**Not relevant**
- lecture notes	3	-	-	-	7
- example	4	4	-	1	1
- homework	1	-	1	3	5
- experiment	1	-	3	5	1
- case study	-	-	2	-	8
Total	9	4	6	9	22

Table 8. 5: The relevancy of the returned links to the requested resource types for concept 'contrast brightness'.

Resource Types	Category				
	All relevant	**Mostly relevant**	**Some relevant**	**A bit relevant**	**Not relevant**
- lecture notes	4	1	-	-	5
- example	7	1	1	1	-
- homework	-	1	1	-	8
- experiment	3	-	3	4	-
- case study	4	-	-	2	4
Total	18	3	5	7	17

Table 8. 6: The relevancy of the returned links to the requested resource types for concept 'hue saturation''.

Result from Table 8.4 shows that for learning concept 'selection tool', 70% of the returned links are significant to the intended resource types, whereas only 30% of the returned links are considered as irrelevant. For learning concept 'contrast brightness', result from Table 8.5 denotes that only 56% of the returned links are delivering the requested resource types. Finally, for learning concept 'hue saturation', result from Table 8.6 depicts a convenience 66% of the returned links are appropriate to the intended resource types.

Overall, the analysis of the relevancy of the returned links to the intended resource types for four learning concepts (i.e. layer, selection tool, contrast brightness and hue saturation) has delivered a total of 69.5%. This result is encouraging after considering the fact that the linkbase is built without any human intervention. It is also important to note that this result is the starting point and will be improved over time when the suggested enhancements described in Section 9.3 are implemented. As the system holds to the assumption that a user with a particular learning style preferred a specific types of learning resources, the relevancy of the links to the requested resource types can also indicate that users with different learning style preferences will receive an appropriate set of learning resources.

8.2.2 Hypothesis 2: Returned Links are Relevant to the Intended Concept

This experiment analyses the relevancy of the returned links to the intended taught concepts for the following concepts: 'layer', 'selection tool', 'contrast brightness' and 'hue saturation' in the Photoshop domain. An assumption is made in this experiment in which all the five resource types (lecture notes, example, homework, experiment and case study) are delivered to the user to supplement each aforementioned taught concept delivered by the SCORM RTE. The assumption is necessary in order to measure the suitability of all the links in the linkbase to the intended concepts. 10 potential links will be delivered for each resource type so there will be 200 links involved in this experiment.

As with the previous experiment the links returned were manually inspected to ascertain the relevance of the resource to each intended concept. The results are shown in Table 8.7. Again, the divisions indicate that 99%, 80%, 50%, 10%-30% or 0% of the resource was judged to be about the intended concept.

	Category				
Learning Concept	**All relevant**	**Mostly relevant**	**Some relevant**	**A bit relevant**	**Not relevant**
Layer	20	6	13	1	10
Selection tool	14	5	12	8	11
Contrast brightness	2	4	13	10	21
Hue saturation	14	5	12	8	11
Total	50	20	50	27	53

Table 8. 7: Results of the content analysis for each referred links in associating to the taught concepts (i.e. in Photoshop) delivered by the SCORM Run-Time Environment

Like in the previous experiment, an assumption is made that the relevant links are represented by the first four categories (from 'All Relevant' to 'A Bit Relevant')

whereas only the last category represents the irrelevant links. Therefore, the overall result shown in Table 8.7 has proven the hypothesis in which 73.5% of the links returned by PLS are relevant to the taught concept of the learning resource delivered by the SCORM Run-Time Environment. Whilst, only 26.5% of the returned links are insignificant in which the referred content does not contribute to the understanding of the taught concept. The percentage of relevant and irrelevant links for each intended learning concept is as shown in Table 8.8.

	Relevant(%)	**Irrelevant (%)**
Layer	80	20
Selection tool	78	22
Contrast brightness	58	42
Hue saturation	78	22

Table 8. 8: The percentage of relevant and irrelevant links for each learning concept

A further analysis was conducted to determine the content type of the irrelevant links. The analysis result is displayed in Table 8.9. The result depicts that pages which conveyed a course outline for the subject domain has made up the largest part of the irrelevant pages. This finding denotes the future requirement for the PLS Pre Run-Time Authoring Environment to include an intelligent search filter that can 'semantically' analyse the content of each search result before they are included in the linkbase, as described in Section 9.3.1.1.

Type of Page Content	**No. of Pages**	**Percentage**
Course outline	31	58.4
Index pages	6	11.3
Pages for promotional purpose (tool, training, or workshop)	5	9.4
Some work experiences which involved Photoshop	2	3.8
About other tool/course which include the use of Photoshop as the image editing tool	3	5.7
Same as primary material	2	3.8
Forums/Discussion page	2	3.8
Page not found	2	3.8
Total number of irrelevant pages	53	
Percentages of irrelevant pages to the total query result	26.5 %	

Table 8. 9: The possible page content referred by the irrelevant links

Another interesting finding from this result is that the quantities of other types of irrelevant pages are small. For instance, there is only 11.3% of the irrelevant links belong to the index pages. Meaning that, only 3% of the total pages (200 pages) are index pages. In this regards, PLS system has achieved its aim to deliver a Web page with appropriate learning concept as opposed to the querying result normally obtained when querying learning object repositories such as MERLOT (see Section 6.3.1 and Section 6.3.2). By eliminating the intermediary navigational layers, PLS's users can have quick and direct access to the required Web page content.

The result also shows that the PLS approach manages to sustain only a small percentage of pages used for the promotional and training purposes of the product (Adobe Photoshop). There is also a small number of link errors ('page not found') in which it constitutes only 1% of the total pages available in the linkbase. Finally, the analysis has also identified two learning resources that are of the same content as the existing primary materials offered by the SCORM package. In this case, links to both resources are considered as irrelevant.

8.3 Justification

The success of both experiments has confirmed that the PLS Run-Time Environment is capable of delivering an appropriate set of alternative links to individual users with different user model attributes (e.g. Preferred Learning Style) and appropriate to the taught concept delivered by the existing SCORM Run-Time Environment. It has also confirmed that the auto-generated linkbase does indeed contain many significant links.

Overall, the experiments' results have justified the feasibility of the Personalised Link Service's architecture in its endeavour to attain the following objectives:

i. To supply SCORM RTE with additional learning resources at run-time in order to complement its predetermined course content

ii. To generate a personalised learning experience by providing alternative learning resources according to the users' preferences for a particular set of resource types
iii. To demonstrate that system's understanding about the learning concept of each learning resource delivered by the SCORM delivery system can be established by deducing the knowledge model from the SCORM course content package
iv. To demonstrate the use of the IMS Learner Information Package to store and maintain a user's learning style attribute and a reference value to adaptive rules definition (dynamic implementation rules)
v. To demonstrate the implementation of Auld Linky as a Web Service.

In addition to the above attainments, the experiments have also demonstrated that the auto-generated linkbase is able to provide a reasonable result. This result denotes that by using a special method to query the existing search engine and by deploying a linkbase based on FOHM data structure, some tedious tasks (locating useful Web resources and constructing a linkbase) can be automated. The ability to automate these processes can promote the realisation of a large scale adaptive hypermedia system.

The same experiments can also be carried out to determine the relevancy of the links returned by the PLS system to support other types of user features for personalisation (see Section 9.3.2.1) providing that the following conditions must first be satisfied:

i. Links must also be marked up according to their content types (audio, video, graphics, and text). To meet this requirement, an additional search mechanism is needed to analyse Web page for its content type and retrieve the related links.
ii. A new set of adaptation input (dynamic rules) is required to define personalisation based on other user features.

In regards to the author's attempt to automate the process of identifying potential Web resources, results from this experiment can provide the statistical data of the types of the Web content that contributes to the irrelevant links. This information will be useful for devising the future solution to improve the relevancy of links returned by the PLS system, as described in Section 9.3.1.1 and Section 9.3.3.1.

8.4 Summary

This chapter reports the experiments' results that confirm the feasibility of the PLS architecture to deliver personalised and adaptive links to students within the SCORM open learning environment. The chapter ends by a providing some reflections on the experiments' results. The following chapter suggests future work that can improve the relevancy of the returned links.

9 Discussion, Conclusion and Future Work

9.1 Discussion

In this section, the author discusses and reflects some issues arising from the design and development of the system.

9.1.1 Reflecting on the Use of the Existing Web Resources as Opposed to Learning Objects

In the current PLS implementation, only existing Web resources are considered as its source for additional links. This section provides the rationales for opting to utilise Web resources rather than Learning Objects.

Conceptually, the use of Learning Object is better than Web resources because it promotes the idea of context independent and self-explanatory learning material. Therefore, a Learning Object should contain a descriptive set of metadata to ease its discovery and promote content trust (via expert verification of its reliability).

However, there are different Learning Object scenarios in most of the learning object repositories today. For example in MERLOT (the most comprehensive learning object repository), many of its so-called Learning Objects are just Web pages that could

hardly be called Learning Objects. For instance, the Web page is not self-explanatory (still lacking important metadata such as learning concept), is not self-contained and contains multiple learning objectives. Moreover, many of the MERLOT query results often deliver index pages in which a user has to go through a long list of URLs in order to get to the appropriate site that host the required Web page.

In this regards, using the existing Web resources seems to be indifferent from using a Learning Object. Author's observation is shared by many others as posted in the (EdTechPost, 2004; cogdogblog, 2005). Unfortunately, there is yet no empirical evidence being published pertaining to this scenario. Furthermore, as Web resources are highly accessible (via Google API) as opposed to Learning Objects, they have become the PLS choice for its source of potential links at this moment.

It is worth mentioning that even there is a Learning Object, there are a few reasons why it would still not be as desirable as Web resource. First is the fact that most Learning Objects are stored in learning object repositories that are not as accessible as the Web. Secondly, in order to be useful, Learning Object needs to be annotated with a comprehensive set of metadata. But "*Who wants to define metadata for every Learning Object*"? Particularly when the generated Learning Object is intended for public domain (has no commercial value). Oppositely, people are more familiar of producing Web resources than Learning Object and it will continually be. Finally, argument about the appropriate size of a Learning Object seems never end. This is reflected in many learning object repositories today in which there exists Learning Objects with variety of sizes, from just an image file to a Web site. Thus, the idea of having a Learning Object that represents an 'atomic' learning concept seems still far from reality so as its promises for reusable (repurpose) content.

9.1.2 Harvesting Educational Metadata from the SCORM Package

Even though the IEEE LOM standard exists for describing resource metadata, public domain digital resources are still hardly equipped with any educational metadata. This makes resource discovery within an educational context difficult. Even the resources within the SCORM content package developed by the ADL technical team that has being used throughout this work, do not contained educational metadata (e.g. learning concept).

In order to alleviate system dependency on a content expert to manually tag resources with the required metadata (learning concept), the PLS system used an automatic approach to harvest the metadata. This approach utilises a term frequency algorithm to extract potential keywords for the metadata from the learning resource in the SCORM package. These potential keywords are then compared with the resource metadata (title element of the learning activity that is associated with the resource) described in the manifest by a course author. Any match will represent the learning concept metadata for the resource. As SCORM imposed the title element as a mandatory, it becomes the best choice to represent human interpretation of the associated resource in terms of its contribution to a particular learning activity.

The choice of using the term frequency algorithm is because it is feasible to deliver the task of identifying the frequency of each term within a Web document. The selection of the possible terms for the concept name (lexicon) is then determined by a dynamic threshold value. This threshold value is obtained by analysing the distribution of a graph that represents the total number of terms with each identified frequency.

9.1.3 How 'Transferable' Is the Linkbase?

An issue that may arise in regards to a linkbase is how 'transferable' the current linkbase is to the other users. At the moment, the linkbase comprises additional links for each potential concept name in the Adobe Photoshop domain. It will benefit those

who are studying or teaching Adobe Photoshop as it automatically accommodates users' needs for supportive materials (e.g. example, exercise, additional notes) to the studied material in hand. However, if the users are from a different subject domain that is totally distinct from Adobe Photoshop, such as physics or biology, they might not benefit from the existing linkbase. Nonetheless, if the users are studying image processing and would like to see how some of the image enhancing techniques are practically applied in an image editing tool, this linkbase might provide some useful help.

9.1.4 FOHM Linkbase as the Intermediary Medium

It is also worth mentioning that the use of linkbases based on FOHM can provide the intermediary medium to overcome the problem of different representations of resource type metadata across many learning object repositories (see Appendix 1). The needs for this kind of intermediary may become evident in the future as it may not seem possible to have only one standard in describing metadata attributes and its vocabulary. This is because each learning object repository normally produces a rather individual reflection of its own perceived organisational needs (Hatala et al., 2004). As further highlighted by (Hatala et al., 2004), it would be useful to provide a mechanism that can 'strive to intermediate between the existing' learning object repositories and services.

In this regards, the author believes that FOHM can be a good alternative for an intermediary platform to maintain contents and links to educational resources for the benefits of other systems such as adaptive hypermedia authoring tools. This is because as a hypermedia model, FOHM has the required level of abstraction for representing links and contents according to the necessary structures needed to serve adaptation as demonstrated in HA^{3}L application (Bailey, 2002). Additionally, future research should be carried out to determine the possibility of maintaining SCORM objects within the FOHM structure. A positive outcome can ensure learning resources especially SCO

(Sharable Content Object) to be easily reused and interoperable with any 'SCORM-compliant' adaptive hypermedia system.

9.1.5 Augmenting Pre-Authored Learning Material with Dynamic Links to Relevant Resources

This work provides an alternative approach to offer personalised learning resources to an open learning environment supported by the e-learning standards and specifications such as SCORM. With this approach, each pre-authored learning material packaged in the PIF (Package Interchanged File) (e.g. SCORM package) can be dynamically associated with other learning resources which the system deems appropriate to supplement the understanding of the pre-authored learning material according to a user with a specific user model.

This approach is fundamentally different from creating dynamic content. It is indeed a pragmatic decision because the aim of the PLS development is to support SCORM and to sustain SCORM RTE's control over the delivery of its content package. Moreover, the author believes that in the context of formal education, it is necessary for the course authors to be in control of the primary learning materials they deliver and the teaching strategy they want it to be delivered to enable a consistent evaluation of their students' performance. Nonetheless, students learning preferences should also be taken into consideration by adapting and personalising the secondary materials (supplementary learning resources) as demonstrated by the PLS. On the other hand, using a dynamic content without a predefined teaching strategy may alleviate this kind of control from a course author, thus, the approach seems more suitable for research-related learning where students are more aware of their learning objectives (the skills often deployed by intelligent or mature students).

The concept forwarded by the PLS can also be advanced to support other applications such as adaptive hypermedia authoring tool and Web search engine. It is anticipated to

be able to support adaptive hypermedia authoring tool by providing learning resources that are suitable for a particular type of adaptations. It can also support the query result of a Web search engine by providing a list of associative links to related Web resources that match the current page concept and user model. In this regard, the PLS system can extend the ability of a recommender system by having a larger scope of adaptations (e.g. user learning abilities, user's knowledge level, user's learning preferences, user's current location, bandwidth of the user's network, and type of user's reception device).

9.2 Conclusion

This thesis presents a Personalised Link Service (PLS) for a SCORM Web-based open learning environment. The novelty of this work is that it integrates Adaptive Hypermedia principles and Auld Linky into SCORM as an adaptive support service without proposing any modification to the SCORM specification. Rather than providing a solution to sequence SCORM course content adaptively, the PLS service offers SCORM users supplementary materials chosen based on their user model values (e.g. Preferred Learning Style).

PLS generates adaptive links dynamically into SCORM by using the information deduced from the SCORM package, its run-time sequencing data, and a user model. The successful implementation of the PLS prototype has demonstrated how the SCORM learning environment can be augmented with links to resources that are about the same topic as the user is studying, and which appear to support the learner's preferred learning style (Abdullah and Davis, 2005). However, the architecture is extensible to support other personalisation.

As the PLS system implements a user model that is compliant to a common specification, consistency can be assured when the model is deployed by many different adaptation services or even between learning management systems.

Moreover, as PLS automates the process of generating and updating the user model for every SCORM-PLS's user, it would increase the likelihood of a standards-based learning environment to adopt an adaptive support service like PLS in the future.

The development of the PLS is also influenced by the motivation to minimise human effort in modelling the domain model and preparing alternative learning resources. An architecture for the PLS Pre Run-Time Authoring Environment has been developed to automate the production of both the concept map and the linkbase. The architecture describes the functional components required for automatically deducing a concept map from the SCORM package, and constructing the linkbase.

In the process of inferring learning concepts from the SCORM's resources, the PLS approach compares the generated outcome from both automatic (computer processing via the term frequency algorithm) and human (via title assignment for the related item element in the manifest file) processing (Abdullah and Davis, 2005). Results from the automatic method will produce a set of lexicons from each lesson page in the SCORM package. Based on the assumption that the title of an element in the SCORM package often conveys a meaningful description of a corresponding learning resource, any match found when comparing the lexicon with the title will infer a reliable learning concept. As stated by Cravern, "*the combination of both human and computer processing is the most effective means to generate metadata*" (Cravern, 2001).

The PLS Pre Run-Time Authoring Environment also aims to alleviate system dependency on a course author to provide alternative learning resources for every learning resource in the SCORM package. This aim is accomplished by automating the process of locating the potential resources from the Web and tagging each resource with the required metadata for adaptation (e.g. learning concept and resource type). Once the resources have been identified, a FOHM structured linkbase will be automatically constructed. Despite the automation process, the experimental results in Chapter 8 have demonstrated that around 70 percent of an automatically generated

linkbase contains links relevant to the intended taught concepts and resource types. This encouraging result also shows that PLS approach can leverage the existing Web as a source for alternatives contents. However, to further improve the relevancy of the retrieved resources, the mechanism used for locating and analysing potential resources should be enhanced by integrating research outcomes from other research fields such as Natural Language Processing as described in Section 9.3.3.1. It is worth to recall that in locating the potential Web pages, the PLS system has utilised the Google's advanced search query (Google API, 2002) in order to reduce the number of irrelevant pages. With a click of a button (see Section 6.4), the process of searching for additional resources for each learning resources in the identified SCORM package will launch. The search parameters will contain the following attributes' values: learning concept (extracted from the SCORM package), the required resource types and the information about the subject domain.

The development of the PLS system has also introduced a new approach for defining adaptation rules. The deployment of a 'configuration table' provides the end-users (course author) with a familiar environment to indicate a collection of resource types (based on pedagogy function or instructional role of the content) that they consider important to accommodate a particular learning style. These learning resources to learning style mapping rules are stored in an XML file which can be directly referenced from a user model (via accessibility element of the IMS LIP). These rules are known as dynamic rules. Note that, even though the prototype has demonstrated an adaptation scenario based on a user's Preferred Learning Style, the concept of a 'configuration table' can be advanced to support other types of personalisation such as learning disabilities and background knowledge.

Finally, the PLS has demonstrated an explicit mechanism in generating dynamic learning styles adaptation. Note that the mechanism holds on to the assumption that different learning styles require different sets of resource types and that a user's learning style may dynamically change while learning. The mechanism comprises of

dynamic and static rules. As described earlier, dynamic rules allow an editable adaptation input. On the other hand, static rules control the execution of the dynamic rules based on the user action (e.g. selection of links) while interacting with the system.

To conclude, this work has achieved its goal in augmenting a SCORM open learning environment with a personalised link service. The implementation of the PLS has demonstrated that the service has enabled the SCORM environment to offer its users with alternative resources at run-time. The relevancy of the returned links to the taught concept and the required resource types has been shown in the experimental results presented in Chapter 8. Above all, PLS has demonstrated a new methodology in integrating Adaptive Hypermedia and AuldLinky into SCORM that is flexible enough to adapt to the future version of the specification. It has also delivered a method to automatically generate a knowledge model from the SCORM package, locate potential additional learning resources and construct the linkbase. Finally, PLS has introduced an alternative way to define adaptation input and provided an explicit mechanism for achieving dynamic learning styles adaptation.

From the educational point of view, PLS makes it possible for the SCORM learning environment to deliver learning materials that are preferable to both teachers (primary materials of their choice) and students (supplementary materials of their choice to assist their understanding of the primary materials). That is, by providing many alternative learning resources to supplement the primary resources, it permits a student to explore around the taught concept using their preferred learning resources (Abdullah and Davis, 2005).

9.3 Future Work

This section describes future work of the PLS system. The description is divided into two parts in which each deliver the recommendations on how the existing architecture for both PLS Pre Run-Time Authoring Environment and PLS Run-Time Environment

could be enhanced to improve the relevancy of the returned links and advance PLS to include other user features and adaptive techniques. This section ends with some recommendations for possible future research directions emanating from this work.

9.3.1 Enhanced PLS Pre Run-Time Authoring Environment

Even though in its current implementation the auto-generated linkbase can only offers around 70% of the relevant links, the architecture of the PLS authoring environment has provided a basic framework that can easily be extended for improving the relevancy of the returned links.

9.3.1.1 Improved the Mechanism to Locate Potential Web Resource

The PLS Pre Run-Time Authoring Environment provides the functional component for the automatic inference of a concept map from a SCORM package, the discovery of the potential learning resources from the Web and the automatic generation of a linkbase.

In the quest to discover high quality potential resources, PLS search mechanism (see Figure 6.1 in Chapter 6) should be enhanced as follows. First is to include a mechanism to analyse and filter search results before a linkbase is built. This mechanism may require adopting the Natural Language Processing techniques in order to 'semantically understand' the content of a referenced Web page so that a more accurate interpretation of the page content can be established.

Next is to have a mechanism that is able to perform micro analysis of a Web page in order to extract related (semantically relevant) paragraphs or other significant objects (e.g. picture, diagram, or graph) within a Web page. Subsequently, the relevant paragraph or page objects will be marked with appropriate metadata (for adaptation purpose). The success of this process will enable PLS to adapt the presentation of the content of the supplementary resources according to a user model. As a result, the relevancy of the supplementary materials delivered by the system can be improved.

Finally is to include a mechanism that is able to locate and refer to the appropriate domain ontology available in the net by using the concept map generated from a SCORM course package as the query object. The well established domain ontology can provide the system with a common description of the knowledge items for a target domain and their relationships. This information can be used to edit or refine the auto-generated concept map.

9.3.1.2 Broaden the Choices for the Source of Potential Links

Apart from optimising the Web resources as the source for its linkbase, the PLS 'Search Mechanism' should also be enhanced to include a mechanism that can query the existing learning object repositories and digital libraries based on the required metadata. However, this process can only be performed to public domain repositories and libraries or those that have granted access. The success of this process can provide high quality learning resources to the system. Above all, students who have a preference towards theoretical materials can definitely benefits the most from this incorporation because the system can now provide them with high quality reading materials such as journal articles.

9.3.1.3 Large-Scale User Evaluation

When the suggested enhancements have been made, the PLS system should then be evaluated by a large number of students across the globe (to avoid bias towards a particular culture) to determine the system's effects on students' learning psychology and performance.

During this evaluation process, students' feedbacks on each visited links (delivered by the PLS) will be contributed to automatically maintain the linkbase. The link with a relevant educational metadata will be given authoritative status according to its popularity among students with similar profile, while an idle link will be automatically

removed from the linkbase. This automatic linkbase maintenance approach is anticipated to overcome the dependency on the subject expert to verify the existing Web content within an educational context thus allow the linkbase to self-evolve over the time.

The author believes that a true effectiveness of an adaptive system like PLS can be accurately reported only if its linkbase is huge and the sampling process for its evaluation is large enough to cover individual differences in learning, taking into account the influential factors towards these differences such as genders, and social cultures. Based on this conviction, the author perceives a small-scale user evaluation which comprises of less than 100 hundreds participants and covers only a similar group of people as not appropriate for justifying the real impact of an adaptive learning application on people.

As this kind of a large-scale user evaluation requires additional research and needs longer time to design and implement, another research is therefore required to evaluate the effectiveness of the PLS system to users. It is also worth noting that it is beyond the scope of this research to evaluate the effectiveness of the alternative learning resources delivered by the PLS system in improving user's learning performance. A successful implementation of the PLS system is instead a proof of concept that by using the information deduced from SCORM package and its environment, SCORM learning environment can be augmented with a personalised link service to alternative learning resources without any mandatory human authoring intervention.

9.3.1.4 As a Collaborative Recommender System

There is no doubt that a massive existence of information available in the Web will continue to grow each day. Adaptive hypermedia techniques can provide users with related cues to facilitate their Web navigation. The techniques, however, require a system designer to have an explicit knowledge of the content space and alternative routes of the application prior to an adaptation. Meaning that, applying adaptive

hypermedia techniques to the existing structure of the Web will require a tremendous effort and time-consuming. Nonetheless, Web users demand a quick solution to the problem of identifying useful Web pages to suit their current needs. This demand has engendered the increasing popularity of recommender systems.

A recommender system can predict items that might be of users' interest based on related information about a user's profile, without the users needing to have sufficient personal experience of all the alternatives (Resnick and Varian, 1997). For example, when a user has the interest in a particular book viewed using the Amazon Web bookshop, the application will offer a list of related books that were bought by the other people who have the same interest with the user. Currently, there are two typical kinds of recommender systems: content-based and collaborative recommender systems. Content-based recommender system makes recommendations by analyzing the similarity between the contents of the items that are ready to be recommended and those that have been previously been marked as liked by the user (Wei et al., 2005). For example, Syskill recommends Web documents based on user's binary rating ("*hot*" and "*cold*") of their interests (Pazzani et al., 1996). Collaborative-based recommender system, on the other hand, match people with similar interests and then recommend one person's highly evaluated items to the others (Goldberg et al., 1992). An example of a collaborative recommender system is GroupLens (Konstan et al., 1997). GroupLens helps people find Usenet news articles on a collaborative basis.

In its current form, PLS system can be regarded as a content-based recommender system as it recommends links to Web resources that are predicted suitable to support both the primary content delivered by the SCORM RTE and the user attribute value. The PLS system can also evolve as a collaborative recommender system by applying the concept of collaborative filtering recommendation technique (Goldberg et al., 1992, Shardanand and Maes, 1995) in maintaining its linkbase. For instance, the recommendation data can be collected by measuring the feedbacks on every link provided by users with related profile.

9.3.2 Enhanced PLS Run-Time Environment

This section describes the possible future works for the PLS Run-Time Environment in order to implement other types of personalisation and enrich the augmented SCORM learning environment with content-based adaptation.

9.3.2.1 Adapting PLS Based on Different User Features

In the experiments carried out in Chapter 8, only the personalisation based on user's Preferred Learning Style is taken into account. However, the PLS architecture design are extensible to support other user model features. For instance, PLS adaptation algorithm can be used for supporting physical learning disabilities. To support this type of personalisation, the possible mapping between the content type of a learning resource and user's learning abilities are as shown in Table 9.1. Once the table entry is completed, the PLS adaptation algorithm will transform the entry into an XML data structure. This XML file will represent the adaptation input (dynamic rules) to support this personalisation.

	Learning Disabilities		
Content Type	Hearing Impaired	Vision Impaired	Dyslexia
Audio/ Narrative text	-	high	-
Video	-	-	high
Graphics	high	-	high
Text	high	-	low

Table 9. 1: Defining adaptation input for user's learning disability.

It is worth recalling that, the adaptation input derived from this table-driven approach is the core ingredient for the adaptation mechanism of the PLS system (see Section 7.4.1).

The second example of a possible user feature for personalisation with the PLS system is a user's background knowledge. This type of user feature is particularly important in an adaptive educational application that serves lifelong learning. This is because in such learning environment, students come from different levels of background knowledge. In this scenario, a service like PLS would be favourable to aid learning in which each primary learning material prepared by a course author is supported by different sets of supplementary materials that suit students with different background knowledge.

	User's Background Knowledge		
Resource Type	An Expert (has lots of working experience in the subject area)	An Intermediate (only knows the basic of the subject)	A Novice (is new to the subject)
Example	-	low	high
Simulation	-	low	high
Assignment	high	low	low
Exercise	low	high	high
Experiment	low	high	low

Table 9. 2: Defining adaptation input for user's background knowledge.

Table 9.2 denotes the possible mapping between learning resource types and users background knowledge. Then based on the PLS adaptation algorithm, an appropriate set of links to supplementary materials can be delivered to each individual users.

9.3.2.2 Implementing the PLS Client as a Collection of Web Services

Currently, PLS Client is not being implemented as a Web Service, however, a possible transformation of its functional components into a collection of Web Services can be done as illustrated in Figure 9.1. This transformation will enable new adaptation Services to be added into the architecture without any modification. For instance, *Knowledge Tracking Service* and *Reasoning Service* can be included to enable PLS to

deliver appropriate links or content to users based on their background knowledge or current knowledge levels.

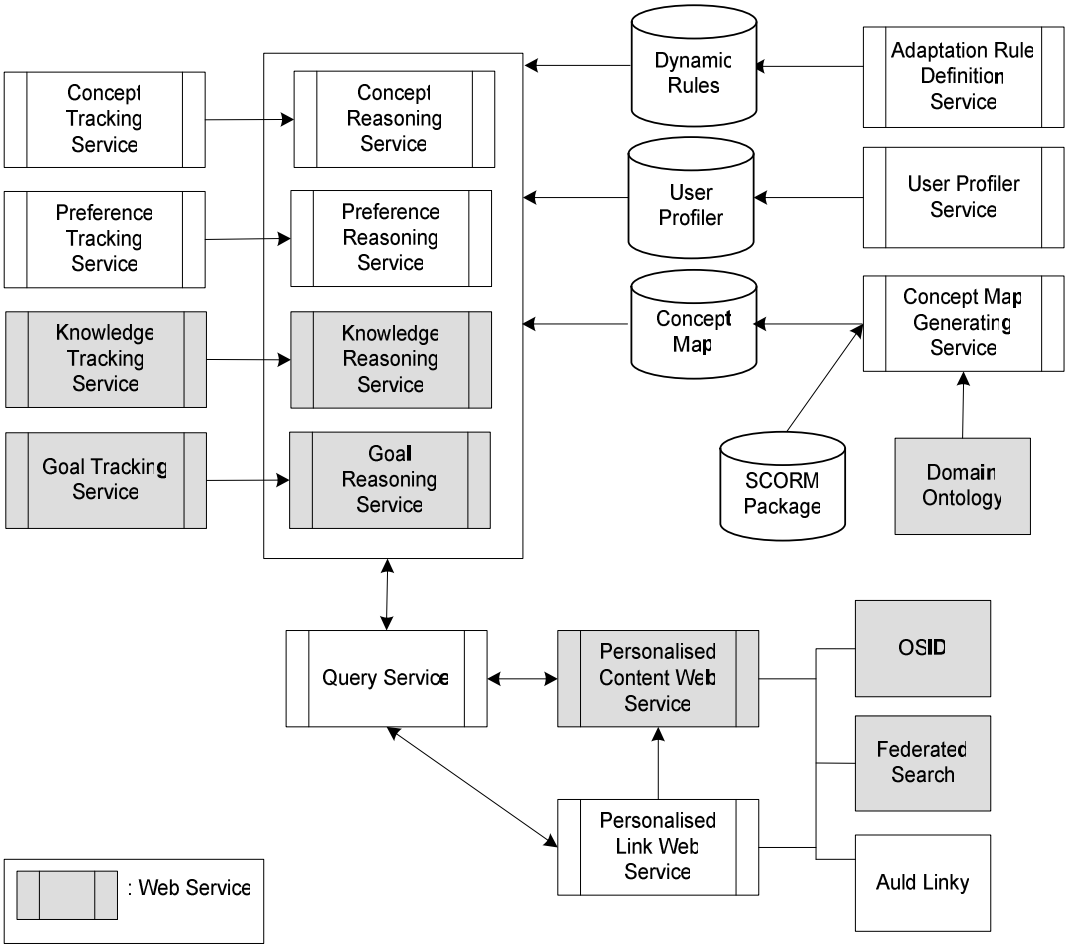

Figure 9. 1: The enhanced version of PLS architecture in which PLS Client is implemented as a collection of Web Services. The diagram also demonstrates that the architecture is extensible to add new functions. The shaded objects denote the possible new Services.

One of the advantages of applying Web Services-based Service Oriented Architecture is its design is loosely coupling in which the way a client of a Service communicates with the Service is essentially independent of the implementation of the Service itself (Ort, 2005). This feature makes the design more flexible as both client and Services can evolve or expand over time. In an example depicted in Figure 9.1, the Service Oriented Architecture design allows PLS to be scaleable in terms of functional components (Services).

The most selling point of the Web Services-based Service Oriented Architecture, however, is on its application of the Web Service technology which is platform, system, and language independence. Additionally, as the technology has being widely accepted and used by many developers today, interoperability can be accomplished among different systems and tools. The ability to interoperate among many systems can enable PLS to provide adaptive hypermedia authoring tools with appropriate links to potential learning resources to facilitate the development of an adaptive course. To conclude, the implementation of the PLS system as a collection of Web Services can ensure an architecture that is scalable and interoperable.

9.3.2.3 Implementing Adaptive Content Presentation Techniques via a Personalised Content Web Service

The extensibility feature of a Web Services-based Service Oriented Architecture makes it possible for the existing architecture of the PLS to include additional functions or services at later design stage. For instance, Figure 9.1 shows the future addition of a Personalised Content Web Service for the PLS system. The availability of this Service will enable PLS to adapt the presentation of the supplementary Web resources according to the current context of a student. As a result, the adaptive content presentation techniques such as adaptation modalities can be used to cater the requirement of students with learning disabilities as shown in Table 9.1.

9.3.3 Future Research Direction

It is anticipated that the future research directions in e-learning will revolve around Learning Object technology. Other research fields such as Natural Language Processing, Data Mining and the Semantic Web will also soon become the essential technologies to support future development of Learning Object based e-learning.

Natural Language Processing and Data Mining are two technologies that are capable of providing a solution to the process of identifying useful Learning Objects from the

existing repertoire of knowledge (e.g. the Web) or composing a Learning Object from existing pieces of information. The success of this automation would be desirable as it will utilise the presence of existing knowledge in the Web and lessen human efforts in the process. This is not dissimilar to Tim-Berners Lee's vision for the future of the World Wide Web (Berners-Lee et. al., 2001). The Semantic Web promises to provide the technologies to enable machines to better reason over Web content. As a result, the accuracy of the Web searches can be improved and potential source for Learning Objects can be discovered. Thereafter, by coupling the research outcomes from these research fields with the PLS approach, the identified Learning Object can be tagged with the necessary metadata to enable adaptation based on a user model. Above all, the thesis has provided the framework to practically apply these integrations to serve an adaptive open learning environment. The rest of this chapter will highlight the current research outcomes from these interesting research fields.

9.3.3.1 Natural Language Processing

Natural Language Processing (NLP) is an area of research that deals with the computer's attempts to process natural language (language normally used by humans), so that machines can derive semantic understanding of the content. NLP is indeed a mature research field. This is reflected in the development of many research and commercial based tools to enable machines to "understand" textual content. One of the most recent and widely talked about NLP toolkit is MontyLingua (Liu, 2004) developed at MIT (Massachusetts Institute of Technology).

MontyLingua is an end-to-end natural language understander for English, enriched with commonsense. MontyLingua provides a semantic interpretation of English text to facilitate information retrieval and extraction. It performs language-processing functions including commonsense-informed part-of-speech tagging, semantic recognition (which includes the process of recognising and extracting subject/verb/object tuples, adjectives, noun phrases and verb phrases, people's names, places, events, etc.), chunking, surface parsing, lemmatisation, and thematic-role

extraction (Liu and Singh, 2004). When an English text document is input into MontyLingua, the verb-subject-object object frames are extracted from the document and then further processed to serve various applications as described in (Liu and Singh, 2004).

Recent trends in NLP research have also demonstrated the interest in NLP techniques and related research to create tools to understand, organise and mine online materials, as reflected in the 2006 special track at the 19th International FLAIRS (The Florida Artificial Intelligent Research Society) conference. A system has even been developed to automatically build ontologies from a corpus of text using NLP techniques (Navigli et al., 2003).

The system named OntoLearn (Ontology Learning) (Navigli et al., 2003) first extracts terminology from a corpus of domain text using the Ariosto language processor (Basili et al., 1996). The extracted terminology is then filtered using NLP and statistical techniques. The filtering process is based on a comparative analysis across different domains, or contrastive corpora. The filtered terms will then be interpreted semantically by referencing to the WordNet and SemCor (Miller, 1990) lexical knowledge bases. Subsequently, WordNet and OntoLearn rule-based inductive-learning method is used to extract relations among the terms according to taxonomic (kind-of) and other semantic relations. As a result a so called 'domain concept forest' will be generated. This concept forest will later be integrated with WordNet to create a pruned and specialised view of the domain ontology. Finally, tools for ontology editing, validation, and management will be used to enrich, correct and update the generated ontology.

Outcomes from the aforementioned NLP research can definitely provide a fundamental ground for improving the query result of Web search engines. However, PLS system's requirement will still imposes a great challenge to NLP research in terms of interpreting the semantic content of a Web page according to an educational context.

Firstly, the PLS requires the deduction of a subject-specific learning concept from the identified Web page. Secondly, the PLS requires the inference of the instructional function of the Web page. Finally, the PLS requires a method to resolve the problem of determining which learning concepts and resource types (i.e. instructional function) should be represented by a Web page if the page contains multiple concepts and resource types. These questions will pose an interesting research agenda for initiating a serious collaboration between NLP and Adaptive Hypermedia research communities.

9.3.3.2 Data Mining

Data mining is the process of extracting patterns and predicting trends from large quantities of data (Fayyad, 1996). It has been largely used in on-line business such as market segmentation, fraud detection, direct marketing, interactive marketing, market basket analysis, and trend analysis (Tsai and Tsai, 2005). Its usefulness has become more prevalent with the tremendous growth of the Web. Furthermore, the emergence of so called 'internet dependence society' has created demands for more effective methods to locate useful Web pages. Now the concern is "*how can a search find high-quality Web pages on a specified topic*" (Han and Chang, 2002).

The integration of data mining techniques into the Web search engines can help facilitate finding high quality Web pages (Brin and Page, 1998) as "*data mining holds the key to uncovering and cataloguing the authoritative links, traversal patterns, and semantic structures of the Web*" (Han and Chang, 2002).

Authoritative links indicate the importance of a Web page because it denotes that the page is favourable (linked to) by many different authors on the Web. The concept of authoritative links can also be found in hubs. A hub is a single Web page or page set that provides collections of authoritative links (Han and Chang, 2002). In other words, a hub provides links to a collection of prominent sites on a common topic. Methods such as PageRank (Brin and Page, 1998) and HITS (Chakrabarti et al., 1999) are built for identifying authoritative Web pages and hubs. These methods are currently used by

Google, and as a result, Google can generate better-quality search results than term-index engines such as Alta Vista and topic directories such as Yahoo (Han and Chang, 2002).

Besides the use of authoritative links, semantic page structure and content recognition will greatly enhance the in-depth analysis of Web page contents (Han and Chang, 2002). A semiautomatic method to extract Web page structures and semantic contents based on the predefined Web page classes (e.g. document, organisation, and directory) can be described as follows (Han and Chang, 2002). The first step is to identify the relevant and interesting structures to extract, either an expert manually specifies this structure for a given Web page class, or the structure is deduced automatically from a set of pre-labelled Web page examples. Subsequently, Web page structure and content extraction methods (Borkar et al., 2001) can be used to analyse a Web page content and automatically extract segments that fits into these predefined structures (Han and Chang, 2002). As PLS aims to serve other applications with learning resources according to an educational context, it may require a creation of a new Web page class or extend the existing 'document' class. The new class should then contain the attributes that represent the required educational metadata such as subject name, learning concept, learning objective, resource types, content type and file type.

The aforementioned data mining techniques can be used to facilitate the automation process of identifying high quality Web content to serve an application such as the PLS. However, the existing data mining techniques have not yet taken into account the issue of content trust and reliability within an educational context which is important to ensure the success of the PLS. This emerging requirement would be the ground for collaborating research between adaptive hypermedia and data mining.

9.3.3.3 The Semantic Web

The Semantic Web enhances the existing Web with the ability to reason about its resources, so that "*machines become much better able to process and 'understand' the data that they merely display at present*" (Berners-Lee et. al., 2001). As a result, search engines would be able to gain an understanding about a query and respond, not only with a set of pages that refer to a precise concept, but also a set of services and other related information (Bailey, 2002).

Metadata and rules constitute the important ingredients of the Semantic Web. Metadata enables machines to locate a particular resource in the Web. On the other hand, rules make inferences about the content of those resources so that the semantic relationships among the resources can be established. The Semantic Web uses XML data format with Resource Discovery Framework (RDF) to uniquely describe each Web resource via their URIs (Unique Resource Identifiers). The interrelationships among the resources are described in terms of named properties and their values using RDF triples (*subject, predicate, object*) (Bray, 2001; Gibbins et al., 2003). A network of RDF triples will form webs of information about related resources (Berners-Lee et. al., 2001). As the Web constitutes billion of interlinked resources which are contributed by different people, there will be circumstances where different identifiers are used to represent the same concept. To address this issue, the Semantic Web uses the concept of ontologies.

According to (Berners-Lee et. al., 2001), an ontology usually comprises of a taxonomy and a set of inference rules. The taxonomy defines classes of objects and relations among them which are normally domain-specific. Whilst, inference rules interpret the outcome of the relations defined in the taxonomy. An ontology can be developed using RDF Schema (RDFS). RDFS defines classes, the taxonomic orderings, and the properties which relate instances of classes together (Gibbins et al., 2003). By exchanging ontologies, consumer and producer agents can share the same understanding about a common concept, and having a common ontologies can solve

problem associated to ambiguous terminology hence improve the accuracy of the Web searches (Berners-Lee et. al., 2001). These promises have made the Semantic Web headline the current research agenda of many institutions in the Web community.

As the Semantic Web adds logical structure to the Web, it makes it easier for machines to make sense of the Web, encouraging the development of more adaptive systems (Bailey, 2002). The PLS system can benefit from Semantic Web technology in mainly the following two areas. Firstly, the subject domain ontologies can provide a 'control' ground to the development of a concept map that is auto-generated from the SCORM package. In other words, once the concept map is generated, it can be edited, refined or updated according to the identified ontologies. Secondly, any relevant ontologies (e.g. ontology of subject domain or instructional objects) that are published in public domain can be used by the PLS Pre Run-Time Authoring Environment to obtain more accurate description of the Web resources and their relations in order to locate useful links. As a result, the relevancy of the links in the linkbase can be improved.

9.3.3.4 Conclusion

The development of the PLS has opened up many doors for integrating research outcomes from other research fields such as Natural Language Processing, Data Mining and Semantic Web.

The integration of these leading research fields will enable an automatic discovery of high quality pieces of information from the existing repertoire of knowledge either be it the Web or structured databases. Semantic Web technology promises that machines will make more sense of the Web content (or distributed resources) so that intelligent reasoning can be accomplished hence potential learning resources can be more accurately discovered. Once a huge collection of potential resources are discovered, data mining and NLP techniques can be used to identify and extract significant learning resources at both macro (e.g. page) and micro (e.g. paragraphs) levels. Then

by applying PLS approach, these resources can be labelled with metadata that will facilitate adaptation in an educational context.

As a result, useful links and contents can be discovered and developed with minimal human effort. Consequently, linkbases of highly significant learning materials can be constructed. As the PLS is using FOHM linkbases, it permits the storage of not only references to the materials but also the content's fragments. As a result, the PLS will not only generate adaptive links but is also capable of adapting the content of the supplementary materials to suit a student's learning requirement and the current learning concept delivered by the SCORM delivery system.

To conclude, the success of integrating related research outcomes from the aforementioned research fields will ensure the development of highly relevant linkbases hence increases the accuracy of the PLS as an adaptive support mechanism within an open learning environment. Such an enhanced PLS is hoped to serve Web-based SCORM-compliant learning environment and adaptive hypermedia authoring tools with good adaptive educational content.

Appendix 1

Learning Object Repositories	**Metadata Name**	**Vocabulary**
IEEE LOM	Resource Type	Diagram, Exam, Exercise, Experiment, Figure, Graph, Index, Narrative Text, Problem Statement, Questionnaire, Self Assessment, Simulation, Slide, Table
MERLOT		Animation, Collection, Drill and Practice, Lecture/Presentation, Quiz/Test, Reference Material, Simulation, Case Study, Tutorial
EdNa	DC.Type	EdNa document: Bibliography Dissertation Guidelines Index Manuscript Policy Document Presentation Reference Report Research Report Serial EdNa Curriculum: Activity Assessment Course Curriculum/Syllabus Exemplar Lesson Plan Online Project Training Package Unit/Module
EducaNext		Educational material type: Case Study, Case Study Guide, Collection, Data Set, Demonstration, Educator's Guide, Exam, Exercise, Experiment, Figure, Lecture Notes, Narrative Text, Presentation, Problem Statement, Questionnaire, Recorded Lecture, Reference Material, Research Paper, Research Study, Self

			Assessment, Simulation, Text Book, Thesis, Tutorial
		Educational activity type:	Case Study, Course, Course Unit, Exam, Exercise, Experiment, Group Work, Lecture, Presentation, Project
ARIADNE	Document Format	Expositive documents : Active documents :	Essays, Video Clips, All Kinds Of Graphical Material And Hypertext Documents Simulations, Questionnaires, and Exercises
iLumina[3]		Lesson, Presentation, Lab, Assessment, Teacher Tool, Syllabus, Example, Exercise, Course, Project, Learner Tool, Book, Demonstration, Simulation, Dataset, Lesson Plan, Manager Tool	
UK LOM Core[4]	Learning Resource Type	Exercise Simulation Questionnaire Diagram Figure Graph Index Slide Table Narrative Text Exam Experiment Problem Statement Self Assessment Lecture	

Table A. 1: Resource type metadata and its vocabularies across several learning object repositories

[3] iLumina (Educational Resources for Science and Mathematics) at http://www.ilumina-dlib.org/
[4] UK LOM Core at http://www.cetis.ac.uk/profiles/uklomcore

Appendix 2

```
<linkbase>
  <learningmodel id="honeymumford">
    <learningstyle type="activist">
      <resource type="notes" priority="none"/>
      <resource type="example" priority="low" />
      <resource type="homework" priority="high" />
      <resource type="experiment" priority="low" />
      <resource type="case" priority="high" />
      <changestyle type="reflector" />
    </learningstyle>
    <learningstyle type="reflector">
      <resource type="notes" priority="low"/>
      <resource type="example" priority="high" />
      <resource type="homework" priority="none" />
      <resource type="experiment" priority="none" />
      <resource type="case" priority="none" />
      <changestyle type="theorist" />
    </learningstyle>
    <learningstyle type="theorist">
      <resource type="notes" priority="high"/>
      <resource type="example" priority="none" />
      <resource type="homework" priority="none" />
      <resource type="experiment" priority="none" />
      <resource type="case" priority="low" />
      <changestyle type="pragmatist" />
    </learningstyle>
    <learningstyle type="pragmatist">
      <resource type="notes" priority="none"/>
      <resource type="example" priority="none" />
      <resource type="homework" priority="low" />
      <resource type="experiment" priority="low" />
      <resource type="case" priority="high" />
      <changestyle type="activist" />
    </learningstyle>
  </learningmodel>
</linkbase>
```

Figure A. 1: Dynamic rules (honeymumford.xml) that represent the association between the type of learning resource (based on its instructional function) and a specific learning style

Appendix 3

```
<?xml version="1.0" encoding="UTF-8"?>
<learnerinformation>
     <comment>PLS user model based on the IMS LIP</comment>
     <contentype>
         <referential>
            <sourcedid>
                <source>PLS_UM_IMS_LIP</source>
                <id>1001</id>
            </sourcedid>
         </referential>
     </contentype>


     <identification>
         <comment> user identification info </comment>
         <contentype>
            <referential>
                <indexid>user7</indexid>
            </referential>
         </contentype>
         <name>
         <typename>
            <tysource sourcetype="imsdefault"/>
            <tyvalue>Preferred</tyvalue>
         </typename>
            <comment>name details</comment>
            <contentype>
                <referential>
                   <indexid>aniza</indexid>
                </referential>
            </contentype>
            <text>Aniza Abdullah</text>

            <partname>
                <typename>
                   <tysource sourcetype="imsdefault"/>
                   <tyvalue>First</tyvalue>
                </typename>
                <text>Aniza</text>
            </partname>
            <partname>
                <typename>
                   <tysource sourcetype="imsdefault"/>
                   <tyvalue>Last</tyvalue>
                </typename>
                <text>Abdullah</text>
            </partname>
         </name>
     </identification>

     <securitykey>
```

```
            <typename>
               <tysource sourcetype="imsdefault"/>
               <tyvalue>Password</tyvalue>
            </typename>
            <contentype>
               <referential>
                   <indexid>Security_1</indexid>
               </referential>
            </contentype>
            <keyfields>
               <fieldlabel>
                   <typename>
                      <tyvalue>PersonalPassword</tyvalue>
                   </typename>
               </fieldlabel>
               <fielddata>aniza</fielddata>
            </keyfields>
            <keyfields>
               <fieldlabel>
                   <typename>
                      <tyvalue>LMSPassword</tyvalue>
                   </typename>
               </fieldlabel>
               <fielddata>LMS_Soton</fielddata>
            </keyfields>
         </securitykey>


         <accessibility>
            <contentype>
               <referential>
                   <indexid>accessibility_01</indexid>
               </referential>
            </contentype>
            <language>
            <typename>
               <tysource sourcetype="imsdefault"/>
               <tyvalue>English</tyvalue>
            </typename>
            <comment>Language</comment>
                <contentype>
                   <referential>
                      <indexid>language_01</indexid>
                   </referential>
                </contentype>
            </language>
            <preference>
            <typename>
                <tysource
sourcetype="proprietary">http://localhost:8080/adlextra/honeymumford.x
ml</tysource>
               <tyvalue>reflector</tyvalue>
            </typename>
            <comment>learning style preferences</comment>
                  <contentype>
                     <referential>
                        <indexid>learningstyle_02</indexid>
                     </referential>
                  </contentype>
```

```
            <description>Honey and Mumford Learning Style Model is
employed here</description>
         </preference>
       </accessibility>
</learnerinformation>
```

Figure A. 2: A sample of the PLS's user model implemented using IMS LIP

References

Abdullah, N.A. and Davis, H.C. (2003). Is Simple Sequencing Simple Adaptive Hypermedia?. In *Proceedings of the 14th ACM Conference on Hypertext and Hypermedia*. Nottingham. August 26-30, 2003. pp. 172-173.

Abdullah, N.A., Bailey, C.P., and Davis, H.C. (2004). Augmenting SCORM Manifest with Adaptive Links. In *Proceedings of the 15th ACM Conference on Hypertext and Hypermedia*. Santa Cruz. 2004. pp. 183-184.

Abdullah, N.A. and Davis, H.C. (2005). A Real-time Personalization Service for SCORM. In *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies (ICALT 2005) (in press)*. Kaohsiung, Taiwan. July 5-8, 2005. pp. 63-64.

ADL (2004). Advanced Distributed Learning, Retrieved February 20, 2006 at http://www.adlnet.org

ADL Technical Team (2004). SCORM Photoshop Examples Version 1.0. Retrieved February 20, 2006 at http://www.adlnet.gov/downloads/index.cfm

AICC (2000). CMI Guidelines for Interoperability Revision 3.0.2. Retrieved February 20, 2006 at http://www.aicc.org/docs/tech/ cmi001v302.pdf

Anderson, K. M. (1997). Integrating Open Hypermedia Systems with the World Wide Web. In *the Proceedings of ACM Hypertext '97*, Southampton UK. 1997. pp. 157-167

Anderson, K. M., Taylor, R. N., and Whitehead, E. J. (1997) A Critique of the Open Hypermedia Protocol. *Journal of Digital Information (JoDI). Special Issue on Open Hypermedia Systems 1, 2* (1997).

Anderson, K. M., Taylor, R. N., Whitehead, Jr., E. J. (1994). Chimera: Hypertext for Heterogeneous Software Environment, In the Proceedings of the 1994 ACM European Conference on Hypermedia Technology, Edinburgh, Scotland. September 19-23, 1994. pp. 94-107.

Anido-Rifon, Luis E., Santos-Gago, Juan, M., Rodriguez-Estevez, Judith, S., Caeiro-Rodriguez, Manuel, Fernandez-Iglesias, Manuel, J., Llamas-Nistal, Martin (2001). A Step Ahead In E-Learning Standardization: Building Learning Systems from Reusable and Interoperable Software Components. In *Proceedings of the International World Wide Web Conference (11)*. Honolulu, Hawaii. January 2001, Retrieved April 3, 2006 at http://wwwconf.ecs.soton.ac.uk /archive/00000180/

Arapi, P., Moumoutzis, N., and Christodoulakis, S. (2003). Supporting Interoperability in an Existing e-Learning Platform Using SCORM, In Proceedings of the 3rd IEEE International Conference on Advanced Learning Technologies (ICALT2003), July 2003, Athens, Greece, Retrieved April 3, 2006 at http://www.ced.tuc.gr/Staff/ ArapiPolyxeni.htm

ARIADNE (2004). Alliance of Remote Instructional Authoring and Distribution Networks for Europe Retrieved February 20, 2006 at http://www.ariadne-eu.org/

Axis Guide (2005) Axis User Guide. Retrieve 2 Mac 2006 at http://ws.apache.org/axis/ java/user-guide.html

Bailey, C. (2002). An Agent-based Framework to Support Adaptive Hypermedia. *PhD Thesis*, University of Southampton. October 2002.

Bailey, C., Hall, W., Millard, D. E. and Weal, M. J. (2002). Towards Open Adaptive Hypermedia. In *De Bra, P., Brusilovsky, P. and Conejo, R., Eds. Proceedings of 2nd International Conference, Adaptive Hypermedia and Adaptive Web-Based Systems 2347 (-).* Málaga, Spain. 2002. pp.36-46.

Bailey, C., Hall, W., Millard, D.E., and Weal, M.J. (2003). Adaptive Hypermedia through Contextual Structures. *ACM Transactions on Information Systems (awaiting publication)*. Retrieved February 20, 2006 at http://eprints.ecs.soton.ac.uk/9045/

Bajraktarevic, N. (2003). Adaptive Hypermedia, Learning Styles and Strategies within the Educational Paradigm. *PhD Thesis*, University of Southampton. September 2003.

Baldoni, M., Baroglio, C., Patti, V., Torasso, L. (2004). Reasoning About Learning Object Metadata for Adapting SCORM Courseware. *In Proceedings of International Workshop on Engineering the Adaptive Web: Methods and Technologies for Personalization and Adaptation in the Semantic Web, (AH2004)*. Eindhoven. August 24-26, 2004. Retrieved April 3, 2006 at http://rewerse.net/publications/download/REWERSE-RP-2004-35.pdf

Basili, R., Pazienza, M.T., and Velardi, P. (1996). An Empirical Symbolic Approach to Natural Language Processing, *Artificial Intelligence*, *85(1-2)*, August, 1996, pp.59-99.

Berners-Lee, T., Cailliau, R. (1990) WorldWideWeb: Proposal for a HyperText Project. *CERN*, Geneva, 1990. Retrieved May 3, 2006 at http://www.w3.org/Proposal.html

Berners-Lee, T., Hendler, J., Lassila, O. (2001). The Semantic Web: A New Form Of Web Content That Is Meaningful To Computers Will Unleash A Revolution Of New Possibilities. *Scientific American.com*. May 17, 2001. Retrieved April 3, 2006 at http://www.scientificamerican.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21&catID=2

Bernd, S. (2002). Survey on Vocabulary Usage of Learning Resource Type. 2002. Retrieved April 3, 2006, from http://nm.wu-wien.ac.at/e-learning/lr-types.htm

Blackmon, W., Brooks, J., Roberts, E., and Rehak, D. (2004). The Overlap and Barriers between SCORM, IMS Simple Sequencing, and Adaptive Sequencing. In *Proceedings of Adaptive Hypermedia Conference*. 2004. Retrieved February 20, 2006 at http://www.lsal.cmu.edu/lsal/expertise/papers/conference/ah2004/wit20040329.pdf

Bouras, C., Nani, M., and Tsiatsos, T. (2003). A SCORM-Conformant LMS. In *Proceeding of ED–MEDIA Conference*. Honolulu, Hawaii, USA. June 23 – 28, 2003. pp. 10–13, at http://ru6.cti.gr Publications/910.pdf

Borkar, V., R., Deshmukh, K., and Sarawagi, S. (2001). Automatic Segmentation of Text into Structured Records, In *Proceedings of ACM-SIGMOD International Conference Mangement of Data, ACM Press*, New York, 2001, pp. 175-186.

Brailsford, T.J., Stewart, C.D., Zakaria, M.R. and Moore, A. (2002). Autonavigational, Links and Narrative in An Adaptive Web-Based Integrated Learning Environment. In *Proceedings of the 11th International World Wide Web Conference (WWW2002)*. Honolulu, Hawaii, USA. May 7-11, 2002. Retrieved April 3, 2006 at http://whurle.sourceforge.net/

Bray, T., (2001). What is RDF?. Retrieved February 20, 2006 at http://www.xml.xom/pub/a/2001/01/24/rdf.html?page=3

Brin, S. and Page, L. (1998) The Anatomy of a Large-Scale Hypertextual Web Search Engine, In *Proceedings of the 7th International World Wide Web Conference (WWW7), ACM Press*, New York, 1998. pp. 107-117.

Brown, E., Cristea, A., Stewart, C., and Brailsford, T. (2005). Patterns in Authoring of Adaptive Educational Hypermedia: A Taxonomy of Learning Styles. *Educational Technology & Society, 8(3).* 2005. pp. 77-90.

Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. *User Modeling and User Adapted Interaction 6, 2-3*, 1996, pp. 87-129.

Brusilovsky, P. (1999). Adaptive and Intelligent Technologies for Web-based Education. In *C. Rollinger and C. Peylo* (*Eds.*), *Special Issue on Intelligent Systems and Teleteaching*, *Künstliche Intelligenz*, *4*. 1999. pp. 19-25.

Brusilovsky, P. (2001). Adaptive Hypermedia. In *Alfred Kobsa* (*Ed.*), *User Modeling and User-Adapted Interaction*, *Ten Year Anniversary*, *11.* 2001. pp. 87-129.

Brusilovsky, P. (2003). Developing Adaptive Educational Hypermedia Systems: From Design Models to Authoring Tools. In *T. Murray, S. Blessing and S. Ainsworth* (*Eds.*): *Authoring Tools for Advanced Technology Learning Environmen*, *Dordrecht*: *Kluwer Academic Publishers.* 2003.

Bush, V. (1945). As We May Think. *The Atlantic Monthly*, *176*(*1*). 1945. pp. 101-108.

Campbell, L. (2002). Introduction to Learning Technology Interoperability Standards and CETIS. *Centre for Educational Technology Interoperability Standards*, *SURF*

Education Days, The Hague, November 2002, Presentation Slide. Retrieved April 3, 2006 at http://www.cetis.ac.uk/groups/20010926111402/FR20030829111244

Clark, D. (2000). Learning Styles: How Can We Go From the Unknown to the Unknown. Retrieved February 20, 2006 at http://www.nwlink.com/ ~donclark/hrd/styles.html

Clark, R., and Rossett, A (2002). Learning Solutions – Learning Objects: Behind The Buzz. 2002. Retrieved April 3, 2006 at http://www.clomedia.com/content/ templates/clo_feature.asp?articleid=24&zoneid=30

Carr, L., A., De Roure, D., Hall, W., and Hill, G., (1998). Implementing an Open Service for the World-Wide Web, *World Wide Web Journal*, *1*(2), 1998, pp. 61-71

Carr, L.A., De Roure, D., Hall, W. and Hill, G. (1995). The Distributed Link Service: A Tool for Publishers, Authors and Readers. *World Wide Web Journal*, *1*(*1*). 1995. pp.647-656.

Carro, R. M. (2002). Adaptive Hypermedia in Education: New Considerations and Trends, In *Proceedings of the 6th World Multi-conference on Systemics, Cybernetics and Informatics*, Orlando, Florida, 2002, pp. 452-458.

Carver, C.A., Howard, R.A., Lane, W.D. (1999). Enhancing Student Learning through Hypermedia Couresware and Incorporation of Student Learning Styles, *IEEE Transactions on Education*, *42*(*1*). 1999. pp 33-38.

Chakrabarti, S., Dom, B., E., Kumar, S., R., Raghava, P., Rajagopalan, S., Tomkins, S., Gibson, D., Kleinberg, J. (1999) Mining the Web's Link Structure. *IEEE Computer*, August 1999, pp. 60-97.

Coffield, F.J., Morseley, D.V., Hall, E. and Ecclestone, K. (2004). Learning Styles and Pedagogy in Post-16 Learning: A Systematic and Critical Review. *London: Learning and Skills Research Centre/University of Newcastle Upon Tyne*. 2004.

Conlan, O., Dagger, D., and Wade, V. (2002). Towards a Standards-based Approach to e-Learning Personalization using Reusable Learning Objects. In *Proceedings of World Conference on E-learning in Cooperate, Government, Healthcare and higher Education* (*E-learn 02*). Montreal, September 2002. Retrieved April 9, 2006 at https://www.cs.tcd.ie/Owen.Conlan/ publications/eLearn2002_v1.24_Conlan.pdf

Conklin, J. (1987). Hypertext: An Introduction and Survey, *IEEE Computer, 20(9)*. 1987. pp. 17-41.

Cravern, T. C., (2001) DESCRIPTION Meta Tags in Public Home and Linked Pages. *LIBRES: Library and Information Science Research, 11(2)*. Retrieved April 3, 2006 at http://libres.curtin.edu.au/LIBRE11N2/index.htm

Cristea, A. I and de Mooij, A. (2003) Adaptive Course Authoring: My Online Teacher, In *the Processing of the 10th IEEE International Conference on Telecommunications, ICT'03*. Papeete, French Polynesia, February 23–March 1, 2003, pp. 1762-1769

Cristea, A.I. (2003). Adaptive Patterns in Authoring of Educational Adaptive Hypermedia. *Educational Technology & Society*, *6*(*4*), 2003. pp. 1-5. Retrieved February 20, 2006 at http://ifets.ieee.org/periodical/6_4/1.pdf

Cristea, A.I., Floes, D., Stash, N., De Bra, P. (2003). MOT meets AHA!. In *Proceedings of the PEG Conference*. Saint-Petersburg, Russia. 28 June – 1 July, 2003. Retrieved 9 April, 2004 at http://wwwis.win.tue.nl/~acristea/HTML/Minerva/papers/PEG03-cristea-floes-stash-debra-final.rtf

cogdogblog (2005). If All The Learning Objects Are Web Pages Who Needs a Repository? February 22, 2005 at http://jade.mcli.dist.maricopa.edu/alan/ archives/2005/02/22/ objects.php

Dagger, D., Wade, V. and Conlan, O. (2003). Towards "anytime, anywhere" Learning: The Role and Realization of Dynamic Terminal Personalization in Adaptive e-Learning. In *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications* (*ED-MEDIA 2003*). Hawaii. 2003. Retrieved 9 April, 2004 at https://www.cs.tcd.ie/Owen.Conlan/ publications/ EdMedia2003v1_Conlan.pdf

Davis, H.C., Hall, W., Heath, I., Hill, G., and Wilkins, R.J. (1992). Towards and Integrated Information Environment with Open Hypermedia Systems. In *Proceedings of the ACM European Conference on Hypermedia Technology* (*ECHT'92*). *ACM Press*. 1992. pp. 181-190.

Davis H.C., Rizk, A., and Lewis, A.J. (1996). OHP: A Draft Proposal for a Standard Open Hypermedia Protocol. *U.K.D.Wiil, S.* (*Eds.*), *Proceedings of the Second Workshop on Open Hypermedia Systems*, *ACM Hypertext '96*. March, 1996. pp.27-53,

DeBra, P. (2002). Adaptive Educational Hypermedia on the Web. *Communications of the ACM*, *45*(*5*), 2002. pp. 60-61.

DeBra, P. and Calvi, L. (1998). AHA! An Open Adaptive Hypermedia Architecture. *The New Review of Hypermedia and Multimedia*, *4*, 1998. pp. 115-139, Taylor Graham Publishers

DeBra, P., Aerts, A. and Roussean, B. (2002a). Concept Relationship Types for AHA! 2.0. In *Proceedings of the AACE ELearn 2002 Conference*. 2002. pp 1386-1389.

DeBra, P., Aerts, A., Smits, D., and Stash, N. (2002b). AHA! Version 2.0: More Adaptation Flexibility for Authors, In *Proceedings of the AACE ELearn'2002 Conference*, October, 2002, pp. 240-246.

DigitalThink (2003). SCORM: The E-Learning Standard. DigitalThink, Inc. 2003. Retrieved May 11, 2006 at http://c-beta.digitalthink.com/dtfs/downloads/products_services/wp_standards.pdf

Dolog, P., Gavriloaie, R., Nejdl, W. and Brase, J. (2003). Integrating Adaptive Hypermedia Techniques and Open RDF-based Environments. In *Proceedings of the 12th International World Wide Web Conference*, (*WWW2003*), Budapest, Hungary. 2003. Retrieved 17 January, 2006. at http://citeseer.ist.psu.edu/dolog03integrating.html

Duval, E. (2004). Advanced Applications of Reusable Learning Objects and the RLO Standards in Web-based Education. *Invited Presentation at the 3rd IASTED International Conference on Web-based Education*. Innsbruck, Austria. February 16-18, 2004.

Duval, E., and Hodgins, M. (2003). A LOM Research Agenda, In *Proceedings of the World Wide Web Conference 2003*. May 20-24, 2003, Budapest, Hungary. Retrieved 3 April 2006 at http://www2003.org/cdrom/papers/alternate/P659/p659-duval.html.html

EducaNext (2001) The EducaNext Portal for Learning Resources. Retrieved 20 April 2006 at http://www.educanext.org/ubp

Eftekhar, N. and Strong, D.R. (1998). Towards Dynamic Modelling of a Teaching/Learning System Part 2: A New Theory on Types of Learners.

International Journal of Engineering Education, *14*(*6*), 1998. pp. 388-406. Tempus Publication.

Fayyad, U., Piatestku-Shapio, G., Smyth, P., Uthurusamy, R. (1996). Advances in Knowledge Discovery and Data Mining, *AAAI/MIT Press*, Menlo Park, CA, 1996.

EdTechPost (2004). Finding Learning Objects: Walking the Talk, *in the EdTechPost, Technologies for Learning, Thinking & Collaborating*, May 06, 2004, Retrieved 3 May, 2006 at http://www.edtechpost.ca/mt/archive/000539.html

Felder R.M. and Silvermann, L.K. (1988). Learning Styles and Teaching Styles in Engineering Education. *Engineering Education*, *78*(*7*), 1998. pp. 674-681.

Felder, M.R. (1996). Matters of Style. *ASEE Prism. 6(4)*, 1996. pp. 18-23. Retrieved February 20, 2006 at http://www.ncsu.edu/felder-public/Papers/LS-Prism.htm

Fischer, G. (2001). User Modeling in Human-Computer Interaction, *Journal of User and User-Adapted Interaction*, *11*, 2001. pp. 65-86.

Gibbins, N., Harris, S., Michaelides, D., T., Millard, D., E., Weal, M., J., (2003). Exploring the Relationship between FOHM and RDF. In *Proceedings of the 1st International Workshop on Hypermedia and the Semantic Web, HT'03*. Nottingham, UK, 2003.

Google API (2002). Google Search API, Beta Version. Retrieved February 20, 2006 at http://www.google.com/apis/ reference.html#2_2

Goldberg, D., Nichols, D., Oki, B. and Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. Communication of the ACM 35, 1992. pp. 61-70.

Grønbæk, K. and Trigg, R.H. Design issues for a Dexter-based hypermedia system. *Communications of the ACM 37*, *2*, February 1994, pp. 40-49.

Hall, W., Davis, H.C., and Hutchings, G. (1996). Rethinking Hypermedia the Microcosm Approach. *Kluwer Academic Publishers*. Dordrecht, Netherland. 1996.

Han, J. and Chang, K.C-C (2002). Data Mining for Web Intelligence, *IEEE Computer, 35(11*), November 2002, pp. 64-70.

Hatala, M., Richards, G., Eap. T., Willms, J. (2004). The Interoperability Of Learning Object Repositories And Services: Standards, Implementations and Lessons Learned. In *Proceedings of the World Wide Web Conference 2004*. May 17-22, 2004.

New York, USA. Retrieved April, 9, 2006 at http://www2004.org/proceedings/docs/2p19.pdf

Hartmann, Virginia F. (1995). Teaching and Learning Style Preferences: Transitions through Technology. *VCCA Journal 9(2)*. 1995. pp. 18-20

Heins, T. and Himes, F (2002). Creating Learning Objects With Macromedia, Flash MX. *Macromedia White Paper*, April 2002. Retrieved 10 May 2006 at http://download.macromedia.com/pub/solutions/downloads/elearning/flash_mxlo.pdf

Heng, S. (2003). Learning Objects – Where Next? In *the Proceedings of the 3rd Annual CM316 Conference on Multimedia Systems*. 2003. Retrieved January 8, 2006 at http://citeseer.ist.psu.edu/556932.html

Hodgins, H. W. (2000). The Future of Learning Objects. In *D. A. Wiley (Ed.), The Instructional Use of Learning Objects* (Chapter 5.3, pp. 1-24). 2000. Retrieved 1 January, 2006 at http://www.reusability.org/read/chapters/hodgins.doc

Honey, P. and Mumford, A. (1992). The Manual of Learning Styles. Maidenhead, Peter Honey. 1992.

IEEE LOM (2002). 1484.12.1 IEEE Standard for Learning Object Metadata. 2002. Retrieved February 20, 2006 at http://ltsc.ieee.org/wg12

IMS (2003). IMS Global Learning Consortium: Overview of Specifications. Retrieved February 20, 2006 from

IMS CP (2003). IMS Content Packaging Information Model Version 1.1.3 Final Specification 2003, Retrieved 20 May 2004 at http://www.imsglobal.org/content/packaging/

IMS LIP (2000). IMS Learner Information Packaging Best Practice and Implementation Guide v1.0. Retrieved February 20, 2006 at http://www.imsglobal.org/profiles/lipbind01.html

IMS SS (2002a). IMS Simple Sequencing Information and Behavior Model, Retrieved 28 May 2003 at http://www.imsproject.org/simplesequencing/v1p0pd/imsss_infov1p0pd.html.

IMS SS (2002b). IMS Simple Sequencing Best Practice and Implementation Guide. Retrieved February 20, 2006 at http://www.imsproject.org/simplesequencing /v1p0pd/imsss_bestv1p0pd.html

Jonassen, D. H. (1993). Effects of Semantically Structured Hypertext Knowledge Bases on Students' Knowledge Structures. In *C. McKnight, A. Dillon and J. Richardson* (*Eds*.) *Hypertext. A Psychological Perspective*. Chichester: Ellis Horwood. 1993.

Karagiannidis, C., and Sampson, D., (2004). Adaptation Rules Relating Learning Styles Research and Learning Objects Meta-Data. *Workshop on Individual Differences in Adaptive Hypermedia, the 3rd International Conference on Adaptive Hypermedia and Adaptive Web-based Systems* (*AH2004*). Eindhoven, Netherlands. August 2004, *1*, pp. 136-145

Kavcic, A. (2000). The Role of User Models in Adaptive Hypermedia Systems. In *Proceedings of the Electrotechnical Conference, MELECON 2000, 10th Mediterranean, 1*, 2000. pp. 119-122.

Kinshuk, and Lin, T. (2004). Application of Learning Styles Adaptivity in Mobile Learning Environments, In *Proceedings of the 3rd Pan-Commonwealth Forum on Open Learning*. Dunedin, New Zealand. July 2004.

Kolb, D.A. (1984). Experiential Learning: Experience as the Source of Learning and Development. *Englewood Cliffs*, NJ, Prentice-Hall. 1984.

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R. and Riedl, J. (1997). Grouplens: Applying Collaborative Filtering to Usenet News. *Communication of the ACM 40(3)*, 1997. pp. 77-87.

Lin, M., Chau, M., Nunamaker Jr, J.F. and Chen, H. (2004). Segmentation of Lecture Videos Based on Text: A Method Combining Multiple Linguistic Features. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences*, (*HICSS'04*), January 5-8, 2004. Track 1, pp. 10003C. Retrieved April 3, 2006 at http://csdl2.computer.org/ comp/proceedings/hicss/2004/2056/01/205610003c.pdf

Liu, H (2004). MontyLingua: An End-to-end Natural Language Processor with Common Sense. 2004. Retrieved April 3, 2006 at http://web.media.mit.edu/~hugo/ montylingua/

Liu, H. and Singh, P. (2004). ConceptNet: A Practical Commonsense Reasoning Toolkit. *BT Technology Journal*. Kluwer Academic Publishers. 2004.

Longmire, W. (2000). A Primer on Learning Objects. *Learning Circuits Online*. 2000. Retrieved April 3, 2006 from

Lucene (2001). The Apache Software Foundation: Apache Lucene. 2001. Retrieved 19 October 2005 at http://lucene.apache.org/java/docs/index.html

MERLOT (2005). Multimedia Educational Resource for Learning and Online Training. Retrieved February 20, 2006 at http://www.merlot.org

Marshall C. C., and Shipman, F. M. (1995). Spatial Hypertext: Designing for Change. *Comunications of the ACM, 38* (1995), pp. 88-97.

Michaelides, D.T., Millard, D.E., Weal, M.J. and De Roure, D.C. (2001). Auld Leaky: A Contextual Open Hypermedia Link Server. In *the Proceedings of the 7th Workshop on Open Hypermedia Systems, ACM Hypertext 2001 Conference*. Aarhus, Denmark. Retrieved April 3, 2006 at http://eprints.ecs.soton.ac.uk/6141/

Millar, G. (2002). Learning Objects 101: A Primer for Neophytes, *Sidebars*, *a Publication of the Learning Resources Unit of the British Columbia Institute of Technology*. 2002. Retrieved May 8, 2003 at http://Online.Bcit.Ca/sidebars 02november/inside-out-1.htm

Millard, D. E. (2000). Navigating the Information Continuum. *PhD Thesis*, University of Southampton, December 2000.

Millard, D. E., Davis, H. C., Weal, M., Aben, K., and DeBra, P. (2003). AHA ! meets Auld Linky : Integrating Designed and Free-form Hypertext Systems. In *Proceedings of the 14th ACM Conference on Hypertext and Hypermedia*. Nottingham. August 26-30, 2003. pp. 161-169.

Millard, D.E., Moreau, L., Davis, H.C. and Reich, S. (2000). FOHM: A Fundamental Open Hypertext Model for Investigating Interoperability between Hypertext Domains. In *Proceedings of the 11th on the ACM Conference on Hypertext and Hypermedia*, 2000. pp. 93-102.

Miller, A., (1990). WordNet:An On-line Lexical Resource. *Journal of Lexicography, 3*(4). December 1990.

Mödritscher, F., García Barrios, V.M., Gütl, Ch., (2004). Enhancement of SCORM to Support Adaptive E-Learning within the Scope of the Research Project AdeLE, In *Proceedings of ICL*, Villach, Austria, September 30, 2004, Retrieved April 3, 2006 at http://www2.iicm.edu/cguetl/papers/adaptiveelarningstandards/adaptivestandard.pdf

Mohan, P., and Greer, J. (2003). Using Learning Object Technology to Tackle Education Challenges of the Carribean, In *the Workshop of the Proceedings for the International Conference in Information Technology: Research and Education*, August 11-13, 2003, Newark, New Jersey, USA.

Montaner, M., Lopez, B. and Dela, J. L. (2003). A Taxonomy Recommender agents on the Internet. *Artificial Intelligence Review, 19*. 2003. pp 285-330.

Murray, T. (2002). MetaLinks: Authoring and Affordances for Conceptual and Narrative Flow in Adaptive Hyperbooks, In *International Journal of Artificial Intelligence in Education.* 2002. Retrieved February 20, 2006 at http://helios.hampshire.edu/~tjmCCS/papers/ MLIJAIED2001. subm1.doc

Murray, T., Condit, C., and Haugsjaa, E. (1998). MetaLinks: A Preliminary Framework for Concept-Based Adaptive Hypermedia. *Workshop Proceedings for WWW-Based Tutoring in ITS-98 Conference*. San Antonio, August, 1998. Retrieved February 20, 2006 at http://helios.hampshire.edu/~tjmCCS/papers/MetaLinksITS98Wkshp/MetaLinksITS98.html

Navigli, R., Velardi, P., and Gangemi, A. (2003). Ontology Learning and Its Application to Automated Terminology Translation. *IEEE Intelligent Systems. Computer Society. 18(1),* Published by IEEE. January/February 2003, pp. 22-31

Nelson T.H. (1965). A File Structure for the Complex, the Changing, and the Indeterminate. In *Proceedings of the 20th ACM National Conference*. August, 1965. pp. 84-100.

Ng, M.H. (2003). Integrating Adaptivity into Web-based Learning. *PhD Thesis*, University of Southampton, March 2003.

Ort, E. (2005) Service-Oriented Architecture and Web Services: Concepts, Technologies, and Tools , Sun Developer Network, April 2005. Retrieved 2 May 2006 at

http://java.sun.com/developer/technicalArticles/WebServices/soa2/SOATerms.html

Østerbye, K. and Wiil, U.K. (1996) The Flag Taxonomy of Open Hypermedia Systems. In *Proceedings of the the Seventh ACM Conference on Hypertext* (Bethesda, Maryland, United States, March 16 - 20, 1996. *HYPERTEXT '96*. ACM Press, New York, NY, pp. 129-139

Paredes, P., and Rodriguez, P. (2004). A Mixed Approach to Modelling Learning Styles in Adaptive Educational Hypermedia, *International Workshop: Authoring of Adaptive and Adaptable Educational Hypermedia, In Proceeding of the WBE 2004*. 19 February, 2004, Innsbruck, Austria. pp. 372-378.

PAPI (2004). IEEE Private and Personals Information. Retrieved May 1, 2006 at http://www.edutool.com/papi/

Pazzani, M., Muramatsu, J. and Billsus, D. (1996). Syskill and Webert: Identifying Interesting Web Sites. In *Proceeding of the 13th National Conference on Artificial Intelligence*. 1996. pp. 54-61.

Perez, Gutierrez, Lopistequy, and Uzandizaga (1995). HyperTutor: From Hypermedia to Intelligent Adaptive Hypermedia'. In *Proceedings of the World Conference on Educational Multimedia* and Hypermedia, (*ED-MEDIA'95*). Graz, Austria, 1995. pp. 529-534.

Power, G., Davis, H.C., Cristea, A.I., Stewart, C. and Ashman, H. (2005). Goal Oriented Personalization with SCORM. In *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies* (*ICALT 2005*) (*in press*), Kaohsiung, Taiwan, July 5-8 2005, pp. 467-471.

Qu, C., and Nejdl, W. (2002). Towards Interoperability and Reusability of Learning Resource: A SCORM-conformant Courseware for Computer Science Education. In *Proceeding of 2nd IEEE International Conference on Adavanced Learning Technologies* (*ICALT, 2002*), IEEE Computer Society Press. Kazan, Tatarstan, Russia. Retrieved April 9, 2006 at http://www.kbs.uni-hannover.de/Arbeiten/Publikationen/2002/icalt24.pdf

Ramirez, P.M. and Mattman, C.A. (2004). Improving Search Engines Using Automatic Concept Extraction. In *Proceeding of the 2004 IEEE Conference on Information Reuse and Integration*, (*IRI–2004*), Publisher IEEE Systems, Man, and Cybernetic Society. 2004. pp. 229 – 234.

Rehak, D., R. and Mason R. (2003). Keeping the Learning in Learning Objects, *in Reusing Online Resources, A sustainable Approach to eLearning*, *A. Littlejohn (Ed).* Kogan Page, London, 2003.

Resnick, P. and Varian, H., R. (1997) Recommender Systems. *Communications of the ACM, 40(3),* March 1997.

Sampson, D., Karagiannidis, C. and Cardinali, F. (2002). An Architecture for Web-based e-Learning Promoting Re-usable Adaptive Educational e-Content. Invited Paper In *Educational Technology and Society Journal of IEEE Learning Technology Task Force*, *Special Issues on Innovations in Learning Technologies*, 5(4). 2002. pp 27-37.

SCORM (2002). SCORM Version 1.3 Application Profile, Working Draft 0.9, 27 November 2002. Retrieved February 20, 2006 from http://xml.coverpages.org/ SCORMV13-SeqAppProfile.pdf

SCORM RTE (2004). Advanced Distributed Learning, SCORM Runtime Environment Version 1.3.1. Retrieved February 20, 2006 at http://www.adlnet.org/downloads/ 197.cfm

Shardanand, U. and Maes, P. (1995). Social Information Filtering: Algorithm for Automating "word of mounth". In Conference on Human Factors in Computing Systems. *ACM Press*. 1995. pp. 210-217.

Sinclair, P., Martinez, K., Millard, D., E., and Weal, M., J. (2002). Links in the Palm of Your Hand: Tangible Hypermedia Using Augmented Reality. In *the Proceedings of the 13th ACM Conference on Hypertext and Hypermedia*, College Park, Maryland, USA. 2002. pp. 127-136.

Sleeman, D.H. and Brown, J.S. (1982). Introduction to Intelligent Tutoring Systems: An Overview. In *Intelligent Tutoring Systems*, *D.H. Sleeman and J.S. Brown* (*Eds*.), *Academic Press*. 1982. pp. 1-11.

Slosser, S. (2002). ADL and Sharable Content Object Reference Model. *Presentation Slide, Joint ADL Co-Laboratory*, Orlando, Florida. 2002.

Smith, J. B. and Weiss (1988). Hypertext *Communications of ACM*, *31*(*7*). July 1998

SOAP Tomcat (2002). Using SOAP with Tomcat. 2002. Retrieved 3 January 2006 at http://www.onjava.com/pub/a/onjava/ 2002/02/27/tomcat.html?page=2

Stash, N. and De Bra, P. (2003). Building Adaptive Presentations with AHA! 2.0. In the *Proceedings of the PEG Conference*. Sint Petersburg, Russia, 2003. Retrieved February 20, 2006 at http://wwwis.win.tue.nl/~debra/ peg2003/ peg2003.pdf

Stash, N., Cristea, A., De Bra, P. (2004). Authoring of Learning Styles in Adaptive Hypermedia: Problems and Solutions. In *the Proceedings of the World Wide Web 2004 Conference*. New York, USA. May 2004. pp. 114-123.

Tsai, C-Y, Tsai, M-H, (2005). A Dynamic Web Service Based Data Mining Process System. In *Proceedings of the 5th International Conference on Computer and Information Technology* (*CIT'05*). September 21-23, 2005, Shanghai, China. IEEE Computer Society. pp. 1033-1039

Van Dyke Parunak, H. (1991). Hypercubes Grow on Trees (and Other Observations from the Land of Hypersets). In *Proceedings of the '93 ACM Conference on Hypertext*. November 14-18, 1993, Seattle, WA. Pp. 73-81

Vassileva, J. (1997). Dynamic Course Generation on the WWW. In *the Proceedings of the Workshop Adaptive Systems and User Modeling on the World Wide Web, 6th International Conference on User Modeling*, Chia Laguna, Sardinia, 1997. Retrieved February 20, 2006 at http://www.contrib.andrew.cmu.edu/~plb/ AIED97_workshop/ Vassileva/Vassileva.html

Vitali and Bieber (1999). Hypermedia on the Web: What Will It Take?. *ACM Computing Surveys (CSUR), 31(4es),* December 1999, ACM Press, New York, USA.

Wei, Y.Z., Moreau, L. and Jennings, N. R. (2005). A Market-based Approach to Recommendr Systems. *ACM Transactions on Information Systems. 23(3).* 2005. pp. 227-266.

Wiley, D., A. (2000). Connecting Learning Objects to Instructional Design Theory: A Definition, A Metaphor, and A Taxonomy. In *D. A. Wiley* (*Ed.*), *the Instructional*

Use of Learning Objects: Online Version. 2000. Retrieved February 20, 2006 from http://reusability.org/read/chapters/wiley.doc

Wu, H., Houben, G.J., De Bra, P. (1998). AHAM: A Reference Model to Support Adaptive Hypermedia Authoring, In *Proceedings of the Conference on Information Science*, Antwerp, 1998. pp. 51-76. Retrieved April 3, 2006 at http://citeseer.ist.psu.edu/ wu98aham.html

Zakaria, M.R. and Brailsford, T. (2002). User Modelling and Adaptive Hypermedia Frameworks for Education, *Technical Notes*, *the Review of Hypermedia and Multimedia*. 2002.

Zwanenberg, V., Wilkinson, N., and Anderson, A. (2000). Felder and Silvermann's Index of Learning Styles and Honey and Mumford's Learning Styles Questionnaires: How Do They Compare and Do They Predict Academic Performance? *Educational Psychology*, *20* (*3*), 2000. pp. 365-381.