The Development of Computer-aided Distance Learning Courses:
An application of formal methods and object-oriented approaches

S K Victor LEE and Sunny SUEN

Abstract

One approach to the development of a distance learning course is to base the study of the course on a number of books within a subject area readily available in the market, and then develop a study guide which consists of a series of study units to help students to understand the course objectives and assimilate the subject matters elaborated in the set books. In order to develop a study guide into an effective computer-aided learning (CAL) tool, we may treat the study guide as a software system to be constructed through the familiar stages of specification, design, coding and maintenance. While CAL software and hardware products can be readily applied to the implementation processes, the specification and design of a study guide is important in determining its composite and functional requirements which are pertinent to distance education learners. As the layout of a study guide is essentially well structured with its contents presented succinctly and systematically and it is often the case that students need to follow a series of steps to retrieve the information they want from the study guide, the specification and design phases should be examined closely. To this end this paper first demonstrates the structural specification of a study guide by formal methods and then proceeds to design it by the object-oriented approach. Finally the implementation issues are outlined with emphasis on hypertext application.

Introduction

The Open University of Hong Kong (OUHK), formerly known as The Open Learning Institute of Hong Kong, founded by the Hong Kong Government in 1989 to provide higher education opportunity to working adults, offers a number of degree programmes in distance learning mode. Each of the degree programmes consists of a number of courses which have been well developed to fit into a coherent whole. The development of such distance learning courses is a major endeavour, among others, carried out by academic staff with the support of the University's Educational Technology and Publishing Unit. One approach we take is to base the study of the course on a number of selected textbooks, and then develop a study guide consisting of a series of study units to help students to understand the course objectives and to assimilate the subject matter. Each of the study units in the study guide consists of a series of sections, each of which contains a number of topics. In the main, each topic provides students with certain study features to facilitate their learning process. Such features normally include explanation and elaboration of study notes, self-answered questions and references to textbooks, to name a few.

The assistance offered by a study guide to students will definitely be enhanced with the computer-aided learning (CAL) techniques. For examples references to textbook can be implemented electronically by hypertext design (Muhlhauser, 1990); the notes of a study guide can be augmented with animation; and self-answered questions can be accommodated in an interactive problem solving environment linking up a knowledge-base and a dialogue interface. In view of the maturity of and rapid advances in the CAL technology where more and more hardware and software products are becoming commercially available or cost-effective to develop, implementation of a CAL-designed study guide can be reduced to a secondary or less important issue. On the contrary more importance should be attached to the specification of a study guide as a software system. In this way the characteristics of a study guide as an aid to distance learners will be analysed and specified correctly for the subsequent implementation by CAL design.

In this paper we illustrate the use of formal methods to specify the components as well as the functions of a study guide. For educational purposes it is essential that a study guide is systematically structured whose contents are presented in a lucid and unambiguous manner so that students are normally able to acquire the desirable information from the study guide through a series of clearly defined procedures. This motivates us to examine the structural specification of a study guide. To elucidate this concept, we resort to a formal specification language made up of Z notations which emphasises the reuse of modular specifications and the construction of complicated operations from simpler ones. Then we illustrate how to design a study guide by means of the object-oriented approach and briefly introduce its implementation in a hypertext environment.

Requirement Analysis of A Study Guide

Based on experience in the OUHK and many other distance learning institutions, particularly the British Open University, we see the study guide as a set of well-structured materials that guide students to acquire knowledge and information in a certain subject area. Thus in the section, we first look at the internal structures of a study guide and its components from a top-down point of view. These will be followed by considering the operations provided for an external user.

Intrinsic and static features of a study guide

1. As a study aid to the distance learning materials of a course, a study guide contains
   - the course title;
   - the introduction and summary of the course;
   - a series of units.

2. A unit of a study guide consists of
   - the unit title;
   - the introduction and summary of the unit;
   - a collection of key terms and their glossaries;
   - a series of section.
3. A section within a unit contains
   - the section title;
   - the introduction and summary of the section;
   - a series of topics.
4. Finally a topic comprises

- its title;
- the introduction of the topic;
- study notes in the form of text, graphics and animation;
- extracts form references to other literature about the topic;
- knowledge about various subject items;
- a series of exercises.

Operations available on the interface between users and a study guide

A study guide software system aims to provide its users with the following facilities:

- Reading the notes about a course. Such notes may take the form of text, graphics and animation.
- Enquiry about a subject that may be covered by the study guide. When the subject is found, the study guide will explain the theories behind the subject which are further illustrated by examples.
- Browsing through the study guides for the introductions and summaries of its units, sections and so on.
- Providing the definitions of technical terms.
- Reference to other literature about various topics covered by the study guide. If a topic is located, the study guide will display the extracts of the reference materials.
- Attempting exercises within the study guide. After the user gives an answer to a question, the study guide will make suitable comments on the answers by linking up a knowledge base and a dialogue interface.

Note that the first three facilities are general features of the whole system whereas the remaining are study features. Next we consider the formulation of the above requirements of a study guide on a rigorous basis in terms of formal specification in Z notations.

Formal Specification in Z - Compositional Specification

To represent knowledge or information concerning a certain topic which is presented by the study guide materials, we consider *Topic* to be the atomic constituent of a study guide. For the time being we are contend that the type *Sentence, Paragraph, Text, GraphicLink, AnimationLink*, *Textbook*, *BookLocation* and *Question* have already been defined elsewhere in advance. As a state, it is specified by the following schema:

Topic

Title : *Sentence*

Introduction : *Paragraph*

Summary : *Paragraph*

Note : *Text*

Note: **P** (*Text* × *GraphicLink* × *AnimationLink*)

reference : **P** (*Textbook* × *BookLocation*)

exercises : Seq [*Question*]

knowledge : *Sentence* → *Text*

From *Topic*, each section of a study guide unit is specified as a schema known as *Section*:

Section

Title : *Sentence*

Introduction : *Paragraph*

Summary : *Paragraph*

s : Seq [*Topic*]

Then a series of sections form a unit of the study guide, which is represented by a schema *Unit*:

Unit

Title : *Sentence*

u : Seq [*Section*]

Introduction : *Paragraph*

Summary : *Paragraph*

Term: **P** (*Word*)

glossary : *Word* → *Paragraph*

---

dom *glossary* = *Term*

Finally a study guide is composed of its units together with other relevant items. It is specified by the following schema *StudyGuide* :

StudyGuide

Title: *Sentence*

Introduction : *Paragraph*

Summary : *Paragraph*
sg : Seq [*Unit*]

Formal Specification in Z II- Functional Requirements

In specifying the operations that can be applicable to a study guide, we resort to the techniques of promotion which are important in Z specification. It is seen that the above specification for the composition of a study guide is developed in a hierarchical structure from the basic component *Topic* to *StudyGuide*. In addition, most of the operations available for *StudyGuide* are derived or promoted from similar operations occurring in its sub-specification *Unit* which are in turn promoted from the sub-specification *Section* and so on. First we would like to consider three promoted operations on *StudyGuide* that belong to the students' interface. These operations are concerned with enquiries about a certain topic, looking for textbook reference to a certain topic and looking up the definitions of a term.
Enquire about a topic

The basic effect of the operation is to make enquiries on the basis of the state *Topic*:

EnquireTopic

Ξ*Topic*
enquiry?: *Sentence*
explanation!: *Text*

enquiry ? $\in$ dom *knowledge*
explanation! = *knowledge* (*enquiry*?)

To appreciate how the operation *EnquireTopic* affect the state *Section* which involves *Topic*, we use a scheme operation known as frame Φ (*Section*) in order to explain the relation between *Topic* and *Section* that makes use of *Topic*.

$\Phi\ (Section)$

$\Delta Section$
$\Delta Topic$

$$\exists\, i : \mathbf{N} \bullet (i \in \operatorname{dom} s \wedge$$
$$s(i) = q\ Topic \wedge$$
$$s'(i) = s \oplus \{i \rightarrow q\ Topic'\})$$

Then according to Hayes (1993), the promoted operation *EnquireSection* on the state *Section* which arises from *EnquireTopic* is formulated as

$$EnquireSection = \exists\, \Delta Topic \bullet \Phi\ (Section) \wedge EnquireTopic$$

Note that the input and output parameters into *EnquireSection* remain *enquiry*? and *explanation*! which are inherited from *EnquireTopic*.

Similarly from *EnquireSection* we derive the operation *EnquireUnit* on the state *Unit*

$$EnquireUnit = \exists\, \Delta Section \bullet \Phi\ (Unit) \wedge EnquireSection$$

where by the same token, the framed schema $\Phi\ (Unit)$ relating the states of *Unit* and *Section* is given by:

$\Phi\ (Unit)$

$\Delta Unit$
$\Delta Section$

$$\exists\, i : \mathbf{N} \bullet (\mathrm{i} \in \operatorname{dom} q\ \wedge$$
$$u(i) = q\ Section \wedge$$
$$u'(i) = u \oplus \{i \rightarrow q\ Section'\})$$

Finally *EnquireUnit* is promoted to *EnquireStudyGuide* on the state *StudyGuide* as

$$EnquireStudyGuide = \exists\, \Delta Unit \bullet \Phi\ (StudyGuide) \wedge EnquireUnit$$

In this approach we start with the operation *EnquireTopic* which directly acts upon the most fundamental basic constituent unit *Topic*. From there the techniques of promotion is used and repeated to propagate *EnquireTopic* from *Topic* to *Section* to *Unit* and finally to *StudyGuide*.

Reference of textbook to a given description of topic

Similar to the enquiry activities, reference to a topic description is made about the relevant parts in textbooks. Therefore the operation first manipulates directly the *Topic* state and is then promoted to act on the *Section* state and so on. The operation *ReferenceTopic* on *Topic* is given by

ReferenceTopic

$\Xi Topic$
$givenitem?: Sentence$
$direction!: \mathbb{P}\ (Textbook \times BookLocation)$

$givenitem\ ? = description$
$direction! = reference$

Then *ReferenceTopic* is promoted to *ReferenceSection* on the state *Section* using the same input and output parameters *givenitem*? and *direction*!:

$$ReferenceeSection = \exists\ \Delta Topic \bullet \Phi\ (Section) \wedge ReferenceTopic$$

where the framed schema Φ (*Section)* is the same as that for the enquiring operation.

Similarly we obtain the promoted operations *ReferenceUnit* and *ReferenceStudyGuide* as follows:

$$ReferenceUnit = \exists\ \Delta Section \bullet \Phi\ (Unit) \wedge ReferenceSection$$

$$ReferenceStudyGuide = \exists\ \Delta Unit \bullet \Phi\ (StudyGuide) \wedge ReferenceUnit$$

It is worthwhile to note that the techniques of promotion enable us to re-use the framed schema Φ (*Section),* Φ (*Unit)* and Φ (*StudyGuide).* Whatever operation is promoted from a constituent state to an assembled state, the same appropriate framed schema is employed.

Definition of a term

It is important to note that the schema *Unit* is directly composed of the set $Term : \mathbb{P}\ (Word)$ and function $glossary : Word \rightarrow Paragraph$. It is these two components that enable us to look up the definition of terms. Therefore the operation starts with the state *Unit* as *LookupUnit*

LookupUnit

$\Xi Unit$
$unknown?: Word$
$definition!: Text$

$unknown? \in Term$
$definition! = glossary\ (unknown?)$

Then from *LookupUnit* is promoted *LookupStudyGude* which relies on the same input and output parameters *unknown*? and *definition*!:

$$LookupStudyGuide = \exists\ \Delta Unit \bullet \Phi\ (StudyGuide) \wedge LookupUnit$$

Apart from the above operations, there are two more ‘classes’ of operations that can be promoted from the elementary state *Topic*. One is concerned with doing exercises on a given topic and is denoted *AttemptTopic*. It is promoted to *AttemptoSection*, *AttemptUnit* and finally to *AttemptStudyGuide*. The other one is called *ReadTopic* which is concerned with reading notes on a given topic. It is subsequently promoted to *ReadSection*, *ReadUnit* and *ReadStudyGuide*. We will leave the formulations of these operations to the readers.

Interface specification - a touch of object-oriented flavours

When we say that a operation is available on a certain state, for example *EnquireSection* on the state *Section*, it implies that the state *Section* forms an interface between a user and the system and that the operation *EnquireSection* is a module in this *Section* interface which can be executed by the user. In order to allow users to move from one interface to another, it is desirable to specify a global schema known as *Interface* which accommodates the states *Topic*, *Section*, *Unit* and *StudyGuide* as classes together with the corresponding operations. On the basis of the above operations, the schema *Interface* is specified briefly as

Interface

state : StateSet
operations : **P** (*OperationsSet*)

---

state = StudyGuide ⇒
operations = {EnquireStudyGuide, ReferenceStudyDuide, DoStudyGuide, ...}
state = Unit ⇒
operations = {EnquireUnit, ReferenceUnit, DoUnit, ...}
state = Section ⇒
operations = {EnquireSection, ReferenceSection, DoSection, ...}
state = Topic Þ
operations = {EnquireTopic, ReferenceTopic, DoTopic, ...}

To allow the user to move to a desirable interface such as *Section*, we define the global schema *InterfaceToSection*

InterfaceToSection

Interface

---

state = Section

The operations available to *Section* are automatically determined by the invariant of *Interface*.

Object-Oriented Design of the Study Guide

As seen above, a study guide system is systematically structured with well-defined components and its operations normally entail straightforward navigation through the hierarchical structure. Together with the above interface specification, these characteristics of a study guide are suitable for object-oriented design where the study guide and its components like section, unit and topic are readily recognised as objects and the operations performed on a study guide are interpreted as the transmissions of suitable messages to and its components which will undertake the required services.

Static behaviours

First we define a class of objects known as user. Its attribute and operation are the intelligent level of the user and the action of giving command to the study guide system. The object belongs to the external environment of the study guide.

User
Intelligence
Issue command

Then to avoid overloading the main object - study guide - with too many operations and to take advantage of separation of concerns, we divide our study guide system into three object classes, namely *Learning System*, *Overview System* and *Study Guide*. All the operations that can be called upon by messages of users are excluded from the sub-object class *Study Guide* whose attributes represent the static intrinsic features of the material while its operation *Search Unit* is internal and concerned with its component unit.

Study Guide
Title Introduction Summary Unit**
Search Unit

Learning System
Study Guide
Read Enquiry Lookup Refer Attempt

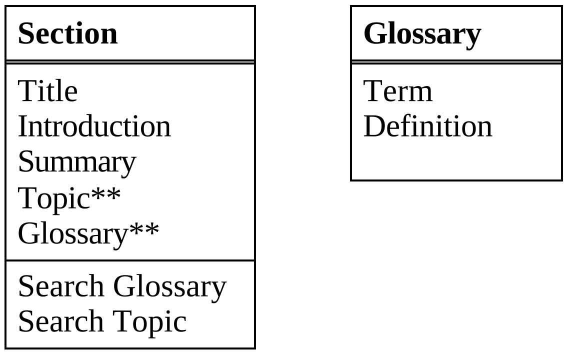
Overview system
Study Guide
Browse Study Guide Browse Unit Browse Section Browse Topic

Learning System is concerned with intensive study activities while *Overview System* enables users to read briefly the introduction and summaries of the study guide, units and etc. The object classes for the Study Guide CAL system are defined in the following and they are denoted with ** as annotation.

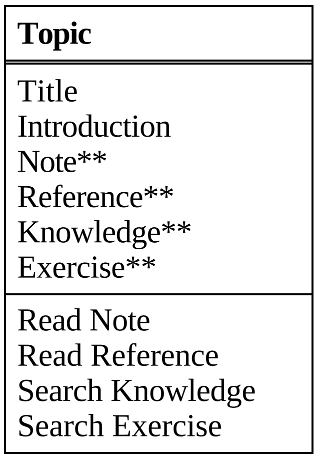
First, below study guide is an object class *Unit*:

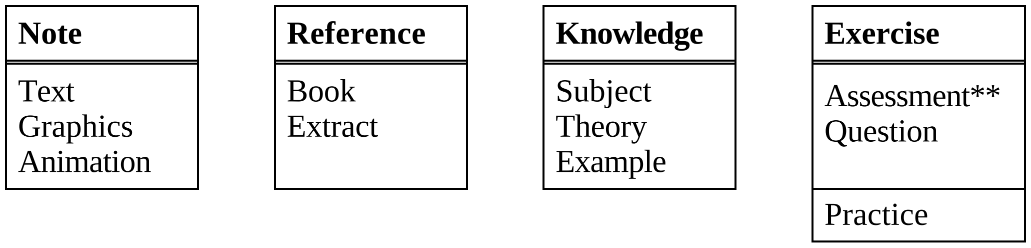
Unit
Title Introduction Summary Section**
Search Section

After unit, there is a class *Section* to which is attached a subclass *Glossary* which allows user to look up the definitions of terms.

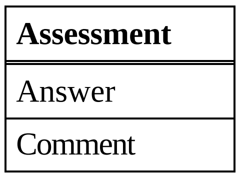


Then there is the ultimate object class *Topic* which represents the fundamental component of a study guide. Under *Topic* there are a number of sub-classes which provides *Topic* with various aspects of information and support its operations as well. These sub-classes of topic are known as *Note*, *Reference*, *Knowledge* and *Exercise*.

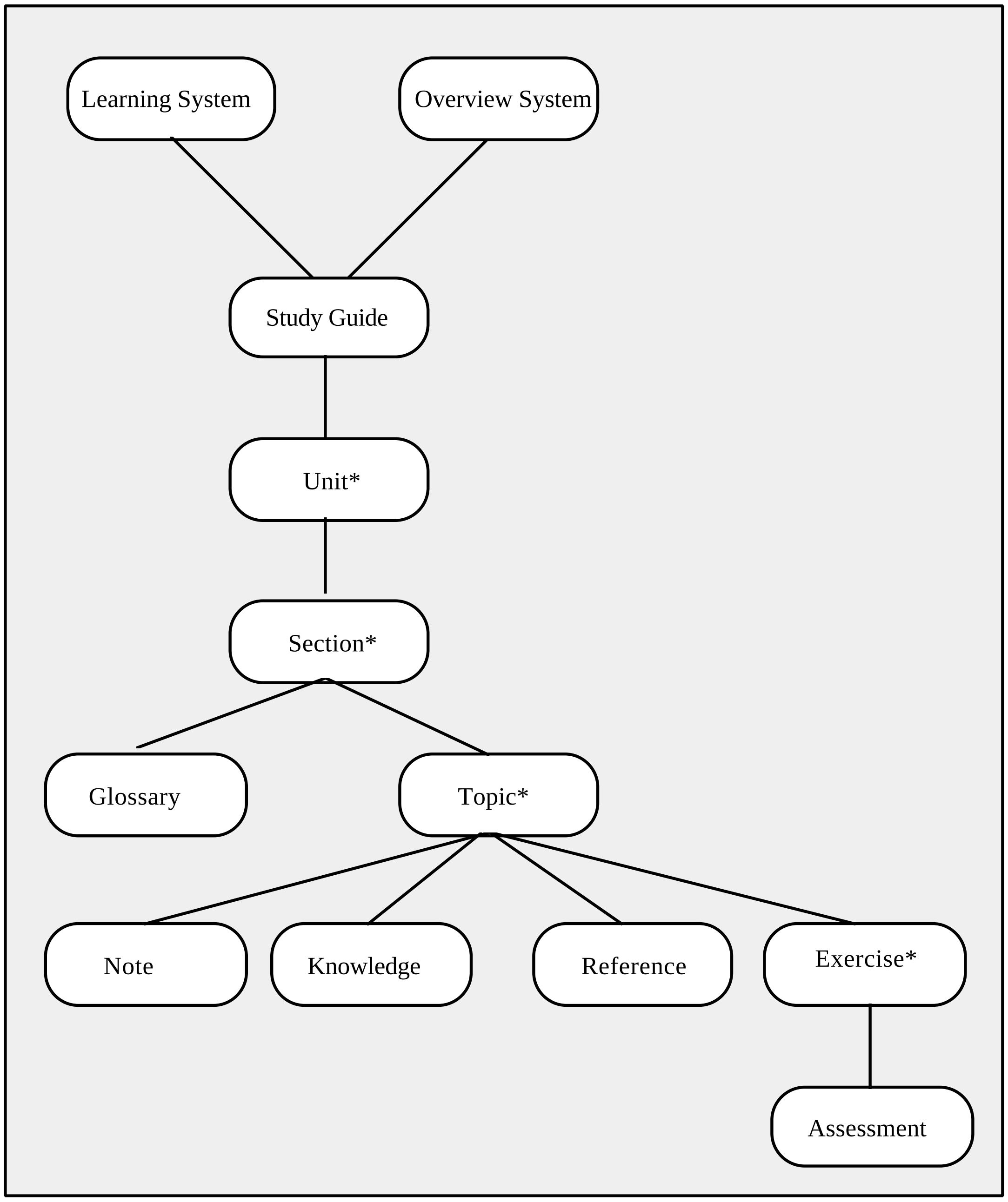




Finally we define the subclass *Assessment* of Exercise which provides service to an object of *Exercise* when it carries out the operation *Practice* via its own operation Comment in response to the answer supplied by the user.



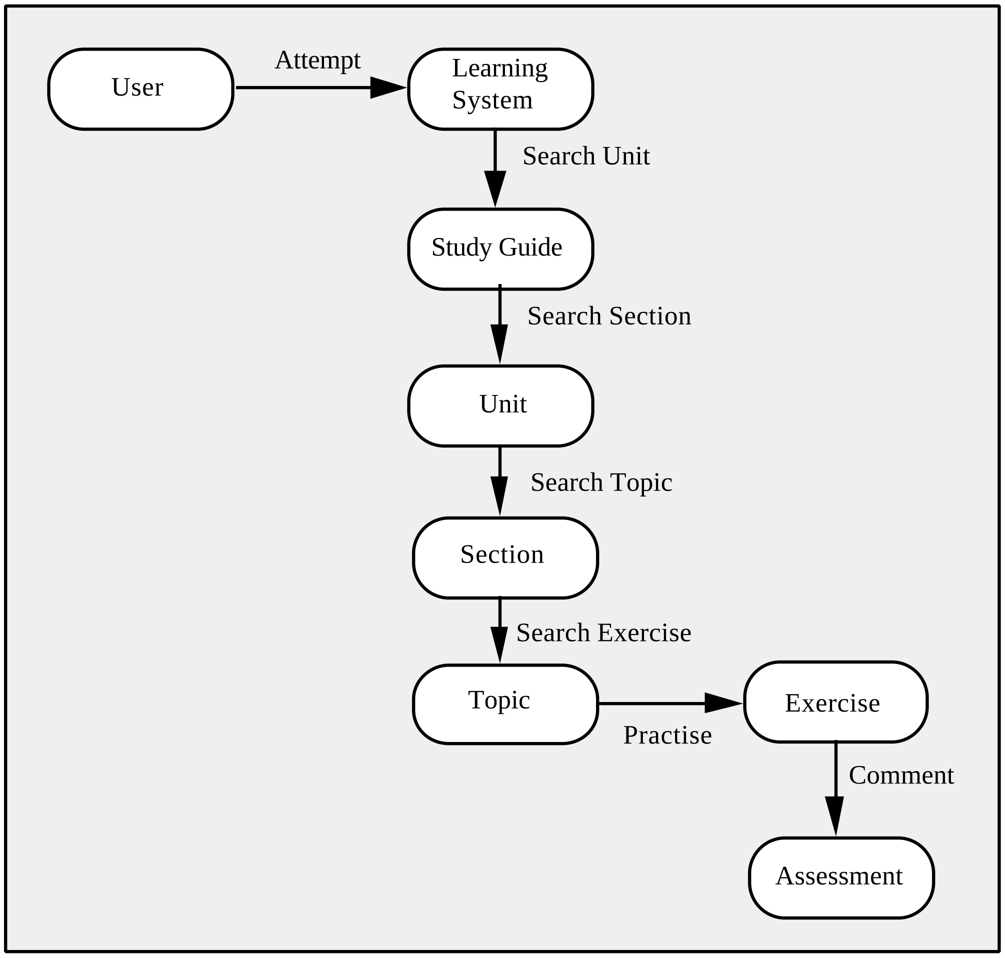
Therefore we have defined 13 object classes including *User* for the study guide CAL system. The static structure showing the inheritance relation among the objects are shown below. For simplicity, the attributes and operations of an object is omitted. Those objects that have sub-classes are denoted with * in the diagram.



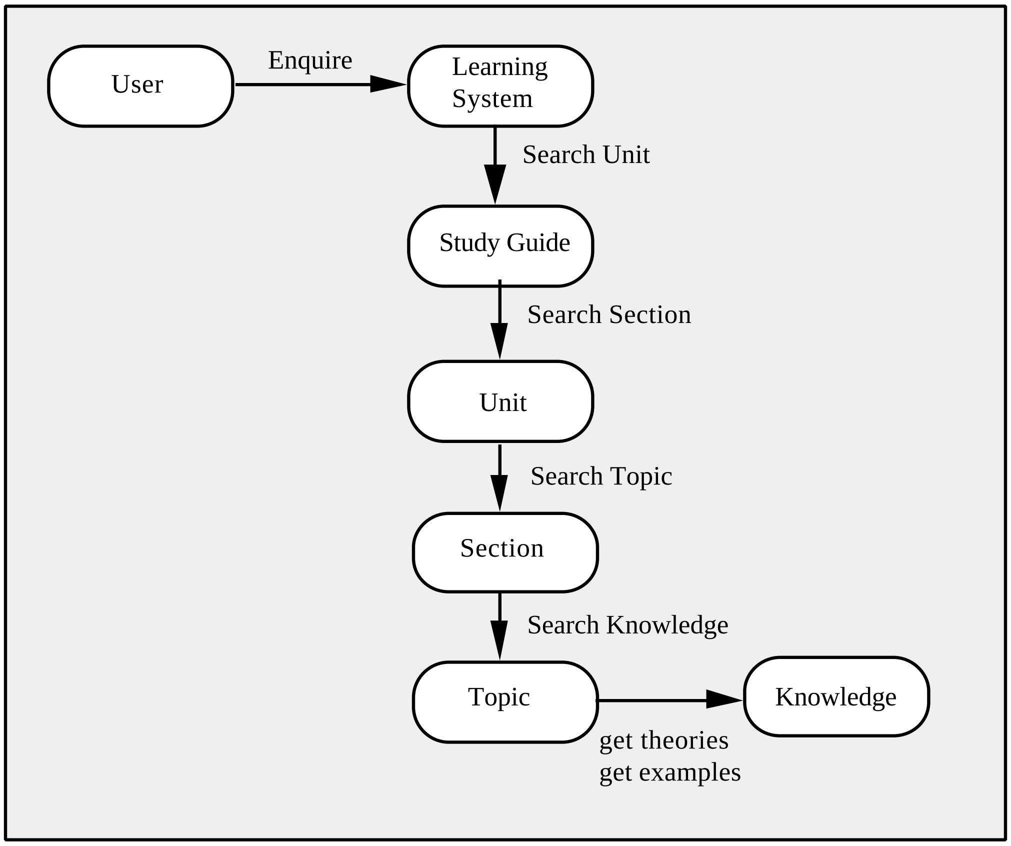
Dynamic behaviours

It will be too complicated and tedious to depict those which are not active among the objects for all the operations performed by the user. So we choose the operations *Enquire* and *Attempt* to illustrate how the objects interact with one other in order to carry out these operations.

When the user issues the *Attempt* command together with a list of given addresses, a message is sent to an object of *Learning System* which then carries out the operation. In doing so, it transmits a message to an object of *Study Guide* to request it to look for the addressed unit. In search of the unit, the study guide requests its units to look for the right section and so on. Finally an object of *Topic* receives a message from a section to find out the correct exercise and then it requests an object of *Exercise* to perform the operation *Practise* which involves displaying the question and asking the user to give an answer. Upon receipt of the answer, it will send a message to an object of *Assessment* which is associated with the answer to make appropriate comments. The diagram follows the practices in (Sommerville, 1996).



Similarly the operation of *Read* is interpreted by object-oriented approach as follows:



Hence we have obtained the object-oriented design of the study guide which reflects the structural and operational characteristics of a study guide.

Note that the "static behaviours" describes the overall structure of the CAL system whereas the "dynamic behaviours" describes the series of operations that are performed for a specific case. It should be made clear that dynamic behaviour does not imply support for dynamic course structure.

Implementation with Hypertext

There are two main reasons for implementing the object-oriented design of a study guide in a hypertext environment. First on an more abstract level, hypertext possesses such capabilities as representation of the organisational structure of information, multimedia information delivery, inter-activity with users, dynamic control of information and etc (Jonassen, 1989). These greatly contribute to the effective performance of a study guide as an instructional media. This encompasses the functions of knowledge conveyance, cognitive learning, problem exploration, literary exchange and browsing facilities.

Secondly the inheritance chart which exists among the objects of a study guide system corresponds naturally to the hierarchical structure of the hypertext that is to be devised to implement it. As each object like *Unit* or *Topic* represents an information fragment, the nodes of the hypertext are mapped conveniently to the objects of the system. Then the associated links can be regarded as the realisation of the inheritance relationships among the objects. They provide the essential channels for the transmission of messages from the super-objects to their sub-objects which are requested to perform their services.

Current advances in information technology enhance the development of a hypertext to implement a study guide. For example the object *Note* resides in a hypertext as a node that contains the elements of text, graphics and animation in a multi-media environment. An expert system shell can be embedded within the object *Assessment* and it is to be furnished with the course-specific questions together with a dialogue interface for the provision of comments and guidance. The enormous storage capacity of CD-ROMs enables the hypertext itself to function as a macro-literacy systems for the object *Reference* which connect user directly to other documents. Finally the up-for-grab commercial product, such as HyperCard, will sufficiently fulfil our purposes.

Conclusion

This paper presents a conceptual framework for the specification, design and (to a lesser extend) implementation of developing a study guide for a distance learning course which makes use of a CAL tool to enhance learning effectiveness. The techniques discussed in this paper have laid a foundation on which we developed a CAL package (Lee et al, 1997) as a supplementary material for a computing course at the OUHK. It is clear that the students who took part in the evaluation of the CAL distance learning course recognised the potential strengths of the medium for the learning and teaching of the OUHK courses. They were keen to see the University build on this experience, and produce more sophisticated learning resources, which relied less on pages of text and more on the available features of hypertext. This project should thus provide encouragement to the OUHK in its effort to use new and emerging information technologies in course development and delivery.

Reference

Hayes I (1993), *Specification Case Studies*, 2nd Edition, Practice Hall International Series in Computer Science

Jonassen D (1989), *Hypertext/Hypermedia*, Educational Technology Publications

Lee SKV, Murphy D, Chan CC and Chung L (1997) "Computer-aided distance Learning: a case study". *Open Learning: The Journal of Open and Distance Learning*, Pitman Publishing, UK

Muhlhauser M (1990), *Hyperinformation in instructional tool environment,* Lecture Notes in Computer Science 438, University of Oxford

Sommerville I (1996), *Software Engineering*, 5th Edition, Addision-Wesley International Computer Science Series