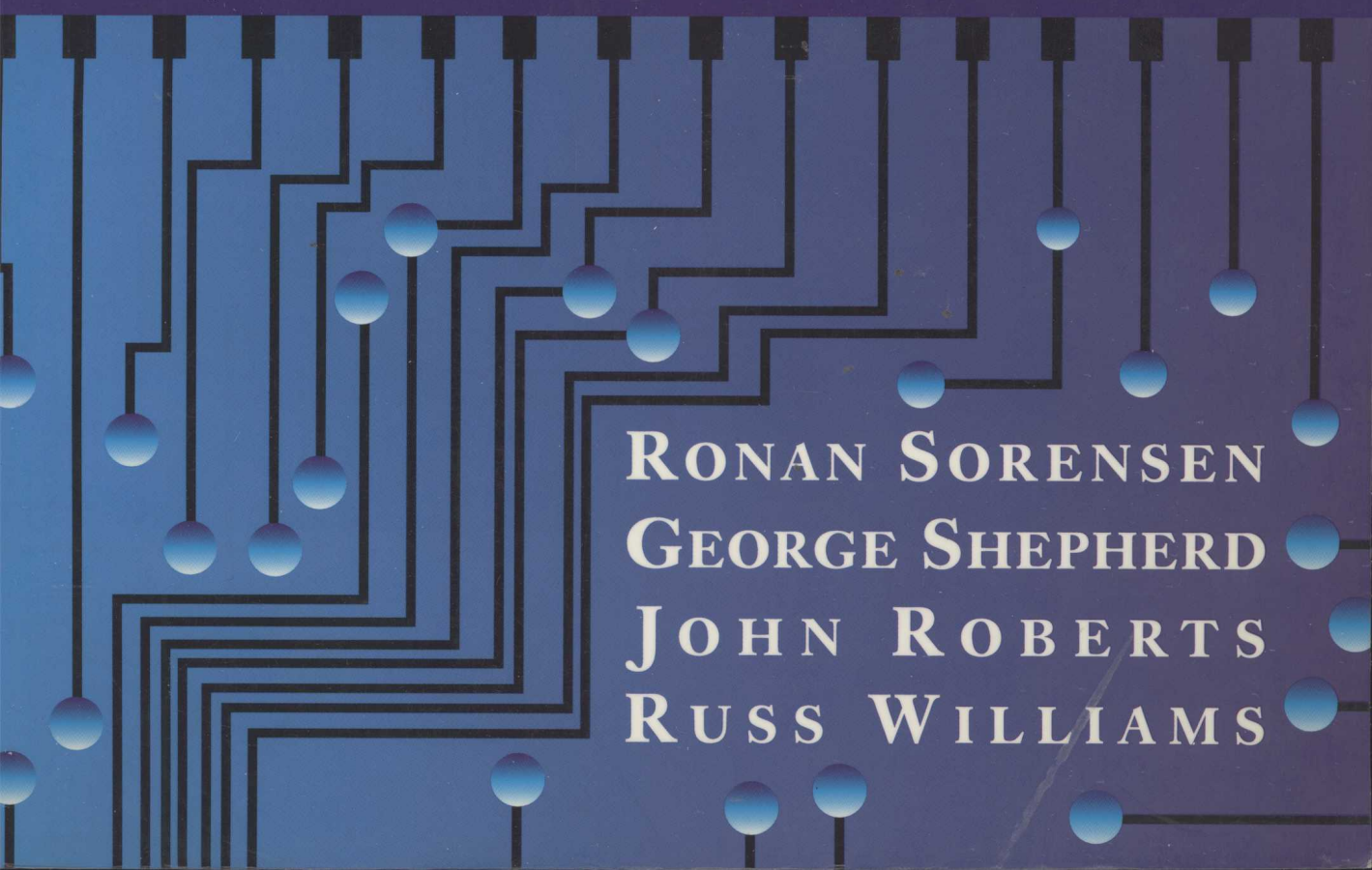




APPLIED



Developing People-Oriented Software Using C#



RONAN SORENSEN
GEORGE SHEPHERD
JOHN ROBERTS
RUSS WILLIAMS

Applied .NET

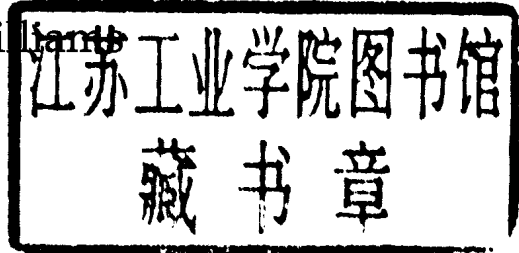
Developing People-Oriented Software Using C#

Ronan Sorensen

George Shepherd

John Roberts

Russ Williams



◆Addison-Wesley

Boston San Francisco New York
London Toronto Sydney Tokyo Singapore Madrid
Mexico City Munich Paris Cape Town Hong Kong Montreal

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book and we were aware of a trademark claim, the designations have been printed in initial capital letters or all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

Copyright © 2002 by Ronan Sorensen, George Shepherd, John Roberts, and Russ Williams

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher. Printed in the United States of America. Published simultaneously in Canada.

Portions from Microsoft PressPass reprinted with permission from Microsoft Corporation. Copyright © 2001 Microsoft Corporation. All rights reserved.

The publisher offers discounts on this book when ordered in quantity for special sales. For more information, please contact:

Pearson Education Corporate Sales Division
One Lake Street
Upper Saddle River, NJ 07458
(800) 382-3419
corpsales@pearsontechgroup.com

Visit us on the Web at www.awl.com/cseng/

Library of Congress Cataloging-in-Publication Data

Applied .NET : developing people-oriented software using C# / Ronan Sorensen . . . [et al.]
p. cm.

ISBN 0-201-73828-7 (alk. paper)

1. C# (Computer program language)
 2. Application software—Development
- I. Sorensen, Ronan

QA76.73.C154 A67 2002
005.2'762--dc21

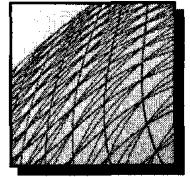
2001045747

Text printed on recycled paper.

1 2 3 4 5 6 7 8 9 10—CRS—05 04 03 02 01

First printing, October 2001

Preface



Like a storm that has built energy out at sea, the first waves of a new era of computing have begun to pound the “beaches” of software development. The forces behind this storm have been building for some time and as the waves make land, a somewhat unsuspecting industry braces itself and prepares to survive the fury.

Although this is a rather dramatic way to characterize the current state of affairs in the computer industry, it is nonetheless accurate. Never before has so much technology been made available to such a large community of developers in such an integrated and distributed fashion. Just as someone standing on a beach can tell there’s a storm approaching, as a developer you can recognize that a change in the industry is underway. Undoubtedly, you’re trying to figure out what all this means and how best to prepare yourself. This book is meant to provide a level of understanding that will prepare you not only to survive in this new era of development but to actually thrive. The information here will help you understand the .NET technologies and show you how they all fit together in a way that will enable you to effectively build next-generation solutions. This book conveys these ideas through the development of several .NET applications using C#.

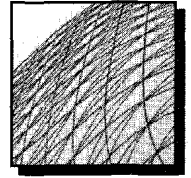
Applied .NET offers a people-oriented perspective on the new forces changing software development and a set of principles that can be applied to building effective Internet software. We use the term *people oriented* to describe the new wave of software that is approaching as it captures the dynamism that stirred up the .NET storm. The origin of this term dates back several years to a book authored by Ronan in 1998—*Inside Microsoft Windows NT Internet Development*. Part I of that book introduced the new paradigm of people-oriented programming and the concepts embodied within this type of software. The second part of the book explained how earlier technologies could be used to develop systems adhering to these principles.

The .NET technologies take such a significant step closer to the goals and ideals first presented in that earlier work that our choice of a subtitle for this book was natural—*Developing People-Oriented Software Using C#*. This serves our desire to present a perspective on how .NET can be applied to build a new and very exciting class of software. Therefore, although this book applies .NET technology, the objective of that application is to create something more transcendent, which is formally referred to as *people-oriented software*.

In the period of time since *Inside Microsoft Windows NT Internet Development* was published, the ideas it presented have matured and sharpened as a result of various discussions among the authors of this current book. Some of those discussions produced more heat than light, but in the end we are all in agreement that the principles laid out in this book are the right ones and that the future will no doubt be people oriented. This became even more evident just recently when Microsoft announced their HailStorm initiative. Any doubts about the people-oriented perspective ended with that announcement. We are entering a new era in which people will not have to be computer oriented to use software—software will be oriented toward how people actually live. Software will be running many everyday devices, and all of them will be connected in unimaginable ways. The core theme running through it all will be how software is embedded within society and oriented to the people who will use it.

From a people-oriented perspective, .NET is a means to an end rather than an end in itself. No doubt, other books will go into more detail in certain areas of .NET than this one does, and they will be very useful in that regard. This book, however, tries to strike a balance between theory and practice so that we can show you not only how to apply .NET but also what you can achieve as a result of that application. As useful as we think the perspectives and principles contained in *Applied .NET* are, we just don't stop there. The book will actually show you how to apply what you've learned by building realistic .NET applications—it takes a practical look forward.

Acknowledgments



The authors, collectively, would like to thank the following people:

- The Addison-Wesley team: Kristin Erickson, Curt Johnson, Chris Kief, Chanda Leary-Coutu, Marilyn Rash, Cathy Comer, Dianne Wood, Karin Hansen, and Mark Bergeron of Publishers' Design; and a special thanks to Stephane Thomas, our fearless editor.
- All our colleagues at Plural for their encouragement, and special thanks to James Watkins and Connie Hughes who assisted with the promotion of this book.
- Miki Bell who provided some of the artwork.
- Sanjay Parthasarathy, Nelson Rossa, Connie Sullivan, and Rodney Miller from Microsoft for their assistance and comments.
- Rob Howard, John McGuire, Greg Hack, Daryl Richter, Don Browning, Maxim Loukianov, and Christophe Nasarre for their technical reviews.

In addition we include the following individual acknowledgments.

First and foremost I would like to thank my wife Irene and my three daughters Mary, Catherine, and Sophia for allowing me to write another book. This time they knew what to expect and so I am particularly grateful that they continue to put up with my ideas and encourage me to write about them. I would also like to thank my extended family in Ireland, America, and Italy from whom I have received faith, hope, and love. Finally, I would like to thank my fellow authors George, John, and Russ for their friendship and for the privilege of writing this book with them.

R.S.

I express my most grateful appreciation to my family, Sandy Daston and Ted Shepherd, for being supportive and gracious while I toiled away on another book. Thanks also to DevelopMentor for being a great training and thinking place for developers. Thanks to Patrick Shepherd for being a great sounding board, participating on the other side of the modern software fence (I kneel toward Redmond while he kneels toward San Jose). Finally, great thanks are due to Ronan Sorensen, John Roberts, and Russ Williams without whose efforts this book would not have been possible.

G.S.

I would like to thank Pete Nash, Mike Cabrera, and Jason Cuplin for their assistance and support. I thank my fellow authors, whom I have had the distinct pleasure of knowing and working with for years. I thank Ronan Sorensen for creating opportunities over the last four years for many fine adventures in software development and publication, including this one. Thanks to Russ Williams for being a continual source of encouragement and inspiration. Thanks to George Shepherd for encouraging me to write back in the early nineties. Working with you all has been the highlight of my software career. Most of all, I wish to thank my wife Sue and my sons Daniel, Luke, and Michael for their patient and understanding support.

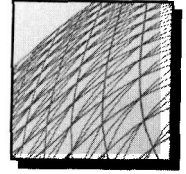
J.R.

I would like to start by thanking my family for the sacrifices they made while I worked on this book. The time spent apart, the baseball games missed, the family gatherings I was unable to attend, and my fatigue and distractedness are just some of the things that I have had to ask them to forgive. I thank my children Ryan, Chase, and McKenzie for their understanding and especially my wife Gina who was so supportive and provided me with much-needed encouragement at just the right time. They are a gift from God for which I am eternally grateful. I would also like to thank my co-authors, all of whom I have worked with and greatly admired. It has been an honor to be a part of this book and I thank them for the quality of the contributions they have made. Special thanks go to Ronan who was the point man on this effort. His hard work, leadership, experience, insight, and instinct have shaped the book that you now hold. Finally, I would like to thank my mother and father for their overconfidence in me through the years. It was an offhand conversation with my father in which he described his passion for programming that gave me the "bug," as he called it, for development. That event not only was the beginning of my chosen career, but it also taught me that there really are no *small* conversations with kids. Mom and dad, I love you both very much.

R.W.

While we have made every effort to avoid inaccuracies in this book, some may be uncovered after it is printed. The Web site at <http://www.people-oriented.net> will provide contact details for reporting errors, and it will also list any corrections or updates.

Contents



Chapter 1 People-Oriented Software 1

The People-Oriented Paradigm 3

Universalization 4

Collaboration 5

Translation 5

A .NET Approach 6

.NET and Universalization 6

.NET and Collaboration 9

.NET and Translation 12

System Interoperability 12

Contract Transformation 13

Conclusion 14

Chapter 2 Applied People-Oriented Software 15

People-Oriented Design 15

People-Types: "Design with Attitude" 16

Miner 17

Conductor 18

Linguist 18

Apply the Concepts: The InternetBaton Application 19

InternetBaton Application Features 19

Universalization Design: Mining the Runtime 21

Step 1: Sign-Up Page with Entry Validation 22

Step 2: Authentication and Authorization of Users 31

Step 3: Creation of New Baton Projects 37

Collaboration Design: Conducting the Orchestra 50

Step 4: Integration of InternetBaton with Other Web Services 54

Translation Design: A Linguist's Delight 62

Step 5: Baton Synchronization 62

Step 6: Translation of Baton Metadata 66

Conclusion 69

Chapter 3 C# 71**This Time, It's Personal 71****What Is C#? 73****What's So Special About C#? 74***Contemporary Perspective 74**It's Elegant 74**It's Object Oriented 75**It's Component Oriented 76**People-Oriented Perspective 77**C# and Universalization 77**C# and Collaboration 78**C# and Translation 79***Language Tour 80***The Basics 81**Program Structure 82**Namespaces 82**Assemblies 85**Variables 85**Expressions 88**Statements 91**Types 99**Value Types 99**Arrays Types 104**The object Type 106**Type Harmony 106**Classes 107**Inheritance 108**Members 109**Constructors 111**Destructors 113**Method and Method Overloading 114**Properties 116**Operators 117**Events 118**Indexers 120**Interfaces 122**Struct 125**Enum 126**Attributes 126**Exceptions 127***Conclusion 131**

Chapter 4 Applied C# 131**ManagedSynergy 131**

The Vision 132

The Functionality 132

The Design 133

*Universalization 134**Collaboration 135**Translation 136*

The Implementation 137

*Opening an Existing Project 138**Creating a New Project 144**Adding a Project Item 150**Deleting a Project Item 154**Checking Out an Item 154**Viewing a Project Item 156**Checking In an Item 157**Reviewing an Item 160**Viewing an Item's Properties 163**Invoking Administration Services 166**Dynamic Status Updates and Overnight Project Replication 166***Conclusion 169****Chapter 5 The Common Language Runtime 171****Windows and Components 173**

Static Libraries 173

Dynamic Link Libraries 174

*Implicit Linking 175**Explicit Linking 178**Upsides and Downsides 180*

COM Tries to Fix It 180

*COM Interfaces 181**COM Objects 185**COM Classes and Class Objects 186**Loading COM DLLs 187**IDL and Type Information 189**COM+ 190**What Is Right in COM? 191**What Is Wrong in COM? 191***Enter the Common Language Runtime 192**

A Pervasive Type System 193

Types Are Fundamental 194

The Common Type System	195
<i>Value Types</i>	198
<i>Value versus Reference</i>	198
The Common Language Specification	198
Boxing	198
How Types Map to C#	198
<i>Fields</i>	199
<i>Methods</i>	199
<i>Properties</i>	199
<i>Constructors</i>	199
Assemblies	200
Assemblies and Modules	201
The Manifest	201
<i>Private versus Public Assemblies</i>	202
.NET Versioning	203
Life within the CLR	203
IL and JIT Compiling	204
.NET Garbage Collection	205
<i>Finalization</i>	206
Threading and the CLR	207
AppDomains	207
Interoperability	208
<i>Platform/Invoke</i>	208
<i>TLBEXP and TLBIMP</i>	208
Conclusion	209

Chapter 6 Applied Runtime 211

Building Assemblies and Applications	211
The Command Line	211
Makefiles	213
Building Projects Using Visual Studio.NET	214
Examining the Manifest	215
Using ILDASM	215
Deployment and Versioning	218
Global Cache	221
Loading Assemblies and Versioning	223
More on Configuration Files	224
Garbage Collection	225
Effects	225
Deterministic Finalization	225

Threading and the CLR	227
Creating Threads	227
Synchronization	229
Method-Level Locks	233
Interoperability	234
Platform/Invoke	234
Interoperating with COM	236
TLBIMP	236
TLBEXP	238
Windows Forms	241
The Forms Class	242
Handling Events	244
Graphics and Rendering	244
Conclusion	244

Chapter 7 ASP.NET Up Close 247

Connective Tissue	248
The Road to ASP	248
Classic ASP versus ASP.NET	249
Deemphasizing ISAPI	250
ASP.NET: A Common Language Runtime Citizen	250
System.Web.UI.Page	251
System.Web.UI.Page Fundamentals	255
ASP.NET Connection Object Model	257
Mixing ASP.NET and C#	258
ASP.NET Configuration Files	260
Web Forms	262
Custom Server-Side Controls	262
Extending the Browser	263
Server-Side Rendering	263
Control Life Cycle	264
Reasons to Use a Custom Server-Side Control	266
Web Services and ASP.NET	266
Web Methods and ASP.NET	267
Service Description Language and ASP.NET	267
Invoking Web Methods	268
Optimizations: ASP.NET Caching	268
Output Caching	268
Data Caching	268

Managing Session State	269
Conclusion	270

Chapter 8 Applied ASP.NET 271

User Interface Controls and the Web	271
HTML Controls	276
Web Controls	277
Web Forms and Visual Studio.NET	280
State Management for Web Applications	288
Application State	289
Session State	290
Session Configuration	291
Caching	293
Output Caching	293
Data Caching	294
HTTP Handlers	297
Conclusion	298

Chapter 9 .NET Enterprise Servers 301

.NET Enterprise Servers and People-Oriented Software	301
Universalization	301
Collaboration	301
Translation	302
Making It All Work Together	302
Point of Critical Solution Mass	303
.NET Enterprise Servers and .NET	305
Role of XML	305
Foundation of Modern Interoperability	305
Structured Data Exchange	306
Business Document: XML	306
Business Document Specification: XML Schema	306
Document Translation: XSLT	306
Business Process: XLANG	307
Remote Object Invocation: SOAP	307
Asynchronous Messaging: SOAP	307
Description of Web Services: WSDL	307
Basics of XML	308
Design Goals	308

Documents	309
Elements	309
Attributes	310
Well-Formed Documents	310
Validity	310
Entity References	310
CDATA Sections	311
Processing Instructions	311
Comments	311
Namespaces	311
XPath	313
XLink	313
XPointer	313
Processing Models	314
Memory Tree	314
Event-Driven	314
Sequential Navigation Based	314
SOAP	315
Description and Purpose	315
Maximized Interoperability	315
Distributed Internet Computing RPC Mechanism	316
Document Messaging Mechanism	316
Operation Over the Internet through Firewalls	316
Definition	316
SOAP Envelope	316
SOAP Headers	316
SOAP Body	317
Call and Response Pattern	317
Data Types	317
Parts of the Implementation Problem	318
Microsoft Implementations	318
SOAP SDK 1.0	318
SOAP SDK 2.0	319
VisualStudio .NET	319
BizTalk Server Essentials: Solving the EAI Problem and Beyond	319
BizTalk Orchestration	320
BizTalk Orchestration Designer	321
Business Process Design	322
Port Implementation	325
Schedule Compilation	327
Schedule Instantiation and Execution	327

BizTalk Messaging	328
Abstractions	329
Submitting Documents	331
Channel Firing	333
Translation	333
Extensibility Framework: BizTalk Hooks	333
Custom Preprocessor	335
Custom Parser	335
Custom Functoid	335
Custom Serializer	336
Custom Transport	336
BizTalk Development Tools	337
BizTalk Editor	337
BizTalk Mapper	337
BizTalk Orchestration Designer	339
BizTalk Administration Tools	339
BizTalk Server Administration	339
BizTalk Messaging Manager	340
BizTalk Document Tracking	340
BizTalk Messaging Object Model	341
Creating a BizTalkConfig Object	341
Creating an Organization	342
Creating a Document	342
Creating a Port	342
Creating a Channel	343
Deleting an Object	343
Issues Addressed	343
Interoperability	343
Transport Protocols	344
Business Document Definitions (Schemas)	344
Business Document Validation	344
Business Process Definition	344
Business Document Translation	344
Integration with Legacy Systems, Internal Applications, and Cross Enterprise Applications	344
Standards Support	345
Encryption and Secure Communications over the Internet	345
Management of the Execution of Business Processes	346
Long-Lived Transactions	346
Tracking of Business Transaction Status	347
Prepackaged Capabilities to Minimize Custom Development	347

<i>Extensible Design</i>	347
<i>Architecture Issues with Current Implementation</i>	347

Commerce Server Essentials 348

Commerce Server Architecture	348
Continuous Improvement Cycle	350
Business Processing Pipelines	350
Profile System	350
Targeting System	352
Product Catalog System	352
Business Analytics System	353
Solution Sites	354

Supplier Enablement Tool Kit 355**Integration Points 356**

Internet Information Server to BizTalk Server Orchestration	356
Internet Information Server to BizTalk Server Messaging	356
<i>From HTTP Request to BizTalk Messaging via Direct Integration</i>	356
<i>From HTTP Request to BizTalk Messaging via Message Queuing</i>	
<i>Receive Function</i>	357
<i>From HTTP Request to BizTalk Messaging via File Receive Function</i>	357
Commerce Server to BizTalk Server	359
<i>Integration to BizTalk Server Messaging</i>	359
<i>Integration to BizTalk Server Orchestration</i>	359

Conclusion 361**Chapter 10 Applied .NET Enterprise Servers: Order Fulfillment with an Outside Vendor 363****Order Processing Pipeline 365****Business Process Definition 365****Port Implementations 366****Integration with Business-to-Consumer Site 373**

Order Translation to Outside Vendor Format	374
Delivery to Outside Vendor's BizTalk Server	375
Commerce Site Status Update and Consumer Notification	375
Outside Vendor BizTalk Processing	376
<i>Ship Notice Handling</i>	377
<i>Charge Credit Card</i>	377

Conclusion 378**Index 379**

People-Oriented Software

The Internet has brought software to the people. For the first time in history, ordinary people all over the world are using software to connect to each other. This trend will surely continue as Internet connectivity enters the realms of television, radio, telephone, personal digital assistant (PDA) technology, and the automobile. In addition, people's lives are becoming the primary focus of software—either directly through human interaction via Web-user interfaces or indirectly through business-to-business (B2B) communication targeted at serving human needs. The increasing connectivity of the populace through software combined with software's more specialized focus on people is revolutionizing software design.

The software of the past focused on modeling the operation of *things*, which gave rise to the object-oriented movement. Although today people could be viewed as just another collection of objects in an object-oriented world, this approach would be impractical and likely fail. There is simply no plausible way to model the dynamic interactions and forces within our society using object-oriented design. Social interaction involves issues such as the use of freedom, multicultural preferences, mobility, unpredictability, and geographical location, just to name a few. Simply put, society cannot be adequately represented using the abstraction of an object model. The real world of people is radically different from the world of things, as philosopher Karl Wojtyla (better known as Pope John Paul II) pointed out years ago:

The world in which we live is composed of many objects . . . As an object, a man is "somebody"—and this sets him apart from every other entity in the visible world, which as an object is always only "something." Implicit in this simple,