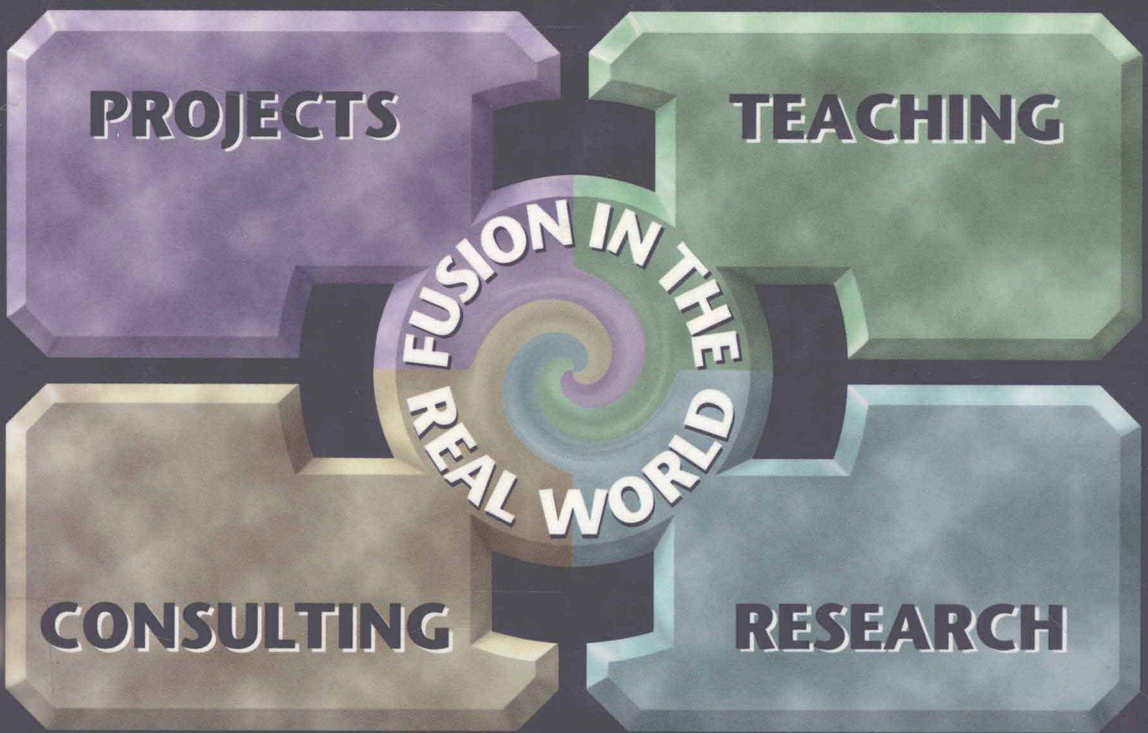
OBJECT-ORIENTED DEVELOPMENT AT WORK

FUSION IN THE REAL WORLD



RUTH MALAN
REED LETSINGER · DEREK COLEMAN

Object-Oriented Development at Work

Fusion In the Real World

Ruth Malan
Reed Letsinger
Derek Coleman

Prentice Hall PTR
Upper Saddle River, NJ 07458

Malan, Ruth.
Object-oriented development at work: fusion in the real world /
Ruth Malan, Reed Letsinger, Derek Coleman.
p. cm. -- (Hewlett-Packard professional books)
Includes bibliographical references and index.
ISBN 0-13-243148-3
1. Object-oriented programming (Computer science) 2. Computer
software--Development. I. Letsinger, Reed. II. Coleman, Derek.
III. Title. IV. Series.
QA76.64.M347 1995 95-42331
005.1'2--dc20 CIP

Editorial/production supervision: **Ann Sullivan**
Cover design: **Paul Gourhan**
Cover manager: **Jerry Votta**
Manufacturing manager: **Alexis R. Heydt**
Acquisitions editor: **Karen Gettman**
Editorial assistant: **Barbara Alfieri**

Published by Prentice Hall PTR
Prentice-Hall, Inc.
A Simon and Schuster Company
Upper Saddle River, NJ 07458

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN: 0-13-243148-3

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Simon & Schuster Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

Preface

This book is about best practices in object-oriented software development. The focus is on the process by which software is developed and in particular the use of the Fusion object-oriented analysis and design method (Coleman et al. 1994). The book contains a collection of papers on the practical aspects of Fusion. The authors come from a wide variety of backgrounds including product development organizations, research laboratories, academia, software houses, and consultancies. From the book the reader will learn about the benefits of using Fusion, how to introduce the method, and how to customize it to meet the needs of a project. Overall, the papers testify to the level of interest in Fusion and the great variety of projects and situations in which the method is making a real and significant contribution.

Audience for Book

The target audience for this book is software engineers and project managers with some familiarity with object-oriented methods. It should be particularly useful to those charged with evaluating methods or introducing a method onto a live project. The book is self-contained and includes an introductory overview of Fusion together with full reference documentation. It is stressed however that the experiences documented in this book are likely to be of general interest to object-oriented practitioners and research workers and not limited to Fusion aficionados.

Evolution of the Fusion Object-Oriented Method

Until the late nineteen eighties, only structured methods, such as structured analysis and design (SA/SD), were available for systematic software development. Of necessity, developers who wanted to approach the new object-oriented paradigm systematically had to apply structured techniques. In most cases these attempts were at best only partially successful because it was not easy to map the concepts of SA/SD to objects.

The problem is that there is a fundamental clash between the computational models that underlie the two paradigms. Structured methods lead to software which is structured as a collection of procedures, all of which can access global state. The object-oriented computational model is fundamentally different because there is no global state and software is structured as a collection of objects, each of which has its own encapsulated state. The objects interact solely by passing messages. Consequently using structured methods for object-oriented development necessarily requires a translation from the structured computational model to the object-oriented model. Packaging the global state of the structured model into objects is, in essence, the same task as designing object-oriented software in the first place. In both cases the developer has the difficult conceptual task of "finding the objects," i.e., partitioning functionality across appropriate object classes.

The problems with the use of SA/SD for object-orientation were soon recognized and prompted investigation into object-oriented analysis and design (OOA/D) methods. By the early nineteen-nineties there were a large number of such methods available. In 1992 the Object Management Group (OMG) compiled a list of nearly thirty OOA/D methods (OMG 1992). Of course, not all of the methods had entered widespread use. Many were speculative research ideas that were not supported by books or training courses. However, there was much commonality between some of the methods; for example, many incorporated some form of object model, or class diagram, based on entity-relationship modeling. Some used models and notations borrowed from structured methods. There were also important differences; some methods emphasized expressiveness of notations while others gave more support for a particular phase of the software life cycle.

For object-oriented software development, unlike traditional software development, there was a wide choice of methods. The great variety of methods, which was growing all the time, caused problems for practical software developers. Before a project could use a method, it was faced with the prospect of carrying out a methods evaluation to ascertain which to use. The prospect of choosing the "wrong" method also increased the risk of using object-oriented techniques.

These problems were very apparent in Hewlett-Packard (HP) company. Many projects embarking on the use of object-oriented techniques were unsure which method should be used. They did not know the strengths and weaknesses of the various methods and did not know which one would best meet their individual needs. In order to alleviate these problems, a research project was begun in 1990 at HP Labs, Bristol, in order to develop OOA/D method training and consultancy suitable for use by the many and diverse HP projects developing software.

The project began by surveying HP's object-oriented projects in order to understand their needs (Coleman and Hayes 1991). The project also conducted an extensive and systematic evaluation of the leading methods (Arnold et al. 1991). At that time, only a few methods had attracted significant numbers of users. The most prominent were Object Modeling Technique (OMT) (Rumbaugh et al. 1991), Booch (1991), Responsibility Driven Design (RDD) (Wirfs-Brock 1990), Shlaer-Mellor (1988) and Coad-Yourdon (1991a, 1991b).

The results of the method evaluation and the studies of user needs confirmed that there was no single best method. It was clear that the best way to meet the needs of software developers was to synthesize a new method based on the best aspects of existing methods. This work lead directly to development of the Fusion OOA/D method, which is a seamless combination of techniques from a number of different methods.

The analysis stage of Fusion is based on OMT, with the addition of a formal methods technique known as pre- and postconditions (Jones 1990). Pre- and postconditions are an essential tool for precisely capturing the desired behavior of software. The analysis stage also employs scenarios which are closely related to the use case concept in Objectory (Jacobson et al. 1992). The Fusion design stage is based on the Class-Responsibility-Collaborator (CRC) (Beck and Cunningham 1989) technique, which is the core component of the RDD method. The design stage also incorporates elements of the Booch method. The different models are integrated to form a systematic process spanning the analysis and design phases of the software life cycle. The Fusion method evolved through several iterations via experimental use in the research labs and on live software development projects. This culminated in the publication of the reference text on Fusion (Coleman et al. 1994) that was published in late 1993.

Since its introduction, the use of Fusion has spread rapidly within HP. It is now being used to develop a broad range of products including printers, network management software and embedded firmware. Today, many companies worldwide are employing Fusion on an even wider range of domains including MIS, telecoms and defense systems. The contents of this book represent just a small fraction of this growing use.

Structure of Book

The book starts with an overview of the Fusion method. This chapter together with the reference material contained in the appendices will give the reader who is unfamiliar with Fusion sufficient context to understand the rest of the book.

The book has four sections. The first section contains in-depth reports of case studies. Chapter 1, by Jerremy Holland, is an account of an HP project developing a distributed telecom test system. The main thrust of the paper is the lessons learned in introducing a method to a team inexperienced in object-oriented development. The lessons are presented as some highly useful guidelines and lists of "dos and don'ts." The second chapter, by Ron Falcone and his colleagues, explains how the analysis stage of Fusion was applied by an HP project developing a breakthrough medical imaging product. The chapter places particular emphasis on project management and how Fusion was customized to meet the circumstances of the project.

Section 2 deals with how Fusion can be applied in a variety of domains and circumstances. Paul Jeremaes, one of the original team that developed Fusion, shows in Chapter 3 how the method can be applied in the face of bottom-up constraints. This chapter tackles one of the key issues for all OOA/D methods—how to apply them to non-greenfield situations. The paper uses the development of telecom network management applications as its motivating example.

In Chapter 4, Laurence Canning and Richard Nethercott report on the application of Fusion to the development of bespoke software, where cost control is a major concern. The authors focus on how Fusion contributes to managing the development process and how it relates to incremental development, project staffing and cost estimation.

In the era of business re-engineering, process modeling is an area of growing importance. This is the subject of Chapter 5 by Colin Atkinson and Michael Weisskopf. The chapter is based on experiences gained working on the development of software for NASA and takes the software maintenance process as the motivating example. The chapter also introduces some interesting and powerful extensions to the Object Interaction Graph notation.

Project management is the key to successful software development and this is the theme of Section 3. In Hewlett-Packard, Fusion is almost universally used in the context of evolutionary development. This combination has been spectacularly successful since it allows a project to systematically tackle business and technical risks. In Chapter 6, Todd Cotton, a principal developer of Evolutionary Fusion and one of HP's internal software consultants, explains how the steps of the Fusion method can be effectively mapped onto an evolutionary and incremental project life cycle.

In Chapter 7, Kris Oosting explains how metrics and defect tracking can be introduced into the management of a project using Fusion. The work is based on Kris's extensive experience as a consultant and software developer using Fusion and other OOA/D methods.

Section 4 of the book focuses on practical extensions to Fusion. Chapter 8, by Andrew Blyth of the University of Newcastle Upon Tyne, introduces a requirements engineering front-end process for the Fusion method. The work has been developed as part of a research project funded by the European Union on Informatics in Healthcare.

Chapter 9 is authored by Howard Ricketts, a designer of the *Fusion*CASE tool and a software consultant. The chapter proposes some extensions to the design phase of Fusion based on Howard's experience as a consultant. The extensions are aimed at making design more systematic through refinement of the object model. The chapter also introduces notations and guidelines for handling concurrency.

Distributed computing is of growing commercial interest and importance.

Many projects today are using Fusion to develop client/server and distributed software. In chapter 10, Kris Oosting extends the Fusion notations to allow the development of client-server applications. The paper is based on the author's extensive experience developing NextStep/OpenStep applications using Fusion.

In the transition to using object technology, education and training of software engineers in OOA/D is of key importance. In Chapter 11, Gabriel Eckert discusses the issues raised in training university students in the use of Fusion. The chapter presents a critique of the Fusion analysis phase based on the observations of student progress developing software in a laboratory project.

The Fusion method was significantly influenced by the ideas and concepts of formal methods. In Chapter 12, Desmond D'Souza and Alan Wills take this a step further by introducing a new methodology which incorporates a more formal notion of refinement into a Fusion-like development methodology. The chapter indicates the extent to which ideas and concepts that were pioneered by Fusion are being incorporated and extended in new experimental methodologies.

Fusion and the Future of OOA/D Methods

From the outset, the originators of Fusion believed that OOA/D methods must be based on the experiences and needs of software developers. Of course, as the challenges facing software developers grow, and understanding of the software development process improves, OOA/D methods themselves must evolve. Users will not benefit from a "fossilized" method which fails to take account of the lessons of experience.

An important aim of this book is to facilitate the evolution by documenting best-practices with Fusion. We also hope that the book will make a contribution beyond Fusion by promoting the cross-fertilization of methods with ideas that work.

As this book shows, the Fusion method is an open "school of thought." Although we at HP Labs are actively working on extending and improving Fusion, we do not consider that we own the method. On the contrary, we enthusiastically welcome all contributions on how Fusion can be developed. We believe that openness must be the watch word for the future evolution of object-oriented analysis and design. The software development community will be best served by methods evolving and converging according to the experience of what works and what does not. This book is a step in that direction.

Derek Coleman
Ruth Malan
Reed Letsinger

REFERENCES

Arnold, P., S. Bodoff, D. Coleman, H. Gilchrist, and F. Hayes. 1991. Evaluation of five object-oriented development methods. In *Journal of Object-Oriented Programming: Focus on Analysis and Design*. pp. 101-121. New York, NY: SIGS Publications.

Beck K. and W. Cunningham. 1989. A laboratory for teaching object-oriented thinking. In *ACM OOPSLA'89 Conference Proceedings.*

Booch, G. 1991.*Object-Oriented Design with Applications.* Redwood City, CA: Benjamin/ Cummings.

Coad, P. and E. Yourdon. 1991a. *Object-Oriented Analysis*, 2nd edition. Englewood Cliffs, NJ: Yourdon Press.

Coad, P. and E. Yourdon. 1991b. *Object-Oriented Design.* Englewood Cliffs, NJ: Yourdon Press.

Coleman, D., P. Arnold, S. Bodoff, C. Dollin, H. Gilchrist, F. Hayes, P. Jeremaes. 1994. *Object-Oriented Development: The Fusion Method.* Englewood Cliffs, NJ: Prentice Hall.

Coleman, D. and F. Hayes. March 1991. Lessons from Hewlett-Packard's experience of using object-oriented technology. In *Tools 4*. pp. 327-333. Paris.

Jacobson, I., M. Christerson, P. Jonsson, and G. Övergaard. 1992. *Object-Oriented Software Engineering*. Reading, MA: Addison-Wesley.

Martin, J. and J.J. Odell. 1992. *Object-Oriented Analysis and Design.* Englewood Cliffs, NJ: Prentice Hall.

OMG. October 1992. Object Analysis and Design Survey of Methods 1992. *Technical Report.* Framingham, MA: Object Management Group.

Rumbaugh, J., M. Blaha, W. Premerlani, F. Eddy, W. Lorensen. 1991. *Object-Oriented Modeling and Design.* Englewood Cliffs, NJ: Prentice Hall.

Shlaer, S. and S.J. Mellor. 1988. *Object-Oriented Systems Analysis - Modeling the World in Data.* Englewood Cliffs, NJ: Yourdon Press.

Wirfs-Brock, R., B. Wilkerson, and L. Wiener. 1990. *Designing Object-Oriented Software.* Englewood Cliffs, NJ: Prentice Hall International.

Acknowledgments

Since its launch in 1992, many people inside and outside of Hewlett-Packard have played an important role in establishing Fusion as an essential success factor in numerous real-world projects. We want to thank the HP engineers who have adopted Fusion, as well as Todd Cotton, Wendell Fields, Bill Crandall, Mark Davey, and Mike Ogush of the HP Software Initiative for the part they have played in making Fusion the dominant object-oriented development method used within HP. We would also like to thank Paul Jeremaes and Chris Dollin for, even though they now working on different projects, they continue to take an active interest in Fusion.

We also thank Patricia Gill, Lisa Guinn, David Bell-Lee, Wolfgang Demmel, Dennis Todd, Tim Ryan, Betty Lin, Wulf Rehder, Steve Mock, Gregson Siu, Larry Marran and Padma Sreenivasan for their support of Fusion as an integral component of HP's training and consulting services.

Fusion has received strong support outside of Hewlett-Packard, and we would like to thank Dr. Lekkos and Carlos Carvahal at ProtoSoft, Howard Ricketts at *Soft*CASE Consulting, Kris Oosting at SHARED OBJECTIVES, Graham Glass and David Norris at ObjectSpace, and Viktor Ohnjec at Semaphore. Special thanks go to all those engineers who have championed the use of Fusion in their organizations, and used Fusion on their projects. It is their experience that forms the basis of this book.

From the initial call for papers to the final production of this book, we have had strong support and encouragement from many people in Hewlett-Packard and the broader Fusion community. We would like to thank them all, and especially Mary Loomis, manager of the Software Technology Lab in Hewlett-Packard Laboratories, for sponsoring our time as editors on this project. We also thank all those who submitted abstracts for the book, but had to withdraw due to work conflicts. We especially thank Justin Murray, whose paper went all the way through the review process receiving high praise from the reviewers but had to be withdrawn at the last moment due to concerns about project confidentiality issues.

Many people participated as reviewers of the papers in this book, including Colin Atkinson, Andrew Blyth, Todd Cotton, Bill Crandall, Lew Creary, Chris Dollin, Gabriel Eckert, Paul Jeremaes, Justin Murray, Kris Oosting, Michael Weisskopf, Kevin Wentzel and Bill Kent. We thank them all.

We would also like to thank Alan Apt for his enthusiastic support for the book concept and Mona Pompili and Sophie Papanikolaou for their advice and assistance with formatting the book. Thanks also to Karen Gettman, Barbara Alfieri, and Pat Pekary for shepherding the book through the publication process, and to Ann Sullivan for coordinating the production. We would like to thank Louise Herndon and William Thomas who did a wonderful job in copy editing this book.

Finally, we would like to thank Paul Malan, Glenna Letsinger and Victoria Stavridou for their support.

Contents