软件工程实践丛书

PEARSON Prentice Hall

需求分析

影印版

Requirements Analysis

From Business Views to Architecture

[美] David C. Hay 著

清华大学出版社

需求分析 影印版

大师新著
助您进行有效的需求分析

内容简介：

本书讲述了有效的需求分析方式。David C.Hay从商业角度到软件构架，阐述了目前业界最好的需求分析方法，还在定义构架的整个过程中提供行之有效的指导。

本书可作为软件学院及大学计算机等专业相关课程的教材，也可以作为软件公司各级管理和开发人员参考。

作者简介：

在穿孔卡、纸带和打字机时代，David C.Hay就已经开发出了基于数据库的交互式系统。他是世界著名的Essential Strategies顾问公司的主管。他使用模式技术为其他公司提供帮助，构造信息策略和构架、定义广泛的组织需求。

ISBN 7-302-06352-4

9 787302 063520 >

定价：39.00元

责任编辑：尤晓东　封面设计：立日新设计公司

读者信箱：Book@21bj.com

信息网站：http://www.ePress.com.cn

限中华人民共和国大陆地区销售

PEARSON Prentice Hall

软件工程
实践丛书

需求分析

影印版

清华大学出版社

软件工程实践丛书

需求分析

（影印版）

[美] David C. Hay 著

清华大学出版社

·北 京·

内 容 简 介

本书讲述的是有效的需求分析方式。David C. Hay 从商业角度到软件构架提供了目前业界最好的需求分析方法的全面阐述。此外，本书还在定义构架的整个过程中提供指导。

本书可作为软件学院及大学计算机等专业相关课程的教材，也可以作为软件公司各级管理和开发人员参考。

EISBN: 0-13-028228-6
Requirements Analysis: From Business Views to Architecture, 1e
David C. Hay

北京市版权局著作权合同登记号：图字 01-2002-5726 号

图书在版编目（CIP）数据

需求分析 Requirements Analysis /（美）赫尔著. 影印版.
—北京：清华大学出版社，2003
（软件工程实践丛书）
ISBN 7-302-06352-4
Ⅰ.需... Ⅱ.赫... Ⅲ.软件开发－系统分析－英文 Ⅳ.TP311.52
中国版本图书馆 CIP 数据核字（2003）第 010094 号

出 版 者： 清华大学出版社（北京清华大学学研大厦，邮编 100084）
http://www.tup.com.cn
http://www.tup.tsinghua.edu.cn
责任编辑： 尤晓东
印 刷 者： 北京人民文学印刷厂
发 行 者： 新华书店总店北京发行所
开　　本： 880×1230 1/32 **印张：** 15.75
版　　次： 2003 年 4 月第 1 版 2003 年 4 月第 1 次印刷
书　　号： ISBN 7-302-06352-4/TP・4794
印　　数： 0001～3000
定　　价： 39.00 元

Library of Congress Cataloging-in-Publication Data

A catalog record for this book can be obtained from the Library of Congress

Editorial/Production Supervision: *Donna Cullen-Dolce*
Cover Design Director: *Jerry Votta*
Cover Design: *Anthony Gemmellaro*
Manufacturing Manager: *Alexis R. Heydt-Long*
Executive Editor: *Greg Doench*
Editorial Assistant: *Brandt Kenna*
Marketing Manager: *Debby van Dijk*

Published by Pearson Education, Inc., Publishing as Prentice Hall PTR
Upper Saddle River, New Jersey 07458

Prentice Hall books are widely used by corporations and government agencies for training, marketing, and resale.

For information regarding corporate and government bulk discounts please contact:
Corporate and Government Sales (800) 382-3419 or corpsales@pearsontechgroup.com

Or write: Prentice Hall PTR, Corporate Sales Dept., One Lake Street, Upper Saddle River, NJ 07458

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

ISBN 0-13-028228-6

Pearson Education LTD.
Pearson Education Australia PTY, Limited
Pearson Education Singapore, Pte. Ltd.
Pearson Education North Asia Ltd.
Pearson Education Canada, Ltd.
Pearson Educación de Mexico, S.A. de C.V.
Pearson Education—Japan
Pearson Education Malaysia, Pte. Ltd.

REQUIREMENTS ANALYSIS

From Business Views to Architecture

David C. Hay

PRENTICE HALL PTR
UPPER SADDLE RIVER, NJ 07458
WWW.PHPTR.COM

To my dear Jola, the fabric of my life

Foreword

Once upon a time, my son was five years old. Like most five-year-old children, he was very curious about the world around him. As I was reading Dave's book, one of my son's curious moments came to mind. While our family was cruising in a small boat, he questioned how it was possible for the boat to float in the water, instead of sinking to the bottom of the bay.

To put this moment of childhood curiosity into proper context, it is important to know that my husband is a civil engineer by background. I am a mathematician by training. So, this is exactly the kind of question we enjoy pursuing with our children. In fact, our enthusiasm for my son's question led to our brilliantly delivered revelation of Archimedes Principle, complete with an explanation of its input and output parameters.

Now, those of you who are familiar with five-year-old children already know how this turned out. Not only did my son begin to squirm immediately, but also he was quite unimpressed with Archimedes. Rather, he was interested only in knowing whether something about the boat made it float in the water, or whether it was something about the water that pushed on the boat.

This was one of those puzzling parenting moments. We were speechless. What kind of question is this? Indeed, what kind of answer does it deserve?

Of course, John Zachman's Framework provides insight into the source of our confusion. John's framework teaches us that there are as many as six different perspectives of the same phenomenon. The business expert sees the world differently from the information architect, who in turn sees things differently from the system designer. More importantly, each perspective is legitimate in its own right.

Typically, a spectator views the boat's behavior in terms of the perspective that is most intuitive to that spectator. It was painfully obvious that my son felt most comfortable with an immediate, concrete perspective, where he perceived that the objects are in control. My husband and I simply felt more comfortable with the perspective concerned with the underlying laws of science (or business policies and parental rules!) that are running the show.

But, both perspectives (and others) were at play in floating the boat, even if we weren't explicitly aware of them.

The confusion over these perspectives has tormented information professionals for decades. Today, with the Zachman Framework as a foundation, we now know better. That is, we now know that if two people discussing a prospective system cannot understand each other, it is probably because they are each operating in a different cell of the Framework. Neither professional is necessarily incorrect or incomplete, but simply viewing the same problem space from a different perspective.

But, the total architecture is the holistic view of how individually organized deliverables relate to each other in a blueprint (not unlike a musical score, as Dave writes in his introduction) and how they interact with each other in the working world.

It is no easy task to create such a blueprint. The Zachman Framework is elegant in pointing out the breadth and depth of the challenge. But, as a practitioner, how do you navigate through it? Until now, you were on your own, but Dave's book is an excellent directory for this journey. Specifically, Dave provides a comprehensive survey of techniques for the Framework cells that defines the path from vision to architecture.

The value of this book is three-fold. First, the book is appropriate for both newcomers and old timers because it is historical and tutorial in nature. Second, the book has something of value for every reader, whatever a reader's emotional attachment to a particular systems development paradigm. That's because the book graciously incorporates techniques from well-known to the lesser known disciplines. It includes proven insights from structured systems analysis to information engineering to object-orientation and to the most recent, a business rules approach. It is destined to become the authoritative source for defining roadmaps from vision to architecture. The book is articulate (a natural strength of Dave's) and to the point. We can be grateful for Dave's opinions on this subject and his generosity in sharing them in a serious, but entertaining way.

Third, and most important of all, this book elevates (necessarily again, thank goodness!) the importance of the requirements analysis process. Why is this so important?

Dave points out that in our business, painfully, we operate most often without a score to follow. We are the epitome of improvisation. Bob Giordano of Knowledge Partners Inc indicates that this is not new. He states, "We have always employed iterative approaches to development. It is just that in the 70s and 80s, each iteration took years and cost millions because monolithic systems morphologically coupled everything to everything else."

Today, the iterative phenomenon is more obvious, prevalent and perhaps exaggerated, with the recent trend toward iterative development methods and extreme everything. Without a doubt, these trends are useful for specific aspects of systems

development, for those pieces that are uncertain or that are discretionary. But, is there a hidden cost to sacrificing the fundamental elements of analysis? Perhaps so.

Dave points out that analysis is needed in order to see simplicity, and to avoid the development of unnecessarily complex systems. Analysis is also needed to understand the elements of change. We live in a world of accelerating change and uncertainty. Therefore, change should not stop when the system goes into production. The ability to change should be designed into the system so that incremental change is the name of the game forever, at a reasonable price. Otherwise the business will march in place and possibly go backwards.

You are setting the pace and direction of business change through information technology. You ought to have a score by which an uncertain future can unfold in an orderly way. Use this book as a roadmap for creating that score.

Time has passed. It has been many years since my son's question about floating boats. Supposedly, he has long forgotten the question. Hopefully, he is old enough to have a greater appreciation for Archimedes' role in the boat's behavior.

What is the answer to my son's question? I still don't know because I don't understand the whole picture, only parts of it.

But, if I wanted to know the answer to my son's question, I could find the answer in the score or blueprint.

It would all be so elegant. It would also seem simple because someone took the time to make it so. As you can see, the score or blueprint is a valuable investment because it represents solid analytical thinking. This thinking has one eye focused on simplicity and another looking to enable change. When these two properties are present (simplicity and agility), the blueprint (or score) should stand the test of time. If you skip Requirements Analysis, you miss this thinking. Dave helps you rediscover it.

Barbara von Halle

Preface

The Inspiration for This Book: Object-Oriented Analysis

Object-oriented programming has, in recent years, radically reduced the amount of time and effort required to build systems. A technology derived from real-time systems, it has been successfully applied to interactive, windows-based user interfaces and the systems they support. One of its appeals is that it is highly modular, allowing pieces to be built and repaired easily without major surgery on an entire system. Moreover, modules can be reused.

In recent years the object-oriented community has ventured into the world of requirements analysis. Numerous books on "object-oriented analysis" have appeared, and the UML has come on the scene as a technique for supporting this.

There are three attendant problems: First, object-oriented programmers, like all programmers, tend to focus on the technology of producing programs, and they find it less interesting to go out and analyze the nature of a business. The skills required to do that are different, so they may be less likely to have them. The idea seems to have arisen that object-oriented designers are natural systems analysts, although this does not necessarily follow.

A second problem is that some authors consider requirements analysis an object-oriented phenomenon. Ideas developed from information engineering and other sources are ignored as though they were irrelevant to the object-oriented world. This has meant, among other things, that disciplines which have been important and valuable to the field for several decades have been simply ignored.

In 1991, for example, James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorensen published *Object-Oriented Modeling and Design*. This book presented an object-modeling notation along with a methodology called the "Object Modeling Technique", or OMT. The notation consisted of symbols for the same concepts that Clive Finkelstein and James Martin had presented ten years earlier in

their work on information engineering. OMT is concerned with object classes (entity types), associations (relationships), and attributes.

The methodology, while it purported to be a significant departure from "traditional software development approaches" [Rumbaugh et al. 1991, p. 146], was very similar to information engineering. Like information engineering, it was based on the principles of "shifting of development effort into analysis", "emphasis on data structure before function", and a "seamless development process". These are precisely the principles that had already been articulated by Messrs. Finkelstein and Martin. One significant difference is that OMT is much more oriented toward an iterative approach.

Deep in Chapter 12 the book does give credit to Peter Chen as the inventor of entity/relationship modeling, and it recognizes that object-modeling's techniques are descended directly from entity/relationship modeling [Rumbaugh et al. 1991, p. 271]. But this is not obvious from the language in the rest of the book.

Modern requirements analysis is in fact the combined work of many people who have been contributing to the industry for over 30 years. The role of requirements analysis in the system-development life cycle is more important than ever, even with the advent of object-oriented technologies. Contrary to what some say, it has not changed fundamentally with the advent of these techniques.

Object orientation has indeed contributed to requirements analysis, but it has less to do with it than some perceive. The requirements analysis process is intended to identify business requirements for information technology, not to determine the technology used to solve those requirements. A properly done requirements specification should be able to guide designers using any technology.

The third problem comes from authors who insist on including "object-oriented features" in the requirements process. "Control objects" and "interface objects" are artifacts from object-oriented *design*, but they are often (inappropriately) described as part of the requirements analysis process.

The fact of the matter is that the process of identifying requirements is fundamentally different from the process of applying technology to address those requirements. It should be possible to identify requirements without necessarily knowing (except in the most general terms) what technology will be applied to address them. The same set of requirements might be satisfied by an object-oriented application, by an application implemented with Oracle Corporation's relational tools, or by a set of COBOL programs.

About the Book

Rather than viewing requirements analysis from the perspective of a particular implementation technology, this book views it as fundamentally an architectural process.

Specifically, it sets out to answer the question: How do we identify and understand the architecture of an enterprise, so that whatever systems we build for it can truly support that architecture? To do this, it attempts to bring together as many as possible of the best techniques and approaches from the entire history of systems development (including some that originated in the object-oriented world), and it will argue the relevance of all of these in developing object-oriented and other kinds of systems.

Merriam Webster defines "architecture" as "a unifying or coherent form or structure" [*Merriam Webster*, 2001]. When the present book describes an architecture for requirements analysis, it is describing the structure of the entire requirements process. The book is informed by the work of John Zachman, who has described the architecture of systems development as a matrix, with the perspectives of the players in the development process as rows, and the things to be seen from each perspective as columns. The various techniques presented are organized in terms of the cells in such a matrix.

This book's premise is that requirements analysis is the translation of a set of business owners' views of the enterprise to a single, comprehensive architectural view of that enterprise. After some introductory chapters, there is a chapter on each of the dimensions (what, how, where, who, when and why) of the two perspectives.

While the book's focus is on these two rows, attention must be paid to the "scope" perspective that puts all our efforts in perspective. Also, where it is useful to our understanding (notwithstanding the remarks above), reference is occasionally made to the implications for technology designers of what we learn.

While, in one sense, this book will cover ground explored by others, it is unique because it describes in one place the full range of artifacts that can be delivered in an analysis project. This should give the book a completeness not found in most. It addresses not only analysis of data and process, but also the analysis of data networks, people and organizations, events, and motivation.

This is not the kind of book that you can read through like a mystery novel. The Introduction and Chapters 1 and 2 provide an overview of the field and should be read first. The remaining six chapters provide both a roadmap and a reference guide. The roadmap describes 12 of the 36 cells in the Architecture Framework and how they relate to each other. The reference guide is to the myriad of techniques that are available for each cell. If there is a technique you've heard of and you would like to know both more about it and how it fits into the general scheme of things, you should be able to find it here.

Note that Mr. Zachman's great insight was to provide a framework for organizing what we know. That we don't know everything equally well becomes quickly apparent when we try to populate the matrix. You will observe as you go through the book that the various chapters do not describe each area of knowledge equally well or equally completely. Indeed, the chapters are of very different lengths. Moreover, each is written

in a somewhat different style, depending on the kinds of information available for that column.

We've had a lot more experience modeling data, for example, than we have in modeling locations or even people and organizations. Mr. Codd gave us a mathematical basis for modeling data that does not exist for any of the other columns. For the others, we are trying to establish discipline, but it doesn't come easy.

A colleague of mine once remarked that he would like a book "to assist people like me who don't have time to learn anything that's not new". I would like this to be such a book. It will show you what's already been invented and point you in the direction of things that have not.

It is the joy and aggravation of our times to have the opportunity to be in on an industry that is building itself before our very eyes. At its best, this book is no more than a snapshot of what we know in the year 2002 of the modern era. With luck, future editions will have a great deal to add. The homework assignment for every reader of this book is to be diligent in expanding this body of knowledge.

The Important Stuff

Arguments about which technique is better are not what this book is about. It attempts to present a wide range of techniques and asks you to choose the best ones for your particular situation. So far in our history, the most models are available for modeling data and modeling activities. The most important chapter here, however, is not about modeling either of those. Chapter 6 is about understanding people and organizations, the roles they play in the success of an enterprise, and the importance of communications channels in making that success happen. This, alas, is something that has not been very well or very systematically addressed in our industry.

For this reason, the chapter digs back twenty years to the field of cybernetics, and it uses insights from this field to discuss the communications channels that constitute the lifeblood of an organization. Specifically, it describes the way we use "amplifiers" and "attenuators" (filters) to manage the proliferation of "variety" (complexity) that confronts every business. Managing variety, after all, turns out to be what everyone's job is all about. Buried in the middle of that chapter is an expression of the true purpose of the requirements analysis process:

Requirements analysis is the examination of an organization to determine the most effective amplifiers and attenuators to build. What are needed? How are those now in place ineffective or counter productive? What should they look like, given the purpose and organization of the enterprise?