enterprise java microservices

reviewed by andrew binstock

february 27, 2020

it is unclear whether the current interest in microservices architecture is a
fad spurred on by the availability of easily accessible cloud services or
whether it will endure and become a standard model of application
design. whatever its ultimate fate, it's clear for the time being that many
java developers will be called on to create and implement microservices.
if you are one of them and have no real idea of where to start, this slim
volume should give you what you need to get up and running, that is, if
you are an experienced developer. the book states up front that it is
intended for developers with at least four years of experience in writing
java.

finnigan starts with basic definitions of microservices, a discussion of the
various technologies, and the concepts and designs that are foundational
to microservices. topics in this first half of the book include building
rest endpoints and choosing a small-scale application server, or what
the author refers to as jeas (just enough application server). the term
is his creation and after examining several possible servers for this role,
he ultimately chooses red hat's thorntail. while i can't argue with the
technology choice per se, thorntail is the project for which the author is
responsible at red hat. while the book stands up fine with this server,
there remains a thread of doubt throughout—would this be done
differently with another app server? am i learning microservices or
thorntail-specific microservices? the correct answer is some of both.
most of the explanations are indeed portable to other platforms.

for example, a lot of the early discussion, such as the differences in
design and architecture between monoliths and microservices, are well
explained and are valid independent of the implementing technology.
one of the topics finnigan pays close attention to both in the design and
implementation phases is testing, which is often overlooked in
discussions of microservices—replaced too often with hand waving and
comments bemoaning the difficulty.

in the second half of this book, finnigan presents the implementation in
considerable detail. the explanations are clear and easy to follow, in
good part because he has skillfully broken the topic into easily
approachable tasks.

in these explanations, finnigan favors widely used libraries and
packages, so this is not strictly a red hat–only tutorial. for example,
finnigan discusses using netflix hystrix, a library described in the
september 2018 issue of this magazine. and he includes a section on
data streaming using kafka (inexplicably missing from the book's index).
later, he explores tools that help deploy and manage microservices,
including security tools. he even explores, albeit briefly, handling
payments with stripe.
When it comes to deploying the sample app, Finnigan shows multiple possibilities beyond just the microservices option. For example, he sagely explains deploying as a monolith and as a smaller monolith with some parts implemented as services. And he covers deployments on hybrid clouds. This thoroughness runs throughout the book and makes it likely that by the time you’ve read through the more than 200 pages, you’ll have a good command of microservices, their architectural requirements, implementation details, and deployment challenges. That’s a lot of value.

My only concern is that Red Hat is a little too prominent in the discussions. Deployment is foremost on the OpenShift cloud, which, with all due respect, seems an unlikely target for most readers. However, the author is good enough to frame the discussions so that you can easily map what he’s saying to non-Red Hat tools and platforms.

This book is ideal for Java developers who are being asked to move from monoliths to microservices. Their natural temptation might be to study some corner of the migration most applicable to them and miss the fact that microservices require an entirely new perspective on the design and implementation of enterprise software. This quick read will make those programmers conversant with the issues and familiar with some of the leading solutions: a solid recommendation for a well-written book.