Event Sourcing

Workshop

Agenda

- What is Event?
- What is Event Sourcing?
- Coding Exercise
- Recap
- Summary

Receive an email

Issue an invoice

Reverse an invoice

A payment

An HTTP POST

A Click

What is an Event?

Type a letter

A thing that happens

Move mouse

An invoice sent

Stand up

A new purchase

A payment settled

An HTTP GET

Change name

Types of events

Interrelated

Technical Events

Happens not because of a business reason

Page Loaded

A Click

An HTTP Post

An HTTP Get

An SNS Notification

Business Events

Happens because of a business reason

A New Purchase

Issue an Invoice

Send an Invoice

Settle a Payment

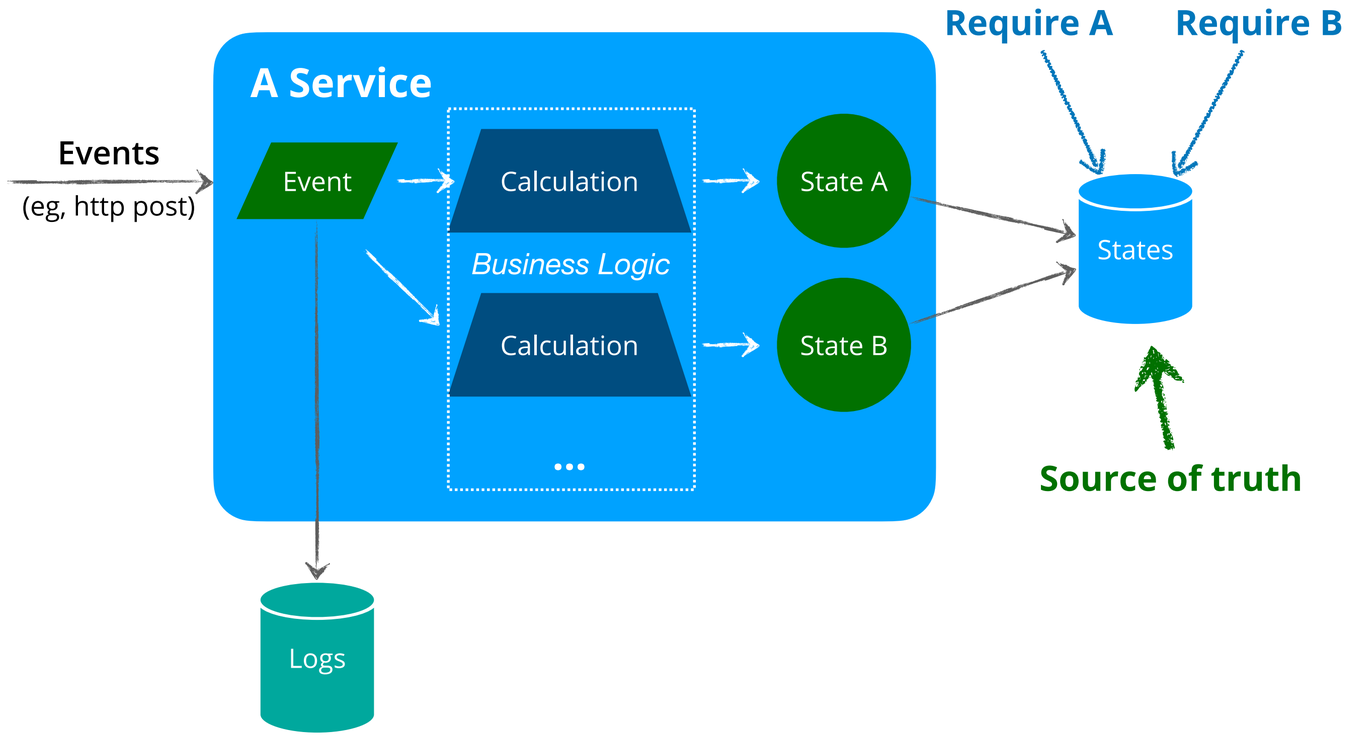
An user sign up

Event Sourcing

With the Business Events

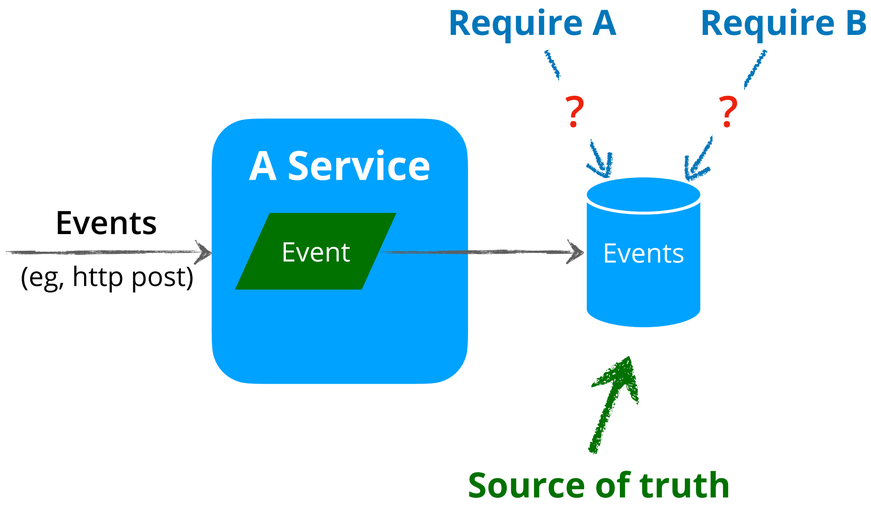
The difference

Traditional DB Design



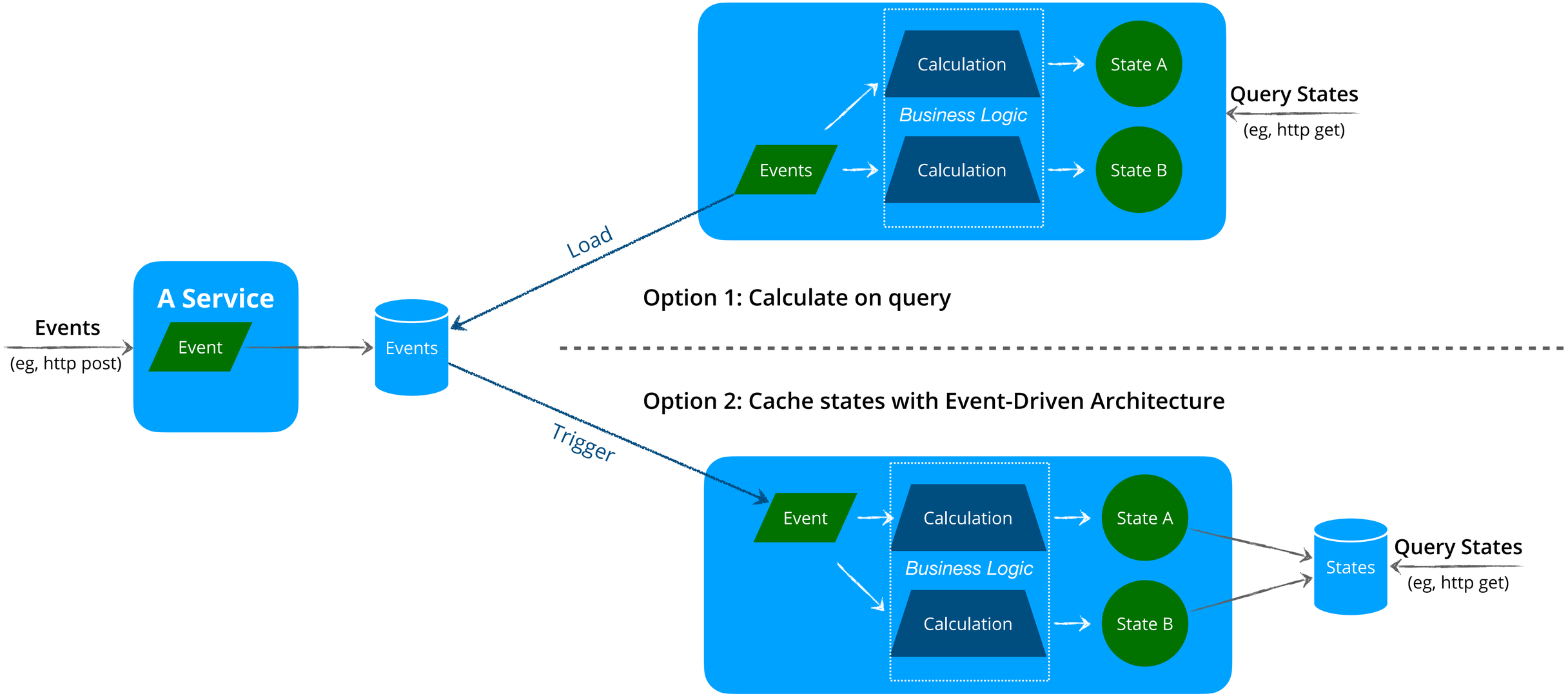
State-Oriented

Event Sourcing



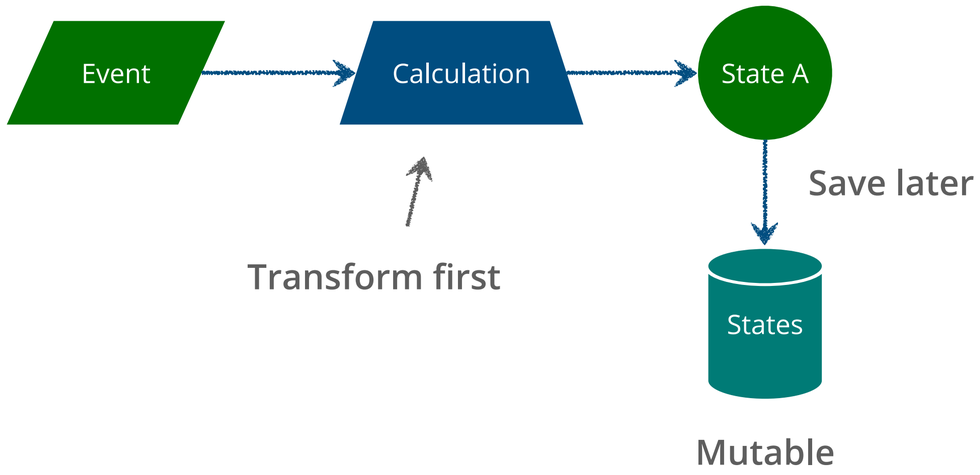
Event-Oriented

Query States

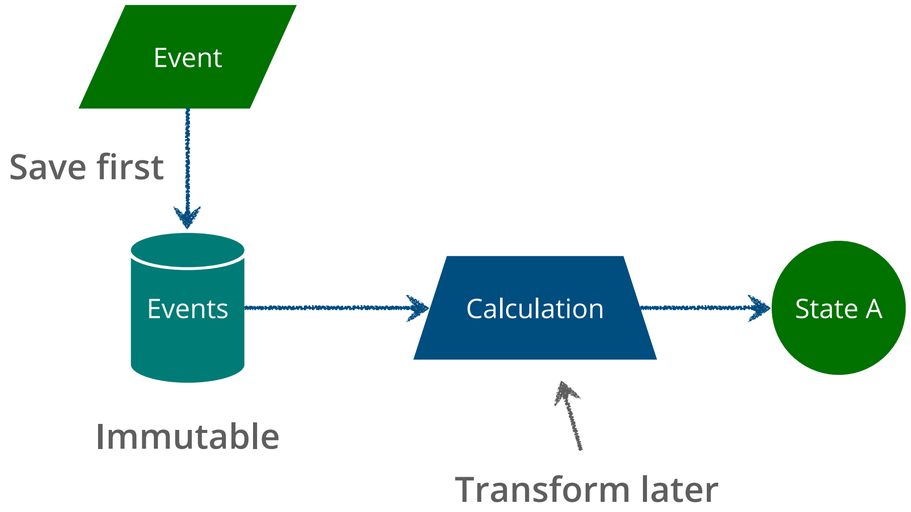


Summary

Traditional DB Design



Event Sourcing



Coding Exercise

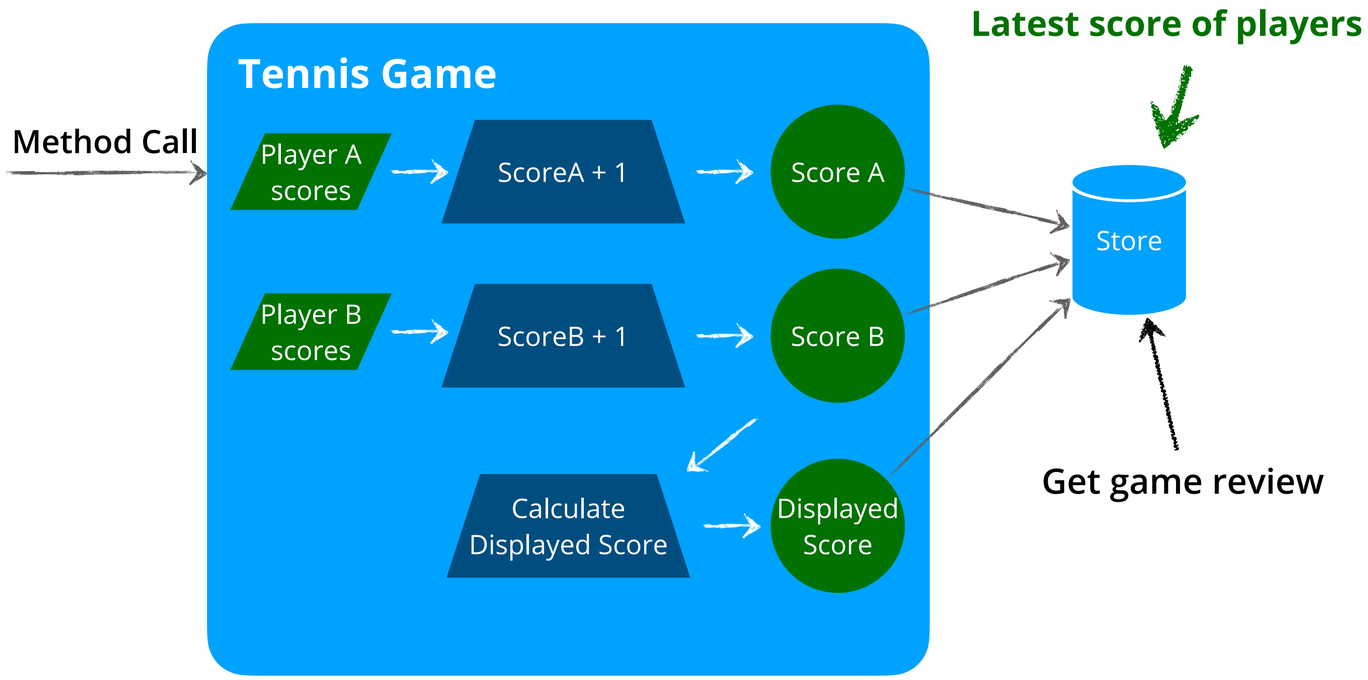
Implement event sourcing with tennis game kata

https://github.com/heaton/tennis-game

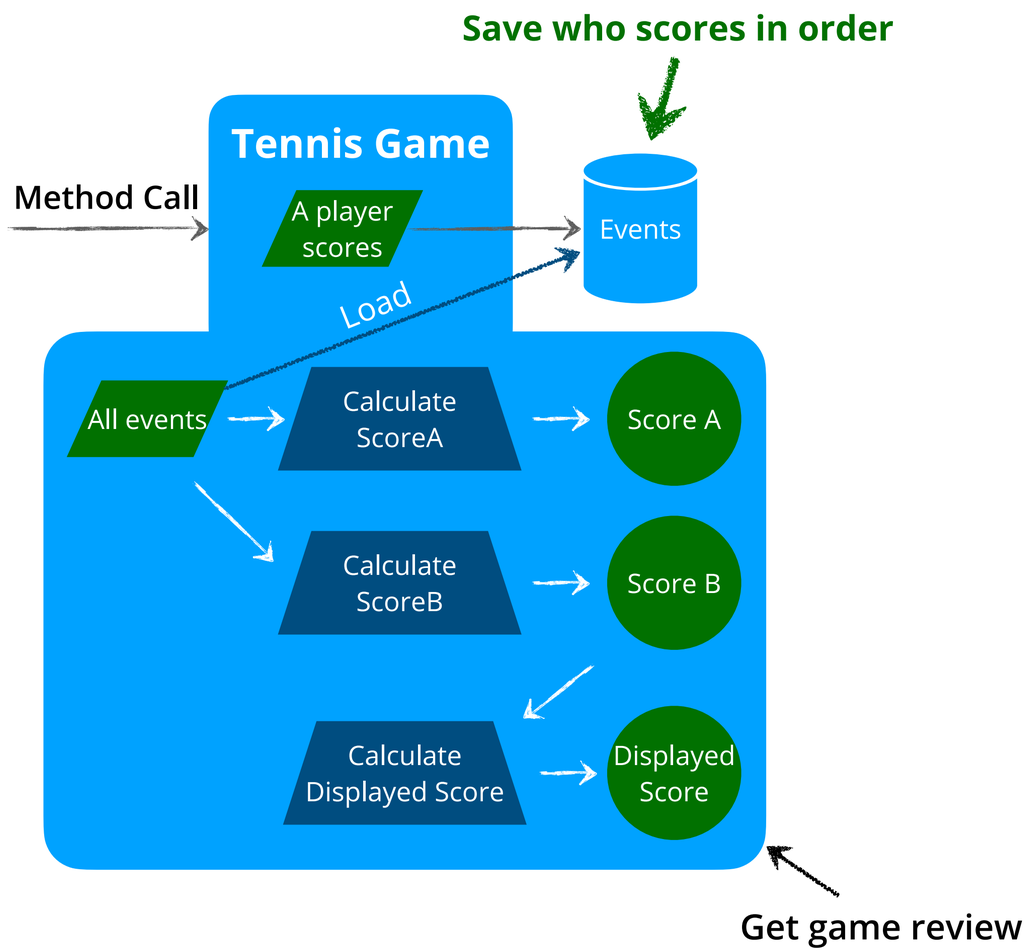
Recap

Tennis Game

Before refactoring



After refactoring



Handle the Business Logic

Traditional DB Design

Event Sourcing

Business Logic

```
fun f(currentState, event): newState
```

Calculate state for each event

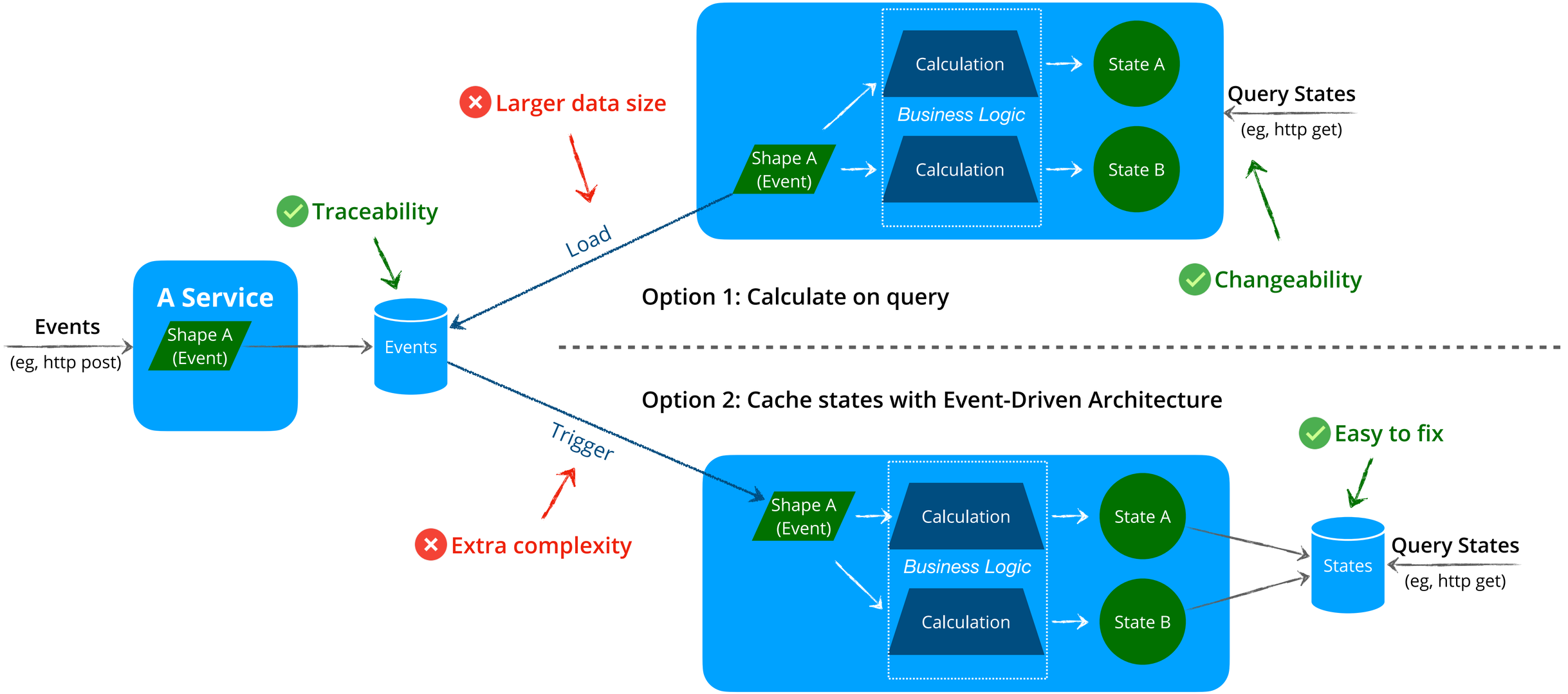
```
val latestState =
  if(firstEvent)
    f(initialState, event)
  else
    f(currentState, event)
```

Calculate latest state from all events

```
val latestState =
  events.fold(initialState, f)
```

https://github.com/heaton/tennis-game/pull/1/commits/40ffcf9f1dd5e142dea51ed5073910bbc8ac70ab

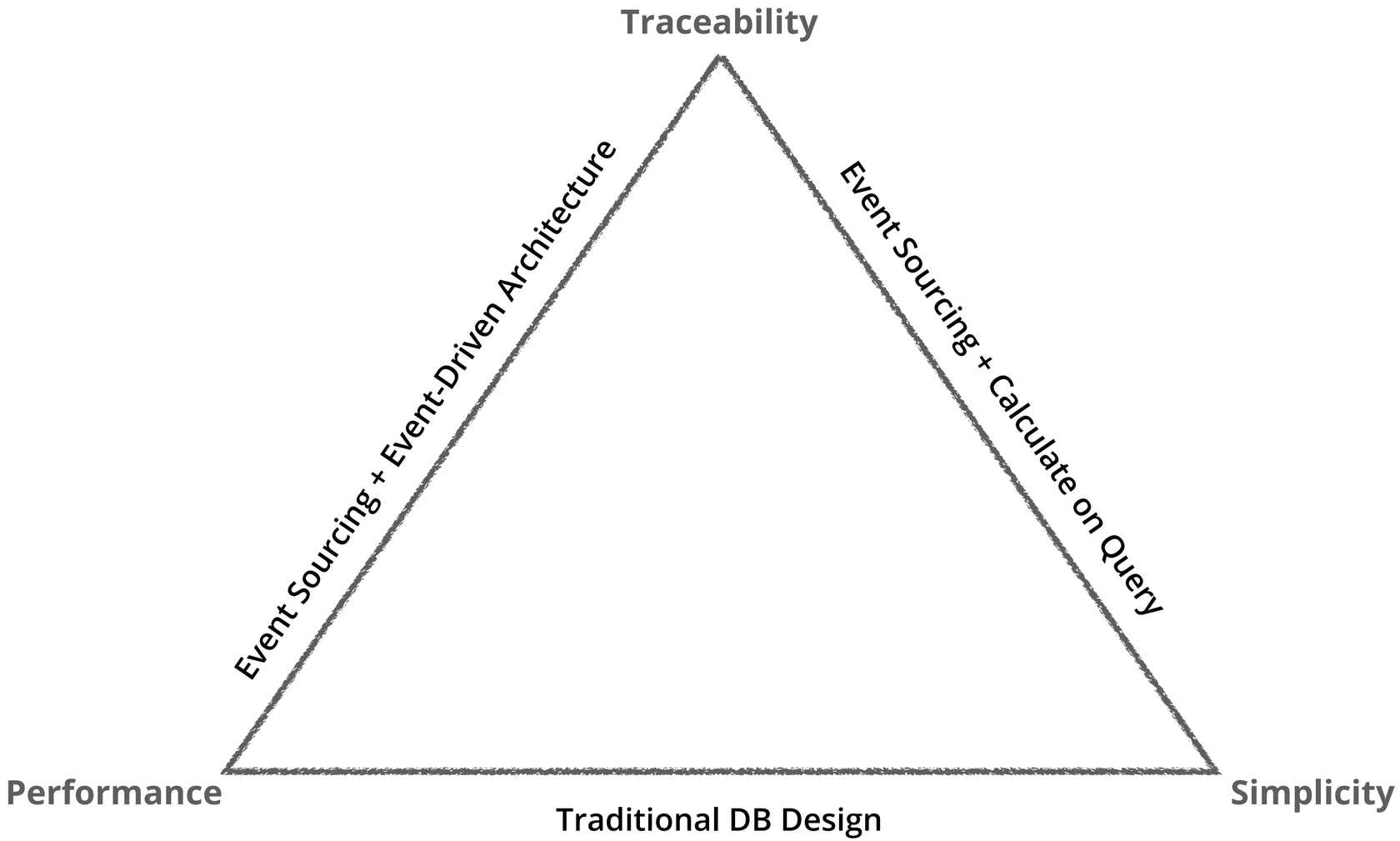
Trade-offs


Larger data size
Traceability
Load
Shape A
(Event)
Calculation
Business Logic
Calculation
State A
State B
Query States
(eg, http get)
Changeability
A Service
Events
(eg, http post)
Shape A
(Event)
Events
Option 1: Calculate on query
Option 2: Cache states with Event-Driven Architecture
Trigger
Extra complexity
Easy to fix
Shape A
(Event)
Calculation
Business Logic
Calculation
State A
State B
States
Query States
(eg, http get)

Trade-offs

Trade-offs	Traditional DB Design	Event Sourcing with Calculation on Query	Event Sourcing with Event-Driven Architecture
Traceability	✗	✓	✓
Performance	✓	✗	✓
Complexity	✓	✓	✗
Changeability	✗	✓	✗
Choose when	No traceability is required	The size of events to calculate an state is small	When you need both traceability and very high performance

Summary


Traceability
Event Sourcing + Event-Driven Architecture
Event Sourcing + Calculate on Query
Performance
Simplicity
Traditional DB Design