Using UML in Software Requirements Analysis – Experiences from Practical Student Project Work

Dirk Frosch-Wilke
University of Applied Sciences, Kiel, Germany

Dirk.frosch-wilke@fh-kiel.de

Abstract

Currently the Unified Modeling Language (UML) is an industry standard for object-oriented analysis and design of software systems. Accordingly, teaching UML is part of curricula in many universities engaged in the field of software engineering. Yet not much has been reported in the literature on how efficiently such courses enable students to use UML in software development projects. In this paper we present the initial results of our ongoing study into the capabilities of students to use the UML in system design projects after having undergone "traditional" and alternative teaching methods in UML classes. In this paper we investigate students' motivation to follow a use-case driven approach in requirement analysis. We furthermore explore specific problems students are confronted with when using the UML. These findings were gathered during continuous evaluation of a project, in which students were exposed to the real world of systems design, by making the requirement analysis for a customer relationship system. With our study we attempt to optimize our methods of teaching UML in university courses and offer recommendations to this end on the basis of our findings.

Keywords: Software Requirements Analysis, Unified Modeling Language, IT-Education, Object-Oriented Analysis

Introduction

Introducing the UML to students is part of curricula in many universities. The reasons for that are manifold:

- the UML is an OMG (Object Management Group) standard,
- the UML has a wide acceptance in industry,
- there are a lot of CASE (Computer Aided Software Engineering)-tools which support the UML,
- use of the UML is independent of software development processes.
- object-oriented design has become very popular in software development projects

There exits also a lot of textbooks about UML (e.g. Alhir, 2002; Martin, 1997; Oesterreich, 2002; Scott, 2001; Schmuller, 2001). Usually you will find in these books the syntax definition of the UML diagrams together with some –often very simple – examples to put the different diagram notations in some practical context.

The syntax of the UML diagrams is often not explained by using the UML metamodel (Kobryn, 1999). Instead a more intuitive way is chosen. Authors often prefer explaining the use of UML diagrams with the aid of isolated simple examples.

Material published as part of these proceedings, 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

Paper Accepted as a Short Paper

No doubt, this might be an appropriate way for books that aim at an initial introduction of the UML.

However we have to ask the question if this approach is sufficient to enable students to develop an understanding of the UML at a depth sufficient to see the advantages of using the UML in software development projects and to build effective models.

For this reason we took a random sample of ten UML courses from universities in North-America, Australia, Asia and Europe (Southern Polytechnic State University (Georgia, USA), University of Missouri (Kansas City, USA), Western Illinois University (USA), University of Calgary (Canada), University of Newbrunswick (Canada), University of Newcastle (Australia), Swinburne University (Australia), Universität Magdeburg (Germany), University of Aarhus (Denmark), University of Hongkong). Our investigations into these ten courses confirm even for university courses that the UML is preferably introduced by appealing to students' intuition and to foster their initial imagination by simple examples to demonstrate how UML is to be used "correctly".

Our own teaching experience and course evaluations show that this approach is inadequate to qualify students to use the UML effectively in software development projects. The students have extensive knowledge of diagram notations but the majority of them is not able to put this (theoretical) knowledge into an application context. Provided that they are successful in drawing some UML diagrams separately, students will fail in realizing relationships and interdependencies between these different UML diagrams. Students are widespread confronted with the problem that they do not really know how the UML helps them in development projects to build good software.

Some of the reasons for this inability to build effective UML models are:

- Definition and exercise of the different UML diagrams are being dealt with separately.
- Students have not been mad familiar with some typical software development process (see e.g. Sommerville 2000), which they could use as a guide for gaining an understanding how software development projects employing UML are being run.
- No presentation of examples extensive enough and of such broad applicability to gather practical knowledge.
- Variety of the modeling possibilities in the UML

To improve the quality of teaching the UML we created –in cooperation with a software development company- a project, exposing students to the real world of object-oriented analysis and design with the UML. The primary objective of the students enrolment in this project is to make the requirement analysis of a customer relationship management system (CRM system) using the UML, feeling in an environment which resembles the real working conditions of systems developer.

After describing the project in more detail we present some evaluation results of this project. Further we suggest consequences for teaching UML in university courses, which are based on the evaluation results.

Certainly the sample size of 30 students in one course is not sufficient to empirically prove the suggestions for improvement of teaching UML, this has to be done in future work and is beyond the scope of this paper.

The purpose of this paper is to present first results of an ongoing study and to initiate a discussion about alternative training methods in the field of object-oriented analysis and design of information systems.

The Project

Aims of the Project

The aim of this project was to simulate a real working environment in which the students could train the their knowledge about object-oriented analysis and design with the UML. Furthermore this project offered the possibility for practical project management and estimations of software development costs. An additional important aim was to allow the students improving their soft skills (like team- or communication skills) that rate very high on the agenda of IT employers (Janczewski, 2001).

Project Organization

Six project teams were established from thirty participants of a software engineering course. In the course of the project these six teams represented fictitious and competing companies, which tendered for a CRM software development contract.

The client was a real software development company, which supported this project with three employees. These three employees represented the marketing, controlling and IT departments of the client.

The students were responsible to organize the project teams. They chose the project leader and defined responsibilities for different project tasks within the teams on their own.

For collaborative work within the project teams and between teams and client the BSCW-GroupWare tool (BSCW, 2002) was used among other communication channels. The students could use two different CASE tools for the UML modeling: Rational Rose (Rational, 2002) and Poseidon for UML (Gentleware, 2002).

The project period was three months.

The Project Exercise

All six projects teams had to work on the following tasks when conceiving a CRM system:

- Drawing up and updating of a project plan
- Keeping a project log
- Requirement analysis and design of a CRM system using the UML
- Calculation of a software development offer
- Final presentation of the project results

The overall rating of these tasks was the decisive factor for the fictitious placement of the development order.

The Project Evaluation

Subject of Evaluation

The continuous evaluation of the project was divided into three subject areas:

- General aspects of the project (e.g. students gain, use of GroupWare tools)
- Cooperation with the software development company
- Use of UML in object-oriented requirement analysis

Subsequently we will present some results from the last subject area, because the results here are most relevant to conclude consequences for teaching the UML in university courses.

The Main Results

General use of the UML in OOA

As Figure 1 shows, the overall student assessment is that practical project work is necessary for learning the UML.

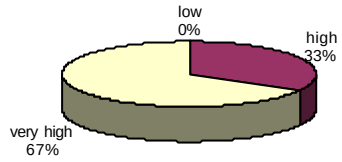


Figure 1: Relevance of practical project work for learning OOA with UML

The relevance of an extensive practical project work for getting a better understanding of the benefits of some modeling language like the UML in software development is underlined by the evaluation results represented in Figure 2. It can be seen on this figure that the students' opinion about the usefulness of the UML for requirement analysis has significantly changed during the practical work experience. Immediately after the course teaching the UML "traditionally" (as described above) only a minority of students was convinced that using the UML could improve the quality of the requirement analysis. This opinion has dramatically changed after the students had the possibility to work with the UML in a bigger software development project.

a) before project work b) after project work

Figure 2: Recommendation of using the UML for software requirement analysis

After the project work the vast majority of students were convinced that the UML helps them to improve the communication within the project team about certain concepts and between the project members and the domain experts. Especially achieving good communication along with good understanding of the users' world is understood to be the key for developing the "right" software system (Fowler, 2000).

All student project teams used use cases for addressing this. After getting an adequate external view of the CRM system by using use cases the students used mainly class diagrams and sequence diagrams to look at more complex details.

Experiences with Using UML Diagrams

One of the goals behind the development of the UML was to keep it as simple as possible while still being able to model the spectrum of systems that needed to be built (Booch et al., 1999). However it is

more complicated than previously developed object-oriented methods, because it is intended to be more comprehensive. As a result, UML diagrams are often difficult to develop especially for the novice.

In what follows we will present some empirical results about how students use the UML in requirement analysis and about the problems they are confronted with when doing this. These results indicate how teaching the UML may be improved which we will discuss in the next paragraph.

Furthermore you can use these results to examine the limitations of an informal definition of UML diagrams. We assume that the absence of some stringent specification in UML courses and textbooks and the isolated treatment of UML elements prevent students from gaining a deep understanding of the UML which is indispensable to handle complex problems in practical work. Proofing the evidence of this, however, is beyond the scope of this paper and has to be explored in future work.

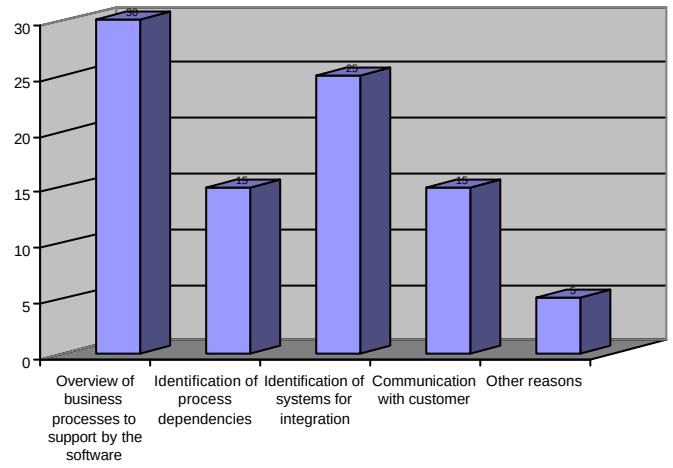


Figure 3: Reasons for using the use-case-diagram
(multiple student answers were possible)

Figure 3 summarizes the reasons and their weightings for following a use-case driven approach in requirement analysis and using the use case diagram.

The main reasons for the students to describe use cases and to visualize them with the use case diagram are to get a better overview of all business processes which should be supported by the CRM software and to know external systems which have to be integrated within the CRM system. Although the majority of students use UML diagrams for communication with the domain experts (as stated before), only 50% communicate with the customer by using a use case diagram. The reason for that might be that the student teams had defined personal responsibility for different tasks in requirement analysis. Therefore not every student was involved in the discussion about business processes with the domain experts.

Figure 3 indicates further that the identification of relationships between use cases, which one can express with three kinds of relationship in the UML, is of less importance for students. The hypothesis that there is a statistical dependency between problems with modeling an use case diagram and not using it for identification of process relationship had to be rejected with Pearson's χ^2 –test (Weiss, 2001).

Using class diagram technique has become truly central within object-oriented analysis. The class diagram is not only widely used, but it also has the greatest range of modeling concepts. The last point might be the reason why many students had problems to describe objects in the CRM system and the various kinds of static relationships among them from a conceptual perspective. Figure 4 shows the main problems of students by using the class diagram technique.

The reason for the difficulty to identify appropriate classes is that there exists no "cooking-recipe" for doing this. Certainly there are some possible methods that can be used to identify classes in some structured process, but most of these methods will produce a lot of insignificant classes. As many students noted in the evaluation process it is difficult for them to identify significant classes because they lack experience in doing this.

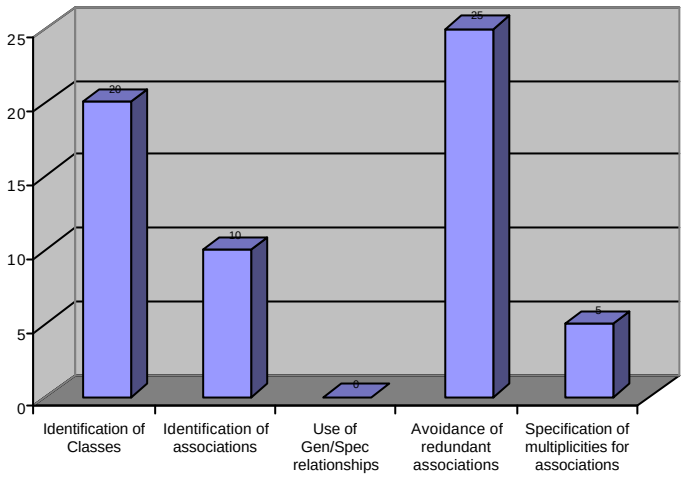


Figure 4: Main problems by using class diagrams (multiple student answe rs were possible)

Among the UML diagrams one of the most difficult diagram to develop is the sequence diagram (Song, 2001). We evaluated some problems students have to develop this form of object interaction diagram. Figure 5 summarizes the results.

The reason for the difficulties summarized in Figure 5 might be that the UML does not provide a process or specific steps that students can follow to produce an effective diagram. Therefore it is often up to the student to develop a method that assists him or her in creating a sequence diagram.

The last range of problems we will address here is about consistency and interrelationships among different UML diagrams used in building conceptual domain models. Although it is good practice to draw diagrams without worrying about there connection at the beginning of requirement analysis in order to get quickly good initial understanding, it is necessary to consolidate the various diagrams into a single consistent domain model. The condition for diagram consolidation is to understand the relationships between different UML diagrams. For this reason we asked the students if they saw relationships between class diagram and sequence diagrams before and after the practical project.

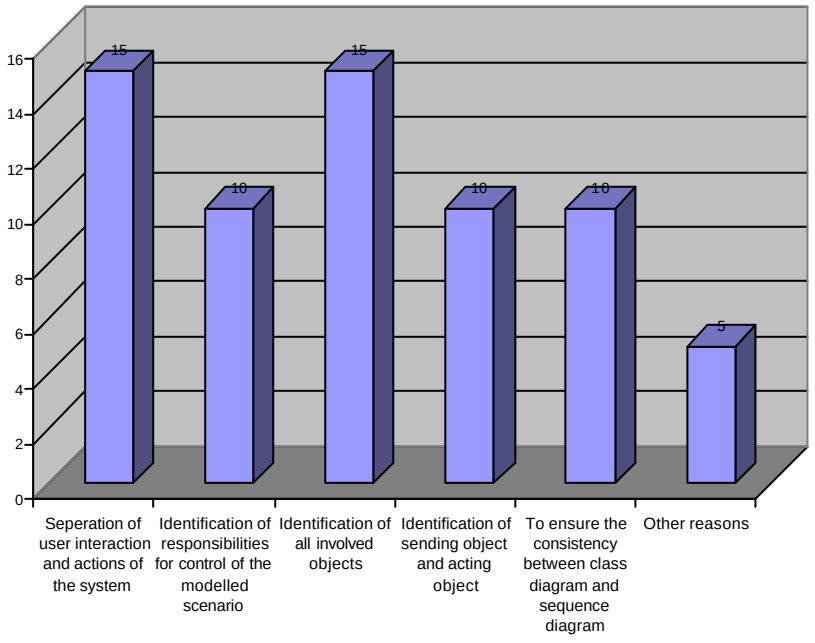
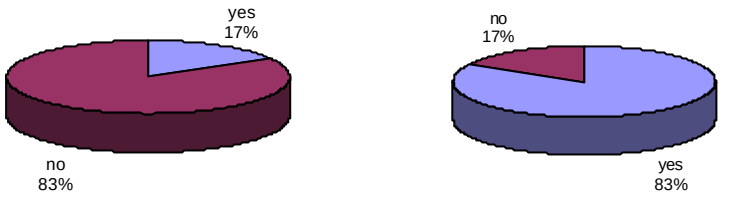


Figure 5: Problems by developing sequence diagrams



a) before project work b) after project work

Figure 6: Students' assessment about a relationship between class- and sequence diagrams

Figure 6 demonstrates that an overwhelming majority of students see connections between different UML diagrams after practical work experiences but had not been able to see this before.

Consequences for Teaching UML

Based on the empirical results presented in the paragraph before, we draw now some conclusions for teaching UML in university courses.

The "traditional" method to teach UML by appealing to students' intuition seems to be a good starting point to reach a basic understanding of the diagram notation. A more rigorous approach by using the UML metamodel would certainly help to define more well-formed models, but only using the meta-model in teaching would prevent students from getting an understanding of how the use of UML can help in software development.

As the empirical results show, however, the "traditional" teaching method is not sufficient to convince students that using the UML in software requirement analysis will give them advantages. One reason for that might be that they do not realize the benefits of the UML diagrams for this task and for further steps in software development processes and that they do not really understand the relationships between different diagrams.

As a result some additional teaching methods and contents are necessary:

- *Integration of practical orientated project work*: "Learning by doing" seems to be the key factor for enabling students to understand the extensive benefits of using the UML in software development, to produce effective diagrams, to build consolidated models and to select UML diagrams commensurate to the problem to be solved. Isolated exercises of UML diagrams are insufficient for gaining these results. A real life project work – ideally in cooperation with real business organization – seems to be indispensable for that purpose. In addition, the project work trains some supplementary skills like communication or project planning – no doubt important abilities for information system students.
- *Use of methods for developing diagrams*: The use of processes for developing diagrams (especially for developing interaction diagrams) seems to be important for a novice in modeling with UML. Beginners are often unable to use the UML effectively in software development because they do not know how to build diagrams which can really help them. Experienced designers will be able to choose or develop a method that assists them creating a model. Beginners like students, however, need advice how to develop UML diagrams. Unfortunately there are only few authors who even mentioned possible methods (see e.g. Song, 2001; Rosenberg, 1999).

- *Draw students attention on diagrams' consistency:* Students are often not able to see the relationships between different UML diagrams because of the sequential and sometimes isolated teaching of UML diagrams. Thus focusing on diagrams' consistency in UML courses helps students to understand the connection between different UML models.
- *Use of the UML within the context of a software development process:* UML techniques are to be put in context of some software development process even if one can use the UML with any process. This will help students to understand which diagrams are valuable for which kind of tasks in software development projects and they can learn how object-oriented development works. We have made good experiences with some kind of iterative software development process though we are convinced that you need various kinds of development processes for developing different software systems.
- *Take care of perspectives:* There are different perspectives you can use in drawing an UML diagram. Following (Cook, 1994) you can distinguish between the conceptual, specification and implementation perspective. Taking care of these perspectives will help students to understand better different kinds of information included in diagrams and, again, how they can use UML diagrams for particular purposes in software development.

Conclusion

In this paper we present results from the evaluation of a software engineering course for Information Systems and Business Administration students. The main objective of this course is object-oriented analysis and design with UML.

Traditional textbooks and teaching methods in this area are often insufficient to enable students to use effectively the UML in software development projects. Based on the empirical results we have drawn some consequences for teaching the UML in university courses.

An integral part of UML courses should be a real-life project. "Learning by doing" seems to be the key factor for students to gain an extensive understanding for benefits of using the UML in software development and to produce effective diagrams. Furthermore, students need assistance like processes or steps that can be followed to produce effective diagrams.

Further work will be concentrated on organizational aspects and content of the practical training of the UML. Although this study did not vary the training methods, use of alternative training approaches will be useful future extension of this ongoing study. Also additional data collection is necessary to use a more robust empirical analysis.

References

Alhir, S. (2002). *Guide to Successfully Applying the UML*. Springer - Telos.

BSCW. (2002). *BSCW – Basic Support for Cooperative Work: Introduction*. Retrieved November 20, 2002 from the World Wide Web http://www.bscw.de/index_en.html

Booch, G., Rumbaugh, J. & Jacobsen, I. (1999). *The Unified Modeling Language: User Guide*. Addison-Wesley.

Cook, S. & Daniels, J. (1994). *Designing Object Systems: Object-Oriented Modeling with Syntropy*. Prentice-Hall.

Fowler, M. (2000). *UML Distilled*. Boston: Addison-Wesley.

Gentleware. (2002). *Poseidon for UML*. Retrieved November 20, 2002 from the World Wide Web http://www.gentleware.com/products/index.php3

Janczewski, L. (2001). Communication Skills for Information System Students. *Proceedings of the 2001 Informing Science Conference*, Krakow (Poland), 274-278.

Kobryn, C. (1999). UML 2001: A Standardization Odyssey. *Communications of the ACM*, Vol. 42, No. 10, 29-37.

Martin, J. & Odell, J. (1997). *Object Oriented Methods: A Foundation, UML Edition.* Prentice Hall.

Oestereich, B. (2002). *Developing Software with UML.* Boston: Addison-Wesley.

Rational. (2002). *Product Features of Rational Rose*. Retrieved November 20, 2002 from the World Wide Web http://www.rational.com/products/rose/index.jsp.

Rosenberg, D. (1999). *Use Case Driven Object Modeling with UML: A Practical Approach*. Addison-Wesley.

Scott, K. (2001). *UML Explained*. Boston: Addison-Wesley.

Schmuller, J. (2001). *Teach Yourself UML in 24 Hours*. Sams.

Sommervile, I. (2000). *Software Engineering*. Addison-Wesley.

Song, I. (2001). A Heuristic for Developing Object Interaction Diagrams. *Proceedings of the 2001 Informing Science Conference*, Krakow (Poland), 487-492.

Weiss, N. (2001). *Introductory Statistics*. Boston: Addison-Wesley.

Biography

Dr. Dirk Frosch-Wilke is a Professor of Business Information Systems at the University of Applied Sciences Kiel. His research interests include the strategic applications of information technology to organizational productivity, electronic commerce and software engineering techniques. He has consulted companies in e-commerce and software development projects. He has presented papers at both national and international conferences. He is editor of the book *Marketingcommunication in the Internet* (Braunschweig: Vieweg:2002; in German) and he has contributed chapters to other texts.