國立交通大學

資訊科學系

碩 士 論 文

以高階派翠網路為基礎建立一個符合
SCORM 標準的課程

Constructing SCORM Compliant Course

Based on High Level Petri Nets

研 究 生：陳家瑜

指導教授：曾憲雄 博士

中 華 民 國 九 十 三 年 六 月

以高階派翠網路為基礎建立一個符合 SCORM 標準的課程
Constructing SCORM Compliant Course
Based on High Level Petri Nets

研 究 生：陳家瑜　　Student：Chia-Yu Chen

指導教授：曾憲雄　　Advisor：Shian-Shyong Tseng

國 立 交 通 大 學
資 訊 科 學 研 究 所
碩 士 論 文

A Thesis
Submitted to Institute 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 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

以高階派翠網路為基礎建立一個符合 SCORM 標準的課程

研究生：陳家瑜　　　　　　　　　　　　　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

摘要

隨著網路的普及，許多網路學習相關標準被制定。在這些標準中，SCORM 是最被廣泛應用的一個。它不僅定義了如何以學習資源來組合成一個樹狀課程架構(Activity Tree)，並提供了 Sequencing 的功能來幫助老師編輯學習的序列，大大提升網路學習的適性化設計。然而，由於 SCORM 定義的相關內容皆是以標記式語言(XML)編寫，而且課程序列的設計往往相當複雜，這使得課程編輯成為相當的棘手的問題。因此，我們以高階派翠網路(High Level Petri Net，HLPN)為基礎將 SCORM 的課程序列物件化，稱為物件化活動樹(Objected Oriented Activity Tree，OOAT)。根據不同的教學策略將這些物件加以組合，可建立出一個高階派翠網路的課程模組；再透過一個知識轉換的過程，將這個課程模組轉換成 SCORM 相符樹狀課程架構，並自動產生相對應的 XML 碼來描述課程的序列。最後，透過 Content Aggregation 的機制加入實體教材資源，產生一個可在 SCORM RTE 上執行的 Content Package。我們將這整個流程稱為”物件導向課程模型(Object Oriented Course Modeling, OOCM) ”。由於物件化的特性，OOCM 可以簡化課程序列編輯的複雜度，透過不同序列模組的組合，可以彈性的設計出符合 SCORM 標準

的教學策略，免去編寫 XML 碼的繁複手續，大大提高了課程編輯的效率。

Constructing SCORM Compliant Course Based on High Level Petri Nets

Student : Chia-Yu Chen **Advisor : Dr. Shian-Shyong Tseng**

Institute of Computer and Information Science
National Chiao Tung University

Abstract

Sharable Content Object Reference Model (SCORM) 2004 is the most popular standard in e-learning. It provides the Sequencing and Navigation (SN) version 1.3 to define the course sequencing behavior. However, the complicated rules and controls of SCORM activity tree make the course sequences hard to design. Therefore, how to provide a user-friendly authoring tool to easily and fast construct SCORM compliant course becomes an important issue. However, before developing the authoring tool, how to provide an approach to analyze the sequencing rules and transform the created course into SCORM compliant are our concerns.

Therefore, in this thesis, we propose a systematic approach, called **Object-Oriented Course Modeling (OOCM),** to construct a SCORM-compatible course. **High-Level Petri Nets** (**HLPN**), which is a powerful language for system modeling and validation, are applied to modularize the complex sequencing behaviors of SCORM into several sequencing components, which we call them **Object-Oriented Activity Trees (OOATs)**. Therefore, these OOATs can be easily and fast used to construct complex sequencing behaviors of course for different learning guidance. Then, we also propose two algorithms to transform the constructed HLPN of course into SCORM compliant file described by XML language, which can be executed on the SCORM RTE system.

Keywords: High Level Petri Nets(HLPN), SCORM, Sequencing Definition Model (SDM), Course Sequencing, Learning Activity

致謝

這篇論文的完成，必須感謝許多協助與支持的人。首先必須感謝我的指導教授-曾憲雄博士。由於老師細心的指導與勉勵，讓我得以順利完成此篇論文，並且在做學問、做事態度、思考邏輯、以及待人接物方面，也有很大的啟發，非常的感謝老師。同時也必須感謝我的論文口試委員:孫春在教授、黃國楨教授, 以及游寶達教授，他們也對這篇論文提出了許多寶貴的意見，讓此篇論文更顯得有價值與意義。

此外，我感謝實驗室中各位學長、同學平日的諸多照顧與協助，尤其是蘇俊明學長和翁瑞鋒學長，他們對於此篇論文的完成，亦花費了許多心血與精神，提供了許多寶貴的意見. 在與學長們互相討論勉勵的過程中，讓我受益良多;不論是在做學問的態度或是待人處世，都有著顯著的成長。

而實驗室的同窗夥伴，Peggy、培綺、阿肯、王威和建豪等人互相幫助鼓勵的情誼，更讓忙碌的碩士班兩年的生涯，能夠愉快而充實的度過。

最後，我要感謝家人對我的支持與鼓勵，以及其他無法一一提及的朋友, 你們的愛護與支持都是讓我努力向前的最大動力。

Table of Content

List of Figures

Chapter 1: Introduction

In recent years, e-learning system has become more and more popular. For the consistency of course format, many standards have been proposed. Sharable Content Object Reference Model (SCORM) 2004, the most popular standard in E-learning, provides the Sequencing and Navigation (SN) Version 1.3 to define the course sequencing behavior. However, the complicated rules and controls of SCORM activity tree make the course sequences hard to design. Therefore, how to provide a user-friendly authoring tool to easily and fast construct SCROM compliant course becomes an important issue. The learning guidance of a course can be represented as a graph which is easier to understand and to create for teachers and authors. Accordingly, if we can provide an authoring tool that teachers and authors can edit the structure of course with sequencing rules by graph representation and then transforms the created structure into SCORM compliant file automatically, the SCORM compliant course can become more and more popular. However, before developing the authoring tool, how to provide an approach to analyze the sequencing rules and to transform the created course into SCORM compliant are our concerns.

Therefore, in this thesis, we propose a systematic approach, called **Object-Oriented Course Modeling (OOCM),** to construct a SCORM-compatible course. **High-Level Petri Nets** (**HLPN**), which is a powerful language for system modeling and validation, are applied to modularize the complex sequencing behaviors of SCORM into several sequencing components-**Object-Oriented Activity Trees(OOATs)**. Therefore, these OOATs can be easily and fast used to construct complex sequencing behaviors of course for different learning guidance. Then, we also propose two algorithms to transform the constructed HLPN of course into SCORM compliant file described by XML language, which can be executed on the SCORM RTE system.

The remainder of the article is structured as follows: In the Chapter 2, we introduce some related works about e-learning environment including SCROM standard, AT technologies, course sequencing applying High Level Petri Nets (HLPNs). The Chapter 3 describes the proposed framework, called OOCM. The process of Activity Tree transformation is described in Chapter 4. Then, some implementations of learning strategies based on OOAT are discussed in the Chapter 5. Finally, the Chapter 6 and 7 are the discussion and conclusion respectively.

Chapter 2: Related Work

In last five years, the researches of e-learning have become more and more important. Therefore, in this Section, we will introduce the popular standard, Sharable Content Object Reference Model (SCORM). Then, the adaptive learning environments applying AI technologies and Petri Nets will be also discussed.

2.1 Sharable Content Object Reference Model (SCORM)

In recent years, many e-learning standards have been developed. The most popular standard, Sharable Content Object Reference Model (SCORM) [2], which contains the definitions about the meta-data of learning material, Content Aggregation Model (CAM) which defines how to organize a course into a tree-like structure called Activity Tree (AT). Figure 1 shows an example of AT. It is a structure that provides the hierarchical organization of learning content. According to SCORM 1.3 specification, an AT is structured by a set of clusters. A cluster is an organized aggregation of activities consisting of a single parent activity and its first level children, but not the descendants of its children. The cluster is considered the basic sequencing building block. The parent activity of a cluster will contain the information about the sequencing strategy for the cluster. The status information of all child activities will be collected and can be used to sequence these activities in the structure.

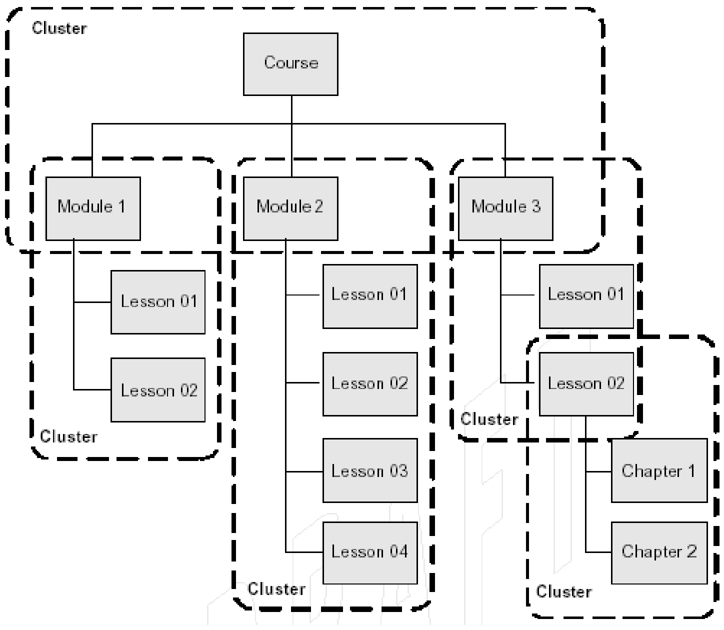


Figure 1: An Activity Tree with clusters

2.1.1 The Sequencing and Navigation (SN) Specification

The *SCORM Sequencing & Navigation (SN) Specification* is based upon the IMS Simple Sequencing Definition Model [1]. It provides a profile about information of specific behaviors between activities and restrictions while learning an activity. The *Sequencing Definition Model* (SDM) defines the following categories: *Sequencing Control Modes, Sequencing Rules, Limit Conditions, Auxiliary Resource, Objectives, Objective Map, Rollup Controls, Selection Controls, Randomization Controls* and *Delivery Controls*. In this thesis, the *Sequencing Control Modes, Sequencing Rules, Objectives* and *Rollup Controls* are our concerns.

(1) **Sequencing Control Mode (SCM):**

The Sequencing Control Mode (SCM) allows the content developer to affect how navigation requests are applied to a cluster and how the cluster's activities are considered while processing sequencing requests. Table 1 describes the SCM that may be applied. Sequencing Control Modes can be applied to any activity in the AT and multiple modes are enabled to create combination of control mode behaviors. Nevertheless, the *Sequencing Control Choice*, *Sequencing Control Flow* and *Sequencing Control Forward Only* modes will have no effect if applied to leaf activities.

Table 1: The Description of Sequencing Control Mode (SCM)

SDM	Description
Sequencing Control Choice	Indicates that a Choice navigation request is permitted to target the children of the activity
Sequencing Control Choice Exit	Indicates that the activity is permitted to terminate if a Choice sequencing request is processed.
Sequencing Control Flow	Indicates the Flow Sub-process may be applied to the children of the activity.
Sequencing Control Forward Only	Indicates that backward targets (in terms of Activity Tree traversal) are not permitted for the children of the activity.
Use Current Attempt Objective Information	Indicates that the Objective Progress Information for the children of the activity will only be used in rule evaluations and rollup if that information was recorded during the current attempt on the activity.
Use Current Attempt Progress Information	Indicates that the Attempt Progress Information for the children of the activity will only be used in rule evaluations and rollup if that information was recorded during the current attempt on the activity.

(2) The Sequencing Rule:

The IMS Simple Sequencing Specification (IMS SSS) employs a rule-based sequencing model. The behaviors between activities are defined by Sequencing Rules. Sequencing Rule is composed of a set of conditions and a corresponding action. The structure of Sequencing Rule is: ***if*** *[condition_set]* ***then*** *[action]*.

The conditions are evaluated using tracking information with the activity. The action of Sequencing Rule will be triggered if its condition-set evaluates to True. There are three kinds of Sequencing Actions SCORM proposes: ***Precondition Actions***, ***Post-condition Actions*** and ***Exit Actions***, which describe different learning strategies.

(3) Objective:

IMS SSS proposes a mechanism of objectives of each activity for sequencing propose. Each learning objective associated with an activity will have a set of tracking status information which is used to decide which sequencing decision should be triggered according to student's current learning progress. Two kinds of learning objective are defined in IMS SSS: ***Local Objective*** and ***Global Shared Objective***. The Local Objective is only referenced by one activity; however, the Global Shared Objective can be shared by sets of activities. Therefore, activities may have more than one associated local objective and may reference multiple global shared objectives. Figure 2 shows an example of objectives. All objectives except Objective 5 are local to their associated activities; Objective 5 is a global shared objective shared between Activity **AA** and Activity **BB**.

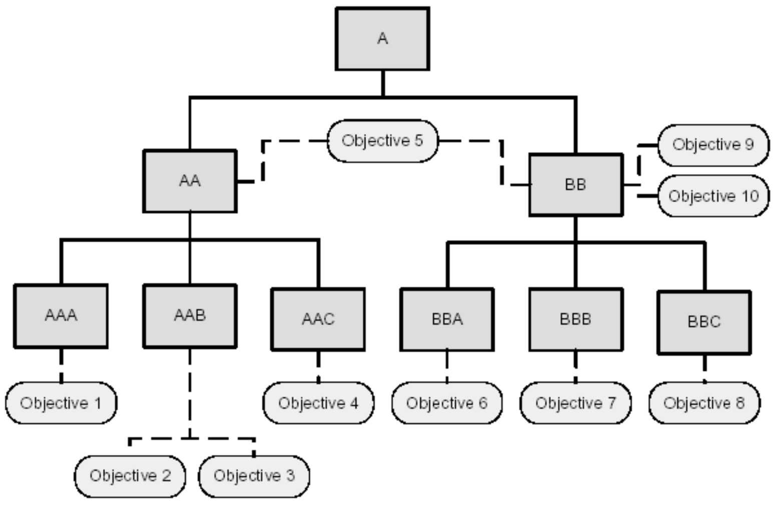


Figure 2: An example of objectives

(4) The Rollup Rule

Cluster activities are not associated with teaching materials; therefore, there is no direct way for learner progress information to be applied to a cluster activity. The IMS SSS defines the way of how to evaluate the learner progress of cluster activity. The structure of Rollup Rule is: ***if** [condition_set] True **for** [child activity set] **then** [action]*. The conditions of Rollup Rule are evaluated against the tracking information of the included child activities, and a corresponding action will set the cluster's tracking status information if the conditions are evaluated to True.

2.1.2 The Tracking Model

The Tracking Model is a collection of dynamic sequencing state information associated with each activity in the Activity Tree for each learner. Tracking Model elements will be updated to reflect learner interactions with the currently launched content object during a learning experience. It defines the following sets of tracking status information:

(1) Objective Progress Information: describe the learner's progress related to a learning objective.

(2) Activity Progress Information: describe a learner's progress on an activity. This information describes the cumulative learner progress across all attempts on an activity.

(3) Attempt Progress Information: describe a learner's progress on an activity. This information describes and the per attempt progress on an activity.

Figure 3 shows the Tracking Models for an Activity Tree.

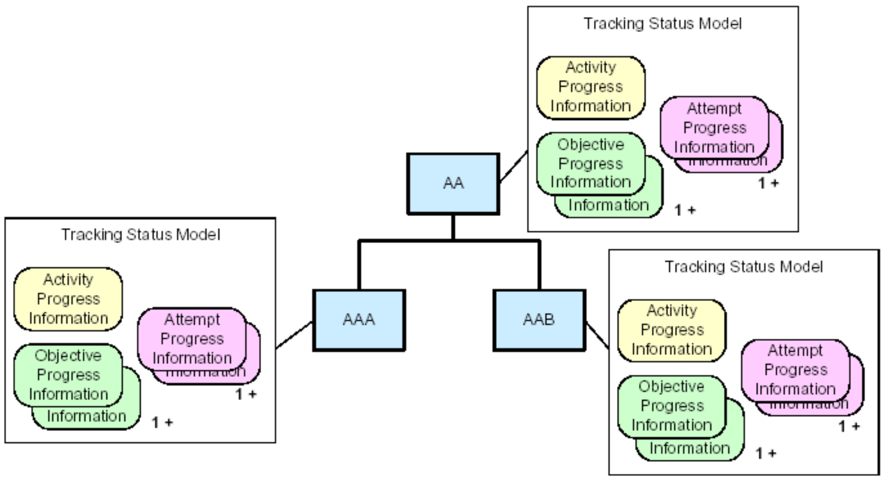

Tracking Status Model
Activity Progress Information
Attempt Progress Information
Objective Progress Information
Information
1 +
1 +
AA
AAA
AAB
Tracking Status Model
Activity Progress Information
Attempt Progress Information
Objective Progress Information
Information
1 +
1 +
Tracking Status Model
Activity Progress Information
Attempt Progress Information
Objective Progress Information
Information
1 +
1 +

Figure 3: The Tracking Models

Currently, more and more researches about constructing an intelligent tutoring system based on SCORM standard. The article of [3] proposes several sequencing templates to support various strategies of instructional design. These templates can be combined with another one, to create a complex learning experience for students. However, the processes of building an Activity Tree and defining sequencing behaviors are very complicated for teachers, because the formats of meta-data and Simple Sequencing are described by XML. The functionality within a lesson or between lessons is hard-coded whether based on linear or an adaptive model. It means teachers must edit lots of XML files for building a course; definitely, it will bring more burdens to teachers and limit the reusability of individual learning objects (SCOs). It also limits the ability to create new or custom content structures from the same instructional materials. Therefore, it is imperative to develop an easy-to-use tool for editing existing SCORM-compatible content packages [9].

2.2 E-Learning Environment Applying AI Technologies

With the development of E-Learning, many articles about course sequencing designs have been proposed. Graph theories are usually used to represent the course model, such as AND/OR graph [11][12][18]. Therefore, teachers can express the relations between sub-courses, such as "*prerequisite*", "*corequisite*", "*related*", which define the order of delivering learning materials, and specify the conditions of achieving learning goals. Some articles [3][15][16] model a course with hierarchical levels of structure. The study on [15] proposed three levels of knowledge abstractions which describe relations between learning goals, concepts and educational materials. The domain model in [16] which is based on the hierarchical structure and concept graph representation describes the knowledge taxonomy.

Course sequencing is an important issue in an intelligent tutoring system. It provides students with the most suitable, individually planned, sequence of knowledge units to learn and learning tasks to work with according to their learning progress. Lots of researches were applied AI technologies to the curriculum sequencing, such as neural network [6][8][13][20], fuzzy theory[6], Baysian network[7], planning algorithm[11][12]etc. Because of the characteristic of self-learning of AI technologies, applying these to develop the e-learning system is able to support more individualized and flexible environment for students, and tailor learning paths according to different students' background knowledge and preferences. However, these researches usually used complex operations during sequencing course and were not applied uniform standard format to specify metadata for content structure and sequencing behaviors. Therefore, these systems may not have interoperability between different systems.

2.3 Petri Net

Petri Net is a graphical and mathematical modeling tool, which can be used to express causality, selection, and synchronization of behaviors, are applicable to many systems [21]. High Level Petri Net (HLPN) [14][19] is the extension of Petri Nets, which is able to model and validate more complex systems. It consists of three different parts: the *net structure*, the *declarations* and the *net instructions*.

(1) ***Net Structure*****:** it is a directed graph with two kinds of nodes, places and transitions. They are interconnected by arcs which are either from a place to a transition or from a transition to a place. A place (drawn as a circle) with their tokens reflects states of a system, while the firing of a transition (drawn as a thin rectangle) reflects notions of state changes.

(2) ***Declarations*****:** define the tokens and variables which can be viewed as the mediums of communicating between places and transitions. Each place can have one or more tokens and each of these carries a data value called token color. A distribution of tokens on the places is called marking; the initial marking represents the initial state. A transition is said to be enabled if there are enough tokens with the correct colors in each input place of the transition. The token colors in the input places will be removed and add correct token colors to the output places while a transition is fired.

(3) ***Net Inscription*****:** can be attached to a place, transition or arc. Two main inscriptions for a transition and arc are *guards* and *arc expressions*. The guard is a firing rule that criticizes whether a transition is fired or not according to the token colors transferred to it. The arc expression determines how many and which kinds of token colors will be removed from each input place of a transition and added to its output places. The transition table is defined in some articles [10] which contain the definitions of guards and arc expressions for a transition.

Figure 4 is an example of high-level Petri Net, where:

- P_1, P_2 and P_3 are places.

- t_1 and t_2 are transitions.

- <b,a>, <b,c> and <a,c> are token colors in P_1. In this example, they are the initial marking.

- In the transition tables of t_1 and t_2 , the left side defines what will be removed from the input places (P_1 for the transition t_1; P_2 for the transition t_2) while the right side specifies what will be added to each output place (P_2 for the transition t_1; P_3 for the transition t_2)

Through a substitution {b|x, a|y, c|z}, the transition t_1 is enabled because P_1 have enough tokens with correct colors (<b,a> and <b,c>)

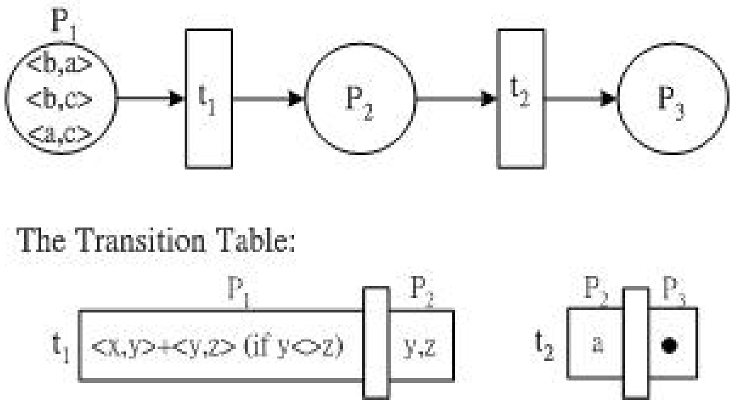


Figure 4: An Example of High-Level Petri Nets

Besides, the HLPN also provides the conception of hierarchical construction. The hierarchy construct is called a subnet which encapsulates parts of the HLPN. In other words, in HLPN the complexity of a model can be divided between the net structure, the net inscriptions and the declarations. Developers can model a simple abstract of a process and then specify a more detailed description of the activity represented by this subnet. Figure 5 shows subnets which are contained in HLPN. The transition t_2 contains a subnet.

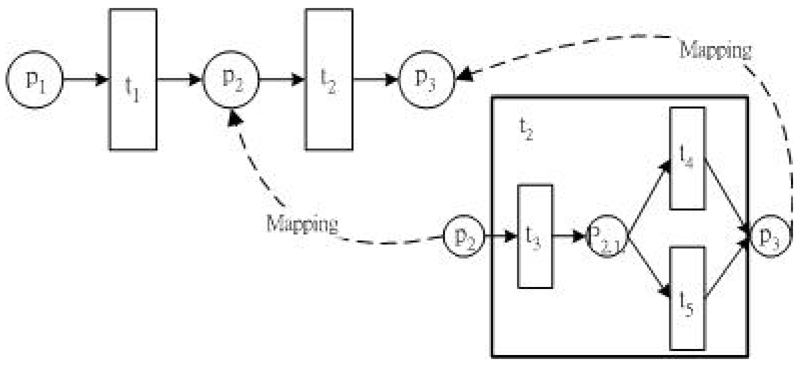


Figure 5: The Hierarchical Construction of a High Level Petri Nets

2.4 Course Module Applying High Level Petri Nets

Researches on modeling teaching plans by Petri nets have been conducted [5][23]. In the study of [5], it proposes an approach to model online instruction knowledge for developing online training systems. Two-level specialized Petri nets are used, one for representing goal-oriented training plans (TP-net) and another for task-oriented training scenarios (TS). A TP-net represents non-sequential causality among tasks, and the TS is a set of corresponding training scenarios for each task. Combining a TP-net and all TS-nets to form a goal-oriented training model Petri net (GOTM-net) which is converted as a set of "*if-then*" rules representing the behaviors a student may perform and the corresponding responses. However, no standard format, such as SCORM, has been applied to describe the content structure and sequencing behaviors. It can reduce the interoperability. Besides, authoring TP-net and TS-nets is complicated without friendly tools.

The research in [23] discussed meta-data structure, which is based on SCORM 1.2 version, makes a base for reusing and aggregation learning resources in e-learning, and provides an aggregation model-Teach net based on High-Level Petri Nets. However, the Teach net is mainly used to model the content aggregation and no course sequencing is mentioned in SCOMR 1.2 version. It is an incomplete module although some routing constructs are proposed for the Teach net to implement flexible navigations, and editing it is not very easy for teachers because of its non-complete structure.

In summary, how to provide a user friendly course sequencing model and apply the SCORM standard is an important issue.

Chapter 3: The Framework of SCORM Compliant Course Construction

As mentioned above, the complicated rules and controls of SCORM activity tree make the course sequences hard to design. Therefore, how to provide a user-friendly authoring tool to easily and fast construct SCROM compliant course becomes an important issue. The learning guidance of a course can be represented as graph which is more easy to understand and create for teachers and authors. Accordingly, if we can provide an authoring tool that teachers and authors can edit the structure of course with sequencing rules by graph representation and then transforms the created structure into SCORM compliant file automatically, the SCORM compliant course can become more and more popular. However, before developing this authoring tool, how to provide an approach to analyze the sequencing rules and to transform the created course into SCORM compliant are our concerns.

Therefore, in this thesis, we apply the High-Level Petri Nets (HLPN), which are a powerful language for system modeling and validation, to modularize the sequencing rules of SN into several sequencing components with corresponding sequencing rules, called **Object-Oriented Activity Tree** (**OOAT**). Thus, by arbitrarily combining these OOATs, we can model a structure of course with different learning guidance. Then, the proposed transformation algorithm can transform the created course into SCROM compliant one described by XML language. The detail will be described in this Chapter.

3.1 The Scheme of OOCM

There are several features for constructing the AT in object-oriented model. Because of the characteristic of hierarchical construction in HLPN, some articles [9][21][22] have proposed several basic structures to construct their specific domains. In this thesis, we also propose sequencing components with modularity and reusability, which represent various sequencing behaviors of SN, to easily manage complex sequences rules of course and to easily reuse. Therefore, we propose the flowchart of OOCM which is used to construct the SCORM compliant course, shown in Figure 6.

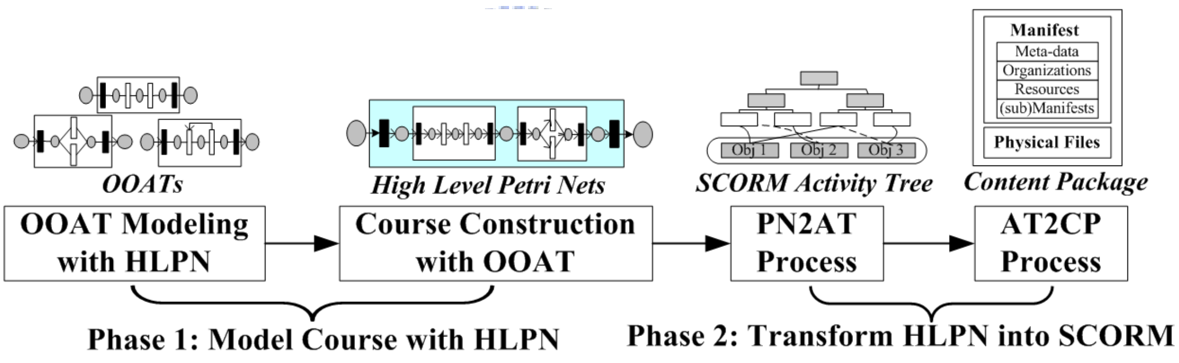


Figure 6: The Flowchart of Object-Oriented Course Modeling (OOCM)**.**

Based upon concept of object-oriented methodology, the OOCM four processes:**1. OOAT Modeling with HLPN**, **2. Course Construction with OOAT**, 3. **PN2AT Process** and **4. AT2CP Process**

(1) **OOAT Modeling with HLPN**: apply HLPN to model five sequencing components as the middleware with corresponding structure of AT and specific basic sequencing behaviors, called *Object-Oriented Activity Tree* (OOAT).

(2) **Course Construction with OOAT**: use these basic sequencing components (OOAT) to model complex structure of course with different learning guidance based upon the HLPN theory.

(3) **PN2AT Process:** transform the modeled course structure into tree-like SCORM-compliant AT with sequencing definition of SN.

(4) **AT2CP Process:** package the transformed AT structure with corresponding physical learning resources and then generate the content packaging course of SCORM.

3.2 Modeling Course with High Level Petri Nets

The concept of OOAT is shown in Figure 7. A course can be composed of a lot of sub-courses, such as Sections or Chapters. Each Section can be viewed as an object which can be created independently. Therefore, these objects can be reused and combined with each other while a teacher authors the course.

In SCORM 2004, a cluster which is a basic sequencing block which can represent an course object. Each cluster has a *Sequencing Definition Model* (*SDM*) which defines a set of elements that can be used to describe and affect various sequencing behaviors. Therefore, we can model a cluster with different sequencing modes as sequencing components which can be used to construct a complex course.

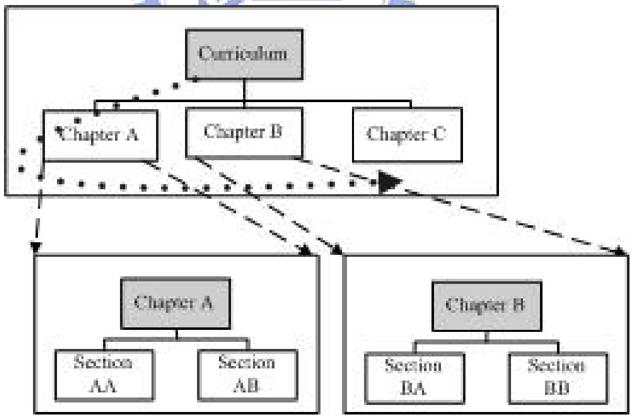


Figure 7: An Object-Oriented Activity Tree.

The Petri Net (PN) is a systematic analysis method which uses graph and mathematics technique to represent and simulate various workflows. Therefore, it is suitable to model and analyze the sequencing behaviors of the activity tree. The HLPN, which is an extension of the traditional PN, supports the *hierarchical* constructing property, which can form a new HLPN by several HLPN, called *subnet*. Thus, we apply HLPN to model clusters with different learning sequencing into several basic sequencing components. The *hierarchical* property of HLPN is suitable to represent the concept of OOAT. The definition of HLPN for course sequencing is described as follows.

Definition 1: The High Level Petri Nets for Course Sequencing is a 6-tuple

Course Sequencing Model CSM = (P, T, Σ, A, G, E) where:

1. **P** = {$p_1, p_2, \ldots, p_m$} is a finite set of places. P includes five types of places: $\mathbf{P_G}$ denotes the global objectives, $\mathbf{P_L}$ denotes the local objectives, $\mathbf{P_M}$ denotes the connector between transitions, $\mathbf{P_R}$ checks whether the transition executes the Rollup process or not and $\mathbf{P_W}$ checks whether the transition sets the global objective ($\mathbf{P_G}$) or not. Besides, in connective places ($\mathbf{P_M}$), we use $\mathbf{P_{in}}$ and $\mathbf{P_{out}}$ to represent the starting place and ending place of a component. P_G and P_L contains tokens recording the tracking status information
2. **T** = {$t_1, t_2, \ldots, t_n$} is a finite set of transitions which disjoint form P ($P \cap T=0$). T includes three types of transitions: $\mathbf{T_A}$ denotes a learning activity or a sub-component (sub-net), $\mathbf{T_M}$ denotes the connector between sequencing components, $\mathbf{T_R}$ will rolls up all learning status of its children, and $\mathbf{T_O}$ will set the global objective ($\mathbf{P_G}$) of an activity according to its local objective ($\mathbf{P_L}$).
3. **Σ** = <C_{TSM}, C_O> is the non-empty finite color sets of tokens. C_{TSM}={C_{Act}, C_{Att}, C_{Obj}}

represents the Tracking Status Model (TSM) which records the information of *Activity Progress Information*, *Attempt Progress Information* and *Objective Progress Information* of learners. These colors have respective values based upon SN. C_O denotes the *ordinary color* which means its corresponding tokens carry no information and is applied to initialize or trigger a learning process.

4. $A \subseteq (P \times T) \cup (T \times P)$ is a finite set of directed arcs. $\overrightarrow{PT}$ is the arc from a place to a transition; $\overrightarrow{TP}$ is the arc from a transition to a place.

5. **G:** is a *guard function*. The firing rule G(t) of a transition ($t \in$ T) is defined as "*if-else*" form in SN. Defining the guard function can generate specific sequencing behaviors. In CSM, we define the following *guard functions*:

   - **G(T_A):** define the sequencing rules of SN and specify whether a learner is ready or not to learn the activity according to her/his previous learning results.
   - **G(T_R):** control the rollup process of an activity based upon the Rollup rules definition of SN.
   - **G(T_O):** set the learning status of the global objective according to local objective of activity (**T_A**). In SN, teachers can define how to read/write a global objective for different course sequencing.

6. **E:** is an *arc expression function*. E(a), $\forall a \in A$, denotes the information of removing how

many and which kinds of token colors should be removed from the input places and added to the output places. In CSM, we define the *expression functions* as shown in Table 2

T**able 2** The Arc Expression Function **E(*a*)** and its Related Token Color

Arc Expression Function	**Token**
E($\overrightarrow{P_GT_A}$), E($\overrightarrow{P_GT_M}$)	**<C_O+C_{TSM}>**
E($\overrightarrow{T_AP_L}$), E($\overrightarrow{T_RP_L}$), ($\overrightarrow{P_LT_R}$), E($\overrightarrow{P_MT_R}$), E($\overrightarrow{T_RP_M}$), E($\overrightarrow{T_OP_G}$), E($\overrightarrow{P_LT_O}$), E($\overrightarrow{P_LT_A}$)	**<C_{TSM}>**
E($\overrightarrow{T_AP_M}$), E($\overrightarrow{P_MT_A}$), E($\overrightarrow{T_AP_G}$), E($\overrightarrow{T_MP_G}$), E($\overrightarrow{P_MT_M}$), E($\overrightarrow{T_MP_M}$), E($\overrightarrow{P_WT_O}$), E($\overrightarrow{P_RT_R}$)	**<C_O>**

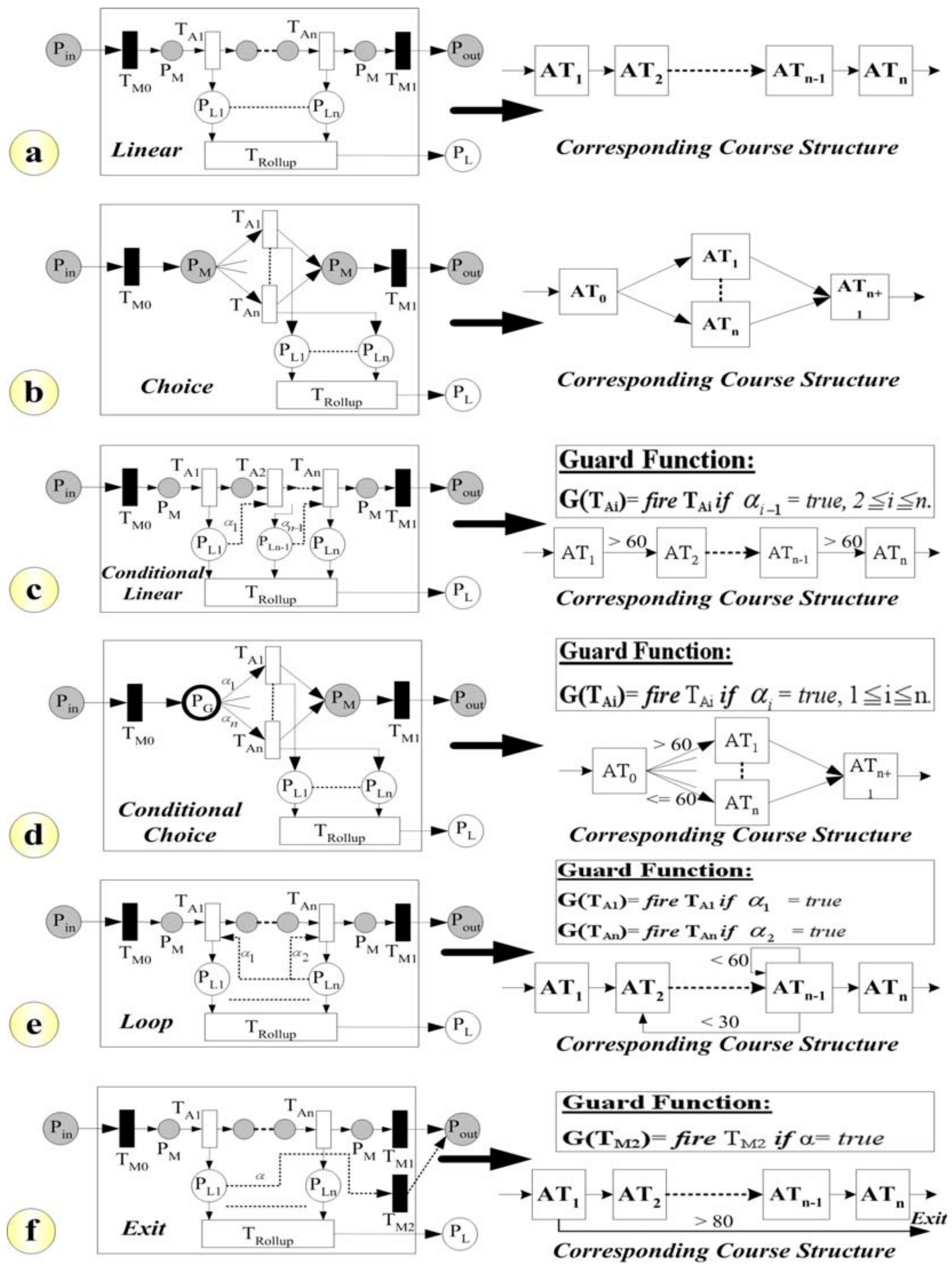
According to the sequencing behaviors in SN specification, we propose five sequencing components, 1. ***Linear***, 2. ***Choice***, 3. ***Condition***, 4. ***Loop***, and 5. ***Exit***, based on HLPNs to model different learning strategies. Figure 8 shows these five sequencing components with its corresponding structure of course and SCORM functions. The SCORM functions include *Sequencing Control Mode (SCM)* which controls the navigation behaviors and *Sequencing Rules* which define the evaluated conditions of course sequencing during learning activity. The default value of control parameters tagged with '*' in *SCM* can be changed by teachers and authors. For example, "*Sequencing Control Forward Only*" has a default value as "*false*", which means that learners can free to navigate the learning content. On the contrary, the navigation is one-way. Moreover, some sequencing components have corresponding *guard function*, which defines the firing rules of CSM in HLPN, to define the triggering criteria of transitions.

In Figure 8, the ***Condition*** component includes ***Conditional Linear*** (8.c) and

Conditional Choice (8.d). The former is a ***Linear*** component with conditional criteria that checks whether an activity is assigned to a student or not is dependent on his/her previous learning progress. For example, a token, $\mathbf{C_{TSM}}$, will be delivered to the local objective (P_{L1}) after learning an activity (T_{A1}). Then, according to the activity's tracking information, the next transition (T_{A2}) may be accessible if the condition α_1 is true. The latter is similar to the ***Choice*** component. According to the previous learning status, an activity (T_A) can be selected by learners if its evaluated criterion (α) is true. In Figure 8.f, the **Exit** component controls the learning process will be terminated or not. For, example, after learners learn the T_{A1}, the token, C_{TSM}, will be delivered to P_{L1}. Then, according to the tracking information of T_{A1}, learners will finish the component if the condition α is true. The related SDM of each OOAT is shown in Table 3.

Table 3. The Related SDM definition of OOAT

OOAT Types	Sequencing Control Mode	Sequencing Rules
Linear	Sequencing Control Flow = *true*	
Choice	Sequencing Control Choice = *true* Sequencing Control Choice Exit = *true*	
Conditional Linear	Sequencing Control Flow=*true* Sequencing Control Choice Exit=*true*	***Postcondition Rule:*** **if** α_i=*true* **then** *continue, 1≦i≦n-1.*
Conditional Choice	Sequencing Control Flow = *true* Sequencing Control Choice = *true* Sequencing Control Choice Exit = *true*	***Precondition Rule:*** $\mathbf{T_{Ai}}$: **Read** OBJ P_G **if** $\alpha_i \neq$*true* **then** *disable, 1≦i≦n.*
Loop	Sequencing Control Flow = *true* Sequencing Control Choice Exit = *true*	***Postcondition Rule:*** $\mathbf{T_{A1}}$: **if** α_1 =*true* **then** *previous* $\mathbf{T_{An}}$: **if** α_2 =*true* **then** *retry*
Exit	Sequencing Control Flow=true Sequencing Control Choice Exit=true	***Postcondition Rule:*** $\mathbf{T_{A1}}$: **if** α =*true* **then** *Exit Parent*

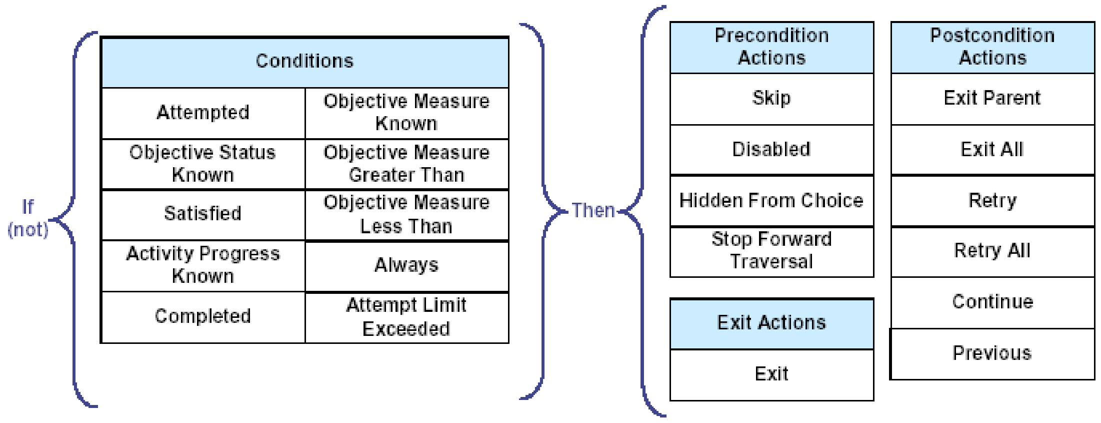


a
Linear
Corresponding Course Structure
b
Choice
Corresponding Course Structure
c
Conditional Linear
Guard Function:
G(T_Ai)= fire T_Ai if α_{i-1} = true, 2≦i≦n.
Corresponding Course Structure
d
Conditional Choice
Guard Function:
G(T_Ai)= fire T_Ai if α_i = true, 1≦i≦n.
Corresponding Course Structure
e
Loop
Guard Function:
G(T_A1)= fire T_A1 if α_1 = true
G(T_An)= fire T_An if α_2 = true
Corresponding Course Structure
f
Exit
Guard Function:
G(T_M2)= fire T_M2 if α= true
Exit
Corresponding Course Structure

Figure 8: Five Sequencing Components

3.3 Sequencing Behaviors Modeling of SN

The SN employs a rule-based sequencing model. A set of zero or more rules can be defined for an activity and must be evaluated at various times in the sequencing and delivery process during run-time. The application of sequencing rules to activity may alter the default sequencing path or may affect the availability of the activity and/or it's sub-activities for delivery. Sequencing rules consist of a set of conditions and a corresponding action or behavior that is performed if the set of condition evaluates to true. The structure of a sequencing rule is shown in Figure 9. We define these sequencing conditions as tokens used to determine whether an activity is accessible or not.



Conditions	
Attempted	Objective Measure Known
Objective Status Known	Objective Measure Greater Than
Satisfied	Objective Measure Less Than
Activity Progress Known	Always
Completed	Attempt Limit Exceeded

Precondition Actions
Skip
Disabled
Hidden From Choice
Stop Forward Traversal

Exit Actions
Exit

Postcondition Actions
Exit Parent
Exit All
Retry
Retry All
Continue
Previous

Figure 9: The Structure of Sequencing Rules

Three kinds of sequencing actions are proposed in SN: *preCondition Action*, *postCondition Action* and *exitCondition Action*. These sequencing actions of SN can be modeled by combining our proposed five sequencing components.

Here, we will describe how to model these sequencing actions of SN based upon these five sequencing components.

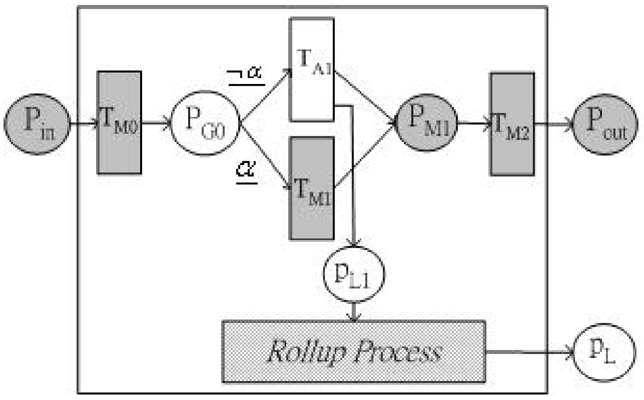
(1) **Precondition Rules:** decide whether an activity will be selected or not for learning. These rules will be executed while an activity will be selected. Its action elements and corresponding OOATs are shown in Table 4.

Table 4. The Action Types and Corresponding OOATs of Precondition in SN.

Action Element	Description	OOATs
Skip	this action will omit an activity to be learned.	*Conditional Choice*
Disabled	this action will block an activity to be learned.	*Conditional Linear*
Stop Forward Traversal	this action will terminate learners to continuously navigate learning activity forward.	*Conditional Linear*
Hidden From Choice	this action will stop the choice of activity	*Conditional Choice* with "*Sequencing Control Choice*" is false.

- ***Skip*****:** this action will omit an activity. It can be modeled by a *Conditional Choice* component. For example, in Figure 10, if the condition α is true, the activity T_{A1} will be skipped and then learners will go to the T_{M1} for learning.
- ***Disabled*****:** this action will block an activity. It can be modeled by *Conditional Linear* component. As shown in Figure 8.c, the activity (T_{A1}) won't be enabled (learned) if the condition α_1 is not true
- ***Stop Forward Traversal*****:** this action will terminate learners to continuously forward navigate learning activity. It can be modeled as ***Disabled*** action.
- ***Hidden From Choice*****:** this action will stop the choice of activity. It can be modeled

by a *Conditional Choice* component with value of “*Sequencing Control Choice*” as false.



SCORM Functions

Sequencing Rules:
Precondition rules:
T_{A1}: Read OBJ P_{G0}
If α=*true* **then** *Skip*

Guard Function:
$G(T_{A1})$=*fire* T_{A1} **if** α=*false*
$G(T_{M1})$=*fire* T_{M1} **if** α=*true*

Figure 10: An Example of modeling Skip Action in SN by Sequencing Components

(2) **Postcondition Rules:** control the sequencing flow according to learning result of learners after learning an activity. These rules will be executed while an activity has been finished. Its action elements and corresponding OOATs are shown in Table 5.

Table 5. The Action Types and Corresponding OOATs of Postcondition in SN

Action Element	Description	OOAT
Exit Parent	this action terminates a activity	*Exit*
Exit All	this action terminates whole activity tree (course)	*Exit*
Retry	this action makes learner to relearn some previous activities if its condition is evaluated as true.	*Loop*
Retry All	this action makes learners to relearn all previous activities if its condition is evaluated as true.	*Loop*
Continue & Previous	this action makes learners to learn next or previous activity respectively.	*Conditional Linear & Loop*

- **Exit Parent:** this action can be modeled by **Exit** component shown in Figure 8.f.
- **Exit All:** this action is similar to the Exit action. However, we have to set the

system commend, *Exit*, to finish whole course if the condition α is true.

- **Retry:** this action can make learner relearn some previous activities under condition α. It can be modeled by *Loop* component as shown in Figure 8.e. For example, in Figure 11, the token <C_{TSM}> of T_{A2} is delivered to P_{L2}. Then, T_{A2} will be relearned under condition α according to its learning status of local objective (P_{L2}).
- **Retry All:** this action will make learners to relearn all previous activities in this course under condition α. As shown in Figure 11, the token <C_{TSM}> will be delivered to P_{L2} after learning T_{A2}. Then, learners will relearn all activiries under condition α according to its learning status of local objective (P_{L2}).
- **Continue** and **Previous:** These actions can be modeled by *Conditional Linear* component and *Loop* component, shown in Figure 8.

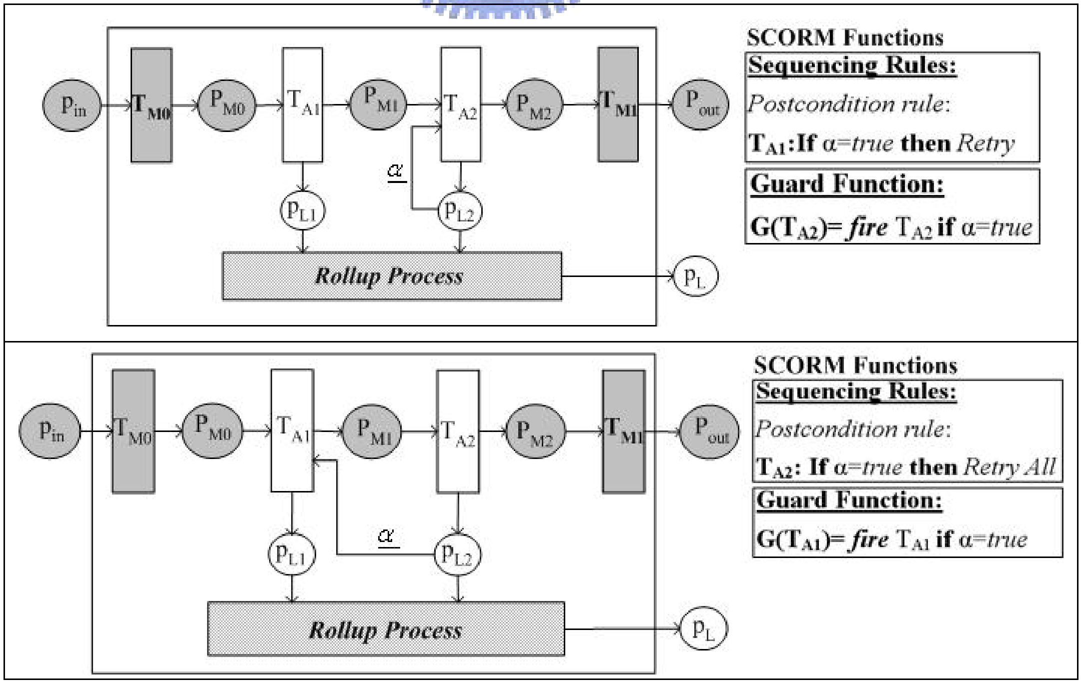


Figure 11: The modeling of *Retry* and *Retry All* Actions in SN by Sequencing Components

(3) Exit Rule: This rule will be executed after a descendant activity has been finished. It is controlled by a SCORM complaint learning system. Thus, we set the system commend, Exit, to finish the whole course.

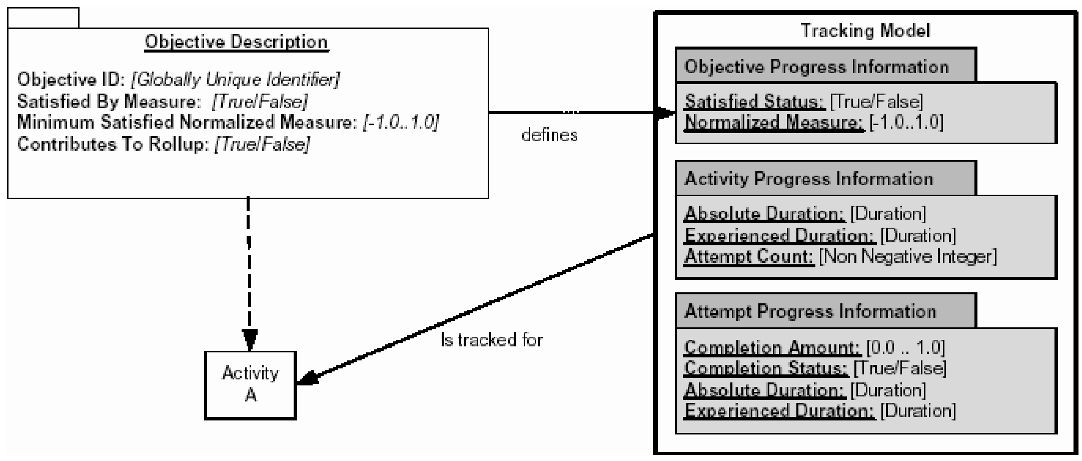
3.4 Objective Modeling

In SN, each activity has unlimited number of associates learning objectives. Two kinds of learning objectives are proposed in SN: local objectives and global objectives. The local objective defines how to evaluate an activity's objective progress; the global one can be shared between activities for the more complex instructional designs. Each learning objective contains a set of information that can be defined. As shown in Table 2, the SN defines the following set of elements to describe each learning objective.

Table 6: Description of Objective

Name	Description
Objective ID	A unique identifier for the objective associated with the activity
Objective Satisfied By Measure	Whether or not the objective satisfaction is determined by a measure (score)
Objective Minimum Satisfied Normalized Measure	Indicate the minimum satisfaction measure of the objective
Objective Contributes to Rollup	Indicates that the *Objective Satisfied Status* and *Objective Normalized Measure* for the objective are used during rollup.

For sequencing purposes, each learning objective of an activity will have a set of tracking status information so that the learning progress of a student can be tracked; then, the conditional sequencing decisions are enabled. The relationship between *Objective Description* and the *Objective Progress Information* of TSM is shown in Figure 12. An activity will only be able to access *Objective Progress Information* of its local objectives. In default, no objective information is shared between activities. It means that no global objective is defined until teachers or authors specify it.



Objective Description
Objective ID: [Globally Unique Identifier]
Satisfied By Measure: [True/False]
Minimum Satisfied Normalized Measure: [-1.0..1.0]
Contributes To Rollup: [True/False]
defines
Tracking Model
Objective Progress Information
Satisfied Status: [True/False]
Normalized Measure: [-1.0..1.0]
Activity Progress Information
Absolute Duration: [Duration]
Experienced Duration: [Duration]
Attempt Count: [Non Negative Integer]
Attempt Progress Information
Completion Amount: [0.0 .. 1.0]
Completion Status: [True/False]
Absolute Duration: [Duration]
Experienced Duration: [Duration]
Is tracked for
Activity A

Figure 12: The Relationship between Objective Description and Objective Progress Information

In CSM, each transition (activity) has one local objective (P_L) and one global objective (P_G). Figure 13 shows the objective model we proposed in this thesis. Figure 13.a is the default model where all values of local objective description are default and no global objectives are defined. (Because P_W does not has ordinary token < C_O>, T_O won't be fired). The token <C_{TSM}>, which contains the learning information in T_{A1}, is delivered to its local objective P_{L1}. Figure 13.b shows user defined model that teachers can specify the objective description of local and global objectives by setting place P_W.

Place P_1 is a connective place in the default model; however, it will be changed to a global objective if P_W is set by giving an ordinary token <C_O>. Then, T_O will evaluate the status of the global objective P_{G1} according to P_{L1} and the global objective description.

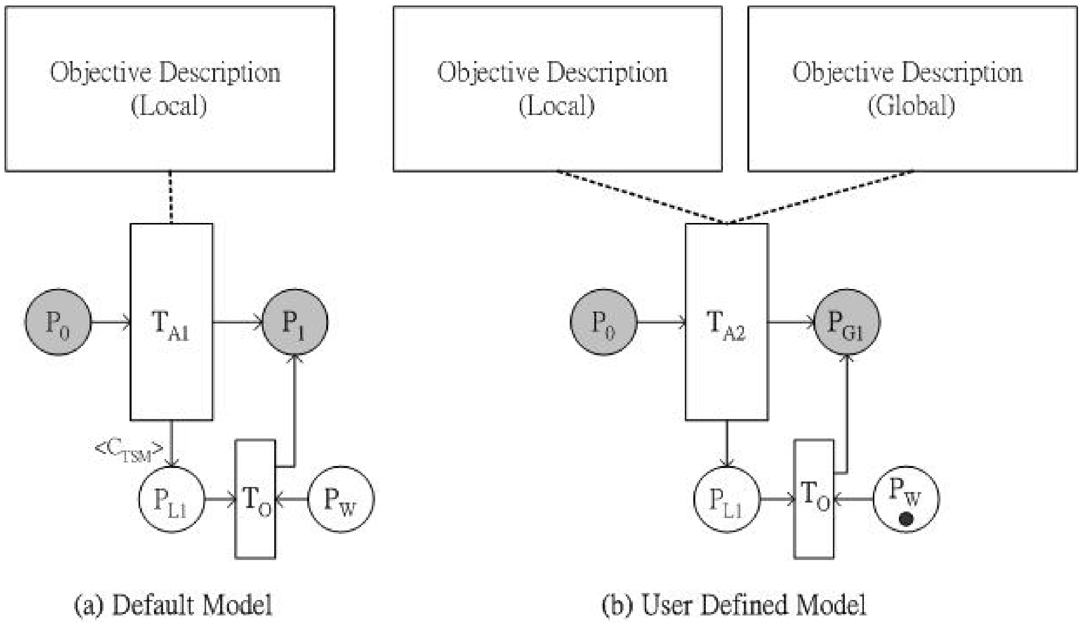


Figure 13: The Objective Model of a Transition (Activity)

Figure 14 shows the objective references. Teacher defines the local and global objective descriptions. The transition T_{A1} is satisfied if the score students get exceeds 0.6. In this case, the score the student obtains is 0.75; therefore, T_{A1} is satisfied. According to the objective descriptions, the guard function $G(T_O)$ is defined as shown in Figure 14. The token $<C_{TSM}>$ carries the status information of T_{A1} and the ordinary token $<C_O>$ are delivered to T_O for setting the global objective P_{G1}. In this example, the objective measure of P_{L1} exceeds 0.6; therefore, the learning status of global objective P_{G1} is satisfied.

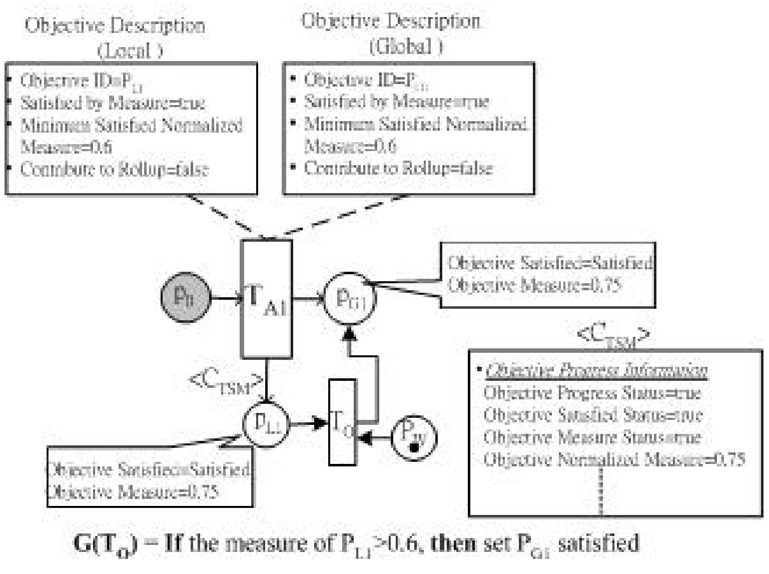

Objective Description (Local)
Objective ID=P_{L1}
Satisfied by Measure=true
Minimum Satisfied Normalized Measure=0.6
Contribute to Rollup=false
Objective Description (Global)
Objective ID=P_{G1}
Satisfied by Measure=true
Minimum Satisfied Normalized Measure=0.6
Contribute to Rollup=false
P_{I1}
T_{A1}
P_{G1}
Objective Satisfied=Satisfied
Objective Measure=0.75
<C_{TSM}>
Objective Progress Information
Objective Progress Status=true
Objective Satisfied Status=true
Objective Measure Status=true
Objective Normalized Measure=0.75
<C_{TSM}>
P_{L1}
T_O
P_W
Objective Satisfied=Satisfied
Objective Measure=0.75
$G(T_O)$ = If the measure of P_{L1}>0.6, then set P_{G1} satisfied

Figure 14: The Process of Objective Reference

3.5 Rollup Rule Modeling

Each activity in the Activity Tree may track a learner's interactions with the activity. The set of tracking information associated with each activity is defined by the Tracking Status Model (TSM). Leaf activities may provide learning resources that are directly experienced by the learner. However, cluster activities cannot provide learning resources and do not directly set their status information. The status of a cluster activity is based on the status of its children; the process of evaluating a cluster's status information is called 'rollup'. The evaluation of rollup rules may alter the default sequencing path or may affect the availability of the activity and/or it's sub-activities for delivery.

Rollup rules consist of a set of child activity conditions and a corresponding action or behavior that is performed if the set of conditions are evaluated as true. Figure 15 shows the structure of rollup rules which are optional and teachers don't need to specify them if necessary.

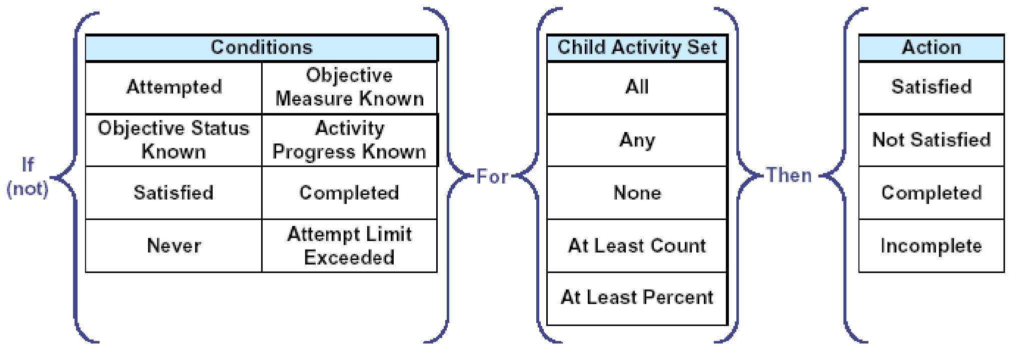


Figure 15: The Structure of Rollup Rules

In CSM, we record status information of each activity in a component. All the status information of transitions (activities) in one component will be aggregated to evaluate the learning results of learners. Based upon the property of hierarchical construction of HLPNs, we can easily evaluate the whole status information of a course by recursively rolling all of the sub-course up. As shown in Figure 16, two models are proposed for teachers to specify the rollup process. The first one is *default model* shown in Figure 16.a. In default model, because no P_R is set as a token $<C_O>$, the rollup process isn't executed (T_R won't be fired). The second one is *user defined model* that teachers can decide which activities should be rolled up and define the rollup rule (the guard function $G(T_R)$) which is applied to evaluate the learning progress of a student in this component. For example, in Figure 16.b, because the P_R of T_{A2} and P_R of T_{A3} are marked by an ordinary token $<C_O>$, T_{A2} and T_{A3} will execute the rollup process. Moreover, according to definition of Rollup Rules, the learning status of this component will be set as satisfied if at least two activities (transitions) within it are satisfied.

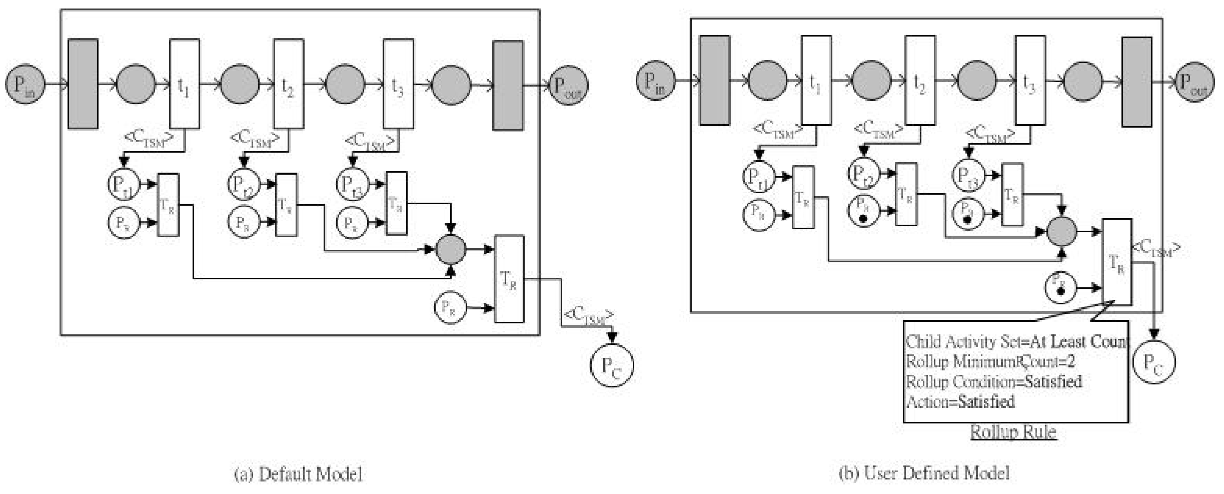


Figure 16: The Rollup Model of a Component

3.6 The Process of Sequencing Components Combination

Up to now, we have propose how to model the sequencing behaviors based on SN, rollup rule specification and objective description by proposing five sequencing components. In this section, we will describe how to construct a complete course by combining sequencing components. Based upon the characteristic of hierarchical construction in HLPN, teachers can edit the course in both top-down and bottom-up manner. Besides, the teacher can define rules for rollup process and objectives for more complex sequencing behaviors if necessary. Therefore, we propose the combination principles of sequencing component for creating complex course. We describe these principles as follows:

(1) **Principle 1:** The two connective places P_{in} and P_{out} are the starting point and ending point for a component respectively. They can be represented as the mediums between components while combining components.

(2) **Principle 2:** P_{in} of one component is connected to another component's P_{out} if the learning procedure is linear; then, we use another connective place to replace the two as shown in Figure 17.a.

(3) **Principle 3:** If the learning procedure is in the branching mode (such as Choice, Condition), P_{in} of one component is connected to other components' P_{in}s and another connective place are used to replace the connected P_{in}s. Then, their P_{out}s are connected to T_M which will be linked to a connective place. Figure 17.b shows this principle.

(4) **Principle 4:** When combining a Condition component with other ones, the replaced connective place must be changed to a global objective which is the same as the Condition component. This principle is shown in Figure 17.c.

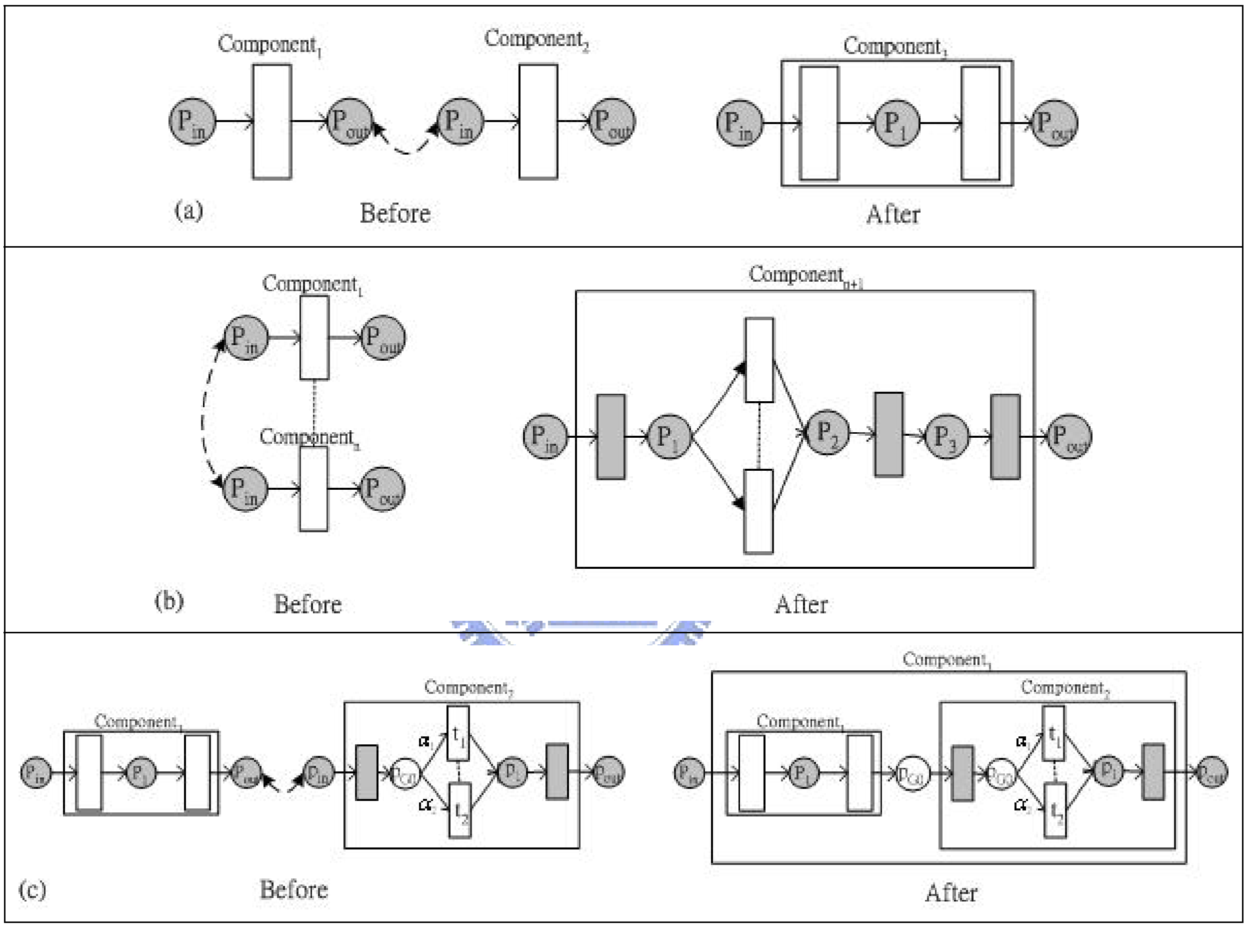


Figure 17: Combination scheme of sequencing components.

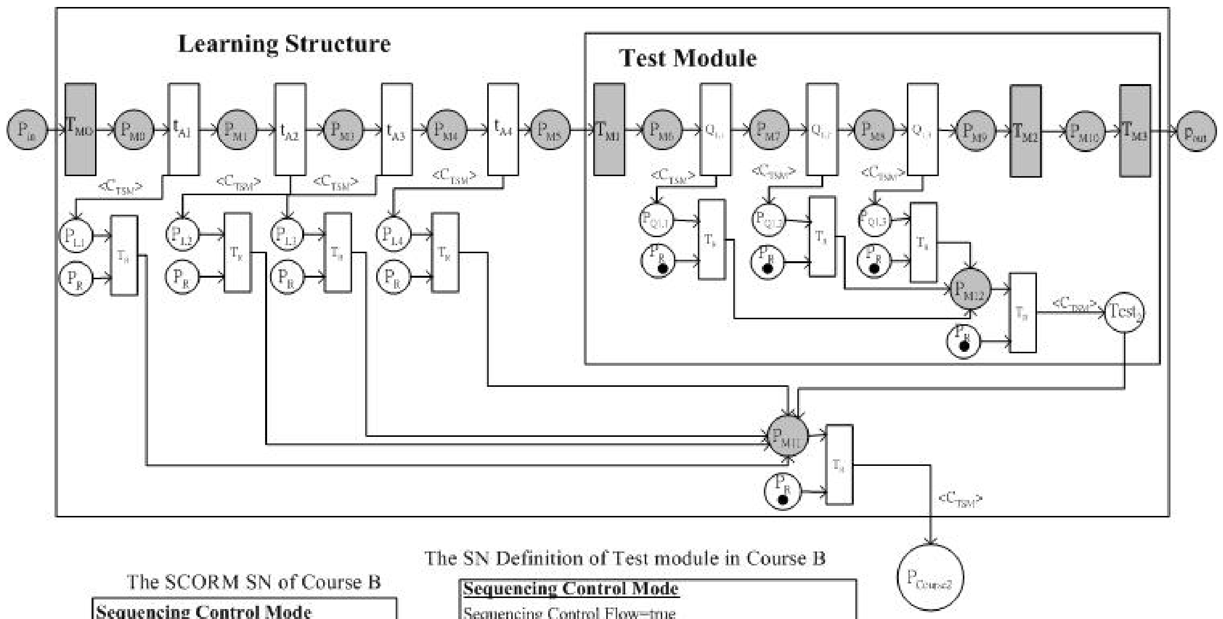
3.7 An Example of Creating a Complex Course

We use the course of “PhotoShop” which SCORM releases to describe the process of constructing a complex course with the corresponding definitions of SN based upon five sequencing components. This course contains four parts with different teaching strategies: **Course A**, **Course B**, **Course C** and **Course D**. For each component, teachers can get the directory information of learning resources related to the activities by searching the learning objective repository (LOR). The process of adding learning resources to the course model will be expressed in the Chapter 4

(1) **Course A:** It is the introduction of “**PhotoShop**” course. This part contains only one activity and no sequencing behaviors.

(2) **Course B:** It is composed of five parts, T_{A1}, T_{A2}, T_{A3}, T_{A4}, and test model, which are organized as a linear structure, shown in Figure 18. Thus, we can apply A *Linear* component to model Course B. The HLPN model of Course B and the corresponding definition of SN are also shown in Figure 18. Moreover, we use the score of test module to evaluate a student’s learning result and execute the rollup process. According to the objective description of test model, a student’s learning status will be set as satisfied if her/his score exceeds 0.6. Then, rollup process of test module will be executed and the learning progress of Course B will be set as satisfied if all of its child activities’ learning statuses are satisfied. Because only the test module can execute the rollup process, Course

B is satisfied as long as the test module is satisfied.



The SCORM SN of Course B

Sequencing Control Mode
Sequencing Control Flow=true
Sequencing Control Forward Only=false
Sequencing Control Choice=false
Sequencing Control Choice Exit=true

Rollup Rules
If All Satisfied then Satisfied

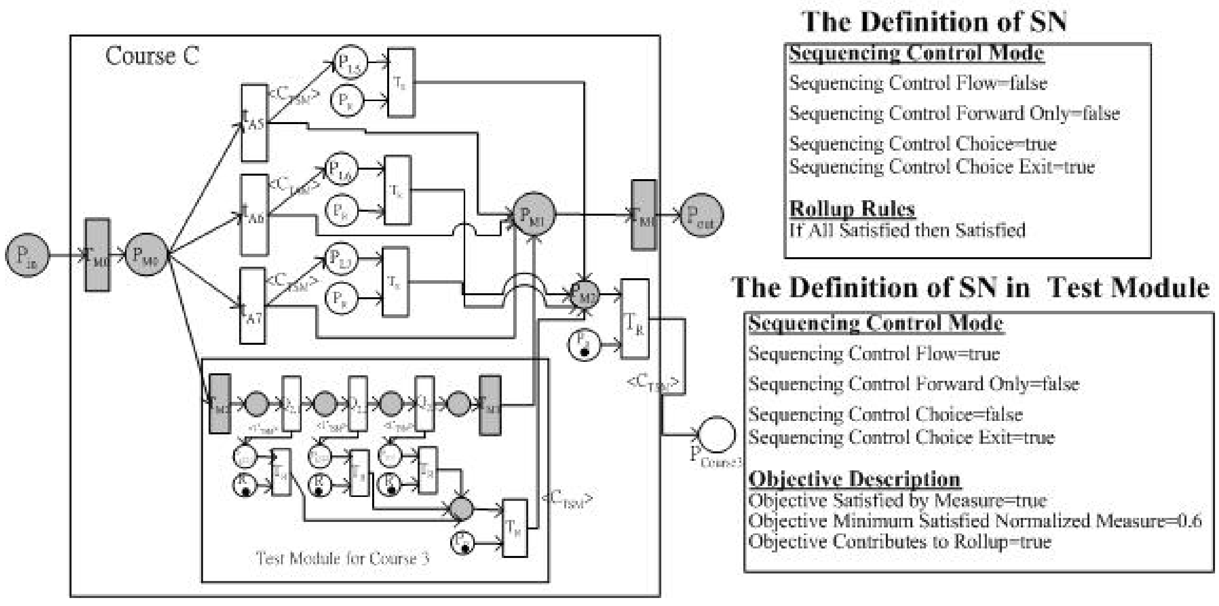
The SN Definition of Test module in Course B

Sequencing Control Mode
Sequencing Control Flow=true
Sequencing Control Forward Only=false
Sequencing Control Choice=false
Sequencing Control Choice Exit=true

Objective Description
Objective Satisfied by Measure=true
Objective Minimum Satisfied Normalized Measure=0.6
Objective Contributes to Rollup=true

Figure 18: The HLPN Model and it related definition of SN in Course B

(3) **Course C:** There are four parts includes t_{A5}, t_{A6}, t_{A7} and a test model. We can use a *Choice* component to model Course **C** so that students can any select activity arbitrarily. Figure 19 shows the HLPN model of Course C and its corresponding definition of SN. The Rollup process of Course C is similar to Course B.



The Definition of SN

Sequencing Control Mode
Sequencing Control Flow=false
Sequencing Control Forward Only=false
Sequencing Control Choice=true
Sequencing Control Choice Exit=true

Rollup Rules
If All Satisfied then Satisfied

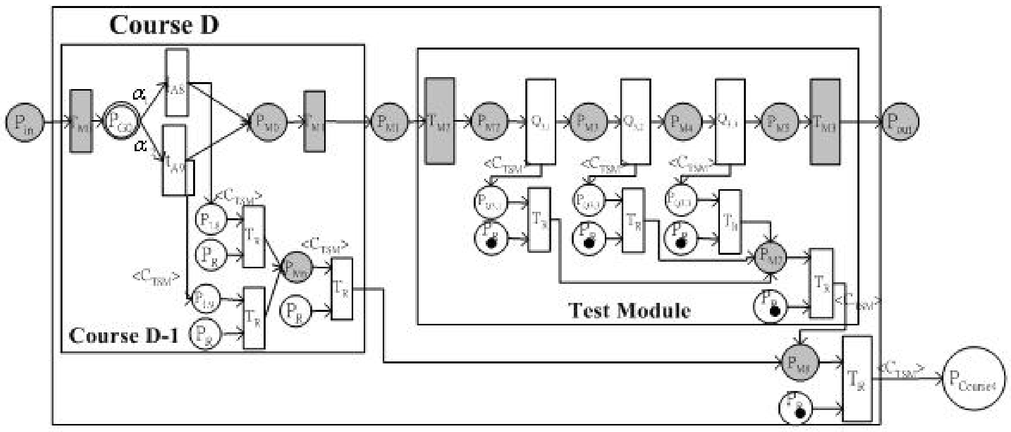
The Definition of SN in Test Module

Sequencing Control Mode
Sequencing Control Flow=true
Sequencing Control Forward Only=false
Sequencing Control Choice=false
Sequencing Control Choice Exit=true

Objective Description
Objective Satisfied by Measure=true
Objective Minimum Satisfied Normalized Measure=0.6
Objective Contributes to Rollup=true

Figure 19: The HLPN Model and related definition of SN in Course C

(4) **Course D:** it can be divided into two parts: Course D-1 and a test model, which can be arranged in a linear manner as shown in Figure 20. Course D-1 is composed of a *Conditional Choice* component with two chapters: T_{A8} and T_{A9}. According to sequencing rules in Course D-1, its availability depends on learning result of Course C.



The Definition of SN in Course D

Sequencing Control Mode
Sequencing Control Flow=true
Sequencing Control Forward Only=false
Sequencing Control Choice=false
Sequencing Control Choice Exit=true

Rollup Rules
If All Satisfied then Satisfied

The Definition of SN in Course D-1

Sequencing Control Mode
Sequencing Control Flow=true
Sequencing Control Forward Only=false
Sequencing Control Choice=true
Sequencing Control Choice Exit=true

Rollup Rules
Rollup Objective Satisfied=false

Sequencing Rules:
Precondition Rule
t_{A8}: Read OBJ P_{G0}
Minimum Measure=0.9
If Objective Measure Less Than then Disabled
t_{A9}: Read OBJ P_{G0}
Minimum Measure=0.6
If Objective Measure Less Than then Disabled

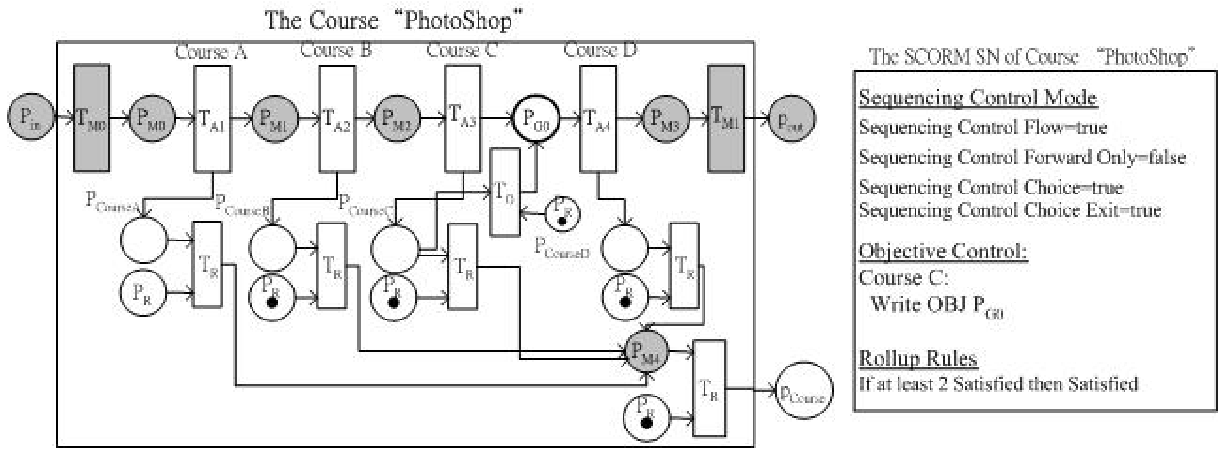
The Definition of SN in Test module

Sequencing Control Mode
Sequencing Control Flow=true
Sequencing Control Forward Only=true
Sequencing Control Choice=false
Sequencing Control Choice Exit=true

Objective Description
Objective Satisfied by Measure=true
Objective Minimum Satisfied Normalized Measure=0.6
Objective Contributes to Rollup=true

Figure 20: The HLPN Model and related definition of SN in Course D

Finally, we combine four Courses HLPN components in a *Choice* manner (Choice component). During the process of combination, a global objective $\mathbf{P_{G0}}$ is used to connect Course **C** and Course **D**. According to Step 4, Course **D-1** is composed of a Conditional Choice component. Therefore, Course C will set the global objective $\mathbf{P_{G0}}$ which can be read by t_{A8} and t_{A9} in the Course **D-1** to decide a student's sequencing behavior. As shown in Figure 21, all except Course **A** execute the rollup process. According to the rollup rule specified for the course "**PhotoShop**", it will be set as satisfied if at least two sub courses among Course **B**, Course **C** and Course **D** are satisfied.



Sequencing Control Mode
Sequencing Control Flow=true
Sequencing Control Forward Only=false
Sequencing Control Choice=true
Sequencing Control Choice Exit=true

Objective Control:
Course C:
Write OBJ P_{G0}

Rollup Rules
If at least 2 Satisfied then Satisfied

Figure 21: The Final HLPNs Model of Course "PhotoShop"

Chapter 4: Activity Tree Transformation Process

Course sequencing is an important component in an intelligent tutoring system. In Chapter 3, we have described how to model HLPN model of a complex course based upon our proposed five sequencing components.

Then, how to transform the HLPN model into SCORM compliant course is an important issue. Therefore, in this thesis, we propose two algorithms to transform from the HLPN course model into a SCORM complaint course which can be executed on SCORM compliant LMS. The process of transformation can be divided to two parts:

(1) The Process of Transforming HLPN into Activity Tree:

Each sequencing component can be represented as a cluster of AT in SCORM. Therefore, we propose an algorithm, called **PN2AT**, to transform each component into a course module as a cluster with related sequencing rules of SN. An AT can be constructed by combining these clusters transformed from HLPNs.

(2) The Process of Packaging Activity Tree into SCORM Course

Then, we propose an algorithm, called **AT2CP**, to package the Activity Tree and related physical learning resources into a SCORM compliant course file described by XML language.

4.1 PN2AT algorithm

The main purpose of PN2AT algorithm is to transform HLPN course model into a SCORM complaint Activity Tree.

In Section 3.2 we have proposed five primitive components used to construct a course module. Each component represents a cluster in SCORM; therefore, components will be transformed and the corresponding clusters are generated during the transformation process.

Because the definition of SN is embedded in sequencing component, teachers can design the sequencing behaviors by editing components instead of authoring complex XML codes.

In PN2AT algorithm, recursively transforms each non-terminated transition into a cluster with associated sequencing definitions in SDM and integrates them to construct the structure of AT. The transformed AT is described by XML codes about the content structure without information of related learning resources. These related physical teaching materials (LOs) will be pooled to the structure during the process of *Content Aggregation* in the algorithm AT2CP that will be described in Section 4.2. The definition of PN2AT algorithm is shown as follows:

Algorithm: PN2AT Algorithm

Definition of Symbols:

AT_{cur}: represent a non-finished AT during transforming process

AT_G : represent the final AT.

C_{par} : represent a composite component includes other HLPN model.

Combine*(C, AT_{cur}):** it is used to combine a cluster ***C with its parent activity in the current AT.

ID={id_1,id_2,…,id_n}**:** it is a set of identifiers. For every leaf activity A_i is assigned a unique identifier id_i to denote the reference of its related learning resources. Initially, ID={ϕ}

Input: The HLPN course model

Output: **AT_G**

```
Step 1.   Input the component COM of HLPN model
Step 2.   Create a cluster C for COM
          $\forall$ $T_i$ (transition) $\in$ COM, add a child activity to C
Step 3.   Generate the corresponding XML codes of Sequencing Definition Model for
          C according to the specification of COM
      Stept 3.1.   Generate Sequencing Rules specification if necessary
      Stept 3.2.   Generate Rollup Rule specification if necessary
      Stept 3.3.   Generate Objective specification if necessary
Step 4.   Create an $AT_G$ by combining cluster C with $AT_{cur}$
      Stept 4.1.   If $AT_{cur} \neq \phi$, $AT_G$=Combine(C, $AT_{cur}$), $AT_{cur}$= $AT_G$
      Stept 4.2.   Otherwise, $AT_G$=C, $AT_{cur}$= $AT_G$
Step 5.   Recursively create clusters in a manner of Depth First Search.
          $\forall$ $T_i \in C$
      Stept 5.1.   If $T_i$ contains another HLPN model,
                    Then $C_{par}$= $T_i$, COM= $T_i$, go to Step 2
      Stept 5.2.   Otherwise, give a reference identifier $id_i$ to $T_i$ and insert it to ID
Stept 6.  Output $AT_G$
```

We take an example to describe the process of PN2AT. As shown in Figure 23, the component T_{A1} is the highest level of component which can be decomposed as a hierarchical structure.In every level, a non-terminated transition, e.g., T_{A1}, will be represented as a root-node (AA) and included sub-transitions (TA'_1 and TA'_2) will be represented as the child nodes (AA and AB), which form a tree-like structure as a cluster with associated sequencing definition of SDM in AT. Then, we can recursively transform all non-terminated transitions by the same process.

4.2 AT2CP Algorithm

An Activity Tree which is transformed by the algorithm PN2AT can be viewed as a map (content structure) that can be used to aggregate learning resources (LOs) into a unit of course. Once AT is created, there is a need to make the content available to learners, authoring tools, repositories or Learning Management Systems (LMSs). The IMS Content Packaging Specification [1][2] was designed not only to provide a standard way to structure and exchange learning content between different systems, but also a mechanism for describing the structure and the navigation behaviors of a collection of content. Therefore, we propose an algorithm AT2CP which is used to package the content structure (AT) and related physical files into SCORM compliant course.

Two major components in SCORM/IMS Content Package Scheme are as follows:

(1.) A special XML document which are used to describe the content structure and associated resources of the package called the *manifest file* (*imsmanifest.xml*). A manifest is required to be present at the root of the content package.

(2.) The physical files form the content package, such as media, graph, tex, etc.

Figure 20 illustrates components of an IMS content package.

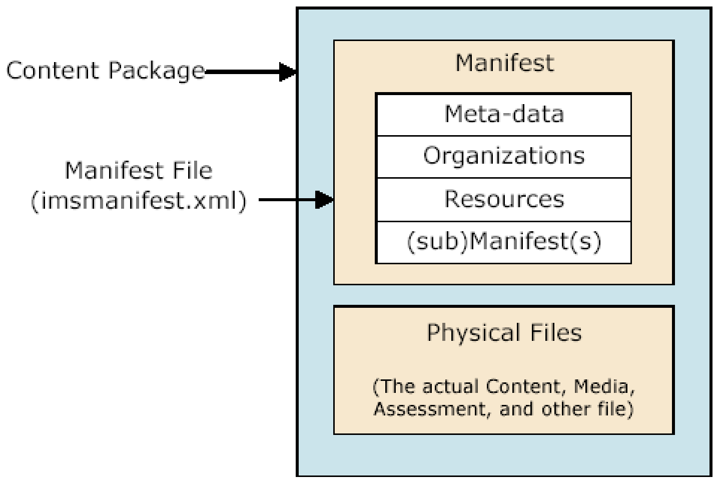


Figure 22: Conceptual Diagram of Content Package in SCORM/IMS

Thus, we propose an algorithm AT2CP that can pack the content structure and the physical files into a content package file. According to the algorithm PN2AT, we can obtain the XML codes describing the content structure of the AT. These XML codes are parts of the information in the *manifest* file; however, no information about physical files is mentioned. By applying AT2CP, the *manifest* file for the AT will be generated. The reference of teaching material for each leaf activity will be included in the file and the related physical files are aggregated into SCORM compliant course. In the following algorithm, the input AT is represented as a non-finished manifest file which describes the content structure and definition of SN.

```
Algorithm: AT2CP Algorithm
Definition of Symbols:
PF: represent a file which contains related physical files of AT. Initially, PF={$\phi$}
CP: represent the content package
Input: Activity Tree (AT)
Output: Content Package (CP)

Step 1.  For each leaf activity with reference identifier $\mathbf{id_i}$, add the information of its
         learning resources into manifest.xml file
    Stept 1.1.  Create a pair of XML tags"<resource></resource>" for this leaf
                activity, e.g., <resource identifierref="$id_i$"></resource>
    Stept 1.2.  Search the LOR to find the suitable learning material and add a
                pair of XML tags <file></file> recording the references of related physical
                files, e.g., <file identifierref="the reference of a physical file"> </file>
Step 2.  Content Aggregation Process
    Stept 2.1.  Add these physical files to PF
Stept 3.  Package the manifest file and PF into the CP;
Stept 4.  output the CP
```

Figure 23 shows an example of AT2CP. The AT of Course with structure and definition of SN is constructed by the algorithm PN2AT. For each leaf activity in the AT, teachers will create a pair of XML tags of "*resource*": *<resource> </resource>*. The information of physical files is represented by a pair of tags of "file": *<file> </file>*. Then, aggregate all related physical files of each leaf activity and package the *manifest* file and all physical files into SCORM content package file.

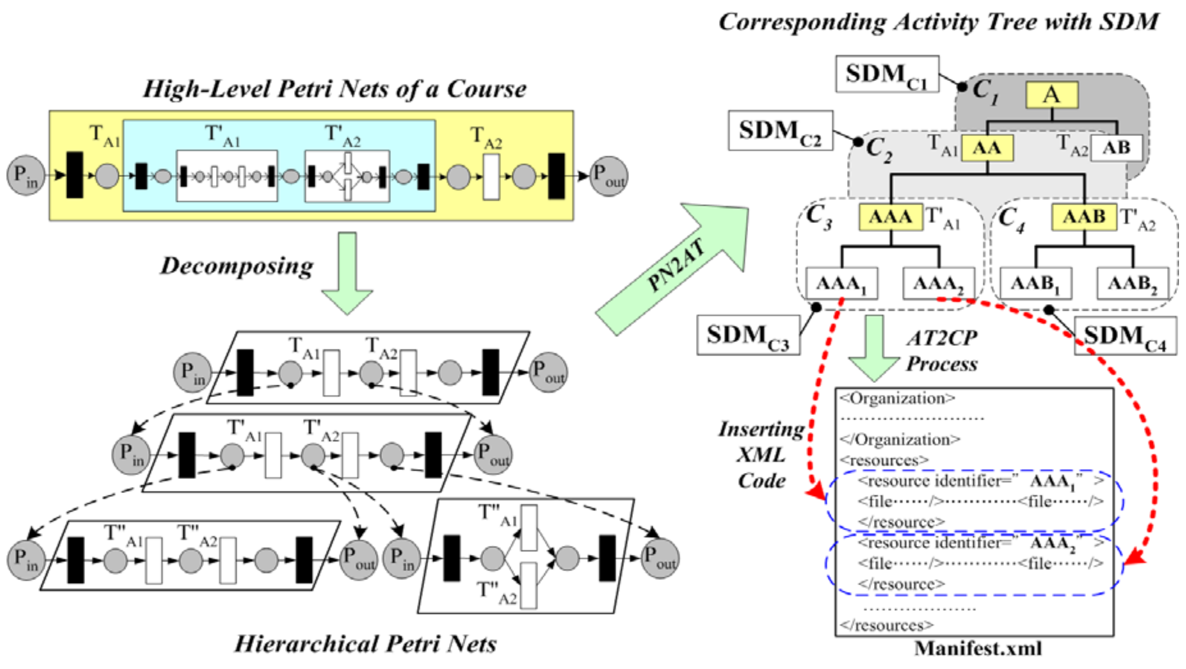


Figure 23: The Activity Tree Transformation Process

4.3 An Example of PN2AT

In this section, we take Figure 24 as an example to express the process of AT transformation. Firstly, the course model based on HLPN is inputted to the algorithm PN2AT. Figure 23 shows the process of transformation from the HLPN into an AT. According to PN2AT, the outer component which is composed of four sub courses- Course A, Course B, Course C and Course D. In a manner of Depth First Search, clusters of the Course B, Test B, Course C, Test C, Course D, Course D-1 and Test D are created step by step as shown in Figure 24.b. Because the Course D-1 is composed of a *Conditional* Choice component, t_{A8} and t_{A9} will read the global objective P_{G0} which is set by Course C to decide which one is available for students to learn. Finally, the complete AT of the "PhotoShop" is completed in Figure 24.c.

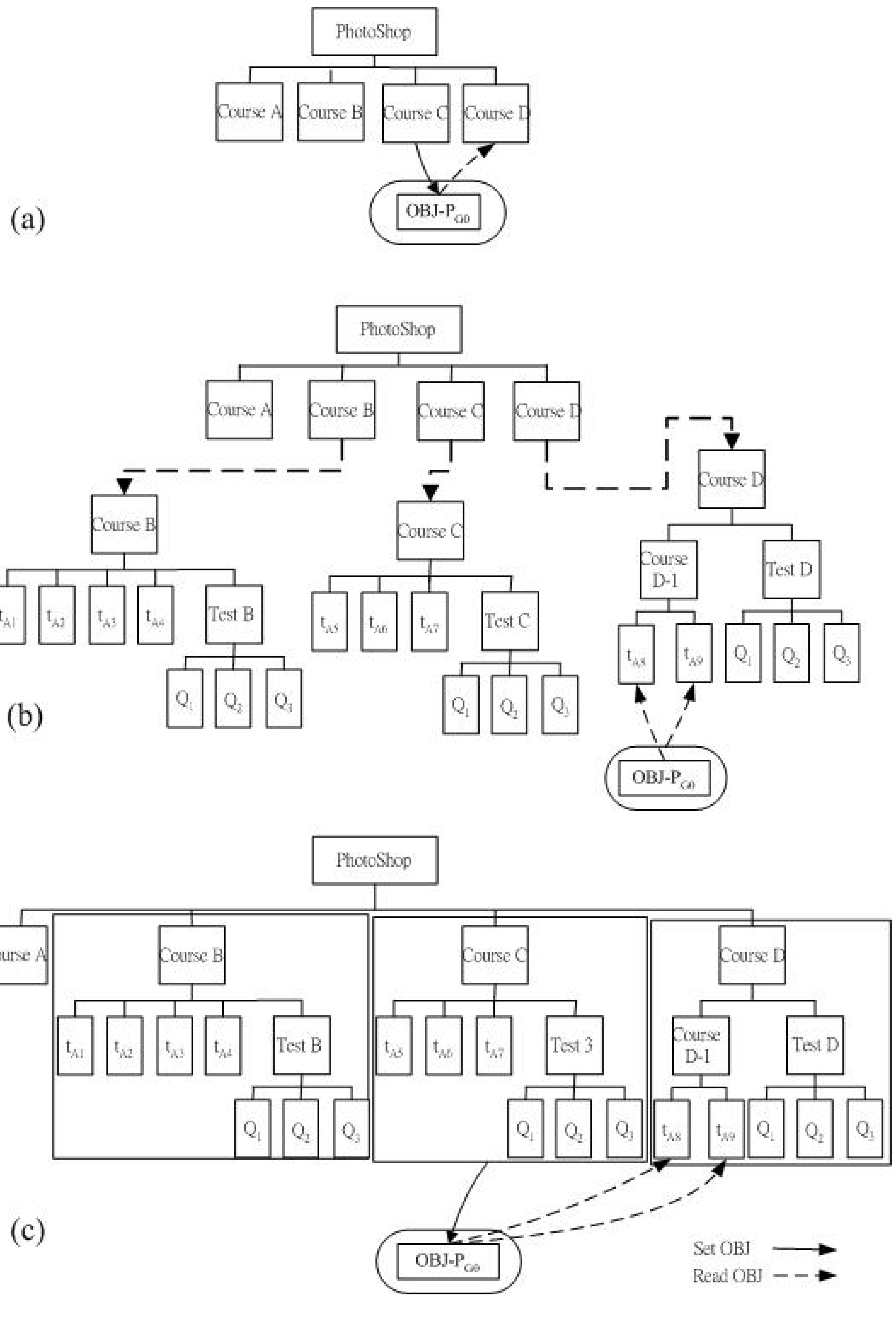


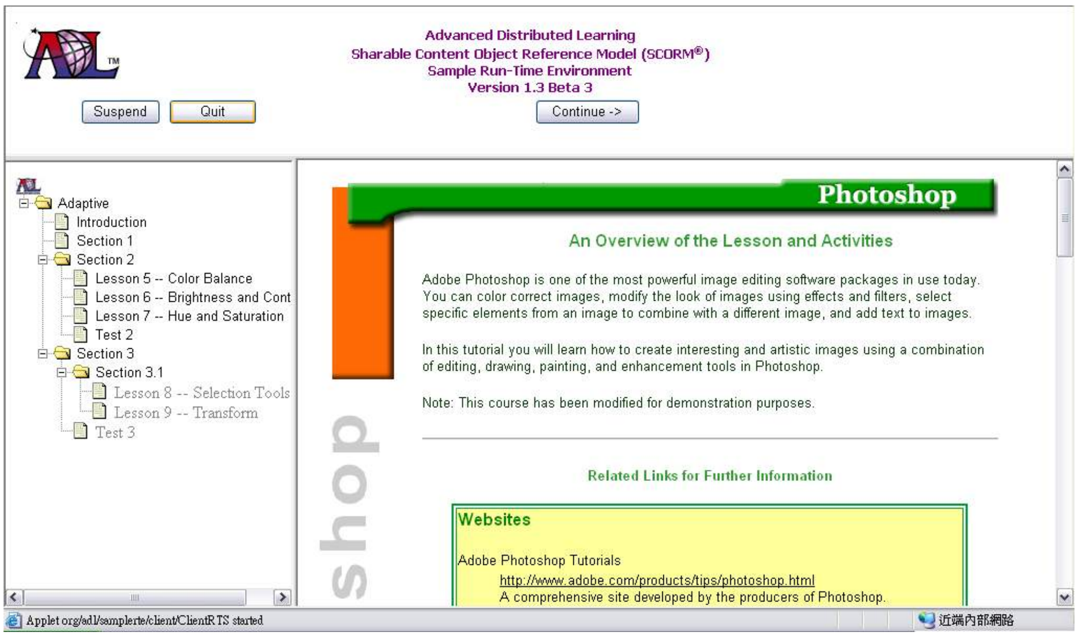
Figure 24: An Example of PN2AT Algorithm in Course "Photoshop"

Chapter 5: Implementation

In this section, we take the course of **PhotoShop** as an example to show the implementation running on SCORM RTE. The course is constructed by the components based on HLPN in Figure 21 and is transformed into a SCORM complaint AT in Figure 25. It contains four sub courses: Course A, Course B, Course C and Course D. Each one is composed of different components for different instructional strategies. We will discuss the sequencing behaviors a student may perform on SCORM RTE.

(1) Course A:

Course A, an introduction of the general conception of PhotoShop, doesn't contain any other sub courses. As shown in Figure 26, the left area is a tree-like structure which represents the content structure (AT) of this course. Because these four sub courses are combined in a Choice component, students can choose any section in all except Course D (it is labeled as Chapter 3 in the tree-like structure) in arbitrary order. The detail of Course D will be expressed later. In the top area, a student can control her/his learning procedure including the direction of learning path (buttons of "Previous" or "Continue") and the action of suspending or terminating the learning (buttons of "Suspend" and "Quit"). The right area shows the teaching material which students learn.



Advanced Distributed Learning
Sharable Content Object Reference Model (SCORM®)
Sample Run-Time Environment
Version 1.3 Beta 3

An Overview of the Lesson and Activities

Adobe Photoshop is one of the most powerful image editing software packages in use today. You can color correct images, modify the look of images using effects and filters, select specific elements from an image to combine with a different image, and add text to images.

In this tutorial you will learn how to create interesting and artistic images using a combination of editing, drawing, painting, and enhancement tools in Photoshop.

Note: This course has been modified for demonstration purposes.

Related Links for Further Information

Websites

Adobe Photoshop Tutorials
http://www.adobe.com/products/tips/photoshop.html
A comprehensive site developed by the producers of Photoshop.

Applet org/adl/samplerte/client/ClientRTS started

近端內部網路

Figure 25: The diagram of Course A (Introduction)

(2) Course B:

According to Figure 18, the Course B is composed of a Linear Component. It contains four chapters and one test model. Students learn these chapters step by step and finally a test model "Test B" is assigned to them to evaluate the learning progress. Because Course B is composed of a *Linear* component, students must follow the predefined learning path instead of choosing activities arbitrarily. Besides, the test models for each sub courses consist of a *Linear* component in our implementation. Figure 27 shows the learning procedure of Course B.

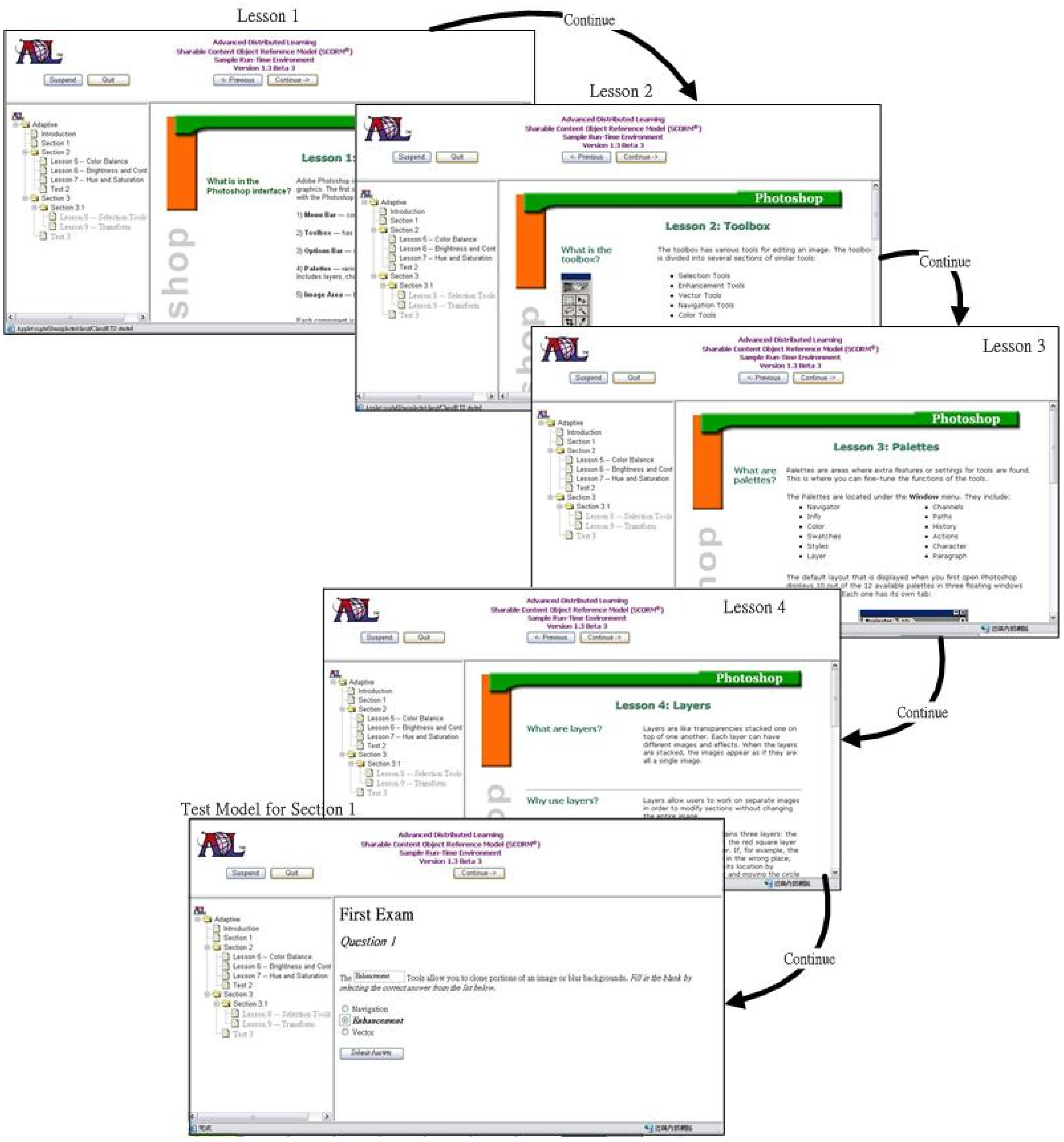
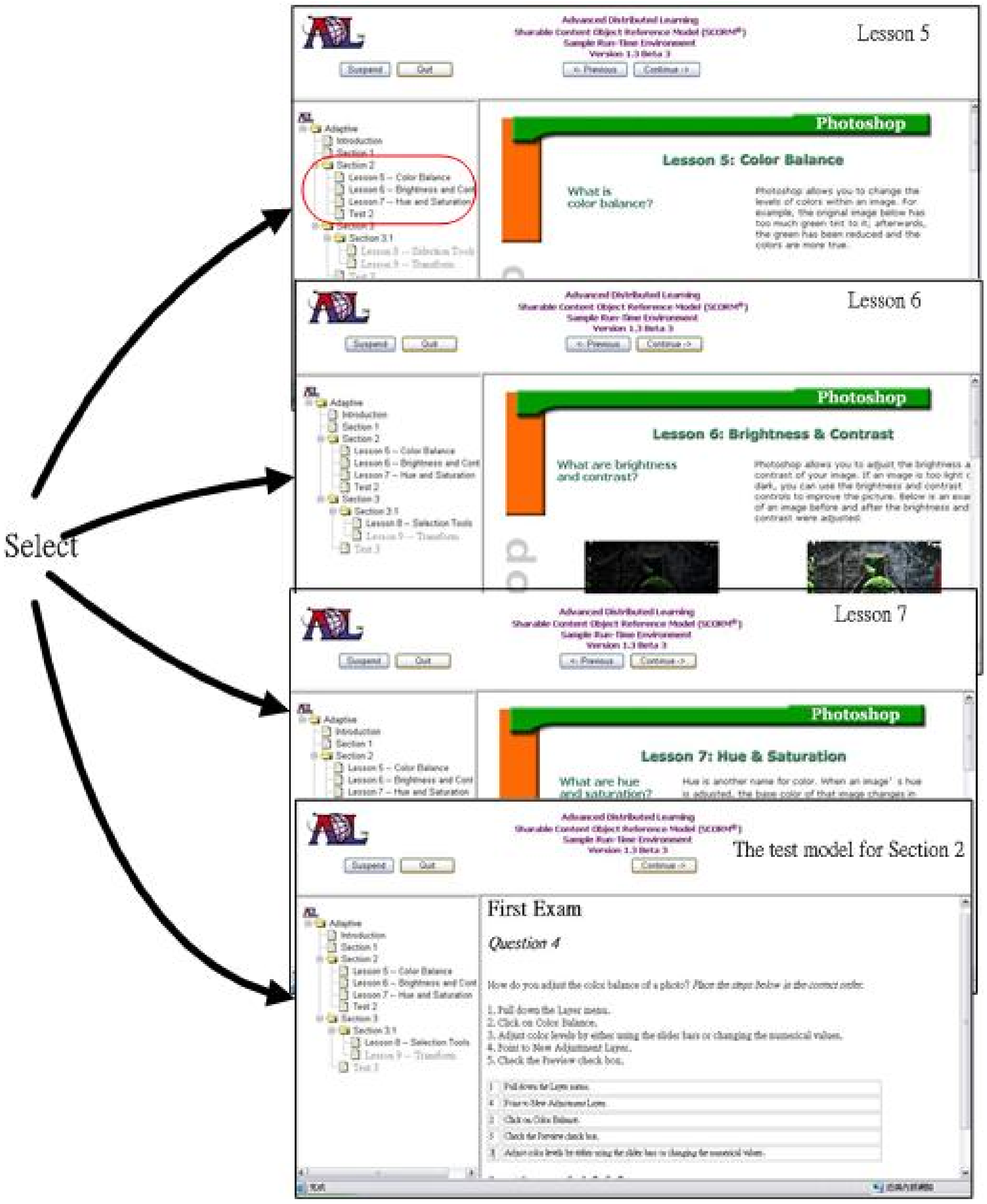


Figure 26: The Learning Procedure of Course B

(3) Course C:

Course C is composed of a *Choice* component which has three chapters and one test model "Test C". Students can freely select any chapter to learn in any order, including the test model. The result of Test C will affect a student's sequencing behaviors in the Course D. Therefore, the learning path will be blocked if Course C isn't performed. Figure 28 shows the learning procedure of Course C.


Select
Lesson 5
Lesson 6
Lesson 7
The test model for Section 2
Advanced Distributed Learning
Sharable Content Object Reference Model (SCORM®)
Sample Run-Time Environment
Version 1.3 Beta 3
Suspend
Quit
<- Previous
Continue ->
Adaptive
Introduction
Section 1
Section 2
Lesson 5 -- Color Balance
Lesson 6 -- Brightness and Cont
Lesson 7 -- Hue and Saturation
Test 2
Section 3
Section 3.1
Lesson 8 -- Selection Tools
Lesson 9 -- Transform
Test 3
Photoshop

Lesson 5: Color Balance

What is color balance?

Photoshop allows you to change the levels of colors within an image. For example, the original image below has too much green tint to it, afterwards, the green has been reduced and the colors are more true.

Lesson 6: Brightness & Contrast

What are brightness and contrast?

Photoshop allows you to adjust the brightness and contrast of your image. If an image is too light or dark, you can use the brightness and contrast controls to improve the picture. Below is an example of an image before and after the brightness and contrast were adjusted.

Lesson 7: Hue & Saturation

What are hue and saturation?

Hue is another name for color. When an image's hue is adjusted, the base color of that image changes in

First Exam

Question 4

How do you adjust the color balance of a photo? *Place the steps below in the correct order.*

1. Pull down the Layer menu.
2. Click on Color Balance.
3. Adjust color levels by either using the slider bars or changing the numerical values.
4. Point to New Adjustment Layer.
5. Check the Preview check box.

1	Pull down the Layer menu.
4	Point to New Adjustment Layer.
2	Click on Color Balance.
5	Check the Preview check box.
3	Adjust color levels by either using the slider bars or changing the numerical values.

Figure 27: The Learning Procedure of Course C

(4) Course D:

Course D consists of a *Linear* component with two transitions: Course D-1 and Test D, which are composed of a *Conditional Choice* component and a Linear component respectively. There are two chapters in the Course D-1. Students will be guided to one of them according to the learning status of Test C in the Course C. Lesson 9 will be assigned if the result of Test C is satisfied; otherwise, Lesson 8 will be delivered to students. Finally, Test 4 will be assigned to evaluate the student' learning progress. Figure 29 shows the learning progress of Course D.

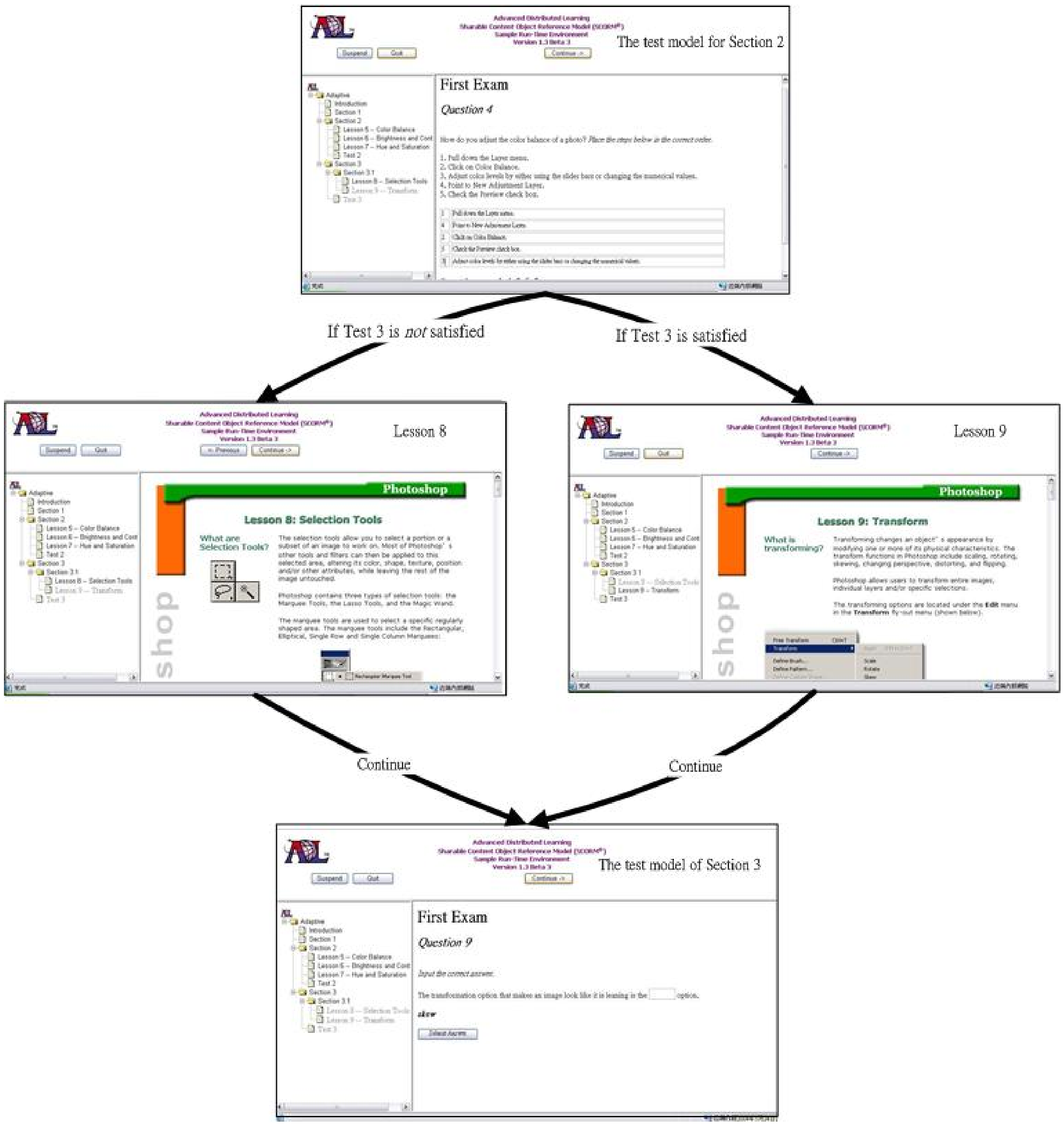


Figure 28: The Learning Procedure of Course D

Chapter 6: Conclusion and Future Work

With the rapidly growing of Internet Technology, e-learning system has become more and more popular. For the consistency of course format, many standards have been proposed. Sharable Content Object Reference Model (SCORM) 2004, the most popular standard, provides the Sequencing and Navigation (SN) Version 1.3 to define the course sequencing behavior. However, the complicated rules and controls of SCORM activity tree make the course sequences hard to design. Therefore, how to provide a user-friendly authoring tool to easily and fast construct SCORM compliant course becomes an important issue. However, before developing the authoring tool, how to provide an approach to analyze the sequencing rules and to transform the created course into SCORM compliant are our concerns.

Therefore, in this thesis, we propose a systematic approach, called **Object-Oriented Course Modeling (OOCM),** to construct a SCORM-compatible course. **High-Level Petri Nets** (**HLPN**), which is a powerful language for system modeling and validation, are applied to modularize the complex sequencing behaviors of SCORM into several sequencing components, which are called **Object-Oriented Activity Trees(OOATs)**. Therefore, these OOATs can be easily and fast used to construct complex sequencing behaviors of course for different learning guidance. Then, we also propose two algorithms, called PN2AT and AT2CP, to transform the constructed HLPN of course into SCORM compliant file described by XML language, which can be executed on the SCORM Compliant system.

In the near future, we will improve the OOAT model for enhancing its scalability and flexibility. Then, based upon proposed sequencing components, an authoring tool will be developed to create the SCORM compliant course with sequencing rules. Moreover, for developing the personalized learning course, applying the Educational Theory to OOATs modeling will also be investigated.

Reference:

[1] IMS (Instructional Management System), http://www.imsproject.org/

[2] SCORM (Sharable Content Object Reference Model), http://www.aslnet/orgScorm/scorm.cfm

[3] "SCORM Best Practices Guide for Content Developers", *Carnegic Mellon Learning Systems Architecture Lab 2003*

[4] Anjaneyuly K, "Concept Modeling on the WWW", *Proceedings of the Workshop on Intelligent Educational Systems on the World Wide Web, the 8th International Conference on Artificial Intelligence in Education 1997*

[5] Fuhua Lin, "Modeling Online Instruction Knowledge Using Petri Nets", *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing 2001*

[6] G.D.Magoulas,M.Grigoriadou, "Neuro-fuzzy Synergism for Planning The Content in A Web-based Course", *Proceeding of the INNS-IEEE international joint conference on neural network 2000*

[7] Hugo Gamboa and Ana Fred, "Designing Intelligent Tutoring Systems: A Bayesian Approach", *3rd International Conference on Enterprise Information Systems, ICEIS'2001*

[8] Jim Prentzas, Ioannis Hatzilygeroudis, and John Garofalakis, "A Web-based Intelligent Tutoring System Using Hybrid Rules As Its Representation Basis", *Proceedings of the ITS-2002*

[9] Jin-Tan David Yang, P.T.Yu, C.Y.Tsai, T.H.Wu, "Visualized Online Simple Sequencing Authoring Tool for SCORM-compatible Content Package", 2004

[10] Jonathan Lee, Lein F .Lai, "A High-Level Petri Nets-Based Approach to Verifying Task Structures", *IEEE Transition on Knowledge and Data Engineering 2002*

[11] Julita Vassileva and Ralph Deters, "Dynamic courseware generation on the WWW", *British Journal of Educational Technologies 1998*

[12] Julita Vassileva, "DCG+GTE: Dynamic Courseware Generation with Teaching Expertise", *Instructional Science 1998*

[13] K.A. Papanikolaou, G.D. Magoulas, and M. Grigoriadou. "A Connectionist Approach for Supporting Personalized Learning in a Web-based Learning Environment", *Internal Conference on Adaptive Hypermedia and Adaptive Web-based Systems 2000*

[14] K.Jensen, "Coloured Petri Nets: A High Level Language for System Design and Analysis", *Proceedings on Advanced in Petri Nets 1990*

[15] Kyparisia A. Papanikolaou, Maria Grigoriadou, Harry Kornilakis and George D. Magoulas, "Personalizing the Interaction in a Web-based Educational Hypermedia System: the case of INSPIRE", *User Modeling and User-Adapted Interaction 2003*

[16] Leonif Sheremetov & Adolfo Guzman Arenas, "EVA: An Interactive Web-based Collaborative Learning Environment", *Journal of Computers and Education 2002*

[17] Mia K. Stern and Beverly Park Woolf, "Curriculum Sequencing in a Web-based

Tutor",*Proceedings of the Fourth International Conference on Intelligence Tutoring Systems (ITS '98)*

[18] Peter Brusilovsky and Julita Vassileva, "Course Sequencing Techniques for Large-Scale Web-based Education", *Journal of Engineering Education and Lifelong Learning 200*

[19] P.Huber, K.Jensen and R.M.Shapiro, "Hierarchies in Coloured Petri Nets", Proceedings on *Advanced in Petri Nets 1990*

[20] R.Stathakopoulou, G.Magoulas, M.Grigoriadou, "A Connectionist Approach for Adaptive Lesson Presentation in a Distance Learning Course", *Proceedings of International Joint Conference on Neural Networks 1999*

[21] Tadao Murata, "Petri Nets: Properties, Analysis and Applications", *Proceedings of the IEEE 1989*

[22] Xiaoou Li and Wen Yu, "Object Oriented Fuzzy Petri Net for Complex Knowledge System Modeling", *Proceedings of the 2001 IEEE International Conference on Control Applications*

[23] X.Q Liu, Min Wu, J.X Chen, "Knowledge Aggregation And Navigation High-Level Petri Nets-based in E-learning", *Proceedings of the First International Conference on Mache Learning and Cybernetics, Beijing 4-5 2002*