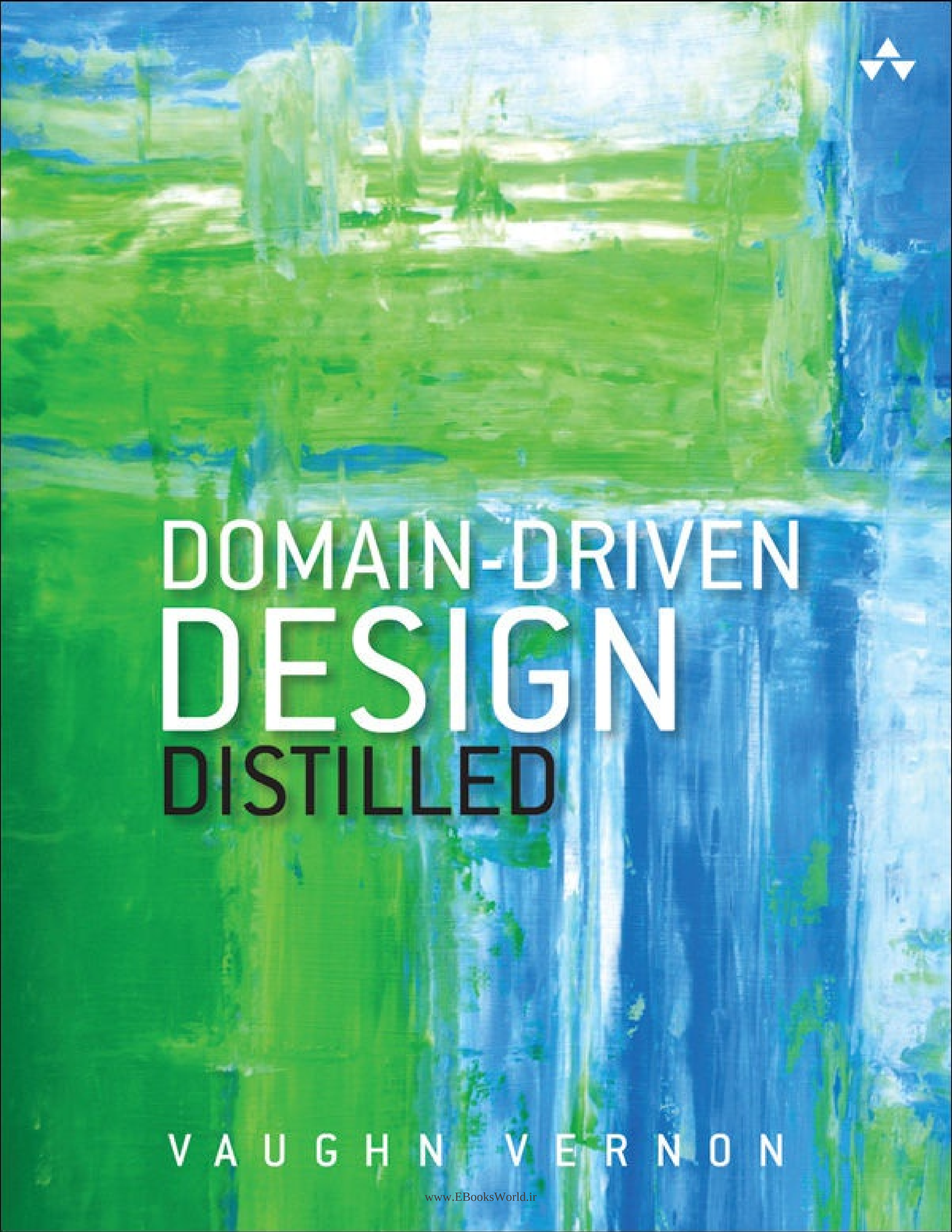

DOMAIN-DRIVEN DESIGN DISTILLED

VAUGHN VERNON

About This E-Book

EPUB is an open, industry-standard format for e-books. However, support for EPUB and its many features varies across reading devices and applications. Use your device or app settings to customize the presentation to your liking. Settings that you can customize often include font, font size, single or double column, landscape or portrait mode, and figures that you can click or tap to enlarge. For additional information about the settings and features on your reading device or app, visit the device manufacturer's Web site.

Many titles include programming code or configuration examples. To optimize the presentation of these elements, view the e-book in single-column, landscape mode and adjust the font size to the smallest setting. In addition to presenting code and configurations in the reflowable text format, we have included images of the code that mimic the presentation found in the print book; therefore, where the reflowable format may compromise the presentation of the code listing, you will see a "Click here to view code image" link. Click the link to view the print-fidelity code image. To return to the previous page viewed, click the Back button on your device or app.

Domain-Driven Design Distilled

Vaughn Vernon

Addison-Wesley

Boston • Columbus • Indianapolis • New York • San Francisco • Amsterdam • Cape Town
Dubai • London • Madrid • Milan • Munich • Paris • Montreal • Toronto • Delhi • Mexico City
São Paulo • Sydney • Hong Kong • Seoul • Singapore • Taipei • Tokyo

The author 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.

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com .

For questions about sales outside the U.S., please contact intlcs@pearson.com .

Visit us on the Web: informit.com/aw

Library of Congress Control Number: 2016936587

ISBN-13: 978-0-13-443442-1
ISBN-10: 0-13-443442-0
Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.
First printing, May 2016

Nicole and Tristan
We did it again!

Contents

Preface

Acknowledgments

About the Author

Chapter 1 DDD for Me

- Will DDD Hurt?
- Good, Bad, and Effective Design
- Strategic Design
- Tactical Design
- The Learning Process and Refining Knowledge
- Let's Get Started!

Chapter 2 Strategic Design with Bounded Contexts and the Ubiquitous Language

- Domain Experts and Business Drivers
- Case Study
- Fundamental Strategic Design Needed
- Challenge and Unify
- Developing a Ubiquitous Language
  - Putting Scenarios to Work
  - What about the Long Haul?
- Architecture
- Summary

Chapter 3 Strategic Design with Subdomains

- What Is a Subdomain?
- Types of Subdomains
- Dealing with Complexity
- Summary

Chapter 4 Strategic Design with Context Mapping

- Kinds of Mappings
  - Partnership
  - Shared Kernel
  - Customer-Supplier
  - Conformist
  - Anticorruption Layer

    - Open Host Service
    - Published Language
    - Separate Ways
    - Big Ball of Mud
  - Making Good Use of Context Mapping
    - RPC with SOAP
    - RESTful HTTP
    - Messaging
  - An Example in Context Mapping
  - Summary

- **Chapter 5 Tactical Design with Aggregates**
  - Why Used
  - Aggregate Rules of Thumb
    - Rule 1: Protect Business Invariants inside Aggregate Boundaries
    - Rule 2: Design Small Aggregates
    - Rule 3: Reference Other Aggregates by Identity Only
    - Rule 4: Update Other Aggregates Using Eventual Consistency
  - Modeling Aggregates
    - Choose Your Abstractions Carefully
    - Right-Sizing Aggregates
    - Testable Units
  - Summary

- **Chapter 6 Tactical Design with Domain Events**
  - Designing, Implementing, and Using Domain Events
  - Event Sourcing
  - Summary

- **Chapter 7 Acceleration and Management Tools**
  - Event Storming
    - Other Tools
  - Managing DDD on an Agile Project
    - First Things First
    - Use SWOT Analysis
    - Modeling Spikes and Modeling Debt
    - Identifying Tasks and Estimating Effort
  - Timeboxed Modeling
    - How to Implement

Interacting with Domain Experts

Summary

References

Index

Preface

Why is model building such a fun and rewarding activity? Ever since I was a kid I have loved to build models. At that time I mostly built models of cars and airplanes. I am not sure where LEGO was in those days. Still, LEGO has been a big part of my son's life since he was very young. It is so fascinating to conceive and build models with those small bricks. It's easy to come up with basic models, and it seems you can extend your ideas almost endlessly.

You can probably relate to some kind of youthful model building.

Models occur in so many situations in life. If you enjoy playing board games, you are using models. It might be a model of real estate and property owners, or models of islands and survivors, or models of territories and building activities, and who knows what all. Similarly, video games are models. Perhaps they model a fantasy world with fanciful characters playing fantastic roles. A deck of cards and related games model power. We use models all the time and probably so often that we don't give most models a well-deserved acknowledgment. Models are just part of our lives.

But why? Every person has a learning style. There are a number of learning styles, but three of the most discussed are auditory, visual, and tactile styles. The auditory learners learn by hearing and listening. The visual learners learn by reading or seeing imagery. The tactile learners learn by doing something that involves touching. It's interesting that each learning style is heavily favored by the individual to the extent that he or she can sometimes have trouble with other types of learning. For example, tactile learners likely remember what they have done but may have problems remembering what was said during the process. With model building, you would think that visual and tactile learners would have a huge advantage over the auditory learners, because model building seems to mostly involve visual and tactile stimulation. However, that might not always hold true, especially if a team of model builders uses audible communication in their building process. In other words, model building holds out the possibility to accommodate the learning style of the vast majority of individuals.

With our natural affinity to learning through model building, why would we not naturally desire to model the software that ever increasingly assists and influences our lives? In fact, to model software appears to be, well, human. And model software we should. It seems to me that humans should be elite software model builders.

It is my strong desire to help you be as human as you can possibly be by modeling software using some of the best software modeling tools available. These tools are packaged under the name "Domain-Driven Design," or DDD. This toolbox, actually a set of patterns, was first codified by Eric Evans in the book *Domain-Driven Design: Tackling Complexity in the Heart of Software* [DDD] . It is my vision to bring DDD to everyone possible. To make my point, if I must say that I want to bring DDD to the masses, then so be it. That is where DDD deserves to be, and DDD is the toolbox that model-oriented humans deserve to use to create their most advanced software models. With this book, I am determined to make learning and using DDD as simple and easy as possible and to bring that to the broadest conceivable audience.

For auditory learners, DDD holds out the prospect of learning through the team communication of building a model based on the development of a *Ubiquitous Language.* For visual and tactile learners, the process of using DDD tools is very visual and hands-on as your team models both strategically and tactically. This is especially true when drawing *Context Maps* and modeling the business process using *Event Storming*. Thus, I believe that DDD can support everyone who wants to learn and achieve greatness through model building.

Who Is This Book For?

This book is for everyone interested in learning the most important DDD aspects and tools and in learning quickly. The most common readers are software architects and software developers who will put DDD into practice on projects. Very often, software developers quickly discover the beauty of DDD and are keenly attracted to its powerful tooling. Even so, I have made the subject understandable for executives, domain experts, managers, business analysts, information architects, and testers alike. There's really no limit to those in the information technology (IT) industry and research and development (R&D) environments who can benefit from reading this book.

If you are a consultant and you are working with a client to whom you have recommended the use of DDD, provide this book as a way to bring the major stakeholders up to speed quickly. If you have developers—perhaps junior or midlevel or even senior—working on your project who are unfamiliar with DDD but need to use it very soon, make sure that they read this book. By reading this book, at minimum, all the project stakeholders and developers will have the vocabulary and understand the primary DDD tools being used. This will enable them to share things meaningfully as they move the project forward.

Whatever your experience level and role, read this book and then practice DDD on a project. Afterward, reread this book and see what you can learn from your experiences and where you can improve in the future.

What This Book Covers

The first chapter, "DDD for Me," explains what DDD can do for you and your organization and provides a more detailed overview of what you will learn and why it's important.

Chapter 2 , "Strategic Design with Bounded Contexts and the Ubiquitous Language ," introduces DDD strategic design and teaches the cornerstones of DDD, *Bounded Contexts* and the *Ubiquitous Language*. Chapter 3 , "Strategic Design with Subdomains ," explains *Subdomains* and how you can use them to deal with the complexity of integrating with existing legacy systems as you model your new applications. Chapter 4 , "Strategic Design with Context Mapping ," teaches the variety of ways that teams work together strategically and ways that their software can integrate. This is called *Context Mapping*.

Chapter 5 , "Tactical Design with Aggregates ," switches your attention to tactical modeling with *Aggregates*. An important and powerful tactical modeling tool to be used with *Aggregates* is *Domain Events* , which is the subject of Chapter 6 , "Tactical Design with Domain Events ."

Finally, in Chapter 7 , "Acceleration and Management Tools ," the book highlights some project acceleration and project management tools that can help teams establish and maintain their cadence. These two topics are seldom if ever discussed in other DDD sources and are sorely needed by those who are determined to put DDD into practice.

Conventions

There are only a few conventions to keep in mind while reading. All of the DDD tools that I discuss are printed in italics. For example, you will read about *Bounded Contexts* and *Domain Events*.