

*Your guide to modeling made easy
with the new UML 2 standard*

UML 2

FOR

DUMMIES®

***A Reference
for the
Rest of Us!®***

FREE eTips at dummies.com®

*Full of practical
advice and examples
for every
enterprise*



**Michael Jesse Chonoles
James A. Schardt**

*Your guide to modeling made easy
with the new UML 2 standard*

UML 2

FOR

DUMMIES®

*Full of practical
advice and examples
for every
enterprise*

***A Reference
for the
Rest of Us!®***

FREE eTips at dummies.com®

**Michael Jesse Chonoles
James A. Schardt**



UML 2 For Dummies®

**by Michael Jesse
Chonoles and James A.
Schardt**



WILEY

Wiley Publishing, Inc.

UML 2 For Dummies®

Published by

Wiley Publishing, Inc.

111 River St.

Hoboken, NJ 07030-5774

www.wiley.com

Copyright © 2003 by Wiley Publishing, Inc.,
Indianapolis, Indiana

Published by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8700. Requests to the Publisher for permission should be addressed to the Legal Department, Wiley Publishing, Inc., 10475 Crosspoint Blvd., Indianapolis, IN 46256, (317) 572-3447, fax (317) 572-4447, e-mail: permcoordinator@wiley.com.

Trademarks: Wiley, the Wiley Publishing logo, For Dummies, the Dummies Man logo, A Reference for the Rest of Us!, The Dummies Way, Dummies Daily, The Fun and Easy Way, Dummies.com and related trade dress are trademarks or registered trademarks of Wiley Publishing, Inc., in the United States and other countries, and may not be used without written permission. All other trademarks are the property of their respective owners. Wiley Publishing, Inc., is not associated with any product or vendor mentioned in this book.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY:

While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at 877-

762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Library of Congress Control Number: 2003105654

ISBN: 978-0-7645-2614-5

Manufactured in the United States of America

10 9 8 7 6 5

1B/TR/RQ/QY/IN



About the Authors

Michael Jesse Chonoles: An established system developer, educator, author, and consultant, Michael has done just about everything that you can do in software and system development — business, requirements, and software analysis; software, system, and architectural design; coding in many languages; testing and quality control — right through marketing, packing, and shrink-wrapping the software. His titles include Chief of Methodology for the Advanced Concepts Center (ACC), Software Development Practice Area Director, Consulting Analyst, Software Standard and Practices Manager, Test Director, Senior Software Engineer, several varieties of Team/Project Lead/Staff, and (his personal favorite) Wizard. At the Advanced Concepts Center, he was responsible for the content and direction of its Object-Oriented and Requirements-Gathering Curricula as well as its Software Development Practice. Together with his co-author, he constructed a software/system-development methodology, CADIT, which was an early attempt to combine agile techniques with aerospace discipline. He continues his quest to make the complicated simple, while increasing the professional rigor, quality, and productivity of his audience's working lives.

Michael has been involved in many aspects of UML, even before there was a UML. He's been an active member of the UML RTF (Revision Task Force) at OMG — and

frequently writes, lectures, speaks, and suggests UML topics.

Michael has an MSE in Systems Engineering from the University of Pennsylvania and BSs in Math and Physics from MIT. He can be contacted at michaeljessechonoles@alum.mit.edu.

James A. Schardt: As the Chief Technologist with the Advanced Concepts Center, James provides 24 years of experience and a firm grounding in object-oriented development, data warehousing, and distributed systems. He teaches and mentors Fortune 50 companies in the U.S. and abroad. His many years of practice in object-oriented systems, database design, change management, business engineering, instructional design, systems-architecture assessment, business engineering, and team facilitation bring a wealth of experience to his assignments.

He authors papers on data warehousing and object technology and also wrote a column for *Report on Object-Oriented Analysis and Design*. James speaks at The Data Warehouse Institute's world conferences on a regular basis. He delivers a two-day presentation on collecting and structuring the requirements for enterprise data-warehouse development.

James is always looking for ways to improve the way that we develop systems and software. Clients request him by name to deliver his exceptional knowledge transfer skills, both in the classroom and as a mentor on projects. Over the years, James has managed major research and development programs, invented new systems methods, developed "intelligent" information-access systems, and

provided unique insights into clients' difficult development problems.

James has an MSE in Systems Engineering from the University of Pennsylvania. He can be reached via schardt@acm.org.

Dedication

Michael dedicates this book to his wife Susann (of blessed memory) and to their son Zev, for their love, support, sacrifice, and silliness.

Jim dedicates this book to his wife Martha for her sustaining love and encouragement, and to M. R. Bawa Muhaiyaddeen as the guiding inspiration in his life.

Authors'

Acknowledgments

We would like to thank all the students whom we have taught over the years for their help in shaping our ideas, and all the members of the Advanced Concepts Center, both past and present, for the chance to work with some of the best practitioners in the business of systems and software development.

Together we acknowledge the absolutely necessary help, encouragement, and moral support of our Wiley editors Terri Varveris and Kala Schrager.

Michael would like to thank a whole bunch of people who have helped him over the years, and specifically with this book: Susann Chonoles for teaching him how to write better and for help in proofreading; Zev Chonoles, for being a Test Dummy For Dummies and reading his chapters; his managers Bob DeCarli, Mike Duffy, and Barbara Zimmerman, who encouraged him even when he messed up; and his high-school buddies Joseph Newmark, Jeffrey Landsman, and Barry Salowitz, who keep on telling him what he's doing wrong. It goes without saying that he's grateful to his parents for everything.

He'd also like to acknowledge Jim Schardt for his work toward understanding UML in all its forms, and Lou Varveris for his insight, recommendations, and for access

to the Popkin's System Architect tool. He's also grateful to all the members of the OMG ADTF and the UML Gurus for their technical advice, encouragement, and support over the years — especially Cris Kobryn, Jim Odell, Jim Rumbaugh, Philippe Desfray, and Bran Selic.

Jim would like to thank a number of individuals who helped him develop his knowledge and skills over the years: David Oliver for his systems perspective; Michael Kamfonas for his data-warehouse development insights; Michael Chonoles for his work toward understanding UML in all its forms; Jim Rumbaugh and Fred Eddy for their mentoring on object-oriented analysis; and Michael Blaha and William Premierlani for their guiding hand in developing database-design techniques using UML.

Publisher's Acknowledgments

We're proud of this book; please send us your comments through our online registration form located at

www.dummies.com/register/

Some of the people who helped bring this book to market include the following:

Acquisitions, Editorial, and Media Development

Project Editor: Kala Schrager

Acquisitions Editor: Theresa Varveris

Senior Copy Editor: Barry Childs-Helton

Technical Editor: Lou Varveris

Editorial Manager: Kevin Kirschner

Media Development Supervisor: Richard Graves

Editorial Assistant: Amanda Foxworth

Cartoons: Rich Tennant, www.the5thwave.com

Production

Project Coordinators: Kristie Rees, Dale White

Layout and Graphics: Seth Conley, Kelly Emkow, Carrie Foster, LeAndra Hosier, Stephanie D. Jumper, Michael Kruzil, Mary Gillot Virgin

Proofreaders: Laura Albert, Susan Moritz, Dwight Ramsey, TECHBOOKS Production Services

Indexer: TECHBOOKS Production Services

Publishing and Editorial for Technology Dummies

Richard Swadley, Vice President and Executive Group Publisher

Andy Cummings, Vice President and Publisher

Mary C. Corder, Editorial Director

Publishing for Consumer Dummies

Diane Graves Steele, Vice President and Publisher

Joyce Pepple, Acquisitions Director

Composition Services

Gerry Fahey, Vice President of Production Services

Debbie Stailey, Director of Composition Services

Contents

[Title](#)

[Introduction](#)

[How to Use This Book](#)

[Some Presumptuous Assumptions](#)

[How This Book Is Organized](#)

[Icons Used in This Book](#)

[Where to Go from Here](#)

[Part I : UML and System Development](#)

[Chapter 1: What's UML About, Alfie?](#)

[Introducing UML](#)

[Appreciating the Power of UML](#)

[Choosing the Appropriate UML Diagram](#)

[Identifying Who Needs UML](#)

[Dispelling Misconceptions about UML](#)

[Chapter 2: Following Best Practices](#)

[Understanding UML Terminology and Concepts](#)

[Improving Your Productivity](#)

[Part II : The Basics of Object Modeling](#)

[Chapter 3: Objects and Classes](#)

[Recognizing Classes and Objects](#)

[Naming Objects and Classes](#)

[Identifying Attributes](#)

[Performing Operations](#)

[Diagramming a System's Parts](#)

[Defining Visibility](#)

[Chapter 4: Relating Objects That Work Together](#)

[Showing Static Relationships in a Class Diagram](#)

[Linking Objects Together](#)

[Associating Classes](#)

[Naming Your Associations](#)

[Relating Many Objects \(Multiplicity\)](#)

[Understanding the Roles That Classes Can Play](#)

[Associating Classes with Themselves](#)

[Using Association Classes](#)

[Qualifying Relationships](#)

[Finding a Way — Navigation](#)

[Creating a Program](#)

[Chapter 5: Including the Parts with the Whole](#)

[Representing the Whole and the Parts](#)

[Showing Ownership: Composition](#)

[Showing What Can Be Shared: Aggregation](#)

[Deciding between Aggregation and Composition](#)

[Using Alternate Composite Notation](#)

Chapter 6: Reusing Superclasses: Generalization and Inheritance

[Making Generalizations](#)

[Specializing Classes](#)

[Using Generalization Sets](#)

[Inheriting from Ancestors](#)

[Exploring the Pros and Cons of Multiple Inheritances](#)

[Reusing Code](#)

Chapter 7: Organizing UML Class Diagrams and Packages

[Modeling Objects and Classes on Diagrams](#)

[Constructing Class Diagrams](#)

[Using Project-Oriented Class Diagrams](#)

Part III : The Basics of Use-Case Modeling

Chapter 8: Introducing Use-Case Diagrams

[Identifying Your Audience](#)

[Casting the System's Actors](#)

[Exposing an Actor's Roles](#)

[Showing Your System's Use Cases](#)

[Distinguishing between Internal and External](#)

[Using Context Diagrams](#)

[Packaging Use Cases](#)

Chapter 9: Defining the Inside of a Use Case

[Creating a Use-Case Specification](#)

[Telling the Use-Case Story](#)

[Indicating Alternative Courses of Behavior](#)

[Chapter 10: Relating Use Cases to Each Other](#)

[Linking Use Cases with «include»](#)

[Using Generalization with Use Cases](#)

[Extending Use Cases](#)

[Part IV : The Basics of Functional Modeling](#)

[Chapter 11: Introducing Functional Modeling](#)

[Modeling Functions from an Object-Oriented Perspective](#)

[Writing Text-Based Behavioral Specifications](#)

[Chapter 12: Capturing Scenarios with Sequence Diagrams](#)

[Diagramming an Interaction Scenario](#)

[Composing Interaction Diagrams](#)

[Chapter 13: Specifying Workflows with Activity Diagrams](#)

[Ordering the Flow of Behavior](#)

[Working through Workflow Diagrams](#)

[Chapter 14: Capturing How Objects Collaborate](#)

[Developing a Collaboration](#)

[Constructing the Communication Diagram](#)

[Conquering Concurrency](#)

[Capturing the Collaboration's Design](#)

[Chapter 15: Capturing the Patterns of Behavior](#)

[Describing Patterns with Collaborations](#)

[Applying Patterns](#)

[Framing Frameworks](#)

[Part V : Dynamic Modeling](#)

[Chapter 16: Defining the Object's Lives with States](#)

[Showing the Life of an Object](#)

[Programming an Object's Memory with State Attributes](#)

[Creating State Diagrams from Scenarios](#)

[Chapter 17: Interrupting the States by Hosting Events](#)

[Making Use of Events](#)

[Indicating Order of Execution on a Diagram](#)

[Showing Transitions as Icons](#)

[Chapter 18: Avoiding States of Confusion](#)

[Simplifying Large State Diagrams](#)

[Handling Concurrency with States](#)

[Building Protocol State Machines](#)

[Part VI : Modeling the System's Architecture](#)

[Chapter 19: Deploying the System's Components](#)

[Defining Your System](#)

[Constructing Logical Pieces](#)

[Working with Components](#)

[Deploying Physical Pieces \(Implementation\)](#)

[Chapter 20: Breaking the System into Packages/Subsystems](#)

[Using Packages and Subsystems](#)

[Exploring Dependencies](#)

[Patterning the Relationships](#)

[Part VII : The Part of Tens](#)

[Chapter 21: Ten Common Modeling Mistakes](#)

[Splitting Attributes and Operations](#)

[Using Too Few or Too Many Diagram Types](#)

[Showing Too Much Detail](#)

[Using Vague Terminology](#)

[Defining the Same Thing Twice](#)

[Linking Everything Together](#)

[Creating Too Many Use Cases](#)

[Completing One Diagram Before Moving On](#)

[Cycling Around Class Diagrams](#)

[Not Listening to the User](#)

[Chapter 22: Ten Useful UML Web Sites](#)

[Weave a Tangled Web](#)

[UML Home Page](#)

[UML Forum](#)

[UML 2 Submitters](#)

[OCL Center](#)

[Magazines and Information Portals](#)

[Search Engines](#)

[Tool Sites](#)

[Training Sites](#)

[Forums and Groups](#)

[Miscellaneous Sites](#)

[Chapter 23: Ten Useful UML Modeling Tools](#)

[Picking a Tool](#)

[Argo/UML](#)

[Konesa](#)

[Ideogramic UML](#)

[Objectteering](#)

[Rational Rose Suite](#)

[Rhapsody](#)

[System Architect](#)

[Tau](#)

[Together](#)

[Visio](#)

[Chapter 24: Ten Ways to Use UML Diagrams](#)

[Context Diagram](#)

[Use-Case Diagram](#)

[Domain Class Diagram](#)

[Sequence Diagram](#)

[State Diagram](#)

[Application Class Diagram](#)

[Package Diagram](#)

[Deployment Diagram](#)

[Communication Diagram](#)

[Activity Diagram](#)

Introduction

If, like us, you're a software developer or computer professional of some sort, you probably have to deal with the stereotype that developers can't express themselves among normal humans about normal things.

Unfortunately, this book may not help you with that particular challenge, but it can help improve your ability to communicate with other developers about technical matters. UML

If you're already familiar with UML, you know how powerful and expressive it is — but don't be surprised if you're impressed all over again by the new features of UML 2. Perhaps you found some parts of UML too complicated or the apparent benefit too obscure. Well, the UML gurus have revamped UML in many areas — making easier to express yourself exactly and clearly — and they have also added fresh capabilities for the latest software- and system-development problems that you're facing.

But because your problems are complex — and your solutions are some-times even more complex — UML is not always simple to learn. It's a large and multifaceted language, capable of helping in all areas of development, from analysis to test as well as from database to embedded-real-time. To some, it's a bewildering array of diagrams and symbols. Sometimes it might appear to you that the UML gurus purposely make it too complicated

(and with UML 2, even more so) for *the rest of us* to understand.

Bottom line: You need a practical, experience-based guide to the ins and outs of this new language. Let this book be that guide. We boiled down our experiences with UML (in many environments) and our skills as educators to focus on key UML capabilities that you need *first* to be more productive.

So, with straightforward English and concrete examples, we give you a leg up on expressing yourself and being more creative on the job. (Hey, it *could* help you get a raise — just don't expect us to help you get a date.)

How to Use This Book

There's a right way and a wrong way to use this book. Luckily (like its subject, UML 2), this book is remarkably versatile. If you're a traditionalist, you can read it from cover to cover (although you'll probably stop at the index). That's a great approach if you're really new to UML. If you're familiar with earlier versions of UML, you can skip around looking for the new UML 2 stuff. You may miss our (ahem) great insights into the rest of UML, but you know why you bought the book — do what works. Using any of these techniques will get you familiar with your book so that you can count on it to help unstick you if you hit a snag with UML.

After you make friends with your book, you'll probably find yourself taking advantage of its *just-in-time* features. With just a bit of page flipping, you'll be at a section that's full of examples, tips, techniques, and warnings that will help you with your UML modeling.

There are other ways to use this book . . . and some of them are wrong ways. It's not going to work that well as a doorstop (wrong size), and it probably won't impress your date (unless you're dating a developer who's new to UML). However, it'll look great on your bookshelf — silently conveying to your boss your desire to improve — but if you never open it, you won't get the full benefit.

Some Presumptuous Assumptions

If you're reading this, we can safely assume that not only have you *already* opened the book, you're probably also a developer of software, systems, or databases, and you want to read or write UML 2 diagrams. Perhaps you're a manager or business analyst in the same boat.

We won't assume that you know any particular computer language, although knowing one will certainly help.

For the most part, we assume that you fall into one of two major categories: Either you're a modeler (with a yen to communicate requirements or how you think the

world works), or you're a developer (looking to explore alternative designs or communicate your results). Either way, this book is for you.

We assume that you're capable of using a tool to draw UML diagrams — we don't care which one. If the only tool that you have your hands on is *in* your hands (as opposed to on-screen), you won't be at a disadvantage when you use this book (although your diagrams won't be quite as tidy if you're drawing with a stick on wet sand). You may even be better off doing some diagrams by hand; electronic UML tools are often expensive and may not yet be up to date with all the neat UML 2 features that we cover. If you're itching for a high-tech UML tool, take a look at Chapter 23 where we list of some of the more useful examples (in all price categories).

How This Book Is Organized

Here's your first practical hint about using UML: *Put about five to nine major elements on a diagram — no more.* Studies have shown (we've always wondered who does this type of study) that most people have a hard time comprehending more than about nine elements at a time. Likewise, when designing this book, we decided to follow our own advice and to divide the book into just seven parts.

Remember that you don't have to read this book in order. Just choose the parts and chapters that you need at the time.

Part I: UML and System Development

If you want to know what UML is (and why knowing it is useful), this is the place to go; it covers the basics of UML and how it can be used. You'll also find some common principles for communicating or developing systems with UML. These principles guided the UML gurus when they created UML; the same principles can guide you to effective use of it. Ways to apply these principles crop up throughout the book.

Part II: The Basics of Object Modeling

When you model by using UML, the basics are the things (or *objects*) that you draw and the relationships among them. You'll find information on classes, objects, associations, inheritances, and generalizations. No matter what type of development you do, understanding this part will probably be essential.

Part III: The Basics of Use-Case Modeling

Use cases (detailed real-world examples) allow you to understand and communicate the purpose of a system or its components. They are great for organizing your thoughts — and your system — when you want to get a value-added product out the door.