國立交通大學

資訊科學系

碩 士 論 文

基 於 本 體 論 及 規 則 建 構 學 習 順 序

Learning Sequences Construction Using Ontology and Rules

研 究 生：陳瑞言

指導教授：曾憲雄　教授

中 華 民 國　九 十 四　年 六 月

基於本體論及規則建構學習順序

研究生：陳瑞言　　　　　　　　　　　　　　指導教授：曾憲雄 博士

國立交通大學資訊科學系

摘要

網域名稱伺服系統（Domain Name System，以下簡稱 DNS）是現今網際網路基礎設施的重要環節之一；然而，根據 Men & Mice (2005) 最新一份網路健康狀況調查的報告指出，目前仍有將近 70%的 .COM 網域存在設定上的錯誤。因此，如果能建構出一個可提供完整 DNS 相關知識的教學輔助系統，就可以幫助許多上述的系統管理者完成 DNS 系統的建構，減少錯誤設定的發生。爲了簡化學習順序建構的複雜度，在這篇論文中，我們提出一個使用本體論及規則建構學習順序的模型，其中包含有三個功能不同的模組：**基於本體論的學習順序建構模組**是用以將本體論轉換成一個基本的課程架構；**後設知識擷取模組**則是用於從規則中擷取出後設知識；最後，我們在**範例及測驗附加模組**中整合這兩類知識，以呈現給學習者更完整的課程架構。另一方面，分享內容元件參考模型（Shareable Content Object Reference Model，以下簡稱 SCORM）是目前最爲廣泛使用的數位學習標準，其藉由學習順序與瀏覽模組來定義使用者在數位學習上的學習行爲與順序；所以我們設計並實做了一個符合 SCORM 標準的 DNS 輔助教學雛型。實際上，只要做點些微的調整，就可將我們所提出的模型套用在其他領域上，幫助學習順序的建構。

關鍵字：學習順序、本體論、規則、網域名稱伺服系統、分享內容元件參考模型

Learning Sequences Construction Using Ontology and Rules

Student: Ruei-Yan Chen Advisor: Dr. Shian-Shyong Tseng

Department of Computer and Information Science
National Chiao Tung University

Abstract

The Domain Name System (DNS) is an essential part of the Internet software infrastructure. However, according to the domain health survey for commercial sites (Men & Mice, 2005), almost 70% of .COM zones have at least one mis-configuration. In practice, during design and deployment phases, DNS tutoring system could provide us with related information to help reduce the percentage of DNS mis-configuration. In this thesis, we propose a learning sequences construction model using ontology and rules to simplify the complexity of learning sequence construction. There are three modules in this model. First, the *Ontology-based Learning Sequences Construction Module* is designed to transform an ontology into a basic course scheme. Second, the *Meta-Knowledge Extraction Module* is used to extract meta-knowledge form rules. And, finally, these two kinds of knowledge would be integrated in the *Example & Quiz Annotation Module*. On the other hand, Shareable Content Object Reference Model (SCORM), which is the most popular e-learning standard, defines the learning sequence behavior and the learner navigation by using sequence and navigation model. For illustrating the ideas, we design and implement a SCORM-based DNS tutoring prototype system. In fact, with a few modifications, it is supposed that the model could easily be applied to other domains for learning sequences construction.

Keywords: learning sequence, ontology, rule, DNS, SCORM

誌謝

這篇論文的完成，必須感謝許多人的協助與支持。首先必須感謝我的指導教授，曾憲雄老師，由於他的耐心指導和勉勵，讓我得以順利完成此篇論文。此外，在老師的帶領下，這兩年來，除了學習應有的專業知識外，在待人處事方面也有不少的啓發，而研究上許多觀念的釐清更是讓我受益匪淺，真的十分感激。同時必須感謝我的口試委員，洪宗貝教授、楊錦潭教授及曾秋蓉副教授，他們給予了我許多寶貴的建議。

其次要感謝陳昌盛博士及劉建良學長，兩年期間讓我學會許多理論和實務技巧，也給予我許多對於此篇論文的寶貴意見，接受我的詢問和討論，並協助我完成論文的修改工作，深表感激。

另外也要感謝實驗室的學長、同學、以及 iDNS 計畫的學弟們，蘇俊銘學長、王慶堯學長、林順傑學長、翁瑞鋒學長、宋昱璋同學、張俊彥學弟、蔡昇翰學弟。不管是論文上或是系統的建置上，都給我許多協助及建議。同時也感謝所有實驗室同窗夥伴，君翰、昱璋、易虹、柏智、政霖、育松、成樑，我永遠不會忘記和你們一起度過的這段忙碌且充實的碩士生涯。

最後要感謝我的家人，謝謝你們在艱困的環境下，仍是毫不猶豫的支持我，讓我可以順利完成碩士學業，謝謝你們。

要感謝的人很多，無法一一詳述，在此僅向所有幫助過我的人，致上我最深的謝意。

Table of Content

List of Figures

List of Tables

List of Algorithms

Chapter 1. Introduction

The Domain Name System (DNS) is an essential part of the Internet software infrastructure. Unfortunately, due to the distributed nature of DNS and lack of efficient knowledge sharing mechanisms among DNS administrators, even though DNS is so important to network operation today, rather few DNS administrators have the expertise to do the jobs well. In the domain health survey for commercial sites (Men & Mice), almost 70% of .COM zones have at least one mis-configuration (i.e., some are minor ones, while others are fatal). In Chen et al. (2003), the authors proposed a framework for the design and implementation of a unifying intelligent system for DNS management. The iDNS system started to provide diagnosis services since 2003 and most of the feedbacks from users are positive.

On the other hand, as mentioned in Chen et al. (2003), many novice DNS administrators suffer from their lack of domain knowledge in conducting DNS management tasks. Therefore, in addition to DNS diagnosis system, DNS tutoring could provide DNS background knowledge for helping these people on the design and deployment of DNS servers to reduce the percentage of DNS mis-configuration.

Next, the reusability and interoperability issues of the teaching material are important as well. When the teaching material is reusable, other tutoring systems could reuse the teaching material directly. On considering these, we adopt the Sharable Content Object Reference Model (SCORM) model for building the web-based tutoring system. SCORM provides sequencing and navigation mechanism for learning objects. General speaking, the SCORM-based learning sequences should be domain-dependent; that is, it usually needs domain experts to get involved in developing the learning scheme. Therefore, the mechanism which could provide required assistance when the domain experts' constructing learning sequence is required.

In this thesis, we propose to adopt and integrate both the ***ontology knowledge model*** and ***rules knowledge model*** to construct SCORM-based learning sequences. An ontology is an explicit specification of a conceptualization (Gruber, 1993). Similar to concept map, an ontology consists of concept classes and relationships. This kind of architecture is suitable for the transformation to course scheme. Besides, in the view of knowledge representation, ontology has the following advantages:

(1) Ontologies are useful in a range of applications, where they provide a source of precisely defined terms that can be communicated across people and applications (Chandrasekaran et al., 1999). In short, ontology could help domain experts to model the domain knowledge.
(2) In essence, ontology representation is suitable for communication and natural for human thinking. The role of ontologies is to capture domain knowledge and provide a commonly agreed upon understanding of a domain. Hence, ontology can facilitate the communication between knowledge engineers and domain experts.
(3) Many useful ontology editors (e.g., Protégé, OilEd, etc.) could provide us with required assistance when constructing ontology.

As described above, ontology could model domain knowledge and could be used to reflect the sequence of the knowledge concept as well. Therefore, we adopt ontology as one of the knowledge resources to construct learning sequences.

On the other hand, one of the most popular approaches for knowledge representation is to use production rules. In general, rule representation is appropriate for the support of decision-making on network system management. Many network services (e.g., DNS, Intrusion Detection System, anti-SPAM software, etc.) adopt rules to perform their management jobs. Liu et al. (2004) proposed an ontology-driven model for rules extraction and applied the rules generated from the model on DNS diagnosis system. In essence, the rules in DNS diagnosis system focus on DNS problems. In other words, when users fire the rules in DNS diagnosis system, the diagnostic system would infer that there should be something wrong in users' DNS configurations and we could further infer that the users might need some background knowledge related to the problems they encounter. Therefore, with some appropriate design, we could extract some meta-knowledge from rules.

In general, our main contributions of the thesis are:

1. We propose a SCORM-based learning sequence construction algorithm, which is based on ontology and rules. The paradigm of integrating ontology and rules could benefit from domain knowledge model and users' behaviors.

2. We design and implement a DNS e-learning prototype system based on the SCORM-based learning sequences constructed from above algorithm.

The rest of this thesis is organized as follows. In Chapter 2, we introduce some preliminaries about knowledge representation, DNS domain knowledge and ontology, and SCORM. Chapter 3 shows the overall system architecture. There are three modules in our learning sequences construction system, which includes the ***Ontology-based Learning Sequences Construction Module***, the ***Meta-Knowledge***

Extraction Module, and the ***Example & Quiz Annotation Module***. The working details of these three modules are described in Chapter 4, Chapter 5, and Chapter 6 respectively. Chapter 7 shows the implementation of our system. Finally, we have concluding remarks about this thesis in Chapter 8.

Chapter 2. Preliminaries

In this Chapter, we will describe preliminaries and general information relating to this thesis. In Section 2.1, we first talk about knowledge representation, especially focusing on ontology representation and rule representation. Some basic DNS domain knowledge and the domain ontology would be covered in Section 2.2. Finally, we give a brief introduction of SCORM in Section 2.3.

2.1 Knowledge Representation

As we know, knowledge representation is one of the most central and familiar concepts in AI. Five distinct roles of knowledge representation are described in Davis et al. (1993). They are listed below:

- A knowledge representation (KR) is most fundamentally a surrogate.
- It is a set of ontological commitments.
- It is a fragmentary theory of intelligent reasoning.
- It is a medium for pragmatically efficient computation.
- It is a medium of human expression.

In our system, we adopt ontology representation and rules representation as the knowledge representation. From the above, we know that a knowledge representation is used as a substitution for the real world object. In principle, it is impossible for us to describe the real object completely because the one that could really denote the object is itself. In general, different knowledge representations focus on different views. Furthermore, different applications may need different representations on the same problem domain.

2.1.1 Ontology Knowledge Representation

An ontology is an explicit specification of a conceptualization (Gruber 1993). Ontologies are useful in a range of applications, where they provide a source of precisely defined terms that can be communicated across people and applications (Chandrasekaran et al. 1999). The role of ontologies is to capture domain knowledge and provide a commonly agreed upon understanding of a domain. Ontology defines the concepts, the attributes of the concepts, and the relationships among concepts. Figure 2.1 shows a simple animal ontology. There are five concepts and one kind of relationships in this simple ontology. With the help of ontology, the knowledge is not only human-readable but also machine-readable (Chandrasekaran et al. 1999; Gaines & Shaw 1993). Furthermore, the graphical representation of ontology could simplify the communication between the domain experts and knowledge engineers.

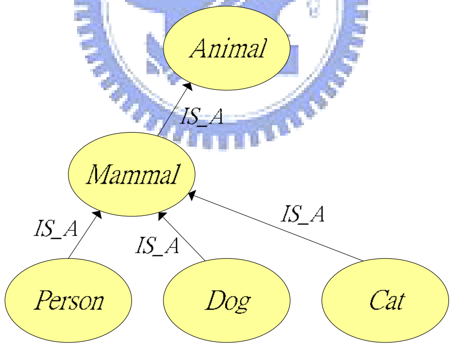


Figure 2.1: Animal ontology hierarchy

As mentioned in Fernandez (1999), the ontology building process is still a craft rather than an engineering activity. Each development team usually follows its own set of principles, design criteria and phases on the ontology development process. In Fernandez et al. (1997), the authors of METHONTOLOGY explain that the life of an ontology moves on through the following states: specification, conceptualization, formalization, integration, implementation, and maintenance. Knowledge acquisition, documentation and evaluation are supporting activities that are carried out during the majority of these states. The evolving prototype life cycle of METHONTOLOGY allows the ontologist to go back from any state to other if some definition is missed or wrong. So, this life cycle permits the inclusion, removal or modification of definitions anytime of the ontology life cycle.

2.1.2 Rule-based Knowledge Representation

One of the most popular approaches to knowledge representation is to use production rules, sometimes called IF-THEN rules. The basic form of the rule representation is:

IF <Condition> THEN <Conclusion>;

When the input facts match the condition (or premise), the inference engine would infer that a certain rule (or a set of rules) should be fired and the action part would be activated.

There are many advantages of using the IF-THEN rule representation. First, the IF-THEN form is similar to natural language and it is easily understood since each defines a relatively small and, at least in principle, independent piece of knowledge. Second, the IF-THEN rules are powerful to define the management mechanism for many application domains. In fact, many network services (e.g., Intrusion Detection System (IDS), anti-SPAM software, information filtering system, etc.) adopt rules as the engine to perform access control jobs. For example, most firewall software systems are typical rule-based system. In practice, the network administrator could define the filtering rules to filter out unwanted network packets or protocols. For example, to the network attacks, it is a common practice for many network administrators to allow web access and there might be some pseudo rule like the following:

IF the port of destination server <> 80 THEN reject the packet;

As described above, rule representation is also suitable for DNS domain as well. In principle, in a typical DNS diagnosis system, we might diagnose DNS problems from real-world DNS configurations. Using our DNS diagnosis system, the provided DNS configuration information of a site could be viewed as the facts and the whole diagnosis process is a typical forward reasoning process. For example, we could define the rule about Single Point of Failure (SPOF) as follows:

IF number of NS records < 2 THEN SPOF=true

2.1.3 Hybrid Knowledge Model

Knowledge acquisition is often the bottleneck of building Knowledge-Based System (KBS). Usually, it is not easy to extract knowledge directly from domain experts. Therefore, some mechanism (e.g., automatic, semi-automatic, or even manual) is required during the knowledge acquisition process. In essence, ontology representation is easily understood by domain experts and knowledge engineers. The concept hierarchy, concept attributes and relationships are similar to the object-oriented design or database schema design. In addition, many existing ontology tools (such as Protégé) can simplify ontology construction. So ontology representation is suitable for knowledge engineers and domain experts to model the domain knowledge.

On the other hand, rules representation is more suitable for many practical domains (e.g., DNS management, firewall management, etc.) since rule representation is powerful for machines to manipulate the concepts (or tasks). However the rule extraction is not a straightforward process. In Liu et al. (2004), we proposed an ontology-driven model for rule extraction. The whole process is to facilitate the domain experts to extract the rules by the help of ontology. The ontology could guide the rules extraction and simplify the whole process.

In this thesis, we adopt both ontology and rule knowledge representations to model the resource knowledge. The advantages of using the hybrid knowledge model are as follows:

- Ontology representation could make domain problem modeling more easily.
- Ontology could facilitate the KBS rules extraction.
- Rule representation is powerful for machine to manipulate the concepts.
- DNS diagnosis could be addressed by rules to model users' behaviors.

2.2 DNS Domain Knowledge and Ontology

2.2.1 Basics of the DNS System

The Domain Name System (Mockapetris 1987-1; Mockapetris 1987-2) is responsible for translating between hostnames and the corresponding IP addresses needed by software. The mapping of data is stored in a tree-structured distributed database where each name server is authoritative (responsible) for a portion of the naming hierarchy tree. The client side query process typically starts with an application program on the end user's workstation, which contacts a local name server via a resolver library. That client side name server queries the root servers for the name in question and gets back a referral to a name server who should know the answer. The client's name server will recursively follow referrals re-asking the query until it gets an answer or is told there is none. Caching of that answer should happen at all name servers except those at the root or top-level domains (.com for example). The working paradigm could be illustrated in Figure 2.2.

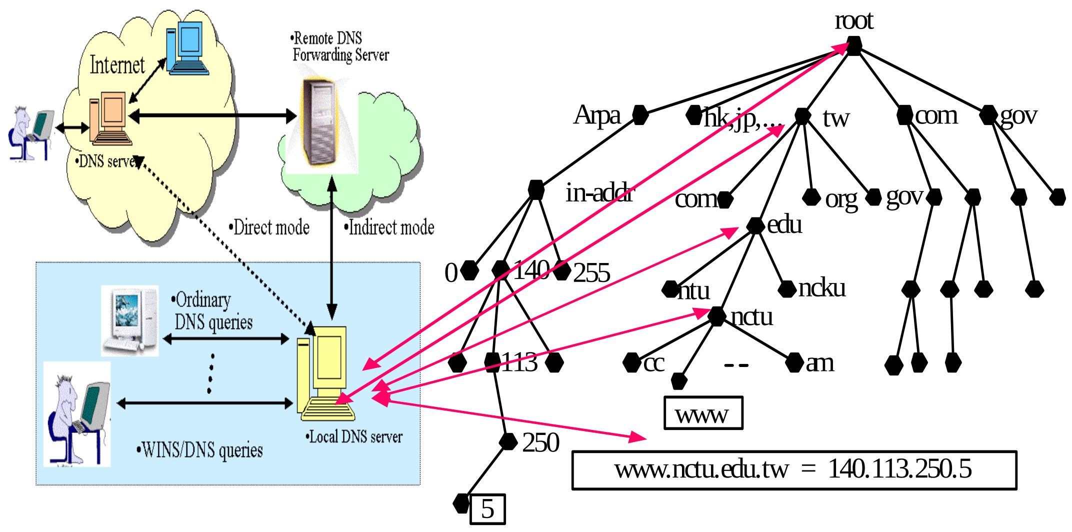


Figure 2.2: DNS operation model

2.2.2 DNS ontology

For many people (e.g., DNS beginners, etc.), information of DNS taxonomy will help them understand operating details of the DNS and describe encountered problems more explicitly. Figure 2.3 shows a snapshot of DNS ontology (Chen et al. 2002). Five types of relationships and four types of constraints are described as follows:

■ **Five types of relationships:**

(1) ***Component of:*** "*Component of*" is a generalization relationship, which could be used to describe the concept taxonomies in the class hierarchy. For example, NS Record is a *component of* Zone Data.

(2) ***Type of:*** "*Type of*" is also a generalization relationship. For example, a Master DNS Server is a *type of* Authoritative DNS Server.

(3) ***Attribute of:*** Denote the theoretic attributes of a concept class. For example, DNS Availability is an *attribute of* DNS Server on theoretical aspect.

(4) ***Process of:*** Denote the actual processing of a concept class. For example, DNS Registration is a *process of* DNS administration while users try to construct a working domain (zone) name.

(5) ***Synonym:*** "*Synonym*" relationship could be used to denote two or more terms with the same meaning. For example, a master DNS server of a specific zone is also called a primary DNS server. Both terms are synonyms.

■ **Identification of Constraints:**

(1) ***Pre-requisite constraint:*** One class depends upon another. For example: if we want to avoid SPOF, we must first adopt policies and enforcement mechanisms to avoid being Single Network and Single Server in advance.

(2) ***Temporal constraint:*** One class must occur before another. For example: DNS Registration is needed as long as Delegated DNS Server exists.

(3) ***Mutually inclusive constraint:*** One class requires another for its existence. For example, to improve the availabilities of DNS zones, we have to avoid SPOF (i.e., having replicate servers for servicing the specified zone) in the first place.

(4) ***Mutually exclusive constraint:*** One term/relationship must not co-exit with another. For example, a caching-only DNS server is not an authoritative-only DNS server and vice versa.

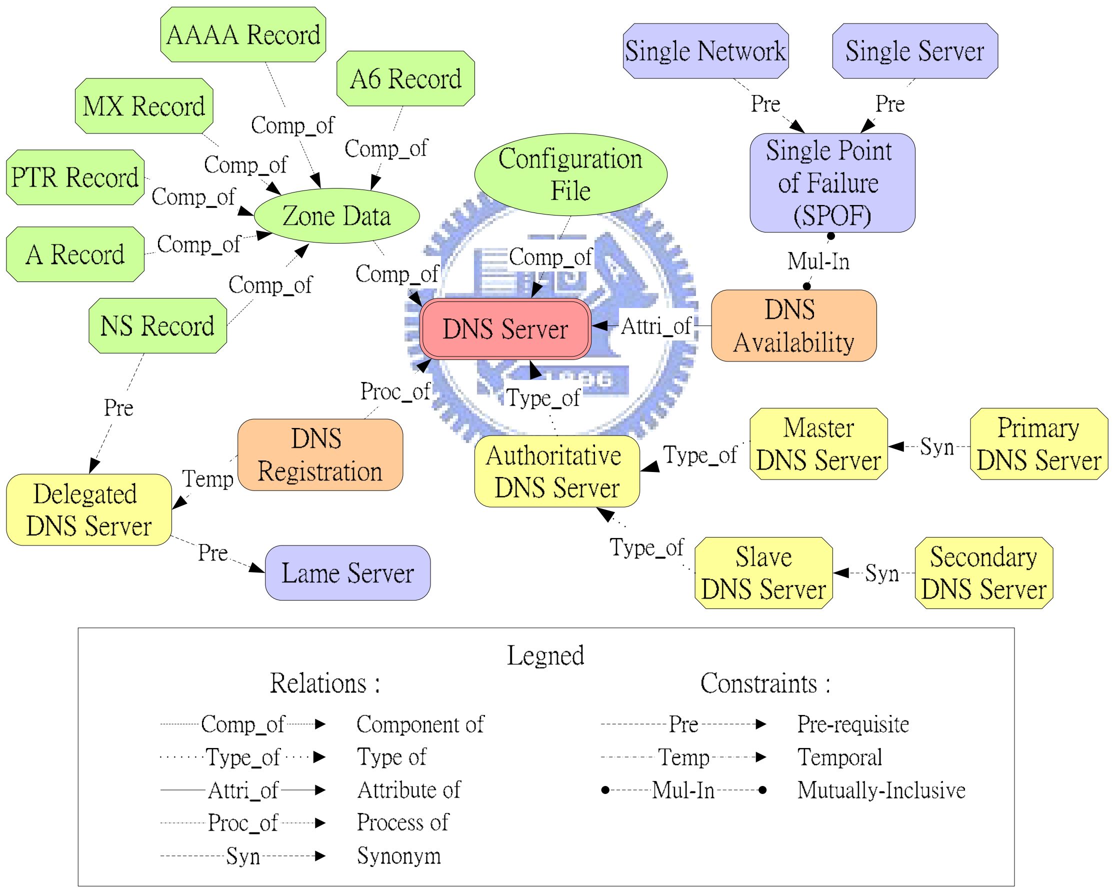
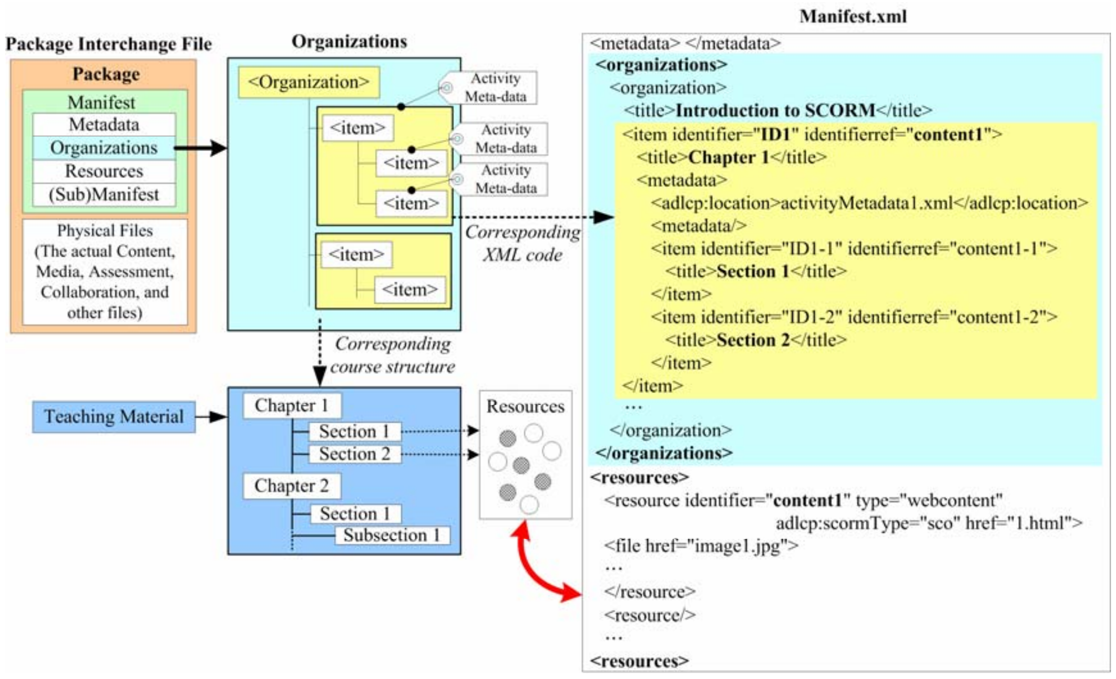


Figure 2.3: A diagram showing part of the DNS ontology

2.3 SCORM (Sharable Content Object Reference Model)

SCORM, which is proposed by the U.S. Department of Defense's Advanced Distributed Learning (ADL) organization in 1997, is currently the most popular one among those existing standards for learning contents. The SCORM specifications are a composite of several specifications developed by international standards organizations, including the IEEE, IMS, AICC and ARIADNE. 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 "RAID:" Reusable: easily modified and used by different development tools, Accessible: can be searched and made available as needed by both learners and content developers, Interoperable: operates across a wide variety of hardware, operating systems and web browsers, and Durable: does not require significant modifications with new versions of system software (Jones 2004).

In SCORM, content packaging scheme is proposed to package the learning objects into standard teaching materials, as shown in Figure 2.4. The content packaging scheme defines a package of teaching materials consisting of four parts - 1) Metadata: describe the characteristic or attribute of this learning content, 2) Organizations: describe the structure of this teaching material, 3) Resources: denote the physical file linked by each learning object within the teaching material, and 4) (Sub) Manifest: describe this teaching material is consisted of itself and another teaching material. In Figure 2.4, the organizations define the structure of whole teaching material, which consists of many organizations containing arbitrary number of tags, called item, to denote the corresponding chapter, section, or subsection within physical teaching material. Each item as a learning activity can be also tagged with activity metadata which can be used to easily reuse and discover within a content repository or similar system and to provide descriptive information about the activity. Hence, based upon the concept of learning object and SCORM content packaging scheme, the teaching materials can be constructed dynamically by organizing the learning objects according to the learning strategies, students' learning aptitudes, and the evaluation results. Thus, the individualized teaching materials can be offered to each student for learning, and then the teaching material can be reused, shared, recombined.



```
<metadata> </metadata>
<organizations>
  <organization>
    <title>Introduction to SCORM</title>
    <item identifier="ID1" identifierref="content1">
      <title>Chapter 1</title>
      <metadata>
        <adlcp:location>activityMetadata1.xml</adlcp:location>
        <metadata/>
        <item identifier="ID1-1" identifierref="content1-1">
          <title>Section 1</title>
        </item>
        <item identifier="ID1-2" identifierref="content1-2">
          <title>Section 2</title>
        </item>
    </item>
    …
  </organization>
</organizations>
<resources>
  <resource identifier="content1" type="webcontent"
                              adlcp:scormType="sco" href="1.html">
  <file href="image1.jpg">
  …
  </resource>
  <resource/>
  …
<resources>
```

Figure 2.4: SCORM content packaging scope and corresponding structure of teaching materials

Chapter 3. System Architecture

Figure 3.1 shows overall system architecture. There are three modules in our learning sequences construction system. The ***Ontology-based Learning Sequences Construction Module*** and the ***Meta-Knowledge Extraction Module*** are preprocessing modules. And, the ***Example & Quiz Annotation Module*** would be used to integrate the preprocessing results into a recommended course scheme. The working principles of these modules will be covered in more details later in this chapter.

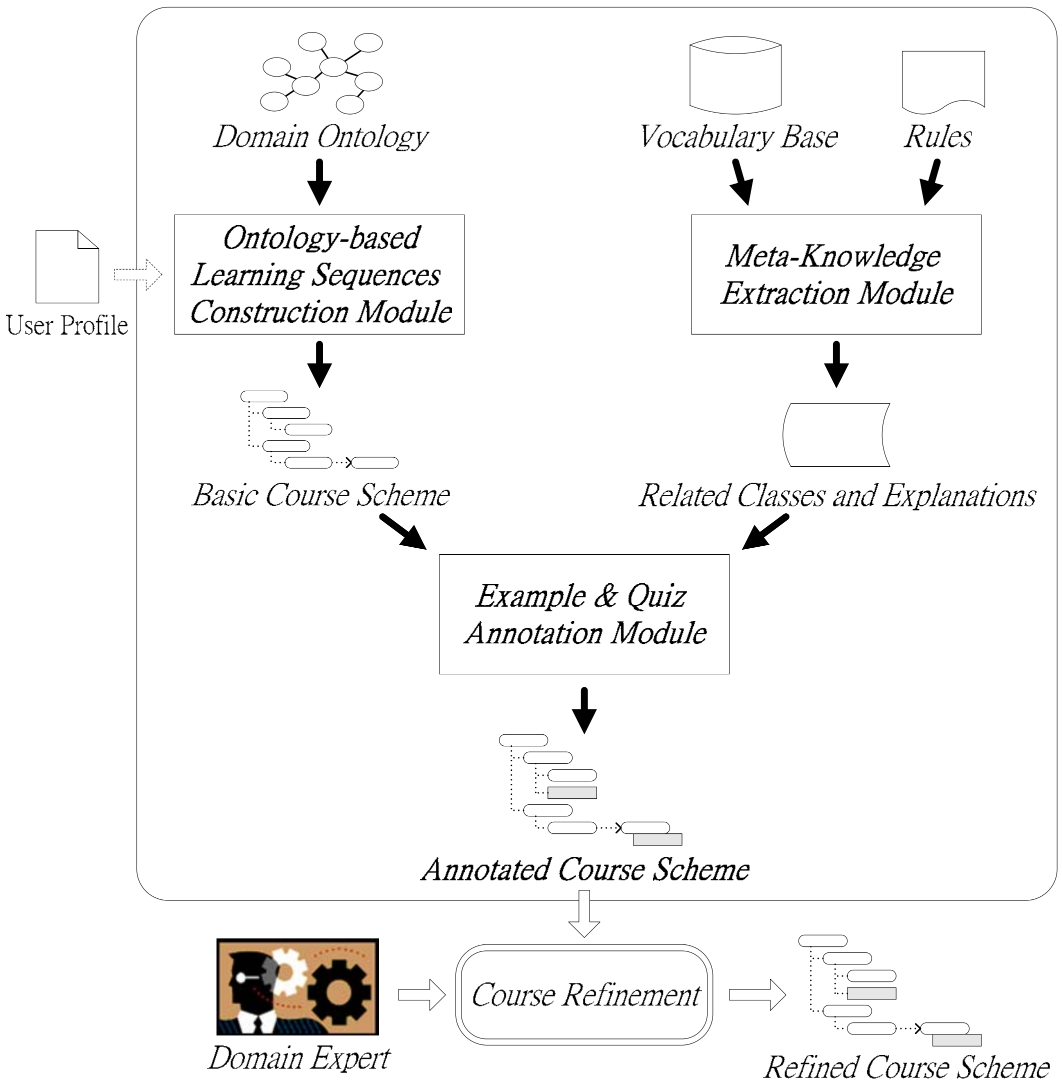

Domain Ontology
Vocabulary Base
Rules
User Profile
Ontology-based Learning Sequences Construction Module
Meta-Knowledge Extraction Module
Basic Course Scheme
Related Classes and Explanations
Example & Quiz Annotation Module
Annotated Course Scheme
Domain Expert
Course Refinement
Refined Course Scheme

Figure 3.1: Overall system architecture

3.1 Learning Sequences Construction Preprocessing

In this thesis, we adopt both domain ontology and domain knowledge rules to construct learning sequences. In essence, ontology representation focuses on concept classes, attributes of the concept classes, and the relationships between the concept classes. On the other hand, the main components of rule representation are facts and actions. Thus, in practice, the integration of ontology and rules is not a straightforward job. Before integrating these two kinds of knowledge representations, it is supposed that there should be preprocessing processes to make the integration process smoother.

In essence, as mention in Section 2.1.1, we know that an ontology is an explicit specification of a conceptualization, and it could be used to model domain knowledge. In addition, an ontology consists of concept classes and relations. Hence, we would like to transform the concept hierarchy and relationships between the concepts into some kind of learning sequence. Therefore, the approach to transform ontology into a course scheme (a set of learning sequences) is suitable. In this thesis, we propose an *Ontology-based Learning Sequences Construction Module* to transform the domain ontology into a basic course scheme. In essence, there are three main issues:

(1) How to decide the transformation mapping between relationships of the domain ontology and learning sequences?

(2) When we confirm the transformations between relationships and learning sequences, the next problem is: how to decide the priority among these transformations?

(3) How to create a course scheme based on the definition of transformations and the priority ordering definition?

The working details in the Ontology-based Learning Sequences Construction Module will be described later in Chapter 4.

In general, rule representation is appropriate for the support of decision-making on network system management. In fact, many network services (e.g., IDS, anti-SPAM software, information filtering systems, etc.) adopt rules as the engine to perform access control jobs. For example, most firewall software systems are typical rule-based systems. As similar to other network management domain, rule representation is suitable for DNS domain as well. In these domains, adding rule-based knowledge is a very efficient way to enhance the content of course.

Usually, rule format was written as follows:

IF <Condition> THEN <Conclusion>;

In order to integrate rule-based knowledge into the course exactly and smoothly, we should get some information from this kind of representation, including:

- What are the key terms of the rule?
- Where is the most appropriate position of the domain course to add the rule-based knowledge?

Therefore, we propose to design the *Meta-Knowledge Extraction Module* to get the necessary information. In essence, this module has two intentions:

(1) Extract meta-knowledge from rules.

(2) Let the meta-knowledge be integrated with ontology knowledge smoothly.

The working details of this module will be covered later in Chapter 5.

3.2 Knowledge Integration

After the preprocessing of ontology knowledge and rule-based knowledge, we could get a basic course scheme and meta-knowledge of rules. Next, in order to offer learners more complete domain knowledge, we will integrate these two kinds of knowledge by adding the meta-knowledge into the basic course scheme. However, we would meet a problem: What form would we add the rule-based knowledge into the ontology knowledge with?

According to the pedagogy theory, we know that learning by examples or quizzes could increase the learning efficiency. In essence, the rules in DNS diagnosis are used to diagnose DNS problems and it is suitable to provide examples when DNS problems occurred. Furthermore, the facts and actions of DNS diagnosis rules are composed by DNS ontology elements. Thus, the rules could provide us the hits to provide appropriate examples at suitable place in the course scheme. In this thesis, we propose the *Example & Quiz Annotation Module* to annotate the rule-based knowledge (i.e., the meta-knowledge extracted from rules) into the ontology knowledge (i.e., the basic course scheme transformed from the domain ontology) with the form of examples and quizzes.

In practice, examples would usually be presented after learners have finished most of the related chapters (or sections). If we want to annotate the meta-knowledge of a rule as an example of the course, we have to know which chapter (or section) is the last one among those related chapters (or sections). Here we will use the learning sequences transformed by the Ontology-based Learning Sequences Construction Module to decide which one is the last studied related chapter (or section) in a rule.

In this thesis, we adopt the methodology proposed by Fischer (2001) to perform the quiz annotation. The main idea is to change the major conceptual terms with the same (or similar) relationships in the ontology to generate some simple, but meaningful quizzes. The quizzes include true-or-false, single choice, and even multiple choices. Both annotation methodologies will be described in Chapter 6.

3.3 Course Refinement

The course scheme generated by domain ontology and rules is a recommended one. In practice, teachers or domain experts could use some authoring tools to refine the course scheme for their own uses (i.e., to fit their own requirements). Because the course scheme follows the standard of SCORM 2004, the most popular standard for learning contents, there are many authoring tools to use. For example, Reusable eLearning Object Authoring & Delivery (RELOAD) provides a *Metadata and Content Packaging Editor*, which can help users organize, aggregate and package learning objects in standard IMS and SCORM content packages. Furthermore, in Su et al. (2005), the authors proposed an Object Oriented Course Modeling (OOCM) to construct the SCORM compliant course and supported a graphic OOCM authoring tool as shown in Figure 3.2.

Moreover, based on our proposed architecture, system administrators can add additional information to make the course scheme more adaptive. For example, system administrators may add user profiles into the *Ontology-based Learning Sequences Construction Module* to influence the definitions of transformations and the priority ordering. In this way, the system could be used to generate more individualized course schemes.

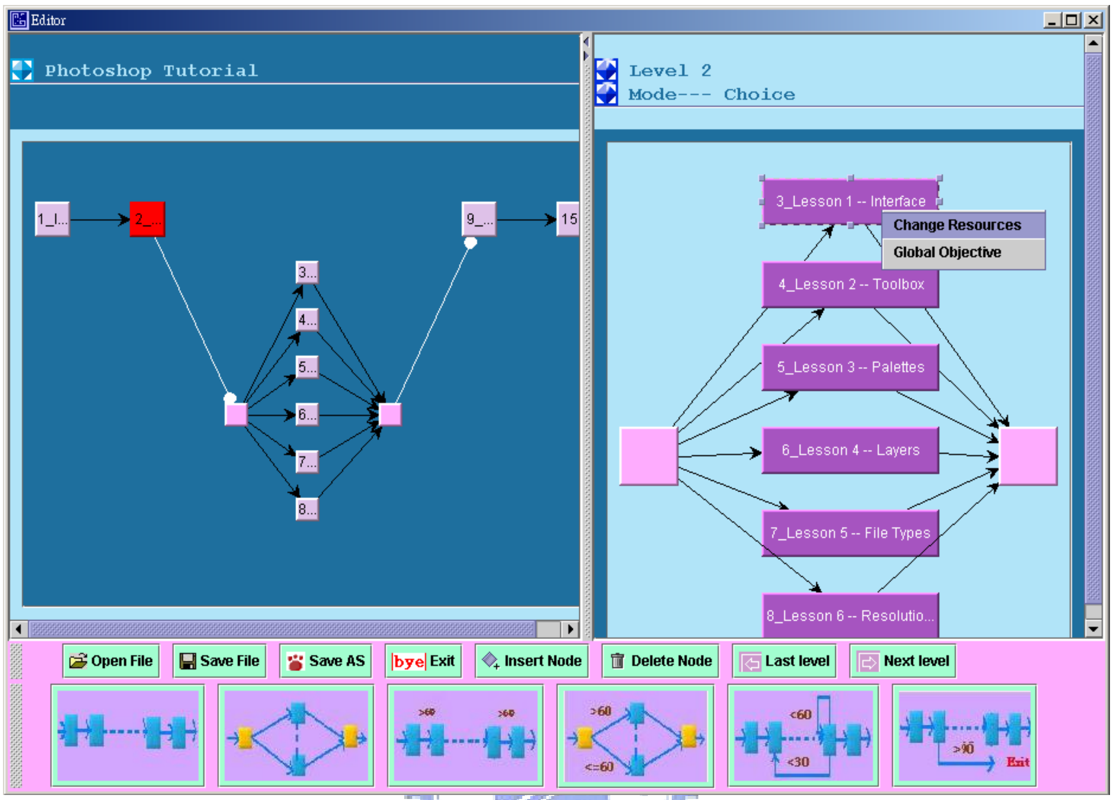


Figure 3.2: The screenshot of the OOCM authoring tool

Chapter 4. Ontology-based Learning Sequences Construction

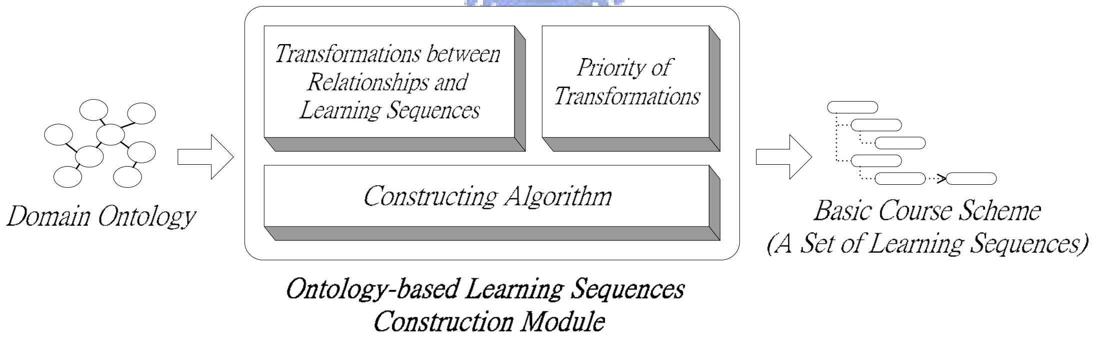
In this chapter, we will introduce the detailed processing of the ***Ontology-based Learning Sequences Construction Module***. As shown in Figure 4.1, we will transform a domain ontology into a basic course scheme (a set of learning sequences) in this module. There are three primary parts in this construction module:

(1) Transformations between domain ontology relationships and learning sequences.

(2) Priority ordering definition among those transformations.

(3) The ontology-based learning sequences constructing algorithm we proposed.

We will discuss these three parts in more detail and explain the whole process using an example with a simple DNS ontology we mentioned in Section 2.2.

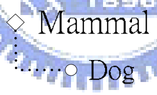


Transformations between Relationships and Learning Sequences
Priority of Transformations
Constructing Algorithm
Domain Ontology
Basic Course Scheme (A Set of Learning Sequences)
Ontology-based Learning Sequences Construction Module

Figure 4.1: Ontology-based learning sequences construction

4.1 Transformation between Ontology Relationship and Learning Sequence

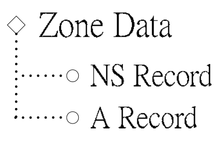
In essence, learning sequence construction is one of the important issues on building SCORM-based tutoring system. Meanwhile, many relationships (e.g., Is_a, or to be more precisely – Subset_of, Component_of, Type_of, etc.) between the ontology concept classes could represent the hierarchical information of the domain knowledge and be used to suggest appropriate learning sequences of the knowledge. Therefore, with the help of transformation process, we could transform the ontology relationship into some kind of learning sequence according to the properties of the relationship. For example, the general relationship *"Is a"* implies hierarchical information and there exists an *"Is a"* relationship exists between the concept class "Dog" and the concept class "Mammal", so we would transform the relationship into parent-child (or chapter-section) structure:



In different domain ontology, there are different kinds of relationships. The problems of polysemy and synonymity may happen among ontologies. Therefore, to deal with these kinds of dilemma, if we want to define the transformations between the domain ontology relationships and learning sequences, it is nature for us to interview the domain experts to confirm the meanings of relationships, and establish the transformations furthermore. In DNS domain, according to the properties of those relationships of the DNS ontology, we define the translations as follows:

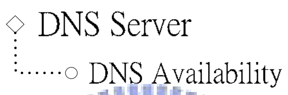
(1) ***Component of:*** If concept class ***A*** is a component of concept class ***B***, it means that ***A*** is at the child level (or sub level) of ***B*** in course construction. For example, NS

Record and A Record are components of Zone Data of a specified zone, so we can construct this kind of learning sequence:



(2) ***Type of:*** Similar to "Component of", this relationship can also be translated into the learning sequence of parent-child structure.

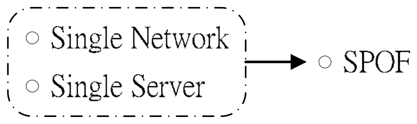
(3) ***Attribute of:*** Just as the literal meaning, we would transform this relationship into the parent-child structure. For example, DNS Availability is an attribute of DNS Server, so we construct learning sequence as follows:



(4) ***Process of:*** If concept class ***A*** indicates a required process of concept class ***B***, it suggests that when we want to learn ***B***, ***A*** is one of the parts we should learn.

(5) ***Synonym:*** Suppose that there exists a synonym relationship between concept class ***A*** and concept class ***B***, it means that they share the same tutoring content. Hence, no new learning sequence will be generated, and we will indicate that ***A*** and ***B*** are synonyms in the tutoring content.

(6) ***Pre-requisite constraint:*** If concept class ***A*** is a pre-requisite to concept class ***B***, then ***A*** should be introduced before ***B*** in the domain course scheme. For example, both the "Single Server" and "Single Network" concepts have Pre-requisite constraints to SPOF avoidance, so we can construct the following learning sequence:



(7) ***Temporal constraint:*** If concept class ***A*** has a temporal constraint to concept class ***B***, it means that concepts ***A*** and ***B*** are highly coupled. Hence, it suggests a natural order that ***A*** might be better to be introduced before ***B*** (although this is not necessary) in the domain course scheme.

(8) ***Mutually inclusive constraint:*** If concept class ***A*** and concept class ***B*** are mutually inclusive and if ***A*** is mentioned earlier in a course scheme transformation, then ***B*** might be at the child level of ***A*** in the generated course construction.

4.2 Priority of Transformations

Usually, there might be more than one kind of relationships in an ontology and each relationship is supposed to correspond to a kind of transformation. Therefore, if we would like to transform an ontology into a basic course scheme, we have to decide the priority ordering definition among these transformations since there are so many co-existed relationships. Just like the definition of transformations, the priority ordering definition is also domain-dependent. Thus, it is supposed that the priority ordering be defined with the help of domain experts.

In DNS domain, we might have some heuristic knowledge about the priority ordering after interviewing the DNS domain experts:

- Generally speaking, there are two basic kind of learning sequences of our generate. One is the parent-child structure (*Comp_of*, *Type_of*, *Attri_of*, *Proc_of*, and *Mul-In*), the other is the ordering structure (*Pre* and *Temp*).

- Some typical priority ordering suggestion from heuristic knowledge:
  (1) The pre-requisite knowledge of ***A*** should be introduced before one learns about the concept class ***A***.
  (2) When studying concept class ***A***, all the child level knowledge of ***A*** should be introduced as well.
  (3) If ***A*** has temporal constraints to other concept classes, then it is supposed that we learn those concept classes later.

- Among all the child level knowledge of concept class ***A***, a suggestive learning order might be:
  1. The components of ***A***
  2. Some types of ***A***
  3. The theoretical attributes of ***A***
  4. Some actual processing of ***A***
  5. The concept classes which are mutually-inclusive with ***A***.

Therefore, according to above principles, we could generate the priority of transformations in the partial DNS ontology as listed below (i.e., denoted by the ontology relationship of the transformation):

Pre-requisite > *Component of* > *Type of* > *Attribute of* > *Process of* > *Mutually-Inclusive* > *Temporal*

4.3 Constructing Algorithm

In this Section, we propose an *ontology-based learning sequences construction algorithm*. As mentioned in Section 4.1 and Section 4.2, before running this algorithm, the transformations between an ontology and the learning sequences and the priority ordering of those transformations must be well-defined. The constructing algorithm is shown below:

Algorithm 4.1: The construction algorithm

Input: The domain ontology

Output: The basic course scheme

Step 1: Locate the *core concept class* and take it as the *now-class*.

Step 2: Find all ***available*** *relationships* and *associated concept classes* of the *now-class*.

- **Step 2.1:** Find all *relationships* and *associated concept classes* of the *now-class*.
- **Step 2.2:** If the *relationship* would not generate any new learning sequence, we would eliminate the *relationship* and its *associated concept class*.
- **Step 2.3:** If the *now-class* has either a *"Pre-requisite"* or a *"Temporal"* constraint to the *associated concept class*:
  - **Step 2.3.1:** If the *associated concept class* has been used in another transformation or has other *relationships*, then we would eliminate this *relationship* and the *associated concept class*.
  - **Step 2.3.2:** If the *associated concept class* has no other *relationships*, then we would convert the *relationship* into *"Temporal"* for construction.

Step 2.4: If the *associated concept class* has a *“Pre-requisite”* or *“Temporal”* constraint to the *now-class*:

Step 2.4.1: If the *associated concept class* has been used in other transformation, then at the level the *associated concept class* stands, we would move the *associated concept class* before the *now-class*. Finally, eliminate this *relationship* and the *associated concept class*.

Step 2.4.2: If the *associated concept class* has not been used in other transformation, then convert the *relationship* into *“Pre-requisite”* for construction.

Step 3: Sort the *relationships* and *associated concept classes* by the priority of all transformations in descending order.

Step 4: According to the order of the sorted list, construct the corresponding learning sequences.

Step 5: Take the *associated concept class* as the *now-class* and go to Step 2 in turn.

Step 6: Return the basic course scheme (a set of learning sequences).

The following is a construction example using the partial DNS ontology shown in Figure 2.3.

(1) Take the core concept class "DNS Server" as the now-class, and find all relationships and associated concept classes of the now-class. Then we find the relationships and associated concept classes are all available, and sort the relationships and associated concept classes. The result is listed in Table 4.1.

Table 4.1: The sorted available relationships and associated concept classes of the now-class "DNS Server"

Relationship	***Associated Concept class***
Component of	*Zone Data*
Component of	*Configuration File*
Type of	*Authoritative DNS Server*
Attribute of	*DNS Availability*
Process of	*DNS Registration*

(2) According to the sorted list, we could construct the corresponded learning sequences of the now-class "DNS Server".

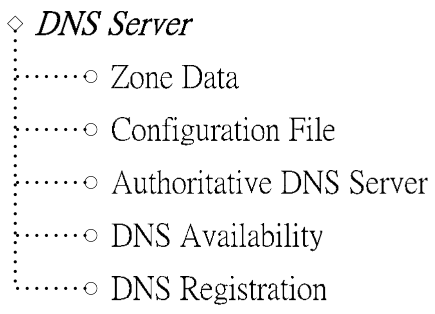


Figure 4.2: Constructing learning sequences about "DNS Server"

(3) Next, take the first associated concept class “Zone Data” as the now-class, and find all relationships and associated concept classes of the now-class. Here we could find the “Component of” relationship associated with “DNS Server” has been used. Thus, we should eliminate this relationship and the associated concept class. Since all the other available relationships are the same, the ordering of relationships and associated concept classes is not changed. The result will be listed in Table 4.2 bellow.

Table 4.2: Available relationships and associated concept classes of the now-class “Zone Data”

Relationship	***Associated Concept class***
Component of	*NS Record*
Component of	*A Record*
Component of	*PTR Record*
Component of	*MX Record*
Component of	*AAAA Record*
Component of	*A6 Record*
~~*Component of*~~	~~*DNS Server*~~

(4) Next, according to the sorted list, we could further construct the corresponding learning sequences of the now-class “Zone Data”. The result is illustrated as Figure 4.3 bellow.

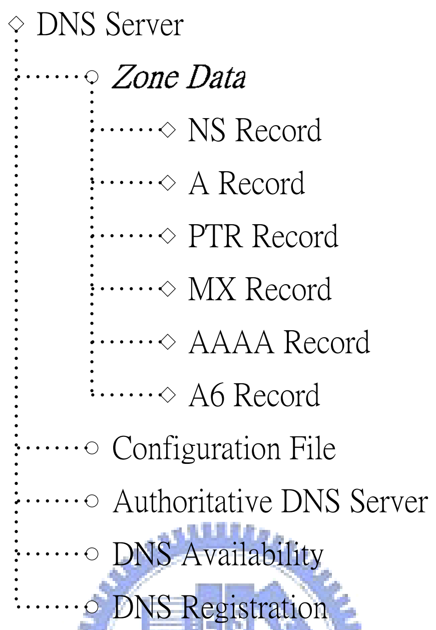


Figure 4.3: The course scheme after constructing learning sequences about “Zone Data”

(5) Next, we will take the concept class “NS Record” as the now-class. We find the “Component of” relationship associated with “Zone Data” has been used, so we eliminate this relationship and the associated concept class. On the other hand, there is another relationship “Pre-requisite” to the concept class “Delegated DNS Server”. Because the associated concept class has other relationships, we eliminate this relationship and the associated concept class, too. Therefore, there is no available relationship around this now-class.

(6) Next, by following the same steps, we take different concept class as the now-class in turn and construct related learning sequences. Until we take the concept class “Single Point of Failure (SPOF)” as the now-class, we could construct the resulting course scheme as illustrated in Figure 4.4 below.

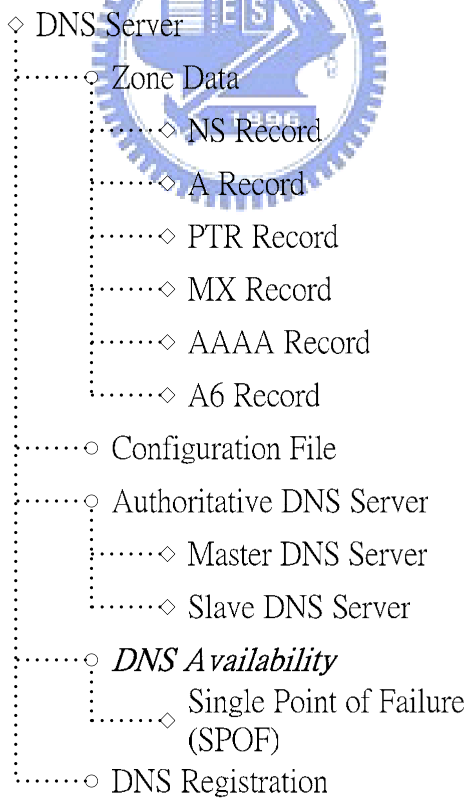


Figure 4.4: The course scheme after constructing learning sequences about “DNS Availability”

(7) Next, we take the concept class "Single Point of Failure (SPOF)" as the now-class. By following the same steps, we can find two available relationships. The two relationships are both "Pre-requisite", and correspond to the definition of transformations, so we can get the following course scheme:

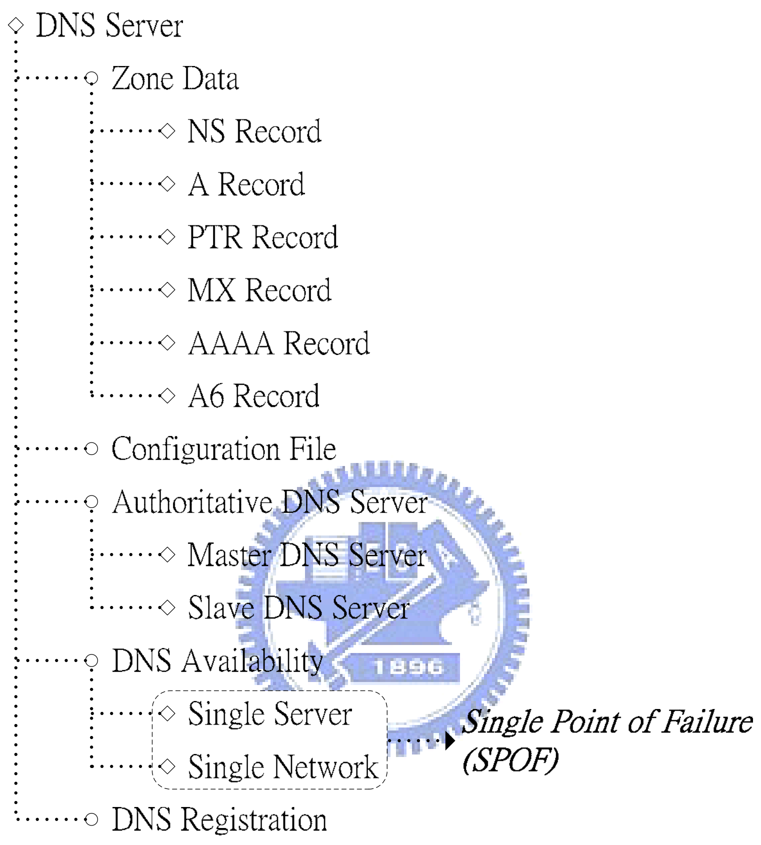


Figure 4.5: Learning sequences constructing about "Single Point of Failure (SPOF)" – insert two necessary sections in front of "Single Point of Failure (SPOF)"

(8) Finally, we generate the basic course scheme, as shown in Figure 4.6, transformed from the partial DNS ontology.

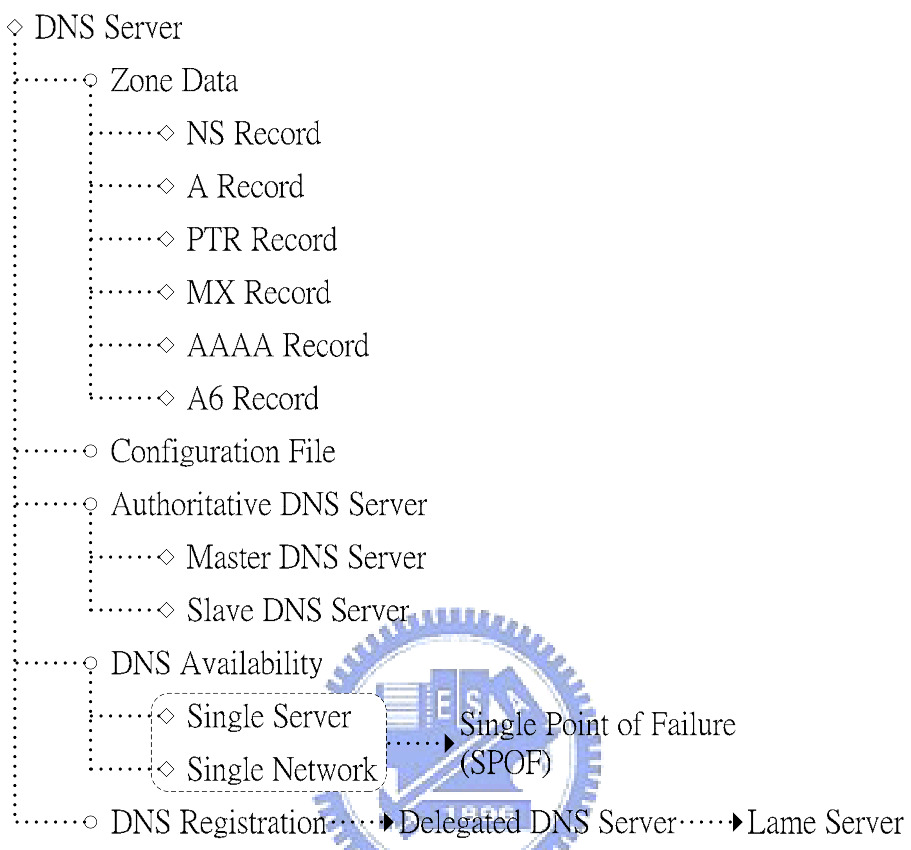


Figure 4.6: Basic DNS course scheme transformed from the partial DNS ontology

Chapter 5. Meta Knowledge Extraction from Rules

As described above, in some domains, rules are often used to represent the domain knowledge in building information systems. In this sense, there would be a lot of knowledge that is represented in rule format in those domains. Hence, to enrich the content of those domain courses, the rule-based knowledge is a very import resource. In this Chapter, we will introduce how to extract meta-knowledge from rules. In addition, we will further explain why this extraction can make the knowledge integration with ontology knowledge and rule-based knowledge smoother.

5.1 Rule-based Knowledge Representation

Before describing the Meta-Knowledge Extraction Module, we will introduce the rule format we used in this system first. Traditionally, rule format could be written as follows:

IF <Condition> THEN <Conclusion>;

For example, a rule about Single Point of Failure (SPOF) in DNS domain is described below:

IF number of NS Records < 2 THEN SPOF = true;

From the traditional rule representation, we know that counting the number of NS Records can check if there is an SPOF (or not) regarding the specified domain zone. However, for some novice DNS administrators, above information is not enough for them to understand why this rule makes sense. In other word, the explanation of the rule is required for some novice DNS administrators. Hence, to meet our tutoring requirement and enhance the readability of the rules, we propose to attach an *Explanation-part* to each rule to explain this rule, and we represent a rule as the extended format:

IF <Condition> THEN <Conclusion>, <Explanation>;

For example, the rule about SPOF with extension would be:

IF number of NS Records < 2 THEN SPOF = true,
Explanation = "If the number of NS Records < 2, it means that there is only one (or even no) DNS server in the specified domain. Therefore, if the only one DNS server is crashed, then users from other Internets sites might not able to access any host in the specified domain, even though they are healthy and available online, since no DNS queries will be acknowledged. This problem of DNS availability is called Single Point of Failure (SPOF)";

In practice, the rules in DNS diagnosis system focus on DNS related problems. In other words, the rules in DNS diagnosis system could provide us the hint about the mis-configuration users may make. The advantages of rule extension above are:

1. Explanation acquisition from domain experts is more easer. From domain experts' point of view, they could provide the explanation from the rules.
2. The information of the rules is more sufficient. From users' view of point, they could gain more DNS related information about the rule.

5.2 Vocabulary Base

As we know, the domain ontology consists of concept classes, and these concepts are usually domain related vocabularies. Thus, when we create a domain ontology, the vocabulary base of the domain would be produced as well. For example, the key terms stored in the vocabulary base of the partial DNS ontology would be *"DNS Server"*, *"Zone Data"*, *"Configuration File"*, *"DNS Availability"*, etc.

In the *Meta-Knowledge Extraction Module*, we need a set of the domain key words to be matched patterns. Based on the following reasons, we use the vocabulary base as the provider of the domain key words:

(1) The concept classes of a domain ontology are also the key terms of the domain.

(2) With the help of the vocabulary base, the meta-knowledge we extract can be used to integrate with the ontology knowledge easily.

5.3 Meta-Knowledge Extraction

In order to add rule-based knowledge into the domain course in an efficient way, we have to process the domain rules in advance. The preprocessing we use here is to extract meta-knowledge from the rules.

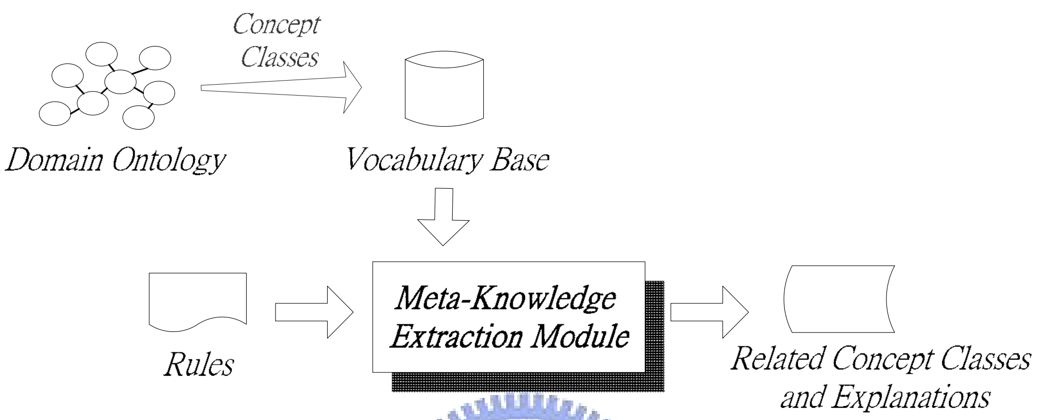

Concept Classes
Domain Ontology
Vocabulary Base
Rules
Meta-Knowledge Extraction Module
Related Concept Classes and Explanations

Figure 5.1: Meta-Knowledge Extraction from Rules

Figure 5.1 shows the process of meta-knowledge extraction. In addition to the domain rules, the vocabulary base, as mentioned in Section 5.2, is also needed. In the Meta-Knowledge Extraction Module, we would extract two kinds of meta-knowledge from each rule. One is the *related concept classes* of the rule, and the other is the *explanation* of the rule.

- Related concept classes

  In this module, we use the condition-part and the conclusion-part as the content of a rule, and perform a pattern-matching process between the content and the vocabulary base. As mentioned in Section 5.2, we know that the vocabulary base stores all the concept classes of a domain ontology, and the concept classes are also key terms in the domain. Hence, the extraction of related concept classes has two advantages:

(1) We can extract the key terms in the domain as the meta-knowledge of each rule.

(2) We can know exactly what concept classes in the domain ontology are related to a rule.

■ Explanation

As specified in Section 5.1, we know that each rule in our system has an explanation-part. Here we would extract this part directly for the tutoring purpose.

The following is an example of meta-knowledge extraction:

◆ **Rule:** IF *number of **NS Records** < 2* THEN ***SPOF*** *=true,*

Explanation = *"If the number of NS Records < 2, it means that there is only one (or even no) DNS server in the specified domain. Therefore, if the only one DNS server is crashed, then users from other Internets sites might not able to access any host in the specified domain, even though they are healthy and available online, since no DNS queries will be acknowledged. This problem of DNS availability is called Single Point of Failure (SPOF)";*

◆ **Related classes:** NS Record

SPOF

◆ **Explanation:** "If the number of NS Records < 2, it means that there is only one (or even no) DNS server in the specified domain. Therefore, if the only one DNS server is crashed, then users from other Internets sites might not able to access any host in the specified domain, even though they are healthy and available online, since no DNS queries will be acknowledged. This problem of DNS availability is called Single Point of Failure (SPOF)"

Chapter 6. Example and Quiz Annotation

Using both the ontology knowledge and the rule-based knowledge of a domain is an efficient way to enrich the content of a course, especially in some domains which have a lot of rule-based knowledge. In this chapter, we will describe the integration of ontology knowledge and rule-based knowledge. The method we used is to annotate the rule-based knowledge into the ontology knowledge with the form of example and quiz. Section 6.2 discusses the working principle of the example annotation, and quiz annotation will be described in Section 6.3.

6.1 Integration of Ontology Knowledge and Rule-based Knowledge

As described in Chapter 4, an ontology is suitable to be transformed into a course scheme. Because the course scheme contains only the ontology knowledge of the domain, we prefer to call it a *basic course scheme*. However, the basic course scheme could not provide individualized learning environment. As described above, rule representation is appropriate in DNS domain and it could reflect users' behavior. Thus, we propose to extract meta-knowledge from the rules and apply the meta-knowledge to enrich the basic course scheme. In this section, we would describe the process of integrating ontology knowledge and rule-based knowledge.

As mentioned in Section 3.2, we know that learning by examples or quizzes could increase the learning efficiency And, the rules could provide us the hits to provide appropriate examples at suitable place in the course scheme. Therefore, we decide to add the rule-based knowledge with the form of ***example annotation*** and ***quiz annotation***. In this thesis, we propose to design the *Example & Quiz Annotation Module* to implement the knowledge integration. As shown in Figure 6.1, there are two main components in this module. *Example Annotation* would decide the position where an explanation should be annotated in the basic course scheme. And, Quiz Annotation would generate some simple quizzes at the bottom of a chapter.

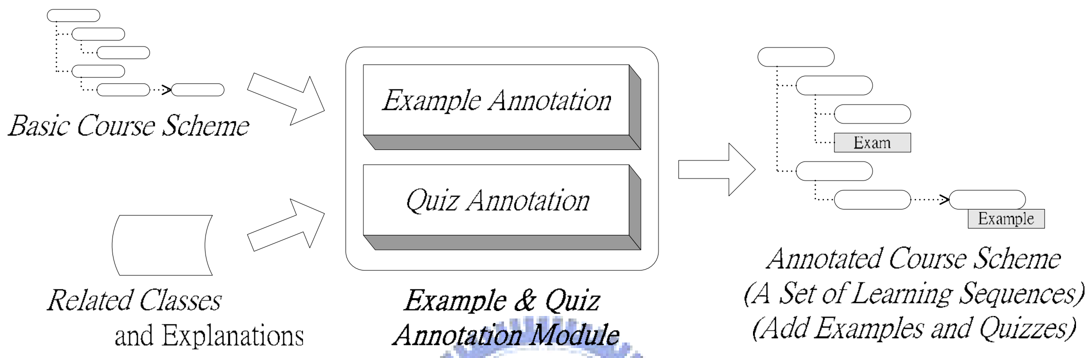

Basic Course Scheme
Related Classes and Explanations
Example Annotation
Quiz Annotation
Example & Quiz Annotation Module
Exam
Example
Annotated Course Scheme
(A Set of Learning Sequences)
(Add Examples and Quizzes)

Figure 6.1: Example & Quiz Annotation Module

6.2 Example Annotation

In practice, examples would usually be shown to a learner after the one has finished studying the related course material since examples can help one understand and remember the content of a course. Hence, if we want to annotate an explanation as an example into the course, we have to know which section is the last related section during the course. In this module, we use the sequence corresponding to the basic course scheme constructed with the domain ontology. By following the sequence of the basic course scheme, we can decide which related concept class of a rule is the last one. The example annotation algorithm is described as follows.

Algorithm 6.1: The example annotation algorithm

Input: The basic course scheme, related classes and explanation of each rule.

Output: The course scheme with examples annotated.

Step 1: Set all the related concept classes of each rule *unmarked*.

Step 2: Start from the first section of the basic course scheme. Take the name of this section as the *now-section*.

Step 3: Check each rule which has any *unmarked* related concept class.

> **Step 3.1:** For a rule, if the *unmarked* related concept class is the same as the *now-section*, then set the related concept class *marked*.
>
> **Step 3.2:** If all related concept classes of the rule are *marked*, then annotate the explanation of the rule as an example into the content of the *now-section*.

Step 4: Go through by the basic course scheme. Take the name of the next section as the *now-section*, and go to Step 2.

Step 5: Return the course scheme which has examples annotated.

Take the basic DNS course scheme we described in Section 4.3 and the DNS rule (about SPOF) we mentioned in Section 5.3 for example, the steps of *Example Annotation* are shown as follows:

(1) The related concept classes of the rule are *"NS Record"* and *"SPOF"*. Set the two related concept classes *unmarked*.

(2) Take "DNS Server" as the *now-section*, and check each *unmarked* related concept class is the same as the *now-section* or not. We found that no *unmarked* related concept class is the same as the *now-section "DNS Server"*.

(3) Then take *"Zone Data"* as the *now-section*, and still no *unmarked* related concept class need to be *marked*.

(4) Take *"NS Record"* as the *now-section*, and set the related concept class *"NS Record" marked*. There is only one related concept class *unmarked* now.

(5) Then take *"A Record"*, *"PTR Record"*, *"MX Record"*, etc. as the *now-section* in turn. Until take *"Single Point of Failure (SPOF)"* as the *now-section*, there is still one related concept class *unmarked*.

(6) Take *"Single Point of Failure (SPOF)"* as the *now-section*. Because the *unmarked* related concept class *"SPOF"* is matched with the *now-section*, we could set this related concept class *marked*.

(7) Now, all related concept classes of the rule are *marked*. Hence, we annotate the explanation of the rule into the content of the *now-section* *"Single Point of Failure (SPOF)"*.

Even though we can use the sequence of the basic course scheme; however, the position that an example will be annotated into is not always suitable for a learner because some sequences in the basic course scheme are not absolute (or fixed in a particular order). We could take the basic DNS course scheme in Section 4.3 as illustration. For example, if we want to learn the content of *"Zone Data"*, we also need to learn the seven sub-level sections. One of the sequences (that we would construct) might be: *"NS Record"* → *"A Record"* → *"PTR Record"*, etc. But, as well as know, this sequence is not absolute. The learner may follow another sequence: *"AAAA Record"* → *"A6 Record"* → *"A Record"*, etc. Or, it would be ok if the learner might even go to learn *"DNS Registration"* directly. But, "Delegated DNS Server" must be introduced after "DNS Registration", because there is a pre-requisite relationship between the two (i.e., this kind of sequence is absolute).

Therefore, if a learner skips the content of *"Zone Data"* and goes to learn the content of *"Single Point of Failure"*, s/he may not understand what the example talks about since s/he doesn't learn *"NS Record"* before. To overcome this problem, we would add the related concept classes of a rule into the corresponding example. A related concept class corresponds to a section in the basic course scheme. Thus, the related concept class can be a link to the corresponding section. By these additional links, learners could know what sections should be learned before the specified example. Even the learner had skipped or forgot the content of some sections, s/he can know which section s/he should go back to learn.

6.3 Quiz Annotation

Just like examples, quizzes can also help learners to understand and to remember the content of course. In this section, we will describe how to generate some simple quizzes using the rule-based knowledge and the domain ontology. In addition, how to decide the location of the quizzes would be introduced as well.

In this thesis, the method to generate simple quizzes refers to Fischer (2001). The main concept is to use the relationships of an ontology. If a concept class ***A*** and a concept class ***B*** have the same (or similar) relationships with another concept class ***C***, then in some points of view, ***A*** and ***B*** have the same (or similar) meanings. For example, suppose there is a rule with key term ***A*** embedded inside. In general, if we replace a key term ***A*** with another key term ***B***, then the semantics of the rule would be wrong, however, it does still have conceptual meanings in some special cases. For example, here is a typical DNS rule concerning SPOF:

IF number of NS Records < 2 THEN SPOF = true,

Explanation = "If the number of NS Records < 2, it means that there is only one (or even no) DNS server in the specified domain. Therefore, if the only one DNS server is crashed, then users from other Internets sites might not able to access any host in the specified domain, even though they are healthy and available online, since no DNS queries will be acknowledged. This problem of DNS availability is called Single Point of Failure (SPOF)";

Based on the partial DNS ontology described in Section 2.2.3, we know that *"NS Record"* and *"A Record"* have the same relationship *"Component of"* with the concept class *"Zone Data"*. Thus, if we replace *"NS Record"* with *"A Record"*, we can generate a syntactically good but semantically wrong rule:

*IF number of **A Records** < 2 THEN SPOF = true,*

*Explanation = "If the number of **A Records** < 2, it means that there is only one (or even no) DNS server in the specified domain. Therefore, if the only one DNS server is crashed, then users from other Internets sites might not able to access any host in the specified domain, even though they are healthy and available online, since no DNS queries will be acknowledged. This problem of DNS availability is called Single Point of Failure (SPOF)";*

In principle, based on the model-tracing tutoring method (Anderson et al., 1990; Anderson and Corbett, 1993), we can generate some candidates of semantically wrong (but syntax-OK) rule-based knowledge. By following the above approach to collocate right rule-based knowledge and wrong rule-based knowledge, we can generate simple quizzes, including true-or-false, single choice, and multiple choices.

On the other hand, similar to the *Example Annotation*, a rule should be annotated to the last related section. But, to generate more variable quizzes (e.g., single choice and multiple choices), we have to collect all annotated rule in a chapter. With more than one rule, we can generate more variable quizzes at the bottom of each chapter.

Chapter 7. Implementation

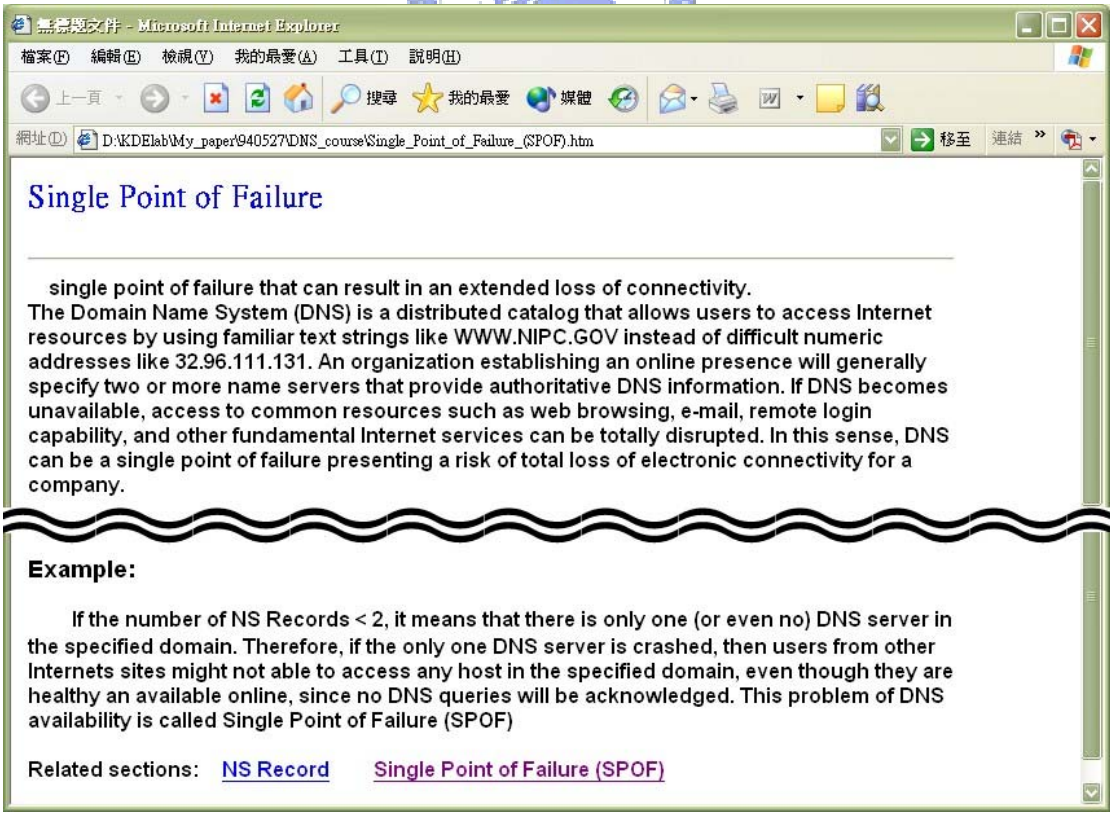
The prototype of our learning sequences construction system is built under Windows XP, Apache web server, and JAVA programming language. We used Protégé 3.0 (http://protege.stanford.edu/) as the ontology editor, and rule-based knowledge was stored in the MySQL database. The generated learning sequences follow the standard of SCORM 2004 and we could put the content package into the SCORM 2004 Sample Run-Time Environment (Version 1.3.3) to show the sample DNS course.

■ Example

In Chen et al. (2003), the authors developed a DNS ontology as the background DNS knowledge model. In this thesis, we adopt partial DNS ontology shown in Figure 2.3 (i.e., adopted from Chen et al., 2003) to be the input domain ontology in our system. For compiling the tutoring materials, we interviewed domain experts to create experimental course materials for each concept class of the DNS ontology. The tutoring materials are composed of HTML files. In principle, each concept class of the DNS ontology has a corresponding HTML file, which was named after the name of the concept class. For example, the concept class "NS Record" has a corresponding HTML file named "NS_Record.htm", which talks about what is NS Record, and How to configure NS Records etc. Through the Ontology-based Learning Sequences Construction Module, the partial DNS ontology could be transformed into a basic DNS course scheme, as shown in Figure 4.6.

The DNS domain rules, used in this thesis, could be separated into two parts. On the one hand, some set of rules were extracted via an ontology-driven model as proposed in Liu et al. (2004). And, the other set of rules were acquired from interviews with domain experts. Both sets of rules are stored in the MySQL database. Moreover, we also extract meta-knowledge from the domain rules in the Meta-Knowledge Extraction Module, and implement the Example & Quiz Annotation Module to annotate rule-based knowledge as examples and quizzes into the basic DNS course scheme.

In the example annotation module, we would annotate examples into some tutoring HTML files. For example, as mentioned in Section 6.2, we would annotate the explanation about NS Record and Single Point of Failure (SPOF) into the “Single_Point_of_Failure_(SPOF).htm”, which is shown in Figure 7.1.


無標題文件 - Microsoft Internet Explorer
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
上一頁 搜尋 我的最愛 媒體
網址(D) D:\KDElab\My_paper\940527\DNS_course\Single_Point_of_Failure_(SPOF).htm
移至 連結

Single Point of Failure

single point of failure that can result in an extended loss of connectivity.
The Domain Name System (DNS) is a distributed catalog that allows users to access Internet resources by using familiar text strings like WWW.NIPC.GOV instead of difficult numeric addresses like 32.96.111.131. An organization establishing an online presence will generally specify two or more name servers that provide authoritative DNS information. If DNS becomes unavailable, access to common resources such as web browsing, e-mail, remote login capability, and other fundamental Internet services can be totally disrupted. In this sense, DNS can be a single point of failure presenting a risk of total loss of electronic connectivity for a company.

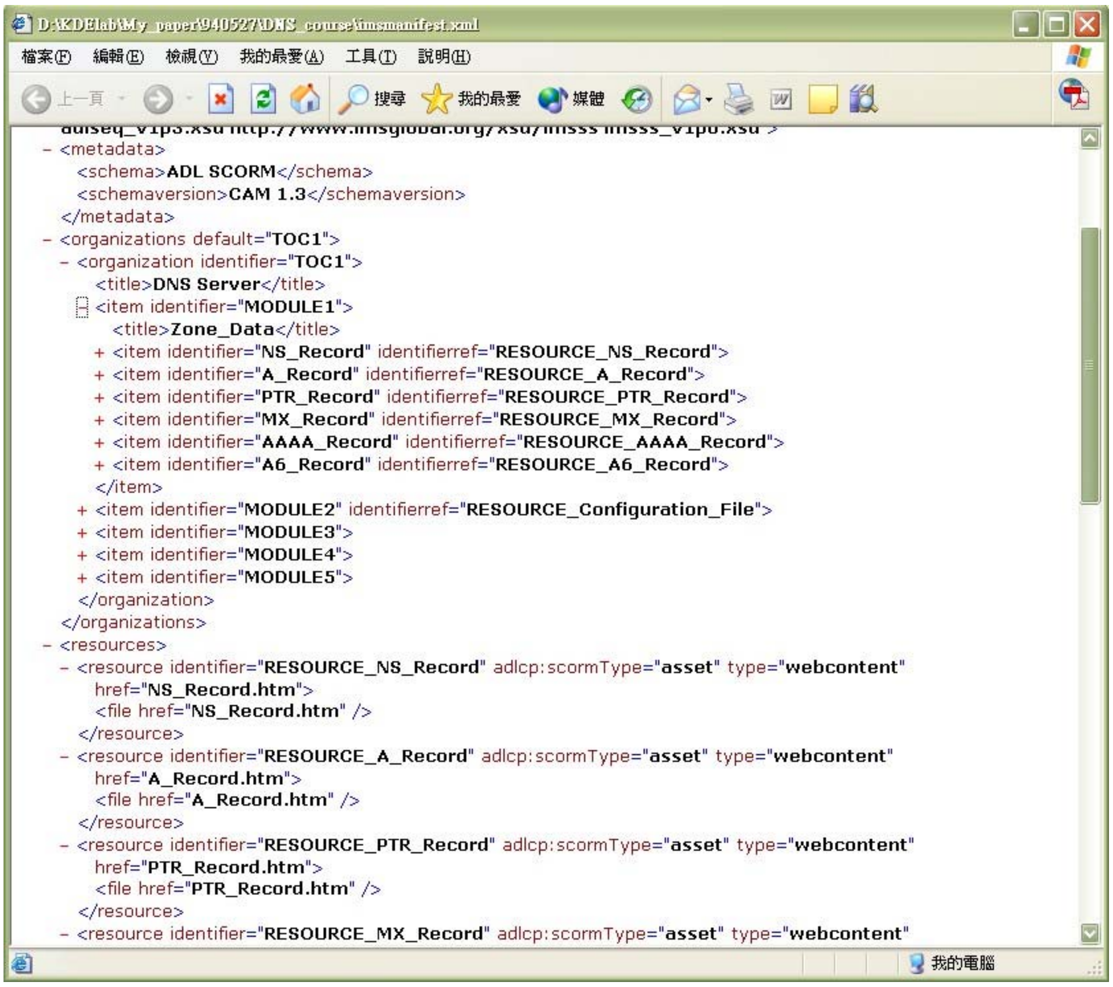
Example:

If the number of NS Records < 2, it means that there is only one (or even no) DNS server in the specified domain. Therefore, if the only one DNS server is crashed, then users from other Internets sites might not able to access any host in the specified domain, even though they are healthy an available online, since no DNS queries will be acknowledged. This problem of DNS availability is called Single Point of Failure (SPOF)

Related sections: NS Record Single Point of Failure (SPOF)

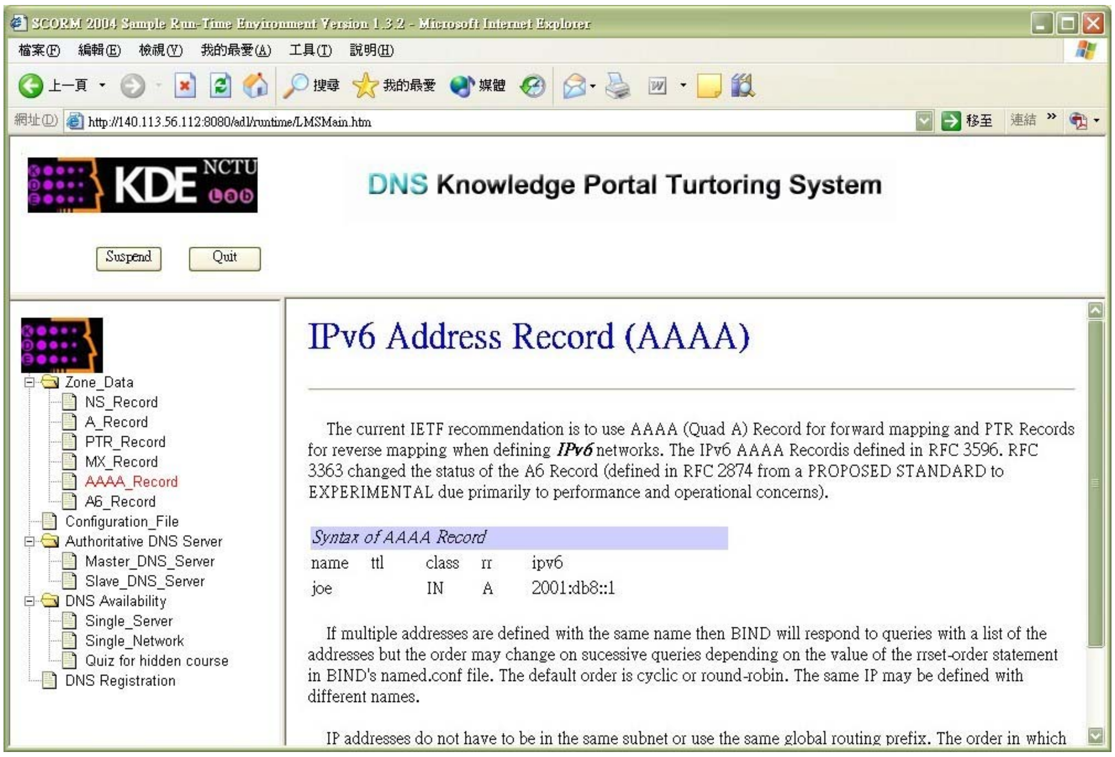
Figure 7.1: The tutoring material of “Single Point of Failure (SPOF)”

In the quiz annotation module, some quiz HTML files would be generated and put into suitable positions of the course scheme. The primary output is an XML file named "imsmanifest.xml", which records all the learning sequences in a well-structured format. Figure 7.2 shows partial content of the XML file. The XML file and all its related HTML files would be packaged into a content package, which is a ZIP file in reality. Then, we put this content package into the SCORM 2004 Sample Run-Time Environment to show the sample DNS course as the one shown in Figure 7.3 (i.e., the section of IPv6 AAAA resource record).


D:\KDElab\My_paper\940527\DNS_course\imsmanifest.xml
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
上一頁
搜尋
我的最愛
媒體
我的電腦

```
  adlseq_v1p3.xsd http://www.imsglobal.org/xsd/imsss imsss_v1p0.xsd">
- <metadata>
    <schema>ADL SCORM</schema>
    <schemaversion>CAM 1.3</schemaversion>
  </metadata>
- <organizations default="TOC1">
  - <organization identifier="TOC1">
      <title>DNS Server</title>
      <item identifier="MODULE1">
        <title>Zone_Data</title>
      + <item identifier="NS_Record" identifierref="RESOURCE_NS_Record">
      + <item identifier="A_Record" identifierref="RESOURCE_A_Record">
      + <item identifier="PTR_Record" identifierref="RESOURCE_PTR_Record">
      + <item identifier="MX_Record" identifierref="RESOURCE_MX_Record">
      + <item identifier="AAAA_Record" identifierref="RESOURCE_AAAA_Record">
      + <item identifier="A6_Record" identifierref="RESOURCE_A6_Record">
      </item>
    + <item identifier="MODULE2" identifierref="RESOURCE_Configuration_File">
    + <item identifier="MODULE3">
    + <item identifier="MODULE4">
    + <item identifier="MODULE5">
    </organization>
  </organizations>
- <resources>
  - <resource identifier="RESOURCE_NS_Record" adlcp:scormType="asset" type="webcontent"
      href="NS_Record.htm">
      <file href="NS_Record.htm" />
    </resource>
  - <resource identifier="RESOURCE_A_Record" adlcp:scormType="asset" type="webcontent"
      href="A_Record.htm">
      <file href="A_Record.htm" />
    </resource>
  - <resource identifier="RESOURCE_PTR_Record" adlcp:scormType="asset" type="webcontent"
      href="PTR_Record.htm">
      <file href="PTR_Record.htm" />
    </resource>
  - <resource identifier="RESOURCE_MX_Record" adlcp:scormType="asset" type="webcontent"
```

Figure 7.2: A snapshot of "imsmanifest.xml" which follows the standard of SCORM 2004


SCORM 2004 Sample Run-Time Environment Version 1.3.2 - Microsoft Internet Explorer
檔案(F) 編輯(E) 檢視(V) 我的最愛(A) 工具(T) 說明(H)
上一頁 搜尋 我的最愛 媒體
網址(D) http://140.113.56.112:8080/adl/runtime/LMSMain.htm
移至 連結
KDE NCTU Lab
DNS Knowledge Portal Turtoring System
Suspend
Quit
Zone_Data
NS_Record
A_Record
PTR_Record
MX_Record
AAAA_Record
A6_Record
Configuration_File
Authoritative DNS Server
Master_DNS_Server
Slave_DNS_Server
DNS Availability
Single_Server
Single_Network
Quiz for hidden course
DNS Registration

IPv6 Address Record (AAAA)

The current IETF recommendation is to use AAAA (Quad A) Record for forward mapping and PTR Records for reverse mapping when defining ***IPv6*** networks. The IPv6 AAAA Recordis defined in RFC 3596. RFC 3363 changed the status of the A6 Record (defined in RFC 2874 from a PROPOSED STANDARD to EXPERIMENTAL due primarily to performance and operational concerns).

Syntax of AAAA Record

name	ttl	class	rr	ipv6
joe		IN	A	2001:db8::1

If multiple addresses are defined with the same name then BIND will respond to queries with a list of the addresses but the order may change on sucessive queries depending on the value of the rrset-order statement in BIND's named.conf file. The default order is cyclic or round-robin. The same IP may be defined with different names.

IP addresses do not have to be in the same subnet or use the same global routing prefix. The order in which

Figure 7.3: DNS course runs on the run-time environment

Chapter 8. Conclusion

In this thesis, we propose a SCORM-based learning sequence construction model using ontology and rules to simplify the complexity of learning sequence construction. The proposed study will include the insights into how various DNS knowledge source could be integrated to help define the learning sequence behavior and the learner navigation by using sequence and navigation model. In essence, our main contributions are listed as follows:

1. We propose a SCORM-based learning sequence construction algorithm, which is based on ontology and rules. The paradigm of integrating ontology and rules could benefit from domain knowledge model and users' behaviors.

2. We design and implement a DNS e-learning prototype system based on the SCORM-based learning sequences constructed from above algorithm.

In this thesis, we adopt both domain ontology and rules as knowledge representations to facilitate constructing DNS learning sequences. As mentioned, we propose two algorithms, including the *ontology-based learning sequence construction algorithm* and the *example annotation algorithm*, to fulfill the learning sequence construction scheme. First of all, the *Ontology-based Learning Sequences Construction Module* and *Meta-Knowledge Extraction Module* form the preprocessing components of the scheme. In essence, ontology representation could help domain experts to model DNS knowledge because it focuses on concept classes, attributes of the concept classes, and the relationships between the concept classes. On the other hand, the main components of rule representation are facts and actions, which could be used to represent users' behaviors. Next, the *Example & Quiz Annotation Module* could be used to integrate the preprocessing results into a recommended course scheme. Besides, the recommended course scheme could be further refined using some existing course refinement tools. Furthermore, additional information (e.g., user profiles) could be integrated into our learning sequences construction process. In summary, the paradigm of using ontology and rules to construct learning sequence would provide an adaptively learning sequence scheme to help alleviate the domain experts' loadings. We have submitted this thesis to the International Conference on Computers in Education, 2005 (Chen et al., 2005).

References

[Anderson et al. 1990] Anderson, J. R., Boyle, C. F., Corbett, A., & Lewis, M. (1990). Cognitive modeling and intelligent tutoring. Artificial Intelligence, 42, 7-49.

[Anderson & Corbett 1993] Anderson, J.R. and Corbett, A.T. (1993). Tutoring of cognitive skill. In J.R. Anderson, Rules of the Mind (pp. 235-255). Hillsdale, NJ: Erlbaum

[Chandrasekaran et al. 1999] B. Chandrasekaran, John R. Josephson, and V. Richard Benjamins, "What Are Ontologies, and Why Do We Need Them?", *IEEE Intelligent Systems*, 14 (1): 20-26, 1999.

[Chen et al. 2002] Chen, C.S., Tseng, S.S., Liu, C.L., Ou, C.H., "Building a DNS ontology using METHONTOLOGY and Protege-2000", *Proceedings of 2002 International Computer Symposium Workshop on Artificial Intelligence*, Dec., 18-21, 2002.

[Chen et al. 2003] Chen, C.S., Tseng, S.S., Liu, C.L., "A unifying framework for intelligent DNS management", *International Journal of Human - Computer Studies*, 58(4): 415 – 445, 2003.

[Chen et al. 2005] Chen, R.Y., Tseng, S.S., Liu, C.L., Chang, C.Y., Chen, C.S., "Learning Sequences Construction Using Ontology and Rules", submitted to the International Conference on Computers in Education, 2005.

[Davis et al. 1993] Randall Davis, Howard Shrobe, and Peter Szolovits, "What is a Knowledge Representation?", *AI Magazine*, 14(1): 17-33, 1993.

[Fernandez 1999] Fernandez Lopez, M., "Overview of Methodologies for Building Ontologies", *Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving Methods, Stockholm, Sweden*, 1999.

[Fernandez et al. 1997] Mariano Fernandez, Asuncion Gomez-Perez, and Nataliz Juristo, "METHONTOLOGY: From Ontological Art Towards Ontological Engineering", *Proceedings of Workshop on Ontological Engineering: AAAI-97 Spring Symposium Series*, 1997.

[Fischer 2001] Stephan Fischer, "Course and Exercise Sequencing Using Metadata in Adaptive Hypermedia Learning Systems", *ACM Journal of Educational Resources in Computing*, 1(1), Spring 2001.

[Gaines & Shaw 1993] Brian R. Gaines and Mildred L. G. Shaw, “Eliciting Knowledge and Transferring it Effectively to a Knowledge-Based System”, *IEEE Transactions on Knowledge and Data Engineering*, 5(1): 4-14, 1993.

[Gruber 1993] Thomas R. Gruber, “A Translation Approach to Portable Ontology Specifications”, *Knowledge Acquisition*, 5(2): 199-220, 1993.

[Jones 2004] Jones, E.R., 2004, Dr. Ed’s SCORM Course, http://www.scormcourse.jcasolutions.com/index.php

[Liu et al. 2004] Liu, C.L., Tseng, S. S. and Chen, C. S., "Design and Implementation of an Intelligent DNS Management System", *Expert Systems with Applications*, 27(2): 223-236, August 2004

[Men & Mice] Man & Mice Company. Domain Health Survey for .COM. http://www.menandmice.com

[Mockapetris 1987-1] P. Mockapetris, “Domain Names - Concepts and Facilities”, *RFC 1034*, November 1987.

[Mockapetris 1987-2] P. Mockapetris, “Domain Names - Implementation and Specification” *RFC 1035*, November 1987

[Ou 2002] Ou, C.H., “Design of An Intelligent DNS Planning and Management System”, *Master Thesis, Dept. of Computer and Information Science, National Chiao-Tung University, Taiwan*, 2002

[RELOAD] Reusable eLearning Object Authoring & Delivery, http://www.reload.ac.uk

[SCORM] Sharable Content Object Reference Model, http://www.adlnet.org/scorm/index.cfm.

[Su et al. 2005] Su, J.M., Tseng, S.S., Chen C.Y., Weng, J.F., and Tsai, W.N., “Constructing SCORM compliant course based on High-Level Petri Nets”, *Computer Standards & Interfaces, 2005*.