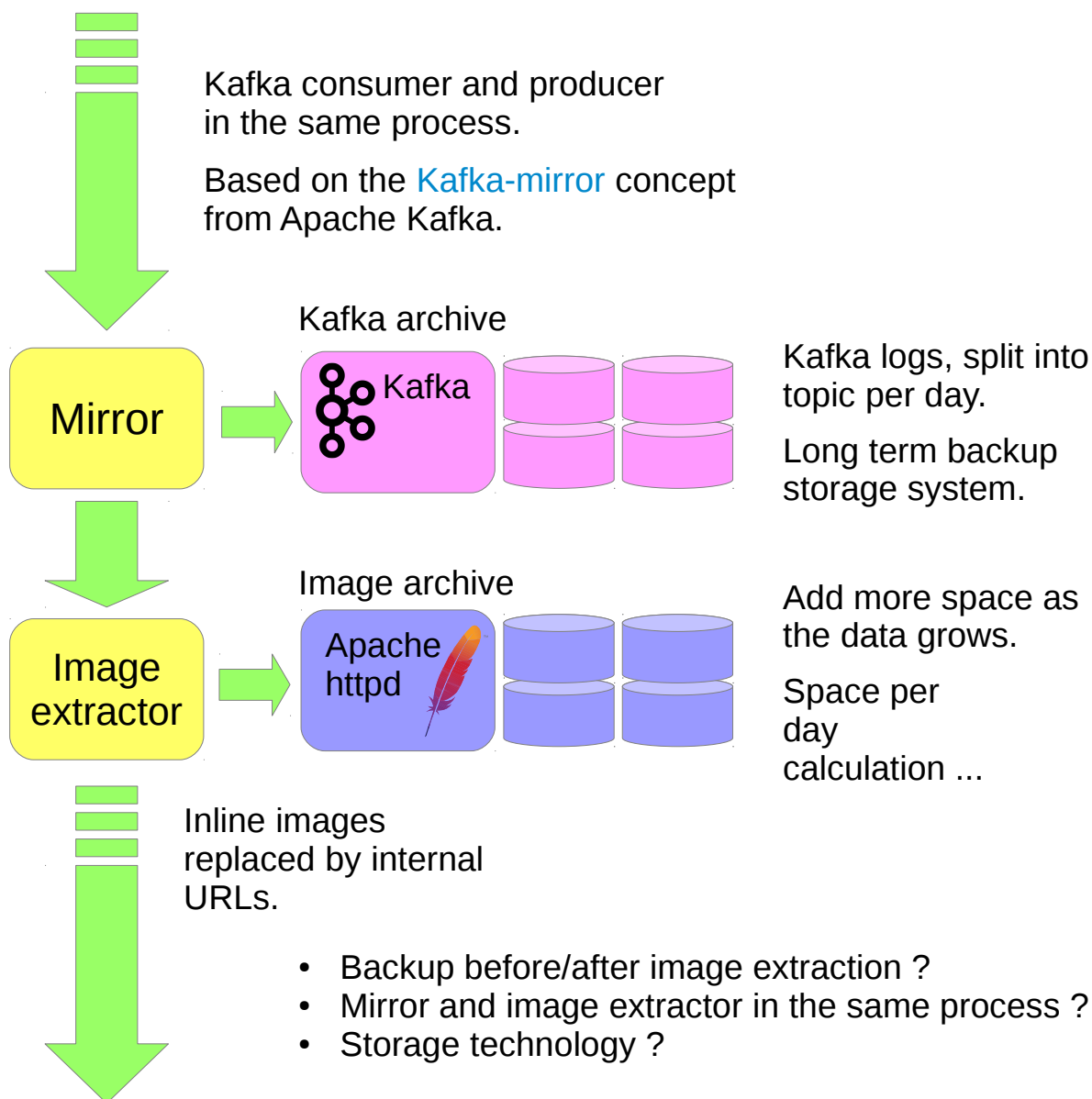
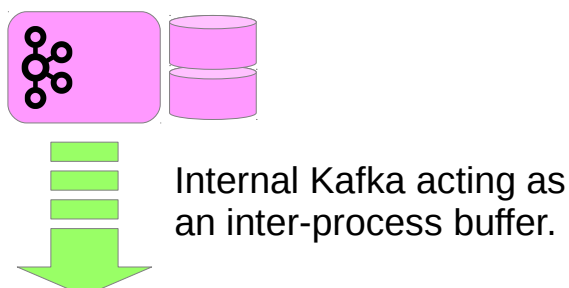


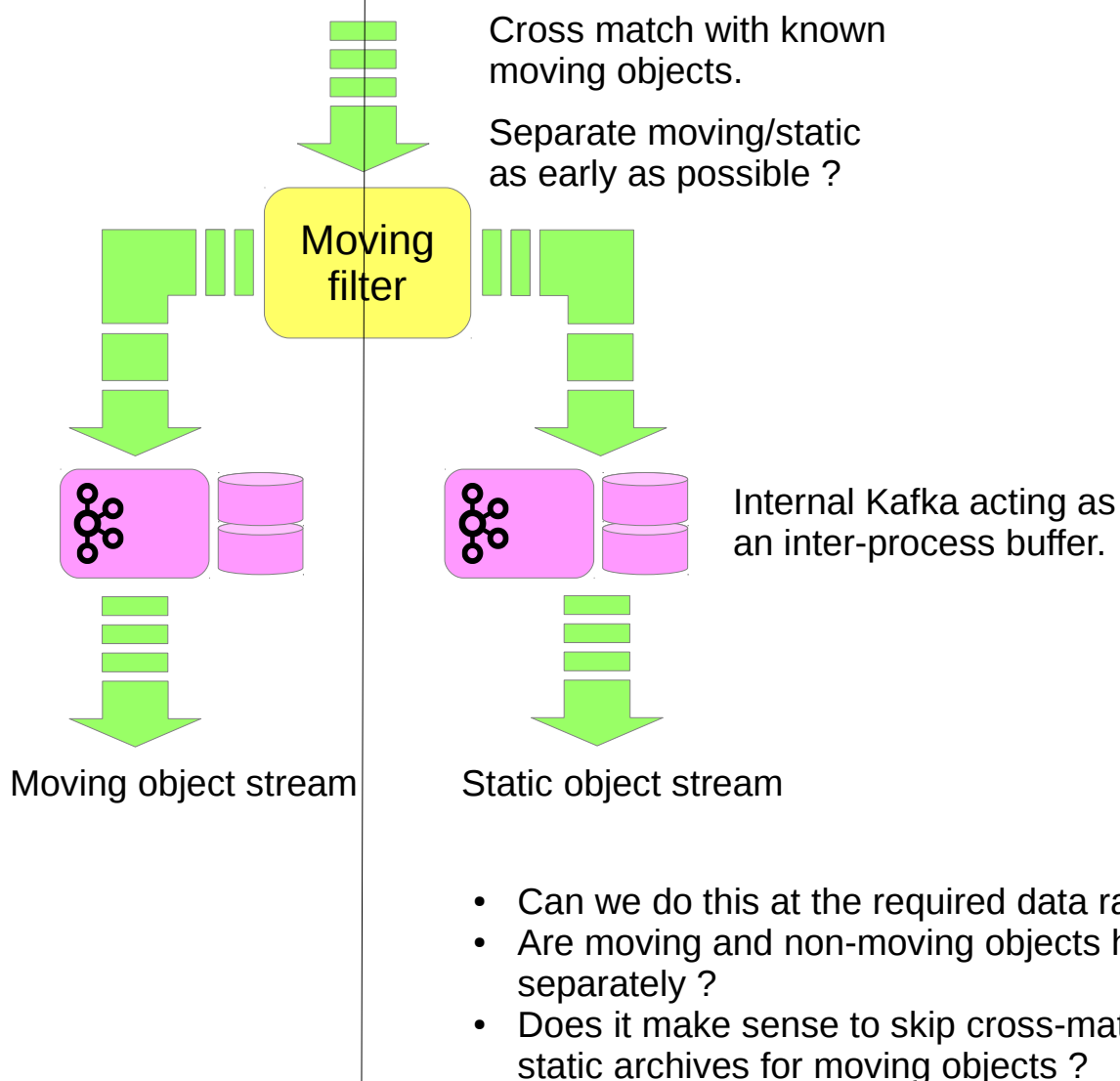
External receiver



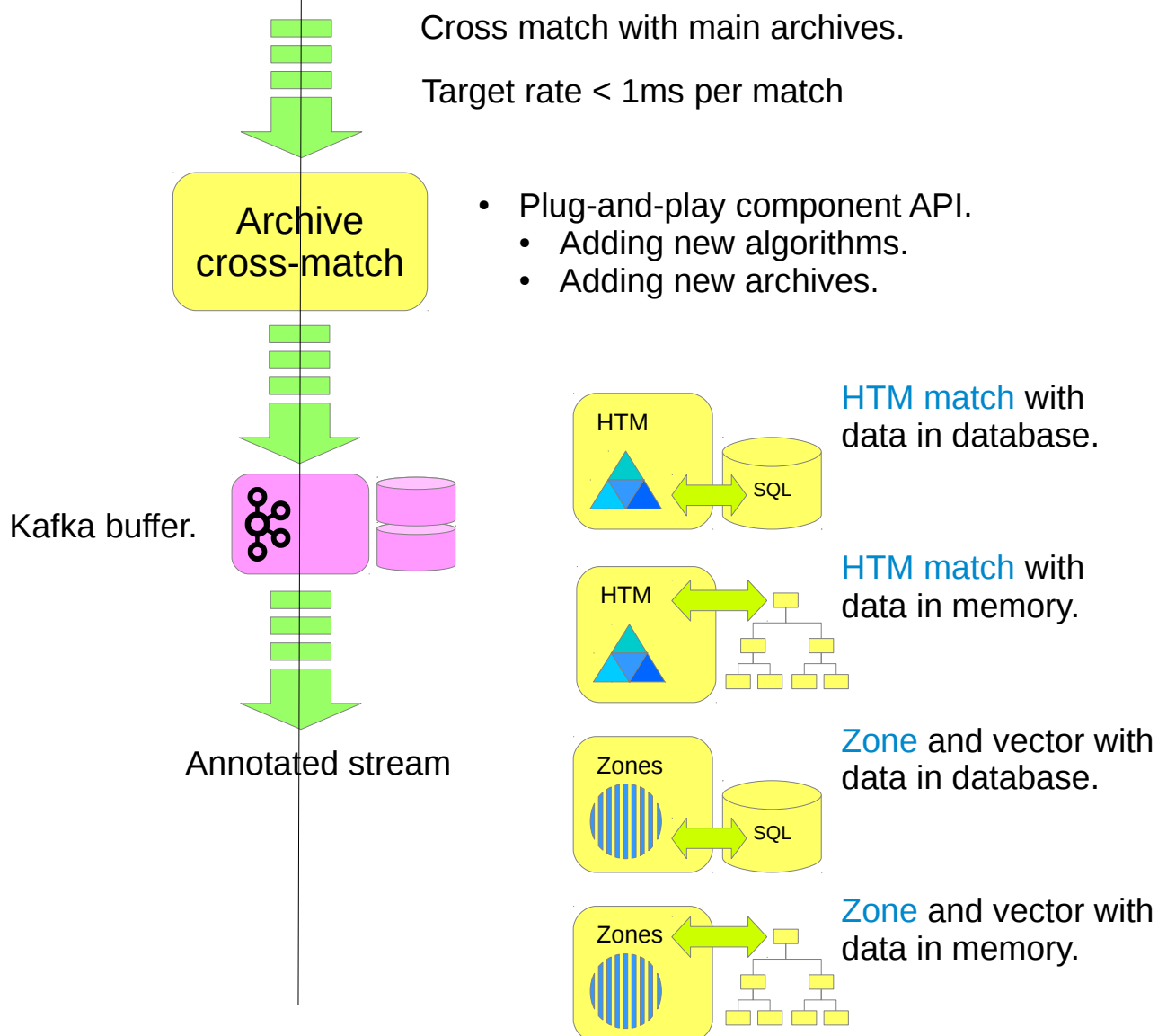
Internal processing



Moving objects



Archive cross-match



Multiple interfaces

JSON/REST webservice

- Control and configuration
- Cone search
- Cross match

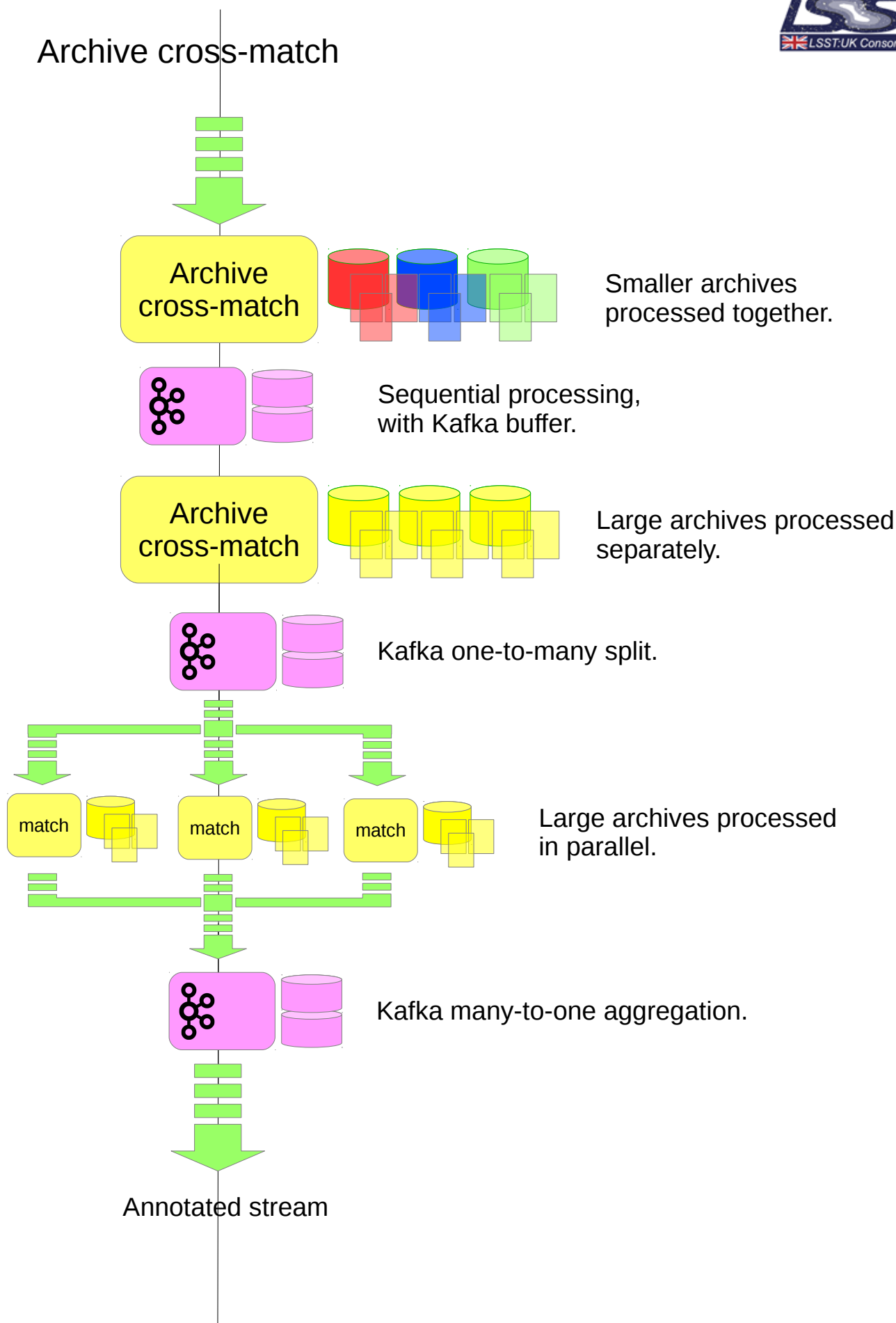
Kafka stream component

- Consumer/Producer API ?
- Kafka Connect component ?

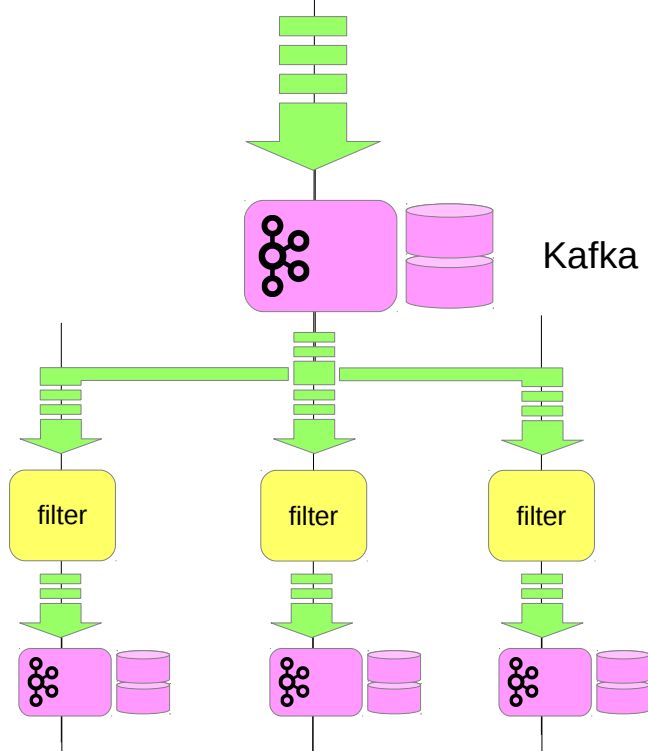
IVOA conesearch

- Publish as IVOA services ?

Archive cross-match



Selective filtering

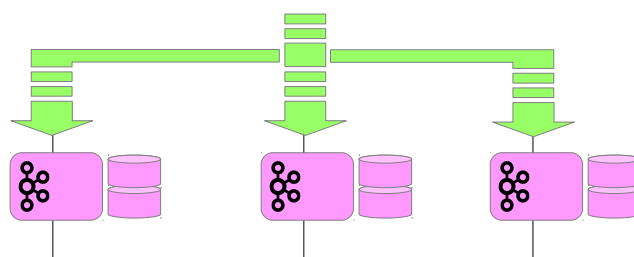
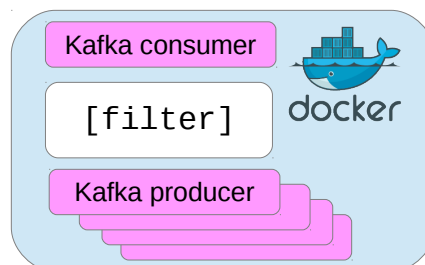


Kafka one-to-many splitter.

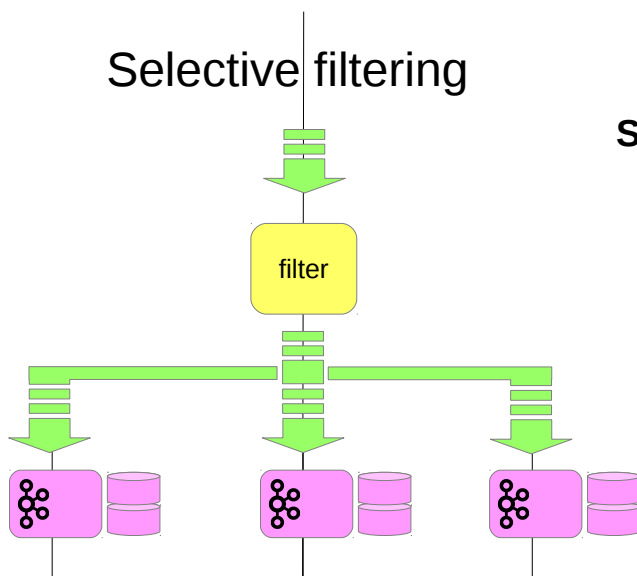
Filters create custom streams based on annotations.

- User defined code (Python).
- User defined code (Java).
- System filters (user = system).

- Multiple output filters



Selective filtering



Single output yes/no filter

```
if (test)
{
    output ..
}
```

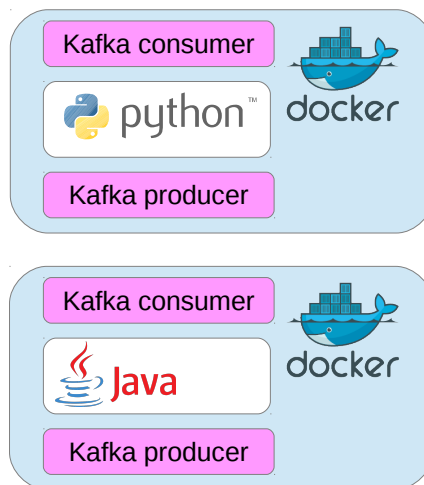
Dual output left/right filter

```
if (test)
{
    output ..
}
else {
    output ..
}
```

Multiple output filter

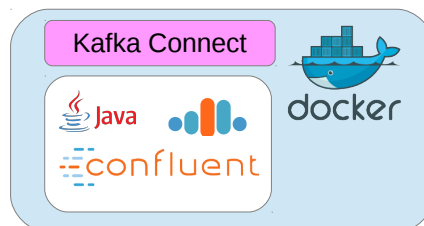
```
switch (test)
{
    case xx :
        output ..
    case yy :
        output ..
    case zz :
        output ..
    default :
        output ..
}
```

Standard Docker containers



Standard input and output interfaces.
Similar to the OGSA-DAI Activity class.

Confluent **Kafka Connect**

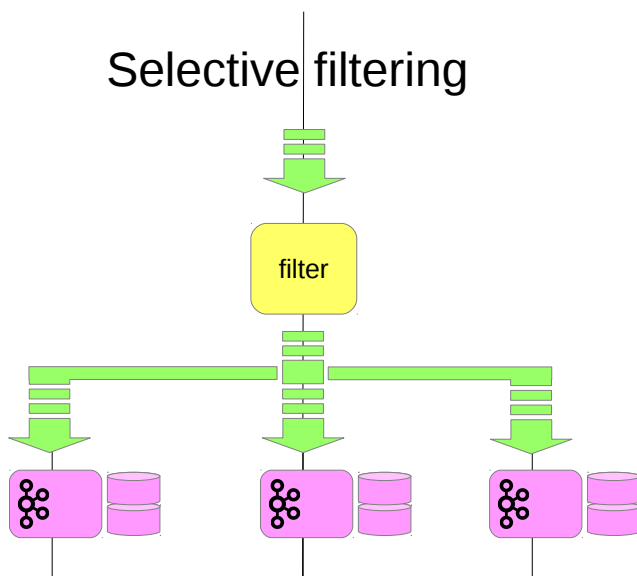


How much of the **Kafka Connect** stack do we use ?

How much do we create ourselves ?

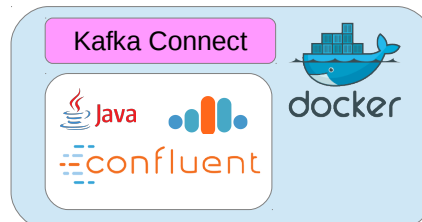
Kafka Connect provides a range of tools for data import and export .. but it adds yet another way of handling message schema.

Selective filtering



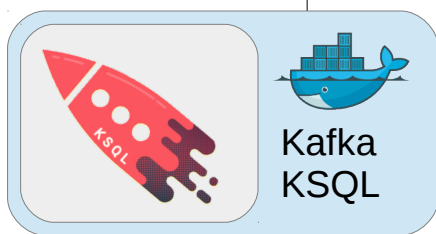
Kafka Connect

Import/export and schema



KSQL

Streaming SQL for Kafka

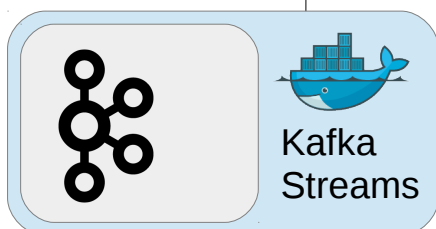


SQL query language

```
CREATE TABLE
    error_counts
AS SELECT
    error_code,
    count(*)
FROM
    monitoring_stream
WINDOW TUMBLING
    (SIZE 1 MINUTE)
WHERE
    type = 'ERROR'
```

Kafka Streams

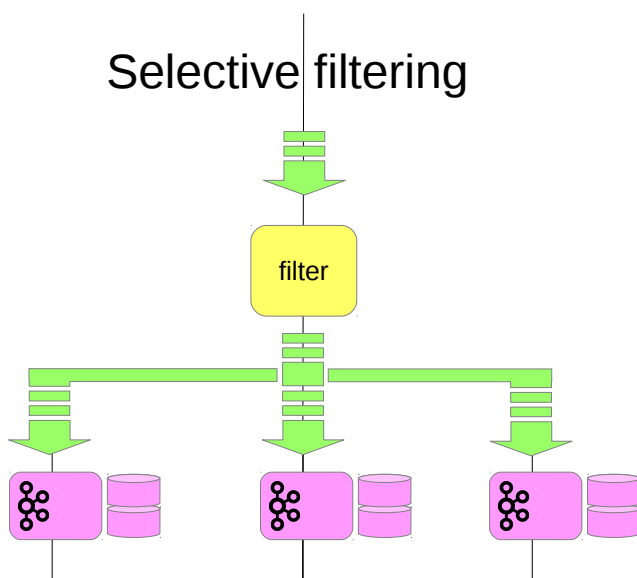
Data streaming with Kafka



Stream processing API

```
KStream<String, Long> stream = ...;

stream.foreach(
    new ForeachAction<String, Long>()
    {
        @Override
        public void apply(
            String key,
            Long value
        ){
            ....
        }
    }
);
```

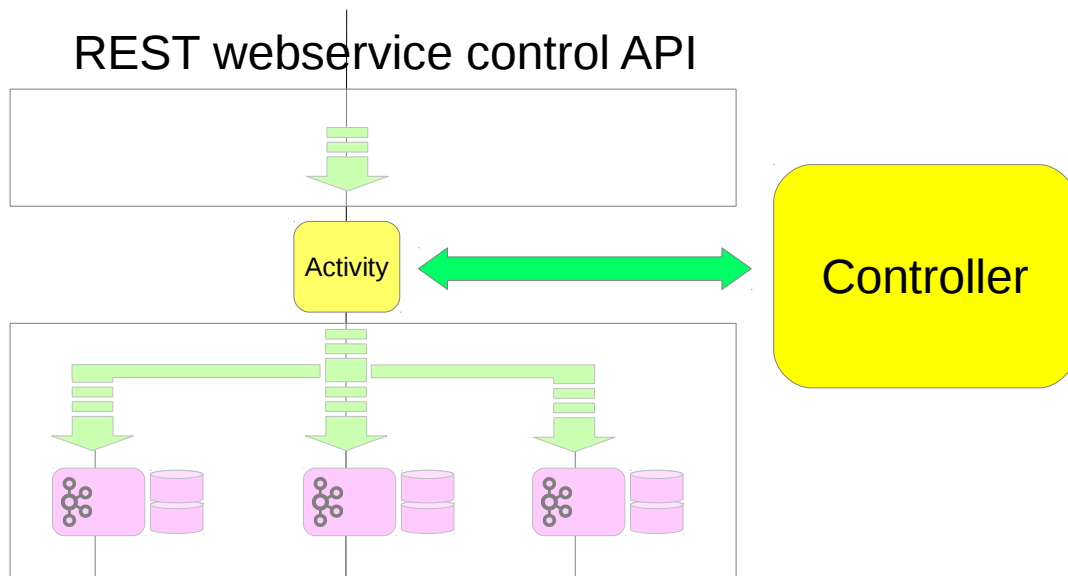


Internal event schema, critical set of attributes needed for filtering.

Event-expander, correlates mini-event with original event and creates a new event with additional params.

HTTP webservice with in memory cache ?

memcache ?



Spring-cloud ... yet another framework
Very easy to start .. harder to control the direction.

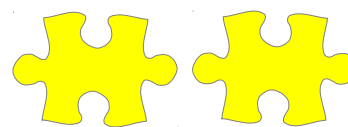
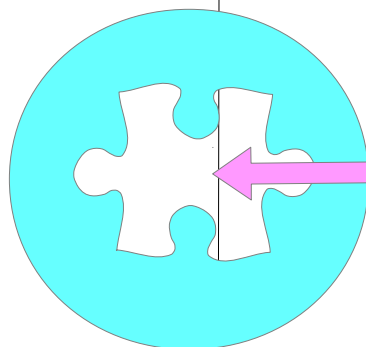
Our own REST API for deploying and controlling Kaffa nodes

Separation of concerns

- Processor Activity interface
- Command and control API

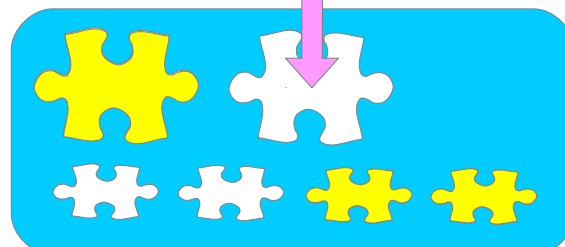
Internal Activity API insulates end user code from external command and control framework.

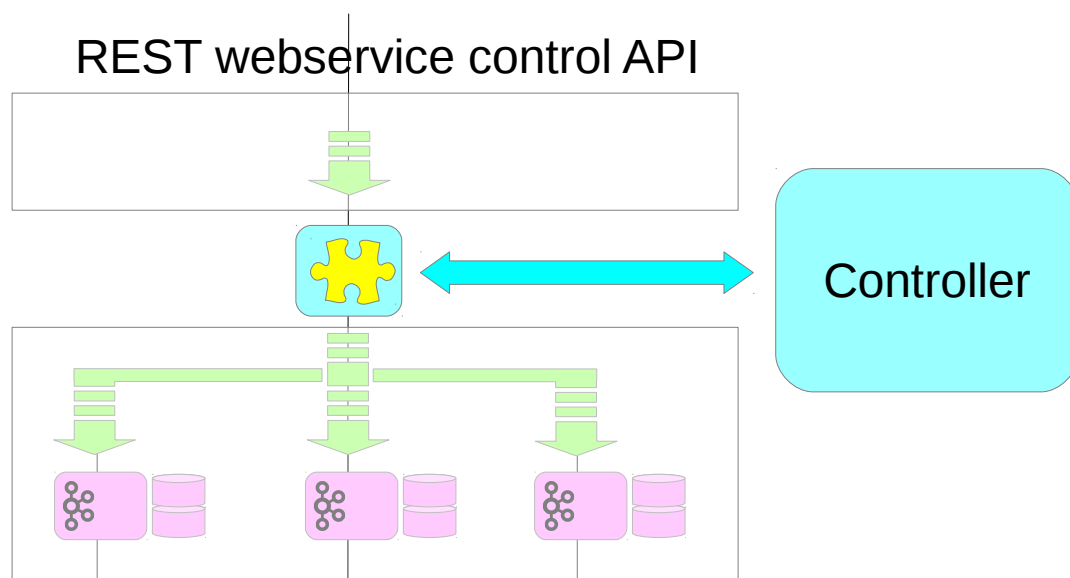
Deployment environment



End user code Activities

Deployment environment





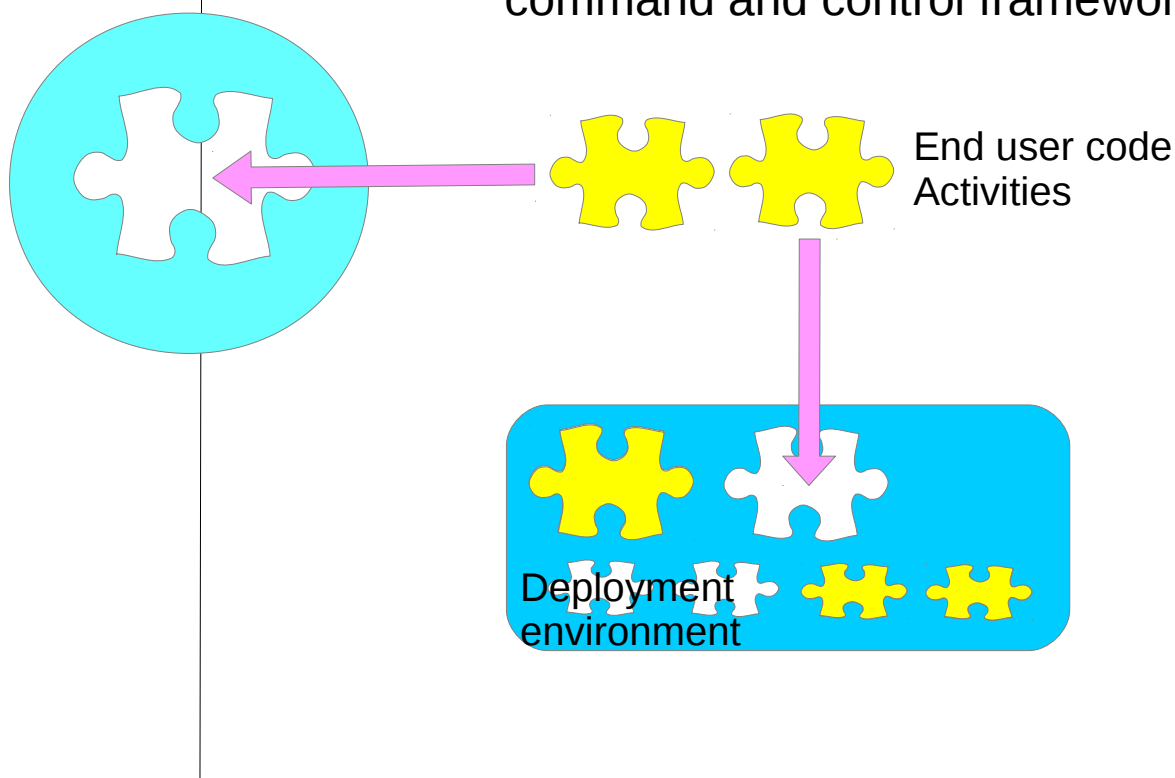
REST API for deploying and controlling Kaffa nodes

Separation of concerns

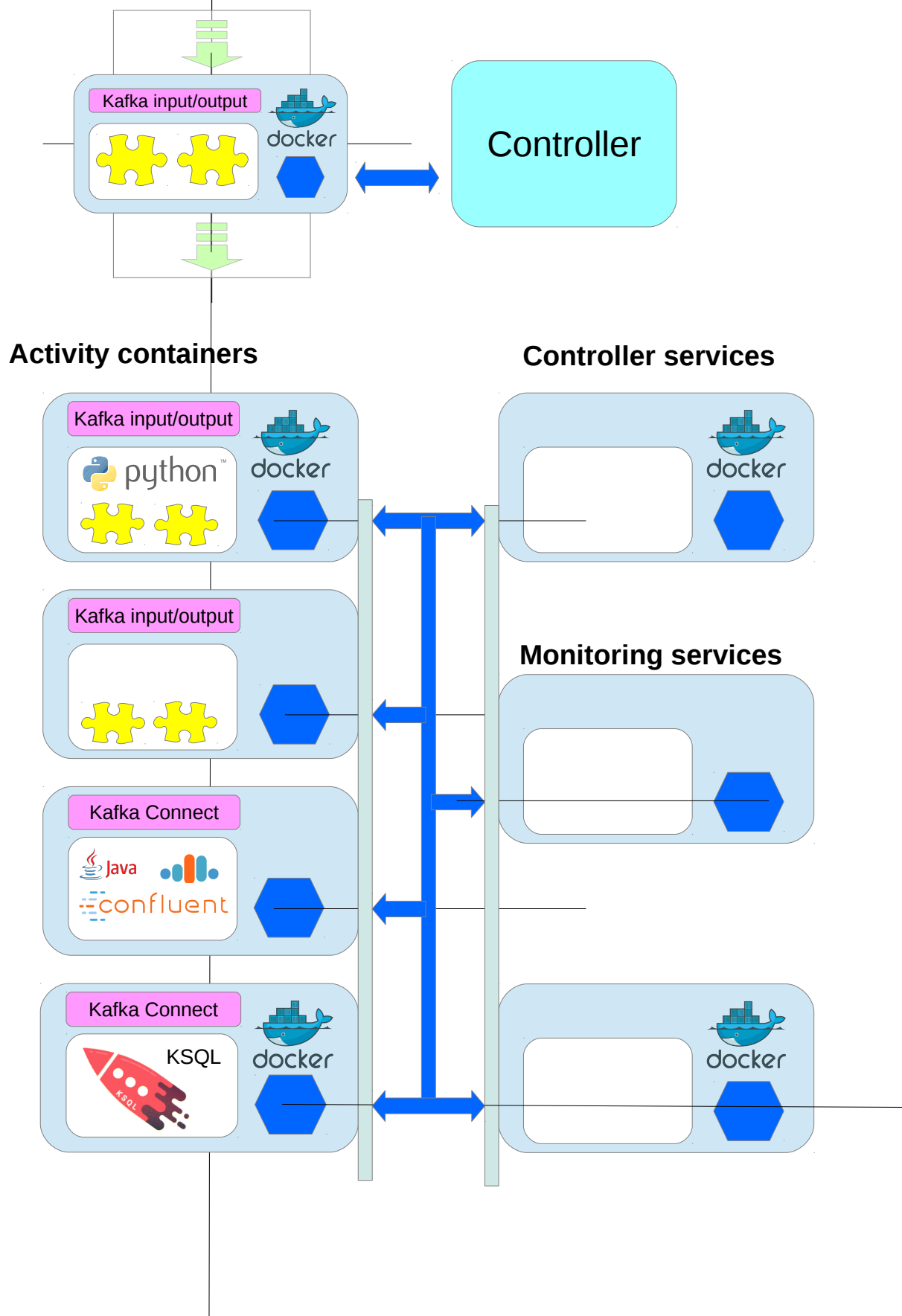
- Processor Activity interface
- Command and control API

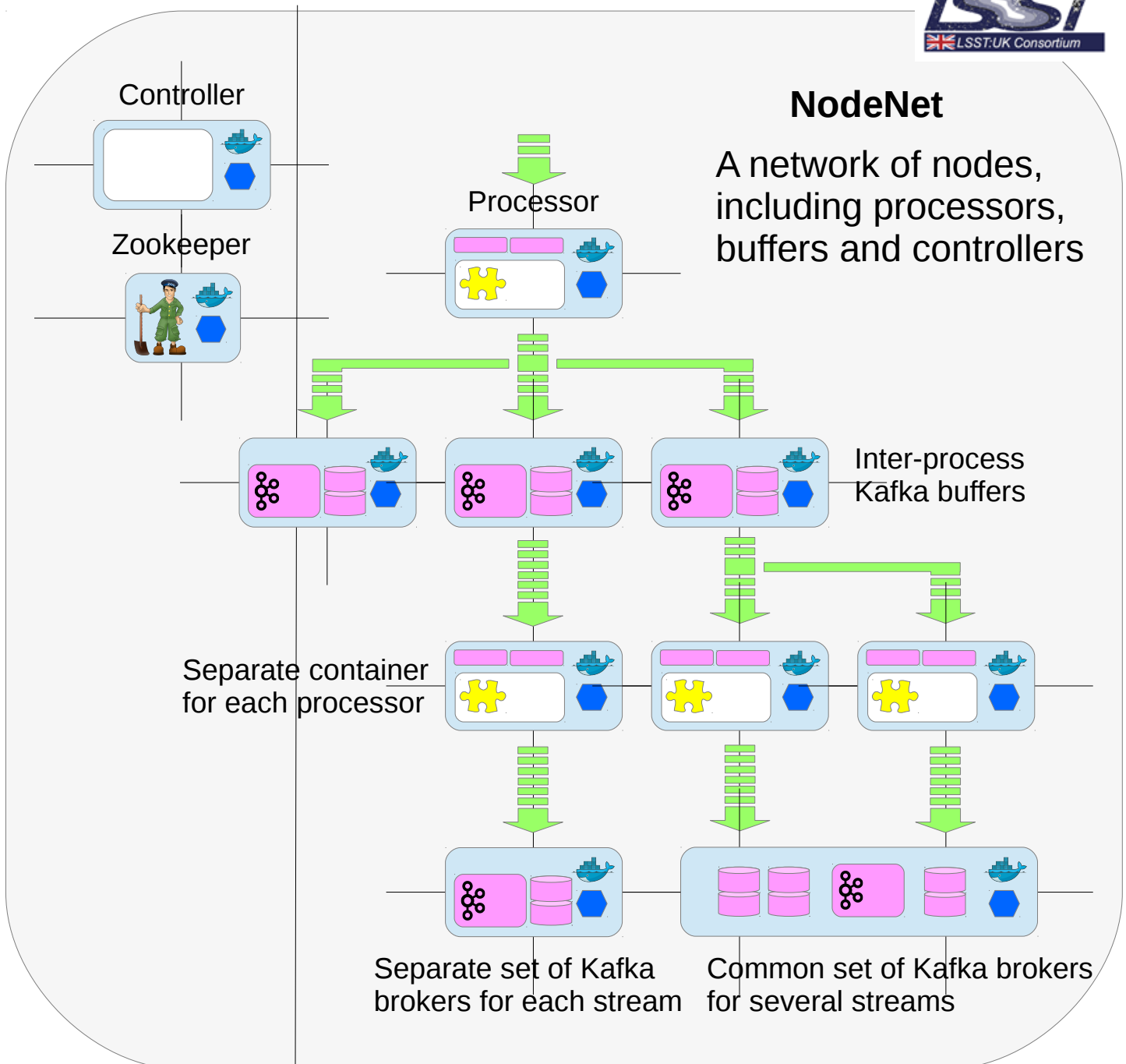
Internal Activity API insulates end user code from external command and control framework.

Deployment environment



REST webservice control API





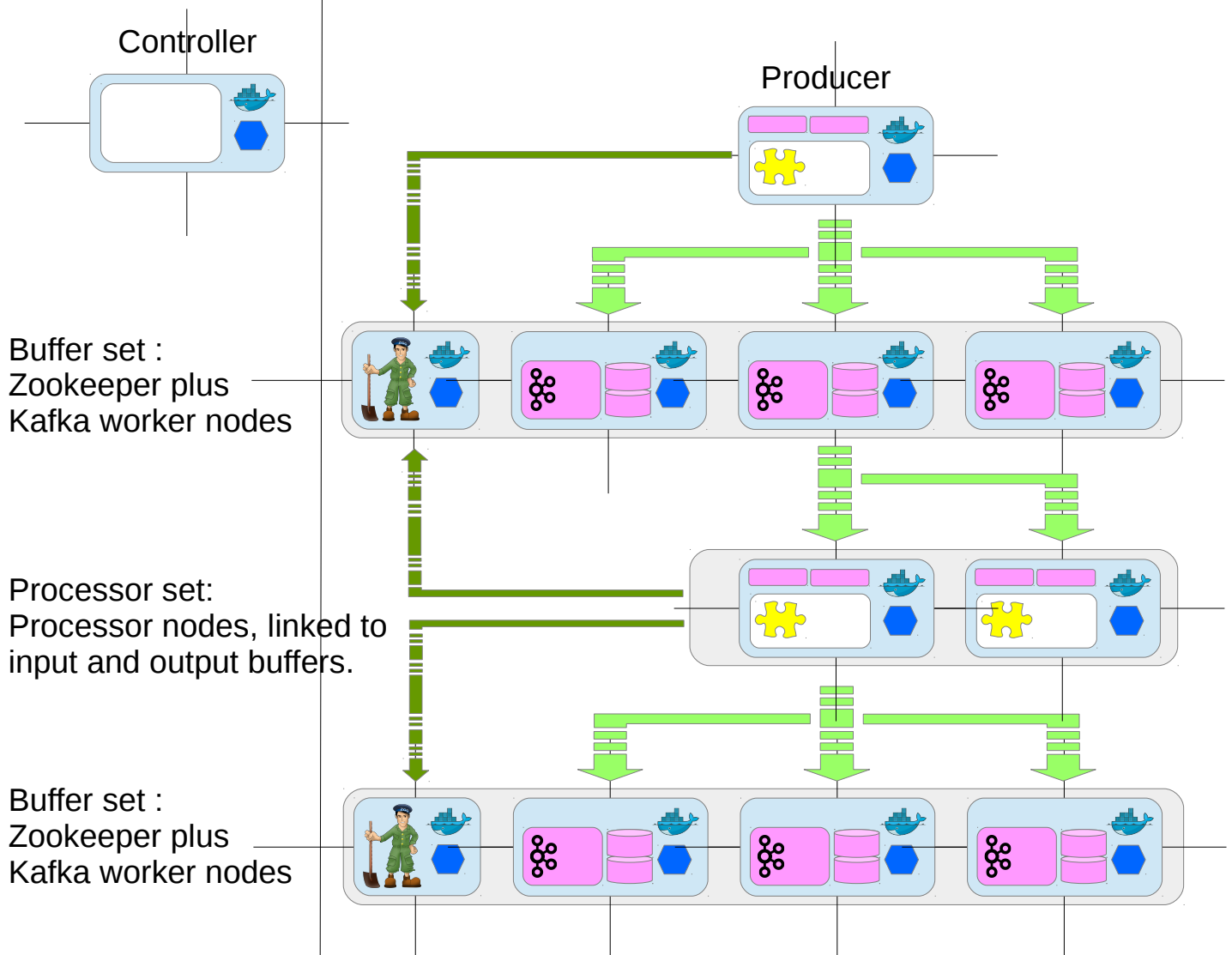
We need to be able to deploy multiple NodeNet networks.

Development, testing and live services.

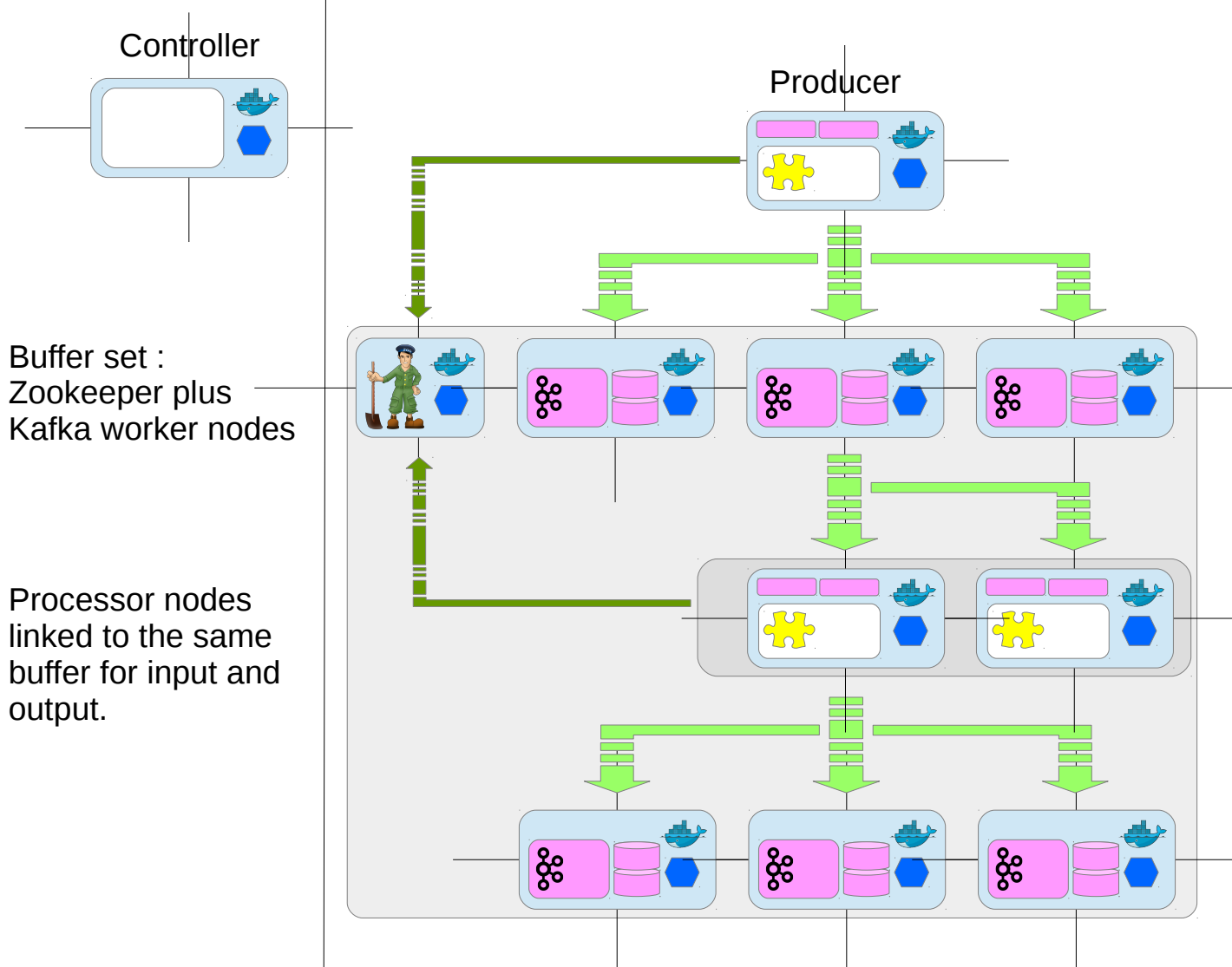
Single VM local instances.

Multiple physical machine deployments.

Multiple Buffer sets, each with its own Zookeeper and Kafka worker nodes

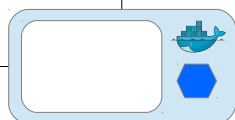


Shared Buffer sets, common Zookeeper and Kafka worker nodes



Multiple Buffer sets, each with its own Zookeeper and Kafka worker nodes

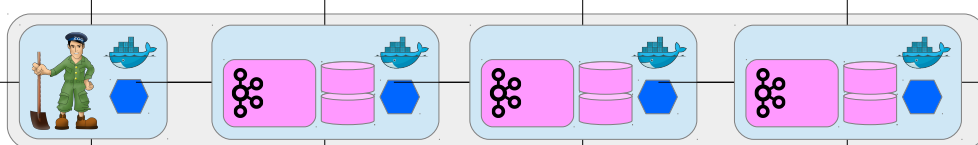
Controller



Buffer set

Parameters:

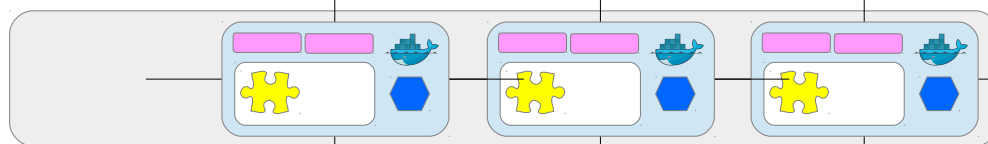
- r replicas
- p partitions



Processor set

Parameters:

- n nodes
- input
 - zookeeper
 - topic
- output
 - zookeeper
 - topic



Jupyter visualization

Elasticsearch

Selective filtering

