

A new local runner appears!

A Deep Drive on the Prism runner

Robert Burke
lostluck@apache.org



BEAM
SUMMIT

September 4-5, 2024
Sunnyvale, CA. USA

What is Prism?

A Local Portable Runner

- Apache Beam Runner
 - Portable Only
- Local
 - Data in memory
 - Runs on the same machine
 - Not Distributed
- Built with Go
 - Compact Standalone binaries
 - Powerful Concurrency
 -
- Originally for the Go SDK
 - Replaced the Go Direct Runner



BEAM
SUMMIT

Demo



BEAM
SUMMIT

History & Rationale



BEAM
SUMMIT

Go Direct Runner

- Batch only
- Basic DoFns
- Side Inputs
- Simple GroupByKey
- Combiners
- Executed SplittableDoFns
- Limited Windowing Support



Observation

If the default runner for an SDK was unable to support a feature, then users make the mistake in thinking that the SDK didn't support the feature either.

Go Direct Runner

- Missing:
 - State, Timers, Triggers, Windowing Strategies, Cross Language
- Too Permissive
 - Didn't require DoFn or Type Registrations
 - Didn't serialize elements
 - Allowed Closure functions
- Made it easy to write code that couldn't work on other runners



The Timeline

- Started authoring a portable first runner. ~2021
 - First commit in personal repo: ~January 2022
 - Migrated code to Beam repo: February 2023
 - Default for the Go SDK: July 2023 for v2.50.0
- [GopherCon Europe 2022 - Stream Processing End to End](#)
 - Revealed the true need
- I gave talks at the last two Beam Summits which outlined some of the progress. You can find them and their slides
 - [Beam Summit 2022](#)
 - [Beam Summit 2023](#)
- Beam v2.59.0 - Available to Java & Python SDKs



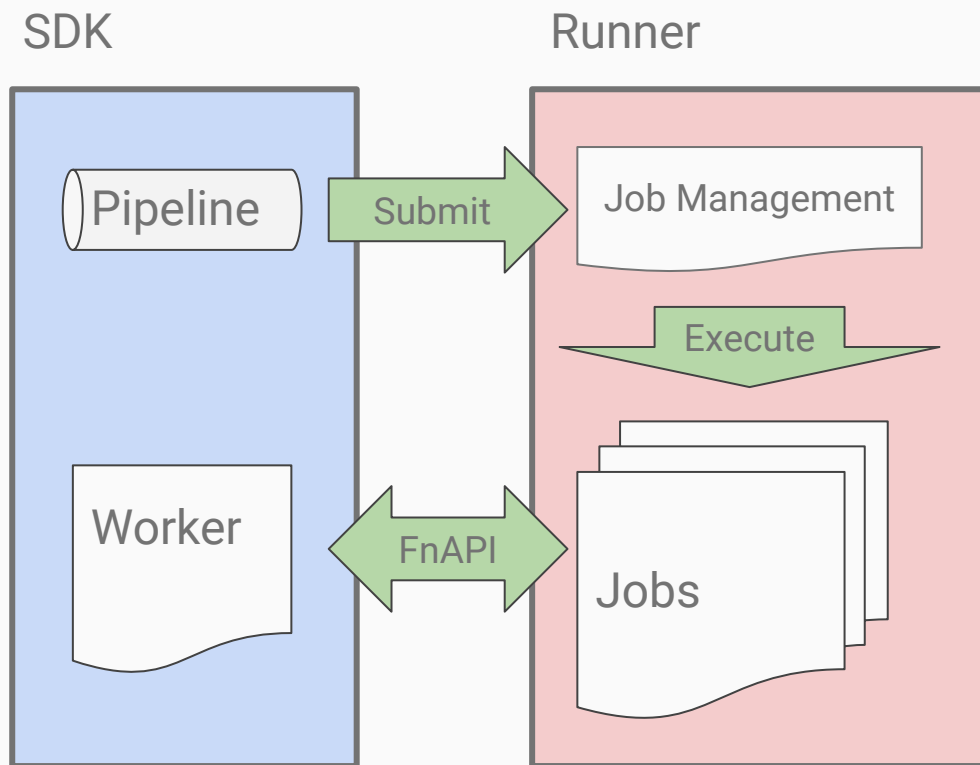
Structure



BEAM
SUMMIT

Portable Beam Runners - Generally

- Allow Job Submission via a JobManagement GRPC Services
- Runner uses SDK configuration to manage SDK workers
 - Loopback
 - Docker
 - Process
- Execute jobs.
- SDK side via FnAPI GRPC services
- Report back results via Job Management



github.com/apache/beam/sdks/go/pkg/beam/runners/prism

- prism.go - external entry point
- /internal - **Job Executor**, Stage config, Pipeline Preprocessing
 - /config - Future fine grain configuration handling infrastructure
 - /engine - Watermark handling, **ElementManager**, StageState
 - /jobservices - **Beam JobManagement** GRPC services
 - /urns - Beam Urns
 - /web - **UI**
 - /worker - **Beam FnAPI & SDK interactions**

A simple system may or may not work. A complex system that works is invariably found to have evolved from a simple system that worked. A complex system designed from scratch never works and cannot be patched up to make it work. You have to start over with a working simple system.

~John Gall (1975) Systemantics: How Systems Really Work and How They Fail

github.com/apache/beam/sdks/go/pkg/beam/runners/prism

- prism.go - **trunk (internal, jobservices, web)**
- /internal - **trunk (config, engine, urns, worker)**
 - /config - **no protos - leaf**
 - /engine - **no protos (config)**
 - /jobservices - **Protos (config, urns, protos)**
 - /urns - **Protos - leaf**
 - /web - **trunk, protos, independant**
 - /worker - **protos, (engine)**

The Loop



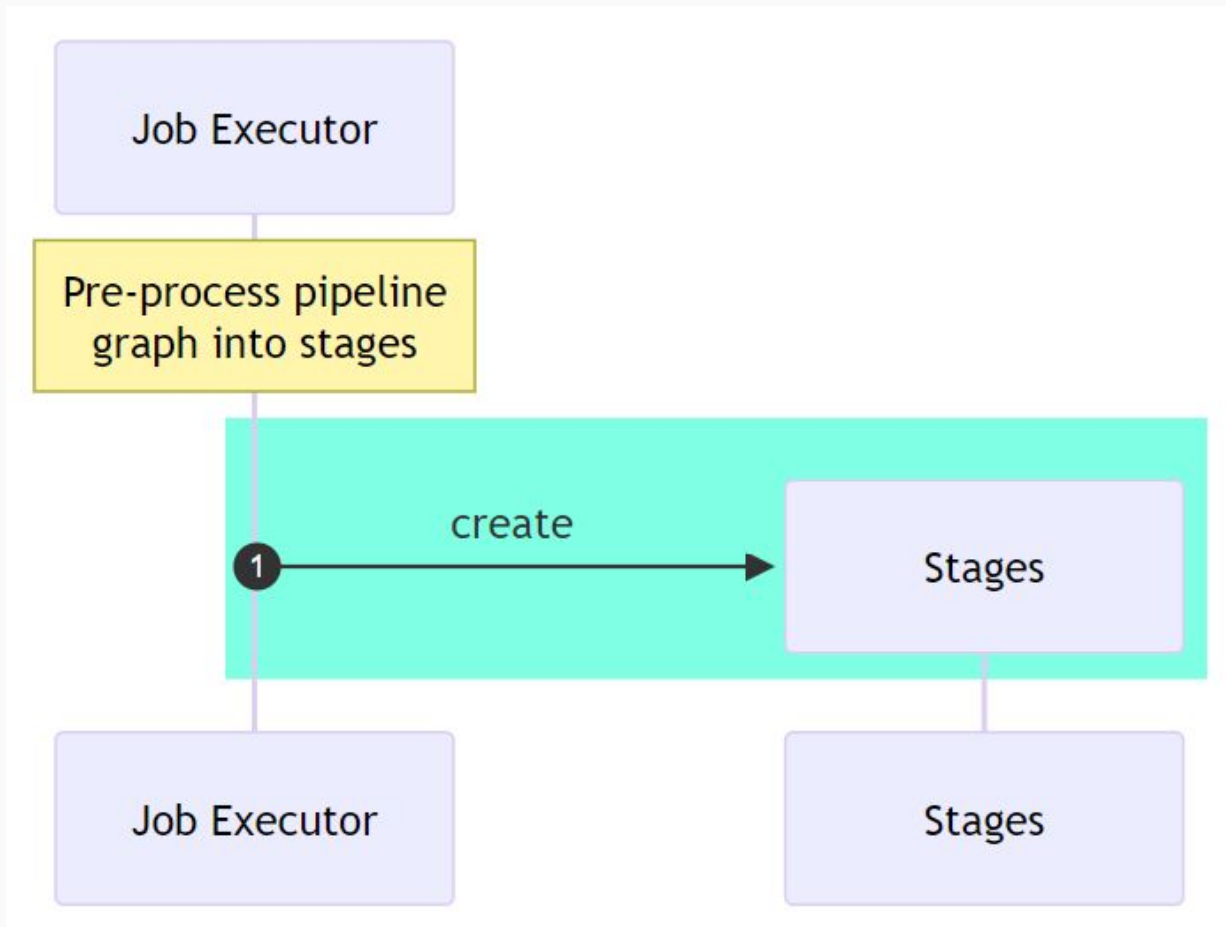
BEAM
SUMMIT


```
graph TD; A[Job Executor] --- B[Pre-process pipeline graph into stages]; B --- C[Job Executor];
```

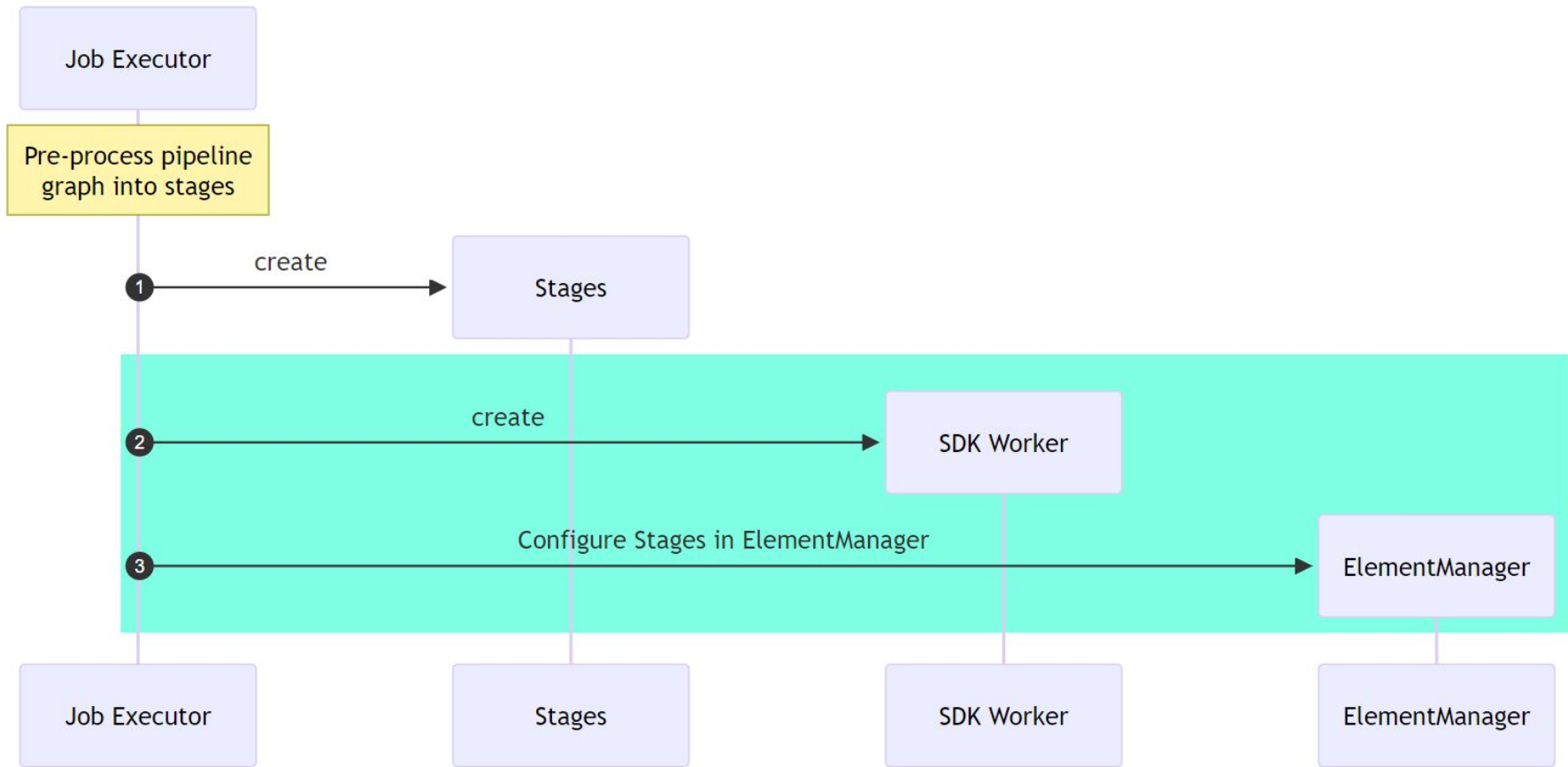
Job Executor

Pre-process pipeline
graph into stages

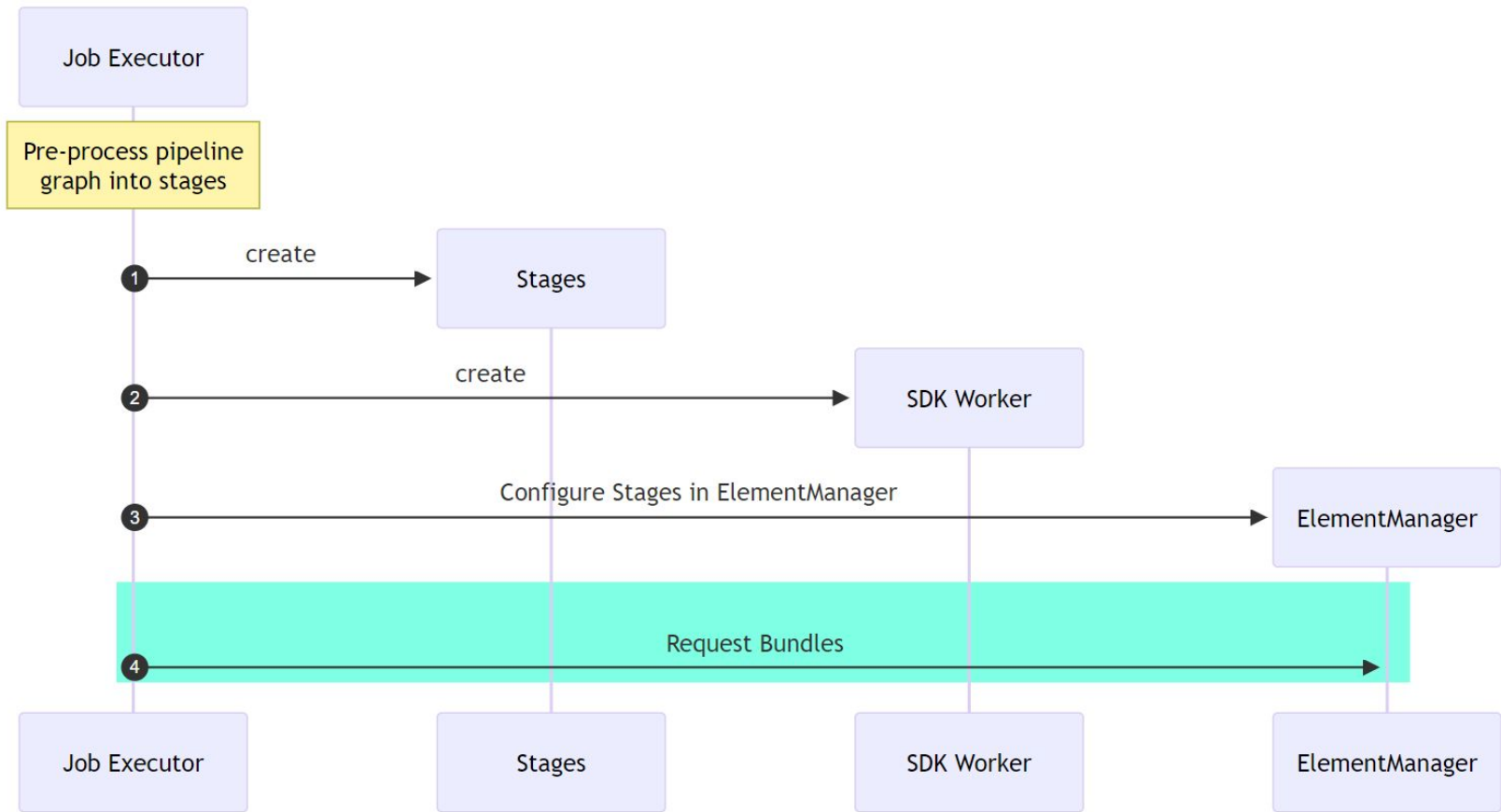
Job Executor



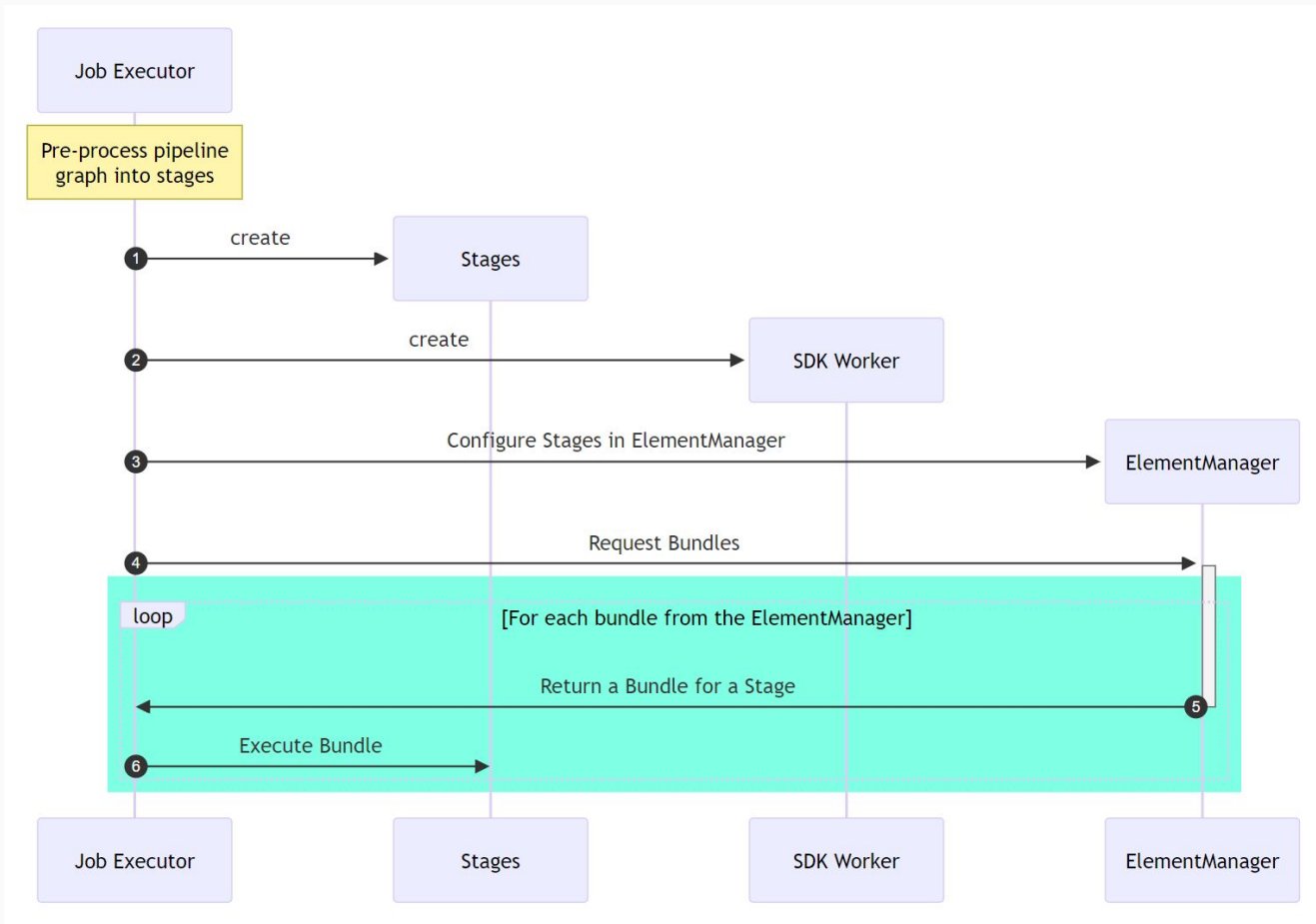
Stages are created to manage runtime metadata



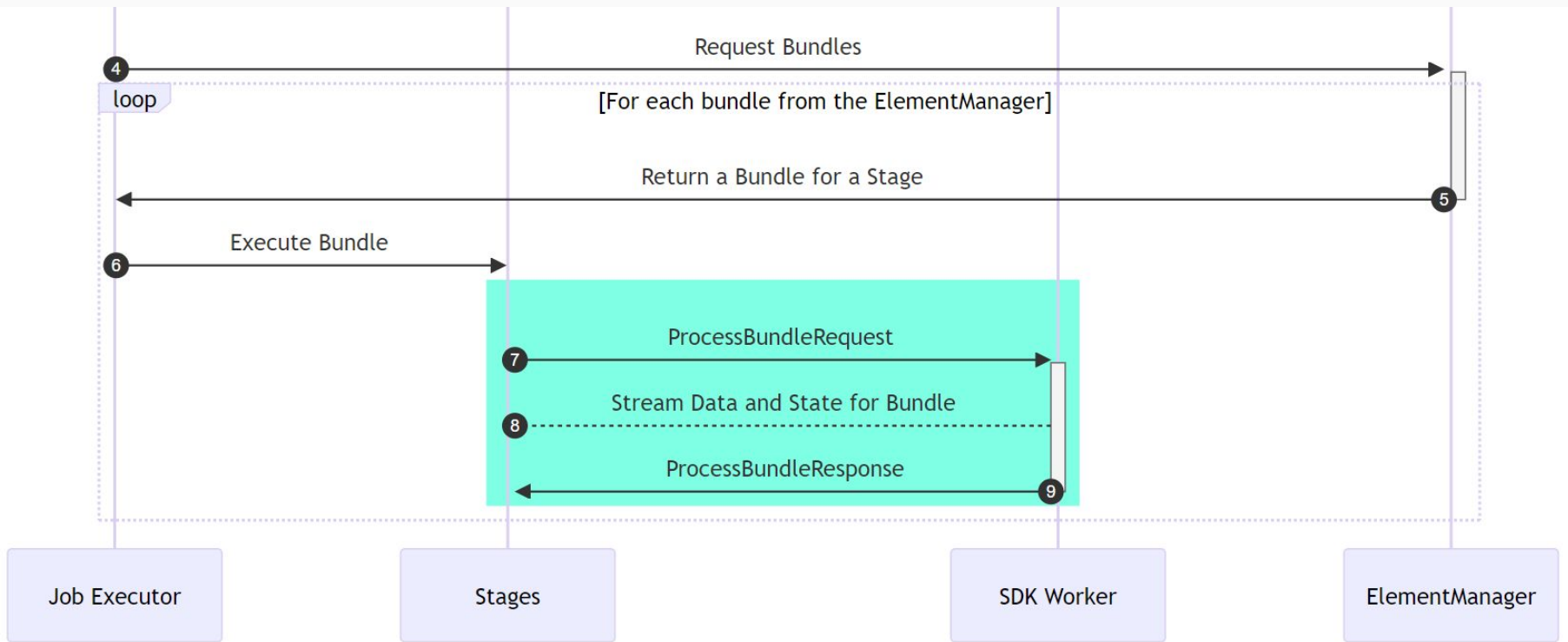
Create stateful handlers - workers for each environment, and configure the Element Manager

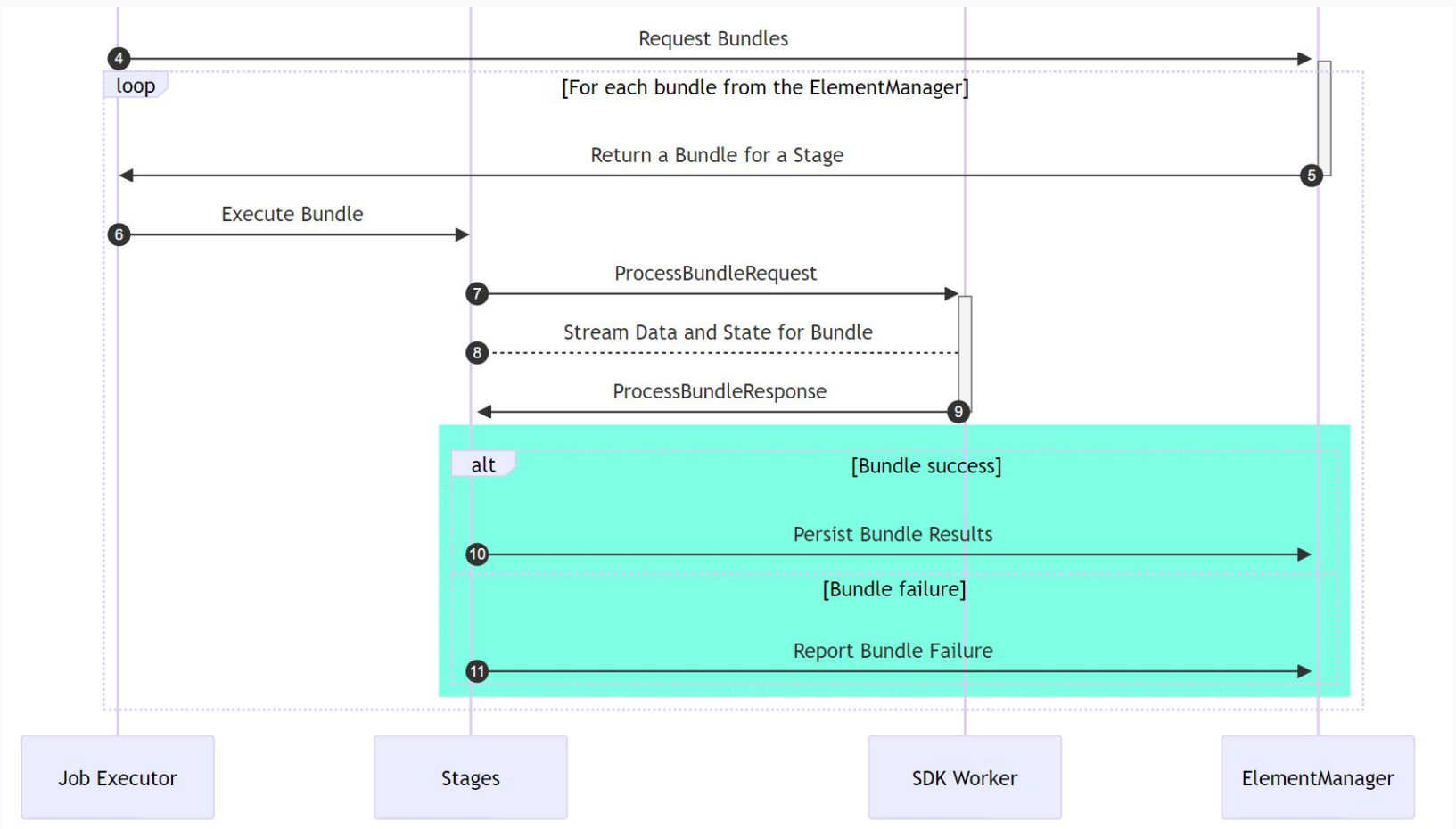


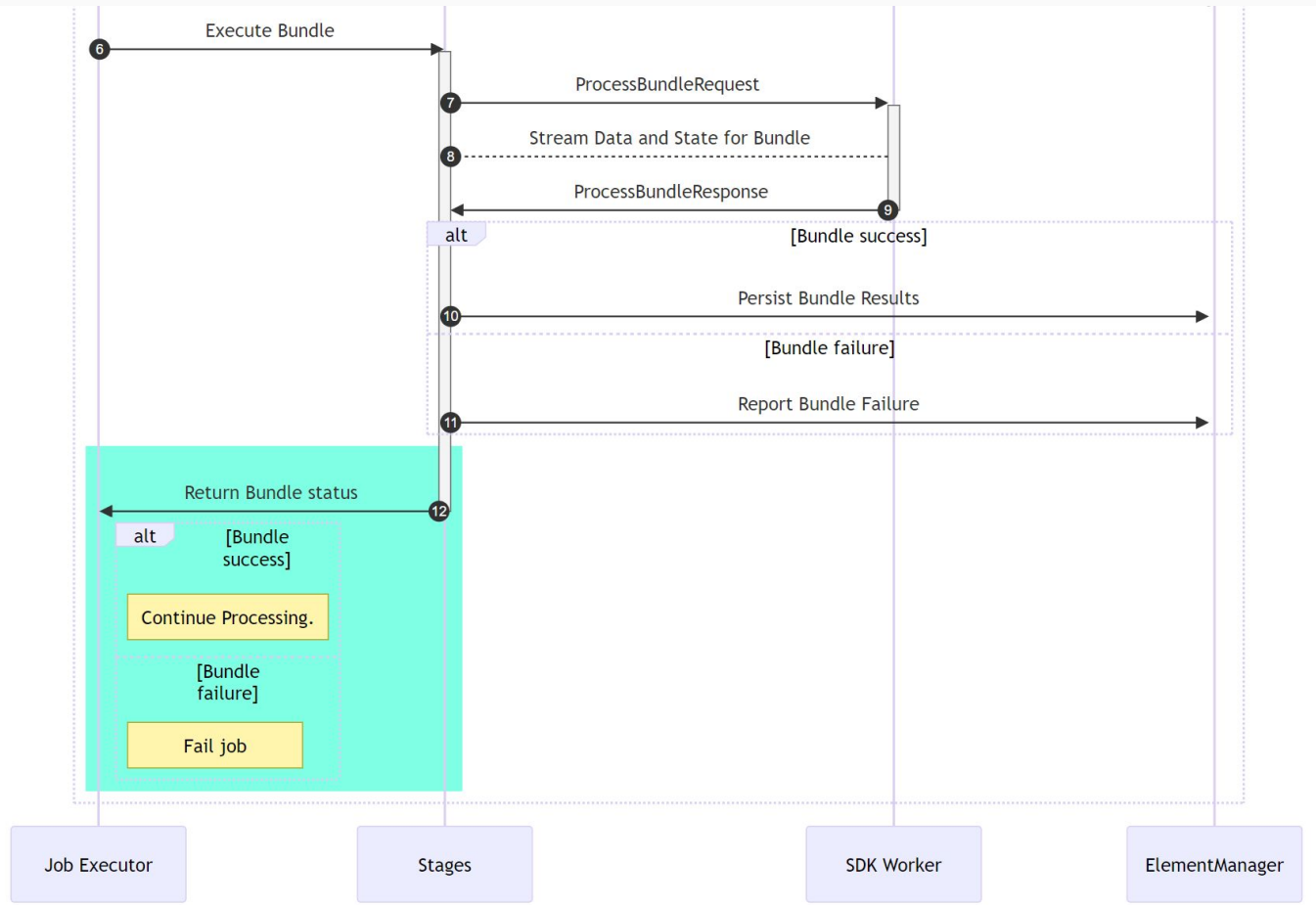
Job Executor requests bundles from the Element Manager



Bundles are directed to their matching stage to be executed







About Time



BEAM
SUMMIT

Updating a Watermark

`Watermark_PCol = Watermark_Out_ProducingPTransform`

U = "Union of"

`Next_Watermark_In =`

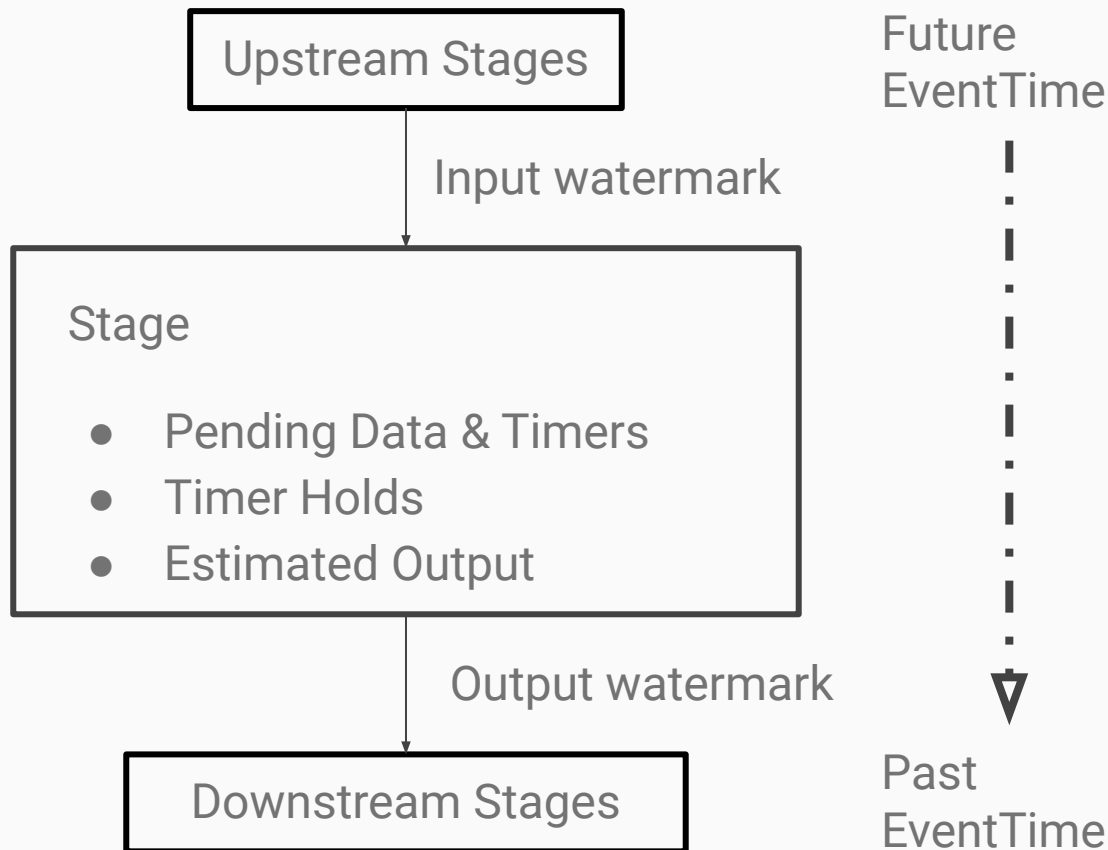
```
MAX(  
  Watermark_In,  
  MIN(  
    U(TS_Pending),  
    U(Watermark_InputPCol)  
  )  
)
```

`Next_Watermark_Out =`

```
MAX(  
  Watermark_Out,  
  MIN(  
    Next_Watermark_In || Estimated_Out  
    U(minWatermarkHold)  
  )  
)
```

Propagate Output Watermark to Downstream Stages if it has changed.

See the "updateWatermarks" method for this in code.



BEAM
SUMMIT

ElementManager

Prism's Heart

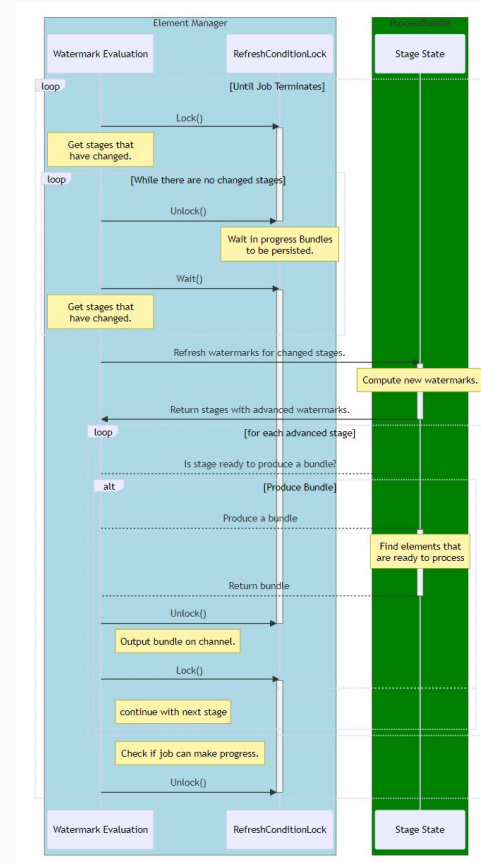
- Tracks state for each stage
 - Pending Elements
 - Derives Watermarks
- Evaluates stages for bundle readiness
- Divides work among bundles



BEAM
SUMMIT

Bundle Generation

- Currently Single threaded
- Determined by a stage's watermark
- Stage marked for evaluation when changed by PersistBundle
 - New pending elements from upstream
 - Old elements successfully processed.
 - Timers fired



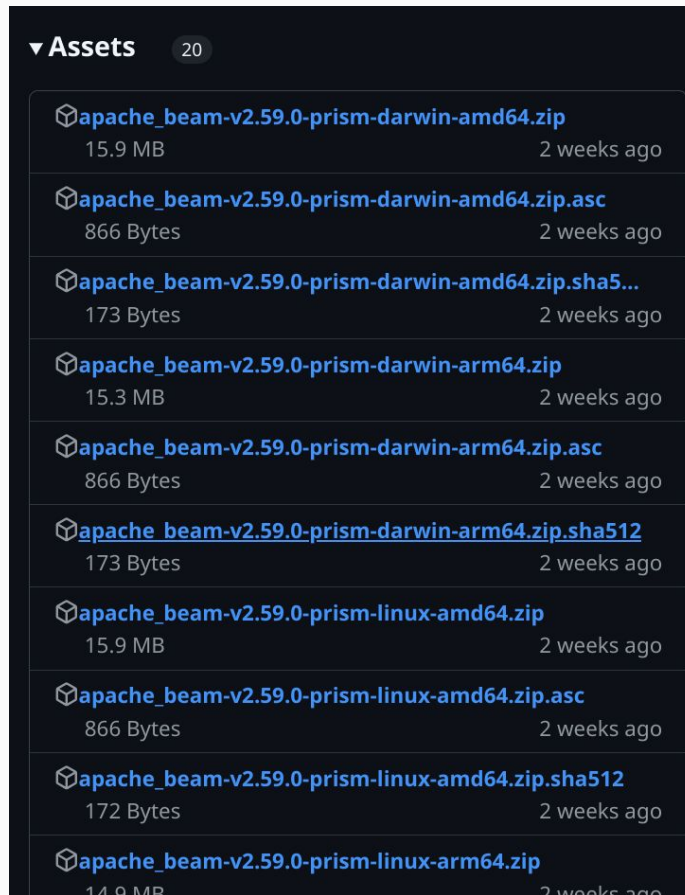
Deployment



BEAM
SUMMIT

Pre-built Binaries

- Built as part of the Beam Release
 - Built for Linux, Windows, and Mac (darwin) for both ARM64 and AMD64
 - Leveraging Go's compact binaries and cross compilation abilities
- Hosted on Github as Release Assets
- With v2.59.0 Automatically downloaded when using PrismRunner in the Java and Python SDKs
 - Try with `-runner=PrismRunner`



▼ Assets 20	
apache_beam-v2.59.0-prism-darwin-amd64.zip	15.9 MB 2 weeks ago
apache_beam-v2.59.0-prism-darwin-amd64.zip.asc	866 Bytes 2 weeks ago
apache_beam-v2.59.0-prism-darwin-amd64.zip.sha5...	173 Bytes 2 weeks ago
apache_beam-v2.59.0-prism-darwin-arm64.zip	15.3 MB 2 weeks ago
apache_beam-v2.59.0-prism-darwin-arm64.zip.asc	866 Bytes 2 weeks ago
apache_beam-v2.59.0-prism-darwin-arm64.zip.sha512	173 Bytes 2 weeks ago
apache_beam-v2.59.0-prism-linux-amd64.zip	15.9 MB 2 weeks ago
apache_beam-v2.59.0-prism-linux-amd64.zip.asc	866 Bytes 2 weeks ago
apache_beam-v2.59.0-prism-linux-amd64.zip.sha512	172 Bytes 2 weeks ago
apache_beam-v2.59.0-prism-linux-arm64.zip	14.9 MB 2 weeks ago



Surprises



BEAM
SUMMIT

Surprises: More Permissive than Expected

- SDKs may
 - “Pre-optimize” pipelines, using typically post Optimized Pipeline constructs on submission.
- Composite transforms may
 - be empty or do non-transform things.
 - May have non-standard unknown URNs.
- Flattens may have
 - no input collections at all.
 - Real usage: “optional” side input data.
 - distinct coders for all involved PCollections
 - The inputs and outputs may not have to match ever.
 - Only truly solvable via removing the flatten via unzipping (currently unimplemented)



Problem: Spuriously passing Java Tests

Root Cause:

- Java Pipeline Tests use a *TestPipeline* and just call `pipeline.run()`
- No blocking means tests spuriously pass, since they don't wait for pipeline termination.

Solution:

- Have a `TestPrismRunner` wrapper to have the blocking behaviour.
 - This is what all the other wrappers do as well.



Problem: Missing Java Metrics

Root cause: Java was sending legacy MonitoringInfo metrics

- Prism only understands the modern ShortID format (for compact representation)
- Java Loopback was never migrated to use the Beam Provisioning APIs
- Runner Capabilities were not being respected.
- The FnHarness kept sending the old format.

Solution: Fix Java Loopback mode ([#32198](#))



The Future



BEAM
SUMMIT

Near Term

- Start to include Prism around the Beam Site
 - Quickstarts, Capability Matrix etc.
- Continue adding new features
 - Bundle Finalization, OnWatermarkExpiry, Triggers
- Improve the Prism documentation
 - <https://github.com/apache/beam/pull/32143>
 - Contents of this talk, and more.
- Blog Post on Beam site.



One Consistent Onboarding Runner

- Become the Default Runner for all current and future Beam SDKs
 - Consistent experience for all SDKs
 - Complete support of the Beam Model
 - Pass all Validates Runner tests in all SDKs
 - Address profiling hotspots
 - Validate support of further IOs and test pipelines.
- Possible Goal: Language agnostic SDK authoring Guide coordinated with Prism
 - Specific Debug logging configurations to assist in implementing different Portable SDK features
 - Eg. Verbose logging around metrics, when working on new Metric
 - Hard disable permissive alternative paths
 - Require that SDK implements a feature correctly instead of using fallback implementations.
 - SDK Authoring Guide becomes a “how does Beam Portability work?” exercise.



Future Work - <https://github.com/apache/beam/issues/29650>

- Implement non-user visible Beam features
 - Elements on ProcessBundleRequest and Response
 - Parameterized Window Coders
 - State Cache management
 - And more...
- Improve Configurability
 - Enable/Disable features via Pipeline Options
 - Configurable Advanced Debug logging
- Improve “Single Box Runner” abilities
 - Improve UI
 - Improve default execution performance
 - Bundle sizing & splitting behaviors
 - Parallelize Bundle generation
 - Reliability
 - Durable Data
 - Pipeline Update & Restarts
- Notebook runner?



Thank You!

Robert Burke

lostluck@apache.org

@lostluck on Github

- Umbrella Tracking Issue
 - <https://github.com/apache/beam/issues/29650>
- Tag @lostluck for prism interest/questions/comments
- Try Prism in Java and Python with `-runner=PrismRunner`



BEAM
SUMMIT