# Learning UML

Keith Alcock, TCS and Developers SIG Member
tcs@keithalcock.com

This book's subject, Unified Modeling Language or UML, has been a topic of discussion at Developers SIG meetings. William Mitchell gave a presentation this January, for example, and his notes are available at the URL below. UML is not a universal translator and is not especially helpful for modeling circuits or designing a house. As the cover states, it is a way of "communicating software design graphically." A more detailed name would be "Booch, Rumbaugh, and Jacobson's consolidated (unified) graphical and textual technique (language) for analysis and design (modeling) of object-oriented software systems." Despite these caveats, UML is becoming the lingua franca of the software design world and is surely worth learning.

The book consists of five parts: Fundamentals, Structural Modeling, Behavioral Modeling, Beyond UML, and Appendixes. The various UML diagrams, after being introduced briefly in part one, are divided into structural and behavioral categories for further elaboration. Class, object, use-case, component, and deployment diagrams belong to the former and sequence, collaboration, state, and activity diagrams belong to the latter. Throughout the book, a project management system is used as the one and only example upon which all diagrams are based. This may be repetitive and boring for many, but it does manage to cover all bases without repeated introductions to new examples.

Each chapter has a similar structure. The author first states what he is going to cover in the chapter. Next, he covers it. In other books a summary would follow, but instead, from part two onward the author includes exercises to reinforce the material and, in the appendix, solutions, so that one can judge whether or not it was learned. Each problem set follows a pattern. First a diagram or two is presented and you are to describe it in prose. Next, you are to modify diagrams based on additional specifications for the project management system. The specifications usually include handy words like "association," "activity, "component," "communication," etc., which have special meaning in the UML context and direct you to the desired solution. Translating requirements to UML in realistic situations would no doubt be much more difficult. I very much approve of the exercises, but wish that they were presented in addition to rather than in place of summaries.

Part one consists of two chapters: Introduction and Object-Oriented Modeling. The introduction contains interesting background information, including a discussion of the software development process. While the book is independent of any particular process, it seems to favor a bottom-up viewpoint. For example, what is usually called inheritance (top-down) is termed generalization (bottom-up) and part two begins with classes rather than use-cases. The second chapter introduces all UML diagrams. Be sure to check the book's web site for errata because some diagrams are confusingly mislabeled. The author goes a little overboard comparing human language to UML and drags out definitions for alphabet, word, sentence, paragraph, section, and document.  If you don't skip this part, note that the smallest unit of meaning in a language is not the word, but rather the morpheme.  After reading the book I was able to redraw these UML diagrams using Visio with minimal error messages, some of which had work-arounds.  I highly recommend reproducing the drawings yourself.

Part two covers structural diagrams.  The class and object diagrams are combined into a chapter of nearly fifty pages.  Use-cases take up fifteen and component and deployment are combined for just ten pages.  In each case I learned how to read a diagram fairly well, but when it came to exercises on drawing one, I was at a loss.  Using a software tool for drawing is easier, since they often include a palette of shapes with names and dialog boxes of properties that when entered are converted into syntactically correct statements.  For a "graphic" language, UML includes a great deal of text. That text can include special symbols, proper names, keywords, punctuation, and adornments like underlining.  The book is only the messenger, so it can't be faulted for UML itself, but I think that it should include a glossary of keywords, list of shapes, reference for the many different arrowheads, summary of dotted line use, etc. rather than strictly examples.

The third part covers behavioral modeling with sequence and collaboration diagrams combined into a chapter of over 25 pages.  State and activity diagrams follow with fewer than ten each.  Much in the way that an instance of a class has a special and well-known name "object," associations between classes when applied to their instances have the special and not well-known name "link."  Similarly, messages become stimuli, collaborations become collaboration instances. . .until one is quite confused with the terminology.  A simple table with one column for the class term and a second for the corresponding instance term would serve well here and is the kind of summary that is missing from the book.

Extension mechanisms and Object Constraint Language make up part four, Beyond UML.  Part five consists of two pages of references, exercise solutions, and an index.  After reading the book twice, I don't think that I have arrived at UML in the first place, much less gone beyond it.  The single example turned into a hundred different UML diagrams with lengthy and often repetitive explanations (see for example pages 73-79 where one paragraph is reproduced six times with little change) was not effective for me.  Others may be quite different, but I think that most would want to supplement this book with another containing additional examples, some designed from the top down, and a great deal more information summarized in tables and non-UML diagrams.  Whatever the book, though, practice makes perfect and that comes highly recommended.

DevSIG URL: http://www.coffeeincodeout.com/DevSig/

Title: Learning UML
Author: Sinan Si Alhir

Publisher: O'Reilly
Pages: 234
Price: US $34.95
ISBN:0-596-00344-7
Website: http://www.oreilly.com/catalog/learnuml/

Keith Alcock is a software developer and consultant interested in agile methodologies and wanting to communicate software designs with UML.