12-2006

Identifying difficulties in learning UML

Keng SIAU
Singapore Management University, klsiau@smu.edu.sg

Poi-Peng LOO

Citation
SIAU, Keng and LOO, Poi-Peng. Identifying difficulties in learning UML. (2006). *Information Systems Management*. 23, (3), 43-51.
Available at: https://ink.library.smu.edu.sg/sis_research/9487

IDENTIFYING DIFFICULTIES IN LEARNING UML

Keng Siau and Poi-Peng Loo

Despite its recognition as a standard object-oriented modeling language, Unified Modeling Language (UML) has been criticized for such deficiencies as semantic inconsistencies, vagueness, and conflicting notations. The relationship between these deficiencies and the difficulties in the learning process is the focus of this study. A concept mapping technique is used to unveil the learning difficulties and suggestions for alleviating them are provided.

KENG SIAU is a professor of management at the University of Nebraska–Lincoln. He currently serves as the editor-in-chief of the Journal of Database Management and editor of the book series Advances in Database Research. He received his Ph.D. degree from the University of British Columbia. Dr. Siau has published more than 10 books, 75 refereed journal articles, and 90 refereed conference papers. His research interests are systems analysis and design, mobile and ubiquitous commerce, and user–database interaction.

POI-PENG LOO received her master's degree from the University of Nebraska–Lincoln. Her research interests include concept mapping.

THE UNIFIED MODELING LANGUAGE (UML) was originally developed by Grady Booch, James Rumbaugh, and Ivar Jacobson in the mid-1990s to facilitate object-oriented systems analysis and design. UML has since become the Object Management Group (OMG) standard and is now accepted as the de facto industry standard (Booch, 1999). Despite this status, UML has been criticized for a number of deficiencies such as semantic inconsistencies, inadequacy of notations, and ambiguities of diagrams and constructs. These problems are also believed to be the primary obstacles in learning UML, yet empirical evidence of this attribution is currently lacking.

With the UML as the standard modeling language for object-oriented modeling, learning UML becomes a necessity for the majority of novice designers, as well as some experienced analysts who are accustomed to only structured and functional paradigms. As such, understanding and alleviating the difficulties in learning UML is important to both practitioners and researchers.

This article reports on the results of a systematic empirical study focusing on identifying the difficulties faced by novices in learning UML. A concept mapping technique is used to develop the various categories of difficulties encountered by the subjects, who were MIS students who had completed a semester course on UML prior to the study.

UNIFIED MODELING LANGUAGE (UML)

UML specifies a set of diagrams and notational conventions for modeling systems based on the object-oriented paradigm. Within the context of software engineering, UML is designed to help software engineers specify, construct, visualize, and document the artifacts of a software-intensive system and to facilitate the communication of ideas (Booch et al., 2005). UML, in its version 1.x, provides nine different types of diagrams with which to model systems. Class diagrams, object diagrams, component diagrams, and deployment diagrams are used to address a system's static aspects; i.e., the organization and structure of the system and its components. Sequence diagrams, collaboration diagrams, statechart diagrams, and activity diagrams are used to portray a system's dynamic aspects; i.e., the behavior, interaction, and states of the components when the system is in execution. In addition, use-case diagrams are used to model the context and requirements of systems (i.e., the use case view of systems). Our study was conducted prior to the

This study empirically examines whether some deficiencies inherent in UML contribute to the difficulties in learning it.

introduction of UML 2.0, but the results still apply and are valid for UML 2.0.

The introduction of UML was intended to unify the various object-oriented methods in the object-oriented marketplace (Booch et al., 2005). Practitioners and researchers (e.g., Simons & Graham, 1999; Dobing & Parsons, 2000; Whittle, 2000; Price et al., 2000; Siau & Cao, 2001; Siau & Loo, 2002; Siau & Lee, 2004; Siau et al., 2005), however, soon pointed out major pitfalls of UML. For example, Siau and Cao (2001) have shown that UML is much more complicated (2 to 11 times more complex) than other modeling methods. Tyson and Frank (2002) wrote that UML 1.x has 144 distinct concepts. Semantic inconsistency is another problem (Whittle, 2000; Dori, 2002). Some parts of the UML models contradict other parts: for example, the "uses" and "extends" in the use-case diagram are defined as stereotypes of generalization and are represented by the standard generalization arrow, but the arrow points to a more general entity (Simons, 1999) instead of the other way around (as in the "include" stereotype). This creates confusion and learning difficulties for novices.

Surveyer (1999/2000) identified three major UML obstacles. First, UML does not include a comprehensive process methodology. Second, UML is not easy to learn because it is fairly sophisticated (complex) and may require other methods and processes in order to map a complex system. Third, UML is skewed more toward the details of how a system is designed and deployed and emphasizes less the analysis and documentation requirements of the system.

Although these are commonly cited UML problems, comprehensive, systematic research on learning difficulties is lacking. This study empirically examines whether deficiencies inherent in UML contribute to the difficulties in learning it.

THE RESEARCH STUDY

The focus of the study is to identify the problems encountered by students in learning and applying UML. The subject of interest can be summarized as, "What difficulties do students have in learning UML?"

The participants in this study were students who had completed the "Object-Oriented System Analysis and Design" (OOSAD) class at a large Midwestern (U.S.) university. The course focuses on introducing the concepts, syntax, semantics, and diagramming techniques of UML; it also covers object-oriented concepts. During the semester, students were required to complete quizzes, in-class exercises, presentations, and case studies to further enhance their understanding of UML; also, as a group project, they had to use UML to analyze a system. Most of these students did not have prior knowledge or experience in object-oriented systems development. This study, therefore, focused on understanding the learning difficulties encountered by UML novices and suggesting ways to help them overcome these difficulties.

The data analysis for this study utilizes a concept mapping technique. Concept mapping is a process that can be used to help a group describe its ideas on any topic of interest (Trochim, 1989a, 1989b) and represent these ideas visually in the form of a map.

As the first step, participants generate a large set of statements relevant to the topic of interest by using techniques such as brainstorming. Then these statements are individually sorted into piles of similar ones and rated on a particular scale. Next, data analysis techniques are used to produce maps in which individual statements are shown in two-dimensional space with more similar statements located near each other. Statements are also grouped into clusters that partition the space on the map. Finally, a structured interpretation session is conducted to help clarify the maps.

The study was conducted in six steps, as specified by Trochim (1989a), and the six steps were carried out in two phases, as shown below. Forty-nine students enrolled in the OOSAD course were recruited to participate in Phase 1 of the study. For Phase 2, we recruited another 30 students who had taken the course.

Phase 1: Preparation and Statements Generation

Step One: Prepare Project. We first identified the problem domain of this study, which is to uncover the difficulties that students face in learning UML. Students were asked to: "Generate statements which describe the difficulties, concerns, and problems in learning UML for systems analysis and design."

Step Two: Generate Statements. The Phase 1 participants were encouraged to generate as many statements as possible that could ideally represent the entire conceptual domain for the topic of interest. The 49 students generated 321 statements. Redundant statements were deleted from the statement list and the rest of the statements were modified to ensure consistency and grammatical correctness. The resulting 192 statements were used for the rest of the study.

We first identified the problem domain of this study, which is to uncover the difficulties that students face in learning UML.

Phase 2: Computing and Utilizing Concept Maps

The criterion for rating the statements generated was developed before Phase 2. A five-point Likert-type response scale was used to measure the difficulty in learning UML, as follows: 1 = not difficult, 2 = somewhat difficult, 3 = moderately difficult, 4 = very difficult, 5 = extremely difficult.

Step Three: Structure Statements. The Phase 2 participants were required to sort and rate the 192 statements generated in Phase 1. Each subject was given a complete set of statements and asked to put the statements they considered to be similar into one pile according to the following sorting rules:

- Each statement can be placed in only one pile (i.e., an item can't be placed in two piles simultaneously).
- All statements cannot be put into a single pile.
- Each pile must consist of more than one statement.

After the participants finished sorting the statements, they were asked to rate each statement using the five-point scale.

Step Four: Compute Maps. After the sorting and rating procedures, researchers generated various concept maps using both a statistical package and manual analysis.

Step Five: Interpret Maps. The main objective of this step was to develop labels for the various categories of difficulties in learning UML.

Step Six: Utilize Maps. This step involved using the maps to help address the original focus; that is, to what extent the clustered difficulties are related to the deficiencies inherent in UML and how future versions can alleviate those issues.

FINDINGS

A 15-cluster solution selected from the concept mapping analysis is shown in Figure 1. The original 192 statements about the difficulties of learning UML from the subjects are divided into 15 clusters, which are enclosed by boundaries. The clusters are labeled based on how the majority of sorters labeled their piles. For example, Cluster 15 is labeled, "lack of prior knowledge/experience of UML or programming."The label was derived from the following wording/phrases provided by the subjects:

"Lack of programming experience"
"Lack of experience in UML"
"Do not have any knowledge of UML before taking this class"

As another example, Cluster 6 is labeled, "too many constructs/concepts/techniques and difficult to remember," based on subjects' statements such as the following:

"Too many constructs in UML"
"Too many concepts in UML"
"Too many diagramming techniques in UML"
"Too many concepts and some of them seem unpractical"

FIVE CATEGORIES OF UML LEARNING DIFFICULTIES

The 15 clusters in Figure 1 were consolidated into five meta-regions, which reflect the major categories of difficulties perceived by the subjects in learning UML. Each of these is discussed below.

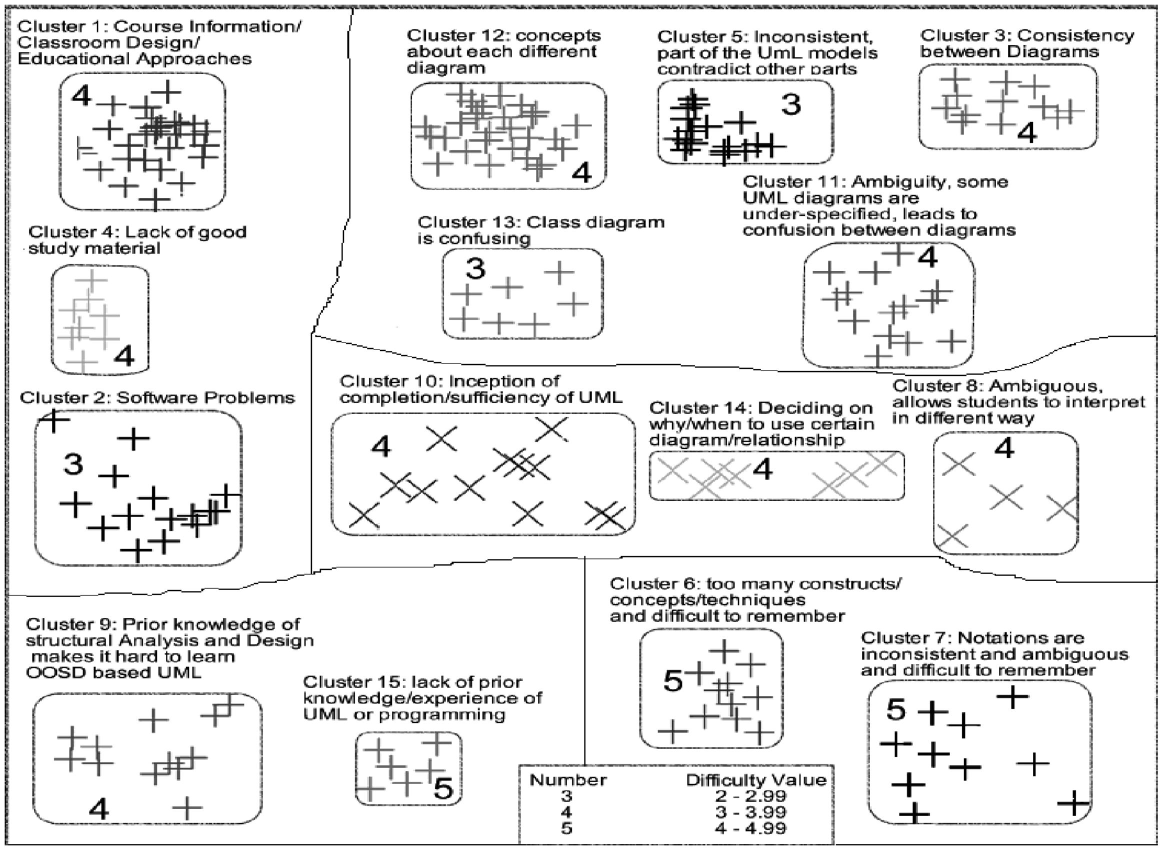
Region #1 — Training Material (Clusters 1, 2, 4)

The subjects claimed that insufficient course information, a crowded classroom, lack of a good textbook, and the user-unfriendliness of the CASE (computer-aided software engineering) tool created moderate difficulty in learning UML. Typical statements about this region include:

"Spend too much time on class diagrams and not enough on others"
"I think we should have learned more about the unified process"
"Not seeing a completely finished product"
"Bugs in UML software (i.e., Rational Rose)"
"UML software is too expensive and slow to access"
"How to link diagram in Rose"
"Many UML examples contain errors"
"Textbook is hard to follow"

According to theories and literature on cognition and learning (e.g., Long, 1990; Langley & Simon, 1981; Shuell & Lee, 1976) the "big jump" on the learning curve is usually created by having assistance during the process of

FIGURE 1 Cluster Rating Map


Cluster 1: Course Information/ Classroom Design/ Educational Approaches
Cluster 12: concepts about each different diagram
Cluster 5: Inconsistent, part of the UmL models contradict other parts
Cluster 3: Consistency between Diagrams
Cluster 11: Ambiguity, some UML diagrams are under-specified, leads to confusion between diagrams
Cluster 13: Class diagram is confusing
Cluster 4: Lack of good study material
Cluster 2: Software Problems
Cluster 10: Inception of completion/sufficiency of UML
Cluster 14: Deciding on why/when to use certain diagram/relationship
Cluster 8: Ambiguous, allows students to interpret in different way
Cluster 6: too many constructs/ concepts/techniques and difficult to remember
Cluster 7: Notations are inconsistent and ambiguous and difficult to remember
Cluster 9: Prior knowledge of structural Analysis and Design makes it hard to learn OOSD based UML
Cluster 15: lack of prior knowledge/experience of UML or programming
Number Difficulty Value
3 2 - 2.99
4 3 - 3.99
5 4 - 4.99

learning. Assistance could be lectures in class, discussion, and additional supporting materials. Without help from these sources, a subject's UML learning curve would be long and steep. With the pervasive adoption of UML as the standard object-oriented modeling language, good learning materials have been introduced, and this aspect is improving rapidly. Software vendors are also enhancing their CASE tools and providing tutoring software for novices to learn UML.

Another training issue relates to the trade-off between the limited memory capacity of humans and the complexity of UML (e.g., Anderson, 1976, 1989, 1993, 1996; Siau, 1999). Within a typical semester schedule (i.e., 48 class hours over 16 weeks), instructors may have difficulty teaching all the UML diagrams and providing plenty of hands-on practice. According to the information processing theory (Anderson, 1976, 1989, 1993, 1996), both lack of rehearsal and information overload can cause the loss of information in short-term memory, resulting in difficulties in the learning process. UML 1.x has nine different diagramming techniques. UML 2.0 has 13 diagramming techniques. It is expected that UML will continue to grow in size. The size and complexity of UML pose problems in learning it (Siau et al., 2005).

Region #2 — Prior Knowledge (Clusters 9, 15)

The typical statements of this meta-region are listed below:

"Difficult to understand UML using concepts from structured analysis and design"
"It is too different from the traditional method"
"A shift from procedural to object orientation was difficult"
"Learning UML is hard if we lack some programming experience"
"Designing a system while learning a new language was very hard"

TABLE 1 Issues Inherent in the UML Diagrams

Typical Statement	What the Statement Means
"Hard to show when actors are involved in diagrams"	Actors are used in the use-case diagram and may appear in the interaction diagrams. The training materials do not "link" the various diagrams together. Students are confused about the use of actors in the use-case diagram and how the actors in the use-case diagram relate to the actors in the interaction diagrams.
"The relationships between different diagrams are confusing"	The "linkages" between different diagrams are not obvious to the students. For example, how is class diagram "linked" to interaction diagrams?
"The role of use-case diagram is confusing"	The use-case diagram is a controversial diagram in UML. It is different from the other UML diagrams and its role has been questioned.
"Hard to decide when to use statechart diagrams and when to use activity diagrams"	The activity diagram is a special case of statechart diagram. Students are confused about when to use activity and when to use statechart diagram.
"Difficult to understand constraints in class diagram"	Constraint specification may need to be included for some class diagrams, but constraint specification is difficult for novices in UML to understand.
"Very difficult to come up with error-free solution/diagram"	UML has many constructs and syntax rules. UML is not easy for a novice to learn.
"<<include>> is hard to understand"	The "include" stereotype is not easy for students to understand.
"<<uses>> and <<extends>> generalizations are inconsistent"	The direction of the arrowhead is different for "uses" and "extends" stereotypes.

According to the information processing theory, when learners acquire new information, they attempt to relate this new piece of information to their existing knowledge structure. Thus, when subjects are introduced to the concepts of UML, the foremost action is to search and recall similar concepts, knowledge, or production rules from their knowledge pool (long-term memory) and then try to create a new production rule that combines the new concepts learned with the old knowledge. If learners have very little or no experience in object orientation or UML, they can encounter difficulties in processing UML concepts and constructs in their memory and combining knowledge.

In addition, prior knowledge of a procedural and functional paradigm may interfere with the learning of UML, which is based on an object-oriented paradigm. Besides the discrimination effect of prior knowledge, Dué (1993) believed that "the change in mindset required to move to the object paradigm seems to be the single biggest obstacle." Pennington, Lee, and Rehder (1995) found that even after object-oriented training, "certain procedural practices crept into" the efforts of novice object-oriented designers. By inference, then, the best candidates to learn UML will be those who have prior knowledge of object orientation or object-oriented programming and those who have not been exposed to structured techniques.

Region #3 — UML Diagrams (Clusters 3, 5, 11, 12, 13)

Inconsistent and confusing diagrams and constructs are obstacles in learning UML. Table 1 lists the typical statements on this issue and what those statements were interpreted to mean.

According to the ACT-R theory (Anderson, 1976, 1989, 1993, 1996), generalization and discrimination are important outcomes of knowledge compilation. However, the inconsistency and confusing diagrams in UML impact the effectiveness of compiling declarative knowledge (e.g., definitions, meanings of diagrams, and other constructs of UML) into procedural knowledge (e.g., how to draw UML diagrams to effectively model an aspect of a system).

In addition, subjects pointed out that a number of concepts could not be captured using UML notations. They also claimed that sometimes they could not express what they were trying to model. These difficulties can result from the inadequacy of UML itself. The findings for Region #3 thus show that these

TABLE 2 Issues Inherent in the UML Semantics

Typical Statement	What the Statement Means
"There is more than one correct solution to a problem"	There is more than one way to represent something, and that is confusing to the students. For example, some relationships can be represented as either associations or aggregations.
"For a huge project, combining every diagram together to make a whole picture of the system is not easy"	Students cannot integrate and synthesize the information from the various diagrams.
"Unsure about the logical correctness of a representation"	Students are unsure whether certain representations are logically correct. For example, whether to use an association or an aggregation depends on how the object is to be stored and how the object is to be used. It is not easy for novice UML users to understand the implications of certain representations and when to use certain representations.
"Knowing when to use each type of diagram"	Students are not sure when to use a certain diagramming technique.
"Determining proper relationship to use in the diagrams"	There are many types of relationships in UML; students are confused when to use a certain relationship.

weaknesses are the source of at least some of the difficulties in learning UML.

Region #4 — UML Semantics (Clusters 8, 10, 14)

The pattern displayed in Region 4 indicates that subjects were unsure about some of the semantics of UML.Table 2 details the typical statements for this region and the interpretations.

These difficulties are mainly due to those UML semantics that are not precisely defined. This is an inherent problem of the unifying process, which produces UML from many different object-oriented modeling techniques (Selic et al., 2002). Furthermore, the ambiguity of UML semantics poses problems for novices and for forward engineering (i.e., generating codes from models).

Region #5 — UML Constructs (Clusters 6, 7)

Subjects had problems memorizing the notations and constructs in UML. The difficulties mentioned are included in Table 3.

The large number of constructs in UML increases the difficulty in learning, because humans are constrained by the size of their short-term memory (Miller, 1956). To make learning UML easier, we need to reduce the number of constructs, resolve inconsistencies among constructs and diagrams, remove ambiguities in UML, and increase hands-on practice during the learning process.

RECOMMENDATIONS

The cluster rating map in Figure 1 suggests that there are five major issues in learning UML. These issues can be divided into inherent and peripheral categories. In the following, we present recommendations for alleviating learning difficulties for the inherent and peripheral categories.

Category One: Inherent problems in UML

- ❑ Issues related to UML diagrams (Region #3)
- ❑ Issues related to UML semantics (Region #4)
- ❑ Issues related to UML constructs (Region #5)

For these problems, the following recommendations are suggested to help alleviate the learning difficulties:

1. *Clearly define UML constructs and concepts.* The semantics of some UML constructs and concepts are poorly defined. This creates learning difficulties for students — especially inexperienced and novice users of UML. For example, the definitions of stereotypes <<Include>> and <Extend>> in Booch et al. (2005, p. 234) are beyond the comprehension of novice UML users. UML constructs and concepts should be clearly defined and discussed in an easily understandable way (rather than using formal technical jargon) to avoid ambiguities and misunderstanding. As the original UML was developed by incorporating many object-oriented modeling techniques, the later UML versions should try to

TABLE 3 Issues Inherent in the UML Constructs

Typical Statement	What the Statement Means
"Too many constructs in UML"	UML has many constructs and it is difficult to learn them and their notations.
"Too many concepts and some of them seem unpractical"	UML has many concepts, some of which are seldom used. For example, qualified association is not commonly used.
"Symbols in UML mean different things in different diagrams"	UML notations are difficult to learn and familiarize.
"Learning the complete syntax was daunting"	The UML syntax can be confusing. For example, filled solid arrowhead means nested flow of control whereas stick arrowhead represents flat flow of control. Also, the direction of arrowhead is important.

better integrate these techniques to provide learners with consistent notations, constructs, and concepts.

2. *Discuss the roles and uses of various diagrams in UML.* The roles and uses of the UML diagrams should be discussed in the training materials and lectures. For example, the use-case diagram is probably the most controversial diagramming technique in UML. The role and uses of the use-case diagram in UML need to be clarified and explained. Students are also confused about when to use the sequence and when to use the collaboration/communication diagrams. Similar issues exist for the activity and the statechart/state diagrams. These issues need to be addressed by the instructors and hopefully alleviated in future versions of UML. Hands-on practice and real-world projects should be important components of a training program.
3. *Discuss the "linkages" between various diagrams.* The current textbooks and training materials on UML do a poor job discussing the "linkages" between various diagrams. The various UML diagrams are used to present different aspects of a problem. However, students find it difficult to integrate and synthesize the information depicted in different diagrams. They also find it difficult to see the connections between the various UML diagrams. For example, how are the use-case and class diagrams linked to the interaction diagrams. The linkages between various diagrams need to be explicitly discussed in the textbooks, training materials, and classrooms.
4. *Focus on core UML diagrams and constructs.* UML is inherently large and complex; thus, teaching all the diagrams and constructs is a daunting task for instructors and learning them is an impossible endeavor for novice UML users. The 20–80 rule can be applied to the learning of UML: only 20 percent of UML constructs are needed to specify 80 percent of the common software solutions. Therefore, it is important to identify the core diagrams and core constructs in UML (Siau et al., 2005) and to give priority in teaching them. Different application domains will require different domain-specific extensions to the core UML (Duddy, 2002; Selic et al., 2002), and these extensions can be learned by the users when necessary.
5. *Enhance UML semantics with a constraint language.* The inherent problems with UML semantics can be enhanced and clarified with a constraint language that will reduce ambiguities and facilitate forward engineering. Although learning the Object Constraint Language (OCL) is not recommended for UML novices, the language can be a basis for devising a novice-friendly language to represent constraints.

Category Two: Peripheral Issues in Learning UML

❑ Issues related to training material/software (Region #1)
❑ Absence/presence of prior knowledge (Region #2)

The following are some suggestions and recommendations:

1. *Better textbooks and training materials.* Most of the current textbooks and training materials use UML as examples in discussing software development or systems analysis and design. The emphasis of these textbooks is not on UML, and the coverage of UML in these textbooks is less than adequate. UML is complex and consists of many diagramming techniques. Complete

Learners who have been exposed to the structured paradigm should be cautioned not to transfer knowledge from the structured paradigm to the object-oriented paradigm.

and comprehensive coverage of UML in textbooks and training materials may not be suitable for novice users, but it is needed in advanced courses on UML. Thus, we need basic but adequate coverage of UML in textbooks written for novice UML users and comprehensive coverage of UML in textbooks targeted at advanced UML users.

2. *Acquire knowledge of object orientation.* Before learning UML, learners should preferably have some knowledge of the object-oriented paradigm. Object-oriented programming experience in C++, Java, or Visual Basic would be helpful for learning UML. Prior knowledge of the object-oriented paradigm will speed up the learning process and will enable learners to better appreciate UML.
3. *Prevent interference from knowledge of structured paradigm.* Learners who have been exposed to the structured paradigm should be cautioned not to transfer knowledge from the structured paradigm to the object-oriented paradigm. Prior knowledge from the structured paradigm will interfere with the learning of UML and make the learning process much more difficult and confusing.

CONCLUSIONS

Undoubtedly, UML is large and complex, and it is going to continue to grow! The number of diagramming techniques increases from 9 in UML 1.x to 13 in UML 2.0. The complexity of UML poses a problem for learning the modeling language.

This study used a concept mapping technique to identify the difficulties in learning UML. Two categories of problems were identified. The first category can be traced to the inherent problems in UML. These problems have been highlighted by some practitioners and researchers, but empirical evidence was previously absent. The second category, peripheral issues in learning UML, relates to the prior knowledge of the learners and the problems of UML training materials.

This study is of interest to both UML researchers and practitioners. It highlights the difficulties in learning UML and relates some of these difficulties to the inherent problems in UML. For practitioners and educators, this study provides recommendations on ways to facilitate UML learning and ease the learning process. For UML researchers and designers, some of the inherent problems in UML can be addressed in future releases of UML.

ACKNOWLEDGMENTS

The authors would like to express their sincere gratitude to the guest editor, Dinesh Batra, the editor-in-chief, Carol Brown, and the anonymous reviewers for their effort and time in assisting the authors to revise and enhance the manuscript. ▲

References

Anderson, J. R. (1976). Language, Memory and Thought. Hillsdale, NJ: Erlbaum Associates.

Anderson, J. R. (1989). Practice, working memory, and the ACT* theory of skill acquisition: A Comment on Carlson, Sullivan, and Schneider. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 15*, 527–530.

Anderson, J. R. (1993). *Rules of the Mind*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Anderson, J. R. (1996). ACT: A simple theory of complex cognition. *American Psychologist, 51*, 355–365.

Booch, G. (1999). UML in Action, *Communications of the ACM*, 42(10), 26–28.

Booch, G., Rumbaugh, J., and Jacobson, I. (2005). *The Unified Modeling Language User Guide, Second Edition,* Addison Wesley.

Dobing, B., and Parsons, J. (2000). Understanding the Role of Use Cases in UML: A Review and Research Agenda, *Journal of Database Management*, 11(4), 28–36.

Dori, D. (2002). Why Significant UML Change is Unlikely, *Communications of the ACM*, 45(11), 82–84.

Duddy, K. (2002). UML2 Must Enable A Family of Languages, *Communications of the ACM*, 45(11), 73–75.

DuÈ, R. T. (1993). Object-oriented technology: The economics of a new paradigm. *Information Systems Management,* 10(3), 69ñ73.

Langley, P., and Simon, H. A. (1981). *The central role of learning in cognition.* In J. R. Anderson (Ed.) *Cognitive Skills and Their Acquisition* (pp. 361-380). Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Long, H. B. (1990). *Understanding Adult Learners. In M. W. Galbraith, Adult Learning Methods: A Guide for Effective Instruction*. Malabar, Florida: Robert E. Krieger Publishing Company.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63*, 81–97.

Pennington, N., Lee, A. Y., and Rehder, B. (1995). Cognitive activities and levels of abstraction in procedural and object-oriented design. *Human-Computer Interaction, 10*, 171ñ226.

Price, R., Tryfona, N., and Jensen, C. (2000). Extended Spatiotemporal UML: Motivations, Requirements, and Constructs, *Journal of Database Management*, 11(4), 14–27.

Selic, B., Ramackers, G., and Kobryn, C. (2002). Evolution, Not Revolution, *Communications of the ACM*, 45(11), 70–72

Shuell, T. J., and Lee, C. Z. (1976). *Learning and Instruction*. Monterey, California: Brooks/Cole.

Siau, K. (1999). Information Modeling and Engineering: A Psychological Perspective, *Journal of Database Management*, 10(4), 44–50.

Siau, K., and Cao, Q. (2001). Unified Modeling Language — A Complexity Analysis, *Journal of Database Management*, 12(1), 26–34.

Siau, K., and Lee, L. (2004). Are Use Case and Class Diagrams Complementary in Requirements Analysis? — An Experimental Study on Use Case and Class Diagrams in UML, *Requirements Engineering*, 9(4), 229–237.

Siau, K., and Loo, P. (2002). Difficulties in Learning UML: A Concept Mapping Analysis, *Seventh CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'02)*, Toronto, Canada, 102–108.

Siau, K., Erickson, J., and Lee, L. (2005). Theoretical versus Practical Complexity: The Case of UML, *Journal of Database Management*, 16(3), 40–57.

Simons, A. J. H. (1999). Use cases considered harmful. Proc. 29th Conf. Tech. Object-Oriented Programming Language and System, (TOOLS-29 Europe), eds. R Mitchell, A C Wills, J Bosch and B Meyer (Los Alamitos, CA: IEEE Computer Society), 194–203.

Simons A. J. H., and Graham, I. (1999). *30 Things That Go Wrong In Object Modelling With UML 1.3 Chap. 16*. In: Kilov H, Rumpe B and Simmonds I (ed.) *Precise Behavioral Specification of Businesses and Systems.* Kluwer Academic Publishers.

Surveyer, J. (1999/2000, Winter). *Java and UML.* [Online] Available: http://www.devx.com/upload/free/features/javapro/1999/10mid99/js1399/js1399.asp

Trochim, W. (1989a). An introduction to concept mapping for planning and evaluation. *Evaluation and Program Planning*, 12(1), 1–16.

Trochim, W. (1989b). Concept mapping: Soft science or hard art? *Evaluation and Program Planning,* 12(1), 87–110.

Tyson, K., and Frank, W. (2002). Be Clear, Clean, Concise, *Communications of the ACM*, 45(11), 79–81.

Whittle, J. (2000). Formal Approaches to Systems Analysis Using UML: An Overview, *Journal of Database Management*, 11(4), 4–13.