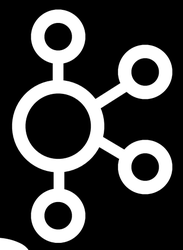
using Kafka for fun&profit

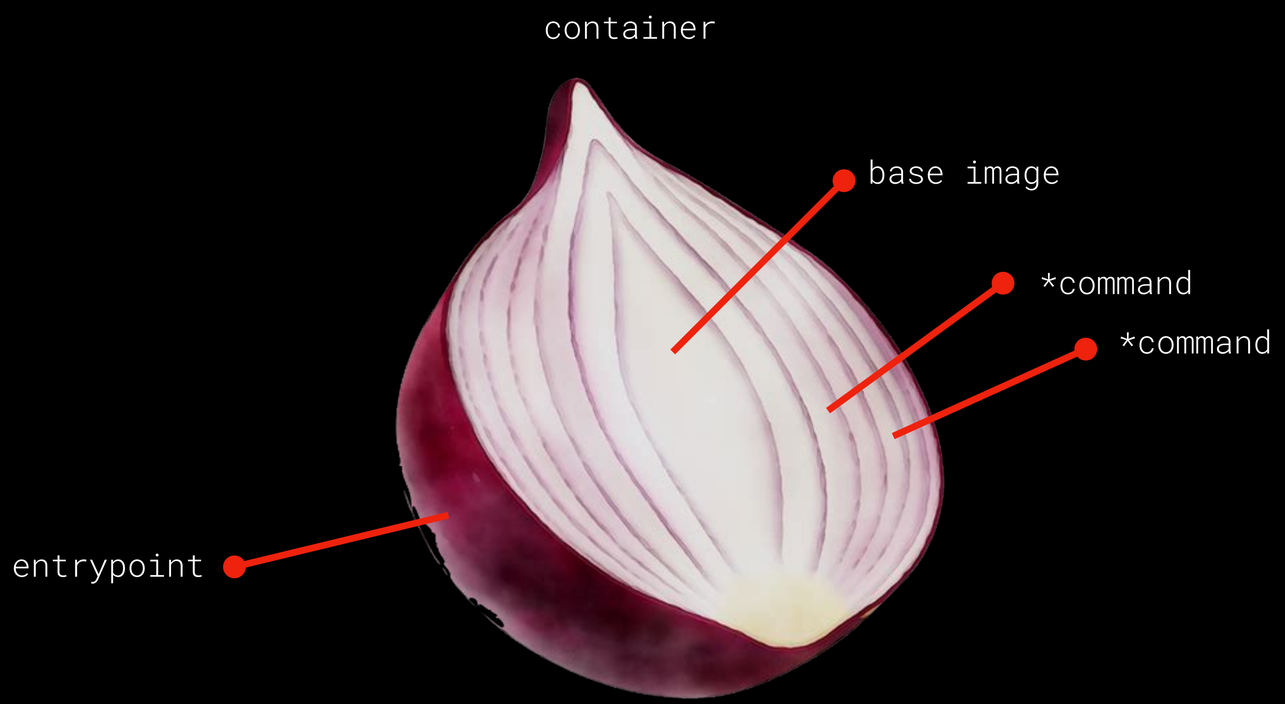


on example of building* Async State machine

@pelegrin 24.03.22 **moss**

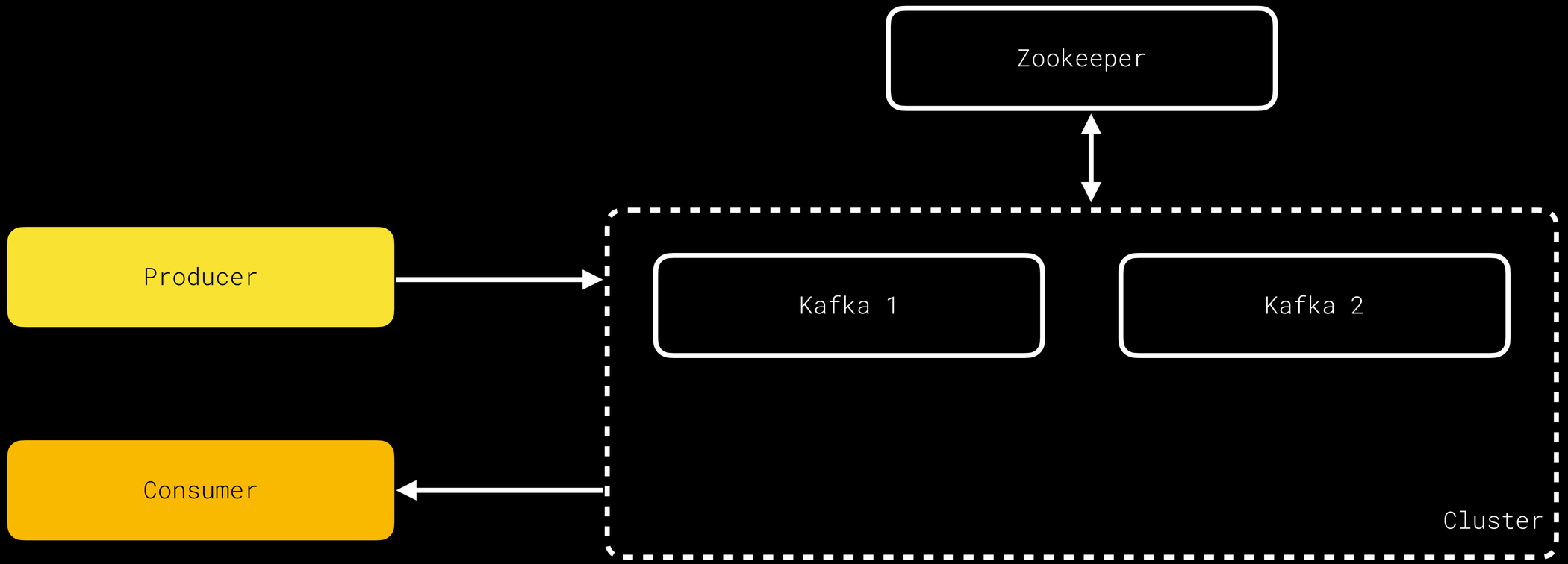
#environment

Creating own image. Kafka cluster



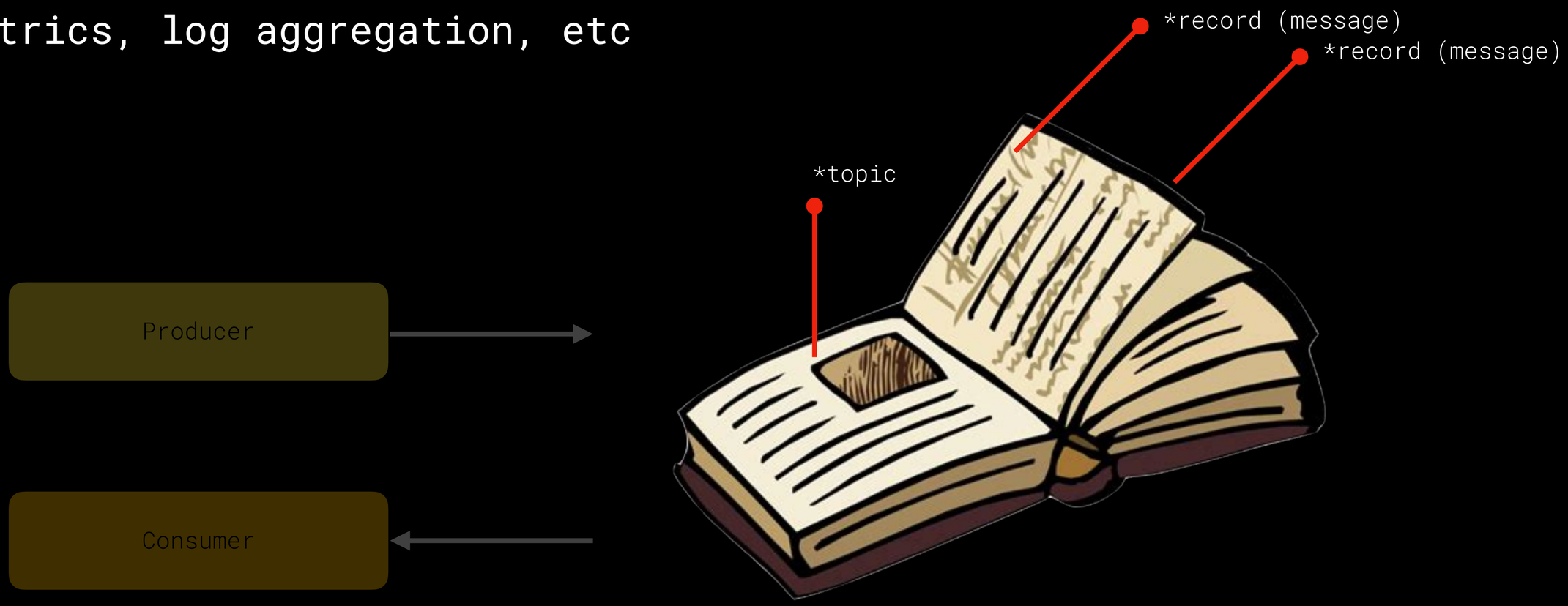
#environment

Creating own image. **Kafka cluster**



#kafka

Messaging (aka Message Broker)
Stream Processing
Event Sourcing
… metrics, log aggregation, etc



#async state machine

Benefits

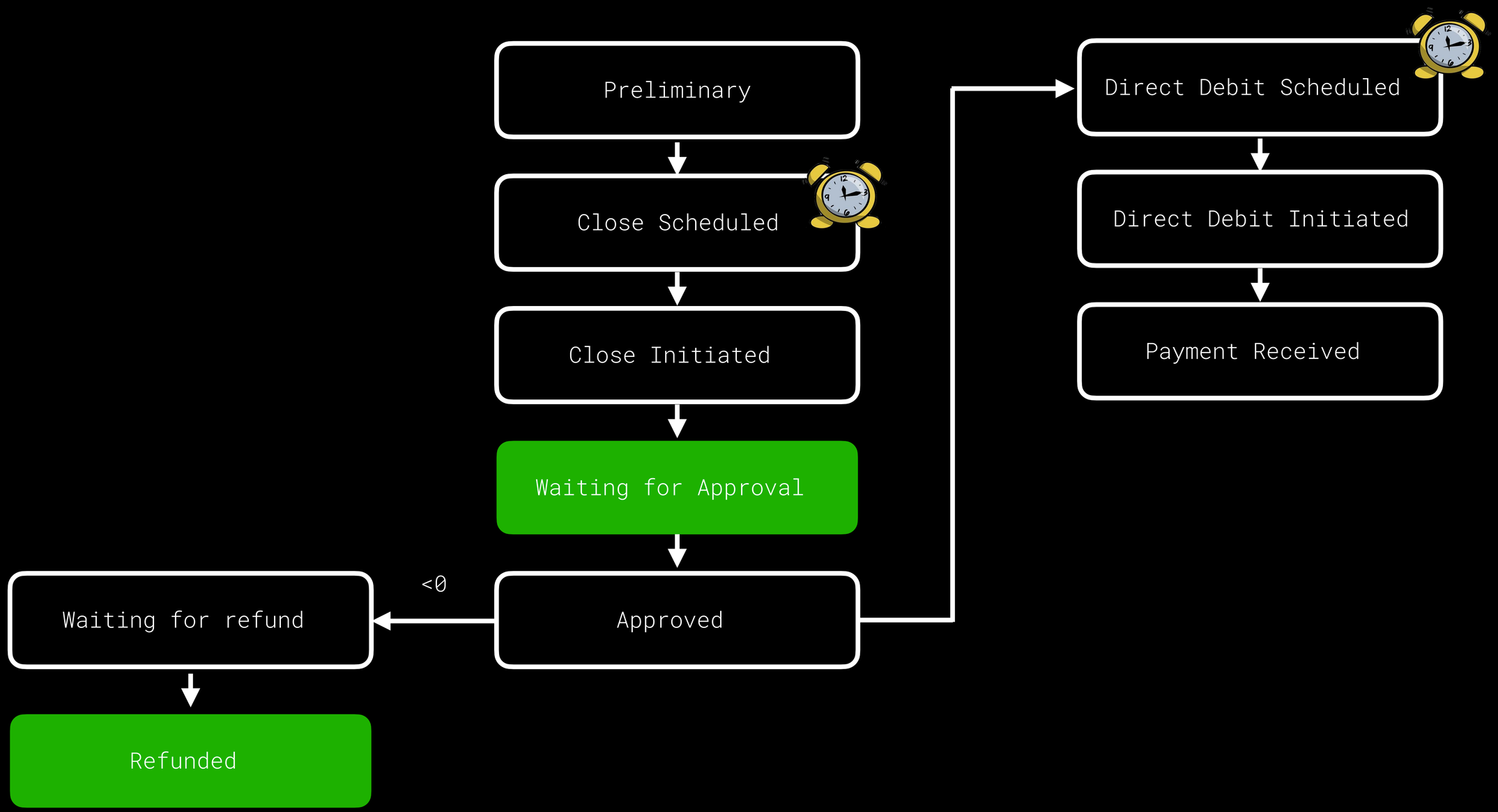
Fast implementation

Fits in k8s (multiple processors)

Distributed State machine

#async state machine

statement generation. **transaction-clearing &co**



#async state machine

implementation details

state machine event handling

IncomingMessageServiceImpl

```
@Override
@KafkaListener(topics = {
        STATEMENT_CLOSE_INITIATED,
        STATEMENT_APPROVED,
        STATEMENT_WAITING_FOR_APPROVAL,
        STATEMENT_DIRECT_DEBIT_INITIATED,
        STATEMENT_PAYMENT_RECEIVED_INITIATED,
        STATEMENT_PAYMENT_RECEIVED,
        STATEMENT_PAYMENT_INITIATED,
        STATEMENT_CLOSED
})
public void onStatementChangeEvent(ConsumerRecord<String, String> record, Acknowledgment acknowledgment) {
    handleMessage(record, Statement.class, acknowledgment, statementStatusChangeHandler::handle);
}
```

state machine event processing

StatementConfiguration

```
return (Statement statement) -> {
    workers.stream().filter(w -> w.isApplicable(statement)).findFirst().flatMap(worker ->
worker.process(statement)).ifPresent(statementGenerationHelper::nextStep);
    return MessageProcessingResult.processingSuccess();
```

to run processing from manual step
call this method as well

```
public void nextStep(Statement statement) {
    transactionHelper.withTransaction(() -> {
        statementRepository.save(statement);
        lifecycleEventRepository.insert(LifecycleEvent.builder()
                        .statementId(statement.getId())
                        .status(statement.getStatus())
                        .createUser(getActor())
                .build());
        repository.insert(OutgoingMessage.builder()
                .topic(statusToTopic.get(statement.getStatus()))
                .partitionKey(statement.getOrganisationId().toString())
                .payload(JsonMapper.DEFAULT.write(statement))
                .build()
        );
    });
```

#async state machine

Pros&**Cons**