AVOID THE DISTRIBUTED MONOLITH

Moving Towards Mature Scalable Microservices with Spring

Philipp Gerhard
@pgerhard92

Philipp Gerhard

Dev, Tech Lead, Tech Principal

pgerhard_dev@icloud.com

@pgerhard92

@philippgerhard@toot.thoughtworks.com

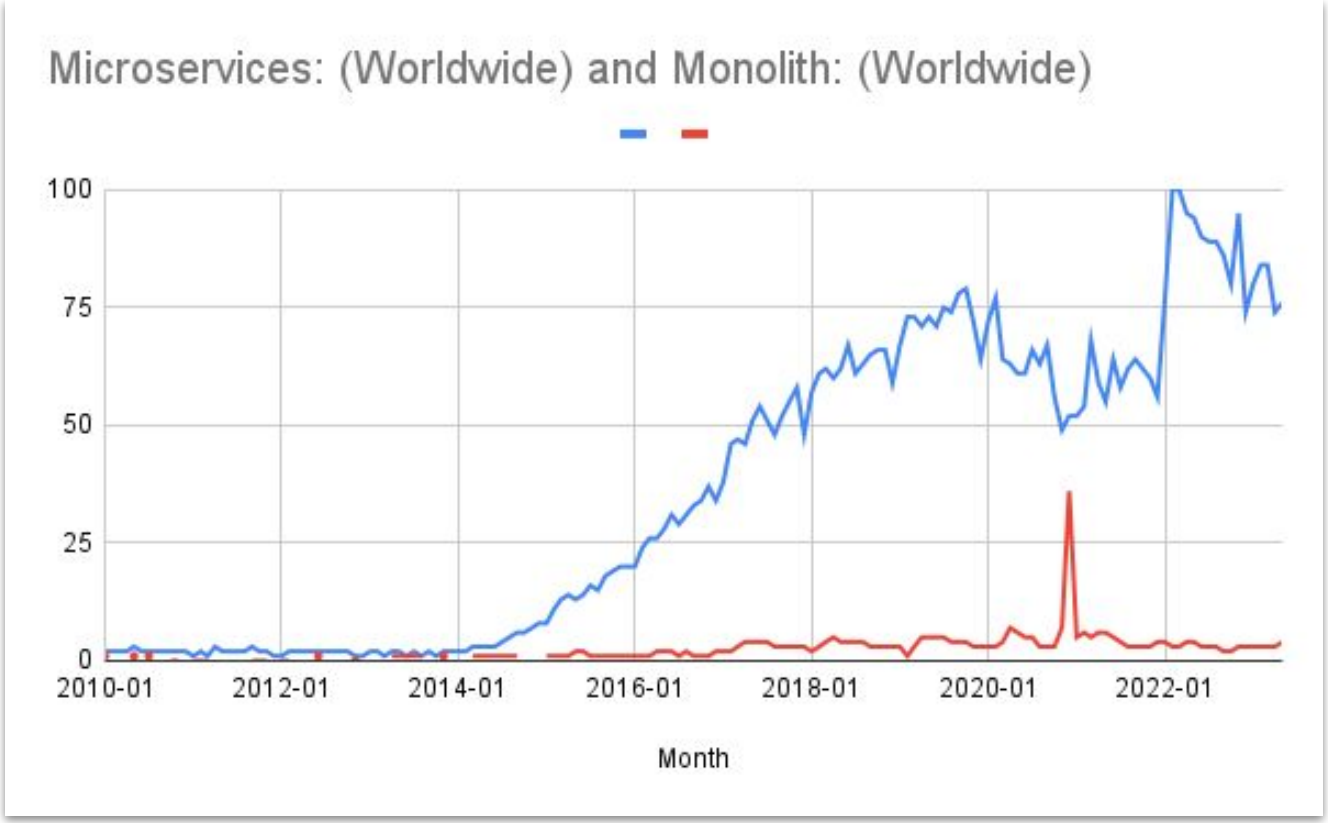
pgerhard

Slides: **TODO insert link**

/thoughtworks

Disclaimer

The views and opinions expressed in this talk are my own and do not necessarily reflect the views or opinions of my current or past employers.


Microservices: (Worldwide) and Monolith: (Worldwide)
100
75
50
25
0
2010-01
2012-01
2014-01
2016-01
2018-01
2020-01
2022-01
Month

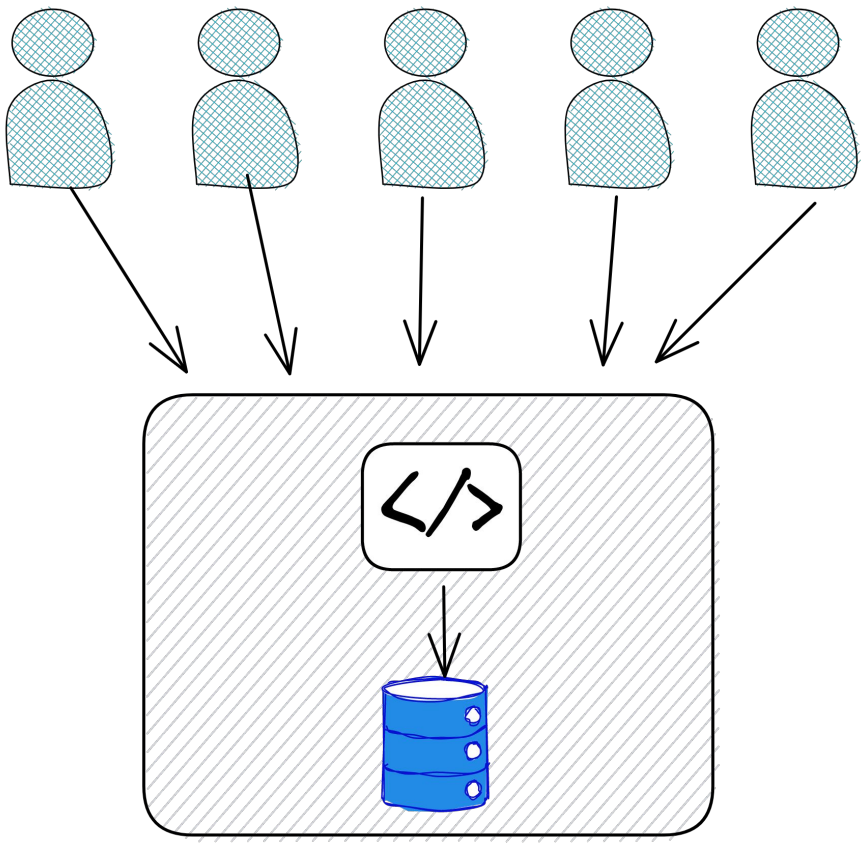
AVOID THE DISTRIBUTED MONOLITH

Moving Towards Mature Scalable Microservices (with Spring)

Philipp Gerhard
@pgerhard92

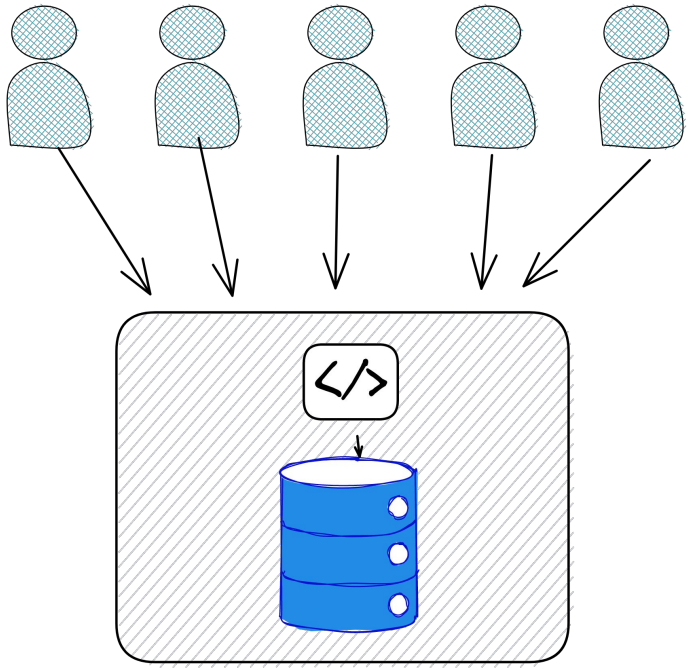
Monolith vs Microservice

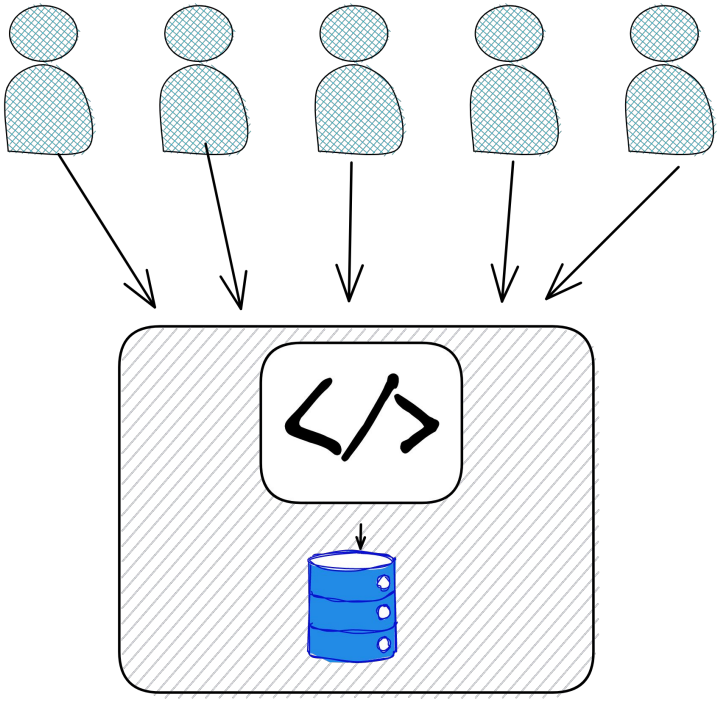
Monolith



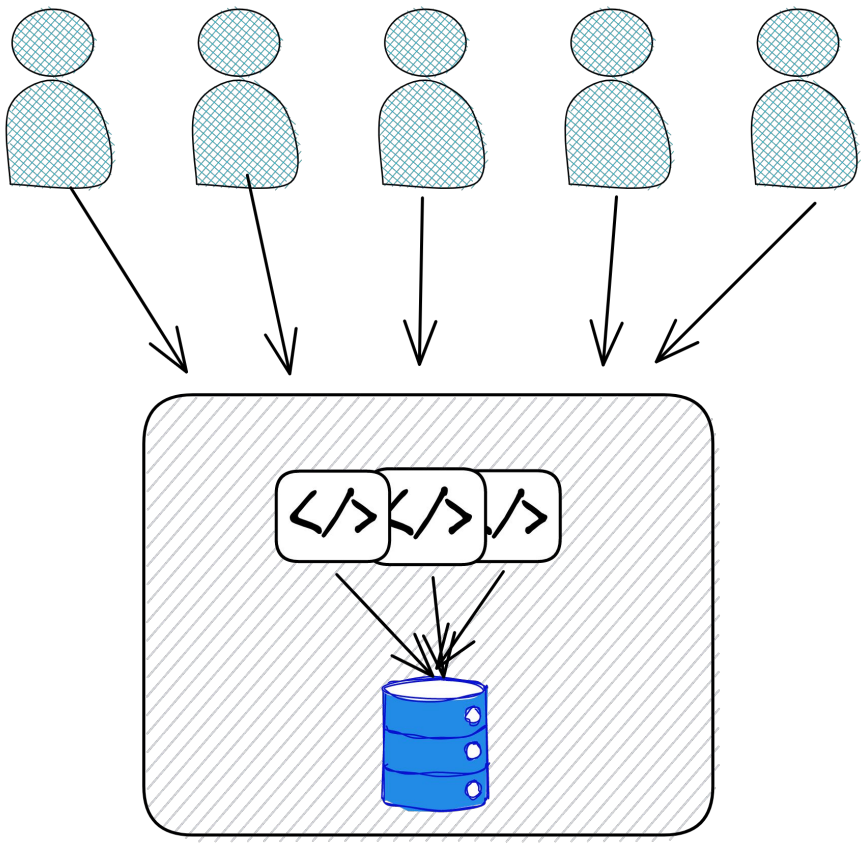
https://excalidraw.com/

Monolith

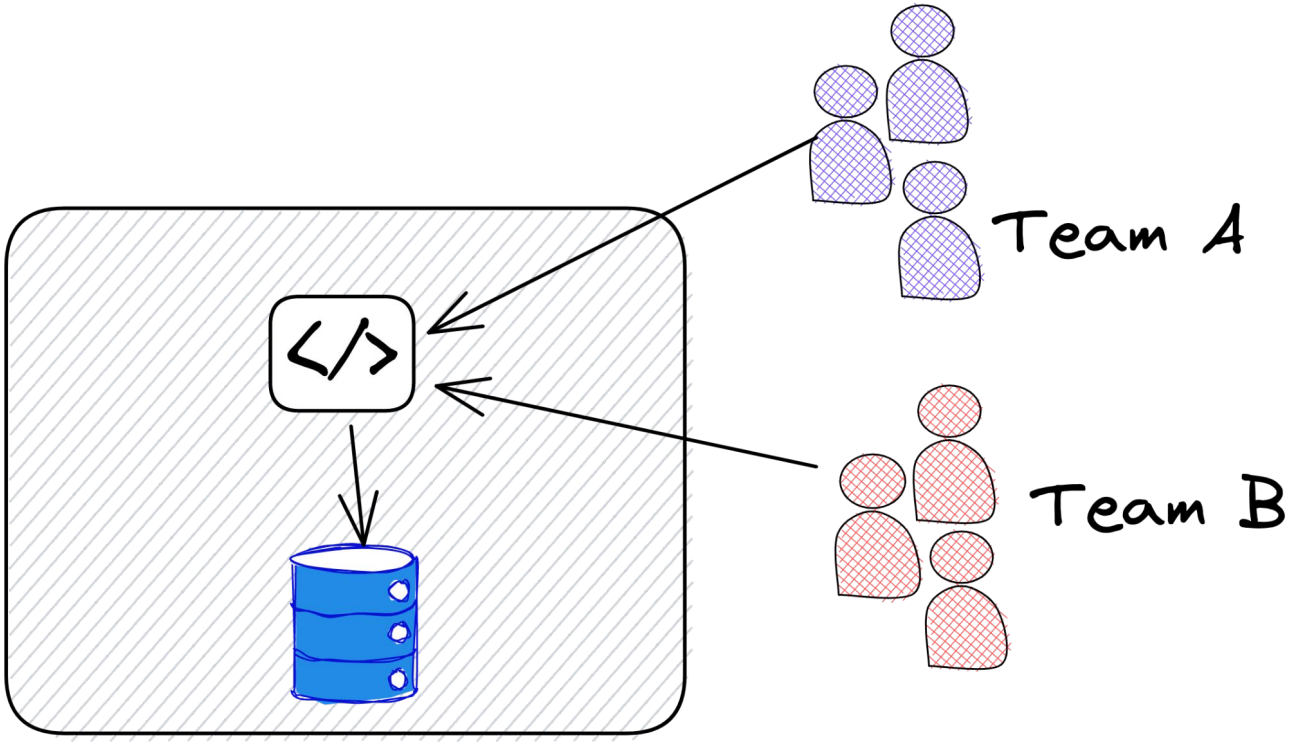




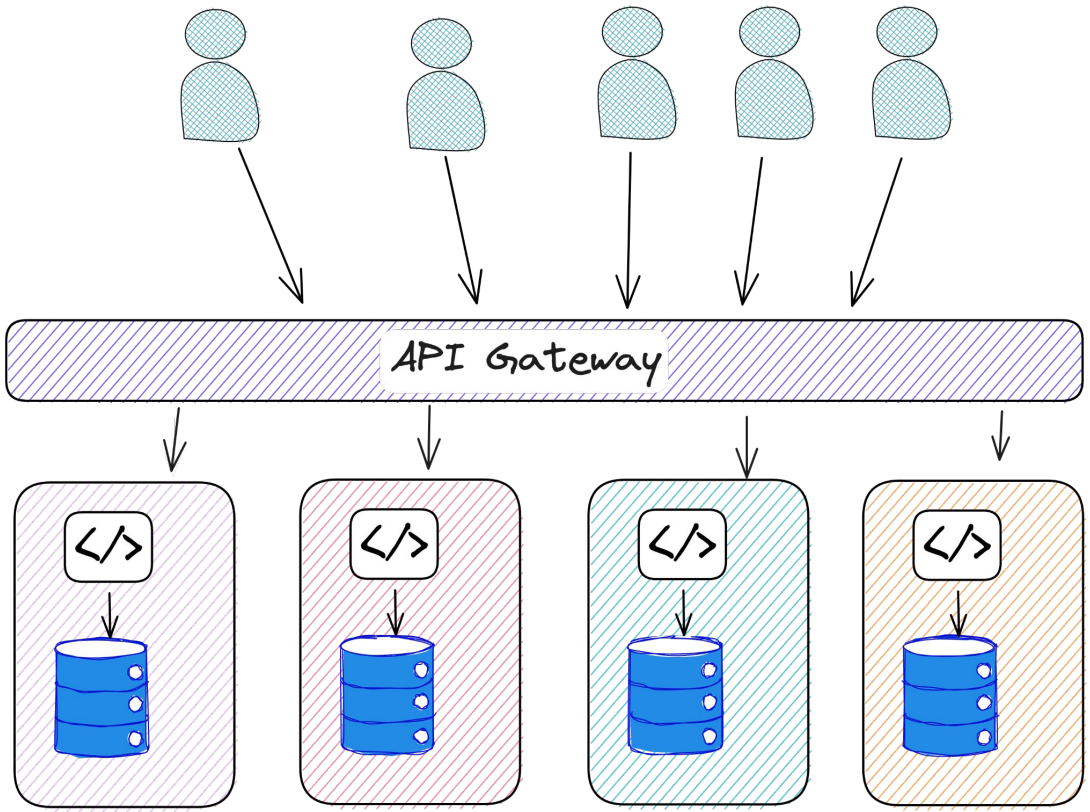
Monolith



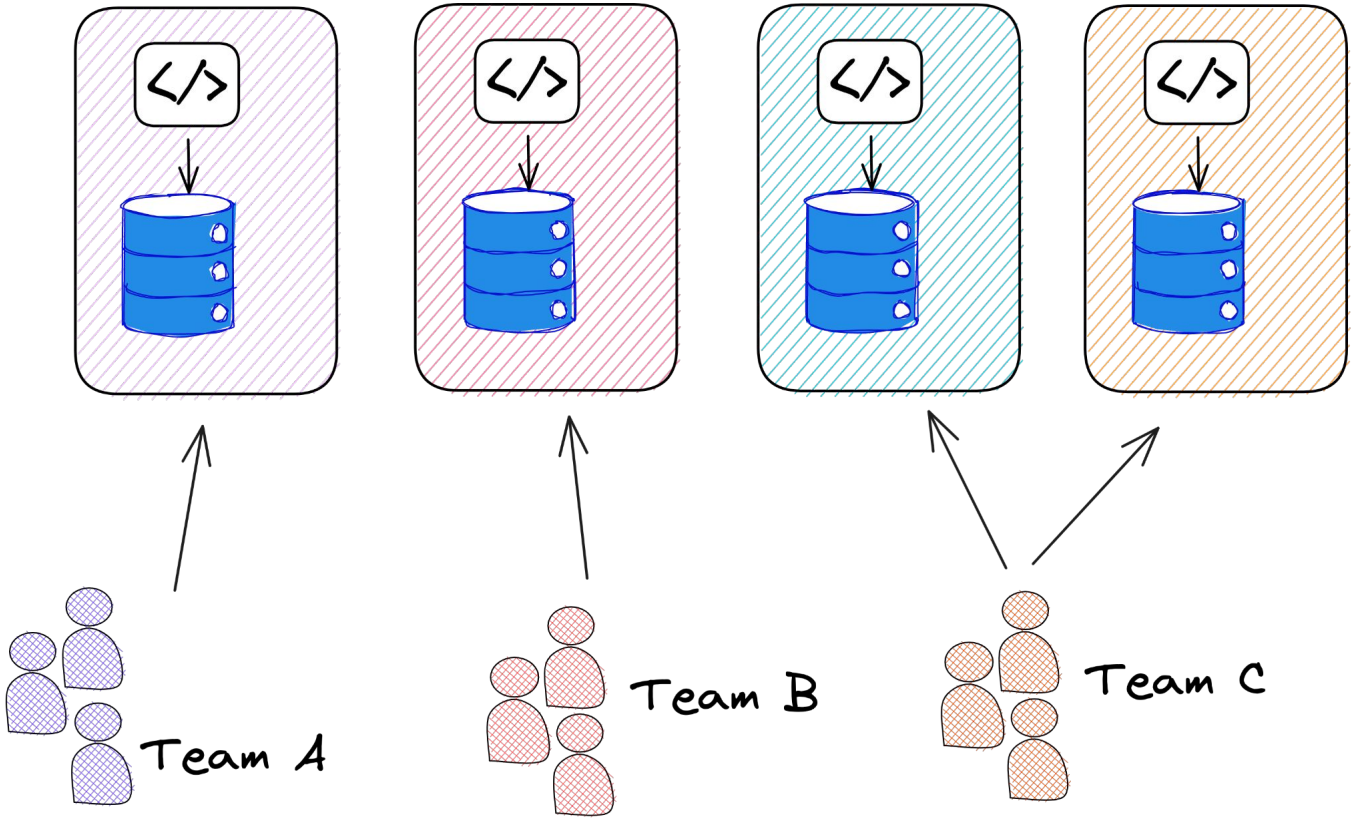
Monolith



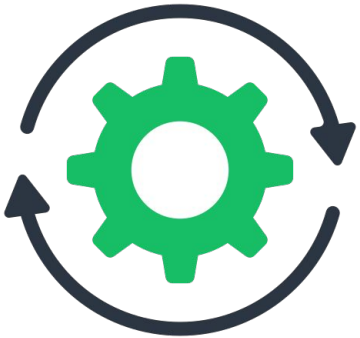
Microservice



Microservice



Prerequisites to Playing



CI/CD

High volume data

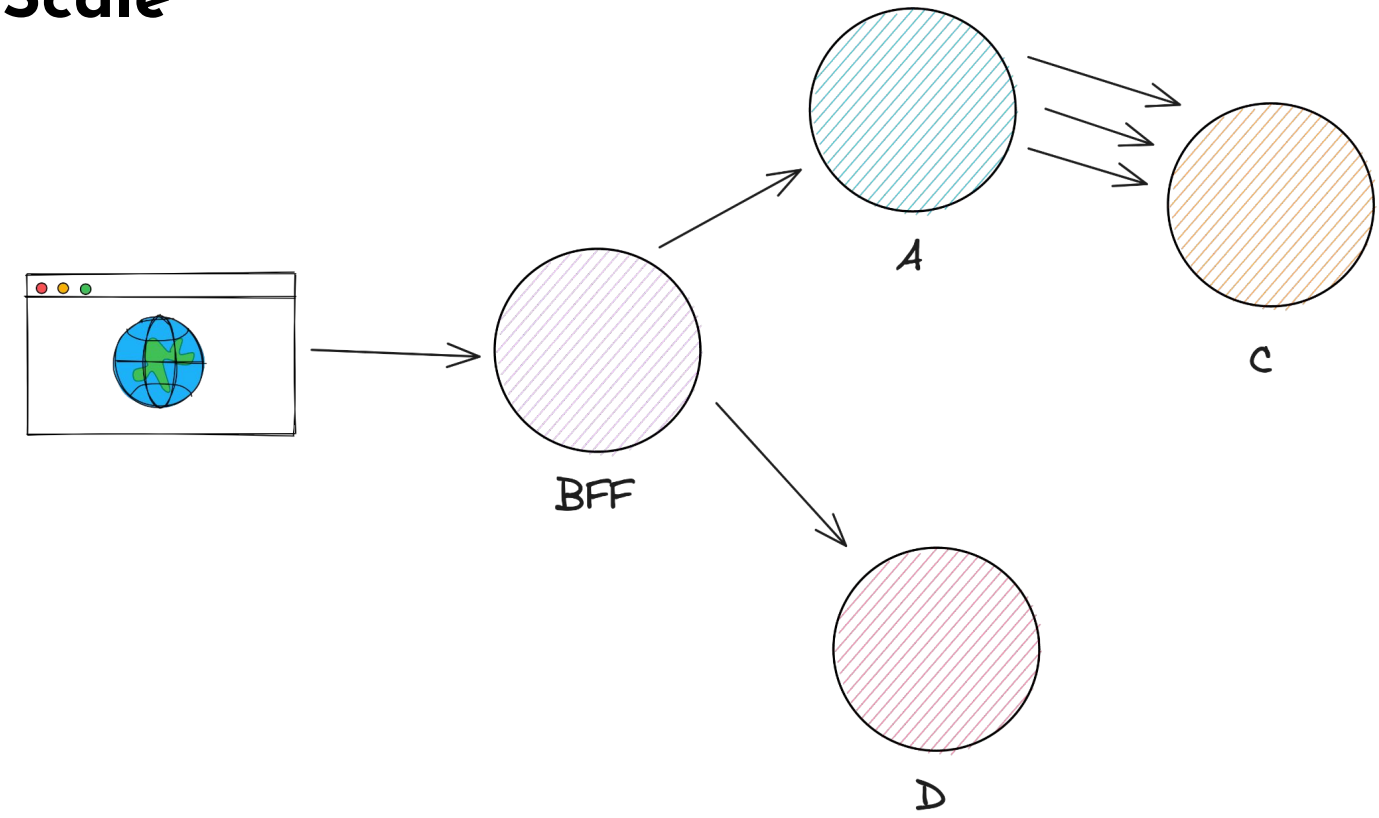
Not Green Field

Is it a Distributed Monolith?

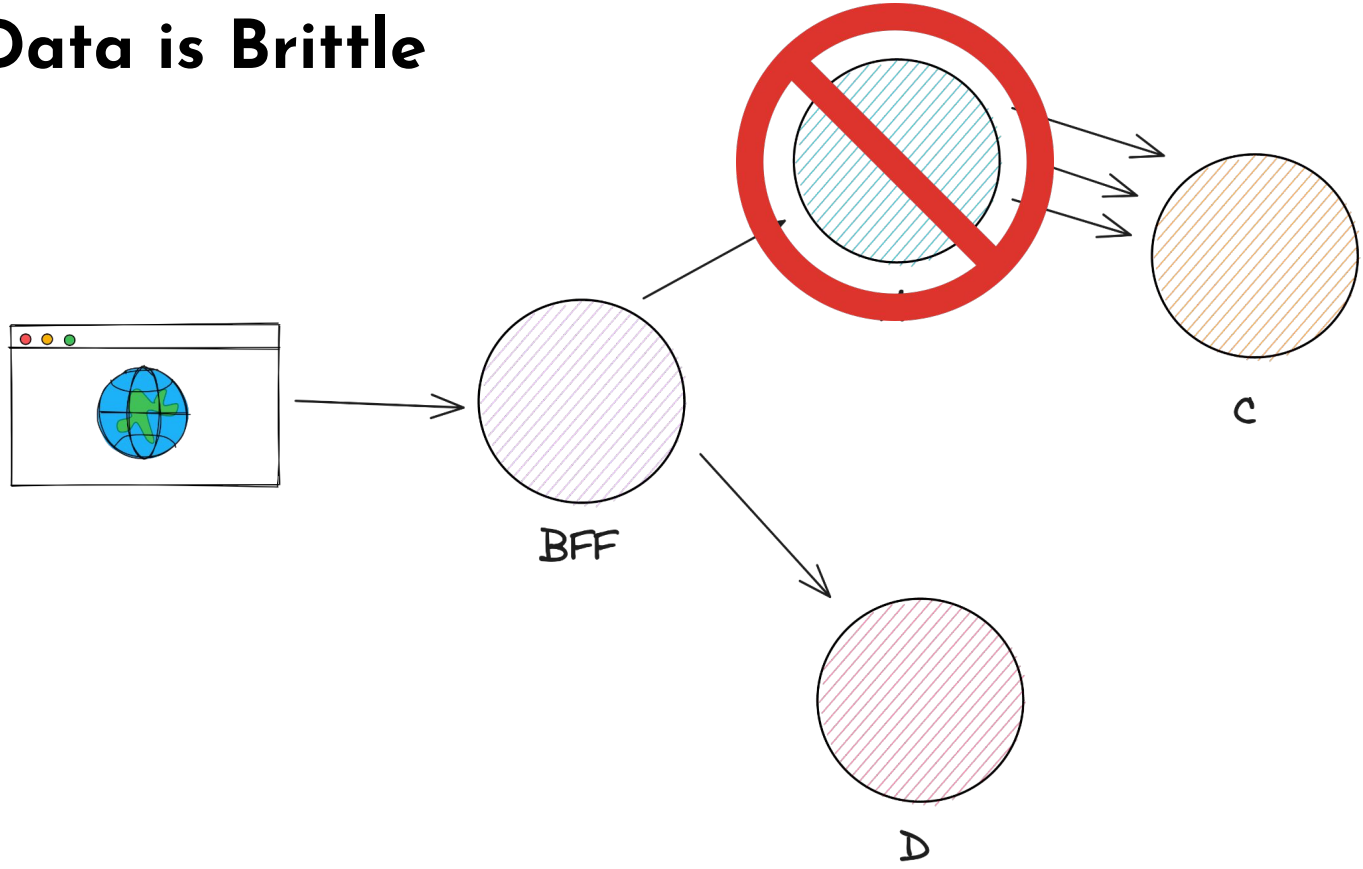
Today's Focus

- Assuming micro-services are always better
- Shared data store or models
- Microservices that are too big
- Microservice that are too small
- Starting from scratch
- Coupling through cross-cutting concerns
- **Use of synchronous communication**
- Breaking Changes to Event Contracts
- Not Automating Build and Release
- Unencapsulated Services

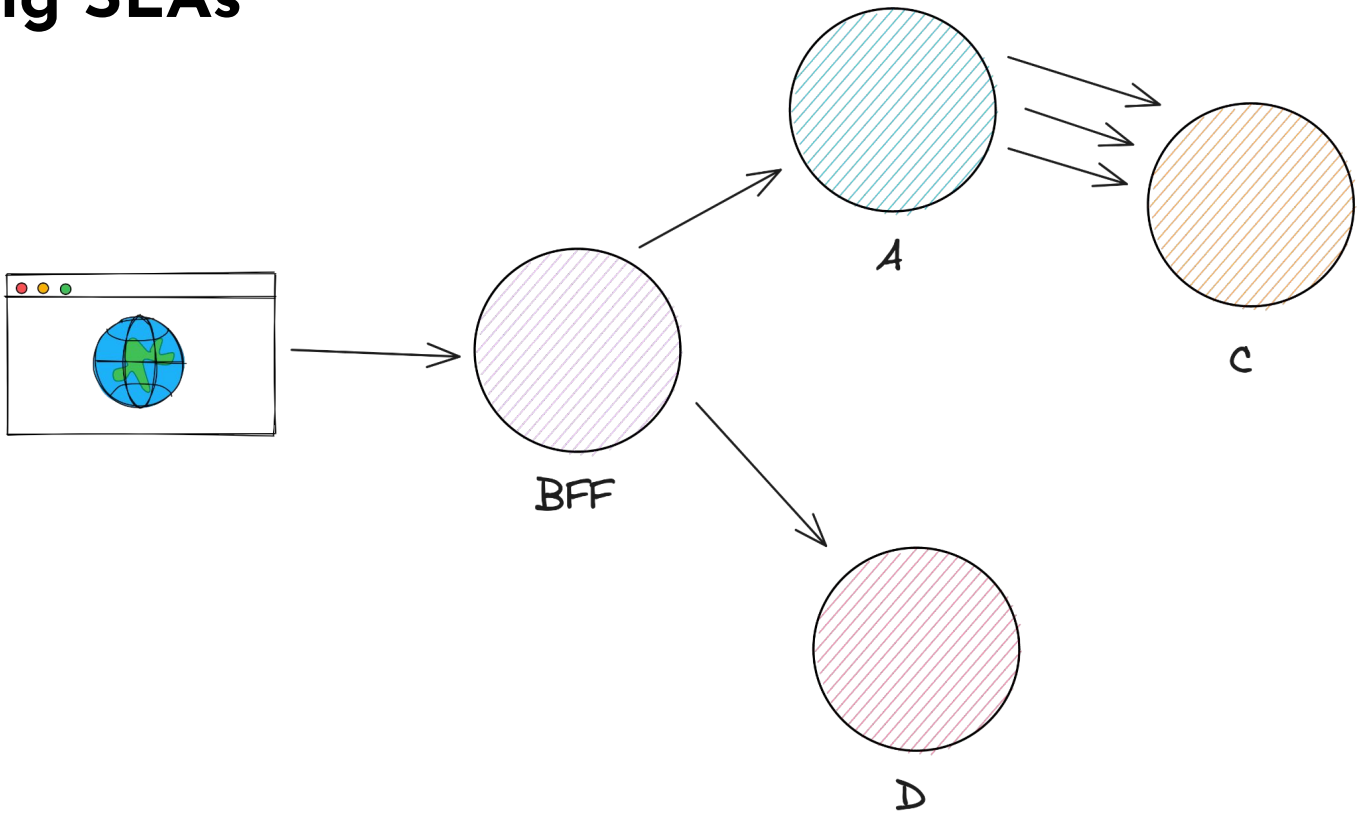
Fails to Scale



Pulling Data is Brittle

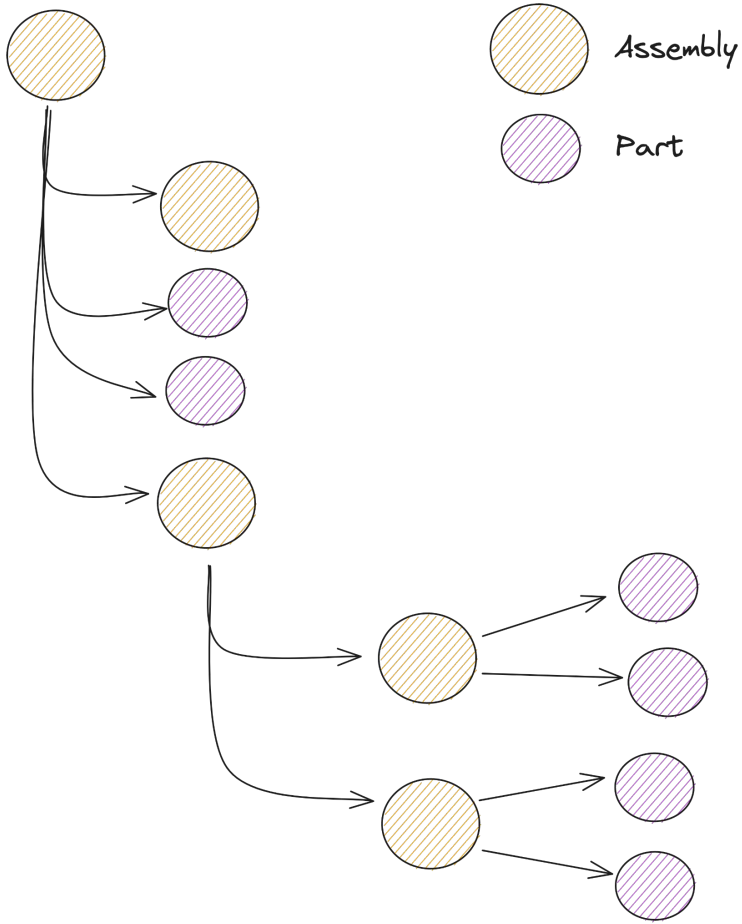


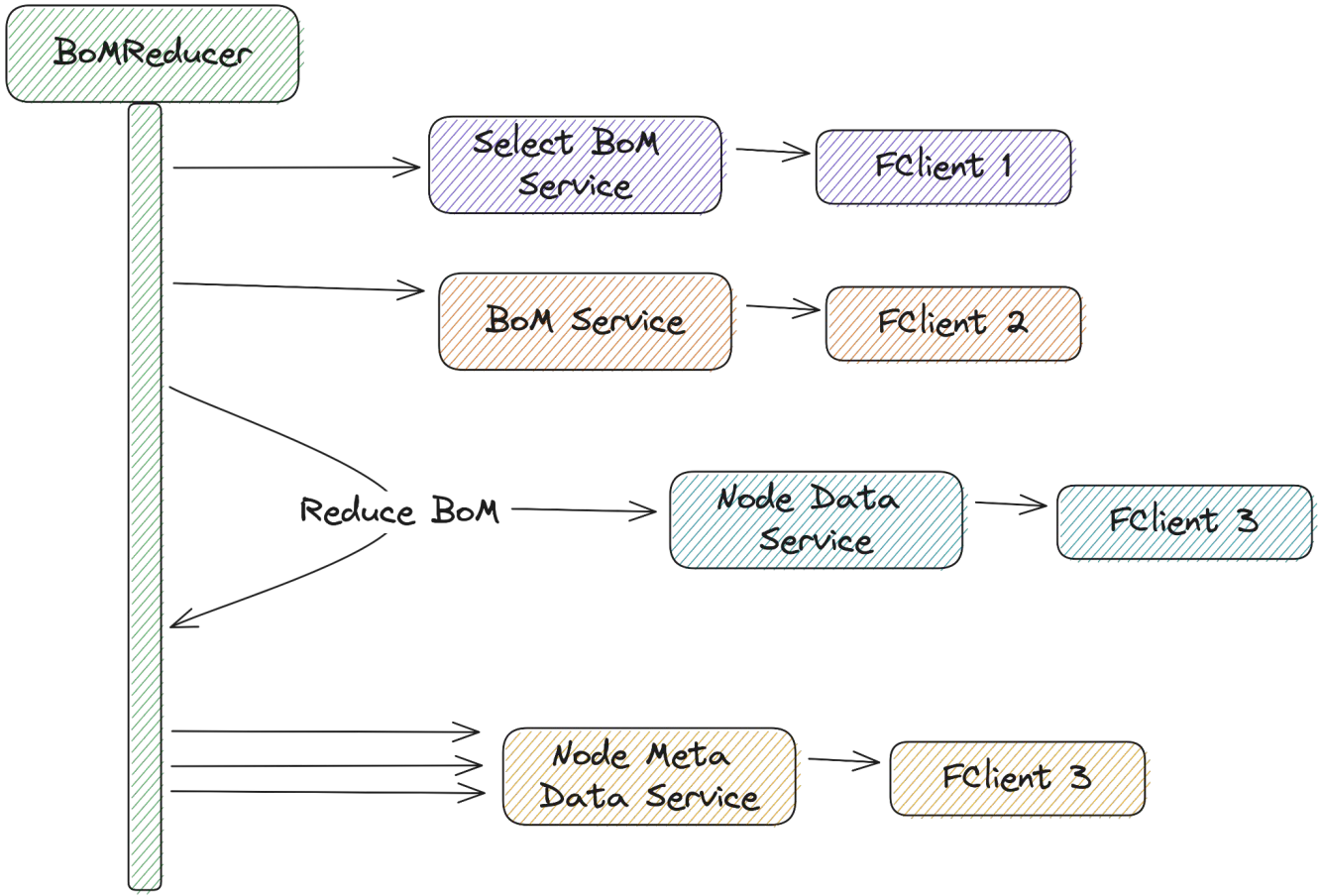
Cascading SLAs

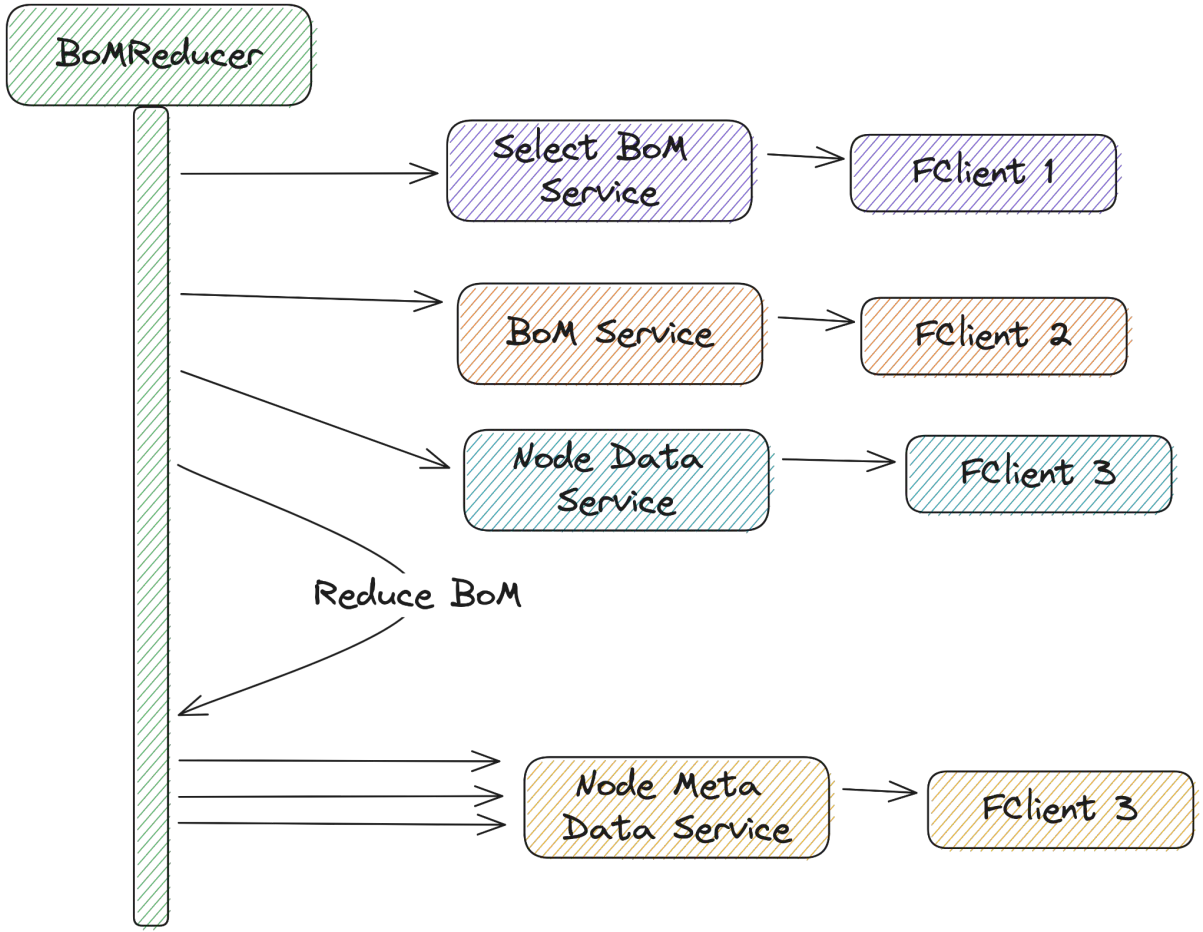


Build seams into your apps

Bill Of Materials




BoMReducer
Select BoM Service
FClient 1
BoM Service
FClient 2
Reduce BoM
Node Data Service
FClient 3
Node Meta Data Service
FClient 3


BoMReducer
Select BoM Service
FClient 1
BoM Service
FClient 2
Node Data Service
FClient 3
Reduce BoM
Node Meta Data Service
FClient 3

Plan for failure

HYSTRIX
END YOUR APP

Resilience4J Patterns

- Retry - Do it more than once
- CircuitBreaker - Fail Fast
- RateLimiter - Limit calls per second
- TimeLimiter - Limit the time waiting
- Bulkhead - Limit concurrent calls

Single Annotation

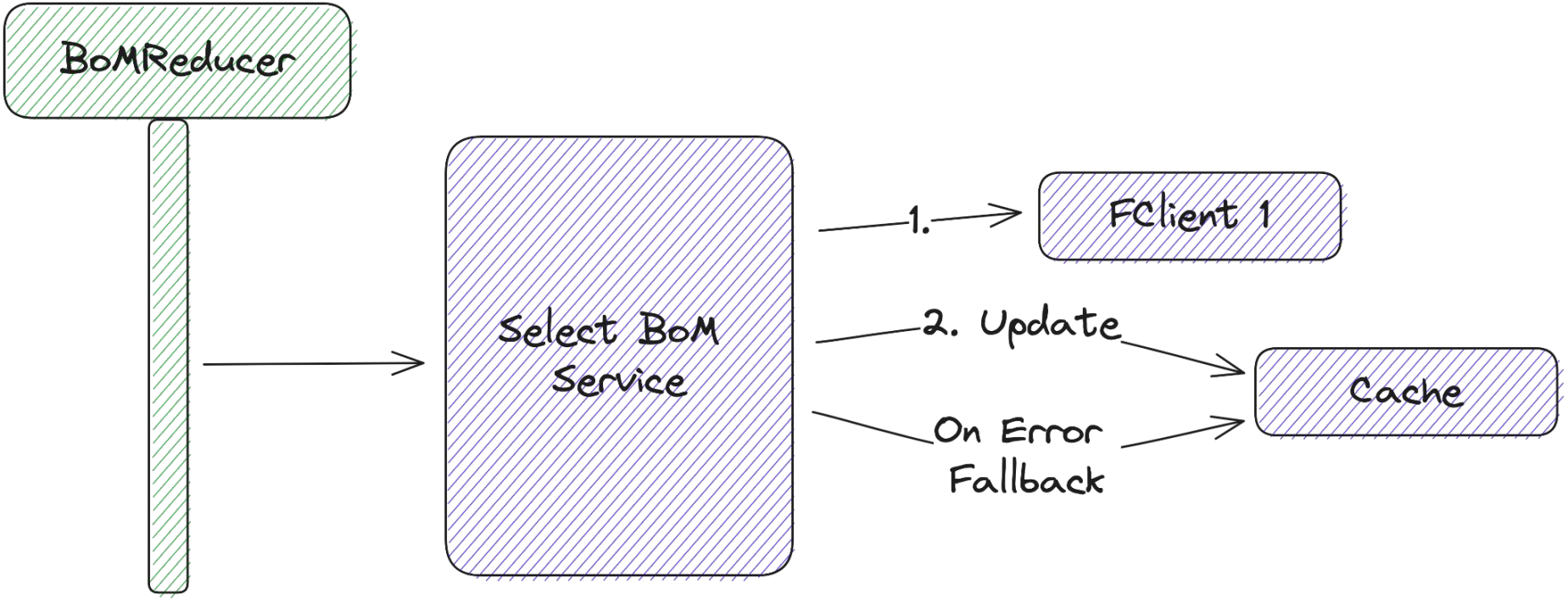
```
1.  @CircuitBreaker(name = BACKEND_A, fallbackMethod = "fallback")
2.  public String failureWithFallback() {
3.      return failure();
4.  }
5.
6.  private String fallback(HttpServerErrorException ex) {
7.      return "Recovered HttpServerErrorException: " + ex.getMessage();
8.  }
9.
10. private String fallback(Exception ex) {
11.     return "Recovered: " + ex.toString();
12. }
```

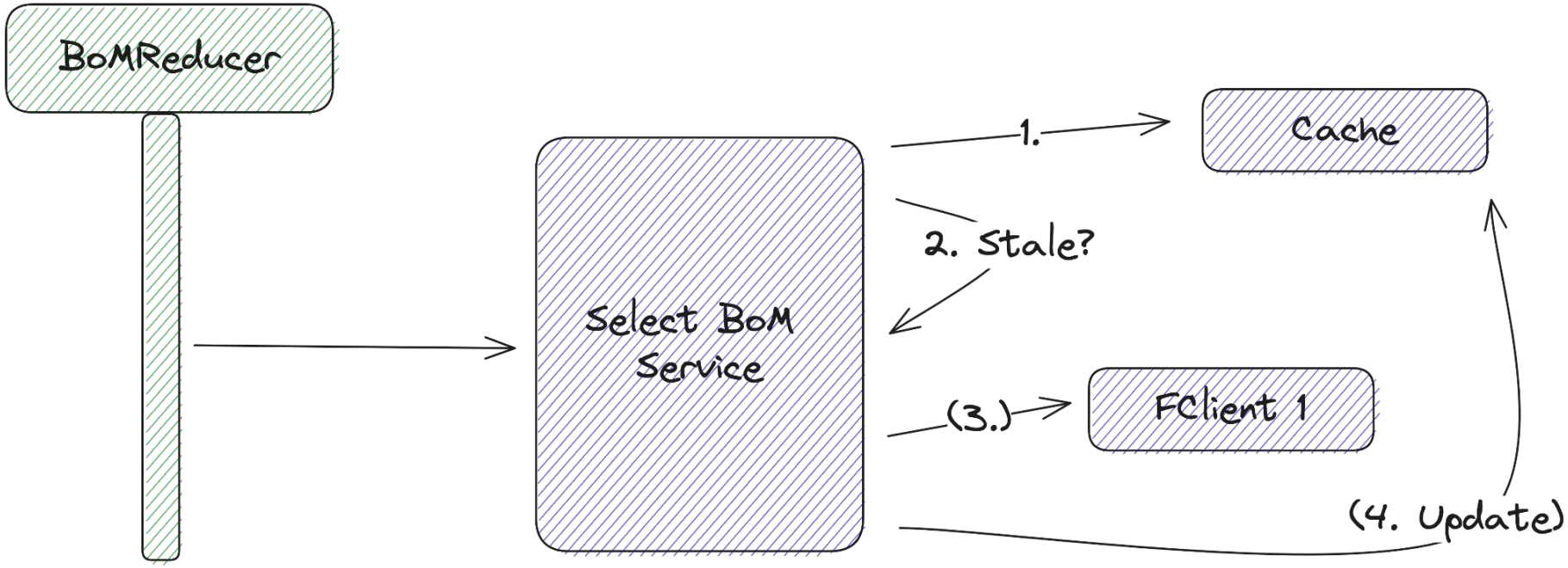
Composed Annotations

```
1.   @CircuitBreaker(name = BACKEND, fallbackMethod = "fallback")
2.   @RateLimiter(name = BACKEND)
3.   @Bulkhead(name = BACKEND, fallbackMethod = "fallback")
4.   @Retry(name = BACKEND)
5.   @TimeLimiter(name = BACKEND)
6.   public BoM method(String bomId) {
7.      return callRemoteService();
8.   }
9.
10.  private BoM fallback(String bomId, HttpServerErrorException e) {
11.        log.warn("Handled the exception when the CircuitBreaker is open")
12.     return new EmptyBoM();
13.  }
14.
15.  private String fallback(String bomId, BulkheadFullException e) {
16.        log.warn("Handled the exception when the Bulkhead is full")
17.     return new LazyBoM();
18.  }
```

Improve Data Handling

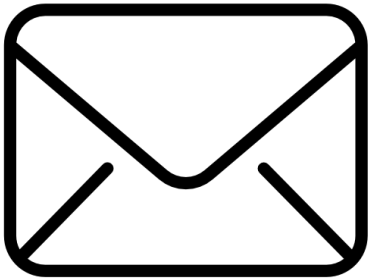
Caching?


BoMReducer
Select BoM Service
1.
FClient 1
2. Update
Cache
On Error Fallback


BoMReducer
Select BoM Service
1.
Cache
2. Stale?
(3.)
FClient 1
(4. Update)

So what else?

Events



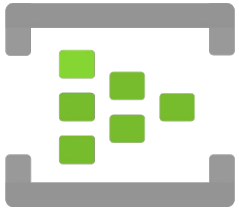
Messages

Events

- Sent via bus
- Past state change
- No guarantees on future actions

Examples

`UserCreated`
`UserModified`



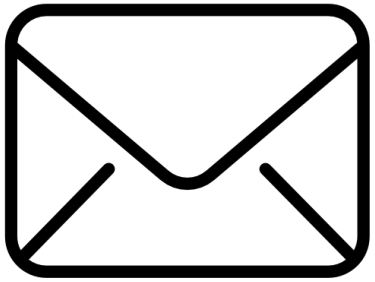
Events != Event Sourcing

Messages

- Sent via bus
- Communicate instructions

Examples

```
SendEmail
AssignRole
```



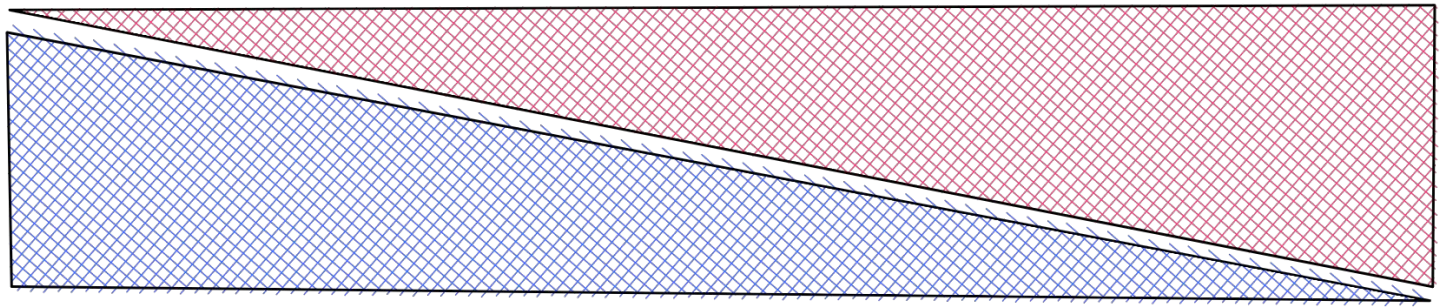
Scale to (almost) Zero with

K What?

- Kubernetes Event-driven Autoscaling
- Lightweight

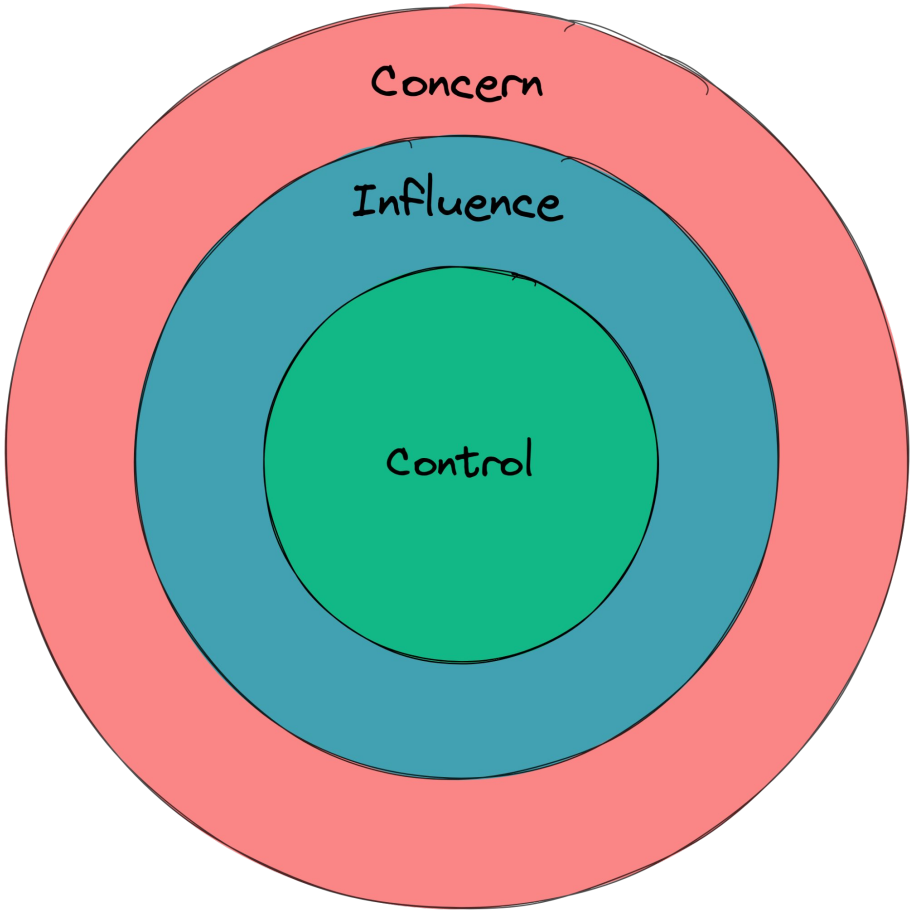
- Full re-use
- Easy Testing

Immediate Consistency

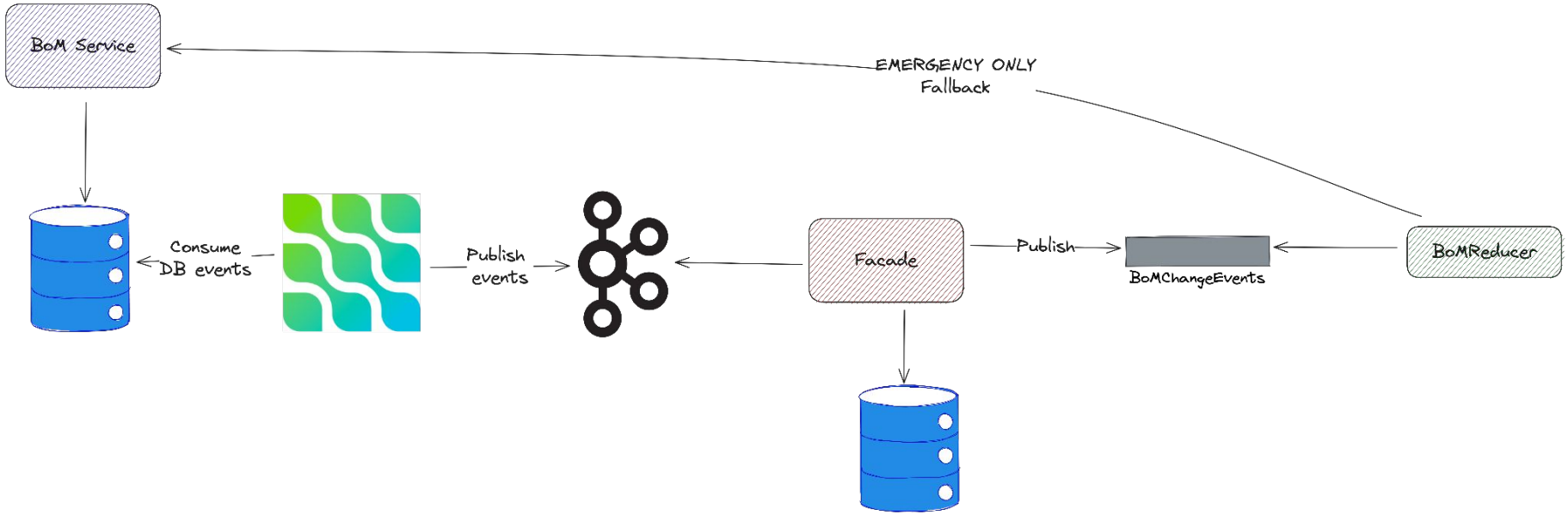


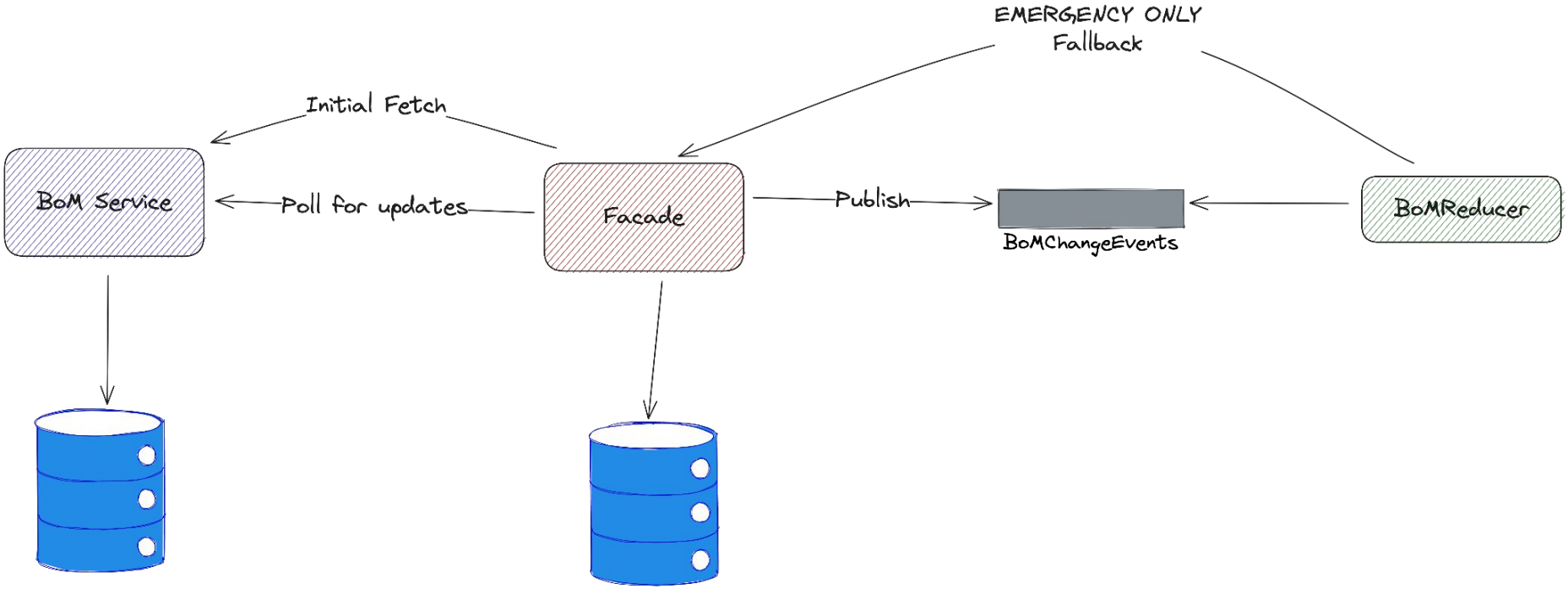
Availability

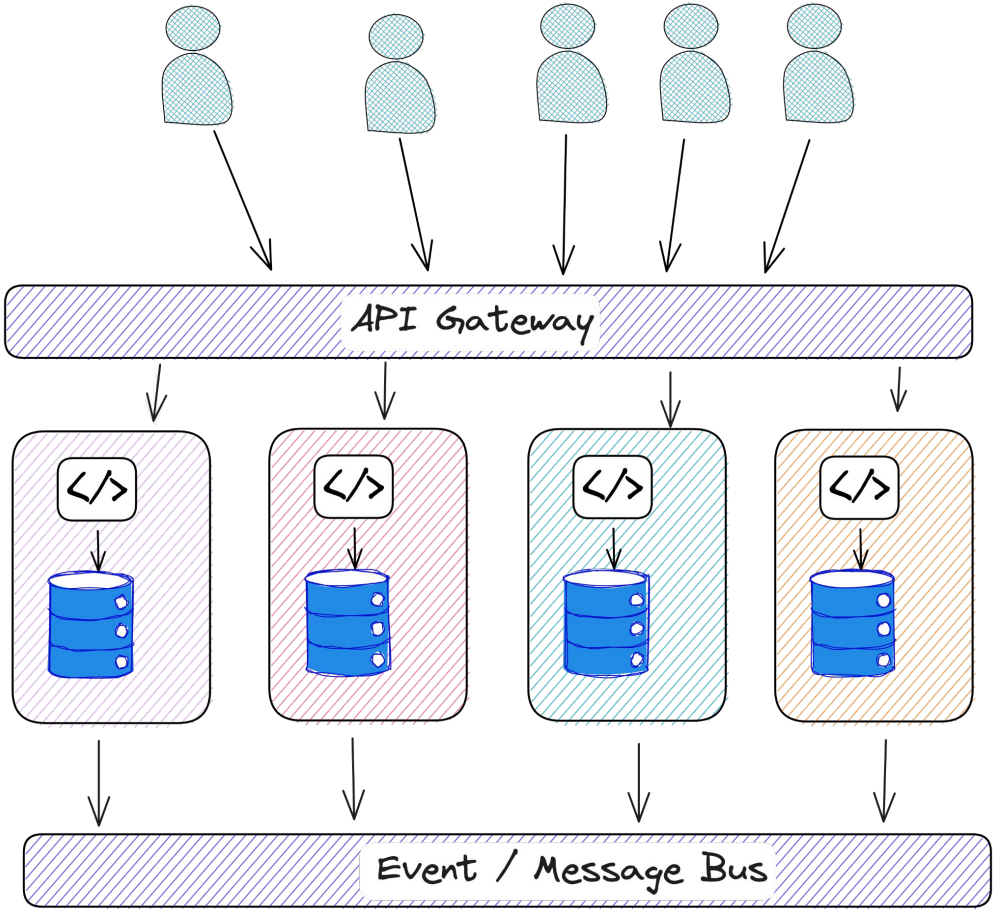
How do we go there?

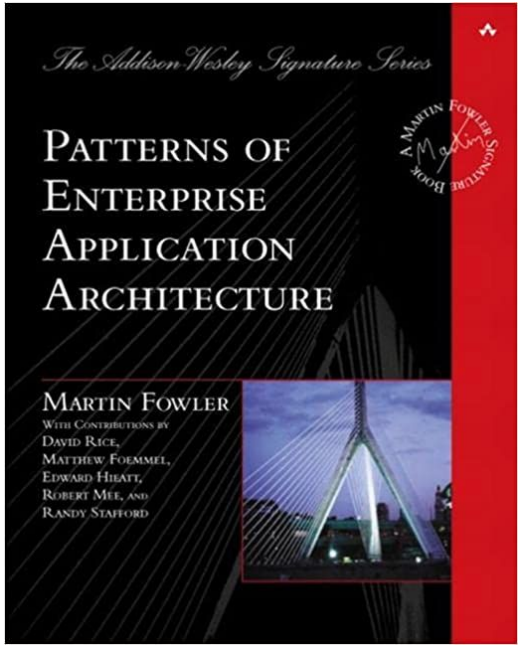

Concern
Influence
Control

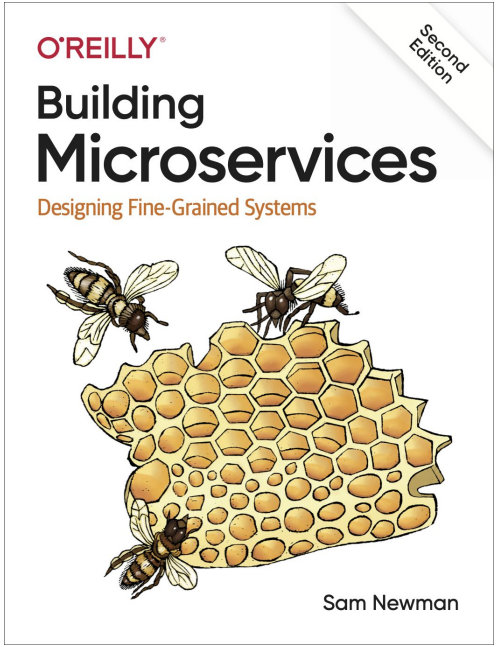
Minimise effort

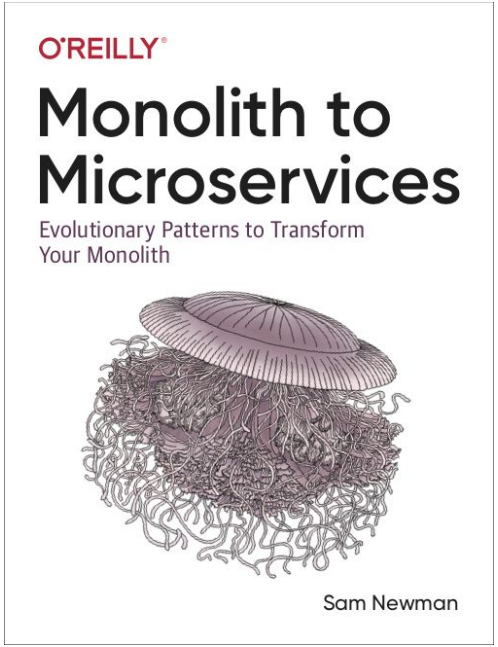

BoM Service
EMERGENCY ONLY
Fallback
Consume
DB events
Publish
events
Facade
Publish
BoMChangeEvents
BoMReducer


EMERGENCY ONLY
Fallback
Initial Fetch
BoM Service
Poll for updates
Facade
Publish
BoMChangeEvents
BoMReducer


API Gateway
Event / Message Bus


The Addison-Wesley Signature Series
Patterns of Enterprise Application Architecture
Martin Fowler
With Contributions by
David Rice,
Matthew Foemmel,
Edward Hieatt,
Robert Mee, and
Randy Stafford

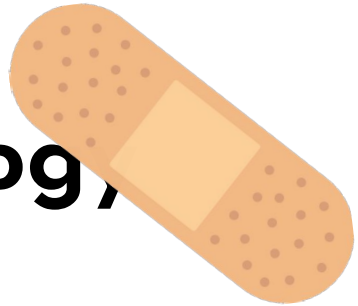

O'REILLY
Second Edition
Building Microservices
Designing Fine-Grained Systems
Sam Newman

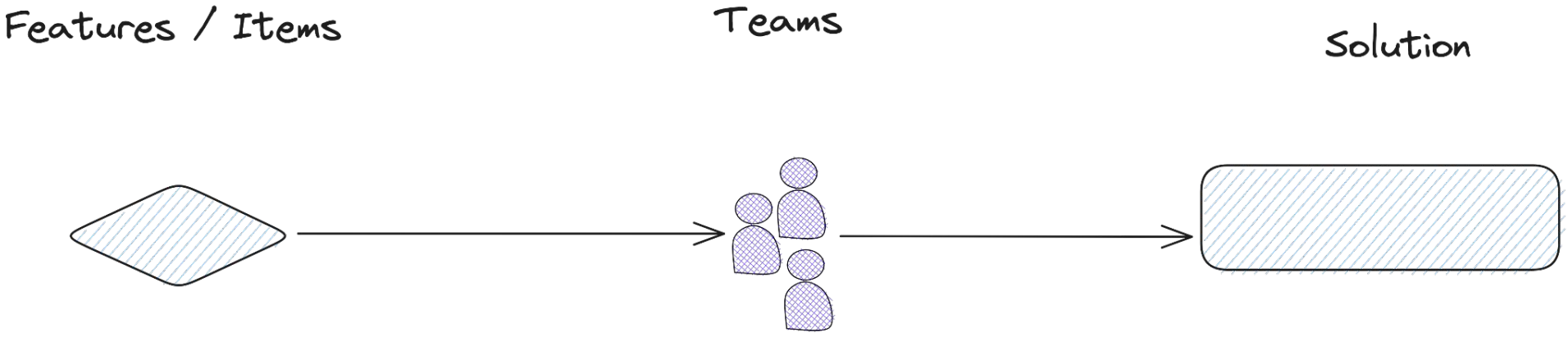

O'REILLY
Monolith to Microservices
Evolutionary Patterns to Transform Your Monolith
Sam Newman

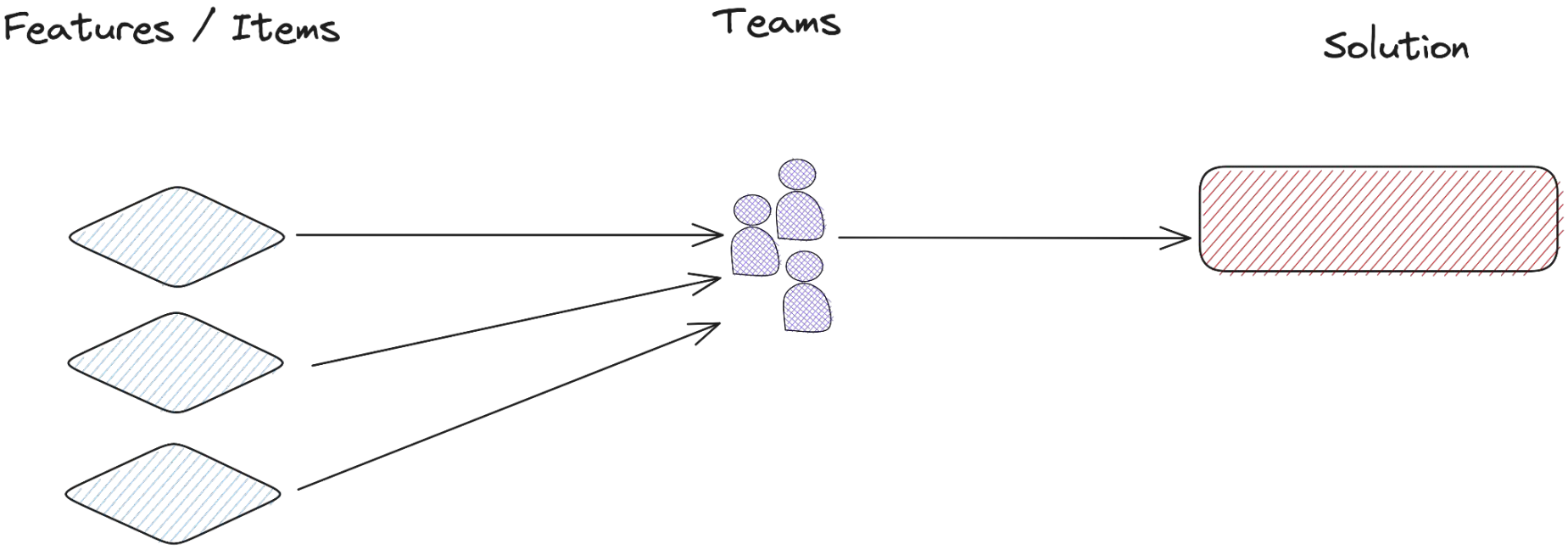
Websites

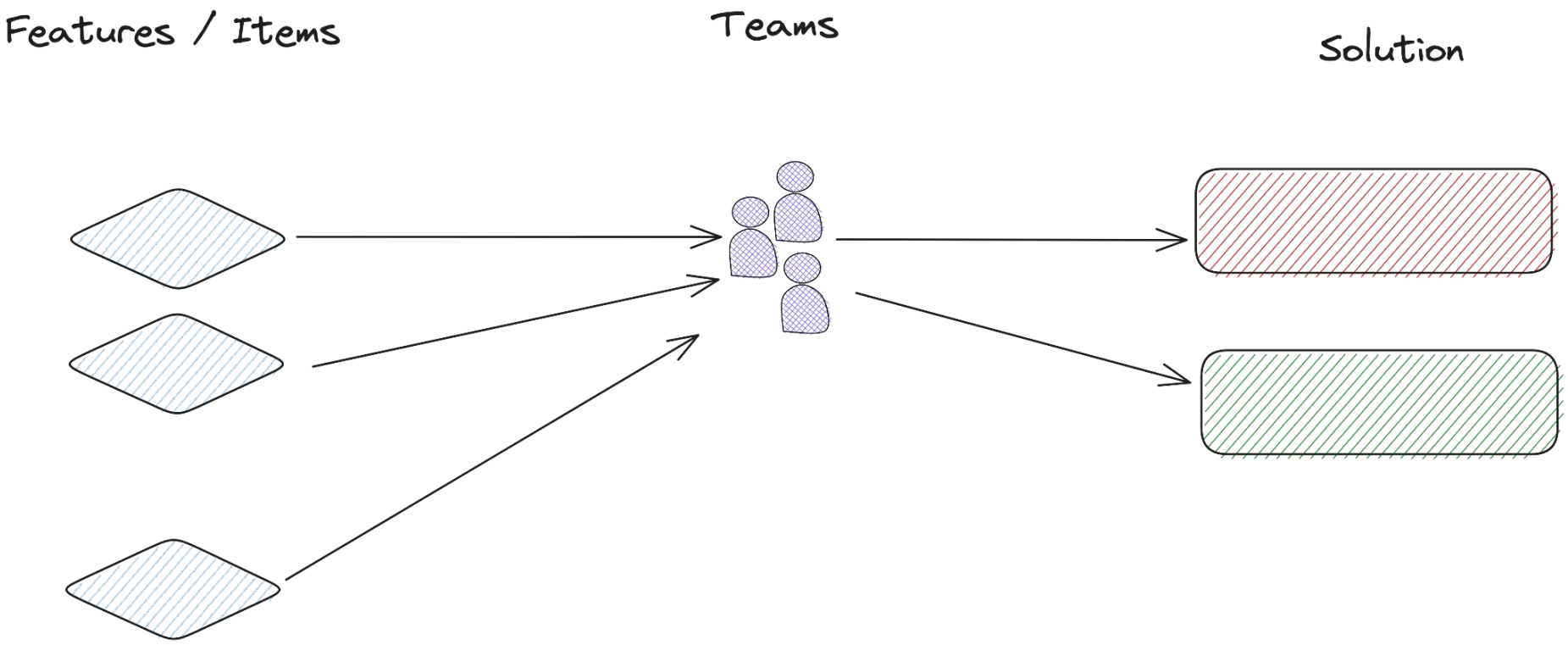
- https://martinfowler.com/eaaCatalog/
- https://www.enterpriseintegrationpatterns.com/
- https://martinfowler.com/articles/patterns-of-distributed-systems/

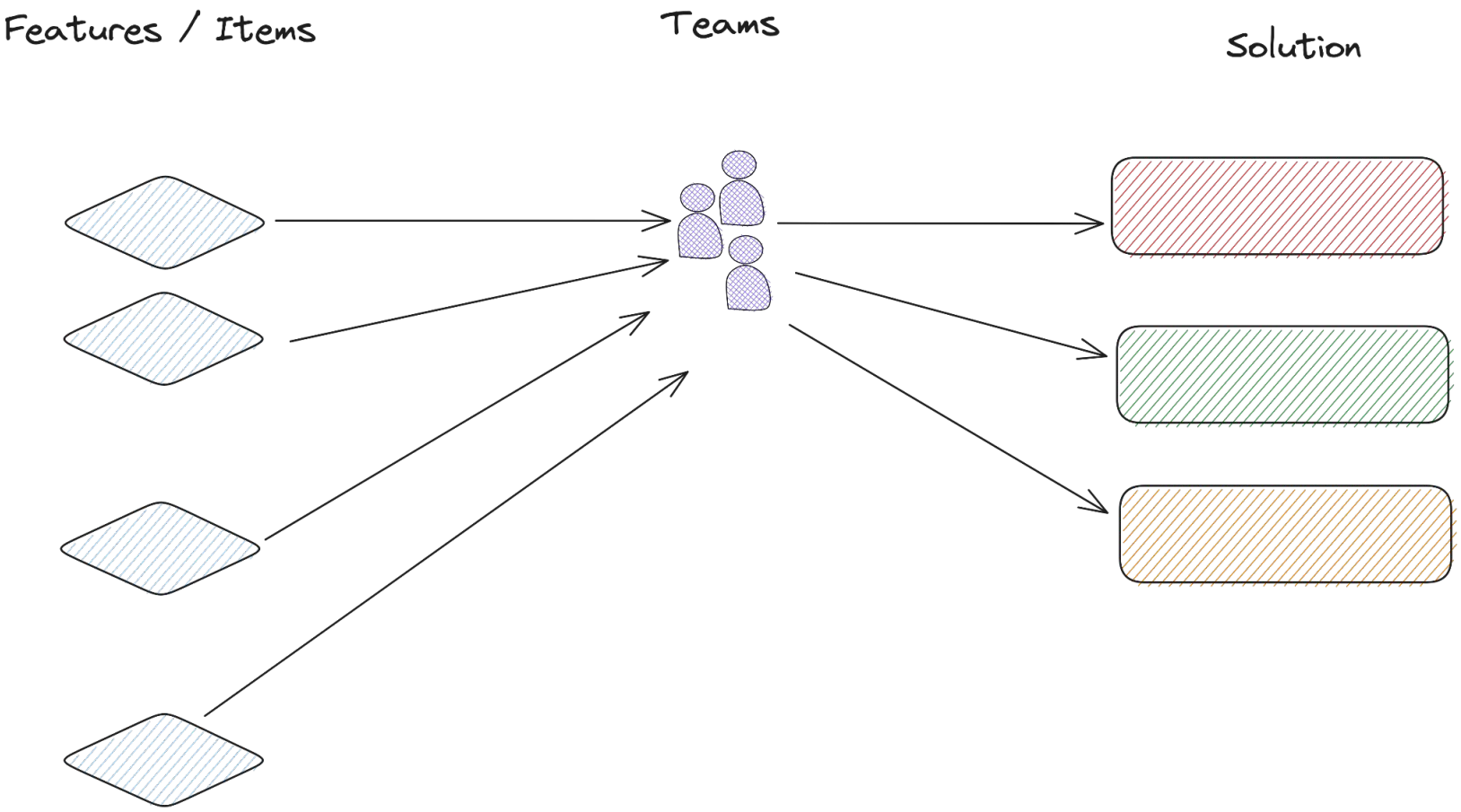
Technology

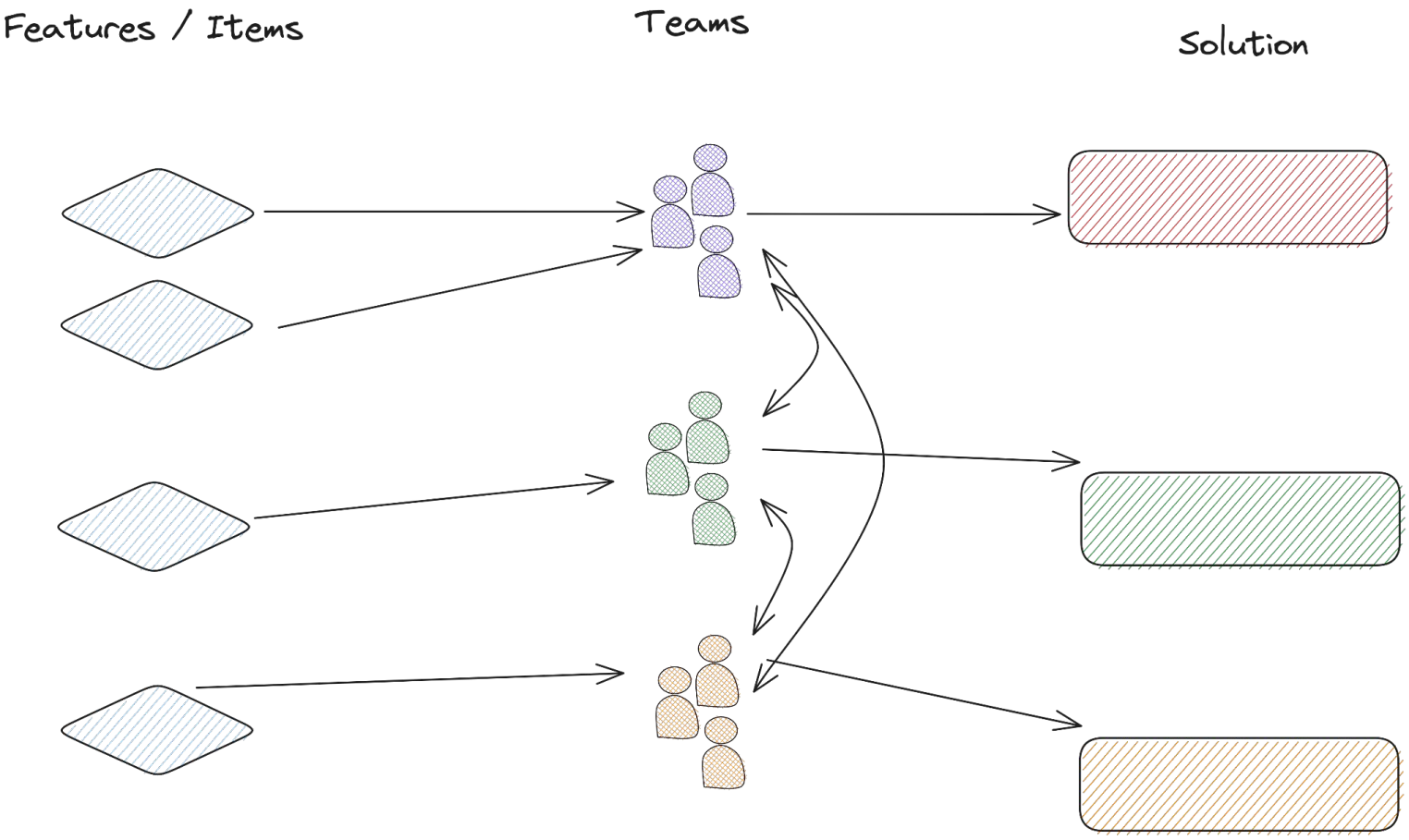


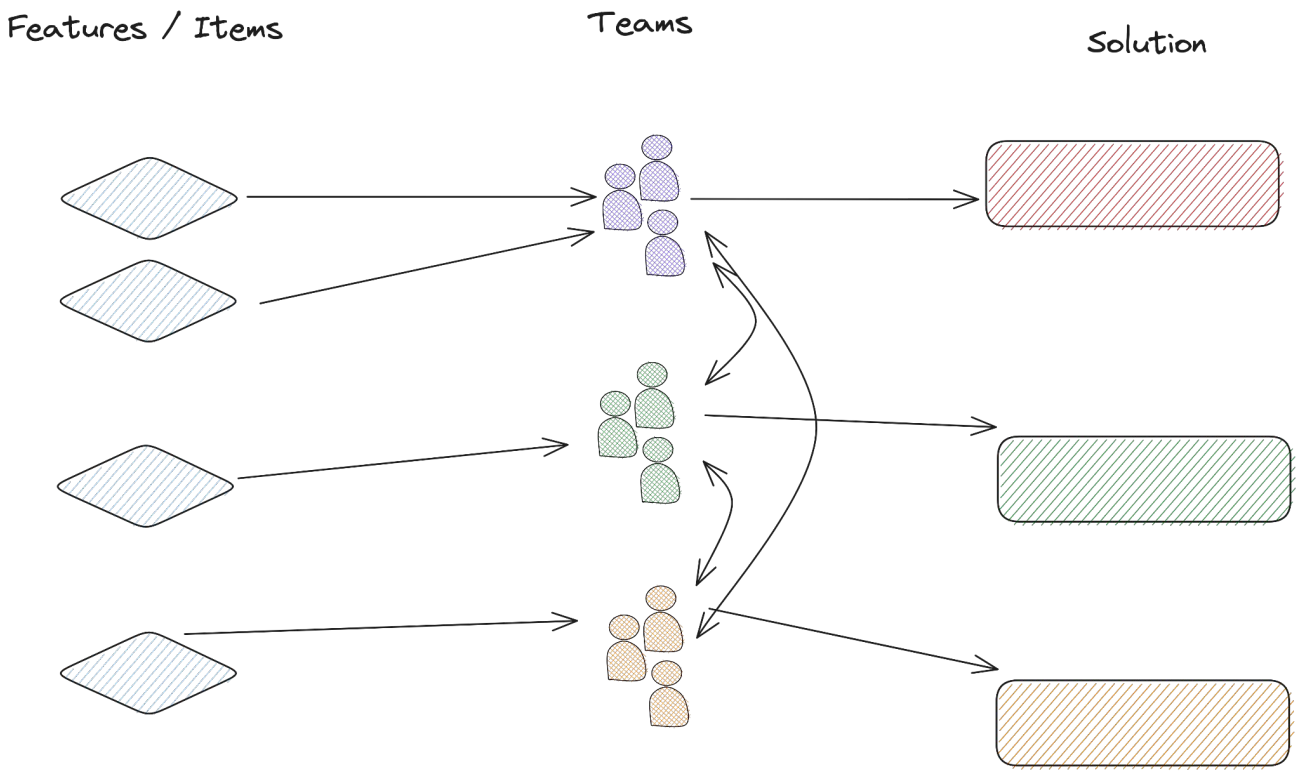

Features / Items
Teams
Solution


Features / Items
Teams
Solution


Features / Items
Teams
Solution

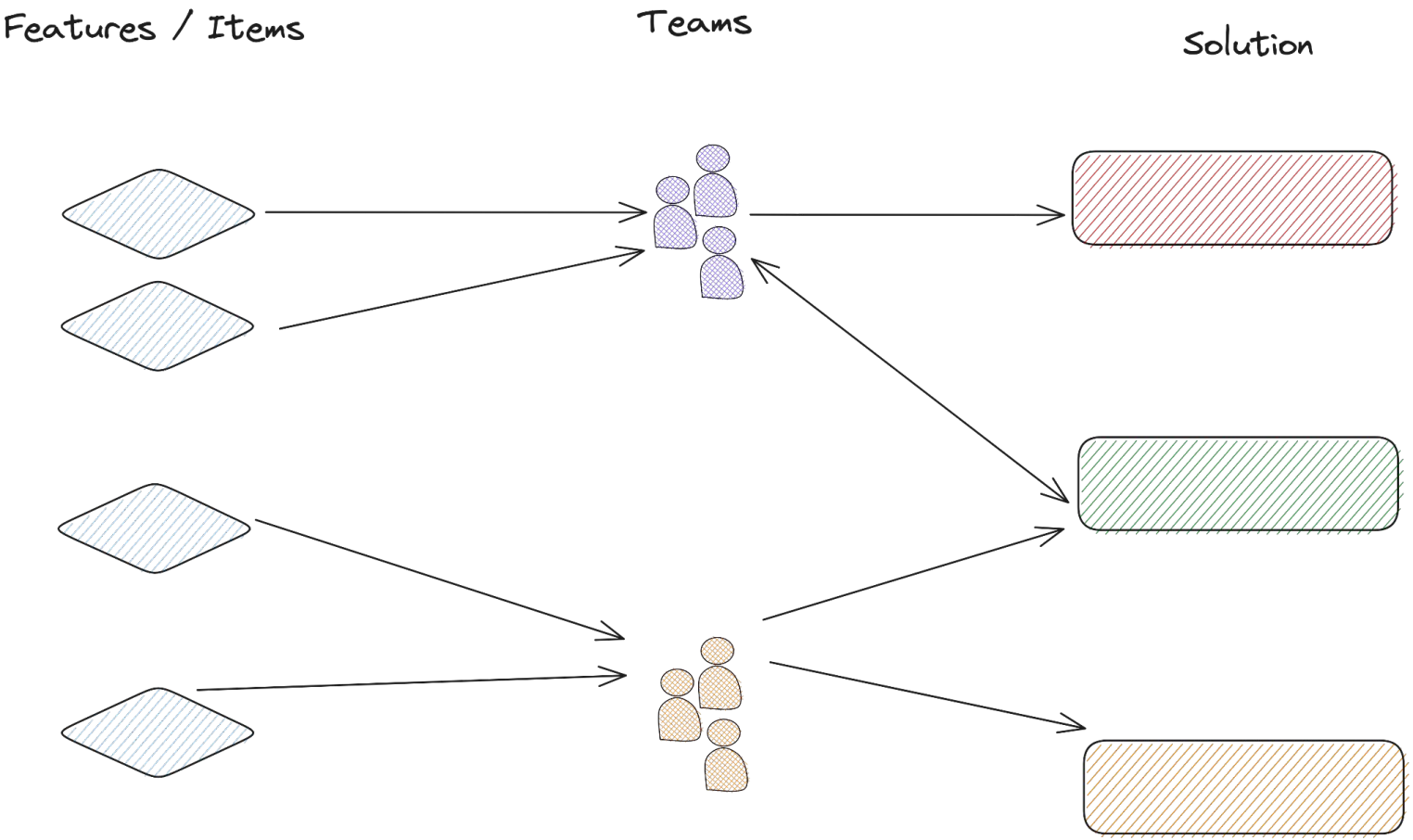

Features / Items
Teams
Solution

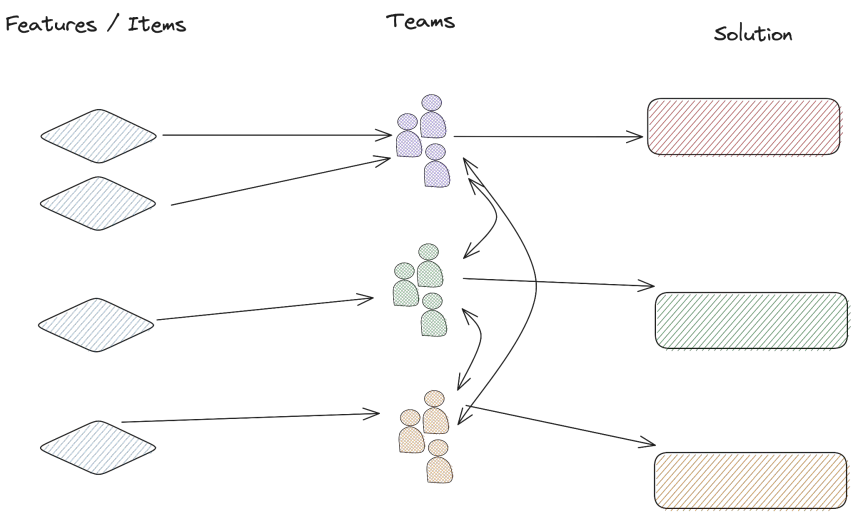

Features / Items
Teams
Solution

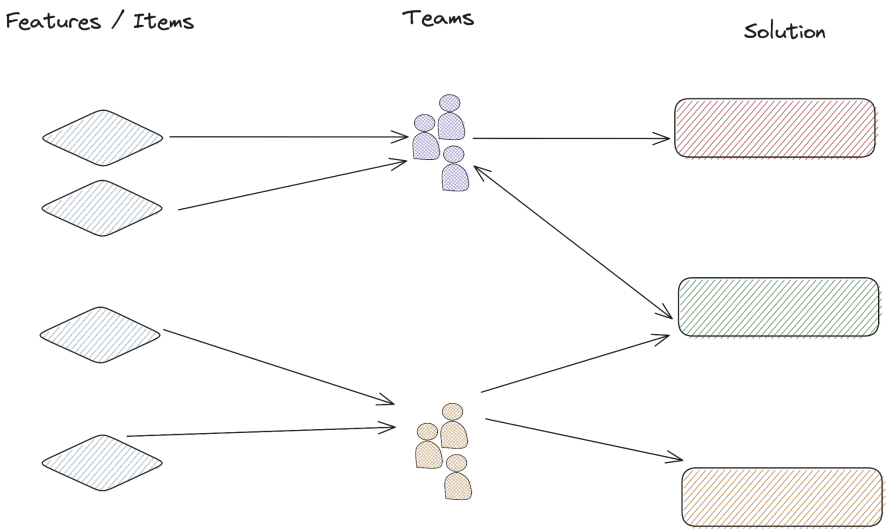


SAFe®
PROVIDED BY SCALED AGILE

LeSS


Features / Items
Teams
Solution


Features / Items
Teams
Solution


Features / Items
Teams
Solution

As always, it Depends!

THANK YOU!
Q&A

Philipp Gerhard
@pgerhard92