Kafka Streams Testing

A Deep Dive

Ivan Ponomarev, John Roesler

Who Are We

Ivan Ponomarev:

- Software Engineer at KURS, tutor at MIPT
- Apache Kafka Contributor

Who Are We

John Roesler:

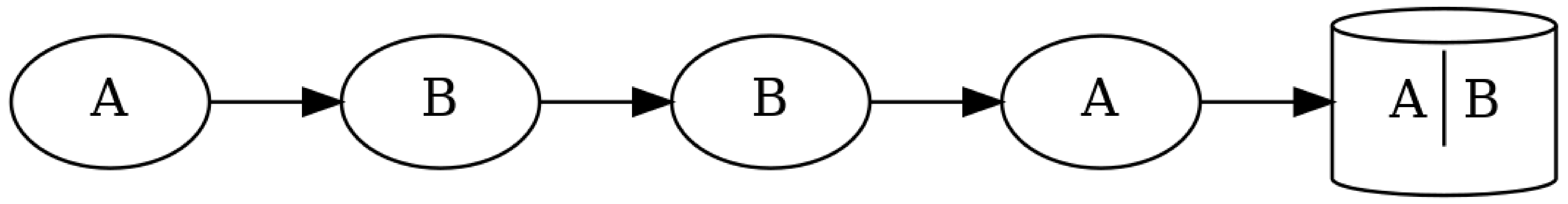
- Software Engineer at Confluent
- Apache Kafka Committer and PMC member

Kafka Streams Testing: A Deep Dive

1. Purpose: cover testing methodologies for Kafka Streams
   - "Unit" Testing: TopologyTestDriver
   - Integration Testing: KafkaStreams
2. Start with motivating example (from Ivan’s production)
3. A flawed testing approach: unit testing doesn’t work for this example
4. Deep-dive into the testing framework
5. Correctly testing the example with integration tests

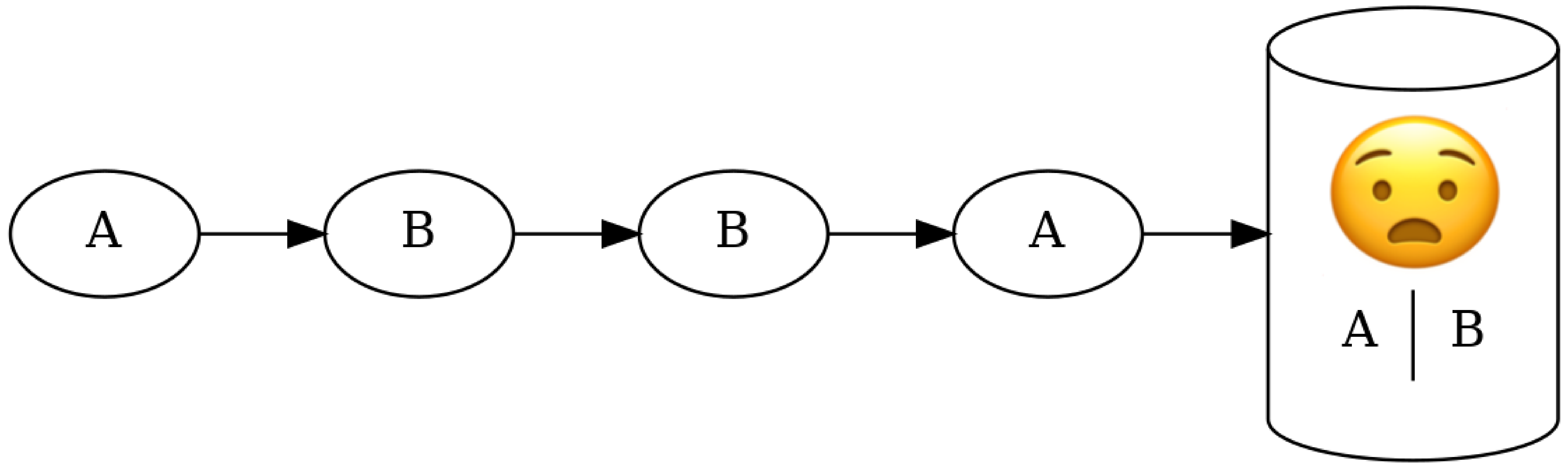
The task

Save different source IDs in the database



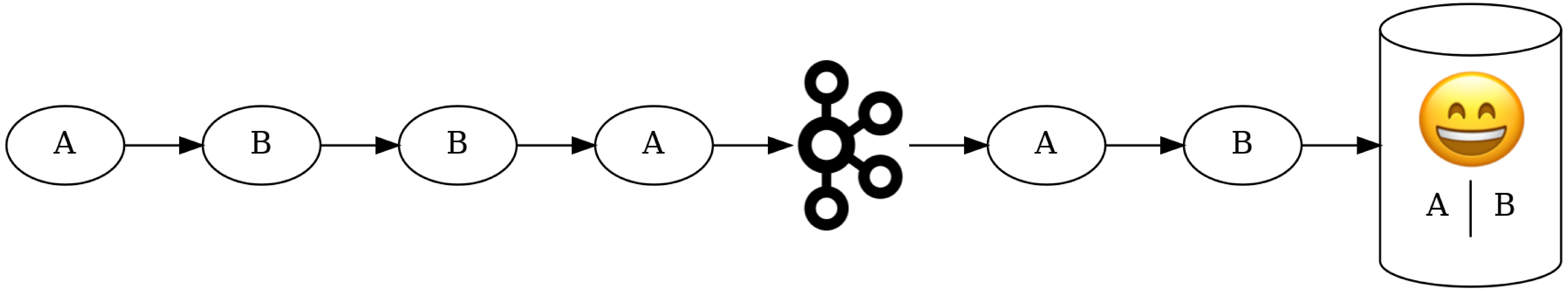
The problem

Too many writes to the database

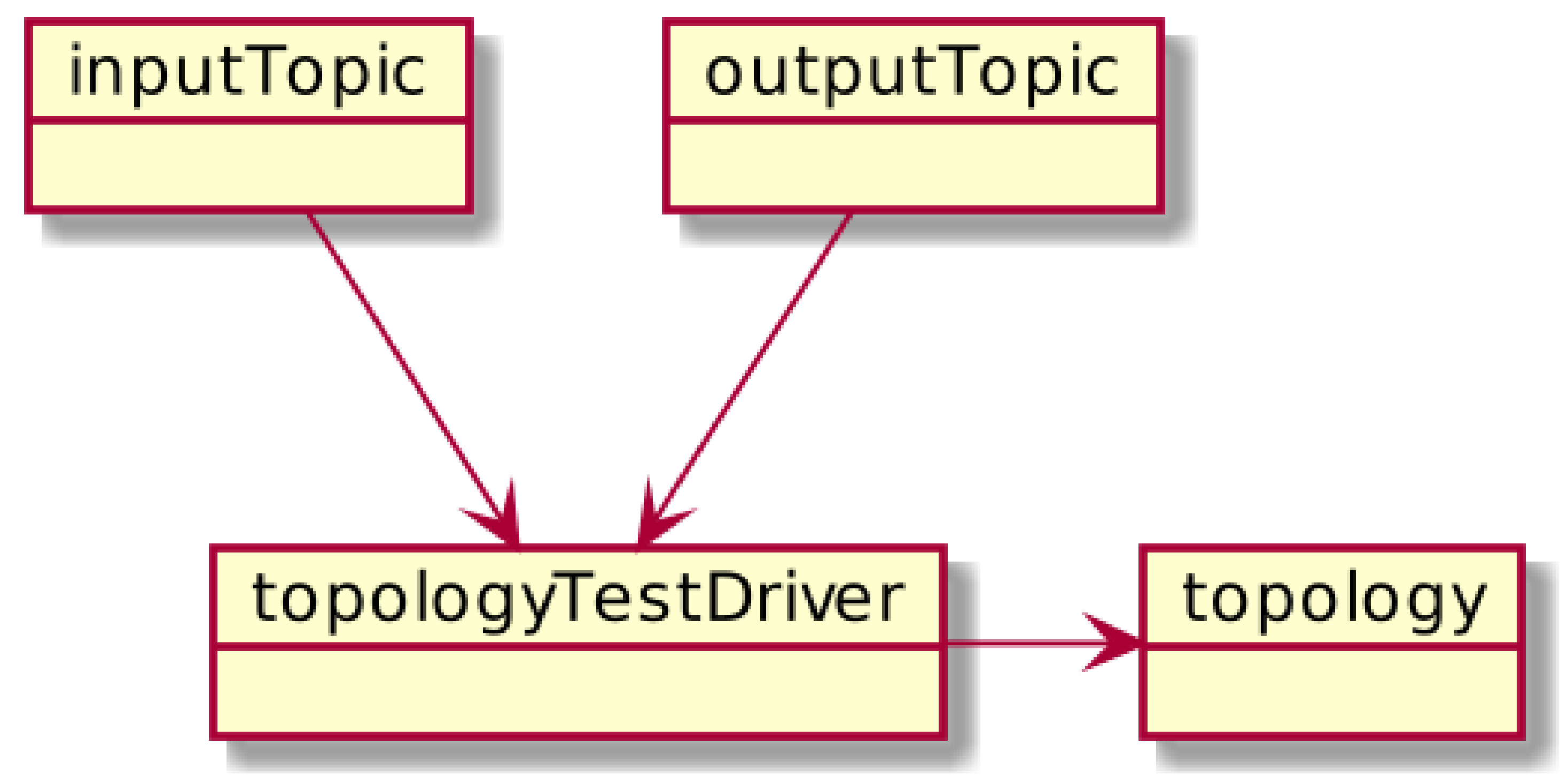


The solution

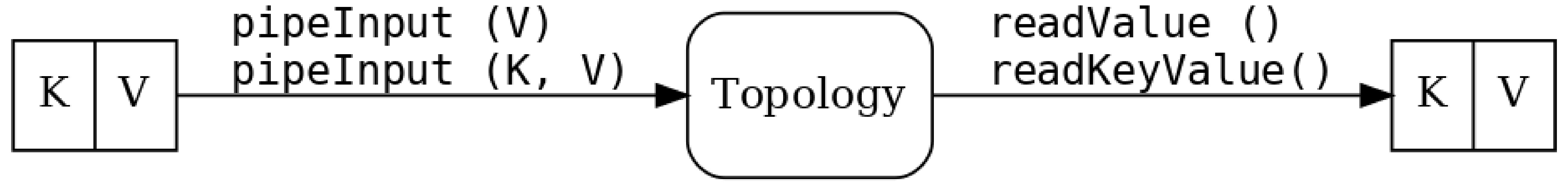
Let’s deduplicate using Kafka Streams!



TopologyTestDriver

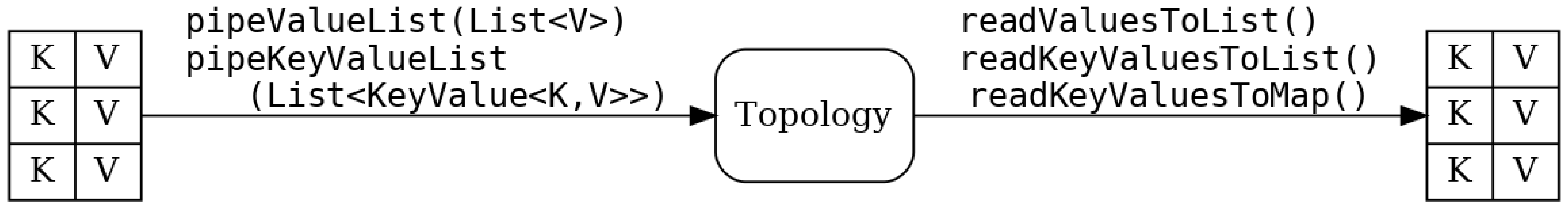


`TopologyTestDriver` capabilities



What is being sent/received	TestInputTopic methods	TestOutputTopic methods
A single value	`pipeInput (V)`	`V readValue ()`
A key/value pair	`pipeInput (K, V)`	`KeyValue<K,V> readKeyValue()`

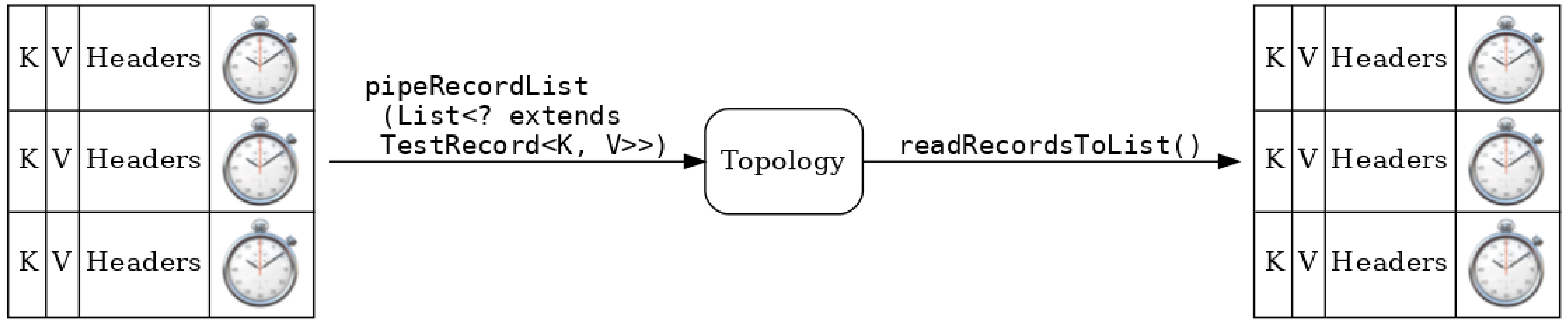
`TopologyTestDriver` capabilities



What is being sent/received	TestInputTopic methods	TestOutputTopic methods
A list of values	`pipeValueList (List<V>)`	`List<V> readValuesToList()`

A list of key/value pairs	`pipeKeyValueList (List<KeyValue<K,V>>)`	`List<KeyValue<K,V>> readKeyValuesToList()`
		`Map<K,V> readKeyValuesToMap()`

`TopologyTestDriver` capabilities

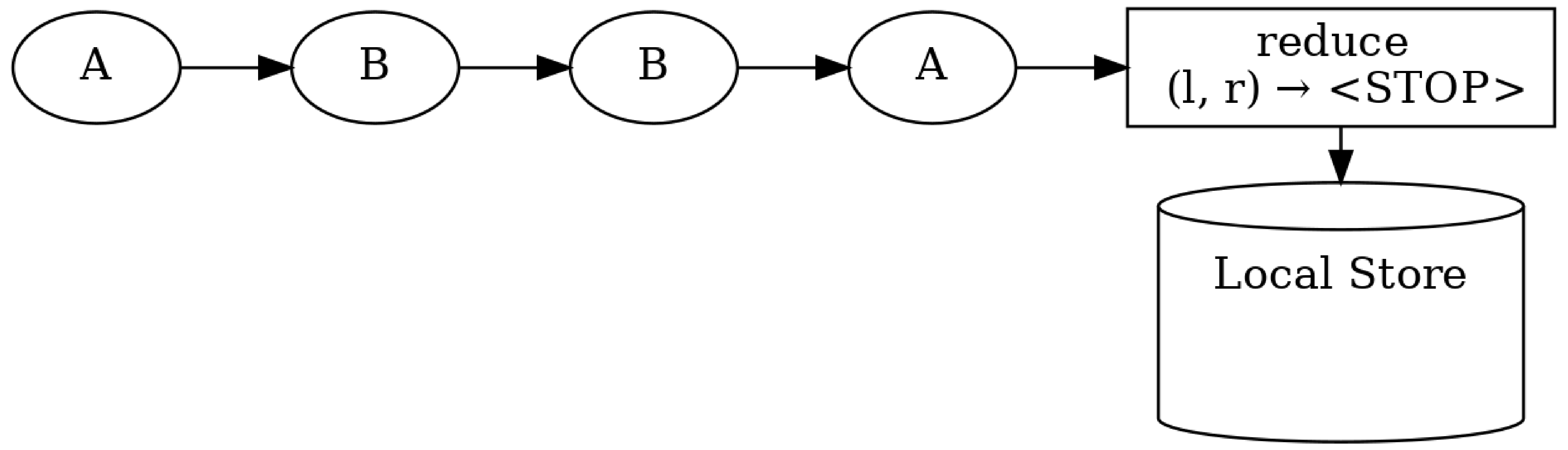


What is being sent/received	TestInputTopic methods	TestOutputTopic methods
A list of Records	`pipeRecordList (List<? extends TestRecord<K, V>>)`	`List<TestRecord<K, V>> readRecordsToList()`

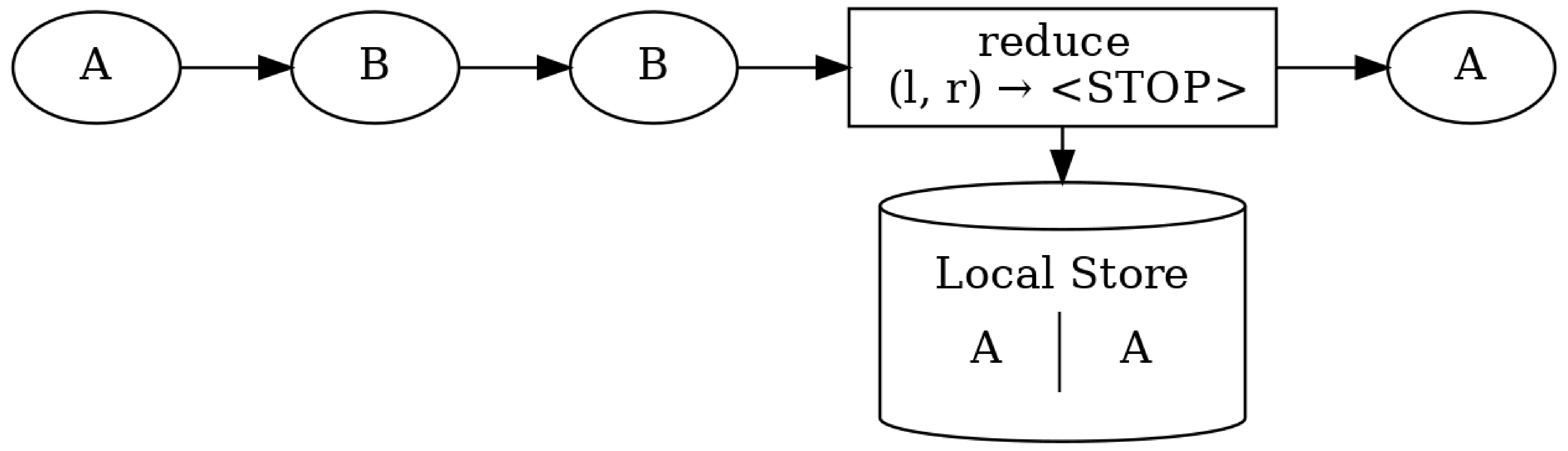
Demo

1. Spring Boot app
2. Let’s do some test-driven development and first write a test
3. Writing a test with TTDriver

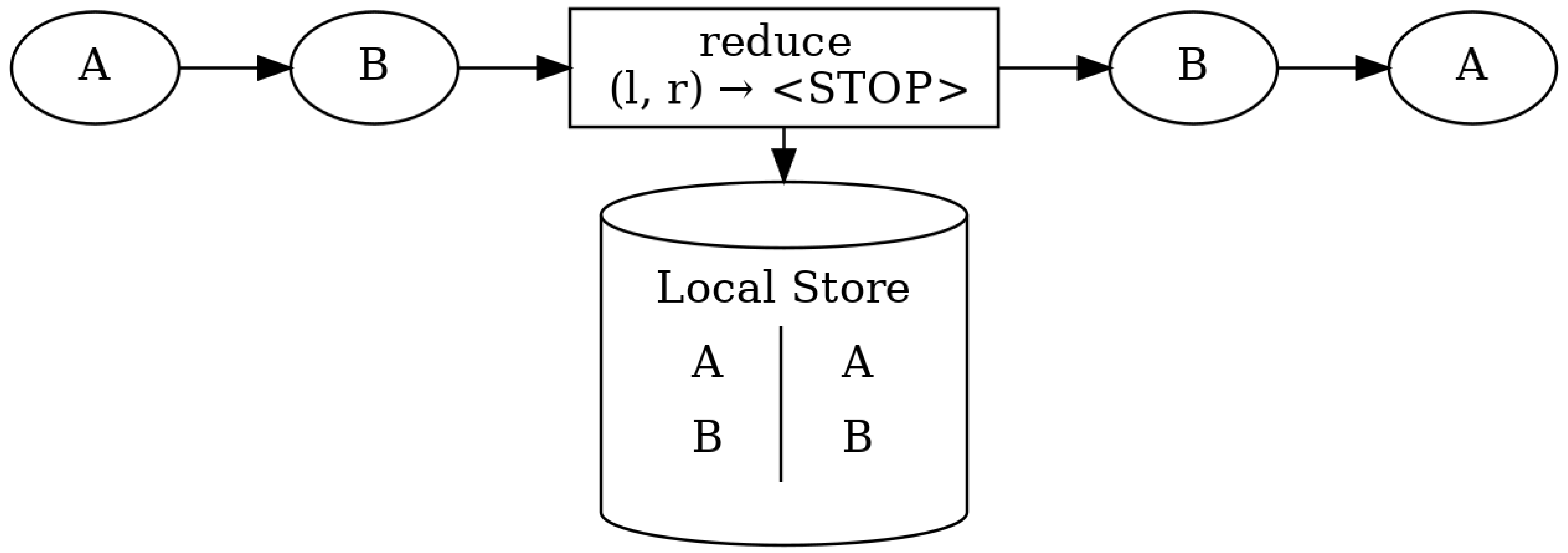
A "Simple Solution"



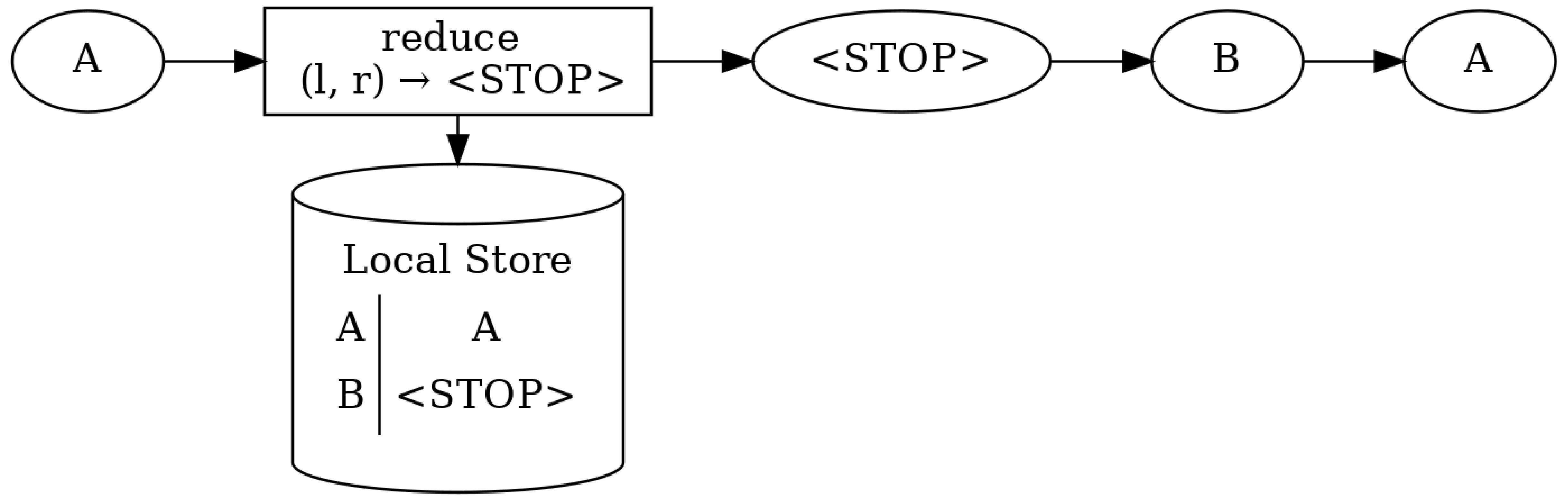
A "Simple Solution"



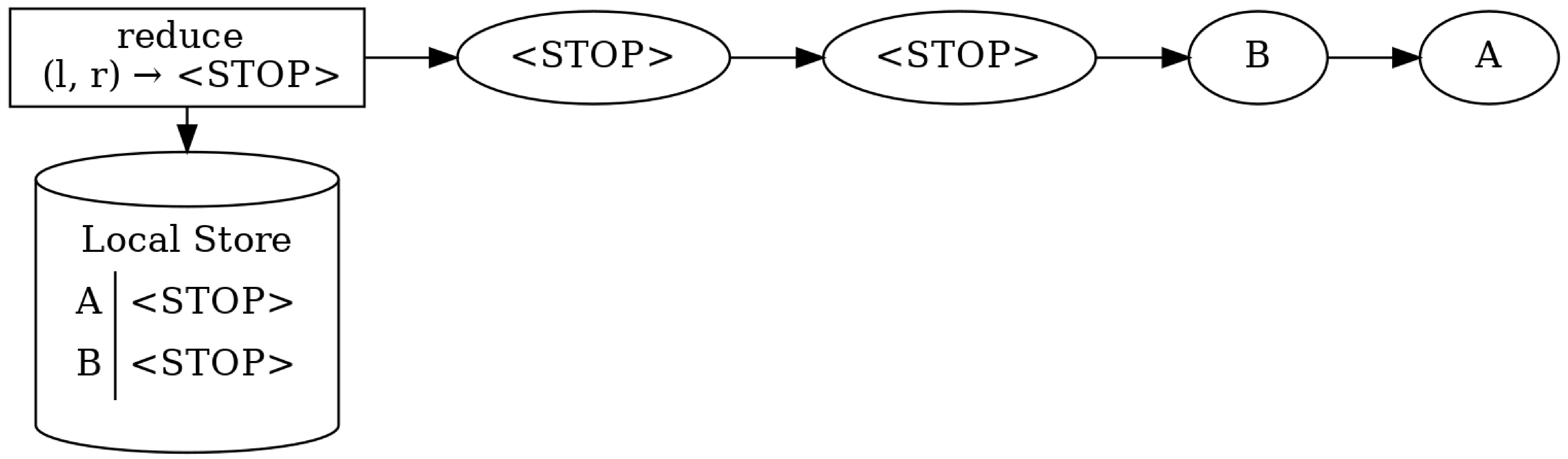
A "Simple Solution"



A "Simple Solution"



A "Simple Solution"



Demo

- writing the topology
- TopologyTestDriver test is green

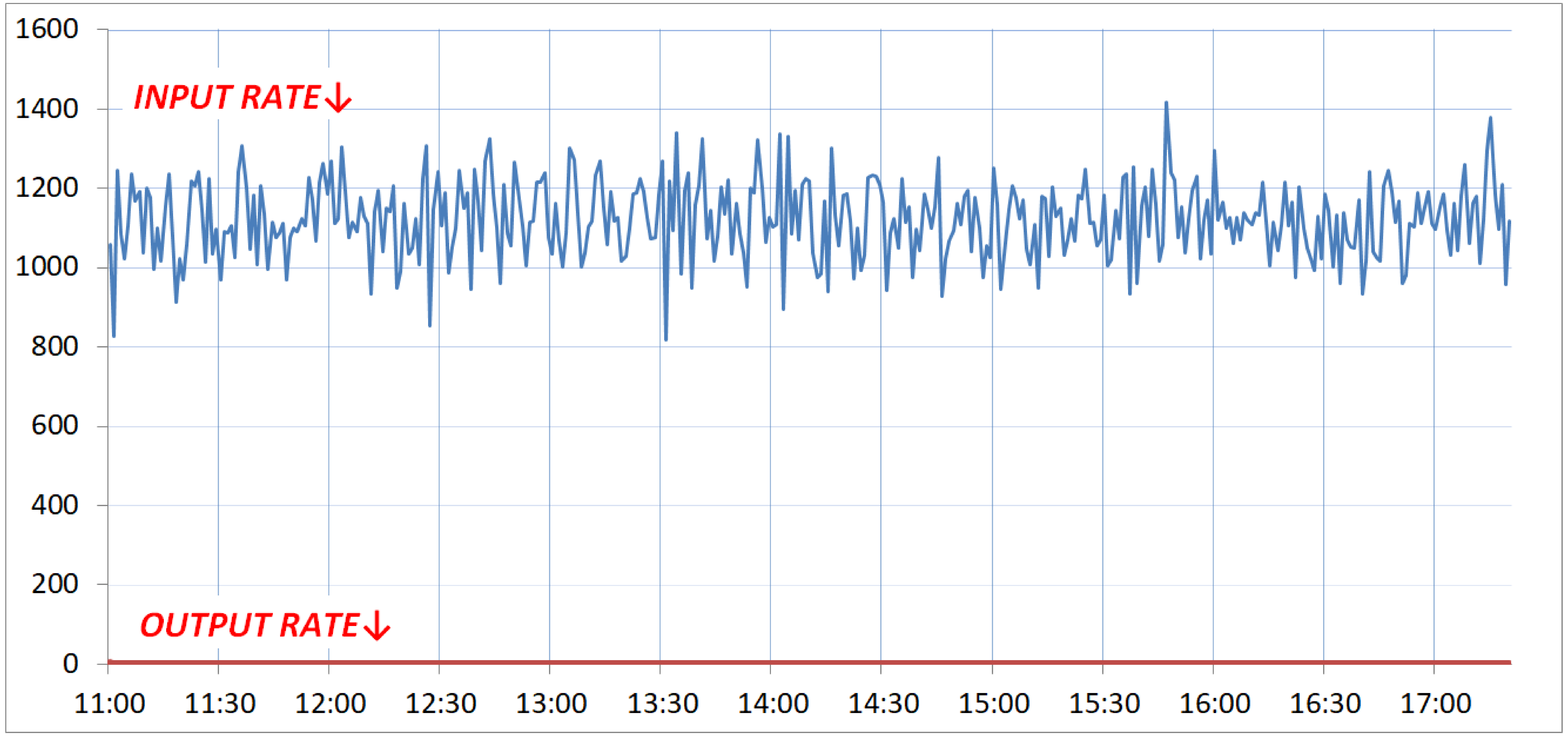
Tests are green

Tests are green

build passing

Should we run this in production?

What we saw in production:



Why it’s not working

Kafka Streams	**TopologyTestDriver**
is a big data streaming framework	is a fast, deterministic testing framework

Why it’s not working

Kafka Streams	TopologyTestDriver
is a big data streaming framework	is a fast, deterministic testing framework

- designed for high throughput
- throughput demands batching, buffering, caching, etc.
- caching is the culprit in this example

Why it's not working

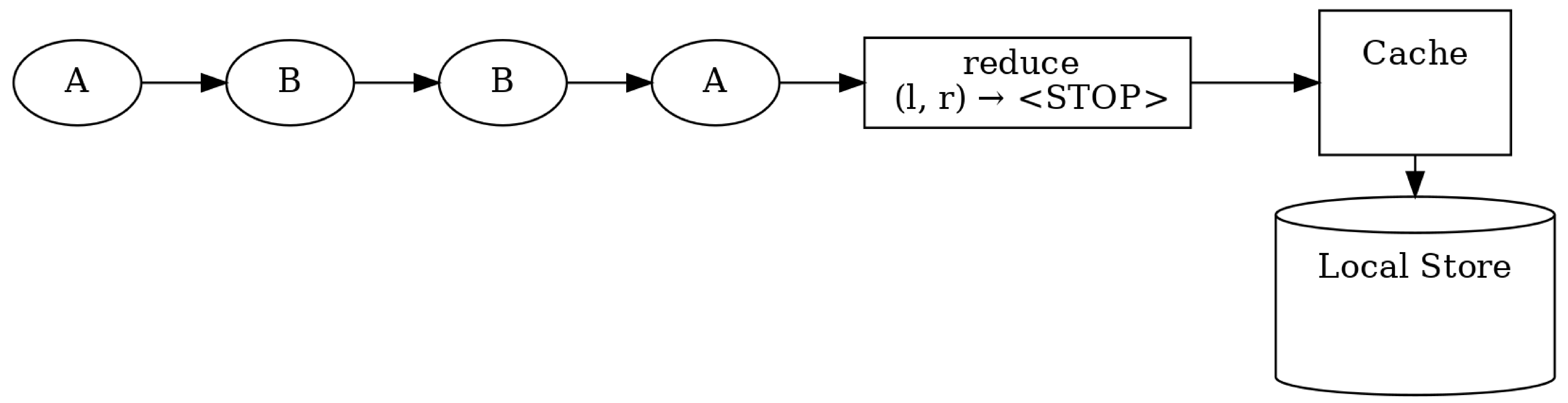
Kafka Streams	**TopologyTestDriver**
is a big data streaming framework	is a fast, deterministic testing framework
• designed for high throughput • throughput demands batching, buffering, caching, etc. • caching is the culprit in this example	• designed for synchronous, immediate results • flush cache after every update

Why it’s not working

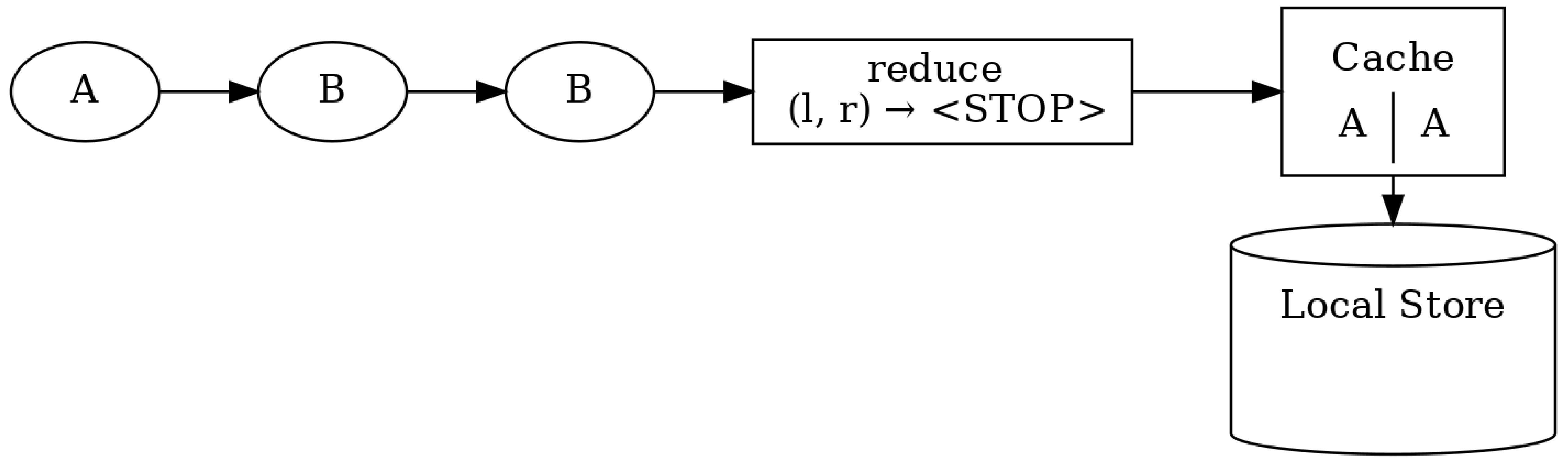
Caching in Kafka Streams

- don’t immediately emit every aggregation result
- "soak up" repeated updates to the same key’s aggregation
- configure cache size: `max.bytes.buffering` (10MB)
- configure cache flush interval: `commit.interval.ms` (30s)
- emit *latest* result on flush or eviction

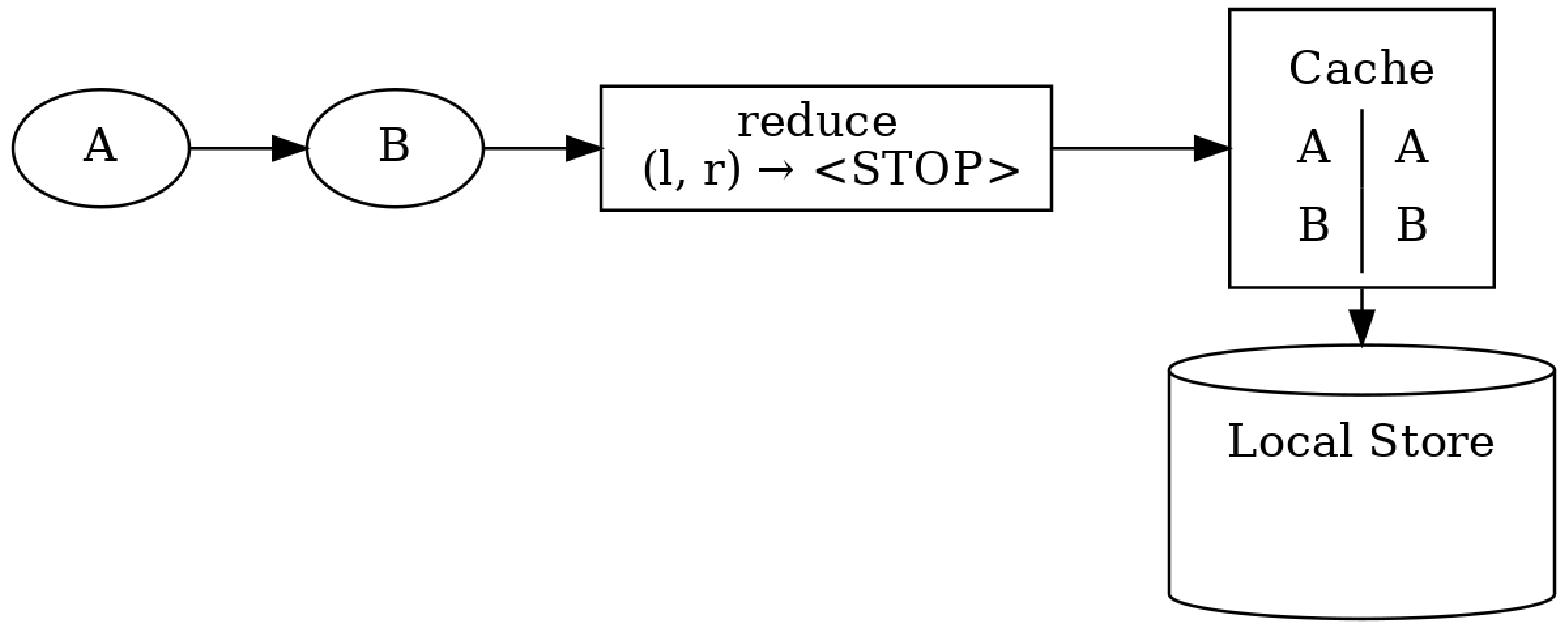
Why it’s not working



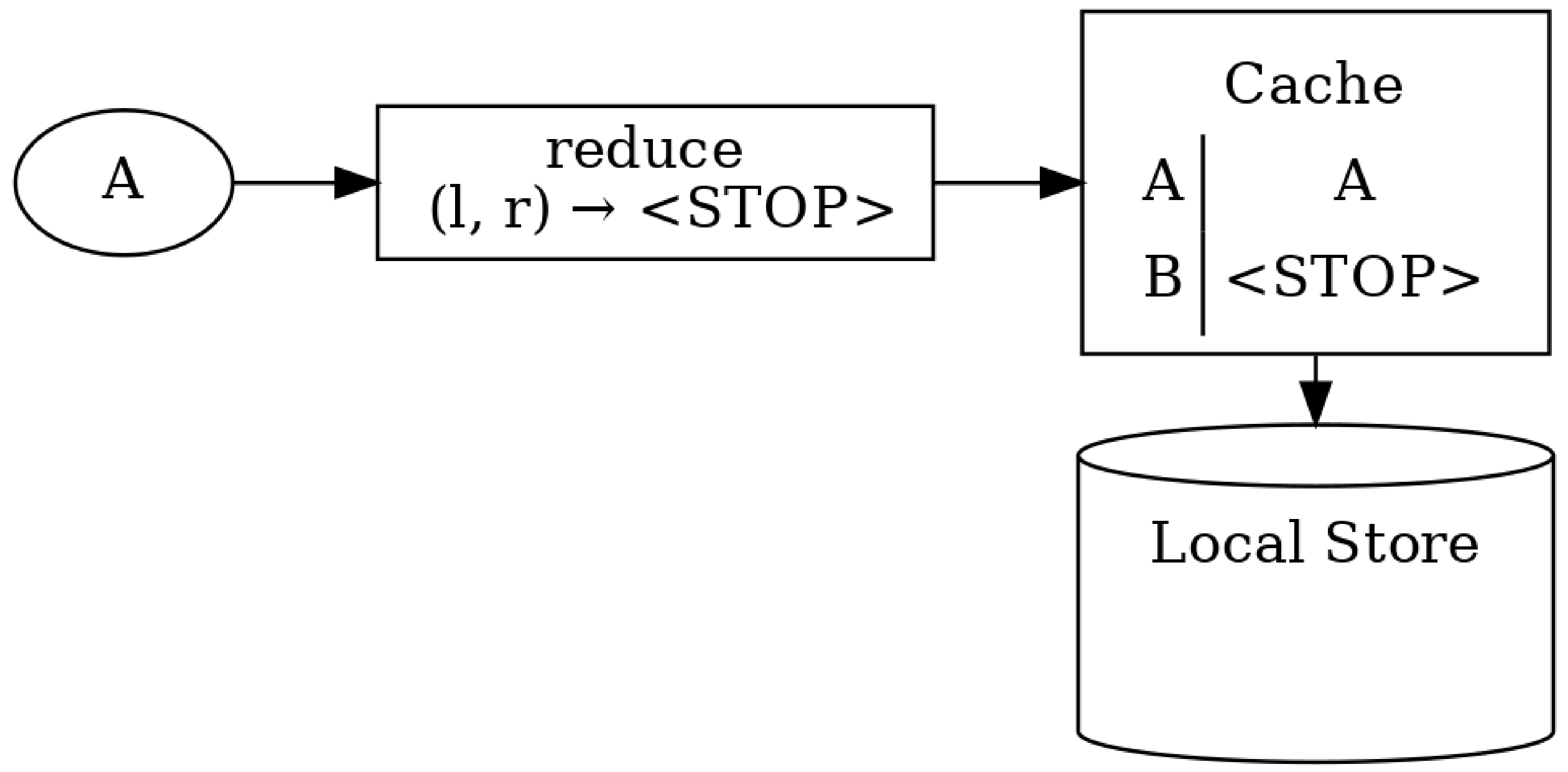
Why it’s not working



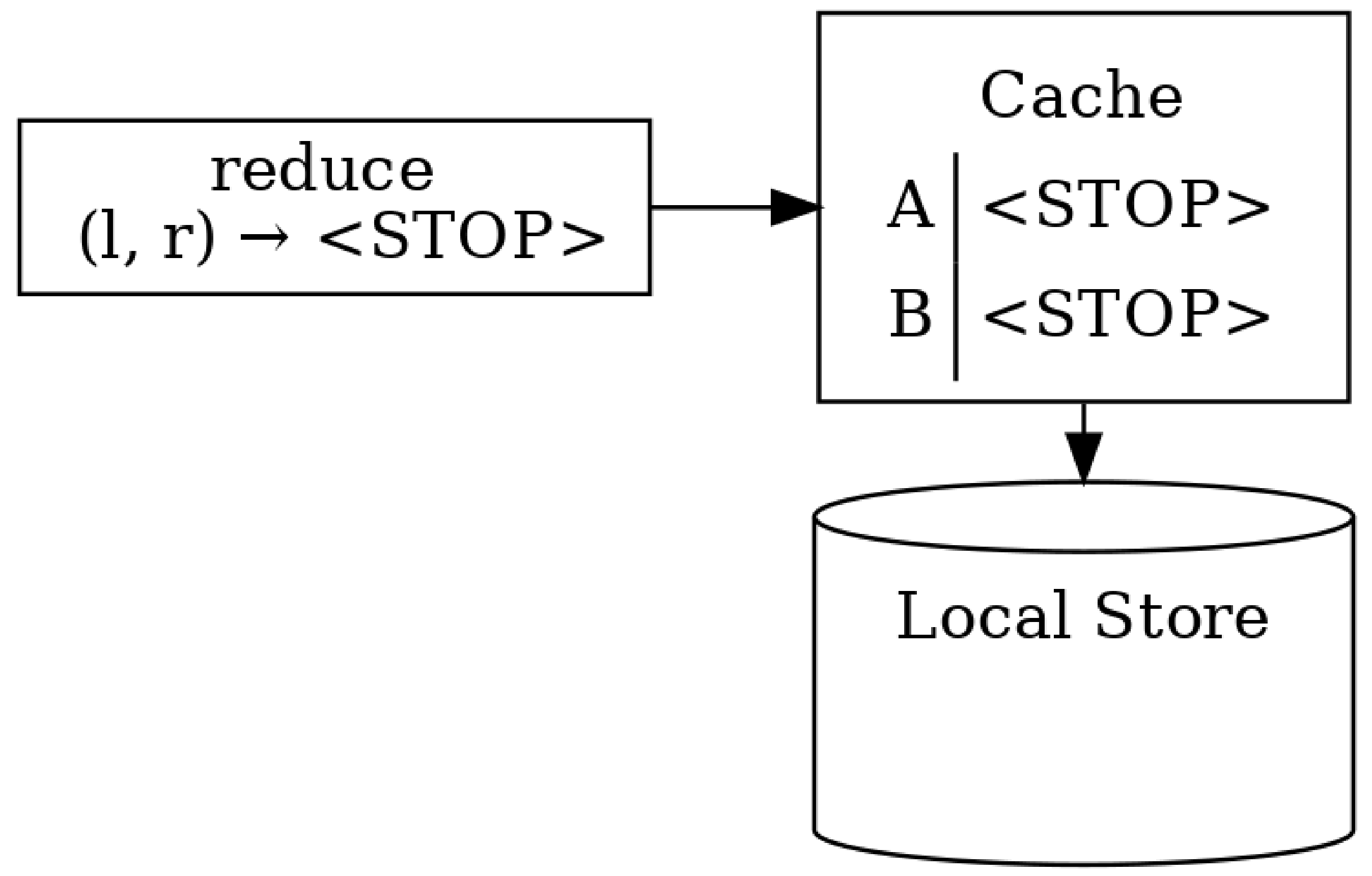
Why it’s not working



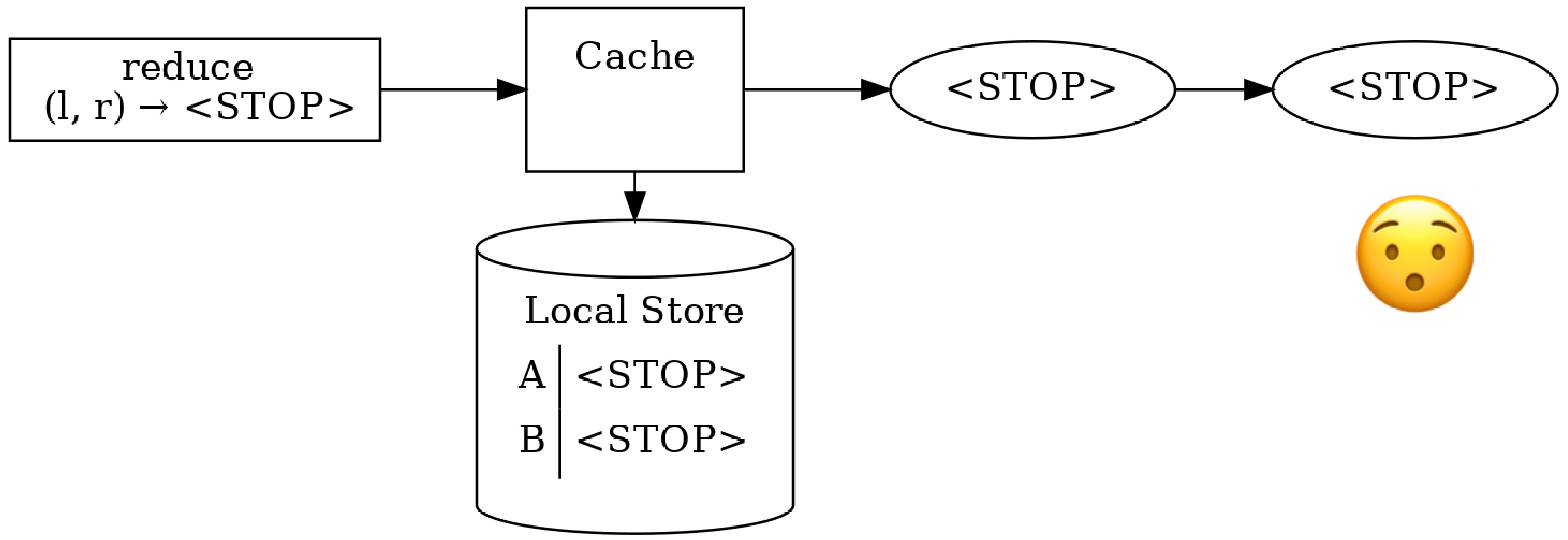
Why it’s not working



Why it’s not working



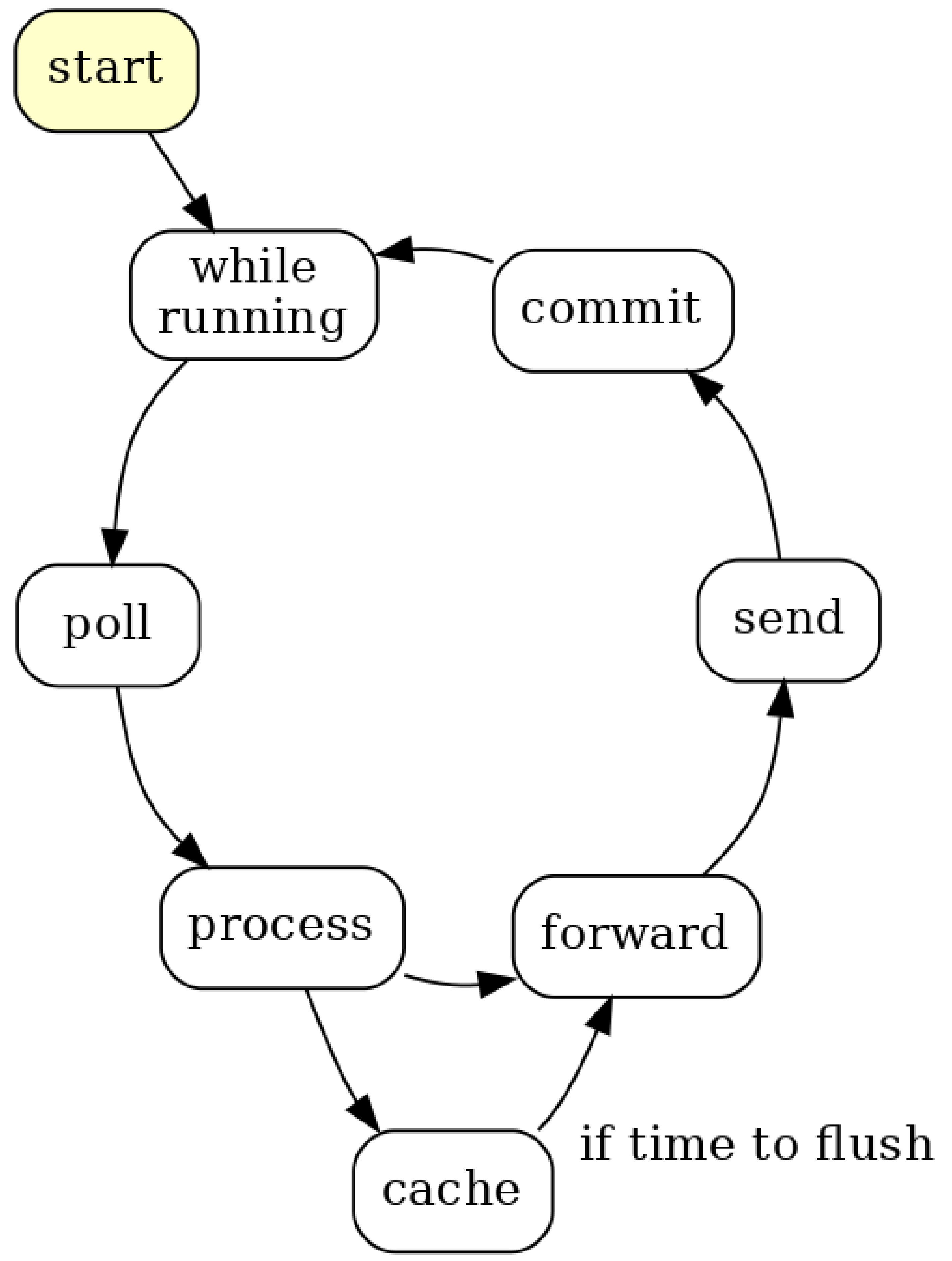
Why it's not working



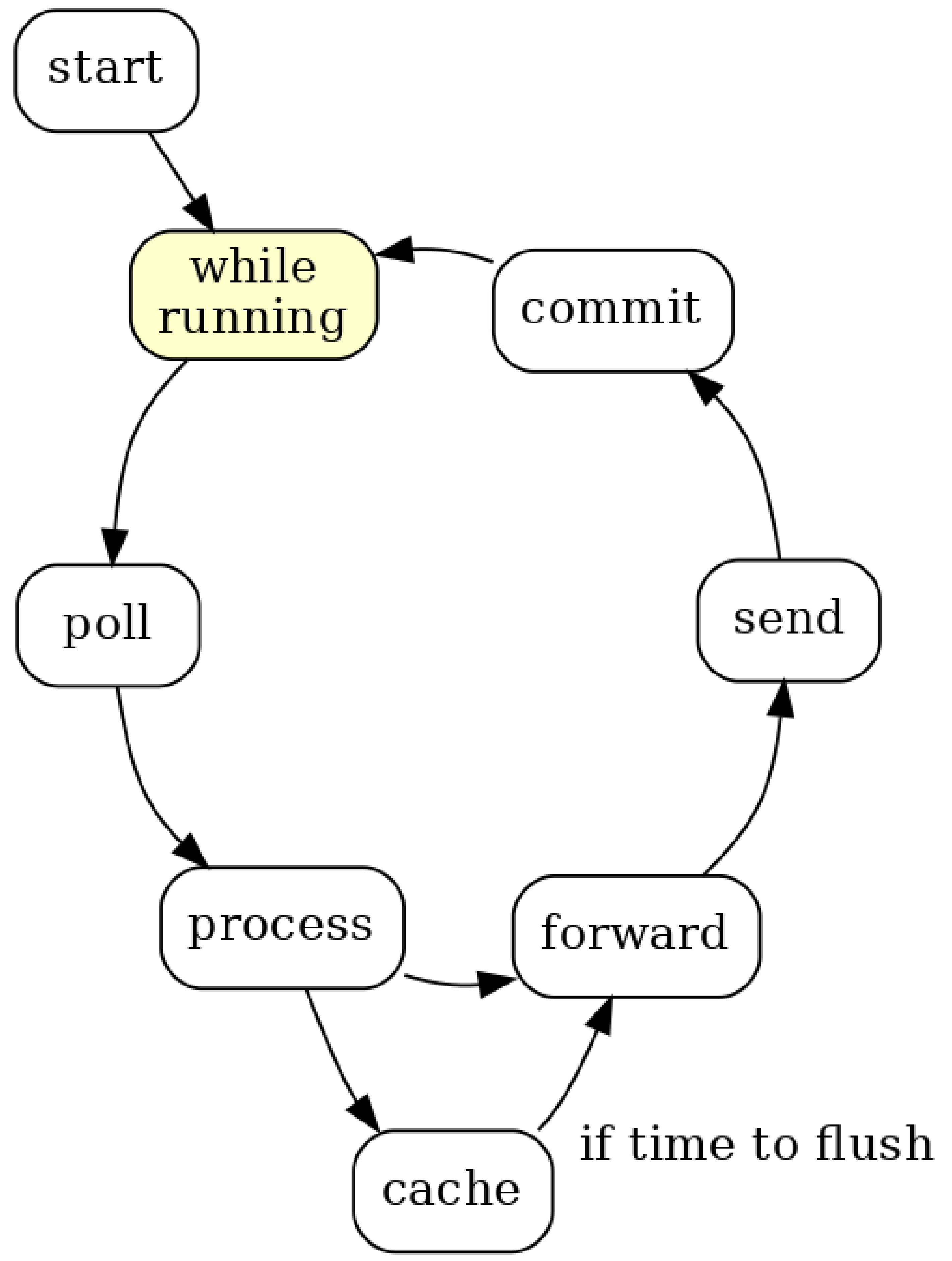
Demo

TopologyTestDriver vs. Kafka Streams execution loop

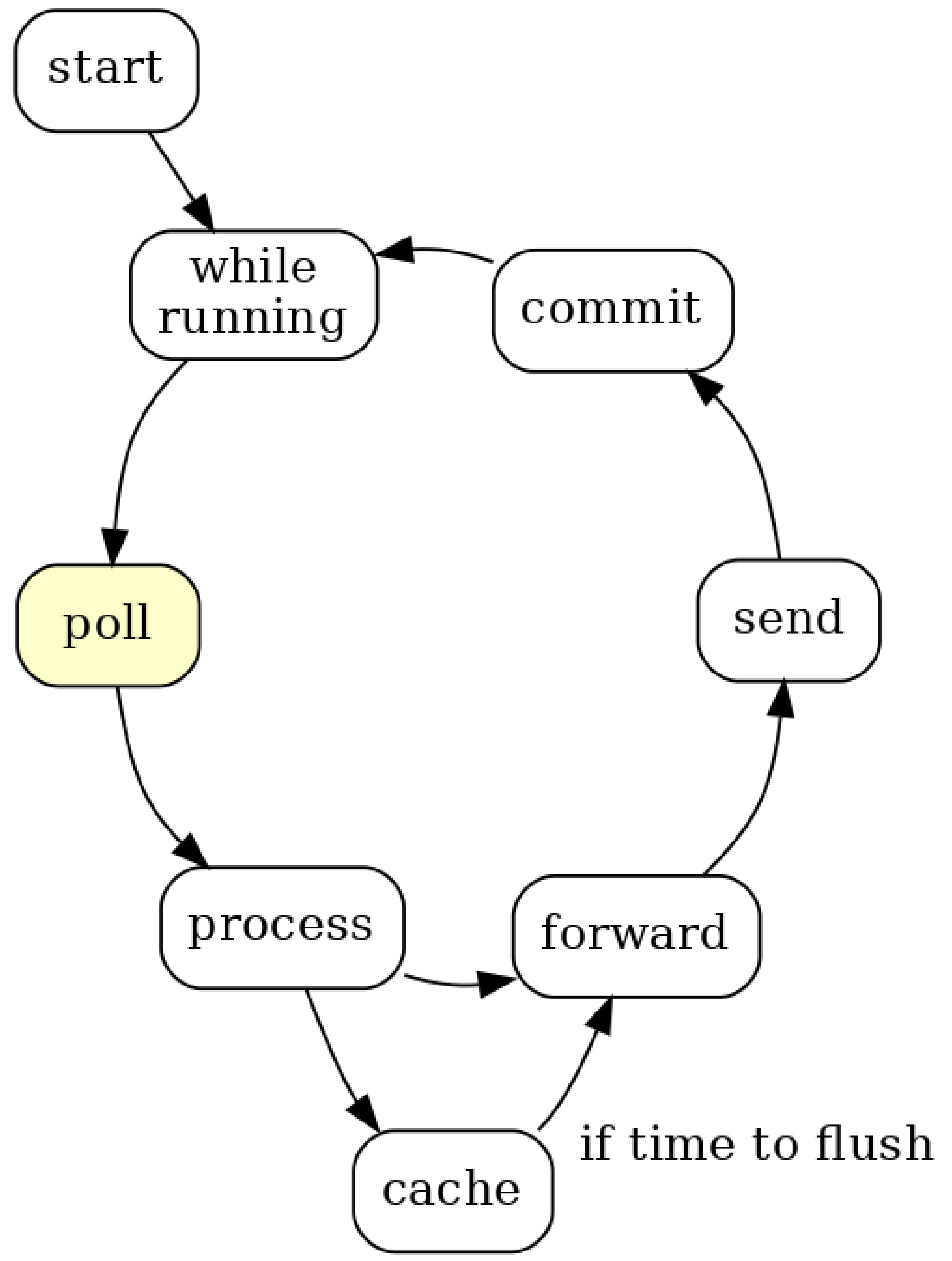
Kafka Streams execution loop



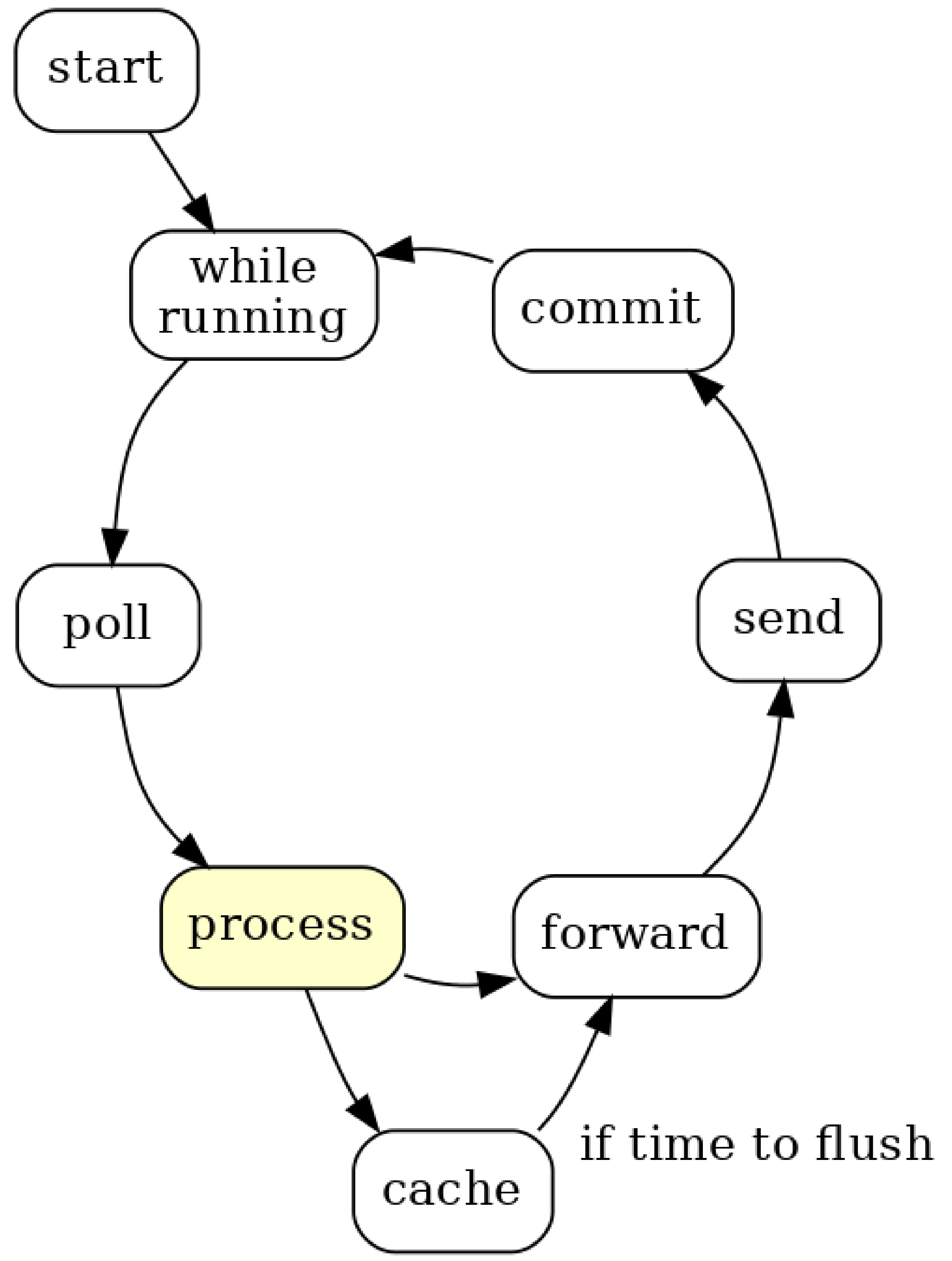
Kafka Streams execution loop



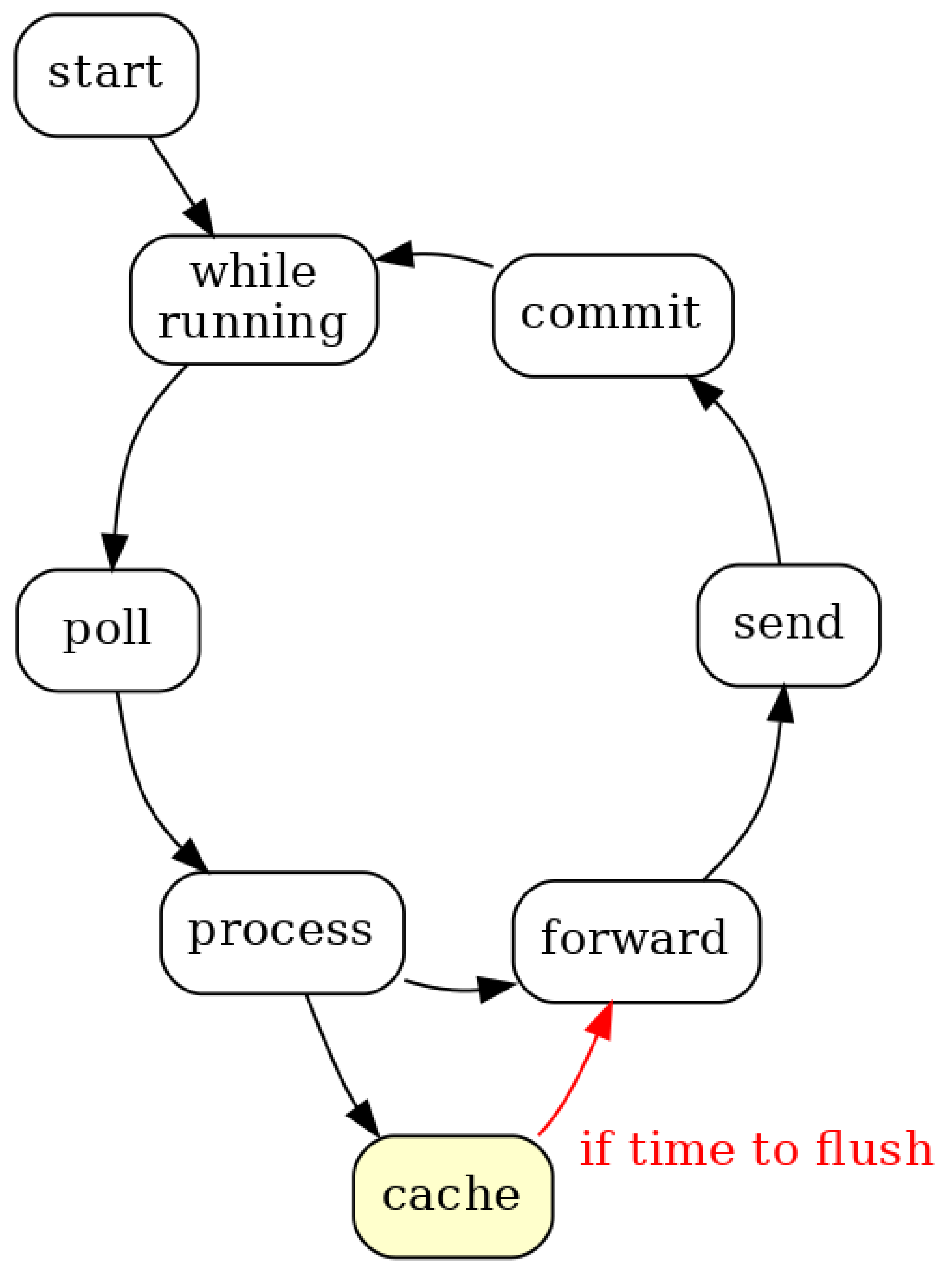
Kafka Streams execution loop



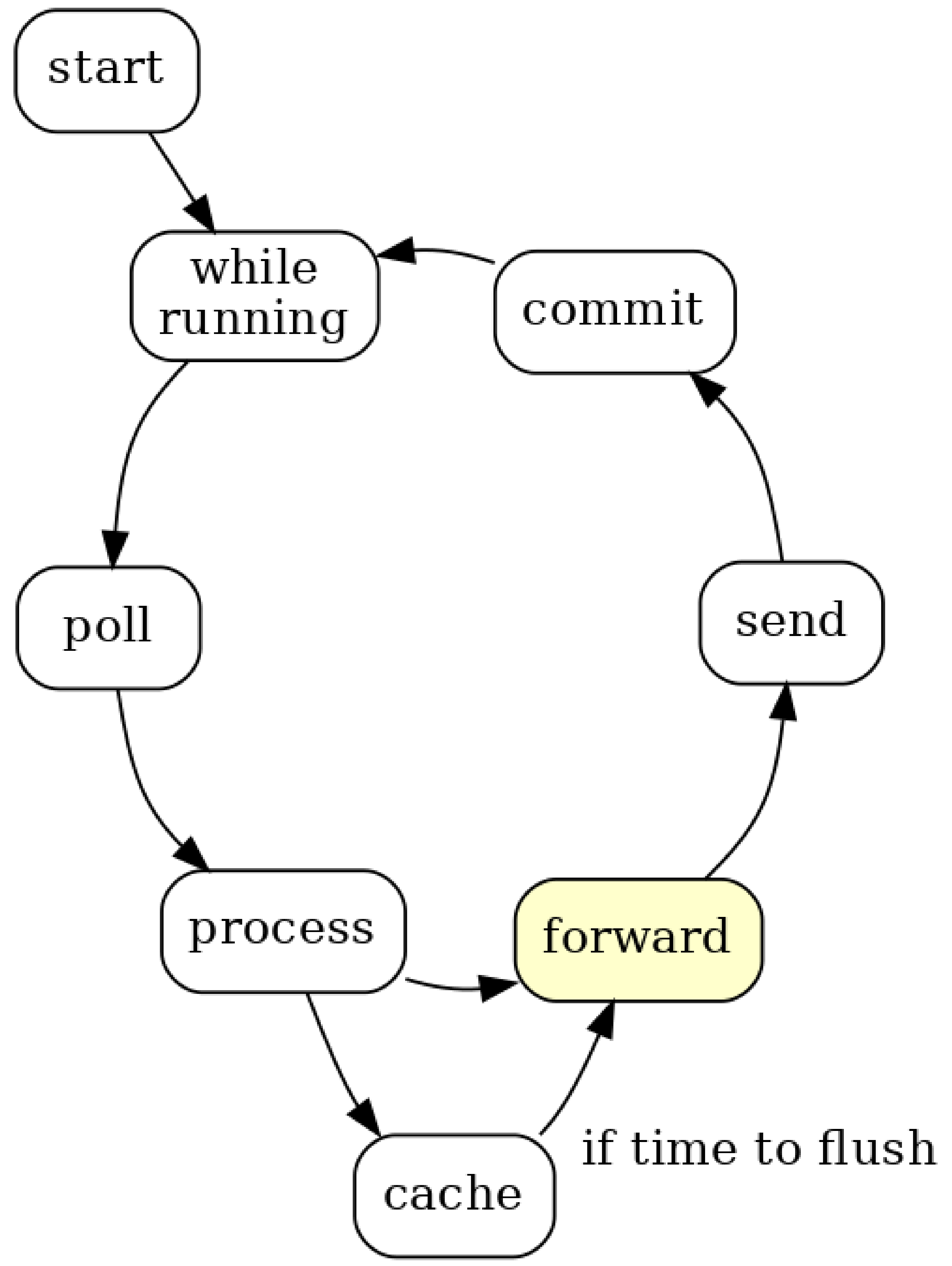
Kafka Streams execution loop



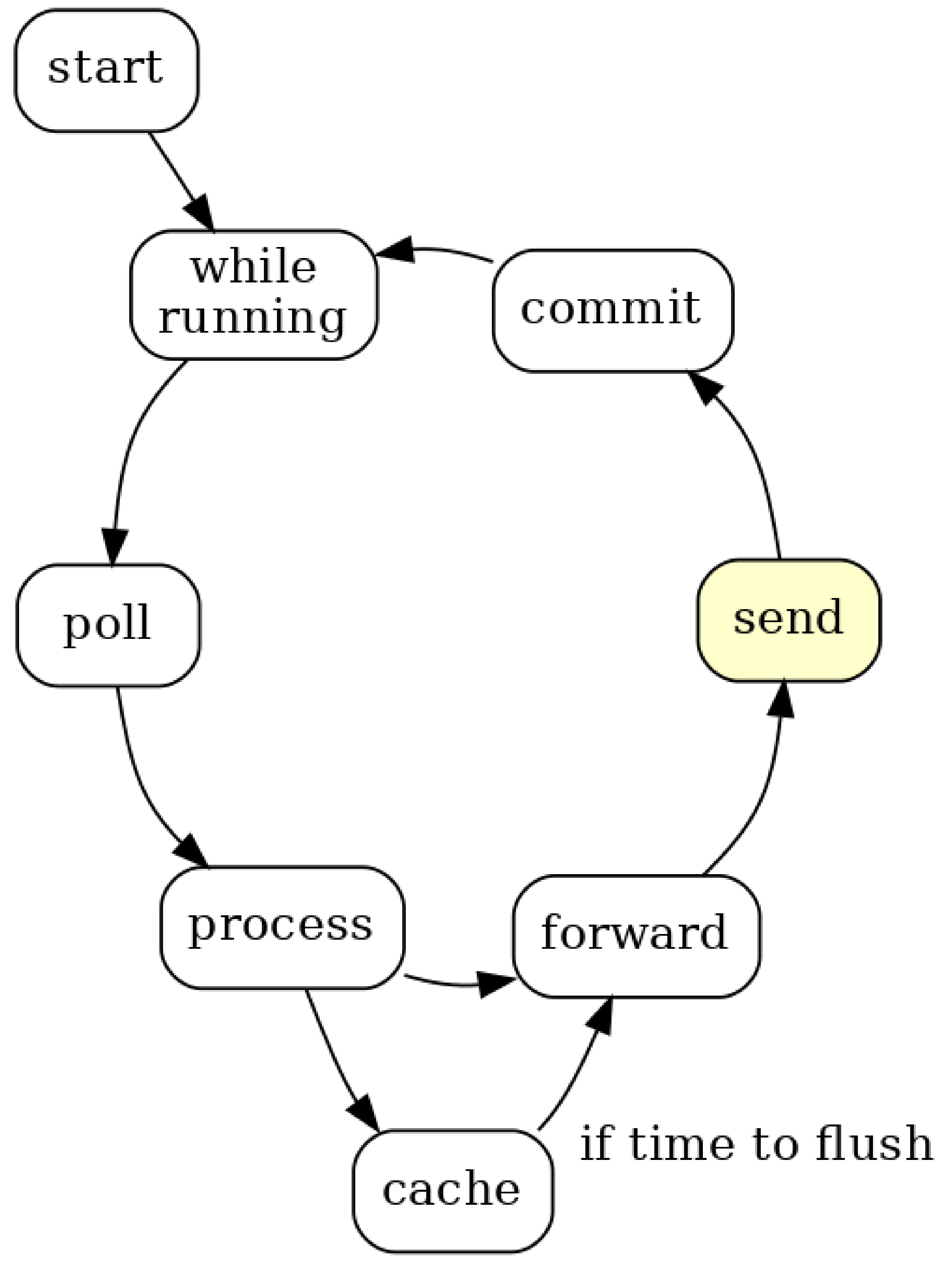
Kafka Streams execution loop



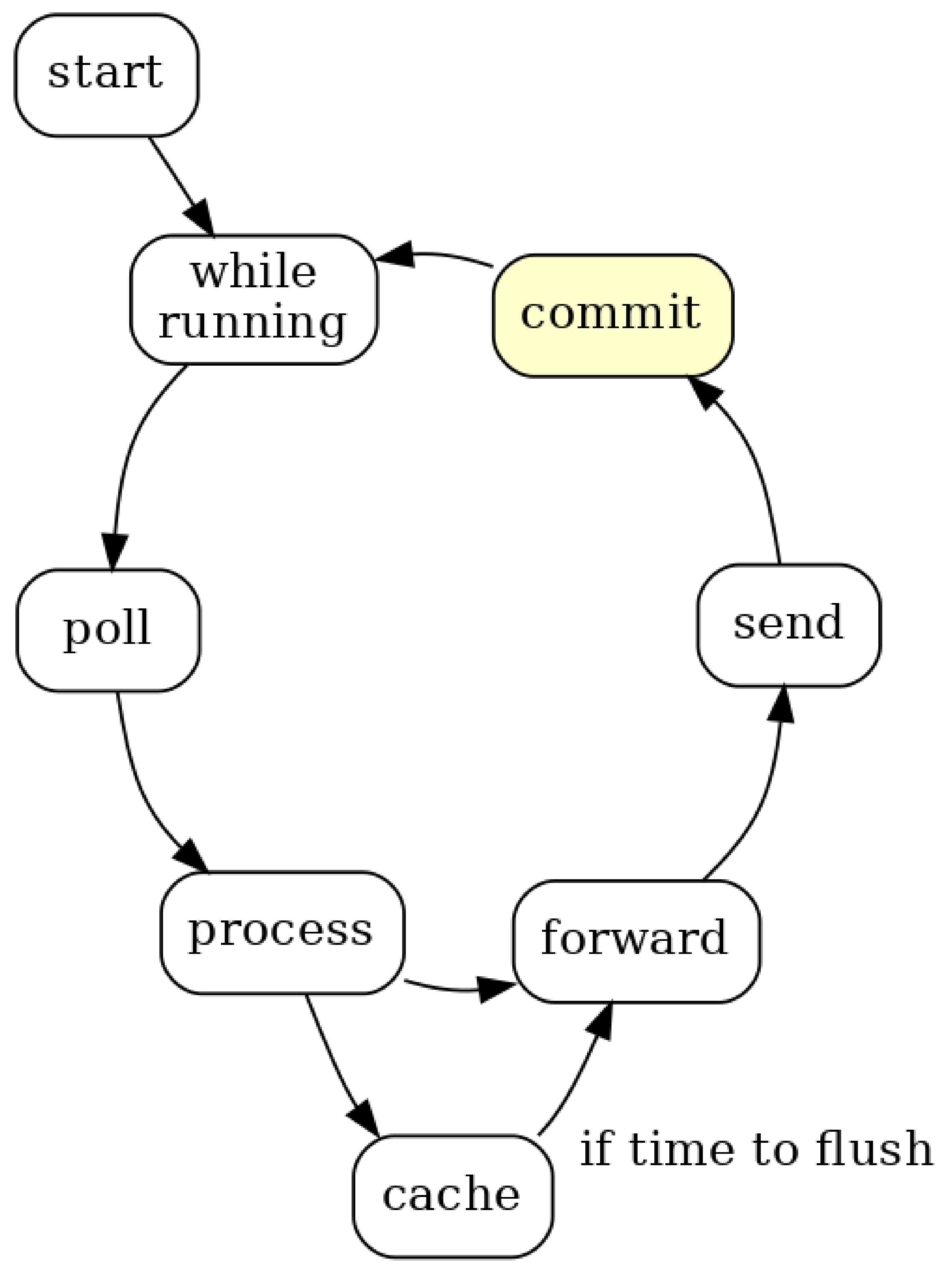
Kafka Streams execution loop



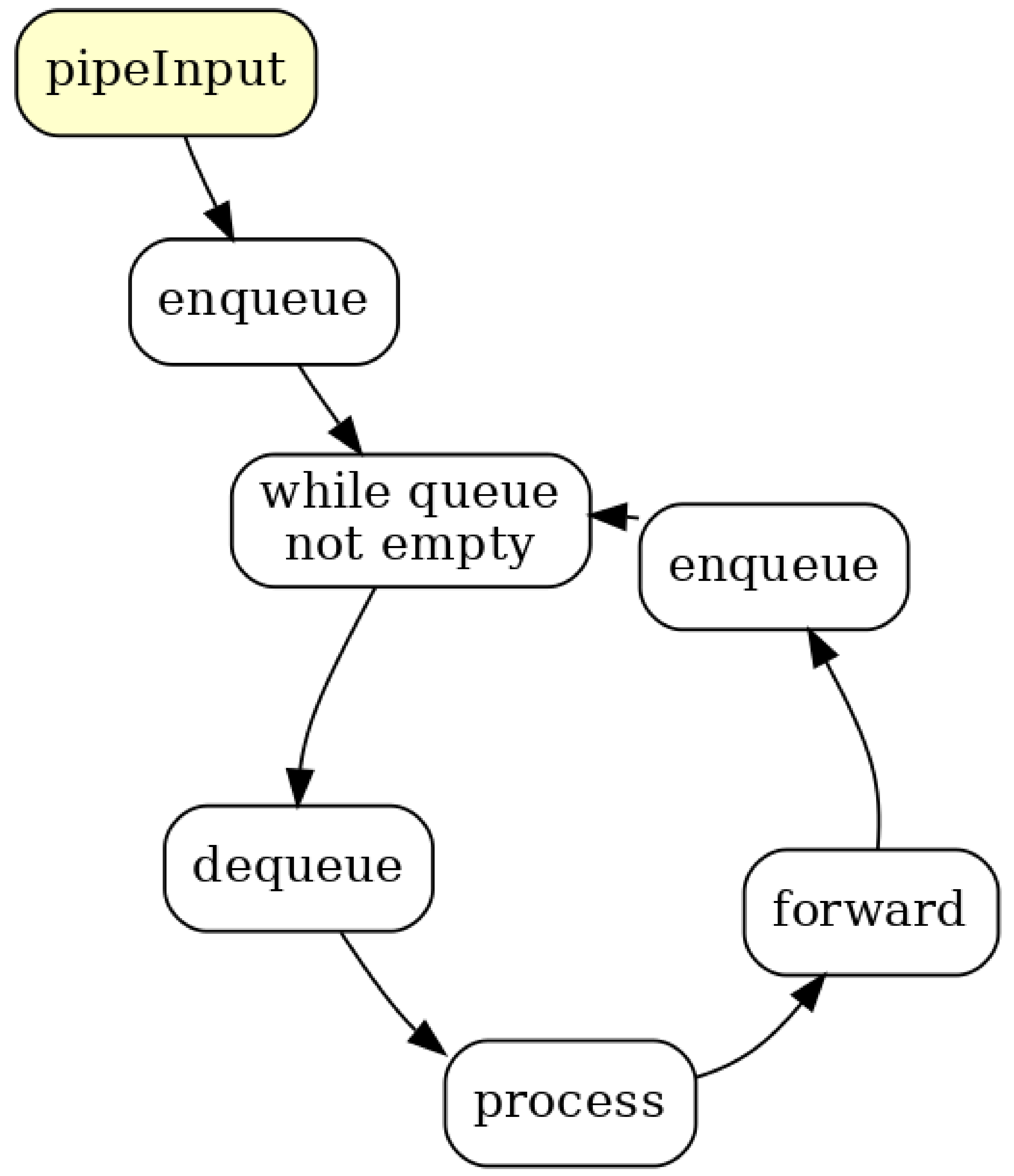
Kafka Streams execution loop



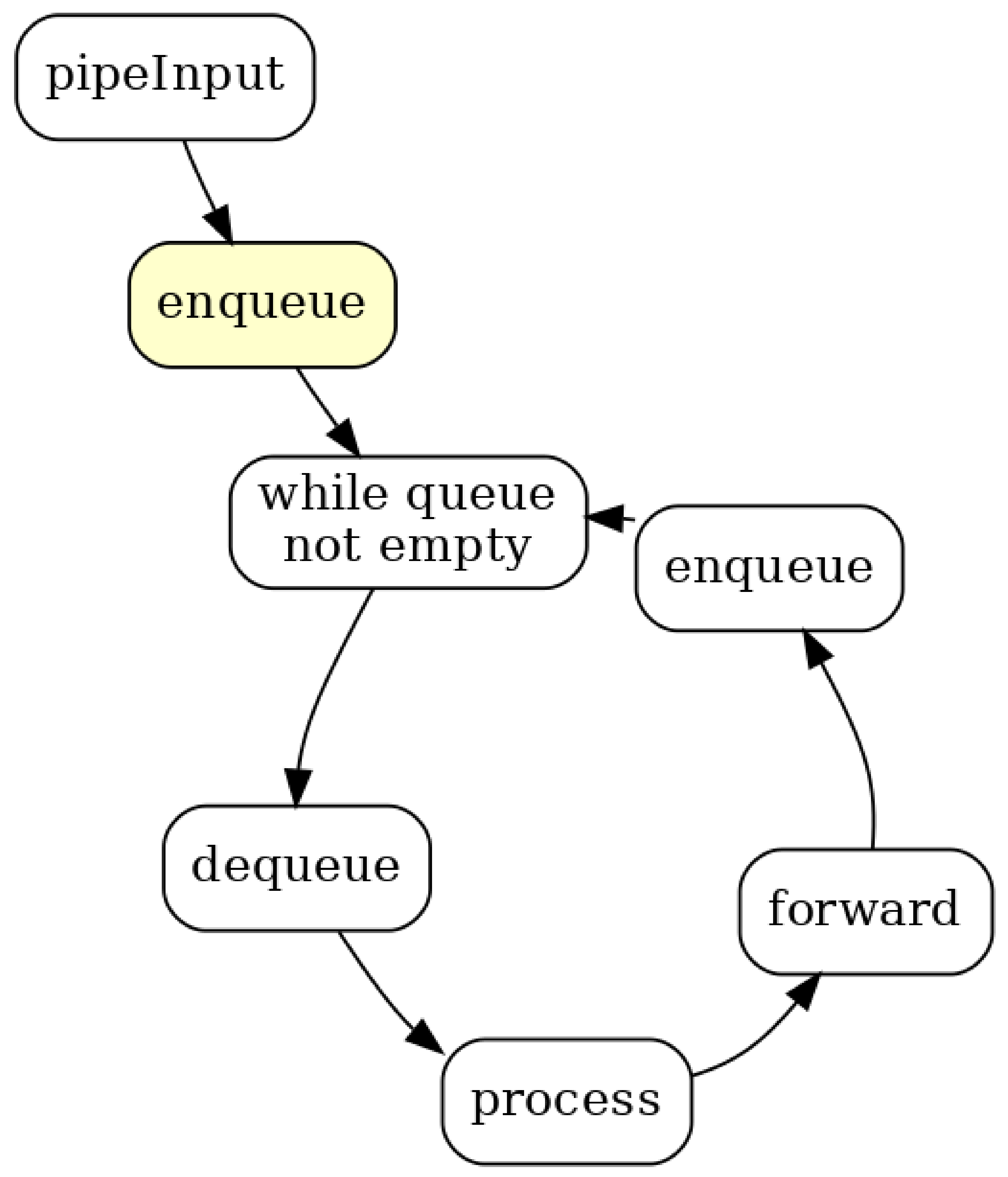
Kafka Streams execution loop



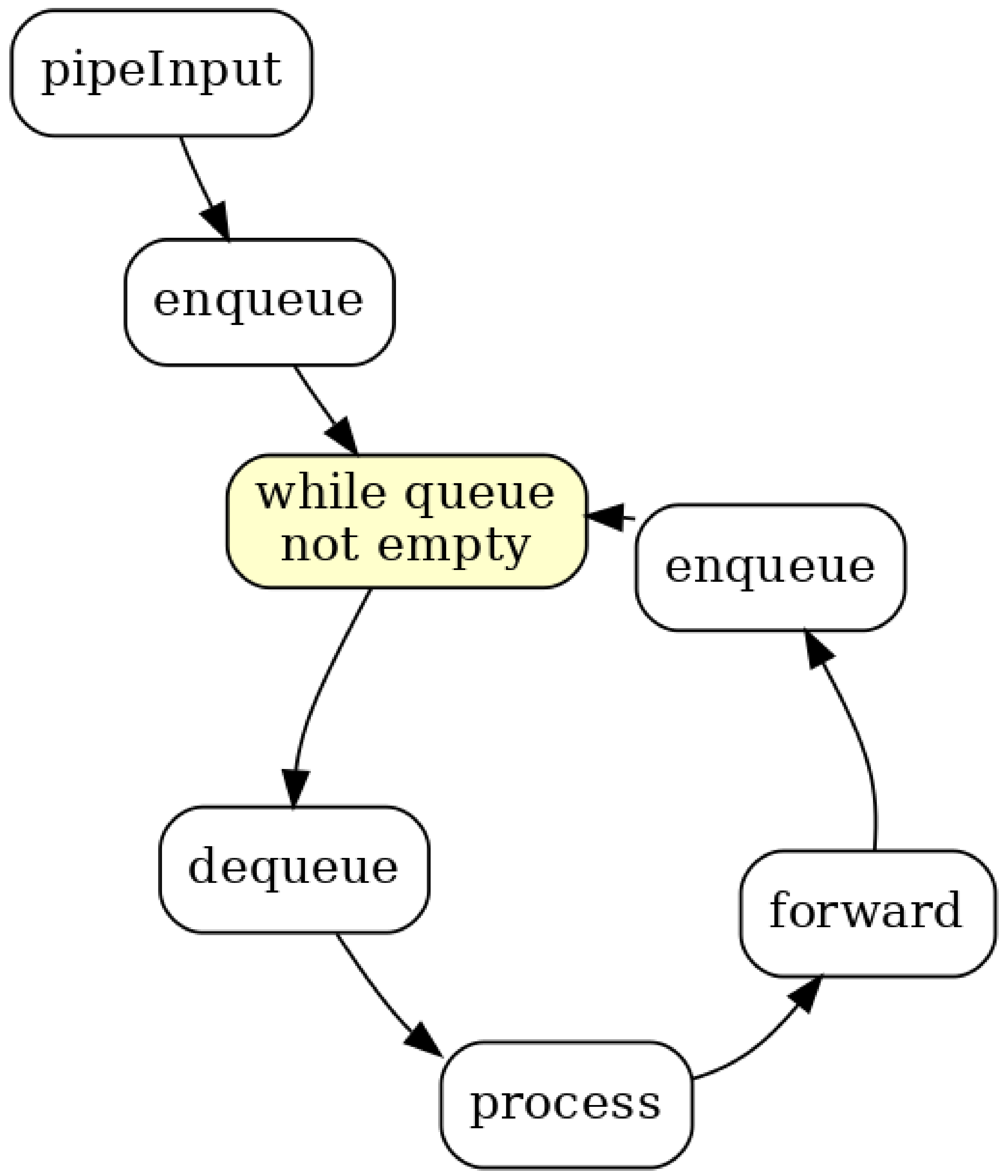
TopologyTestDriver execution loop



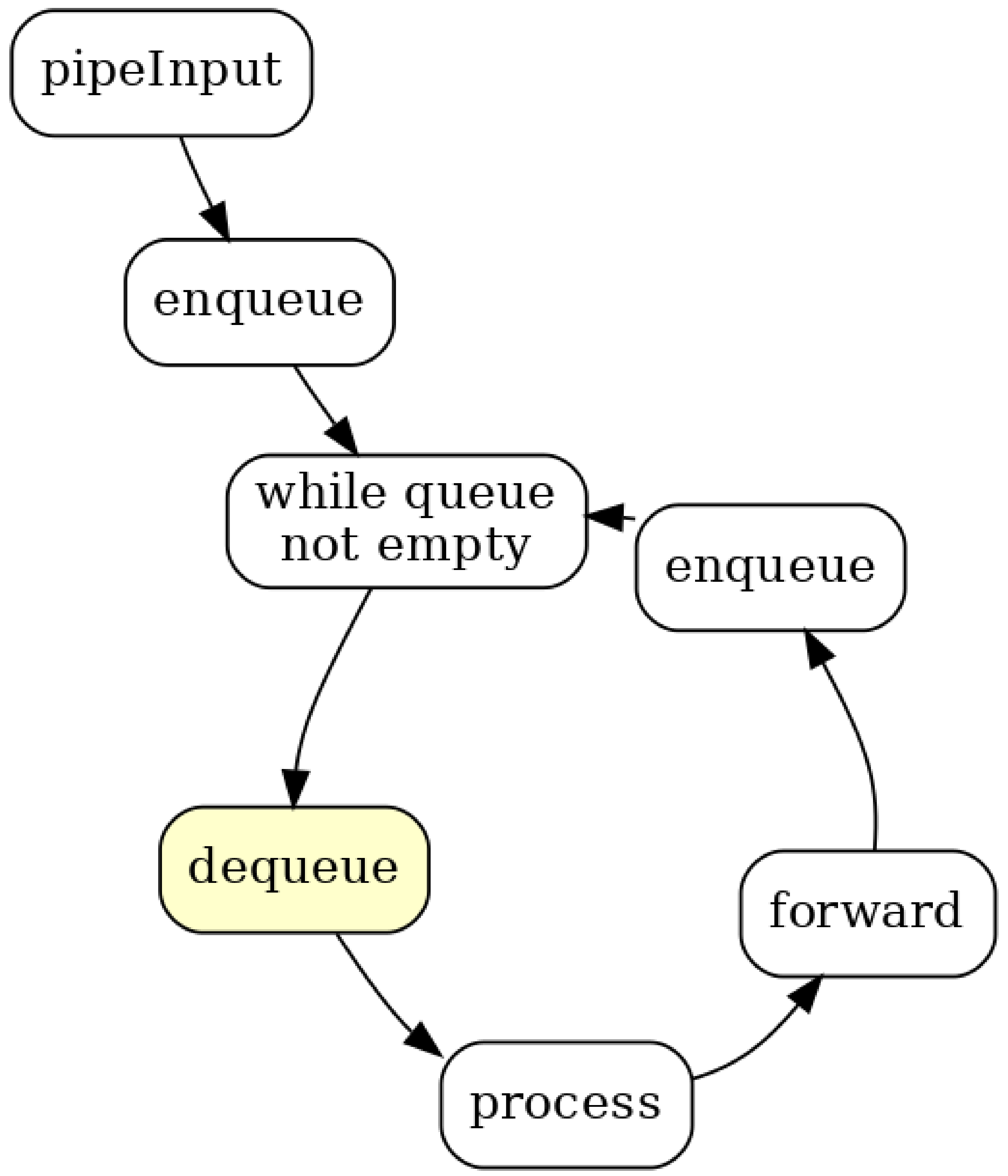
TopologyTestDriver execution loop



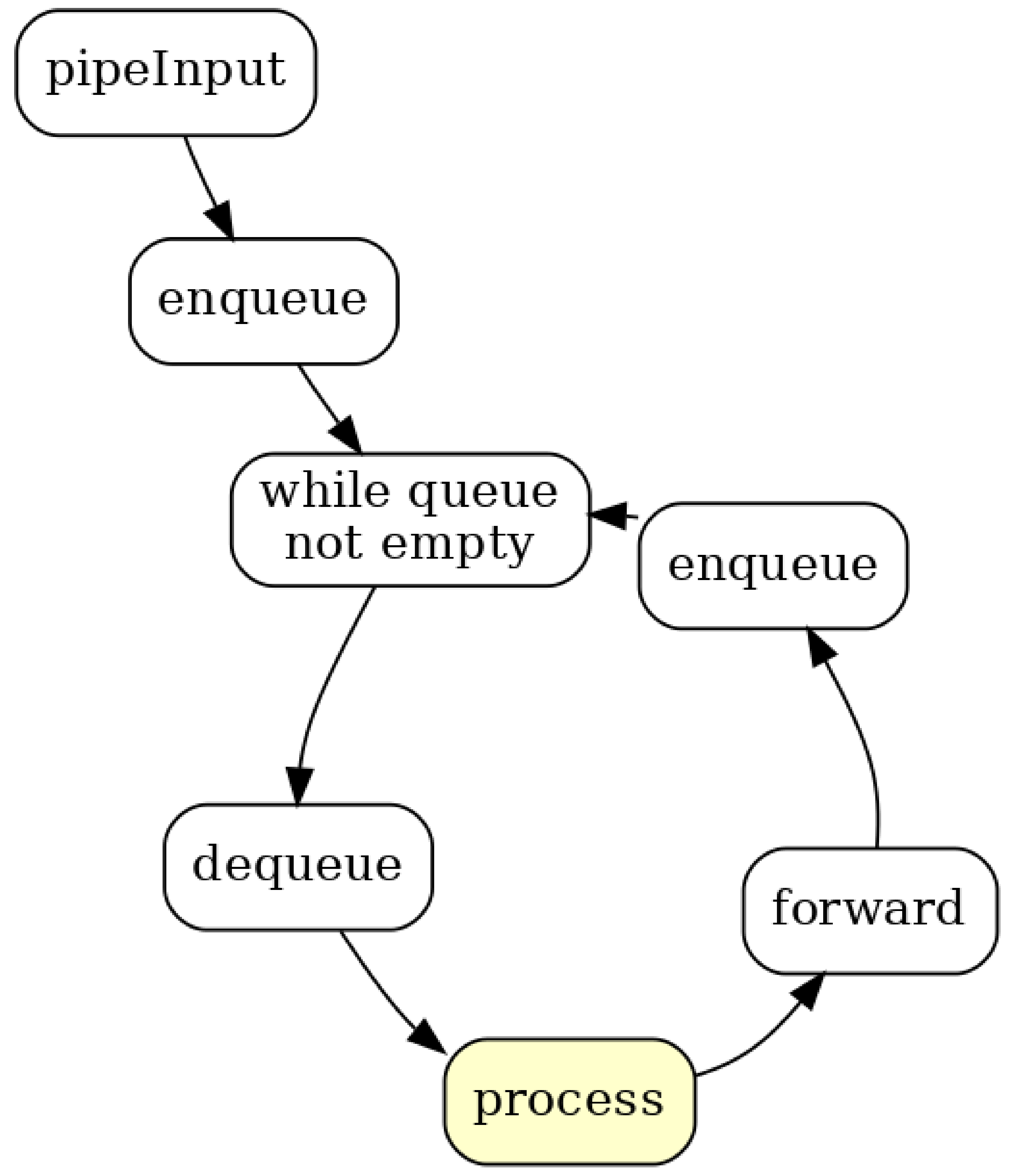
TopologyTestDriver execution loop



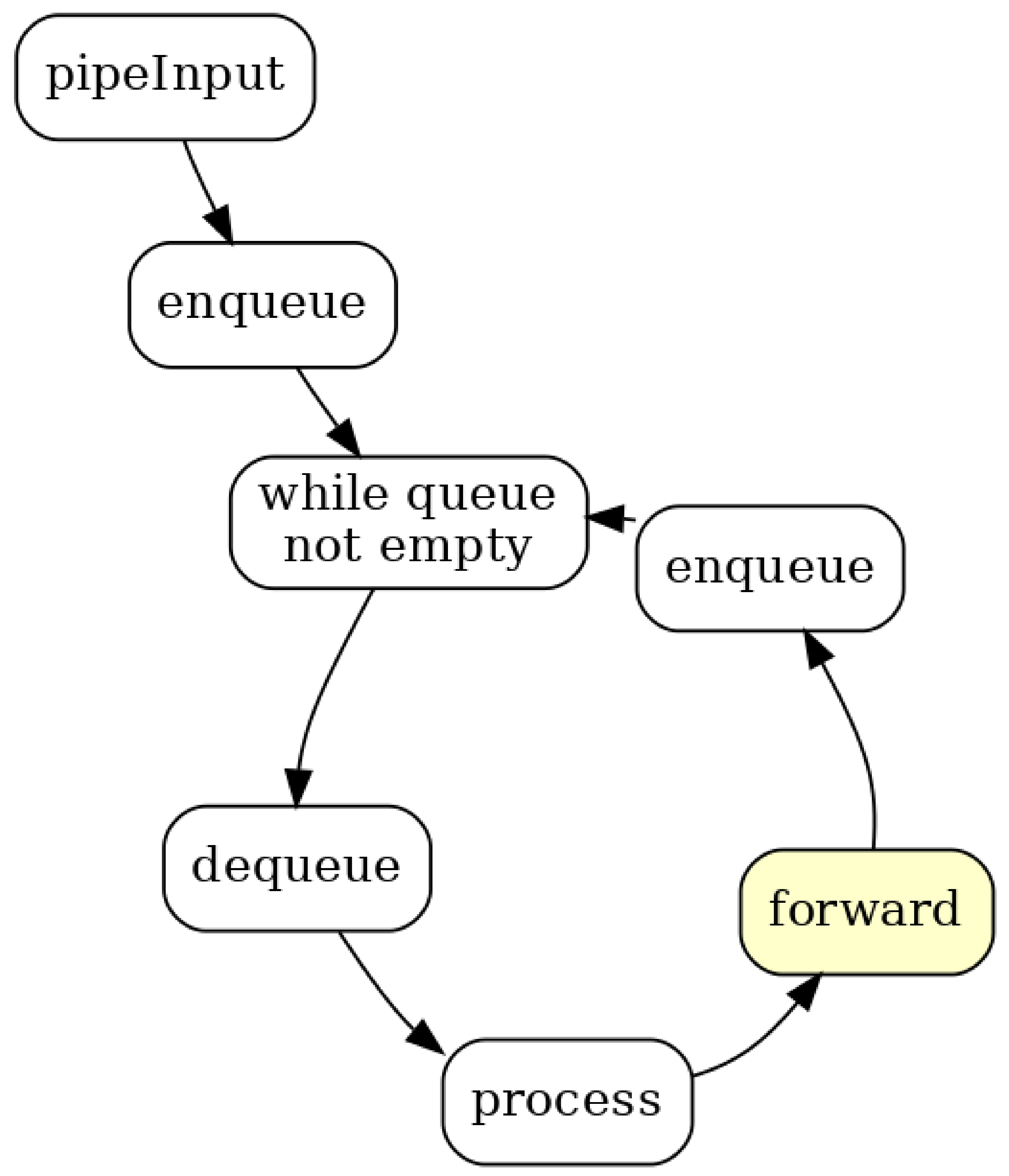
TopologyTestDriver execution loop



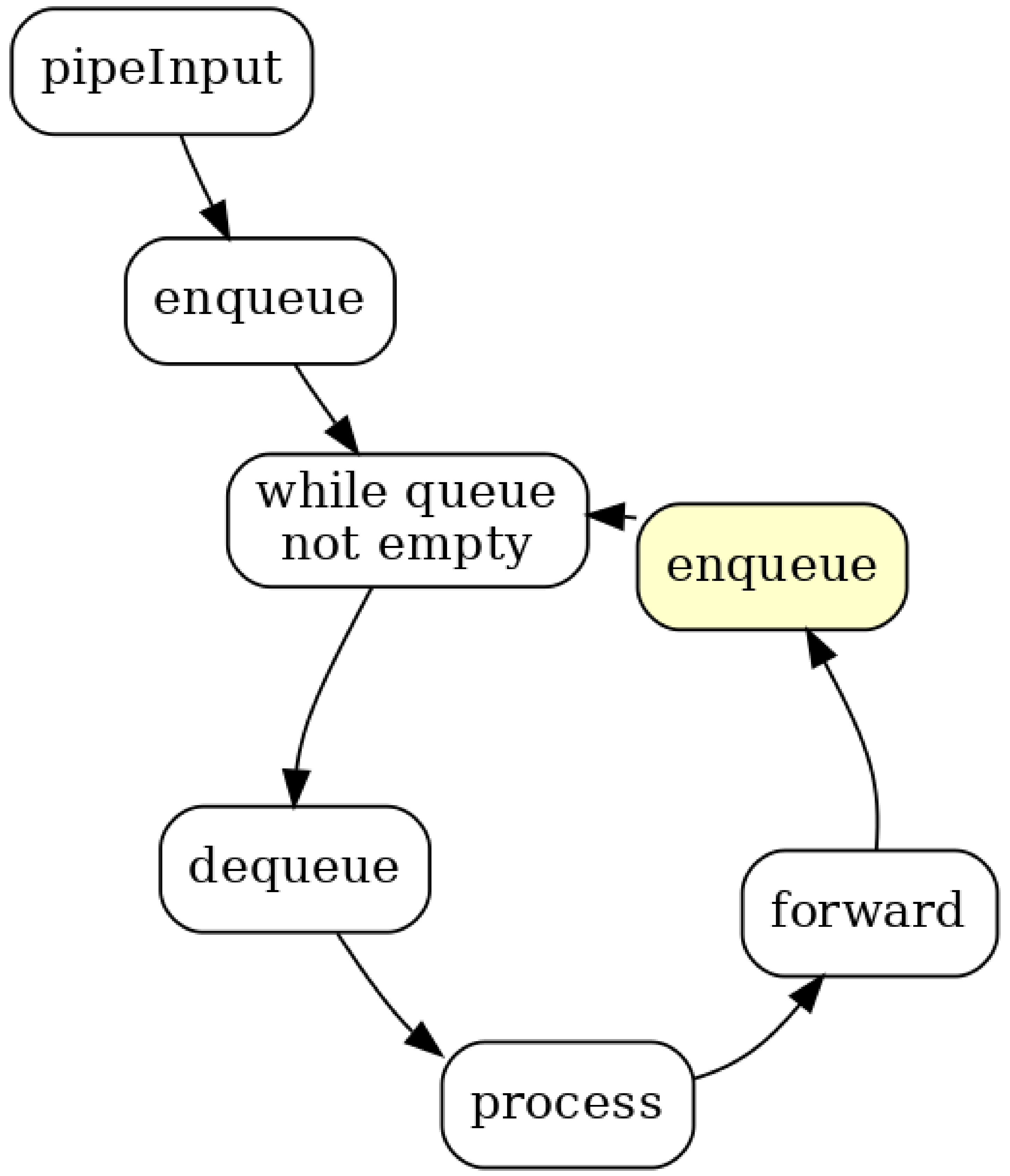
TopologyTestDriver execution loop



TopologyTestDriver execution loop



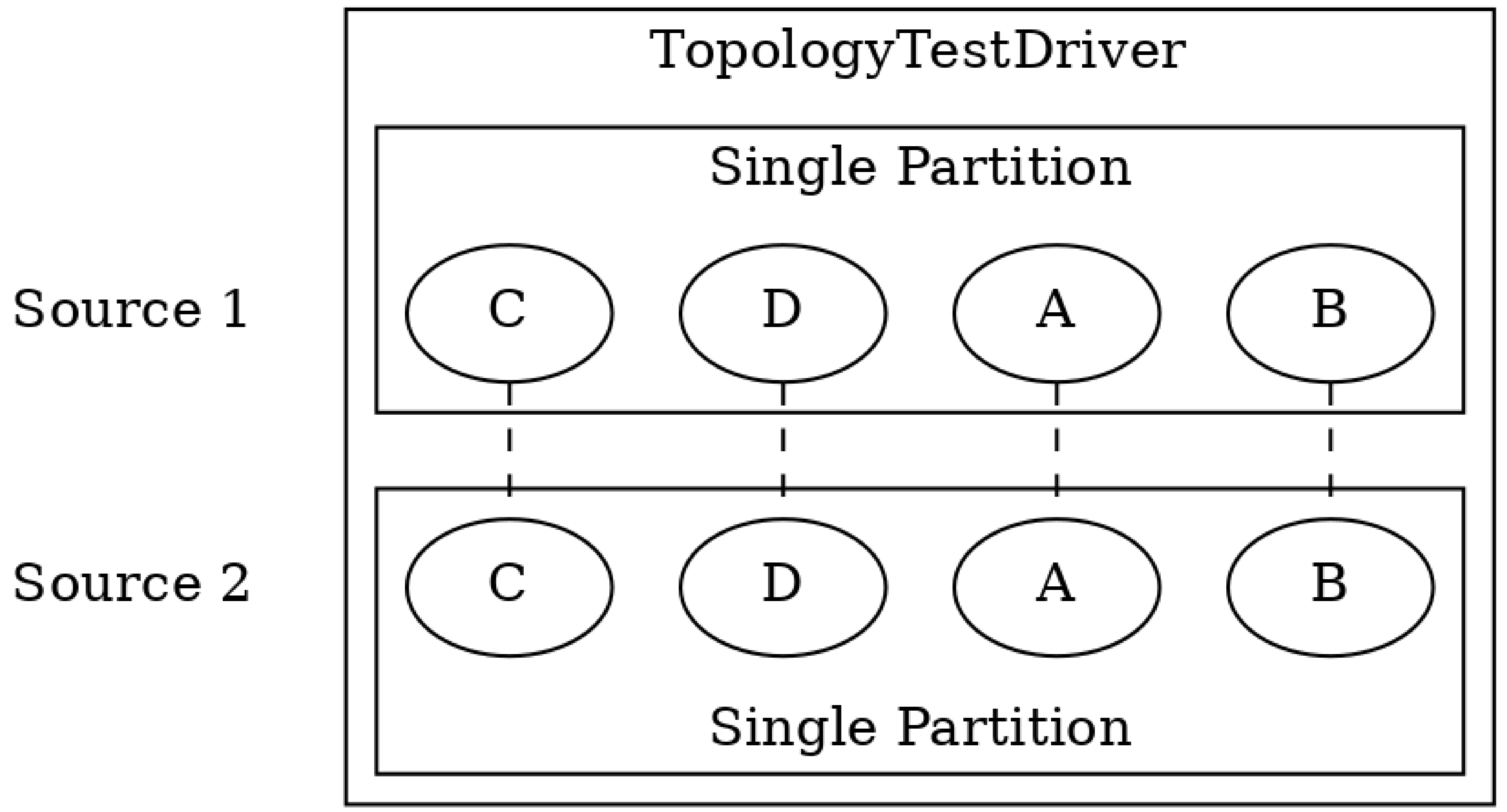
TopologyTestDriver execution loop



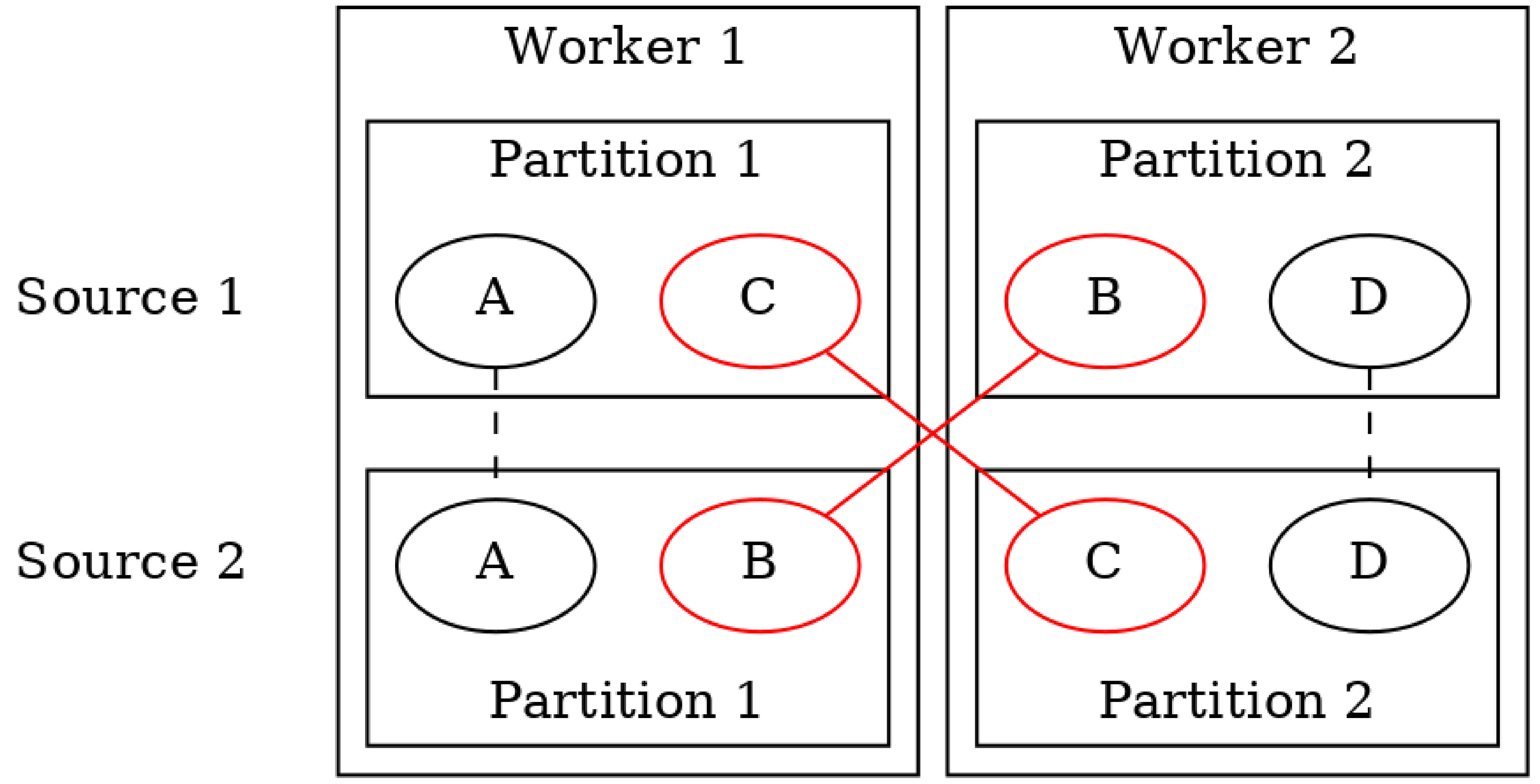
What else?

What are other problems that can’t be surfaced with `TopologyTestDriver`?

`TopologyTestDriver`: single partition

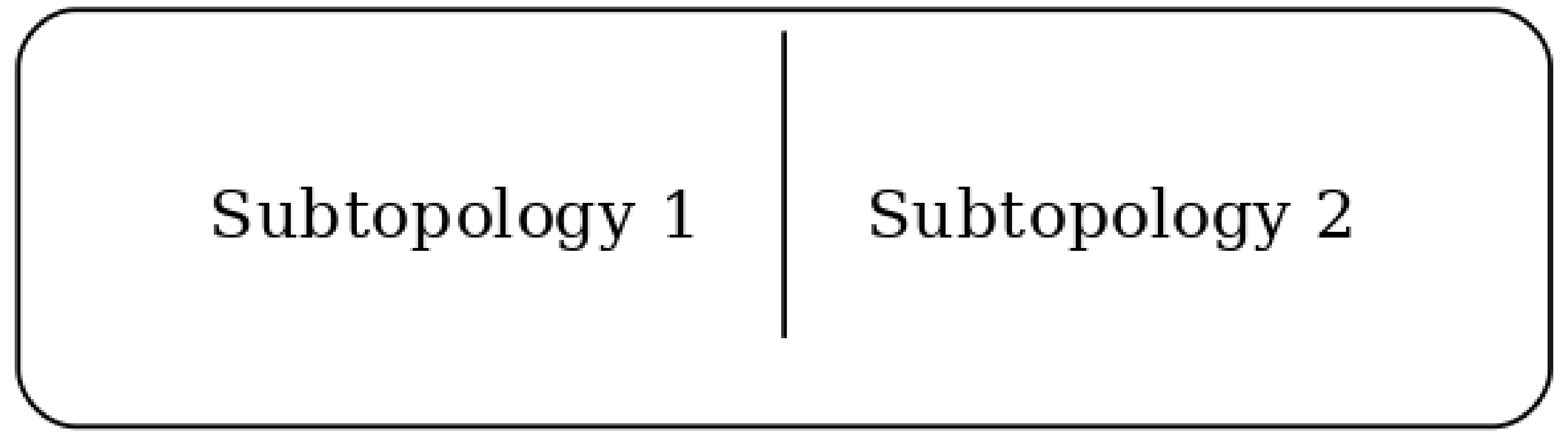


Kafka Streams: co-partitioning problems

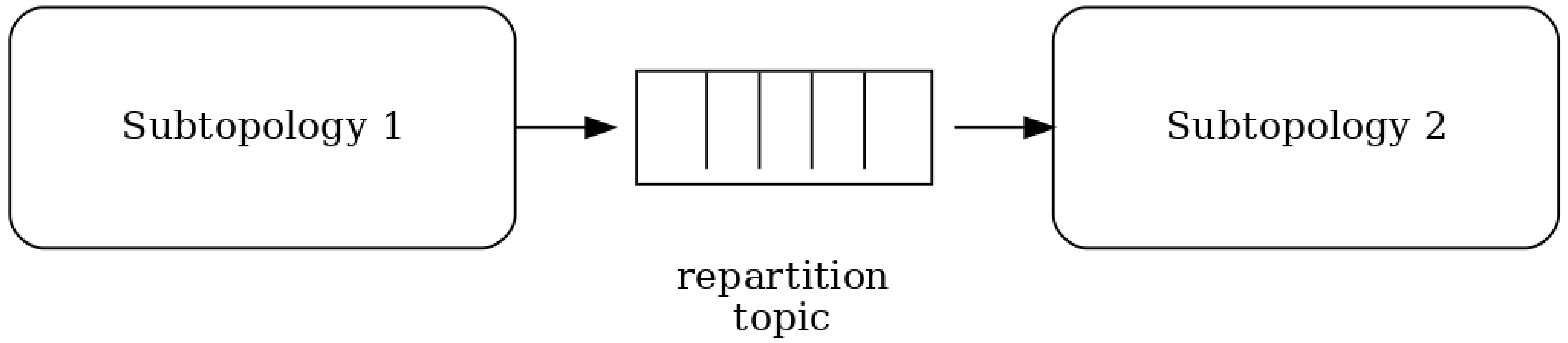


`TopologyTestDriver`: "Fused" subtopologies

`ToplogyTestDriver`

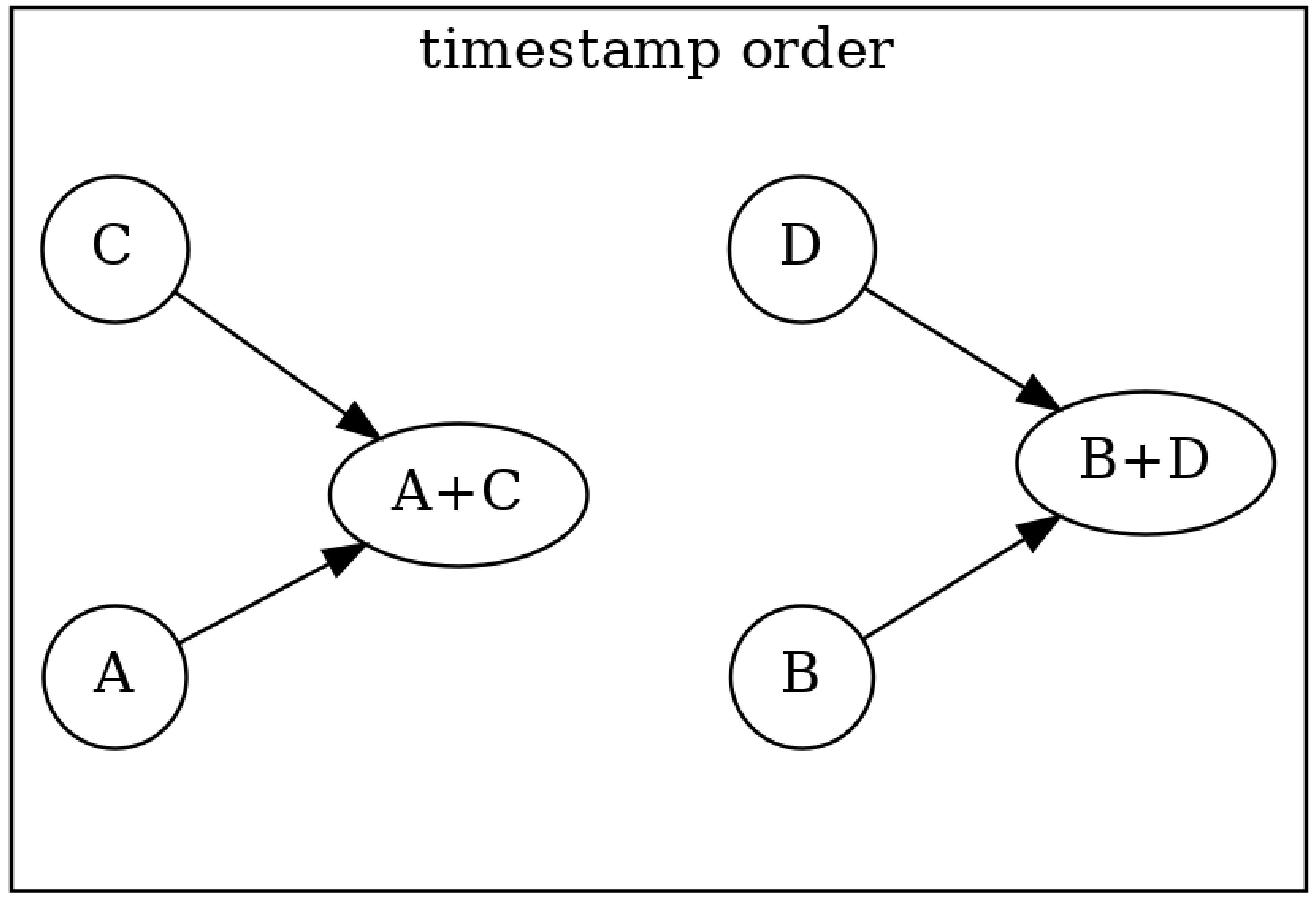


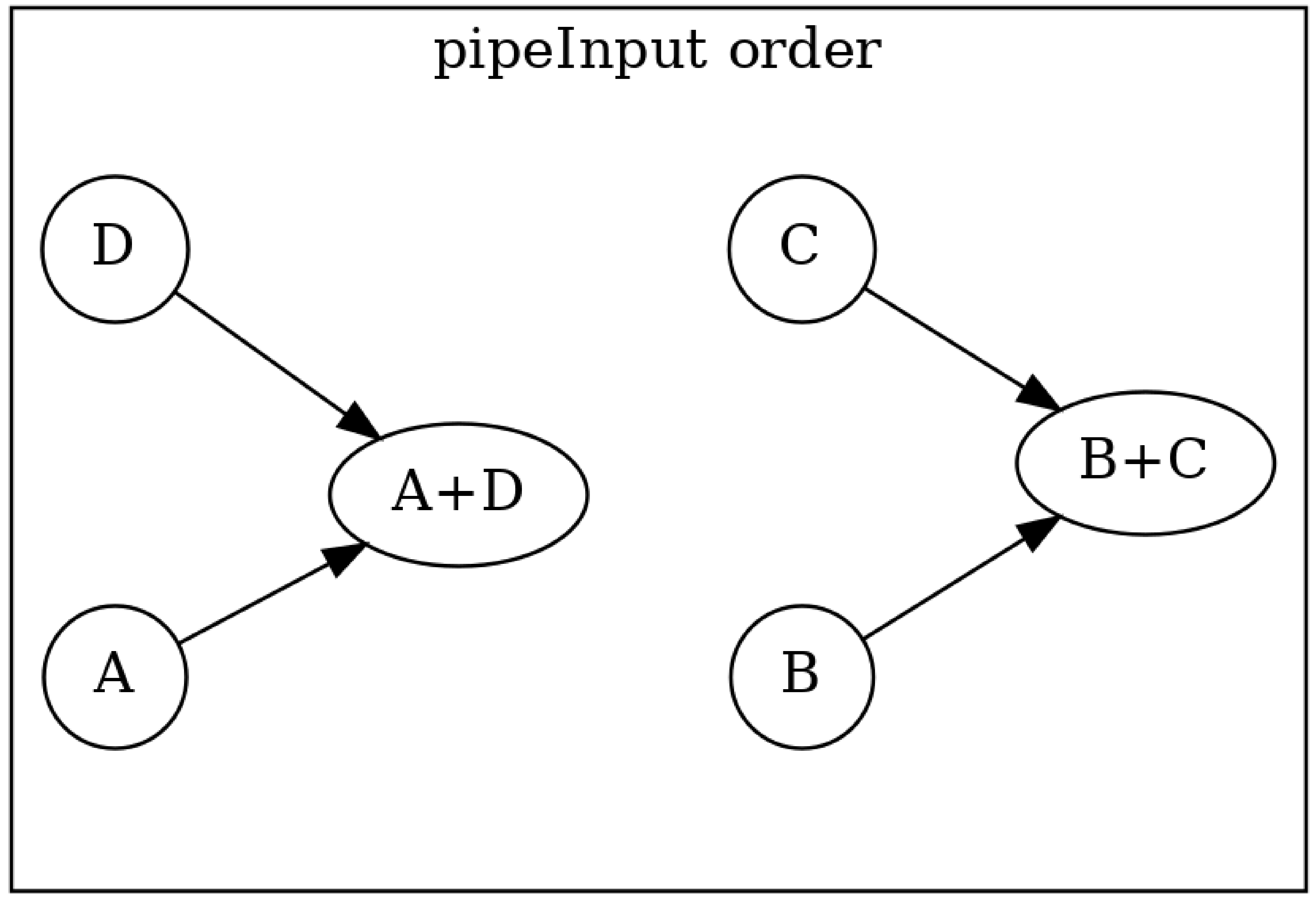
Kafka Streams



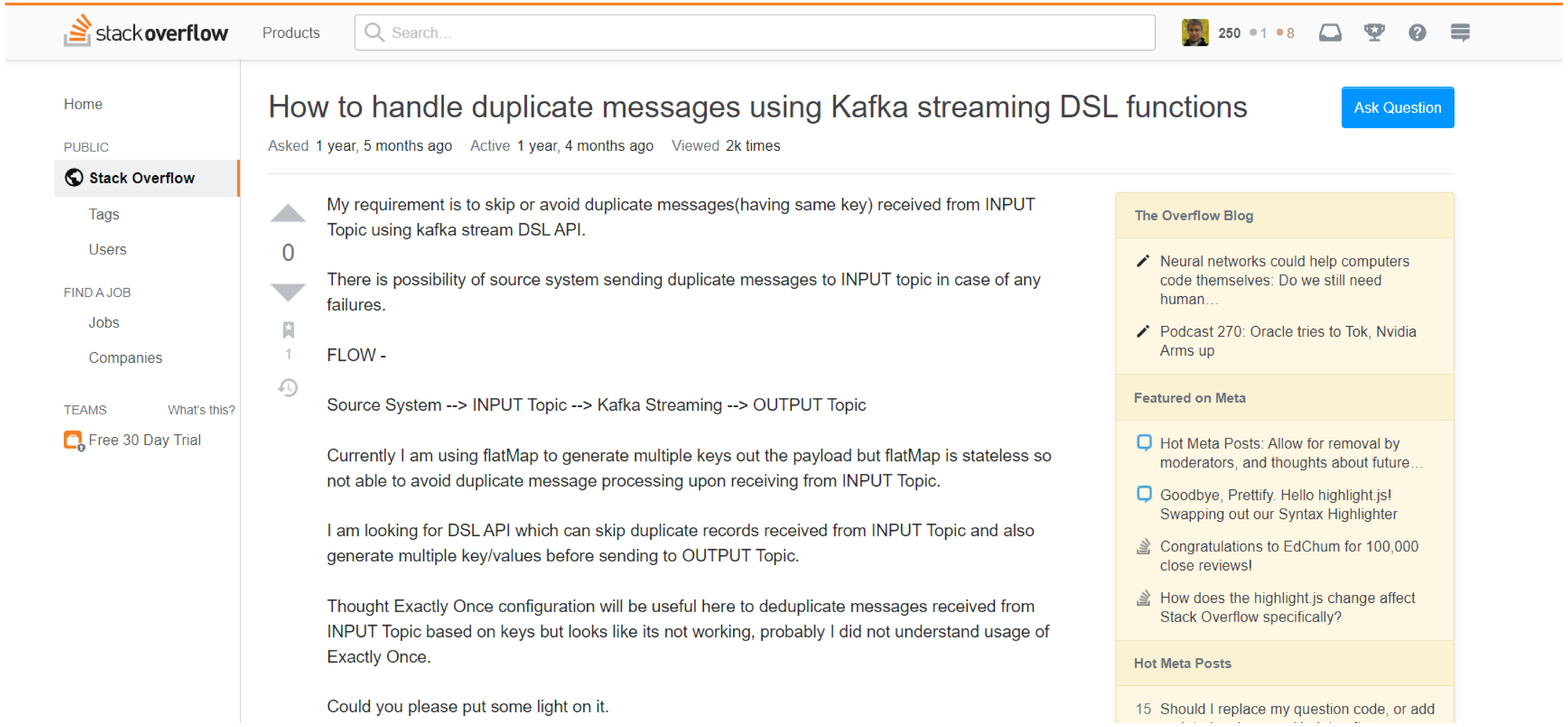
Timing

- stream-stream joins can behave differently (`pipeInput` order vs. timestamp order)
- logic that depends on stream time (such as `suppress`) can behave differently





Should we trust StackOverflow?



stackoverflow Products Search... 250 •1 •8

Home

PUBLIC

Stack Overflow

Tags

Users

FIND A JOB

Jobs

Companies

TEAMS What's this?

Free 30 Day Trial

How to handle duplicate messages using Kafka streaming DSL functions

Ask Question

Asked 1 year, 5 months ago Active 1 year, 4 months ago Viewed 2k times

0

1

My requirement is to skip or avoid duplicate messages(having same key) received from INPUT Topic using kafka stream DSL API.

There is possibility of source system sending duplicate messages to INPUT topic in case of any failures.

FLOW -

Source System --> INPUT Topic --> Kafka Streaming --> OUTPUT Topic

Currently I am using flatMap to generate multiple keys out the payload but flatMap is stateless so not able to avoid duplicate message processing upon receiving from INPUT Topic.

I am looking for DSL API which can skip duplicate records received from INPUT Topic and also generate multiple key/values before sending to OUTPUT Topic.

Thought Exactly Once configuration will be useful here to deduplicate messages received from INPUT Topic based on keys but looks like its not working, probably I did not understand usage of Exactly Once.

Could you please put some light on it.

The Overflow Blog

- Neural networks could help computers code themselves: Do we still need human…
- Podcast 270: Oracle tries to Tok, Nvidia Arms up

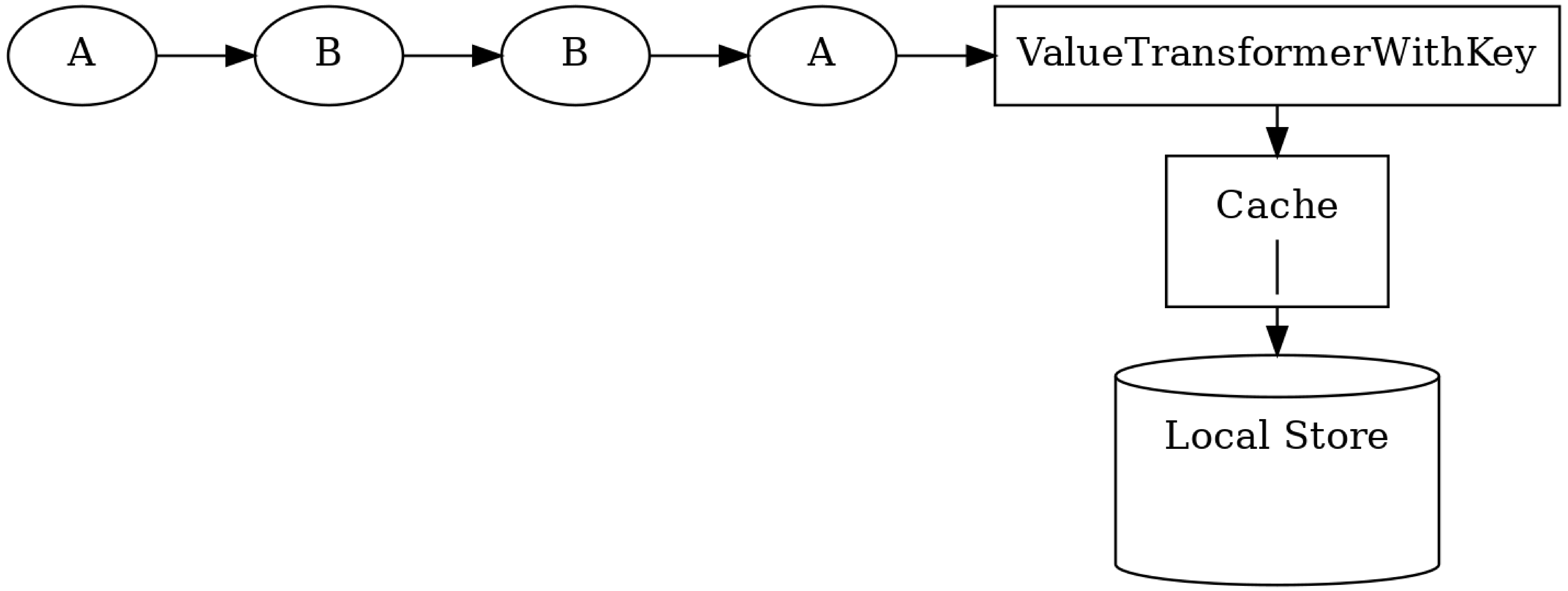
Featured on Meta

- Hot Meta Posts: Allow for removal by moderators, and thoughts about future…
- Goodbye, Prettify. Hello highlight.js! Swapping out our Syntax Highlighter
- Congratulations to EdChum for 100,000 close reviews!
- How does the highlight.js change affect Stack Overflow specifically?

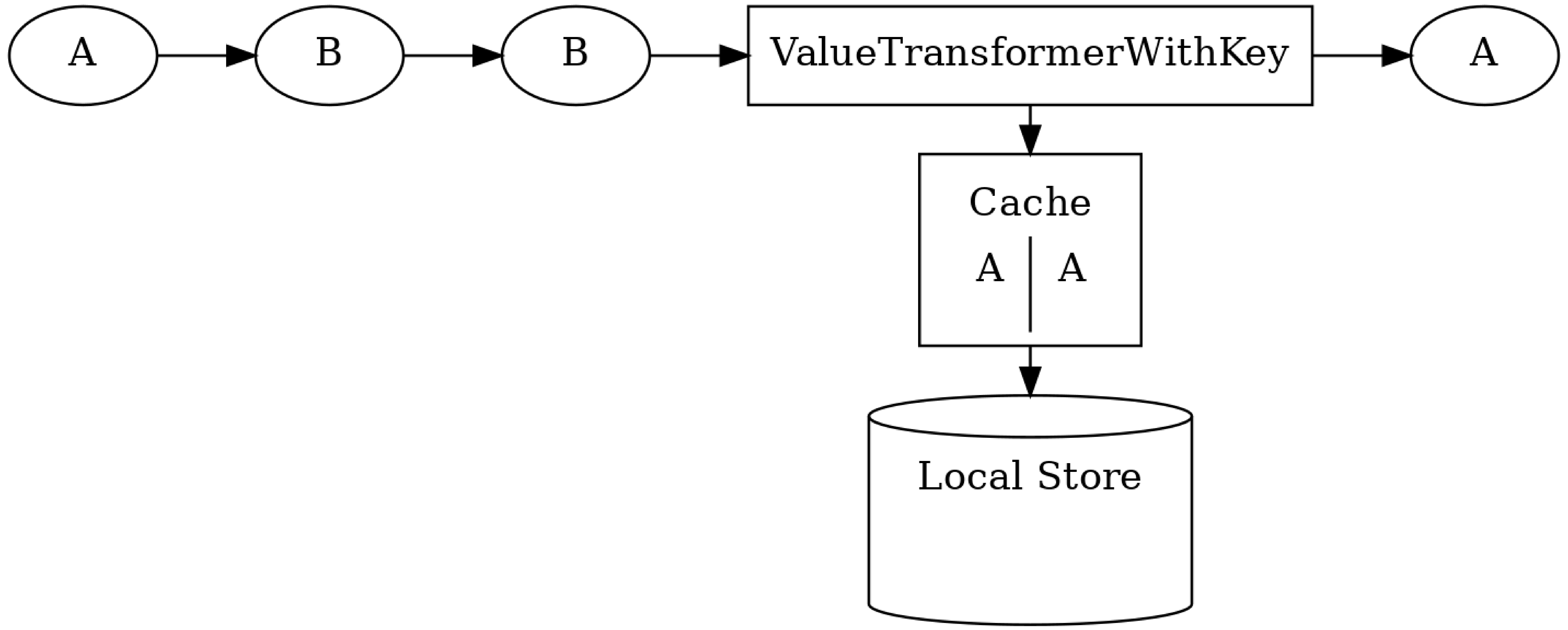
Hot Meta Posts

15 Should I replace my question code, or add

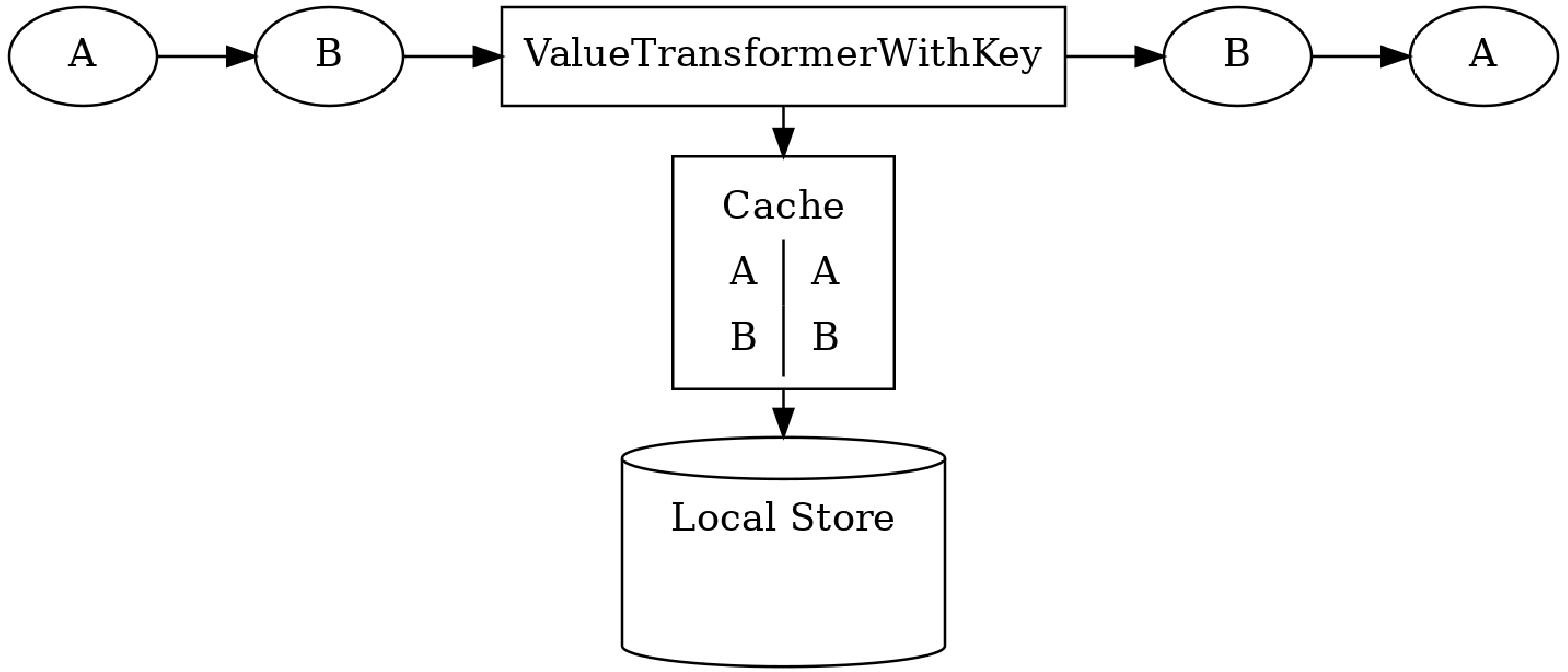
Using Transformer



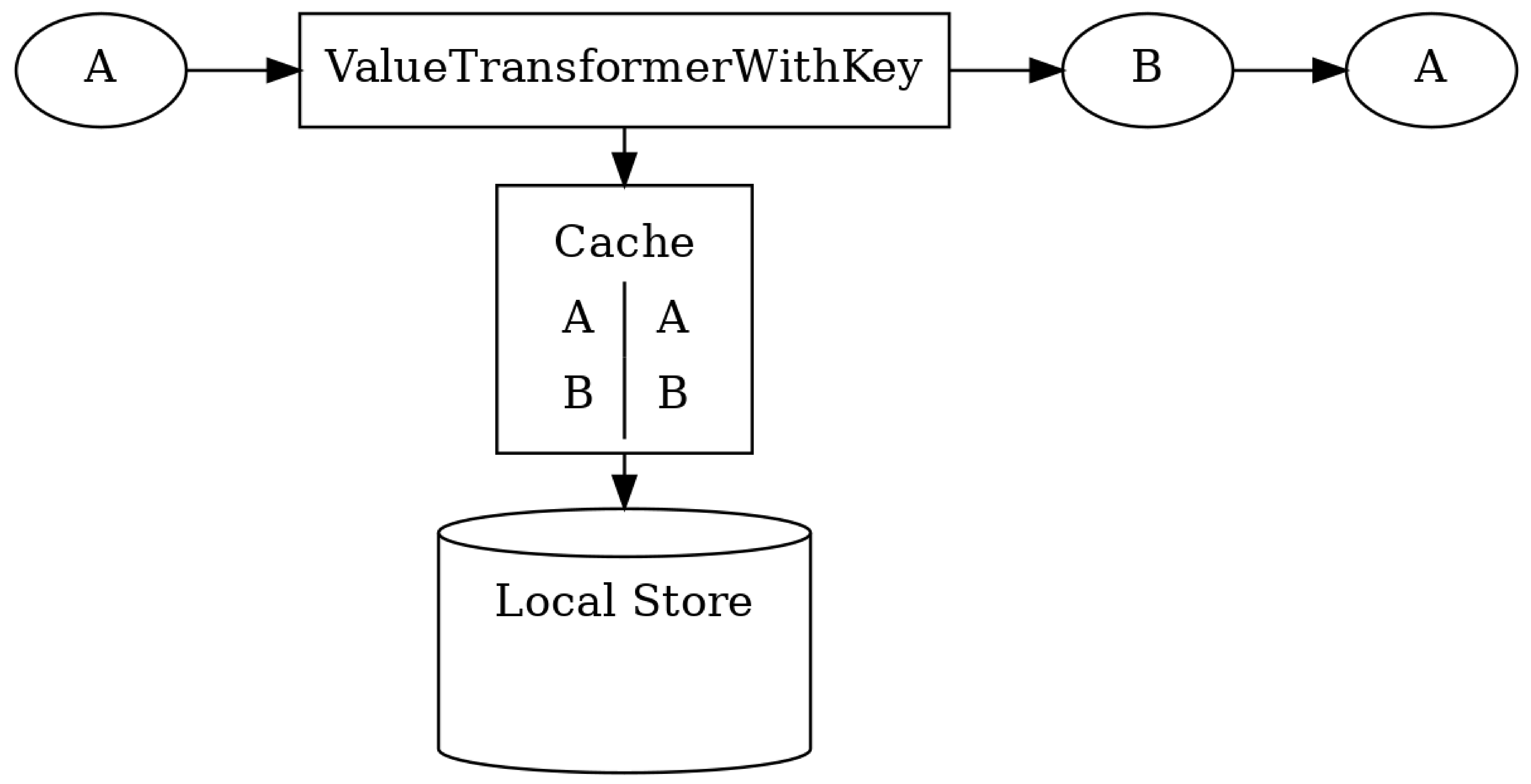
Using Transformer



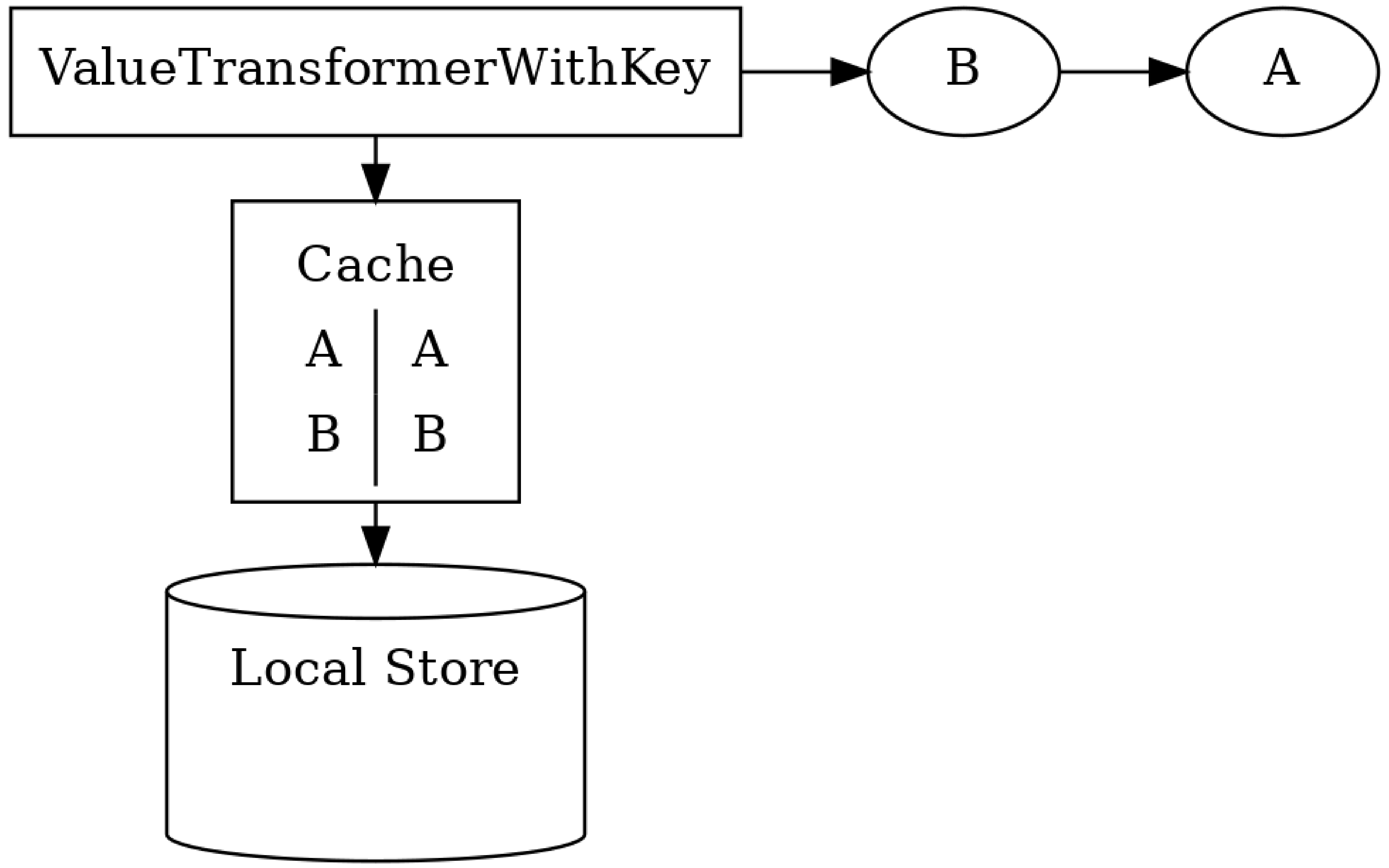
Using Transformer



Using Transformer



Using Transformer



Let’s run tests on real Kafka!

- EmbeddedKafka
- TestContainers

EmbeddedKafka vs TestContainers

EmbeddedKafka	TestContainers
• Pro: ▪ Just pull in a dependency • Contra: ▪ Pulls in Scala ▪ Runs in the same JVM	• Pro ▪ Runs Kafka isolated in Docker ▪ Not only for Kafka testing • Contra ▪ Needs Docker ▪ Requires some time for the first start

Demo

- Writing TestContainers test
  - An easy part: pushing messages to Kafka
  - A not so easy part: how do we check the output?

Demo

- Deduplication: the correct implementation
- Now the test is green, but takes 5 seconds!

Does it have to be so slow?

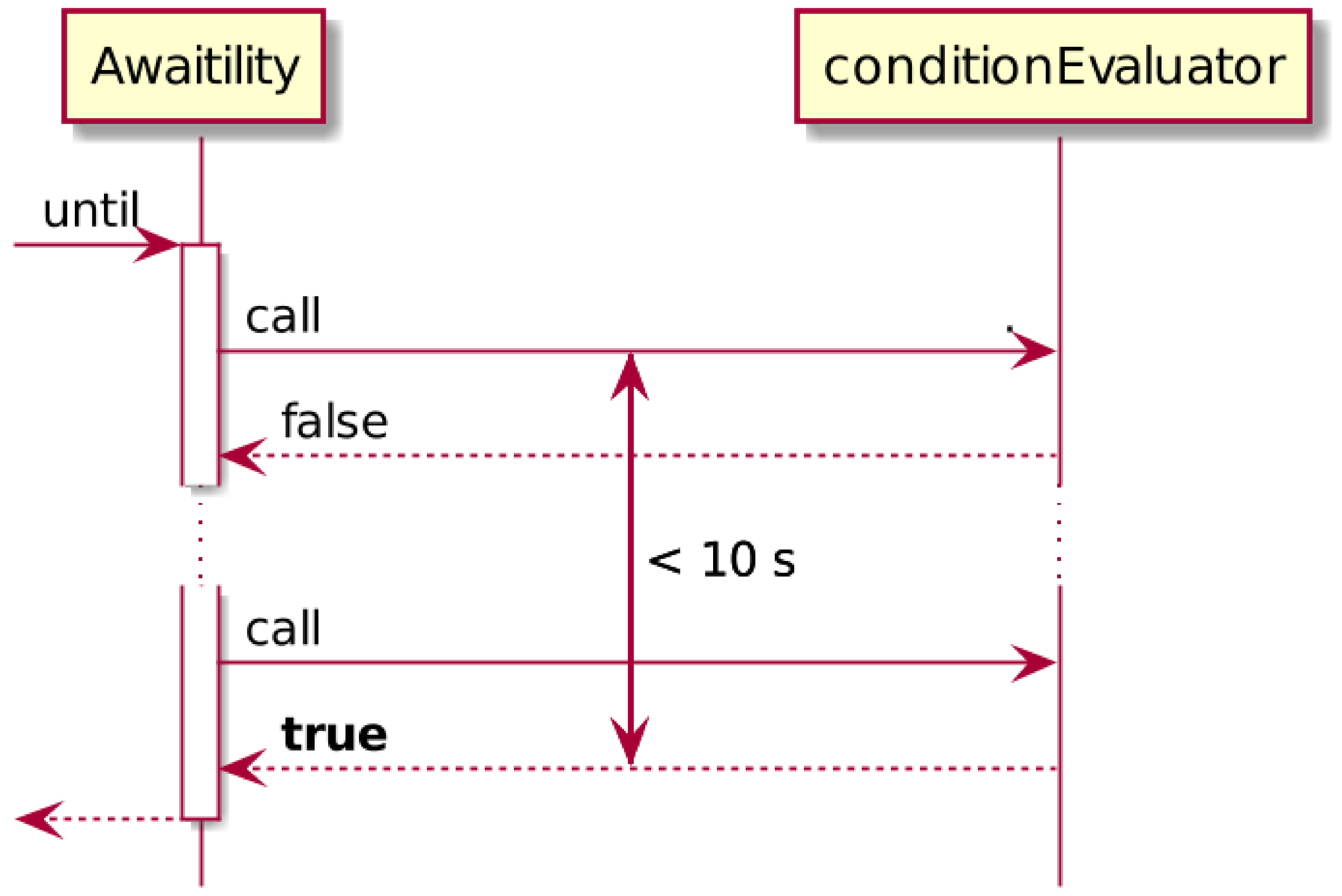
```
List actual = new ArrayList<>();

while (true) {
  ConsumerRecords<String, String> records =
    KafkaTestUtils.getRecords(consumer, 5000 /* timeout in ms */);
  if (records.isEmpty()) break;
  for (ConsumerRecord<String, String> rec : records) {
    actual.add(rec.value());
  }
}

assertEquals(List.of("A", "B"), actual);
```

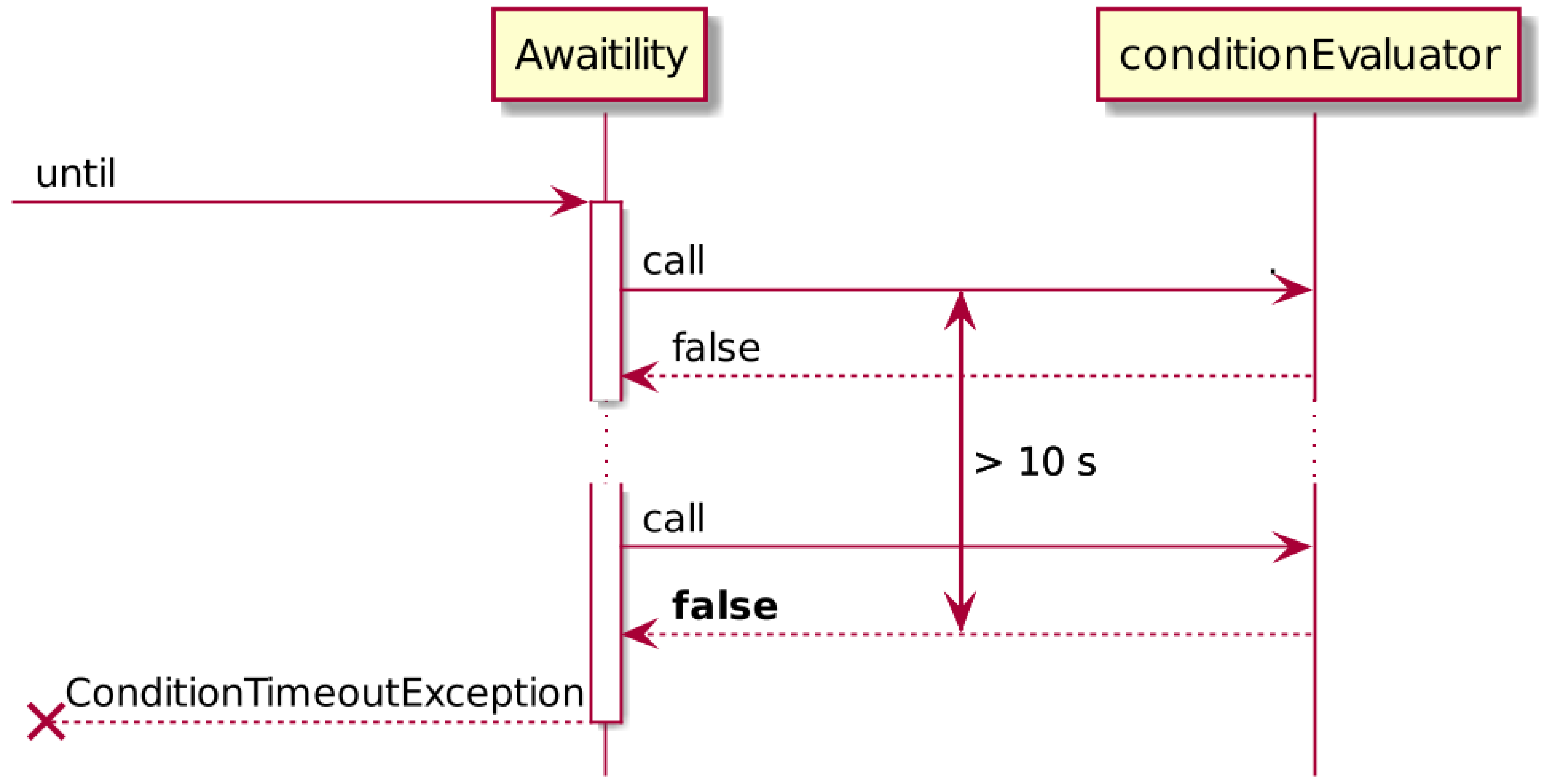
Awaitility

```
Awaitility.await().atMost(10, SECONDS).until(
                  () -> List.of("A", "B").equals(actual));
```



Awaitility

```
Awaitility.await().atMost(10, SECONDS).until(
                    () -> List.of("A", "B").equals(actual));
```



Things we must keep in mind

- Cooperative termination
- Thread-safe data structure

Demo

- Green test runs faster

Will any extra messages appear?

- We can wait for extra 5 seconds (bad choice)
- We can put a 'marker record' at the end of the input and wait for it to appear in the output (not always possible)

Summary

Summary

- **Both** `TopologyTestDriver` and integration tests are needed

Summary

- **Both** `TopologyTestDriver` and integration tests are needed
- Write unit tests with `TopologyTestDriver`. When it fails to surface the problem, use integration tests.

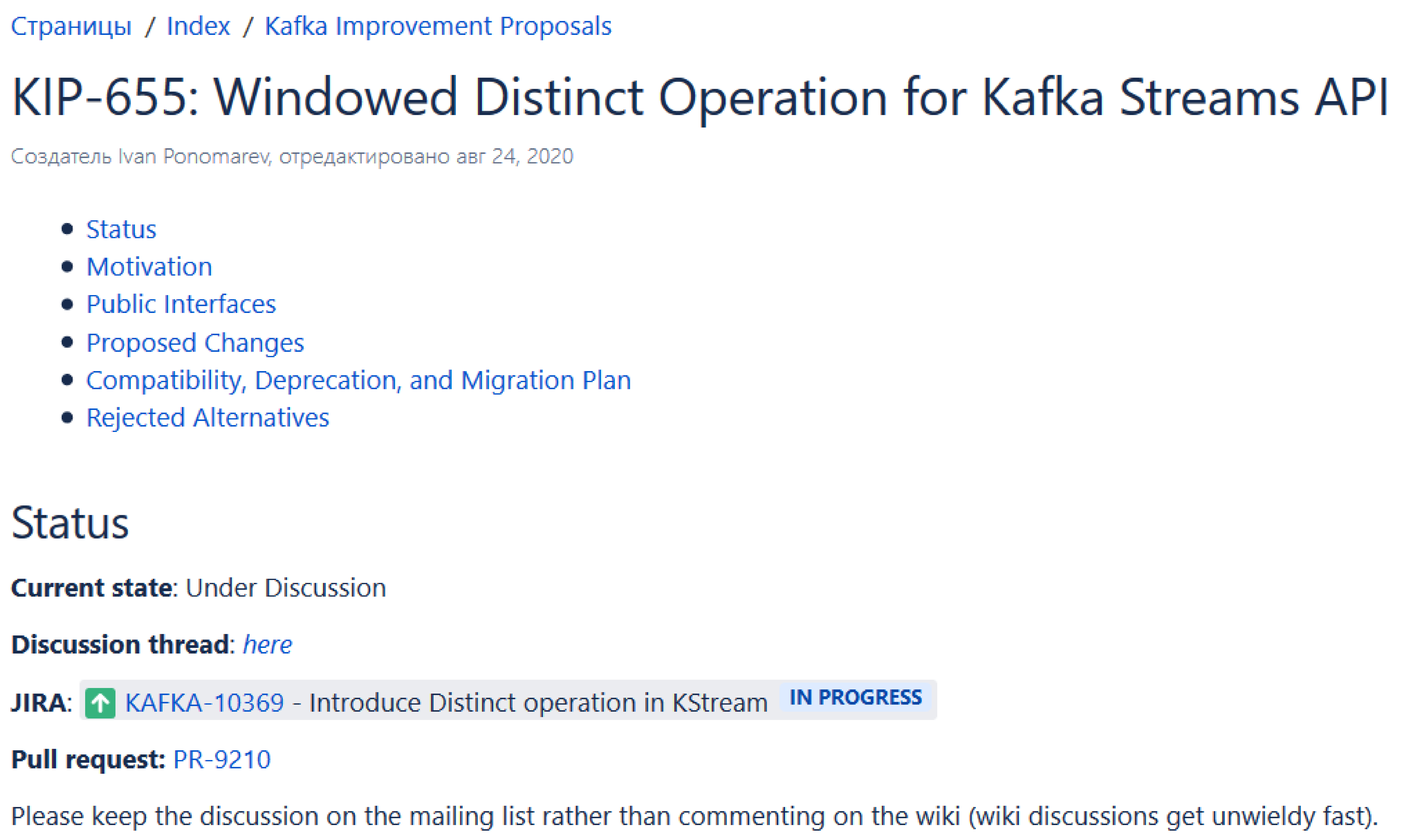
Summary

- **Both** `TopologyTestDriver` and integration tests are needed
- Write unit tests with `TopologyTestDriver`. When it fails to surface the problem, use integration tests.
- Know the limitations of `TopologyTestDriver`.

Summary

- **Both** `TopologyTestDriver` and integration tests are needed
- Write unit tests with `TopologyTestDriver`. When it fails to surface the problem, use integration tests.
- Know the limitations of `TopologyTestDriver`.
- Understand the difficulties and limitations of asynchronous testing.

KIP-655 is under discussion



Страницы / Index / Kafka Improvement Proposals

KIP-655: Windowed Distinct Operation for Kafka Streams API

Создатель Ivan Ponomarev, отредактировано авг 24, 2020

Status

Current state: Under Discussion

Discussion thread: *here*

JIRA: KAFKA-10369 - Introduce Distinct operation in KStream **IN PROGRESS**

Pull request: PR-9210

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Useful links

- Confluent blog: Testing Kafka Streams – A Deep Dive
- **pro.kafka**: Russian Kafka chat in Telegram: https://t.me/proKafka
- Confluent community Slack: https://cnfl.io/slack

Thank you!

Ivan Ponomarev

- iponomarev@curs.ru
- @inponomarev
- inponomarev

John Roesler

- john@confluent.io
- vvcephei@apache.org
- vvcephei