Scaling CQRS in Theory, Practice and Reality

Allard Buijze
Founder & CTO, AxonIQ
✉️ allard@axoniq.io
✈️ allardbz

Nakul Mishra
Sr Engineer & Architect, Casumo
✉️ nklmish@protonmail.com
✈️ nklmish
Layered architecture

- User Interface
- Service Layer
- Data Access Layer
- Domain Model
'Normal' SQL QUERY

CREATE ALGORITHM = UNDEFINED DIRECTIVES 'SET SESSION 'NOCACHE' = OFF;' SET SESSION 'NOCHECKTABLE' = OFF;' SELECT * FROM some_table WHERE condition;

22 JOINS

6 SUBQUERIES
Layered architecture

*Ponder and deliberate before you make a move.* —Sun Tzu

**User Interface**
- Web Cache
- Session replication

**Service Layer**
- Method invocation Cache
- Worker pools

**Data Access Layer**
- Distributed 2nd level cache
- Query Cache

**Domain Model**
Microservices systems

• Splitting up systems into smaller, simpler components
  • Agility
  • Scalability
CQRS
Command Query Responsibility Segregation

Events

Command

T: 1 mln / s Resp: < 10 ms
T: Thr. 20 / s Resp: < 100 ms

Projections

T: 10 mln / s Resp. < 100 ms
T: 1 / s Resp. < 10 ms

Client
Are you tall enough?

You must be this tall to use microservices

Source: martinfowler.com/bliki/MicroservicePrerequisites.html
“Noun Driven Design”
“Entity Services”
Monoliths

St Breock Downs Monolith - www.cornwalls.co.uk
Location transparency

A component should neither be aware of nor make any assumptions about the location of components it interacts with.

Location transparency starts with good API design

*(but doesn’t end there)*
Microservices Messaging

Commands
- Route to single handler
- Use consistent hashing
- Provide result

Events
- Distribute to all logical handlers
- Consumers express ordering req's
- No results

Queries
- Route with load balancing
- Sometimes scatter/gather
- Provide result

"Event" and "Message" is not the same thing
EventStore (Percona)

- Writes / event sourcing

ComponentA

- ComponentB

- ComponentC

- Projection

BigData Platform

- Reads historical events

- Publish event-A

- Consume event-A

RabbitMQ

- Consumes new events

- Publish event - A

- Publish event - C

Read / replay

Built with Axon Framework

AxonIQ  Casumo

@allardbz  @nklmish
Casumo

- Event store with >30 billion events
- Hundreds of millions of events, every day
- Every event is EQUALLY important

“I was thinking I might win 50 pounds but when it went all the way to the jackpot I was shocked.” - Mega Fortune £2,700,000 jackpot won on the 3rd spin.
Event Store operations

- Append
- Validate ‘sequence’
Event Store operations

• Full sequential read
Event Store operations

• Read aggregate’s events
Serialized form

• Keep it small

• Use aliases & abbreviations

• Carefully select serializer
Partitioning and archiving
Loading aggregates - snapshots

- **0:** Wallet created
- **1:** €10 deposited
- **2:** €1 wagered
- **3:** €2 wagered
- **4:** €1 wagered
- **5:** €2000 won
- **6:** €1000 withdrawn
- **7:** €10 wagered
- **8:** €1M jackpot won!!

Wallet balance:
- €996 in wallet @ 7
- €7 in wallet @ 3

Twitter handles: @allardbz, @nklmish
Replays

```
TRUNCATE TABLE xyz;
```
Partial replays

Idempotent event handling
Perform selective, ad-hoc, replays

“Reasonable” timeframe

TRUNCATE TABLE xyz;

Event Handler
Projection
Evolutionary microservices

Commands

Events

Queries
Unmanageable mess

As a Wallet module I want to know when to withdraw money from the wallet

Bet Placed
Bet Accepted
Bet Won
Bet Lost
Bet Reverted
Communication = Contract
Bounded context

Explicitly define the context within which a model applies. Explicitly set boundaries in terms of team organization, usage within specific parts of the application, and physical manifestations such as code bases and database schemas. Keep the model strictly consistent within these bounds, but don’t be distracted or confused by issues outside.
Within a context, share ‘everything’
Between contexts, share ‘consciously’

As a Wallet module I want to know when to withdraw money from the wallet.

Bet placed → Claim Funds

Bet Placed
Bet Accepted
Bet Won
Bet Lost
Bet Reverted
Lessons learned
References

• Axon
  • axoniq.io
  • github.com/axonframework
  • @axonframework

• Casumo
  • casumo.com
  • @casumotech