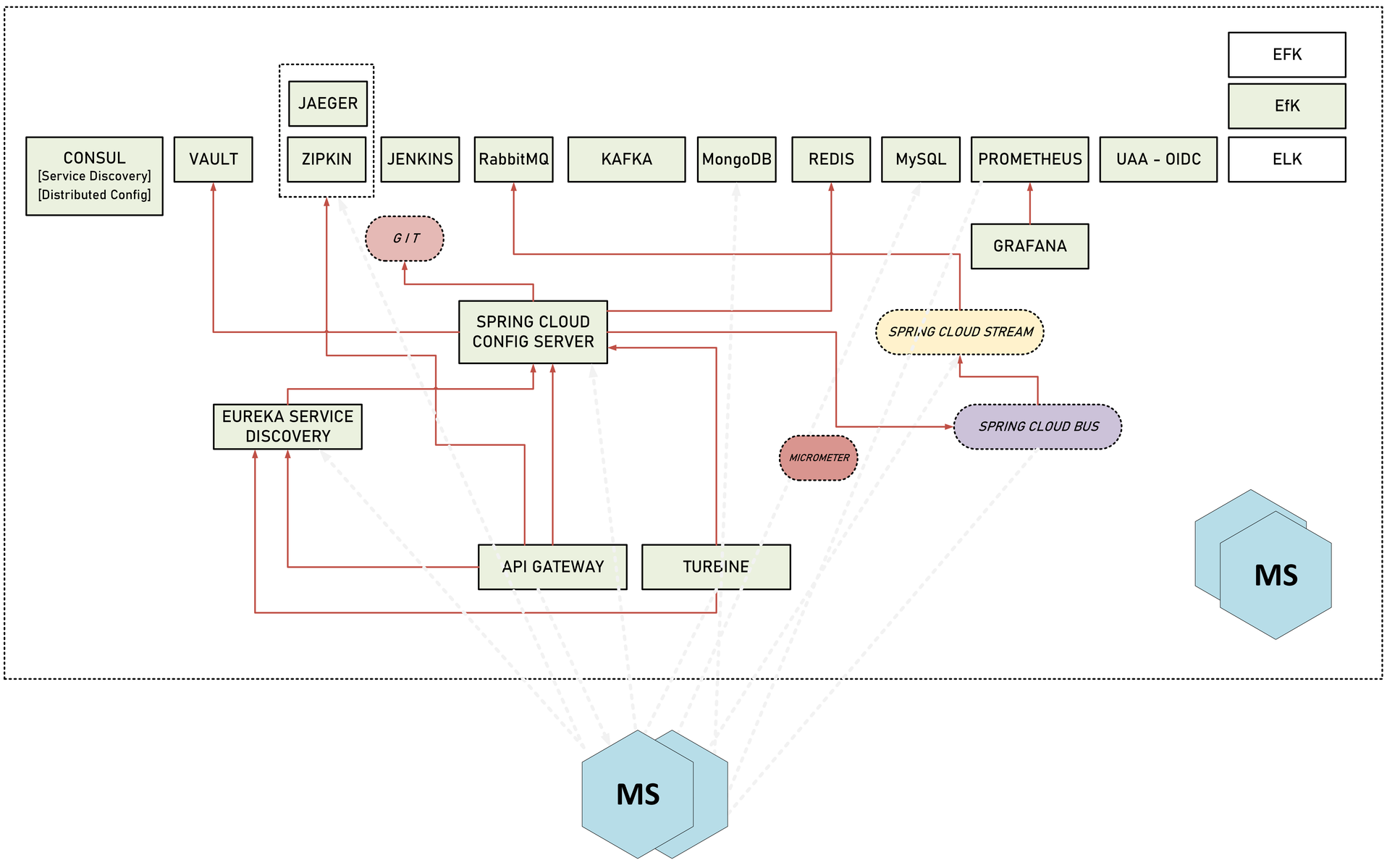
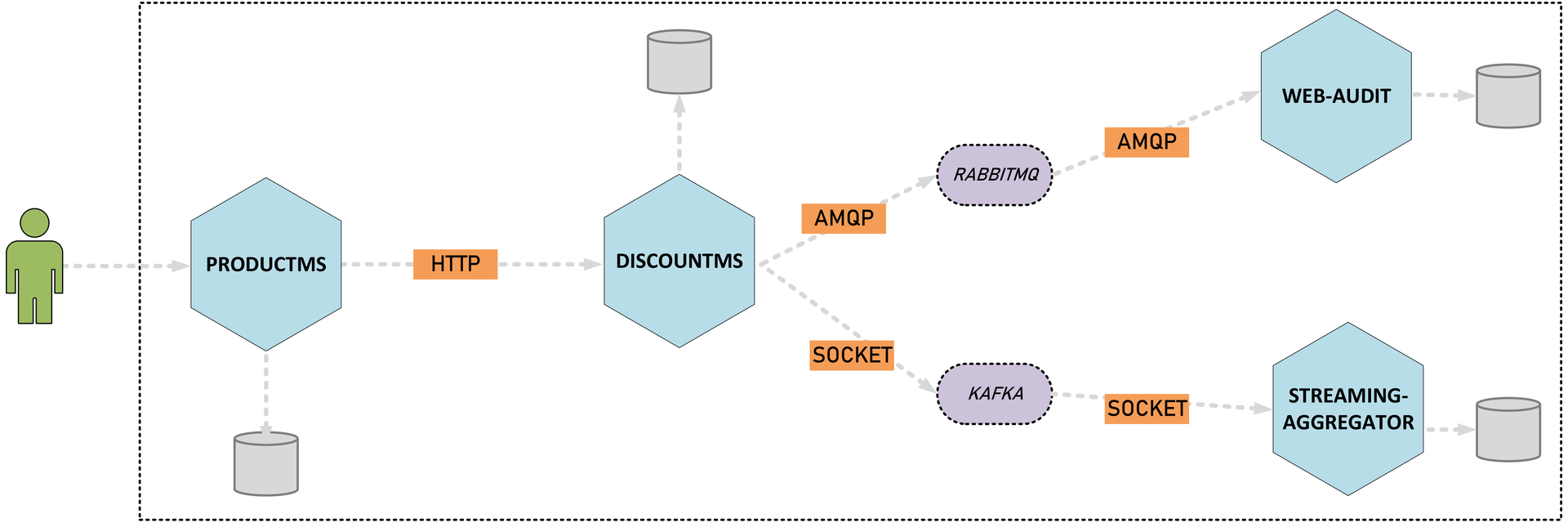
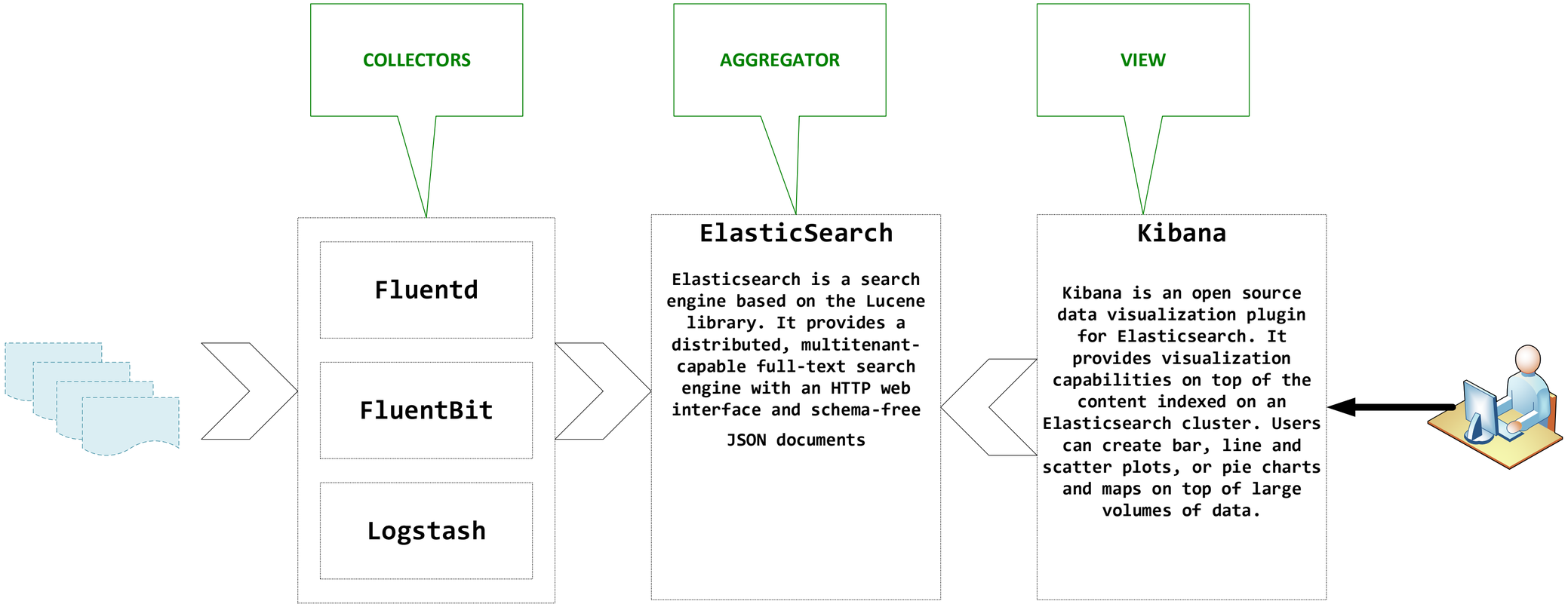
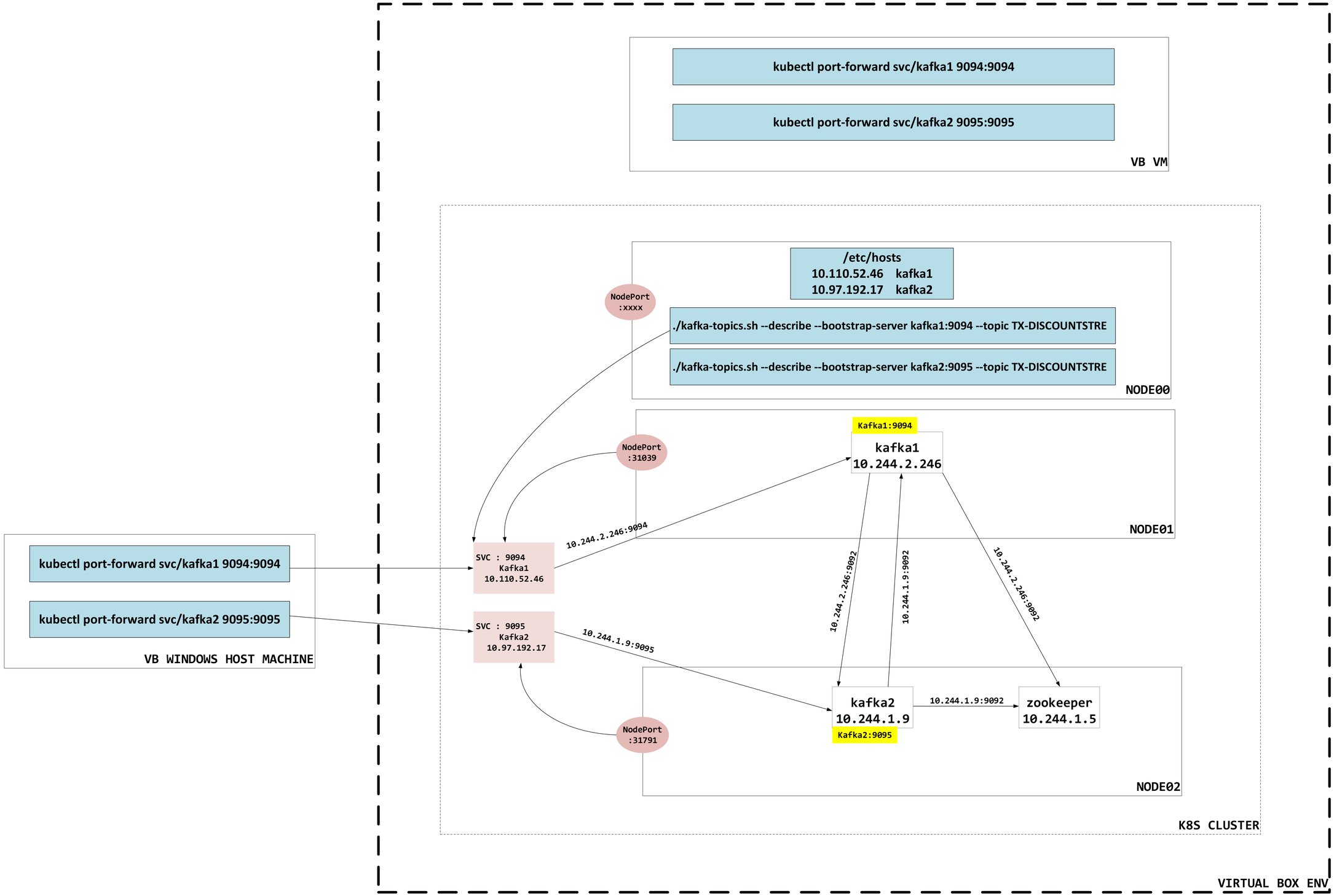
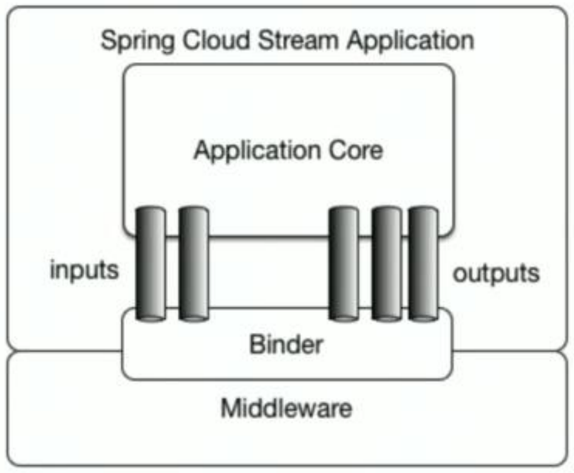

EFK
EfK
ELK
JAEGER
ZIPKIN
CONSUL
[Service Discovery]
[Distributed Config]
VAULT
JENKINS
RabbitMQ
KAFKA
MongoDB
REDIS
MySQL
PROMETHEUS
UAA - OIDC
GRAFANA
GIT
SPRING CLOUD
CONFIG SERVER
SPRING CLOUD STREAM
EUREKA SERVICE
DISCOVERY
SPRING CLOUD BUS
MICROMETER
API GATEWAY
TURBINE
MS
MS


PRODUCTMS
HTTP
DISCOUNTMS
AMQP
RABBITMQ
AMQP
WEB-AUDIT
SOCKET
KAFKA
SOCKET
STREAMING-
AGGREGATOR

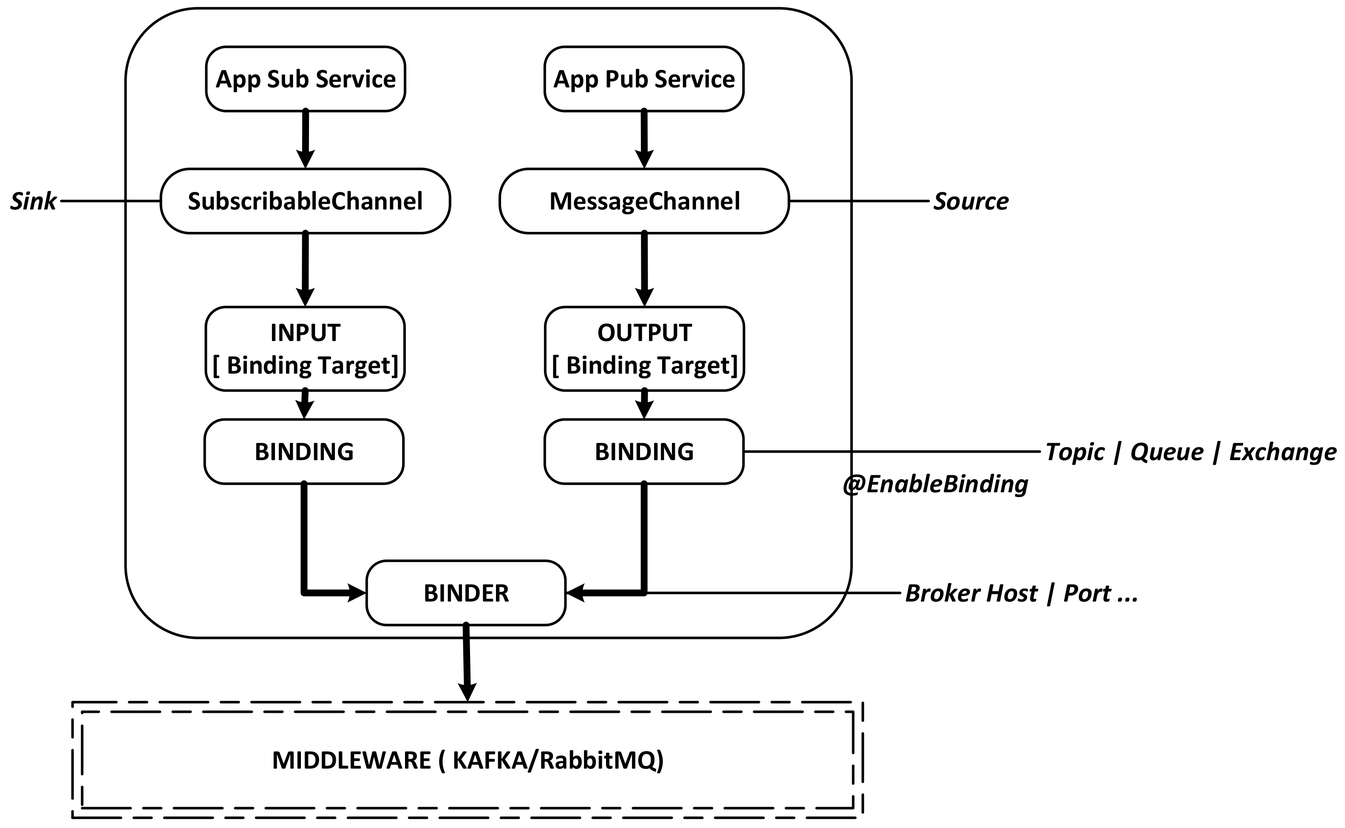

COLLECTORS
AGGREGATOR
VIEW
Fluentd
FluentBit
Logstash
ElasticSearch
Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents
Kibana
Kibana is an open source data visualization plugin for Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster. Users can create bar, line and scatter plots, or pie charts and maps on top of large volumes of data.

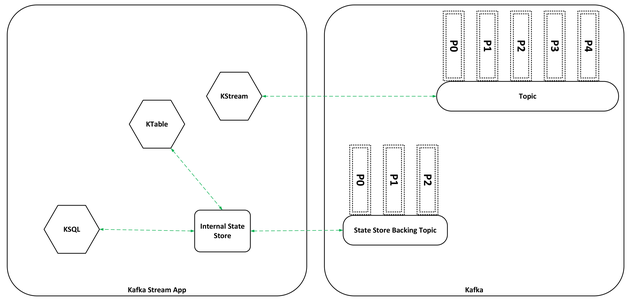

kubectl port-forward svc/kafka1 9094:9094
kubectl port-forward svc/kafka2 9095:9095
VB WINDOWS HOST MACHINE
VB VM
/etc/hosts
10.110.52.46 kafka1
10.97.192.17 kafka2
NodePort :xxxx
./kafka-topics.sh --describe --bootstrap-server kafka1:9094 --topic TX-DISCOUNTSTRE
./kafka-topics.sh --describe --bootstrap-server kafka2:9095 --topic TX-DISCOUNTSTRE
NODE00
Kafka1:9094
kafka1
10.244.2.246
NodePort :31039
NODE01
SVC : 9094
Kafka1
10.110.52.46
10.244.2.246:9094
SVC : 9095
Kafka2
10.97.192.17
10.244.1.9:9095
10.244.2.246:9092
10.244.1.9:9092
10.244.2.246:9092
kafka2
10.244.1.9
Kafka2:9095
10.244.1.9:9092
zookeeper
10.244.1.5
NodePort :31791
NODE02
K8S CLUSTER
VIRTUAL BOX ENV

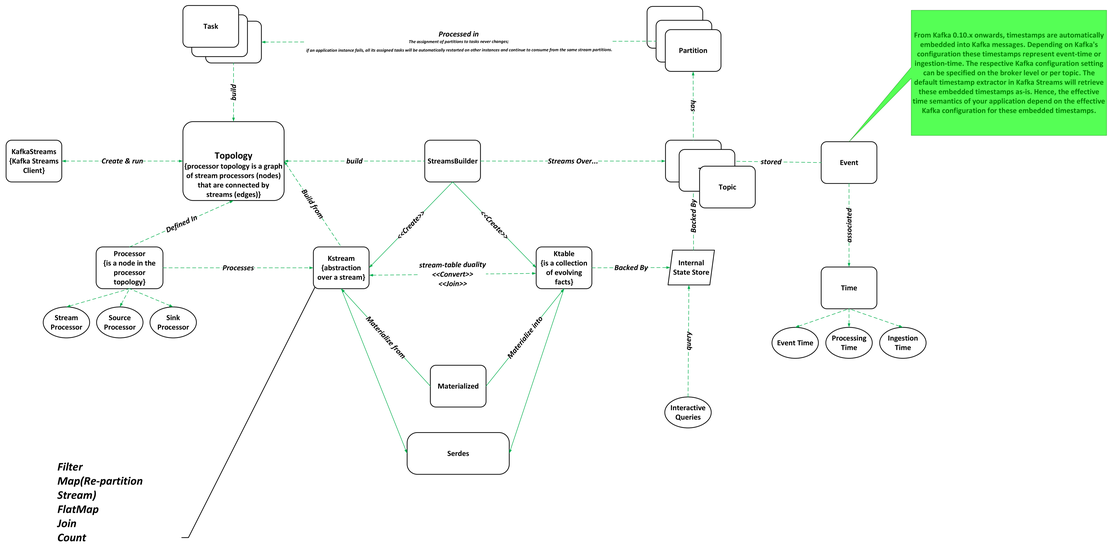
Spring Cloud Stream



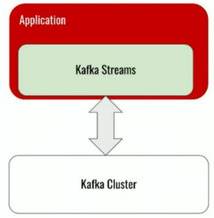
Kafka Binding Framework







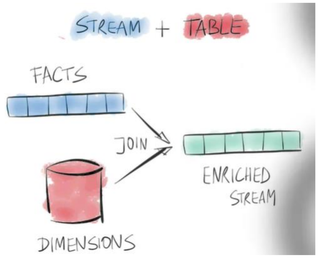
Kafka Streams



- Client library for stream processing
  - Embed stream processing features into regular Java applications (microservice model)
  - Create sophisticated topologies of independent applications
- Functional transformations via DSL:
  - Mapping, filtering, flatMap
  - Aggregation, joins (multiple topics)
  - Windowing
- Kafka-to-Kafka semantics
- One-record-at-a-time processing (no microbatching)
- Stateful processing support
- Transactions/exactly once

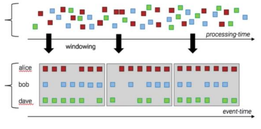
Kafka Streams - important concepts

- KStream
  - Record stream abstraction
  - Read from/written to external topic or produced from other KStream via operators such as map/filter
- KTable/GlobalKTable
  - Changelog stream abstraction (key is meaningful)
  - Read from external topic as a sequence of updates
  - Produced from other tables or stream joins, aggregations etc
- State Store
  - Key-value store for intermediate aggregation data, KTable materialized views, arbitrary key-value data produced during
  - Replicated externally
- Time windowing



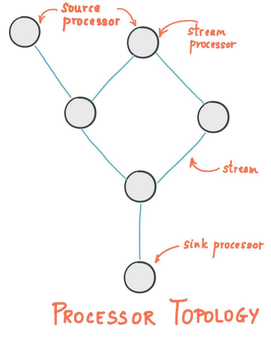
Windowing

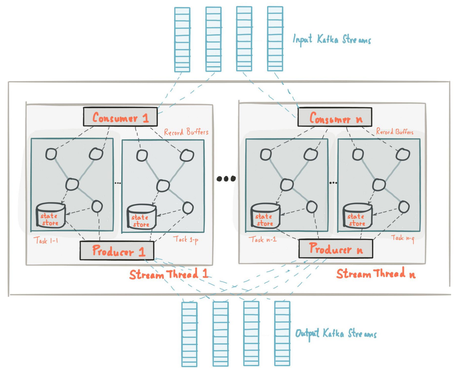
- You can window streams of data by time, i.e., group into buckets

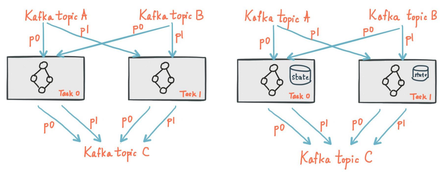


- Compare implementation without and with windowing

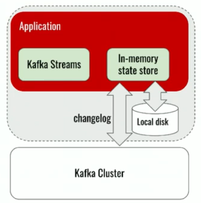
```
1 groupedBySongId.count("song-play-count");
2 groupedBySongId.count(TimeWindows.of(TimeUnit.MINUTES.toMillis(5)), "song-play-count-windowed");
```







Kafka Streams stateful processing (default stores)



- Pluggable state store model
- Key-value data store
- Default strategy:
  - In-memory (fast access)
  - Local disk (for fast recovery)
  - Replicated to Kafka (for resilience)
- Tightly integrated with Kafka: state updates are correlated with offset commits

- Changes propagated to changelog topic
- Stored locally for recovery/restart
- Fully stateless deployments require to replay the topic on restart/failover
- State store recovery can be optimized by providing access to stateful deployments