

《实现领域驱动设计英文版》

图书基本信息

书名：《实现领域驱动设计英文版》

13位ISBN编号：9787121272741

出版时间：2016-4

作者：Vaughn Vernon（沃恩·弗农）

页数：648

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

《实现领域驱动设计英文版》

内容概要

领域驱动设计（DDD）是教我们如何做好软件的，同时也是教我们如何更好地使用面向对象技术的。它为我们提供了设计软件的全新视角，同时也给开发者留下了一大难题：如何将领域驱动设计付诸实践？VaughnVernon的这本《实现领域驱动设计》为我们给出了全面的解答。《实现领域驱动设计英文版》分别从战略和战术层面详尽地讨论了如何实现DDD，其中包含了大量的优秀实践、设计准则和对一些问题的折中性讨论。《实现领域驱动设计英文版》共分为14章。DDD战略部分讲解了领域、限界上下文、上下文映射图和架构等内容；战术部分包括实体、值对象、领域服务、领域事件、聚合和资源库等内容。一个虚构的案例研究贯穿全书，这对于实例讲解DDD实现来说非常有用。《实现领域驱动设计英文版》在DDD的思想和实现之间建立起了一座桥梁，架构师和程序员均可阅读，同时也可以作为一本DDD参考书。

《实现领域驱动设计英文版》

作者简介

Vaughn Vernon 是一个经验丰富的软件工程师，在软件设计、开发和架构方面拥有超过25年的从业经验。他提倡通过创新来简化软件的设计和实现。从20世纪80年代起，他便开始使用面向对象语言进行编程；在20世纪90年代早期，他便在领域建模中应用了领域驱动设计，那时他使用的是Smalltalk语言。他在很多业务领域都有从业经验，包括航空、环境、地理、保险、医学和电信等领域。同时，Vaughn在技术上也取得了很大的成功，包括开发可重用的框架和类库等。他在全球范围之内提供软件咨询和演讲，此外，他还在许多国家教授《实现领域驱动设计》的课程。你可以通过www.VaughnVernon.co访问到他的最新研究成果。他的Twitter：@VaughnVernon。

书籍目录

序

前言

致谢

关于作者

如何使用本书、

Chapter 1 Getting Started with DDD

Can I DDD?

Why You Should Do DDD

How to Do DDD

The Business Value of Using DDD

1. The Organization Gains a Useful Model of Its Domain

2. A Refined, Precise Definition and Understanding of the Business Is Developed

3. Domain Experts Contribute to Software Design

4. A Better User Experience Is Gained

5. Clean Boundaries Are Placed around Pure Models

6. Enterprise Architecture Is Better Organized

7. Agile, Iterative, Continuous Modeling Is Used

8. New Tools, Both Strategic and Tactical, Are Employed

The Challenges of Applying DDD

Fiction, with Bucketfuls of Reality

Wrap-Up

Chapter 2 Domains, Subdomains, and Bounded Contexts

Big Picture

Subdomains and Bounded Contexts at Work

Focus on the Core Domain

Why Strategic Design Is So Incredibly Essential

Real-World Domains and Subdomains

Making Sense of Bounded Contexts

Room for More than the Model

Size of Bounded Contexts

Aligning with Technical Components

Sample Contexts

Collaboration Context.

Identity and Access Context.

Agile Project Management Context

Wrap-Up

Chapter 3 Context Maps

Why Context Maps Are So Essential

Drawing Context Maps

Projects and Organizational Relationships

Mapping the Three Contexts

Wrap-Up

Chapter 4 Architecture

Interviewing the Successful CIO

Layers

Dependency Inversion Principle

Hexagonal or Ports and Adapters

- Service-Oriented
- Representational State Transfer—REST
- REST as an Architectural Style
- Key Aspects of a RESTful HTTP Server
- Key Aspects of a RESTful HTTP Client
- REST and DDD
- Why REST?
- Command-Query Responsibility Segregation, or CQRS
- Examining Areas of CQRS
- Dealing with an Eventually Consistent Query Model
- Event-Driven Architecture
- Pipes and Filters
- Long-Running Processes, aka Sagas
- Event Sourcing
- Data Fabric and Grid-Based Distributed Computing
- Data Replication
- Event-Driven Fabrics and Domain Events
- Continuous Queries
- Distributed Processing
- Wrap-Up
- Chapter 5 Entities
- Why We Use Entities
- Unique Identity.
- User Provides Identity
- Application Generates Identity
- Persistence Mechanism Generates Identity
- Another Bounded Context Assigns Identity
- When the Timing of Identity Generation Matters
- Surrogate Identity
- Identity Stability.
- Discovering Entities and Their Intrinsic Characteristics
- Uncovering Entities and Properties
- Digging for Essential Behavior
- Roles and Responsibilities
- Construction
- Validation
- Change Tracking
- Wrap-Up
- Chapter 6 Value Objects
- Value Characteristics
- Measures, Quantifies, or Describes
- Immutable
- Conceptual Whole
- Replaceability
- Value Equality.
- Side-Effect-Free Behavior
- Integrate with Minimalism.
- Standard Types Expressed as Values
- Testing Value Objects

Implementation.

Persisting Value Objects

Reject Undue Influence of Data Model Leakage.

ORM and Single Value Objects

ORM and Many Values Serialized into a Single Column

ORM and Many Values Backed by a Database Entity.

ORM and Many Values Backed by a Join Table.

ORM and Enum-as-State Objects

Wrap-Up

Chapter 7 Services

What a Domain Service Is (but First, What It Is Not)

Make Sure You Need a Service.

Modeling a Service in the Domain

Is Separated Interface a Necessity?

A Calculation Process

Transformation Services.

Using a Mini-Layer of Domain Services

Testing Services.

Wrap-Up

Chapter 8 Domain Events

The When and Why of Domain Events

Modeling Events

With Aggregate Characteristics

Identity

Publishing Events from the Domain Model

Publisher

Subscribers

Spreading the News to Remote Bounded Contexts

Messaging Infrastructure Consistency

Autonomous Services and Systems

Latency Tolerances

Event Store

Architectural Styles for Forwarding Stored Events

Publishing Notifications as RESTful Resources

Publishing Notifications through Messaging Middleware

Implementation

Publishing the NotificationLog

Publishing Message-Based Notifications

Wrap-Up

Chapter 9 Modules

Designing with Modules

Basic Module Naming Conventions

Module Naming Conventions for the Model.

Modules of the Agile Project Management Context

Modules in Other Layers

Module before Bounded Context

Wrap-Up

Chapter 10 Aggregates

Using Aggregates in the Scrum Core Domain

First Attempt: Large-Cluster Aggregate
Second Attempt: Multiple Aggregates
Rule: Model True Invariants in Consistency Boundaries
Rule: Design Small Aggregates
Don ' t Trust Every Use Case
Rule: Reference Other Aggregates by Identity
Making Aggregates Work Together through Identity
References
Model Navigation
Scalability and Distribution
Rule: Use Eventual Consistency Outside the Boundary
Ask Whose Job It Is
Reasons to Break the Rules
Reason One: User Interface Convenience
Reason Two: Lack of Technical Mechanisms
Reason Three: Global Transactions
Reason Four: Query Performance
Adhering to the Rules
Gaining Insight through Discovery.
Rethinking the Design, Again
Estimating Aggregate Cost
Common Usage Scenarios
Memory Consumption
Exploring Another Alternative Design
Implementing Eventual Consistency.
Is It the Team Member ' s Job?
Time for Decisions
Implementation
Create a Root Entity with Unique Identity
Favor Value Object Parts
Using Law of Demeter and Tell, Don ' t Ask
Optimistic Concurrency.
Avoid Dependency Injection.
Wrap-Up
Chapter 11 Factories
Factories in the Domain Model
Factory Method on Aggregate Root
Creating CalendarEntry Instances
Creating Discussion Instances
Factory on Service
Wrap-Up
Chapter 12 Repositories
Collection-Oriented Repositories
Hibernate Implementation
Considerations for a TopLink Implementation
Persistence-Oriented Repositories
Coherence Implementation
MongoDB Implementation
Additional Behavior

Managing Transactions
A Warning
Type Hierarchies
Repository versus Data Access Object
Testing Repositories
Testing with In-Memory Implementations
Wrap-Up
Chapter 13 Integrating Bounded Contexts
Integration Basics
Distributed Systems Are Fundamentally Different
Exchanging Information across System Boundaries
Integration Using RESTful Resources
Implementing the RESTful Resource
Implementing the REST Client Using an Anticorruption Layer
Integration Using Messaging
Staying Informed about Product Owners and Team Members
Can You Handle the Responsibility?
Long-Running Processes, and Avoiding Responsibility
Process State Machines and Time-out Trackers
Designing a More Sophisticated Process
When Messaging or Your System Is Unavailable
Wrap-Up
Chapter 14 Application.
User Interface
Rendering Domain Objects
Render Data Transfer Object from Aggregate Instances
Use a Mediator to Publish Aggregate Internal State
Render Aggregate Instances from a Domain Payload Object
State Representations of Aggregate Instances
Use Case Optimal Repository Queries.
Dealing with Multiple, Disparate Clients
Rendition Adapters and Handling User Edits
Application Services
Sample Application Service
Decoupled Service Output
Composing Multiple Bounded Contexts
Infrastructure
Enterprise Component Containers
Wrap-Up
Appendix A Aggregates and Event Sourcing: A+ES
Inside an Application Service
Command Handlers
Lambda Syntax
Concurrency Control.
Structural Freedom with A+ES
Performance
Implementing an Event Store
Relational Persistence
BLOB Persistence

《实现领域驱动设计英文版》

Focused Aggregates
Read Model Projections
Use with Aggregate Design
Events Enrichment
Supporting Tools and Patterns
Event Serializers
Event Immutability
Value Objects
Contract Generation
Unit Testing and Specifications
Event Sourcing in Functional Languages
Bibliography
Index

《实现领域驱动设计英文版》

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:www.tushu000.com