Experiences with Apache Beam

Dan Debrunner

Programming Model Architect – IBM Streams
STSM, IBM

Apache Beam: An advanced unified programming model

Implement batch and streaming data processing jobs that run on any execution engine.

Background

- To define my point of view …

IBM Streams brief history

- 2002 – IBM Research/DoD joint research project – *System S*
- 2002-2009 – Multiple releases to development partners
- 2008 – IBM Software Group adopts project for product release
- 2009 – First release of *IBM Streams* (née *IBM InfoSphere Streams*)
- 2009-… – Multiple releases of *IBM Streams*
- 2015 – *Streaming Analytics* managed service on IBM Cloud
- 2017 – Inclusion in IBM Watson Data Platform

IBM Streams: High volume, low latency, continuous streaming analytics

- React to each event as it occurs
  - Customer: "*If you have to write it to disk you've already lost*"
- Maintain current state of thousands to millions of entities
  - *Context of Now!*
- Analytics run 24/7

IBM Streams programming models

- SPL (**S**treams **P**rocessing **L**anguage) –
  - Domain specific language
    - Operators, streams, windows
    - Data flow graph with cycles allowed
    - Toolkits with analytical & adapters operators
  - Structured tuples – similar to database table definition
    - `stream<rstring id, timestamp ts, float64 value>`
- Java/Scala/Python
  - Typical source, map, filter, flat map, for each, aggregate functional api
  - Integration with SPL
- Streams Designer – High-level visual pipeline creator
- Microservice approach
  - Topic based publish/subscribe model for streams

Building an Apache Beam Java runner for IBM Streams

- 1.0 supporting Apache Beam 2.0 Java SDK released early November 2017

Why?

- **Potential** single “standard” programming model for streaming applications

Some concerns

- Beam may not become the/a standard streaming api
- Real-world adoption of Beam not apparent
- Is the model too focused on event-time?
- Can it address scenarios our customers need

Potential upside

- Somewhat early in lifecyle, can we (IBM) & others help drive Beam to be *the* standard api

Terminology “confusion”

- “ParDo” – ParallelDo – but seemed little discussion about what parallel meant
- Sliding windows – Not the same as our definition – seems strange that fixed/sliding distinction while fixed is a sub-class of sliding.
- Unaligned windows <-> partitioned windows
- Watermarks – “Magic”
  - microsecond or millisecond
- Partition – Split
- Bounded/unbounded – batch/streaming

The good …

- SDK package well documented
- One page concept tutorials good
- The runners.core package significantly simplifies the runner implementation, which helped us to quickly get started.
- Large number of core tests that could be run to verify our runner
- The number of IO connectors keep growing

Some “bad” …

- Documentation in the runners package could be improved
- Not all concepts have one page tutorials
  - Initially many specific to pre-Beam Google Data Flow.
- Real-world sample applications would help

Some “bad” …

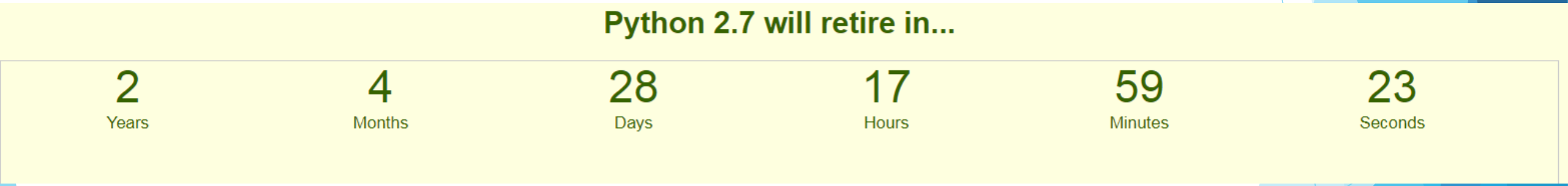
- Redundancy between View/Combine transformations
- Lack of tests for IO connectors slowed development
- Backwards compatibility
  - Unexpected classes not found after moving from Beam 2.0 to 2.1
  - Many features marked experimental
  - What happened to 1.x?
- No mechanism to capture pipeline source locations
- Footprint - ~60MB of dependencies
- Probably a shame didn’t start with Java 8

Just different?

- Streams runner can just produce a Streams Application Bundle (`sab` file)
  - Self contained application
  - Configured through submission parameters and “application configurations”
  - What does it mean to read metrics after creating a `sab`?

Python

- Streaming not yet supported
- Python 2.7!

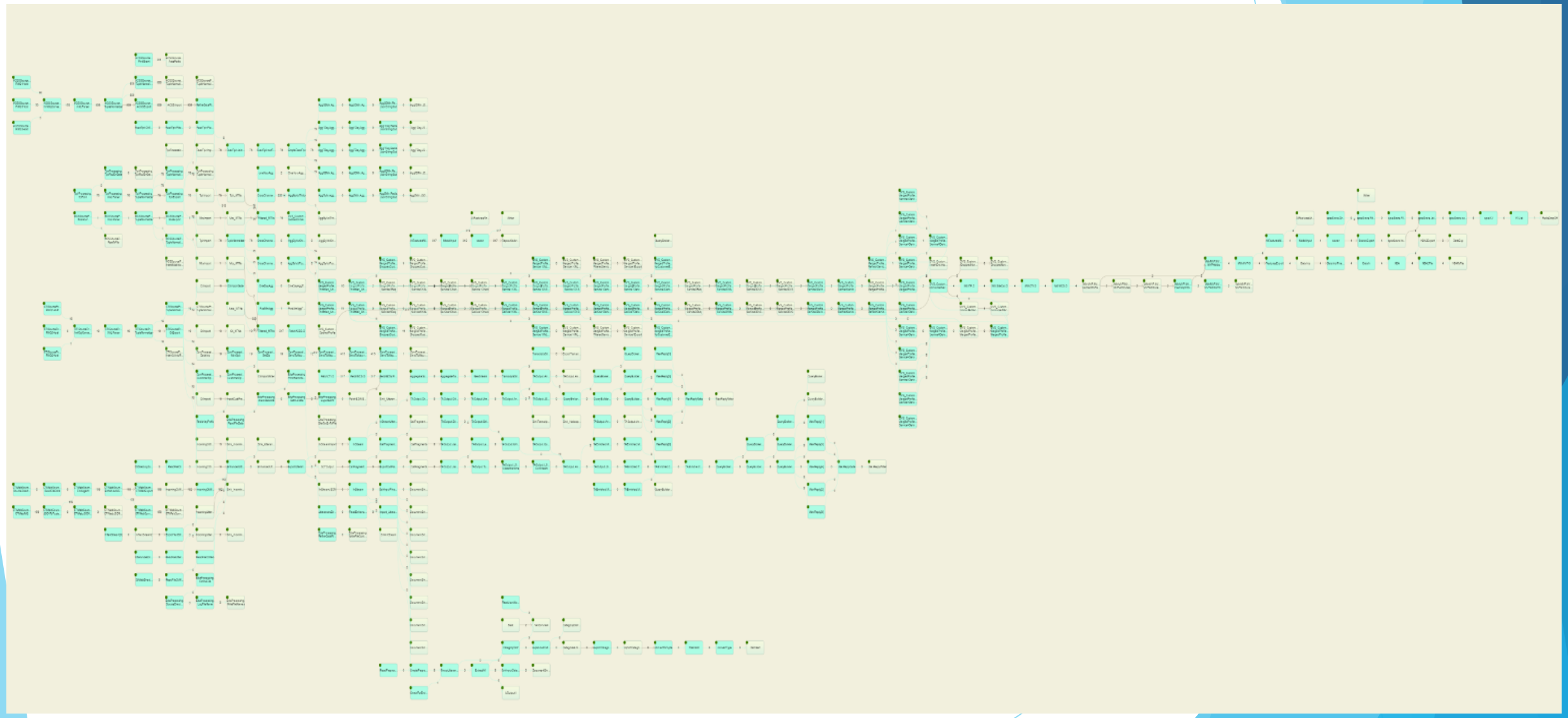


Developing pipelines

Naïve view …

- ***What*** results are being calculated?
  - Multiple outputs based upon multiple input data streams
  - Real-time state per entity, with potentially multiple entities per event
- ***Where*** in event time?
  - As soon as the event is received ..
- ***When*** in processing time?
  - As soon as possible …
- ***How*** do refinements of results relate?
  - Probably too late by then …

IBM Streams customer application



Pipeline configuration

- Pipeline configuration/tuning will be needed:
  - Only this host has access to the data source
  - Only these hosts have a $$ licensed library installed.
- Degree of parallelism
  - May be better known by application developer
- How to securely access credentials?
  - Streams has application configurations which hold credentials etc.
    - Can be set by system admins.
  - How to generically expose them in the model

Reusable analytics

- Is a model needed to allow reusable analytics against streaming data
- SPL has concept of toolkits
  - Collection of operators, functions and types.
  - Many toolkits open source at github.
- Aided by having a structured schema
  - Many operators support any schema though parameters
  - Most operators copy matching attributes from input to output
  - E.g. geospatial operator only needs say lat,long, time, id – but any additional attributes are carried from input to output automatically.

Monitoring API

- Is a standard monitoring API needed for complete application portability?

Impressions

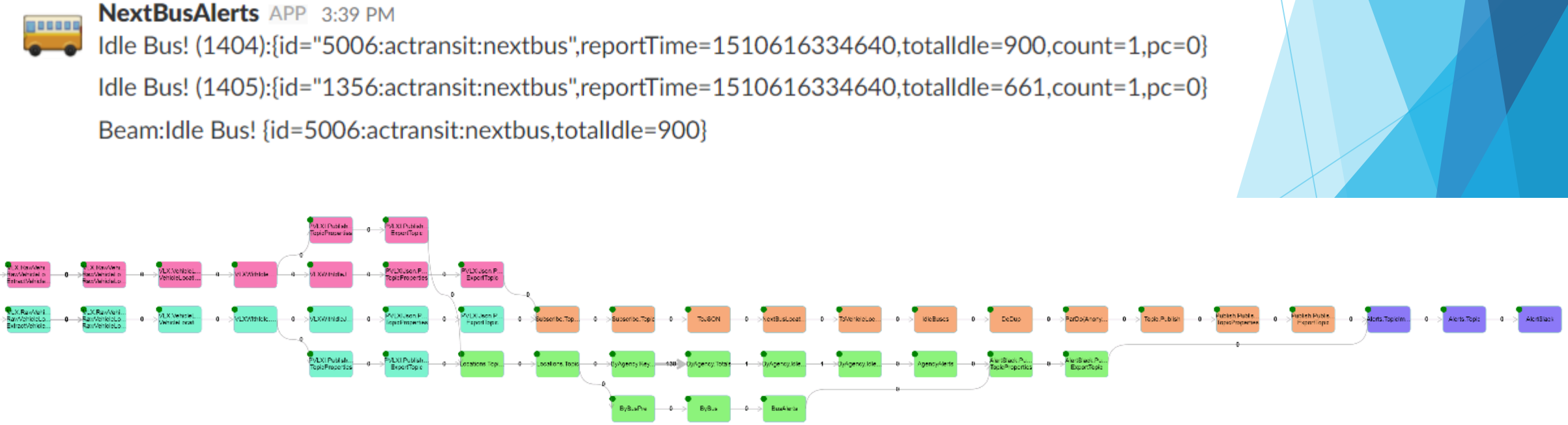
- *I found the model mostly quite simple to understand, but its realization in the APIs made writing my first Beam app much less simple. That is, I thought I understood how I was going to write my simple but actually coding it up was more difficult than I expected. The reference documentation is not bad; quite good in places, a bit weak in others, but the API is **big** and there is a gap between the programming model overview / quickstart tutorials and the reference docs. Any time I wanted to do something not covered in the quickstart, I'd end up spending quite a bit of time looking around the API reference to find things that looked like they were what I was after, and then how to use them.*

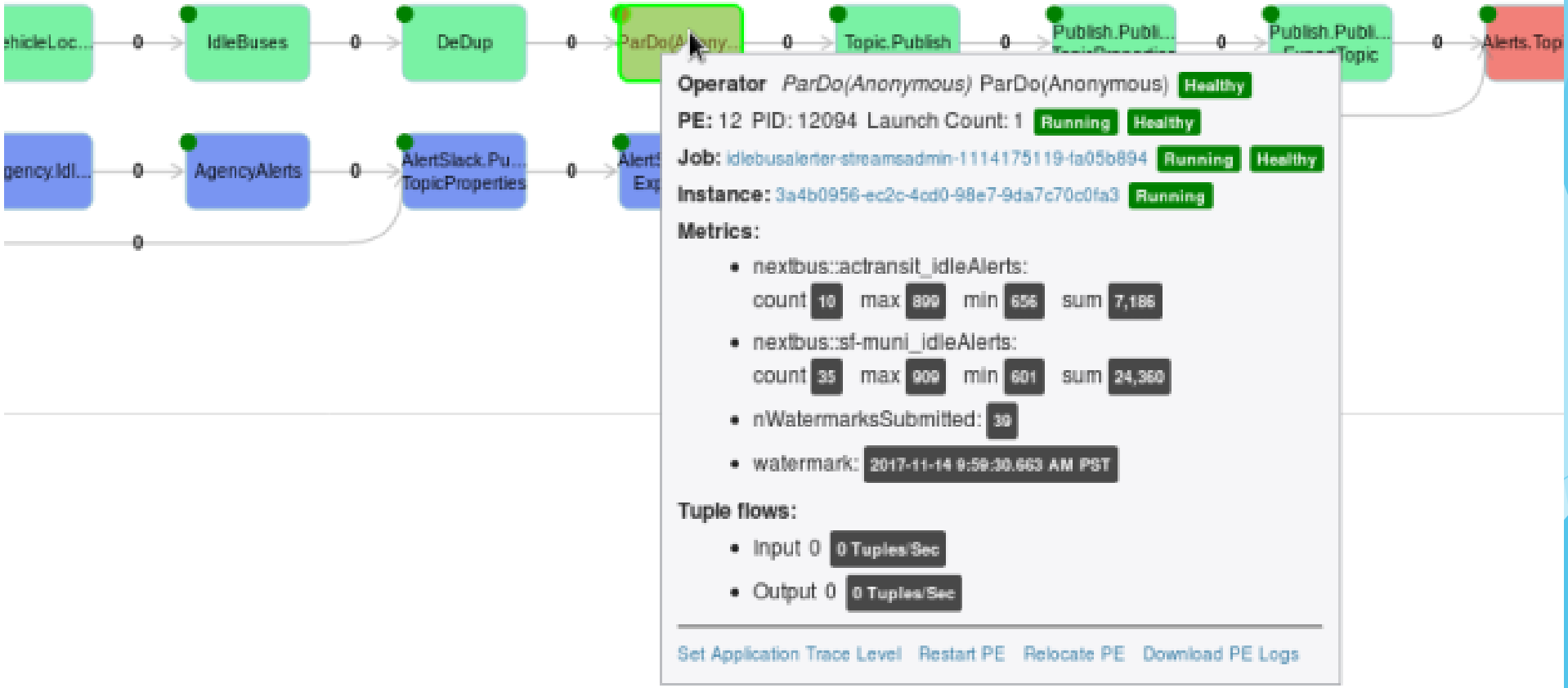
Impressions

- Builder approach makes sense
  - Have to dig out the available transformations
  - Generics + builders seems to lead to many levels of <> and ()
    - maybe confusing Eclipse along the way
  - Seemed to be able to use lambda expressions less than I wanted to
- @ProcessElement -> no auto-complete
- Sometimes seemed to have to set a coder when ideally it would be determined automatically
- Tuple ordering or lack of …

Vehicle Location Pipelines

- Existing streams of NextBus vehicle location data enriched with idle stats
- Create pipelines that continually monitor vehicles and agencies for idle alerts




IdleBuses
DeDup
Topic.Publish
AgencyAlerts
Operator ParDo(Anonymous) ParDo(Anonymous) Healthy
PE: 12 PID: 12094 Launch Count: 1 Running Healthy
Job: idlebusalerter-streamsadmin-1114175119-fa05b894 Running Healthy
Instance: 3a4b0956-ec2c-4cd0-98e7-9da7c70c0fa3 Running
Metrics:
nextbus::actransit_idleAlerts:
count 10 max 899 min 656 sum 7,186
nextbus::sf-muni_idleAlerts:
count 35 max 909 min 601 sum 24,360
nWatermarksSubmitted: 39
watermark: 2017-11-14 9:59:30.663 AM PST
Tuple flows:
Input 0 0 Tuples/Sec
Output 0 0 Tuples/Sec
Set Application Trace Level
Restart PE
Relocate PE
Download PE Logs

Some Issues

- Uncorrelated streams of locations from unknown number of different agencies
  - How to determine watermark?
- How to maintain state per-bus, per-agency, per-route
  - Window/watermark woes
  - Window(Last15Mins(last locations)) -> Window(Aggregate(ByAgency))
- Use of a timer implied a stateful ParDo then required a KV coder but not where …
  - Timer/state marked experimental

If I did it again …

- First try to better understand windowing/grouping concepts
  - using direct runner
  - using small fixed datasets

Experience Summary

- Apache Beam provides the foundation for a single model for streaming systems
  - Transforms and builders make sense
  - Documentation could be improved
- Still unclear on suitability for our customer needs
  - State handling
  - Non-event time apps
  - Tuple order
  - Configuration
- Really up to streaming framework providers to get involved in Beam community

Thank You

IBM

The **Watson** & **Cloud** Platform

IBM