Efficiency through Stateless Microservices: Streamlining Request Handling

Mike Malyi

Team leader - Youhodler

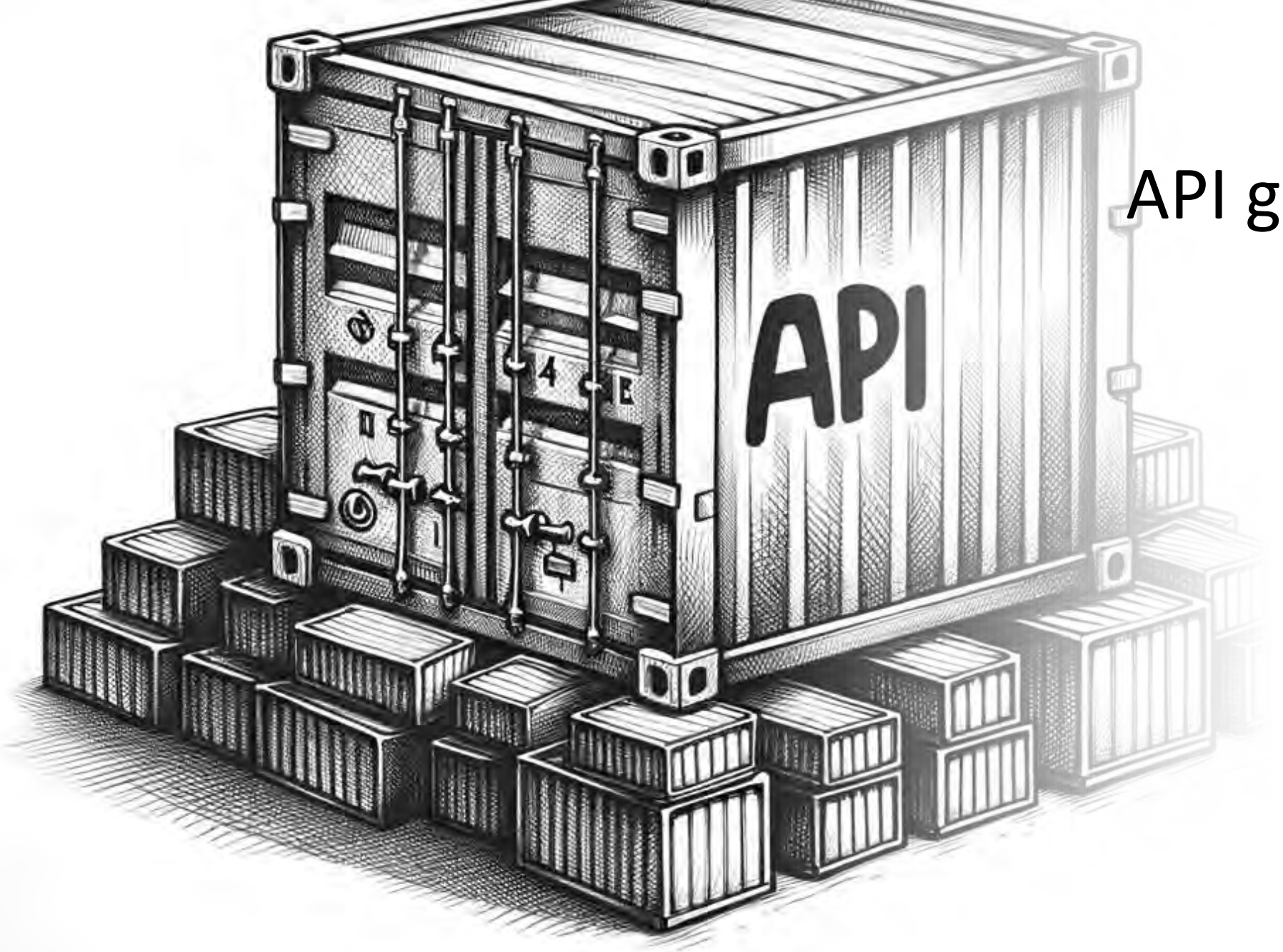
entropy

Microservices

- The Rise of Containerization and Orchestration
- Adoption of DevOps and CI/CD
- Utilization of Cloud Technologies
- Observability and Monitoring

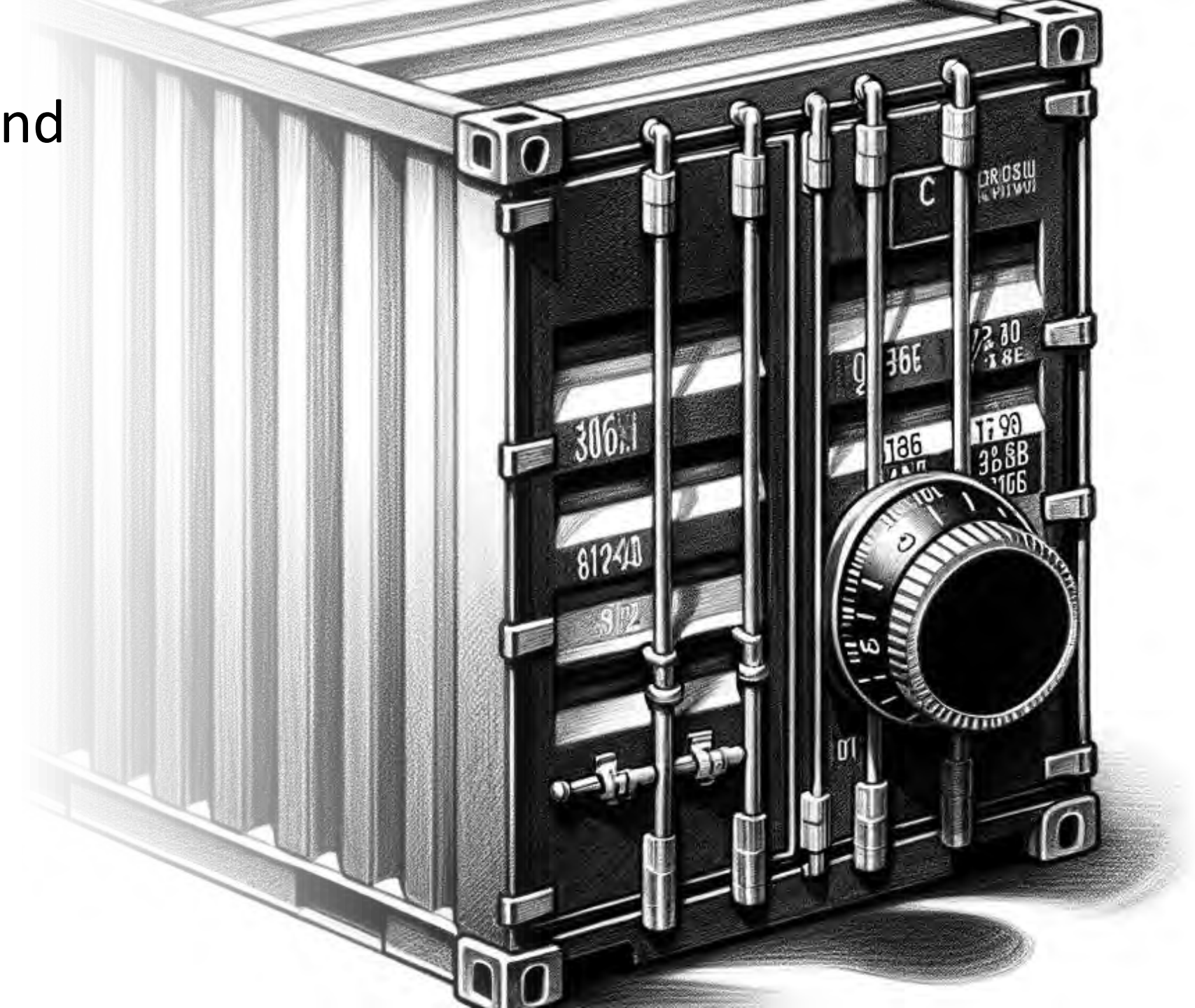
The Importance of Stateless Microservices

- Scalability
- Reliability and Fault Tolerance
- Simplified Management
- Load Balancing
- Security Assurance
- Flexibility and Independence

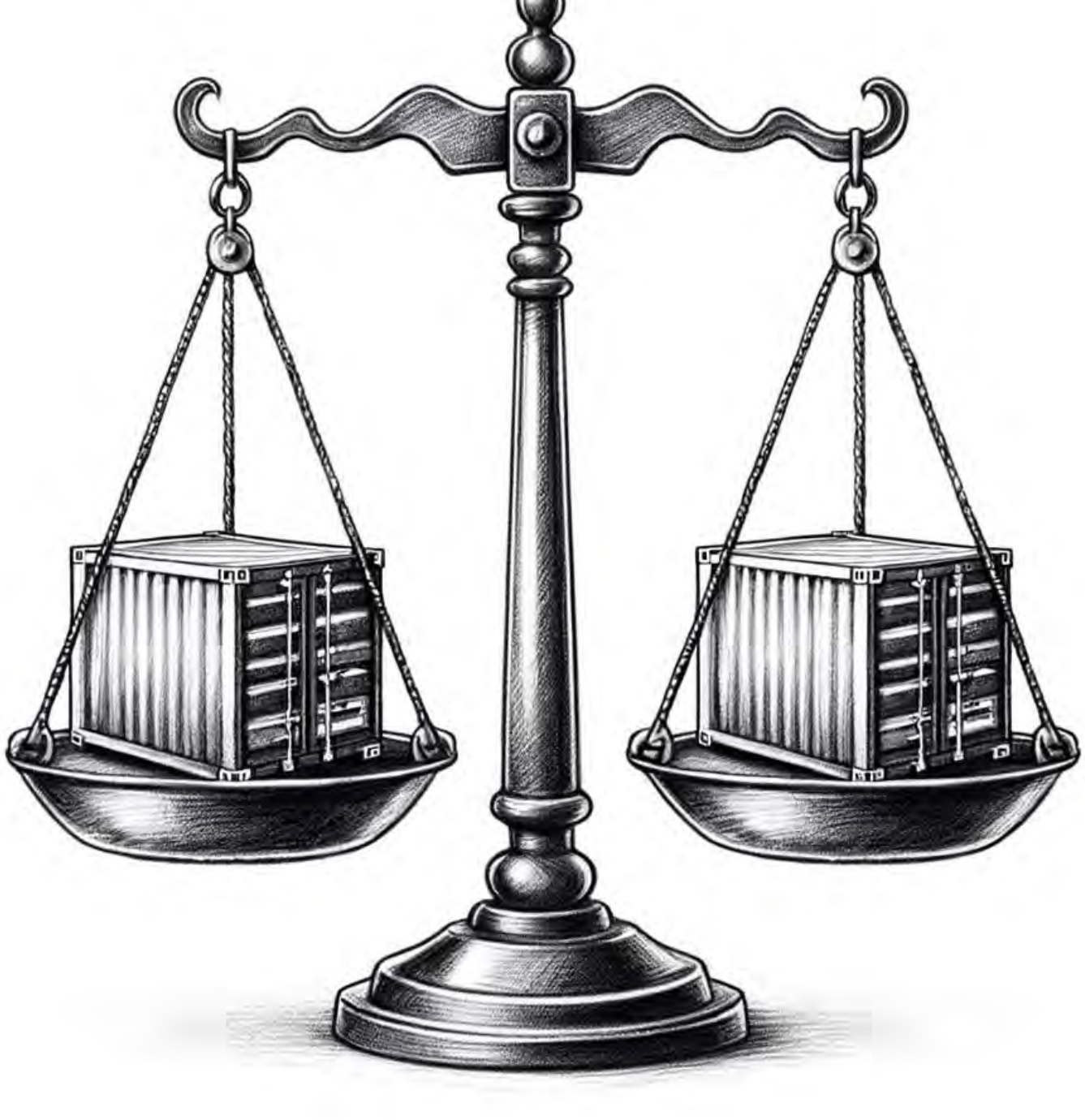


API gateway

Authorization and Authentication Servers

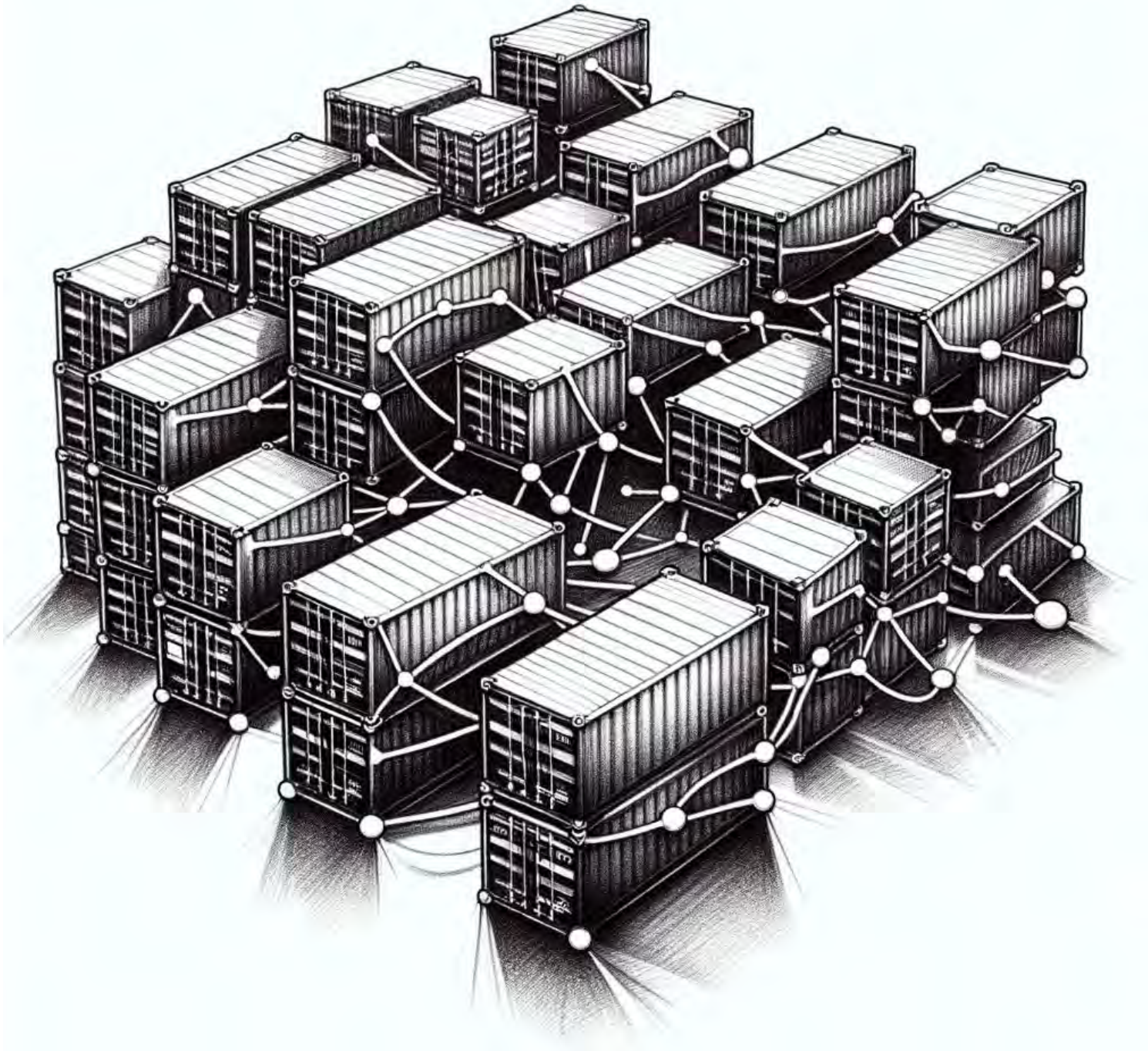


Load Balancers



Notification services

Pre-trained Model Services



What to Do with Stateful Services?

Hybrid microservice

- Stateful component
- Stateless component

CQRS

Synchronous Interaction

- The stateless component initiates a request.
- The stateful component processes the request and returns a result.
- The stateless component receives the response and completes the request processing.

Synchronous Interaction

- Load Balancers
- Timeouts and Retries

Race condition

Solving race condition problem

- Locks
- Idempotency
- Data Versioning

Asynchronous Interaction based on MQ (CQRS again)

Statelss component receves a request

Send command

Message waiting in queue

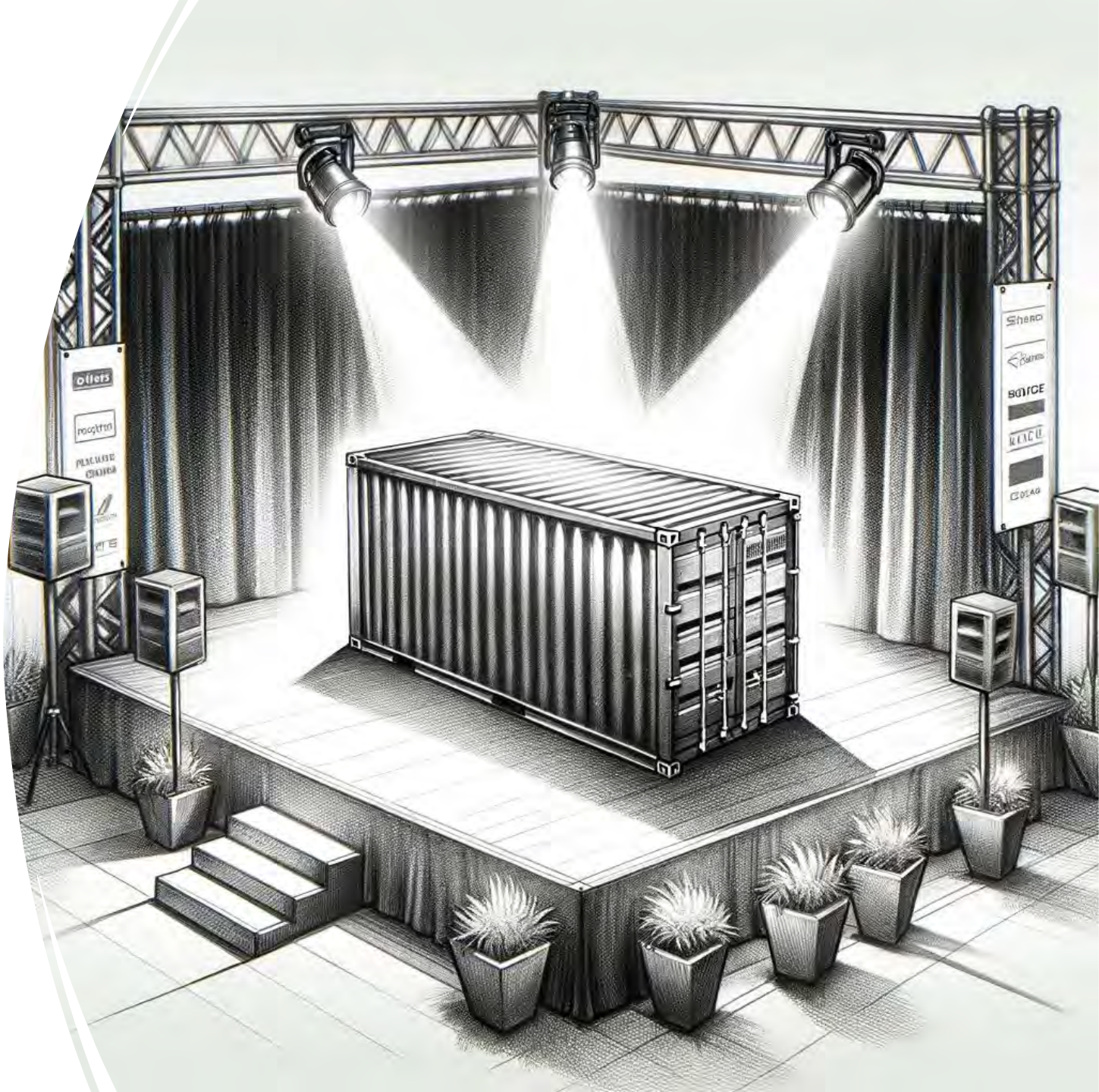
Stateful component process the message

Stateful component send confirmation message to statelss component.

Or optimistic response

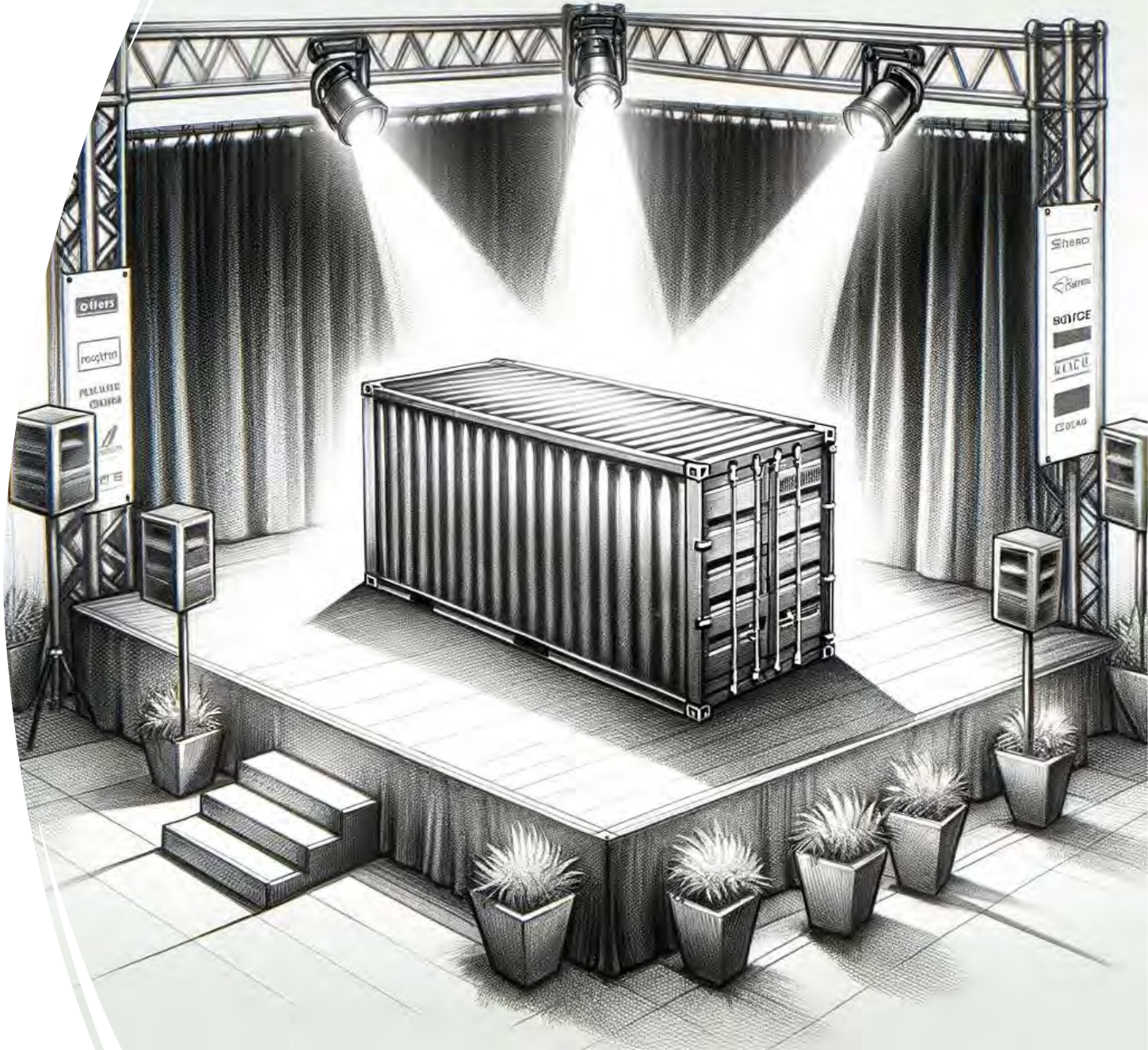
Event-Driven Architecture

- Event Generation
- Event Processing
- Optimistic response by design



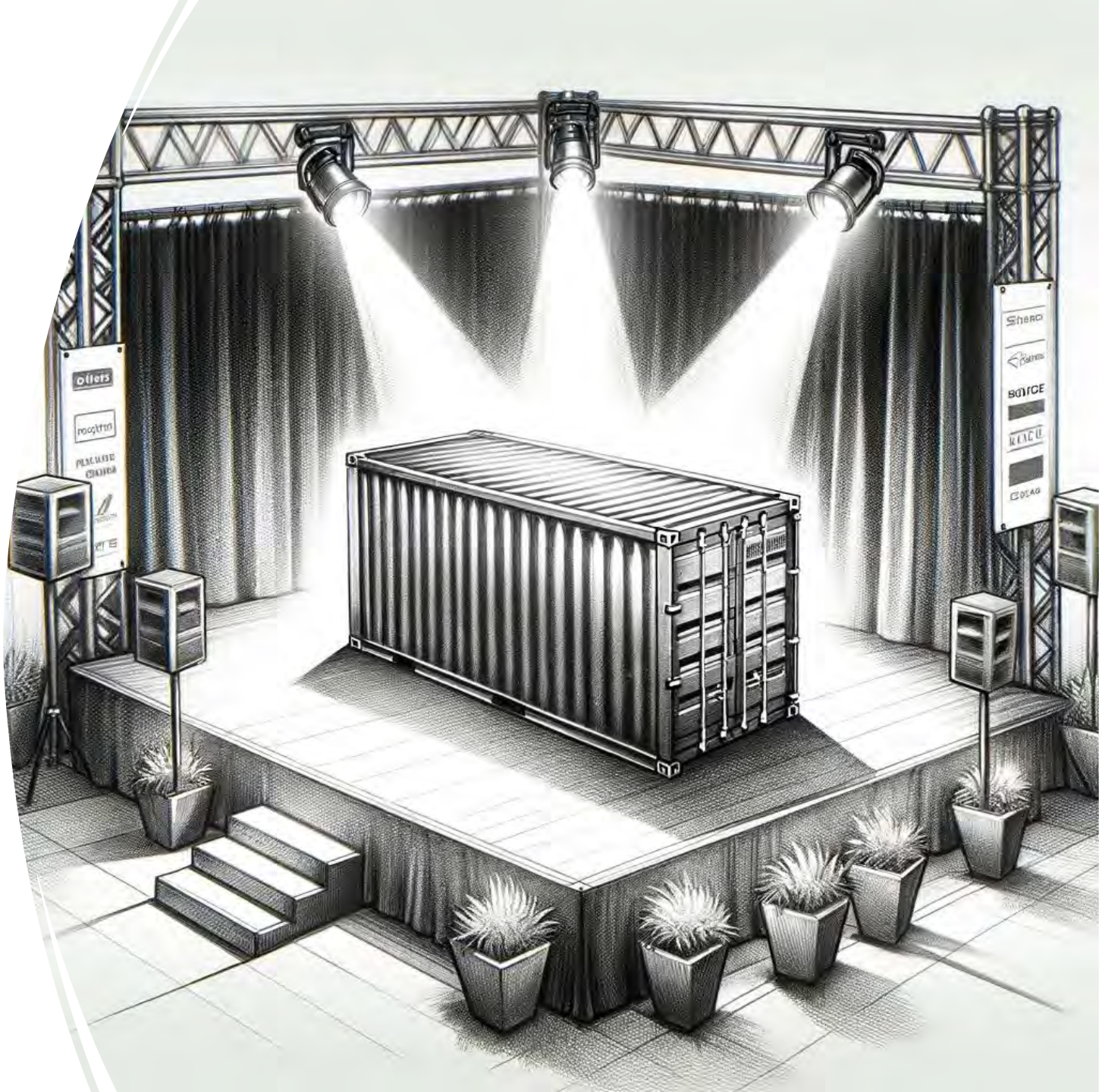
Event-Driven Architecture

- Reliability
- Scalability
- Flexibility



Event-Driven Architecture

- Monitoring and Alerting
- Dependency Management



Finally

- Statless better than stateful
- Hybrids
- CQRS
- Async interaction
- Event-Driven Architecture


THANKS