The Practical Guide to Structured Systems Design

Second Edition

Meilir Page-Jones

The Practical Guide to Structured Systems Design

Second Edition

Meilir Page-Jones

YOURDON PRESS
Prentice Hall Building
Englewood Cliffs, New Jersey 07632

LIBRARY OF CONGRESS
Library of Congress Cataloging-in-Publication Data

Page-Jones, Meilir.
Practical guide to structured systems design / Meilir Page-Jones.
-- 2nd ed.
p. cm.
Bibliography: p.
Includes index.
ISBN 0-13-690769-5
1. System design. 2. Electronic data processing--Structured techniques. I. Title.
QA76.9.S88P33 1988
004.2'1--dc19 87-34000
CIP

Editorial/production supervision: WordCrafters Editorial Services, Inc.
Cover design: Wanda Lubelska
Manufacturing buyer: Margaret Rizzi

This book can be made available to businesses and organizations at a special discount when ordered in large quantities. For more information contact:

Prentice-Hall, Inc.
Special Sales and Markets
College Division
Englewood Cliffs, N.J 07632

Previously published by Yourdon Press

Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

ISBN 0-13-690769-5

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*

Foreword

The systems development field has seen many fads and follies. During the past 25 years, we have witnessed "silver bullets" ranging from new versions of 3rd generation languages to brand new programming languages, from code generators to application generators, from diagramming tools to CASE technology, and from project management methodologies to "structured" methodologies.

Structured design is one of those structured methodologies, and it is the subject of Meilir Page-Jones' superb new edition of *The Practical Guide to Structured Systems Design*. While other of the silver bullets may disappear after a time, structured design has become solidly entrenched since it was first introduced in the mid 1970s. Hundreds of thousands of programmers and software designers around the world have learned how to apply the top-down, functional decomposition approach to systems design; a whole generation of college computer science students is now working in industry, building systems with this approach. And along the way, some 100,000 copies of Meilir's classic book have been sold to students and veterans alike.

But even classics need to be updated from time to time. Since the first edition of *The Practical Guide to Structured Systems Design* was published in 1980, we have seen an explosion in new technological developments. On-line, real-time, distributed database systems are common today, in contrast to the single-machine, mainframe-based systems we built in the 1960s and 1970s. And the various "silver bullets" mentioned above have begun to affect the way

software designers do their work: automated CASE tools, for example, make it practical for the designer to create and maintain structure charts, which are discussed throughout the book: this was something that had to be done manually when Meilir Page-Jones first wrote his book, and it meant that in many cases, the work wasn't done at all. And finally, we are learning that we must strike a reasonable compromise between those who want to develop a high-quality design for a high-quality, maintainable system, and those who want to use prototyping tools and fourth generation languages to quickly and effectively ascertain the end-user's real requirements.

The result of all this revamping and revising is a book that is still easy--no, *fun*--to read. As I pointed out in the first edition, part of this is due to Meilir's command of the English language, and some of it is due to his unique Welsh sense of humor. Whatever the reason, it's nice to know that some things haven't changed!

A lot of books on structured design have appeared since Meilir first wrote *The Practical Guide to Structured Systems Design*. But I still think his is the clearest explanation of the subject. After you read it, I think you'll agree.

Edward Yourdon
Water Mill, New York

Notes to the Second Edition

Eight years have passed since I wrote the first edition of this book. During that time much has happened in the field of software development. For example, we've seen the coming of age of information modeling; the maturity of structured analysis through such ideas as essential modeling; further investigation of the allocation of technology to the essential model; the notion of prototyping; and the extension of structured techniques to real-time systems development. In addition, in those seven years—through teaching, consulting, and actually doing honest work—your author has learned more about structured software development.

Nevertheless, despite all this, I've found that the principles of structured design (first laid out by Larry Constantine in the late 1960s) have remained remarkably resilient throughout these eight years. Thus this second edition contains no bombshells that will invalidate any major concept that you may have encountered in the first edition.

Specifically, then, these are the changes in the second edition:

- The rewording of a few passages that readers of the first edition found less than crystal clear
- Presentation of diagrams in a more legible format
- A slight resequencing of the chapters to make each section of the book more cohesive

- An incorporation of those ideas that I've learned since I wrote the first edition and that are relevant to this book
- A discussion of essential structured analysis
- The extension of structured analysis notation (to include, for example, that of real-time systems specification)
- The inclusion of modern information-modeling notation
- An entirely new chapter on the system-configuration activities that occur between essential analysis and structured design
- A fusion of the two chapters on transform and transaction analysis into one chapter
- A fusion of the two chapters on packaging and implementation into one chapter
- The rework of appendix E (the case study) to reflect the changes in the body of the book
- A new appendix on the use of automated tools for structured software development
- A bibliography.

Preface

As you pick up this book, I imagine that you have several questions in your mind. Let me try to anticipate some of them.

The Practical Guide to Structured . . . what?

This book is about design: the design of computer programs and computer systems. Although it's directed particularly at the design of systems for commercial and scientific applications, most of the ideas it sets forth apply to the design of any computer system (for example, compilers, operating systems, or process-control systems). Some of the ideas apply even to the design of non-computer systems.

Is this book for me?

It is, so long as you're not a complete novice to computer software development. To get the most out of this book, you should be a programmer with at least a year's experience under your belt. (Alternatively, you should be a second- or third-year computer-science student.) You should also be yearning for something bigger and better than mere programming: a view of the structure of systems from well above the statement-level of code.

But isn't this structured stuff only for academics?

Structured Programming, it's true, owes its existence to the farsightedness of several people working in the academic environment. However, Structured Analysis and Structured Design were developed almost exclusively in the real world of commercial and scientific shops. These two techniques have been tried and tested in large systems since the late 1960s by DP professionals for whom deadlines, careers, and finicky users were far more real than any academic considerations.

Why should I be interested in the high-level structure of systems and programs?

The reason is that at this level are the clues to producing maintainable, reliable, and adaptable systems, and, ultimately, to reducing the lifetime costs of systems. Throughout the 1970s and 1980s, software costs grew malignantly until users began balking at the high price tags on their systems. Yet these expensive systems typically continued a sad tradition of having no better than mediocre quality. No longer can we afford ad hoc and unpredictable methods of creating software. The future will belong to those of us who bring professionalism to the process of building software and who thus imbue our eventual systems with the highest possible quality.

Is this the only book on structured design?

No. In the past few years—as structured design has grown in popularity—many books on software development have included a chapter or two on structured design. However, the classic tome in the field remains Structured Design [Yourdon & Constantine, 1979]. This brilliant (but often difficult and theoretical) work covers almost every conceivable aspect of the structure of a system. *The Practical Guide to Structured Systems Design* takes the theoretical concepts of structured design and makes them applicable to real-world software development. This book also integrates the approach of structured analysis with that of structured design. It also gives a brief outline of the tools of structured analysis and shows how these tools are a boon not only to the analyst, but also to the designer of a computer system.

Will this book teach me a language?

No. I assume that you're already familiar with at least one of the following languages: COBOL, FORTRAN, PL/I, Assembler, APL, C, or Pascal.

Will this book teach me better programming techniques?

No. I assume you're already a reasonably competent programmer.

Won't it even teach me about Structured Programming?

No, not even Structured Programming.

Well, what *will* it teach me?

By the time you reach the end of this book, you should know about the fundamental ideas behind Structured Design; how to use the tools of Structured Design; how to interact with a structured analyst and how to read, understand, and use a structured specification; objective criteria for evaluating and improving design quality; strategies for creating a good, maintainable design in a systematic way; the modifications that you must make to a design before you can program from it; the impact of Structured Design on management; and when to use each Structured Design tool in a typical systems development project.

Acknowledgments

Too many people have contributed to the development of my ideas for me to acknowledge you all individually. However, I would like to thank two broad groups: my irrepressible colleagues and the thousands of professionals who have attended my seminars. I know that I've learned as much from you as you've learned from me.

I'd also like to thank my principal reviewers: Pete Coad, Hilary Ives, Max Mansur, Chris Pidgeon, Michael Silves, Bob Spurgeon and Ed Yourdon.

I thank too the staffs of Prentice-Hall Inc. and WordCrafters, Inc. for their work on the various incarnations of this manuscript.

Finally, I'd like to thank all of you who have taken the time to write and call me since the first edition was published. I've really appreciated your kind words and constructive criticisms, together with the opportunities you've given me to consult on your projects and to train your people.

Meilir Page-Jones
Wayland Systems Inc., Seattle, Washington

Contents