

國立交通大學

資訊科學系

碩士論文

符合 Learning Design 標準學習活動的推薦系統
之研究

A Recommendation Scheme of Personalized Learning Activities
Based on Learning Design Standard

研究生：林易虹

指導教授：曾憲雄 博士

中華民國九十四年六月

符合 Learning Design 標準學習活動的推薦系統之研究
A Recommendation Scheme of Personalized Learning Activities
Based on Learning Design Standard

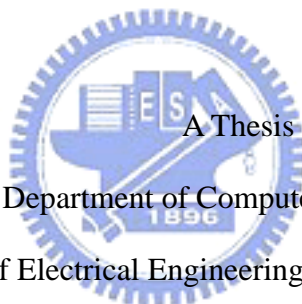
研究生：林易虹

Student：Yi-Hung Lin

指導教授：曾憲雄

Advisor：Shian-Shyong Tseng

國立交通大學
資訊科學系
碩士論文



Submitted to Department of Computer and Information Science

College of Electrical Engineering and Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer and Information Science

June 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年六月

符合 Learning Design 標準學習活動的推薦系統之研究

研究生：林易虹

指導教授：曾憲雄博士

國立交通大學電機資訊學院資訊科學系碩士班

摘 要

隨著數位化學習的普及，許多網路學習的相關標準已被提出與制訂。在這些標準中，SCORM 和 Learning Design (LD) 最為被廣泛應用。然而，SCORM 在描述教學活動的情境上，有諸多限制與不便，例如：無法明確表達各種教學理論的情境及不支援多人學習的模式。相較之下，LD 豐富描述教學理論的能力使其成為近來描述學習活動之熱門標準，故目前的趨勢則以 LD 作為上層教學活動描述的標準而 SCORM 為下端內容描述之標準；同時很多組織或研究單位亦投入開發相關平台與編輯工具。隨著產生符合 LD 標準的學習活動越來越多，如何搜尋、進而如何有效地重複利用等將會成為重要的問題。因此，在本論文中提出了一套學習活動的推薦機制，稱為 Learning Activity Recommendation (LAR) Scheme，以達到能夠有目標性地搜尋並重複利用符合 LD 標準之教學活動的目的。根據多元相似度計算的機制，LAR 能夠輔佐使用者找到適合他們需求的學習活動；另外結合規則庫推論，能夠動態控制使用者互動介面及動態配置相似度計算。LAR 不僅能夠尋找到符合使用者需求的教學活動，簡化使用者在編輯符合 LD 標準教學活動之過程，同時整個架構也具備相當的彈性與擴充性。本論文亦導入布魯姆教學目標分類法的相似度計算來有效地延伸 LAR 的相似度計算能力與擴充能力，同時也藉由實驗來驗證所提出的結構相似性之效能。

關鍵字：教學活動重複利用、相似度計算、學習設計、規則庫

A Recommendation Scheme of Personalized Learning Activities Based on Learning Design Standard

Student : Yi-Hung Lin

Advisor : Shian-Shyong Tseng

Institute of Computer and Information Science
National Chiao Tung University

Abstract

As the popularity of e-learning, the related standards which are used for providing the interoperability, reusability, portability, and sharability of learning resources are promoted. Among these standards, the most famous are SCORM and Learning Design (LD). However, SCORM has some disadvantages such as unsupported of the pedagogy theory and cooperative learning. Therefore, LD becomes the main standard in the use of expressing learning scenario. Many institutes are developing the related platforms and authoring tools. As more LD compliant learning activities are generated, how to retrieve and how to reuse them will be critical issues. Thus, in this thesis, we propose a learning activity reusing mechanism, called Learning Activity Recommendation (LAR) Scheme, to achieve the purposes that search efficiently and reuse the learning activities compliant with LD. With the multiple similarity calculations, LAR can help users to find the learning activity they desired. In the framework, we also adopt the expert system shell, DRAMA, to handle the interactive interface with users and the generation of the similarity equation. In summary, LAR is full of flexibility and scalability to simplify the process of editing learning activity compliant with LD. The thesis also adopts the pedagogy theory, Bloom, to extend the framework of LAR. The experimental result shows that the similarity functionality really works.

Keyword: Reuse Learning Activity, Similarity measurement, Learning Design, Rule base.

致謝

這篇論文的完成，首先要感謝我的指導教授，曾憲雄老師。在研究所兩年的歲月裡，無論是在學術研究或是為人處世方面，皆讓我受益匪淺。在碩二撰寫論文的這段時間，老師花了很多心思幫我批改、給我建議，心裡覺得很感動；同時也感謝我的口試委員，黃國禎教授、楊鎮華教授、和楊錦潭教授，他們給予了我相當多的寶貴意見，讓本篇論文更有意義與價值。

第二要感謝的是蘇俊銘學長和翁瑞峰學長，這段期間內，他們花費了許多心力與我討論以及給我建議，協助我修改論文；此外從他們身上，我學習了不少生活態度及為人處事的方法，在此深表感激。還有實驗室的同窗們：君翰、昱璋、政霖、瑞言、柏智、育松、成樑，在這兩年的時光裡，能和你們一起度過，互相扶持鼓勵甚至互開玩笑，我覺得能遇到你們很開心也很滿足。還有其他在身邊鼓勵我的朋友們：姿文、倫武、文如等，眾多親朋好友們雖無法在此一一提及，但我心裡真的非常感激有你們在我身邊！

也感謝我的男朋友弘育，在我身邊不停地鼓勵我，讓我在遇到挫折的時候能夠又繼續樂觀地向前走；最後要感謝的是我的家人，默默地支持與鼓勵，並不時地關心我。日後，我會更加努力地繼續前進！

Tables of Content

摘 要.....	I
ABSTRACT.....	II
致謝.....	III
TABLES OF CONTENT	IV
LIST OF FIGURES	VI
LIST OF TABLES.....	VII
CHAPTER 1. INTRODUCTION	1
CHAPTER 2. RELATED WORK	4
2.1 SCORM.....	4
2.2 LEARNING DESIGN	6
2.3 BLOOM.....	12
CHAPTER 3. SCHEME OF LEARNING ACTIVITY RECOMMENDATION	16
3.1 LAR SCHEME.....	17
3.2 DATA REPRESENTATION IN LAR SCHEME.....	19
CHAPTER 4. PRE-PROCESSING OF LAR SCHEME.....	24
4.1 THE DEFINITION OF USER INPUT PARAMETERS	24
4.2 UoL PARSER.....	26
4.3 CANDIDATED COMPUTABLE ACTIVITIES SELECTION (CCAS)	29
4.4 RULES DEFINITION OF SIMILARITY SELECTION (RDSS).....	30
CHAPTER 5. THE METHODOLOGY OF SIMILARITY MEASUREMENT	32
5.1 COVERAGE SIMILARITY (CS)	32
5.2 SEQUENCE SIMILARITY (SS).....	34
5.3 PATH SIMILARITY (PS).....	36
5.4 DISTRIBUTION SIMILARITY (DS)	38
CHAPTER 6. APPLYING PEDAGOGY THEORY TO COMPUTABLE ACTIVITY.....	39
6.1 BLOOM SIMILARITY	39
6.2 OTHER PEDAGOGY APPROACHES	47
CHAPTER 7. SYSTEM IMPLEMENTATION	48

7.1 TRANSFORMATION EXAMPLE OF UOLPARSER	48
7.2 SYNTHETIC COMPUTABLE ACTIVITY MATRIX AND EXPERIMENT RESULTS	53
7.3 SYSTEM INTERFACES	56
CHAPTER 8. CONCLUSION	62
REFERENCES	63
APPENDIX A	66



List of Figures

Figure 2.1: An Example of Activity Tree (AT) with Clusters	5
Figure 2.2: the structure of learning	7
Figure 2.3: Learning Design Rationale[22]	8
Figure 2.4: Example of Learning Design.....	9
Figure 2.5: Conceptual model of overall Learning Design.....	10
Figure 3.1: Scheme of LAR	17
Figure 3.2: Unit of Learning	21
Figure 4.1: Concept of UoL Parser	26
Figure 4.2: The hierarchy of attributes we used.....	27
Figure 6.1: Example of BP.....	40
Figure 6.2: Knowledge Dimension Tree (Up)	42
Figure 6.3: Cognitive Processing Dimension Tree (CPDT)	43
Figure 6.4: The Extended Scheme of LAR.....	45
Figure 7.1: The templates of learning flow and its corresponding CAM	54
Figure 7.2: The screenshot of the synthetic CAM	54
Figure 7.3: The screenshot of experiment result.....	55
Figure 7.4: The result of satisfaction.....	56
Figure 7.5: Editing rules of LAR with DRAMA Editor	57
Figure 7.6: The first inferencing interactive process	58
Figure 7.7: The inference process and the result of LAR	59
Figure 7.8: The editing process with RELOAD.....	60
Figure 7.9: Upload page of CopperCore.....	60
Figure 7.10: The Simulation on CopperCore.....	61

List of Tables

Table 2.1: Knowledge Dimension [27]	12
Table 2.2: Cognitive Process Dimension [27].....	13
Table 3.1:Meta-data of Computable Activity.....	20
Table 6.1:BI of figure3.2.....	40



List of Algorithms

Algorithm 4.1: UoL Parser Algorithm (UoLPAIgo)	27
Algorithm 4.2: Candidated Computable Activity Selection Algorithm (CCASAIgo).....	29
Algorithm 5.1: Loop Path Detection Algorithm (LPDAIgo)	37
Algorithm 5.2: Condition Path Detection Algorithm (CPDAIgo)	38



Chapter 1. Introduction

Recently, learning over Internet has become a trend of education, such as e-learning, online-learning, or distance learning. Plenty of e-learning systems make people conveniently study at any time and any location. Several international organizations have proposed several e-learning standards to provide the interoperability, reusability, portability, and sharability of learning resources among different e-learning systems. Therefore, more and more authoring tools have been developed for teachers to edit the teaching materials and learning activities compatible with these standards. Therefore, a large number of learning contents will be created. Undoubtedly, the maintenance of these learning sources will become a critical issue.



Based on these standards, another key challenge to e-learning today, is the development of a framework based on sound pedagogical principles that will promote the exchange and interoperability of learning concepts, materials and teaching strategies [14][1][21]. The process of designing a learning scenario or learning flow is called “Teaching/Learning Plan”, “**Instructional Design (ID)**” or “Learning Lesson”. Teachers can inspect or review the learning flow by examining the ID. In other words, an ID implies the relationship between resources, such as the manner in which they are sequenced or presented to the learner. Alternatively, an instructional context may define the role that a given resource plays in a learning scenario; for example, it may be an illustration, an example, an explanation or an exegesis [1]. For those teaching-beginners without plentiful teaching experiences, they may face some troubles while constructing their own learning activities. First, they may be confused

about how to design a better learning activity that helps students learn more efficiency. Second, they may be lack of experiences about how to apply some pedagogy theory while designing their learning activity. Besides, the ID is a tedious work. Mostly, the well-known learning strategies change little while different teacher applying to their learning activity. So, if the knowledge is not well-reuse, these dreary works will be repeatedly done by different teachers. Therefore, we can realize that IDs constructed by other veterans of many teaching experiences for reference are useful and helpful. They are worthy to be shared and reused in education.

Among these learning standards, the most famous standards may be SCORM [5]. However, its weakness such as unsupported of pedagogy theory and cooperative learning make another standard, Learning Design (LD), on the up grade [6]. LD has powerful expression ability of learning activity to overcome the disadvantages SCORM has. Nowadays, the trend of e-learning standards is to combine the LD and SCORM. That's the reason why we choose it for our e-learning standard here.

In this thesis, we propose an innovative scheme called **Learning Activities Recommendation (LAR)** that can reuse the learning activities compliant to Learning Design smartly. Users only need to input the information about their teaching activities, and then LAR will retrieve the matched learning activities in the repository by the intelligent similarity methodologies. The similarity methodologies are inferred by the *Rules Definition of Similarity Selection (RDSS)* of LAR. RDSS controls the interaction with users and generates the similarity equation. This framework provides LAR flexibility and scalability.

The organization of this thesis is as follows: In Chapter 2, we introduce some

related work about e-learning standards. The scheme of LAR and the data representation we propose is introduced in Chapter 3. Each phase of LAR is introduced in Chapter 4 and Chapter 5. And we adopt the pedagogy theory into LAR to prove the extension ability of LAR in Chapter 6. The implementation of LAR is discussed in Chapter 7. At the end, concluding remarks are given in Chapter 8.



Chapter 2. Related work

In this chapter, some related background knowledge will be introduced. First, we will go through the e-learning standards: SCORM and Learning Design. Then the famous taxonomy for Educational Objectives, Bloom, will be discussed later.

2.1 SCORM

SCORM (Sharable Content Object Reference Model) was developed by ADL Initiative, which was launched by The White House Office of Science and Technology Policy (OSTP) together with government, industry, and academia. Among those existing standards for learning contents, SCORM, which was proposed by the U.S. Department of Defense's Advanced Distributed Learning (ADL) organization in 1997, is currently the most popular one. The SCORM specifications are a composite of several specifications developed by international standards organizations [14][14][1]. In a nutshell, SCORM is a set of specifications for developing, packaging and delivering high-quality education and training materials whenever and wherever they are needed. SCORM-compliant courses leverage course development investments by ensuring that compliant courses are Reusable, Accessible, Interoperable, and Durable (RAID) [21]. At present, the Sequencing and Navigation (SN) [28] in SCORM 1.3 (or called SCORM 2004) adopting the Simple Sequencing Specification of IMS relies on the concept of learning activities, each of which may be described as an instructional event, events embedded in a content resource, or an aggregation of activities to describe content resources with their contained instructional events. Content in SN is organized into a hierarchical structure, namely activity tree (AT) as a learning map. The example of AT is shown in Figure 1. Each activity in the Activity Tree includes

two data models: Sequencing Definition Model (SDM) including an associated set of desired sequencing behaviors of content designer and Tracking Status Model (TSM) including the information about a learner’s interaction with the learning objects within associated activities. The SN uses information in SDM and TSM to control the sequencing, selecting and delivering of activities to the learner. The sequencing behaviors describe how the activity or how the children of the activity are used to create the desired learning experience. SN enables users to share not only learning contents, but also intended learning experiences. It also provides a set of widely used sequencing method so that the teacher could do the sequencing efficiently. However, how to create, represent and maintain the activity tree and associated sequencing definition is an important issue.

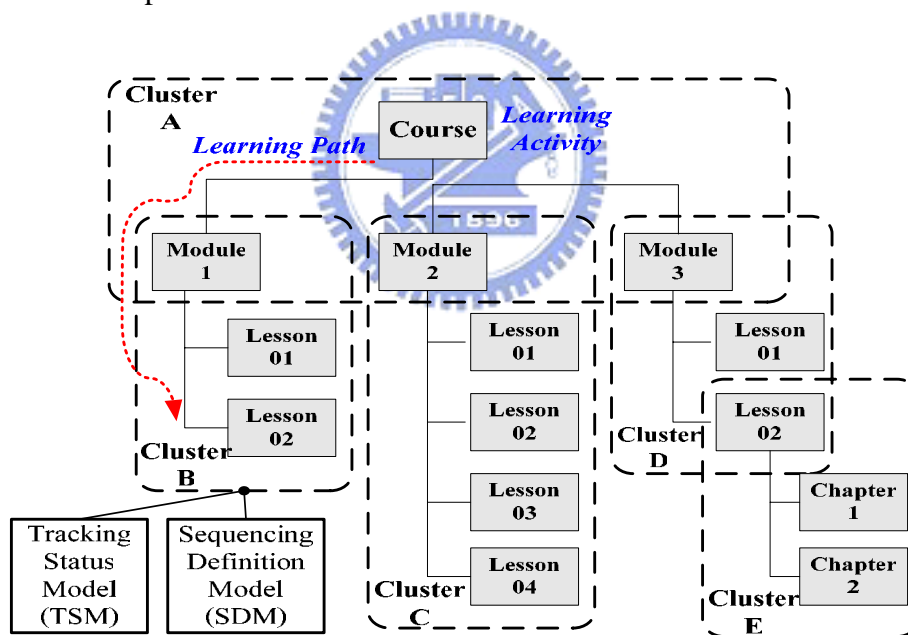


Figure 2.1: An Example of Activity Tree (AT) with Clusters

The complicated sequencing rule definitions of SN in SCORM 2004 make the design and creation of course hard. The sequencing rule has less flexibility to support cooperative learning. Moreover, SCORM is hard to imply the teaching pedagogic theory.

2.2 Learning Design

The core concept of the Learning Design framework is that regardless of the pedagogical strategy, learners attain learning objectives by performing a specific order of learning activities. It has emerged as one of the most significant developments in e-learning [22][14][10][2][3][7][10][11].

From a standards/specifications perspective, IMS Global Learning Consortium has recently released the IMS Learning Design specification[20], based on the work of the Open University of the Netherlands (OUNL) on “Educational Modeling Language” [23], a notational language to describe a “meta-model” of instructional design. The OUNL that coordinates an international EML/IMS Learning Design implementation group known as the Valkenburg group [31] has recently stated their intention to no longer develop EML continuously, but instead focus their energies of the new IMS Learning Design specification [29]. Thus it can be seen that LD has come into more and more notice.

2.2.1 IMS Learning Design Information Model

The primary use of IMS Learning Design is to model units of learning (uol) by including an IMS Learning Design in a content package. The Manifest is the information structure defined in the Content Packaging specification. It contains a package as an XML file with a fixed, pre-defined name (imsmanifest.xml). The integration of a Learning Design into the Content Packaging Structure is set out in the Figure 2.2.

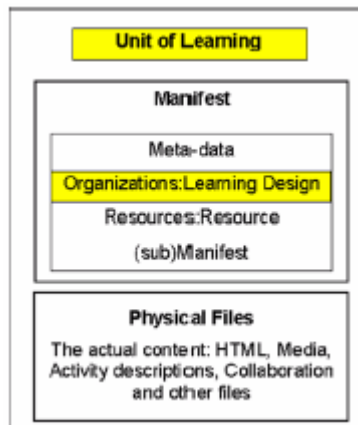


Figure 2.2: the structure of learning.

To create a unit of learning, IMS Learning Design is integrated with an IMS Content Package by including the learning design element as another kind of organization within the <organizations> element, using the standard namespace for Learning Design. When the standard namespace is “[standard-namespace-for-learning-design]”, then learning design elements are included as follows (ignoring irrelevant elements and attributes):

```

<manifest>
  <metadata/>
  <organization>
    <learning-design xmlns=”[standard-namespace-for-learning-design]”>
      [add learning design elements here]
    </learning-design>
  </organization>
  <resources>
</manifest>

```

The core components of the Learning Design Framework are shown in Figure 2.3 and summarized below:

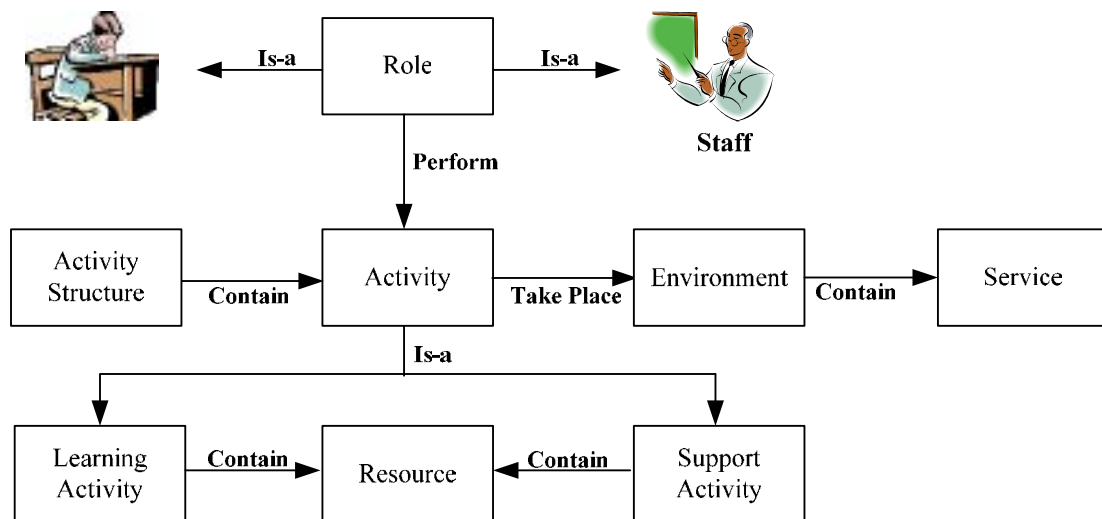


Figure 2.3: Learning Design Rationale[24]

- *Role* component specifies the type of participant in a unit of learning. There are two basic types: *Learner* and *Staff*. These roles can be sub-typed to allow learners to play different roles in certain types of learning activity such as task-based, role-play and simulations. Similarly support staff can be sub-typed and given more specialized roles, such as Tutor, Teaching Assistant, Mentor, etc. Thus, Roles set the basis for multi-user models of learning. The name that a certain role is given depends on the underline pedagogy and the setting in use.

- *Activities* are one of the core structural elements of the ‘learning workflow’ model for learning design. They form the link between the roles and the learning objects and services in the learning environment. They describe the activities that a role has to undertake within a specified environment composed of learning objects and services. They also specify their termination conditions and the actions to be taken on termination. There are two basic types of activities: learning activities and support activities. A learning activity is directed at attaining a learning objective per individual user. Any user performs a learning activity only once (until completion). A support activity is meant to facilitate a role performing one or more learning activities.

The controller of the role and activities flow is the attribute “Method”. It contains two core parts of the LD specification: the play and conditions. A play specifies the actual learning design, the teaching-learning process, referring to the components declared earlier. And conditions are used in conjunction with properties to further refinement and to add personalization facilities in the learning design.

Here we illustrate the architecture more detail by showing an example of UoL.

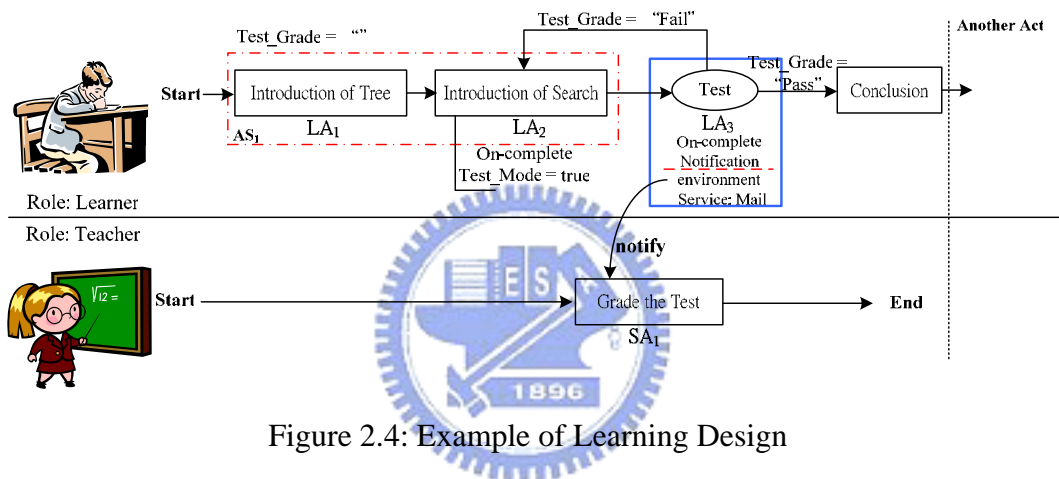


Figure 2.4: Example of Learning Design

Figure 2.4 illustrates a LD compliant learning flow. There are two roles: Learner and Teacher. The property “Test_Mode” is default to be false and “Test_Grade” =”. In the teacher’s learning flow, the only task is to grade every student’s test which is defined as a support activity (SA). This support activity will be executed several times until finishing all students’ grades. The learner takes two lessons: ”Introduction of Tree” and “Introduction of Search” (Learning Activity). While completing the lesson “Introduction of Search”, the property “Test_Mode” becomes true and the Test activity will show up. After the learner finishes the test, the system will notify the teacher to grade by using the service: send mail of the environment. After the teacher grading this student’s test, the learning flow of the students will continue or go back to

the lesson "Introduction of Search" depending on the value of the property "Test_Grade". If the "Test_Grade" is true, the learner will take the lesson: "Conclusion" and then go to another act.

According to the expressive ability, LD specifies three levels of implementation and compliance. The level wise conceptual view of LD is provided in Figure 2.5. In this figure, the emphasis is on the functional differences between each level.

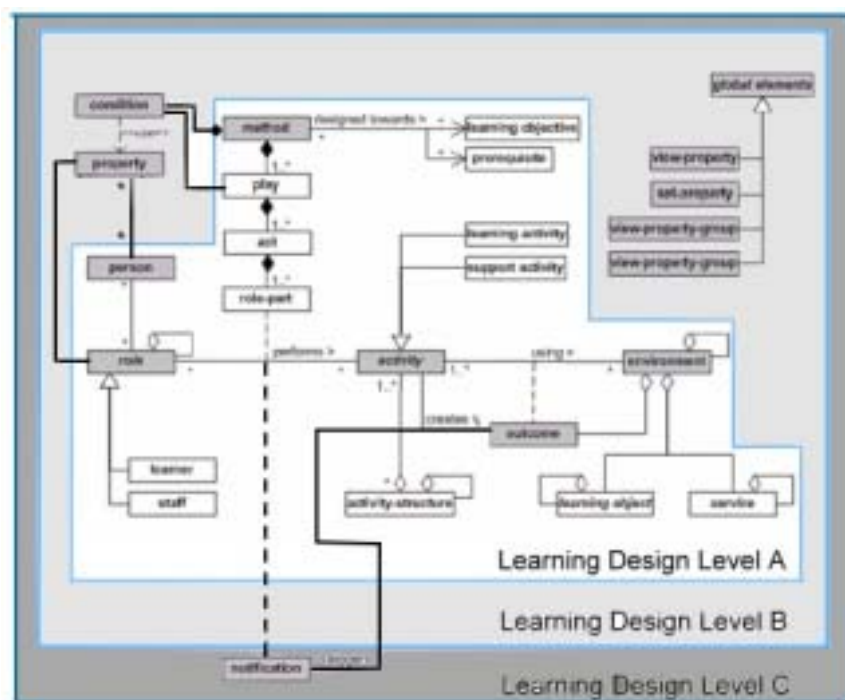


Figure 2.5: Conceptual model of overall Learning Design

LD Level A contains all the core vocabulary needed to support pedagogical diversity. And Level B adds Properties and Conditions to level A, which enable personalization and more elaborate sequencing and interactions based on learner portfolios. It can be used to direct the learning activities as well as record outcomes. Afterward, LD Level C adds Notification, which can be used for notifying or "messaging" both between system components and between roles. This adds a new dimension by supporting real-time event-driven work/learning flow. Activities can

then be set as a consequence of dynamic changes to the learner's profiles and/or of events generated in the courses of the learning activities.

2.2.2 Compared with SCORM Simple Sequence

While the definitions of Learning Design vary, the main elements tend to include greater focus on “context” dimensions of e-learning (rather than simply “content”), a more “activity” based view of e-learning (rather than “absorption”), and greater recognition of the role of “multi-learner” (rather than just single learner) environments. While Learning Design does not exclude single learner, self-paced modes of e-learning, it draws attention to a wider range of collaborative e-learning approaches in addition to single learner approaches. Much of the focus on Learning Design arises from a desire for re-use and adaptation at a level above simply re-using and adapting content objects. These above advantages make Learning Design become more popular than SCORM.



2.2.3 Related Tools of LD Specification

Currently many systems and tools which are based on Learning Design specification are developing. The most popular related researches are two projects: RELOAD Editor [26] and CopperCore [8].

The RELOAD Editor supports the full IMS Learning Design specifications for Levels A, B and C. It provides the graphical User Interface for all elements. Besides, it also provides fully featured Help system and easy file management. For the designer of LD compliant learning activity, the most convenient function is the wizards to import and export as IMS Learning Design Zip Package.

The CopperCore is a J2EE runtime engine for IMS Learning Design released by the Open Universiteit Nederland (OUNL). It supports all three levels of IMS Learning Design (A, B, and C). It includes a command line interface to most of the API calls, an example of a publication interface, and an example of a web delivery interface. These two tools will be helpful in the experiment of this thesis.

2.3 Bloom

In 1956, Benjamin Bloom who headed a group of educational psychologists developed a classification of levels of intellectual behavior important in learning [30]. However, to the need of real need in education [3], Bloom taxonomy was revised in 2001 [4]. The revision divides the learning goal into two dimensions: knowledge dimension and cognitive process dimension. The former assists the teachers classify what to teach; the latter promotes the students retain and transfer the knowledge they learned. The following tables depict two dimensions in detail.

Table 2.1: Knowledge Dimension [4]

MAJOR TYPES AND SUBTYPES	EXAMPLES
A. Factual knowledge – The basic elements students must know to be acquainted with a discipline or solve problem in it.	
AA. Knowledge of terminology	Technical vocabulary, musical symbols.
AB. Knowledge of specific details and elements	Major natural resources, reliable sources of information
B. Conceptual knowledge – The interrelationships among the basic elements within a larger structure that enable them to function together.	
BA. Knowledge of classifications and categories	Periods of geological time, forms of business ownership
BB. Knowledge of principles and generalizations	Pythagorean theorem, law of supply and demand

BC. Knowledge of theories, models, and structures	Theory of evolution, structure of Congress
C. Procedural knowledge – How to do something, methods of inquiry, and criteria for using skills, algorithms, techniques, and methods	
CA. Knowledge of subject-specific skills and algorithms	Skills used in painting with watercolors, whole-number division algorithm
CB. Knowledge of subject-specific techniques and methods	Interviewing techniques, scientific method
CC. Knowledge of criteria for determining when to use appropriate procedures	Criteria used to determine when to apply a procedure involving Newton’s second law, criteria used to judge the feasibility of using a particular method to estimate business costs.
D. Metacognitive knowledge – Knowledge of cognition in general as well as awareness and knowledge of one’s own cognition	
DA. Strategic knowledge	Knowledge of outlining as a means of capturing the structure of a unit of subject matter in a text-book, knowledge of the use of heuristics.
DB. Knowledge of subject-specific techniques and methods	Knowledge of the types of tests particular teachers administer, knowledge of the cognitive demands of different tasks
DC. Knowledge of criteria for determining when to use appropriate procedures	Knowledge that critiquing essays is a personal strength, whereas writing essays is a personal weakness; awareness of one’s own knowledge level

Table 2.2: Cognitive Process Dimension [4]

PROCESS CATEGORIES	COGNITIVE PROCESSES AND EXAMPLES
1. Remember – Retrieve relevant knowledge from long-term memory.	
1.1 Recognizing	(e.g., Recognize the dates of important events in U.S. history)
1.2 Recalling	(e.g., Recall the dates of important events in U.S. holiday)
2. Understand – Construct meaning from instructional messages, including oral, written, and graphic communication	
2.1 Interpreting	(e.g., Paraphrase important speeches and documents)

2.2 Exemplifying	(e.g., Give examples of various artistic painting styles)
2.3 Classifying	(e.g., Classify observed or described cases of mental disorders)
2.4 Summarizing	(e.g., Write a short summary of the events portrayed on videotapes)
2.5 Inferring	(e.g., In learning a foreign language, infer grammatical principles from examples)
2.6 Comparing	(e.g., Compare historical events to contemporary situations)
2.7 Explaining	(e.g., Explain the causes of important eighteenth-century)
3. Apply – Carry out or use a procedure in a given situation	
3.1 Executing	(e.g., Divide one whole number by another whole number, both with multiple digits)
3.2 Implementing	(e.g., Determine in which situations Newton’s second law is appropriate)
4. Analyze – Break material into constituent parts and determine how parts relate to one another and to an overall structure or purpose	
4.1 Differentiating	(e.g., Distinguish between relevant and irrelevant numbers in a mathematical word problem)
4.2 Organizing	(e.g., Structure evidence in a historical description into evidence for and against a particular historical explanation)
4.3 Attributing	(e.g., Determine the point of view of the author of an essay in terms of his or her political perspective)
5. Evaluate – Make judgments based on criteria and standards	
5.1 Checking	(e.g., Determine whether a scientist’s conclusions follow from observed data)
5.2 Critiquing	(e.g., Judge which of two methods is the best way to solve a given problem)
6. Create – Put elements together to form a coherent or functional whole; reorganize elements into a new pattern or structure.	
6.1 Generating	(e.g., Generate hypotheses to account for an observed phenomenon)
6.2 Planning	(e.g., Plan a research paper on a given historical topic)
6.3 Producing	(e.g., Build habitats for certain species for certain purposes)

By applying the Bloom taxonomy in education, educators can examine objectives from the student’s point of view and consider the panorama of possibilities

in education. Most important of all, it can help educators see the integral relationship between knowledge and cognitive processes inherent in objectives.

In summary, how to help the teachers who have little teaching experience to design the learning activity compliant with Learning Design and adapt Bloom taxonomy as course assessment is an important issue.



Chapter 3. Scheme of Learning Activity

Recommendation

As mentioned previously, the standard trend about LD plus SCORM must be the main stream in e-learning. LD's powerful expression ability of learning activity is attracting more and more scholars and instructional designers to develop related tools or systems. Under the circumstances, the issue of learning activities management is arising. Based on the management issue, the first important thing is how to reuse these learning activities. Besides, the reusing mechanism must be efficient to simplify users edit their learning activities. As we know, the real methodology hasn't been proposed yet. In our related work, only some scholars proposed the need in the future. Besides, each learning activity has its unique structure. So how to model it and how to design the similarity functions are another issues. The LAR (Learning Activity Recommendation) we propose in this chapter will solve the above problems.

According the articles in these websites: JISC [16] and CETIS [17] , the reusing unit should be a completed learning design (learning activity) rather than its subset. That is the reason why we choose a completed learning activity to be our reusing unit in this thesis.

In the following section, the data presentation is discussed. And then the scheme of LAR is introduced in Section 3.2.

3.1 LAR Scheme

The procedure of LAR that we propose to solve the issues mentioned above is shown in Figure 3.3.

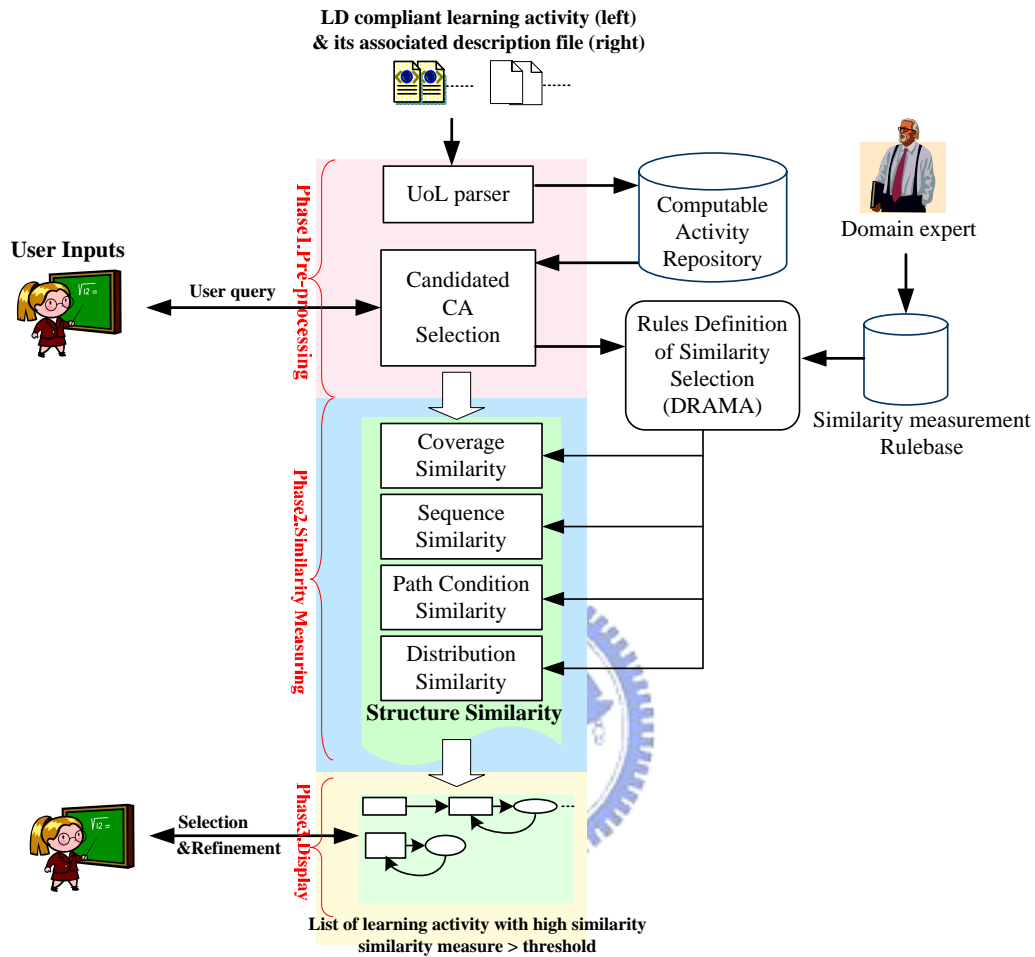


Figure 3.1: Scheme of LAR

The LAR is composed of three phases:

Phase 1. Pre-Processing deals with the raw data and determines how to calculate the similarity in phase 2 by inference. It includes: *UoL Parser*, *Candidated CA Selection (CCAS)*, and *Rules Definition of Similarity Selection (RDSS)*. Their functionality is described as following:

- **UoL Parser**: it transforms the LD compliant learning activity associated with meta-data into the data representation, Computable Activity, which we defined in

next section. The parsed CAs will be stored in the Computable Activity Repository (CAR).

- **Candidated CA Selection (CCAS):** it filters the CAs in CAR according to the user's input. So the number of CAs that needs to be calculated in Phase 2 would be reduced.
- **Rules Definition of Similarity Selection (RDSS):** it handles the interaction with users and generates the similarity equation based on the user's input parameters. The infer mechanism is based on the rules in the Similarity Measurement Rule Base (SMRB). These rules are defined by domain experts. The RDSS provides quite flexibility to extend the similarity calculations.

Phase 2. Similarity Measuring calculates the similarity value of each CAs of *CCAS* according to the similarity equation that inferred from *RDSS*. As we analyze the structure of a learning activity, we propose four similarity functions as follows.

- **Coverage Similarity (CS):** it computes the similarity level between the courses the learning activities has and the courses users want to use.
- **Sequence Similarity (SS):** it computes the similarity level between the courses' learning flow in the learning activities and the learning flow users want.
- **Path Similarity (PS):** it computes the similarity level between the path condition of each course in a learning activity and the path condition users want.
- **Distribution Similarity (PS):** it computes the similarity level to match the user's need of pre-requisite.

The inference engine of *RDSS* will trigger the suitable functions to compute the similarity value.

Phase 3. Display lists the learning activities to end-users in the order that depend on

their similarity score from high to low. Users can choose their favorite learning activity recommend by LAR to do some refinement that make it become their desired learning activity.

The process in Figure 3.1 can be described as follows. First, the user inputs their queries by interacting with *RDSS*. The query includes two parts: the requirements about learning activity and the meta-data. *UoL Parser* transforms the original learning activity documents compliant with LD (xml file) which describes a learning activity into Computable Activity (CA). And *CCAS* gets the meta-data in user's query to reduce the searching space. The *RDSS* not only controls the interactive input interface with users but also generates the suitable similarity measurement equation that is combined with the similarity formulas according to the rules defined by domain experts in the *SMRB*. Then the similarity calculation is executed based on the similarity equation. After computing similarity functions, the learning activities will be ranked decreasingly depending on the similarity value and be shown to end-users to refine.

These phases will be introduced in detail in Chapter 4. In order to compute the similarity between user's queries and these LD compliant learning activities, we propose a data representation which is called Computable Activity (CA). In the following section, its definition is shown explicitly.

3.2 Data Representation in LAR Scheme

We define the storage unit for reusing is a **Computable Activity (CA)**. A CA

describes the learning flow in an activity and some meta-data associated with the learning activity. The formula of CA can be illustrated as followings:

$$CA = \{ \text{Meta-data, unit of learning (UoL)} \}$$

In the LAR scheme, the place to store the unit of learning associates with its meta-data for reusing is the repository which is called Computable Activity Repository (CAR). We assume all CA stored in CAR must be represented in Learning Design Standard (LD) and verified by education domain experts.

The Meta-data is used for assisting our similarity computation. It is a pair with attribute and corresponding value. The following table shows the attributes we maintain and their description.

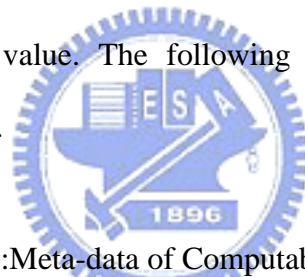


Table 3.1: Meta-data of Computable Activity

Attribute	Description
Suitable-Learning-Target	It refers to which grades students can attend to this learning activity.
LD-Level	It describes what kind of LD this learning activity matched. Its value are: A or B or C.
Role	It describes what kinds of Roles are involved in this learning activity.
Depth	The Longest path from start course to final course.

In Chapter 2, we mentioned the definition of UoL. Here we'll reduce the range of UoL. In this thesis, we focus on the Organizations: Learning Design which is the core of the UoL. The UoL is composed of a set of **Activity Structures (AS)** and a method including several rule definitions which can be used to control the learning guidance.

An AS in turn consists of references to one or more of: a learning activity, a support activity, an (sub) Activity-structure, and another (separate) unit of learning. A learning activity is directed at attaining a learning objective per individual user. Any user performs a learning activity only once (until completion). It can be a text, an audio-file, a video file or any other cue to the user. A support activity is meant to facilitate a role performing one or more learning activities. The part to joint these activities is the rules in method. So we can adapt the UoL for a graphical based representation. We first describe some definitions and then illustrate the graphical formula and a figure to depict.

Definition 3.1: Unit of Learning (UoL)

$UoL = \{ V, E \}$, where

- $V = \{v_1, v_2, \dots, v_m\}$. In UoL, V denotes an activity which can be the Learning Activity (LA) or Support Activity (SA). Several activities are organized into an activity structure (AS).
- $E = \{e_1, e_2, e_3, \dots, e_n\}$, where $e_i = \langle A_i, A_j \rangle$

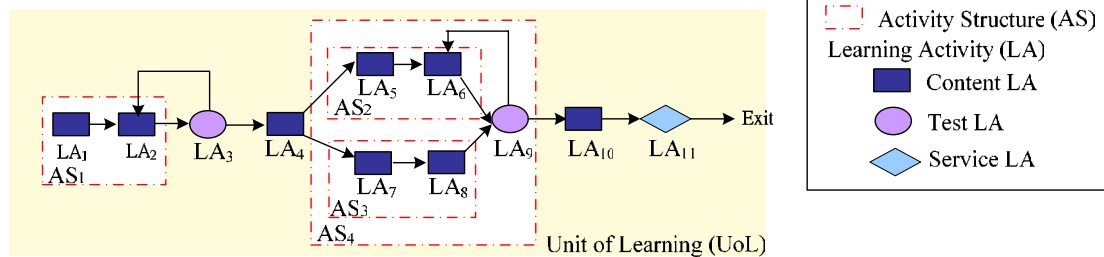


Figure 3.2: Unit of Learning

Figure 3.1 represents a UoL that has 11 learning activities and edges. These activities are connected by the edges. In this example, (LA₁ & LA₂), (LA₅ & LA₆), and (LA₇ & LA₈) are grouping into AS₁, AS₂, and AS₃ respectively. The AS₄ is composed of AS₂ and AS₃. LA₃ is a Test LA, so there are two edges spilt from here:

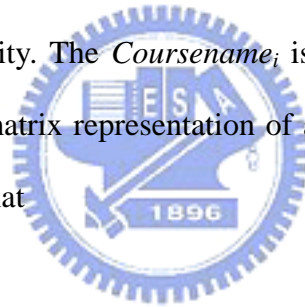
one is going backward to LA₂ and another one is continuing learning LA₄. This is a kind of remedy teaching strategy. So we can learn that different UoL represents different unique learning flow.

In order to calculate the similarity between user's query and UoLs, we apply a matrix called **Computable Activity Matrix (CAM)** to represent the learning flow of UoL. The related definitions and example are shown as followings.

Definition 3.2: Computable Activity Matrix (CAM)

CAM is the adjacency-matrix representation of the graph UoL, we assume that the vertices are presented as *Coursename₁*, *Coursename₂*, ..., *Coursename_n* in the order as in the learning activity. The *Coursename_i* is presented as *i* for short in the matrix. Then the adjacency-matrix representation of a graph UoL consists of a $|V| \times |V|$ matrix CAM = (a_{ij}) such that

$$a_{ij} = \begin{cases} 1, & \text{if } \langle i, j \rangle \in E \\ 0 & \text{otherwise.} \end{cases}$$



Example 3.2: The CAM and Meta-data of Figure 3.2

CAM_{11,11}

	1	2	3	4	5	6	7	8	9	10	11
1	0	1	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0
3	0	1	0	1	0	0	0	0	0	0	0
4	0	0	0	0	1	0	1	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0	0
6	0	0	0	0	0	0	0	0	1	0	0
7	0	0	0	0	0	0	0	1	0	0	0
8	0	0	0	0	0	0	0	0	1	0	0
9	0	0	0	0	0	0	0	0	0	1	0
10	0	0	0	0	0	0	0	0	0	0	1
11	0	0	0	0	0	0	0	0	0	0	0

Meta-data

Suitable-Learning-Target	6 grades
LD-Level	B
Roles	<learner>
Depth	8

This example describes the CAM and Meta-data of Figure 3.1. The attributes of CAM represent the names of learning activity of Figure 3.1. Their relationship is one-to-one mapping, such as 1 means LA₁. There is an edge connect LA₁ from LA₂, so the CAM₁₂ = 1; while there is no edge from LA₆ to LA₈, so the CAM₆₈ = 0. The values of LD-Level and Suitable-Learning-Target are gotten from the input document. By counting the longest path number between the start course:LA₁ and the final course: LA₁₁, the Depth is assigned to 8.

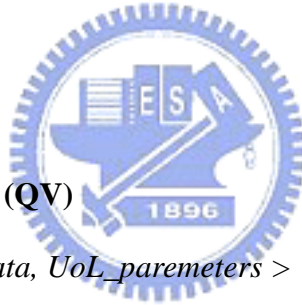


Chapter 4. Pre-Processing of LAR Scheme

Pre-processing can be used to transform data and reduce the searching spaces. There are three components in this phase: UoLP (UoL Parser), CCAS (Candidated Computable Activities Selection) and Rules Definition of Similarity Selection (RDSS). In the following sections, these components will be described in detail.

4.1 The Definition of User Input Parameters

The proposed scheme provides users with application user interface to execute queries and get results. We define the following query vector to record the user's requirements.



Definition 4.1: Query Vector (QV)

Query vector (qv) = $\langle \text{Mata-data}, \text{UoL_parameters} \rangle$

- *Mata-data* = $\{LD_level, SLA\}$, where
 - *LD_level*: the level of learning Design,
 - *SLA*: the suitable learning ages of the learner.
- *UoL_parameters* = $\{CN, CS, CP, PreR, Weights\}$, where
 - *CN* (*Course Name*): the course-names the user wants to use. In this thesis, the user can only choose the existing courses in our Computable Activity Repository.
 - *CS*: the courses sequence while navigating the user desired courses,
 - *CP*: the courses' navigating path condition. Users can fill the sequence types (condition, loop) they want between two courses. That means from the course_i to course_j, the navigation path is like what users defined in the candidated learning

activity.

- *PreR*: this parameter means pre-requisite of learning activity. The input type is boolean. If the value is true, that means LAR will search the learning activity that has more courses than user desired. Then these courses not in user's input parameters will become recommended courses for reference.

- *Weights*: the weight of each similarity calculation. If the user doesn't configure them, the *RDSS* will assign the default weights.

Example 4.1: User Input

Query vector (qv) = {(B, 6 grades), (A,C,D), (C A D), (C A:Condition, D A:Loop), Yes}

After calculating, the matched learning activities will be ranked in the order of their similarity scores and display to users. Users can edit their favorite one.



4.2 UoL Parser

The inputs of UoLP are the xml files compliant with LD. We have mentioned the xml structure of LD in Chapter 2.1. The following figure 3.4 is a simple transform concept.

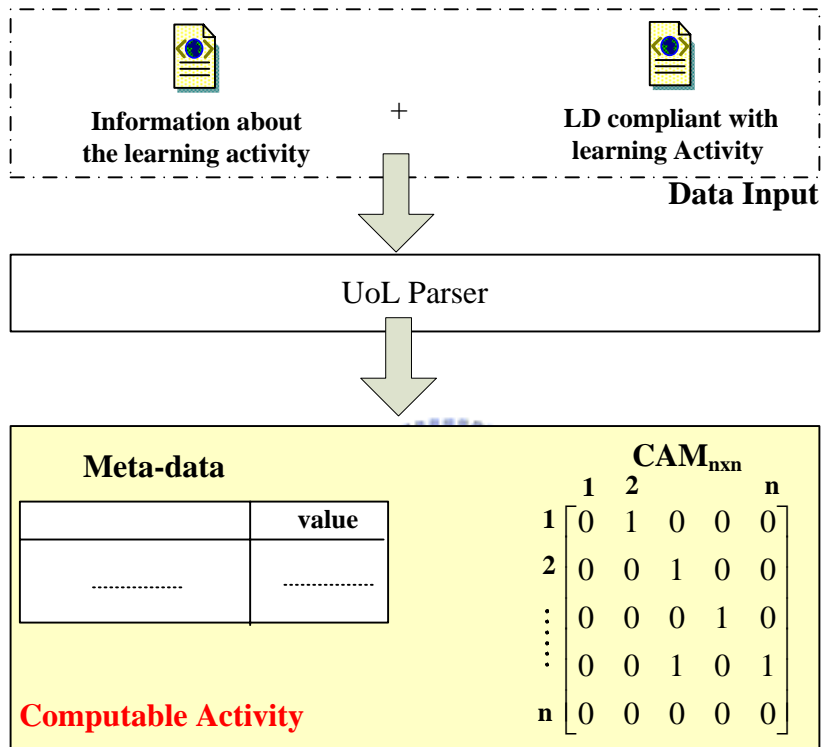


Figure 4.1: Concept of UoL Parser

The above figure depicts the concept of UoL Parser. Each learning activity and its associated description file are the input of UoL Parser. After the process of UoL Parser, the Computable Activity will be generated. The description file is a xml file we defined. Its format will be shown in Appendix 1. It contains most attributes of Meta-data, while the LD compliant with Learning Activity is transformed into the Computable Activity Matrix.

As we illustrated in Chapter 2, Learning Design has a powerful ability of expressing the learning scenario. Hence it has too many attributes so that is so

complicated. So, in the thesis, we focus on the following attributes to represent the Scheme, LAR.

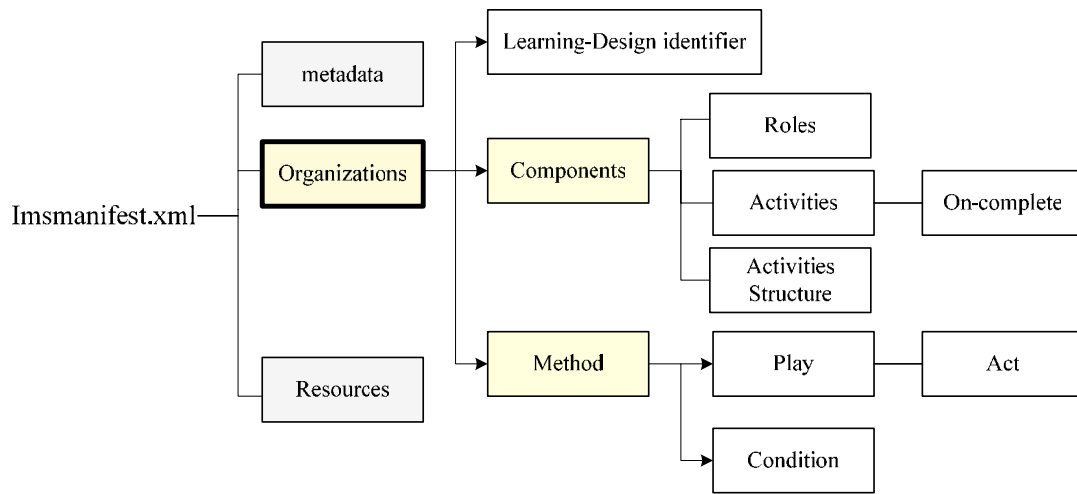


Figure 4.2: The hierarchy of attributes we used

The definition of UoL Parser algorithm is shown as follows:

Algorithm 4.1: UoL Parser Algorithm (UoLPAalgo)

Algorithm: UoL Parser Algorithm (UoLPAalgo)

Definition of Symbols:

Roles = {role₁, role₂, ... , role_n}: it is a set of roles which is used to record what kind of roles participates in this unit of learning.

TmpActivity: it represents a structure that record this activity's identifier, title, its related properties and status after changing.

CAM_Attri: it represents these learning activities in this unit of learning in sequence.

TmpCondition: it represents the conditions of all properties in this unit of learning. It records the property' name, its changed criteria, and the results.

Input: LD compliant with learning activities and their description file.

Output: Meta-data and CAM

Step1. Read the description file.

Step1.1. Set each attributes of description file to the corresponding attributes in Meta-data.

Step2. Read the LD compliant with learning activities into DOM tree.

Step 2.1 Read the element “learning-design identifier”.

Step 2.2 Read the value of its attribute “level” into the “LD-Level” of Meta-data.

Step 2.2 Enter the element “components”.

Step 2.3 Read the element “roles”.

Step 2.4 Add each value of “learner identifier” into **Roles**.

Step 2.5 Set **Roles** into Meta-data.

Step 2.5 For each element “activities”

Step 2.5.1 set each related value into **TmpActivity**.

Step 2.5.2 Initialize eacho elements of $CAM_{nn} = 0$ where n is the number of learning activities in this unit of learning.

Step3. Read the element “Method”.

Step 3.1 Read the element “conditions”.

Step 3.1.1 set each “if-else” into **TmpCondition**.

Step 3.2 Enter the element “role-part”.

Step 3.2.1 Read the activity-structure-ref of each role-part.

Step 3.3 Read each activity-structure-ref recursively.

Step 3.3.1 Expand each learning activity in the activity structure

Step 3.3.2 Assign the learning activity’s name into **CAM_Attri**.

Step 3.3.3 Read the element “on-complete” of the learning activity.

Step 3.3.4 Map the changed-property with the **TmpCondition**.

Step 3.3.5 If there are any learning activity are shown after this property changed, appdend it to the **CAM_Attri**.

Step 3.3.6 Mark the $CAM_{ij} = 1$ if the learning acitivity i can achieve the learning activity j

4.3 Candidated Computable Activities Selection (CCAS)

The processing of LAR may take a great deal of time to calculate the similarity scores of all the learning activities while there are a large number of CAs in CAR. However, the performance is a key index for modern computer systems; a candidated computable activity selection (CCAS) to avoid calculating all CAs during the retrieval process can be very helpful to improve the performance.

The attribute to filter CA in CCAS are the Suitable Learning Target, and LD level in meta-data parameters. Here, we adopt the principle of “exactly match”. The concerned attributes can still be extended if there is more information about learning activities in the future. The CCAS algorithm is shown below:

Algorithm 4.2: Candidated Computable Activity Selection Algorithm (CCASAlgo)

Algorithm: Candidated Computable Activity Selection Algorithm (CCASAlgo)

Input: All learning activities in LAKB

Output: Only the learning activities that match the user’s input attribute.

Step 1: For each learning activity in LAKB, check their meta-data.

Step 1.1: Check the level of LD.

Step 1.2: Check the age of students.

Step2: If any attribute in a learning activity is not equal to user’s input parameters, then ignore this learning activity.

4.4 Rules Definition of Similarity Selection (RDSS)

After Candidated Computable Activity Selection, the inference engine will infer the suitable similarity equation according the user's inputs and rules in the similarity measurement rule base (SMRB). By adapting this inference mechanism, the similarity measurement will be more flexible and scaleable. In NORM [15], a new knowledge model, proposed to process knowledge modularization and knowledge relations, is used to represent the generating equation rule in LAR. In the rule base, there is a rule class, called **MainRC**. MainRC is used for controlling the interactive interface between LAR and users and the sequencing of the similarity calculation. In the thesis, we concern four similarity calculations: Coverage Similarity (CS) which concerns the possession rate of courses that users need, Sequence Similarity (SS) which concerns the learning order, Path Similarity (PS) which concerns the types of navigating paths in the UoL, and Distribution Similarity (DS) which concerns the need of pre-requisite courses. Following is illustrated the example of our rule base.

Rule base = {MainRC }

MainRC contains these rules:

Facts: CS, SS, PS, DS (boolean)

Rules:

R1: If CS, then SS

R2: If CS & SS, then PS

R3: If CS, then DS

The rules above are defined in what condition which similarity calculation should be executed. In the similarity we concern, the basic calculation is Coverage

Similarity (CS). Other three similarity calculation are based CS. So the user must input the course names for CS, or the LAR will only search the activities that match the meta-data of the user's desire. There is another dependency relation of SS and PS. So in R2, they have the "and" relationship in rule condition.

The similarity score will be the summary of each similarity score multiple their corresponding weight.

$$\mathbf{Score} = W_C \times S_C + W_S \times S_S + W_P \times S_P + W_D \times S_D, \sum W_i = 1$$

The weights corresponding to each similarity calculation are obtained from two ways: the system default weight or user configuration. User can configure these weights according their need while interacting with IE. If users don't assign any weight, the system default weight is shown as following formula:

$$w = \frac{1}{num(Fact)}$$

Example 4.3: weight configuration

If the user don't input any parameter and only the fact CS, SS are true, the w will

become: $w = \frac{1}{2}$. And the **Score** = $\frac{1}{2} \times S_C + \frac{1}{2} \times S_S + 0 \times S_P + 0 \times S_D$

■

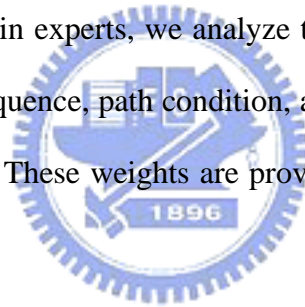
The detail of calculating Similarity will be introduced in Chapter 5.

Chapter 5. The Methodology of Similarity

Measurement

In LAR, the most important part is to calculate the similarity function between user's query and computable activity (CA) in CAR. In this chapter, we will introduce our similarity measurement.

The most important part in a learning activity is its learning flow. A well-structured learning flow will affect the outcome of learning. After discussing with some teachers and domain experts, we analyze the structure similarity based on four viewpoints: coverage, sequence, path condition, and distribution. Each viewpoint has its corresponding weight. These weights are provided by our inference engine or configured by users.



In the following section, we'll introduce the calculation of each structural viewpoint more detailed.

5.1 Coverage Similarity (CS)

The coverage means that the number of courses that user wants are in the learning activities to compare with. Here, we define a heuristic function called $Match(C_i)$. This function is used for detect that if the course of query is in the comparing learning activity. Following formulas are used for computing Coverage Similarity.

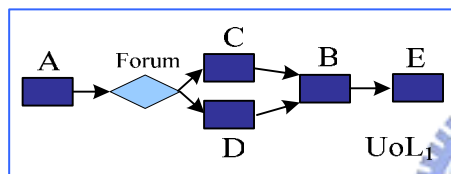
$Match(C_i) = 1$ if the learning activity has course i that user want.

$user_query$ = total courses that user inputs.

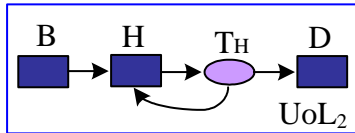
$$Score = \frac{\sum Match(C_i)}{\sum user_query}$$

Example 5.1: Coverage Similarity

There are two learning activities below. Now a user inputs that he/she wants to search a learning activity that has courses: A, B, and D. We can calculate the coverage similarity score of each learning activity by the formula.



$$Score(LA_1) = \frac{1+1+1}{3} = 1$$



$$Score(LA_2) = \frac{1+1}{3} = \frac{2}{3}$$

LA_1 has the courses A, B, and D that exactly matched the user's inputs, so LA_1 gets the score: 1; while LA_2 only has the courses B and D, so it only gets the score:

$$\frac{2}{3}$$



5.2 Sequence Similarity (SS)

In learning activity, another important part is the learning sequence. The proper arrangement of the courses will affect the students' learning achievement. As Section 3.2 illustrated, we use the CA to compute the similarity. The characteristic, transport closure, of CAM can present the path condition step by step while traversing the UoL in CA. And the depth of Meta-data in a UoL refers to the CAM's multiplication times. We define each multiplication result, M^n , $n = 1 \dots \text{Depth of Meat-data}$. By doing the "or" operation to each M^n , we will get another new matrix, CheckedMatrix (CM). Later we will explain how to use CM to calculate the sequence similarity. The calculating formula is illustrating following:

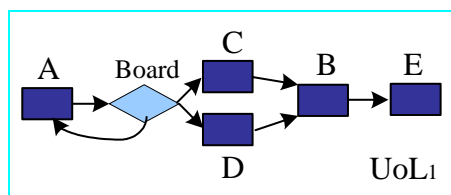
$$\text{Score} = \frac{OMP}{C_2^n}, \text{ where}$$

- n : the number of courses that user input as sequence parameter,
- C_2^n : The combination of courses. For example, if user inputs courses in order: A B D. There are 3 combinational pairs: (A,B), (B,D), (A,D),
- OMP: Order Matched Pattern.

Example 5.2: Sequence Similarity

In this example, the user inputs the courses sequences he wants: "A D B".

The detail computing process will be seen in LA₁.



$$CM_{LA1} = \begin{matrix} & \begin{matrix} A & Board & C & D & B & E \end{matrix} \\ \begin{matrix} A \\ Board \\ C \\ D \\ B \\ E \end{matrix} & \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{matrix}$$

$$\begin{matrix} \text{CAM}^1 \\ \begin{array}{c} \text{A} \\ \text{Board} \\ \text{C} \\ \text{D} \\ \text{B} \\ \text{E} \end{array} \end{matrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{matrix} \text{CAM}^2 \\ \begin{array}{c} \text{A} \\ \text{Board} \\ \text{C} \\ \text{D} \\ \text{B} \\ \text{E} \end{array} \end{matrix} \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

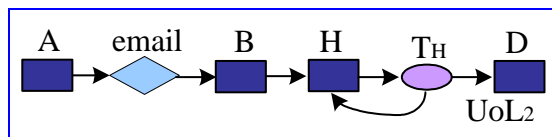
$$\begin{matrix} \text{CAM}^3 \\ \text{CAM}^2 \times \text{CAM}^1 = \end{matrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 2 & 0 \\ 1 & 0 & 1 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{matrix} \text{CAM}^4 \\ \text{CAM}^3 \times \text{CAM}^1 = \end{matrix} \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 2 \\ 0 & 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\text{CAM}^1 \cup \text{CAM}^2 \cup \text{CAM}^3 \cup \text{CAM}^4 = \text{CM}_{\text{LA}_1}$$

By checking the $\text{CM}[\text{A}][\text{D}]$, $\text{CM}[\text{A}][\text{B}]$, and $\text{CM}[\text{D}][\text{B}]$, we can count the sequence similarity score of LA_1 .

$$\text{Score}(\text{LA}_1) = \frac{(1+1+1)}{3} = 1$$

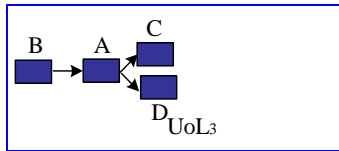
The remaining two calculations of sequence similarity are based the same approach.



$\text{CM}_{\text{LA}_2} =$

$$\begin{matrix} \begin{array}{c} \text{A} \\ \text{Email} \\ \text{B} \\ \text{H} \\ \text{TH} \\ \text{D} \end{array} \end{matrix} \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\text{Score}(\text{LA}_2) = \frac{(1+1)}{3} = \frac{2}{3}$$

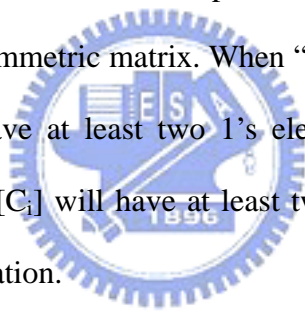


$$\text{Score(LA}_3) = \frac{1}{3}$$



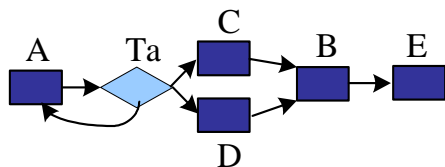
5.3 Path Similarity (PS)

This part is used for detecting the learning sequence condition. It's an extended request of sequence similarity. By means of the characteristics of $CAM_{n \times n}$, the path condition will be discovered. There are three path conditions that can be detected: "Loop", "Condition", and "And". When "Loop" occurs, if the path is from C_j to C_i , $CAM_{\max(i,j) \times \max(i,j)}$ will be a symmetric matrix. When "Condition" occurs, if the branch node is C_i , $CAM[C_i]$ will have at least two 1's elements. If "And" occurs at the convergence node C_i , $CAM[][C_i]$ will have at least two 1's elements. The following example illustrates the observation.



Example 5.3: Path Similarity

There under we show a learning activity and its $CAM_{n \times n}$. By examining the markers of $CAM_{n \times n}$, the path conditions reveal.



$$CAM_{n \times n} \begin{matrix} & A & T_A & C & D & B & E \\ A & 0 & 1 & 0 & 0 & 0 & 0 \\ T_A & 1 & 0 & 1 & 1 & 0 & 0 \\ C & 0 & 0 & 0 & 0 & 1 & 0 \\ D & 0 & 0 & 0 & 0 & 1 & 0 \\ B & 0 & 0 & 0 & 0 & 0 & 1 \\ E & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

The circle marks the condition "Loop" from C_{T_A} to C_A . It is a symmetric

sub-matrix of $A_{2 \times 2}$. The $A[T_A][C]$ and $A[T_A][D]$ reveal that there is a conditional branch at C_{Ta} . While $A[C][B]$ and $A[D][B]$ show that there is a merge condition at C_B .

Here we only concern two conditions: “Loop” and “Condition”. Because the two conditions are often used for remedy learning in education [5]. User’s input may like “ $C_A \rightarrow C_B$, Condition”. The score formula is as following:

$$\text{Score} = \frac{PMP}{RMP}, \text{ where}$$

- PMP: Path Matched Pattern,
- RMP: Required Matched Pattern.

The following two algorithms are for detecting these path conditions.

Algorithm 5.1: Loop Path Detection Algorithm (LPDAlgo)

Algorithm: Loop Path Detection Algorithm (LPDAlgo)

Symbol Def:

C_s : start course

C_e : end course

Input: Which kind of paths from C_i to C_j

Output: The boolean value

Step 1: Check $Mn[C_s][C_e] = 1$

Step 1.1 Check index of $C_s > C_e$. If true, then Loop = true , else Loop = false.

Algorithm 5.2: Condition Path Detection Algorithm (CPDAlgo)

Algorithm: Condition Path Detection Algorithm (CPDAlgo)

Input: Which kind of paths from C_i to C_j

Output: The boolean value

Step 1: At $Mn[Cs][Ce] = 1$

Step 1.1 for ($i=index(Ce);i<total_course;i++$)

Check $Mi[Cs][i] == 1$.

Step2: If true, then Condition = true ,else Condition = false.

5.4 Distribution Similarity (DS)

The Distribution here means the pre-requisite in a learning activity. Briefly speaking, a longer learning activity means that there are more pre-requisites than short one. The similarity calculation provides an optional service to users especially for the teachers that have fewer experiences in instructional design. This function offers users what should be learned that others think for reference. The following formula represents the score calculating mechanism.

$$\text{Score} = \frac{\sum Path(C_i, C_j)}{10^{\text{len}(x = \text{MAX}(L A_i))}}, \text{ where}$$

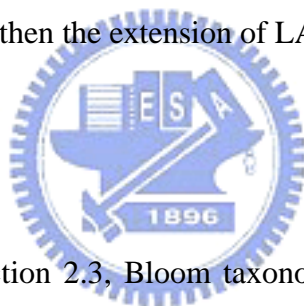
- $\sum Path(C_i, C_j)$: the # of edges from C_i to C_j . Where $i < j$
- $\text{len}(x) = \ell \text{MAX}(L A_i)$
- $\text{MAX}(L A_i) = n$, n = the depth that the longest LA has

Concerned with the normalization, the denominator of the function is based on the longest learning activity at each selection interval. The length will be the base number 10's exponent. Then the score will be normalized between 0~1.

Chapter 6. Applying Pedagogy Theory to Computable Activity

The pedagogy theory plays an important role in education. If the teacher can adopt suitable pedagogy theory into the appropriate learning activities, the learner could learn well. At present, many pedagogy theories have been proposed. In Chapter 3, we have proposed the LAR scheme to reuse learning activities by similarity measurement. The scheme can be extended to support diverse similarity functions based on these pedagogy theories. Here, we take the famous taxonomy for Educational Objectives, Bloom, for example. First the similarity calculating mechanism is introduced, and then the extension of LAR will be described in detail.

6.1 Bloom Similarity



As we mentioned in Section 2.3, Bloom taxonomies has become an important assessment while designing a learning activity. The purpose of this similarity calculation can help teachers find a learning activity with desired courses and associated Bloom. In order to take Bloom into the similarity measurement of LAR, we first define a structure for Bloom which is called BI and then show the extension Computable Activity (CA) formula.

Definition 6.1: BI (Bloom Indicator)

BI is a mapping table that records courses and their corresponding Bloom taxonomies. The key set is courses' name. And each key refers to a set of Bloom taxonomy. The notation is as following:

$$(C_i, \langle B_1, B_2, \dots, B_j \rangle), i = 1 \dots m, j = 1 \dots n.$$

Take Figure 3.2 for example. Its BI may look like the following table:

Table 6.1:BI of figure3.2

Name	Mapping Bloom Category
LA ₁	{AA1.1, BC2.3}
LA ₂	{AA1.2}
.....
LA ₃	{AA1.3, C1.1, DC1.2}

Therefore, the formula of CA can be extended as followings:

$$CA = \{ \text{Meta-data, unit of learning (UoL), BI} \}$$

In the Bloom Similarity calculation, we encode Bloom taxonomy into four-bit representation, called “Bloom Pattern” (BP). As we know, A BP divides two dimensions and each dimension has two hierarchical levels. The former two bits stand for knowledge dimension (KD) and the later two bits mean cognitive processing dimension (CPD). No matter how precisely each dimension allocated, a BP just has the flex length, 4. Take following pictures for example:

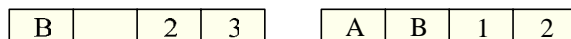


Figure 6.1: Example of BP

According to the precision level, we design two functions to give user flexibility while counting the Bloom Similarity. The basic function only computes the similarity while two BP’s prefix are the same and the reinforced function can compute the

similarity between two BPs that have different prefix.

Besides the functions, we also provide the idea of important weight. For two dimensions, we give two important weights respectively: α , β . User can adapt each weight to their need. The score of Bloom Similarity can be illustrated as following:

$$\text{Score} = \alpha \sum_{i=1}^{i=2} \text{Match}_i(q_i, BP_i) + \beta \sum_{j=3}^{j=4} \text{Match}_j(q_j, BP_j)$$

6.1.1 Basic Function

This operation held if the prefixes of two BPs are the same either in the KD or CPD. The following formula describes the rule while calculating the similarity between user's query and the target courses' Bloom taxonomy.

$$\text{Match}_i(q_i, BP_i) = \begin{cases} 1, & \text{if } (q_i = BP_i \mid i=4), q_3 = BP_3 \\ 0.5, & \text{if } BP_i \text{ not exist \& } BP_{i-1} = q_{i-1}, i = 2, 4 \\ 0, & \text{otherwise} \end{cases}$$

We can learn that in basic function, while comparing bit i , either KD or CPD, as long as the prefix of query and target are the same and $query_i = target_i$, $\text{Match}_i(q_i, BP_i)$ is equal to 1. In order to compare two Bloom taxonomies in different hierarchical level, we design a heuristic strategy to calculate. For example, the similarity score between the set of "A" and "AB" and the set of "AA" and "AB", the "A" will get score, 1.5. The empty part earns score, 0.5; while "AA" only gets score, 1. That's because "A" is in higher concept, it may include the concept "AA" and "AB". So it counts. But "AA" and "AB" is exclusive part under "A". So the "B" of "AB" will get nothing. We can figure our function by the following example.

Example 6.1: Bloom Similarity

The user wants to find a course named “A” corresponding to the Bloom taxonomy, AA1.1. Now in our candidated learning activities (CLA), there are 5 LAs that have the course “A”. The important weight defaults to be 0.5 respectively. Reader can notice that the calculation of LA₄ in basic function.

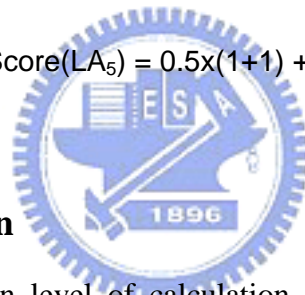
$$\boxed{\text{LA}_1 : \text{A1.2}} \quad \text{Score}(\text{LA}_1) = 0.5 \times (1+0.5) + 0.5 \times (1+0) = 1.25$$

$$\boxed{\text{LA}_2 : \text{AA2.1}} \quad \text{Score}(\text{LA}_2) = 0.5 \times (1+1) + 0.5 \times (0) = 1$$

$$\boxed{\text{AA1.1}} \quad \boxed{\text{LA}_3 : \text{AB1.1}} \quad \text{Score}(\text{LA}_3) = 0.5 \times (1+0) + 0.5 \times (1+1) = 1.5$$

$$\boxed{\text{LA}_4 : \text{BC1.2}} \quad \text{Score}(\text{LA}_4) = 0. \quad \underline{\text{BC}} \text{ vs. } \underline{\text{AA}}$$

$$\boxed{\text{LA}_5 : \text{AA1.2}} \quad \text{Score}(\text{LA}_5) = 0.5 \times (1+1) + 0.5 \times (1+1) = 1.5$$



6.1.2 Reinforced function

To enhance the precision level of calculation, reinforced function provides a flexibility to compute the similarity between two Bloom taxonomies that have difference prefixes. Here, we adopt the computing principle bases on the concept “distance in a tree”. First, we transform the two dimension of Bloom into a tree structure (Figure 6.2).

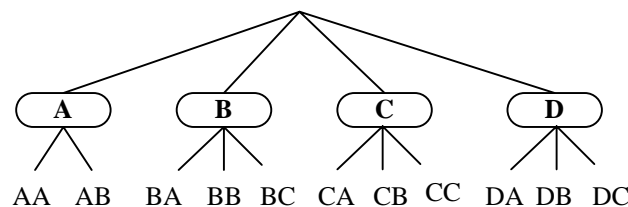


Figure 6.2: Knowledge Dimension Tree (Up)

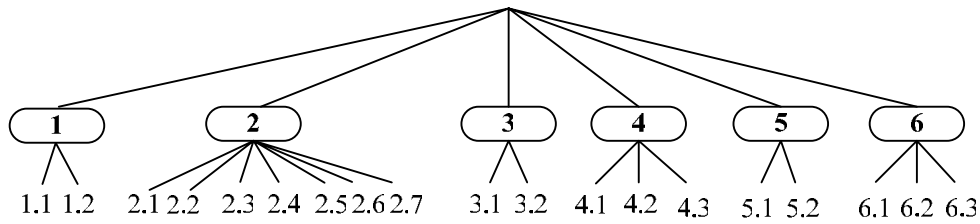


Figure 6.3: Cognitive Processing Dimension Tree (CPDT)

We use the algorithm, “Preorder Traversal”, to compute the similarity between two Bloom categories with different prefix. For each node, it can get a path length while traversal recursively from root. Here we also face the problem that compares node at different level. The solution is to compute their average path. The adaptation formula, $Match(q_i, BP_i)$, is as following:

$$Match_i(q_i, BP_i) = \begin{cases} 1 - \frac{Path(q, BP)}{total_length} & , \text{ if } q \text{ and } BP \text{ are at the leaf level} \\ 1 - \frac{AvgPath(q, BP)}{total_length} & , \text{ otherwise} \end{cases}$$

where

- $Path(q, BP)$ denotes the distance between two points while traversing the tree. The value is the distance from root to right node minus the distance from root to left node.
- $AvgPath(q, BP)$ denotes the distance is calculated by the average of shortest path and longest path between two Bloom taxonomies in different tree level or in the first level.
- $Total_Length$ denotes the total path length of the tree while traversing it.

Example 6.2: Reinforced Bloom Similarity

The user wants to find a course named “A” corresponding to the Bloom taxonomy, AA1.1. Now in our candidate computable activities (CCA), there are 3 LAs that have the course “A”. The important weight defaults to be 0.5 respectively. .

AA1.1	LA ₁ : BC1.2	$\text{Score}(\text{LA}_1) = 1 - (0.5 \times \frac{(12-2)}{30} + 0.5 \times \frac{(8-2)}{50}) = \frac{116}{150}$
	LA ₂ : CC2.4	$\text{Score}(\text{LA}_2) = 1 - (0.5 \times \frac{(20-2)}{30} + 0.5 \times \frac{(14-2)}{50}) = \frac{87}{150}$
	LA ₃ : D6	$\text{Score}(\text{LA}_3) = 1 - (0.5 \times \frac{(28+32)-2}{30} + 0.5 \times \frac{(44+48)-2}{50}) = \frac{14}{150}$

From these three examples, the longer path from AA1.1 will get the fewer scores. ■



6.1.3 The Extension of LAR

We have extended the formula of CA and explained how to compute the similarity of Bloom taxonomy above. Corresponding to the extension, the scheme is extended as shown in Figure 6.4. In this figure, the Bloom similarity function has been imported into the Phase 2. It is calculated before the Structure Similarity function. Along with the Bloom similarity, the components: *Query Vector* and *Rules Definition of Similarity Selection* and schema of the information description file, have to be extended to support the function. They will be described in Figure 6.4.

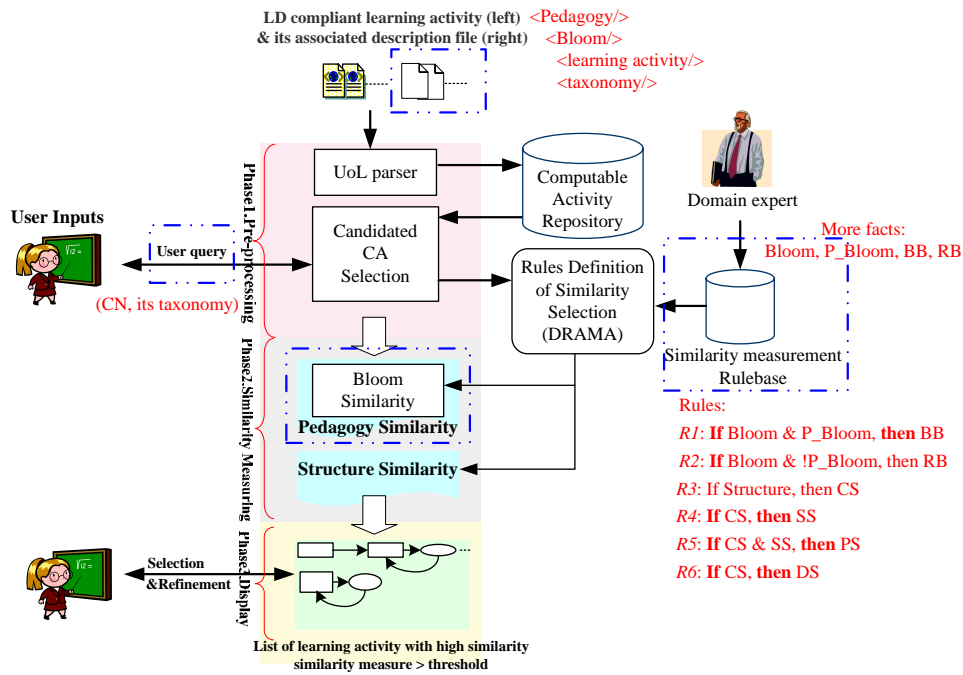


Figure 6.4: The Extended Scheme of LAR

As we mentioned in the previous page, the *Query Vector* and *Similarity Measurement Rule Base* are also extended to support the pedagogy theory. Their definitions are illustrated as follows.

Definition 6.2: The Extended Query Vector (QV')

Extended Query vector (qv') = $\langle \text{Meta-data}, \text{UoL_parameters}' \rangle$

- *Meta-data*
- $\text{UoL_parameters}' = \{ \text{CN}, \text{BI}, \text{CS}, \text{CP}, \text{PreR} \}$, where
 - *BI(Bloom Indicator)*: it represents the corresponding Bloom taxonomies of a *CN*. The definition is illustrated in Definition 6.1,
 - Others parameters are the same with Section 3.2.

We have described the extended qv' of LAR above. It stands to reason that if we want to calculate the Bloom similarity, the rules in *Rules Definition of Similarity*

Selection (RDSS) must expand to support this function. The following rules explained the extended RDSS.

Rule base = {MainRC' }

MainRC' contains:

Facts: Structure, Bloom, P_Bloom, BB, RB, CS, SS, PS, DS (boolean)

Rules:

R1: If Bloom & P_Bloom, then BB

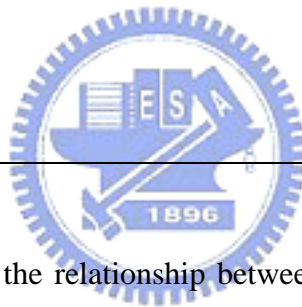
R2: If Bloom & !P_Bloom, then RB

R3: If Structure, then CS

R4: If CS, then SS

R5: If CS & SS, then PS

R6: If CS, then DS



These rules demonstrate the relationship between each similarity function. The Bloom function is divided into two precision levels according to the prefix. The default weight has changed a little. Here the default weight of Bloom function is 0.5 and the Structure function is the same. Unlike the Section 4.3, the weight of Coverage Similarity, Sequence Similarity, Path Similarity, and Distribution Similarity are summed up to 0.5. That's because these functions are derived from Structure Similarity function. Thus, LAR provides flexibility and scalability for designers to enhance the similarity functions they need.

6.2 Other pedagogy approaches

In the previous section, we adopt the famous taxonomy for learning objective, Bloom, into the scheme, LAR. There are still many pedagogy theories that can be modeled into the LAR. For example, we can define some learning patterns and transform them into the similarity functions, such as Problem-based learning, Inquiry-based learning, and so on. Although the LAR we proposed in this thesis haven't concern the similarity calculation of these pedagogy theory, it is full of flexibility to achieve this goal. Besides, LAR can be used for recommending any learning activities based on different standard that can be modeled as learning flow.

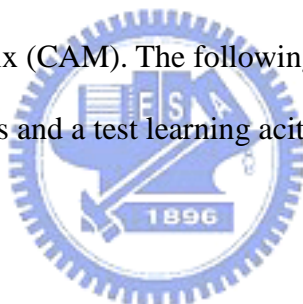


Chapter 7. System Implementation

For evaluating the performance of LAR, in the chapter, the experiment by synthetic computable activity matrix has been done. All the experiments are running on Intel P4 2.8 GHz processor with 512 MB DDR RAM, executed in the Windows 2000 Server operating system, and developed in the IDE, Eclipse, with jdk1.4.2_08 version.

7.1 Transformation Example of UoLParser

In this section, we demonstrate a transformation from a learning design unit to the Computable Activity Matrix (CAM). The following learning design unit contains four content learning activities and a test learning activity. The original file is illustrated as following:



```
<?xml version="1.0"?>
<!--Edited with XMLSPY Home Edition Version 2005 by Owen ONeill, Open University of the
Netherlands-->
<manifest xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"
xmlns:imsld="http://www.imsglobal.org/xsd/imsld_v1p0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.imsglobal.org/xsd/imscp_v1p1
http://www.imsglobal.org/xsd/imscp_v1p1p3.xsd http://www.imsglobal.org/xsd/imsld_v1p0
http://www.imsglobal.org/xsd/IMS_LD_Level_B.xsd" identifier="learningactivity-example">
<metadata>
  <schema>IMS Metadata</schema>
  <schemaversion>1.2</schemaversion>
</metadata>
<organizations>
  <imsld:learning-design identifier="Course-learningactivity" level="B"
uri="http://ou.nl/examplelearningactivity">
```

```

<imsld:title>Learning Activity Example With Conditions</imsld:title>
<imsld:components>
<imsld:roles>
<imsld:learner identifier="Learner">
  <imsld:title>Learner role</imsld:title>
</imsld:learner>
</imsld:roles>
<imsld:properties>
<imsld:locpers-property identifier="P-availability-examples">
  <imsld:datatype datatype="boolean"/>
  <imsld:initial-value>>false</imsld:initial-value>
</imsld:locpers-property>
</imsld:properties>

<imsld:activities>
  <imsld:learning-activity identifier="computer introduction">
    <activity-description>
      <item identifierref="RES-computer-introduction" identifier="I-computer-introduction">
    </activity-description>
  </imsld:learning-activity>
  <imsld:learning-activity identifier="software introduction">
    <activity-description>
      <item identifierref="RES-software-introduction" identifier="I-software-introduction">
    </activity-description>
  </imsld:learning-activity>
  <imsld:learning-activity identifier="hardware introduction">
    <activity-description>
      <item identifierref="RES- hardware -introduction" identifier="I- hardware -introduction">
    </activity-description>
  </imsld:learning-activity>
  <imsld:learning-activity identifier="Preparation">
<imsld:title>Optional Extra Help</imsld:title>
<imsld:activity-description>
  <imsld:item identifierref="R-Preparation" identifier="I-preparation"/>
</imsld:activity-description>
<imsld:complete-activity>
  <imsld:user-choice/>
</imsld:complete-activity>

```

```

<imsld:on-completion>
  <imsld:change-property-value>
    <imsld:property-ref ref="P-availability-examples" />
    <imsld:property-value>true</imsld:property-value>
  </imsld:change-property-value>
</imsld:on-completion>
</imsld:learning-activity>

<imsld:learning-activity identifier="Assignment-1">
  <imsld:title>Assignment - Answer these questions</imsld:title>
  <imsld:environment-ref ref="E-study-resources" />
  <imsld:activity-description>
    <imsld:item identifierref="R-Assignment-1" identifier="I-assignment-1"/>
  </imsld:activity-description>
  <imsld:complete-activity>
    <imsld:user-choice/>
  </imsld:complete-activity>
</imsld:learning-activity>

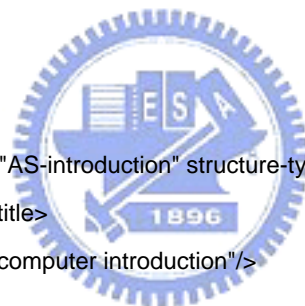
<imsld:activity-structure identifier="AS-introduction" structure-type="sequence">
  <imsld:title>Introduction</imsld:title>
  <imsld:learning-activity-ref ref="computer introduction"/>
</imsld:activity-structure>

<imsld:activity-structure identifier="AS-subject-introduction" structure-type="select"
number-to-select="1">
  <imsld:title/>
  <imsld:learning-activity-ref ref="software introduction"/>
  <imsld:learning-activity-ref ref="hardware introduction"/>
</imsld:activity-structure>

<imsld:activity-structure identifier="AS-test" structure-type="selection" number-to-select="2">
  <imsld:title>Learning Activities</imsld:title>
  <imsld:learning-activity-ref ref="Preparation"/>
  <imsld:learning-activity-ref ref="Assignment-1"/>
</imsld:activity-structure>

<imsld:activity-structure identifier="AS-learningactivity" structure-type="sequence" >
  <imsld:title>Learning Activities</imsld:title>
  <imsld:learning-structure-ref ref=" AS-introduction "/>
  <imsld:learning-structure-ref ref=" AS-subject-introduction "/>

```



```

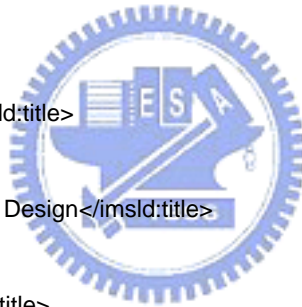
<imsld:learning-activity-ref ref="Preparation"/>
<imsld:learning-activity-ref ref="Assignment-1"/>
</imsld:activity-structure>
</imsld:activities>

<imsld:environments>
  <imsld:environment identifier="E-study-resources">
    <imsld:title>Study resources</imsld:title>
    <imsld:learning-object identifier="LO-article">
      <imsld:item identifierref="R-article" identifier="I-article"/>
    </imsld:learning-object>
  </imsld:environment>
</imsld:environments>

</imsld:components>

<imsld:method>
<imsld:play>
<imsld:title>learning LD activity</imsld:title>
<imsld:act>
<imsld:title>Learning about Learning Design</imsld:title>
<imsld:role-part>
<imsld:title>Role part learner</imsld:title>
<imsld:role-ref ref="Learner"/>
<imsld:activity-structure-ref ref="AS-learningactivity"/>
</imsld:role-part>
</imsld:act>
</imsld:play>
<imsld:conditions>
<imsld:if>
<imsld:is>
<imsld:property-ref ref="P-availability-examples"/>
<imsld:property-value>true</imsld:property-value>
</imsld:is>
</imsld:if>
<imsld:then>
<imsld:show>
  <imsld:class class="P-availability-examples" />

```

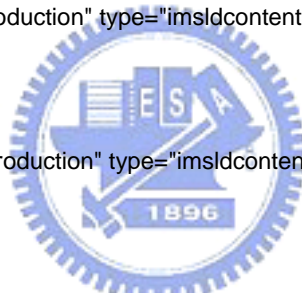


```

</imsld:show>
</imsld:then>
<imsld:else>
<imsld:hide>
<imsld:class class="P-availability-examples" />
</imsld:hide>
</imsld:else>
</imsld:conditions>
</imsld:method>
</imsld:learning-design>
</organizations>

<resources>
<resource identifier="R-computer-introduction" type="imsldcontent" href="index.xml">
<file href="index.xml"/>
</resource>
<resource identifier="R-software-introduction" type="imsldcontent" href="software_intro.xml">
<file href=" software_intro.xml"/>
</resource>
<resource identifier="R-hardware-introduction" type="imsldcontent" href="hardware_intro.xml">
<file href=" hardware_intro.xml"/>
</resource>
<resource identifier="R-article" type="imsldcontent" href="article.xml">
<file href="article.xml"/>
</resource>
<resource identifier="R-Preparation" type="webcontent" href="preparation.html">
<file href="preparation.html"/>
</resource>
<resource identifier="R-Assignment-1" type="webcontent" href="assignment1.html">
<file href="assignment1.html"/>
</resource>
</resources>
</manifest>

```



After the processing of UoLParser, the CAM is shown as following figure:

Computer-Intro	Software-Intro	Hardware-Intro	Preparation	Task1
0	1	1	0	0
0	0	0	1	0
0	0	0	1	0
0	0	0	0	1
0	0	0	0	0

Figure 7.1: The result of UoLParser

7.2 Synthetic Computable Activity Matrix and Experiment

Results

We use synthetic computable matrix to evaluate the precision of our proposed Learning Activity Recommendation (LAR) scheme. All synthetic computable matrixes are generated by the following principles: 1) No continuous learning activity for testing or service learning activity. 2) Two learning activities have no connection if they are split from the same learning activity. 3) Only learning activity for testing or service learning activity can loop backward to another learning activity. 4) Each learning activity has to be passed. We design an authoring tool that generates the test data. The generating mechanism is based on combining the CAM of each template. There are three templates in the template pool: linear, condition, and loop. The name of each learning activity is generated randomly. The following figure illustrates the template flow and its corresponding CAM. The template can be extended as need.

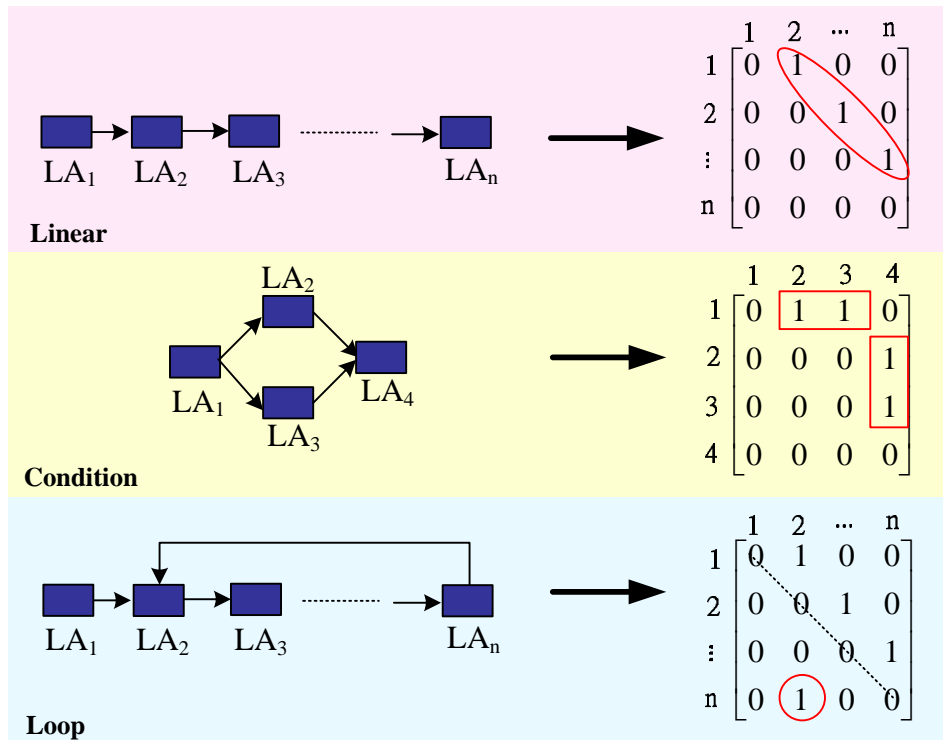


Figure 7.2: The templates of learning flow and its corresponding CAM

By combining these templates, we can generate the synthetic CAMs. The result is shown as follows:

```

系491個
[R, Q, G, L, K, C, S, B]

0 1 1 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 1 0 0 0
0 0 0 0 0 1 1 0
0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0

系492個
[G, Test, Service, V, Y, E]

0 1 0 0 0 0
1 0 1 0 0 0
0 0 0 1 0 0
0 0 0 0 1 0
0 0 0 0 0 1
0 0 0 0 0 0

系493個
[N, Service, Test, W, Z, Q, S, P, C, R, A, E, U, G, R, Service, Test, L, K, O, M, U]

```

Figure 7.3: The screenshot of the synthetic CAM

In LAR Scheme, the most important part is the similarity measuring mechanism. We'll use these synthetic computable matrixes that generated by the tool to verify the structure similarity functions that we analyze in the thesis. The experiment generates 500 synthetic computable matrixes randomly. In this experiment, we input 5 different queries. Then we will verify the result of each query to examine the functionality of LAR.

The following figure depicts the result of the experiment. The top of the figure represents the user's input parameters. After the similarity measurement, the results are listed in the order of scores from high to low. The CAM₁ gets the highest score, because it exactly matches the user's query.

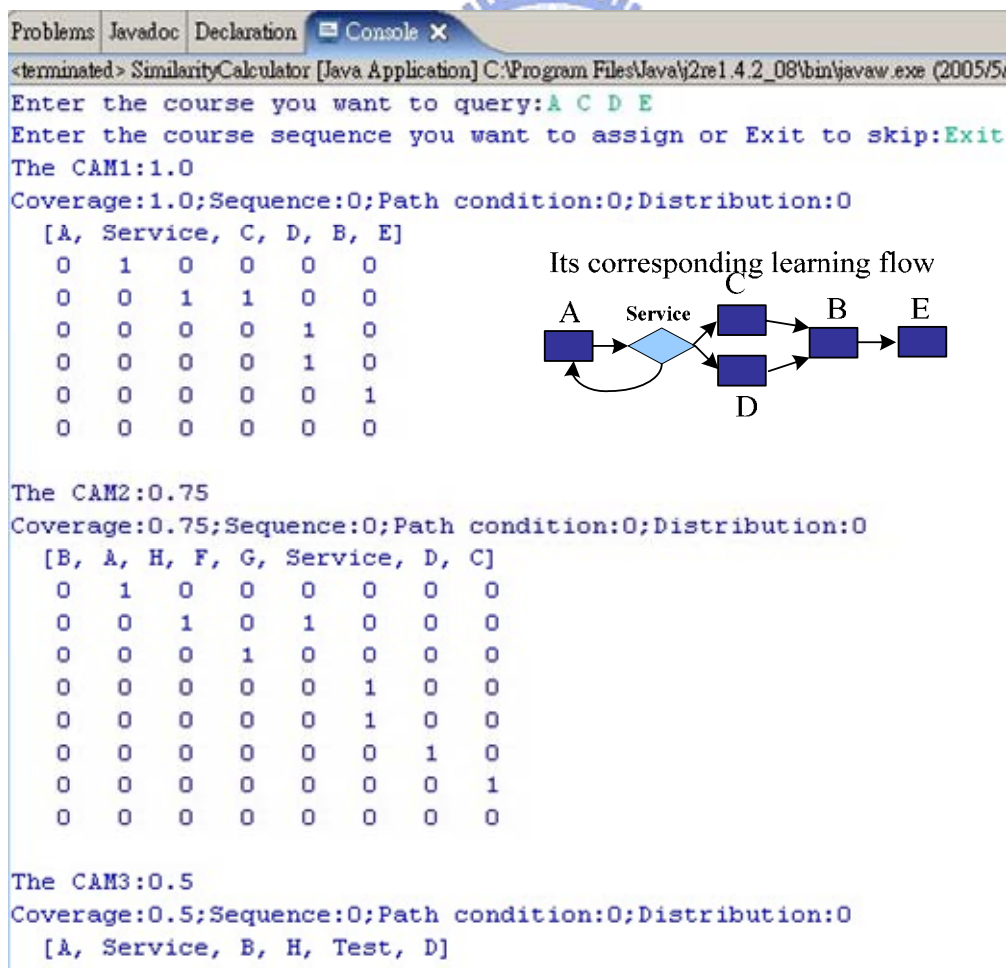


Figure 7.4: The screenshot of experiment result

The experiment result shows that our approach is workable and beneficial. By these four similarity functions, we can retrieve a desired learning activity efficiently. Another experiment is focused on the satisfaction of users. In the experimental scenario, we invite 10 persons that include the roles of teachers and students to use the simulator. We offer the testers two environmental scenarios: 1) Retrieve learning activities only by course names (GradeA). 2) Retrieve learning activities based on LAR approach (GradeB). The testers can score from points 0~10 for each scenario. The result shows in Figure 7.4.

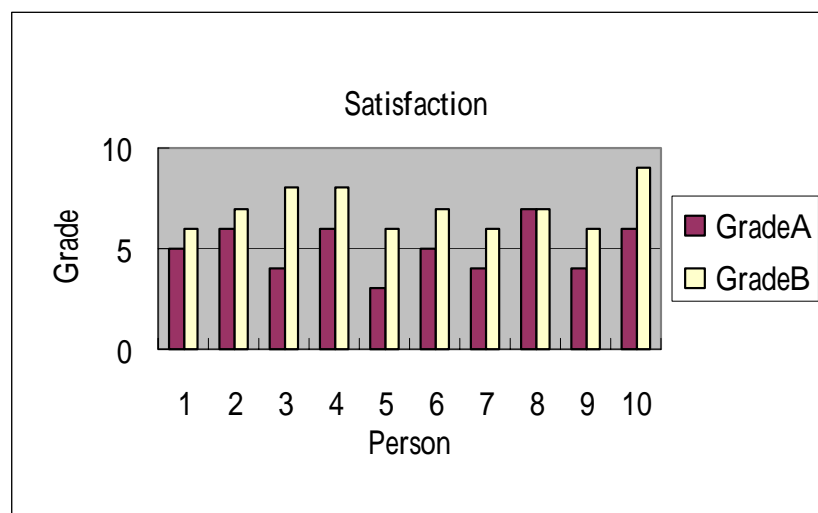


Figure 7.5: The result of satisfaction

7.3 System Interfaces

We use the editor of DRAMA to edit the rule in LAR which is shown in Figure 7.5. The rules' prototype is described in Section 4.3.

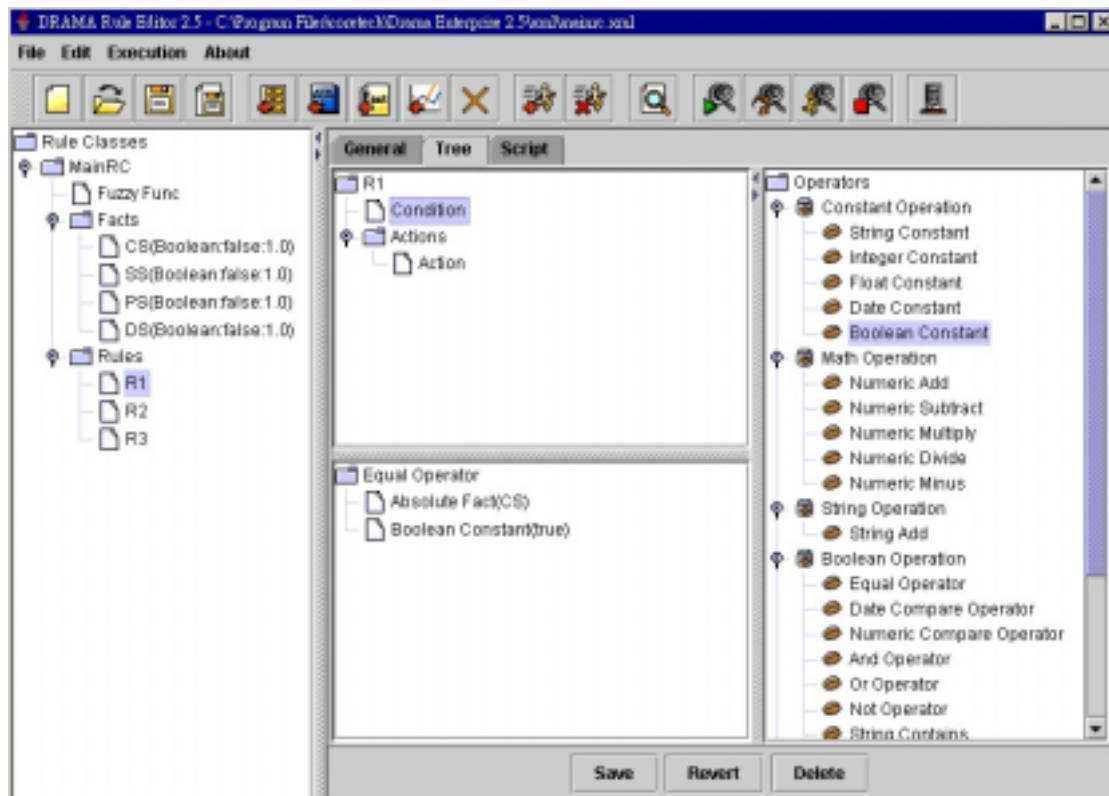


Figure 7.6: Editing rules of LAR with DRAMA Editor

The input scenario is controlled by *Rules Definition of Similarity Selection*. In the following figures, Figure 7.6 illustrates the first inferring process: when user inputs the desired courses, the rule1 will be triggered. Then the user is required to input the information of course sequence and the needness of course distribution. And in next inferring process, the user is asked to input the courses' path condition. After the user inputs their parameters, LAR will calculate the similarity functions and then display the results to the user. In Figure 7.7, the inference process and the result of LAR are shown.

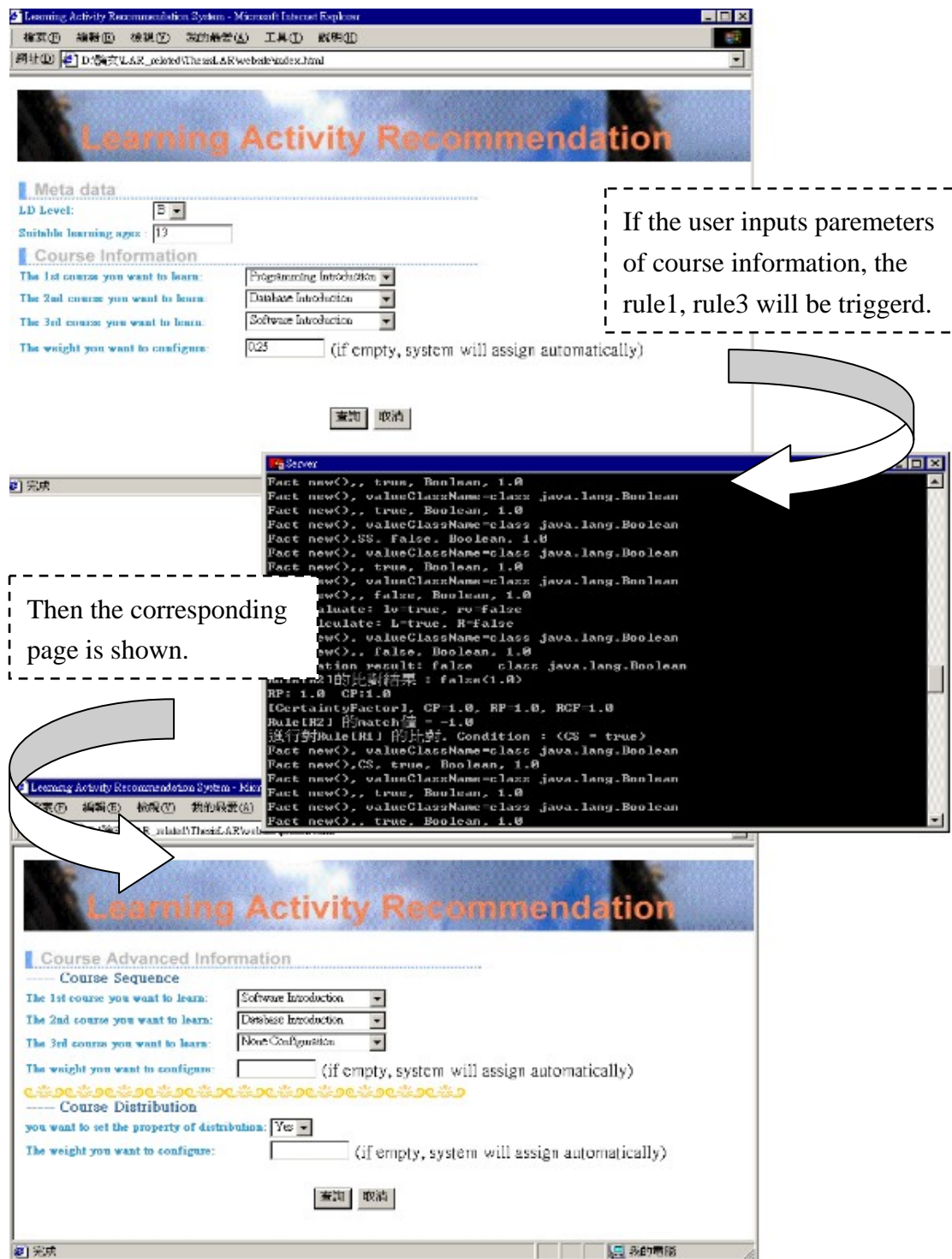


Figure 7.7: The first inferencing interactive process

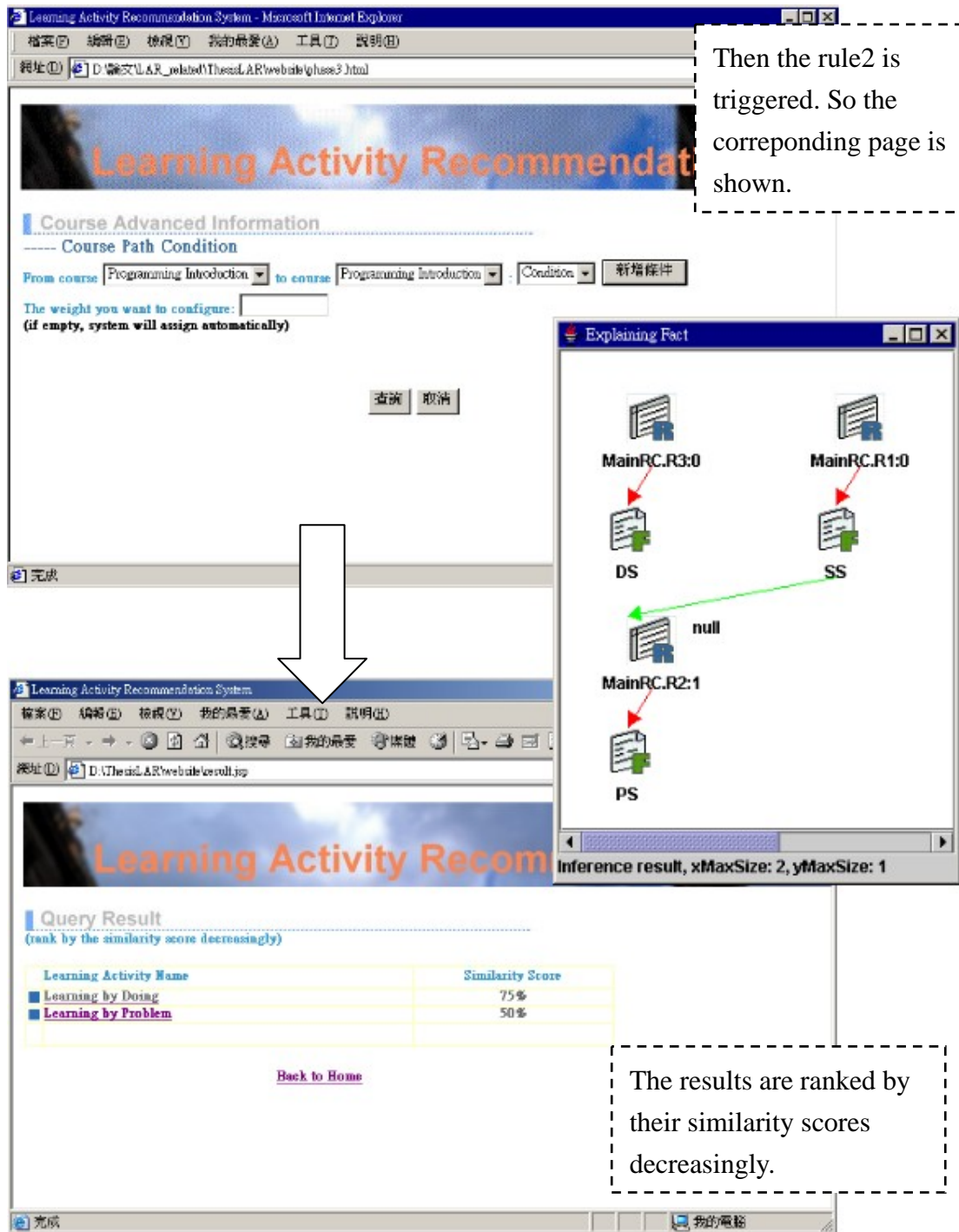


Figure 7.8: The inference process and the result of LAR

Then we use the project mentioned in Section 2.2.3, RELOAD, to edit the learning activity LAR recommended. After the user refines the learning activity they choose from the recommendation list, the user has to upload the learning activity content package to the CopperCore Server. And then the simulation of the learning

activity compliant with LD is done by CopperCore. The following figures show the interfaces.

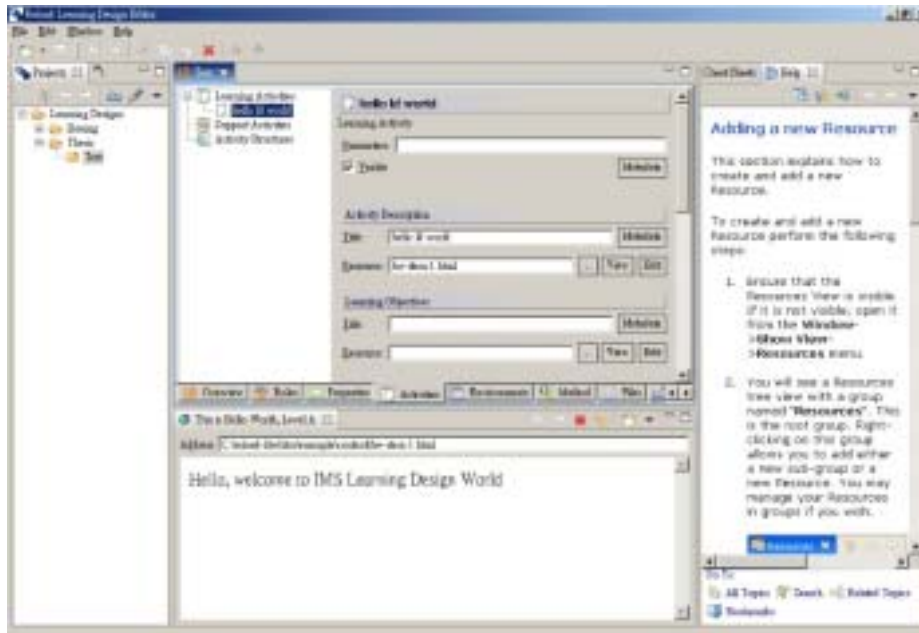


Figure 7.9: The editing process with RELOAD

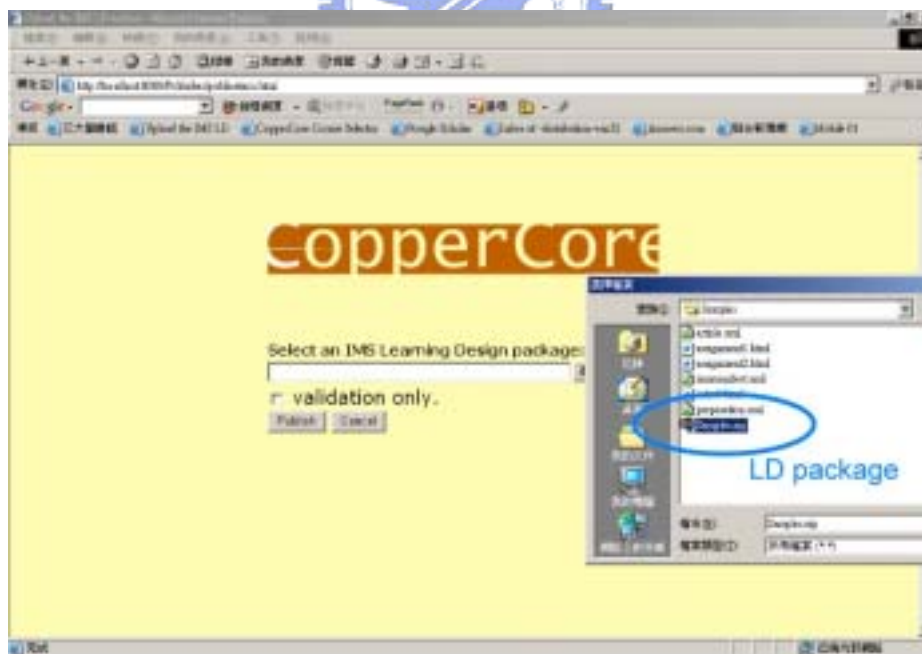


Figure 7.10: Upload page of CopperCore

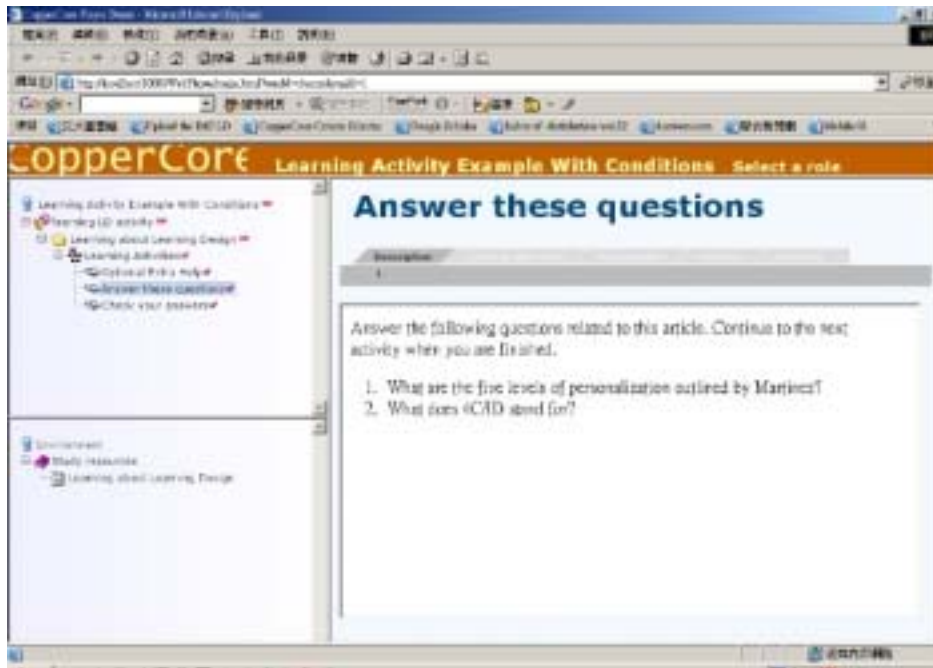


Figure 7.11: The Simulation on CopperCore




Chapter 8. Conclusion

Due to the growth of e-learning Standards, more and more institutes are developing the related tools, such as authoring tools and learning management systems. As many learning activity compliant with LD are generated, how to retrieve and how to reuse them will be critical issues. Thus, in this thesis, we propose a reusing learning activity mechanism, called **Learning Activity Recommendation (LAR) Scheme**, to achieve the purposes that search efficiently and reuse the LD compliant learning activities. According to the multiple similarity calculations, LAR can help users to find the learning activity they desired. In the framework, we also adopt the expert system shell, DRAMA, to handle the interactive input interface and the generation of the similarity equation. In summary, LAR is full of flexibility and scalability to simplify the process of editing learning activity compliant with LD. We also adopt the pedagogy theory, Bloom, to extend the framework of LAR. The experimental results are also shown to verify the similarity functionality.

In the near future, we will improve the data representation of **Computable Activity (CA)** for enhancing its scalability, such as to calculate the multi-role scenario of Learning Design. Then, the similarity functions will be extended to support more pedagogy theory, and we'll develop a LAR system. Besides, LAR can adopt the retain mechanism of CBR. Therefore, the LAR can recommend more adaptive learning activities to users.

References

- [1] Alliance for Remote Instructional and Authoring and Distribution Networks for Europe (ARIADNE) 2004, ARIADNE: Foundation for The European Knowledge Pool. <http://www.ariadne-eu.org>
- [2] AUTC (2002). AUTC conference: Reusable Learning Designs: opportunities and challenges. University of Technology, Sydney, December 2002. [Online] <http://www.iml.uts.edu.au/autc/> [30th July 2003]
- [3] Anderson, W., Sosniak, L. A. (Eds) (1994). "Bloom's taxonomy: A forty-year retrospective". Chicago, IL: The National Society for the Study of Education.
- [4] Anderson, W., Krathwohl, D. R. (Eds) (2001). "A taxonomy for learning, teaching, and assessing: A revision of Bloom's educational objectives". New York, NY: Longman.
- [5] BMC Remedy Service Management 
<http://www.remedy.com/solutions/services/education/lpc.htm>
- [6] CETIS (2003a). Pedagogy Forum. [Online] <http://www.cetis.ac.uk/members/pedagogy/> [30th July 2003]
- [7] CETIS (2003b). RELOAD: Reusable E-Learning Object Authoring and Delivery. [Online] <http://www.cetis.ac.uk/members/x4l/articles/reload> [30th July 2003]
- [8] CopperCore, <http://coppercore.org/>
- [9] Design, Standards, and Reusability, <http://www.downes.ca/cgi-bin/website/view.cgi?dbs=Article&key=1059622263>
- [10] Downes, S. (2003). Design, Standards and Reusability. [Online] <http://www.downes.ca/cgibin/website/view.cgi?dbs=Article&key=1059622263> [30th July 2003]
- [11] Griffiths, D. (2003). SCOPE: Structuring Content for Online Publishing

- Environments. [Online] <http://www.tecn.upf.es/scope/> [30th July 2003]
- [12] Harper, B. & Oliver, R. (2002). Reusable Learning Designs: information and communication technologies and their role in flexible learning. Presentation for the “AUTC Reusable Learning Designs: opportunities and challenges” Conference, UTS, Sydney, December 2002. [Online], <http://www.learningdesigns.uow.edu.au/Publications/AUTCICTProject.ppt> [30th July 2003]
- [13] LOM Standard, <http://ltsc.ieee.org/wg12/20020612-Final-LOM-Draft.html>
- [14] Laurillard, D. (2002). Design Tools for E-learning. Keynote presentation for ASCILITE2002.[Online], http://www.unitec.co.nz/ascilite/proceedings/papers/key_laurillard.pdf [30th July 2003]
- [15] Lin, Y.T., Tseng, S.S., Tsai, C.F. (2003), “Design and implementation of new object-oriented rule base management system”, Expert Systems with Applications, vol. 25, pp. 369-385, 2003.
- [16] Learning Design Issue: Learning Object, JISC. <http://www.jiscinfonet.ac.uk/InfoKits/effective-use-of-VLEs/designing-for-sustainability/designing-issues-learning-objects>
- [17] Learning Design and reuseability, Wilbert Kraan, CETIS staff (2003). <http://www.cetis.ac.uk/content/20030902133812>
- [18] IEEE Learning Technology Standards Committee (LTSC) 2004, IEEE LTSC | WG12, <http://ltsc.ieee.org/wg12/>
- [19] IMS (Instructional Management System), <http://www.imsproject.org/>
- [20] IMS Learning Design Information Model Version1.0 Final Specification
- [21] Jones, E.R. (2004), Dr. Ed’s SCORM Course, <http://www.scormcourse.jcasolutions.com/index.php>

- [22] Kraan, W. (2002). DfES' e-learning guru: Learning Design is the way ahead.
[Online] <http://www.cetis.ac.uk/content/20020930092048> [30th July 2003]
- [23] Koper, R. (2001). From change to renewal: Educational technology foundations of electronic environments. EML website. [Online]. Available:
<http://www.eml.ou.nl/> [30th April 2001]
- [24] Karampiperis, P. Sampson, D. (2004), "A Flexible Authoring Tool Supporting Learning Activities". In Proc. Of LADIS International Conference on cognition and Exploratory Learning in Digital Age.
- [25] Oliver, R., Harper, B., Hedberg, J., Wills, S., Agostinho, S. (2002) "Exploring strategies to formalise the description of learning designs". J. Herrington (Eds.) Proceedings of HERDSA. Joondalup: Edith Cowan University, 2002.
- [26] RELOAD, <http://www.reload.ac.uk/>
- [27] SCORM (Sharable Content Object Reference Model),
<http://www.aslnet.org/Scorm/scorm.cfm>
- [28] Sequencing and Navigation (SN) 2004, 'Sharable Content Object Reference Model (SCORM) Sequencing and Navigation (SN) Version 1.3', Advanced Distributed Learning.
<http://www.adlnet.org/index.cfm?fuseaction=DownFile&libid=648&bc=false>
- [29] Tattersall, C. (2003). EML and IMS Learning Design. Presentation for the Valkenburg Group, Vancouver, February 2003.
- [30] Taxonomy of educational objectives: The classification of educational goals, Handbook : Cognitive domain. New York, NY: Longman, Green.
- [31] Valkenburg Group (2003). Valkenburg Group. [Online],
<http://www.valkenburggroup.nl/> [30th July 2003]

Appendix A

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd=http://www.w3.org/2005/XMLSchema
  targetNamespace=http://e-learning.nctu.edu.tw
  xmlns="http://e-learning.nctu.edu.tw/XMLSchema"
  elementFormDefault="qualified">

  <xsd:element name="extension_LD">
    <xsd:complexType content="elementOnly">
      <xsd:sequence>
        <xsd:element name="learning_target_info" type="LTI"/>
        <xsd:element name="pedagogy_involved" type="Pedagogy">
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="LTI" content="elementOnly">
    <xsd:sequence>
      <xsd:element name="suitable-learning-target" type="string"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Pedagogy" content="elementOnly">
    <xsd:sequence>
      <xsd:element name="Bloom" type="BloomType"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="BloomType" >
    <xsd:sequence>
      <xsd:element name="learning activity" type="string"/>
      <xsd:element name="Taxonomy" type="string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

