.NET DEVELOPER DAYS

Streamlining Microservice Communication with CQRS and Event Sourcing

Oli Sturm

oliver@oliversturm.com

@olisturm@mastodon.world

DevExpress®

Oliver Sturm

Training Director at DevExpress

Consultant, trainer, author, software architect and developer for over 30 years

11 year Microsoft C# MVP, Docker Captain

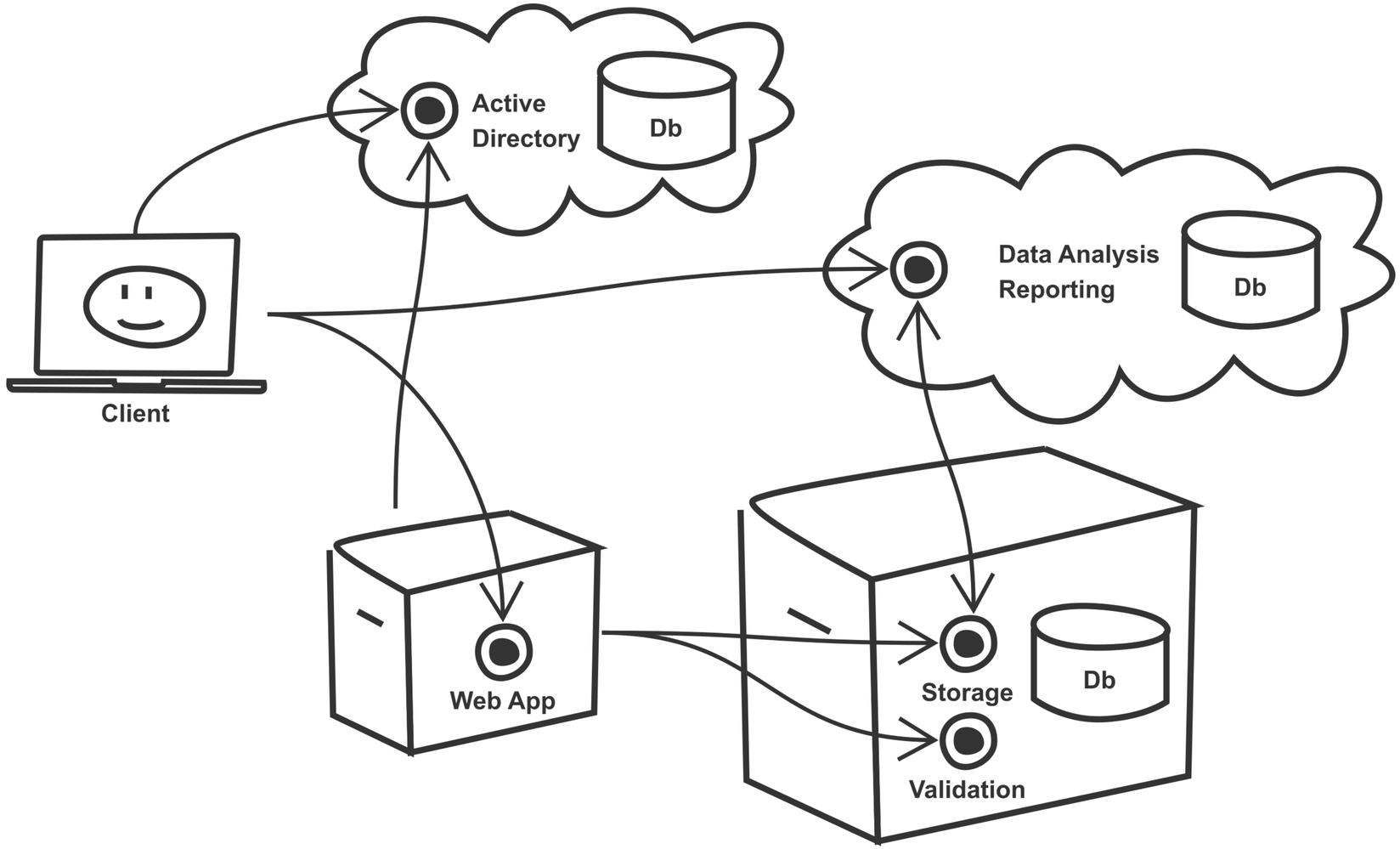
Contact: oliver@oliversturm.com

Mastodon: @olisturm@mastodon.world

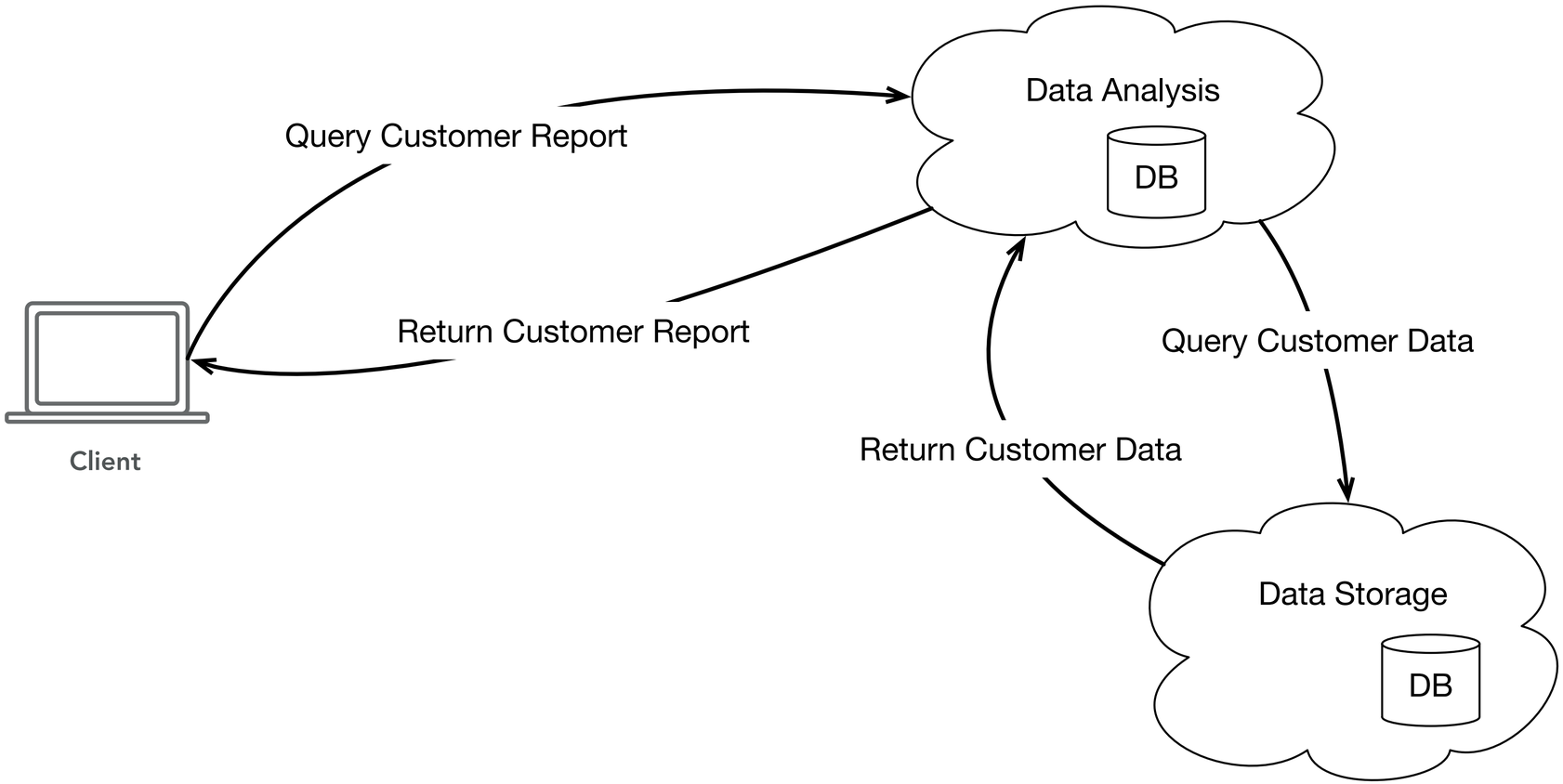
The Plan

- A look at a “normal” service-based system
  - … and how it can be a bit chaotic
- CQRS/ES — how does that work?
- Unidirectional communication paths FTW!

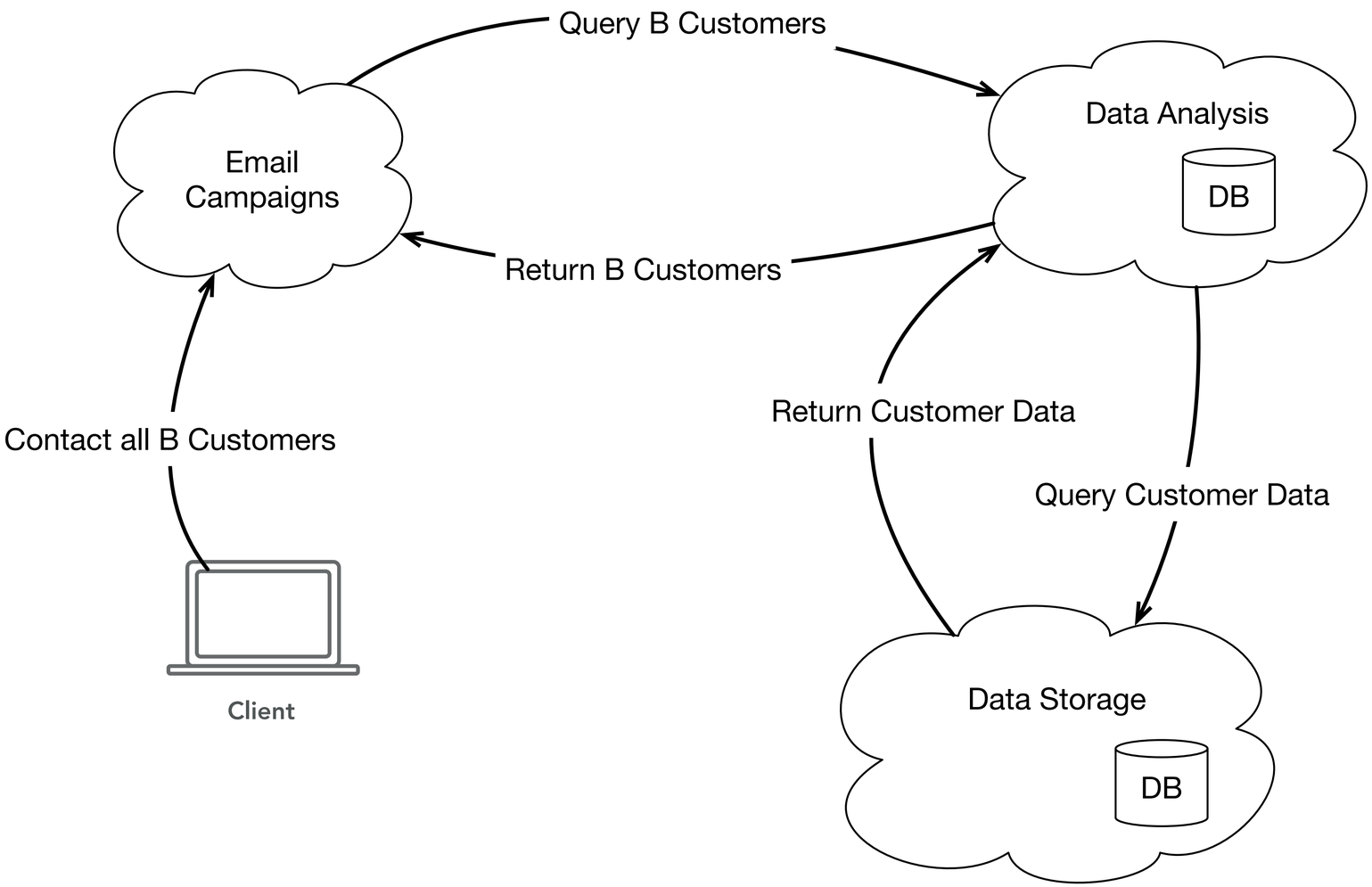
General Service Structure



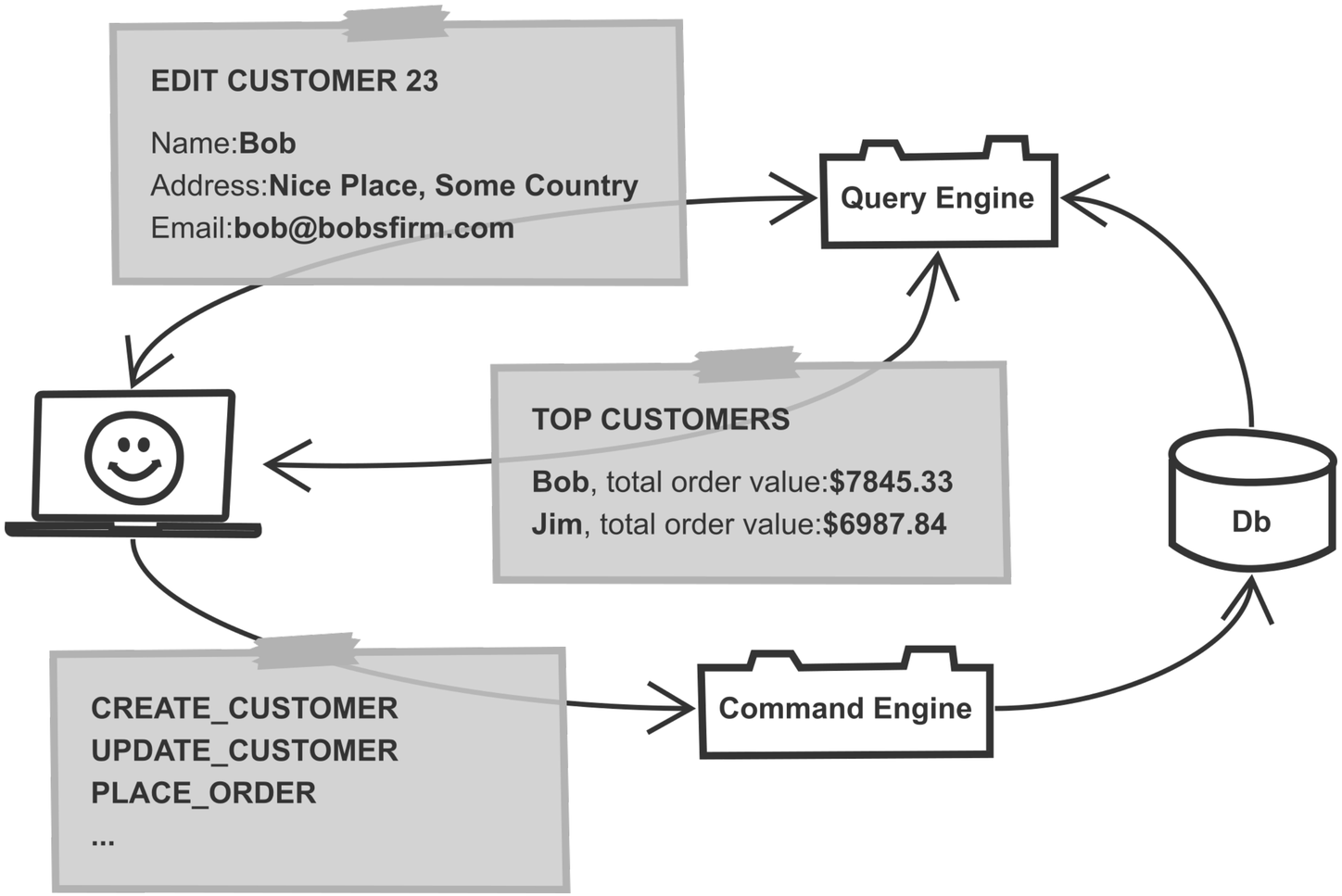
Customer Report



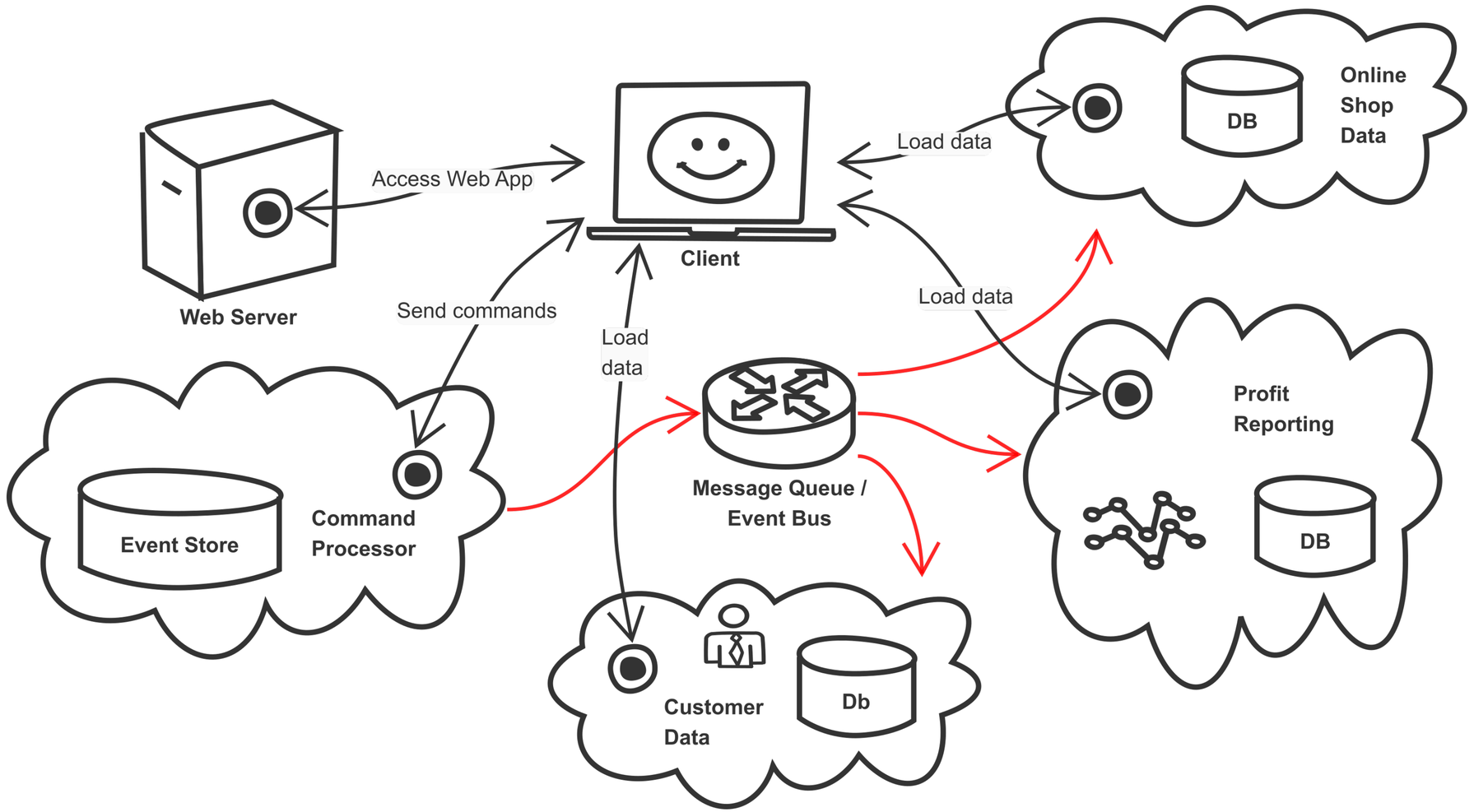
B Customers Email Campaign



CQRS — Command/Query Responsibility Segregation



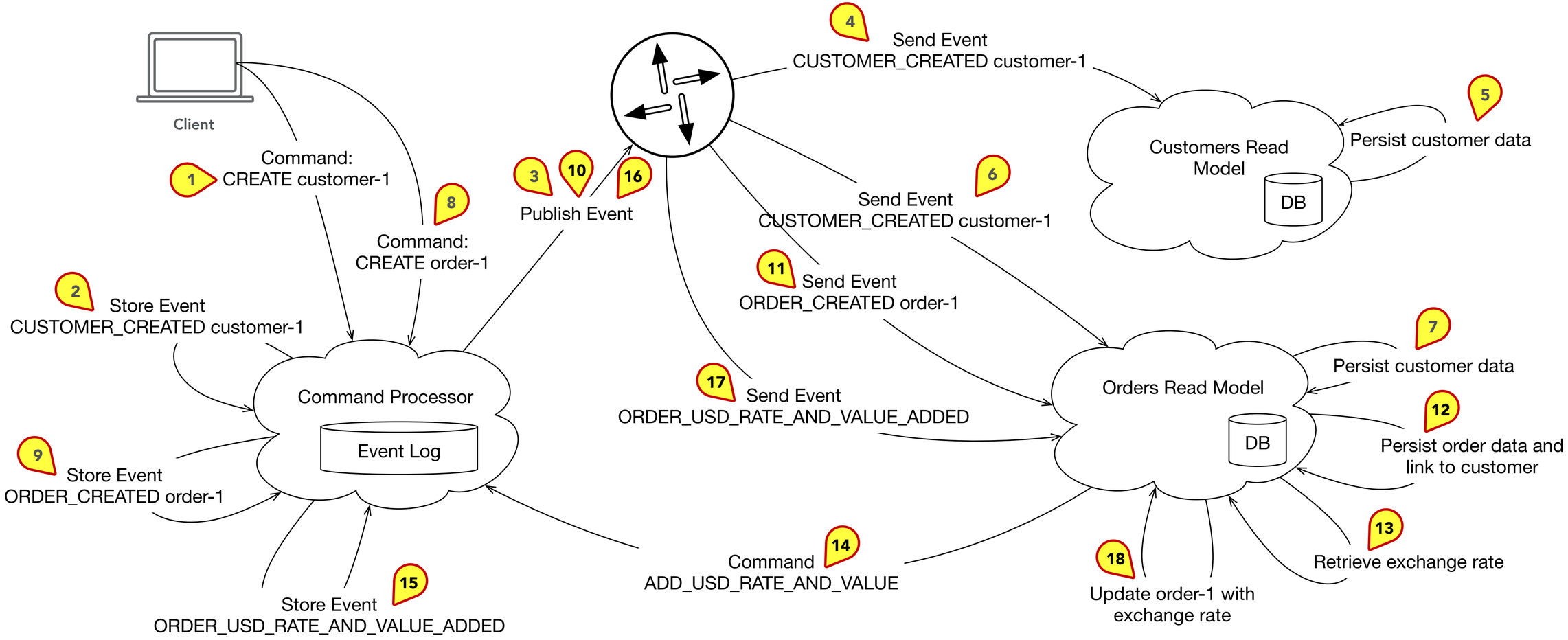
CQRS with Event Sourcing



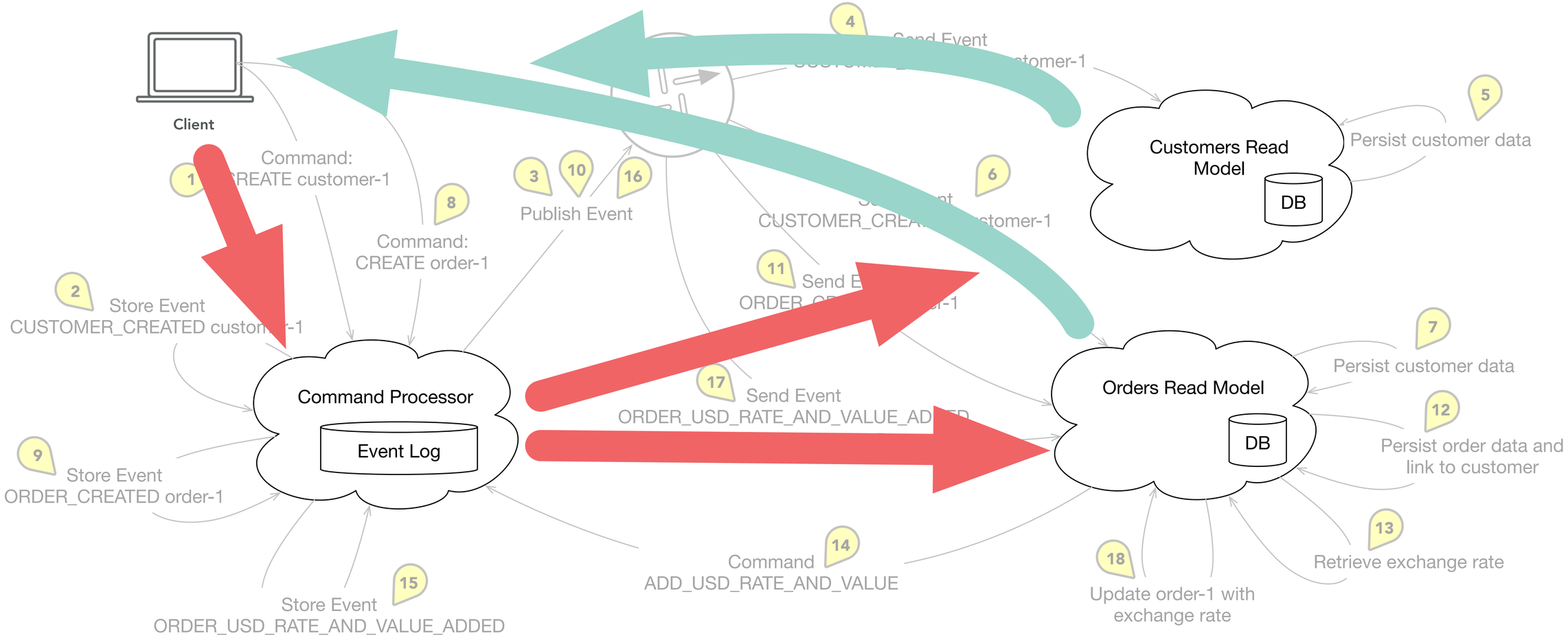
Demo

A CQRS/ES System

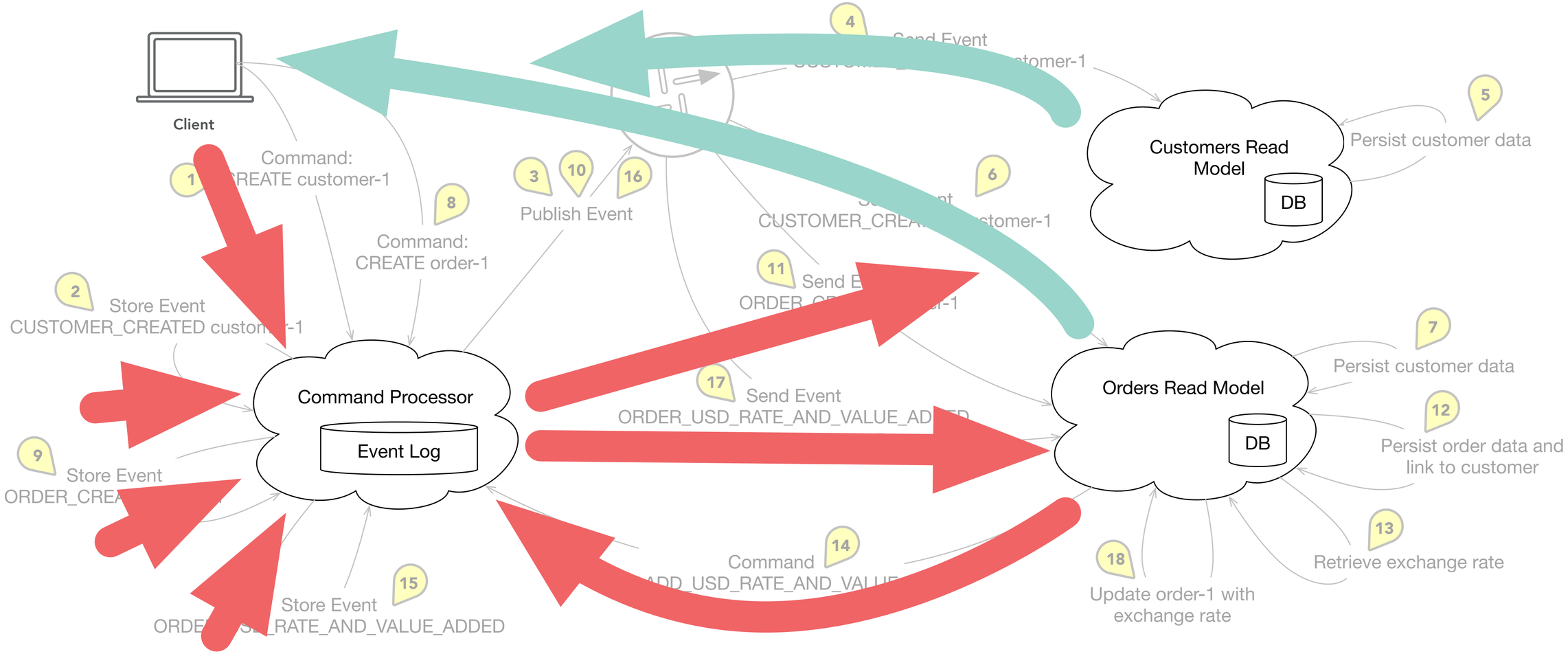
Demo Call Sequence



Demo Call Direction



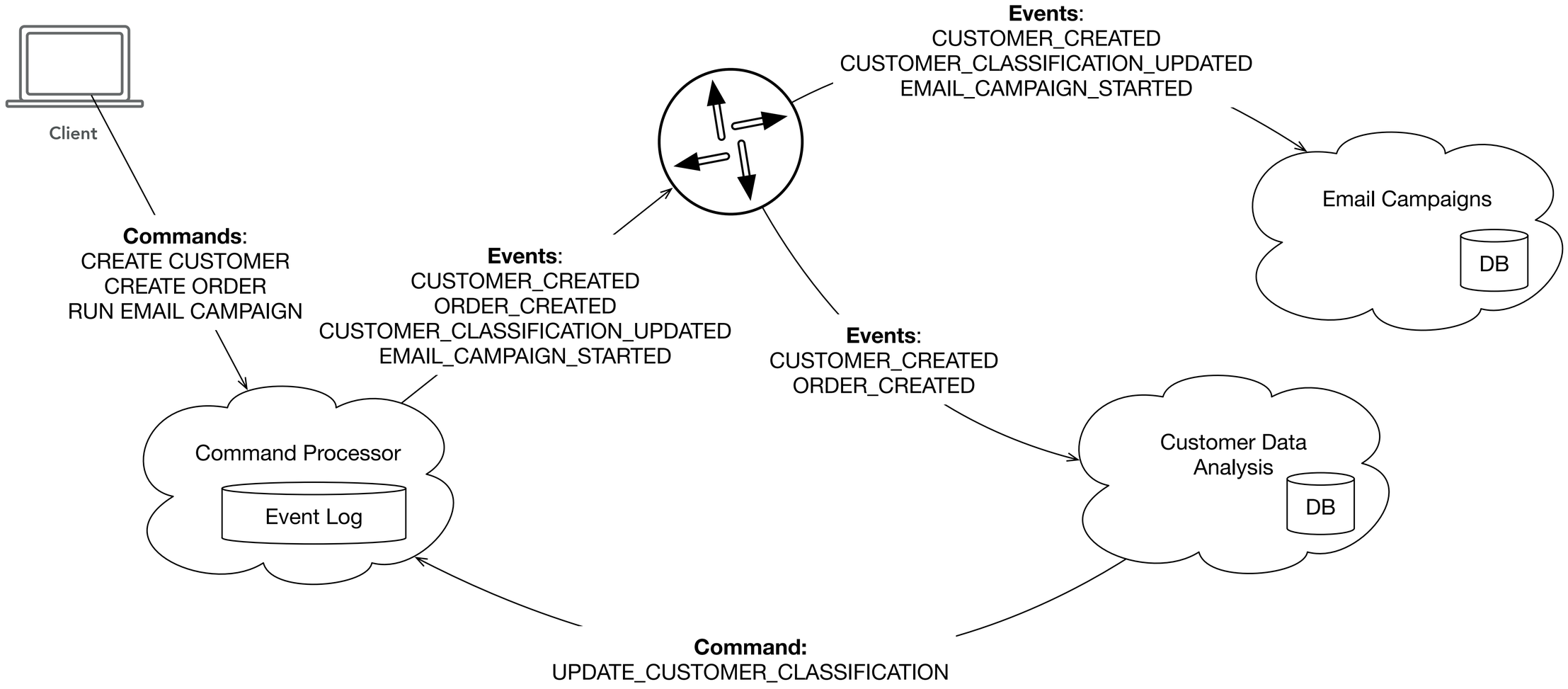
Demo Call Direction



B Customers Email Campaign

- "Email Campaign" read model / saga
- *Could* retrieve customer classification "on the fly"
- Much better though to persist the info in the event stream
  - There is also other data — customer permissions for email, logs of sent/reacted, …
- *Could* share classification implementation with data analysis service, or call the service as an external resource
- Much better to make that service calculate classification and publish `CUSTOMER_CLASSIFICATION_UPDATED` as needed!
- ***Event store should be used to persist all relevant information!***

B Customers Email Campaign



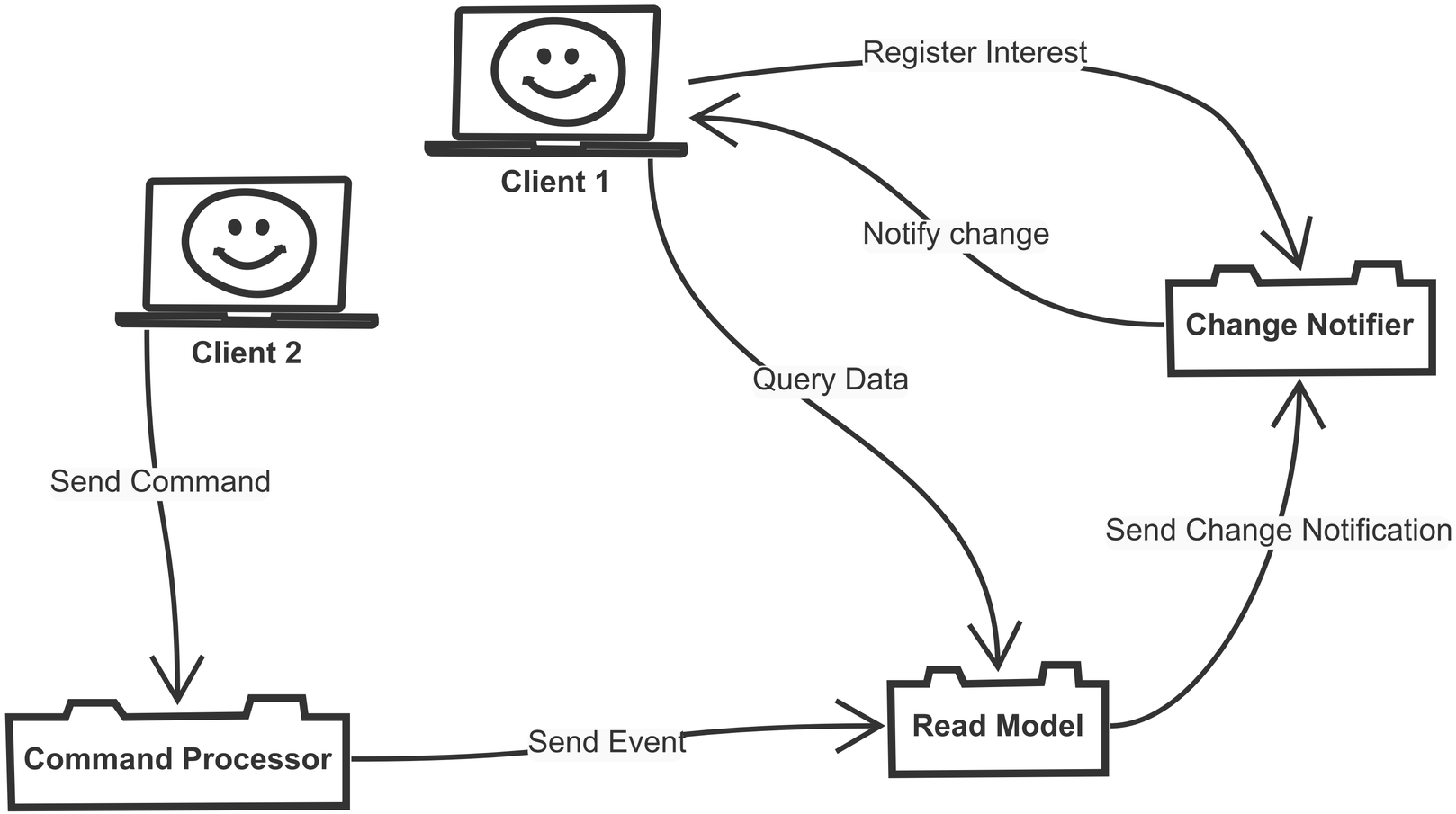
The Missing Bit: Eventual Consistency

- Transactions in distributed systems are difficult, expensive and disruptive
- Most distributed systems are eventually consistent
  - Fun fact: statistics say that this makes them more frequently consistent than other systems
- Advice: deal with it, don't combat it!
  - Your software will be more stable as a result
  - Like with every good rule, there are exceptions here.
- Think about every consistency issue by considering compensation
- Meanwhile, keep your users up to date…

Demo

Change Notification

Extra Communication Path: Change Notification



Summary

- The pattern(s) CQRS/ES define service responsibilities and communication structures are standardized as an automatic consequence
- The resulting system is fast, extensible and scalable, and maintenance as well as lifecycle management is fantastically simple
- Check out CQRS/ES frameworks for your favorite language!

Sources

- This presentation
  https://osturm.me/microservice-comms-cqrs-es
- Source code:
  https://github.com/oliversturm/lazyapps-public

Thank You

Please feel free to contact me
about the content anytime.

Oli Sturm
oliver@oliversturm.com
@olisturm@mastodon.world

DevExpress®

Please rate this session using

.NET DeveloperDays Mobile App
(available in AppStore & Google Play)