Guest Editorial Preface

Unified Modeling Language (UML) Topics:

The Past, the Problems, and the Prospects

Dinesh Batra, Florida International University, USA

Unified modeling language (UML) is a modeling language sponsored by a prominent consortium of companies, the object management group (OMG), for object-oriented (OO) systems development (Kobryn, 1999, 2004). It is widely acknowledged that UML has become the de facto standard for modeling object-oriented software development (Pender, 2003). It is used in specifying, visualizing, constructing, and documenting the artifacts of software-intensive systems (OMG, 2005). Although UML is popularly used in object-oriented software development, it is designed to be a general-purpose language with a rich set of diagrammatic notations that can model any type of software application. In a project lifecycle, it can be used for requirement determination, analysis, and design, and can serve as a basis for coding (Bell, 2004; George, Batra, Valacich, & Hoffer, 2007). It is supported by powerful tools such as the ones based on the rational unified process (RUP). The adoption rate of UML and its related tools has even exceeded OMG's most optimistic predictions (Selic, Ramackers, & Kobryn, 2002). It appears UML is here to stay.

There are a number of interesting research areas on UML topics, such as usability concerns (Agarwal & Sinha, 2003), cognitive matters (Sin & Batra, 2007), complexity issues (Siau & Cao, 2001), Web applications (Conallen, 1999), data modeling and warehousing aspects (Trujillo, Lujan-Mora & Song, 2004), modeling constraints (Warmer & Kleppe, 2003), model driven architecture (Kleppe, Warmer & Bast 2003), domain modeling (Rosenberg & Scott, 2001), component-based development (Dahanayake, Sol & Stojanovic, 2003), object constraints (Warmer & Kleppe, 2003), UML's role in agile methodologies (Ambler, 2004), ontology concerns (Evermann & Wand, 2006), training issues (Siau & Loo, 2006), and curriculum (Batra & Satzinger, 2006). The articles in this and forthcoming special issues on UML topics attempt to address some of these research areas. This is the first of two special issues that are being published in the *Journal of Database Management*, which has been one of the key journals that have covered a variety of UML topics in recent years.

The first article in the issue, "Dimensions of UML Diagram Use: A Survey of Practitioners," by Brian Dobing and Jeffrey Parsons, employs the survey research strategy to investigate how UML is used in practice. The authors report that there is considerable variety in the ways the

UML diagrams are used in different projects. Overall, they found that the class diagram, use case diagram, and the sequence diagram are the most frequently used, while the written use case is used less often. The authors, therefore, challenge the "use-case driven" assumption. They report a number of interesting findings related to UML usage based on organization size, project size, respondent experience, UML tools' availability, industry, and system type. For instance, contrary to widespread belief, clients frequently approve, review, and even help develop UML diagrams.

The second article in the issue, "Conflicts, Compromises, and Political Decisions —Methodological Challenges of Enterprise-Wide E-Business Architecture Creation," by Kari Smolander and Matti Rossi, reports the inadequacies of UML in certain organizational aspects of e-business architecture creation. They employed a grounded-theory approach to study an implementation that involved the integration of data and legacy systems from independent business units to create a uniform Web-based customer interface. They found that the e-architecture emerges through somewhat non-deliberate actions dictated by the situation and its constraints, conflicts, compromises, and political ramifications. They classify 13 issues in this context into technical, language, and organization requirements, and rate the effectiveness of UML relative to each issue. The study needs to be viewed with the understanding that although UML is only a specification that defines the grammar of the modeling language, it has been adopted by a number of methodologies such as the RUP, which are expected to address organizational and project management issues (Kruchten, 2003).

The third article, "BROOD: Business Rules-driven Object Oriented Design," by Pericles Loucopoulos and Wan Kadir, employs UML as the modeling framework and addresses the important issue of representing business rules from the early requirements stage, expressed in user-friendly terms, to downstream system design components, and maintaining these throughout the life cycle of the system. Any user-oriented changes can then be traced, and if necessary propagated from requirements to design specifications and evaluated by both end-users and developers regarding the changes impact on the system. The BROOD approach aims to provide seamless traceability between requirements and system designs through the modeling of business rules and the successive transformations. To illustrate their approach, the authors describe a tool that provides a high degree of automation in supporting the propagation of changes from requirements to designs. The usefulness of BROOD is demonstrated by employing it in an application from healthcare.

The fourth article, "Enhancing UML Models: A Domain Analysis Approach," by Iris Reinhartz-Berger and Arnon Sturm, reports on a laboratory experiment that evaluates the effectiveness of application-based domain modeling, a domain analysis approach. When developing a particular application in the domain, the domain model is used as a reference, and the application model can be validated against the relevant domain model in order to detect completeness, and correctness errors. The results suggest that the availability of the domain model helps produce more complete models without reducing the comprehensibility of these models.

These are the four articles included in the current issue. In the next special issue on UML, we expect to publish an equally diverse group of articles covering a range of topics. The call for articles released in 2006 resulted in a large number of submissions, and, after a round of reviews, it was decided to have two special issues. The interest generated is a good indicator of the popularity of UML in research and industry. Therefore, It is not surprising, that UML has become important in training and education. In response to the trend of UML's growing popularity in practice, many undergraduate and graduate programs in information systems, information technology, and computer science areas have adopted UML as the primary object modeling language in their SA&D courses (Batra & Satzinger, 2006). But despite the popularity of UML, it has several shortcomings that are beginning to be revealed and researched.

Some researchers claim that the UML specifications have become considerably bulky with the release of the major revision in 2003, UML version 2.0 (Kobryn, 2004). The number of diagrams in UML 2.0 has been augmented to 13 from 9 in UML 1.0. Most of the diagrams added in UML 2.0 depict higher level views of the system under development to enhance component-based development (Kobryn, 2002). However, the ideas behind the original UML diagrams have not changed much and are expected to remain the same for the foreseeable future (Dori, 2002; Miller, 2002; Selic, Ramackers, & Kobryn, 2002).

Researchers had complained about the large number of diagrams even in UML 1.0 versions; UML 2.0.has even more This problem has resulted in a number of nagging concerns and criticisms of the language. Some obvious research issues include the way in which the diagrams relate to each other, whether the dependencies can be clearly mapped out, and whether traceability can be achieved. There is some research on these topics. For example, Selonen, Koskimies, and Sakkinen (2003) provide four categories of transformations of operations from one diagram to another: full transformations, strong transformations, supported transformations, and weak transformations. This kind of research remains limited, however, and there is a need for more investigation.

Some redundancy and overlap in notations has resulted from incorporating the diverse interests of the major stakeholders during the standardization process (Kobryn, 1999). For example, two of the interaction diagrams—the sequence diagram and the collaboration diagram—basically employ the same set of notations to describe the same phenomenon in different arrangements, which is often considered redundant. Hence, UML is often criticized for introducing a large number of competing constructs (Hesse, 2001), increasing the complexity of the language (Siau & Tian, 2001).

With more than 150 distinct constructs (Dori, 2002), UML is indeed a large and complex language. Dori (2002) claims that UML requires several models to completely specify a system, and is, therefore, low in usability. Dori (2001) has proposed the object-process methodology to provide a simpler methodology that provides a single, integrated graphic model. It achieves model integration by incorporating the three major system aspects—function, structure, and behavior—into a single model in which both objects and processes are adequately represented without either suppressing the other. Some studies have concluded that it is difficult to model a correct and consistent application using UML, as well as to understand such a specification (Peleg & Dori, 2000).

UML is often criticized for its poorly-defined syntax and semantics, and its spatial layout (Tilley & Huang, 2003). For example, some UML notations are often shown in a number of ways, which can easily confuse a novice analyst, who has very little experience. However, the predominant usability concern is its structural complexity.

Siau and Cao (2001) analyzed the complexity of UML in comparison with other OO methods by using Rossi and Brinkkemper's (1996) structural complexity metrics, which are primarily based on the structural properties of modeling constructs such as the number of objects, associations, and properties in a diagram. The complexity measures of the important UML diagrams in UML 1.x are: 26.40 for class diagrams, 10.39 for use case diagrams, 11.18 for activity diagrams, 7.87 for sequence diagrams, 8.12 for collaboration diagrams, 5.92 for object diagrams, 15.39 for state chart diagrams, 15.65 for component diagrams, and 9.95 for deployment diagrams. When these measures are put together, UML is revealed to be two to 11 times more complex than other OO methods (Siau & Cao, 2001). To be fair, however, the practical complexity of UML is estimated to be lower than the above theoretical complexity measures (Siau, Erickson, & Lee, 2005).

It has been conjectured that the structural complexity of UML increases the neural load on systems analysts developing UML diagrams. The mental burden or increased neural load on users caused by the design characteristics of an artifact such as UML is called the cognitive complexity (Reeves, 1999, p. 18). The structural complexity can serve as a surrogate for the cognitive complexity (Siau et al., 2005). Time has

been used as a measure of general complexity by Zendler, Pfeiffer, Eicks, and Lehner (2001), who showed that the coarse-grained concepts of some object-oriented approaches are superior to those of UML when modeling a database-oriented application.

Cognitive complexity may provide a theoretical basis to study the usability of UML. The theories of cognitive complexity have been applied to software engineering processes from coding or programming (Cant, Jeffery, & Henderson-Sellers, 1995) to OO system design (Rosson and Alpert, 1990). Cognitive complexity principles are also applied to other areas of research such as the design of user interfaces for information-intensive products (Reeves, 1999) and computer-based learning systems (Norman & Spohrer, 1996; Soloway & Pryor, 1996).

Usability research has used methods like the GOMS model (Card, Moran, & Newell, 1983) to measure the cognitive complexity. The GOMS model attempts to represent specific human problem solving behavior in terms of goals, operators, methods, and selection rules after Newell and Simon's (1972) information processing approach. Siau and Tian's (2001) study employed GOMS, a cognitive approach, to analyze the complexity of UML and had results similar to those of Siau and Cao's (2001) study, which followed a structural approach, thereby showing the association between the structural and cognitive complexity. One issue with the GOMS model is that its successful application requires a certain amount of expertise in the subject's areas being evaluated. If the scope of a study is limited to novice analysts with little or no expertise, the GOMS model may be difficult to use.

UML has also not been rated highly in behavioral measures like ease-of-use, although one study provides moderate support. Agarwal and Sinha (2003) used ease-of-use measures derived from Davis (1989) to assess and compare the usability of important UML diagrams: use case, class, state, and sequence diagrams. Mean ease-of-use scores of the diagrams tested were 5.064 for the use case diagram, 4.295 for the class diagram, 5.301 for the state diagram, and 4.487 for the interaction diagram (sequence or collaboration diagram). All mean scores except that of the interaction diagram were found to be significantly higher than four which was the indifference score, indicating that subjects perceived use case, class, and state diagrams as easy to use. On the other hand, none of the diagrams were rated very highly by subjects. Researchers also discovered that prior knowledge in OOSA&D techniques had significant effects on subjects' perception of the ease of use of diagrams, which is consistent with Siau and Loo's (2006) findings.

A related issue is that empirical research so far indicates that UML may be a difficult language to learn. Siau and Loo (2006) used the concept mapping technique to identify two categories of programs in UML, namely inherent and peripheral problems. Inherent problems are related to (1) inconsistencies in UML diagrams and constructs, (2) unclear definitions of UML semantics, and (3) the large number of UML constructs. Peripheral problems are related to (1) training materials, (2) software tools, and (3) prior knowledge. These problems are believed to cause difficulties in novice analysts' learning process. Improving the learnability of UML diagrams by addressing the UML's inherent problems is the responsibility of the UML standards group. On the other hand, designing and proposing better, more effective training or modeling techniques to make UML easier to learn is the purpose of research studies.

Many suggestions have been made to improve UML (Halpin, 2001, 2002). Shoval and Kabeli (2001) remark that one problem with UML is that there is no methodology that ties the diagrams together. They suggest a methodology called FOOM (functional and object-oriented analysis and design methodology), which tries to blend data flow diagram (DFD), entity relationship diagram (ERD), and object-oriented (OO) constructs. Dori (2001) has suggested the object-process methodology, which has a similar goal of tying diagrams together into a parsimonious representation.

Researchers, thus, justifiably claim that UML suffers from a number of limitations. UML has been found to be a difficult language for novices because there seem to be too many diagrams. Although the UML grammar is well

known, precise techniques for training novice analysts are missing. For example, although the syntax of a sequence diagram is quite obvious, there is practically no research explaining how it can be taught to novice designers. The usability of many other diagrams is largely unexplored. There is a need to model business constraints in a natural way. Contrary to the initial expectations, the object constraint language (OCL) has not gained popularity, and its usability seems to be questionable. MIS textbooks on systems analysis and design typically do not cover OCL. There has been little research in the organizational capabilities of UML. Popular methodologies like the RUP have not been well researched in the organizational context by academic researchers.

There are other important issues with UML that pertain to development environment. One potential area of research is the role of UML in component-based development (CBD). Dahanayake et al. (2003) have evaluated existing CBD methods and concluded that current methods do not consider component as a life cycle concept; instead components are handled at the implementation and deployment phases. They conclude that a more formal and systematic CBD approach is needed. In fact, they recommend a common CBD language that can be used throughout the life cycle for integration of different services, concepts, and perspectives.

UML has been applied in several recent and emerging domains such as designing data warehouses and OLAP applications (Trujillo et al., 2004). Extensions have been proposed to employ UML for space- and time-dependent applications (Price, Trfona, & Jensen, 2002). For software-intensive system applications, SysML (see http://www.sysml.org/) has been proposed.

Although the technical aspects of UML are being extended, the focus of researchers, as evidenced by the articles included in this issue, is on usability, organizational issues, and how UML is actually used in practice. Despite all of these issues, the adoption of UML and its tools has been widespread (Selic et al., 2002).

This special issue includes papers that have attempted to address some of the previously- mentioned issues. A large number of high-quality manuscripts were received for the special issue. Each submitted manuscript was sent out to three reviewers. Since the accepted articles cannot fit into one special issue, a second issue on UML topics is forthcoming. Each of the four accepted articles in this special issue went through one round of comprehensive revision based on reviewer comments. I thank the reviewers, most of whom are members of the AIS special interest group on systems analysis and design (SIGSAND), for their meticulous reading of the papers, and the substantive comments that improved the quality of the papers. I also thank the editor-in-chief, Keng Siau, who counseled me on various matters. I am confident that the readers of the journal will find the papers of high quality, and I hope this will lead to future research in this area of considerable contemporary interest.

REFERENCES

Agarwal, R. & Sinha, A. P. (2003). Object-oriented modeling with UML: A study of developers' perceptions. *Communications of the ACM, 46*(9), 248-256.

Ambler, S.W. (2004). *The object primer: Agile model-driven development UML 2.0* (3rd ed.). Cambridge, UK: Cambridge University Press.

Batra, D. & Satzinger, J. W. (2006). Contemporary approaches and techniques for the systems analyst. *Journal of Information Systems Education, 17*, 257-266.

Bell, A. E. (2004). Death by UML fever. *Queue, 2,* 72-80.

Cant, S. N., Jeffery, D. R., & Henderson-Sellers, B. (1995). A conceptual model of cognitive complexity of elements of the programming process. *Information and Software Technology, 37*(7), 351.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Conallen, J. (1999). Modeling web application architectures with UML. *Communications of the ACM, 42*(10), 63-70.

Dahanayake, A., Sol, H., & Stojanovic, Z. (2003). Methodology evaluation framework for component-

based system development. *Journal of Database Management*, *14*(1), 1-26.

Davis, F.D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly, 13*(3), 319-340.

Dobing, B., & Parsons, J. (2002). The role of use cases in the UML: A review and research agenda. In K. Siau (Ed.), *Advanced topics in database research* (pp. 367-382). Hershey, PA: IGI Publishing.

Dobing, B., & Parsons, J. (2008). Dimensions of UML diagram use: A survey of practitioners. *Journal of Database Management*, *19*(1), 1-18.

Dori, D. (2001). Object-Process Methodology applied to modeling credit card transactions. *Journal of Database Management*, *12*(1), 4-14.

Dori, D. (2002). Why significant UML change is unlikely. *Communications of the ACM, 45*(11), 82-85.

Evermann, J., & Wand, Y. (2006). Ontological modeling rules for UML: An empirical assessment. *Journal of Computer Information Systems*, *47*(1).

George, J.F., Batra, D., Valacich, J. S., & Hoffer, J. A. (2007). *Object-oriented systems analysis and design* (2nd Ed.). Upper Saddle River, NJ: Prentice Hall.

Halpin, T. (2001). Supplementing UML with concepts from ORM. In K. Siau & T. Halpin (Eds.), *Unified modeling language: Systems analysis, design and development issues* (pp. 61-74). Hershey, PA: Idea Group Publishing.

Halpin, T. (2002). Information analysis in UML and ORM: A comparison. In K. Siau (Ed.), *Advanced topics in database research* (pp. 307-323). Hershey, PA: Idea Group Publishing.

Hesse, W. (2001). RUP: A process model for working with UML. In K. Siau & T. Halpin (Eds.), *Unified modeling language: Systems analysis, design and development issues* (pp. 61-74). Hershey, PA: Idea Group Publishing.

Kleppe, A.G., Warmer, J.B., & Bast, W. (2003). *MDA explained—the model driven architecture: Practice and promise*. Boston, MA: Addison-Wesley.

Kobryn, C. (1999). UML 2001: A standardization odyssey. *Communications of the ACM, 42*(10), 29-37.

Kobryn, C. (2002). Will UML 2.0 be agile or awkward? *Communications of the ACM, 45*(1), 107–110.

Kobryn, C. (2004). UML 3.0 and the future of modeling. *Software & Systems Modeling, 3*(1), 4-8.

Kruchten, P. (2003). *The rational unified process: An introduction* (3rd ed.). Boston, MA: Addison-Wesley Professional.

Loucopoulos, P., & Kadir, W. (2008). BROOD: Business rules-driven object oriented design. *Journal of Database Management*, *19*(1), 41-73.

Miller, J. (2002). What UML should be. *Communications of the ACM, 45*(11), 67-69.

OMG. (2005). Introduction to OMG's unified modeling language (UML). Retrieved August 2007.

Peleg, M., & Dori, D. (2000). The model multiplicity problem: Experimenting with real-time specification methods. *IEEE Transactions of Software Engineering*, *26*(8), 742-759.

Pender, T. (2003). *UML bible.* Indianapolis, IN: Wiley Publishing.

Price, R., Trfona, N., & Jensen, C.S. (2002). Extending UML for space- and time-dependent applications. In K.Siau (Ed.), *Advanced topics in database research* (pp. 342-366). Hershey, PA: Idea Group Publishing.

Reeves, W.W. (1999). *Learner-centered design: A cognitive view of managing complexity in product, information, and environmental design*. Thousand Oaks, CA: Sage Publications.

Reinhartz-Berger, I., & Sturm, A. (2008). Enhancing UML models: A domain analysis approach. *Journal of Database Management*, *19*(1), 74-94.

Rosenberg, D., & Scott, K. (2001). *Applying use case driven object modeling with UML: An anotated e-commerce example.* Addison-Wesley, Boston.

Rossi, M., & Brinkkemper, S. (1996). Complexity metrics for systems development methods and techniques. *Information Systems, 21*(2), 209-227.

Rosson, M.B., & Alpert, S. R. (1990). The cognitive consequences of object-oriented design. *Human-Computer Interaction, 5*(4), 345-379.

Selic, B., Ramackers, G., & Kobryn, C. (2002). Evolution, not revolution. *Communications of the ACM, 45*(11), 70-72.

Selonen, P., Koskimies, K., & Sakkinen, M. (2003). Transformations between UML diagrams. *Journal of Database Management, 14*(3), 37-55.

Shoval, P., & Kabelli, J. FOOM: Functional- and object-oriented analysis & design of information systems: An integrated methodology. *Journal of Database Management*, *12*(1), 15-25.

Siau, K., & Cao, Q. (2001). Unified modeling language (UML)—A complexity analysis. *Journal of Database Management, 12*(1), 26-34.

Siau, K., Erickson, J., & Lee, L. Y. (2005). Theoretical vs. practical complexity: The case of UML. *Journal of Database Management, 16*(3), 40-57.

Siau, K., & Loo, P.-P. (2006). Identifying difficulties in learning UML. *Information Systems Management, 32*(3), 43-51.

Siau, K., & Tian, Y. (2001). *The complexity of unified modeling language: A GOMS analysis.* In the Proceedings of International Conference on Information Systems, New Orleans, LA.

Sin, T., & Batra, D. (2007, May 12-13,). *Improving usability of analysis sequence diagram in transaction-oriented applications*. Symposium on Systems Analysis and Design. Tulsa, Oklahoma.

Smolander, K., & Rossi, M. (2008). Conflicts, compromises, and political decisions—Methodological challenges of enterprise-wide e-business architecture creation. *Journal of Database Management*, *19*(1), 19-40.

Soloway, E., & Pryor, A. (1996). The next generation in human-computer interaction. *Communications of the ACM, 39*(4), 16-18.

Tilley, S., & Huang, S. (2003). *A qualitative assessment of the efficacy of UML diagrams as a form of graphical documentation in aiding program understanding.* Paper presented at the 21st Annual International Conference on Documentation. San Francisco, CA.

Trujillo, J., Lujan-Mora, S., & Song, I. (2004). Applying UML and XML for designing and interchanging information for data warehouses and OLAP warehouses. *Journal of Database Management*, *15*(1), 41-72.

Warmer, J., & Kleppe, A. (2003). *The object constraint language: Getting your models ready for MDA*. The Addison-Wesley Object Technology Series. Addison-Wesley, 2nd edition, August 2003.

Zendler, A., Pfeiffer, T., Eicks, M., & Lehner, F. (2001). Experimental comparison of coarse-grained concepts in UML, OML, and TOS. *Journal of Systems and Software, 57*(1), 21-30.

Dinesh Batra is a Knight-Ridder research professor in the Department of Decision Sciences and Information Systems in the College of Business Administration at the Florida International University. Dr. Batra has published articles in journals such as Management Science, Communication of the ACM, Journal of MIS, International Journal of Human Computer Studies, Data Base, European Journal of Information Systems, Journal of Database Management, Communications of the AIS, Decision Support Systems, Requirements Engineering, Computers and OR*, and* Information & Management*. His research interests focus on systems analysis and design, and usability issues in systems and databases. He has served as an associate editor in the J*ournal Data Base*, and is currently an associate editor in the* Journal of Database Management*, and* Information Systems Management*. He is a co-author of the book* Object-Oriented Systems Analysis and Design*. He has served as a president of the AIS Special Interest Group on Systems Analysis & Design (SIGSAND).*