

Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

# **A Story Environment for Learning Object Annotation and Collection**

**A thesis presented in partial fulfilment of the requirements for the degree of  
Master of Science  
in  
Computer Science  
at Massey University, Palmerston North, New Zealand**

**Tianjiao Chen  
2005**

## Abstract

---

With the increase in computer power, network bandwidth and availability, e-learning is used more and more widely. In practice e-learning can be applied in a variety of ways, such as providing electronic resources to support teaching and learning, developing computer based tutoring programs or building computer supported collaborative learning environments. Nowadays e-learning becomes significantly important because it can improve the quality of learning through using interactive computers, online communications and information systems in ways that other teaching methods cannot achieve. The important advantage of e-learning is that it offers learners a large amount of sharable and reusable learning resources. The current approaches such as Internet search and learning object repository does not effectively help users to search for appropriate learning objects.

The original story concept introduces a new semantic layer between collections of learning objects and learning material. The basic idea of the story concept is to add an interpretative, semantically rich layer, informally called 'Story' between learning objects and learning material that links learning objects according to specific themes and subjects (Heinrich & Andres, 2003a). One motivation behind this approach is to put a more focused, semantic layer on top of untargeted metadata that are commonly used to describe a single learning object. Speaking from an e-learning context the stories build on learning objects and become information resources for learning material.

The overall aim of this project was to design and build a story environment to realize the above story concept. The development of the story environment includes story metadata, story environment components, the story browsing and authoring processes, and tools involved in story browsing and authoring. The story concept suggests different types of metadata should be used in a story. This project developed those different metadata specifications to support story environment. Two prototypes of tools have

been designed and implemented in this project to allow users to evaluate the story concept and story environment. The story browser helps story readers to read the story narrative and look at a story from different perspectives. The story authoring tool is used by the story authors to author a story. The future work of this project has been identified in the area of adding features of current tools, user testing and further implementation of the story environment.



# Acknowledgements

---

First of all, I would like to thank Dr Eva Heinrich, my project supervisor, for her support throughout the project. Without the advice, comments and helpful guidance from her, this thesis would not have been possible. I thank her especially for her great patience and her professional, unconditional, dedicated encouragement and support.

Many thanks to the staff in Computer Science group of the Institute of Information Science and Technology at Massey University. Without their encouragement and support, I could not finish the research work.

Finally, I would like to thank my family for their infinite love and support throughout my life.

Tianjiao Chen

Master of Science (Computer Science) Candidate,  
Massey University

Computer Science,  
Institute of Information Science and Technology  
Massey University  
Palmerston North  
New Zealand

# Table of Content

---

<b>Chapter 1 Introduction</b> .....	<b>1</b>
1.1 Research Background .....	1
1.2 Motivation and Objectives for the Research.....	4
1.2.1 Background & Motivation .....	4
1.2.2 Research Objectives.....	7
1.3 Work Included in This Thesis .....	8
<b>Chapter 2 Learning Object Searching Approaches</b> .....	<b>10</b>
2.1 Internet Search .....	11
2.2 Specialized Knowledge Collection .....	12
2.3 Learning Object Repositories .....	14
2.4 Story Concept .....	15
2.5 Conclusion .....	18
<b>Chapter 3 Conceptual Design of Story Environment</b> .....	<b>21</b>
3.1 Story Repository & External Story Metadata .....	21
3.2 Internal Story Metadata .....	24
3.2.1 Domain Independent Metadata .....	25
3.2.2 Domain Specific Metadata.....	25
3.3 Story Authoring .....	26
3.4 Story Browsing .....	28
3.5 Conclusion .....	29
<b>Chapter 4 Story Metadata Design</b> .....	<b>31</b>
4.1 Introduction.....	31
4.2 External Story Metadata .....	32
4.3 Internal Story Metadata .....	35
4.3.1 Domain Independent Metadata .....	35
4.3.2 Domain Specific Metadata.....	52
4.3.3 Example of a Story Annotation.....	53
4.4 Summary.....	54
<b>Chapter 5 Story Environment Tools Design</b> .....	<b>56</b>
5.1 Story Authoring Tool .....	56
5.1.1 Requirements Analysis .....	56
5.1.2 Technology Issues.....	60
5.1.3 Implementation Issues .....	62
5.1.4 Interface Design.....	72
5.2 Story Browser .....	80
5.2.1 Requirements Analysis .....	80
5.2.2 Technology for Story Browser .....	81
5.2.3 Implementation Issues .....	86
5.2.4 Interface Design.....	95

5.3 Summary.....	100
<b>Chapter 6 Evaluation.....</b>	<b>101</b>
6.1 Annotea and the Story Environment.....	101
6.2 User Testing.....	106
6.3 Summary.....	108
<b>Chapter 7 Conclusion and Future Work.....</b>	<b>109</b>
7.1 Thesis Summary .....	109
7.2 Contributions .....	110
7.3 Future Work .....	111
<b>References .....</b>	<b>114</b>

## List of Figures

---

Figure 2 - 1 Shabajee's re-purposing process with additional story layer .....	16
Figure 2 - 2 An example of the story .....	17
Figure 3 - 1 Learning objects and their metadata in learning object repository.....	22
Figure 3 - 2 Stories and their external metadata. ....	22
Figure 3 - 3 Relationship between story repository, story repository management system and story authors/users.....	23
Figure 3 - 4 Process of creating a story .....	27
Figure 3 - 5 Process of browsing a story .....	29
Figure 3 - 6 Story environment structure .....	30
Figure 4 - 1 Use of multimedia cross reference metadata.....	49
Figure 4 - 2 Hierarchical tree of the story domain independent metadata.....	51
Figure 4 - 3 An annotated example from the cultural heritage story in XML format..	53
Figure 4 - 4 Hierarchical position of each story metadata type .....	54
Figure 5 - 1 A view of XML spy data input grid.....	57
Figure 5 - 2 The XML editing window of cook top.....	58
Figure 5 - 3 Process of story authoring .....	62
Figure 5 - 4 Process of loading XML schema.....	63
Figure 5 - 5 Structure of metadata definition list. ....	64
Figure 5 - 6 Structure of metadata annotation list.....	66
Figure 5 - 7 Translation of metadata annotation element to XML tag .....	69
Figure 5 - 8 An example of story design file.....	71
Figure 5 - 9 Main interface of the story authoring tool.....	74
Figure 5 - 10 The narrative view under different metadata mode.....	75
Figure 5 - 11 Detail of annotation area in the story authoring tool.....	77
Figure 5 - 12 A screen shot of tool box in the story authoring tool .....	78
Figure 5 - 13 The screen shot of information area in the story authoring tool.....	79
Figure 5 - 14 Process of loading a story.....	86
Figure 5 - 15 DOM tree of story XML document.....	88
Figure 5 - 16 Process of searching metadata information.....	90
Figure 5 - 17 Process of opening a linked learning object.....	91

Figure 5 - 18 HTML code example of play a media file.....	93
Figure 5 - 19 Process of displaying structure diagram.....	93
Figure 5 - 20 Basic interface of story browser.....	95
Figure 5 - 21 Interface of control area in story browser .....	96
Figure 5 - 22 Structure diagram of a story .....	98
Figure 5 - 23 A snap shot of display area in story browser .....	99
Figure 6 - 1 The basic architecture of Annotea (Kahan & Koivunen, 2001) .....	102
Figure 6 - 2 A screen shot of AMAYA (Kahan & Koivunen, 2001) .....	103
Figure 6 - 3 Physical architecture of the story environment .....	105

## List of Tables

---

Table 2 - 1 A table of comparison between story concept and other learning object searching methods.....	20
Table 4 - 1 Elements in Dublin Core metadata standard.....	34
Table 4 - 2 Summary of three internal metadata annotation standards.....	38
Table 4 - 3 PBL phases in the story context.....	46
Table 5 - 1 Properties and methods of MSXML to travel XML DOM tree.....	89

---

## Chapter 1 Introduction

---

### 1.1 Research Background

With the increase in computer power, network bandwidth and availability, e-learning is used more and more widely. The definition of e-learning generally refers to “learning in a way that uses information and communication technology” (Clarke, 2005). This is a broad definition. Different people, groups and organizations have created different terms that generally narrow the scope of the e-learning definition. In particular, e-learning can be defined as “training delivered in an electronic form, whether via a CD-ROM, DVD, the Internet, or a company intranet. It can occur synchronously (all learners participate learning at the same time and interact each other even if not in the same place) or asynchronously (learners take the course at their desktops at different times)” (Smith, 2004, p22). In practice e-learning can be applied in a variety of ways, such as providing electronic resources to support teaching and learning, developing computer based tutoring programs or building computer supported collaborative learning environments.

Nowadays e-learning becomes significantly important because it can improve the quality of learning through using interactive computers, online communications and information systems in ways that other teaching methods cannot achieve (Clarke, 2005). In comparison with traditional classroom based learning, e-learning has the following strengths:

- Learner-centered and self-paced: traditional classroom-based learning centers on instructors who have control over class content and learning process. E-learning offers learners centered, self-controlled learning environments in which learners control learning content and progress at their own pace.
- Time and location flexibility: e-learning is available anywhere anytime as long as learners have digital equipment such as computers and/or network connections.
- Access to a large knowledge base and archival capabilities for knowledge reuse and

sharing: Taking the advantage of computer power and high speed networks, e-learning is able to provide learners with a large collection of learning resources in diverse formats such as text, image, sound and video. Furthermore, learning resources in e-learning are stored in electronic format which allows for easy sharing, exchange and reuse in different e-learning applications.

As described above, one of the important advantages of e-learning is that it offers learners a large amount of sharable and reusable learning resources. In general learning resources have following the potential:

- Generativity - generative from primitive objects rather than pre-composed.
- Reusability - reusable in different applications.
- Adaptability - adaptive to the individual.

An instructional technology called “learning object” has been introduced for the next generation of instructional design, development, and delivery (Wiley, 2000, p3).

The IEEE Learning Technology Standards Committee (an IEEE organization which develops accredited technical standards, recommended practices, and guides for learning technology) (LTSC, 2004) defines learning object as

“Any entity, digital or non-digital, which can be used or referenced during technology supported learning. Examples of technology supported learning include computer-based training systems, interactive learning environments, intelligent computer-aided instruction systems, distance learning systems, and collaborative learning environments. Examples of learning objects include multimedia content, instructional content, learning objectives, instructional software and software tools, and persons, organizations, or events referenced during technology supported learning” (LTSC, 2004, chapter1).

This is a general definition of learning objects which includes almost everything related to learning. Accordingly different groups outside the Learning Technology Standard Committee have explained this definition from their own perspectives. For example:



- The Multimedia Educational Resource for Learning and On-Line Teaching (MERLOT) project defines learning objects as “Online learning materials” (MERLOT, 2004).
- Wiley defines learning objects as “any digital resource that can be reused to support learning, this includes anything can be delivered across the network on demand, be large or small” (Wiley, 2000).

Wiley (2000) also has identified and categorized learning objects into five different levels. These are:

- Fundamental: an individual, digital resource; commonly used to exhibit and display. Examples of fundamental learning objects are images and figures.
- Combined-closed: a small number of digital resources combined to one object, for example, a video clip that consists of images and sound. Combined-closed learning objects may also contain a small amount of logic such as the sequencing. Combined-closed learning objects generally have a single purpose for instruction or practice.
- Combined-open: a large number of digital resources combined when a request for a learning object is made. A web page is a good example of combined-open learning object, because its component images, video clips, text and other media are combined into a learning object at the request time.
- Generative-presentation: the logic and structure provided to combine learning objects of the lower-level types (i.e. fundamental and combine-closed). A generative-presentation learning object is mostly used in reference, instruction, practice and testing. For example, a piece of Java program which can generate an online test depending on learner’s choice.
- Generative-instructional: the complex logic and structure provided to combine learning objects types (fundamental, combined-closed types, and generative-presentation) and evaluate students’ interactions with those combinations. For example, a tutoring program which can deliver the knowledge, offer the practice and evaluate learner’s performance with feedback.

Based on the above explanations, in this research the term ‘learning object’ is defined as any simple and basic reusable and sharable multimedia digital source that is used to support e-learning. Learning objects in this project are limited to three types: fundamental, combine-closed and combine-open from Wiley’s classification. Examples of the learning objects include electronic course materials, digital pictures and figures, multimedia video/audio clips and so on. The reason of restricting learning object definition to the above three types (fundamental, combine-closed and combine-open) is that those low level learning objects are basic and simple learning objects which can be easily reused and combined to make complex learning materials. Generative-presentation and generative-instructional learning objects normally have complex structure and are used individually rather than components of other learning objects.

## **1.2 Motivation and Objectives for the Research**

### **1.2.1 Background & Motivation**

With the popularity of e-learning, more and more learning resources will be converted to digital format as learning objects and accessed through the Internet. An individual teacher or instructor needs to collect appropriate learning objects to support himself in compiling learning materials for students. In this research, the term of “learning material” is defined as a complicated learning object which falls within the categories of generative-presentation and generative-instructional learning objects from Wiley’s classification. To create learning material one usually selects appropriate basic and fundamental learning objects (fundamental or combined-closed or combined-open learning objects) under a certain topic and includes the logic or instructional information for arranging those learning objects properly and providing context background. Examples of learning materials are lecture notes, study guides and lab exercises and computer-based teaching programs. Usually learning materials are produced by teachers

or instructors. High quality learning material requires teachers or instructors to find suitable learning objects for the learning material's subject and educational purpose. In order to compile the learning materials, teachers or instructors should be able to identify the appropriate learning objects. Learning objects in the learning material need to be closely related to the learning material's topic and contribute to the realization of its educational purpose.

However, for an individual teacher, the task of collecting learning objects for learning materials is very difficult based on the following reasons:

- Lack of specialized learning object repositories/collections: A learning object repository is a searchable database that houses digital resources and metadata that can be reused to search and identify collected learning objects (Daniel, 2004). Metadata is broadly defined as 'data about data'. Generally speaking, metadata provides 'hooks' by which resources can be extracted from or discovered within a repository or a database (Haynes, 2004). The traditional library catalogue index card is a classic example of metadata. The card catalogue identifies what books are in the library and where they are physically located. It can be searched by subject area, author, or title. By showing the author, number of pages, publication date, etc, the catalogue helps people determine which book will satisfy their needs. Following the same principle, metadata in learning object repositories help users to decide which learning objects fulfill their needs. Although most learning object repositories have a reasonable amount of learning objects, they usually contain learning objects in various subjects and domains, which are not specialized for a small topic or subject. Furthermore most current learning object repositories only can provide general and surface metadata description about learning objects rather than description and explanation from a specific subject point of view. Hodgins (2005) suggests that learning objects should be put into context to meet the individual needs of learners in different locales and cultures. The subject or domain context descriptions of the learning objects would definitely assist teachers or instructors to select learning

objects in different situations. Without the specialized knowledge and context description of the learning objects, it is difficult for teachers and instructors to reuse the learning objects in their areas.

- Lack of learning-focused specialized knowledge collections: Specialized knowledge collections such as online museum collections normally contain a large amount of learning objects under a single subject or topic. Learning objects in specialized knowledge collections are carefully selected and described by domain experts to ensure the authenticity. However the problem of specialized knowledge collections is that specialized knowledge collections do not focus on teaching and learning environments. Commonly, specialized knowledge collections provide sufficient descriptions of learning objects from a subject point of view but ignore explaining how learning objects would be used to teach students. Instructional and pedagogical information is quite important for teachers or instructors to correctly reuse learning objects in their teaching context. Due to the educational purpose of learning objects, instructional information must be incorporated in any learning object implementation that aspires to facilitate learning (Wiley, 2005).
  
- A teacher may not have sufficient and highly specialized domain knowledge to find the best learning objects. Lots of learning objects on the Internet have not been evaluated, assessed and approved by domain or subjects experts. The individual teacher or instructor might lack the specialized knowledge to select and identify learning objects of high quality.

The story concept (Heinrich & Andres, 2003b) has been developed and introduced to address these issues. The basic idea of a story is to add an interpretative, semantically rich layer, called ‘Story’, between learning objects and learning materials that links learning objects according to specific themes and subjects. A story collects different learning objects under a specific subject and also provides the context narrative and explanation for those collected learning objects. Speaking from an e-learning context

the stories build on learning objects and become information resources for learning materials.

A story consists of three main elements: the narrative, links to learning objects and story metadata.

- The narrative is what the author of the stories writes to describe the collected learning objects. The author can use the narrative to tell facts, provide interpretations, make comparisons, draw attention or similar.
- A link is a connection between a part of the textual narrative and related learning objects. The link can refer to a particular part in a learning object or a whole learning object.
- The story metadata is to make it possible to search for a story and customize a story according to a user's point of view.

### **1.2.2 Research Objectives**

The main objective of this research is to develop and implement a story environment based on the basic story concept. The objectives of this research can be summarized to the following points.

- Design and develop the story environment: The story environment will implement the story concept. The design and development of the story environment include identifying the components in the story environment and demonstrating the functions offered by the story environment. The following issues need to be concerned by the story environment:
  - Classify different roles and components in the story environment and their responsibilities.
  - Identify different functions and processes in the story environment.
  - Specify different applications involved in the story environment.

- Further extend the concept of the story metadata and develop the story metadata specification: The original story concept only introduced and suggested the story metadata on a conceptual level. It is necessary to further extend and design the story metadata to achieve the following purposes:
  - Find appropriate stories from story collections.
  - Find appropriate learning objects in a story.
  - Annotate the story narrative from different perspectives.
  - Provide instructional information to the story narrative.
  
- Develop and implement applications to support the story environment: In order to evaluate the story concept and the story environment, it is necessary to develop and implement applications in the story environment. Basically, two applications should be developed at first.
  - An application to create and author a story including linking learning objects, editing story metadata data and writing the story narratives.
  - An application to browse a story including viewing the story narratives, searching for learning objects and narratives by story metadata.

### 1.3 Work Included in This Thesis

The objectives of this research have been identified in the previous section. To realize the research goal, the following work has been undertaken and reported in this thesis.

- Review literature. This part reviewed and discussed the original story concept and other learning object search methods such as learning object repositories, specialized knowledge collections and general Internet search. By reviewing the existing methods and story concept, it can explain advantages and strengths of the story concept and create the literature background to develop the story environment.

- Conceptualize story environment. This part developed and conceptualized the story environment. In detail this part explained the story environment components and their relations and discussed issues to be developed and implemented in this research.
  
- Develop and implementing the story environment. Story metadata have been specified in detail with a specific metadata specification. Applications involved in the story environment have been designed and implemented in this part as well.
  
- Evaluate the story environment. The story environment has been evaluated by users from different areas to approve its usefulness and efficiency. Applications implemented in the story environment have been tested by users to evaluate the functions and interface design.

## Chapter 2 Learning Object Searching Approaches

---

Over the last years there have been an increasing number of approaches and projects for indexing and searching for learning objects. All of these approaches aim to provide users with large collections of learning objects and a mechanism for searching for the learning objects efficiently and effectively. Examples are:

- Internet search – search for learning objects with Internet search engines.
- Learning object collections/repositories – search for learning objects in a special database which stores learning objects and metadata for the collected learning objects.
- Specialized knowledge collections/repositories – search for learning objects in a special collection that is highly specialized for a specific subject or domain, such as online museum collection.

The story concept has been suggested by Heinrich and Andres in 2003. The idea of the story concept is to provide users a new intermediate layer called ‘story’ on the top of basic learning objects which includes learning objects of a specific subject, context description and metadata.

This chapter provides an overview of the above approaches and discusses the literature background of the story concept. Some guidelines have been used in this chapter to evaluate and compare different approaches:

- *Information source.* The knowledge base can be searched and extracted by each approach.
- *Metadata.* Metadata evaluates issues of how metadata are involved in each technology. For example, whether metadata have been used, whether metadata are explicit or implicit and whether metadata can be changed or defined by users.



- *Quality of control.* The quality of learning objects in the information source.
- *Author/User.* This criteria looks at authors of the learning objects in the information source and at users who search for the learning objects.

## 2.1 Internet Search

Internet search powered by web search engines may be considered as the best known and widely used approach for people to search for information. Internet search provides users the easiest way to search for information by simply typing some keywords related to search criteria.

Currently there are many search engines available on the Internet such as Google, Yahoo and so on. The typical characteristic of search engines is that they index a huge amount of different resources on the Internet including web pages, documents, images and movies. Most Internet search engines build their database by employing a program to crawl through web space from link to link, identifying and caching pages (Bones, 2004). Once the program gets to a web site, it typically indexes most of the words on the web pages at that site. When a user searches for the Internet by using a search engine, the search engine basically scans its index and matches the submitted query keywords within the engines database. In recent years, some new technologies have arrived to improve the search efficiency. For example, Google rank a web page based on both keywords match and popularity which means how frequently other sites links to that page.

In terms of searching learning objects, Internet search has its own strengths. Because Internet search searches general information, it can provide access to a huge number of the web resources. Learning objects are part of the web resources. Therefore by using Internet search users are able to get a large number of the potential learning objects.

Further, the searching database provided by Internet search engines is updated and refreshed frequently to collect the latest pages. Therefore most learning objects indexed by search engines are active and available for users.

On the downside, firstly the performance of Internet search is restricted and limited by its searching methods and algorithms. Most Internet search engines only search their caching databases by matching the keywords. The simple keyword matching is not accurate to deliver the search results in terms of relationship to the search topic. Internet search engines do not include specialized metadata to describe learning resources from different aspects. Consequently Internet search is not able to return precise search results and offer users context information about learning resources. Secondly the Internet is an open network. Anyone can publish his or her work to the Internet. Learning objects distributed on the Internet have not been properly classified and authorized by domain or subject experts. Therefore the quality of learning objects cannot be guaranteed and approved. Finally resources on the Internet can be from everywhere. Internet search is not designed to be used in a learning environment. It is general for all contexts and environments, which makes it difficult to reuse resources from the Internet in teaching and learning context.

## **2.2 Specialized Knowledge Collection**

Specialized knowledge collections gather information resources in a very specific manner. They only store resources of high quality and authority and closely related to subject and topic. A typical example of a specialized knowledge collection is an online knowledge collection maintained by a museum or domain experts, such as the Digital Silk Road collection. Digital Silk Road is a global multimedia repository for the study of the historical Silk Road. Digital Silk Road collection contains various resources of well known 'silk road' (Ono, 2004). The resources in specialized knowledge collections

are carefully selected and described within the subject context. Domain experts manually collect resources for specialized knowledge collections, so that they can evaluate resources and ensure the quality of collected resources.

From the quality control point of view, specialized knowledge collection offers high quality resources. However compared with Internet search, the search space of the specialized knowledge collection is relatively small because of human effort required and the focus on a specialised subject or domain. It can promise the authenticity since resources in the specialized knowledge collections are reviewed by domain experts with predefined criteria. A good specialized knowledge collection involves high level of expert knowledge that will be used to filter and select resources.

The important advantage of specialized knowledge collections is that they include sufficient context descriptions of collected learning resources from a subject point of view. But specialized knowledge collections put a strong focus on the subject and domain context rather than the learning and teaching environment. Thus a specialized knowledge collection does not provide the learning context for the resources such as instructions for reading resources and difficult level for resources, which makes it difficult to reuse the resources in teaching and learning. As with Internet search specialized knowledge collections do not include specific metadata for annotating and commenting the collected resources. Because specialized knowledge collections do not focus on using metadata to annotate resources, it makes hard and difficult for teachers to locate the most useable resources efficiently.

## 2.3 Learning Object Repositories

Learning object repositories have been developed to allow users to easily search and reuse learning objects by using metadata. This helps reusing and sharing learning objects more efficiently and focusing more on learning criteria rather than other areas. A learning object repository is a searchable database that houses digital resources and/or metadata that can be used to identify learning objects (Daniel, 2005). Learning object repositories are designed to either store the digital resources directly or maintain links to learning resources stored elsewhere on the Internet. Nowadays more and more global learning object repositories are appearing. There are two big learning object repositories CAREO (CAREO, 2004) and MERLOT (MEROLOT, 2004). They are essentially global catalogs for learning objects that are available at different levels of granularity. They do not physically store the learning objects and both of these two learning object repositories use metadata to index the collected learning objects.

As one of the advantages of using learning object repositories, learning object repositories include metadata sets to provide descriptions of learning objects and facilitate search. Currently some metadata standards have already been developed and designed by different institutes and organizations such as Dublin Core (DC, 2004) and LOM (LOM, 2004). The details of metadata will be discussed in Chapter 4.

With the support of metadata, most learning object repositories can easily identify the learning objects and produce precise search results. Metadata standards include many individual fields to describe a learning object from general point of view to a teaching and learning point of view, such as author, date, difficulty and format. As a result, learning object repositories work very well for people who are actually familiar with learning objects and metadata. For those people who are not familiar with learning objects and metadata, they might need time to find out the way of using metadata.

Learning object repositories allow their users or members to submit their learning objects and edit the metadata descriptions for their learning objects. The information source of learning object repositories is of medium size which is smaller than the general Internet search. Although most of users and members of learning object repositories are engaged in teaching and learning field, they might not be domain experts who have sufficient knowledge and background in a specific subject or domain.

## 2.4 Story Concept

The basic idea of a story is to add an interpretative, semantically rich layer, informally called 'Story' between the learning objects and learning material that collects learning objects according specific theme and subject (Heinrich & Andres, 2003b). The major difference between the story concept and other approaches is that teachers do not search for individual learning objects directly. Each story, which is created by highly specialized domain experts, contains links to theme or subject related learning objects and a narrative description of those learning objects. Instead of searching for learning objects, users locate the appropriate learning objects by reading through the categorized stories. One motivation behind this approach is to add a more focused, semantic layer on top of untargeted metadata that are commonly used to describe a single learning object (Heinrich & Andres, 2003a). In principle, stories build on learning objects and become information resources for learning material (Heinrich & Andres, 2003b).

A story can be placed in Shabajee's (Shabajee, 2002) repurposing diagram. A story can sit between the digital assets with their metadata and the composite objects. Learning objects and their metadata are called raw assets. The 'raw' digital assets have metadata linked to them and then are stored with their metadata in a database. Through searching the metadata, suitable raw documents are identified. The composite objects are learning material. The story includes the links to the assets and narratives that describe or

explain the assets or create semantic relationships between assets (Heinrich & Andres, 2003b). The story can be treated as a semantically rich layer between the assets collection and composite objects. It collects the assets according to the themes or subjects and provides the interpretation for those assets at the same time. Figure 2.1 is the Shabajee's re-purposing process with additional story layer.

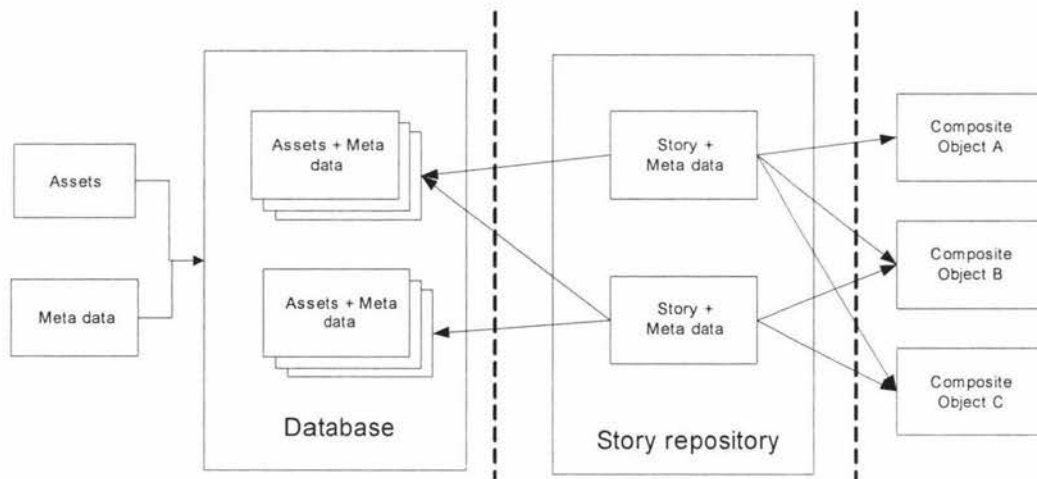


Figure 2 - 1 Shabajee's re-purposing process with additional story layer

A story consists of three main elements: the narrative, links to learning object segments and story metadata. Figure 2.2 indicates an example of the story.

- The narrative is textual data that the author of the stories writes to describe the collected learning objects. The story author can use narrative to tell facts, provide interpretations, make comparisons, draw attention or similar.
- A link is a connection between a part of the textual narrative and related learning objects. The linked learning objects can provide examples, illustration, proof, further explanation and additional material for a part of narrative description. Usually, a link will be able to refer to a particular segment in a learning object. While the links refer to specific document sections, the whole document is still reachable via the link.

- The role of the story metadata is to make it possible to search for a story and customize story from a user's point of view. Two types of metadata have been proposed. One allows users to locate a suitable story in a story repository. Another describes a particular story from inside. This kind of metadata annotates the story narrative, which enables users to find different pathways through a story or to focus on specific aspects provided in a story.

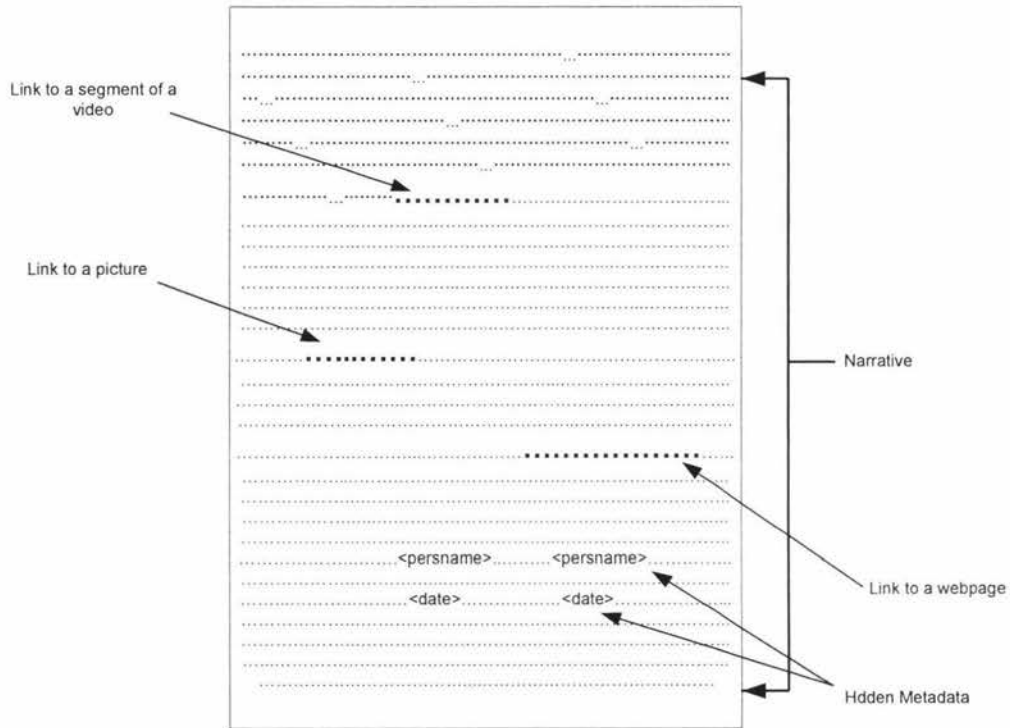


Figure 2 - 2 an example of the story

As described above, each story refers to a list of learning objects which are related to a certain topic and theme. Metadata are widely used in the story concept. Story metadata describe stories from different angles and perspectives and story authors are allowed to define their own metadata to describe stories. This provides users both context and educational information about a story itself and learning objects inside the story. Also by using metadata users are able to easily and quickly find out the appropriate stories from the story repository and the appropriate sections and parts inside a story.



Because learning objects in stories are selected by domain experts, the quality of learning objects is controlled by domain experts. But due to the human effort involved in collecting learning objects, writing narrative and specifying metadata, the story concept can not have a very large information source.

## 2.5 Conclusion

In the previous sections, the different learning object searching approaches have been reviewed including Internet search, specialized knowledge collection and learning objects repository. Also the story concept has been introduced and explained. After reviewing each approach with the predefined criteria, the following points can be summarized.

Internet search has the biggest search space. There are no limitations on search scope and subjects. Everything on the Internet can possibly be picked up by Internet search. But as Internet search only searches for resources by matching keywords rather than using metadata, Internet search may not be able to deliver precise and accurate search results. Since the Internet is an open network and everyone can publish their work on the Internet, the quality of learning resources identified by Internet search is not guaranteed and approved.

Specialized knowledge collection has the best quality control of resources. Every single resource in specialized knowledge collection is selected by domain experts who are familiar with their areas. Because of high demand in human effort specialized knowledge collections can not have a fairly large search space. Also the problem with specialized knowledge collection is that it does not strongly focus on teaching and learning environment. There are no sufficient educational instructions or information appended to collected resources.



Learning objects repository focuses on metadata and learning context. Learning object repositories collect learning objects by members or users submit. From the quality of control point of view, the authors of the learning objects in the learning object repository approach are not domain or subject experts. Accordingly the quality of learning objects in learning object repository is not fully controlled. One advantage of learning object repository is that it offers metadata option to describe each resource that provides users learning and teaching information.

The story concept emphasizes on providing users with both context and learning information about learning objects. The story concept asks domain experts to collect learning objects and write the narrative description of learning objects, which limits the information source of the story concept. However domain experts' effort can ensure the quality of learning objects and provide context and learning description about learning objects. Metadata are widely used in the story concept to help story users to search for learning objects and locate specific narrative section. Furthermore domain experts are allowed to define their own metadata definition to describe a story.

Table 2.1 shows a table of comparison between story concept and other learning objects searching methods.

	Internet Search	Specialized Knowledge Collection	Learning Object Repository	Story Concept
Information source	Wide	Subject Specific	Learning and teaching specific	Subject and learning specific
Metadata	None	few	Predefined	Predefined & User defined
Quality Control	Low	High	Medium	High
Generic	Commercial and general area	Specific for a single subject area	Specific for Learning and teaching area	Specific for learning and teaching in a specific subject area
Author	Anyone	Domain experts	People involved in teaching and learning	Domain experts

Table 2 - 1 a table of comparison between story concept and other learning object searching methods.

As the comparison shows, the story concept has many advantages over the other approaches. Especially, the story concept has different kinds of metadata to facilitate searching and asks domain experts to control the quality of learning objects. In the work before this thesis research the story concept has only been introduced in a very basic conceptual level without specific design. So it is necessary to create a story environment to actually implement the ideas in the story concept.

In the next chapter, the story environment will be designed to realize the story concept. Meanwhile tools and metadata involved in the story concept are necessary to be classified before implementation.

## Chapter 3 Conceptual Design of Story Environment

---

The story concept has many advantages, especially of providing context description for learning objects and using different metadata to annotate the story narrative. In order to implement the story concept this chapter will conceptualise an environment for the story concept. Also story metadata and tools involved in story authoring and browsing are discussed and explained in detail.

In the last chapter, the story concept has been explained and compared to other learning object searching approaches and technologies. Also the structure of a story has been introduced. This chapter concentrates on discussing and revealing a story environment to support to create, browse, store and exchange stories.

### 3.1 Story Repository & External Story Metadata

Because a story can be considered as a learning object, a story repository is basically a container or database which functions like a learning object repository to store stories. All stories would be centrally stored in a story repository for users to browse. The central component of a story repository is the external story metadata.

In the learning object repository approach the function of metadata is to locate learning objects from their repositories. Each learning object will be associated with a set of metadata to describe its properties and attributes. Users are able to find the appropriate learning objects by specifying the corresponding metadata information. Figure 3.1 shows learning objects and their metadata in the learning object repository.

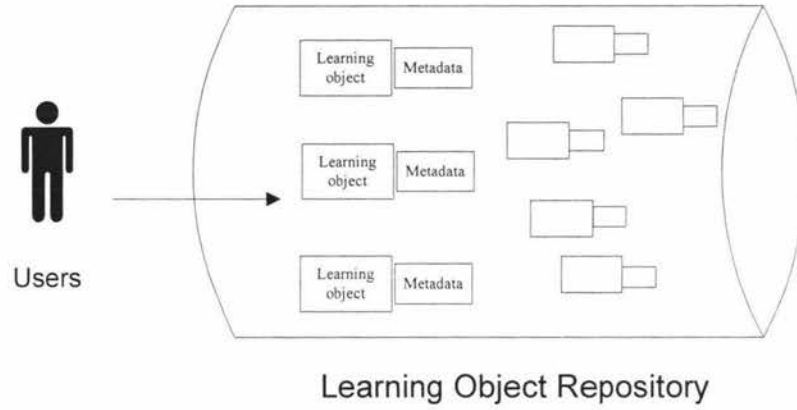


Figure 3 - 1 learning objects and their metadata in learning object repository.

A story itself can be treated as a learning object and it also will be stored in the story repository. Therefore stories should have the same mechanism to describe them from outside and help story users find stories. External story metadata are introduced to play the role of learning object repository metadata. In the story environment external story metadata are associated with each story in the story repository to describe the learning and educational properties of the story. External story metadata help teachers to retrieve and identify the appropriate stories from story repository. Figure 3.2 shows stories and their external metadata.

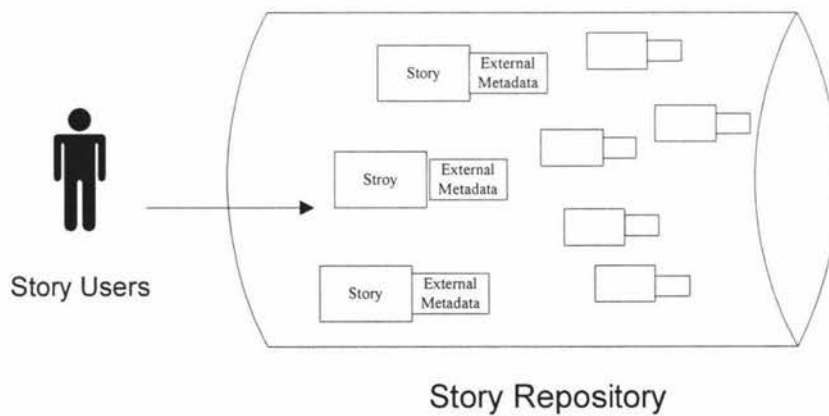


Figure 3 - 2 stories and their external metadata.

From the above description, it is important to have external story metadata to identify and locate stories in the story repository. Consequently, there should be a program that builds on the story repositories to allow users to specify external story metadata for stories or search stories by matching external story metadata. Also story authors and users need to communicate with the story repositories to upload and browse stories. Therefore in this project a system called story repository management system is required. The story repository management system manages stories and their metadata in the story repository. The basic requirements or functions of the story repository management system include:

- Upload and download stories to/from the story repository.
- Search and match the proper stories by using external story metadata.
- Allow story authors to specify external story metadata for their stories
- Manage the stories inside the story repository by providing operations such as copy, delete.

Figure 3.3 shows the relationship between story repository, story repository management system and story authors/users.

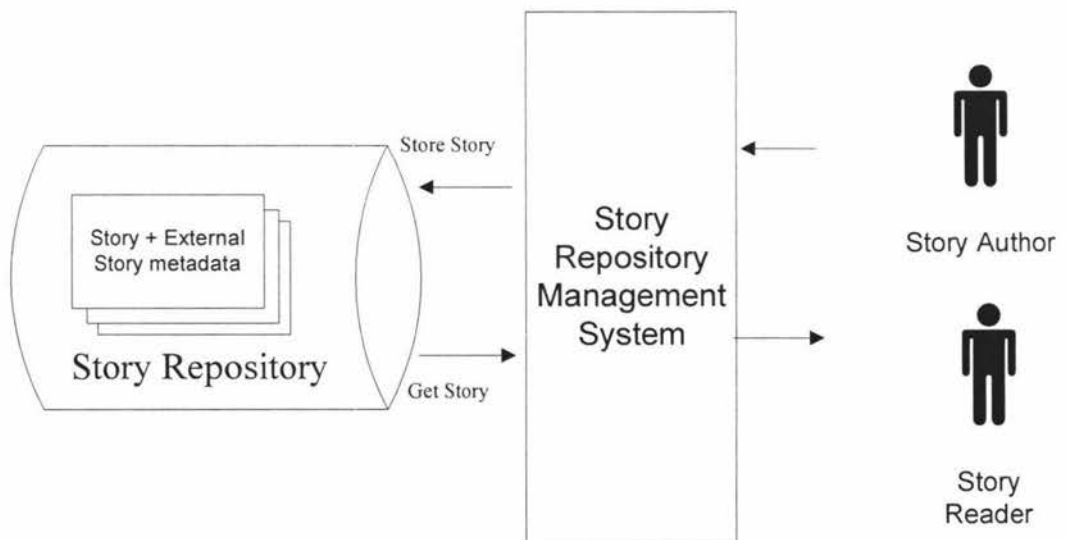


Figure 3 - 3 relationship between story repository, story repository management system and story authors/users.

## 3.2 Internal Story Metadata

Before actually starting to explain story authoring and story browsing, it is very important to clearly introduce and classify internal story metadata which are involved in both story authoring and browsing.

The concept of the external story metadata has already been introduced in the last section. The function of the external story metadata is to describe the story as a learning object that is used to locate stories from the story repository.

In Chapter 2 another kind of metadata has been mentioned as part of the story concept. The task of those metadata is to guide story readers to search inside the story which allows them to locate a specific segment in the story narrative or find different pathways through the story narrative. This project defines this kind of metadata as internal story metadata.

Internal story metadata are used to annotate a story narrative from different perspectives, which allows users to view a story according to their personal needs.

Because stories focus on a small subject, internal story metadata should be able to describe stories from both general and subject perspectives. There are two different sets of internal story metadata. Domain independent metadata are independent of any theme or subject area. They group metadata categories that are common across stories from different subject domains. Domain specific metadata are created by story authors and may vary with the domains and themes of stories. Domain specific metadata can link across stories via a shared domain specific metadata definition.

### 3.2.1 Domain Independent Metadata

Domain independent metadata are designed to capture the common features of stories that are the same across different domains. Most general features of stories will be represented by domain independent metadata. The following are some areas which are considered to be important for the story annotation.

- Story narrative structure – the general structure of the story narrative, such as heading, division, paragraph and sentence. By defining this kind of metadata, the structure of the story narrative is captured.
- Special terms – Many important special terms in the story narrative should be highlighted by domain independent metadata elements as well, for example, date, person name, address etc.
- Instructional information – Due to the educational purpose of stories, domain independent metadata must have some elements to afford instructional information.

### 3.2.2 Domain Specific Metadata

Domain specific metadata elements are defined by story authors. The quality of the story domain specific metadata determines the value of a story. If the story domain specific metadata are not accurate and precise, it will locate and navigate irrelevant and unimportant information for story readers. In order to produce stories of high quality, authors of stories must be experts in a specific domain and subject area who have the specialist knowledge, language skills. Also story author should be familiar with both specifics of the domain and of the learning objects collections they are discussing in the stories. Domain specific metadata are closely related to domains and subjects of the stories. Domain experts can define domain specific metadata schema for a group of stories under a certain topic or theme.

### 3.3 Story Authoring

This section will discuss how story authors create stories.

Stories are ideally authored by domain experts who are familiar with both specifics of the domain and the learning object collections they are discussing in the story narratives. In story authoring, domain experts, who might be involved in learning objects collection and annotation, will acquire intimate knowledge about the data repositories for their specialist areas. Domain experts are required to create the domain specific metadata for their stories. When a story has been created, it will be stored in the story repository together with its external story metadata. Story repositories within the story environment will be formed according to domains or subjects. In order to facilitate the creation of stories the story authoring tool is introduced to help story authors to compile stories. The story authoring tool needs to provide functionalities for editing the story narrative, linking to collected learning objects and editing internal story metadata.

Figure 3.4 illustrates the process of story creation by the domain expert. In principle five steps are involved in creating a new story.

- 1) The domain expert collects the related learning objects under a topic or subject. These selected learning objects could be from everywhere, e.g., the Internet or a digital library.
- 2) The domain expert has to create the metadata definition for the story domain specific metadata or find an existing domain specific metadata definition for the story.
- 3) The domain expert uses the story authoring tool to write the story narrative, link the collected learning objects and use the internal story metadata to annotate the story narrative.



4) Domain expert creates the external story metadata for the story.

5) Domain expert adds the story to the story repository with its outside story metadata.

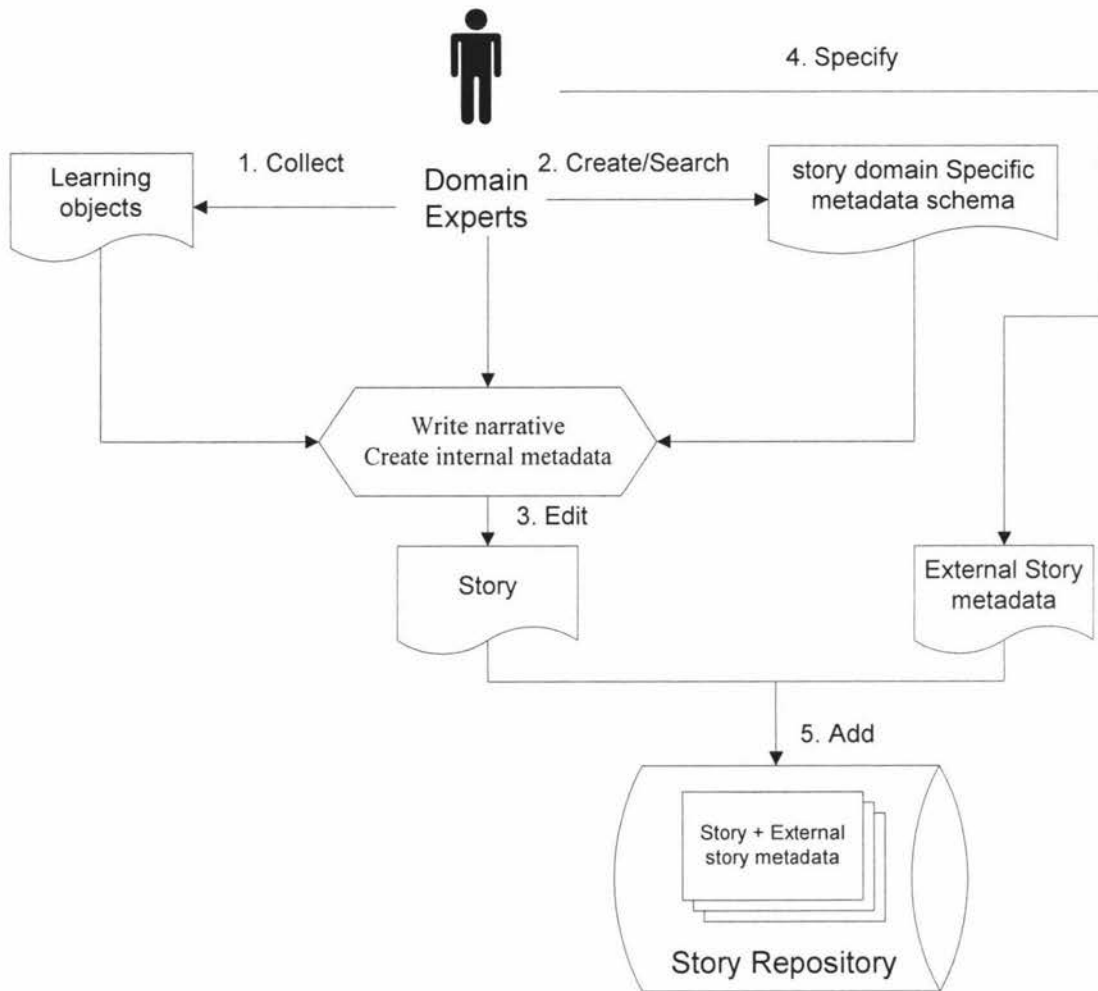


Figure 3 - 4 process of creating a story

### 3.4 Story Browsing

This section mainly focuses on how a story will be displayed to story readers and how story readers can locate stories.

Once a story is written and annotated with the external story metadata, it has to be placed in the story repository. The story reader locates the story via the external story metadata from repositories and navigates in the story via the internal story metadata. The task of retrieving stories from the story repository will be performed by story repository management system which allows users to find appropriate stories by specifying the external story metadata. For using a story, the story narrative has to be presented and the story internal metadata have to be made accessible to allow users to search in a story. Therefore a tool is required to analyze the story. In the story environment, this tool is named story browser. The basic function of the story browser is to retrieve the story narrative and the internal story metadata from a story, search for the story narrative using the internal story metadata and open the linked learning objects inside the story narrative.

In the story concept, the users of stories are teachers who draw on the knowledge transmitted in the stories and support the referenced learning objects to develop the learning materials. Figure 3.5 shows the process of a teacher browsing a story.

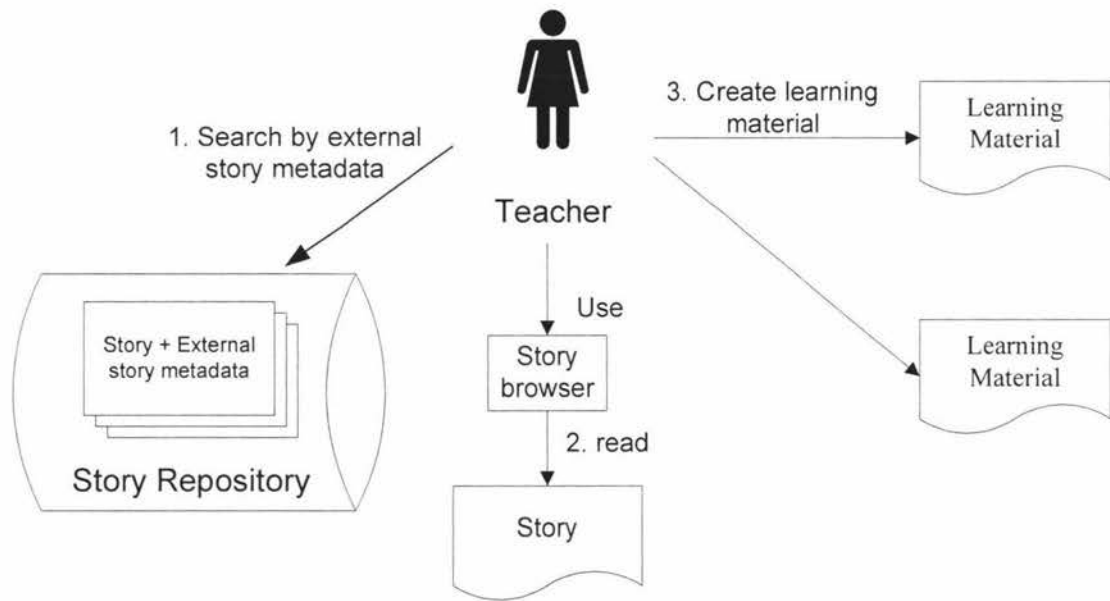


Figure 3 - 5 process of browsing a story

### 3.5 Conclusion

In this chapter the main features of the story environment have been conceptualized. In conclusion, the story environment can be summarised as in the Figure 3.6. Story authors use the story authoring tool to create stories. Story readers use the story browser to view stories. Both the story authoring tool and story browser can communicate with the story repository via the story repository management system. The story repository management system manages stories inside the story repository and also allows the story authoring tool and story browser to upload and download stories from the story repository.

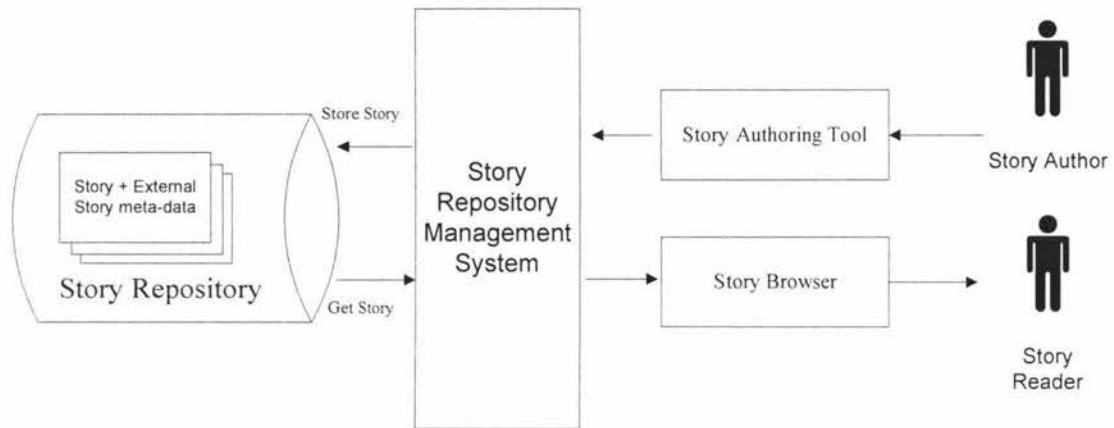


Figure 3 - 6 story environment structure

In addition this chapter explained two major issues to be further investigated in next chapters.

Firstly, the different kinds of metadata and their functions have been classified.

- External story metadata identify stories in the story repository as learning objects.
- Internal story metadata annotate the story narratives in a story. They show the structure, highlight the important terms and create the link to subject or domain related learning objects.

Secondly, the processes of story authoring and browsing have been explained. There are two tools involved in story authoring and browsing that need to be implemented in this project:

- The story authoring tool provides story authors with an environment to visually create stories.
- The story browser retrieves the story narrative from the story and displays it to the story readers. The metadata information in a story is also extracted by the story browser, which will be used to enable story readers to search for the story narratives and view the links to the external learning objects.

## Chapter 4 Story Metadata Design

---

### 4.1 Introduction

The idea of metadata is not new. Long before the emergence of learning objects in the learning community, metadata had been widely popularized and utilized in many other information areas such as library science, publishing industry and web community. In fact, metadata and metadata standards lie behind almost all technological systems involving display and exchange of information (Brody, 2003). As mentioned before, traditional search engines and indexes are inadequate for locating information due to the lack of the detail and precision to deal with large number of resources, but metadata is an effective solution to this problem. EdNA (EdNA, 2004) describes, “the creation of metadata to describe resources assists in resource discovery and resource management. Consistent cataloguing of online resources means maximized opportunities for searchers to find the most relevant and comprehensive set of resources for their purposes. Metadata can also be used to organize, store and retrieve items for information management purposes”. The advantages of using metadata can be summarized as follows (Hayness, 2004).

- Metadata enables resource description and enhances retrieval performance.
- Metadata provides a way of managing information resources.
- Metadata helps document ownership and determines authenticity of data.
- Metadata is the key to reusability and interoperability.

The last chapter introduced the story environment and explained the different types of the metadata involved in the story environment. Two types of metadata have been distinguished in the story environment.

- Internal story metadata are used to annotate the story narrative from different perspectives, which allows users to view a story according to their personal needs.
- External story metadata describe a story as a learning object, which is used to locate stories from story repository. As discussed previously, a story itself is seen as a learning object.

## 4.2 External Story Metadata

As described in last chapter external story metadata are added to each story in the story repository to describe its learning and educational properties. External story metadata help story users to retrieve and identify the appropriate stories from the story repository.

Many external learning object metadata specifications have already been developed around the world, such as LOM, Dublin etc. Because there is not too much difference between learning objects and stories from an outside view, the current learning object metadata specification elements can be reused in the external story metadata. The following section reviews the LOM and Dublin Core metadata standards.

### *LOM*

LOM specifies the syntax and semantics of learning objects. LOM focuses on the minimal set of attributes needed to allow these learning objects to be managed, located, and evaluated (LOM, 2004). At the moment, more than 70 attributes have been defined by LOM. Relevant attributes of learning objects to be described include type of object, author, owner, terms of distribution, and format. Moreover, LOM also includes

pedagogical attributes, for example teaching or interaction style, grade level, mastery level, and prerequisites. Data elements in LOM have been grouped into 9 categories in LOM specification. Each category contains a group of related data elements (LOM, 2004).

- *General* – general features of a learning object like identifier, title or Language.
- *Lifecycle* – features related to history of a learning object like version or status.
- *Meta-metadata* – Origin and edition of the metadata
- *Technical* – technical requirements and technical characteristics of a learning object.
- *Educational* – the educational and pedagogic features of a learning object.
- *Rights* – rights and conditions of use for a learning object.
- *Relation* – features of the learning object in relationship to other learning objects.
- *Annotation* – provides comments on the educational use of a learning object and provides information on when and by whom the comments were created.
- *Classification* – where a learning object falls within a particular classification system.

### *Dublin Core*

Dublin Core metadata is an interoperable metadata standard which provides the specialized metadata vocabularies for describing resources that enable more intelligent information discovery systems (DC, 2004). The Dublin Core metadata element set is a standard for cross-domain information resource description (DC, 2004). The following elements have been included in Dublin Core (see Table 4.1)

Element Name	Description
Title	A name given to resource
Creator	An entity responsible for making the content of resource
Subject	A topic of content of the resource
Description	An account of the content of the resource
Publisher	An entity responsible for making the resource available
Contributor	An entity responsible for making contributions to the resource
Date	A date of an event in the lifecycle of the resource
Type	The nature or genre of the content of the resource
Format	The physical or digital manifestation of the resource
Identifier	An reference to the resource within a given context
Source	A reference to a resource from which the present resource is derived
Langue	A language of the content of the resource
Relation	A reference to a related resource
Coverage	The extent or scope of the content of the resource
Rights	Information about rights held in and over the resource

Table 4 - 1 elements in Dublin Core metadata standard

LOM and Dublin Core are the two fundamental standards in the learning object metadata area. The Dublin Core metadata standard has the broadest level of commonality of elements, understood semantics, extensibility, international acceptability and the flexibility (MatThlha, 2004). Dublin Core also provides for extensions to the basic elements to meet local needs. On the downside, Dublin Core does not have sufficient information about educational approaches and rights management. In contrast to Dublin Core, LOM provides a rich set of elements to describe a learning object. LOM describes educational learning objects with an organized hierarchical metadata structure. Furthermore, LOM includes an education



metadata category to provide some brief educational information about a learning object. In the story context, external story metadata can be implemented by LOM or DC directly. The actual design of the external story metadata is not in the scope of this project, which will be developed in future. This project focuses on developing the internal story metadata.

### **4.3 Internal Story Metadata**

It is very important to have metadata to mark up the important features inside a document such as structure, special terms and references. Internal metadata are embedded into a document to annotate these features inside a document, so that it can help readers to read and understand the document. Internal story metadata focus on commenting the story narrative. By using internal story metadata story readers can find different pathways through a story or focus on specific aspects provided in a story.

In this project, the internal story metadata include domain independent metadata and domain specific metadata. Domain independent metadata annotates the general features of stories, which will be the same for all stories. Domain specific metadata strongly relies on the subjects of stories, which will be defined by story authors.

#### **4.3.1 Domain Independent Metadata**

##### **4.3.1.1 Document Annotation Standards**

Some metadata standards projects have already been conducted and developed to capture the common features of documents. In order to design the story domain independent metadata specification, it is worth to look at those standards first. The following sections will look at the current document annotation projects.

*TEI (Text Encoding Initiative)*

“The TEI is an international and interdisciplinary standard that helps libraries, museums, publishers, and individual scholars represent all kinds of literary and linguistic texts for online research and teaching, using encoding schema that is maximally expressive and minimally obsolescent” (TEI, 2004, chap1). TEI aims to encode all the semantically significant aspects of literary texts. These significant features do not only include words themselves which are most important semantic features, but also contain some other important information such as various types of emphasis, indentation and margins, tables and even decorations. TEI enables effective searching, indexing, comparison and print publication.

Following are important levels in TEI which are used to annotate narratives in a document (TEI, 2004).

*Reading Level.* Reading level focuses on document narratives. The narratives at this level will be marked up to support basic reading, browsing, retrieval, and navigation.

*Pedagogical Level.* The markup at this level encodes document structure to enable search, retrieval, and display for the purposes of teaching or basic research. Additionally, it may contain references to external documents for purposes of text comparison, alignment, and reference.

*Dictionaries Level.* This level provides the dictionary information about electronic documents, such as headworks, grammatical information, morphological forms, definitions and translations etc.

In summary TEI provides users a broad range of metadata tags to mark up a document from different perspectives. Metadata tags in TEI cover most general aspects of a text document including structure of a text; page and line numbers; notes; cross reference and links; omissions. Metadata tags in TEI have been organized in a hierarchical manner.

*Global Document Annotation (GDA)*

GDA (Global document annotation) aims at having authors annotate their electronic documents with a common standard linguistic tag set which enables computers to automatically recognize the semantic structures of the documents (GDA 2004). GDA produces an integrated global platform for electronic document authoring, presentation and reuse. The motivation of GDA annotation is that documents can be retrieved and translated with higher accuracy. Further, GDA can easily facilitate information interchange, sharing and reuse.

Actually, lots of elements in GDA overlap with the TEI reading level specification. For example, both specifications have the element <dv> that is used to mark up a division in a text and the element <date> to mark up a date in the text. However compared with TEI, GDA offers more specific tags which even enable text authors to mark the structure of a sentence in detail. For example in GDA, <np> is used to mark up a noun phrase in a sentence; <vp> is a verb phrase in a sentence. Another advantage of GDA is that GDA introduces the concept of relation identifier. A relation identifier can represent primitive binary relations between elements in a sentence. A purpose of relation identifiers is to associate complement elements (subjects, objects, indirect objects, and so forth) with the corresponding arguments of verbs, adjectives, etc (GDA, 2004). For example, relational identifier 'agt' means agent of action. It can be used in the following scenario <np opr="agt">Tom </np><v>came</v>.

Compared with TEI, GDA focuses on the syntax of a document, which would be very useful in natural language processing. The hierarchical tags in GDA are used to mark up the structure of a document and syntax structure of a sentence. However GDA has not provided specific metadata tags to annotate a document from pedagogical perspective.

*Corpus Encoding Standard for XML (XCES)*

The XML Corpus Encoding Standard (XCES, 2004) is another standard related to document annotation. XCES is designed as an encoding standard for natural language applications. The annotation features of XCES are almost the same as in TEI and GDA, and include basic document structure, linguistic annotation and data structure for linguistic annotation. At present, XCES provides XML implementations for encoding basic document structure (down to paragraph-level elements), morphosyntactic annotation and alignment among parallel texts, and primary data (XCES, 2004).

The elements of XCES have been organized in a hierarchically specified structural mode, beginning at the most abstract level and then defining derived types for general classes of annotation, such as speech, discourse etc. At the lowest level of the hierarchy, precise annotation values are specified. Also users can create their own variant types for existing ones by defining new derived or extended types.

XCES is similar to TEI and GDA, which offers plenty of tags to comment a document. XCES basically highlights the structural and typographic information as well as general as the linguistic annotation. The applications of XCES are close to GDA including natural language processing, machine translation and lexicography.

*Summary*

Table 4.2 shows a summary of three internal metadata annotation standards.

	Number of Tags	Multiple level tags	User defined tags
TEI	Large	Yes	Yes
GDA	Medium	Yes	No
XCES	Medium	Yes	Yes

Table 4 - 2 summary of three internal metadata annotation standards.

According to the review of previous document annotation standards: TEI, GDA and XCES, all of these three standards have many common features which are summarised to the following points:

- Support of a broad range of annotation types for language data;
- Multiple annotation levels, where the various annotation levels can be related to each other;
- Open with respect to the information levels and categories within each level;
- Co-existence of a multitude of coding schemes and standards;
- Easily modifiable XML support for a broad range of annotation types.

Also looking at these standards in detail, all of the three standards define the structural annotation elements such as <header>, <p> and annotation elements for the special phrases in text such as <date>, <person>, <organization> and annotation elements for the decoration and alignment. Some elements appear in all of three standards with the exactly same name such as <dv>, <p>, <date>. Compared with the other two standards, TEI offers a very rich element set which covers lots of areas about a text document and potentially is a common standard for document annotation metadata. GDA and XCES more emphasize the syntactic structure of a text, which has been highly used in nature language processing. The development of the story domain independent metadata will be based on these standards. But all of these standards have not provided any metadata tags which can annotate the document from learning and teaching context.

#### **4.3.1.2 Instructional Strategies**

The original story concept suggested including instructional information in the story narrative. The narrative of a story presents the discussion and interpretation of the knowledge contained in the learning objects. The advantage of adding instructional information to the story narrative is that a story can offer readers pedagogical and educational background of each story section and direct readers to appropriate section

within a story. Instructional strategy is a kind of the learning and teaching strategy which consists of knowledge structure, presentation, exploration, practice and learner guide (ID2 Research Group, 1997).

Gagne's nine events instruction, Ausubel's expository teaching and problem based learning are famous and fundamental instructional strategies around the world, which have been widely used in different applications.

### *Gagne's nine events of instruction*

Gagne's nine events of instruction (Gagne, Briggs, & Wager, 1985) is an instructional strategy that sequences and chunks the educational material. Gagne identifies the different stages of the learning process according to students' mental conditions and creates a nine step process which correlates to and addresses the conditions of learning (Gagne, Briggs, & Wager, 1985). Following are the nine learning stages involved in Gagne's nine events of instruction.

- *Gain attention.* Capturing the attention of the learner.
- *Describe the goal.* Informing the learner of the outcomes or objectives.
- *Stimulate recall of prior knowledge.* Associating new information with prior knowledge can facilitate the learning process.
- *Present the material to be learned.* Actually presenting new content to the learner.
- *Provide guidance for learning.* To help learners encode information for long-term storage. An additional guidance should be provided along with the presentation of new content.
- *Elicit performance.* The learner is required to practice the new skill or knowledge. Eliciting performance provides an opportunity for learners to confirm their correct understanding.

- *Provide informative feedback.* An important way to assess and facilitate learning is to provide immediate feedback after practice as to how well the learner performed.
- *Assess performance.* Upon completing instructional modules, learners should be given the opportunity to take a post-test or final assessment.
- *Enhance retention and transfer to the job.* This event informs learners of similar problems, provides additional practice.

### *Ausubel's expository teaching*

Ausubel's theory is a cognitive learning strategy. Ausubel's theory deals with how learners learn large amounts of meaningful materials from textual presentations. According to Ausubel, learning is well organized by representational and combinatorial processes that occur during the presentation of information (Ausubel, 1984).

Ausubel's learning sequence consists of four major learning phases:

- *Advance organizer.* Prepare for integration of new knowledge and build the bridge between new learning material and existing related ideas. Present introductory material that helps learners relate new information to existing knowledge schema.
- *Progressive Differentiation.* Present the most general ideas of the subject and then progressively introduce new knowledge by comparing details and specifics.
- *Practice.* Practice and apply the new knowledge.
- *Integrating and connecting.* Integrate and link new knowledge to other fields of knowledge and context areas as well as to the advance organizer. Instructional materials should attempt to integrate new material with previously presented information through comparisons and cross-referencing of new and old.

### *Problem Based Learning*

Problem based learning (PBL) is a strategy for instructional development and delivery that is becoming increasingly popular in the learning environment (Boud & Feletti, 1997). PBL involves the presentation of a problem to learners at the beginning of the learning process. Learners then learn the content in an attempt to solve the problem.

There are seven phases in PBL (Allert, Dhraief & Nejd, 2002).

- *Goal Description.* Present problem to be solved and set ultimate goal.
- *Specify Criteria.* Specify one or more criteria that the solution should meet.
- *Background knowledge.* Identify knowledge needed.
- *Generate ideas.* Generate ideas and draft provisional hypotheses.
- *Implement solution.* Generate and develop solutions; implement and compare different solutions
- *Reflect.* Evaluate solutions, reflect on solutions, reflect on product, and reflect on process.
- *Generalize.* Conceptualize, integrate, and generalize learner's knowledge. Move from example to theory.

Instructional information will be added to the story narrative as part of the story independent metadata. By labeling story narrative sections with instructional strategy information, story readers are able to find out the educational background of each section in the story narrative and select suitable story narrative sections.

Gagne suggests that learning tasks should be organized according to complexity: recognition, response generation, procedure following, use of terminology, discriminations, concept formation, rule application, and problem solving. Also Gagne identifies prerequisite knowledge and learning outcome of each learning stage. But Gagne's instructional strategy asks for too many interactions between learner and instructor, so it is not suitable to be adopted in the story narrative.



The Ausubel's expository teaching has the good structure. Ausubel's expository teaching is most commonly used in curriculum components where the primary objective is standardized facts, concepts, rules and procedures (Ausubel, 1984). It is also used in cases when the learning resources are too complex to be understood by the learners. Teachers need to subdivide, translate, and structure the information into more digestible form for learners, and introduce the knowledge in a logical and systematic manner. From the story point of view Ausubel's expository teaching is too brief to describe the learning process. The learner cannot locate each part of the story narrative accurately and precisely by Ausubel's strategy. Further, Ausubel's expository approach usually involves a high degree of teacher-directedness which is not appropriate in the story concept.

In comparison with the above two instructional strategies, Problem Based Learning puts the emphasis on motivating and activating learners to acquire knowledge by solving the problems. Problem Based Learning assists learners in solving problems by the process of continually encountering the type of problems. In the story context, Problem Based Learning is good to describe the story narrative, because it clearly states the task and the objectives of each learning step and directs the learner towards solving problems from easy problems to complex problems. Furthermore, the learning process of Problem Based Learning is quite close to our natural learning process.

#### **4.3.1.3 Domain Independent Metadata Elements**

Before looking at the story domain independent metadata elements, it is necessary to determine how to present metadata. XML is an ideal medium for representing metadata because it can be understood by many different applications and systems. Furthermore metadata should support many functional requirements, such as electronic libraries, application integration, and web resource discovery (Ahmed et al., 2001). Also metadata are needed in many different system environments, such as local networks, large

modular corporate intranets, and the WWW. XML can offer flexible and practical solutions for different metadata requirements. XML has a lot of advantages and becomes more and more popular for presenting metadata, so XML has been adopted to represent the story metadata. Generally a story can be seen as a standard XML file which annotates the narrative with metadata tags. Because both narrative and metadata tags are stored in an XML file, this can simplify the implementation of the story environment tools to just write story narrative and metadata tags to a XML file and retrieve it completely later on.

According to the previous description, the story domain independent metadata should mark up the important features of a story narrative including structure, special terms, link to learning objects and references. Thus the story domain independent metadata will be further divided to 4 sub-categories which are structural metadata, phrasal metadata, multimedia link metadata and reference metadata.

### *Structural metadata*

Structural metadata are basically in charge of presenting the story narrative structure. The structure of the story narrative is annotated by structural metadata. All of TEI and GDA and XCES have their own hierarchical elements to mark up a text document. Based on those existing elements, It is decided that story metadata have maximum seven levels. Each level has its own corresponding elements.

#### ➤ Level 1

TEI's level 1 elements are <front>, <group>, <body>, <back> (TEI U5, 2004). It is obvious that these elements are not very meaningful for a reader. GDA and XCES do not have elements at that level. The story project defines its own level 1 elements. A story narrative can be roughly divided into five parts which are abstract, introduction, main body, summary and reference. Each of these five parts has a relevant element in

the structural metadata. There are no attributes related to these elements. The elements are optional except for main body. As default, the whole story narrative is recognized as a <mainbody> element.

Elements: <Abstract>, <Introduction>, <Mainbody>, <Summary>, <Reference>.

### ➤ Level 2

One or more paragraphs can be joined to be a division. The Division element is the most common element in current document annotation standards. In the story domain independent metadata, an element <dv> is used to annotate a division under the level 1 metadata elements. For example a <Abstract> can have one or many divisions there. If a user does not specify the division, a whole level 1 part will be recognized as a division. An attribute, called ‘description’, is associated with the <dv> element, which allows story authors to write a short description or interpretation for a specific division.

Elements: <dv description=”...”>

### ➤ Level 3

The level after division level is supposed to be the paragraph level. The definition of the paragraph is a simple paragraph in the story narrative. Paragraph is a popular element in many document annotation standards as well. Here two new attributes have been attached to a paragraph level element.

Elements: <p language=”...” phase=”...”>

- ✓ Attribute ‘language’ indicates the language of that specific paragraph.
- ✓ Attribute ‘phase’ stores instructional information concerning problem based learning stages of each paragraph. In the last section Problem Based Learning (PBL) is identified to be suitable for the story context. All of the six PBL phases can be reused in a story. A paragraph ‘Goal Description’ can explicitly express the

objective or learning outcome of a story. A paragraph ‘Specify Criteria’ shows the purpose or topic of a story. ‘Background knowledge’ paragraph basically talks about the background knowledge of a story and also the pre-requirements of reading a story. ‘Generated idea’ paragraph gives an overview or outline of a story. The content or body of story can be marked as ‘Implementation Solution’ paragraph. A ‘Reflect’ paragraph discusses the ideas presented in a story. The paragraph with phase ‘Generalize’ can be a summary paragraph which summarizes the whole story. Only one of six PBL phases can be applied to each paragraph in the story narrative. Table 4.3 describes each PBL phase with original purpose and the purpose in story context.

<b>Phase</b>	<b>Instructional purpose</b>	<b>Story context</b>
Goal Description	Present problem to be solved. Set ultimate Goal	Story objective
Specify Criteria	Specify one or more criteria your solution should meet. What aspects do you want to focus on? How do you know your reached goal?	The purpose and/or topics of the story
Background knowledge	Identify knowledge needed. Sample and share knowledge. Ask experts.	Background knowledge
Generate ideas	Generate ideas. Draft provisional hypotheses.	Outline the story
Implement solution	Generate and develop solution. Implement, compare different solutions	Detail of a story content
Reflect	Evaluate solution, reflect solution, reflect product, and reflect process.	The discussion of the concepts presented in the story
Generalize	Conceptualize integrate and generalize your knowledge. Move from example to theory	Summary

Table 4 - 3 PBL phases in the story context

➤ Level 4

It is reasonable to have a level after paragraph. In TEI, the level after paragraph is <l> which is defined as a single, possibly incomplete, line of verse. In GDA, <s> is the sentence level metadata element for commenting a sentence. In XCES, the element paragraph is also defined to mark a sentence. A story is needed to have a sentence level element since the level 5 phrasal metadata introduced in the next section are defined to annotate a small phrase in a sentence. If there is no sentence level element, there will be lots of sub-elements under paragraph level. Also a phrase can only be in a sentence. By defining sentence level elements, it can simply force story authors to avoid defining a phrase across sentences. This issue will be covered in detail in the implementation part.

Elements: <s>

### *Phrasal metadata*

Phrasal metadata annotate a phrase in a sentence, which contains special terms. Those most valuable and accepted phrasal level elements in TEI, GDA and XCES are picked to create the phrasal metadata in the domain independent metadata set. The hierarchical position of the phrasal metadata is level 5 that is under a sentence element. All the phrasal metadata elements are under the structural element <phrase>. It is important to notice that the domain specific metadata elements defined by story authors will be inserted under the element <phrase> as well.

The phrasal metadata elements include:

- <date> — Date. Example <date>10 July</date>.
- <time> — Time. Example <time>10:20</time>.
- <abbr fullName='...'> — Abbreviation. Example <abbr fullName='By The Way'>BTW</abbr>
- <num type='...' value='...'> — Number. The attribute 'type' stands for the type of the number which can be either decimal, float, or integer. The attribute 'value' is the

numeric representation of that number. For example `<num type='integer' value='23'>twenty three</num>`

- `<persname language='... '>` — Person name. The attribute 'Language' is the language of that person name. For example `<persname>John Brush</persname>`.
- `<orgname language='... '>` — Organization name. For example `<orgname>New Zealand Immigration Service</orgname>`.
- `<address language='... '>` — Address. For example `<address>124 PO box, Wellington</address>`.
- `<bibref language='... '>` — Bibliography reference. For example `<bibref>(Wood, 2004)</bibref>`.
- `<geoname language='... '>` — Geography name. For example `<geoname>Cook mountain</geoname>`.
- `<Link>` - Used to create a link to a learning object.

### *Multimedia cross reference metadata*

Multimedia reference metadata specify links in a story. Basic information such as the URI to an external learning document is included in multimedia reference metadata. Since some learning documents are extremely large, multimedia reference metadata should be able to locate a small segment inside a large learning document. For example multimedia reference metadata can point to pages 10–12 in a novel of 100 pages length. Such information is recorded in multimedia reference metadata by sub-elements of start position, end position and duration or length. The level of multimedia reference metadata is same as phrasal metadata which is level 5 and under the `<phrase>` element. In principle, multimedia cross reference metadata cannot be used separately. It has to be appended to any phrasal metadata, so that any phrasal metadata may have a link to external documents. Figure 4.1 is an example of using multimedia reference metadata to create a link for a person name.

```
<phrase>
<pername>...</persname>
<MediaLocator>.....</MediaLocator>
</phrase>
```

Figure 4 - 1 use of multimedia cross reference metadata

The top element of multimedia reference metadata is `<mediaLocator>`. Under `<mediaLocator>`, it includes:

➤ `<mediaUrl>` — the URL of a link

➤ For a textual link, the multimedia cross reference metadata element is `<textPos>`.

```
<textPos>
```

```
  <unit>      — the unit of the text file.
```

```
  <start>     — record the start position of in a text link
```

```
  <length>   — include the length of a segment.
```

```
</textPos>
```

➤ For an audio/visual link, the multimedia cross reference metadata element is `<mediaTime>`.

```
<mediaTime>
```

```
  <startTime> — include the start time of the audio/video file.
```

```
  <duration>  — include the duration of a media segment.
```

```
</mediaTime>
```

*Reference metadata*

References are especially helpful for a story reader to find the original resources of a story. It can also expand the background knowledge of a story and further direct to more resources. Reference metadata is crucial, so it can be found in almost every document annotation metadata standard. The story domain independent metadata have also created reference metadata for such a facility. Story reference metadata is at second level in the metadata hierarchical tree, which is just under the level 1 element <reference>. According to the APA (APA style, 2004) reference style, each piece of reference can be recorded within the following elements. Each <ref> element represents a reference.

<ref>

<author> — the author of a reference

<year> — the date the reference has been published

<title> — the title of the reference

<publication> — the publication details of a reference including publisher, page number and place of publication, etc.

</ref>



Figure 4.2 shows a hierarchical tree of the story domain independent metadata

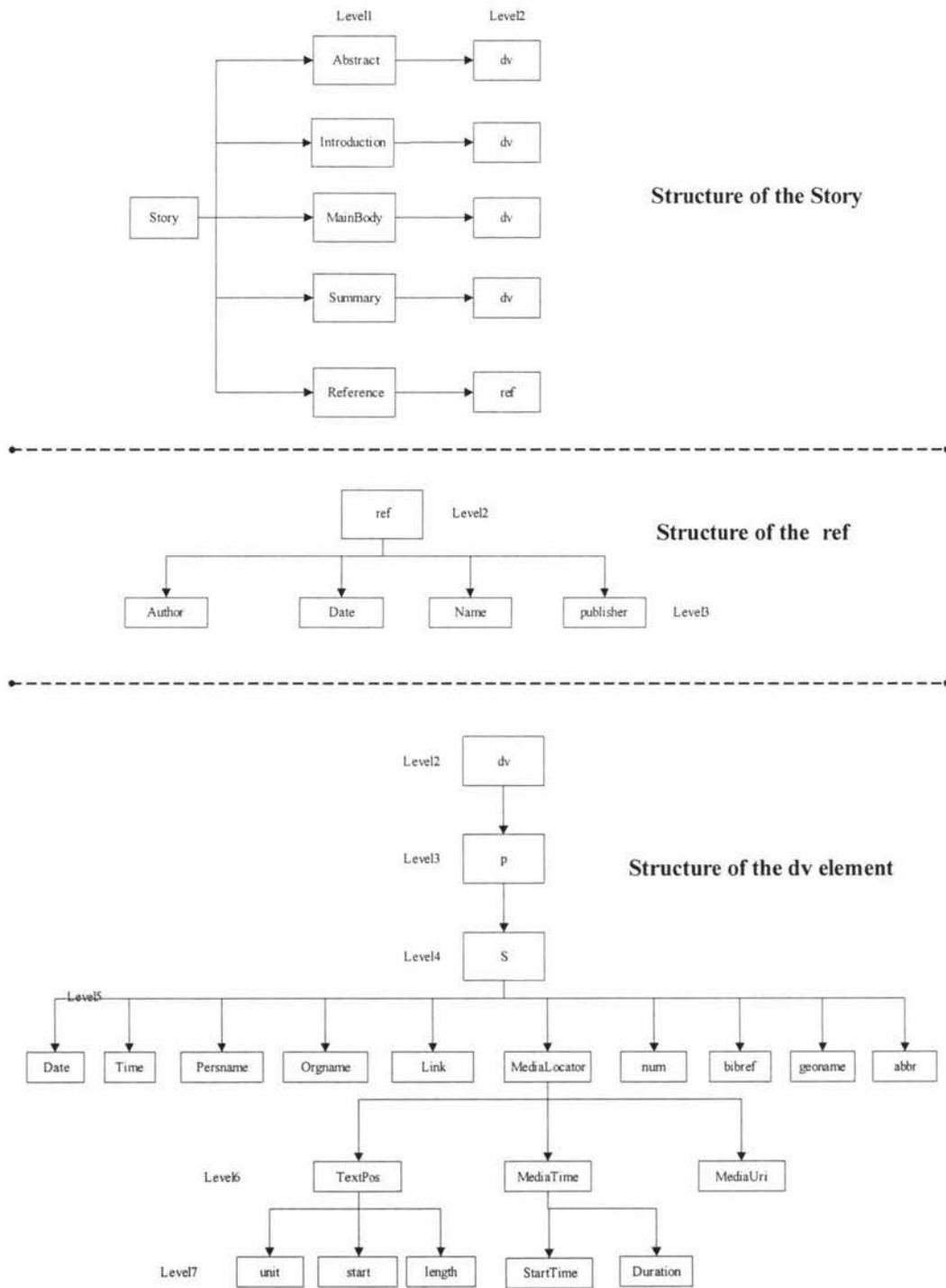


Figure 4 - 2 hierarchical tree of the story domain independent metadata

### 4.3.2 Domain Specific Metadata

Domain specific metadata are designed for story authors to define their own metadata definition related to their specific subjects. Currently domain specific metadata only allow story authors to define a single level metadata definition, which means story authors can only have one level of domain specific metadata elements. As mentioned above, the hierarchical position of domain specific metadata is in level 5 which is the same as phrasal metadata. All the domain specific metadata will be put under the sentence tag <s>. Therefore domain specific metadata are only able to annotate the phrases in a sentence under the current specification.

The following gives an example of the domain specific metadata elements about culture heritage. These elements can be used to describe culture heritage categories, the law about cultural heritage and actions related to cultural heritage. The elements include:

<MovableHeritage>	— Moveable cultural heritage
<ImmovableHerigate>	— Immovable cultural heritage
<IntangibleHeritage>	— Intangible cultural heritage
<CommonHeritage>	— Common cultural heritage
<NationalHeritage>	— National cultural heritage
<HeritageLaw>	— Cultural heritage law
<HeritageAppropriation>	— Heritage appropriation
<HeritageRepatriation>	— Heritage repatriation
<HeritageStakeholder>	— Heritage Stakeholder

### 4.3.3 Example of a Story Annotation

In this section a part of a story is shown. Figure 4.3 is an annotated example from a cultural heritage story in XML format. This story basically outlines the different types of cultural heritages and specifies actions related to cultural heritage. In Figure 4.3 different types of the internal story metadata have been highlighted.

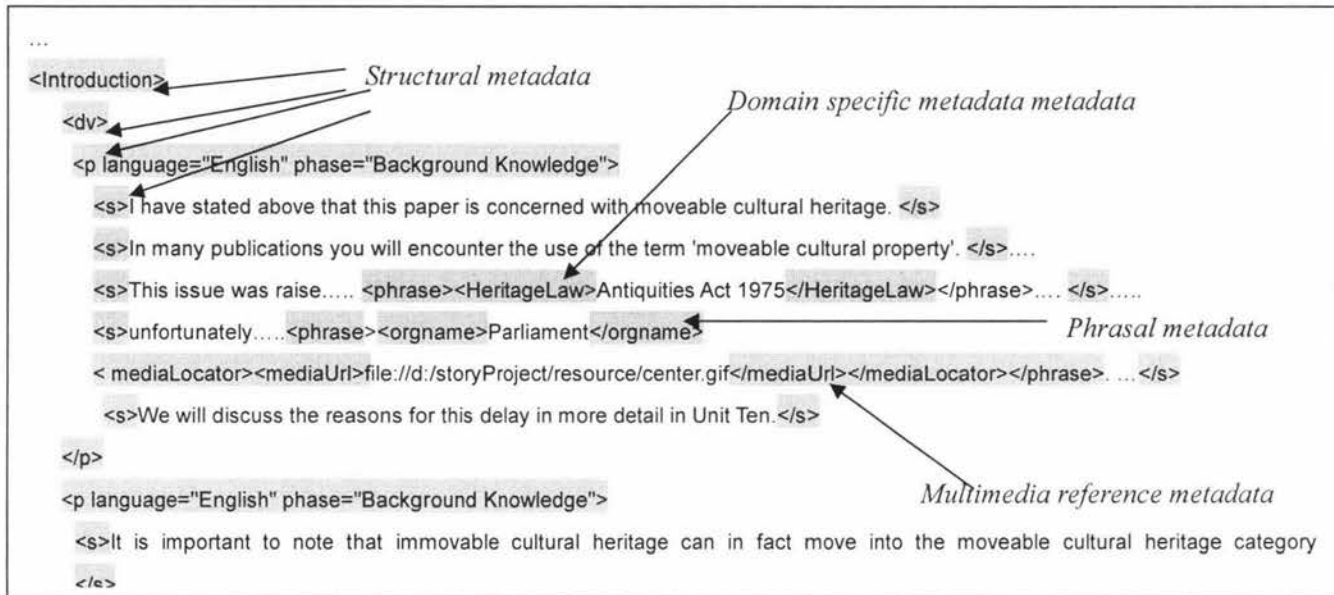


Figure 4 - 3 an annotated example from the cultural heritage story in XML format

## 4.4 Summary

In this chapter, both external story metadata and internal story metadata have been specified and designed. Figure 4.4 shows the overall structure of story metadata and the hierarchical position of each story metadata type.

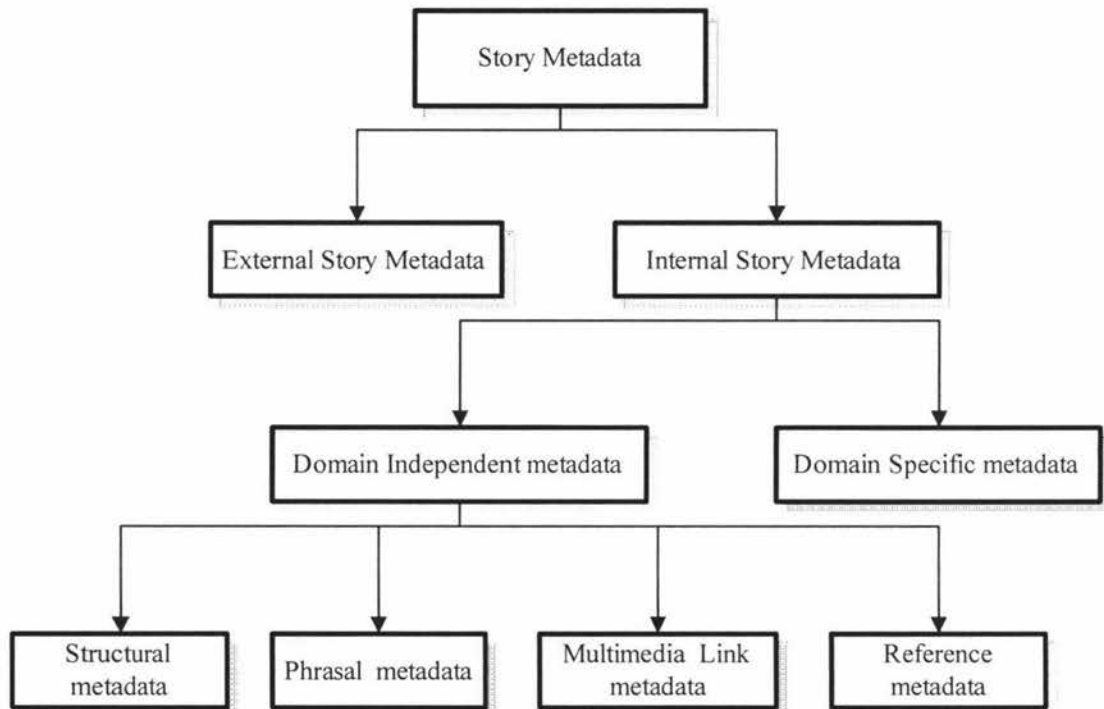


Figure 4 - 4 hierarchical position of each story metadata type

Although this chapter has not designed the specific metadata elements of the external story metadata, this chapter reviewed current external metadata standards such as LOM and Dublin Core that make design of the external story metadata relatively simple by directly using the current standards or reusing elements of the current standards.

The story domain independent metadata specification has been outlined in this chapter. Domain independent metadata include the common features of stories, so that these can be reused by every single story. Based on the requirements of story domain independent

metadata, four different categories of domain independent metadata have been created:

- Structural metadata - The structure of the story narrative marked by structural metadata.
- Phrasal metadata - Phrasal metadata annotate a special phrase in a sentence such as date, person name and address etc. Also domain specific metadata created by story authors are fitted into this category.
- Multimedia cross reference metadata - Multimedia reference metadata annotate a link to an external learning document.
- Reference metadata - Reference metadata are especially important for a reader to find the references in a story.

Story domain specific metadata are defined by story authors. So it is difficult to decide domain specific metadata in this specification. In principle, story authors are allowed to define their own domain specific metadata tags in a single level.

In the next chapter, the implementation of the story applications will be introduced which includes how to load XML metadata tag definitions to story applications, annotate story narratives with story metadata tags and retrieve story narratives and metadata from a story.

## Chapter 5 Story Environment Tools Design

---

According to the specification of the story environment, the implementation of the story environment includes:

- Story authoring tool – create stories;
- Story browser – browse stories;
- Story repository management system – manage stories in the story repositories;
- Story repository – store stories with their metadata.

Due to time constraints, the development of the story repository management system and the story repository are not included in this project. In this chapter, the implementation of the story authoring tool and the story browser is introduced, based on the explanations in the conceptual design chapter. Each tool will be discussed with requirements analysis, technologies, implementation issues and interface design.

### 5.1 Story Authoring Tool

#### 5.1.1 Requirements Analysis

The story authoring process requests the story environment to provide story authors an easy-to-use tool that allows them to compile a story focusing on their subject matter and not the technical details of metadata. The requirements of a story authoring tool can be summarized to assist story users in editing the story narrative, creating links to learning objects, annotating units in the narrative and finally generating a story XML file.

Currently, there are many XML tools available, which are able to help users to create XML documents. It is very useful to look at some other applications that potentially could be used in the story environment as the story authoring tool.

In principle, an XML document can be edited by any text editor such as Notepad, WordPad or Word. Since story authors are not computer experts, it is not realistic to expect them to write XML directly by using a text editor. Therefore, some other applications have been developed to facilitate creation of XML files.

XML spy (XML spy, 2004) offers users a data grid which uses each metadata tag as a grid head. Users can enter metadata under each metadata tag head. XML spy can automatically generate XML files for users. Figure 5.1 shows a screen shot of a XML data input grid.

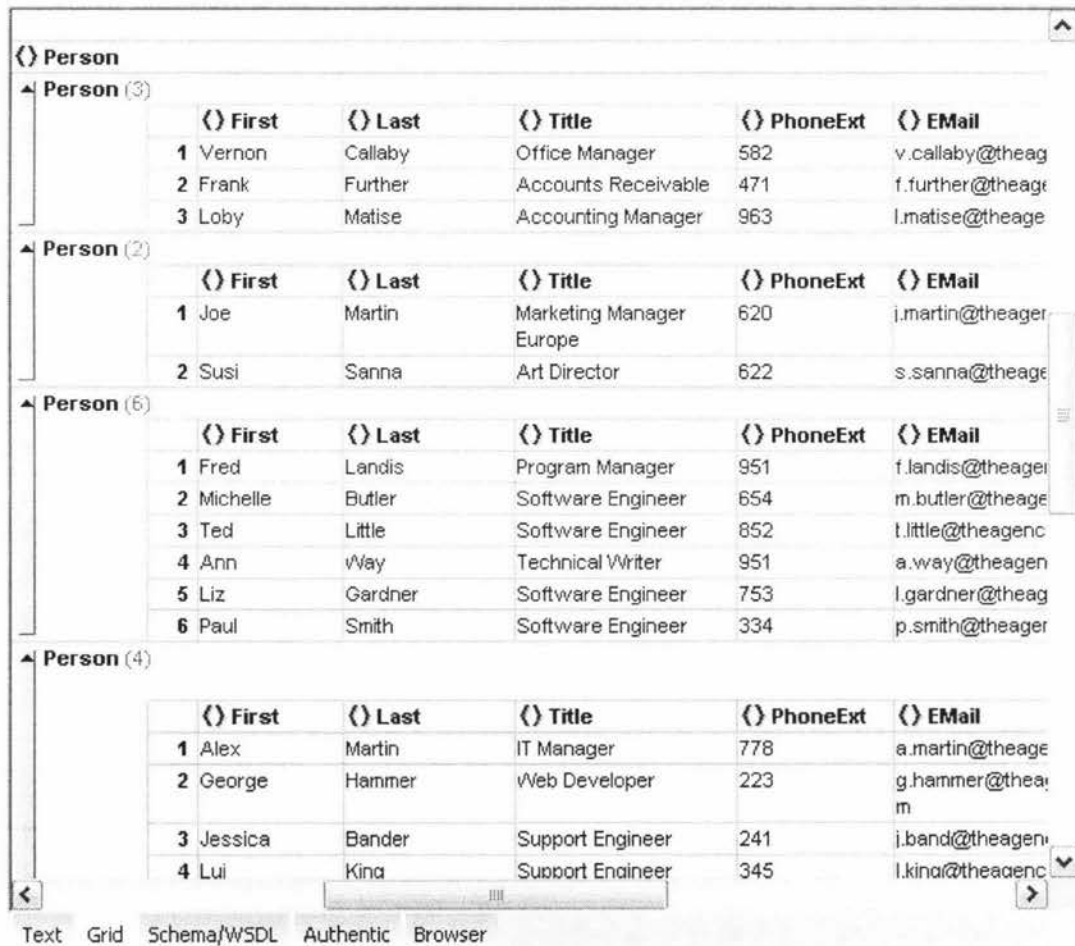


Figure 5 - 1 XML spy data input grid.

CookTop(Cooktop, 2004) is a freeware tool for XML editing. CookTop does not have a data grid or table to facilitate users inputting metadata. But CookTop has some auto-features to assist users editing XML documents quickly and efficiently. For example, CookTop can check the syntactic correctness of the current XML documents against its XML schema definition. Figure 5.2 shows a XML file editing window from CookTop.

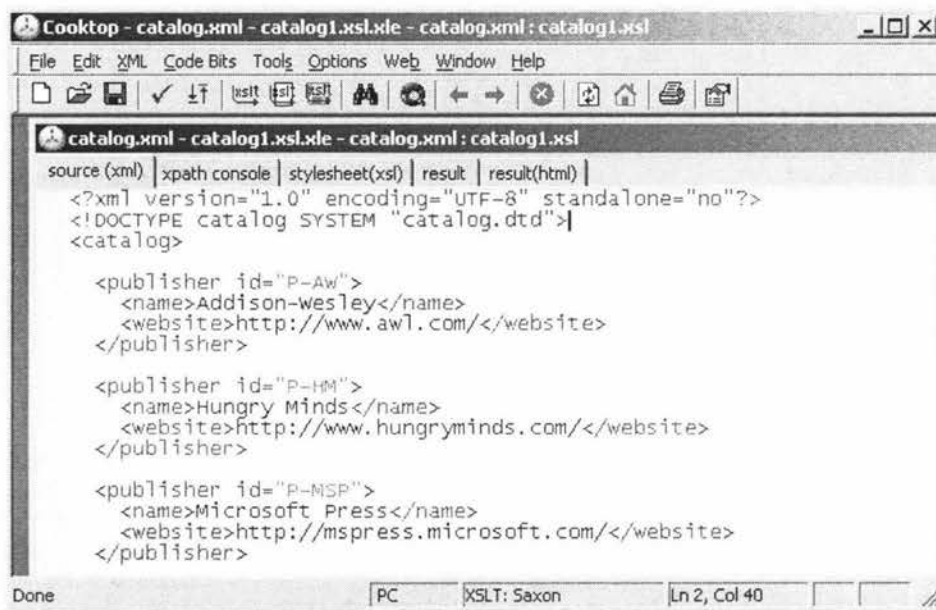


Figure 5 - 2 XML editing window of cook top

Both of the above tools definitely help users to create XML documents at a certain level, but they are not quite suitable in the story environment. In the story environment story authors are supposed to mark up the story narrative with the internal story metadata when they are actually writing the story narrative. That means story authors should be able to enter data first then decide metadata tag for that piece of data. The way of XML spy and CookTop to edit metadata is to enter data under each metadata tag. It is an obvious difficulty for story authors to create a story by using current XML tools. In addition, when story authors write stories, they should be able to concentrate on content (story narrative) and not technical metadata issues. The actual XML metadata should be



hidden behind the story narrative. Story authors only need to focus on the story narrative to be annotated. Therefore, in story the story environment a new tool is required to allow story authors to compile stories.

In this project the story authoring tool is supposed to assist story authors in editing the story narrative, creating links to learning objects, annotating text units in the story narrative and finally generating story XML files. The following are functions of the story authoring tool that have to be considered during the implementation.

- *Edit the story narrative* – One major requirement of the story authoring tool is to allow story authors to edit the story narrative. Story authors need to be given an opportunity to focus on the textual narrative without worrying about the metadata tags behind the narrative.
- *Read internal XML schema* – There are two different types of internal story metadata. Domain independent metadata definitions should be built into the story authoring tool. Domain specific metadata definitions are specified by authors of stories. The story authoring tool needs to import the story domain specific metadata definition, so that story authors can use these metadata tags to comment on their narratives.
- *Mark the story narrative with metadata tags* – The story authoring tool needs to provide story authors with a visual interface for marking up the story narrative. Where possible, such as for metadata tags <s> and <p> the story authoring tool should automatically insert these tags by recognizing a sentence or paragraph delimiters.
- *File storage* – The story authoring tool needs to be able to save the author's current work, which includes the narrative and metadata annotations.
- *Generate story XML file & preview story* – the final step of the story creating process is to generate a story XML file. The preview function should be available to facilitate story authors to preview their stories at any time.

### 5.1.2 Technology Issues

This section looks at some of underlying technologies that have been used to implement the story authoring tool and asks the questions why these were chosen for satisfying the requirements.

In this project the story authoring tool is implemented as a stand-alone application to allow story authors to create stories on their local computers. At this stage, the story repository and story repository management system have not been implemented. The story authoring tool does not contact or communicate with any other applications in the story environment. Another reason of implementing it as a stand-alone application is to make the implementation simple and extensible. In future, another module can be added to the story authoring tool to connect to the story repository management system to update stories and specify their external metadata.

The story authoring tool is written in the Java programming language. The Java programming language is a general-purpose, concurrent, class-based, object oriented language (Gosling, Joy, Steele & Bracha, 2005). More and more applications and systems are being developed in Java because of its cross-platform abilities. Java also has a number of APIs for the developer to cooperate with other technologies, for example, databases, HTML, XML, TCP/IP and mobile information technology.

As a popular programming language, Java also has the following advantages (Java, 2004).

*Simple.* The syntax of Java is very close to other popular programming languages. Also, Java adds automatic garbage collection which can increase the performance of an application. In Java memory is automatically allocated when new dynamic structures (i.e. objects) are created and automatically disposed of by garbage collection when not needed.

*Object Oriented.* Java is an object-oriented language. Object-oriented design focuses on the data or object design and the interface to it. The object-oriented ability of Java is essentially for the high performance and structural program.

*Interpreted and compiled.* Rather than being compiled directly into machine code, normally Java is compiled into an intermediate representation called Java Byte Code, which is then interpreted by the Java Virtual Machine. The intermediate byte code provides cross platform ability to Java.

*Multithreaded.* Unlike conventional programming language, Java has a sophisticated set of synchronization primitives that are based on the widely used monitor and condition variable paradigms. By integrating these concepts into the language, a multithreaded program can be implemented very easily.

The main reason of using Java to implement the story authoring tool is for its cross-platform capabilities. A Java program can be executed on all popular platforms such as Windows, Mac OS and OS2. As mentioned in the last section, XML has cross-platform ability. For the story authoring tool it is important to have cross-platform ability too. Story authors should not be restricted to one platform only. The story authoring tool should enable story authors to create stories under most popular platforms.

Another reason for choosing Java is because of its XML API. One of the requirements of the story authoring tool is to load the XML metadata definition (XML schema) defined by story authors and use those metadata tags to mark up the story narrative. In supporting this function, story authoring tool should have a good parser to parse the XML schema and extract metadata definitions. Java includes a wide range of XML parse APIs based on either SAX (Harold, 2001) or DOM (DOM, 2004). Thus it makes the implementation of parsing an XML schema much easier.

### 5.1.3 Implementation Issues

This section concentrates on the implementation issues of the story authoring tool. The basic functions of the story authoring tool have been summarized as editing the story narrative, creating links to learning objects, annotating the story narrative and finally generating the story XML file. This section will illustrate the implementation of the story authoring tool by going through the processes of story authoring as shown in Figure 5.3.

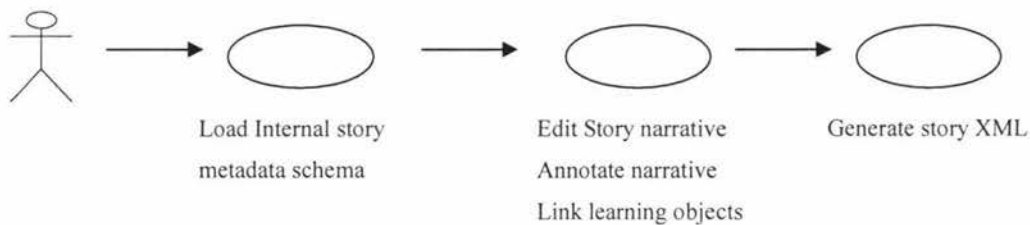


Figure 5 - 3 process of story authoring

#### 5.1.3.1. Load Internal Story Metadata Schema

Before story authors start to edit a story, they need to load the story metadata specifications for a story first. The metadata tags in the story metadata specifications will be loaded into the story authoring tool and be used later on to mark up the story narrative. There are two different types of the story metadata that need to be loaded. Domain independent metadata are generic and identical for all stories and should be built into the story authoring tool. The story authoring tool will not ask story authors to provide a XML schema for domain interdependent metadata. For domain specific metadata the story authoring tool wants the story author to supply a XML schema for his domain specific metadata specification. The domain specific metadata XML schema will be parsed by the story authoring tool to retrieve the metadata tags and its attributes

defined in the schema. Figure 5.4 demonstrates the process of loading the story XML metadata schema.

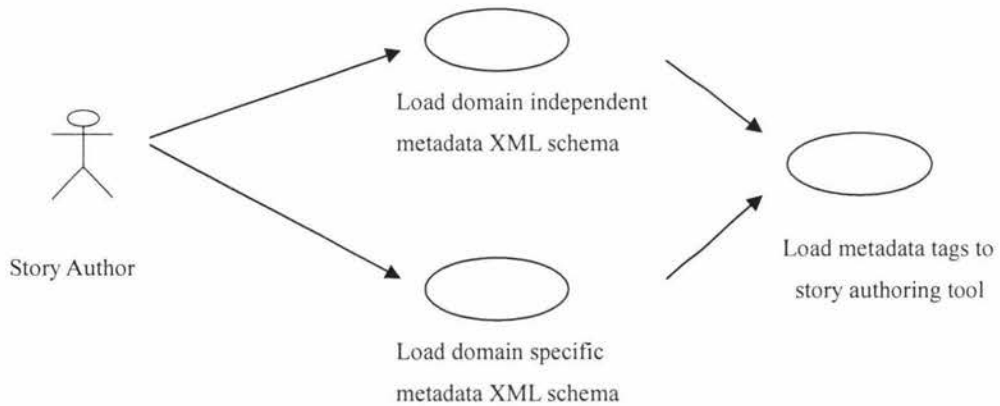


Figure 5 - 4 process of loading XML schema.

The implementation of loading story XML metadata schema needs to load both the story independent metadata and story domain specific metadata XML schema. Although a XML schema only includes tags definition of a XML file, a XML schema is a valid XML file as well. Any valid XML schema file can be parsed by XML parsers too. Because the story authoring tool is implemented by Java, Java DOM parser (Sall, 2002) is selected as a XML parser to retrieve XML metadata tags from a XML schema. Compared with other Java XML parsers such as SAX, JDOM are is very good at processing big and complicated XML files. Also the JDOM API is very easy to use. The details of comparing XML parsers will be explained in the story browser section(5.2.2). After loading a XML schema, JDOM generates an object tree which includes all the metadata tags and their attributes as defined in the XML schema. Then the story authoring tool travels through the JDOM tree from the top level to the bottom level and extracts the metadata tags and their attributes. In the story authoring tool, those retrieved metadata tags and attributes are saved in an object list named metadata definition list. Each object in the list represents a metadata tag definition. A metadata tag object consists of the name of the metadata tag, the level of the metadata tag and an attributes

list which contains all the attributes' names related to that specific metadata tag. Initially there are two lists in the story authoring tool. One is for the domain independent metadata definition, and another is for the domain specific metadata definition. Figure 5.5 shows the structure of metadata definition list.

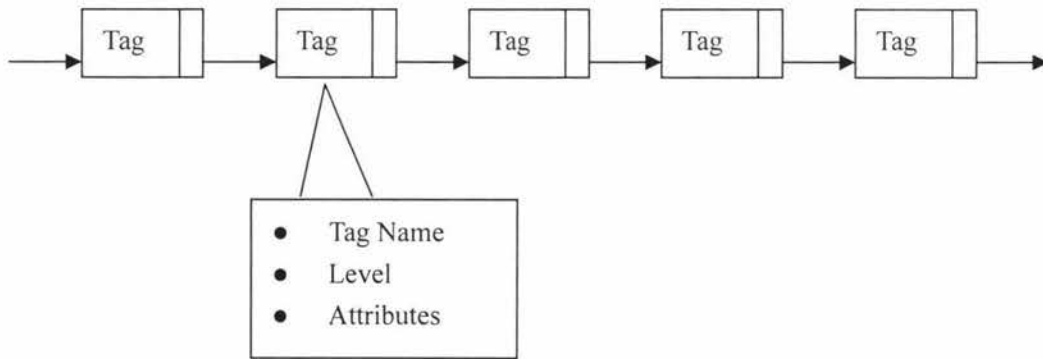


Figure 5 - 5 structure of the metadata definition list.

### 5.1.3.2 Story Editing

This section describes the major functions of editing stories. Following are some specific requirements of the story editing.

- Edit story narrative
  - 1) Store and update users' narrative. The story authoring tool should save and update the story narrative at any time.
  - 2) Maintain and update metadata information. Whenever story authors change the story narrative, the previous metadata annotations should be updated and maintained as well.
  
- Narrative mark up/annotate
  - 1) Create a new metadata annotation on any narrative text.
  - 2) Delete the existing story narrative annotations.

- 3) View the existing story narrative annotations such as the metadata attributes behind each metadata tag.

➤ Record metadata annotation

The idea of storing a story narrative is very simple. The story narrative is treated as a long string in the story authoring tool. However, how to record users' metadata annotations is difficult. In a story each annotation is expressed by metadata which are related to a segment of the story narrative. The possibly easiest way to record a metadata annotation is to save the start and end position of the annotation text in the story narrative, the metadata tag name and its attributes values. This information is saved as an object in a list. For example, in the following paragraph users want to mark up the text '(Linsay, 1988)' with the metadata tag <Reference>.

I have stated above that this paper is concerned with moveable cultural heritage (*Linsay,1988*). In many publications you will encounter the use of the term 'moveable cultural property.....

The metadata tag <Reference> has two properties named author and year. In this case author is 'Linsay' and year is '1988'. The start position of the text '(Linsay, 1988)' in paragraph is at character 80 and it finishes at character 94. So the following information will be saved by the story authoring tool:

Metadata tag Name: Reference:  
Start position: 80  
End Position: 94  
Attributes/Property:  
Name: Author      Value: Linsay  
Name:Year        Value :1988

In the story authoring tool the object which saves information about a metadata annotation is called *metadata annotation element*. The list including metadata annotation elements is called metadata annotation list. Figure 5.6 shows the structure of the metadata annotation list.

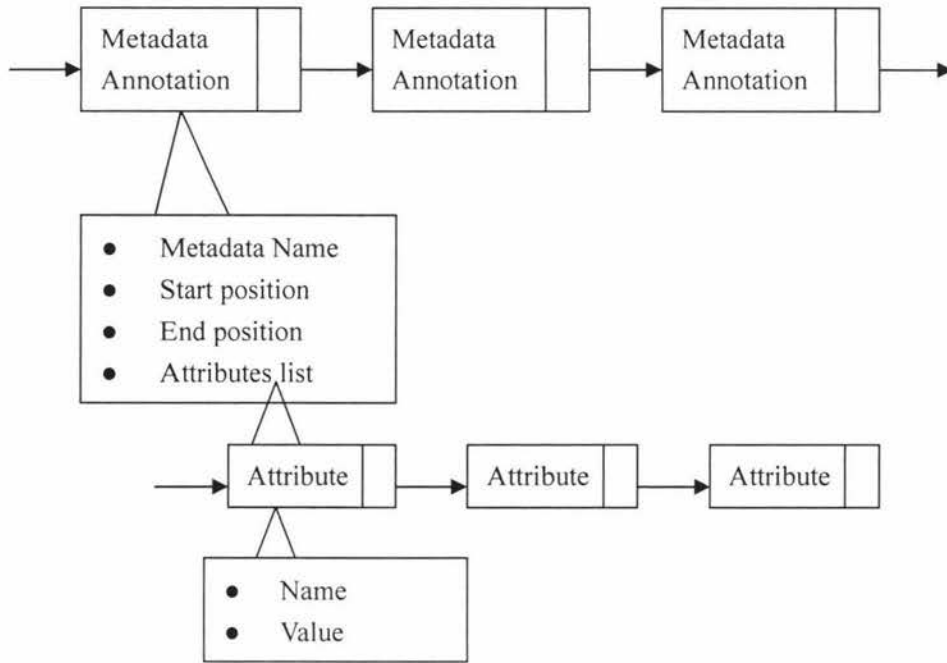


Figure 5 - 6 structure of the metadata annotation list

➤ Update and maintenance of story metadata annotations

*Create a new metadata annotation*

First of all story authors need to highlight the narrative text they want to mark up. Then they need to specify the metadata information for the selected text. As the story metadata specifications have already been loaded to the story authoring tool before story authors edit a story, specifying the metadata information can be done by selecting the suitable tags from the preloaded metadata tag list and filling the attributes values for the specific metadata tags. When story authors finish a mark up, a new metadata annotation element will be added to the metadata annotation list.



*Delete metadata annotations*

Story authors need to select the annotated text that they want to remove. The story authoring tool will search the metadata annotations from the metadata annotation list based on the position of the selected text. When a metadata annotation has been found, it will be removed for the metadata annotation list.

*Retrieve metadata annotations*

This function helps story authors to review a metadata annotation. Story authors select the annotated text which has been annotated previously. The story authoring tool will find the metadata annotation from the metadata annotation list by matching the annotated text and its position. Then the story authoring tool retrieve metadata tag name and its properties values.

*Maintain current metadata annotations*

In the story authoring tool each story metadata annotation will include information about the specific position of an annotated text in the story narrative, whenever story authors change the story narrative, the metadata annotation recorded previously by the story authoring tool should be updated as well. For example, if story authors enter a new character into the story narrative, every metadata annotation marked after that new character's position should be updated by incrementing the start and end position. Following the same principle, if story authors delete a paragraph in the story narrative, all the marked metadata annotations in that paragraph should be removed and all the metadata annotations marked after the position of that paragraph should be updated by decreasing the start and end positions by the number of characters in that paragraph.

*Avoid metadata annotation overlap*

Due to the restriction of XML the text in the story narrative can only be annotated by one metadata tag from the same level. For example, there is a sentence 'Victoria University is located in Wellington'. If story authors want to annotate 'Wellington' by

using both the metadata tag <Place> and <Capital> from the same level, the XML will look like following

```
Victoria University is located in <Place><Capital>Wellington</Capital></Place>
```

When this piece of XML text has been loaded by the XML parser, XML parser will complain that this XML is not valid and conflicts with XML schema definition. In order to avoid this problem, every time when story authors create a new metadata annotation tag, the story authoring tool will check the metadata annotation list. If the story narrative has already been annotated by a metadata tag from the same level, it will not allow story authors to annotate this narrative by new metadata tag again. In future implementation, this problem can be solved by duplicating the annotated text in XML. As example above, the text 'Wellington' be will duplicated in order to be annotated by both 'Place' and 'Capital' metadata tags.

```
Victoria University is located in <Place>Wellington</Place><Capital>Wellington</Capital>
```

### 5.1.3.3 Generate Story XMLFile

When story authors complete their stories, they can export stories to story XML files. For story authors, the exporting process should be very simple by just specifying the path of the story XML file. For generating the story XML file, firstly the story authoring tool will collect recorded metadata annotations specified by story authors. Then the story authoring tool will automatically add default tags where story authors have not specified. For example, if a story author has not marked up a paragraph, the story authoring tool will annotate that paragraph with a default setting. Finally the story authoring tool will assemble metadata annotations to an XML file.

The basic work of generating the story XML file is to insert metadata annotations in the metadata annotation list into the story narrative. In order to avoid metadata tag overlap,

the sequence of inserting metadata annotation should be in accordance with the hierarchical relations of the story metadata. The level 1(top level) metadata elements in the story metadata definition will be inserted first and then level 2. The last level of metadata elements to be added to the story narrative is phrase level metadata.

The information included in the metadata annotation element such as position, name and attributes will be used to generate XML tags for metadata annotations. In the following, the example (Figure 5.7) shows how the metadata annotation element in the metadata annotation list would be translated to an XML tag and inserted to the story narrative.

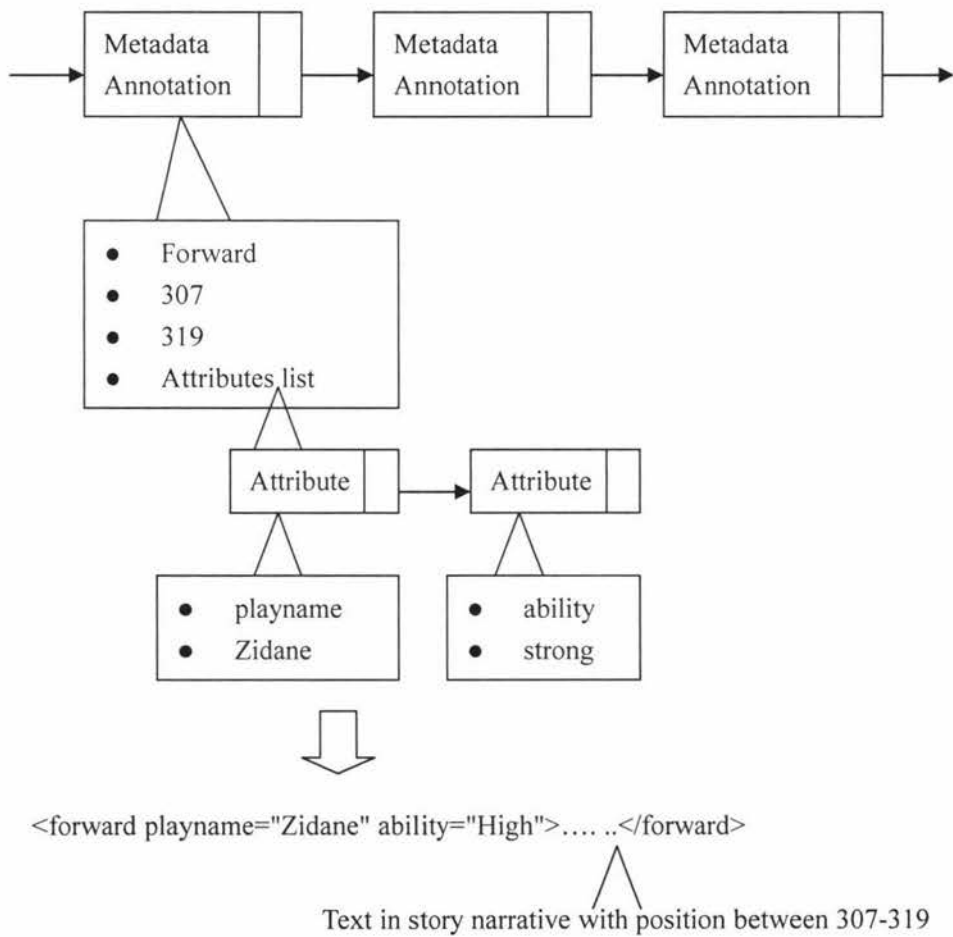


Figure 5 - 7 translation of metadata annotation element to XML tag

#### 5.1.3.4. Story Design File

A story XML file is generated by the story authoring tool when story authors complete the story design. A story design file is a text file which is used to save a story author's temporary design work. A story design file includes the information about story narrative text and details of each metadata annotation such as tag name, attributes and values. When story authors save a story, all the metadata annotations and story narrative in the story authoring tool will be written to a story design file. Vice versa if story authors load a story, the story authoring tool will read the information in a story design file and reassemble the story narrative and metadata annotations in the story authoring tool.

In the implementation, a story design file can be divided into two parts. One is the story narrative and the other is the metadata annotation part. The storage of the story narrative is very straightforward. The story narrative is written to the story design file as plain text at first. Then a separator symbol is added to the story design file. After the narrative part, metadata annotations will be saved in the story design file. Each metadata annotation is a separate line in the story design file. The information of metadata annotations includes the annotation tag name, the position of the annotation in the narrative and attributes and their values related to this metadata tag. Figure 5.8 shows a part of the story design file from a soccer story example.

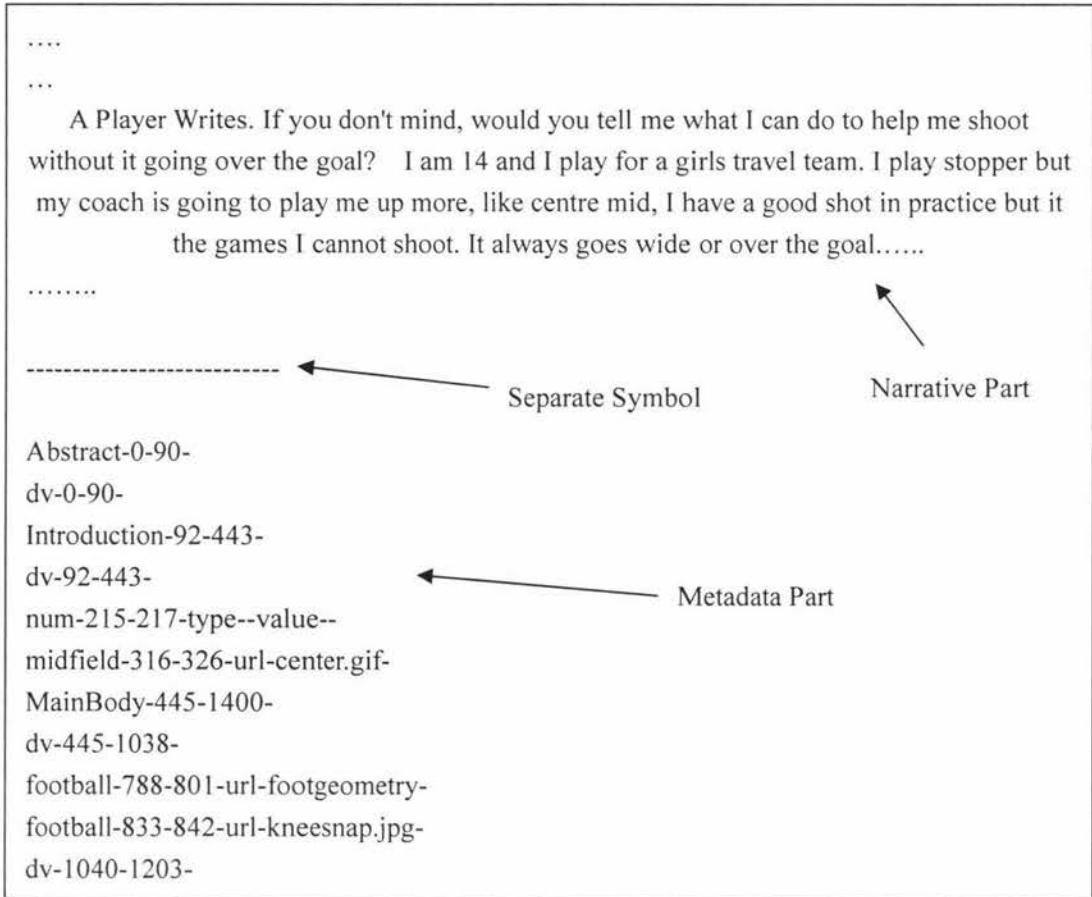


Figure 5 - 8 an example of the story design file

### 5.1.4 Interface Design

In the above section the implementation of the story authoring tool has been discussed and explained. In following, the interface of story authoring tool is demonstrated.

#### 5.1.4.1 Java Interface Component

The story authoring tool is implemented with the Java language. Java provides developers two sets of components to build an interface, which are the Abstract Window Toolkit (AWT) (AWT, 2005) and Swing (Swing, 2005).

AWT is a core Java Package in the Java SDK. It is a portable GUI library for stand-alone Java applications and Java applets. AWT connects Java applications and the native GUI. AWT hides from the programmer the underlying details of the GUI that Java application will be running on. AWT features include:

- A rich set of user interface components;
- A robust event-handling model;
- Graphics and imaging tools, including shape, colour and font classes;
- Flexible windows layouts that do not depend on a particular window size or screen resolution;
- Data transfer classes, for cut and paste through the native platform clipboard;

Swing implements a set of GUI components that build on AWT technology and provides a plugable look and feel. Swing is implemented entirely in the Java programming language. Swing features include:

- All the features of AWT;
- 100% pure java implemented version of the existing AWT component set;

- A rich set of high level advanced component such as trees, tables and tabbed panes.

Swing components contain far more functionalities than the AWT toolkit does. Also swing components provide many new features and capabilities which AWT does not have. In general, AWT components are appropriate for simple application development or single platform target. Swing is suitable for complex applications. Swing is selected for developing story authoring tool.

#### **5.1.4.2 Story Authoring Tool Interface**

The general user interface of the story authoring tool is shown in Figure 5.9. There are four major areas in the story authoring tool. The narrative editing area includes a text pane which enables story authors to input their story narrative. The annotation area contains controls to mark up the story narrative text. The toolbox area provides a set of tools to load and store story design file, load metadata schema, retrieve metadata information, delete metadata information and preview stories. The information area displays the information about a metadata annotation when a story author wants to review metadata annotations.

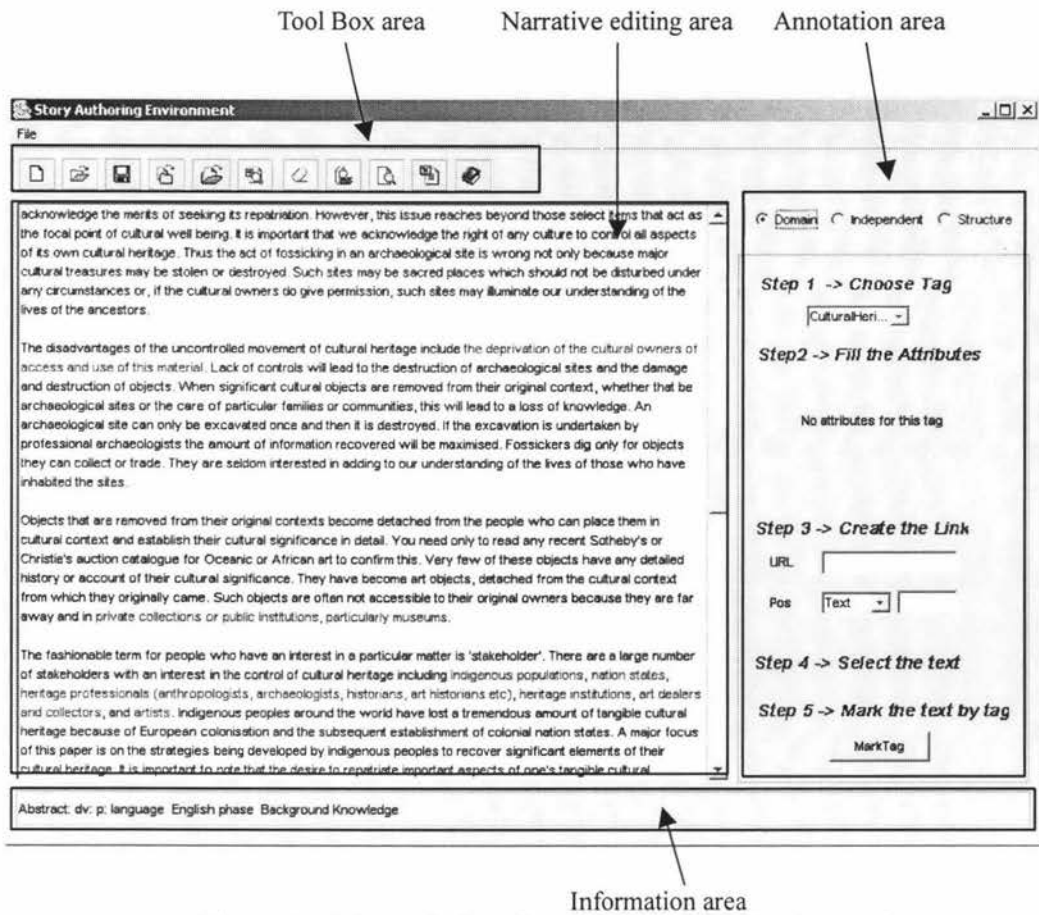


Figure 5 - 9 the main interface of the story authoring tool

### ➤ Narrative editing area

Narrative editing area is implemented by a Java text pane. Basically story authors edit their story narrative in narrative pane. In order to markup the story narrative, story authors need to select the text in the narrative pane. Also the narrative editing pane will highlight the narrative text which has been marked by different metadata tags. Because lots of different metadata tags can be used to mark the story narrative, it is necessary to categorize metadata tags. Three modes have been introduced in the story authoring tool including domain specific metadata mode, domain independent metadata mode and structure mode. The domain specific metadata mode is basically for marking up and retrieving domain specific metadata defined by story authors. The domain independent metadata mode deals with the domain independent metadata elements,



especially phrasal metadata. The structure metadata mode is used when story authors want to mark up the structure of a story by using structural metadata. The story narrative pane will only highlight the annotated text under a certain mode and different highlight colors will be adopted in a different mode. With this design, it is very convenient for story authors to view the different types of metadata annotations clearly without mixing all metadata tags together. Figure 5.10 shows the narrative view under different metadata modes.



Domain specific metadata mode



Domain independent metadata mode



Structure metadata mode

Figure 5 - 10 the narrative view under different metadata mode

➤ Annotation area

The major function of the annotation area is to help story authors to specify information of a metadata annotation. Figure 5.11 shows the details of the annotation area in the story authoring tool. There are a few components in this area for efficiently inputting metadata details. As introduced above, there are three modes in the story authoring tool. Model selection (Figure 5.11) consists of three radio buttons corresponding three modes in story authoring tool. By selecting the different mode, the different metadata tags will be loaded to the tag list and the highlighted text and color in the narrative pane will be changed. The tag list is a combo box which contains the metadata tags under a specific mode. Story authors only need to select a proper tag from the tag list. When a metadata tag has been selected from the tag list, the attribute fields will be automatically displayed depending on the attribute specifications for that metadata tag in the XML schema. In Figure 5.11, when a story author selects the 'num' tag from the tag list, the attribute type and value will be shown to allow the story author to specify the number type and value for a selected text in the story narrative. If story authors want to create a link for a metadata tag, they need to fill the URL attributes in the external link part and furthermore they can specify the position or segment information of a linked document by filling the 'pos' attribute. Finally, story authors can press the 'MarkTag' button to mark up the selected text with the metadata information in the annotation area. At the same time, the selected text in the narrative pane will be highlighted.

Mode Selection

Domain  Independent  Structure

**Step 1 -> Choose Tag**

num

Tag list

**Step 2 -> Fill the Attributes**

type

value

Attributes field

**Step 3 -> Create the Link**

URL

Pos

Text

External Link

**Step 4 -> Select the text**

**Step 5 -> Mark the text by tag**

MarkTag

Mark up button

Figure 5 - 11 detail of annotation area in the story authoring tool

### ➤ Toolbox area

The toolbox area contains a set of buttons to provide story authors with a variety of functions. New, Load and Save buttons allow story authors to create a new story design file, save the current story design file, and load a story design file from a local drive. Because each story includes two sets of internal story metadata, story domain specific metadata and story domain independent metadata, there are two buttons in the tool box to load each kind of internal story metadata definition. Story domain independent metadata has already been built into the story authoring tool, so when story authors click load story domain independent metadata button, they do not need to specify anything. The story domain specific metadata specification is created by story authors,

so if story authors click the load domain specific metadata button, the story authoring tool will ask story authors to select the path of their story domain specific metadata XML schema. Retrieve and delete metadata annotation buttons are basically used to view and delete the metadata annotation in the story narrative. Before story authors click those two buttons, they have to select the annotated text in the story narrative to find a metadata annotation first. The edit reference button helps story authors to input the references of a specific story. Each reference contains information based on the APA reference style, such as author, date, title, and publisher. The preview function writes the current story to a temporary file and calls the story browser to open that temporary file. The export story function will prompt story authors in a file dialog to save the path of the exported story XML file. Figure 5.12 is a screen shot of toolbox in the story authoring tool.

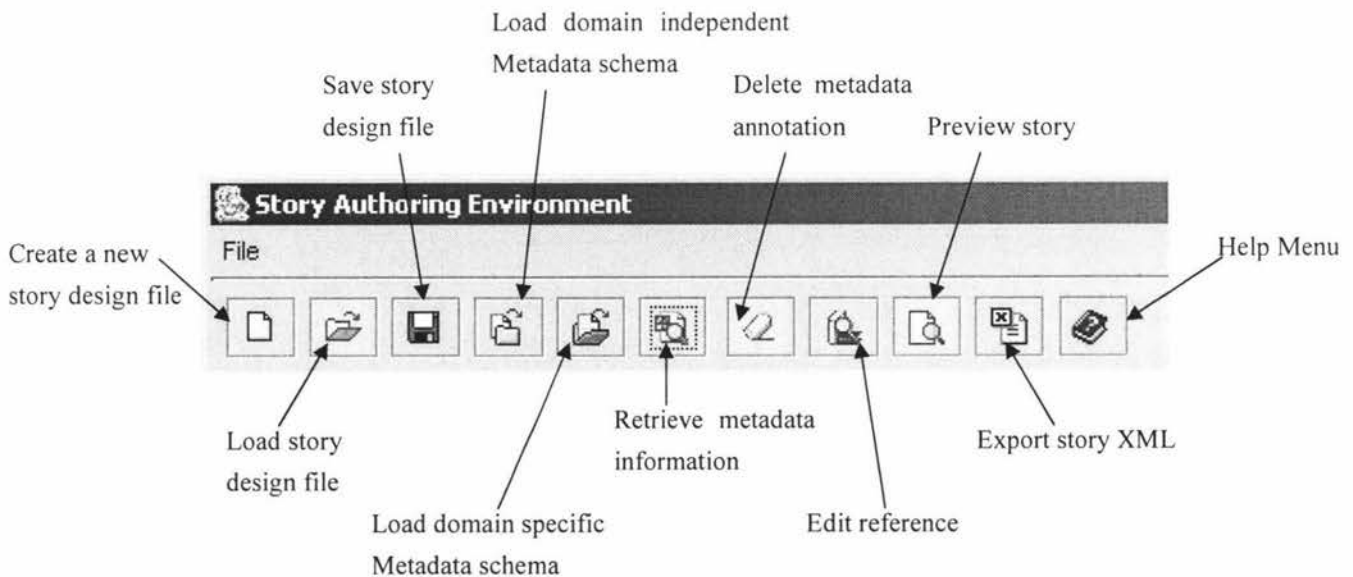
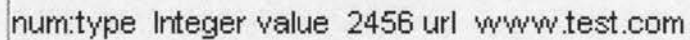


Figure 5 - 12 a screen shot of tool box in the story authoring tool

➤ Information area

The information area is placed at the bottom of the story authoring tool interface. The purpose of information area is to display the metadata information about an annotated text when the story author clicks the retrieve button. The displayed metadata information includes the metadata tag name, attribute names and values. Figure 5.13 shows information of a metadata annotation. In this example the metadata tag of the metadata annotation is num which is used to represent a number in the story narrative. Also there are two attributes related to the 'num' metadata tag. 'Type' indicates the number type of marked number in the story narrative which is integer here. 'Value' represents the value of marked number which is 2456. Moreover this metadata annotation has an external link which is [www.test.com](http://www.test.com).



```
num:type Integer value 2456 url www.test.com
```

Figure 5 - 13 the screen shot of information area in the story authoring tool

## 5.2 Story Browser

### 5.2.1 Requirements Analysis

The story browser is developed for story users to read the story narrative and search in a story with annotated metadata information. The basic functions of a story browser are to parse the story XML file, retrieve the story narrative and extract the metadata annotation information. The following section is a description of the story browser functions that need to be implemented

- Narrative and metadata extraction – The story browser should correctly retrieve the story narrative and the metadata details which have been embedded in a story.
- Search function – A search function is the important functionality of the story browser. The search function enables story users to look at the story from different perspectives and quickly locate a specific position in the story narrative. All of the metadata tags defined in a story can be searched.
- Present linked documents – A story contains various links to multimedia learning objects. Consequently, the story browser has to play these multimedia files. The story browser should be able to play multimedia files starting at a specific offset instead of the media start point.
- Structure of a story – Because structural and instructional information have been stored in the story metadata, the story browser should be able to display a structure diagram of the story narrative with the instructional information. By displaying this structure diagram, story users can quickly locate a suitable section for their needs.

- Visualize metadata – A phrase in the story narrative annotated by the story metadata should be highlighted or commented in some way. The story browser may pop up the comments to remind the user when the user’s mouse moves over a special term or phrase.

## 5.2.2 Technology for Story Browser

### 5.2.2.1 XML parser

A major responsibility of the story browser is to correctly extract the story narrative and metadata information from a story XML file. Although an XML document can be read as a text file, retrieving the story metadata from a story XML document by directly using traditional sequential file-access techniques is neither practical nor efficient. An XML parser is able to check an XML document’s syntax and enable software programs to process mark up data. A good XML parser is a critical component to implement the story browser. Currently there are two different types of XML parser APIs available, Document Object Model (DOM) and Simple API for XML (SAX). Before choosing the XML parser for the story browser, the following sections evaluate both the DOM and SAX XML parsers.

#### ➤ *DOM*

The general definition of DOM from W3C is that “the Document Object Model is a platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style documents. The document can be further processed and the results of that processing can be incorporated back into the presented page” (DOM, 2004). XML DOM can be defined as a programming interface for XML documents. It defines the way an XML document can be accessed and manipulated. When a DOM parser successfully parses an XML

document, it provides a standard set of objects to represent that XML file. Basically, the DOM parser creates a tree structure in memory that contains the XML documents data. The hierarchical tree structure is called Document Object tree.

Each tag name in a XML document represents a tree node. A node that contains other nodes is called parent node. A parent node can have many children, but a child node can only have one parent node. Nodes that are peers are called sibling nodes. A node's descendant nodes include its children, its children's children and so on. A node's ancestor nodes include its parent, its parent's parent and so on. In DOM implementation each node is treated as an object which has properties, methods and events. Properties associated with a node include names, values and child nodes. Methods enable programs to create, delete and append nodes, load XML documents and so on.

#### ➤ *SAX*

SAX is another API for dealing with XML documents. "SAX (the Simple API for XML) is a standard API for event-driven processing of XML data, allowing parsers to deliver information to applications in digestible chunks" (Neil, 1999). SAX gives developer access to the information in a XML document by a sequence of events, not as a tree of nodes. The DOM parser does almost everything for users: read the XML document in, create the object model and give the reference to this object model so that the user can manipulate it. SAX is really simple. SAX does not expect the parser to do much, and what SAX requires is that the parser should read in the XML document, and throw events based on tags it encounters in the XML document. The developer is responsible for interpreting those events by writing an XML document handler class which is responsible for making sense of all the tag events and creating objects in the developer's object model.

DOM and SAX are appropriate for different situations. SAX can be really fast at runtime if the XML document is relative simple. In this case, it is faster than DOM.



Because DOM needs to create a tree based object for a XML document and SAX just throws events. On the other hand, if an XML document has a very complicated structure; it is extremely hard to write SAX document handlers to interpret all the SAX events.

In this project, due to the complexity and size of the story XML document, it is much easier to parse XML by using DOM rather than SAX. Also, by creating the DOM tree it can reduce the workload of the story browser implementation. Otherwise, the story browser will be very complicated by writing a XML tag handler for each XML metadata tag.

The parser used in this project is the MSXML parser 4.0 (MSXML, 2004), which is DOM based and fully compatible with W3C standards. MSXML DOM provides an API containing a set of interfaces for parsing XML documents, working with tree nodes, as well as dynamic validation against XML schema. Also, MSXML has good compatibility with current popular web browsers. MSXML has already been built into to Internet Explore 6. Some script languages such as JavaScript and VBScript are able to use the method and properties directly in IE without installing MSXML SDK. MSXML can also support other web browsers very well by installing MSXML SDK.

#### **5.2.2.2 JavaScript**

According to the design requirements of the story environment, the story browser is used to parse a story XML file and display it to end user. Actually, the story browser itself can be treated as a stand alone application, which means it does not need to contact any server to help it parse a story XML file. Consequently, it is pointless to use server side web application technologies such as JSP, ASP and PHP to implement the story browser.

JavaScript is the best option to implement the story browser for the following reasons:

- Firstly, unlike VBscript, JavaScript has been embedded in Firefox, Internet Explorer and other web browsers and embellished for web programming with the addition of objects that represent the web browser window and its contents (Flanagan, 2002). Nowadays, JavaScript is supported by almost all web browsers. If the story browser is implemented by JavaScript, it can be used widely in different environments and platforms.
- Secondly, JavaScript is a powerful scripting language, which enables users to control the web browser, and also enables the web browser to interact with users and dynamically create HTML content. For the story browser, the story narrative should be dynamically written to the web browser after parsing the story XML file. And some of story narrative should be highlighted based on the story metadata information. JavaScript can achieve these tasks without any potential difficulties.
- Thirdly, JavaScript has similar syntax as Java and it includes most programming constructs such as the if statement, the while loop, and the logical and math operators. This makes it easier for developers to quickly master the syntax of JavaScript.

The following are the most important features of JavaScript to be used in the implementation of story browser.

#### *Control Document Appearance and Content*

JavaScript can fully control a HTML document in the Web Brower by accessing the Document Object. For example, by using a JavaScript document object, programmers are able to write arbitrary HTML into a document as the document is being parsed by the browser. Moreover, the Document object allows programmers to specify colors for

the document background, the text, and the hypertext links and so on. These abilities are very helpful and critical to generate dynamic and conditional HTML documents. Another important feature of JavaScript is that it works particularly well in multiframe documents. Indeed, in some cases dynamic generation of frame content allows a JavaScript program to replace some traditional server-side scripts. These abilities are very important for the story browser that enables the story browser to dynamically display the story narrative and highlight the metadata information.

### *Control the Browser*

Several JavaScript objects allow control over the behavior of the browser. For example, the Window object supports methods to pop up dialog boxes to display simple messages to the user and get simple input from the user. This object also defines a method to create and open entirely new browser windows, which can have any specified size and any combination of user controls.

JavaScript also allows control over web pages displayed in the browser. The Location object allows developers to download and display the contents of any URL in any window or frame of the browser. The History object allows developers to move forward and back within the user's browsing history, to simulate the action of the browser's Forward and Back buttons.

This part of JavaScript is also very useful for the story browser. In the story browser, a build-in media player will be called to play media sources at a specific position. With the control of the browser, JavaScript can easily control the media player as a control object in the web browser.

### 5.2.3 Implementation Issues

This section will explain the implementation details of the story browser. The basic idea of implementing the story browser is to translate a story XML file to a HTML document by using a JavaScript program. In the technologies review section, the strengths of the XML DOM parser and JavaScript have already been discussed.

➤ *Load story*

This section talks about the implementation of loading a story to the story browser. Figure 5.14 shows the process of loading a story.

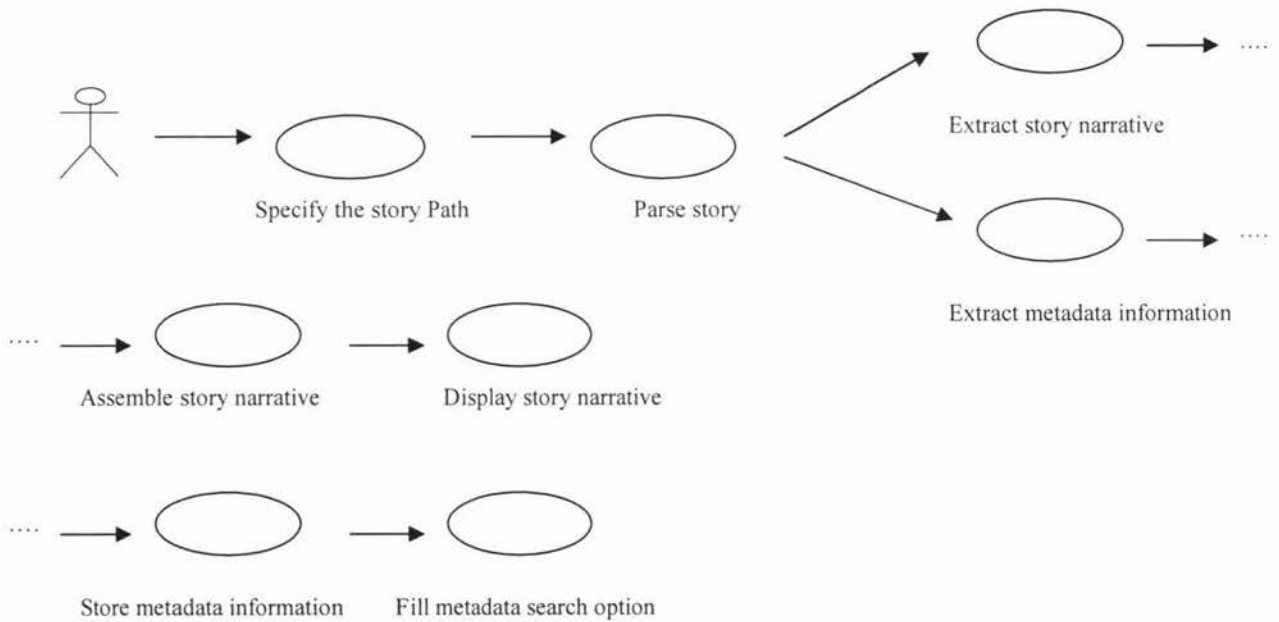


Figure 5 - 14 process of loading a story

First of all, the story browser should be able to load a story by the specified file path. This path can be either a local file or a file in a remote server. When a story has been loaded into the story browser, the story browser parses the story XML file and goes through each XML tag in a story and takes out the value under that tag. Afterwards the story narrative will be assembled and transformed to user readable text by the story

browser. At the same time the metadata mark up information will be stored as well such as the annotated metadata tag name and attributes for each tag.

When the parsing process is finished, the story browser will display the story narrative to story users and also the retrieved metadata annotation information will be used to fill the metadata search tag list in the story browser.

➤ *Parse story*

In the above technology part, MSXML has been chosen as the story XML file parser. Because MSXML is a DOM based XML parser, when a story XML file has been loaded to MSXML, it will create a tree that contains all story metadata XML tags. Figure 5.15 is an example of the XML DOM tree created by MSXML for a story XML file. The story browser would travel this tree to extract the story narrative and all the metadata tags (a node in the tree).

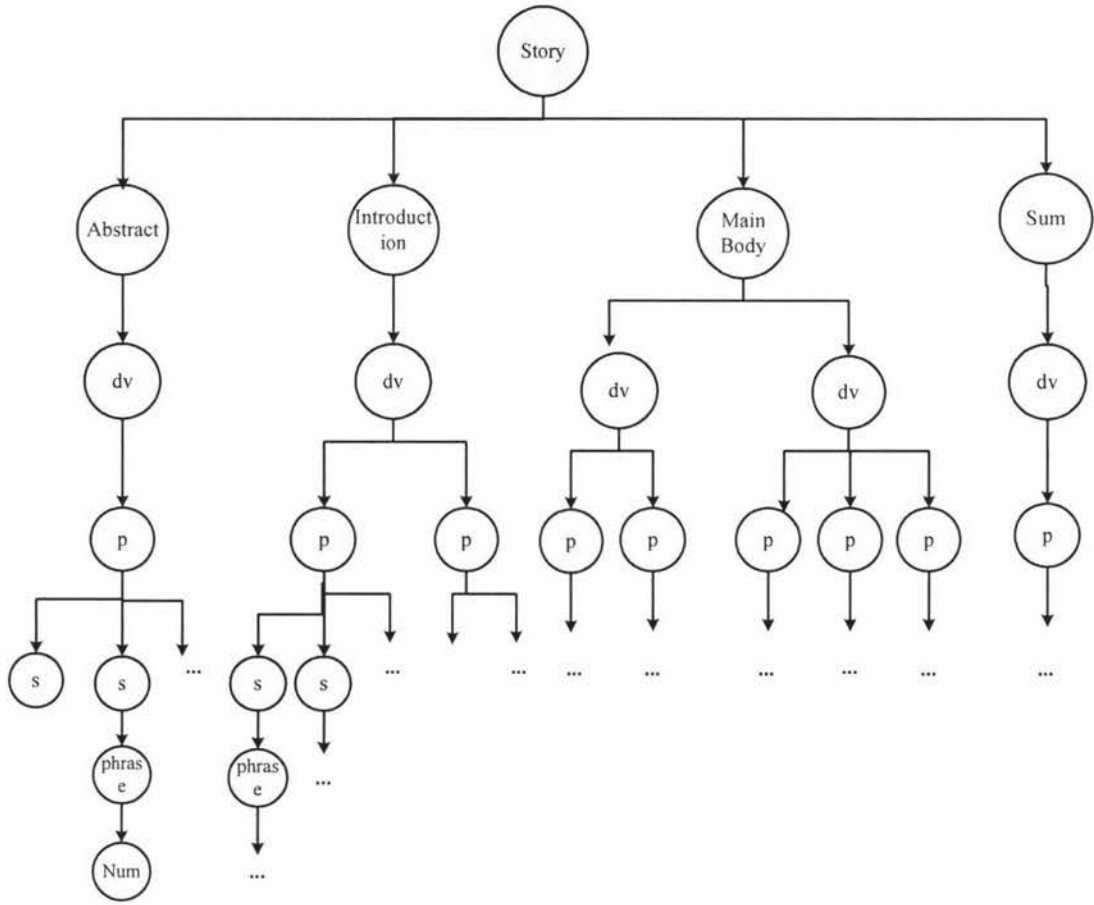


Figure 5 - 15 DOM tree of story XML

When MSXML successfully parses a story XML document, it will return to the root of tree object. In the above example, the node of 'Story' will be returned by MSXML. MSXML also provides a set of the methods and functions to help the story browser to travel the tree. Table 5.1 shows the properties and methods provided by MSXML to help the developer to travel the DOM tree.

Property	Description
attributes	Returns a NamedNodeMap containing all attributes for this node
childNodes	Returns a NodeList containing all the child nodes for this node
firstChild	Returns the first child node for this node
lastChild	Returns the last child node for this node
nextSibling	Returns the next sibling node. Two nodes are siblings if they have the same parent node
nodeName	Returns the nodeName, depending on the type
nodeType	Returns the nodeType as a number
nodeValue	Returns, or sets, the value of this node, depending on the type
ownerDocument	Returns the root node of the document
parentNode	Returns the parent node for this node
previousSibling	Returns the previous sibling node. Two nodes are siblings if they have the same parent node

Table 5 - 1 properties and methods of MSXML to travel XML DOM tree

There are many algorithms used to travel a tree, such as depth first, breadth first, greedy and A-Star travel. Since the story narrative is encoded into the metadata elements `<s>` and `<phrase>`, in order to get the story narrative in sequence, the story browser has to access elements from the left-most branch to right-most branch. Based on this principle, the depth first travel method is the only choice to satisfy this requirement. For instance, in the above example in Figure 5.15, the element `<s>` in the leftmost branch will be accessed by the story browser, then the tag `<num>` will be visited, after that is the tag `<phrase>` and `<s>` in the middle branch.

Once the story narrative has been extracted, it will be written to the display frame as HTML in the story browser.

➤ Search metadata information

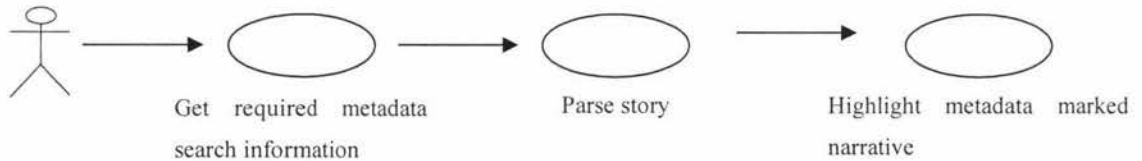


Figure 5 - 16 process of searching metadata information.

Figure 5.16 shows the processes involved in searching metadata information. Firstly, story users need to provide information about the metadata search, such as the metadata tag name and keywords. Because the metadata tags defined in a story have already been retrieved when a story is loaded to the story browser, story users they can just select metadata tags from a list provided by the story browser. Also story users are able to filter the metadata tag search by providing keywords. Therefore, only the metadata annotation that contains search keywords will be returned. Secondly, the story browser will parse the story XML file again to find the required metadata tags. Finally, the story browser will rebuild the story narrative and highlight the searched metadata annotation narrative.

The implementation of searching for a metadata tag is almost the same as loading a story. However, some additional tasks will be processed when the story browser encounters a searched metadata tag. When a searched metadata tag has been met by the story XML parser, the narrative in this tag will be highlighted by different colors in HTML. Moreover the story browser will create an anchor for this piece of text in the story narrative, thus if there are many narrative segments marked with same metadata tag, story users are able to locate to a specific metadata annotated text by clicking the search results.



For example, if there is a piece of the text '1-5 Victoria street' in the story narrative annotated as the metadata tag <address>

```
<address> 1- 5 Victoria Street </address>
```

When the user wants to search all the metadata tags for <address>, this metadata annotation will be translated to the following HTML code.

Highlight Color	Anchor	Hint Message when users move mouse over the text
↓	↓	↓
<pre>&lt;b&gt;&lt;font color="#FF0000"&gt;&lt;a name="address" title="Address metadata"&gt; 1 – 5 Claremont Grove &lt;/a&gt;&lt;/font&gt;&lt;/b&gt;</pre>		

➤ View the linked learning objects

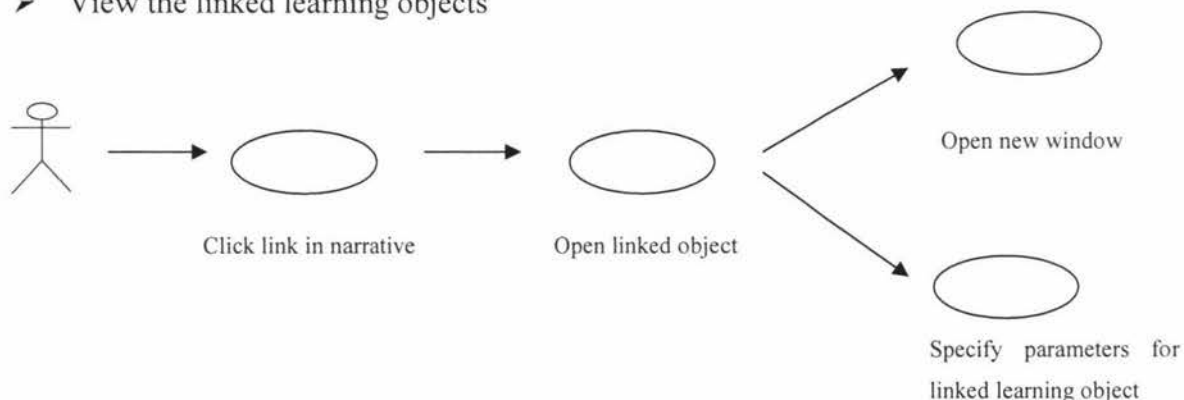


Figure 5 - 17 process of opening a linked learning object

How to deal with the learning object link is an important issue of implementing the story browser. A story contains links to different learning objects related to each theme and subject. In principle, each learning object will be associated with a piece of text in the story narrative. When a story has been parsed by the story browser, the story browser creates a link for each learning object inside the story narrative, so that story

users can jump to the learning object easily by pressing the link. Figure 5.17 shows the process of opening a linked learning object.

In a story, an external learning object link is recorded by the <mediaLocator> metadata element. The information of a learning object link includes the URL of that learning object and the segment information such as start point, end point and duration. This information enables the story browser to point to a specific part inside a learning object. In general, the type of learning object can be any media, for example, web page, word document and audio movie file. The easiest way to open those different multimedia files is to just open those files by web browser directly, so web browser can open those multimedia files by using the appropriate applications. Web pages, text files and word processing documents can be display directly by the web browser. Audio/visual files can be played by calling the media player for them.

As mentioned previously, a story should be able to link to a specific segment inside a learning object rather than the whole learning object, which is especially useful when a learning object is big. Therefore, when a story user clicks the learning object link in the story narrative, basically the story browser can pop up a new window to display that learning object. Meanwhile the segment information in the multimedia link metadata annotation would be used to customize the linked learning object by specifying the parameters in the new window. For example, if a learning object is a web page, the story browser can locate a specific segment in that page. If a learning object is an audio/visual file, the story browser should be able to play the multimedia file at a specific time that has been specified by story authors.

JavaScript provides a powerful function to control the web browser and content displayed in the web browser. This makes the implementation relatively easy. For a text document or web page the story browser still opens the link in a new window. But an additional JavaScript function has been implemented which can scroll the web page

depending on the segment information specified in the <mediaLocator> tag. For an audio/visual link, the story browser dynamically adds a media player control into the new web browser window by JavaScript code to play that audio/visual file. The story browser can specify the file path, start time, play rate, volume and other parameters for the embedded media player control by using JavaScript. This enables the story browser to play a multimedia file at a specific segment. Figure 5.18 is a simple example of HTML code generated by JavaScript to open a video clip in web browser starting at 3 seconds.

```
<object classid='clsid:6BF52A52-394A-11D3-B153-00C04F79FAA6' id='WindowsMediaPlayer1'>
<param name='URL' value='file:///d:/storyProject/resource/van.wmv'>
<param name='currentPosition' value='3'>
</object>
```

Figure 5 - 18 HTML code example of playing a media file

➤ Display structure diagram

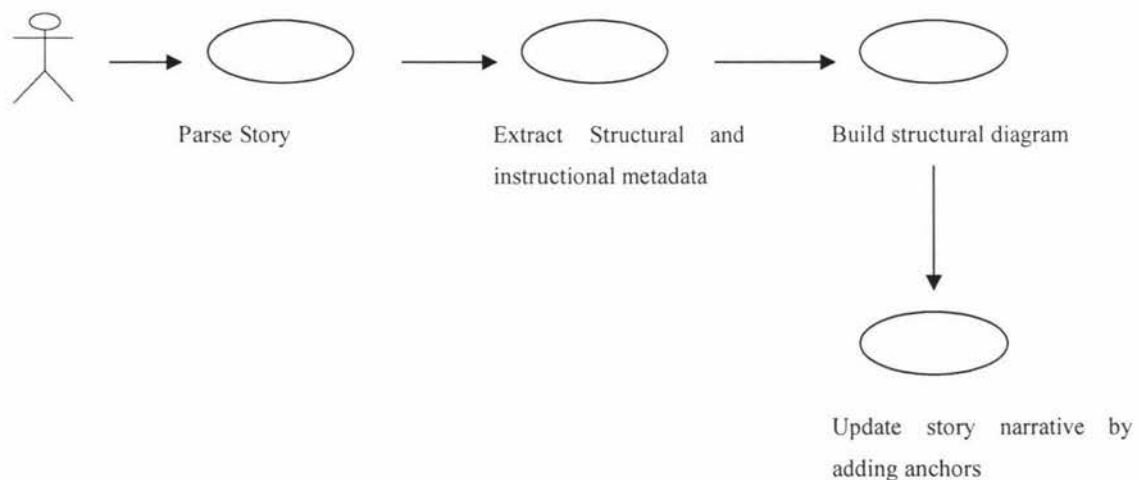


Figure 5 - 19 process of displaying structure diagram

Figure 5.19 shows the process of displaying a structure diagram. Because a story includes structural and instructional metadata tags to mark up the structure of the story narrative and the instructional strategy behind each story narrative section, it is possible

for the story browser to display the structure diagram of the story narrative with instructional strategy descriptions.

When the story browser loads a story, the story browser extracts the structural metadata tags such as <Abstract>, <dv>, <p>. Also the instructional information embedded in the paragraph metadata tag <p> will be retrieved. Then the story browser uses this information to build a hierarchical structure diagram to show the structure of the story narrative. Moreover, in order to help story users locate story narrative sections quickly, the story narrative will be refreshed by adding anchors for each corresponding element in the structure diagram. Therefore, story users can locate the story narrative section by clicking each item in the story structure diagram.

The implementation of building the structure diagram is similar to the metadata search. In this case whenever the story browser meets a structural metadata tag such as <abstract>, <dv> and <p>, it adds an item to the structure diagram. Also when the story browser finds the paragraph tag <p>, it retrieves the value attribute 'phase' of the paragraph tag to the structure diagram which contains the PBL information of each paragraph. Anchor points will also be added to the story narrative to remember the position of each structural tag, thus story users are able to quickly locate a segment in story narrative.

When a structural item in the structural diagram has been clicked, the story browser highlights the whole section of that structural unit, and scrolls to that specific section using the previous created anchor.

## 5.2.4 Interface Design

### ➤ *Story Browser Interface*

There two major areas in the story browser: control area and display area. The control area contains different controls to help story users to search different metadata tags defined inside a story. The display Area displays the story narrative which will be updated by the change in the control area. Figure 5.20 shows the basic interface of the story browser

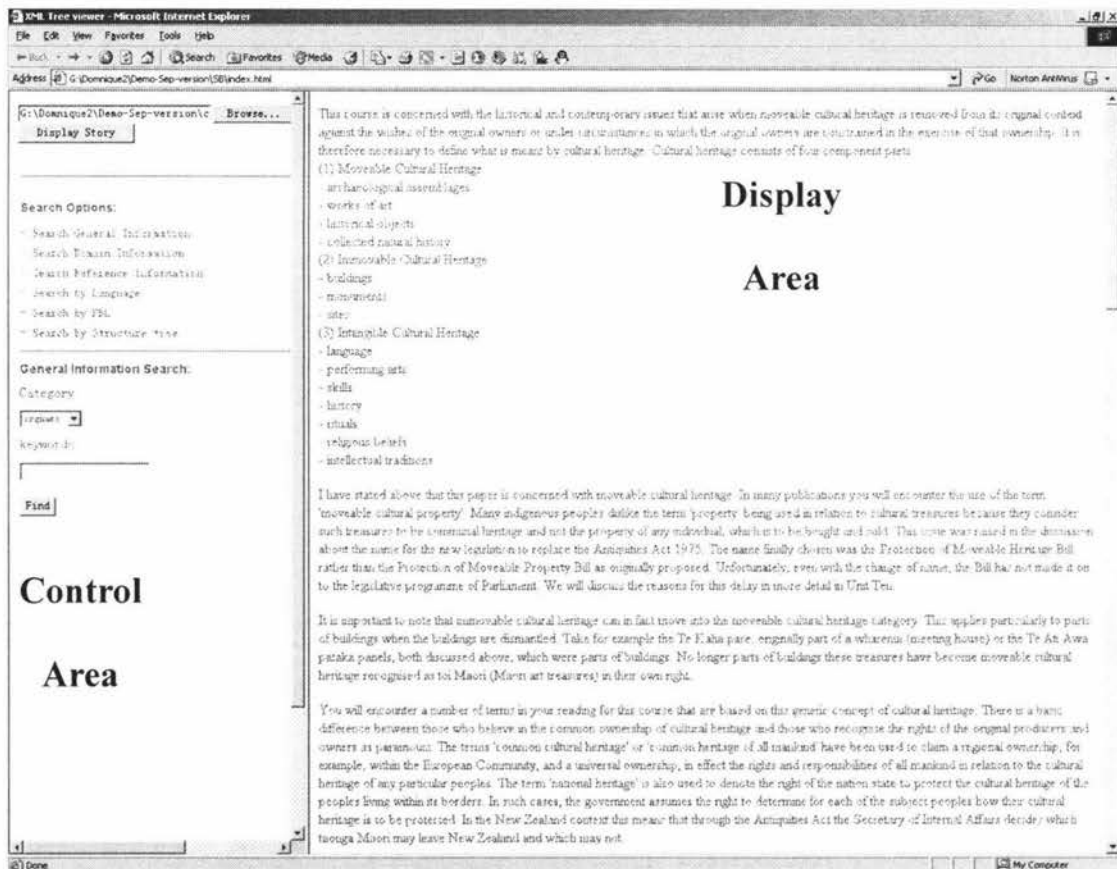


Figure 5 - 20 basic interface of the story browser

➤ *Control Area*

Figure 5.21 shows three different sub parts in the control area.



Figure 5 - 21 control area in the story browser

The loading area enables story users to specify the file path of a story. Story users can either type the URL of a story or select the local file path by pushing the browse button. When the file path of a story has been specified, story users can press the display story button to display the story narrative in display area.

The metadata area provides the different categories of the metadata defined in a story. General information includes all the domain independent metadata annotations such as number, address, person name. Domain information contains all the domain specific metadata annotations. Reference information is the metadata annotations for the references of a story, such as author, date, and publisher. Language and PBL are two attributes associated with the paragraph metadata tag <p>. Language provides the language information of each paragraph. PBL enables story users to find the PBL stage behind each story narrative paragraph. The structure tree shows story users a structural diagram of the story narrative which enables story users to become quickly familiar with the structure of the story narrative.

The metadata search area provides the search option for the story users. When story users click each metadata area in the above Figure 5-21, the story browser loads all available metadata tags defined in a story under that area into the combo box. Therefore story users can easily select search metadata tag from the tag list and type keywords. When story users click the find button, the search results will be dynamically written to the control area. Story users are able to locate each search result in the story narrative by directly selecting each search result. Also, the target metadata annotation narrative will be highlighted in different colors in the display area.

When story users click the structure tree in the metadata area, the metadata search area will display the structure diagram of a story. The structure diagram shows the basic structure of the story narrative. When a user clicks on an item in the structure tree, the story browser automatically scrolls to that narrative section in the display area and highlights with different colors. Figure 5.22 shows the structure diagram of a story narrative.

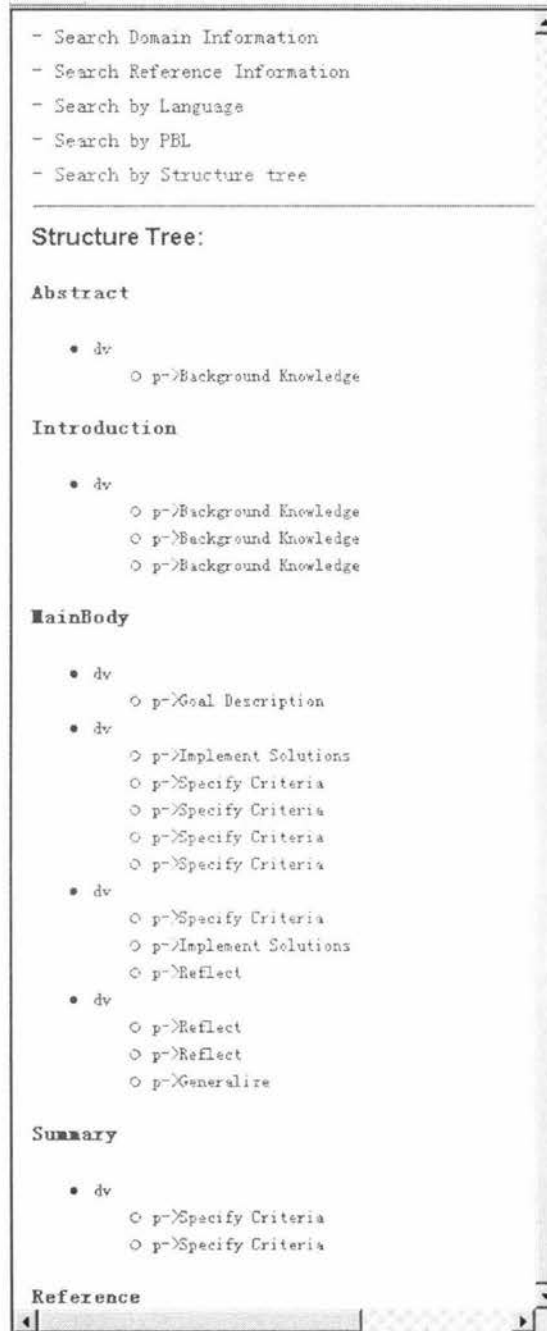


Figure 5 - 22 structure diagram of a story narrative



➤ *Display Area*

The display area displays the story narrative. The content in the display area will be refreshed when the story users want to search in a story narrative.

For each link to a learning object, display area will add a text '(Link)' to point to a specific learning object. The search result of metadata annotation search will be highlighted by different colors. Figure 5.23 shows a snap shot of the display area in the story browser.

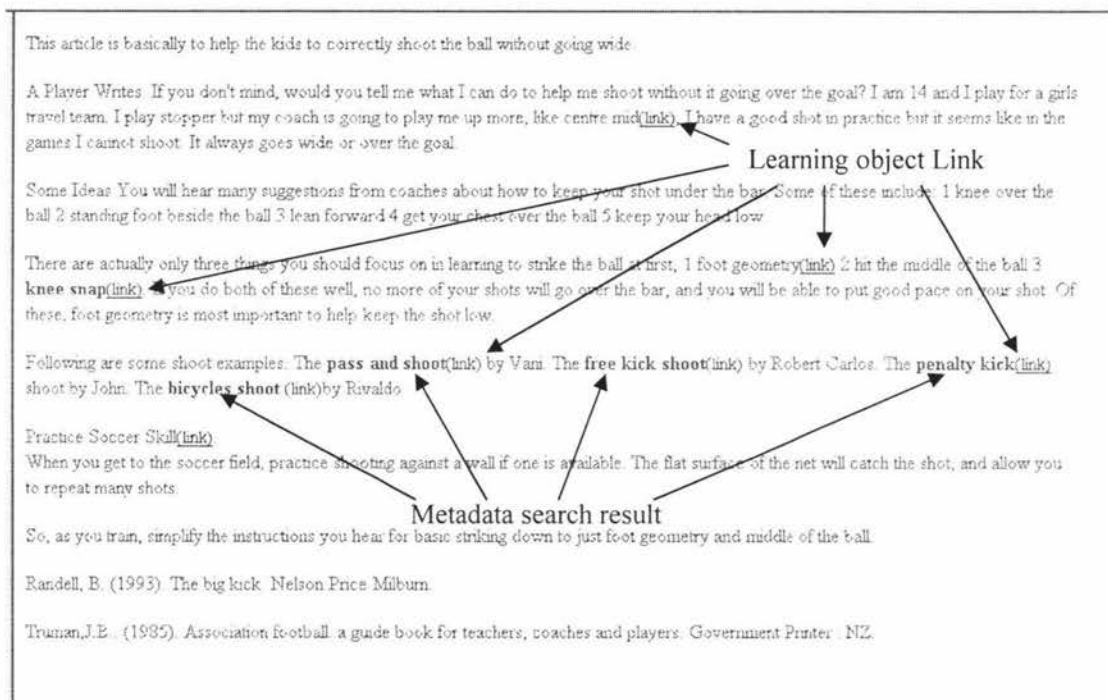


Figure 5 - 23 a snap shot of display area in the story browser

### 5.3 Summary

In this chapter, the design and implementation details of the story authoring tool and the story browser have been introduced. Both tools have been developed based on the concept described in Chapter 3. The main features of the story authoring tool and the story browser include:

- Story Authoring Tool – edit the story narrative, create links to learning objects, annotate text units in the story narrative and generate story XML files.
- Story Browser – parse the story XML files, retrieve the story narrative and metadata annotations, search the metadata annotation in the narrative, play multimedia learning objects.

To realize these functions well, detailed analysis and design were performed. The development and implementation of the story authoring tool and story browser went through the four stages of requirements analysis, technology review, implementation design and interface design.

## Chapter 6 Evaluation

---

The story authoring tool and story browser have been implemented in this project. This chapter will discuss the story environment implementation in comparison to Annotea which is web document annotation project from W3C to evaluate the story environment's strengths and weaknesses. Also some testing results and further testing plans will be explained.

### 6.1 Annotea and the Story Environment

#### ➤ **Annotea**

Annotea (Annotea, 2004) is a W3C project under the section of Semantic Web Advanced Development. The aim of Annotea is to enhance the collaboration of a web document via the shared metadata-based web annotations. These annotations include users' comments, notes, explanations, or other types of external links. In Annotea, annotations are viewed as statements made by authors or readers about a web document. Every user of Annotea can read and write annotations about a specific web document by consulting the annotation server. The basic information of an Annotea annotation includes the date of creation, name of the author, the annotation type (such as comment, query, and correction). Also, Annotea users are able to define their own metadata elements for annotation. The annotation metadata information is defined according to an RDF schema (RDF, 2004) and saved to database.

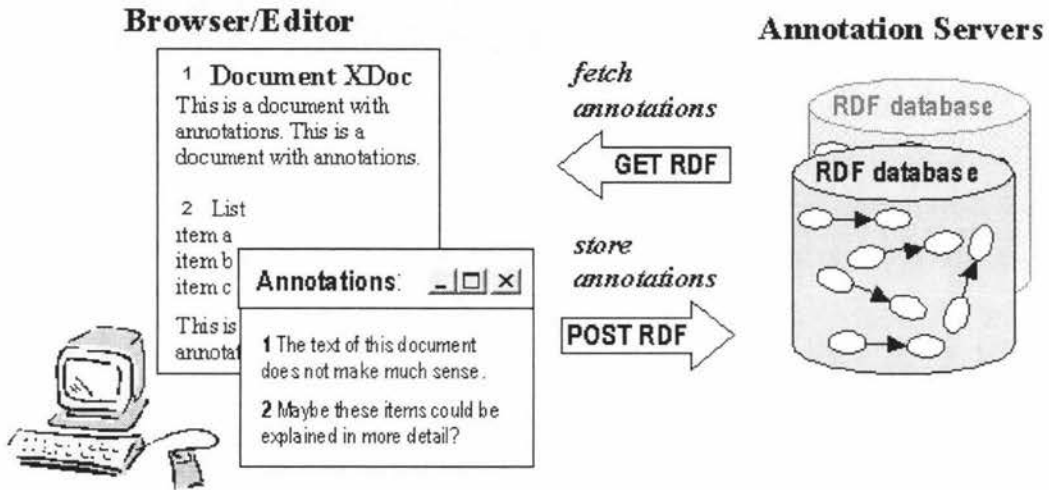


Figure 6 - 1 the basic architecture of Annotea (Kahan & Koivunen, 2001)

The implementation of Annotea is based on a generic RDF database and Apache HTTP server. The RDF database stores all the annotation data based on RDF schema. An Apache HTTP Server gets the request from the Annotea browser to update the RDF annotation database or finds annotations for a specific web document from the RDF database and sends annotations back to the Annotea browser. All communications between client and an annotation server use the standard HTTP methods such as GET and POST. Figure 6.1 shows the basic architecture of Annotea (Kahan&Koivunen, 2001). In Annotea, a tool named AMAYA (Annotea Browser) has been implemented to display a web document and create and browse annotations for a specific web document. Figure 6.2 is a screen shot of AMAYA.

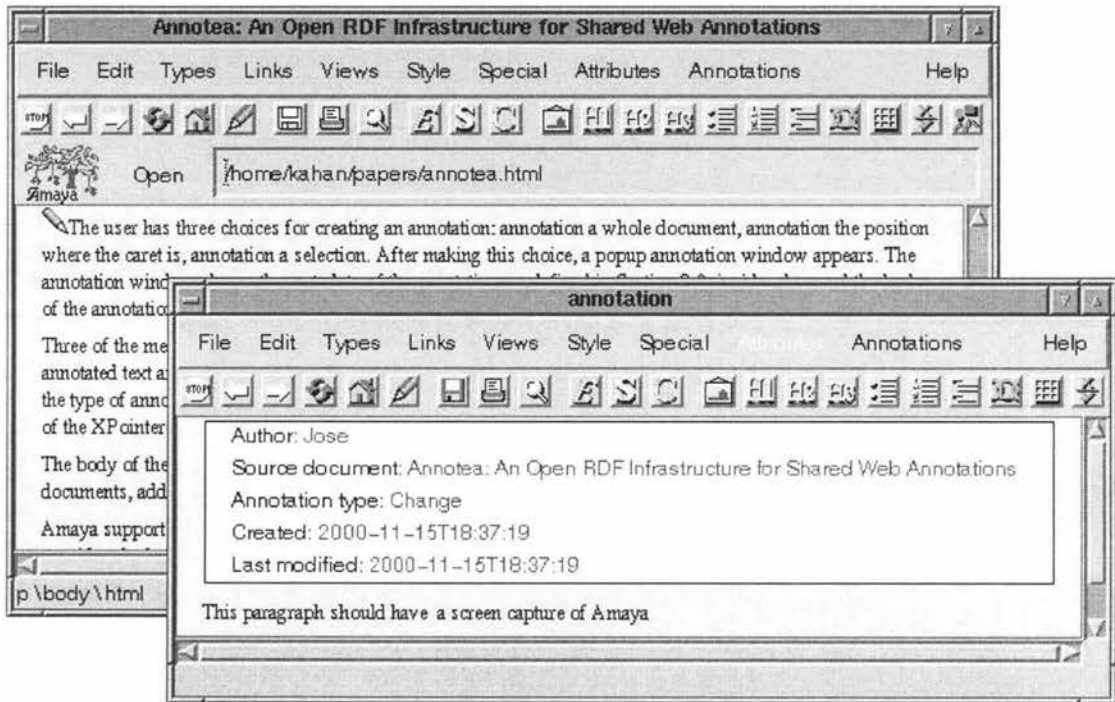


Figure 6 - 2 a screen shot of AMAYA (Kahan & Koivunen, 2001)

Following paragraphs describe the process of creating and browsing annotations.

#### *Creating Annotations (Kaha & Koivunen, 2001)*

- The user browses a document
- The user selects some text on the document and tells the browser that he wants to annotate this text.
- The browser pops up a new window, where the user can type the text of his annotation and choose the type of annotation.
- The browser publishes the annotation to a given annotation server. To do this, the browser generates an RDF description of the annotation that includes metadata and the body and sends it to the server.

*Browsing Annotations (Kaha & Koivunen, 2001)*

- The user browses a document
- The browser queries each of the annotation servers, requesting via an HTTP GET method the annotation metadata that is associated with the document's URL.
- Each annotation server replies with an RDF list of the annotation metadata.
- For each list of annotations that it receives, the browser parses the metadata of each annotation and highlights the annotated text.
- If the user clicks on the highlighted text, the browser will use an HTTP GET method to fetch the body of the annotation from the URI specified in the metadata.
- Finally, the browser will open up a window showing the metadata and the body.

**➤ Story Environment**

Unlike Annotea, a story integrates both narrative text and its metadata to a single XML file. So the basic architecture of story environment is different from the Annotea's structure. The story browser is used to fetch a story and display the story text narrative and highlight the metadata information and link to learning objects. The story authoring tool is used by story authors to create a story including both text narrative and metadata information. Stories are supposed to be stored in the story repositories. Store repository stores stories which are categorized by the subject domain. A story repository has a story repository management system. This system provides story authors a platform to specify external story metadata and upload stories to story repository. For story readers, story repository management system supports story reader to locate stories by external metadata information and download corresponding stories from story repository. The physical architecture can be represented as in Figure 6.3.

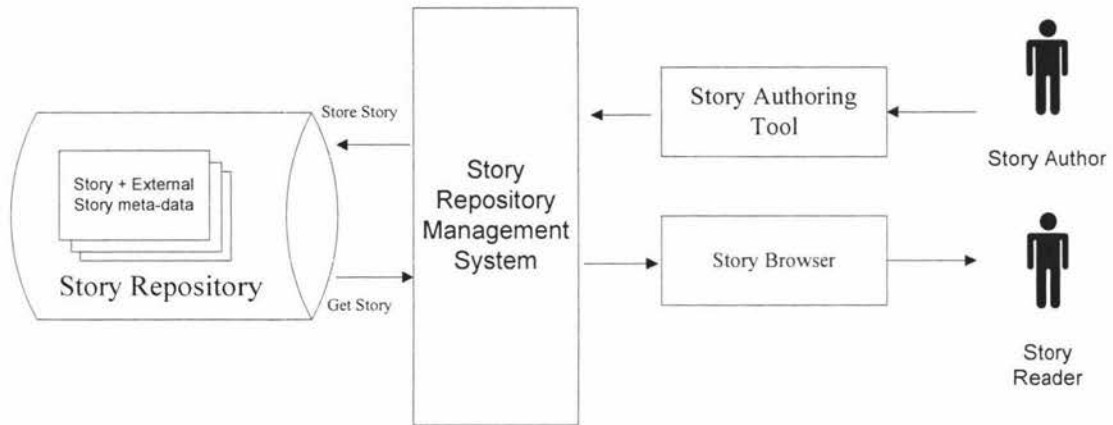


Figure 6 - 3 physical architecture of the story environment

### ➤ Evaluation

Both the story environment and Annotea focus on using metadata to mark up a document. But the implementation approaches are different. This section will discuss strengths and weaknesses of the both approaches.

The physical structure of Annotea is more complicated than the story environment. It requires a database server to separately store the metadata annotations and also needs the Apache HTTP server to communicate between Annotea user and Annotea annotation server. Also Annotea requires users to install their own tool (AMAYA) to create and browse annotations. It cannot work properly with normal web browser such as Internet Explorer. The whole architecture of Annotea is highly dependent on the metadata annotation server and the HTTP apache server. Users need to contact the HTTP server to post and download metadata for metadata annotation. But Annotea has its own strengths. Firstly the Annotea solution can mark up most web documents existing on the Internet with the AMAYA Tool. Secondly, Annotea can show users the annotation for a web document from different people by AMAYA tool.

Compared with the Annotea approach, the story environment has a simple physical structure. There is no metadata annotation server required. A story is a XML file which contains both story narrative and metadata. Only a normal file server is required to store stories. The story environment is also more flexible than Annotea. Story authors can create stories at any time even without contacting story repository. The story browser can display a story by either URL or file path of local drive. The actual implementation of the story browser is just a piece of JavaScript program. A standard web browser such as IE can correctly display the story text and highlight the metadata information without any problem. From the story reader's point of view, there is no additional software requirement. The advantages of Annotea are the weaknesses of the story environment. The story browser can only view the story which is created by the story authoring tool. It is hard for the story environment to mark up the existing web documents. Also each story author will create their own stories. Currently, the story browser cannot integrate different author's metadata annotations for a specific story.

## 6.2 User Testing

After the story authoring tool and the story browser were implemented, they have been evaluated by several people from different backgrounds. They helped this project to make a few story examples. Currently, two story examples have been made. One is a soccer coaching example and the other is on cultural heritage. The feedback was generally positive and it seems that a story would certainly be a useful tool for help teachers to find proper learning objects and also offer teachers the context information behind learning objects. The story authoring tool is commented on to provide users with an easy way to write the story narrative make annotations and link learning objects. The story browser is also evaluated as a nice tool to view the story narrative, search inside the story narrative and open/play learning objects. The flexible search options and structure diagram are appreciated by most people. Also the visual representation of story



narrative and metadata is tidy and clear. The interfaces of the story authoring tool and the story browser are found to be general intuitive and therefore the user's input needed for the system has been minimized.

A very interesting point has been raised by the users. The story concept can be shifted to represent learning materials. Originally in the story concept, a story acts as the information source to help teachers to composite learning material such as lecture notes, exercises and tests. A teacher collects learning objects by reading stories. However, some users mentioned that a story itself can be a very useful tool to make learning material rather than just function as an information source. That means some learning materials such as lecture notes or study guides can be represented as stories. A story can help and guide students, especially extramural students, to quickly and efficiently read and understand some large lecture notes and study guides, since those students can not attend lectures and tutorials. In this case teachers will use internal story metadata to mark up the learning materials structure, important terms, paragraphs, sentences inside the learning material and build in instructions for studying the learning materials. Thus extramural students are able to find and catch important information behind narrative text by using the story browser. Also the search function facilitates students to look at the particular points and sections inside extensive learning material.

Currently, first user feedback has been given and has led to sample stories on cultural heritage and sports history. In future more user tests are required to improve the story environment. User acceptance testing is an important testing that should be conducted next with more users from different groups. This testing may include the following areas.

- The usefulness of the story concept: Basically the story concept is introduced to help teachers to quickly locate the learning objects. So it is sensible to have users test how helpful of stories can be to locate learning objects. The target of this testing could potentially be school teachers who can compare the story approach with their

usual resource searching technologies.

- The nature of GUI of the story browser and the story authoring tool: This testing focuses on validating the interface design of the story browser and the story authoring tool and identifying the further requirements/improvements for each GUI. A testing plan is required to test the efficiency of the interface, which may consist of story authoring/browsing tasks for testers to complete.

### 6.3 Summary

This chapter evaluates the implementation of the story environment. The structure and implementation of the story environment has been compared to another web document annotation project, Annotea. Compared with Annotea the story environment has a simple and flexible structure and also no additional software requirements for the story users. However, unlike the Annotea implementation in the story environment the story browser only works with the documents created by the story authoring tool rather than any web page on Internet.

Some initial user feedback has been explained in this chapter. One of the important issues arising from user feedback is that a story cannot only act as information source for composing learning materials, but also the story itself can be a very useful medium to represent learning material. Learning material such as lecture notes or study guides can be represented as stories and improve the performance and efficiency of the self learning. It helps allowing students to catch the important information annotated by teaches and facilitates students to search in a big learning material by using different sort of metadata

---

## Chapter 7 Conclusion and Future Work

---

This chapter summarizes this thesis in addressing three issues. Firstly the thesis will be reviewed. Then the contribution of this research is concluded. The final part identifies future work for the continuation of this research project.

### 7.1 Thesis Summary

First of all, the literature relevant to developing the story concept was reviewed in this thesis. The original story concept has been compared with other learning object searching methods: Internet search, specialized knowledge collection and learning object repository with the criteria of information source, metadata, quality of control and authors/users. The story concept has advantages compared to other methods. It provides high quality learning objects collections, offers the subject context descriptions of collected learning objects and uses different types of metadata to search and customize stories.

In the second part of this thesis, the original story concept has been further developed. The story environment has been conceptualized and designed in this part. The components involved in the story environment have been described and explained one by one. Story metadata, story authoring tool and story browser were identified for development in this project.

The third part of this thesis is the development of the story metadata. LOM and Dublin Core are two metadata standards which can be metadata standards for the external story metadata. TEI, GDA and XCES are three document annotation metadata standards which have been reviewed to support developing the internal story metadata

specifications. Moreover, as suggested in the original story concept, a story should include some instructional or pedagogical information for story readers. So this thesis examines three different instructional strategies including Ausubel's expository teaching, Gagne's nine events of instruction and Problem Based Learning (PBL). PBL has been chosen to be included in the internal story metadata to offer users instructional information about the story narrative.

The fourth part of this thesis was to develop and implement the story browser and the story authoring tool based on the design of the story environment. The development of those two tools has been performed with requirements analysis, technology reviews, interface design and implementation.

## 7.2 Contributions

The core contributions of this research project are the further development of the story concept and story metadata and the development of the story environment and related tools.

### ➤ Story metadata

In this research project, the story metadata have been designed in detail. External story metadata annotate a story as a learning object from outside which helps users to identify stories from the story repository. Internal story metadata mark up a story from inside. Internal story metadata include domain dependent metadata and domain independent metadata. Domain dependent metadata are defined by story authors, Domain dependent metadata are varied with different domains or subjects. Domain independent metadata focus on the common features of the story narrative and can be applied to every story. The elements and attributes in domain independent metadata have been defined and formulated as a XML schema.

➤ Story environment development

The story environment has been developed in this research. Three major components are identified in the story environment.

- The story repository stores all the stories with their metadata.
- The story repository management system helps story authors to upload their stories to the story repository and assist the story readers in finding appropriate stories from the story repository using external story metadata.
- The user side applications to create and browse stories.

Further this research project defines the story browsing and authoring processes, and tools involved in the story browsing and authoring. Finally, two applications have been implemented in this research. The story browser helps story readers to read the story narrative and to look at a story from different perspectives. The story browser includes the functions of parsing stories, searching the metadata annotation, displaying the story structure and opening the multimedia learning objects. The story authoring tool is used by the story authors to author a story. The functions of the story authoring tool include parsing the user-defined domain dependent metadata schema, marking up the story narrative with metadata tags and generating the story XML document.

### **7.3 Future Work**

A number of issues have arisen for the further development of this project. These issues fall into the area of further conceptual development of the story environment, user testing of the story tools, and further development of the story authoring tool and the story browser.

➤ Further development of the story environment

This project only developed and implemented the story browser and the story authoring tool. The Story repository and story repository management have only been mentioned in high level description. The next step in this project is to develop and implement the story repository and story repository management system, so that these two systems and the story browser and the story authoring tool form a complete story environment. From the current point of view, the story repository can be developed as a database system which stores stories. The story repository management system can be a web based application which interacts between the story repository and the story browser/story authoring tool to allow users to upload and download stories.

➤ Further development of story authoring tool and story browser

The story browser and story authoring tool are implemented sufficiently to demonstrate the idea of the story concept and satisfy the main requirements of the story authoring/browsing based on the story environment. Further work is needed to provide more functionalities and better usability for the systems.

- Text decoration: in this research the story narrative can only be displayed as plain text or highlighted with different colors. From the user's point of view, it should include functions which enable users to change the font, size, and color of the story narrative. Potentially more metadata tags can be added to the internal story metadata to record the decoration information about the story narrative text. Current HTML already defined lots of tags to decorate text, such as `<b>` for bold. Those decoration tags in HTML can be included into internal story metadata in future.
- Domain specific metadata schema creator: as mentioned in the story concept, story domain specific metadata specification need to be defined by story authors. The form of the metadata specification is a XML schema. Although story authors can use other tools to define their story domain dependent metadata

XML schemas, it would be good for the story authoring tool to provide story authors with a visual interface to define their domain metadata elements. When story users want to create a new story, they can use a domain specific metadata schema creator built into the story authoring tool to create metadata tags for domain specific metadata directly.

- Creating stories: currently only two story examples have been made. In order to continuously evaluate and test the story authoring tool and the story browser, more stories should be created and used in teaching and learning environments to collect useful feedback.

## References

Ahmed, L., Ayers, D., Birbeck, M., Cousins, J., Dodds, D., Lubell, J., Nic, M., Rivers-Moore, D., Watt, A., Worden, R. & Wrightson, A. (2001). Professional XML meta edata. Birmingham, UK : Wrox Press.

Annotea (2004). Annotea Project.

<http://www.w3.org/2001/Annotea/>. Accessed 12/20/2004

APA Style (2004). Using American Psychological Association (APA) Format (Updated to 5th Edition). [http://owl.english.purdue.edu/handouts/research/r\\_apa.html](http://owl.english.purdue.edu/handouts/research/r_apa.html). Accessed 01/02/2005.

AWT (2005). The AWT in 1.0 and 1.1. <http://java.sun.com/products/jdk/awt/>. Accessed 02/02/2005.

Asusbel, D.P. (1978). Educational psychology – A cognitive view second edition. New York, 1978.

Bones, B. (2004). Search Engines.

<http://www.sc.edu/beaufort/library/pages/bones/lesson1.shtml>. Accessed 13/08/2004.

Brody, R. (2003). Information ethics in the design and use of metadata. IEEE Technology and Society Magazine, 22(2), Summer 2003, pp. 34-39.

Boud, D., & Feletti, G. (1997). The challenge of problem-based learning. London:Krogan Page.

CanCore (2004). <http://www.cancore.ca/en/>. Accessed 05/07/2004.

CAREO (2004). CAREO Overview & Goals.

<http://www.careo.org/documents/overview.html>. Accessed 12/07/2004.

Cooktop (2004). <http://www.xmlcooktop.com/>. Accessed 02/04/2004.

Daniel, G. (2004). Learning Object Repositories. WWW tools for teachers Newsletter, 3 May 2004. <http://www.e-learningcentre.co.uk/eclipse/Resources/contentmgt.htm>. Accessed 04/03/2005.



- Clarke, C. (2005). Towards a unified e-learning strategy. <http://www.dfes.gov.uk/consultations/downloadableDocs/towards%20a%20unified%20e-learning%20strategy.pdf>. Accessed 24/02/2005.
- DC (2004). Dublin Core Metadata Initiative. <http://www.dublincore.org/groups/education>. Accessed 15/07/2004.
- DOM (2004). Document Object Model (DOM). <http://www.w3.org/DOM>. Accessed 09/10/2004.
- EDNA (2004). EdNA Online. <http://www.edna.edu.au/edna/page1.html>. Accessed 18/05/2004.
- Flanagan, D. (2002). JavaScript: The Definitive Guide, 4th Edition. O'Reilly.
- Gagne, L., Briggs, L., & Wager, W. (1986). Principle of instructional Design. Holt, Rinehart and Winston, 4th edition.1985.
- GDA (2004). Global Document Annotation. <http://www.i-content.org/GDA/>. Accessed 07/05/2004.
- Gill, T. (2000). Metadata and the World Wide Web. Getty website, Introduction to Metadata. [http://www.getty.edu/research/conducting\\_research/standards/intrometadata/2\\_articles/gill/index.html](http://www.getty.edu/research/conducting_research/standards/intrometadata/2_articles/gill/index.html). Accessed 09/07/2004.
- Google (2005). <http://www.google.com>. Accessed 18/11/2004.
- Gosling, J., Joy, B., Steele, G. & Bracha, G. (2005). The Java Language Specification, Second Edition. [http://java.sun.com/docs/books/jls/second\\_edition/html/intro.doc.html#22191](http://java.sun.com/docs/books/jls/second_edition/html/intro.doc.html#22191). Accessed 12/11/2004.
- Harold, E. R. (2001). Processing XML with Java: A Guide to SAX, DOM, JDOM, JAXP, and TrAX. Addison-Wesley.
- Haynes, D. (2004). Metadata for information management and retrieval. London: Facet Publishing.
- Heinrich, E., Andres, F. (2003a). Introducing an Interpretive Layer between Document Collections and Learning Material. EDS. Proceedings of Ed-Media2003 World Conference on Educational Multimedia, Hypermedia & Telecommunications, pp1022 - 1027, Norfolk, USA, Association for the Advancement of Computing in Education.

- Heinrich, E., Andres, F. (2003b). Enriching Document Collections through the writing of 'Stories'. Proceedings of ICALT2003, IEEE International Conference on Advanced Learning Technologies, pp101 - 105, Los Alamitos, USA, IEEE.
- Hodgins, H. W. (2005). The future of Learning objects. In the instructional Use of Learning Objects.(Online version). <http://www.reusability.org/read/#1>. Accessed 25/05/2005.
- ID2 Research Group (1997). Instructional Strategies that Teach. CBT Solutions Nov/Dec 1997. <http://www.id2.usu.edu/Papers/Consistency.PDF>. Accessed 03/11/2004.
- Ide, N & Romary, L. (2004). XML Support for Annotated Language Resource.
- Jacobs, M. (2004). Component Display Theory.  
<http://www.gsu.edu/~mstsw/courses/it7000/papers/componen.htm>.  
Accessed 29/05/2004.
- Java (2004). Java Language Overview.  
<http://java.sun.com/docs/overviews/java/java-overview-1.html>. Accessed 30/10/2004.
- Kahan, J., Koivunen, M. (2001). Annotea: An Open RDF Infrastructure for Shared Web Annotation. Proceedings of the tenth international conference on World Wide Web. HongKong.
- Kinji Ono. (2002). A Global Multimedia Repository Concept for Digital Silk Roads Studies - Heritage of Historical and Cultural Resources in the Digital Age.  
<http://pnclink.org/annual/annual2002/pdf/0920/3/p200301.pdf>. Accessed 04/03/2005
- Kruse, K. (2004). Gagne's Nine Events of instruction: An introduction.  
[http://www.e-learningguru.com/articles/art3\\_3.htm](http://www.e-learningguru.com/articles/art3_3.htm). Accessed 20/05/2004.
- LOM (2004). IEEE Learning Technology Standards Committee (LTSC).  
<http://ltsc.ieee.org/w12g>. Accessed 22/04/2004.
- LTSC (2004). Learning technology standards committee.  
[http:// http://ltsc.ieee.org/wg12/](http://http://ltsc.ieee.org/wg12/). Accessed 23/04/2004.
- MatTalha, N. (2004). Metadata Management System.  
[http://students.washington.edu/jtennis/dconf/Paper\\_24.pdf](http://students.washington.edu/jtennis/dconf/Paper_24.pdf). Accessed 07/07/2004.
- MERLOT (2004). <http://www.merlot.org>. Accessed 11/06/2004.

MSDN (2004). Understanding XML

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnxml/html/UnderstXML.asp>. Accessed 14/09/2004.

MSXML (2004). MSXML.

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xmlsdk/html/xmmscXML.asp>. Accessed 01/10/2004.

Neil, B. (1999). The XML companion. Harlow: Addison-Wesley.

PBL (2004). Problem Based Learning. [http://www.fsw.ucalgary.ca/docs/teaching/slide\\_shows/text\\_versions/problem\\_based\\_learning.rtf](http://www.fsw.ucalgary.ca/docs/teaching/slide_shows/text_versions/problem_based_learning.rtf). Accessed 25/05/2004.

Richards, G., McGreal R., Friesen, N. (2002). Learning Object Repository Technologies for TeleLearning: The Evolution of POOL and CanCore.

<http://209.87.57.212/upload/Richa242Learn.pdf>. Accessed 20/07/2004.

RDF (2004). <http://www.w3.org/RDF/>. Accessed 09/08/2004.

Shabajee, P. (2002). Primary Multimedia Objects and 'Educational Metadata'. D-Lib Magazine, Vol8, No6, ISSN 1082-9873.

Sall, K B. (2002). XML family of specifications : a practical guide.

Boston : Addison-Wesley, c2002.

Smith, J. (2004). E-Learning: When and Why. Intercom, p22 Sep/Oct 2004.

Swing (2005). Swing Introduction. <http://java.sun.com/developer/Books/swing2/>.

Accessed 01/03/2005.

TEI (2004). The Text Encoding Initiative. <http://www.tei-c.org>. Accessed 01/04/2004

TEI U5 (2004). [http://www.tei-c.org.uk/Lite/teiu5\\_en.html#U5-pln](http://www.tei-c.org.uk/Lite/teiu5_en.html#U5-pln).

Accessed 03/04/2004.

XCES (2004). <http://www.xml-ces.org>. Accessed 10/04/2004.

XML basic (2004). Introduction to XML.

[http://www.w3schools.com/xml/xml\\_what.asp](http://www.w3schools.com/xml/xml_what.asp). Accessed 10/12/2004.

XML spy (2004). [http://www.altova.com/products\\_ide.html](http://www.altova.com/products_ide.html). Accessed 02/09/2004.

XSLT (2004). <http://www.w3schools.com/xsl/default.asp>. Accessed 28/08/2004.

- Wiley, D.A. (2000). Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. [www.elearning-reviews.org/.../learning-objects/2001-wiley-learning-objects-instructional-design-theory.pdf](http://www.elearning-reviews.org/.../learning-objects/2001-wiley-learning-objects-instructional-design-theory.pdf). Accessed 01/05/2005
- Zhang, D., Zhao, J., Zhou, L. & Nunamaker, J. (2004). Can e-learning replace classroom learning. *Communications of the ACM*, 2004, Volume 47 Issue, pp75 -79