Issues in Informing Science and Information Technology

On-line Learning and Ontological Engineering

Samuel Sambasivam and Tao Li
Azusa Pacific University, Azusa, CA, USA

ssambasivam@apu.edu tli@apu.edu

Abstract

The effectiveness of delivering course contents for distance learning depends on the organization of the course material, interaction methods and selection of exercise/test questions. The selection of questions plays a role equally important as the course presentation material. The use of multimedia may help ease the learning experience and so does the interaction among the students. A systematic approach to structure the course knowledge is perhaps the most important thing to effective learning. We adopt a novel approach to distance learning in which students are made to learn the ontology of the course through a template based approach.

Keywords: Distance learning, knowledge structure, exercise and test, Multimedia, Web based course, learning reinforcement.

Introduction

Although the burst of the internet bubble resulted in a slow down of network structural expansion, the use of the Web for distance learning is steadily growing. Web-based learning (for example, Boysen, & Van Gorp; 1997 and Frasson, Gauthier, & Lesgold, 1996) is increasing its presence all over the world. Web-based material is also increasing for traditional classroom teaching. Web-based methods make learning material available anywhere and anytime. This is a tremendous advantage over other methods.

The primary focus of our university is teaching. Hence, the effectiveness of teaching is of high importance. Our basic approach is classroom based lecturing. However, we have been experimenting with distance learning and Web-based presentation. We offer remote learning classes and Web-based materials for traditional classes. In this paper we present some research results for web-based distance learning.

This paper is organized as follows:

- This section serves as an introduction to the subject of the paper. It also outlines the contributions of the paper.
- The second section gives an overview of the web site organization of our courses.
- The third section shows the ontological database organization of the courses.
- The fourth section discusses our ontology discovery approach to learning. This is the key component of our system.
- The rest of the paper is about tests and learning reinforcement methods.

The key feature that differentiates our approach from others is in the use of learning templates. Our approach aims at helping students to learn the ontology of the course contents. In addition

Material published as part of this journal, either on-line or in print, is copyrighted by Informing Science. Permission to make digital or paper copy of part or all of these works for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage AND that copies 1) bear this notice in full and 2) give the full citation on the first page. It is permissible to abstract these works so long as credit is given. To copy in all other cases or to republish or to post on a server or to redistribute to lists requires specific permission from the publisher at Publisher@InformingScience.org

to the qualitative relationship among the concepts, we also make students learn the quantitative relationships manifested as mathematical formulae. However, this is based on the assumption that the instructors are able to afford significant amount of time themselves. The ***focus*** of this paper is on ontological engineering in the delivery of on-line computer courses.

In addition to using ontological engineering and multimedia presentation, we also find that the making and selection of exercise and test questions is an important factor in effective learning. We provide ample exercises to the students for reinforcement learning.

Organizing Course Material for the Web

Each course web site usually includes the lecture material such as syllabus, slides and lecture notes (reading), assignments and projects, discussion forum, course related software and links to other sites. Some courses also include video and multimedia presentations, learning templates and ontological learning aids such as guided learning tools and automatic question generators. Student information is stored in a database on the server.

Course Site Organization

A typical course web site starts with a top level page that includes the syllabus and links to other pages for the course. The structure of the site is shown in Figure 1. A sample of top level page is shown in Figure 2.

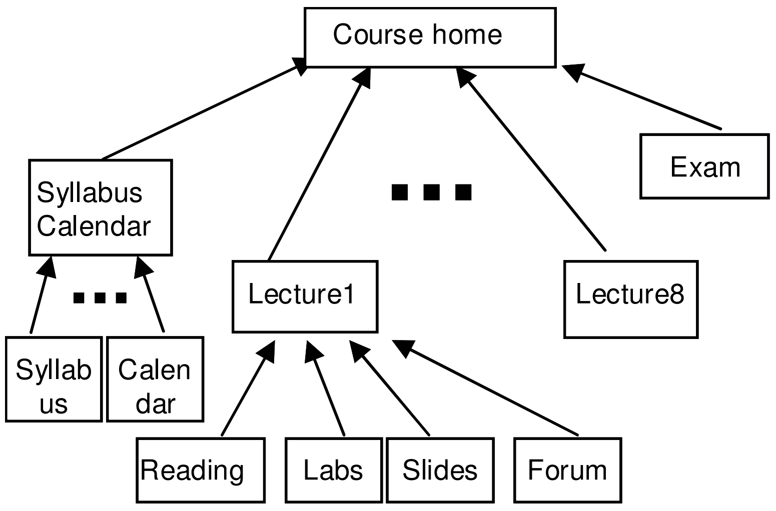


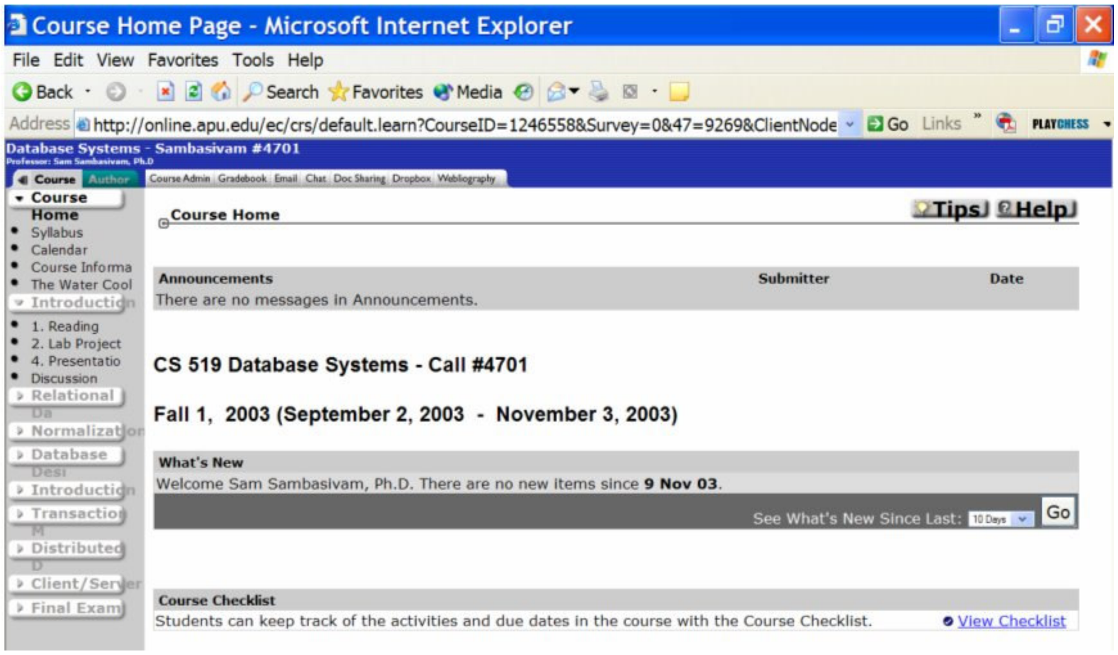
Figure 1. Course Site Structure

From the top level page, the user can navigate to slides-and-notes pages, the code sample pages, the calendar, the lab and assignment pages and the class material pages.

A user may also take the link to a discussion forum for the course. Students can express their opinions and views in the forum. They may also discuss about difficult concepts and assignment questions.

An assessment page is sometimes available and the user can take a test to assess his/her understanding of the materials.

In the end of a course period, students take on-line exams through an examination page. The results are stored in the database.


Course Home Page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://online.apu.edu/ec/crs/default.learn?CourseID=1246558&Survey=0&47=9269&ClientNode
Go Links PLAYCHESS
Database Systems - Sambasivam #4701
Professor: Sam Sambasivam, Ph.D
Course Author
Course Admin Gradebook Email Chat Doc Sharing Dropbox Webliography
Course
Home
Syllabus
Calendar
Course Informa
The Water Cool
Introduction
1. Reading
2. Lab Project
4. Presentatio
Discussion
Relational Da
Normalization
Database Desi
Introduction
Transaction M
Distributed D
Client/Server
Final Exam
Course Home
Tips
Help
Go
View Checklist

Announcements	Submitter	Date
There are no messages in Announcements.		

CS 519 Database Systems - Call #4701

Fall 1, 2003 (September 2, 2003 - November 3, 2003)

What's New

Welcome Sam Sambasivam, Ph.D. There are no new items since **9 Nov 03**.

See What's New Since Last: 10 Days

Course Checklist

Students can keep track of the activities and due dates in the course with the Course Checklist.

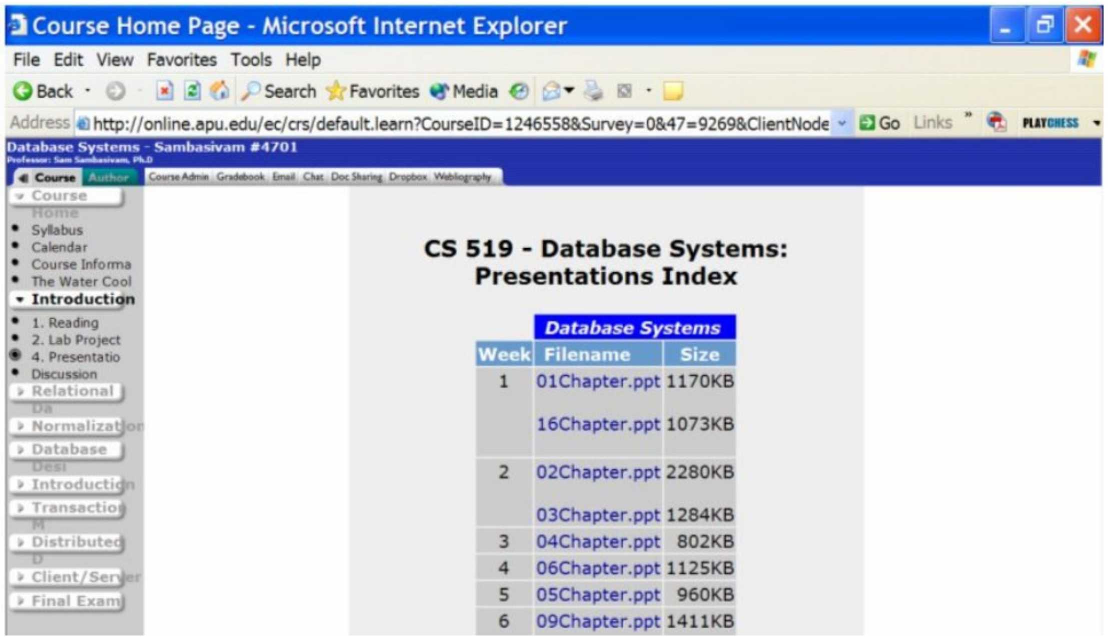
Figure 2. An Example of a Top Level Page

Slides and Lecture Notes

In this paper, we address the issue of presentation and lecture notes in a unique way.

Each week, we provide PowerPoint slides for the week's topic and they are periodically updated on the server. The learning aids and multimedia graphics are provided to enhance the learning.

A simplified view of the PowerPoint slides and lecture notes is given in Figure 3.


Course Home Page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://online.apu.edu/ec/crs/default.learn?CourseID=1246558&Survey=0&47=9269&ClientNode
Go Links PLAYCHESS
Database Systems - Sambasivam #4701
Professor: Sam Sambasivam, Ph.D
Course Author
Course Admin Gradebook Email Chat Doc Sharing Dropbox Webliography
Course
Home
Syllabus
Calendar
Course Informa
The Water Cool
Introduction
1. Reading
2. Lab Project
4. Presentatio
Discussion
Relational Da
Normalization
Database Desi
Introduction
Transaction M
Distributed D
Client/Server
Final Exam

CS 519 - Database Systems: Presentations Index

	Database Systems	
Week	Filename	Size
1	01Chapter.ppt	1170KB
	16Chapter.ppt	1073KB
2	02Chapter.ppt	2280KB
	03Chapter.ppt	1284KB
3	04Chapter.ppt	802KB
4	06Chapter.ppt	1125KB
5	05Chapter.ppt	960KB
6	09Chapter.ppt	1411KB

Figure 3. A Sample Syllabus Of A Top Level Page

The Importance of Ontological Engineering

We find that structuring the knowledge of course content plays a particularly important role in learning. Very often texts do not use explicit and organized knowledge structures to present the relationship among the concepts, neither do they link explicitly the variables associated with concepts through organized hierarchy of formulae. For example, the pioneering work of Hennessy and Patterson was presented in their well-known book about computer architecture (Hennessy & Patterson, 1996). The book is a valuable resource to students, teachers and practitioners. However, it does not provide an explicit, structured view of the relationships among the architectural concepts. Students taking the course often find themselves lost in the quantitative hierarchy and have difficulties solving the problems.

On the other hand, when the conceptual relationship is captured through ontological engineering (Russell & Norvig, 2003), students learn the subject more effectively.

For some courses, we spent significant effort on analyzing the course content and material and we extract the conceptual relationships of the subject into a knowledge structure. The structure ties together the threads of subject matter. A simplified conceptual structure for computer architecture is shown in Figure 4.

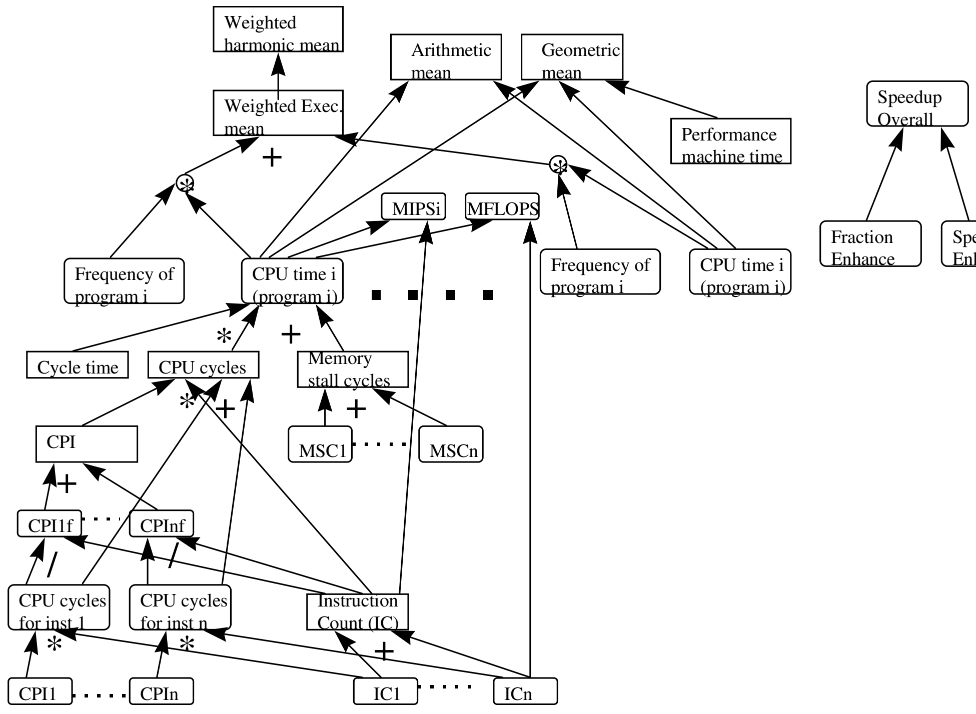


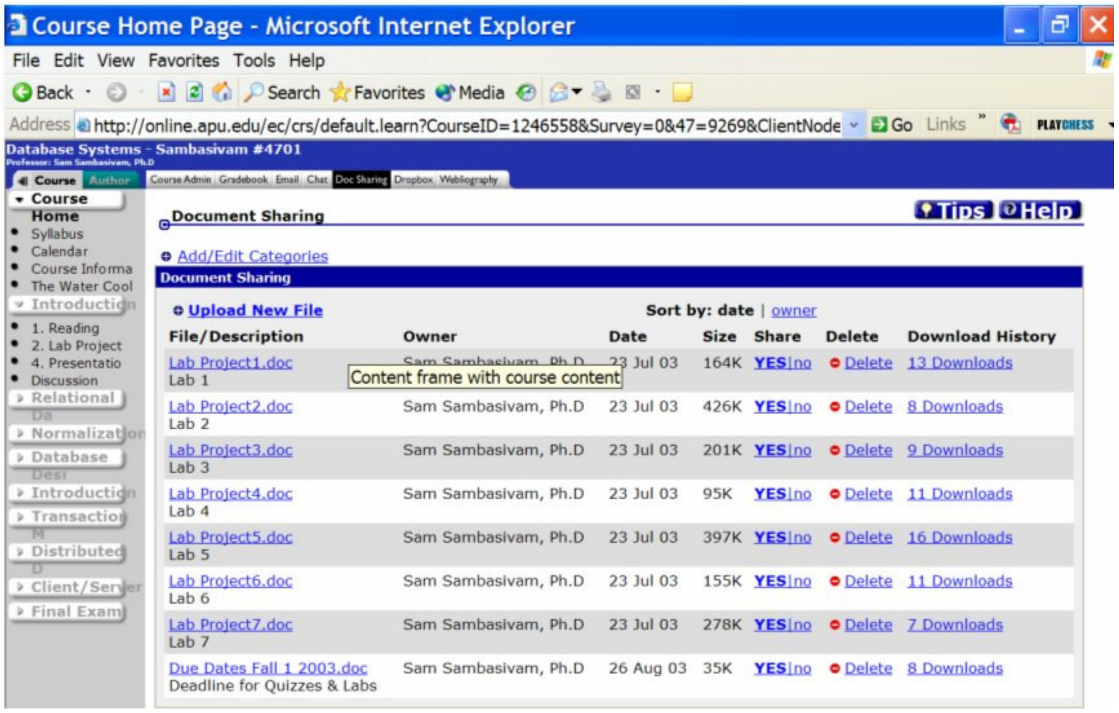
Figure 4. A Simplified Ontology for Computer Architecture.

Lab and Assignment Pages

In this paper, we address the issue of difficulty assessment in problem solving with static knowledge structures.

Practically, the measuring of the degree of difficulty is a subjective matter. However, it is beneficial to make the measuring procedure as objective as possible. Without a proper knowledge model, it is difficult to realize this goal.

We find, from the experience of using exercise questions from some textbooks, that students tend to find a question more difficult when more concepts are involved and more equations are used in solving a problem. This is intuitively easy to understand since it takes more time to organize thoughts when more items are involves and more steps are needed in solving a problem. Of course, the complexity of each equation also adds to the degree of difficulty. But this has less impact on the overall difficulty. A sample lab page is shown in Figure 5.



Document Sharing

Add/Edit Categories

Upload New File — Sort by: date | owner

File/Description	Owner	Date	Size	Share	Delete	Download History
Lab Project1.doc Lab 1	Sam Sambasivam, Ph.D	23 Jul 03	164K	YES|no	Delete	13 Downloads
Lab Project2.doc Lab 2	Sam Sambasivam, Ph.D	23 Jul 03	426K	YES|no	Delete	8 Downloads
Lab Project3.doc Lab 3	Sam Sambasivam, Ph.D	23 Jul 03	201K	YES|no	Delete	9 Downloads
Lab Project4.doc Lab 4	Sam Sambasivam, Ph.D	23 Jul 03	95K	YES|no	Delete	11 Downloads
Lab Project5.doc Lab 5	Sam Sambasivam, Ph.D	23 Jul 03	397K	YES|no	Delete	16 Downloads
Lab Project6.doc Lab 6	Sam Sambasivam, Ph.D	23 Jul 03	155K	YES|no	Delete	11 Downloads
Lab Project7.doc Lab 7	Sam Sambasivam, Ph.D	23 Jul 03	278K	YES|no	Delete	7 Downloads
Due Dates Fall 1 2003.doc Deadline for Quizzes & Labs	Sam Sambasivam, Ph.D	26 Aug 03	35K	YES|no	Delete	8 Downloads

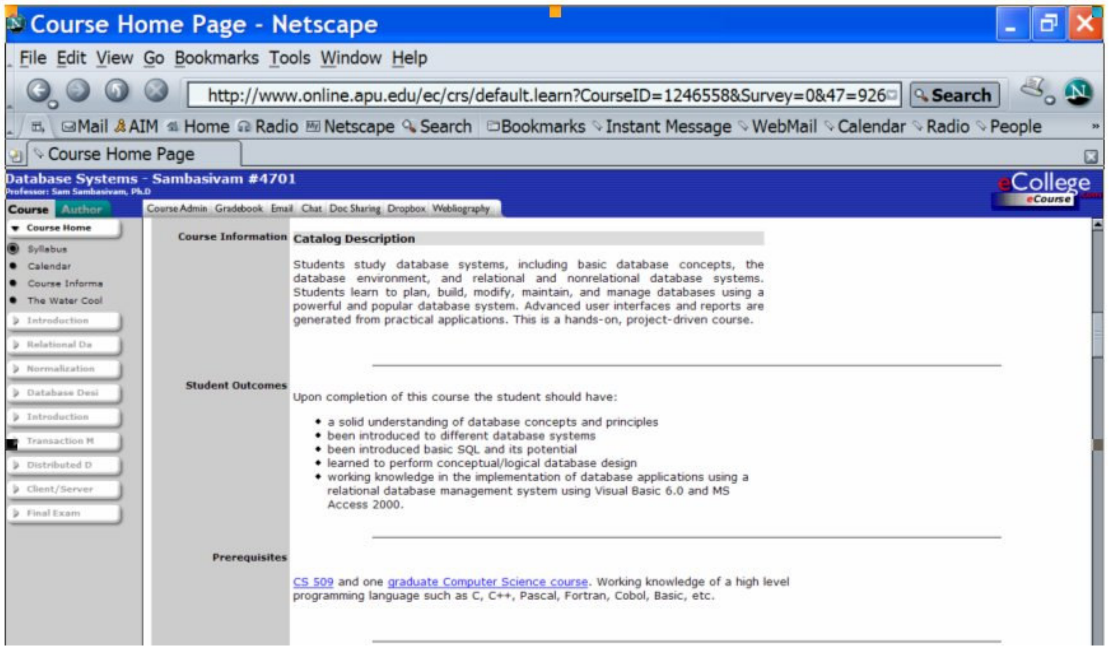
Course Home Page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://online.apu.edu/ec/crs/default.learn?CourseID=1246558&Survey=0&47=9269&ClientNode
Go Links
Database Systems - Sambasivam #4701
Professor: Sam Sambasivam, Ph.D
Course Author
Course Admin Gradebook Email Chat Doc Sharing Dropbox Webliography
Course Home
Syllabus
Calendar
Course Informa
The Water Cool
Introductio
1. Reading
2. Lab Project
4. Presentatio
Discussion
Relational Da
Normalization
Database Desi
Introductio
Transaction M
Distributed D
Client/Server
Final Exam
Tips
Help
Content frame with course content

Figure 5. An Example of a Lab Page (Document Sharing Area)

Calendar, Syllabus and Course Policy Pages

The calendar page describes the course schedule, listing the content of each lecture, etc. A single page suffices to show the calendar.

Under the home page, the syllabus section describes the requirements of the course. It also in-



Course Information — Catalog Description

Students study database systems, including basic database concepts, the database environment, and relational and nonrelational database systems. Students learn to plan, build, modify, maintain, and manage databases using a powerful and popular database system. Advanced user interfaces and reports are generated from practical applications. This is a hands-on, project-driven course.

Student Outcomes

Upon completion of this course the student should have:

- a solid understanding of database concepts and principles
- been introduced to different database systems
- been introduced basic SQL and its potential
- learned to perform conceptual/logical database design
- working knowledge in the implementation of database applications using a relational database management system using Visual Basic 6.0 and MS Access 2000.

Prerequisites

CS 509 and one graduate Computer Science course. Working knowledge of a high level programming language such as C, C++, Pascal, Fortran, Cobol, Basic, etc.

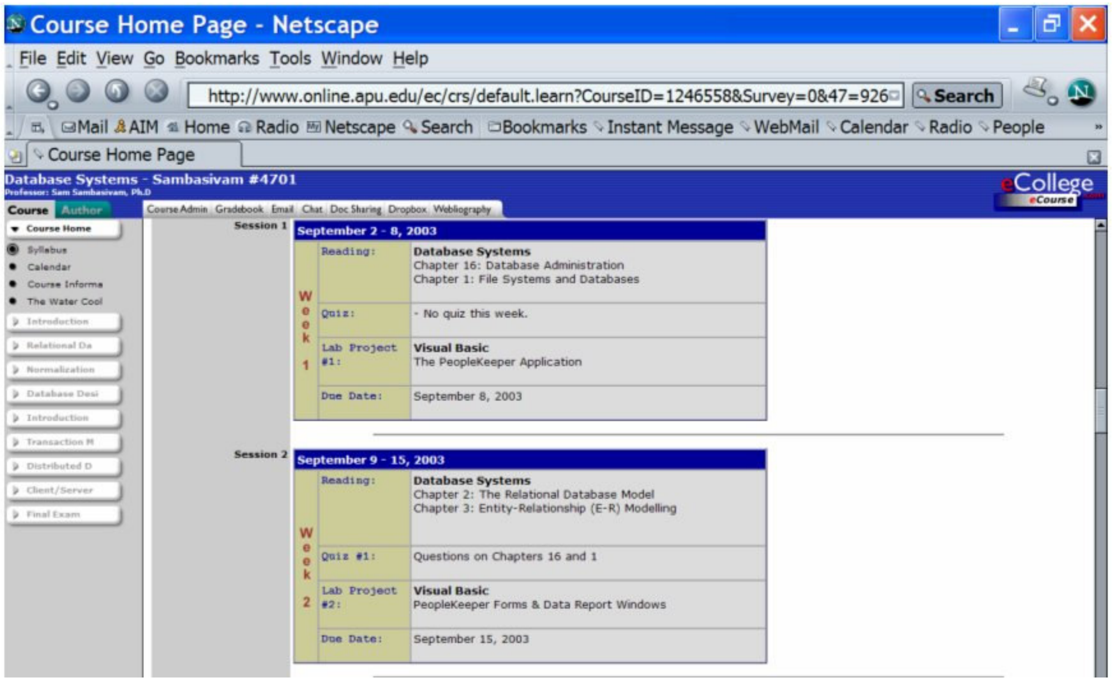
Course Home Page - Netscape
File Edit View Go Bookmarks Tools Window Help
http://www.online.apu.edu/ec/crs/default.learn?CourseID=1246558&Survey=0&47=926
Search
Mail AIM Home Radio Netscape Search Bookmarks Instant Message WebMail Calendar Radio People
Course Home Page
Database Systems - Sambasivam #4701
Professor: Sam Sambasivam, Ph.D
eCollege
eCourse
Course Author
Course Admin Gradebook Email Chat Doc Sharing Dropbox Webliography
Course Home
Syllabus
Calendar
Course Informa
The Water Cool
Introduction
Relational Da
Normalization
Database Desi
Introduction
Transaction M
Distributed D
Client/Server
Final Exam

Figure 6. An Example of a Syllabus Page

cludes the catalog course description, student outcomes, prerequisites, the required texts and software, weekly schedule, evaluation/assessment rationale for grade determination, other course policies, downloads and bibliography. A sample top level page is shown in Figure 6.

The course policy is typically the same for all the courses, hence is shared among the sites for various courses. This is also a single page.

The project and assignment page give student information about each assignment/project and the related requirements. A sample calendar page is shown in Figure 7.


Course Home Page - Netscape
File Edit View Go Bookmarks Tools Window Help
http://www.online.apu.edu/ec/crs/default.learn?CourseID=1246558&Survey=0&47=926
Search
Mail AIM Home Radio Netscape Search Bookmarks Instant Message WebMail Calendar Radio People
Course Home Page
eCollege
Course Author
Course Admin Gradebook Email Chat Doc Sharing Dropbox Webliography
Course Home
Syllabus
Calendar
Course Informa
The Water Cool
Introduction
Relational Da
Normalization
Database Desi
Introduction
Transaction M
Distributed D
Client/Server
Final Exam

Database Systems - Sambasivam #4701
Professor: Sam Sambasivam, Ph.D

Session 1

September 2 - 8, 2003		
Week 1	Reading:	**Database Systems** Chapter 16: Database Administration Chapter 1: File Systems and Databases
	Quiz:	- No quiz this week.
	Lab Project #1:	**Visual Basic** The PeopleKeeper Application
	Due Date:	September 8, 2003

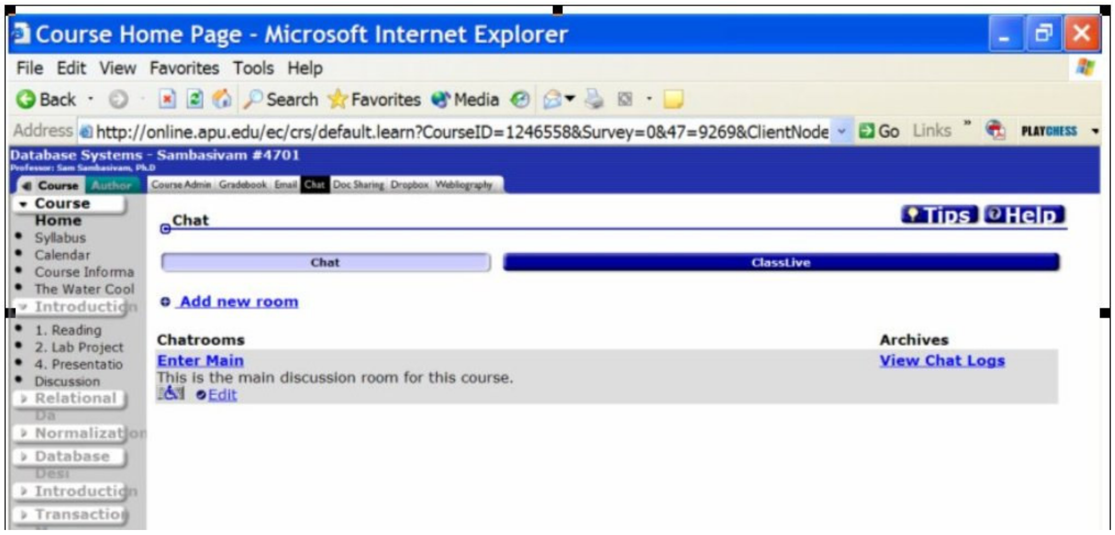
Session 2

September 9 - 15, 2003		
Week 2	Reading:	**Database Systems** Chapter 2: The Relational Database Model Chapter 3: Entity-Relationship (E-R) Modelling
	Quiz #1:	Questions on Chapters 16 and 1
	Lab Project #2:	**Visual Basic** PeopleKeeper Forms & Data Report Windows
	Due Date:	September 15, 2003

Figure 7. An Example of a Calendar Page

Discussion Forum and Chat Room

A discussion forum is provided so that students in the course can help each other in the learning process. We encourage the sharing of learning experience and the discussion forum provides a


Course Home Page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://online.apu.edu/ec/crs/default.learn?CourseID=1246558&Survey=0&47=9269&ClientNode
Go Links PLAYCHESS
Course Author
Course Admin Gradebook Email Chat Doc Sharing Dropbox Webliography
Course Home
Syllabus
Calendar
Course Informa
The Water Cool
Introduction
1. Reading
2. Lab Project
4. Presentatio
Discussion
Relational Da
Normalization
Database Desi
Introduction
Transaction
Tips Help
Chat
ClassLive
Add new room
Edit

Database Systems - Sambasivam #4701
Professor: Sam Sambasivam, Ph.D

Chat

Chatrooms

Enter Main
This is the main discussion room for this course.

Archives

View Chat Logs

Figure 8. An Example of a Chat Page

good tool for this sharing.

A chat room is sometimes provided so that students can have live discussions online. Interactive discussions among students can be a very effective learning method. A sample chat forum is shown in Figure 8.

Ontological Database

The ontological database is typically organized according to chapters of the courses. Ontology is extracted from each chapter. The chapter ontologies are put together and the relationships among the chapters are summarized and included also.

In the beginning, we tried to use the first order logic (FOL) to capture the ontology although FOL is not particularly suited for describing such things as time and event. Later, we adopted an object-oriented representation, similar to the use of frames. This turns out to provide a solution that is readily implemented in current technology. We choose not to use existing markup languages such as SHOE (Heflin, 2001) because of our familiarity with frames and objects and the ease of converting between database tables and frames/classes. In our approach,

- A type is similar to a structure in an object-oriented language.
- A frame is essentially a class in object-oriented design.
- A list or an array of objects is allowed to be a field of another object.
- An inference is a function that produces a result when given a set of parameters.

At the bottom level, concepts are represented with primitive data types such as integers, floats, strings, etc. In some ontologies, for example computer architecture, we associate a numerical value with each concept and an equation with each relation. The use of frames and instances together with lists and arrays help to simplify our solutions since the numerical values associated with concepts and the equations associated with relations can be explicitly encoded in the frames and types.

For example, Amdahl's law (Hennessy & Patterson, 1996) in computer architecture is described by the following equation,

$$Speedup = \frac{1}{(1-\alpha_{enh}) + \left(\frac{\alpha_{enh}}{Speedup_{enh}}\right)}$$

While it is straightforward to compute the Speedup from the equation when α_{enh} and $Speedup_{enh}$ are known, computing α_{enh} from Speedup and $Speedup_{enh}$ requires additional mathematical manipulations. A simple encoding as Speedup(Alpha, Speedenh) is insufficient. Using three predicates to encode the relation for mathematical manipulation is one of several complete representations.

The situation becomes exacerbated when an object is related to a variable number of other objects. Recursion and indexing are used to evaluate such complex formulae. This makes the solution appear complicated in FOL while it is natural and simple in frames and instances.

Frames are coded as classes in object-oriented languages. The program files are our ontological database and they are located on our Web server. These files are consulted when our logic server first starts. We implemented these with Microsoft .NET framework.

Through the modeling of many categories and relations with numerical equations, we learned many valuable lessons of practical issues. The formation of the ontologies also helps to deepen our understanding of the subject under study.

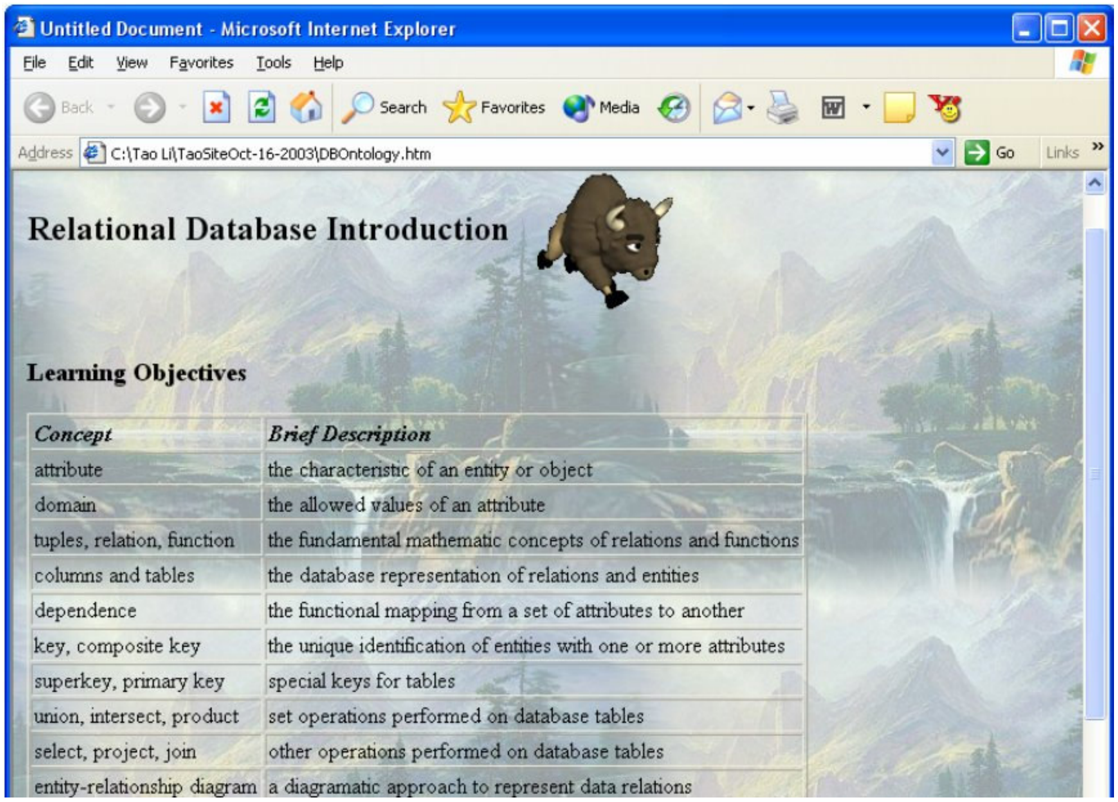
An illustrative database ontology and computer architecture ontology are given in the Appendix. Interested readers may consult the ***Appendix*** for details.

Ontology Discovery Learning

Each lecture comes with a learning template. Our learning templates provide the following information to students:

- Objectives of the lectures.
- Important concepts in the lecture. Some concepts are not listed in the initial templates. They are left for students to complete.
- Relationships among the concepts are left blank (for students to discover and to describe).
- Inference rules, algorithms, and quantitative relations are also left blank. Students are asked to describe these in semi-formal format.

An example is a page for relational database introduction, as shown in Figure 9.


Untitled Document - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\Tao Li\TaoSiteOct-16-2003\DBOntology.htm
Go Links

Relational Database Introduction

Learning Objectives

Concept	*Brief Description*
attribute	the characteristic of an entity or object
domain	the allowed values of an attribute
tuples, relation, function	the fundamental mathematic concepts of relations and functions
columns and tables	the database representation of relations and entities
dependence	the functional mapping from a set of attributes to another
key, composite key	the unique identification of entities with one or more attributes
superkey, primary key	special keys for tables
union, intersect, product	set operations performed on database tables
select, project, join	other operations performed on database tables
entity-relationship diagram	a diagramatic approach to represent data relations

Figure 9. A relational database introduction page

Students are asked to formally describe the concepts, relations and inference rules using object-oriented terminology after the lectures. For example, in the lectures about the basics of relational database, we list the following concepts the students are supposed to learn:

- attributes
- tuples, tables
- functional dependences
- keys and composite keys
- super keys, primary keys and foreign keys

- union, intersection, difference, product, selection, and projection
- join columns, natural join, and other joins

We encourage students to describe these concepts and relations using object-oriented design. These are like the meta level description of relational database entities. They may use pseudo code description for methods/inferences to avoid implementation details.

A number of test questions are given to a student after each section. The results are used to evaluate the student's comprehension of the subject material. The student is then given advice as to proceed to the next lecture or review the subject of the current lecture.

Assessment questions for each lecture are carefully selected simple ones. These questions reinforce the concepts included in the lecture. They also serve diagnostic purposes.

Our learning templates are organized such that students are guided towards self-discovery of the ontology. When a student finishes filling out a template, it is submitted on-line. The instructor can then review the result and diagnose the problem if there is any. For example, we have the following learning templates for database attributes (underlined empty places are left for students to fill out):

```
Frame attributeType  // --- Allowable attribute types ---
   aType ISA Enum { Integer, Float, Char, Text, Null, Binary }
End Frame
Type AttributeRef  // ----- The attribute ref  type -----
   ________ // How to refer to an attribute
   // How to access the reference implemented?
   ________
   ________
End Type
Frame Attribute  // ----- The attribute class definition -----
   ________ // How do you identify the type of data
   ________ // How do you identify an attribute
   ________ // Repeating groups ?
   // What do you want to do with repeating groups?
   ________ // Find out if repeating
   ________ // What to do with repeating groups?
End Frame
```

In the above example, a Type is essentially a structure and a Frame is a class. The fields are placed in the first part of each frame/type and the inferences are placed in the second part.

This approach explicitly guides the students through the ontology discovery steps. Test questions are used to reinforce what they learn from the lectures. The last step is done by the instructor since it is difficult to automate the review process.

Question Selection

One of the major hurdles in building a successful tutoring system is the diagnosis of student's progress. It is extremely difficult to build a comprehensive knowledge base that can model nearly unpredictable student behavior. A typical student model does not cover many unexpected behavior.

Consequently, we do not rely on building student models. Instead, we choose to use test questions for diagnostic purpose. When chosen properly, these tests can also serve the purpose of learning reinforcement. For this reason, it is of particular importance to select the right questions. Other learning assistants use different approaches, for example Culwin (1998), Larkin, Chabay and Sheftic (1990), Wenger (1987), and Wolz (1993).

The main purpose of these questions is learning reinforcement. Hence challenging questions are not suited for our purpose. It would be most effective if the questions are directly mapped to the categories and relations in the ontology. Unfortunately, it is difficult to find questions, in most computer textbooks that directly reflect the ontology. Therefore, we need to make our own test questions.

In making/selecting questions, we use the ontology to guide the making process. Each question is directly related to a few categories and relations in the ontology. The types of questions that we use include the following,

- Direct use of relations and categories. For example, a question may state “Given that the parallelizable portion (i.e., α_{enh}) of a program is 90% and a parallel computer can execute that portion of the program at 4 times the sequential speed execution speed (i.e., $speedup_{enh}$=4), what is the overall speedup when the program runs on the parallel computer.”
- Use of mathematical manipulation. For example, “Given that the parallelizable portion of a program is 90% and the overall speedup of parallel computer is 3 when compared with sequential execution, what is the speedup on the parallel computer for the parallelizable portion.”
- Comparison of results from multiple instances. For example, “Computer A can parallelize 80% of a program. Computer B can parallelize 85% of that program. For the parallelizable portion, computer A can execute 4 times faster and computer B can execute 3.5 times faster. Both computers execute at the same speed (or clock rate). Which computer has better overall speedup?”
- Combination of inputs from a given pool of inputs. This may generate several results for comparison based on the selected combinations.

We select/make many questions so that the sub-ontology of each lecture is thoroughly covered and the number is sufficient for us to assess the student’s understanding of and familiarity with each part of the ontology.

Assessment

For each lecture, the test questions selected should cover the entire sub-ontology of the lecture with sufficient number to provide reliable statistics for assessment.

Assume that the relations in an ontology receive approximately equal coverage by the selected set of questions and the weighted usage of each relation R is U_R. The mistake ratio P_R on relation R is thus

$$P_R = E_R / U_R$$

where E_R is the weighted number of errors made on R.

The weighted usage of a relation R is defined as

$$U_R = \sum_{i=1}^{m} W_i \times F_i$$

Where W_i is the weighting factor of a relation in a specific question and F_i is the number of times that relation is used in the question. We use three weighting factors and the definition of these can be found in [Li, Sambasivam 03].

The weighted error is very similar to U_R and is defined,

$$E_R = \sum_{i=1}^{m} W_i \times G_i$$

where G_i is the number of mistakes made in solving a question.

If the P_R for a relation is below a certain threshold, the student is asked to review the relevant lecture material. An additional set of questions is then selected for the student. All these questions cover the same relation to reinforce the learning of the problematic area.

Summary

This paper presents our effort for the delivery of online courses. To improve the quality of online learning, we adopt advanced methods in course organization.

We present both the course Web site organization and course content presentation through ontological engineering. Such an approach is not widely adopted yet. We have partially automated some aspects of online learning and we hope to incorporate more advanced technology in our online learning environment in the near future.

References

Boysen, P., & Van Gorp. M.J. (1997). CLASSNET: Automated support of Web classes. *ACM SIGUCCS XXV.*

Culwin, F. (1998). Web hosted assessment – Possibilities and policy, *ITiCSE 98 Conference*, Dublin, Ireland.

Frasson, C., Gauthier, G., & Lesgold, A. (1996). Intelligent tutoring systems. LNCS-1086, *3rd International Conference on Intelligent Tutoring Systems* (ITS'96). Montreal, Canada, June 1996.

Heflin, J. (2001). Towards the semantic Web: Knowledge representation in a dynamic distributed environment. Ph.D. Thesis, University of Maryland, College Park. Retrieved from http://www.cs.umd.edu/projects/plus/SHOE/pubs/#heflin-thesis

Hennessy, J. L., & Patterson, D. A. (1996). *Computer architecture: A quantitative approach* (2nd ed.). Morgan Kaufmann.

Larkin, J., Chabay, R., & Sheftic, C. (1990). *Computer assisted instruction and intelligent tutoring systems: Establishing communication and collaboration.* Erlbaum.

Li, T. & Sambasivam, S. (2003). Question Difficulty Assessment In Intelligent Tutor System for Computer Architecture. *Proceedings of ISECON 2003*, San Diego.

Russell, S. & Norvig, P. (2003). *Artificial intelligence: A modern approach.* Upper Saddle River, NJ: Prentice Hall.

SHOE. (2000). The simple HTML ontology extensions (SHOE language specification). Retrieved April 2000from http://www.cs.umd.edu/projects/plus/SHOE/spec.html

Sowa, J. (1984) *Conceptual structures: Information processing in mind and machines.* Reading, MA: Addison-Wesley.

Wenger, E. (1987). *Artificial intelligence and tutoring systems.* Morgan Kaufmann.

Ursula, W. (1993). Providing opportunistic enrichment in customized on-line assistance. *Intelligent User Interface, 93.*

Biographies

Dr. Samuel Sambasivam is the chairman of the Department of Computer Science of Azusa Pacific University. Professor Sambasivam has done extensive research, publications, and presentations in both computer science and mathematics. His research interests include optimization methods, expert systems, Fuzzy Logic, client/server, Databases, and genetic algorithms. He has taught computer science and mathematics courses for over 20 years. Professor Sambasivam has run the regional Association for Computing Machinery (ACM) Programming Contest for six years. He has developed and introduced several new courses for computer science majors. Professor Sambasivam teaches Database Management Systems, Information Structures and Algorithm Design, Microcomputer Programming with C++, Discrete Structures, Client/Server Applications, Advanced Database Applications, Applied Artificial Intelligence, JAVA and others courses. Professor Sambasivam coordinates the Client/Server Technology emphasis for the Department of Computer Science at Azusa Pacific University.

Dr. Tao Li is at the Department of Computer Science, Azusa Pacific University, Azusa, CA 91702. Dr. Li graduated from the University of Utah in 1985 with a Ph.D in computer science. He taught at Adelaide University and Monash University in Australia and Concordia University in Canada. He has offered a wide range of computer science courses. Dr. Li has done research in parallel computing, VLSI design, neural networks and data networking. He served on the editorial board of International Journal of Computer-Aided VLSI Design and on organizing committees of international conferences as well as session chairs of international conferences. He was also invited speaker at conferences and various institutions. His research focus is currently on intelligent systems for computer science education and on hardware based systems for networking.

Appendix: Database and Computer Architecture Ontologies

To simplify the presentation, we use an abstract syntax to specify the ontologies. The inference rules are not given in details. We use pseudo code to describe the functionality of the inference rules.

In the following, a *Type* is a structure, a *Frame* is essentially a class and an *Inference* is implemented with a method.

Ontology for Relational Database:

```
Frame attributeType
   aType ISA Enum { Integer, Float, Char, Text, Null, Binary , etc }
End Frame

Type AttributeRef  // ----- The attribute index class -----
   aIndex AS Integer      // Index to an attribute
   Inference GetRef()
   Inference SetRef()
End Type

Frame Attribute // ----- The attribute class definition -----
   aType AS attributeType         // The type of attribute: Int, Char, Text, etc
   eType AS Boolean     // Is this a par of primary key or non-key
   aName AS String       // The name of the attribute
   repeatFlag AS Boolean           // Repeat group allowed if set true
   // Some methods are defined below
   Inference Check_Repeating_Group()
   Inference Remove_Repeating_Group()
```

```
End Frame

Frame Dependence  // ----- Dependence between attributes -----
   Source AS AttributeRef
   numDepends AS Integer
   DependOn AS List of AttributeRef
   Inference CheckDependence()
End Frame

Frame Key  // ----- The Key class definition -----
   numKeyColumns AS Integer  // The number of attributes of a key
   keyRefs AS List of AttributeRef         // The indexes to key attributes
   Inference IsKey() // Verifies key property
End Frame

Frame foreignKey ISA Key  // ----- The foreign key class -----
   Cardinality As Integer           // Cardinality of a relation
   Inference Check_Cardinality()
End Frame

Frame Table  // ----- The table class -----
     numAttributes As Integer     // The number of attributes in table
   primaryKey As Key              // The primary key
   numFKeys                       // The number of foreign keys
   foreignKeys As List of foreignKey // The groups of foreign keys
   Inference CheckDependence()
   Inference CheckPartialDependence()
   Inference CheckTransitiveDependence()
End Frame

Frame Relation1D  // ----- Single direction of a relation -----
   Cardinality As Integer                    // Cardinality of a relation
   srcTable AS RefTo(Table)      // A reference pointer to a table
   srcColumns AS List of AttributeRef // The indexes of related attributes
   destTable As RefTo(Table)     // A reference pointer to a table
   destColumns AS List of AttributeRef // The indexes of related attributes
End Frame

Frame Database// ----- The database class -----
   Number-of-tables As Integer  // The number of table in database
   Tables As List of Table                   // References to the table in database
   Relations As List of Relation1D          // Relations among the tables
   Inference FindRelations()
   Inference IsNorm1()
   Inference IsNorm2()
   Inference IsNorm3()
End Frame
```

Ontology for Computer Architecture:

```
Frame Computer
   CPU AS component
   MEM AS MemComponent
   Disk AS MemComponent
   SlowDevice1 SlowIOComponent
   SlowDevice2 SlowIOComponent
   FastDevice1 FastIOComponent
   FastDevice2 FastIOComponent
End Frame

Frame MemComponent
   Name AS String
   Capacity AS Integer
   BlockSize AS Integer
   Speed AS Integer
End Frame

Frame IOComponent
   Name AS String
   ItemSize AS Integer
   Address AS Integer
   Status AS enum { Ready, Busy }
End Frame

Frame SlowIOComponent ISA IOComponent
   DataRegister AS Register
   StatusRegister AS Register
End Frame

Frame FastIOComponent ISA IOComponent
   srcAddress AS Integer
   srcSize AS Integer
   destAddress AS Integer
   destSize AS Integer
End Frame

Frame Performance
   Measure AS Float
   BenchMark AS List of Programs
   Type AS enum { MIPS, ExTime, etc }
   Inference ComputePerformance()
End Frame

Frame Program
   CycleCount AS Integer
   Weight AS Float
   ExecTime AS Float
   Inference GetExecTime()
   Inference GetWeightedTime()
End Frame

Frame Cycle_Count
   InstructionCount AS Integer
   CPI AS Float
   Inference GetCycleCount()
End Frame
```