Collibra

Give me my data... now!

Beam use-cases in review

Alex Van Boxel

Alex Van Boxel
Principal Engineer
Collibra

Apache Beam
Committer

Google Developer Expert

Disclaimer: This is my personal Beam journey.

This crosses different companies

What’s the goal of this talk?

Give you my personal view and use-cases I bumped upon throughout my career working with Apache Beam

What will you not learn in this talk?

You will not learn the **hottest** new beam features. That’s where the Beam College come in right?!

What!!! You do sessions in 2 lines of code? We spend months developing sessionization pipelines.

famous quote by “someone that didn’t do Beam” I met on a meetup

Dataflow love

Beam on Cloud DataFlow

Amazing **Uptimes**

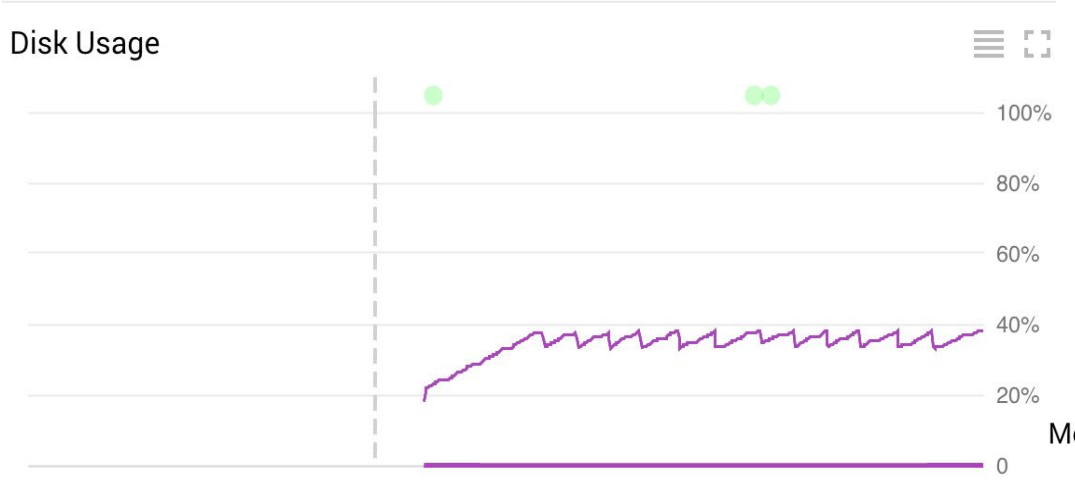
ingresscalrawpipeline-alexvanboxel-1216151020	Streaming	—		490 days 1 hr	Mar 1, 2017, 9:17:35 AM
ingresspaymentpipeline-alexvanboxel-1216152350	Streaming	—		503 days 0 hr	Feb 16, 2017, 10:12:31 AM

Magic **Updates**

ingressemail	Streaming	—	1 min 9 sec	Jul 4, 2018, 11:47:48 AM	Not started
ingressemail	Streaming	—	129 days 19 hr	Feb 24, 2018, 2:53:18 PM	Running

Technology - Apache Beam Windows

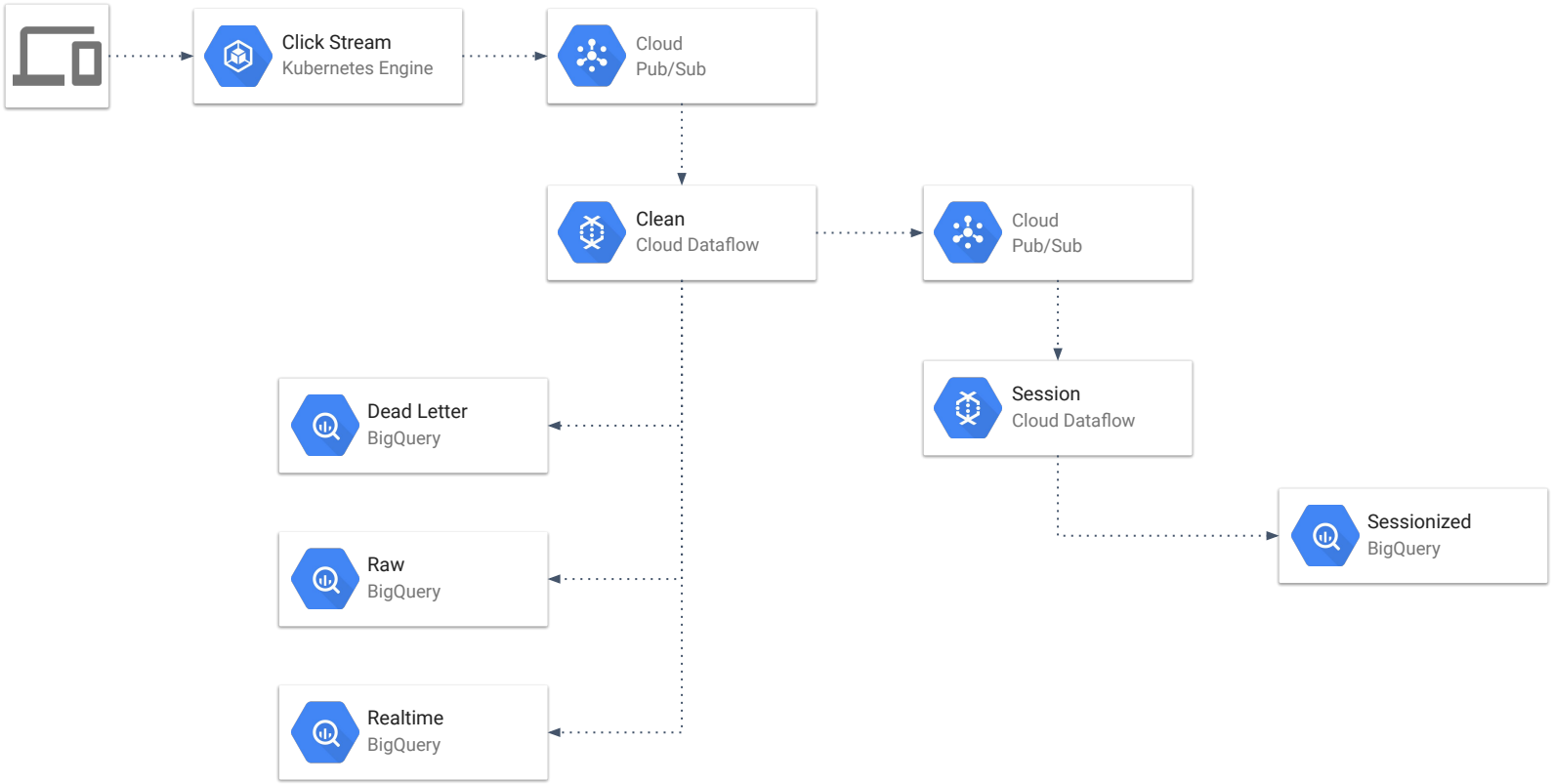
and the famous large window experiment



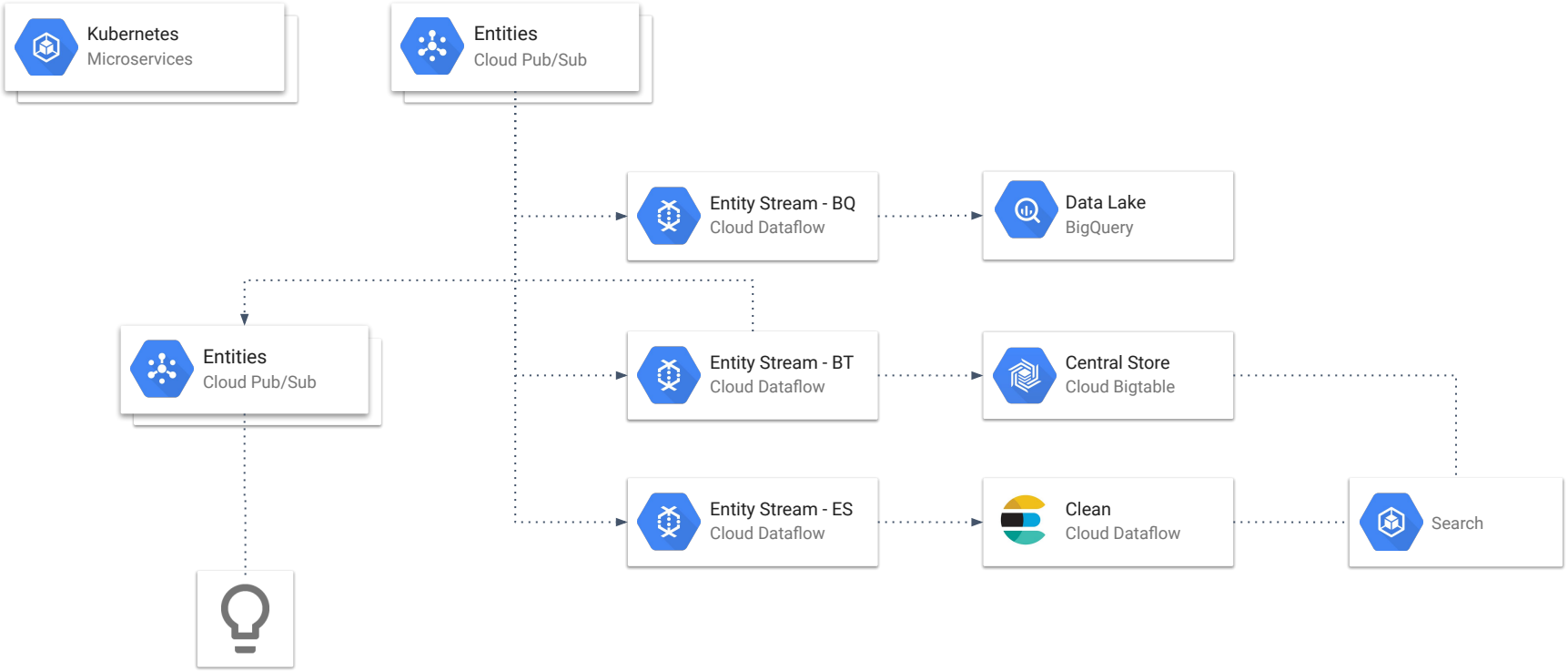
Source: [Medium: Cloud Dataflow and large beam windows]

Trip down memory lane

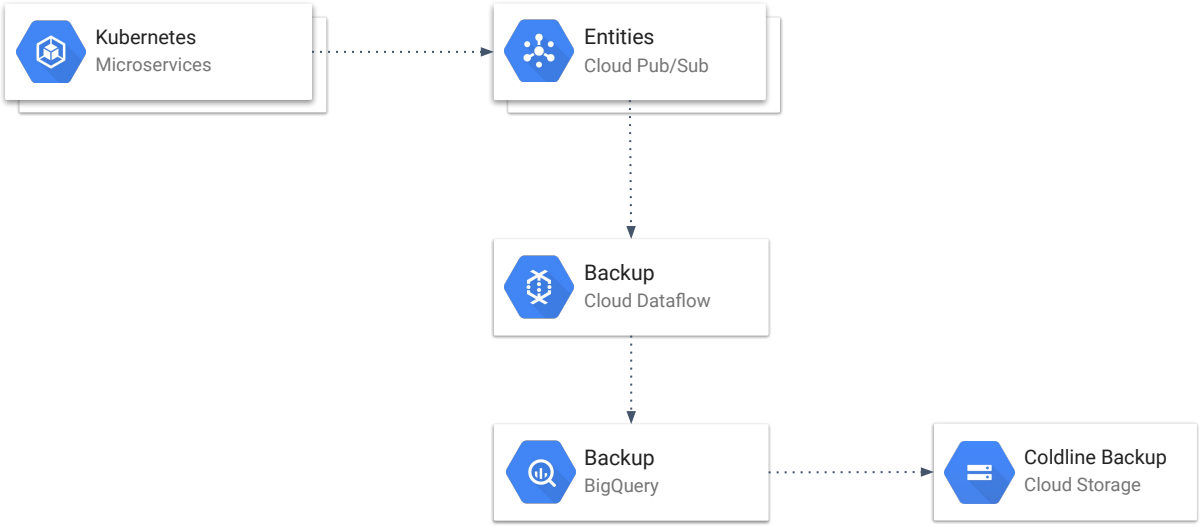
Streaming - Activity



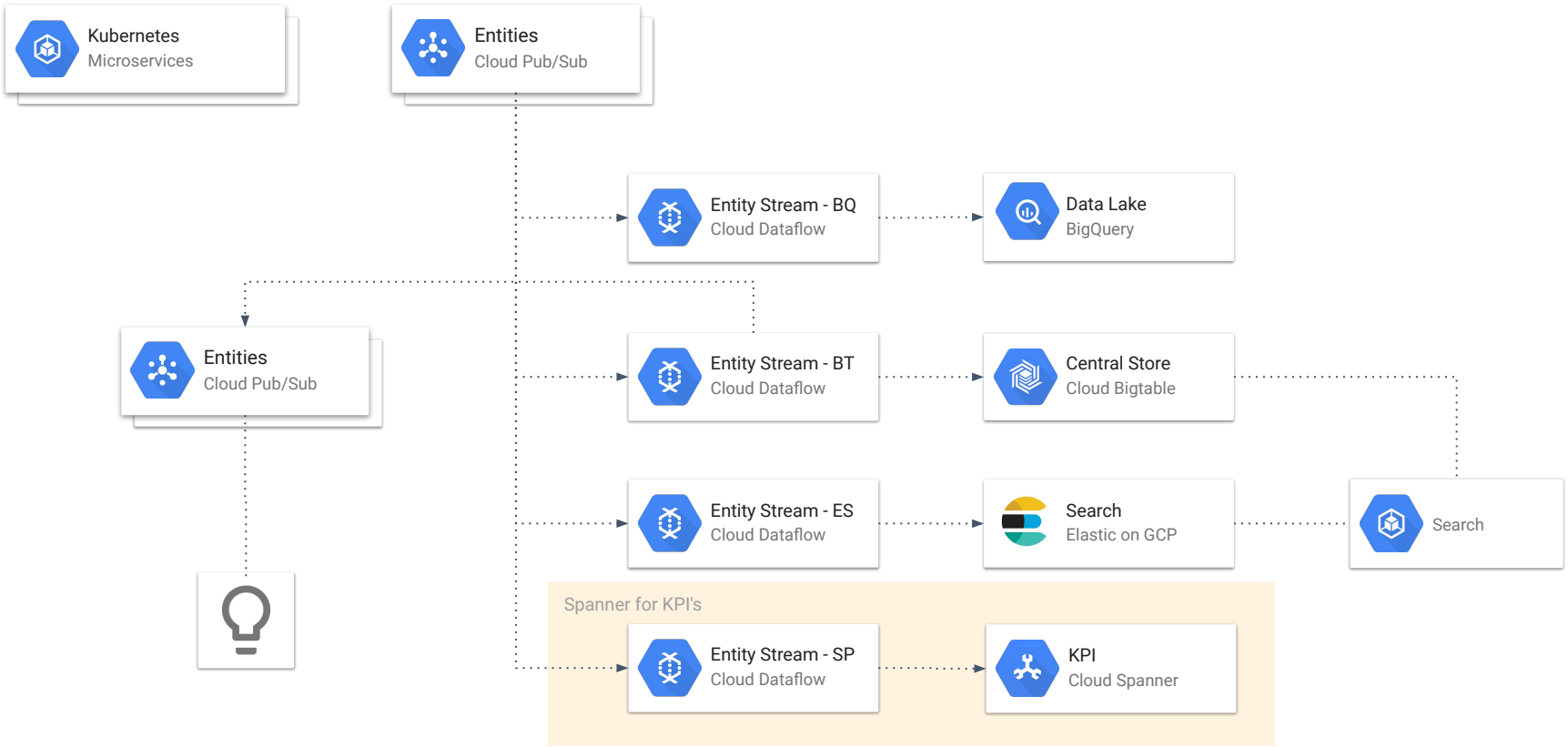
Streaming - Entities



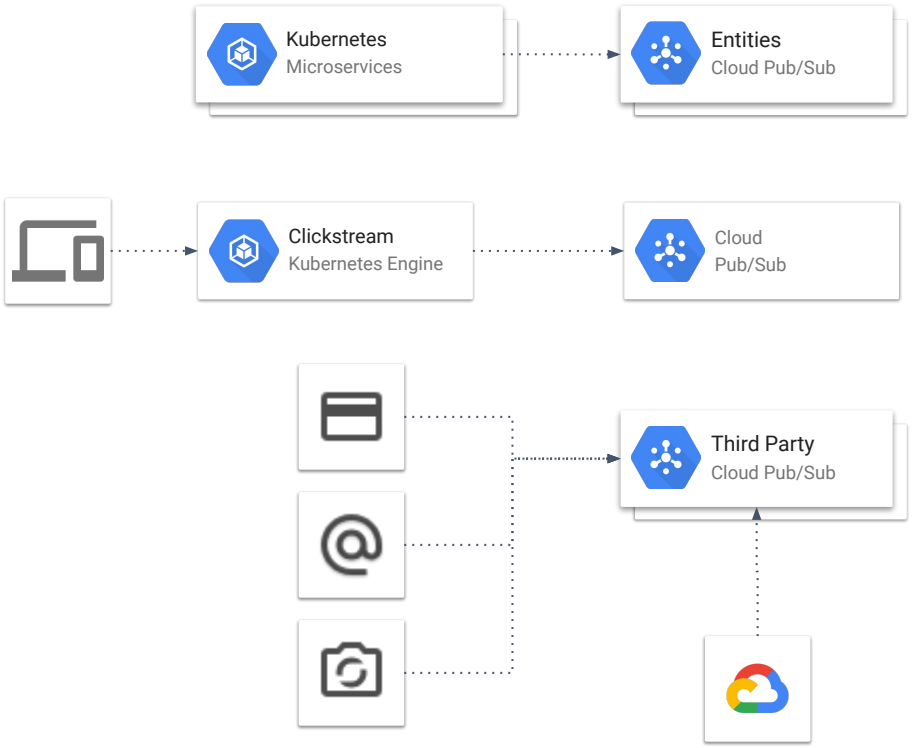
Streaming - Backup



Streaming - KPI refocus > Business aggregates

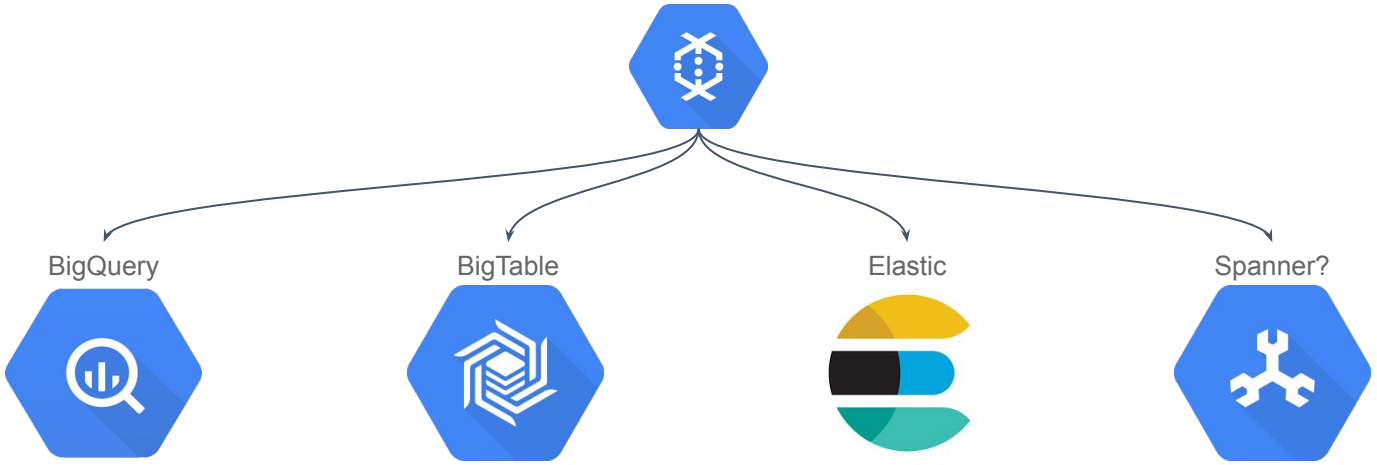


Streaming - Data Backbone



Consuming the stream

Different database technologies for different purposes



Reporting data
-- real-time!
-- full change history
-- self-service BI
-- excel report generation
-- permission based
-- GDPR!
-- SaaS
-- Standard SQL language

-- High-availability
-- low-latency
-- real-time
-- PaaS
-- backup + history

-- search

-- low-latency
-- real-time
-- distributed
-- Standard SQL
-- complex KPI's

Metadata driven pipelines

Changing the mindset

from **coding** pipelines to adding **dynamic transforms**

Pipeline Architecture

DoFn: change type X -> X’

DoFn: change type X -> X’

DoFn: change type Y -> Y’

DoFn: change type Z -> Z’

DoFn: change type X -> X’

DoFn: change type Y -> Y’

DoFn: change type Z -> Z’

DoFn: change type A -> A’

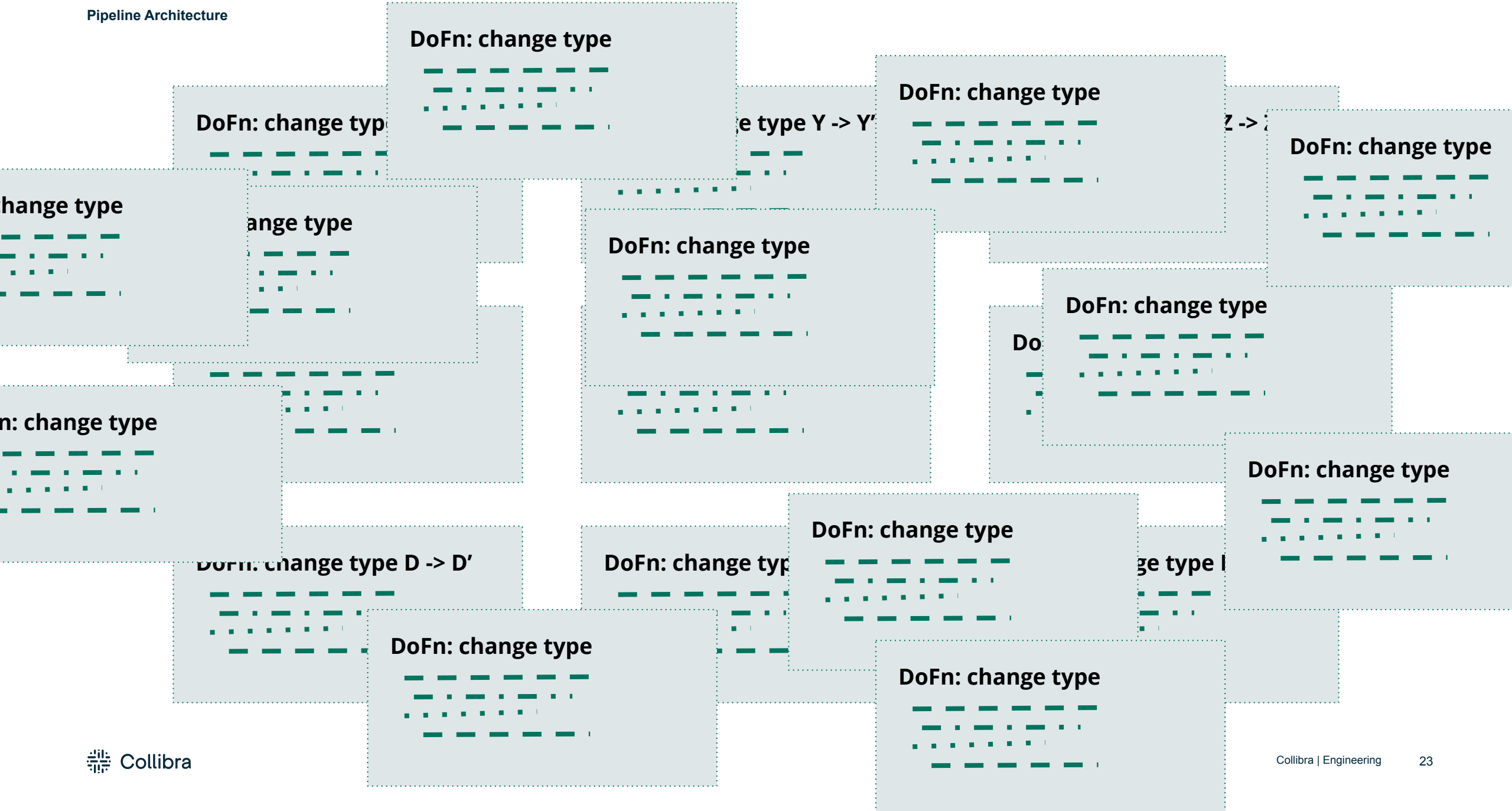
DoFn: change type B -> B’

DoFn: change type C -> C’

DoFn: change type D -> D’

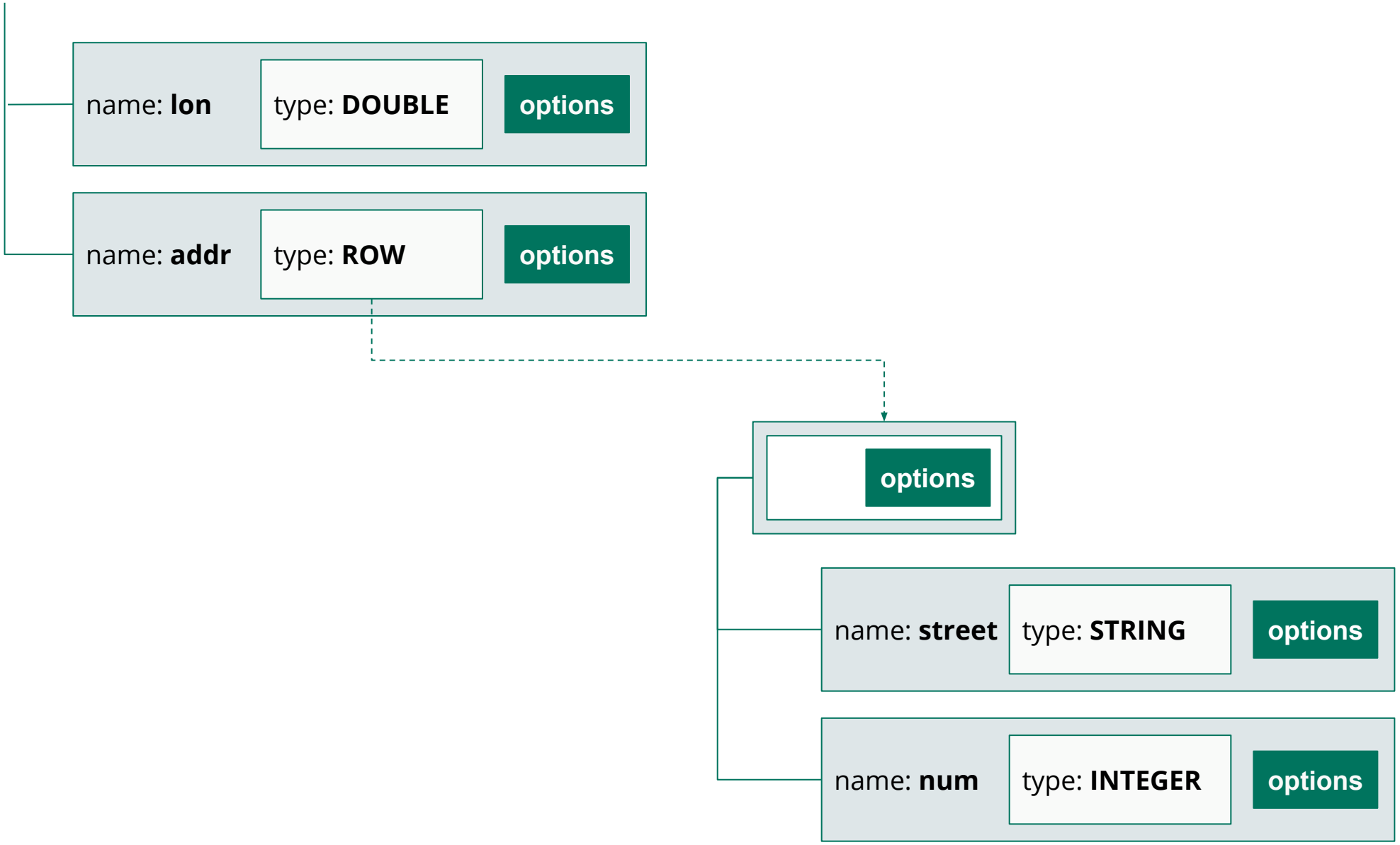
DoFn: change type E -> E’

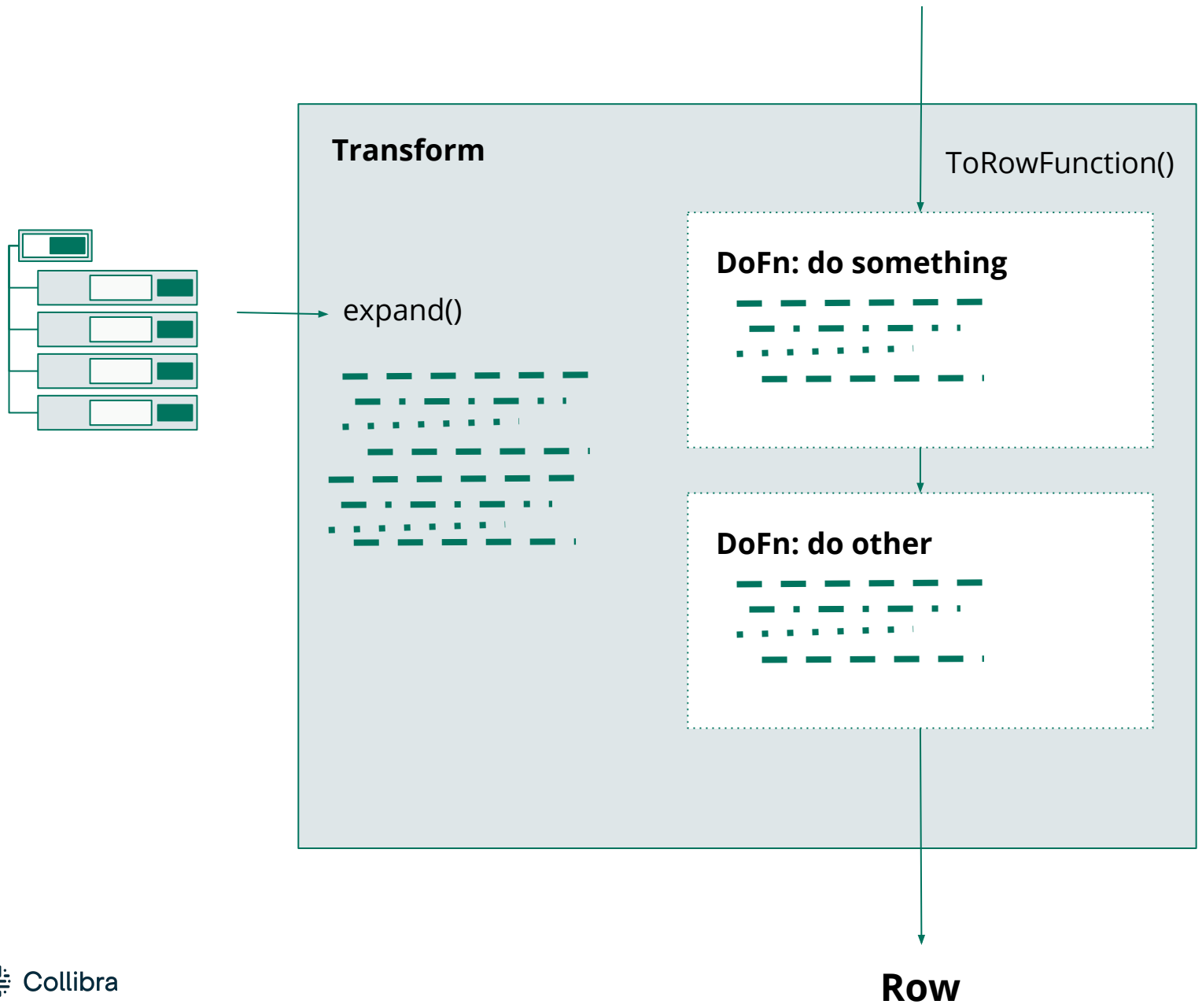
DoFn: change type F -> F’

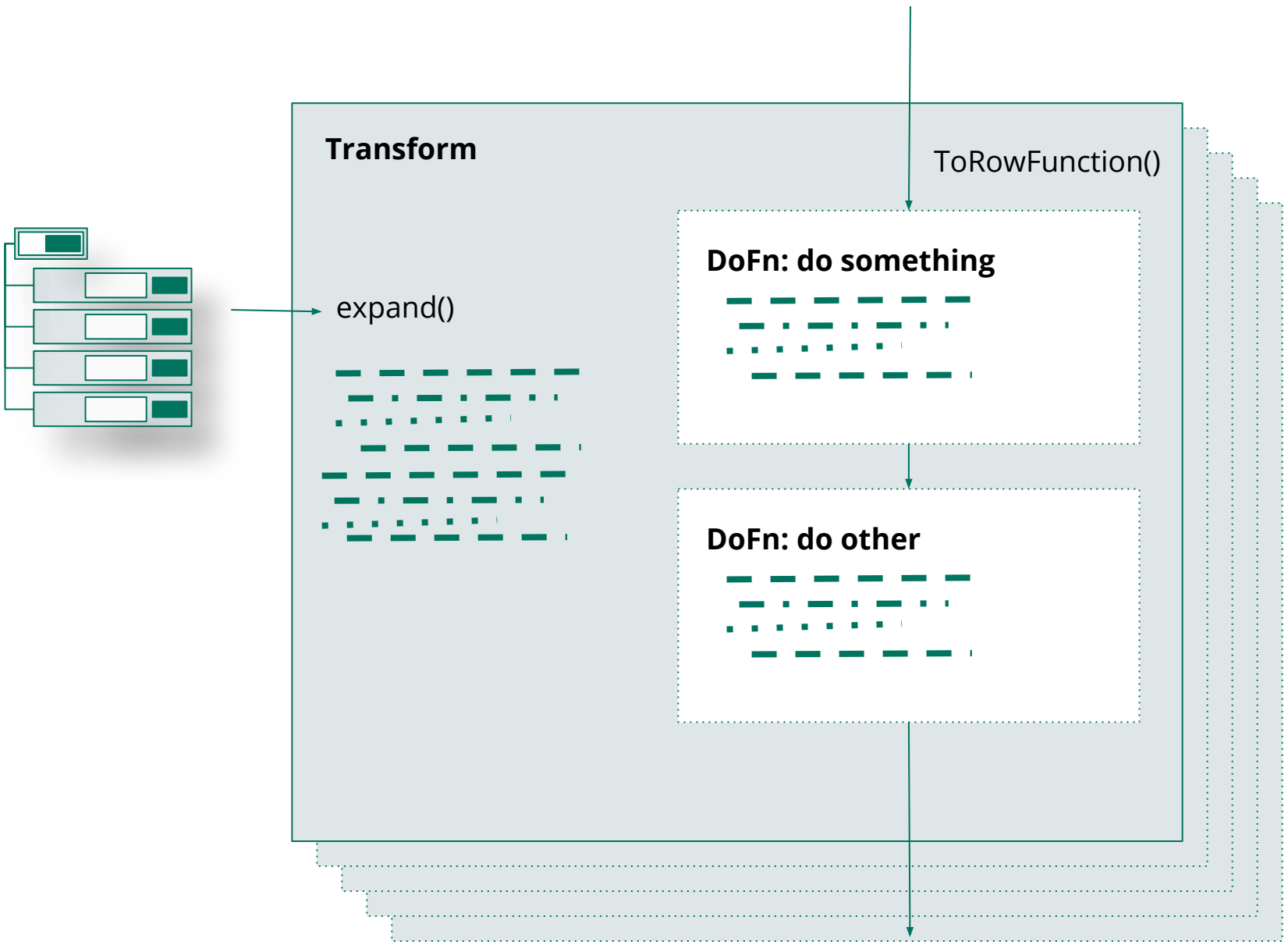

DoFn: change type
DoFn: change typ
e type Y -> Y'
DoFn: change type
Z -> Z
DoFn: change type
hange type
ange type
DoFn: change type
DoFn: change type
Do
DoFn: change type
n: change type
DoFn: change type
DoFn: change type D -> D'
DoFn: change typ
DoFn: change type
ge type I
DoFn: change type
DoFn: change type

Replacing the DoFn chaos

from **coding** pipelines to adding **dynamic transforms**


name: lon
type: DOUBLE
options
name: addr
type: ROW
options
options
name: street
type: STRING
options
name: num
type: INTEGER
options


Transform
ToRowFunction()
expand()
DoFn: do something
DoFn: do other
Row


Transform
ToRowFunction()
expand()
DoFn: do something
DoFn: do other

Row

Schema Options Use-Case

Data Quality

Verify elements on expected values

- Is the value within a range?
- Complies to a regex?

Checking on data quality

https://github.com/envoyproxy/protoc-gen-validate/blob/master/validate/validate.proto

```
...
// FloatRules describes the constraints applied to `float` values
message FloatRules {
    // Const specifies that this field must be exactly the specified value
    optional float const = 1;

    // Lt specifies that this field must be less than the specified value,
    // exclusive
    optional float lt = 2;

    // Lte specifies that this field must be less than or equal to the
    // specified value, inclusive
    optional float lte = 3;
...
```

Semantic Meaning Beyond Logical types

Collibra

When primitives are not enough

- Encoding inside string, binary, numbers
- Extra information about the type

Debezium, example

```
Schema.Builder out = Schema.builder();
for (Schema.Field inField : in.getFields()) {
  String fieldName = inField.getName();
  Schema.Field outField = null;
  Row semanticType = inField.getOptions().getValue("io.debezium.v1.semantic_type", null);

  switch (inField.getType().getTypeName()) {
    // ...
    case STRING:
      if (semanticType.getValue(0).equals("io.debezium.time.ZonedTimestamp")) {
        outField = Schema.Field.of(fieldName, Schema.FieldType.DATETIME).withNullable(true);
      } else {
        outField = Schema.Field.of(fieldName, inField.getType());
      }
      break;
```

Debezium, example

```
Schema.Builder out = Schema.builder();
for (Schema.Field inField : in.getFields()) {
  String fieldName = inField.getName();
  Schema.Field outField = null;
  Row semanticType = inField.getOptions().getValue("io.debezium.v1.semantic_type", null);

  switch (inField.getType().getTypeName()) {
    // ...
    case INT64:
      if (semanticType.getValue(0).equals("io.debezium.time.Timestamp")) {
        outField = Schema.Field.of(fieldName, Schema.FieldType.DATETIME).withNullable(true);
      } else {
        outField = Schema.Field.of(fieldName, inField.getType());
      }
      break;
```

Debezium, example

```
Schema.Builder out = Schema.builder();
for (Schema.Field inField : in.getFields()) {
  String fieldName = inField.getName();
  Schema.Field outField = null;
  Row semanticType = inField.getOptions().getValue("io.debezium.v1.semantic_type", null);

  switch (inField.getType().getTypeName()) {
    // ...
    case BYTES:
      if (semanticType.getValue(0).equals("org.apache.kafka.connect.data.Decimal")) {
        outField = Schema.Field.of(fieldName, Schema.FieldType.DECIMAL).withNullable(true);
      } else {
        outField = Schema.Field.of(fieldName, inField.getType());
      }
      break;
```

GDPR Use-Case

GDPR Use-Case

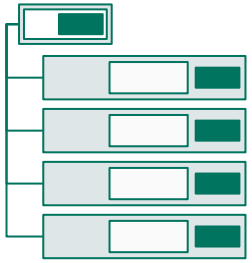
Requirements

- Encrypt PII information
- Each user has his own encryption key
- Right to forget
- Keep non-PII information (legal requirements)

GDPR Use-Case

Building Blocks

- Schema Options
- AEAD encryption (google/tink) in Beam
- Beam Stateful processing
- Key in BigTable (atomicity!)
- Federated table to BigQuery



expand()

Transform

ToRowFunction()

KV<uid, Row>

- **Do we have PII options?**
- **YES**
- **Create output schema (encrypted = bin)**
- **Link DoFn**

- **Encrypt key for uid**
- **Keygen and store in BigTable**
- **Traverse row tree and encrypt field (w/option)**
- **Output Row**

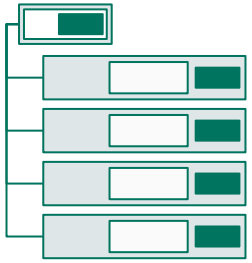
Row

rest of pipeline

BigTable

BigQuery Federation

BigQuery



Transform

expand()

- **Do we have PII options?**
- **NO**
- **Link DoFn**

- **Output Row**

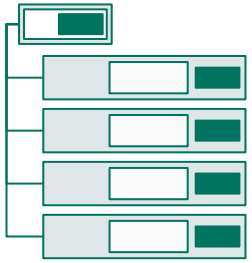
Row

rest of pipeline

BigQuery

Go over each field, and encrypt the field (naive)

```
Row.Builder builder =  Row.withSchema(row.getSchema());
row.getSchema().getFields().forEach(
  field -> {
    switch (field.getType().getTypeName()) {
      case STRING:
        String userField = field.getOptions().getValue("encrypt.ppi")
        String userId = row.getValue(userId);
        if(userId != null) {
          builder.addValue(
            encryptForUser(row.getValue(field), userId);
        }
        break;
      default:
        builder.addValue(row.getValue(field);
   }
```



expand()

Transform

ToRowFunction()

KV<uid, Row>

- **Do we have PII options?**
- **YES**
- **Create output schema (encrypted = bin)**
- **Create Transformation Code (byteBuddy)**
- **Link DoFn**

- **Encrypt key for uid**
- **Keygen and store in BigTable**
- **Execute generated code**
- **Output Row**

BigTable

BigQuery Federation

BigQuery

Row

rest of pipeline

Thank you

Questions?