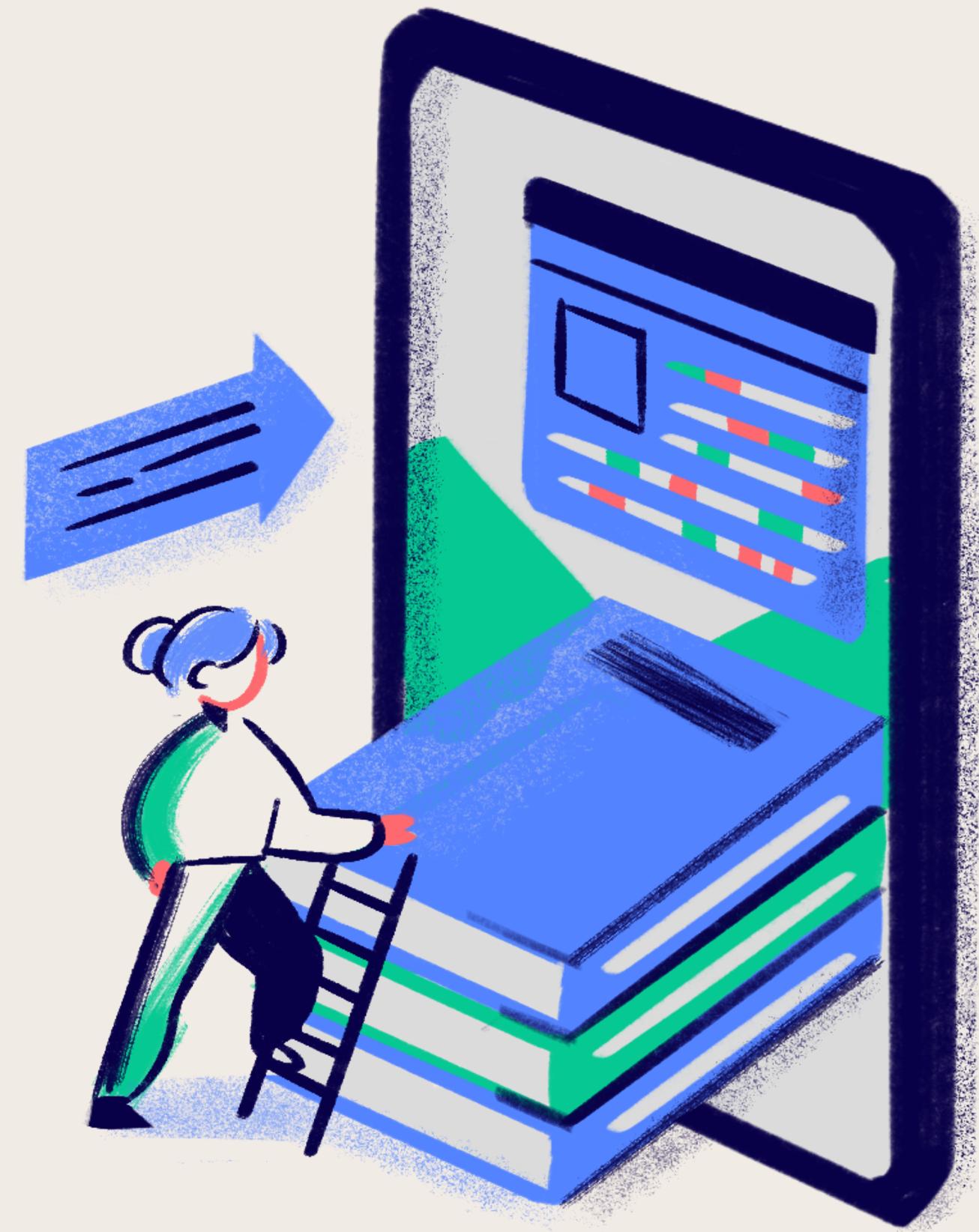


APACHE BEAM

GVHD: THS. NGUYỄN HỒ DUY TRI

TRƯỜNG VĨNH THUẬN
HOÀNG QUỐC VIỆT
PHÙNG THIÊN PHÚC

21522653
21522790
21521297



1. Giới thiệu chung

2. Đặc trưng, ưu/nhược điểm

3. Ứng dụng thực tế

4. So sánh với Spark

5. Demo



1

GIỚI THIỆU CHUNG

APACHE BEAM



- Apache Beam (Batch + strEAM) là một framework mã nguồn mở dùng để xây dựng các luồng dữ liệu có thể chạy trên nhiều hệ thống xử lý dữ liệu phân tán khác nhau.
- Beam cung cấp một API thống nhất để lập trình các pipeline xử lý dữ liệu theo cả hai chế độ: batch processing và stream processing

APACHE BEAM

Java

```
input.apply(  
    Sum.integersPerKey())
```

Python

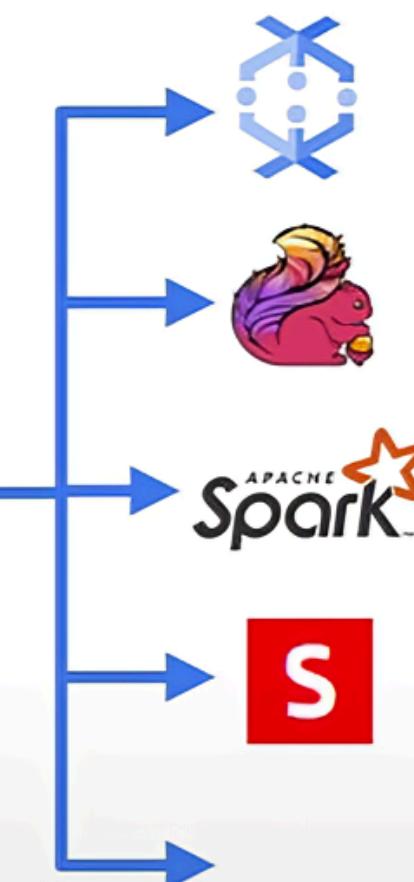
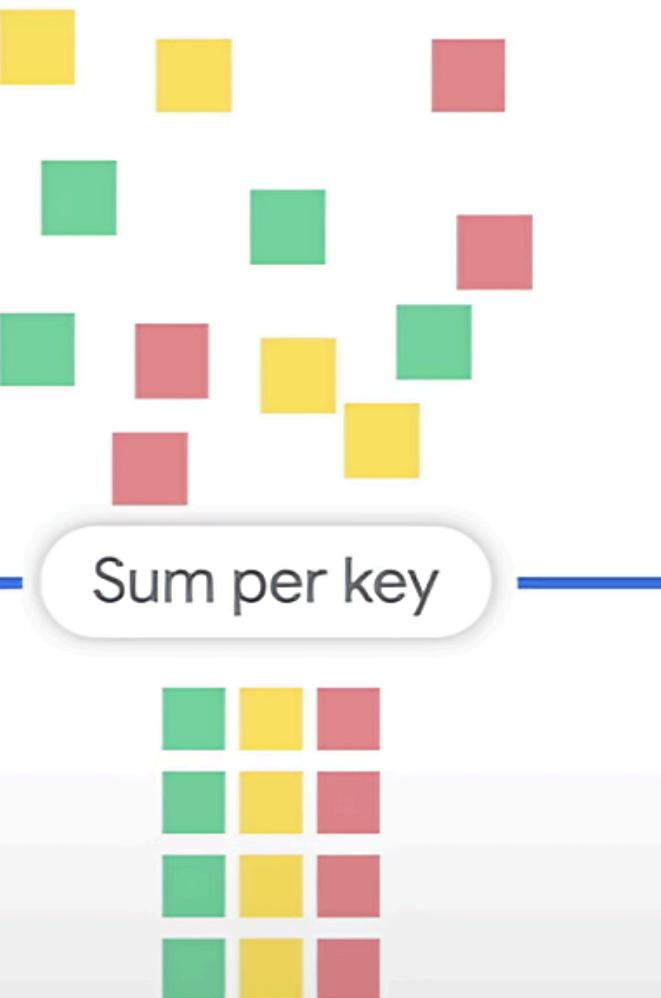
```
input | Sum.PerKey()
```

Go

```
stats.Sum(s, input)
```

SQL

```
SELECT key, SUM(value)  
FROM input GROUP BY key
```



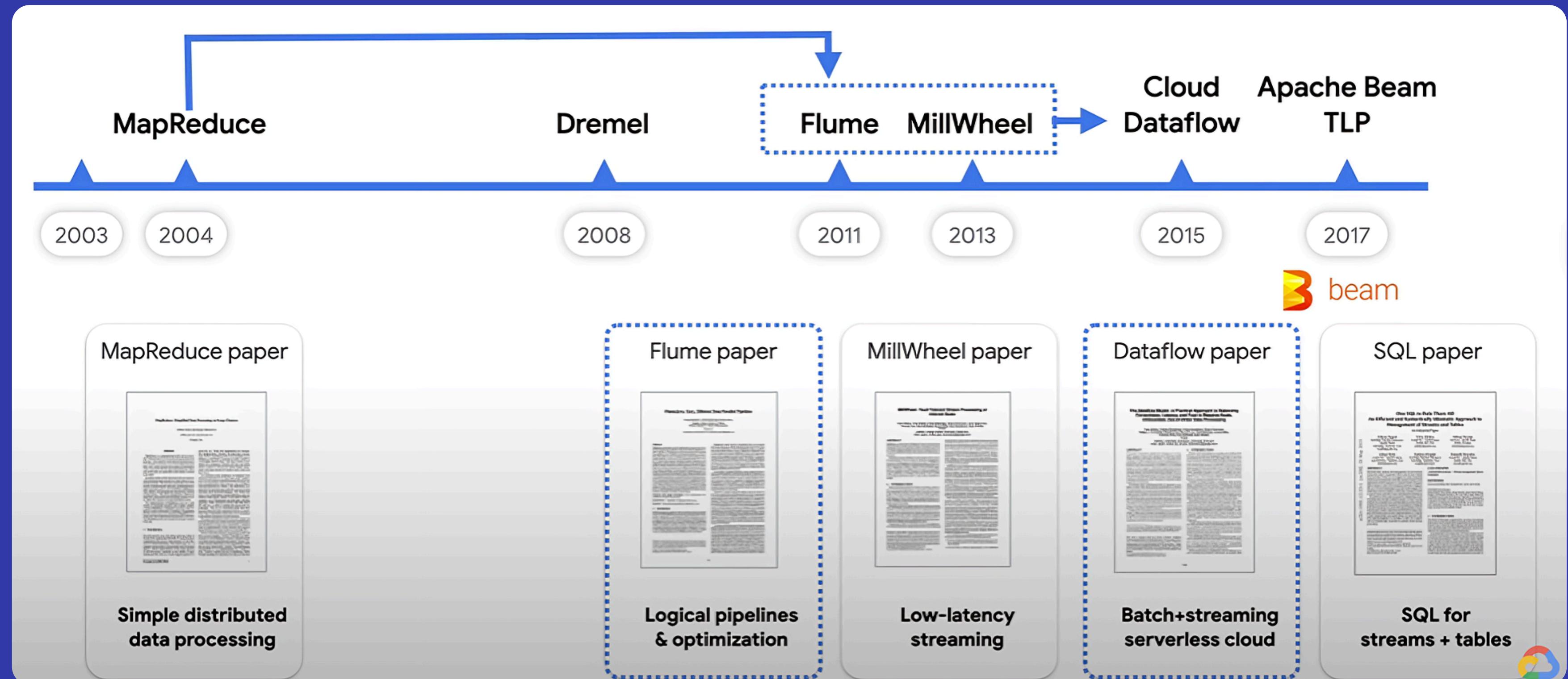
Cloud Dataflow

Apache Flink

Apache Spark

Apache Samza

SỰ RA ĐỜI CỦA APACHE BEAM



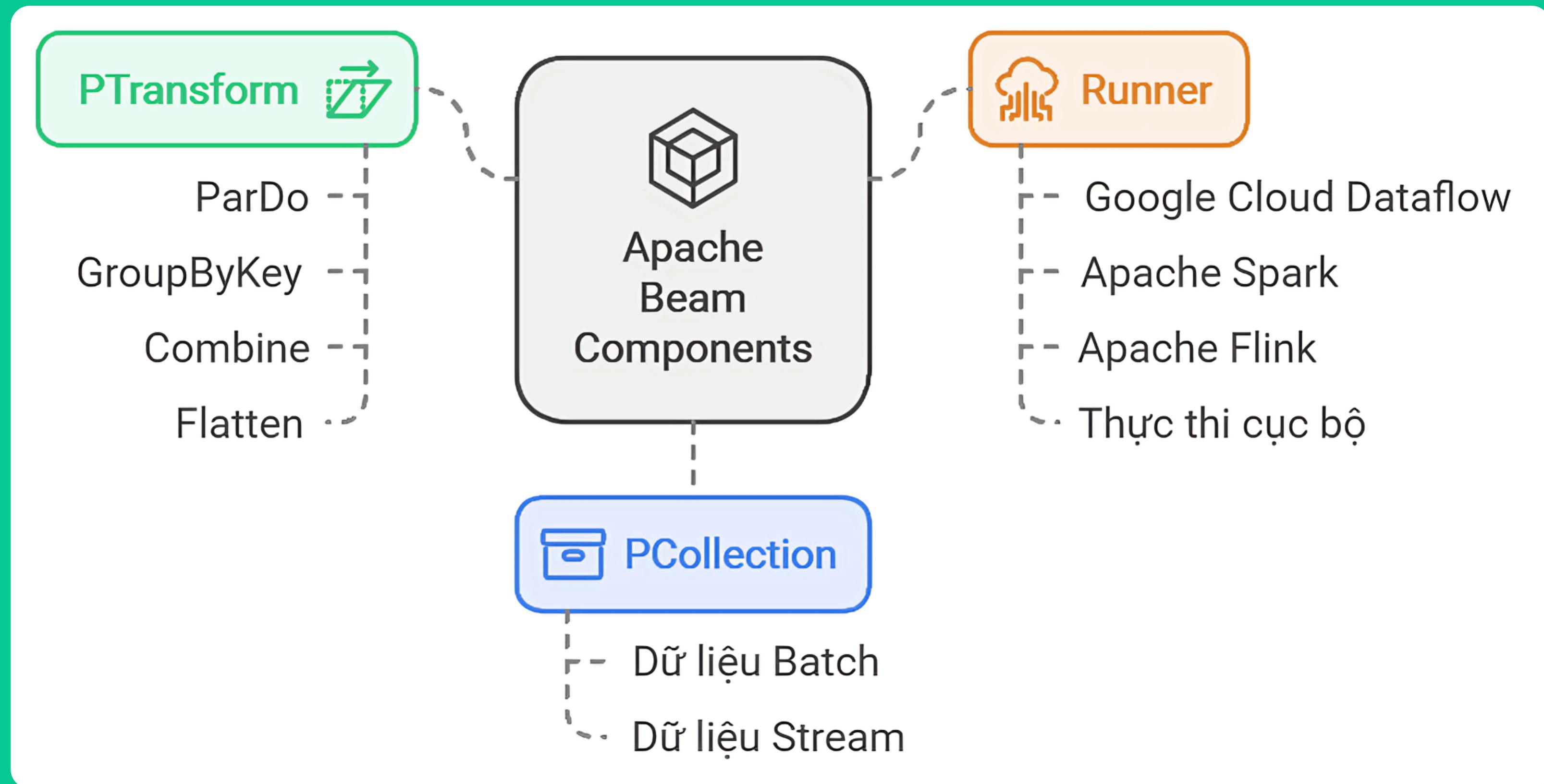
*Nguồn: googlecloudtech

TẠI SAO NÊN SỬ DỤNG APACHE BEAM

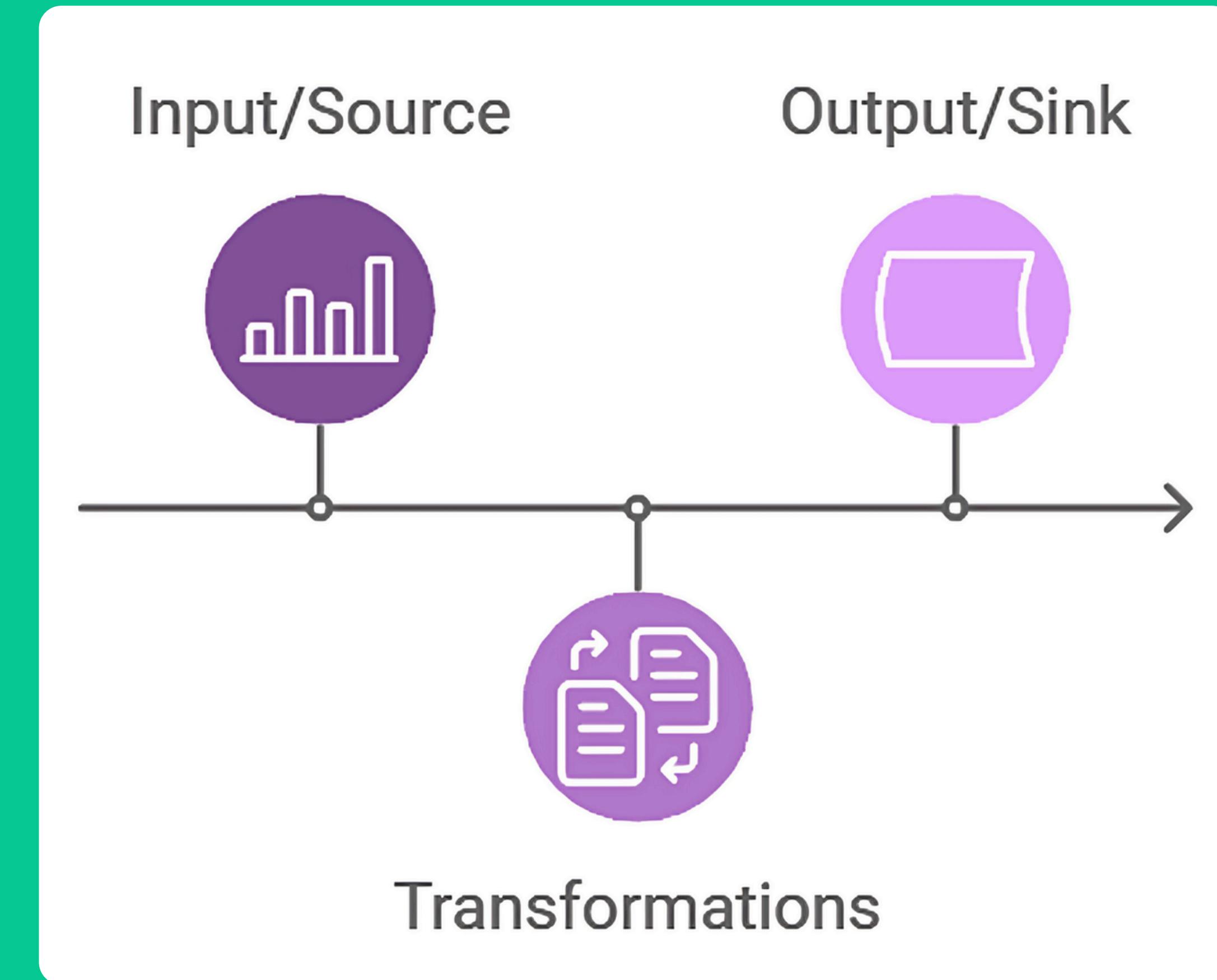
1. Platform Independence
2. Unified Model
3. Scalability
4. Flexibility



THÀNH PHẦN TRONG APACHE BEAM PIPELINE



CÁCH HOẠT ĐỘNG CỦA BEAM PIPELINE



2

ĐẶC TRƯNG ƯU-NHƯỢC ĐIỂM

ĐẶC TRƯNG CỦA APACHE BEAM

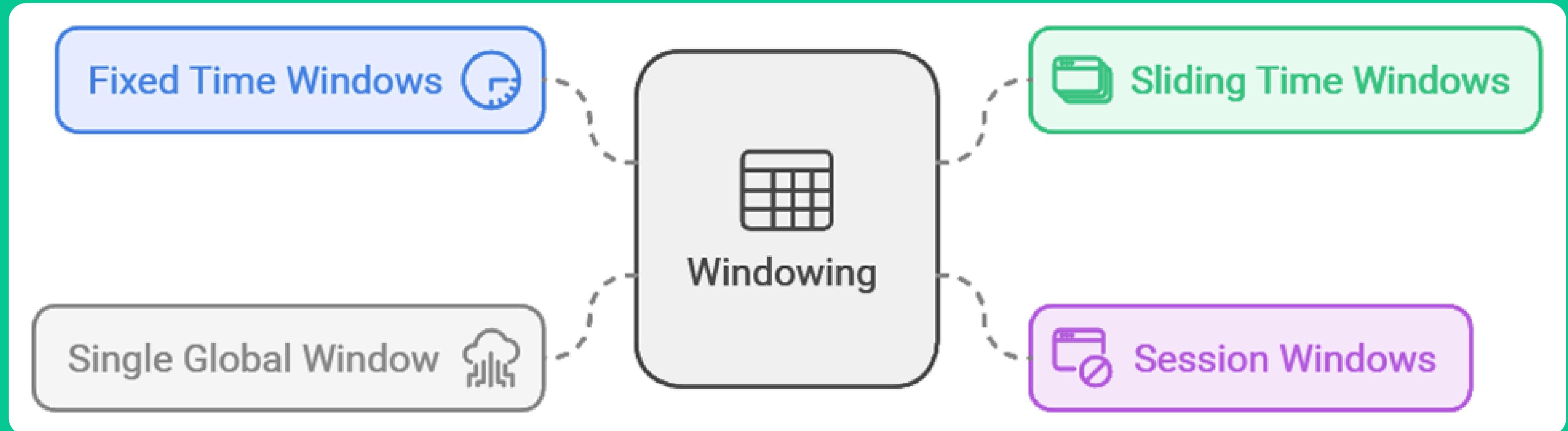
WINDOWING

Windowing chia nhỏ một PCollection dựa trên dấu thời gian của các phần tử trong nó.

Các phép biến đổi (transforms) nhóm nhiều phần tử theo một khóa chung, thực hiện việc nhóm theo từng window.

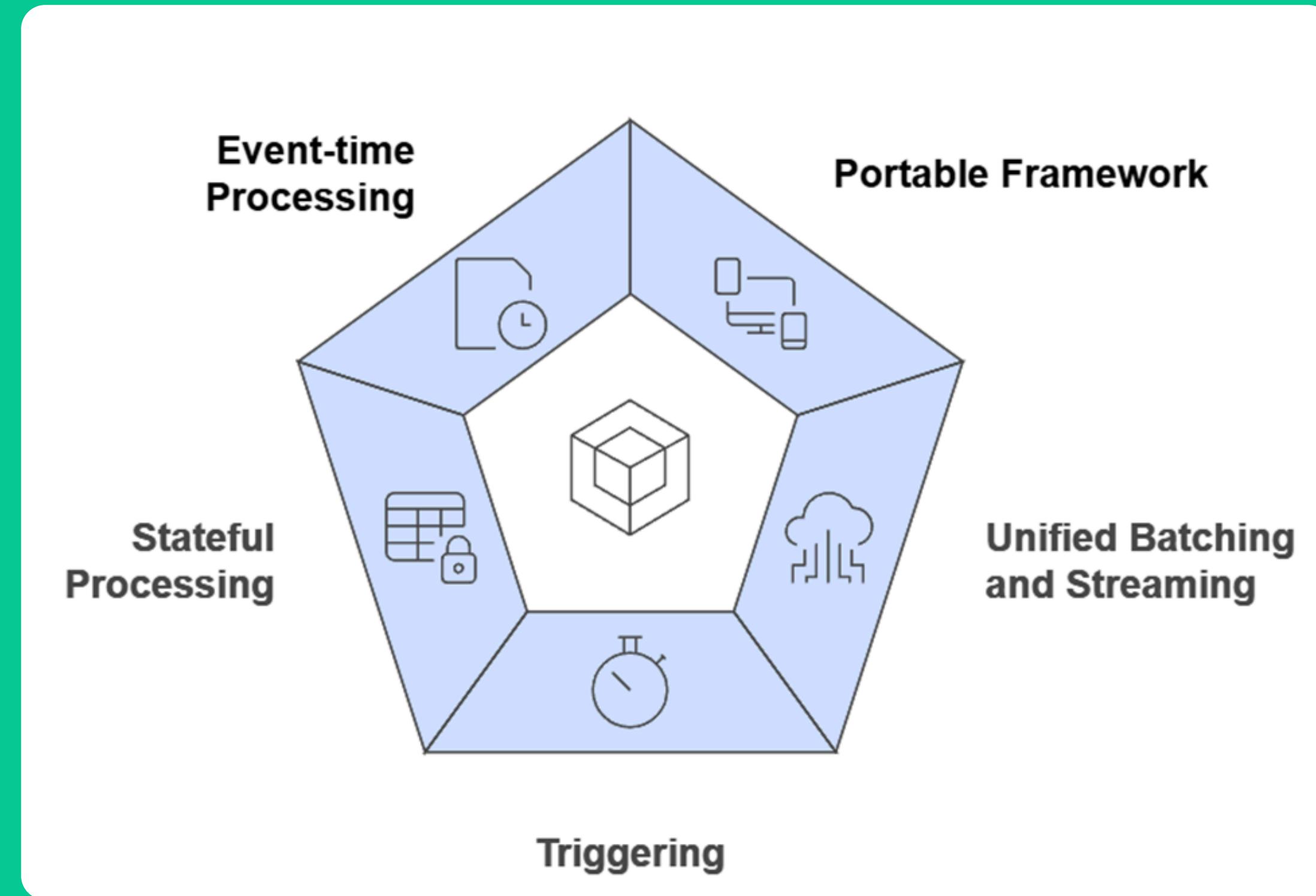
ĐẶC TRƯNG CỦA APACHE BEAM

WINDOWING



ĐẶC TRƯNG CỦA APACHE BEAM

CÁC ĐẶC TRƯNG KHÁC



ƯU - NHƯỢC ĐIỂM CỦA APACHE BEAM

Ưu điểm



- HỖ TRỢ ĐA DẠNG PLATFORM
- TÍNH THỐNG NHẤT
- XỬ LÝ DỮ LIỆU TÙNG PHẦN TỬ
- TÍNH MỞ RỘNG TỐT

ƯU - NHƯỢC ĐIỂM CỦA APACHE BEAM

Nhược điểm



- KHÓ SỬ DỤNG
- HIỆU SUẤT PHỤ THUỘC RUNNER
- KHÔNG MẠNH MẼ NHƯ SPARK/FLINK

3

ỨNG DỤNG THỰC TẾ

ỨNG DỤNG THỰC TẾ



"I want to **ingest data** into the Data Warehouse/Data Lake for analytics and DB for OLTP"



"I want to **aggregate, process, and enrich data** for machine learning and analytics"



"I want to aggregate events, join with other data and **make operational decisions quickly**"

Ingestion and Replication
(Batch and Streaming)

Unified ETL
(Batch and Streaming)

Continuous Intelligence
(Real-time Decision Automation and Insights)

CASE STUDY 1: SPOTIFY



Spotify đã sử dụng Dataflow (với Beam) để xử lý dữ liệu phát nhạc của người dùng theo thời gian thực, cung cấp các thống kê cho người dùng ngay lập tức.

CASE STUDY 2: AIRBUS

The Airbus logo is displayed in a large, bold, dark blue sans-serif font. It is centered within a white rounded rectangular box with a thin black border, which is set against a solid orange background.

AIRBUS

Airbus đã sử dụng Beam để xử lý và phân tích một lượng lớn dữ liệu cảm biến từ các hệ thống máy bay.

4

SO SÁNH VỚI APACHE SPARK

SO SÁNH VỚI APACHE SPARK

Tiêu chí	Apache Beam	Apache Spark
API thống nhất	Thống nhất cho cả batch và streaming	Batch và streaming có API riêng biệt
Platform	Hỗ trợ nhiều runner (Spark, Flink, Dataflow, etc.)	Chạy chủ yếu trên Spark
Cộng đồng phát triển	Mới hơn, nhưng phát triển nhanh	Đã phát triển lâu hơn và có cộng đồng lớn



SO SÁNH VỚI APACHE SPARK

Tiêu chí	Apache Beam	Apache Spark
Windowing & Triggering	Hỗ trợ linh hoạt hơn cho xử lý stream	Spark Streaming 2.x hạn chế hơn Beam
Khả năng mở rộng	Phụ thuộc vào runner	Tối ưu hóa tốt trên Spark
Trạng thái	Hỗ trợ tốt stateful processing	Hỗ trợ stateful nhưng ít linh hoạt hơn Beam
Tính di động	Di động, chạy được trên nhiều nền tảng	Tập trung vào Spark, không di động như Beam



5

DEMO

CÁCH CÀI ĐẶT

- Cài đặt môi trường
- Cài đặt Apache Beam

```
# Create a new Python virtual environment.  
python3 -m venv env
```

```
# Activate the virtual environment.  
source env/bin/activate
```

```
pip install apache-beam
```

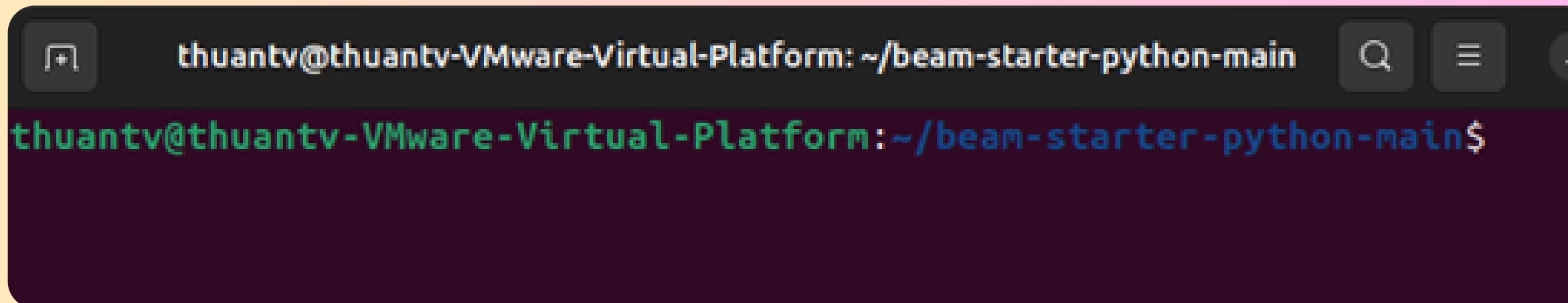
CÁCH CÀI ĐẶT

- Cú pháp thực thi 1 pipeline

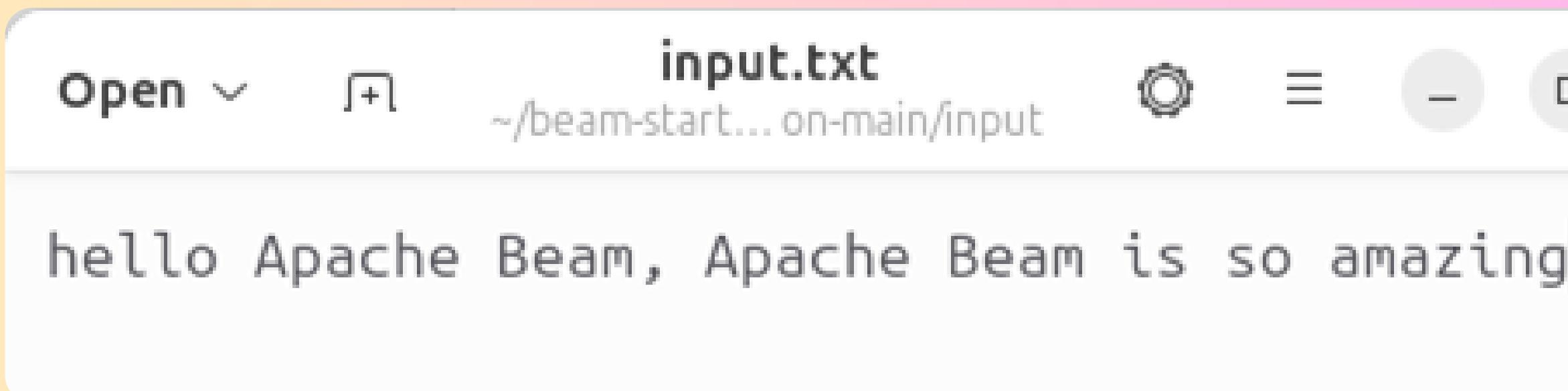
```
python -m apache_beam.examples.wordcount --input /path/to/inputfile \
--output /path/to/write/counts \
--runner SparkRunner
```

WORDCOUNT ĐƠN GIẢN VỚI APACHE BEAM

- Mở Terminal trở đến thư mục chứa file pipeline

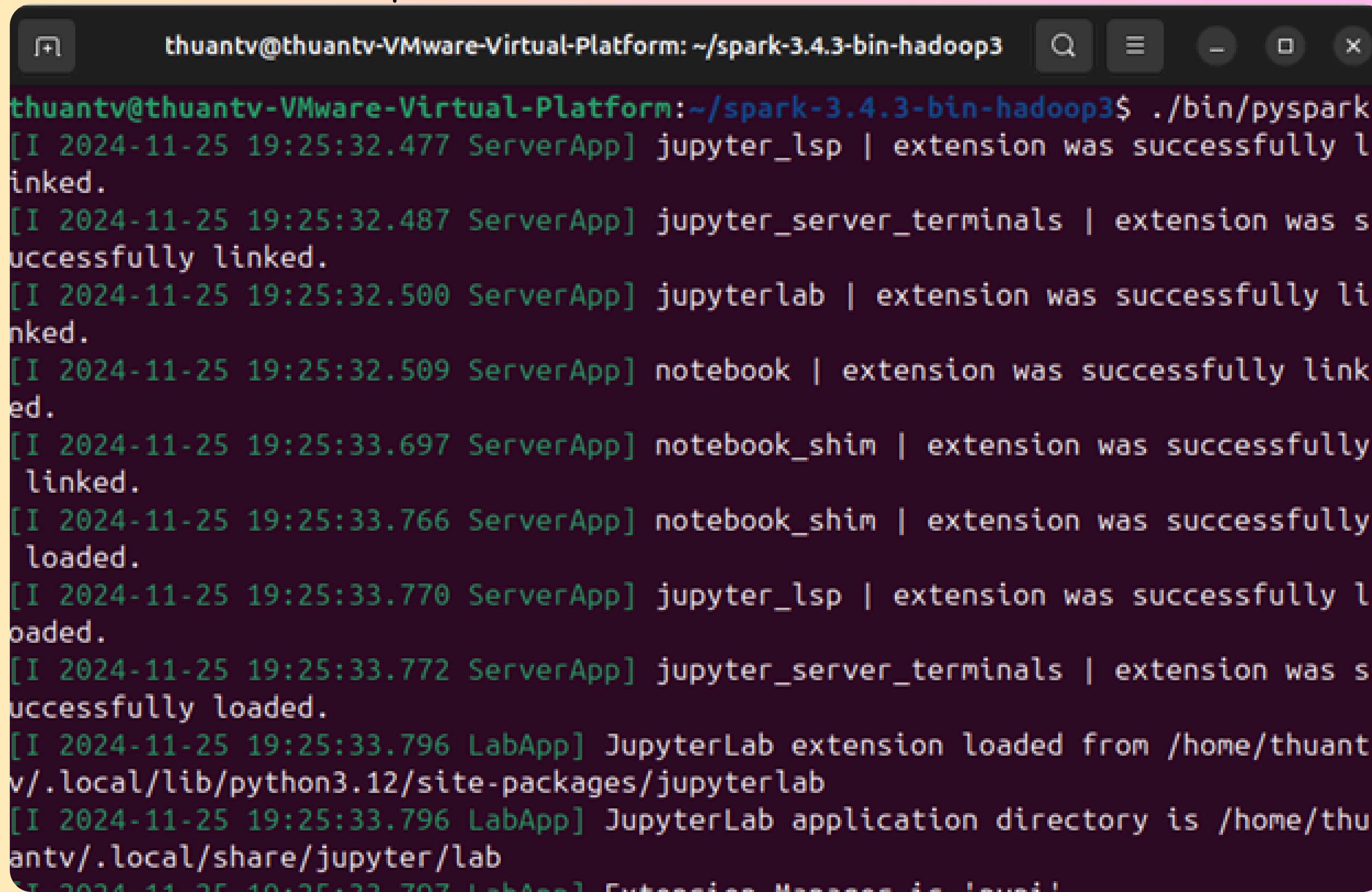


- File input.txt trong thư mục input để thực hiện đếm từ



WORDCOUNT ĐƠN GIẢN VỚI APACHE BEAM

- Mở Spark để triển khai trên Spark Runner



```
thuantv@thuantv-VMware-Virtual-Platform:~/spark-3.4.3-bin-hadoop3$ ./bin/pyspark
[I 2024-11-25 19:25:32.477 ServerApp] jupyter_lsp | extension was successfully linked.
[I 2024-11-25 19:25:32.487 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2024-11-25 19:25:32.500 ServerApp] jupyterlab | extension was successfully linked.
[I 2024-11-25 19:25:32.509 ServerApp] notebook | extension was successfully linked.
[I 2024-11-25 19:25:32.697 ServerApp] notebook_shim | extension was successfully linked.
[I 2024-11-25 19:25:33.766 ServerApp] notebook_shim | extension was successfully loaded.
[I 2024-11-25 19:25:33.770 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2024-11-25 19:25:33.772 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2024-11-25 19:25:33.796 LabApp] JupyterLab extension loaded from /home/thuantv/.local/lib/python3.12/site-packages/jupyterlab
[I 2024-11-25 19:25:33.796 LabApp] JupyterLab application directory is /home/thuantv/.local/share/jupyter/lab
[1 2024-11-25 19:25:33.797 LabApp] Extension Manager is loaded.
```

WORDCOUNT ĐƠN GIẢN VỚI APACHE BEAM

- Pipeline

```
# The pipeline will be run on exiting the with block.
with beam.Pipeline(options=pipeline_options) as p:

    # Read the text file[pattern] into a PCollection.
    lines = p | 'Read' >> ReadFromText(known_args.input)

    counts = (
        lines
        | 'Split' >> (beam.ParDo(WordExtractingDoFn()).with_output_types(str))
        | 'PairWithOne' >> beam.Map(lambda x: (x, 1))
        | 'GroupAndSum' >> beam.CombinePerKey(sum))

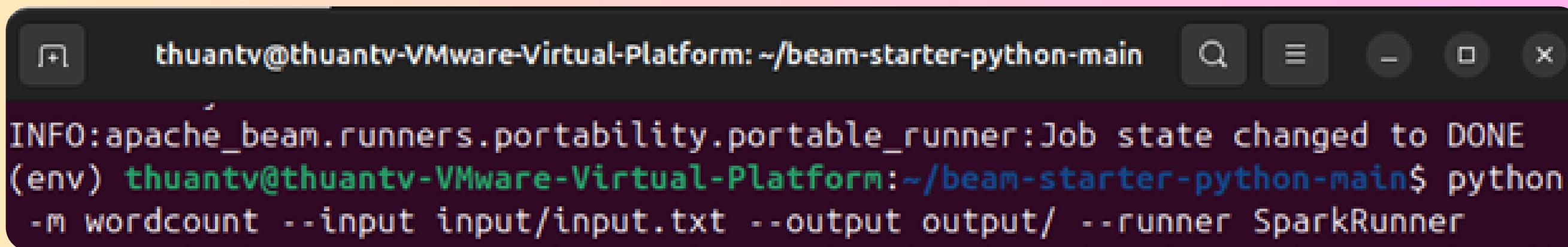
    # Format the counts into a PCollection of strings.
    def format_result(word, count):
        return '%s: %d' % (word, count)

    output = counts | 'Format' >> beam.MapTuple(format_result)

    # Write the output using a "Write" transform that has side effects.
    # pylint: disable=expression-not-assigned
    output | 'Write' >> WriteToText(known_args.output)
```

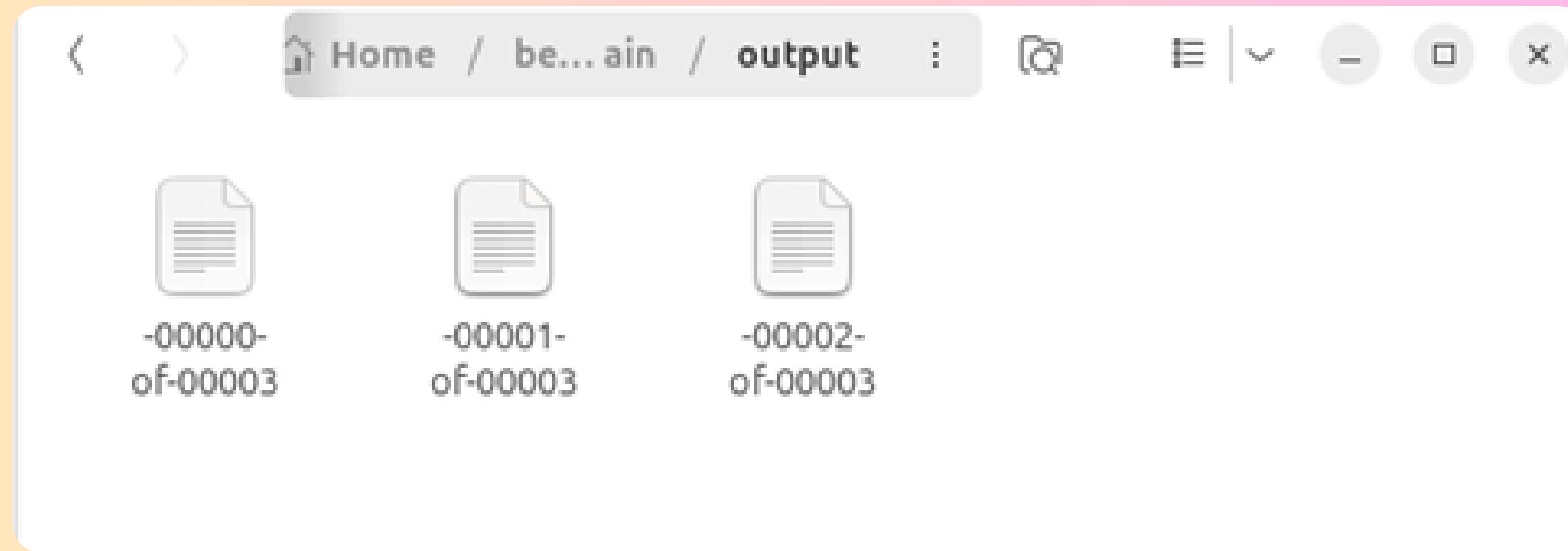
WORDCOUNT ĐƠN GIẢN VỚI APACHE BEAM

- Thực hiện pipeline trên Spark Runner



```
INFO:apache_beam.runners.portability.portable_runner:Job state changed to DONE  
(env) thuantv@thuantv-VMware-Virtual-Platform:~/beam-starter-python-main$ python  
-m wordcount --input input/input.txt --output output/ --runner SparkRunner
```

- Kết quả: tạo ra 3 file output



WORDCOUNT ĐƠN GIẢN VỚI APACHE BEAM

- File output 1

```
-00000-of-00003 × -00001-of-00003 | -00002-of-00003  
so: 1
```

- File output 2

```
-00000-of-00003 -00001-of-00003 × -00002-of-00003  
hello: 1  
Beam: 2
```

- File output 3

```
-00000-of-00003 -00001-of-00003 -00002-of-00003  
amazing: 1  
Apache: 2  
is: 1
```

WORDCOUNT ĐƠN GIẢN VỚI APACHE BEAM

- Thực hiện trên Direct Runner

```
(env) thuantv@thuantv-VMware-Virtual-Platform:~/beam-starter-python-main$ python  
-m wordcount --input input/input.txt --output output/
```

- Kết quả: Output chỉ có 1 file duy nhất



```
-00000-of-00001  
~/beam-starter...-main/output
```

```
hello: 1  
Apache: 2  
Beam: 2  
is: 1  
so: 1  
amazing: 1
```

WORDCOUNT VỚI DỮ LIỆU STREAMING

- Pipeline

```
with beam.Pipeline(options=options) as pipeline:  
    (  
        pipeline  
        | 'Read from socket' >> ReadFromSocket('localhost', 9990)  
        | 'Parse JSON and Add Timestamps' >> beam.ParDo(ParseAndAddTimestamp())  
        | 'Window' >> beam.WindowInto(  
            beam.window.FixedWindows(5),  
            trigger=beam.trigger.AfterWatermark(),  
            accumulation_mode=beam.trigger.AccumulationMode.DISCARDING  
        )  
        | 'Extract name' >> beam.Map(lambda plant: plant['name'])  
        | 'PairWithOne' >> beam.Map(lambda name: (name, 1))  
        | 'GroupAndSum' >> beam.CombinePerKey(sum)  
        | 'Add dummy key' >> beam.Map(lambda x: ('dummy', x))  
        | 'Group by key' >> beam.GroupByKey()  
        | 'Convert to dict' >> beam.Map(lambda kv: dict(kv[1]))  
        | 'Format result' >> beam.Map(  
            lambda d: ', '.join([f'{k}: {v}' for k, v in d.items()])  
        )  
        | 'Print results' >> beam.ParDo(PrintResult())  
    )
```

WORDCOUNT VỚI DỮ LIỆU STREAMING

- Socket gửi dữ liệu streaming mỗi 1s gồm tên và timestamp của event

```
(env) thuantv@thuantv-VMware-Virtual-Platform:~/socket$ python3 socket-server.py
Server listening on localhost:9990
Connection from ('127.0.0.1', 60686)
Sent: {"name": "Apple", "season": 1732543775.535184}
Sent: {"name": "Carrot", "season": 1732543776.535851}
Sent: {"name": "Carrot", "season": 1732543777.5369644}
Sent: {"name": "Apple", "season": 1732543778.5386634}
Sent: {"name": "Carrot", "season": 1732543779.5399363}
Sent: {"name": "Apple", "season": 1732543780.5412452}
Sent: {"name": "Apple", "season": 1732543781.5431101}
Sent: {"name": "Apple", "season": 1732543782.5444434}
Sent: {"name": "Banana", "season": 1732543783.5463877}
Sent: {"name": "Banana", "season": 1732543784.5472267}
Sent: {"name": "Carrot", "season": 1732543785.5481608}
Sent: {"name": "Carrot", "season": 1732543786.549086}
```

WORDCOUNT VỚI DỮ LIỆU STREAMING

- Với Fixed Window 60s, kết quả:

```
(env) thuantv@thuantv-VMware-Virtual-Platform:~/beam-starter-python-main$ python  
3 socket-pipeline.py  
Window 2024-11-25 14:09:00 - 2024-11-25 14:10:00: Apple: 5, Carrot: 5, Banana: 2
```

- Với Fixed Window 5s, kết quả:

```
(env) thuantv@thuantv-VMware-Virtual-Platform:~/beam-starter-python-main$ python  
3 socket-pipeline.py  
Window 2024-11-25 14:11:15 - 2024-11-25 14:11:20: Apple: 2, Carrot: 3  
Window 2024-11-25 14:11:20 - 2024-11-25 14:11:25: Apple: 3, Banana: 2  
Window 2024-11-25 14:11:25 - 2024-11-25 14:11:30: Carrot: 2
```

**THANK YOU
FOR
WATCHING!**

