

Usability of Analysis Sequence Diagram

Thant Sin
Florida International University
Thant.Sin@fiu.edu

Abstract

The Unified Modeling Language (UML) has become a widely-accepted modeling language for object-oriented software development since it became the industry standard in 1997. Although researchers and practitioners have extensively studied various aspects of UML in the last couple of years, they have not yet explored the usability of the analysis sequence diagram. Drawing analysis sequence diagrams is often found to be very challenging for students in systems analysis and design classes even though basic guidelines are provided in popular textbooks. Given the important role of analysis sequence diagrams in object-oriented systems analysis and design, there is the need for an effective sequence diagram modeling technique that can facilitate novices in developing these diagrams. This paper presents a research framework which is developed based on difficulties students encounter in drawing sequence diagrams, identified in the qualitative exploratory study conducted in an undergraduate systems analysis and design class. The proposed framework will enable the design of a modeling technique and laboratory experiments to evaluate the efficacy of the new technique.

Introduction

Describing various aspects of an information system is one of the most important activities in systems development. Systems analysts and designers use these models to communicate system specifications with other stakeholders. Effective communication requires a common set of notations that stakeholders can understand. This is the primary role of modeling languages such as the Unified Modeling Language (UML) in software development.

UML is a language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems (OMG, 2003). The Object Management Group (OMG) adopted the UML version 1.1 as the standard modeling language for object-oriented (OO) software development in November 1997. Since that time on, UML has quickly gained a stronghold in the software development industry. “No other technology has so quickly and deeply permeated the software engineering life cycle quite like UML” (Bell, 2004). Consequently, UML has become one of the most important skills systems analysts and designers need to possess in addition to OO software development methodologies to be able to remain competitive in today’s highly competitive job market which is now opened up to global competition mainly driven by growing popularity of offshore software development.

In response to this trend, many undergraduate and graduate programs in information systems, information technology, and computer science areas have adopted UML as the primary modeling language in their systems analysis and design (SA&D) courses and are using it to de-

velop object-oriented (OO) systems (Batra & Satzinger, 2006). As a result, UML has become very important not only in the software development industry but also in training and education.

However, there seems to have been little progress in proposing effective techniques that employ the UML grammar. In fact, there has been little research in improving the usability of the behavioral aspects of UML. Specifically, there is practically no literature addressing techniques aimed at novice analysts engaged in developing sequence diagrams. This greatly contrasts with the extensive amount of empirical work done in the area of data modeling (see Topi and Ramesh (2002) for a detailed survey).

A number of systems analysis and design textbooks employing UML (e.g., George *et al*, 2004) have appeared in the market. Recommendations provided in textbooks to develop sequence diagrams do not seem to be very helpful for novices like students. These recommendations are more likely to be useful for experts with prior experience. Hence, the usability of the sequence diagram is a significant research gap.

The purpose of this paper is to attempt to address this research gap by providing guidelines for training novice designers engaged in developing sequence diagrams. This includes information systems undergraduate and graduate students taking the first (and usually the only) course in systems analysis and design. The guidelines are based on novice designer difficulties observed in an experiment conducted in summer 2006.

This paper is structured as follow. The relevant literature is reviewed in next section which is followed by research questions and hypotheses. Then, the research method and the details of the exploratory study are presented. Finally, the paper ends with the formulation of the research framework and the conclusion section.

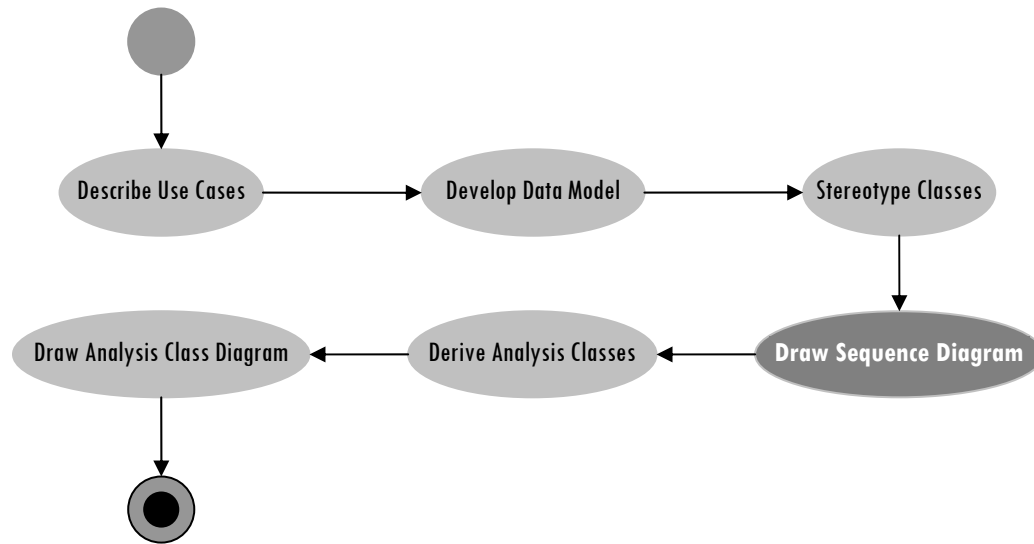
Literature Review

Object-oriented systems analysis and design (OOSA&D) primarily employ use case diagrams (together with corresponding use case descriptions), sequence diagrams, and class diagrams to describe the analysis model of a software system (George et al, 2004). Important roles of these diagrams in SA&D are also confirmed by a recent empirical study where they were found to be the three most commonly used UML diagrams in practice (Dobing & Parsons, 2006).

These three UML diagrams portray the vital aspects of the system. Structural or static aspects of the system are mainly depicted in the class diagram which shows relationships among object classes whereas behavioral or dynamic aspects of the system are captured in the use case diagram at a high level and in the sequence diagram at a detailed level. The use case diagram is likely to be used at the requirements phase while the sequence diagram at analysis and design phases. The sequence diagram is typically used to describe interactions among object classes.

Figure 1 illustrates roles of these UML artifacts in a stepwise process to develop important analysis models (George et al, 2004). The analysis class diagram is often considered the major deliverable of the systems analysis process. In contrast to design models that reflect various implementation decisions such as choice of programming language and hardware/software platforms, analysis models are mainly used to capture domain-level system specifications (Booch *et al*, 2005). Focus on analysis models attenuates the influence of programming language constructs on the quality of software models developed by novices with different backgrounds.

Figure 1 Stepwise technique to develop the analysis class diagram



Source: George et al (2004)

Use cases which portray system requirements captured in requirements determination are the primary driver of OOSA&D (Jacobson *et al*, 1999). A use case represents a particular functionality or functional requirement of a software system. There are two forms of use case deliverables: use case diagram and written use case description. Use case diagrams, derived from system requirements, depict interactions between use cases and relevant actors together with relationships among use cases. Developing use case diagrams is more of an art than science. Although requirements determination is a very complex process (Browne & Ramesh, 2002), deriving use case diagrams from requirements is a relatively simple process that involves identifying major functionalities and relationships among them, together with actors they interact with and putting them together in graphical format.

Describing individual use cases in detailed textual format is the first step in the stepwise process to develop the analysis class diagram. The most important piece of information included in a written use case is the flow of events that describes the main success scenario of actions re-

quired for successful completion of the use case. There are guidelines or approaches available to follow in developing use case descriptions from system requirements (Rolland & Achour, 1998). In addition, this area of research has also benefited from research in scenario construction (Leite *et al*, 2000) and management (Jarke *et al*, 1998). There are also templates available to use in writing use case descriptions as well (Cockburn, 1997, 2000). Therefore, novices have plenty of resources available in performing this activity.

Data modeling is also a well-studied research area where well-defined guidelines and techniques are readily available for novices. Entity-relationship (ER) modeling for conceptual data modeling and relational data modeling for logical data modeling have widely been used in practice (Batra & Zanakis, 1994). Prior research has investigated relationships among data modeling formalisms, user characteristics, task characteristics, and performance by following the framework for human factors research on data modeling (Batra *et al*, 1990; Topi & Ramesh, 2002). Generally, positive outcomes are often identified with ER and extended ER (EER) models. Techniques for modeling these popular models have also been widely taught in training and education as well. Hence, novices may find themselves well-equipped for the data modeling task.

The sequence diagram is a UML diagram that portrays the behavioral aspects of a system in conjunction with the use case diagram. Dobing and Parsons (2006) discover that the sequence diagram is more commonly used in practice than the collaboration diagram. Common practices described in the literature identify the sequence diagram as the major UML diagram that captures the detailed behavior of object classes in the system (George *et al*, 2004). In addition, prior research has also shown that errors in the sequence diagram are one of the major sources of defects in the final product (Agarwal & Sinha, 2003; Bolloju & Leung, 2006). Therefore, drawing the sequence diagram correctly is often considered very important in OOSA&D.

However, the importance of the sequence diagram as the primary means of modeling analysis classes' behavior has been largely overlooked in the current research stream. Consequently, there is practically no formal approach to modeling sequence diagrams. There are only a few recommendations prescribed in some textbooks (George et al, 2004). Anecdotal experience in teaching UML to undergraduate and graduate students in OOSA&D classes informs that these recommendations are of limited help for students. This suggests that there is an urgent need for an effective modeling technique that can facilitate novices in drawing sequence diagrams.

Developing analysis classes and the class diagram is a relatively straightforward process if correct data model and sequence diagram have been developed. Attributes of analysis classes are derived from entity classes in the data model while operations are modeled based on responsibilities and messages described in the sequence diagram. It is clear that to be able to develop analysis classes and the corresponding class diagram correctly, it is necessary to have the data model and sequence diagram modeled correctly beforehand.

From the above review of the literature, lack of research on the usability of the analysis sequence diagram is found to be the missing link in the stream of research on the usability of UML diagrams in SA&D. This is the research gap that needs to be addressed.

Research Questions & Hypotheses

Prior research has identified that UML in general is 2 to 11 times more complex than other OO techniques (Siau & Cao, 2001). This structural complexity of UML also causes the cognitive complexity of UML (Siau *et al*, 2005). The cognitive complexity is related to human cognition

and perception in contrast to the structural complexity which is related to the structural properties of diagrams and constructs. *Is developing the analysis sequence diagram a cognitive complex task for novices?* The qualitative exploratory study was conducted to provide some clue to answer this question. Observing students in OOSA&D classes suggests that developing sequence diagrams is indeed a difficult task.

Then, *how can the cognitive complexity involved in drawing sequence diagrams be addressed?* A research framework is developed to answer this question. It will also enable the design of a modeling technique that can facilitate novices in drawing sequence diagrams. The modeling technique should be able to address the cognitive complexity that may have caused difficulties in the exploratory study. Meanwhile, the resulting technique should promote the usability of the sequence diagram for novices. *Does the proposed modeling technique promote the usability of the sequence diagram?* The answer to this question will be observed in laboratory experiments that will measure the efficacy of the technique by evaluating performance and perception of student subjects in using the new technique.

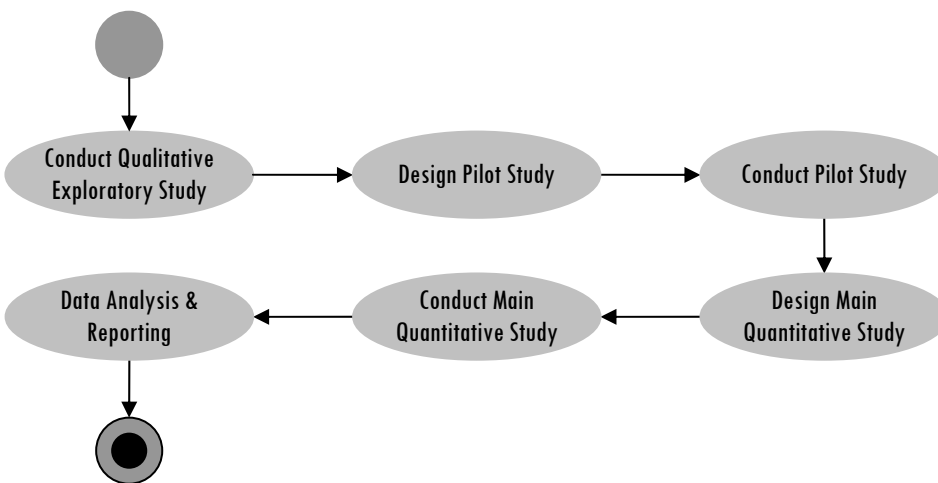
Research Methodology

In general, this research study employs a mixed method that contains both qualitative and quantitative methods (Creswell, 2003). Qualitative methods have been used in “identifying unanticipated phenomena and influences, and generating new grounded theories about the latter” (Maxwell, 1996). Information systems researchers have also employed qualitative research methods in their studies to help them develop questionnaires and identify variables for experimental studies (Kaplan & Duchon, 1988). Furthermore, qualitative methods are considered most

appropriate when organizational or human aspects of information systems are involved (Myers, 1997).

A qualitative exploratory study was conducted in summer 2006 to gain insights into difficulties novices encounter in drawing sequence diagrams. Based on these insights, a sequence diagram modeling technique can be developed. The exploratory study also enables the design of future experimental studies to evaluate the efficacy of the proposed modeling technique. Figure 2 shows the preliminary design of the research study.

Figure 2 Design of the research study



Exploratory Experiment

The qualitative exploratory study was conducted in an undergraduate OOSA&D class in the summer of 2006. The study included a sequence diagram drawing task which was followed by brief interviews with individual subjects within a few days. Subjects were randomly assigned to either the control or treatment group. Although most subjects in both groups attended regular class sessions on data modeling and sequence diagram concepts, those in the treatment group

were given additional training on data modeling. The reason was to verify the role of data modeling in sequence diagram drawing process. Interview questions were customized based on mistakes individual subjects committed in their solutions. Experts' judgment on subjects' solution together with feedbacks in follow-up interviews were used to design the research framework.

Table 1 Research framework

Observation	Possible Cause	Complexity Factor	Complexity Source
Lack of an overall strategy and inability to break down the procedure into discrete steps	There are too many choices, which seem to overwhelm the designers	Variety	Information Overload
	The breadth of the task overwhelm designers	Large/Long tasks	Design Complexity
Not following each activity in the use case to completion	There is a lack of templates and heuristics to help translate a business activity into key application and data oriented activities	Lack of patterns	
Missing certain vital steps, most importantly, data modeling and going straight to modeling the sequence diagram	There is a lack of a strict order in modeling the various activities. This leads to designers avoiding certain vital steps. Since the sequence diagram is the key deliverable, the designers delve straight into it	Disorder	Information Overload
Incorrect and missing entity classes even when going through data modeling	Designers don't understand or follow the rules and heuristics of data modeling	Ill structured	Problem Solving
Inability to distinguish the "transaction" in the task from the transaction in the training example which resulted in borrowing and "copying" objects and logic from the example	On the surface, it appears as an anchoring effect, but based on the interviews, it appears that subjects were unable to solve the task, and resorted to "copying" from the example provided during training	Large/Long tasks	Design Complexity
A sense of vagueness about how the pieces connect to one another	There is a sense of vagueness of sub-goals	Vagueness of goal (or sub-goals)	Problem Solving
	Designers don't know how to move systematically from one activity to another	Guessed next action	Problem Solving and Design Complexity

Research Framework

The research framework is developed by identifying cognitive complexity factors that may have caused difficulties in the exploratory study (see Table 1). For example, insufficient anchor adjustment, which was found to be an issue in requirements determination (Browne & Ramesh, 2002), was also identified in subjects' copying objects and logic from the sample solution provided in the exploratory study. These factors are then identified with sources of cognitive complexity as explained by Reeves (1999). These sources are 1) metasocial/information overload, 2) complex problem solving, 3) system complexity, and 4) design complexity (Reeves, 1999).

Addressing these complexity factors will lead to the development of a sequence diagram modeling technique. The pilot study will provide preliminary assessment of the technique and allow modifications if necessary. The final quantitative study will evaluate the efficacy of the proposed technique.

Conclusions

Analysis sequence diagram is an important UML artifact in OOSA&D. However, the usability of this important diagram from novice's perspective has not been studied in the literature. As a result, novices tend to come across difficulties in drawing sequence diagrams. This research study has identified these difficulties as the first step in developing a modeling technique that will facilitate novices in drawing analysis sequence diagrams by addressing the underlying cognitive complexity.

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.
- Batra, D., Hoffer, J. A., & Bostrom, R., P. (1990). Comparing Representations with Relational and EER Models. *Communications of the ACM*, 33(2), 126-139.
- Batra, D., & Satzinger, J. W. (2006). Contemporary Approaches and Techniques for the Systems Analyst. *Journal of Systems Education*, (Upcoming).
- Batra, D., & Zanakis, S. H. (1994). A Conceptual Database Design Approach Based on Rules and Heuristics. *European Journal of Information Systems*, 3(3), 228-239.
- Bell, A. E. (2004). Death by UML Fever. *Queue*, 2(1), 72-80.
- Bolloju, N., & Leung, F. S. K. (2006). Assisting Novice Analysts in Developing Quality Conceptual Models with UML. *Communications of the ACM*, 49(7), 108-112.
- Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The Unified Modeling Language User Guide* (2nd ed.). Upper Saddle River, NJ: Addison-Wesley.
- Browne, G. J., & Ramesh, V. (2002). Improving Information Requirements Determination: A Cognitive Perspective. *Information and Management*, 39(8), 625-645.
- Cockburn, A. (1997). Structuring Use Cases with Goals. *Journal of Object-Oriented Programming*, Sept-Oct and Nov-Dec 1997.
- Cockburn, A. (2000). *Writing Effective Use Cases*. Addison-Wesley Longman.
- Creswell, J. W. (2003). *Research Design: Qualitative, Quantitative, and Mixed Method Approaches* (2nd ed.). Thousand Oaks, CA: Sage Publications.
- Dobing, B., & Parsons, J. (2006). How UML is Used. *Communications of the ACM*, 49(5), 109-113.
- George, J. F., Batra, D., Valacich, J. S., & Hoffer, J. A. (2004). *Object-Oriented Systems Analysis and Design*. Upper Saddle River, NJ: Prentice Hall.
- Jacobson, I., Booch, G., & Rumbaugh, J. (1999). *The Unified Software Development Process*. Reading, Mass: Addison-Wesley.
- Jarke, M., Bui, X. T., & Carroll, J. M. (1998). Scenario Management: An Interdisciplinary Approach. *Requirements Engineering*, 3(3-4), 155-173.

- Kaplan, B., & Duchon, D. (1988). Combining Qualitative and Quantitative Methods in Information Systems Research: A Case Study. *MIS Quarterly*, 12(4), 571-586.
- Leite, J. C. S. d. P., Hadad, G. D. S., Doorn, J. H., & Kaplan, G. N. (2000). A Scenario Construction Process. *Requirements Engineering*, 5(1), 38-61.
- Maxwell, J. A. (1996). *Qualitative Research Design: An Interactive Approach*. Thousand Oaks, CA: Sage Publications.
- Myers, M. D. (1997). Qualitative Research in Information Systems. *MIS Quarterly*, 21(2), 241-242.
- OMG. (2003, March 2003). OMG Unified Modeling Language Specification - Version 1.5. Retrieved August 20, 2006, from <http://www.omg.org/docs/formal/03-03-01.pdf>
- Reeves, W. W. (1999). *Learner-Centered Design: A Cognitive View of Managing Complexity in Product, Information, and Environmental Design*. Thousand Oaks, CA: Sage Publications.
- Rolland, C., & Achour, C. B. (1998). Guiding the Construction of Textual Use Case Specifications. *Data & Knowledge Engineering*, 25(1), 125-160.
- 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.
- Topi, H., & Ramesh, V. (2002). Human Factors Research on Data Modeling: A Review of Prior Research, An Extended Framework and Future Research Directions. *Journal of Database Management*, 13(2), 3-19.