Monolith to Microservices

Evolutionary Patterns to Transform Your Monolith

Sam Newman

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY

Table of Contents

Preface ix

1. Just Enough Microservices 1
What Are Microservices? 1
Independent Deployability 2
Modeled Around a Business Domain 2
Own Their Own Data 5
What Advantages Can Microservices Bring? 6
What Problems Do They Create? 6
User Interfaces 7
Technology 8
Size 8
And Ownership 10
The Monolith 12
The Single Process Monolith 12
The Distributed Monolith 14
Third-Party Black-Box Systems 14
Challenges of Monoliths 15
Advantages of Monoliths 15
On Coupling and Cohesion 16
Cohesion 17
Coupling 17
Just Enough Domain-Driven Design 28
Aggregate 29
Bounded Context 31
Mapping Aggregates and Bounded Contexts to Microservices 31
Further Reading 32
Summary 32

2. Planning a Migration 33
Understanding the Goal 33
Three Key Questions 35
Why Might You Choose Microservices? 35
Improve Team Autonomy 35
Reduce Time to Market 37
Scale Cost-Effectively for Load 37
Improve Robustness 38
Scale the Number of Developers 40
Embrace New Technology 41
When Might Microservices Be a Bad Idea? 42
Unclear Domain 43
Startups 43
Customer-Installed and Managed Software 44
Not Having a Good Reason! 45
Trade-Offs 45
Taking People on the Journey 47
Changing Organizations 47
Establishing a Sense of Urgency 48
Creating the Guiding Coalition 48
Developing a Vision and Strategy 49
Communicating the Change Vision 50
Empowering Employees for Broad-Based Action 51
Generating Short-Term Wins 51
Consolidating Gains and Producing More Change 52
Anchoring New Approaches in the Culture 52
Importance of Incremental Migration 53
It's Production That Counts 53
Cost of Change 54
Reversible and Irreversible Decisions 54
Easier Places to Experiment 56
So Where Do We Start? 56
Domain-Driven Design 56
How Far Do You Have to Go? 57
Event Storming 58
Using a Domain Model for Prioritization 58
A Combined Model 60
Reorganizing Teams 62
Shifting Structures 62
It's Not One Size Fits All 63
Making a Change 65
Changing Skills 68

How Will You Know if the Transition Is Working? 71
Having Regular Checkpoints 71
Quantitative Measures 72
Qualitative Measures 72
Avoiding the Sunk Cost Fallacy 73
Being Open to New Approaches 73
Summary 74

3. Splitting the Monolith 75
To Change the Monolith, or Not? 76
Cut, Copy, or Reimplement? 76
Refactoring the Monolith 77
Migration Patterns 78
Pattern: Strangler Fig Application 79
How It Works 79
Where to Use It 81
Example: HTTP Reverse Proxy 83
Data? 86
Proxy Options 86
Changing Protocols 90
Example: FTP 93
Example: Message Interception 94
Other Protocols 97
Other Examples of the Strangler Fig Pattern 97
Changing Behavior While Migrating Functionality 97
Pattern: UI Composition 98
Example: Page Composition 99
Example: Widget Composition 99
Example: Micro Frontends 103
Where to Use It 104
Pattern: Branch by Abstraction 104
How It Works 105
As a Fallback Mechanism 111
Where to Use It 112
Pattern: Parallel Run 113
Example: Comparing Credit Derivative Pricing 113
Example: Homegate Listings 115
Verification Techniques 116
Using Spies 116
GitHub Scientist 117
Dark Launching and Canary Releasing 118
Where to Use It 118

Pattern: Decorating Collaborator 118
Example: Loyalty Program 119
Where to Use It 120
Pattern: Change Data Capture 120
Example: Issuing Loyalty Cards 120
Implementing Change Data Capture 121
Where to Use It 124
Summary 124

4. Decomposing the Database 125
Pattern: The Shared Database 125
Coping Patterns 127
Where to Use It 127
But It Cant Be Done! 127
Pattern: Database View 128
The Database as a Public Contract 129
Views to Present 130
Limitations 131
Ownership 131
Where to Use It 132
Pattern: Database Wrapping Service 132
Where to Use It 134
Pattern: Database-as-a-Service Interface 135
Implementing a Mapping Engine 136
Compared to Views 137
Where to Use It 137
Transferring Ownership 137
Pattern: Aggregate Exposing Monolith 138
Pattern: Change Data Ownership 141
Data Synchronization 143
Pattern: Synchronize Data in Application 145
Step 1: Bulk Synchronize Data 145
Step 2: Synchronize on Write, Read from Old Schema 146
Step 3: Synchronize on Write, Read from New Schema 147
Where to Use This Pattern 148
Where to Use It 148
Pattern: Tracer Write 149
Data Synchronization 152
Example: Orders at Square 154
Where to Use It 158
Splitting Apart the Database 158
Physical Versus Logical Database Separation 158

Splitting the Database First, or the Code? 160
Split the Database First 161
Split the Code First 165
Split Database and Code Together 170
So, Which Should I Split First? 170
Schema Separation Examples 171
Pattern: Split Table 171
Where to Use It 173
Pattern: Move Foreign-Key Relationship to Code 173
Moving the Join 175
Data Consistency 176
Where to Use It 178
Example: Shared Static Data 178
Transactions 187
ACID Transactions 187
Still ACID, but Lacking Atomicity? 188
Two-Phase Commits 190
Distributed Transactions—Just Say No 193
Sagas 193
Saga Failure Modes 195
Implementing Sagas 199
Sagas Versus Distributed Transactions 205
Summary 206

5. Growing Pains 207
More Services, More Pain 207
Ownership at Scale 209
How Can This Problem Show Itself? 209
When Might This Problem Occur? 210
Potential Solutions 210
Breaking Changes 210
How Can This Problem Show Itself? 211
When Might This Problem Occur? 211
Potential Solutions 212
Reporting 215
When Might This Problem Occur? 216
Potential Solutions 216
Monitoring and Troubleshooting 217
When Might These Problems Occur? 218
How Can These Problems Occur? 218
Potential Solutions 218
Local Developer Experience 222

How Can This Problem Show Itself? 223
When Might This Occur? 223
Potential Solutions 223
Running Too Many Things 224
How Might This Problem Show Itself? 224
When Might This Problem Occur? 224
Potential Solutions 224
End-to-End Testing 226
How Can This Problem Show Itself? 226
When Might This Problem Occur? 226
Potential Solutions 227
Global Versus Local Optimization 229
How Can This Problem Show Itself? 229
When Might This Problem Occur? 229
Potential Solutions 230
Robustness and Resiliency 232
How Can This Problem Show Itself? 232
When Might This Problem Occur? 232
Potential Solutions 232
Orphaned Services 233
How Can This Problem Show Itself? 233
When Might This Problem Occur? 234
Potential Solutions 234
Summary 236

6. Closing Words 237

A. Bibliography 239

B. Pattern Index 243

Index 245