Java™ CAPS Basics

Java™ CAPS Basics

Implementing Common EAI Patterns

Michael Czapski
Sebastian Krueger
Brendan Marry
Saurabh Sahai
Peter Vaneris
Andrew Walker

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Visit us on the Web: www.prenhallprofessional.com

Library of Congress Cataloging-in-Publication Data:

Java CAPS basics : implementing common EAI patterns / Michael Czapski ...
[et al.].
p. cm.
Includes bibliographical references and index.
ISBN-13: 978-0-13-713071-9 (hardover : alk. paper)
ISBN-10: 0-13-713071-6 (hardcover : alk. paper)
1. Java (Computer program language) 2. Enterprise application integration (Computer systems) I. Czapski, Michael.
QA76.73.J38J3633 2008
005.13'3—dc22 2008007526

ISBN-13: 978-0-13-713071-9
ISBN-10: 0-13-713071-6
Text printed in the United States on recycled paper at Courier in Westford, Massachusetts.
First printing April 2008

Contents

Preface

In their book *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions* [EIP], Gregor Hohpe and Bobby Woolf elaborate on the subject of enterprise application integration using messaging. They present, discuss, and illustrate over sixty EAI design patterns. These patterns, they believe, are key patterns most designers of EAI solutions will use when building enterprise integration solutions. Most examples in [EIP] use raw C# and raw Java to illustrate details of EAI patterns under discussion. Most of these patterns can be implemented succinctly, elegantly, and comprehensively using tools and technologies provided in the Sun Java Composite Application Platform Suite [Java CAPS].

This book is about implementing selected enterprise integration patterns, discussed in [EIP], using Java CAPS as the means to building practical enterprise integration solutions. It bridges the gap between the somewhat abstract pattern language and the practical implementation details. It is designed for integration architects, solution architects, and developers who wish to quickly implement enterprise solutions with Java CAPS. It discusses how EIP patterns can be implemented quickly and efficiently by leveraging the Java CAPS tools and the authors' field experience.

While this book discusses Java CAPS implementation of [EIP] patterns, it does not discuss the patterns in depth. It is assumed that you are already familiar with the subject and need to apply the theoretical knowledge using Java CAPS.

This book is also about basics of the essential Java CAPS Suite components, based on the premise that you cannot apply patterns if you cannot effectively use the tools with which to do it. Since the complete Java CAPS offering has so many components, including ones that are not essential to integration, that this book elaborates only on the basic integration tools: eGate, eInsight, eWays, and Java Message Service (JMS).

This book also provides information you may need to effectively use Java CAPS. A considerable amount of Java CAPS-related material, provided in the text, is not published anywhere else.

The companion text, *Java™ CAPS Basics: Implementing Common EAI Patterns* (located on the accompanying CD-ROM), provides over 60 detailed examples that

illustrate concepts and patterns under discussion. Some examples are high level, illustrating specific points. Other examples follow a step-by-step approach.

Java CAPS projects discussed and developed as examples are available for import and perusal.

How This Book Is Organized

The book is divided into three sections. Section I, "Preliminaries," contains chapters that discuss integration and background Java CAPS topics, including enterprise integration styles, Java CAPS architecture, and project structure and deployment.

Section II, "Patterns Review and Application," covers most [EIP] patterns with discussion of Java CAPS approaches to implementing them. This section includes chapters dealing with Message Exchange patterns, message correlation, messaging infrastructure, message routing, message construction, message transformation, messaging endpoints, and system management patterns and concepts. While discussing Java CAPS implementation of specific patterns, relevant Java CAPS concepts and methods are also discussed. When discussing implementations of the Message Sequence pattern, for example, Java CAPS concepts of JMS serial mode concurrency, Sun SeeBeyond JMS Message Server FIFO modes, and serializing eInsight Business Processes via JMS and XA are also discussed.

Section III, "Specialized Java CAPS Topics," discusses non-pattern matters of importance like solution partitioning, subprocess and Web Services implementation, management, reusability, scalability and resilience options, and others that are not covered elsewhere. This section also covers security features of Java CAPS.

The accompanying CD-ROM contains over 60 detailed examples implementing most of the patterns and concepts under discussion as well as two complete example solutions using many of the patterns discussed and illustrated in both books. The CD-ROM also contains a detailed practical walkthrough of generation and use of cryptographic objects such as X.509 Certificates, PKCS#12 and JKS Keystores, and related matters.

About the Examples

Conventions

Java CAPS Enterprise Designer (eDesigner) is a NetBeans-based Integrated Development Environment (IDE), which developers use to design and build Java CAPS integration solutions. The vast majority of tasks can be accomplished in eDesigner

by means of manipulating components represented by graphical objects, connecting graphical objects with lines, filling information in dialog boxes and property sheets, and choosing components in drop-down menus. The intention was to make development of integration solutions easy for business analysts and similar persons whose coding skills might not be up to the task in nongraphical environments. Since development of Java CAPS solutions results in production of J2EE Enterprise Applications, this graphical orientation might come as a bit of a surprise to hardcore J2EE developers used to writing raw Java and the fine-grained control they exercise through deployment descriptors and other J2EE artifacts. Be that as it may, you will find most Java Collaboration examples shown using Java source code rather than its graphical equivalent. This is principally to make the samples concise, clearly showing essential parts of each solution, and to minimize wasting space on graphics that add no particular value to the discussion. Every one of the Java Collaborations could have been shown in "Standard mode," but that would have required numerous pictures, pretty much one for each Java statement, to illustrate what just a few lines of Java code can show in just a few lines. Figure P-1 shows an example of a Java Collaboration in Standard mode.

Evident are several lines of pseudocode in the Business Rules pane and just one mapping in the Business Rules Designer pane.

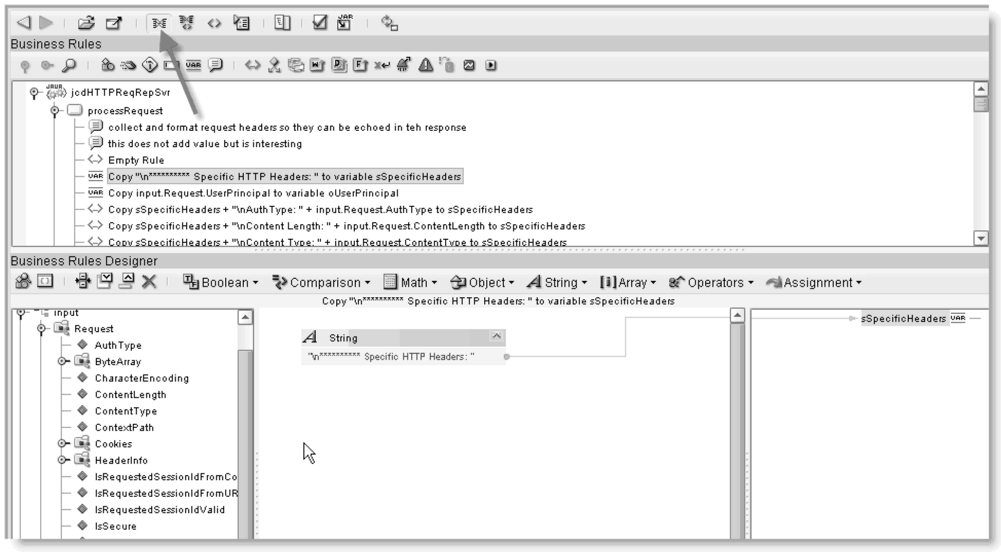

FIGURE P-1: Java Collaboration in Standard mode

In contrast, the same Java Collaboration in Source Code mode is much more illuminating, as seen in Figure P-2, as it shows all there is to know about 30 or so lines of code all at once.

Since switching between Standard mode and Source Code mode is a button-click away, we chose to use Source Code mode for Java Collaboration examples.

eInsight Business Processes are much easier to understand when presented in the graphical view, appearing similar to what is shown in Figure P-3. Object icons, taken from the Business Process Modeling Notation, are quite pleasant to look at, in this author's opinion.

In contrast, working directly with BPEL4WS XML source, an example of which is shown in Figure P-4, which is possible, is in this author's opinion bordering on cruel and unusual punishment, just as working directly with any other XML-based procedural language would be.

So, Java Collaborations are mostly shown in the Source Code mode, and eInsight Business Processes are mostly shown with the Business Rules Designer.

How you work with the tool is still up to you.

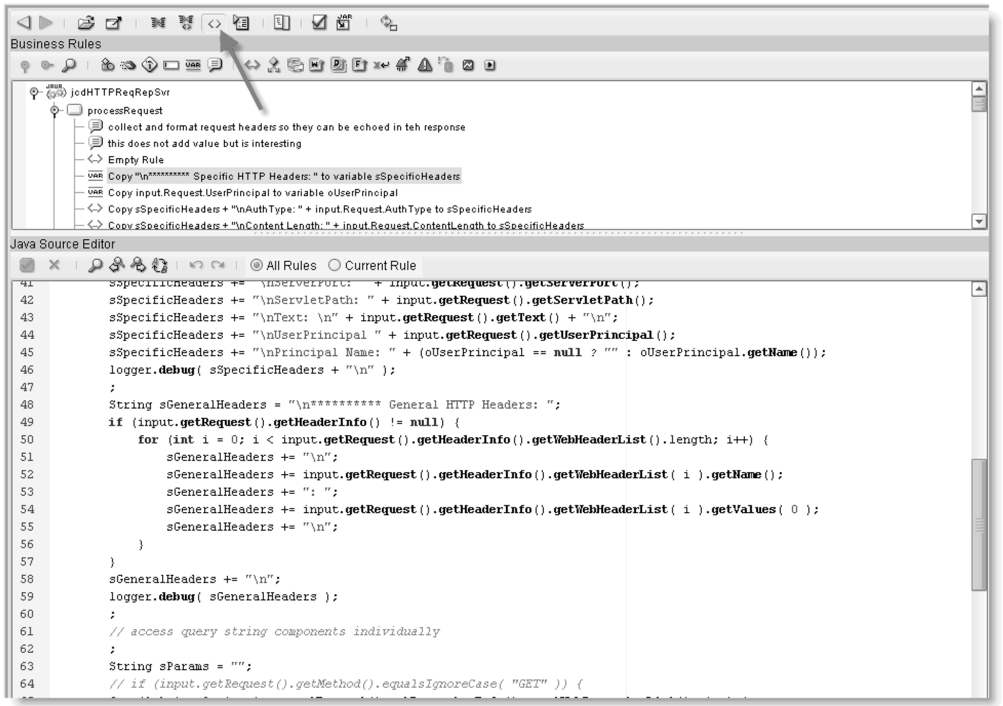

FIGURE P-2: Java Collaboration in Source Code mode

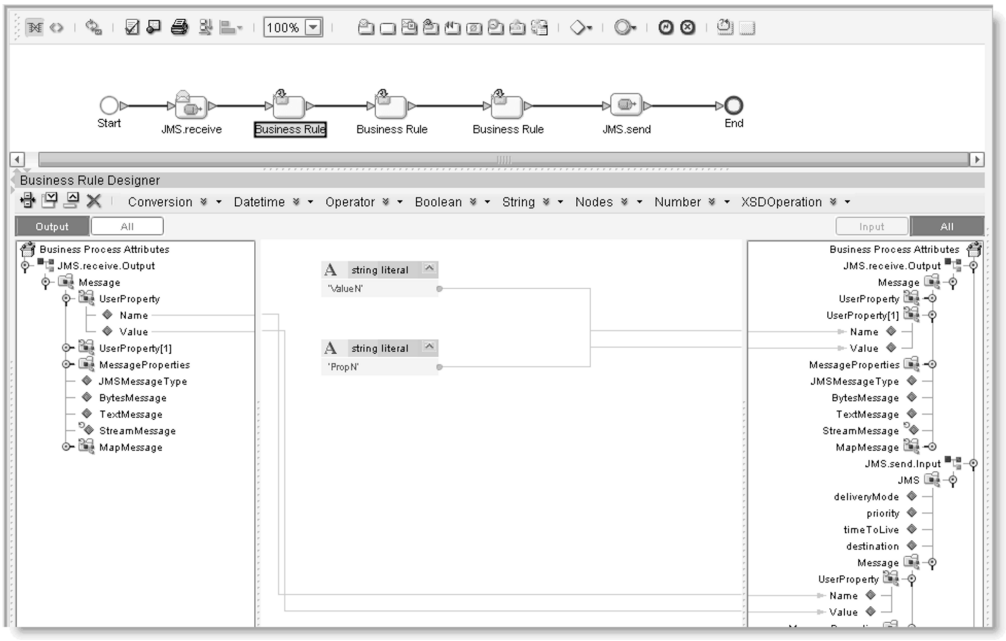

FIGURE P-3: eInsight Business Processes in the graphical view

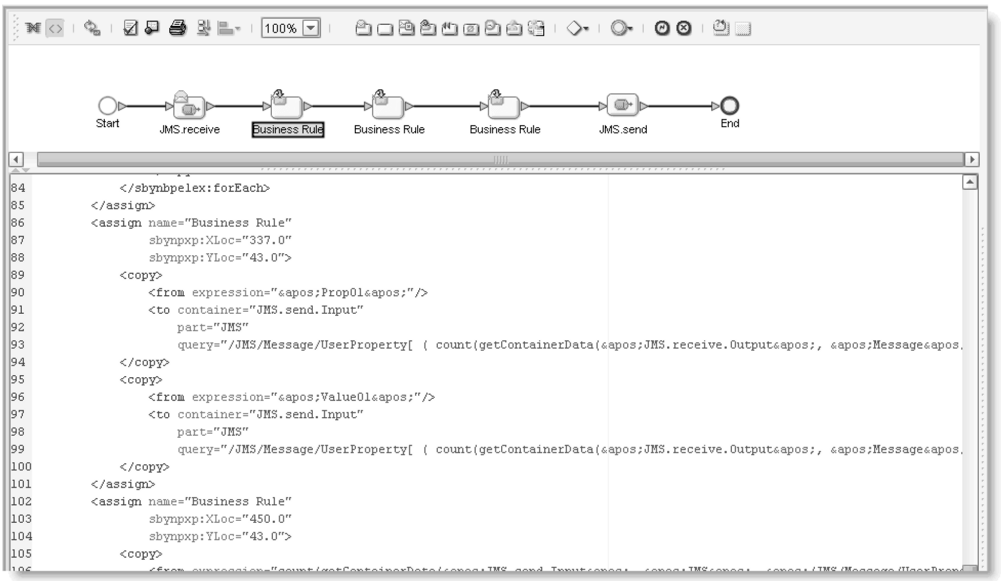

FIGURE P-4: BPEL4WS XML source

LIST OF ILLUSTRATIONS AND EXAMPLES

To include both discussion and all relevant examples in one book would have made it over 1,000 pages in length—too large for printing. As a consequence, the original manuscript has been broken up into two parts. Part I, *Java™ CAPS Basic: Implementing Common EAI Patterns* (this book) discusses Java CAPS facilities, focusing on their application to implementation of EIP patterns, with high-level illustrations and references to detailed examples in Part II, which is contained on the CD-ROM that accompanies this book. Part II provides both detailed illustrations for most of the patterns as well as two completely worked-through Java CAPS–based case study solutions that implement a number of the patterns discussed in this book. Part II also contains a chapter dealing with cryptographic objects used to configure security-related aspects of the suite.

Illustrations and examples from Part II are listed in Table P-1, with section headers in the order of appearance. Of the over 60 examples, most are developed in a step-by-step manner, deployed, and exercised. In most cases, results of execution are shown and discussed.

TABLE P-1: Illustrations and Examples in Part II

Chapter/Section	*Example Topic*
Hello Java CAPS World	Introductory step-by-step example. Basic file transfer projects with separate implementations using eGate and eInsight.
Event Message Using Scheduler	Event Message pattern implementation. Basic scheduled solution using Event Message triggered by a Scheduler eWay.
External Scheduler Example	Event Message pattern implementation. Scheduled solution using external scheduler and an external TCP Sender client injecting an Event Message through a TCP Server eWay.
JMS Request/Reply Invoker for eInsight	Request/Reply pattern implementation. New Web Service Java Collaboration for invoking JMS Request/Reply functionality from an eInsight Business Process.

TABLE P-1: Illustrations and Examples in Part II *(continued)*

Chapter/Section	*Example Topic*
JMS Request/ Response Auction Pattern	Request/Reply pattern implementation. Java Collaboration and JMS Request/Reply–based implementation of an Auction pattern where the fastest responder wins.
HTTP Request/ Response	Request/Reply pattern implementation. A series of HTTP requestor and HTTP responder implementations using both HTML and XML payloads, prefaced by a recap of HTTP principles and mechanics.
SOAP Request/ Response	Request/Reply pattern implementation. Specialization of a HTTP request/response implementation using explicitly constructed and parsed SOAP XML messages as requests and responses.
Web Service Request/ Reply	Request/Reply pattern implementation. Web Service request/response implementation as an example of a Request/Response pattern using both eInsight Business Processes and Java Collaborations.
JMS Serial Mode Concurrency	Message Sequence pattern. Using Sun SeeBeyond JMS Message Server facilities for implementation of message sequence preserving solutions.
Sun SeeBeyond JMS FIFO Modes	Using Sun SeeBeyond JMS Message Server facilities for implementation of message sequence preserving solutions. Series of examples demonstrates the impact of different Sun SeeBeyond JMS Message Server FIFO modes on message sequence.
Serializing Business Processes with XA	Message Sequence pattern. Examples illustrating the impact of imposing XA transactionality on eInsight Business Processes and how it affects message sequence preservation (new in v. 5.1)
Message Expiration	Java Collaboration and JMS-based example illustrating Message Expiration.

(continues)

TABLE P-1: Illustrations and Examples in Part II *(continued)*

Chapter/Section	*Example Topic*
Batch Local File Streaming	Data streaming examples using Batch Local File eWay with discussion of buffering and its impact on message throughput.
eTL Streaming	Very basic eTL example streaming data from a flat file to a database table and a functionally equivalent Java Collaboration example.
Temporary JMS Destinations	Anti-example illustrating the use of an explicitly created JMS temporary destination.
Static Selector	Example illustrating the use of static JMS selectors.
Dynamic Selector	Example illustrating the use of dynamic JMS selectors, constructed at runtime and used in a Java Collaboration to explicitly choose messages to receive.
Resilient JMS with JMS Grid	Example of a simple JMS Grid-based automatic JMS Client failover.
JMS Message Body Formats	Example collaboration that inspects JMS header properties to determine JMS message body format and branch.
Dead Letter Channel in 5.1.2	Example exercising JMS Redelivery Handling functionality with undeliverable message being delivered to a Dead Letter Queue.
eInsight XA Transactionality	Example inducing success and failure outcomes that demonstrate XA transactionality of an eInsight Business Process with side effects in non-XA-capable resources.
eInsight Persistence	Illustration of eInsight persistence, eInsight monitoring, and eInsight restart-recovery of in-flight process instances.
Resequencer: Basic Version	Simple resequencer with memory-based message buffer.
Resequencer: Persisted Version	More sophisticated resequencer with RDBMS-based message buffer and message sequence persistence.
Routing Slip	Routing slip-based message routing solution using JSM and Java Collaborations.
JMS User Properties Envelope Wrappers	Series of examples demonstrating Envelope Wrapper pattern implemented using JMS message header properties in Java Collaborations and eInsight Business Processes.

TABLE P-1: Illustrations and Examples in Part II *(continued)*

Chapter/Section	*Example Topic*
Content Enricher	Content Enricher implementation using eInsight and Oracle eWay to receive a Purchase Order, enrich it with pricing information for an Oracle database, and produce an Invoice.
Polling File System	Series of examples polling local file system using Scheduler eWay–driven and Batch Inbound–driven Batch Local File eWay to implement polling solutions.
Polling JMS Destination	Try-wait-retry FTP delivery solution implementing JMS Destination polling for retry scheduling.
Durable Subscriber	Example illustrating the concept of a JMS topic durable subscriber and behavioral differences between nondurable and durable subscribers.
Idempotent Receiver	Example illustrating message duplication detection–based Idempotent receiver solution.
Multi-Input Service Activator	Example using the OpenTravel Alliances XML Schema documents to implement a Service Activator solution that allows the business service to be activated through a file submission, JMS message submission, and Web Services invocation.
Monitoring eInsight-based Solutions	Example illustrating eInsight persistence, persistence for reporting, and runtime monitoring of eInsight Business Process instances.
Simple Alert Processor for a JMS Channel	Example implementing a simple solution that receives and processes Alert Agent alerts delivered through the JMS Alert Channel.
Catching "Uncatchable" Exceptions	Example using Alert Agent infrastructure to catch and process exceptions that occur outside the Java Collaborations and Business Processes and therefore cannot be caught and processed in JCDs or BPs. Catching and processing a database connectivity exception is the subject of this example.
Programmatic Management	Example Java Collaboration that uses the JMX instrumentation to programmatically start and stop a specified Java CAPS component, such as another Java Collaboration or a Business Process service.

(continues)

TABLE P-1: Illustrations and Examples in Part II *(continued)*

Chapter/Section	*Example Topic*
JMS Latency	Java Collaboration and eInsight Business Process examples illustrating calculation of JMS message delivery latency.
eInsight Correlation Processor: First Cut	Example of an incorrect, naïve implementation of eInsight correlation.
eInsight Correlation Processor: Second Cut	Example of correct implementation of a simple eInsight correlation with a simple correlation key.
Derived Correlation Identifiers	Example of a more complex eInsight correlation with a structured, message-derived correlation key based on subprocess-based correlation key derivation solution.
Derived Correlation Identifiers: Alternative	Alternative example of a more complex eInsight correlation with a structured, message-derived correlation key based on Java Collaboration correlation key derivation preprocessor.
Message Relationship Patterns	Series of examples of using eInsight correlation facilities to implement Message Relationship patterns: Header-Items-Trailer Correlation, Any Order Two Items Correlation, Any Order Two Items Correlation with Timeout, Items-Trailer Correlation, Header Counted Items Correlation, Counted and Timed Items Correlation, Timed Items Correlation, Scatter-Gather Correlation.
Items-Trailer Correlation	Reimplementation of the Items-Trailer Correlation using a Java Collaboration, JMS, and dynamic JMS selectors—no eInsight.
Using New Web Service Collaborations	Example of using New Web Service Java Collaborations to implement reusable modules for use as activities in eInsight Business Processes.
Using eInsight Subprocesses for Reusability	Series of examples of using eInsight subprocesses as reusable components for use as activities in eInsight Business Processes: Request/Response, OneWay Operation, and Notification Subprocess implementations.
Using eInsight Web Services for Reusability	Series of examples of using Web Services as reusable components for use as activities in eInsight Business Processes: Request/Response, OneWay Operation, and Notification Web Service implementations.
JMS-Triggered Java Collaborations	Example of exception and JMS redelivery handling in a JMS message–triggered Java Collaboration.

TABLE P-1: Illustrations and Examples in Part II *(continued)*

Chapter/Section	*Example Topic*
Other Java Collaborations	Example of exception processing in a non-JMS message–triggered Java Collaboration.
JMS-Triggered Business Processes	Example of exception handling in a JMS message–triggered eInsight Business Process demonstrating behavior differences between XA and non-XA processes.
Fault Handlers	Example of using Fault Handlers in eInsight Business Processes.
Secure Sockets Layer (SSL, TLS)	Series of examples illustrating the use of SSL in Java CAPS solutions, both HTTP eWay- and Web Services-based. Covers server-side and mutual authentication for both server and client endpoints.
Web Service, Stored Procedures, and XA	Complete case study implementing an Employee Database Maintenance process with a multi-database update, Web Services, Oracle Stored Procedures, and an XA Business Process. This example implements and exercises a large number of patterns and suite features.
Example Travel Reservation	Complete Travel Reservation case study using Web Services orchestration and eInsight exception handling and compensation. This example implements and exercises a large number of patterns and suite features.
Handling Repeating Nodes in BPEL	Example of handling repeating nodes from XSD-based OTD in eInsight.
XML Deep Parse vs. Shallow Parse	Example implementing lazy XMLparse.
Using Multi-Operation WSDL	Example of implementing a multi-operation Web Service using WSDL and eInsight.
Cryptographic Objects	Step-by-step discussion of cryptographic objects required to configure PKI-related aspects of Java CAPS, with tools, commands, and scripts necessary to create certificate signing requests, convert between various certificate formats, and create various keystore types.

Acknowledgments

First and foremost, I would like to acknowledge my family, Lorraine, Natalie, and Daniel, who graciously tolerated the long hours I put into this project, encouraged me through the process, and rejoiced with me as I reached each significant milestone. Without their support, this project would not be possible.

Sun's announcement of the Red October program in November 2005, under which all Sun Software was going to be made available for download and use, was met with mixed reaction among some of the former SeeBeyond folk. Java CAPS is a complex enterprise application integration platform, not a lone-developer IDE or a data center technologist–configured infrastructure package. Some felt that making the Java CAPS suite freely downloadable would be unhelpful. This was particularly so as a major gap was perceived to exist between material in product documentation and the product and context knowledge an enterprise architect and integration developer would need to effectively use Java CAPS. I felt that a book bridging this gap, and showing examples of Java CAPS solutions implementing enterprise integration patterns, would be appropriate and timely.

Although the manuscript took three times as long to write as I originally anticipated, I made some of the material available to individuals both inside and outside Sun well before it was completed. This was to validate the original intent, to help answer specific questions, and to get early feedback on the content and coverage of various topics. I wish to acknowledge these individuals, particularly Jason Baragry, who took time to review most of the chapters and provided valuable feedback that helped shape the manuscript. I also wish to acknowledge other members of the seebeyond-I community at the ITtoolbox, who took time to provide feedback and who encouraged me to continue.

While this project is largely my own and was almost exclusively undertaken in my own time, there were occasions when I worked on it during my regular hours. Ray Gear at SeeBeyond and subsequently at Sun, and Angelo Joseph at Sun, are due credit for encouraging me to finish the project and tolerating the occasional diversion.

I wish to thank my collaborators, Sebastian Krueger, Brendan Marry, Saurabh Sahai, Peter Vaneris, and Andrew Walker, all talented Java CAPS field practitioners, for providing material for various chapters and for reviewing the manuscript. In particular, the topics Brendan and Peter addressed would likely not have been covered if they had not provided the material. I would also like to thank Dean Hansen, who provided notes on some of the material in early chapters and reviewed parts of the material.

Special thanks go to Jason Fordham and Peter C. Berkman, my colleagues at Sun, for trying to make a virtual image of the Solaris 10-based Java CAPS installation available in time for inclusion on the CD-ROM accompanying the book. Although through no fault of theirs this did not happen I am grateful to them for the effort.

Last, but not least, I would like to thank Carol J. Lallier, who made sure the text is in English, however American it may be; Kim Arney, who laid out the book; Elizabeth C. Ryan, who managed the publication process; and Greg Doench, who took a punt and walked me through the process of writing and publishing a book.

Michael Czapski
Sydney, December 2007.

About the Authors

Michael Czapski has 25 years of experience in the IT industry, the last 10 in the field of EAI. He provides Java CAPS expertise and leverages Java CAPS capabilities in solutions spanning the spectrum of Sun Microsystems software offerings. Michael has written a number of technical whitepapers on various topics for ICAN and Java CAPS, addressing, among others, Java CAPS security configuration, WS-Security implementation in Java CAPS, and application of EAI patterns to Java CAPS solutions. He is a Java CAPS Apostle, an active contributor to Java CAPS communities and forums, and a presenter at various industry conferences.

Saurabh Sahai has over 13 years of experience in IT, developing enterprise-class middleware software and commercial solutions for major software vendors. Over the past 4 years, he has worked as an integration architect within the Sun SOA/EAI professional service practice, where he is responsible for the architecture and delivery of advanced Sun Java CAPS–based solutions to major commercial and government clients within Australia and New Zealand.

Prior to Sun Microsystems, he worked for around 9 years as a J2EE/middleware architect for Fujitsu Australia Software Technologies, developing Java/J2EE/C++–based middleware software for Fujitsu's INTERSTAGE enterprise product set. He has extensive experience developing commercial J2EE applications using major application servers and open-source frameworks.

Saurabh is based in Sydney, Australia, and loves listening to Jazz in his spare time.

Andrew Walker has 18 years of experience in IT and originally joined SeeBeyond in January 1999, where he started working with one of the early EAI software products, then known as DataGate. Subsequently, he has worked with all the EAI software products released by SeeBeyond and now Sun Microsystems. Andrew has broad experience in architecting and implementing EAI and SOA solutions for customers in the Asia–Pacific region. He is currently based in Singapore and provides Java CAPS consulting services throughout the Asia–Pacific region as part of his job role in Sun Microsystems Professional Services.

Brendan Marry has over 10 years of experience in IT and is currently an integration solutions architect for Sun Microsystems in Auckland, New Zealand, responsible for the design and delivery of enterprise integration architectures using Java CAPS.

He has over 4 years of experience at Sun, specifically around the Sun Java CAPS. Brendan worked in the Java Mobile space and Java Enterprise space in Europe before immigrating to New Zealand and joining Sun. He enjoys providing project management and solution architectural advice, vision, and guidance to his clients using the Java CAPS products.

Sebastian Krueger started working on EAI software with SeeBeyond ICAN 5.0.5 in late 2005 and has since worked on all Sun Java CAPS eGate, eInsight, and eXchange product components, as well as on JMS Grid.

Initially providing Java CAPS consulting services to the New Zealand market, he now works for the Inland Revenue Department of New Zealand, where he is a senior analyst programmer.

Sebastian is a Sun-Certified Java Programmer and an LPI-Certified Linux Professional.

Peter Vaneris has 19 years of experience in the IT industry, the last 2 in the field of Java CAPS support. Prior to working with Java CAPS, Peter specialized in system administration, monitoring, automation, and enterprise management.