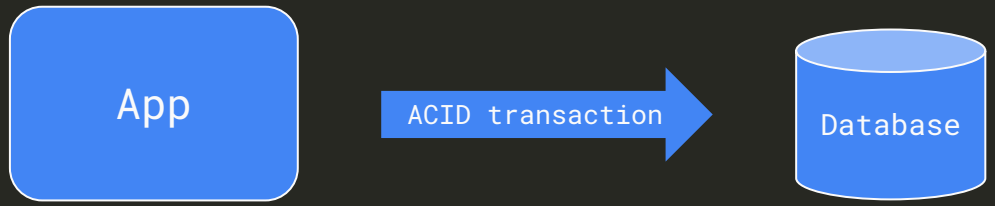
An Introduction to Saga Pattern for Distributed Transactions

Author: Dmitry Khorev

Database transaction

- Series of operations performed as a single unit of work
- Can include business logic checks
- Success - will commit all operations to the database
- Fail - will rollback all operations



ACID properties

A
C
I
D

- Atomicity
- Consistency
- Isolation
- Durability

About me

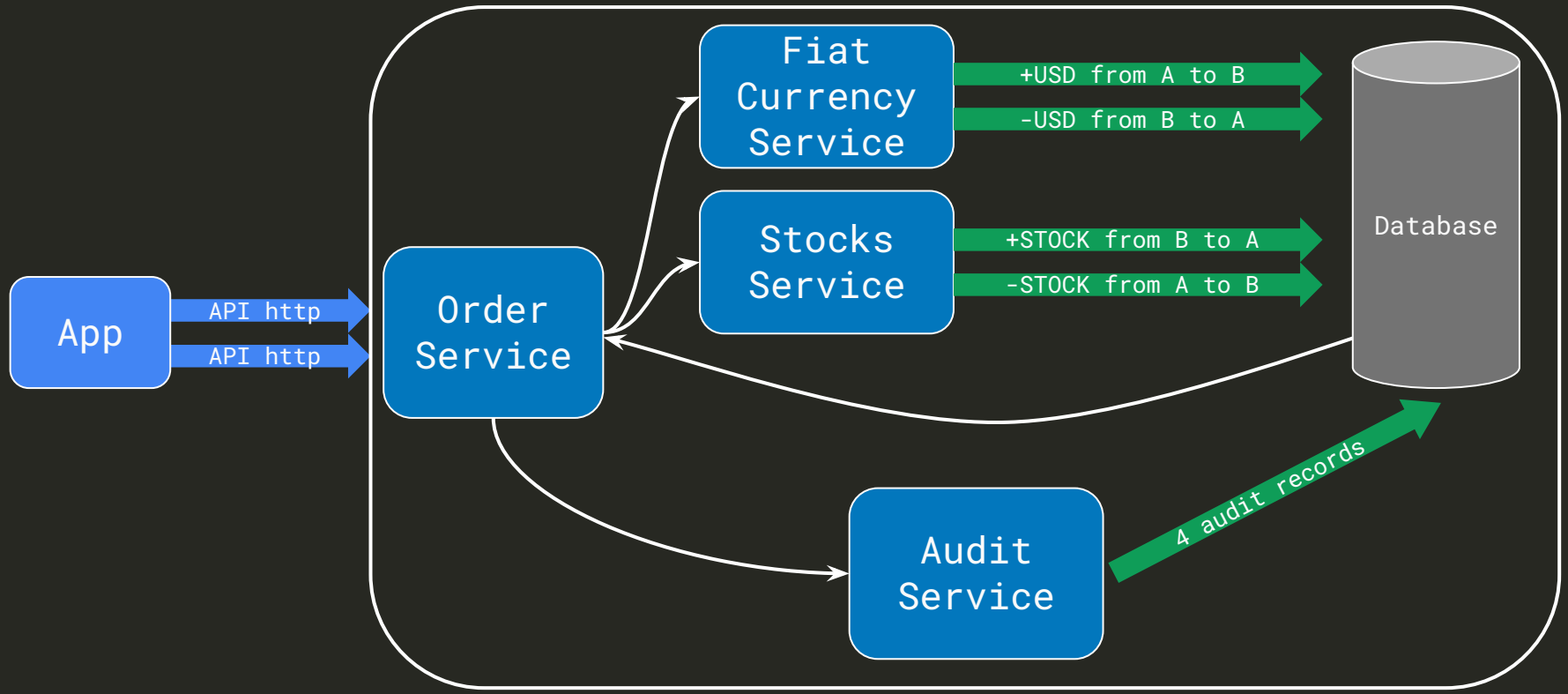
Dmitry Khorev

Software Engineer

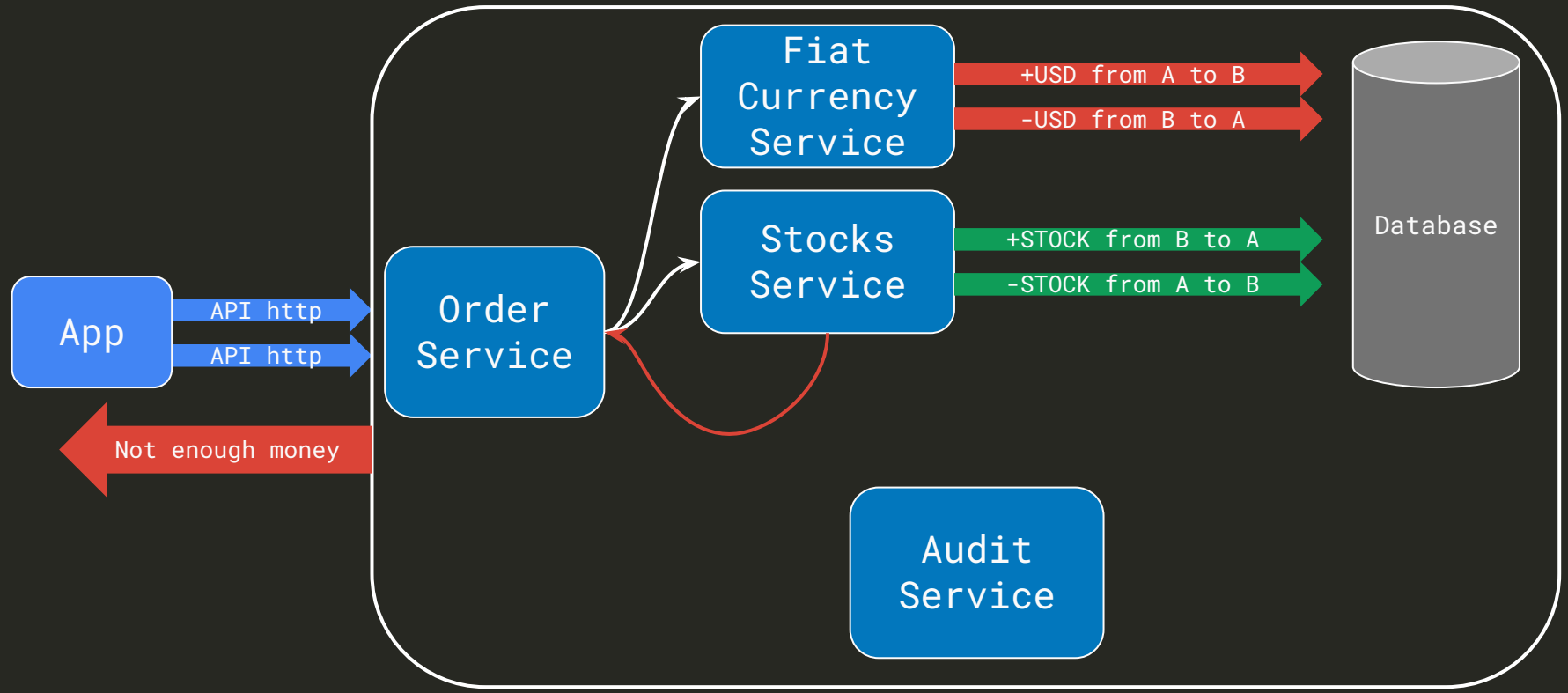
www.linkedin.com/in/dmitry-khorev | github.com/dkhorev | https://medium.com/@dkhorev

- **8+ YoE** in producing quality, scalable server and client applications**.**
- Skilled at creating **well-designed, testable, and efficient back-end services** using best practices.
- I write articles about exciting **Back-End technologies** on Medium.
- **Certified** Node.js Services Developer and Node.js Application Developer.
- Well-versed in Node.js and PHP back-end development. I also possess DevOps skills - Kubernetes, Terraform, and CircleCI, and have deep knowledge of some AWS services.

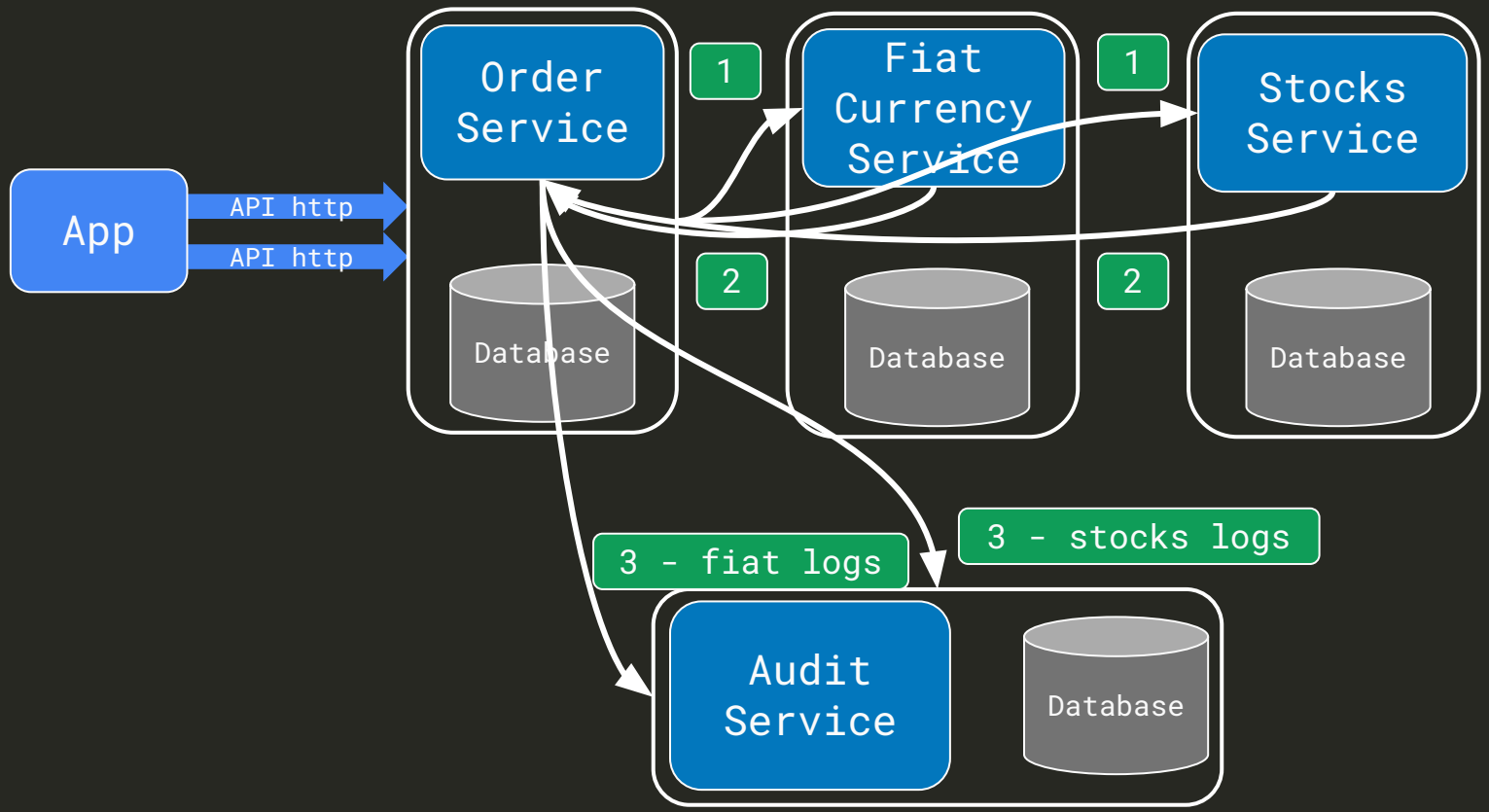
Monolithic application example



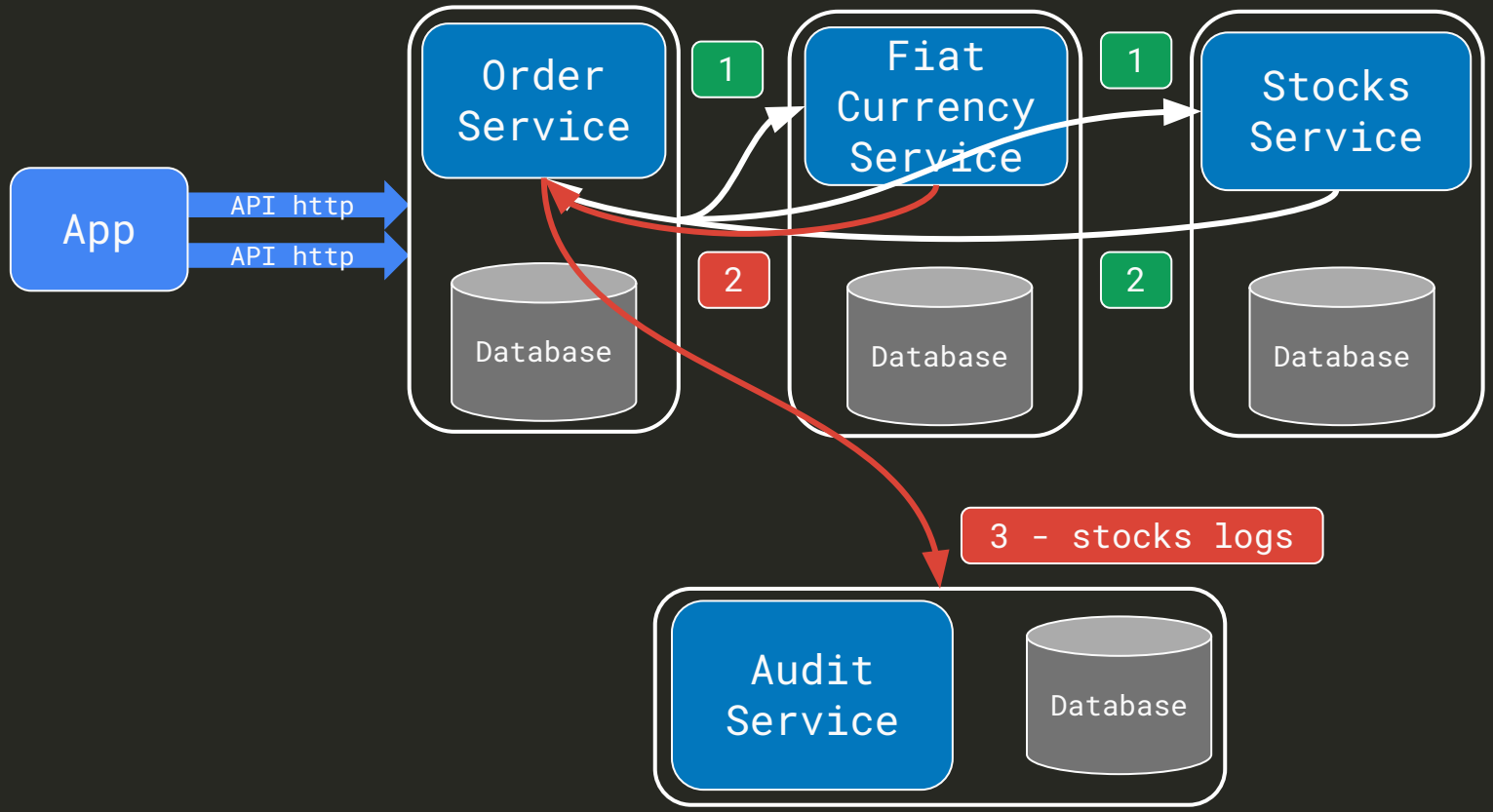
Monolithic application example



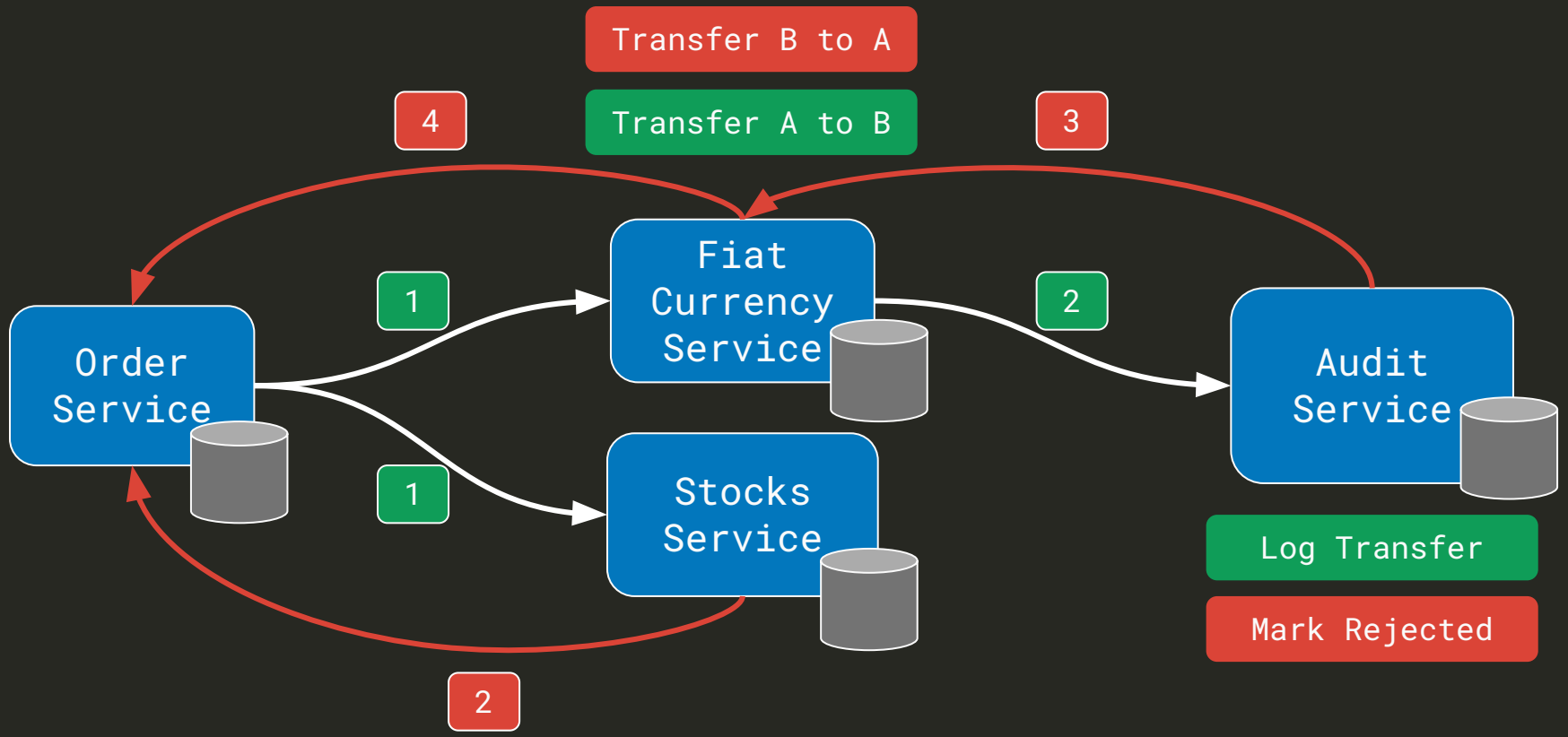
Microservices example



Microservices example

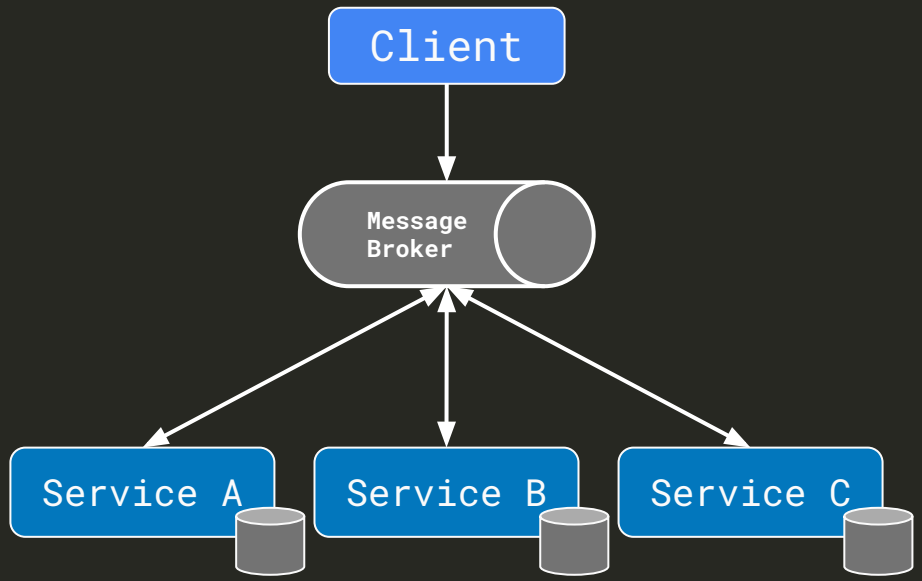


SAGA Pattern

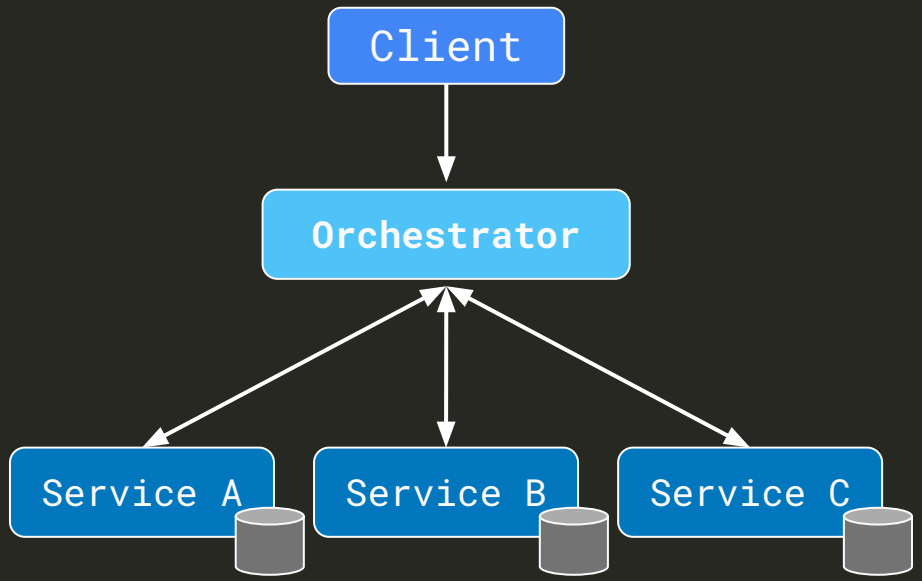


SAGA Pattern

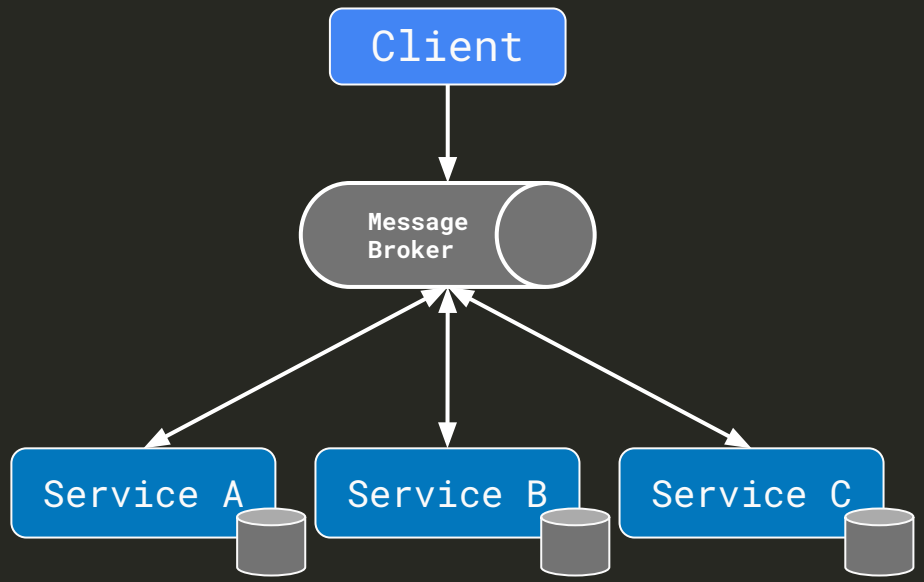
Choreography



Orchestration



Choreography



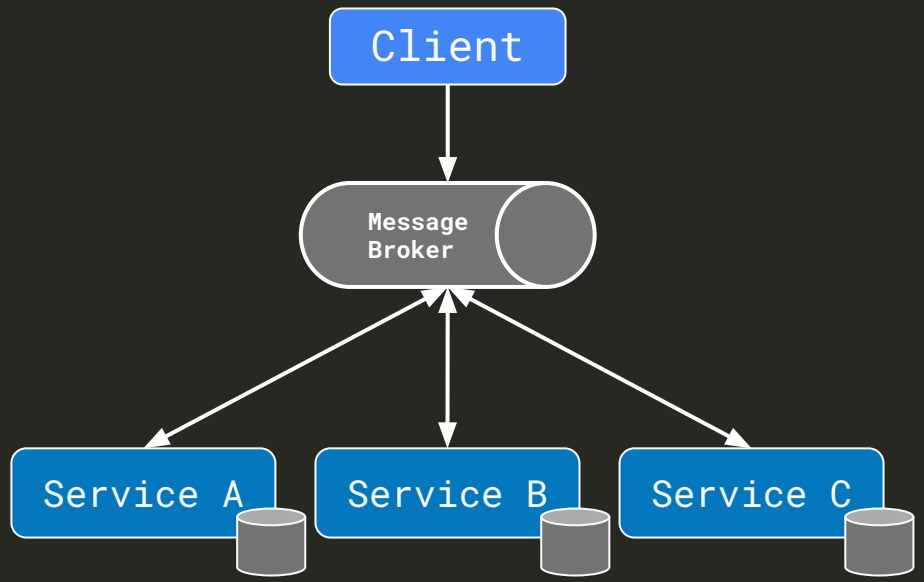
Pros

Good for simple workflows

No additional service

No single point of failure

Choreography



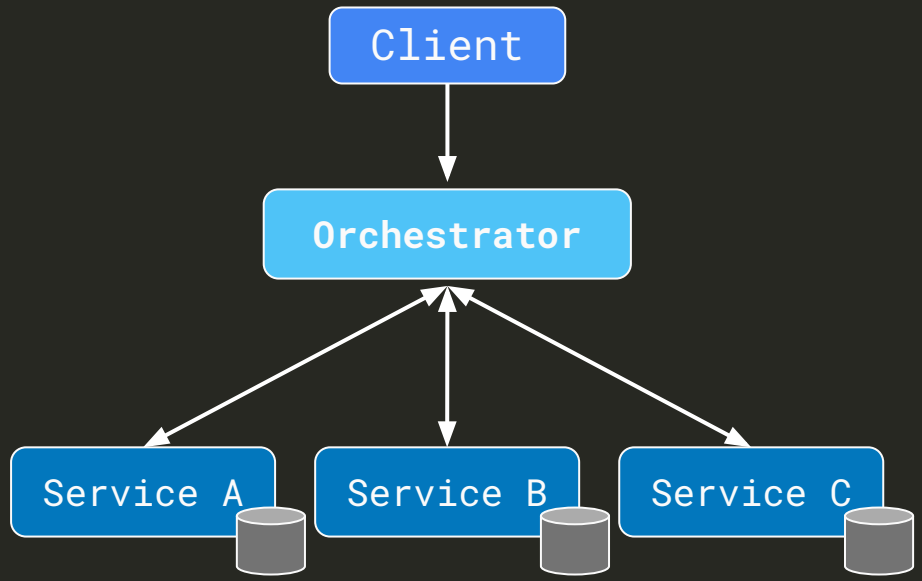
Cons

Workflow is confusing if too many steps

Cyclic dependencies

Integration testing is difficult

Orchestration



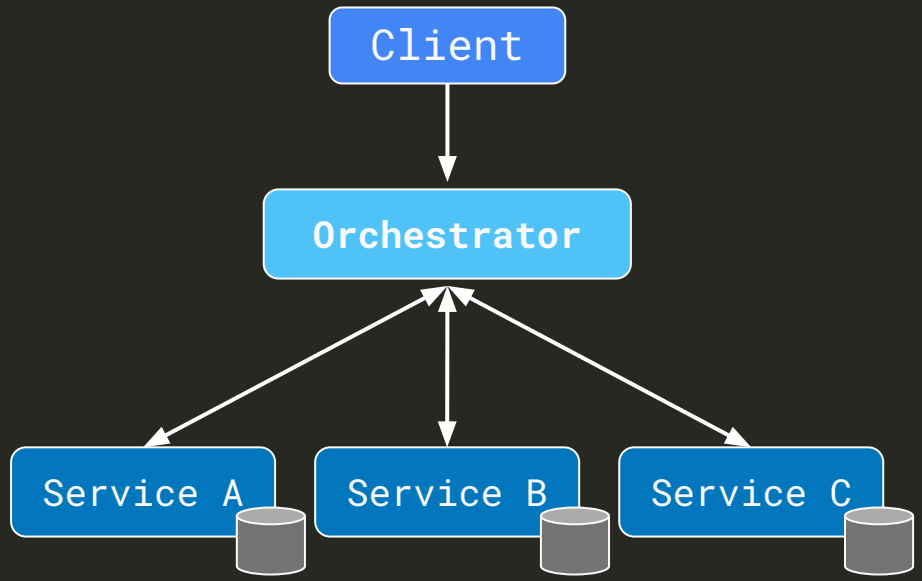
Pros

Good for complex workflows

No cyclic dependencies

Saga participants don’t need to know about each other

Orchestration

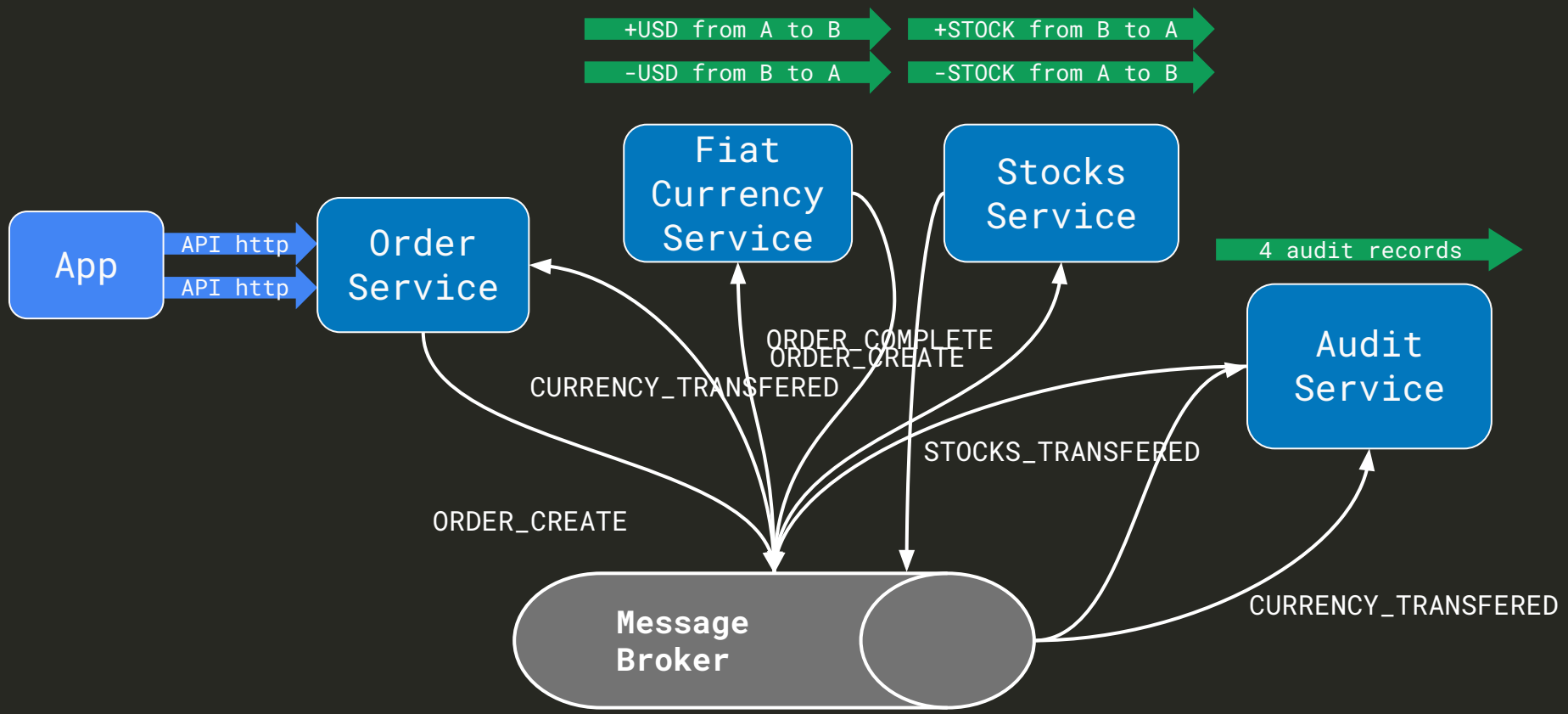


Cons

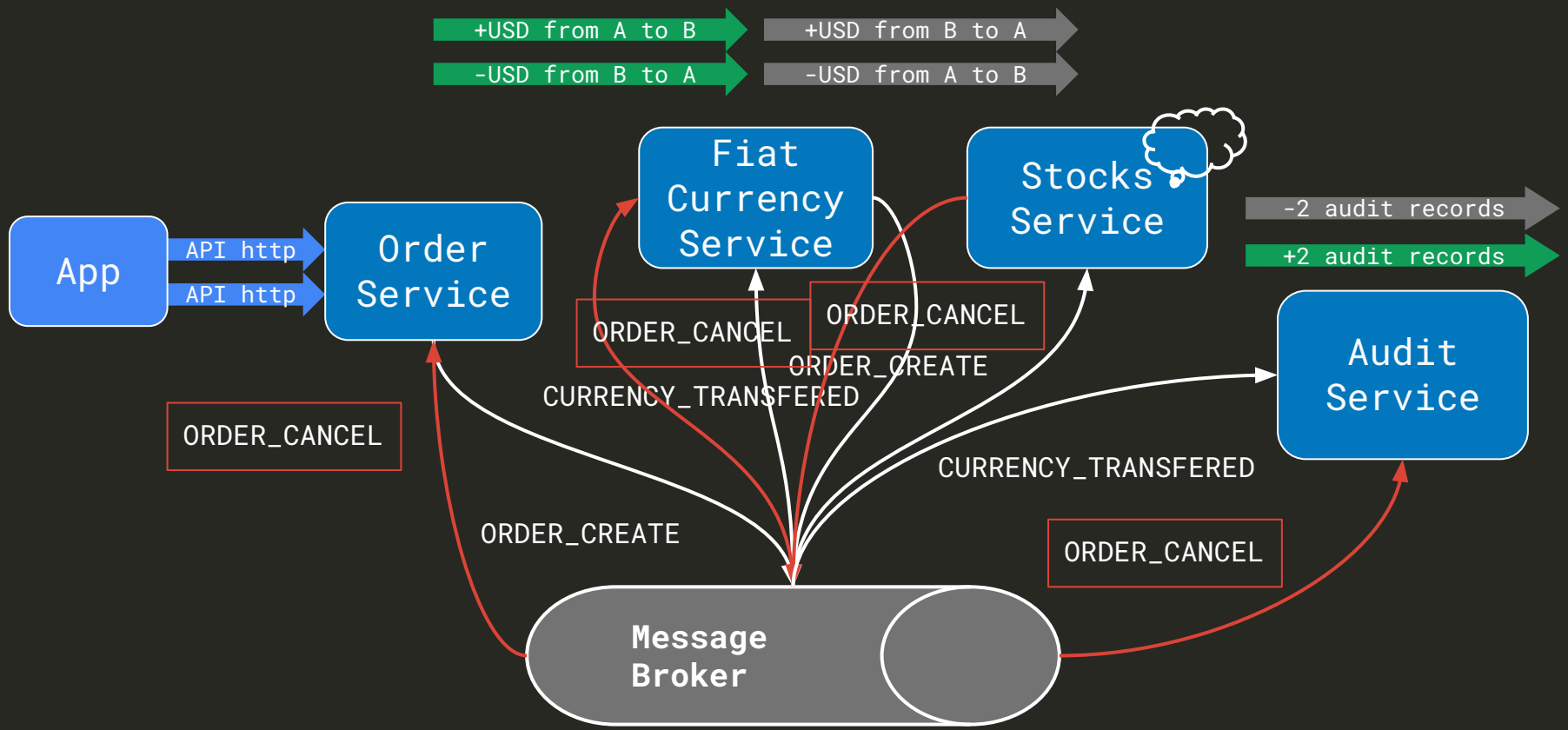
Design complexity and coordination logic

Single point of failure

Choreography - happy path



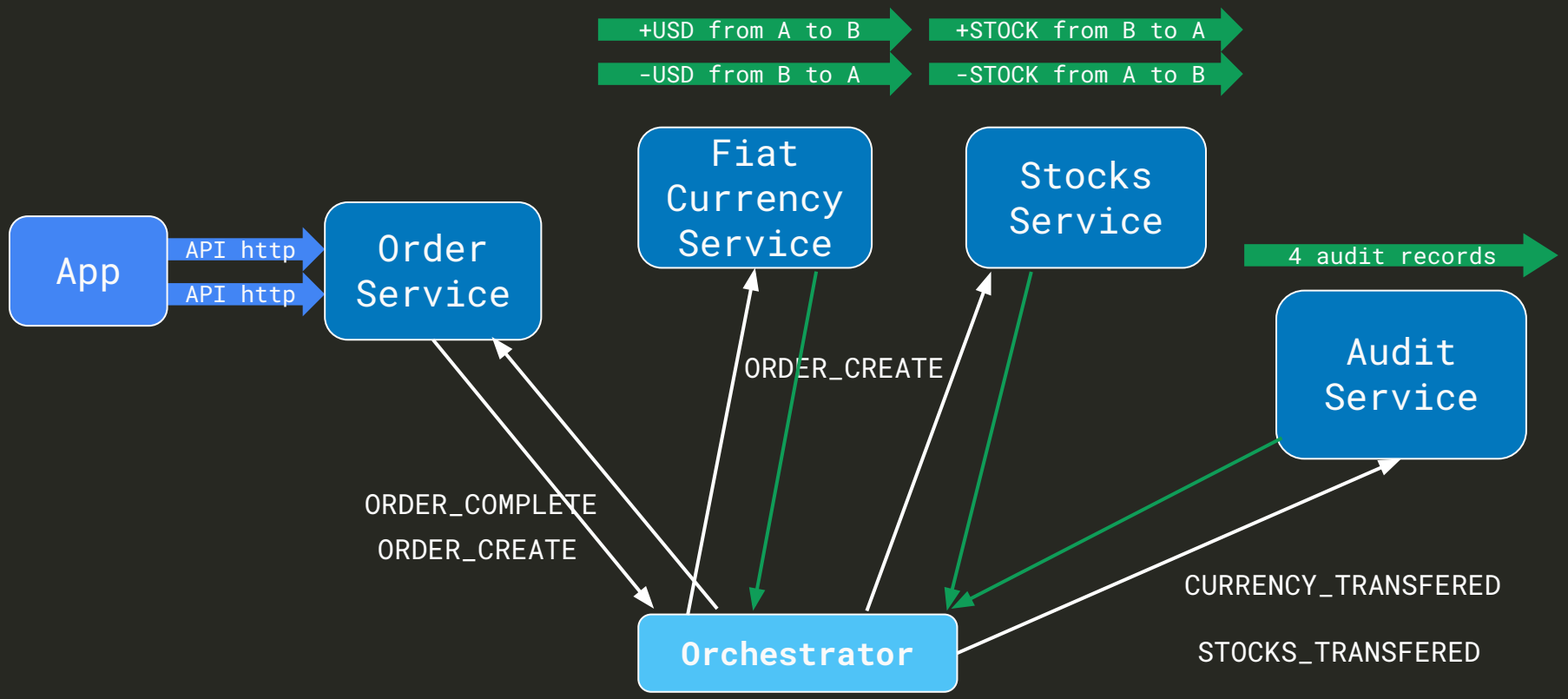
Choreography - error path



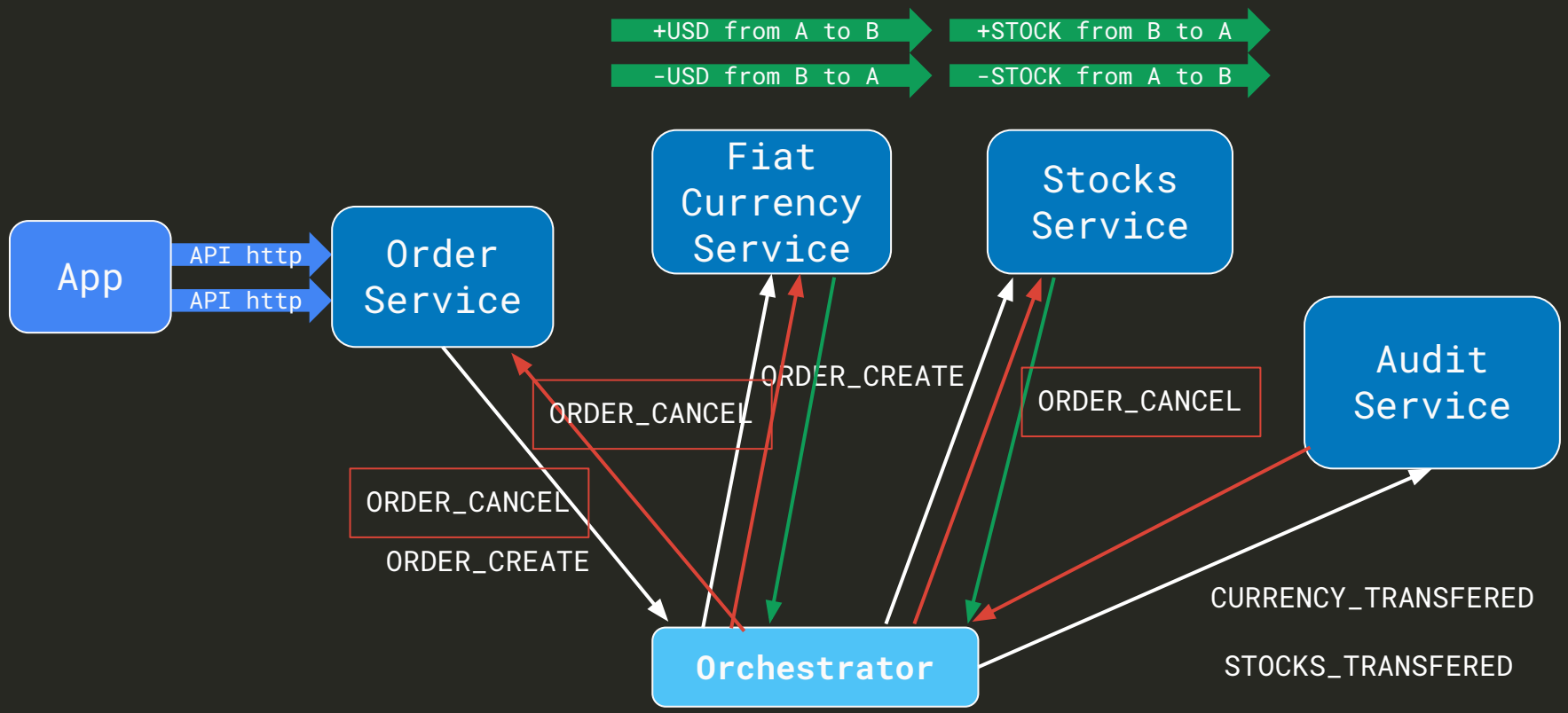
Choreography - events mapping

	Emits	Listens
Order Service	ORDER_CREATE	ORDER_CANCEL
Fiat Currency Service	CURRENCY_TRANSFERED ORDER_CANCEL	ORDER_CREATE ORDER_CANCEL
Stocks Service	STOCKS_TRANSFERED ORDER_CANCEL	ORDER_CREATE ORDER_CANCEL
Audit Service	ORDER_CANCEL	STOCKS_TRANSFERED CURRENCY_TRANSFERED ORDER_CANCEL

Orchestration - happy path



Orchestration - error path



Things to care about

- May initially be challenging
- Hard to debug
- Potential transient failures, idempotence required
- Monitoring is recommended

Thank you!

Dmitry Khorev

Software Engineer

www.linkedin.com/in/dmitry-khorev | github.com/dkhorev | https://medium.com/@dkhorev