# Learning Domain-Driven Design

*Aligning Software Architecture and Business Strategy*

*Vlad Khononov*

Beijing • Boston • Farnham • Sebastopol • Tokyo

O'REILLY®

**Влад Хононов**

# Изучаем DDD — предметно-ориентированное проектирование

Санкт-Петербург
«БХВ-Петербург»
2024

УДК 004.4'236
ББК 32.973.26-018
Х77

**Хононов В.**

Х77 Изучаем DDD — предметно-ориентированное проектирование: Пер. с англ. — СПб.: БХВ-Петербург, 2024. — 320 с.: ил.

ISBN 978-5-9775-1886-4

Книга посвящена методологии DDD (предметно-ориентированному проектированию), что особенно актуально в условиях дробления предметных областей и усложнения бизнес-взаимодействий. Рассказано, как оценить масштаб и сложность предметной области, измерить темпы ее развития, учесть необходимые зависимости, применять событийно-ориентированную архитектуру и структурировать создаваемое ПО, эффективно вписывая его в сеть данных (Data Mesh). Материал будет особенно интересен при развитии стартапа и разработке наукоемких отраслевых систем.

*Для архитекторов ПО, бизнес-аналитиков*
*и разработчиков корпоративного программного обеспечения*

УДК 004.4'236
ББК 32.973.26-018

**Группа подготовки издания:**

| | |
|---|---|
| Руководитель проекта | *Олег Сивченко* |
| Зав. редакцией | *Людмила Гауль* |
| Перевод с английского | *Николая Вильчинского* |
| Редактор | *Анна Ардашева* |
| Компьютерная верстка | *Ольги Сергиенко* |
| Оформление обложки | *Зои Канторович* |

© 2023 BHV
Authorized Russian translation of the English edition of ***Learning Domain-Driven Design*** ISBN 9781098100131
© 2022 Vladislav Khononov.
This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

Авторизованный перевод с английского языка на русский издания ***Learning Domain-Driven Design***
ISBN 9781098100131 © 2022 Vladislav Khononov.
Перевод опубликован и продается с разрешения компании-правообладателя O'Reilly Media, Inc.

"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20

ISBN 978-1-098-10013-1 (англ.)
ISBN 978-5-9775-1886-4 (рус.)

© Vladislav Khononov, 2022
© Перевод на русский язык, оформление.
ООО "БХВ-Петербург", ООО "БХВ", 2023

# Оглавление

**Предисловие редакторской группы** ..... **15**

**Предисловие** ..... **19**

**Введение** ..... **21**
Зачем я написал эту книгу ..... 22
Кому следует прочитать эту книгу ..... 22
Навигация по книге ..... 23
Пример предметной области: WolfDesk ..... 24
Соглашения, используемые в этой книге ..... 25
Порядок использования примеров кода ..... 26
Благодарности ..... 27

**Вступление** ..... **29**

**ЧАСТЬ I. СТРАТЕГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ** ..... **31**

**Глава 1. Анализ предметной области** ..... **33**
Так что же такое предметная область? ..... 33
Что такое поддомен (subdomain)? ..... 34
Типы поддоменов ..... 34
Основные поддомены (core subdomains) ..... 34
Универсальные поддомены (generic subdomains) ..... 36
Вспомогательные поддомены (supporting subdomains) ..... 36
Сравнение поддоменов ..... 37
Конкурентное преимущество ..... 37
Сложность ..... 38
Изменчивость ..... 39
Стратегия решения (solution) ..... 40
Определение границ поддоменов ..... 41
Выделение поддоменов (Distilling subdomains) ..... 42
Поддомены с позиции согласующихся сценариев использования ..... 42
Сосредоточьтесь на главном ..... 44
Примеры анализа предметной области ..... 44
Gigmaster ..... 44
Предметная область (домен) и поддомены ..... 45
Архитектурные решения ..... 45
BusVNext ..... 46
Предметная область (домен) и поддомены ..... 46
Архитектурные решения ..... 47

Кто такие специалисты в предметной области? ..... 47
Выводы ..... 48
Упражнения ..... 49

**Глава 2. Экспертные знания о предметной области ..... 50**
Задачи бизнеса (business problems) ..... 50
Выявление экспертных знаний ..... 51
Общение ..... 51
Что такое единый язык? ..... 53
Язык бизнеса ..... 54
Сценарии ..... 54
Согласованность ..... 55
Неоднозначные понятия ..... 55
Понятия-синонимы ..... 55
Модель предметной области ..... 56
Что такое модель? ..... 56
Эффективное моделирование ..... 56
Моделирование предметной области ..... 57
Непрерывная работа ..... 57
Инструменты ..... 58
Сложности ..... 59
Вывод ..... 60
Упражнения ..... 61

**Глава 3. Как осмыслить сложность предметной области ..... 62**
Противоречивые модели ..... 62
Что такое ограниченный контекст? ..... 64
Границы модели ..... 65
Уточнение термина «единый язык» ..... 65
Область применения ограниченного контекста ..... 66
Сравнение ограниченных контекстов и поддоменов ..... 67
Поддомены ..... 67
Ограниченные контексты ..... 68
Взаимодействие поддоменов и ограниченных контекстов ..... 68
Границы ..... 70
Физические границы ..... 70
Границы владения ..... 70
Ограниченные контексты в реальной жизни ..... 71
Семантические области ..... 71
Наука ..... 72
Покупка холодильника ..... 73
Вывод ..... 75
Упражнения ..... 75

**Глава 4. Интеграция ограниченных контекстов ..... 77**
Сотрудничество (Cooperation) ..... 77
Партнерство (Partnership) ..... 78
Общее ядро (shared kernel) ..... 78
Общие рамки (Shared scope) ..... 79
Реализация ..... 79
Когда следует воспользоваться общим ядром ..... 80

Потребитель-Поставщик (Customer-supplier) ... 81
Конформист (Conformist) ... 81
Предохранительный слой (Anticorruption layer) ... 82
Сервис с открытым протоколом (Open-Host Service) ... 83
Разные пути (Separate Ways) ... 84
Проблемы общения ... 84
Универсальный поддомен (Generic Subdomain) ... 84
Различия в моделях ... 85
Карта контекстов (Context Map) ... 85
Поддержка в актуальном состоянии ... 86
Ограничения ... 86
Вывод ... 87
Упражнения ... 87

**ЧАСТЬ II. ТАКТИЧЕСКИЙ ЗАМЫСЕЛ ... 89**

**Глава 5. Реализация простой бизнес-логики ... 90**
Транзакционный сценарий ... 90
Реализация ... 91
Это не так-то просто! ... 91
Отсутствие транзакционного поведения ... 91
Распределенные транзакции ... 93
Неявные распределенные транзакции ... 94
Когда следует применять транзакционный сценарий ... 96
Активная запись ... 97
Реализация ... 97
Когда следует применять активную запись ... 98
Придерживайтесь прагматичного подхода ... 99
Вывод ... 99
Упражнения ... 100

**Глава 6. Проработка сложной бизнес-логики ... 102**
Предыстория ... 102
Модель предметной области (доменная модель) ... 102
Реализация ... 103
Сложность ... 104
Единый язык ... 104
Строительные блоки ... 104
Объект-значение ... 104
Сущности ... 110
Агрегаты ... 111
Доменные сервисы (domain service) ... 119
Управление сложностью ... 121
Вывод ... 122
Упражнения ... 123

**Глава 7. Моделирование фактора времени ... 125**
События как источник данных (Event Sourcing) ... 125
Поиск ... 130
Анализ ... 132

Источник истины ... 133
Хранилище событий ... 133
Модель предметной области, основанная на событиях ... 134
Преимущества ... 137
Недостатки ... 138
Часто задаваемые вопросы ... 139
Производительность ... 139
Удаление данных ... 141
А почему просто нельзя ... 141
Вывод ... 142
Упражнения ... 142

**Глава 8. Архитектурные паттерны ... 144**
Сопоставление бизнес-логики и архитектурных паттернов ... 144
Слоеная архитектура (Layered Architecture) ... 145
Слой представления (Presentation layer) ... 145
Слой бизнес-логики (Business logic layer) ... 146
Слой доступа к данным (Data access layer) ... 146
Связь между слоями ... 147
Вариация ... 148
Сервисный слой (Service layer) ... 148
Терминология ... 151
Когда предпочтительнее использовать слоеную архитектуру ... 151
Дополнительно: сравнение слоев и уровней ... 152
Порты и адаптеры (Ports and adapters) ... 152
Терминология ... 152
Принцип инверсии зависимостей (Dependency inversion principle) ... 153
Интеграция инфраструктурных компонентов ... 154
Варианты ... 155
Когда предпочтительнее использовать порты и адаптеры ... 155
Разделение ответственности команд и запросов (Command-Query Responsibility Segregation) ... 155
Мультипарадигменное моделирование (Polyglot modelling) ... 156
Реализация ... 156
Модель выполнения команд ... 157
Модели чтения (проекции) ... 157
Проецирование моделей чтения ... 157
Синхронные проекции ... 158
Асинхронные проекции ... 159
Сложности ... 160
Разделение моделей ... 160
Когда предпочтительнее использовать CQRS ... 160
Область применения ... 161
Вывод ... 162
Упражнения ... 163

**Глава 9. Паттерны взаимодействия ... 164**
Преобразование моделей ... 164
Преобразование моделей без сохранения состояния ... 165
Синхронный режим ... 165
Асинхронный режим ... 167

Преобразование моделей с отслеживанием состояния ..... 168
Агрегирование входящих данных ..... 168
Объединение нескольких источников ..... 169
Интеграция агрегатов ..... 170
Паттерн исходящих сообщений (Outbox) ..... 172
Извлечение неопубликованных событий ..... 173
Сага ..... 174
Согласованность ..... 177
Диспетчер процессов ..... 177
Вывод ..... 180
Упражнения ..... 181

**ЧАСТЬ III. ПРИМЕНЕНИЕ ПРЕДМЕТНО-ОРИЕНТИРОВАННОГО ПРОЕКТИРОВАНИЯ НА ПРАКТИКЕ ..... 183**

**Глава 10. Эвристика проектирования ..... 184**
Эвристика ..... 184
Ограниченные контексты ..... 184
Паттерны реализации бизнес-логики ..... 186
Архитектурные паттерны ..... 188
Стратегия тестирования ..... 189
Пирамида тестирования ..... 190
Ромб тестирования ..... 190
Перевернутая пирамида тестирования ..... 190
Дерево тактических проектных решений ..... 191
Вывод ..... 192
Упражнения ..... 192

**Глава 11. Эволюция проектных решений ..... 194**
Изменения в предметных областях ..... 194
Из основного в универсальный ..... 195
Из универсального в основной ..... 195
Из вспомогательного в универсальный ..... 196
Из вспомогательного в основной ..... 196
Из основного во вспомогательный ..... 196
Из универсального во вспомогательный ..... 196
Стратегические аспекты проектирования ..... 197
Тактические аспекты проектирования ..... 198
Преобразование транзакционного сценария в активную запись ..... 198
Преобразование активной записи в модель предметной области ..... 199
Преобразование модели предметной области в модель предметной области, основанную на событиях ..... 200
Генерация прошлых переходов состояния ..... 201
Моделирование событий миграции ..... 202
Организационные изменения ..... 202
Переход от партнерства к отношениям потребитель-поставщик ..... 203
Переход от отношений потребитель-поставщик к модели разных путей ..... 204
Знания предметной области ..... 204
Рост проекта ..... 205
Поддомены ..... 205

Ограниченные контексты ....206
Агрегаты ....207
Вывод ....207
Упражнения ....208

**Глава 12. EventStorming ....210**
Что такое EventStorming? ....210
Кто принимает участие в EventStorming? ....210
Что нужно для проведения EventStorming? ....211
Процесс проведения EventStorming ....212
Этап 1: Проведение неструктурированного исследования ....212
Этап 2: Выстраивание в хронологическом порядке ....213
Этап 3: Проблемные места (pain points) ....213
Этап 4: Выявление ключевых событий (pivotal events) ....214
Этап 5: Выявление команд (commands) ....215
Этап 6: Выявление правил (policies) ....216
Этап 7: Выявление моделей чтения (read model) ....217
Этап 8: Выявление внешних систем (external systems) ....217
Этап 9: Выявление агрегатов ....218
Этап 10: Выявление ограниченных контекстов ....218
Варианты ....219
Когда следует проводить EventStorming ....220
Советы по проведению ....221
Отслеживание динамики проведения семинара ....221
Проведение EventStorming с удаленными участниками ....222
Вывод ....222
Упражнения ....222

**Глава 13. Предметно-ориентированное проектирование на практике ....224**
Стратегический анализ ....225
Осмысление предметной области ....225
Основные поддомены (core subdomains) ....225
Универсальные поддомены ....226
Вспомогательные поддомены (supporting subdomains) ....226
Изучение текущего проекта ....226
Оценка тактического замысла ....227
Оценка стратегического замысла ....227
Определение стратегии модернизации ....228
Стратегическая модернизация ....228
Тактическая модернизация ....230
Развитие единого языка ....230
Паттерн «Душитель» (Strangler) ....231
Рефакторинг тактических проектных решений ....233
Прагматичное предметно-ориентированное проектирование ....234
Как «продать» предметно-ориентированное проектирование? ....234
Законспирированное предметно-ориентированное проектирование ....235
Единый язык ....235
Ограниченные контексты ....236
Тактические проектные решения ....236
Модель предметной области, основанная на событиях ....237

Вывод....237
Упражнения....238

**ЧАСТЬ IV. ВЗАИМООТНОШЕНИЯ С ДРУГИМИ МЕТОДОЛОГИЯМИ И ПАТТЕРНАМИ....239**

**Глава 14. Микросервисы....240**
Что такое сервис?....240
Что такое микросервис?....241
Метод как Сервис (Method as a Service): путь к созданию идеальных микросервисов?....242
Цель проектирования....243
Сложность системы....244
Микросервисы как «глубокие» сервисы (deep services)....245
Микросервисы как глубокие модули....246
Предметно-ориентированное проектирование и границы микросервисов....248
Ограниченные контексты....248
Агрегаты....250
Поддомены....251
Сокращение публичных интерфейсов микросервисов....252
Сервис с открытым протоколом....252
Предохранительный слой (anticorruption layer, ACL)....253
Вывод....254
Упражнения....254

**Глава 15. Событийно-ориентированная архитектура....256**
Событийно-ориентированная архитектура....256
События....257
События, команды и сообщения....257
Структура....258
Типы событий....258
Уведомление....258
Передача состояния с помощью события....260
События предметной области (domain event)....262
Сравнение событий предметной области и уведомлений....262
Сравнение событий предметной области с ECST-сообщениями....262
Типы событий: Пример....263
Проектирование событийно-ориентированной интеграции....264
Распределенный большой ком грязи....264
Временна́я связанность (связанность по времени)....265
Функциональная связанность....266
Связанность на уровне реализации....266
Реорганизация событийно-ориентированной интеграции....266
Творческий подход к событийно-ориентированному проектированию....267
Предполагайте худшее....267
Используйте публичный интерфейс и приватные события....268
Оценивайте требования к согласованности....268
Вывод....269
Упражнения....269

**Глава 16. Сеть данных (Data Mesh) ..... 271**
Сравнение аналитической модели данных (OLAP) с моделью транзакционных данных (OLTP) ..... 271
Таблица фактов ..... 272
Таблица измерений ..... 274
Аналитические модели ..... 274
Платформы управления аналитическими данными ..... 276
Хранилище данных — Data Warehouse ..... 276
Озеро данных — Data Lake ..... 279
Проблемы архитектур хранилища данных и озера данных ..... 280
Сеть данных (Data mesh) ..... 281
Разбиение данных по предметным областям ..... 281
Данные как продукт ..... 283
Обеспечение автономии ..... 284
Построение экосистемы ..... 284
Совмещение сети данных (data mesh) и предметно-ориентированного проектирования ..... 285
Вывод ..... 287
Упражнения ..... 287

**Заключение ..... 289**
Задача ..... 289
Решение ..... 290
Реализация ..... 291
Рекомендуемая литература ..... 291
Дополнительные сведения о предметно-ориентированном проектировании ..... 291
Архитектурные и интеграционные шаблоны ..... 292
Модернизация устаревших систем ..... 292
EventStorming ..... 293
Вывод ..... 294

**Приложение 1. Применение DDD: пример из практики ..... 295**
Пять ограниченных контекстов ..... 295
Предметная область ..... 295
Ограниченный контекст № 1: Маркетинг ..... 297
Своеобразная магия ..... 297
Наши ранние взгляды на предметно-ориентированное проектирование ..... 297
Ограниченный контекст № 2: CRM ..... 297
Еще больше «агрегатов»! ..... 298
Разработка решения: Дубль два ..... 299
Вавилонская башня 2.0 ..... 299
Более широкий взгляд на предметно-ориентированное проектирование ..... 300
Ограниченный контекст № 3: Обработчики событий ..... 301
Ограниченный контекст № 4: Бонусы ..... 302
Проектирование: Дубль два ..... 303
Единый язык ..... 303
Классическое понимание предметно-ориентированного проектирования ..... 303
Ограниченный контекст № 5: Центр маркетинга ..... 304
Микро — что? ..... 304
Реальная проблема ..... 305

Обсуждение ..... 305
Единый язык ..... 305
Поддомены ..... 306
Сопоставление проектных решений с поддоменами ..... 307
Не игнорируйте боль ..... 308
Границы ограниченных контекстов ..... 308
Вывод ..... 309

**Приложение 2. Ответы на вопросы упражнений ..... 310**
Глава 1 ..... 310
Глава 2 ..... 311
Глава 3 ..... 311
Глава 4 ..... 312
Глава 5 ..... 312
Глава 6 ..... 313
Глава 7 ..... 313
Глава 8 ..... 313
Глава 9 ..... 314
Глава 10 ..... 314
Глава 11 ..... 314
Глава 12 ..... 315
Глава 13 ..... 315
Глава 14 ..... 315
Глава 15 ..... 315
Глава 16 ..... 316

**Предметный указатель ..... 317**

**Об авторе ..... 320**

# Предисловие редакторской группы

**Баранов Сергей Александрович**

Архитектор, консультант, основатель конференции ArchDays, соучредитель российской ассоциации ИТ-архитекторов.

**https://github.com/jsergey**

Мое знакомство с Domain-Driven Design произошло около пятнадцати лет назад. На тот момент для меня это было что-то новое, перепрошивающее разум и от того притягательное и вовлекающее. Прочитав книгу Эванса «Предметно-ориентированное проектирование. Структуризация сложных программных систем» (**https://habr.com/ru/articles/580972/**), я сразу попытался применить новую информацию на практике. И оказалось, что многое из прочитанного понял не так, потому что продолжал находиться в собственном контексте понимания, обусловленном прошлым опытом.

Со временем мое понимание Domain-Driven Design улучшилось, о чем говорил положительный опыт в разработке продуктов, спроектированных с использованием DDD. Это понимание достаточно сложно описать словами, это своего рода инсайт, озарение и я долгое время искал подходящие формулировки, определения, примеры. Зачем? Потому что, несмотря на то, что Domain-Driven Design действительно позволяет взять под контроль управление сложностью проектируемых систем, сама по себе эта методология все-таки достаточно сложна.

После появления концепции микросервисов Domain-Driven Design пошел в массы как один из наиболее эффективных методов разграничения микросервисов. Назрела необходимость нового метода, который снизил бы порог вхождения в саму практику Domain-Driven Design, и такая практика появилась — ее назвали Event Storming.

Широкие массы проявили интерес к описываемой парадигме, и следом неминуемо должна была появиться книга, которая бы простым и в то же время профессиональным языком познакомила читателя с Domain-Driven Design. Вы держите в руках ровно такую книгу, с простыми и понятными формулировками, определениями и примерами.

Выражаю благодарность за возможность провести научную редактуру столь значимого для индустрии труда и желаю Вам увлекательного погружения в мир Domain-Driven Design!

## Лукьянов Евгений Александрович

Архитектор ПО

**https://github.com/stringconcat/**

С современной IT-отраслью явно что-то не так. Мы научились уверенно решать технические задачи: у нас есть бесчисленные библиотеки, фреймворки и СУБД на все случаи жизни. Мы парсим HTTP-запросы и легко обрабатываем огромные массивы данных. Но когда дело доходит до управления сложностью предметной области, начинается настоящий кошмар для всех участников проекта.

Когда код не отражает предметную область, технологии не помогут. Заказчики получают что угодно, кроме того, что заказывали, проекты гибнут, а разработчики продолжают перебирать СУБД и фреймворки в поисках Священного Грааля, попутно пополняя резюме востребованными технологиями. И даже если такие проекты переписывают заново, они неизбежно превращаются в прежнее болото.

Нельзя сказать, что для управления сложностью предметной области у нас нет хороших инструментов. Есть, просто ими не пользуются. Например, концепция Domain-Driven Design появилась 20 лет назад, но на реальных проектах следование принципам DDD встречается все еще довольно редко.

Конечно, нельзя сказать, что DDD — это просто и очевидно. Если вы пробовали пройти по этому пути, то знаете, как много там граблей и как мало вменяемых описаний практического применения. Книга, которую вы держите в руках, призвана исправить эту ситуацию. Если вы только присматриваетесь к DDD, книга объяснит аспекты этой концепции и даст понимание, зачем все это нужно. А если вы опытный разработчик, книга поможет понять, как реализовать те или иные конструкции, и даст полезные эвристические приемы, которые позволят ориентироваться в условиях постоянной неопределенности.

## Промышленников Родион Александрович

Руководитель отдела разработки

**https://github.com/rpromyshlennikov**

С оригиналом этой книги я был удостоен чести ознакомиться еще до того, как она увидела свет в виде печатного издания на английском языке. Я был очарован стилем повествования автора, простотой подачи информации и отношением количества полезных знаний к объему самой книги. Мы, как участники IT-отрасли, читаем много технической литературы, но не каждый раз нам выдается держать в руках подобную жемчужину. Поверьте, эта книга стоит таких слов. С момента публикации Фреда Брукса «No Silver Bullets: Essence and Accidents of Software Engineering» (Brooks, 1987) прошло уже более 30 лет, и наконец-то появился подход, который позволяет нам ответить на один из вопросов, поднятых в той статье и его книге: как отделить существенную сложность от несущественной и управлять этой сложностью — DDD. Эта книга будет хорошим подспорьем для любого IT-инженера, который решает проблемы реальной жизни в сложных предметных областях. Не может не радовать, что Влад Хононов не ограничился теоретическим обзором на DDD, но и разобрал множество практических проблем, с которыми мы сталкиваемся в ходе разработки ПО, причем некоторые выходящие за рамки DDD. Редакторы приложили немало усилий, чтобы книга сохранила все свои ценные качества и в переводе. Надеюсь, что вы будете довольны, желаю приятного чтения.

## Круглов Геннадий Григорьевич

Кандидат технических наук, независимый ИТ-эксперт, автор тренингов, спикер

Начал я применять Domain-Driven Design через пару лет после выхода первой книги Эрика Эванса. Мне тогда эта книга «чудом» попала в руки. На тот момент у меня уже был опыт использования Domain Model, да и в целом опыт предметно-ориентированного проектирования. Но Domain-Driven Design «зашел» сразу, ведь он дает все необходимые паттерны для согласования решения с бизнесом и по понятийному аппарату, и по коду.

Применять Domain-Driven Design доводилось в разных архитектурных стилях — и в монолитных решениях, и в SOA, и потом уже в микросервисных

решениях. Мы в какой-то момент обнаружили забавный факт. Применяя Domain-Driven Design в сервис-ориентированной архитектуре (SOA), мы пришли к микросервисам, поскольку наши решения уже давали нужную гибкость и были по-настоящему выровнены с бизнесом. А это именно то, чем микросервисы фундаментально отличаются от SOA.

Однако при всех плюсах методология Domain-Driven Design требует времени на изучение, при работе с ней нужно набить руку. Часто обучение происходит методом проб и ошибок. Поэтому, будучи консультантом, я не всегда рекомендую Domain-Driven Design в «горячих» проектах, а только тогда, когда команда достаточно зрелая для освоения этого подхода, а компания готова инвестировать в наработку опыта.

С выходом книги Владика изучение Domain-Driven Design стало проще. Книгу, более простую для понимания Domain-Driven Design и в то же время более емкую, я еще не встречал. Рекомендую к прочтению и выражаю благодарность за возможность участия в редактуре этой замечательной книги.

# Предисловие

Предметно-ориентированное проектирование (Domain-driven design, DDD) представляет собой набор методов совместного подхода к созданию программных продуктов с позиции бизнеса, то есть предметной области и ее задач, на которые вы нацеливаетесь. Первоначальный замысел принадлежит Эрику Эвансу (Eric Evans), который в 2003 году опубликовал то, что в DDD-сообществе с любовью называют «Синей книгой». В действительности эта книга называется «Domain-Driven Design: Tackling Complexity in the Heart of Software» (в переводе на русский язык — «Предметно-ориентированное проектирование (DDD): структуризация сложных программных систем»).

Хотя целью предметно-ориентированного проектирования является преодоление сложности и достижение ясности, этой методологией предлагается великое множество замечательных идей, применимых и к менее сложным программным проектам. DDD напоминает нам, что разработчики программных продуктов — не единственные, кто занимается их созданием. Эксперты предметной области, для которых разрабатываются программы, являются носителями критического понимания решаемых задач. Мы формируем партнерские отношения на всех этапах создания программного продукта, поскольку сначала нами применяется «стратегическое проектирование», позволяющее разобраться в бизнес-задаче, также известной как предметная область, и разбить задачу на более мелкие, проще решаемые взаимосвязанные задачи. Партнерство с экспертами в предметной области также побуждает нас общаться на языке этой области, а не заставлять тех, кто занимается бизнесом, изучать технический язык разработки программных продуктов.

Второй этап проекта, разрабатываемого по методологии DDD, — это «тактическое проектирование», когда все, что было усвоено при стратегическом проектировании, превращается в архитектуру и реализацию программного продукта. Опять же, в DDD на этот счет предоставляются рекомендации и шаблоны для организации конкретных предметных областей и исключения ненужной сложности. На этапе тактического проектирования продолжается сотрудничество с экспертами предметной области, которые могут узнать язык своей бизнес-сферы даже при просмотре программного кода, созданного командами разработчиков программного продукта.

За годы, прошедшие со времени публикации «Синей книги», кроме той пользы от изложенных в ней идей, что была получена многими организациями, возникло сообщество опытных практиков DDD. А характерная для DDD природа совместного творчества привела к тому, что это сообщество стало делиться своим опытом и

сложившимися взглядами и создало инструменты, помогающие командам применять эти идеи, извлекая из них практическую пользу. В программном докладе на конференции «Explore DDD» в 2019 году Эрик Эванс призвал сообщество продолжать развитие DDD не только в области практического применения, но и в поиске способов более эффективного обмена идеями.

Это как раз и объясняет причину моей ярой приверженности идеям, изложенным в этой книге. Я уже была поклонницей Влада после прослушивания его выступлений на конференциях и чтения его статей. Им был накоплен солидный опыт практического применения DDD в работе над весьма сложными проектами, и он щедро делился своими знаниями. Эта книга представляет собой уникальное «повествование» о DDD (не историческое, а концептуальное), предоставляя всем читателям превосходную перспективу изучения данной методологии. Книга предназначена для новичков, но, как давний практик DDD, который также пишет и говорит об этой методологии, я обнаружила, что, прочитав книгу, многому научилась, взглянув на DDD с позиции автора. Мне не терпелось сослаться на его книгу в моем курсе DDD Fundamentals на Pluralsight еще до того, как она была опубликована, и я уже делилась точкой зрения автора в беседах с клиентами.

Поначалу работа по методологии DDD может стать обескураживающей. Мы использовали DDD для снижения сложности проектов, а Влад, наряду с этим, старается представить DDD таким образом, чтобы упростить саму тему его применения. И он выходит далеко за рамки простого объяснения принципов DDD. В последней части книги рассказывается о некоторых важных практических приемах, ставших производными от DDD, таких как EventStorming, рассматриваются вопросы развития бизнес-направления или самой организации и возможного влияния этого развития на программный продукт. В ней также рассматривается порядок согласования DDD с микросервисами и приводятся способы интегрирования DDD с множеством широко известных шаблонов, используемых при разработке программных продуктов. Я думаю, что эта книга станет отличным начальным пособием по DDD для новичков, а также очень достойным чтением для опытных специалистов-практиков.

*— Джули Лерман (Julie Lerman),*
*инструктор по разработке программных продуктов,*
*автор книг издательства O'Reilly и приверженец методологии DDD*

# Введение

Я хорошо помню тот день, когда впервые приступил к настоящей работе по созданию программного продукта. Во мне одновременно уживались чувства восторга и ужаса. После взлома в школьные годы программной системы местного бизнеса я очень хотел стать «настоящим программистом» и написать код для одной из крупнейших аутсорсинговых компаний страны.

В мои первые рабочие дни новые коллеги демонстрировали мне все основные приемы работы. Настроив корпоративную почту и пройдя систему учета рабочего времени, мы наконец перешли к самому интересному: корпоративному стилю программирования и стандартам. Мне сказали, что «здесь всегда создается только качественно спроектированный код и используется многоуровневая архитектура». Мы рассмотрели определение каждого из трех уровней — доступа к данным, бизнес-логики и представления, а затем обсудили технологии и платформы, удовлетворяющие потребностям этих уровней. В то время общепринятым решением для хранения данных была платформа Microsoft SQL Server 2000, интегрированная с уровнем доступа к данным посредством ADO.NET. Уровень представления щеголял применением либо WinForms для настольных приложений, либо ASP.NET WebForms для веб-приложений. На знакомство с этими двумя уровнями была потрачена уйма времени, поэтому меня удивил явный дефицит внимания к уровню бизнес-логики:

- «А что вы можете сказать об уровне бизнес-логики?»
- «Да все очень просто. Здесь реализуется бизнес-логика».
- «А что такое бизнес-логика?»
- «Ну, бизнес-логика — это всевозможные циклы и инструкции `if-else`, необходимые для реализации требований».

В тот же день я приступил к выяснению, что же такое бизнес-логика и как она должна быть реализована в качественно спроектированном коде. На поиск ответа у меня ушло более трех лет.

Ответ был найден в культовой книге Эрика Эванса (Eric Evans) «Domain-Driven Design: Tackling Complexity in the Heart of Software» (в переводе на русский язык — «Предметно-ориентированное проектирование (DDD): структуризация сложных программных систем»). Оказалось, что я не ошибся. Бизнес-логика действительно важна: это основа программного продукта! Но, к сожалению, мне потребовалось еще три года, чтобы понять все премудрости, которыми поделился Эрик. И сама

книга была новаторской, и тот факт, что английский — мой третий язык, мне не помог.

Но со временем все встало на свои места, и я примирился с методологией предметно-ориентированного проектирования (domain-driven design, DDD). Я изучил принципы и шаблоны DDD, тонкости моделирования и реализации бизнес-логики, а также способы преодоления сложностей, свойственных создаваемому мною программному продукту. Несмотря на необходимость преодоления всевозможных препятствий, дело того стоило. Знакомство с предметно-ориентированным проектированием стало для меня явлением, изменившим всю мою карьеру.

## Зачем я написал эту книгу

За последние 10 лет я знакомил с предметно-ориентированным проектированием своих коллег из разных компаний, проводил очные занятия и преподавал на онлайн-курсах. Преподавание не только помогло углубить мои знания, но и позволило оптимизировать способы объяснения сути принципов и шаблонов предметно-ориентированного проектирования.

Как это часто бывает, преподавать еще сложнее, чем учиться. Я большой поклонник работ и учения Элиягу М. Голдратта (Eliyahu M. Goldratt). Элиягу говорил, что даже самые сложные системы, если смотреть на них под правильным углом, по своей сути просты. В течение многих лет преподавания DDD я искал модель методологии, которая раскрыла бы простоту, присущую предметно-ориентированному проектированию.

Результатом моих усилий стала эта книга. Ее цель — изложить все, что касается предметно-ориентированного проектирования, простым, доступным языком, сделать проектирование более понятным и пригодным для использования. Я считаю, что методологию DDD трудно переоценить, особенно при разработке современных программных систем. Эта книга даст вам достаточное количество инструментов, позволяющее приступить к применению предметно-ориентированного проектирования в повседневной работе.

## Кому следует прочитать эту книгу

Я считаю, что знание принципов и шаблонов предметно-ориентированного проектирования будет полезно инженерам-программистам всех уровней: младшего, старшего, штатного и руководящего. DDD не только предоставляет инструменты и методы, пригодные для моделирования и эффективного внедрения программного продукта, но и высвечивает часто упускаемый из виду аспект разработки программного продукта: контекст. Обладая знаниями о бизнес-задаче системы, можно будет с гораздо большей эффективностью подобрать подходящее решение, которое не будет страдать недостаточностью или избыточностью, и при этом станет отвечать потребностям и целям бизнеса.

Предметно-ориентированное проектирование имеет куда более важное значение для архитекторов программных средств и тем более для тех, кто только начал делать карьеру архитектора. Присущие этой методологии инструменты принятия решений по стратегическому проектированию помогут разбить большую систему на компоненты — сервисы, микросервисы или подсистемы — и спроектировать порядок интеграции компонентов друг с другом для формирования всей системы.

И наконец, в этой книге будут рассмотрены не только вопросы проектирования программных продуктов, но и порядок развития проекта синхронно с изменениями в его бизнес-контексте. Этот важнейший аспект разработки программного продукта поможет поддерживать конструкцию системы «в форме» в течение длительного времени и не давать ей превращаться в большой ком грязи.

## Навигация по книге

Эта книга состоит из четырех частей: стратегического проектирования, тактического проектирования, практического применения DDD и связи DDD с другими методологиями и шаблонами. В первой части книги рассматриваются инструменты и методы, применяемые для принятия широкомасштабных решений по проектированию программного продукта. Во второй части основное внимание уделяется коду, т. е. различным способам реализации бизнес-логики системы. В третьей части рассматриваются методы и стратегии применения DDD в реальных проектах. А в четвертой части продолжается рассмотрение предметно-ориентированного проектирования, но теперь уже в контексте других методологий и шаблонов.

А вот краткое изложение содержимого каждой главы:

- В *главе 1* определяется контекст проекта разработки программного продукта: область бизнеса, его цели и то, как программное средство собирается их поддерживать.
- В *главе 2* вводится понятие «единого языка»: практики предметно-ориентированного проектирования, призванной добиться эффективного общения и обмена знаниями.
- В *главе 3* рассматриваются приемы, позволяющие справиться со сложностью бизнес-областей и спроектировать высокоуровневые архитектурные компоненты системы: ограниченные контексты.
- В *главе 4* исследуются различные модели организации обмена данными между ограниченными контекстами и их интеграции.
- В *главе 5* рассматриваются шаблоны реализации бизнес-логики, начинающиеся с двух шаблонов, предназначенных для применения в случаях несложной бизнес-логики.
- В *главе 6* осуществляется переход от простой бизнес-логики к сложной, в ходе которого происходит знакомство с шаблоном модели предметной области, который позволяет преодолеть сложности, задаваемые бизнес-логикой.

- В *главе 7* добавляется фактор времени и рассматривается более совершенный способ моделирования и реализации бизнес-логики: модель предметной области, основанная на событиях.
- В *главе 8* основное внимание смещается к более высокому уровню и изучаются три архитектурных шаблона, используемые для структурирования компонентов.
- В *главе 9* рассматриваются шаблоны, необходимые для организации работы компонентов системы.
- В *главе 10* шаблоны, рассмотренные в предыдущих главах, объединяются и из них выводятся простые эмпирические правила, упрощающие процесс принятия проектных решений.
- В *главе 11* исследуются предполагаемые изменения в конструкции программного продукта, вносимые с течением времени, и развитие этой конструкции на протяжении всего жизненного цикла продукта.
- В *главе 12* происходит знакомство с EventStorming: простым семинаром по эффективному обмену знаниями, выработке общего понимания и разработке программных продуктов.
- В *главе 13* рассматриваются трудности, с которыми можно столкнуться при внедрении предметно-ориентированного проектирования в уже существующие проекты.
- В *главе 14* рассматриваются взаимоотношения архитектурного стиля микросервисов и предметно-ориентированного проектирования: в чем их отличия, и где они дополняют друг друга.
- В *главе 15* рассматриваются шаблоны и инструменты предметно-ориентированного проектирования в контексте архитектуры, управляемой событиями.
- В *главе 16* основное внимание смещается от рабочих систем к аналитическим системам управления данными и рассматриваются вопросы взаимодействия предметно-ориентированного проектирования с архитектурой сетки данных.

В конце каждой главы имеются упражнения с рядом вопросов для закрепления полученных знаний. В некоторых вопросах с целью демонстрации различных аспектов предметно-ориентированного проектирования фигурирует вымышленная компания WolfDesk. Прочтите следующее описание WolfDesk и возвращайтесь к нему при ответе на соответствующие вопросы упражнений.

# Пример предметной области: WolfDesk

WolfDesk предоставляет в качестве услуги систему управления заявками службы поддержки. Если вашей начинающей компании необходимо поддерживать клиентов, то с помощью решения WolfDesk можно будет приступить к работе в самые короткие сроки.

Компания WolfDesk использует не такую модель оплаты, как у ее конкурентов. Она взимает плату не за каждого пользователя, а позволяет арендаторам иметь сколько