教育部高等教育司推荐
国外优秀信息科学与技术系列教学用书

软件建筑师实训教程

（影印版）

SOFTWARE ARCHITECT BOOTCAMP

■ Raphael Malveau
Thomas J. Mowbray

PEARSON Education

高等教育出版社
Higher Education Press

教育部高等教育司推荐
国外优秀信息科学与技术系列教学用书

软件建筑师实训教程

（影印版）

SOFTWARE ARCHITECT BOOTCAMP

Raphael Malveau
Thomas J. Mowbray

高等教育出版社

图字：01-2003-0686 号

Software Architect Bootcamp, First Edition

Raphael Malveau, Thomas J. Mowbray

图书在版编目（CIP）数据

软件建筑师实训教程/（美）马尔维奥（Malveau, R.），（美）莫布雷（Mowbray, T. J.）著. —影印版. —北京：高等教育出版社，2003.3
ISBN 7-04-012661-3

Ⅰ.软… Ⅱ.①马…②莫… Ⅲ.软件-高等学校-教材-英文 Ⅳ.TP31

中国版本图书馆 CIP 数据核字(2003)第 012576 号

出版发行	高等教育出版社	购书热线	010-64054588
社　　址	北京市东城区沙滩后街 55 号	免费咨询	800-810-0598
邮政编码	100009	网　　址	http://www.hep.edu.cn http://www.hep.com.cn
传　　真	010-64014048		
经　　销	新华书店北京发行所		
印　　刷	廊坊市科通印业有限公司		
开　　本	787×1092　1/16	版　　次	2003 年 3 月第 1 版
印　　张	21.25	印　　次	2003 年 3 月第 1 次印刷
字　　数	350 000	定　　价	26.00 元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换。

前　言

20世纪末，以计算机和通信技术为代表的信息科学和技术对世界经济、科技、军事、教育和文化等产生了深刻影响。信息科学技术的迅速普及和应用，带动了世界范围信息产业的蓬勃发展，为许多国家带来了丰厚的回报。

进入21世纪，尤其随着我国加入WTO，信息产业的国际竞争将更加激烈。我国信息产业虽然在20世纪末取得了迅猛发展，但与发达国家相比，甚至与印度、爱尔兰等国家相比，还有很大差距。国家信息化的发展速度和信息产业的国际竞争能力，最终都将取决于信息科学技术人才的质量和数量。引进国外信息科学和技术优秀教材，在有条件的学校推动开展英语授课或双语教学，是教育部为加快培养大批高质量的信息技术人才采取的一项重要举措。

为此，教育部要求由高等教育出版社首先开展信息科学和技术教材的引进试点工作。同时提出了两点要求，一是要高水平，二是要低价格。在高等教育出版社和信息科学技术引进教材专家组的努力下，经过比较短的时间，第一批引进的20多种教材已经陆续出版。这套教材出版后受到了广泛的好评，其中有不少是世界信息科学技术领域著名专家、教授的经典之作和反映信息科学技术最新进展的优秀作品，代表了目前世界信息科学技术教育的一流水平，而且价格也是最优惠的，与国内同类自编教材相当。

这项教材引进工作是在教育部高等教育司和高教社的共同组织下，由国内信息科学技术领域的专家、教授广泛参与，在对大量国外教材进行多次遴选的基础上，参考了国内和国外著名大学相关专业的课程设置进行系统引进的。其中，John Wiley公司出版的贝尔实验室信息科学研究中心副总裁Silberschatz教授的经典著作《操作系统概念》，是我们经过反复谈判，做了很多努力才得以引进的。William Stallings先生曾编写了在美国深受欢迎的信息科学技术系列教材，其中有多种教材获得过美国教材和学术著作者协会颁发的计算机科学与工程教材奖，这批引进教材中就有他的两本著作。留美中国学者Jiawei Han先生的《数据挖掘》是该领域中具有里程碑意义的著作。由达特茅斯学院Thomas Cormen和麻省理工学院、哥伦比亚大学的几

位学者共同编著的经典著作《算法导论》，在经历了 11 年的锤炼之后于 2001 年出版了第二版。目前任教于美国 Massachusetts 大学的 James Kurose 教授，曾在美国三所高校先后 10 次获得杰出教师或杰出教学奖，由他主编的《计算机网络》出版后，以其体系新颖、内容先进而倍受欢迎。在努力降低引进教材售价方面，高等教育出版社做了大量和细致的工作。这套引进的教材体现了权威性、系统性、先进性和经济性等特点。

教育部也希望国内和国外的出版商积极参与此项工作，共同促进中国信息技术教育和信息产业的发展。我们在与外商的谈判工作中，不仅要坚定不移地引进国外最优秀的教材，而且还要千方百计地将版权转让费降下来，要让引进教材的价格与国内自编教材相当，让广大教师和学生负担得起。中国的教育市场巨大，外国出版公司和国内出版社要通过扩大发行数量取得效益。

在引进教材的同时，我们还应做好消化吸收，注意学习国外先进的教学思想和教学方法，提高自编教材的水平，使我们的教学和教材在内容体系上，在理论与实践的结合上，在培养学生的动手能力上能有较大的突破和创新。

目前，教育部正在全国 35 所高校推动示范性软件学院的建设和实施，这也是加快培养信息科学技术人才的重要举措之一。示范性软件学院要立足于培养具有国际竞争力的实用性软件人才，与国外知名高校或著名企业合作办学，以国内外著名 IT 企业为实践教学基地，聘请国内外知名教授和软件专家授课，还要率先使用引进教材开展教学。

我们希望通过这些举措，能在较短的时间，为我国培养一大批高质量的信息技术人才，提高我国软件人才的国际竞争力，促进我国信息产业的快速发展，加快推动国家信息化进程，进而带动整个国民经济的跨越式发展。

教育部高等教育司

二〇〇二年三月

Preface

Software architecture is an emerging discipline and an exciting career path for software professionals. We encourage both new and experienced practitioners to read this book as an aid to becoming better software architects. You may have noticed that most software books today do not say much about software architecture. Here, in this volume, we've concentrated the knowledge that you need to be the most effective architect possible.

As co-authors, we have lived through the experience of graduating from "member of technical staff" developers to becoming practicing software architects at the most senior levels of our respective companies. We are technical people, not managers, and we enjoy the technical nature of our work. We enjoy parity of salary and benefits with the senior managers at our respective firms. In other words, we are none-the-worse-for-wear as a consequence of choosing a software architecture career. We think that many of our readers would like to gain from our experience. Hence this book.

This is more than a book about software architecture. It is a field manual that can train you. We choose the pseudomilitary style, because it embodies an essential attitude. As a software architect, you need many survival skills—some technical, some political, some personal. While neither author has military experience, we have seen software architecture become a battleground in many ways. It is a battleground of ideas, as developers compete to forward their own concepts. It is a battleground for control of key design decisions that

may be overruled by managers or developers, perhaps covertly. It is a battleground with many risks, since architects are responsible for a much wider range of technical and process risks than most managers or individual developers.

If you are a practicing software architect, we know that you are a busy professional. After buying this book, we would suggest that you peruse the table of contents and the index for topics that are new to you. Focus on those sections first. When you have time, we suggest that you attempt a cover-to-cover read-through, to familiarize yourself with all of the covered topics and terminology.

If you are new to architecture and want to become a software architect, we suggest that you do a cover-to-cover read-through beginning with the first chapter. Work the exercises provided, which will add an experiential learning element to your experience base.

RAPHAEL MALVEAU
THOMAS J. MOWBRAY, PH.D.
McLean, Virginia, U.S.A.

ACKNOWLEDGMENTS

We would like to express our thanks for all of the generous support of our friends and the technical contributions of our fellow software architects. In particular, we wish to recognize: Jan Putman, Kirstie Bellman, Liz Zeisler, Thad Scheer, Marc Sewell, Laura Sewell, Hernan Astudillo, Theresa Smith, Roger Hebden, Chip Schwartz, Jack and Gillian Hassall, John Eaton, Dr. Amjad Farooq, John Holmes, John Weiler, Kevin Tyson, Kendall White, Chibuike Nwaeze, Dave Dikel, David Kane, John Williams, Bhavani Thuraisingham, Jim Baldo, Eric Stein, John Hetrick, Dave Gregory, John Bentley, Nigel Pates, Richard Taylor-Carr, Dan Lam, Garrett Fuller, David Broudy, Mike Baba, Burt Ellis, Matthew Presley, Robert Davis, Peter Lee, Linda Kemby, Georgene Murray, Alfredo Aunon, Jim Gray, and Woody Lewis.

CONTENTS

O N E

INTRODUCTION

So you want to become a software architect? Or perhaps you are already a software architect, and you want to expand your knowledge of the discipline? This is a book about achieving and maintaining success in your software career. It is also about an important new software discipline and technology, software architecture. It is not a book about getting rich in the software business; our advice helps you to achieve professional fulfillment. Although the monetary rewards are substantial, often what motivates many people in software architecture is being a continuous technical contributor throughout their career. In other words, most software architects want to do technically interesting work, no matter how successful and experienced they become. So the goal of this book is to help you achieve career success as a software architect and then maintain your success.

In this book we cover both heavyweight and lightweight approaches to software architecture. The role of software architect has many aspects: part politician, part technologist, part author, part evangelist, part mentor, part psychologist, and more. At the apex of the software profession, the software architect must understand the viewpoints and techniques of many players in the IT business. We describe the discipline and process of writing specifications, what most people would consider the bulk of software architecture, but we also cover those human aspects of the practice which are most challenging to architects, both new and experienced.

So what does a software architect do? A software architect both designs software and guides others in the creation of software. The architect serves both as a mentor and as the person who documents and codifies how tradeoffs are to be made by other software designers and developers. It is common to see the architect serve as a trainer, disciplinarian, and even counselor to other members of the development team. Of course, leadership by example will always remain the most effective technique in getting software designers and developers on the same page.

1.1 ADVICE FOR SOFTWARE ARCHITECTS

"Success is easy; maintaining success is difficult."—J.B.

If you have a focus for your career, gaining the knowledge you need in order to advance can be relatively easy. For software professionals, simply building your expertise is all that is needed in most corporate environments. For example, we often ask software people what books they have read. In the West, most professionals are familiar with design patterns (see Section 1.3). And many have purchased the book by Erich Gamma and co-authors that established the field of design patterns [Gamma 94]. Some have even read it. However, it always surprises us how few people have read anything further on this important topic.

For software architect books, the situation is even worse. Possibly the reason is that there are fewer popular books, but more likely it is that people are not really focused on software architecture as a career goal. In this book series, by publishing a common body of knowledge about software architecture theory and practice, we are eliminating the first obstacle to establishing a software architecture profession. However, making this information available does not automatically change people's reading habits.

So, if the average software professional only reads about one book per year, just think what you could do in comparison. If you were to read three books on design patterns, you would have access to more knowledge than the vast majority of developers on that important topic. In our own professional development, we try even harder—at least a book each month, and if possible, a book every week. Some books take longer than a week—for example, the 1000-page book on the Catalysis Method [D'Souza 98]. In our opinion, it contains breakthroughs on component-oriented thinking, but so few people are likely to read it thoroughly (except software architects), that it becomes a valuable intellectual tool