Book Review

Applying Use-Case Driven Object Modeling with UML: An Annotated e-Commerce Example
by Doug Rosenberg and Kendall Scott

Addison Wesley Professional, 2001
ISBN: 0-20173-039-1
Cover Price: US$34.99
(176 Pages)

If you need to create a system model for analysis and design, or if you need to review such a model, then this book might be for you. Doug Rosenberg and Kendall Scott have written a clear description of one effective way to get from requirements to code using the Unified Modeling Language (UML). Both experienced UML users and novice or occasional UML users -- like me -- and will find something useful here.

The book is designed to be a companion to Rosenberg and Scott's Use-Case Driven Object Modeling with UML (Addison-Wesley, 1999). It is readable on its own as long as you have a basic understanding of UML.

The authors describe how to do modeling in the context of the ICONIX process (see Figure 1), which is based heavily on Ivar Jacobson's Objectory process and will be familiar to Rational Unified Process® (RUP®) users. They describe the process in a logical order, beginning with developing a domain model, which is a "kind of glossary of the main abstractions." Once you have a domain model, you can use it to help generate the sequence of diagrams -- use-case, robustness, sequence, and class -- that make up the design model. Because the process uses only these four types of diagrams, it works well for novice UML modelers. It keeps to a minimum the amount of new knowledge that's required but provides a powerful tool for attacking system complexity. Whether you adopt the ICONIX process in whole or in part, you can benefit from the practical information in the book.
The authors claim the process is flexible. They state early in the book that:

> Although the full approach presents the steps in a specific order, it's not crucial that you follow the steps in that order. Many a project has died a horrible death because of a heavy, restrictive, overly prescriptive "cement collar" process, and we are by no means proponents of this approach. What we are saying is that missing answers to any of these questions will add a significant amount of risk to a development effort.

I'm not sure, however, how you might do the steps in a different order. Also, as I read the book, I had a distinct feeling that the authors would never omit any of the steps, as they do not discuss alternative paths. Although I can see ways to use some pieces of the process and not others, in truth, I think the authors are quite dogmatic about the importance of doing everything in their process in precisely the order they lay out.

With three chapters devoted, respectively, to three different types of reviews, this book is an excellent resource for development teams that want to conduct effective reviews of the design as it progresses. The three reviews these chapters address correspond to the three milestones in the ICONIX process:

1. **Requirements Review (RR).** This review is based upon the use-case model, user interface prototypes, domain model, and initial packaging of use cases. The authors emphasize traceability from requirements to the use cases. They recommend avoiding the inclusion of items such as pre-conditions and post-conditions in use cases in order to keep them simple. I've found that there are many way of describing use cases effectively and suspect that most of them will be appropriate for the requirements review.

2. **Preliminary Design Review (PDR).** The preliminary design review occurs when you have done the analysis of your system and
produced robustness diagrams, which use entity, boundary, and control classes to describe the system. Robustness diagrams are not explicitly part of UML; they are derived from the Objectory process and easy to create using a collaboration diagram. Users of the Rational Unified Process will recognize these diagrams, which are part of the analysis model.

3. **Critical Design Review (CDR).** This is the last topic covered in the book. When you have finished the design by creating detailed sequence diagrams and class diagrams, and assigned all behavior required for all of your use cases' flows of events, you are ready to perform the CDR and translate it to code. If you get this far in the process, then you will have a very detailed UML model -- from which you will probably be able to generate a significant amount of code. You have to decide if you want to get to this level of detail.

Unfortunately, the authors seem to have a contract mentality when it comes to including customers in these reviews. In describing the preliminary design review they say: "You can think of PDR as representing a line beyond which customers are no longer welcome to actively participate in the process." I find this very troubling. I always want my customers involved, right up until the day I ship the software. Admittedly, I do frequently meet customers who are required to give a fixed price up front, and these customers might be able to adopt the posture of excluding customers after the PDR. This approach, however, sets up a needless barrier between the customer and the development team that can lead to misunderstanding as things change. Change inevitably happens throughout the process, and it is by working directly with the customer that you can best manage it.

As the title indicates, the book includes a running e-commerce example; the authors say they will describe the application in the Introduction (first chapter) -- which they do -- and then use it in each chapter's exercises. The chapter exercises present a faulty artifact (e.g., a use-case description, part of a diagram, or the like) along with one or more hints about the top ten errors that have occurred; it is then up to you to identify and correct the problems. Although I found these exercises to be quite simple and useful, they do not convey the feel of a consistent example. The Appendix is where the example is actually worked out. (In the Preface, the authors also provide a URL for obtaining the complete, worked-out example.)

One feature I like is that each chapter includes a list of the top ten common modeling errors that relate to the chapter's subject. I find these are great checklist items to use for my models, although some of the error descriptions are worded in a way that makes it difficult to remember they are describing things you should not do. In the chapter on robustness diagrams, for example, Error #6 states, "Allocate behavior to classes on your robustness diagrams." If you read this without thinking about it carefully, it's easy to mistake it for a guideline you should follow. I recommend that you reword these items if you plan to use them in a review checklist.

Overall, the book does deliver on its basic promises. The back cover copy
claims that "With the information, examples, and exercises found here, you will develop the knowledge and skills you need to apply use-case modeling more effectively to your next application." I think the authors do succeed in enabling readers to do this -- and that is a significant accomplishment.

- **Gary Pollice**
  Evangelist
  The Rational Unified Process

Read a review of *Jack: Straight From the Gut* by Jack Welch with John A. Byrne.