Оглавление

Предисловие **15**
Условные обозначения, принятые в книге 16
Благодарности 16

Предисловие научного редактора к русскому изданию **19**

Глава 1. Почему именно событийно-управляемые микросервисы? **21**
Что такое событийно-управляемые микросервисы? 22
Введение в предметно-ориентированное проектирование и ограниченные контексты 23
Использование моделей предметной области и ограниченных контекстов 24
Привязка ограниченных контекстов к бизнес-требованиям 25
Структуры обмена информацией 26
Структуры обмена бизнес-информацией 27
Структуры обмена технической информацией 27
Структуры обмена данными 28
Закон Конвея и структуры обмена информацией 29
Структуры обмена информацией в традиционных вычислениях 30
Вариант 1: создать новый сервис 30
Вариант 2: добавить новое бизнес-требование в существующий сервис 31
Плюсы и минусы каждого варианта 31
Командный сценарий: продолжение 32
Конфликтующие обстоятельства 33
Событийно-управляемые коммуникационые структуры 33
События являются основой обмена информацией 34
Потоки событий обеспечивают единственный источник истины 34
Потребители занимаются своим моделированием и выполняют свои запросы 34
Обмен данными улучшается по всей организации 35
Доступные данные поддерживают изменения в обмене информацией 35
Асинхронные событийно-управляемые микросервисы 35
Пример команды, использующей событийно-управляемые микросервисы 36
Синхронные микросервисы 37
Недостатки синхронных микросервисов 38
Связь «точка-точка» 38
Зависимое масштабирование 38
Обработка отказов сервисов 39
Управление версиями API и зависимостями 39
Доступ к данным с привязкой к реализации 39
Распределенные монолиты 39
Тестирование 39

Преимущества синхронных микросервисов 39
Резюме 40

Глава 2. Основы событийно-управляемых микросервисов 41
Построение топологий 41
Топология микросервисов 41
Бизнес-топология 42
Содержимое события 43
Структура события 43
Событие без ключа 44
Сущностное событие 44
Событие с ключом 44
Материализация состояния из сущностных событий 45
Определения и схемы данных событий 47
Принцип единственного источника событий 47
Взаимодействие микросервисов с помощью брокера событий 47
Хранение и обработка событий 48
Дополнительные факторы, которые следует учитывать 49
Брокеры событий или брокеры сообщений? 51
Потребление из неизменяемого журнала 52
Потребление в качестве потока событий 52
Потребление в форме очереди 53
Обеспечение единственного источника истины 53
Масштабное управление микросервисами 54
Размещение микросервисов в контейнерах 54
Размещение микросервисов на виртуальных машинах 54
Управление контейнерами и виртуальными машинами 55
Уплата налога на микросервисы 56
Резюме 56

Глава 3. Обмен информацией и контракты на передачу данных 57
Событийно-управляемые контракты на передачу данных 57
Использование явных схем в качестве контрактов 58
Комментарии к определению схемы 58
Полнофункциональное развитие схемы 59
Поддержка генератора кода 60
Разрушительные изменения схемы 61
Учет критических изменений схемы для событий 62
Выбор формата события 63
Дизайн событий 64
Говорите правду, всю правду и ничего, кроме правды 64
Используйте одно-единственное определение события в расчете на поток 64
Используйте самые точные типы данных 64
Держите события узконаправленными 65
Пример: перегрузка определений событий 66
Минимизируйте размер событий 68
Привлекайте потенциальных потребителей к дизайну события 69
Не используйте события в качестве семафоров и сигналов 69
Резюме 70

Глава 4. Интеграция событийно-управляемых архитектур с существующими системами 71
Что такое освобождение данных? 72
Компромиссы для освобождения данных 73
Конвертация освобожденных данных в события 75
Шаблоны освобождения данных 75
Фреймворки освобождения данных 76
Освобождение данных по запросу 77
Массовая загрузка 77
Инкрементная загрузка временных меток 77
Загрузка с автоинкрементируемым ID 77
Выполнение пользовательских запросов 78
Инкрементное обновление 78
Выгоды от обновления по запросу 78
Недостатки обновления по запросу 79
Освобождение данных с помощью журналов захвата изменений в данных 80
Выгоды от использования журналов хранилища данных 81
Недостатки использования журналов базы данных 82
Освобождение данных с помощью исходящих таблиц 82
Некоторые мысли по поводу производительности 84
Изоляция внутренних моделей данных 84
Обеспечение совместимости схем 86
Выгоды от создания событий с помощью исходящих таблиц 88
Недостатки создания событий с помощью исходящих таблиц 89
Захват изменений в данных с помощью триггеров 89
Выгоды от использования триггеров 92
Недостатки использования триггеров 92
Внесение изменений определения данных в захватываемые наборы данных 93
Обработка произведенных постфактум изменений определения данных для шаблонов запросов и журналов CDC 93
Обработка изменений определения данных для шаблонов захвата таблиц изменений в данных 94
Запись данных о событиях в хранилища данных 94
Влияние на бизнес записи данных в приемники и получения их из источников 95
Резюме 97

Глава 5. Основы событийно-управляемой обработки 99
Топологии без поддержки состояния 100
Преобразования 100
Ветвление и слияние потоков 101
Переподразделение потоков событий 101
Пример переподразделения потока событий 102
Соподразделение потоков событий 103
Пример соподразделения потока событий 103
Назначение разделов экземпляру потребителя 104
Назначение разделов с помощью контроллера разделов 104
Назначение соподразделенных разделов 105
Стратегии назначения разделов 105
Назначение по круговой схеме 105

Статическое назначение 106
Индивидуально настраиваемое назначение 107
Восстановление после отказов экземпляра обработчика без поддержки состояния 107
Резюме 107

Глава 6. Детерминированная обработка потоков 108
Детерминизм событийно-управляемых рабочих процессов 109
Временные метки 109
Синхронизация распределенных временных меток 111
Обработка с помощью событий, помеченных временными метками 111
Пример: выбор порядка событий при обработке многочисленных разделов 112
Планирование событий и детерминированная обработка 112
Индивидуально настраиваемые планировщики событий 113
Обработка на основе времени события, времени обработки и времени получения 113
Извлечение временной метки потребителем 114
Вызовы к внешним системам в форме «запрос-ответ» 114
Водяные знаки 114
Водяные знаки в параллельной обработке 115
Время потока 117
Время потока в параллельной обработке 118
Неупорядоченные и запоздалые события 119
Запоздалые события с водяными знаками и временем потока 120
Причины и последствия неупорядоченных событий 120
Загрузка из источников с неупорядоченными данными 120
Несколько производителей в несколько разделов 121
Чувствительные ко времени функции и управление окнами 122
Переворачивающиеся окна 122
Скользящие окна 123
Сеансовые окна 124
Обработка запоздалых событий 124
Повторная обработка или обработка в режиме, близком к реальному времени 126
Периодические сбои и запоздалые события 127
Проблемы с подключением производителя/брокера событий 127
Резюме и дополнительная литература 129

Глава 7. Потоковая передача с поддержкой состояния 130
Хранилища состояний и материализация состояний из потока событий 130
Запись состояния в поток событий журнала изменений 131
Материализация состояния во внутреннее хранилище состояний 132
Материализация глобального состояния 133
Преимущества использования внутреннего состояния 134
Требования к масштабируемости снимаются с разработчика 134
Высокопроизводительные возможности, основанные на использовании дисков 134
Гибкость использования подключенного через сеть внешнего диска 134
Недостатки использования внутреннего состояния 135
Ограничения при использовании локальных дисков 135
Потери при использовании дискового пространства 135

Масштабирование и восстановление внутреннего состояния 136
Использование «горячих реплик» 136
Восстановление и масштабирование из журналов изменений 138
Восстановление и масштабирование входных потоков событий 138
Материализация состояния во внешнее хранилище состояний 139
Преимущества внешнего состояния 140
Полная локализация данных 140
Технологии 140
Недостатки внешнего состояния 140
Управление многочисленными технологиями 140
Потеря производительности из-за сетевой задержки 141
Финансовые затраты на сервисы внешнего хранилища состояний 141
Полная локальность данных 141
Масштабирование и восстановление с помощью внешних хранилищ состояний 142
Использование исходных потоков 142
Использование журналов изменений 142
Использование моментальных снимков 143
Перестраивание хранилищ состояний или их мигрирование? 143
Перестраивание 143
Мигрирование 144
Транзакции и практически однократная обработка 145
Пример: сервис учета запасов 146
Практически однократная обработка с транзакциями клиент-брокер 147
Практически однократная обработка без транзакции клиент-брокер 148
Генерация дублирующихся событий 149
Идентификация дублирующихся событий 149
Защита от дубликатов 150
Поддержание согласованного состояния 151
Резюме 153

Глава 8. Построение рабочих процессов с помощью микросервисов 154
Шаблон «хореография» 155
Простой пример событийно-управляемой хореографии 156
Создание и изменение хореографического рабочего процесса 157
Мониторинг хореографического рабочего процесса 157
Шаблон «оркестровка» 158
Простой пример событийно-управляемой оркестровки 159
Простой пример оркестровки с прямым вызовом 160
Событийно-управляемая оркестровка в сравнении с окрестровкой с прямым вызовом 160
Создание и изменение рабочего процесса на основе оркестровки 162
Мониторинг процесса на основе оркестровки 162
Распределенные транзакции 162
Хореографические транзакции: шаблон «Сага» 163
Пример хореографической транзакции 163
Транзакции с оркестровкой 165
Компенсационные процессы 167
Резюме 168

Глава 9. Микросервисы с использованием технологии «Функция как сервис»..... 169
Проектирование функционально-ориентированных решений в качестве микросервисов 169
Обеспечивайте строгое членство в ограниченном контексте 169
Фиксируйте смещения только после завершения обработки 170
Когда функция завершила свою обработку 170
Когда функция только что запустилась 170
Меньше значит больше 171
Выбор провайдера FaaS 171
Построение микросервисов из функций 172
«Холодный старт» и «теплые старты» 173
Запуск функций с помощью триггеров 174
Запуск по новому событию: слушатель потока событий 174
Запуск по задержке группы потребителей 176
Запуск по расписанию 177
Запуск с использованием веб-перехватчиков 177
Запуск по событиям ресурсов 177
Решение бизнес-задач с помощью функций 178
Поддержание состояния 178
Функции, вызывающие другие функции 179
Шаблон событийно-управляемого обмена информацией 179
Шаблон прямого вызова 180
Хореография и асинхронный вызов функций 180
Оркестровка и синхронный вызов функций 182
Завершение работы функции и выключение 183
Тонкая настройка функций 184
Выделение достаточных ресурсов 184
Параметры пакетной обработки событий 184
Масштабирование решений FaaS 185
Резюме 186

Глава 10. Микросервисы на основе базового шаблона производителя и потребителя 187
Где BPC работают хорошо? 187
Интеграция с существующими и унаследованными системами 188
Пример: шаблон «Коляска» 188
Бизнес-логика с поддержкой состояния без учета порядка событий 189
Пример: книгоиздание 189
Когда уровень данных выполняет значительную часть работы 190
Независимое масштабирование уровня обработки и данных 191
Пример: агрегирование данных о событиях для создания профилей взаимодействия с пользователями 191
Гибридные приложения BPC с внешней потоковой обработкой 192
Пример: использование внешнего фреймворка обработки потоков для объединения потоков событий 192
Резюме 194

Глава 11. «Тяжеловесные» фреймворки для микросервисов 195
Краткая история «тяжеловесных» фреймворков 196
Внутренняя логика работы «тяжеловесных» фреймворков 197
Выгоды и ограничения 199
Варианты настройки кластера и режимы исполнения 200
Используйте сервис, размещенный на хосте провайдера 201
Постройте свой собственный полный кластер 201
Создайте кластер, интегрированный с CMS 201
Развертывание и запуск кластера с использованием CMS 201
Указание ресурсов для одного задания с использованием CMS 202
Режимы передачи приложений в кластер 203
Драйверный режим 203
Кластерный режим 203
Обработка состояния и использование контрольных точек 204
Масштабирование приложений и обработка разделов потока событий 205
Масштабирование приложения во время его работы 206
Масштабирование приложения путем его перезапуска 209
Автоматическое масштабирование приложений 209
Восстановление после отказов 210
Рекомендации по части мультитенантности 210
Языки и синтаксис 211
Выбор фреймворка 211
Пример: обработка щелчков и просмотров с помощью сеансовых окон 212
Резюме 215

Глава 12. «Легковесные» фреймворки для микросервисов 216
Выгоды и ограничения 216
«Легковесная» обработка 217
Обработка состояния и использование журналов изменений 218
Масштабирование приложений и восстановление после отказов 218
Перераспределение событий 219
Назначение состояния 219
Репликация состояния и «горячие реплики» 220
Выбор «легковесного» фреймворка 220
Apache Kafka Streams 221
Apache Samza: режим встраивания 221
Языки и синтаксис 221
Операция соединения «поток-таблица-таблица»: шаблон обогащения 222
Резюме 226

Глава 13. Интегрирование событийно-управляемых микросервисов с микросервисами типа «запрос-ответ» 227
Обработка внешних событий 227
Автономно генерируемые события 228
Реактивно генерируемые события 228
Обработка автономно генерируемых аналитических событий 229
Интегрирование со сторонними API «запрос-ответ» 230
Обработка и обслуживание данных с поддержкой состояния 232
Обслуживание запросов реального времени с помощью внутренних хранилищ состояний 233

Обслуживание запросов реального времени с помощью внешних хранилищ состояний 236
Обслуживание запросов путем материализации событийно-управляемого микросервиса 236
Обслуживание запросов через отдельный микросервис 237
Обработка запросов внутри событийно-управляемого рабочего процесса 239
Обработка событий для пользовательских интерфейсов 240
Пример: рабочий процесс публикации газет (шаблон одобрения) 241
Разделение сервисов одобрения редактором и рекламодателем 245
Микрофронтенды в приложениях на основе запросов-ответов 247
Выгоды от микрофронтендов 248
Композиционные микросервисы 249
Простота привязки к бизнес-требованиям 249
Недостатки микрофронтендов 249
Потенциальное отсутствие единообразия элементов пользовательского интерфейса и стилизации 249
Различающаяся производительность микрофронтендов 250
Пример: приложение поиска источников впечатлений и отзывов о них 251
Резюме 254

Глава 14. Вспомогательные инструменты 255
Система закрепления микросервисов за командами 255
Создание и модифицирование потока событий 256
Разметка потока событий метаданными 256
Квоты 257
Реестр схем 258
Оповещение о создании и модификации схем 259
Управление смещением 259
Разрешения и списки контроля доступа для потоков событий 260
Управление состоянием и сброс приложения 261
Мониторинг запаздывания смещения потребителей 262
Оптимизированный процесс создания микросервисов 263
Элементы управления контейнерами 264
Создание кластеров и управление ими 264
Программная доводка брокеров событий 265
Программная доводка вычислительных ресурсов 265
Межкластерная репликация событийных данных 266
Программная доводка инструментария 266
Отслеживание зависимостей и визуализация топологии 266
Пример топологии 268
Резюме 271

Глава 15. Тестирование событийно-управляемых микросервисов 272
Общие принципы тестирования 272
Модульное тестирование функций топологии 273
Функции без поддержки состояния 273
Функции с поддержкой состояния 273
Тестирование топологии 274
Тестирование развития и совместимости схем 275

Интеграционное тестирование событийно-управляемых микросервисов 275
Локальное интеграционное тестирование 276
Создание временной среды внутри среды выполнения тестируемого кода 278
Создание временной среды, внешней по отношению к тестируемому коду 280
Интеграция размещенных на хосте сервисов с использованием возможностей имитации и симуляции 281
Интеграция удаленных сервисов, не имеющих локальных возможностей 282
Полное интеграционное тестирование 283
Программное создание временной среды интеграционного тестирования 283
Заполнение событиями с использованием реальных производственных данных 284
Заполнение событиями из подготовленного источника тестирования 285
Создание имитационных событий с использованием схем 285
Тестирование с использованием единой общей среды 286
Тестирование с использованием производственной среды 287
Выбор стратегии полного интеграционного тестирования 288
Резюме 289

Глава 16. Развертывание событийно-управляемых микросервисов 291
Принципы развертывания микросервисов 291
Архитектурные компоненты развертывания микросервисов 292
Системы непрерывной интеграции, доставки и развертывания 292
Системы управления контейнерами и стандартное аппаратное обеспечение 294
Базовый шаблон полного развертывания 294
Шаблон непрерывного обновления 296
Шаблон разрушительных изменений схемы 297
Продолженная миграция через два потока событий 298
Синхронизированная миграция в новый поток событий 299
Шаблон «сине-зеленого» развертывания 300
Резюме 301

Глава 17. Заключение 302
Уровни обмена информацией 302
Предметные области бизнеса и ограниченные контексты 302
Совместные инструменты и инфраструктура 303
Схематизированные события 304
Освобождение данных и единственный источник истины 304
Микросервисы 305
Варианты реализации микросервисов 305
Тестирование 306
Развертывание 307
Завершающие слова 308

Предметный указатель 309

Предисловие

Я написал эту книгу, потому что сам хотел бы иметь такую книгу, когда начинал свое путешествие в мир событийно-управляемых микросервисов. Она стала квинтэссенцией моего личного опыта, дискуссий с другими людьми и бесчисленных блогов, книг, постов, бесед, конференций и документов, связанных с той или иной частью предметной области событийно-управляемых микросервисов. Я обнаружил, что во многих работах, которые я читал, упоминались событийно-управляемые архитектуры либо только вскользь, либо с недостаточной глубиной. Некоторые из них охватывали только отдельные аспекты этих архитектур и, хотя и были полезными, но составляли лишь малую часть пазла. Другие работы показались мне упрощенными и пренебрежительными, поскольку утверждали, что событийно-управляемые системы на самом деле полезны лишь в том, чтобы одна система отправляла асинхронное сообщение непосредственно другой, подменяя синхронные системы «запросов-ответов». Как подробно рассказывается в этой книге, событийно-управляемые архитектуры — это гораздо большее, чем такое ограниченное представление.

Инструменты, которые мы используем, существенно влияют на то, какую пользу мы с их помощью для себя извлекаем. Архитектуры событийно-управляемых микросервисов становятся возможными благодаря целому ряду технологий, которые только недавно стали легкодоступными. В этой книге в основе архитектур и шаблонов дизайна лежат распределенные, отказоустойчивые, высокопроизводительные и высокоскоростные брокеры событий. Эти технологические решения основаны на конвергенции больших данных и необходимости обработки событий почти в реальном времени. Микросервисам способствует простота контейнеризации и получения вычислительных ресурсов, что обеспечивает простоту в размещении, масштабировании и управлении сотнями тысяч микросервисов.

Технологии, поддерживающие событийно-управляемые микросервисы, оказывают значительное влияние на то, что мы думаем о задачах и как их решаем, а также на то, как структурированы наши предприятия и организации. Событийно-управляемые микросервисы меняют характер работы предприятия, способы решения задач и взаимодействия коллективов, людей и бизнес-единиц. Эти инструменты дают нам действительно новый способ осуществлять виды деятельности, которые до недавнего времени были невозможны.

Условные обозначения, принятые в книге

В книге используются следующие типографские условные обозначения:

- *Курсив*

  Им выделяются новые термины.

- **Полужирный шрифт**

  Им выделяются интернет-адреса (URL) и адреса электронной почты.

- Шрифт Arial

  Им выделяются пути к файлам, имена файлов и расширения файлов.

- `Моноширинный шрифт`

  Он используется для листингов программ, а также внутри абзацев для выделения упоминаемых там элементов программ — таких как переменные, инструкции языка и ключевые слова.

- **`Полужирный моноширинный шрифт`**

  Им выделяются команды либо другой текст, который должен быть напечатан непосредственно пользователем.

- *`Моноширинный шрифт курсивом`*

  Им выделяется текст, который должен быть заменен значениями пользователя либо значениями, определяемым по контексту.

Такой значок обозначает подсказку или совет.

Такой значок обозначает общее примечание.

Такой значок обозначает предупреждение или предостережение.

На веб-странице этой книги по адресу **https://oreil.ly/building-event-driven-microservices** приведены описания ошибок, примеры и прочая дополнительная информация.

Благодарности

Я хотел бы выразить свое уважение и благодарность сотрудникам компании Confluent, которые, наряду с изобретением Apache Kafka, являются одними из первых, кто по-настоящему «понимает», когда речь идет о событийно-управляемых архитектурах. Мне посчастливилось получить от одного из них, Бена Стопфорда

(Ben Stopford), ведущего технолога офиса технического директора, обширные и ценные отзывы. Скотт Моррисон (Scott Morrison), технический директор PHEMI Systems, также поделился со мной ценными идеями, отзывами и рекомендациями. Я выражаю свою благодарность и признательность Скотту и Бену за то, что они помогли сделать эту книгу тем, чем она является сегодня. Будучи главными ее рецензентами и техническими экспертами, они помогли мне усовершенствовать содержащиеся в ней идеи, улучшить качество контента, не дали мне разместить неверную информацию и помогли правильно рассказать о событийно-управляемых архитектурах.

Я также хотел бы выразить благодарность моим друзьям Джастину Токарчуку (Justin Tokarchuk), Гэри Грэму (Gary Graham) и Нику Грину (Nick Green), которые вычитали и отредактировали множество моих черновиков. Вместе со Скоттом и Беном они помогли мне выявить в моем повествовании наиболее существенные слабые места, предложили пути их улучшения, а также поделились своими мыслями и личным опытом в отношении излагаемого материала.

Я также благодарю сотрудников издательства O'Reilly за то, что они помогали мне самыми разнообразными способами. По ходу дела я работал со многими замечательными людьми, но, в частности, хотел бы поблагодарить моего редактора, Корбина Коллинза (Corbin Collins), за то, что он поддерживал меня в трудные времена и помогал мне удержаться на верном пути. Он стал отличным коллегой в этом начинании, и я ценю усилия, которые он приложил, чтобы меня поддержать.

Рэйчел Монаган (Rachel Monaghan), мой редактор, напомнила мне о школьных годах, когда мои сочинения возвращались ко мне расчерканными красным цветом. Я чрезвычайно благодарен ей за острый глаз и знание английского языка — она помогла сделать эту книгу намного проще для чтения и понимания. Спасибо, Рейчел.

Кристофер Фошер (Christopher Faucher) был со мной очень терпелив, предоставлял мне отличные подсказки и не моргнув глазом позволил мне в последний момент внести в книгу ряд важных нетривиальных изменений. Спасибо, Крис.

Майк Лукидес (Mike Loukides), вице-президент по контентной стратегии, был одним из моих первых контактов в O'Reilly. Когда я обратился к нему со своим пространным многословным предложением, он терпеливо со мной работал, чтобы конкретизировать его и превратить в основу для книги, которую вы сегодня читаете. Я благодарен ему за то, что он нашел время поработать со мной и в конечном счете помог мне продвинуться вперед в этой работе. Я изо всех сил старался прислушиваться к его предостережениям, чтобы не создать том, который соперничал бы по объему со словарем.

Хочу сказать моим маме и папе, что я благодарен им за то, что они научили меня по достоинству ценить письменную речь. Я благодарен вам за любовь и поддержку. Мой отец познакомил меня с Маршаллом Маклюэном (Marshall McLuhan), и хотя стоит признаться, что я не читал большинство из того, что он написал, я чрезвычайно признателен ему за его утверждение о том, что информационная среда влияет на сообщение. Это изменило мой взгляд на системные архитектуры и их оценку.

Наконец, спасибо всем остальным, кто так или иначе внес большой или малый вклад в поддержку меня и этой работы. И таких людей, которые внесли свой вклад каждый по-своему — в разговорах, блог-постах, презентациях, открытом исходном коде, курьезах, личном опыте, историях и импровизированных рассуждениях, — очень много. Спасибо вам всем и каждому.

Работа над этой книгой доставила мне и удовольствие, и разочарование. Нередко я проклинал себя за то, что ее начал, но, к счастью, еще больше я был рад, когда ее завершил. Надеюсь, что эта книга поможет вам, дорогой читатель, в какой-то мере прикоснуться к миру событийно-управляемых микросервисов и найти в нем место для себя.

Предисловие научного редактора к русскому изданию

Работа над современной книгой по информационным технологиям — большая ответственность для переводчика и редактора. Профессиональная литература в переводе доходит до русскоязычного читателя с опозданием минимум на год. Поэтому в нашем сообществе не всегда успевает сложиться русскоязычная терминология, отражающая соответствующие слова в английском первоисточнике. Отсюда — сложности в использовании тех или иных терминов в переводе книги.

При редактировании перевода этой книги я старался применять терминологию, которую ранее использовали в переводных книгах издательства O`Reilly по созданию микросервисов и платформы обработки потоковых данных Apache Kafka — например, «микросервис», «слабая связанность», «сильное зацепление», «производитель», «потребитель», «материализация». Если аналогов не было, обращался к терминологии в отрасли из собственного опыта работы — например, «оркестровка», «хореография», «шаблон», «предметно-ориентированное проектирование». Если на практике в большинстве случаев используется английский эквивалент, и он уже привычен и понятен читателю (например, «фреймворк»), я без раздумий использовал его. Если же таких аналогов в отрасли и в книгах нет (например, «grandfathered systems» — буквально «дедушкины системы»), то старался подбирать максимально точный аналог в русском языке.

Для сохранения точности перевода и отсылки читателя к англоязычным первоисточникам ко всем важным для понимания книги терминам в скобках приведен оригинальный английский вариант термина из первоисточника.

Верю, что изучение этой книги позволит сократить временной разрыв между получением столь важной для специалистов информации в английском и русском варианте.

Научный редактор
Тимур Напреев

ГЛАВА 1

Почему именно событийно-управляемые микросервисы?

> The medium is the message[1].
> – *Маршалл Маклюэн*

Маклюэн утверждает, что на человечество влияет и вносит фундаментальные изменения в общество не содержимое средств массовой информации, а вовлеченность в их среду. Газеты, радио, телевидение, Интернет, мгновенные сообщения и социальные сети изменили человеческое взаимодействие и социальные структуры благодаря нашей коллективной в них вовлеченности.

То же самое относится и к архитектуре компьютерных систем. Достаточно взглянуть на историю компьютерных изобретений, чтобы увидеть, как обмен данными по сети, реляционные базы данных, разработки в области больших данных (big data) и облачные вычисления существенно изменили способы построения таких архитектур и выполнения работы с их помощью. Каждое из этих изобретений изменило не только то, как та или иная технология использовалась в различных программно-информационных проектах, но и то, как организации, коллективы и люди обменивались между собой информацией. От централизованных мейнфреймов до распределенных мобильных приложений — каждая новая среда принципиально изменила отношение людей к вычислительной технике.

Информационная среда асинхронно производимого и потребляемого события претерпела принципиальный сдвиг под воздействием современных технологий. Эти события теперь могут храниться бесконечно, в чрезвычайно больших масштабах и потребляться любым сервисом столько раз, сколько необходимо. Вычислительные ресурсы могут легко приобретаться и подключаться по требованию, что облегчает создание микросервисов и управление ими. Микросервисы могут хранить свои данные и управлять ими в соответствии со своими собственными требованиями, причем делать это в масштабах, которые ранее ограничивались пакетно-ориентированными решениями на основе больших данных. Эти усовершенствования скромной и простой событийно-управляемой среды имеют далеко идущие послед-

[1] Не утихают споры о том, что Маршалл Маклюэн в действительности хотел сказать своим знаменитым афоризмом: «The Medium is the Message». По мнению авторитетного российского маклюэноведа И. Б. Архангельской: «…Маклюэн, возможно, хотел сказать, что тип и форма медиа (generic form of media) важнее того значения (meaning) или содержания (content), которое оно передает, т. е. сама форма средства коммуникации меняет наше сознание». — *Прим. ред.*

ствия, которые не только изменяют архитектуру компьютеров, но и полностью меняют то, как коллективы, люди и организации создают системы и строят предприятие.

Что такое событийно-управляемые микросервисы?

Микросервисы и архитектуры в стиле микросервисов существуют уже много лет в самых разных формах и под самыми разными названиями. Сервисно-ориентированные архитектуры (Service-Oriented Architectures, SOA) часто состоят из многочисленных сервисов, синхронно обменивающихся информацией непосредственно друг с другом. В архитектурах, основанных на передаче сообщений, события используются для асинхронного обмена данными между приложениями. Обмен информацией на основе событий, безусловно, не нов, но необходимость в обработке больших массивов данных в реальном времени является новинкой и требует изменения старых архитектурных стилей.

В современной архитектуре событийно-управляемых микросервисов системы обмениваются информацией, *отправляя* (issuing) и *потребляя* (consuming) события. Эти события не уничтожаются при потреблении, как в системах передачи сообщений, но вместо этого остаются легко доступными для других потребителей, которые могут их читать по мере необходимости. В этом состоит важное различие, поскольку оно позволяет задействовать мощные шаблоны, описанные в этой книге.

Сами сервисы малы и создаются специально, чтобы выполнять специфические функции бизнеса. В типичной ситуации под *малым размером сервиса* понимается то, что на его написание уходит не более двух недель. По другому определению сервис должен быть способен (концептуально) «укладываться в голове». Эти сервисы потребляют события из входных потоков событий, выполняют свою специфическую бизнес-логику и могут генерировать свои собственные исходящие события, предоставлять данные для доступа по схеме «запрос-ответ», обмениваться информацией со сторонним API или выполнять другие необходимые действия. Как будет подробно описано в этой книге, такие сервисы могут хранить или не хранить состояние, быть сложными или простыми, могут быть реализованы как автономные приложения или как функция с использованием технологии «Функция как сервис».

Эта комбинация потоков событий и микросервисов образует взаимосвязанный граф активности всей деловой организации. Традиционные компьютерные архитектуры, состоящие из приложений-монолитов и межмонолитного обмена информацией, имеют сходную графовую структуру. Оба этих графа показаны на рис. 1.1.

Выяснение того, как заставить эту графовую структуру работать эффективно, включает в себя рассмотрение двух основных компонентов: узлов и соединений. В этой главе мы рассмотрим каждый из них по очереди.

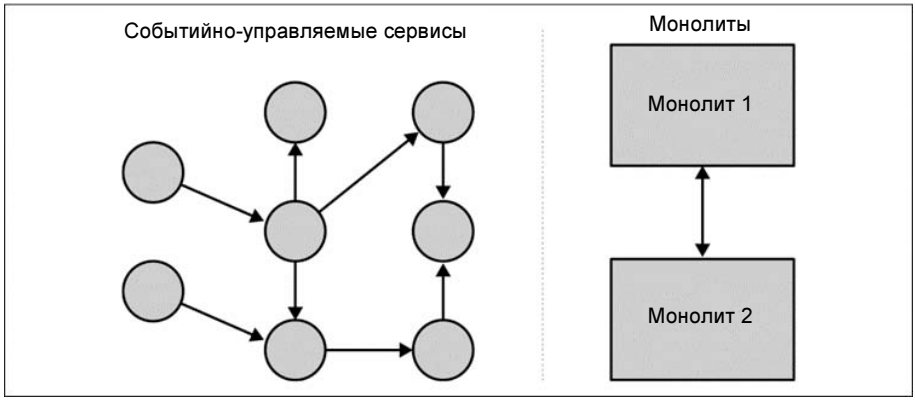

Рис. 1.1. Графовые структуры микросервисов (*слева*) и монолитов (*справа*)

Введение в предметно-ориентированное проектирование и ограниченные контексты

Дизайн на основе предметно-ориентированного проектирования, описанный Эриком Эвансом в его книге «Domain-driven design» (**https://oreil.ly/3fGwK**), вводит несколько концепций, необходимых для построения событийно-управляемых микросервисов. Обсуждая эту тему с учетом значительного количества готовых статей (**https://oreil.ly/zAXqd**), книг (**https://oreil.ly/XwjR3**) и лучших блог-постов месяца, я буду здесь предельно краток.

Итак, в основе предметно-ориентированного проектирования лежат следующие концепции:

- *Домен.*

  Предметная область, в которой работает бизнес. Сюда входит все, с чем предприятие повседневно имеет дело, включая правила, процессы, идеи, специфичную для предприятия терминологию и все, что связано с его пространством решений, *независимо от того, описано ли это на предприятии или нет*. Предметная область существует независимо от существования предприятия.

- *Поддомен.*

  Компонент предметной области. Каждый поддомен сфокусирован на конкретном подмножестве функций и, как правило, отражает некую организационную структуру предприятия (например, Склад, Продажи и Инжиниринг). Поддомен может рассматриваться как самостоятельный домен. Поддомены, как и домены, принадлежат пространству решений.

- *Доменная (и поддоменная) модель.*

  Абстракция реальной предметной области, полезная для целей предприятия. Для генерирования модели используются наиболее важные для предприятия части и свойства предметной области. Главная модель предметной области предприятия

видна через продукты, которые предприятие предоставляет своим клиентам, через интерфейсы, с помощью которых клиенты взаимодействуют с продуктами, и через различные другие процессы и функции, с помощью которых предприятие достигает своих заявленных целей. Модели часто нуждаются в доработке по мере изменения домена и смены бизнес-приоритетов. Модель предметной области является частью пространства решений, поскольку это конструкция, которую предприятия используют для решения своих задач.

- *Ограниченный контекст.*

  Логические границы, включая входы, выходы, события, требования, процессы и модели данных, относящиеся к поддомену. Хотя в идеале ограниченный контекст и поддомен будут полностью совпадать, унаследованные системы, технические взаимосвязи и интеграции со сторонними компонентами часто создают исключения. Ограниченные контексты также являются свойством пространства решений и оказывают значительное влияние на то, как микросервисы взаимодействуют друг с другом.

Ограниченным контекстам следует быть *сильно зацепленными* (highly cohesive). Внутренние операции контекста должны быть интенсивными и тесно взаимосвязанными, причем подавляющий объем обмена информацией происходит внутри него и не выходит за его границы. Сильное зацепление позволяет сократить объем проектирования и упростить реализацию.

Соединения между ограниченными контекстами должны быть *слабо связанными* (loosely coupled), поскольку изменения, вносимые в пределах одного ограниченного контекста, должны минимизировать или исключить их влияние на соседние контексты. Слабая связь может гарантировать, что изменение требований в одном контексте не потребует внесения зависимых изменений в соседние контексты.

Использование моделей предметной области и ограниченных контекстов

Каждая организация образует единый домен между собой и внешним миром. Каждый, кто работает в организации, предпринимает усилия для удовлетворения ее потребностей в ее предметной области.

Этот единый домен подразделяется на поддомены. Для технологически ориентированной компании — это могут быть, например, поддомены инженерного отдела, отдела продаж и отдела поддержки клиентов. Каждый поддомен имеет свои собственные требования и обязанности и сам может быть подразделен на еще более мелкие «подподдомены». Такой процесс разделения повторяется до тех пор, пока модели поддоменов не станут детализированными и действенными и пока команды разработчиков не смогут реализовать их в малые и независимые сервисы. Ограниченные контексты устанавливаются вокруг этих поддоменов и формируют основу для создания микросервисов.