School of Computer Science and Engineering

Faculty of Engineering

The University of New South Wales

Requirements to theses submitted in the Faculty of Engineering

by

Yee Hang Choo

Thesis submitted as a requirement for the degree of

Bachelor of Engineering in Computer Engineering

Supervisor: Prof. Maurice Pagnucco Student ID: z5157656

Abstract

The current education industry has popularised online education service to provide educational resources on a global scale. For this reason, many LMS systems have been developed to manage and distribute education resources known as learning objects (LO). However, at present, the learning objects in current LMS systems do not offer the ability to repurpose and reuse them easily. Although most LMS provides a way to import and reuse previous course's learning objects, these methods are usually limited in flexibility. This work presents "Meta LMS", a learning management system that supports previously mentioned repurposable and reusable learning object. Among the available approaches, this thesis will focus on metadata building based on semantic extraction technologies from Semantic Web.

Acknowledgements

The work of theis has been inspired by Prof. Maurice Pagnucco and extradionary suggestion by Dr John Shepherd. I also would like to thank my thesis group during my time when developing and designing Meta LMS. Specifically, (Tammy) Xue Qing ZHONG, Suphachabhar Nigrodhananda, Ki Fung Kelvin Ting and Apoorva Bhagchandani. They inspired me and make sure that we can come together to develop an amazing world-class game changing LMS.

Glossary

LMS A platform to deliver educational material. E.g: Moodle, CanvasLMS, WebCMS3, Decebo, Adobe Captivate Prime,

Learning Object "Any entity, digital or non-digital, that may be used for learning, education or training"; definition by IEEE.

Concept The smallest unit of entity to describe what a Learning Object is.

Dependencies The context of we are referring to is concepts might have dependencies on other concepts in order for an individual to completely understand a concept.

Contents

Chapter 1

Introduction

The growing demands for both quantitative and qualitative improvements around the world have challenged the education institution to provide more efficient use of educational resources. As technology evolves, education institution has also made steady progress in the use of computing and web services technology to remain competitive in the industry. One of the approaches to delivering industry standards of teaching is to offer an online service with a type of system that organises and manages all perspective of education administration and resources, which is known as learning management system (LMS).

Over the years, the modern institute has evolved continuously to attempt reusing and repurposing existing course resources to save cost. However, the exponential growth and dynamics of educational resources on LMS has posed many challenges for this attempt, namely lack of efficient resource retrieval causing inflexibility in translating learning object into another course and the also problem of dependencies between learning objects. Thus, in this paper, we propose a solution by developing a Meta Learning Management System (Meta LMS). The core of Meta LMS aims to connect learning object data (e.g: filename, description), external data sources like a knowledge base and internal semantics definitions of learning objects to resolving the challenges mentioned above.

1.1 Goals and Objectives

The main thesis goal is to devise an architecture to solve the task of efficient resource retrieval and learning object dependencies. The vision of the final architecture should be able to improve and extend the capabilities of current existing LMS. Thus, the core Meta LMS system architecture shall cover:

1. A proper metadata to classify and express existing learning objects
2. A framework for content re-purposing. This can be supported by metadata devised in this project
3. A framework for dealing with different metadata standards and input sources

The rest of the thesis is organised as follows. Chapter 2 presents mode details of the problem in state of the art regarding the current LMS technology. Chapter 3 introduces the selection of techniques used in Meta LMS. Chapter 4 provides statistical results and discussion regarding Meta LMS's implementations. Chapter 5 concludes the report and recommends some future work.

Chapter 2

Background

2.1 Learning Management System

LMS is a web-based software application that supports many educational administration tasks and promotes the ability of content sharing between learning entities and educational service providers. LMS takes learning objects such as lecture content, quizzes, assignments, course discussions, etc., and organises them into chunks known as modules. Now, modern LMS has evolved to integrate and support features such as personalised student learning and comprehensive course analytic [1]. Figure 2.1 summarises all other common elements of LMS features that exist in state of the art.

2.2 Existing LMS Survey

In this section, we will look at a few popular 'classic' LMS that has been used globally. We survey its existing features and the degree of it able to do content re-usability & purposing and information retrieval.

Moodle. Moodle is an open-source LMS and is one of the popular LMS used in the education industry. Their pedagogical approach is towards education are constructivist

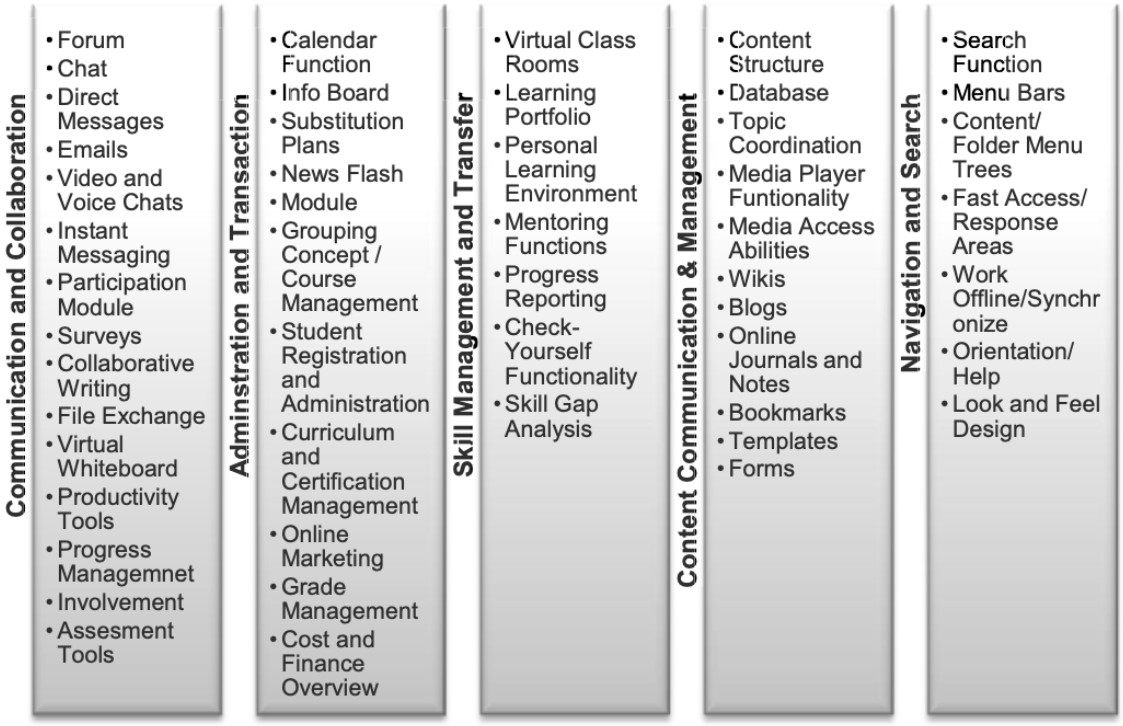

Figure 2.1: Summary of Common LMS Features [1]

and social constructionist education [1], thus most of their work involve around having a variety of plugins and features that can support as many external integration and existing metadata standards like xAPI and SCORM, marketing themselves to offer highly flexible and fully customisable learning platform to users. In Moodle, it is easy to reuse an existing course or a part of a course's content. Moodle does this by "backing up" the course into the system and split each component of the course into what Moodle called "activities". Users can restore to reuse their content by selecting the content they have back up in the system.

Canvas LMS. Canvas LMS is also an open-source LMS and is currently one of the fastest-growing platforms in the LMS industry. It became popular due to its modern-looking interface and ease of use compared to other LMS. In terms of its functionality, it is very similar to Moodle. Canvas LMS also allows the reuse of existing courses and part of the course [2]. Canvas LMS also allows user to easily repurpose existing courses into any other course using their "copy courses" feature, as shown in Figure 2.2 and 2.3.

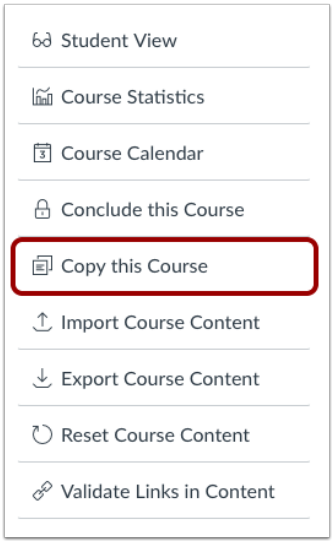

Figure 2.2: Copy Course Feature in Canvas LMS: Getting Started with Copy Course Feature

Adobe Captive Prime Adobe Captive Prime (ACP) is an LMS that adopts AI for

[1]https://docs.moodle.org/310/en/Philosophy

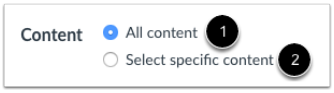

Figure 2.3: Copy Course Feature in Canvas LMS: Selecting Migration Mode

social learning and content curation. [2]. Adobe Captive Prime keeps a course catalogue of its learning object and uses a direct-text search engine to help users retrieve their learning objects. Since Adobe Captive Prime evolved from being an LMS authoring tool to a fully functional LMS, there is a well-defined structure that Adobe Captive Prime has created and used in their system to help users to easily migrate content within the system.

2.3 Problem Statement

2.3.1 Analysis of current LMS

Most classic LMS offers a great deal of re-usability and re-purpose features. Many examples are given above partition their courses into smaller pieces to allow a section of a course to be re-context into another format. Many LMS also allow the ability to import and export courses and learning objects to extend these functionalities.

One thing that is still lacking in most classic LMS is **efficient resource retrieval**. Since there is widely different course content due to many different pedagogical approaches, content formats and standards, the current search engine technology's ability is somewhat lacking as only full-text search is supported [3]. Resource retrieval becomes almost impossible without some proper description of the resource. As a result, an existing learning object is difficult to find and is reducing the chances of it translating into another course.

Another problem that is still not done well in the current classic LMS is the **management of learning objects around their knowledge dependencies**. Almost all the

[2]https://www.adobe.com/au/products/captivateprime/features.html

learning objects will have some knowledge dependencies. To ignore these dependencies will cause a knowledge gap in the learning material and confuses learners. As a result, this problem also has contributed to reducing the reusability of the learning object.

2.4 Possible Solutions

2.4.1 Reusability

Artificial Intelligence

AI Integration offers a way to analyse both learning objects and users to understand the content of each learning objects and user behaviour. The enhanced understanding can be used to improve the discoverability of learning objects for the users by its recommendation features and user behaviour recognition. These can be achieved by applying intelligence in the AI field like Knowledge Representation and Reasoning, Machine Learning, Natural Language Processing and Pattern Recognition.

Meta Search Engine

Efficient resource retrieval can be improved by shifting the focus towards the search engine itself. An appropriate search engine that can utilise documents ranking algorithms like PageRank and other principles can group documents based on these algorithms and provide satisfactory results to the user. However, this often results in high recall and precision retrieval of documents and users do not always get relevant results from their query due to the search engine being highly sensitive to vocabulary. One recent proposal to solve this is using a meta-search engine [4]. Meta-search engine gives a personalised search according to the user's academic background, search history and context. It collects and builds the metadata around the user's search pattern, decisionmaking strategy and third party resources like Google or Bing [5] so that users can extract information faster.

2.4.2 Dependencies

Semantic Web

Semantic Web can enrich information contain inside the learning object, Many current LMS that adopted Semantic Web has used it to provide a personalised user learning paths. They build data models around learners to provide a personalised adaptation of LOs sequencing; and whenever a module is modified (e.g deleted, added, learned, changed to optional), the semantic web will automatically reconfigure the internal learning and course path of a user. Such technology would bring the utility to help to determine and keep track of the dependencies of learning objects

Topic Trees

A topic tree is just like a tree in the computer science literature where it is an abstract data type that simulates a hierarchical tree structure, with a root value and subtrees of children with a parent node, represented as a set of linked nodes.

Charles Sturt University(CSU) Topic Tree has given inspiration to this project for how to solve some dependencies issues. CSU Topic Tree aims to provide self-directed learning through the on-demand online curriculum to students, where it follows the traditional Netflix "binge" model for students to consume its content [11].

Figure 2.4: Example of section of topic tree [6]

CSU topic tree consists of individual fine-grained learning activities called "topics". As shown in figure 2.4, each topic operates on a three-hour chunk of learning modules for students to complete. As the name suggests, the topics are arranged into a tree-like structure or more formal a directed acyclic graph where each node represents a topic and each edge represents the prerequisite relationships between topics.

We can see that if we group the core ideas or knowledge of the learning objects into a

basic format like in the CSU topic tree. It is possible to organise the learning objects for dependencies checking.

2.5 Relevant Literature Details

Concerning our problems and challenges, we discuss the relevant technical details that exist in the state of art. The general element of approaches can be broken down to their task as follows:

- Document Processing: How do we deal with and process input sources?
- Semantic Extraction: How do we gather semantic from processed documents?
- Metadata Building: How do we store extracted data that suit both our purpose and requirement?

2.5.1 Document Processing

There are diverse formats that can exist in an LMS environment. This can include: unstructured format(RTF, TXT), web documents (HTML), semi-structured format (DTD, RDF, XML) and educational standards (SCORM).

The general core steps of pre-processing are as follow:

1. Extraction & Cleansing: Aims to convert these differing formats into machine-processable plain text. The techniques used for this step varies based on the input format. For HTML, off-the-shelf parsers exist like HTML Tidy 8. For images, video or scanned documents, Optical Character Recognition tools are used. For most popular general-audience document format such as PDF files, presentations(PPT) and spreadsheets, software tools including Apache Tika 9 which extracts metadata and text can be used.

2. Tokenisation: Aims to break down the plain text into a sequence of atomic tokens while preserving the order of the text.

3. Sentence Segmentation: Aims to identify the start and end of a sentence and organise them into self-contained sentences. The simplest technique used is by analysing punctuation marks. Existing tools to help with this process includes a python tool called segtok [3].

4. Part-Of-Speech (POS) Tagging: Aims to analyse and label the grammatical category to each word in a sentence. Many techniques exist and are often adopted by the NLP community [7]. This includes Rule-Based Methods, Transformation-Based Learning, Markov Model Taggers and other machine learning techniques.

5. Structural Parsing: Aims to parse text into a syntactic structure with grammatical roles in mind. A common technique is to use a Probabilistic Context-Free Grammar (PCFG). A full description of the implementation of PCFG can be found in this paper [7].

2.5.2 Semantic Extraction

Semantic Web

The vision of the semantic web is to allow data to be easily shared and reused on the web. Therefore, concepts from the semantic web are very closely related to the task at hand and thus in this section we discuss the technology used in semantic web to explain the subsequence methodology used later in this thesis.

The semantic web is an initiative led by Tim Berners-Lee to extend World Wide Web standards. The initiative's goal is to solve the problem of web content which is not machine-processable. The technology approaches the problem by tackling the internal data to enable encoding of semantics with data. This has led to the forming of an

[3]https://github.com/fnl/syntok

explicit definition of terms and relation in data, or formally known as ontology. Ontology models the structure of knowledge and to organise information that enhances the contextual meaning of data [8]. In general, ontology consists of both terms and relationships between terms. The terms refer to important concepts (classes of objects) of the domain while relationships denote hierarchies of classes. The hierarchy specifies if every object in A exists in B, then A is the subclass to A [9].

In the current state of the art, the most prominent ontology language exists Resource Description Framework (RDF) and Web Ontology Language (OWL) [9]. RDF is a data model that uses an object-attribute-value triple as its basic building block. As an example, we want to describe that "An apple is a type of fruit", the statement in RDF would be "Apple-IsA-Fruit". RDF has a known limitation in which it is unable to express a more complex semantic definition of objects. This includes local scope of properties, disjointness of classes, cardinality restriction and boolean combination of types [9]. Thus to compromise for these limitations, a richer ontology language is created, named OWL.

Manual Ontology Development In Semantic Web

Most manual ontology development uses ontology editing tools such as Protégé [4]. The summary of the manual ontology development process in this section follows closely to what Noy and McGuinnes have presented in this paper [10]. The main steps in ontology development are as follow:

1. Determine Scope. Ontology is defined for specific domains and purposes. Therefore ontology must be defined according to the scope and domain of task. By asking competency questions like "What is the domain that the ontology will cover", Who is going to use the ontology, etc., can help to limit the scope of ontology.

[4]protege.stanford.edu

2. Consider reuse. Considering other people's work can save time and extend existing sources for our domain. There are a few reusable ontologies libraries on the web, namely: Ontolingua ontology library [5], Swoogle, SemWebCentral and DAML ontology library 13

3. Enumerate terms. Enumerating terms helps to define all relevant terms that could appear in the ontology. Common knowledge engineering tools (e.g. grid analysis and laddering)can be helpful in this process.

4. Define taxonomy. Terms that are defined in the enumerate process must then be organised in a taxonomic hierarchy. There are several ways to define this hierarchy, either top-down (Concepts to classes), bottom-up or both.

5. Define properties. Classes themselves often do not provide enough information. Thus additional properties of defined classes must be added to explain the internal structure of concepts.

6. Define facets. This step helps to further enhance the ontology by enriching previously defined properties such as cardinality, required values, relational characteristics.

7. Define instances. In this step, ontology is populated by a set of instances.

8. Check for anomalies. Finally, we check and refine for any inconsistencies that exist in the ontology itself.

Semantic Annotation

Semantic annotation goal is to annotate documents with entities using a technique called Entity Extraction & Linking. Entity Extraction & Linking (EEL) is a two-step process that identifies and disambiguates mentions of entities in a document [11]. An entity in this context is any named object that exists in the document In the identification process, the entity mentions in the text are marked as candidates based

[5]http://www.ksl.stanford.edu/software/ontolingua/

on NER or a dictionary. This can be token-based, Part-of-speech-based, parser-based or machine learning methods. On the other hand, the disambiguation process then aims to remove ambiguous KB labels by finding the best match mentions in the dictionary. The best match criteria can be based on multiple features including mention-based, keyword-based, graph-based, category-based and hybrid system like ensemble systems (e.g., WESTLAB).

EEL can also have the involvement of a popular machine learning technique, Natural Language Processing which is an AI-based approach that sees words as atomic units. A common technique used to process these units as a feature is linear statistical models like n-grams [12]. While a pre-trained model exists off the shelf, these techniques often require training on a specific corpus to adapt the algorithm to the domain. There exists an off the shelf tool named Named Entity Relation (NER), however often this tool has limited types of entities that it can extract. Therefore, the general method to identify entities is to use a knowledge base (KB) such as Wikipedia, DBpedia or Freebase, as a dictionary to guide entity extraction.

Ontology Learning

While manual ontology development exists, there is also an automated version of ontology development called ontology learning. Ontology learning is the automatic or semi-automatic acquisition of ontologies using Concept Extraction & Linking (CEL). Concepts refer to a conceptual grouping of elements (e.g. Topics) in the document.

The first step in CEL is to collect domain-relevant terms from the text. The methodology used here is similar to what is done in EEL, but there are some subtle differences between them. CEL focuses on extracting concepts rather than entities in EEL and concepts in CEL are hierarchical. Furthermore, the process in extracting concepts is a bit more complicated, particularly in the syntactic patterns used to search for terminology. Therefore, various systems exist and are proposed for extracting terms such as TermeX, YaTeA and KEA. The second step in CEL is filtering. Filtering selects best concepts using linguistic or statistical features of the text that reflects the scope and

domain. For linguistic features, we take the lexical or syntactic aspects of the term as a feature. For example, the most basic feature would be the number of words forming the term. Other linguistic features can also include POS-tagging described in earlier reports. As for statistical features, we can measure the relevance of the term to the domain at hand. The most common way to do this is using term frequency-inverse document frequency(TF-IDF) which weights the importance of a term to a document or corpus. The third step is the concept of hierarchy induction. This is where we define the taxonomy of extracted terms just like in manual ontology development except. The common approaches in the literature are extracting taxonomies from machine-readable dictionaries, FCA, Distributional Similarity and co-occurrence analysis [12].

The optional final step is topic modelling. Topic modelling links extracted terms from previous steps and link them to referenced KB/ontology (the domain). Topic Modeling process involves applying statistical analysis over the term-concept mapping. For example, one can take the ratio of extracted terms mapped to the subconcepts and indicate the relevance of concept [13]. Other approaches include applying traditional semantic matching methods such as pLSA, LDA or Latent Semantic Analysis (LSA) [12].

2.5.3 Metadata Building

Metadata building refers to both construction of metadata and compiling definition from the previous process. Metadata is data that describes other data. Metadata can have a collection of keywords, comments and attributes that gives essential information about resources to both machines and humans to understand how to categorise and manage the data [3].

Educational metadata provides information regarding their support and learning objects. This metadata can be used by educational and pedagogical professionals to establish a well-designed education offering, and also by the learners searching for any education information [3].

There are several important uses of metadata in education systems. We summarise the following main roles:

- Information discovery and retrieval
- Facilitates reusability of learning objects
- Assist interoperability of education resources [14]
- Enables personalised learning for an individual learner [15]
- Provides analysis of courses, resources and curriculums for comparability [3]

Educational Metadata Standards

There are metadata elements grouped into sets to cater to the needs in the eLearning industry [3]. These are known as metadata standards. Metadata standards help to ensure consistency of object content; mainly, it defines the data structure, education activities, communication protocols, distribution format, etc. [15]. Many different metadata standards have been developed for a variety of user requirements. For example, IMS and SCORM have developed an XML-based interoperable specification for launching and sequencing learning objects. A summary of metadata standards is explained and evaluated in [3].

The existing metadata standards have focused on well-defined system implementation of LO, interoperability across different LMS and their run-time sequencing. Still, there are no semantic inherent to provide a fine-grained view of each learning object's knowledge in educational activities. This has caused the current metadata standards to have difficulties suggesting any personalised relationship of content to the author. According to Samsuzzaman [16], if these LOs can be framed by additional semantics, then this e-learning system will introduce the capabilities of reusable and adaptable educational content. As a result, this has led to forming the central origin and idea of building a meta LMS system.

There exist other representation formats that exist in the literature besides OWL and RDF. If one needs to have a powerful expressiveness of their ontology, the most generic go-to is the Simple Knowledge Organisation System (SKOS). This format aims to standardise and model more general forms of semantic relations than OWL/RDF format. Another possible semantic representation format that has been proposed is LEMON 15. This format links semantic definition with linguistic properties. Other hybrid formats have also been proposed using SKOS and LEMON.

Simple Knowledge Organization System (SKOS)

SKOS is an OWL/RDF specifications to the representation of a set of terms. SKOS is part of the Semantic Web family of standards built upon RDF and RDFS so it can encode knowledge in an interoperable way. SKOS is designed exactly to describe concept schemes, *concept* is its basic structural element. A SKOS concept can be viewed as a unit of knowledge. Therefore, concepts are abstract entities [17]. SKOS introduces the class skos:Concept to indicate that a particular term is a concept. The individuals of the skos:Concept class can belong to a specific concept scheme. A concept scheme is expressed through the skos:ConceptScheme class. Also, they are documented with various types of notes and interconnected with semantic relations through informal hierarchies. To express these characteristics, the SKOS model uses a set of properties, firstly to define a concept itself and secondly to relate it with other counterparts in a concept scheme. Table 2 summarizes available SKOS properties, organized into categories according to their purpose, and gives a brief description of their usage. In Meta LMS, SKOS will be used as a starting point to construct our metadata. The core idea of SKOS having the basic element, *concept*, ensures that every learning object can be broken down into its atomic pieces, thus creating a very atomic view of the learning objects.

2.5.4 Computer Science Ontology (CSO)

Computer Science Ontology(CSO) [6] is a large-scale research topic ontology mainly in the field of Computer Science [17]. The ontology is automatically generated by running Klink-2 algorithm over a large corpus of scientific papers based on Rexplore dataset. Some relationships in the ontology were also revised manually by experts for any abnormalities in the ontology. CSO can characterise higher-level research areas utilizing hundreds of sub-topics and related terms, which enables to map very specific terms to higher-level research areas. Some methodologies that have used in CSO are EEL and CEL.

There are several application has been implemented using CSO ontology. For example, Smart Topic miner (STM) uses CSO to classify scholarly publications. STM analyses in real-time a collection of publications and return a description of the given corpus like a taxonomy of research topics drawn from a large scholarly ontology, a set of classification tags, analytics of terms and topics to of each paper. The evaluation of STM has a high score in topics quality, average 4.0 (from a scale 1-5). In terms of publication coverage, the topics produced by STM were much better than enriched keywords (keywords extracted from titles and abstracts) and keywords defined by authors [18]

We will see later that CSO is adopted as an external ontology to support MetaLMS in Method Section of the report. Besides being a computer science field ontology, CSO has two main advantage. The first is that CSO has a large number of terms that had been gathered over the past years. As a result, it has powerful capabilities to classify very high-level concepts from learning objects. The second is that CSO is constantly automatically updated using Klink-2, This means that CSO can include recent novel research areas to help Meta LMS in unknown areas of metadata definitions.

[6] https://cso.kmi.open.ac.uk

2.5.5 Resnik Distance

To distinguish between concepts, a similarity calculation is used. It can aid in determining whether a concept is the same, discover mapping between ontology entities and estimate semantic orientation [19]. Resnik Distance one of the examples of similarity calculation exists in the state of art. It is a lexical similarity measure based on the information content measurement [20]. It represents core content, concepts or terms, as nodes and builds them into a Directed Acyclic Graph (DAG) and calculates the similarity using the formula below:

$$sim_{Resnik}(t_1, t_2) = \max_{c \in S(t_1, t_2)} [-\log p(t)]$$

where $S(t_1, t_2)$ is the set of common ancestors of terms t_1 and t_2 in the ontology, ranging in $[0, \infty]$ and $p(t)$ can be any function that calculates the probability of finding the concepts in the DAG, such as $\frac{freq(t)}{maxFreq}$.

Intuitively, Resnik distance based on the information content that two concepts share [21]. It defines the distance between two disjunctive sets of concepts as the minimum path length from any element of the first set to any element of the second. This approach has a drawback. When a concept has multiple inheritances, it only considers one of the possibilities for each concept and neglect other inheritance in the DAG.

Table 2. The SKOS core vocabulary.

SKOS Term	Description
skos:Concept	*An abstract idea or notion; a unit of thought*
Concept Schemes	
skos:ConceptScheme	*A concept scheme in which the concept is included*
skos:inScheme	*Relates a resource to a concept scheme in which it is included*
skos:hasTopConcept	*A top level concept in the concept scheme*
skos:topConceptOf	*Is top concept in scheme*
Lexical Labels	
skos:prefLabel	*The preferred lexical label for a resource, in a given language*
skos:hiddenLabel	*A lexical label for a resource that should be hidden when generating visual displays of the resource.*
skos:altLabel	*An alternative lexical label for a resource*
Semantic Relations	
skos:broader	*A concept that is more general in meaning*
skos:narrower	*A concept that is more specific in meaning*
skos:broaderTransitive	*Has broader transitive*
skos:narrowerTransitive	*Has narrower transitive*
skos:related	*A concept with which there is an associative semantic relationship*
skos:semanticRelation	*A concept related by meaning*
Mapping Properties *(to other concept schemes)*	
skos:exactMatch	*Has exact match*
skos:closeMatch	*Has close match*
skos:broadMatch	*Has broader match*
skos:narrowMatch	*Has narrower match*
skos:relatedMatch	*Has related match*
skos:mappingRelation	*Is in mapping relation with*
Notations	
skos:notation	*A string used to uniquely identify a concept within the scope of a given concept scheme*
Documentation Properties	
skos:changeNote	*A note about a modification to a concept*
skos:definition	*A statement or formal explanation of the meaning of a concept*
skos:editorialNote	*A note for an editor, translator or maintainer of the vocabulary*
skos:example	*An example of the use of a concept*
skos:historyNote	*A note about the past state/use/meaning of a concept*
skos:note	*A general note*
skos:scopeNote	*A note that helps to clarify the meaning and/or the use of a concept*
Concept Collections	
skos:Collection	*A meaningful collection of concepts*
skos:OrderedCollection	*An ordered collection of concepts, where both the grouping and the ordering are meaningful*
skos:member	*A member of a collection*
skos:memberList	*An RDF list containing the members of an ordered collection*

Figure 2.5: Summary of SKOS Properties[17]

Chapter 3

Methods

3.1 Meta LMS System

3.1.1 System Overview

The main idea of the methodology is to extract as much information (e.g semantic definition, relation and annotation) from the learning objects itself and saving them as metadata in the system. Then by using these semantic definitions, we can provide fine-grained information of each learning object that the search engine can easily search for and recommend. Besides, dependencies of the learning objects can also be known by storing the required concepts from each learning objects (Like a tagging system).

To store the metadata, we use some inspiration from how Semantic Web enriches and store their metadata. As discussed in the background, one way to define and store the learning objects semantic definition can be done in SKOS format. SKOS describes the knowledge atoms as *concepts*. With further annotations, SKOS is also able to express characteristics and semantic relationship with other concepts. This will help to classify and describe the semantics from a Learning object.

3.1.2 Metadata Extraction Process

With a way to store our metadata, we turn our attention to how to extract the information and turn them into metadata. Figure 3.1 illustrations the overall metadata extraction process.

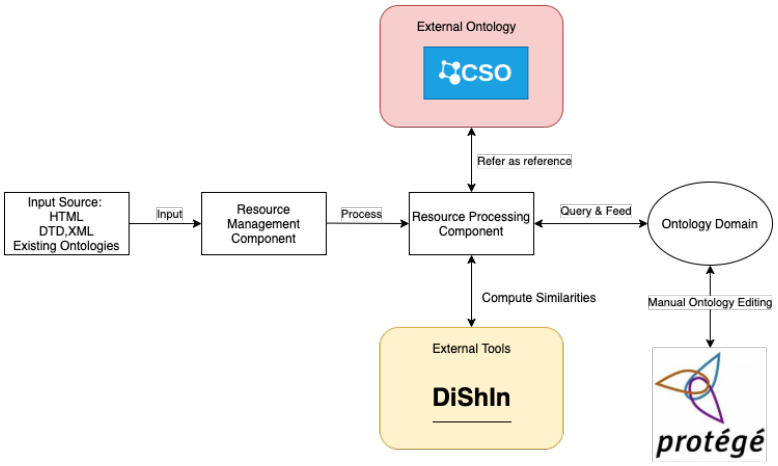

Figure 3.1: Metadata Extraction Process Overview

Resource Management Component. Resource Manage Component manages all input sources (e.g. HTML, XML, RDF, SCORM, xAPI, etc) that come into the system. For existing education standards such as SCORM and xAPI, the system would call related Saas APIs (SCORM Cloud) to extract the metadata them and store them directly to the database; while other document text will get filtered and then chooses resource processing methods for the resource processing component.

Resource Processing Component. Resource Processing Component extracts metadata from documents that get filtered by Resource Management Component. Most of the documents will first be converted into a full-text form. With PDF files, we slice each page as a mean of partitioning. Then we feed each of the pdf pages into CSO to extract semantic definition from them. Extracted Semantic will be used to populate the ontology domain. In the final step of Resource Processing Component, we re-calculate the similarity of all concepts inside the ontology domain using an external library called DiShin. DiShin is an ontology-based-similarity-calculator python library. The metric that we used for the calculator is Resnik Distance.

Ontology Domain. Ontology domains stores all the semantic relations between all the concepts.

Ontology Editing. Further pruning and editing by an ontology engineer to check for any anomalies and refine the ontology. The proposed tool for this is Protégé as it supports RDF/OWL ontology editing.

3.2 System Architecture

Meta LMS system consists of two main components: Application Architecture and Ontology Architecture. Application architecture describes our full-stack technology for our LMS. Meanwhile, Ontology architecture denotes the architecture to perform semantic definition from any document. This includes the metadata proposed to describe all the learning objects semantic definitions. More details in the Application Architecture section and Ontology Architecture section respectively.

3.2.1 Application Architecture

Table 3.1 describes each individual components that exist inside the Meta LMS architecture. As seen in Figure 3.2, Meta LMS consist of a standard full-stack technology to deliver its content. The design of this architecture is motivated by its high level of flexibility in terms of application integration. As one of the visions of the project is to extend the metadata to existing, any LMS can directly call our API to use our service without impacting their application with this architecture.

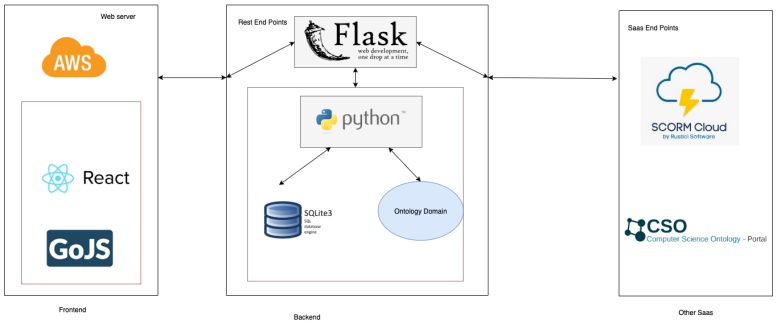

Figure 3.2: Meta LMS Architecture

Database and Course Structure Design

The current database design for Meta LMS can be illustrated in Figure 3.3. The database will store information regarding the learning objects, structure of courses, the metadata to link physical learning object to the ontology domain.

As you might notice, we structured our course internal structure into component and sub-components. Course Component can also be referred to as section types. They are meant to represent the type of section for a given course, i.e: Lectures, Lab, Extra Training, etc. The sub-component is the collection of learning objects the course distributes, such as Slides, Code Demo and Short Presentation Clips. This structure is aim to provide unrestricted course structure as there are many courses in the state of the art that has varying types of course components. Currently, each course is on a weekly basis to imitate most universities course offering structure.

3.2.2 Ontology Architecture

Metadata

As mentioned in the system overview, we are using SKOS as a basis to construct our metadata. Table 3.2 shows all the ontology vocabulary that is used in the ontology domain. Some examples of annotations can be found in the appendix. There were a few changes made when adopting SKOS metadata. The main reasons for these adaptions

Name	**Type**	**Purpose**
AWS	Web Hosting	Deliver Meta LMS to the public web.
ReactJs + GoJs	Frontend	Web application of Meta LMS. To help user in visualising the ontology and some level of ontology editing + course creations
Flask	Web Framework	Connect Endpoints between Frontend and Backend. Also provides APIs to other LMSs.
Python	Backend	Meta LMS's written programming language.
Ontology Architecture	Ontology	Stores Semantic Definitions of Learning Objects. *More info in next section.*
Sqlite3	Database	Stores physicals bytes of data regarding learning objects.
SCORM Cloud	Saas	Responsible to support and extract different learning object file types, including SCORM 1.2, SCORM 2004, AICC, xAPI, cmi5 content and PDF and MP4 files. The extracted JSON Schema can be found in the appendix
CSO	Saas	An external Computer Science Ontology to help classify and provide ontology definitions.

Table 3.1: Meta LMS System Architecture Components

are to keep things simple for our first attempt in metadata creation. It also helps to maintain the ontology easily and provide the means to calculate the similarity between the concepts and also track the dependencies of concepts. The list of changes are as follow:

1. **Replacement of concept collection with the scheme.** When a user wants to describe the same group of concept, they would use schemes instead.

2. **Addition of annotation.** Extra annotation are added, i.e *Requires* (for dependencies of LO), *loId* (link ontology instances with database) and *csoConcepts* (cso related annotations).

3. **Removal of unused annotation.** Unused or unimportant annotation for this project is removed or ignored in the ontology.

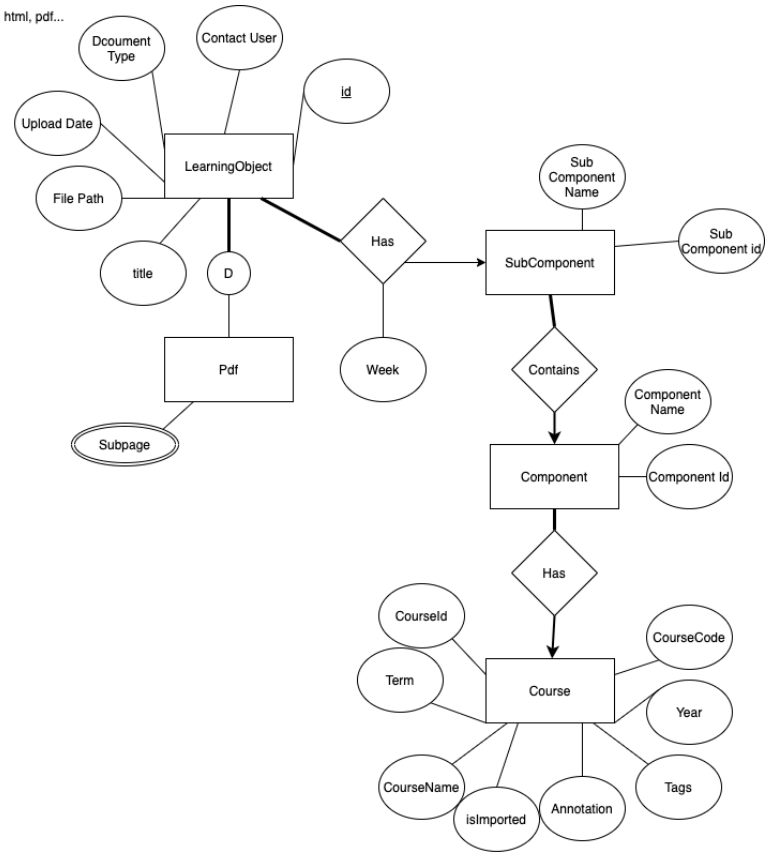

Figure 3.3: Meta LMS database ER Diagram

Metadata in Ontology Properties

One of the biggest difference and motivation of metadata stored in ontology than in classical DAG data structure is its capability to provide extra characteristics of the properties mapping the concepts together without extra work. These characteristics help to check for invariants set on to the properties like domain & ranges of concepts, irreflexive relation and functional relation. For example, suppose if a concept A is depended on concept B, we can logical to define all the pre-requisite dependency of concept A to be also depended on concept B. It can also check for unfavourable situations in the concept mapping like cyclic relation which will cause a paradox in the system. One thing that is worth to mention here, although not currently used due to the lack of knowledge and experience, is that ontology can be processed by *reasoners* or *classifiers*. These reasoners can perform consistency checking on the classes of concepts

Term	Description
concept	An abstract idea or notion; Every Concept is will have a prefix of "Concept..."
conceptSchemes	Concept scheme in which the concept is included.
Lexical Labels	
prefLabel	The preferred lexical label for a resource, in a given language
hiddenLabel	A lexical label for a resource that should be hidden when generating visual displays of the resource.
altLabel	An alternative lexical label for a resource
Semantic Mapping Relation Properties	
broadMatch	LO is 90∼100% similar
closeMatch	LO is 60∼80% similar
exactMatch	LO is 40∼50% similar
narrowMatch	LO is <40% similar
subClassOf	LO is a subclass of Another LO
Annotations	
requires	Dependencies of the concepts
csoConcepts	CSO Extracted Concepts
loId	Id associated in the database

Table 3.2: Meta LMS Ontology Vocabulary

or determine whether a concept is a subclass of another class.

There is a total of 7 characteristics available in RDF. The official definition and all the characteristics is described in table 3.3; while the definition of each properties characteristics in MetaLMS is described in table 3.4

3.2.3 Similarity Calculation

Meta LMS uses disjunctive shared information [1] based on Resnik distance to measure the similarities of concepts. Even with its drawback describe in the background, Resnik will perform reasonably with the current ontology as it is small and has little to none inheritances. This might change in the future and with DiSHin, it is easy to change the metric used.

To determine the similarity of a concept in Meta LMS, we consider all 5 of our semantic

[1]https://dishin.readthedocs.io/en/latest/

Name	**Description**
Functional	If a property is functional, for a given individual, there can be at most one individual that is related to the individual via the property. Also known as single valued properties and also features
Inverse Functional	The inverse property is functional
Transitive	If a property is transitive, this means the if property relates individual a to b, b to c, then the property relates a to c.
Symmetric	If a property is symmetric, this means the individual a and b have binary relation of "equal to".
Anti-Symmetric	If a property is anti-symmetric, this means if an individual a is related to b, then b cannot be related to a via the same property
Reflexive	If a property is reflexive, the individual a relate the property to itself
Irreflexive	If a property is irreflexive, the individual a cannot relate the property to itself

Table 3.3: The 7 Property Characteristics

Property	**Property Characteristic Used**
subClassOf	transitive, functional, irreflexive
.*Match	symmetric, irreflexive
required	transitive, anti-symmetric, irreflexive

Table 3.4: MetaLMS property characteristic

mapping in the Ontology Domain, defined as *base* below and CSO Ontology, defined as *CSO*. Since Resnik distance can only consider one semantic mapping at the time, we need to perform similarity calculation on each of our Semantic mapping and CSO itself. We apply weightage to each of them as each semantic mapping contributes to the similarity metric differently. Currently, the following formula is used:

$$Score_{total} = w_{base} * (\sum_{i} Score_{base\_i} * w_i) + w_{CSO} * Score_{CSO}, \tag{3.1}$$

where $\sum_i Score_{base\_i}$ is the sum of all semantic mapping scores with their respective weightage. The current assigned weightage is based off logical description of semantic mapping relations. The current weightage is defined as follow:

Name	Weightage
broadMatch	0.3
closeMatch	0.5
exactMatch	0.8
narrowMatch	0.1
subClassOf	0.7
CSO	0.5
Base	0.5

Table 3.5: List of Weightage Definitions

Chapter 4

Evaluation

4.1 User Study

We conducted a qualitative study on Meta LMS to assess whether Meta LMS can achieve our project goals mentioned in Section 1.1. During the study, we organise individual sessions with 5 academic members and 5 public audiences. On average, there the academic members are a tutor who has taught at the University of New South Wales (UNSW). We introduced the main functionalities regarding Meta LMS for 10 minutes and asked them to use the application for about 5 minutes. We added two scenario-based tasks for general audiences to familiarise them with the application and step into the perspective as an academic staff. A demo version of Meta LMS is available at this link [1], which has some pre-populated concepts regarding computer science courses in UNSW. The survey consists of close-ended questions on a 1 to 5 scale regarding the quality of Meta LMS and the degree of it solving the goals compared to existing LMS, follow up by open-ended questions to justify their answers. We will summarise the results and answers to these set of questions.

We map some close-ended questions on a 1 to 5 scale, where 1 is the most negative expression and 5 is the most positive expression of respond; and map some questions on

[1] http://3.26.26.205:8080/#/

a Yes or No answer basis. Figure 4.1, 4.2, 4.3 shows all the responded for the respective answers.

Q1. Was MetaLMS able to fully capture and express your learning objects? Majority of responded liked the idea of the fined-grained view of concepts of learning objects. One of the veteran academic respondents stated that the fined-grained details of each learning objects provide great flexibility than in most LMS systems that the respondent has used in the past.

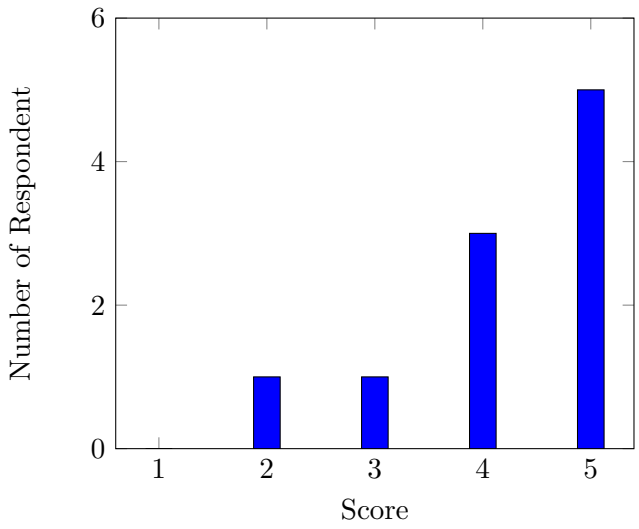

Figure 4.1: Survey question 1

Q2. Was MetaLMS able to help you navigate and find your learning objects? 2 of the respondent found it nice to have all the relevant of the details of learning objects in a single page. However, there is some respondent that found the concept page needs more improvements in terms of UI presentation to get a "better overview in a single page".

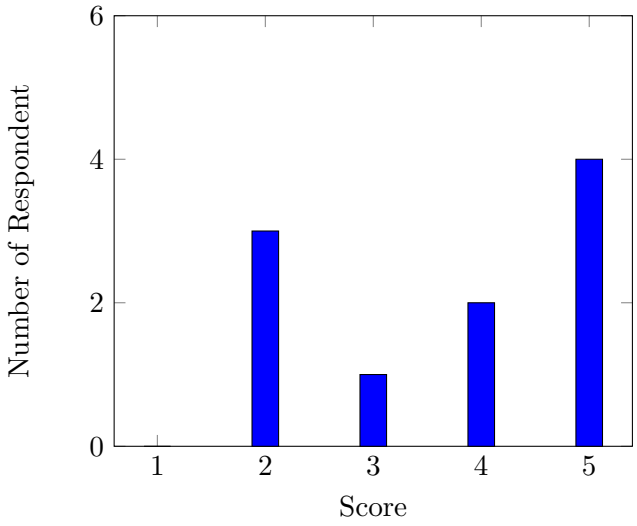

Figure 4.2: Survey question 2

Q3. Was MetaLMS able to help you structure your courses better? Most respondents find it useful to have a cross-check system across the learning objects.

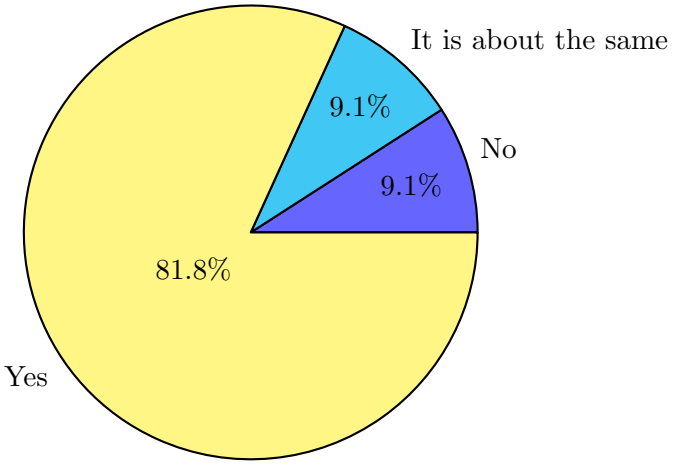

Figure 4.3: Survey Question 3

Q4. What are your overall thoughts and experiences on MetaLMS? Almost half of the respondent expressed a need for improvements on the UI. One of the general audience group respondent "feel lost" when interacting with Meta LMS and gave feedback that "Have better explicit definition and insert a help section somewhere." On

another note, some respondent thinks that the focus on concepts provides more flexibility and integration of any external document, which directly translate to offering a better personalised experienced for students and course conveners.

Q5. How does meta LMS compare to existing LMSs, would you prefer meta LMS? Most respondents liked the course comparison idea as the feature dives deeper to analyse the courses itself unlike the convention user written description, but would still currently prefer existing LMS as it offers better UI. Most agreed that Meta LMS offers better functionality than most existing LMS and has a lot of room for growth.

4.2 Discussion

4.2.1 Usability

Based on a user survey, the approach Meta LMS has proven to be in the right direction towards providing re-usability to users. As a preliminary prototype, MetaLMS used the minimal features to showcase the supporting power of metadata to provide re-usability and tracking of dependency to the user. Many users have shown appreciation to have a tool to have the extra fine-grained look on the learning objects as they can venture to more uncharted learning objects that exist around their LMS.

Secondly, From the user survey, one key lesson that we learnt is that it is critical to provide a user-friendly interface. Many users still prefer existing LMS even though MetaLMS provides favourable functionalities for course convenors.

Lastly, there is currently still a lack of proper authoring tool on our metadata like in Protégé due to Python RDF library limitations and time constraints. Adding more features to provide more flexibility in editing the explicit semantic metadata is going to be helpful in the usability of an application.

4.2.2 Ontology

During the manual ontology process, there are few insight found during development. For one, one of the fundamental issues that are constantly found in the user bias on domain knowledge. When there are grey areas during classifying the semantic relation between concepts, the result of the ontology domain will differ from each individual or group. Thus leading up to difficulties in determining the correctness of semantic relation. The addition of CSO automated generated ontology certainly helped to reduce the fundamental issue and is also part of the contributing reasons to weight manually edited ontology and automated ontology equally.

Another obvious fundamental issue with manual ontology process was the time-consuming aspect of developing one. For every concept added, there are exponential checks needed to determine the real semantic relation with it between all other concepts. The aid of semantic web's property characteristic invariant checking helped a lot in cross-checking any human error that we made, thus saving some work to verify the classes manually.

CSO

CSO covers a huge range of concepts that exist in computer science. As a result, there is almost over 40 MB size of ontology available. Due to this large size of ontology, the current DiShln tool will run out of memory before being able to calculate all the similarity distance. To avoid this issue, we use part of the CSO, such as using a semantic relationship that has lower mean distance and used the trimmed version of CSO ontology. The result of metadata generation from CSO still able to provide a meaningful tagging result when compared to the CSO web version [2]

[2]https://cso.kmi.open.ac.uk/classify/

4.2.3 Resource Processing Component

Resource Processing Component is responsible for extracting metadata from input sources. Currently, only existing educational metadata and pdf files are supported. The metadata tagging from CSO performs reasonable well as it can cover most of the core concepts from each pdf pages when compared to explicit metadata definition. However, the Resource Processing Component falters when the pdf structure has many images in the sub-pages. As CSO only analyses the texts from sub-pages, images that provide meaningful metadata is not considered at all. As a result, it reduces the correctness and fine-grains of metadata being able to be extracted from the documents.

Chapter 5

Conclusion

In this project, we have looked at some issues revolving around the current LMS system that exists in the current state of the art. We have presented Meta LMS to aid academic members to easily re-purpose and reuse digital content. The evaluation on MetaLMS, performed with a mix of veteran UNSW academic members and general public, shown that Meta LMS provides a right approach towards providing re-usability to users.

5.1 Future Work

Following on from this project, there are many ways to enhance the capabilities of MetaLMS.

1. **Automation on Annotating**. With many available automation available in the state of the art, it is possible to develop an automation on ontology development rather than manual work.
2. **Cater features for student** Che current metadata can be easily used to support student model which would cater for student's personalised learning.
3. **Recommendation System** The system can find out which learning object,

courses is relevant by similarity benchmark in Meta LMS. By recommending these medias can provide better re-usability.

4. **Enhancement and additional support on document processing components** Adding more support to varying types of documents help Meta LMS to populate itself at scale.

Bibliography

[1] Sven-Michael Wundenberg. *Requirement Engineering for Knowledge-Intensive Processes: Reference Architecture for the Selection of a Learning Management System*. BestMasters. Springer Fachmedien Wiesbaden GmbH, Wiesbaden, 2015 edition, 2015.

[2] Xue Qing ZHONG. Building a meta learning management system.

[3] Hajiram Beevi. Survey of metadata standards for e-learning resources. 10 2014.

[4] Marco Alfano, Biagio Lenzitti, Davide Taibi, and Markus Helfert. Ulearn: Personalized medical learning on the web for patient empowerment. In *Advances in Web-Based Learning – ICWL 2019*, volume 11841 of *Lecture Notes in Computer Science*, pages 217–228. Springer International Publishing, Cham, 2019.

[5] Hyeon Kyeong Hwang, Ivana Marenzi, Maria Bortoluzzi, and Marco Ronchetti. The role of context for user annotations in searching shared materials. In *Advances in Web-Based Learning – ICWL 2017*, volume 10473 of *Lecture Notes in Computer Science*, pages 91–100, Cham, 2017. Springer International Publishing.

[6] Jim Morgan, Euan Lindsay, Colm Howlin, and Maartje E. D. Van den Bogaard. Pathways of students' progress through an on-demand online curriculum. In *Association for Engineering Education - Engineering Library Division Papers*, Atlanta, 2019. American Society for Engineering Education-ASEE.

[7] Gauri and Suhas Patil. Natural language query processing based on probabilistic context free grammar. 11 2020.

[8] Georgia Troullinou, Haridimos Kondylakis, Evangelia Daskalaki, and Dimitris Plexousakis. Rdf digest: Ontology exploration using summaries. 10 2015.

[9] Grigoris Antoniou and Frank Van Harmelen. *Semantic Web Primer*. Cooperative information systems. MIT Press, Cambridge, 2008.

[10] N. Noy and Deborah Mcguinness. Ontology development 101: A guide to creating your first ontology. *Knowledge Systems Laboratory*, 32, 01 2001.

[11] Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. A framework for benchmarking entity-annotation systems. In *Proceedings of the 22nd international conference on world wide web*, WWW '13, pages 249–260. ACM, 2013.

[12] Jose L Martinez-Rodriguez, Aidan Hogan, and Ivan Lopez-Arevalo. Information extraction meets the semantic web: A survey. *Semantic Web*, 11(2):255–335, 2020.

[13] Sonal Jain and Jyoti Pareek. Automatic topic(s) identification from learning material: An ontological approach. *Computer Engineering and Applications, International Conference on*, 2:358–362, 01 2010.

[14] Devshri Roy, Sudeshna Sarkar, and Sujoy Ghose. A comparative study of learning object metadata, learning material repositories, metadata annotation & an automatic metadata annotation tool. *M. Joshi H. Boley*, 2, 01 2010.

[15] Nicola Henze. Personalized e-learning in the semantic web. *International journal of emerging technologies in learning*, 1(1), 2006.

[16] M Samsuzzaman, M.T Islam, Sharmin Rashid, Faysal Ahmed, and Md. Ridgewan Khan. Proposed model of e-learning management system using semantic web. *Journal of applied sciences research*, 8(5):2484–2492, 2012.

[17] Dimitrios A Koutsomitropoulos and Georgia D Solomou. A learning object ontology repository to support annotation and discovery of educational resources using semantic thesauri. *IFLA journal*, 44(1):4–22, 2017.

[18] Francesco Osborne, Angelo Salatino, Aliaksandr Birukou, and Enrico Motta. Automatic classification of springer nature proceedings with smart topic miner. In *The Semantic Web – ISWC 2016*, volume 9982 of *Lecture Notes in Computer Science*, pages 383–399, Cham, 2016. Springer International Publishing.

[19] You Bin, Liu Xiao-ran, Li Ning, and Yan Yue-song. Using information content to evaluate semantic similarity on hownet. In *2012 Eighth International Conference on Computational Intelligence and Security*, pages 142–145. IEEE, 2012.

[20] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. 1995.

[21] Francisco M Couto and Mário J Silva. Disjunctive shared information between ontology concepts: Application to gene ontology. *Journal of biomedical semantics*, 2(1):5–5, 2011.

Appendix

A.1 SCORM Cloud JSON Schemas

```
{
    "id": String,
    "title" : String,
    "xapiActivityId" : String,
    "created" : String,
    "updated" : String,
    "version" : String,
    "registrationCount" : String,
    "activityId" : String,
    "courseLearningStandard" : String,
    "tags" : String,
    "dispatched" : String,
    "metadata" : MetadataSchema,
    "rootActivity" : CourseActivitySchema,
}
```

Listing 1: Course Schema

```
{
    "title": String,
    "titleLanguage" : String,
    "description" : String,
    "descriptionLanguage" : String,
    "updated" : String,
    "duration" : String,
    "typicalTime" : String,
    "keywords" : String,
}
```

Listing 2: Metadata Schema

A.2 Ontology Domain Annotation Examples

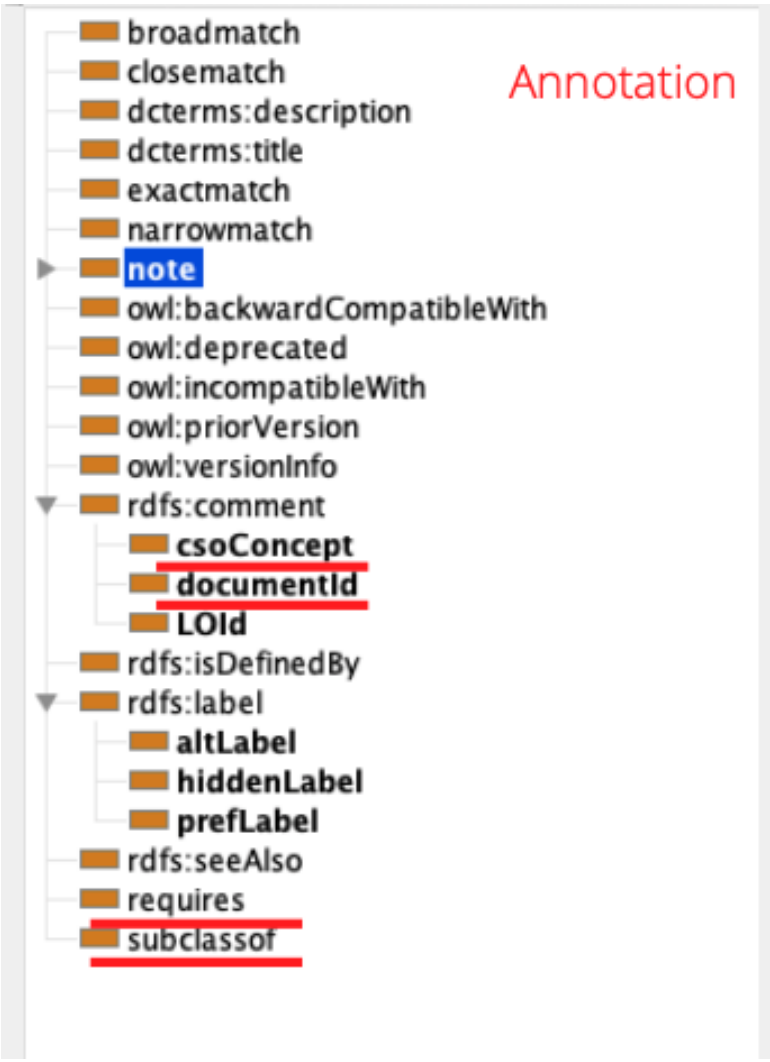

Figure A.1: Ontology Domain Annotations

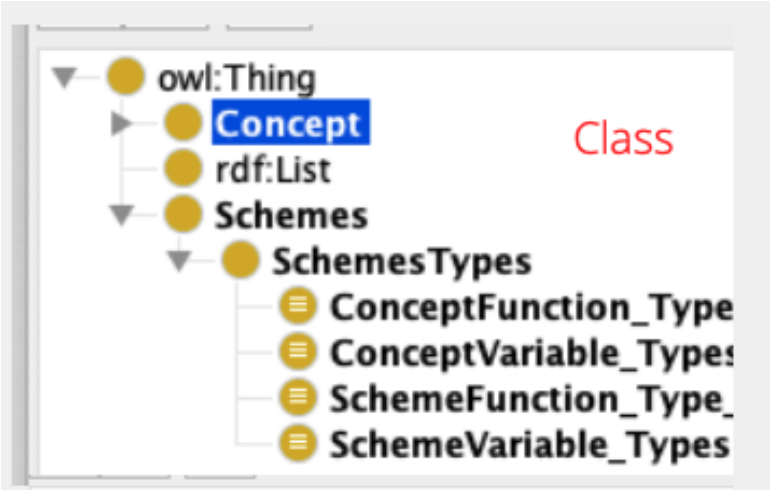

Figure A.2: Ontology Domain Classes

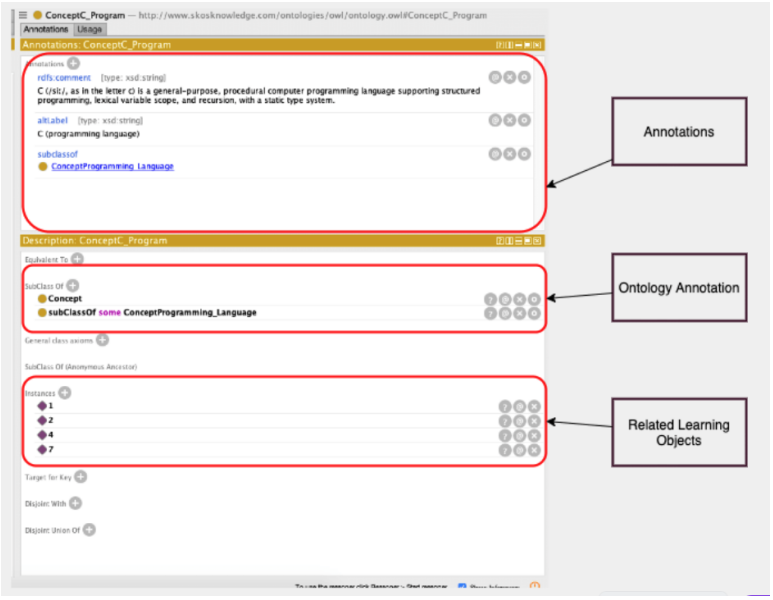

Figure A.3: Ontology Domain Classes Annotations