OBJECT-ORIENTED INFORMATION ENGINEERING

Analysis, Design, and Implementation

Stephen L. Montgomery

AP PROFESSIONAL

Object-Oriented Information Engineering

This page intentionally left blank

Object-Oriented Information Engineering

Analysis, Design, and Implementation

Stephen Montgomery

AP PROFESSIONAL
A Division of Harcourt Brace & Company

Boston San Diego New York
London Sydney Tokyo Toronto

This book is printed on acid-free paper. ♾

Copyright © 1994 by Academic Press, Inc.
All rights reserved.
No part of this publication may be reproduced or
transmitted in any form or by any means, electronic
or mechanical, including photocopy, recording, or
any information storage and retrieval system, without
permission in writing from the publisher.

AP PROFESSIONAL
955 Massachusetts Avenue, Cambridge, MA 02139

An imprint of ACADEMIC PRESS, INC.
A Division of HARCOURT BRACE & COMPANY

United Kingdom Edition published by
ACADEMIC PRESS LIMITED
24–28 Oval Road, London NW1 7DX

Library of Congress Cataloging-in-Publication Data

Montgomery, Stephen.
Object-oriented information engineering: analysis, design, and implementation / Stephen Montgomery.
p. cm.
Includes bibliographical references and index.
ISBN 0-12-505040-2 (alk. paper)
1. Object-oriented databases. 2. Systems engineering. I. Title.
QA76.9.D3M649 1994
658.4'038'0285421—dc20 93-43044
CIP

Printed in the United States of America
94 95 96 97 98 ML 9 8 7 6 5 4 3 2 1

Contents

Preface xiii

1 What Is Information Engineering? 1

1.1 Overview of Information Engineering 1

Enterprise-Wide Approach 1 • Engineering Approach 2 • Data Sharing 2 • Automated Tools 2 •

1.2 The Phases of Information Engineering 3

Finkelstein's Information Engineering Approach 4 • Martin's Information Engineering Approach 7 •

1.3 Object-Oriented Information Engineering 8

End Notes 9

2 What Is Object Orientation? 11

2.1 Why Objects? 11

The Need for Better Systems Development 11 • Assembling Systems from Objects 13 • Integration of Processing with Information 13 •

2.2 Benefits of Object Technology 15

2.3 What Are Objects? 17

2.4 Abstraction 18

2.5 Encapsulation 20

2.6 Hierarchies and Inheritance 22

Encapsulation 23 • Generalization 23 • Aggregation 23

2.7 Association 25

2.8 Messages 26

2.9 Polymorphism 27

End Notes 28

3 Business Information Requirements 31

3.1 Types of Information Models 31

The Enterprise Planning Model 32 • The Enterprise Logical Information Model 33 • Systems Logical Information Models 34

3.2 Characterize Subject Areas 34

3.3 Refine Subject Area Classes 37

3.4 Characterize Object Classes 39

Identifying Object Classes 39 • Guidelines for Selecting Object Classes 40

3.5 Describe Object Associations 41

Cardinality 41 • Optionality 42 • Guidelines for Selecting Associations 42

3.6 Define Object Attributes 43

Guidelines for Selecting Attributes 44

End Notes 45

4 Detailed Object Modeling 47

4.1 Aggregation 47

4.2 Abstract Classes 50

4.3 Generalization 51

4.4 Multiple Inheritance 53

4.5 Keys 55

4.6 Constraints 55

Referential Integrity 55 • Insert Rules 56 • Delete Rules 56 • Domain Integrity 56 • Triggering Operation Integrity Rules 58

End Notes 59

5 Business Functions and Subject Areas 61

5.1 Characterize Business Functions 63

5.2 Characterize Object Classes within Each Function 64

5.3 Identify Operations for Objects within a Function 67

5.4 Characterize Information Requirements across Functions 69

5.5 Build an Inventory of Business Events 70

5.6 Build Function Context Models 71

End Notes 73

6 Individual Object Behaviors 75

6.1 Identify Object States 76

6.2 Characterize Object Operations 80

6.3 Define Trigger Conditions and Events 82

6.4 Model Object Life Cycles 83
Build an Object State Matrix 84 • Hierarchies of Object States 85

End Notes 86

7 Object Interactions 87

7.1 Build Function Context Models 87

7.2 Build Event Lists 89

7.3 Describe Event Response Scenarios 89

7.4 Build an Event Response Model for Each Event 93

7.5 Build an Object Interaction Model 93
Message Flow Diagrams 95 • Control Flow Diagrams 96 • Use of Message Flow Diagrams 97 • Describing Object Interactions 100 • Describing Object Interaction Activities 101

End Notes 105

8 Integrate and Validate Analysis Models 107

8.1 Why Model Integration and Validation Are So Important 107

8.2 Compare Diagrams 108

8.3 Resolve Diagram Conflicts 110

8.4 Examine the Relationships between Models 110

8.5 Perform Object Life-Cycle Analysis 112

8.6 Perform Data Use Analysis 113

8.7 Validate Object Structures 116

Modeling Time 116 • One-to-One Associations 116 • Parallel Associations 117 • Recursive Associations 117 • Summary of Object Structure Validation 117

End Notes 117

9 Object Structure Designs 119

9.1 Design Classes 119

9.2 Design for Objects 120

9.3 Design Attributes 122

9.4 Design Class Hierarchies 123

9.5 Design for Derived Data 123

9.6 Design for Efficient Access 124

9.7 Design for Data Management Systems 125

9.8 General Database Design Considerations 126

Object-Oriented Database Characteristics 126 • Modeling of Objects 127 • Identifying Object Instances 128 • User-Definable Database Operations 129 • Design for Encapsulation 130 • Design for Class Hierarchies 130

9.9 Issues in Object Database Design 131

Transactions 131 •

End Notes 134

10 Application System Object Behavior 135

10.1 Procedure-Driven Control 135

10.2 Event-Driven Control 136

10.3 Concurrent Task Control 139

End Notes 140

11 User Interface Requirements 143

11.1 Analyze Users 144

11.2 Design Menu/Command Structures 144

11.3 Design the User Interface Interaction 147

11.4 Build a Prototype of the User Interface 148

11.5 Design User Interface Classes 151

End Notes 154

12 General System Designs 155

12.1 Design Problems Addressable during Analysis 155

Acceptance of Incomplete Requirements 157 • Problems Partitioning and Decomposing Analysis Models 157 • Service-Level Problems 158 • User Interface Problems 159

12.2 Modules 160

12.3 Subsystems 161

12.4 Process Architecture 162

Allocating Subsystems to Processors and Tasks 162 • Depicting Process Architecture 164

12.5 Software Architectures 165

Interactive Interfaces 166 • Transaction Management 167 • Batch Transformation 167 • General Design Considerations 168

End Notes 168

13 Design for System Distribution 171

13.1 Reasons to Distribute or Centralize Objects 171

13.2 Mapping Objects to Locations 174

13.3 Estimating Traffic Between Locations 178

End Notes 181

14 Redesigning Existing Systems for the Future 183

14.1 What Is Reengineering? 183

14.2 Why Bother Reengineering Systems? 184

14.3 How to Get Started on the Path to Building Objects 186

Assess Impact on Systems and their Components 186 • Business Process Reengineering 188

14.4 Identify Candidate File Structures for Subject Databases 190

14.5 Model Databases and File Structures as Objects 192

14.6 Model System Interfaces as Object Interfaces 193

Encapsulation 193 • Inheritance 194 • Methods 195 • Object Wrapping 195

14.7 Model Distributed Systems as Networks of Objects 196

14.8 Process Normalization 198

Unique Processes 199 • Flow of Control from Common to Unique 200 • Module Cohesion 201

14.9 Migrating Structured Analysis Models to Object Models 202

Carve Structured Analysis Models 203 • Examine External Entities 204 • Examine Data Stores 205 • Examine Data Flows 205 • Examine Processes 206 • Results after Carving 207 • Synthesize Classes 207 • Identify Classes 207 • Refine Classes 207

14.10 Design an Object-Oriented Systems Architecture 208

End Notes 210

15 Programming Languages 213

15.1 Object-Oriented Languages 213

Smalltalk 213 • Object Pascal 214 • C++ 214 • Eiffel 215

15.2 Traditional Languages 215

Abstract Data Types 215 • Encapsulation 217 • Inheritance 218 • Methods 219 • Languages 221

15.3 Selecting a Programming Language 221

End Notes 223

16 Database Environments 225

16.1 Persistent Data 225

16.2 Database Models 226

Hierarchical 226 • Network 226 • Relational 226 • Object 228

16.3 Object-oriented DBMS 229

Class Invariants 231 • Schema Changes 232 • GemStone 233 • ONTOS DB 235 • ITASCA 237 • ObjectStore 239 • VERSANT 240

16.4 Extended Relational DBMS 242

POSTGRES 243 • Informix 243 • EXODUS 243

16.5 Traditional DBMS with Object-Oriented Designs 244

Object Identifiers 244 • Classes 244 • Associations 245 • Operations 247

16.6 Disadvantages of Object-Oriented DBMS 248

End Notes 250

17 Client/Server and Cooperative Processing 251

17.1 Distributed Databases 251

Environments 251 • Requirements 252 • Features 253 • Reasons for Using Distributed Databases 255

17.2 Cooperative Processing 256

Types of Cooperative Processing 257

17.3 Networking 260

Connectivity 260 • Network Topologies 261 • Advantages of LANs 264 • Network Management 264

17.4 Client/Server Architecture 264

The Client/Server Model 265 • Types of Client/Server Processes 267

17.5 Client Characteristics 271

The Client's Role 271 • Client Services 272

17.6 Server Characteristics 273

The Server's Role 273 • Server Functions 274

End Notes 275

18 Object-Oriented Systems Architecture 277

18.1 Design an Object-Oriented Systems Architecture 277
18.2 The Object Management Group 283
18.3 Object Management Architecture 284
18.4 Standardized Messaging 287
18.5 Interfacing to Traditional Systems 287
End Notes 287

Glossary 289

Bibliography 305

Index 311

Preface

Object orientation is becoming very important to application developers now that major vendors such as IBM, Apple, Hewlett-Packard, Microsoft, Borland and others are becoming heavily involved with object-oriented technologies. Structured analysis and design techniques are well established for building traditional systems, but object-oriented development still has few good techniques.

The two seemingly separate worlds of traditional development and object-oriented development need not be separated at all. Many techniques appearing in object-oriented literature have close parallels in more traditional structured techniques, which themselves are new to many developers. This book shows the reader how to integrate the two into a basic framework for developing advanced information systems.

In the first two chapters, the reader is exposed to the basic concepts of information engineering and object orientation. From the third chapter on, an integrated view of the two approaches is presented. Although information engineering is the foundation for business planning and analysis, the emphasis on object orientation in analysis and design has become the primary approach in implementation.

A strong emphasis is placed on analysis in this book. I feel that this is the development phase where the transition to object orientation can best be introduced to development projects. Design can proceed in a traditional fashion, but I discourage this because I present an object-oriented design approach.

Many readers will still need to design and implement systems in an information engineering environment, but I assume that this approach is somewhat understood. Object orientation is lacking in current development approaches, and information engineering is no exception. The later chapters address the need for extending existing approaches with object orientation and give many examples of how this extension can be accomplished.

The last portion of the book deals with the choices that must be made when it comes to implementing an object-oriented design. Object-oriented languages and database management systems are not required in order to implement a design successfully, but they can greatly ease the burden of doing so. I present the most likely program and database development options as a guide to build object-oriented systems or to migrate existing systems to more object-oriented ones.

1

What Is Information Engineering?

1.1 OVERVIEW OF INFORMATION ENGINEERING

The term "information engineering" was coined by Clive Finkelstein in the early 1980s and popularized by James Martin in the mid- and late-1980s. The term refers to an integrated set of methodologies for creating and operating information systems. Information engineering relies upon fully normalized data models of information in a business enterprise. Such models are maintained in an automated system that uses workstation tools to build business applications that use enterprise data. The modeling tools support enterprise planning, business analysis, business and technical design and building and maintaining the resulting information systems. A major objective of the information engineering approach to system development is avoidance of common development and maintenance problems.

Enterprise-Wide Approach

One concept that separates information engineering from many other system development approaches is the application of structured development techniques to an entire business enterprise, not just a single development project or group of projects. The enterprise focus creates integrated data and process models that form the basis of separate systems developed together into a common architecture or framework.

Engineering Approach

Information engineering can also be contrasted with software engineering. Software engineering focuses on structured techniques for specifying, designing and constructing application programs. Information engineering, on the other hand, focuses on information that is stored and maintained by various application programs. It also is concerned with structured analysis and programming, but embraces development using nonprocedural languages, specification languages, application generators and computer-assisted design (CAD) in order to minimize development effort.

Data Sharing

Data concepts, rather than programming ones, form the basic premise of information engineering. Data are stored and maintained by processes that create, modify and delete the data. All data in an enterprise should be captured and recorded with controls on data accuracy and format. Data are seen as central to information system development because data may be used in several information systems concurrently, stored in different ways, distributed and often updated and modified by way of network links and nodes.

Another basic premise of information engineering is that data types do not change often. Entity types do not usually change, but new types may occasionally be added. Attributes that describe entity types do not often change, either. The instances of entity types and the values of the corresponding attributes will change often but the basic structure of the data model does not change.

Automated Tools

One development that makes information engineering feasible is the creation of advanced development tools for business systems planning, analysis, design and construction. In order to build elaborate, extensive networks of integrated information systems, we need automated support in order to handle the complexity and speed up routine tasks. We not only can change the way that systems are generated but also handle many more types and numbers of systems. Organizations must automate not only the management