# arrow-rs + arrow2

Summary of https://github.com/apache/arrow-rs/issues/1176

Disclaimer: this is a work in progress. Comments / suggestions welcome. Please see https://github.com/apache/arrow-rs/issues/1176 for full details

# Background

arrow2 is port/rewrite of arrow-rs by @jorgecarleitao that provided improved type safety and ergonomic APIs to an Arrow Rust library

arrow-rs is the “official” implementation of Arrow, governed by the Apache Software Foundation. It has regular biweekly releases, a substantial committer base, and large featureset

Many of the ideas that arrow2 innovated have since been incorporated into arrow-rs (e.g. decimals as primitives), and vice versa (e.g. arrow2/1287 )

# Major users of arrow2/arrow-rs

Arrow2

- Polars
- Databend
- crates.io reverse dependencies (23)

Arrow-rs

- DataFusion and its known users (21)
- crates.io reverse dependencies (65)

# Why Unify

Unify limited maintenance bandwidth on one implementation rather than 2 (e.g. to get full nested parquet support and late materialization support)

Avoid forcing potential Arrow users to pick between two “similar but different” implementations of Arrow which are hard to switch between and fragments the overall Arrow ecosystem

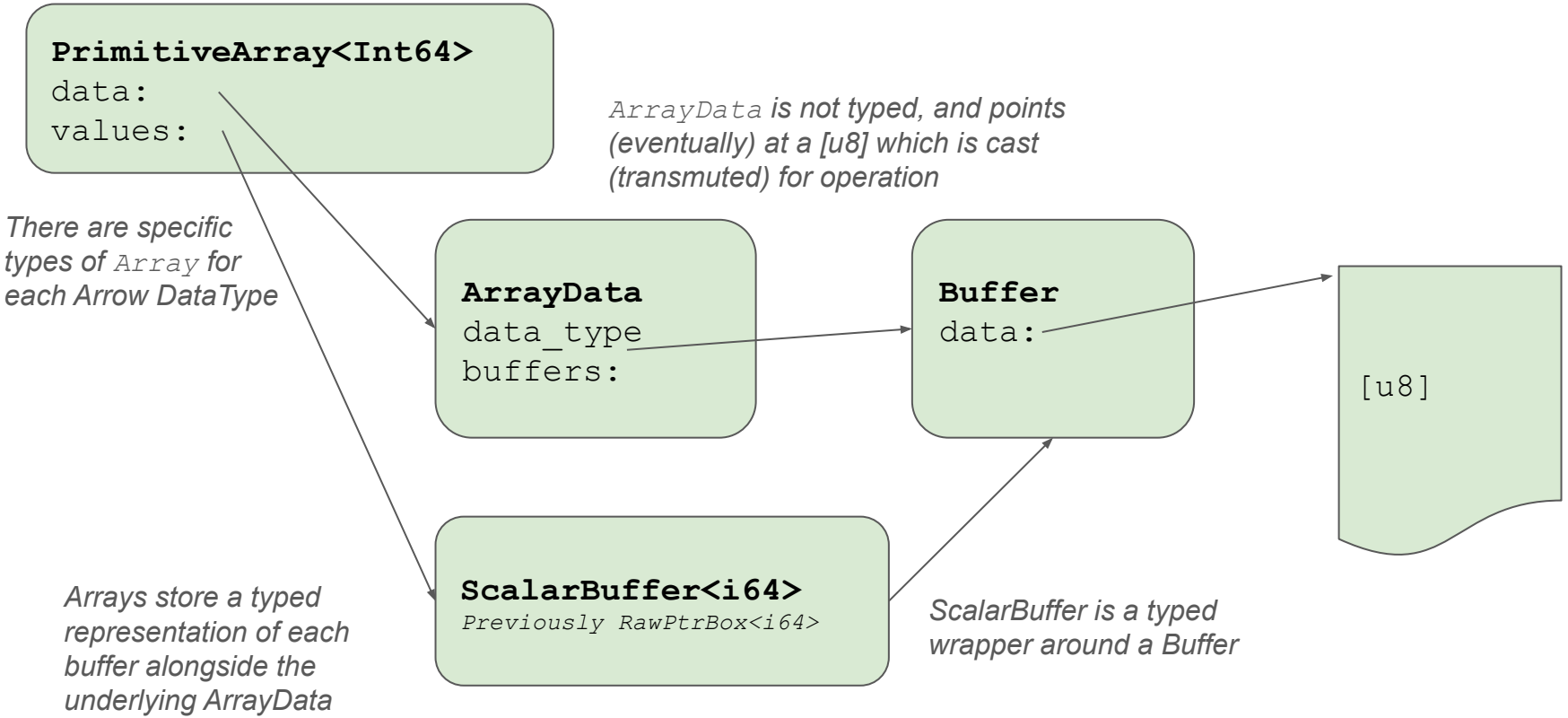
# Steps to get us there

1. Common low level code structure (diagrams next)
2. Apache Software Foundation IP Clearance
3. Community continuance (ensure relevant arrow2 contributors are committers on the ASF project)

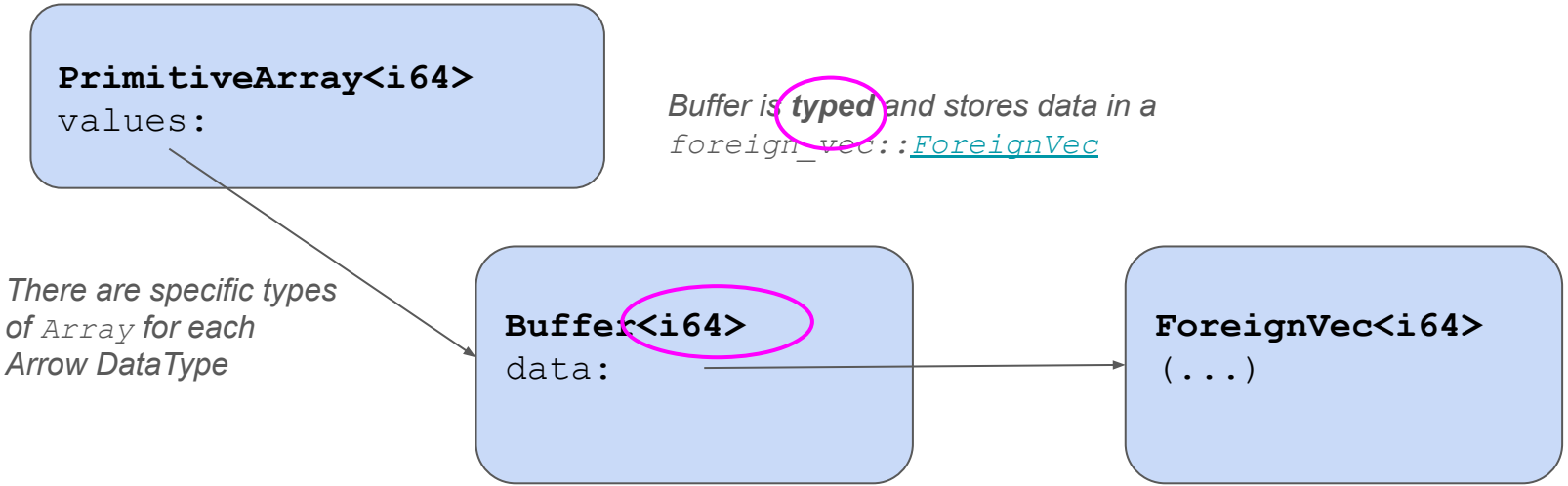
Longer term

1. Unify kernels
2. Unify IO
3. Unify documentation
4. Unify tests

# Current structure (arrow-rs)



# Current structure (arrow2)




*ForeignVec: is a `Vec<T>` with a custom `Drop` and handles unsafe conversions between (potentially foreign allocations).*

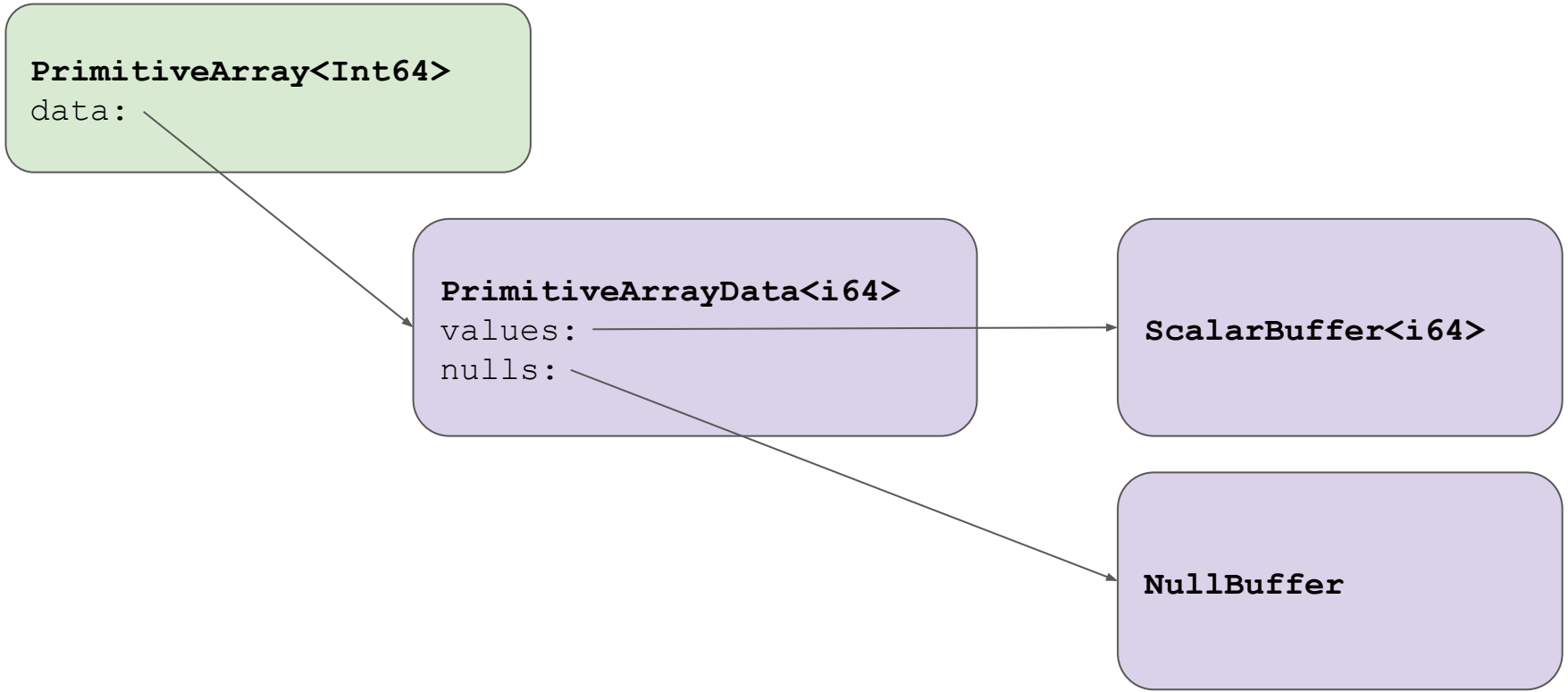
# Unifying ArrayData

*`ArrayData` becomes an enum of one of several typed variants*

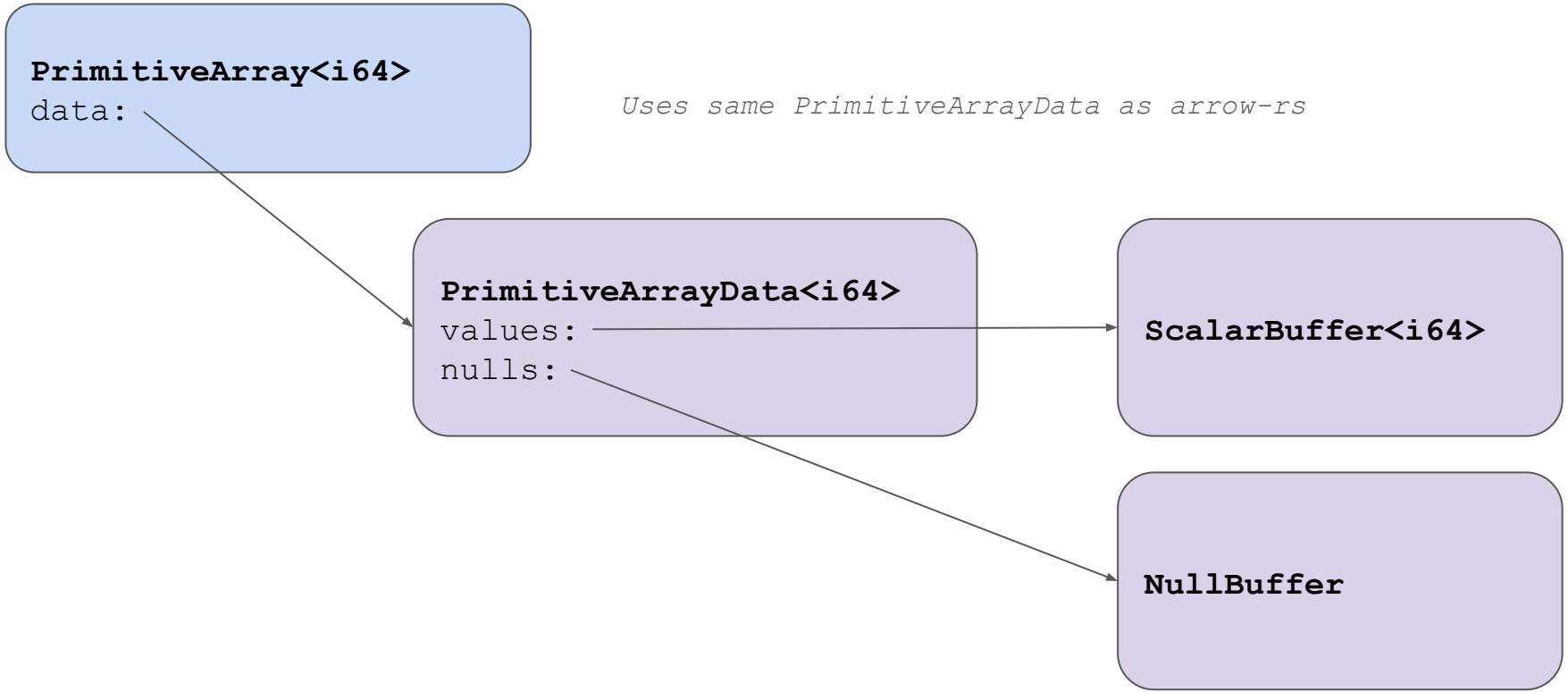
```
enum ArrayData {
  Primitive(ArrayDataPrimitive),
  Bytes(ArrayDataBytes)
  Dictionary(ArrayDataDictionary),
  …
}
```

```
enum ArrayDataPrimitive {
  Int8(PrimitiveArrayData<i8>),
  Int16(PrimitiveArrayData<i16>),
  Int32(PrimitiveArrayData<i32>),
  Int64(PrimitiveArrayData<i64>),
  …
}
```

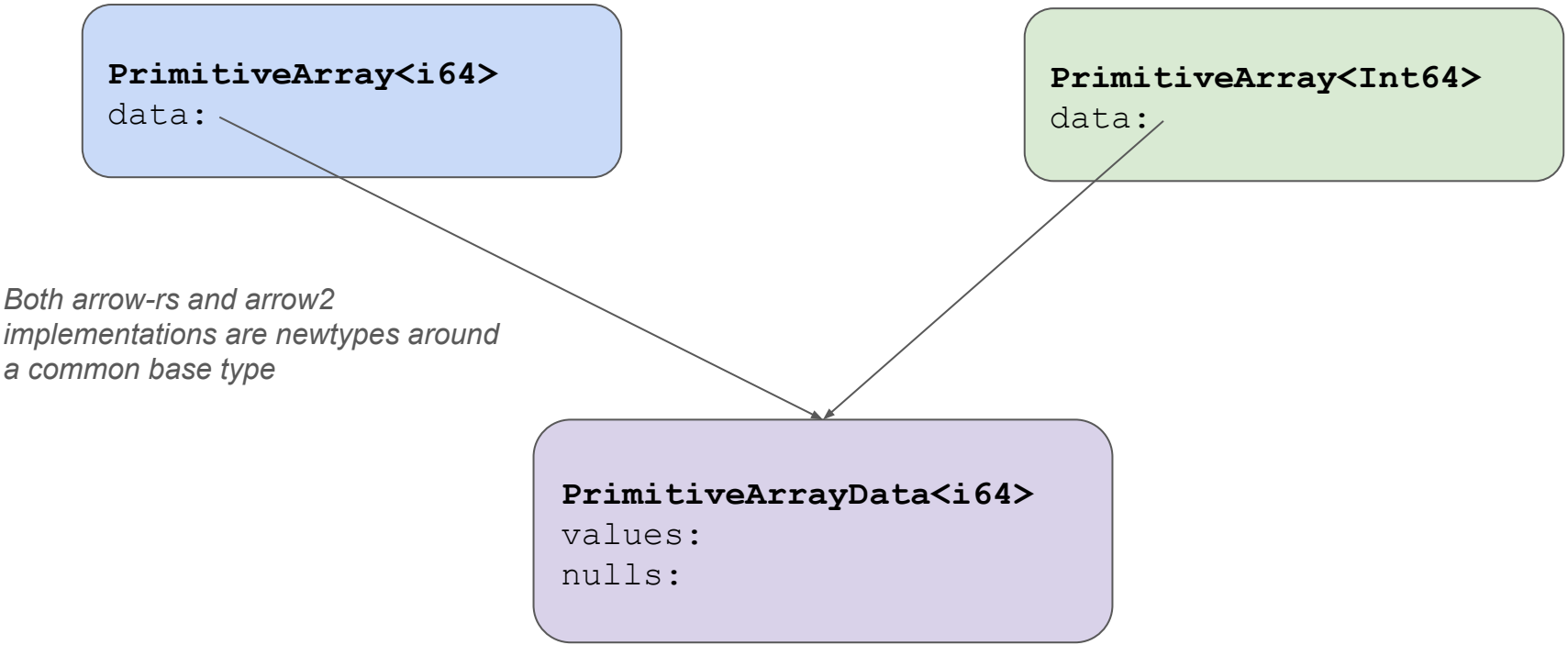
# Proposed new structure of arrow-rs



# Proposed new struture of arrow2

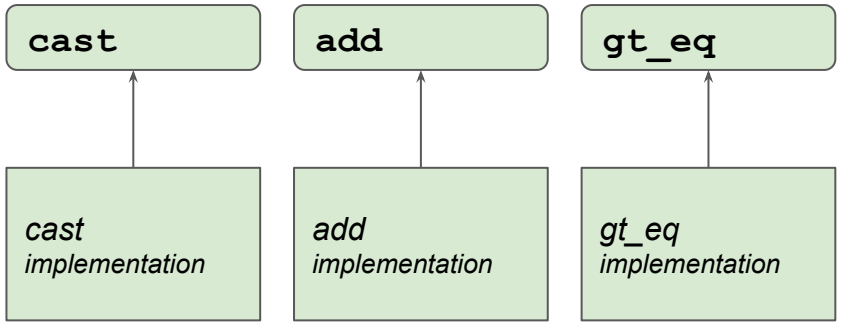


# Interoperability

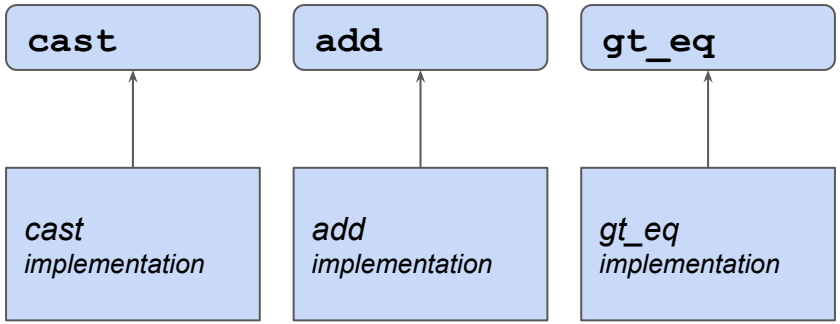



*Both arrow-rs and arrow2 implementations are newtypes around a common base type*

# Current Structure: Kernels




*Implementation in terms of arrow-rs::Arrays / ArrayData*



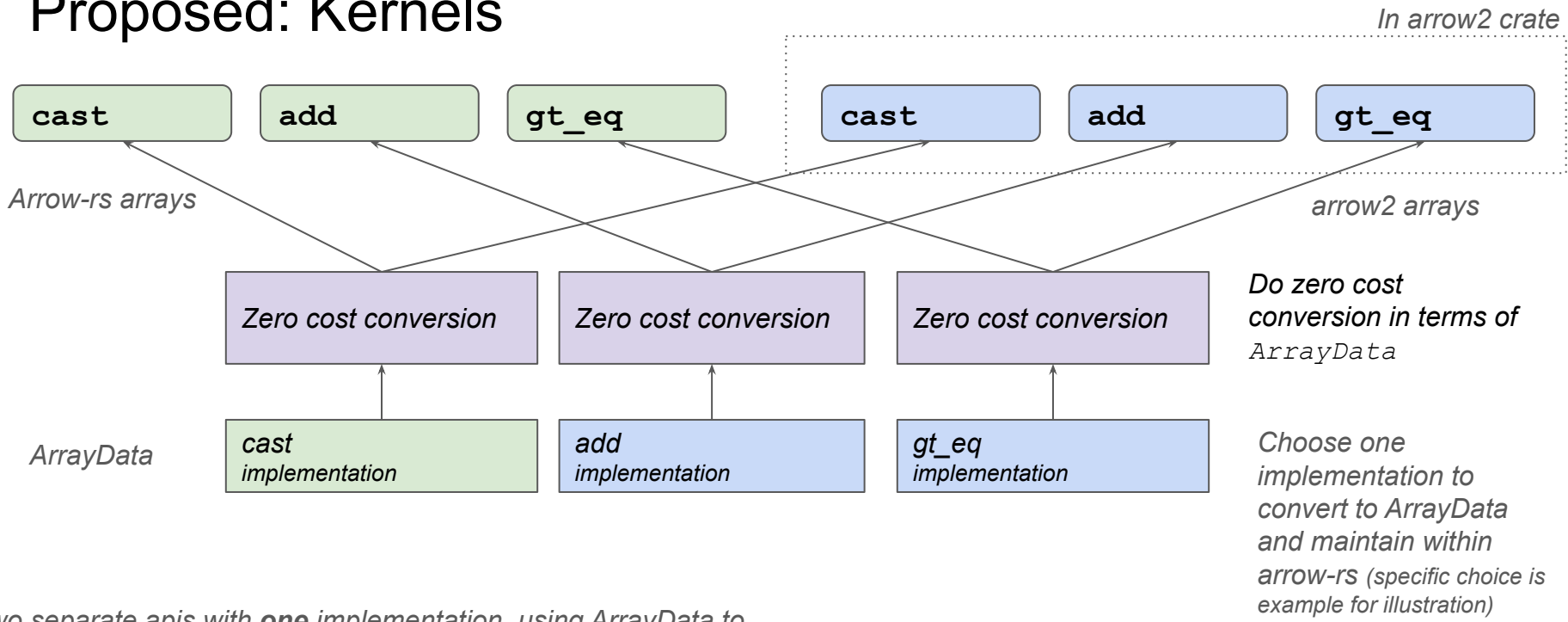

*Implementation in terms of arrow2::Arrays / Buffer<T>*

*Two separate apis with two separate implementations*

arrow-rs

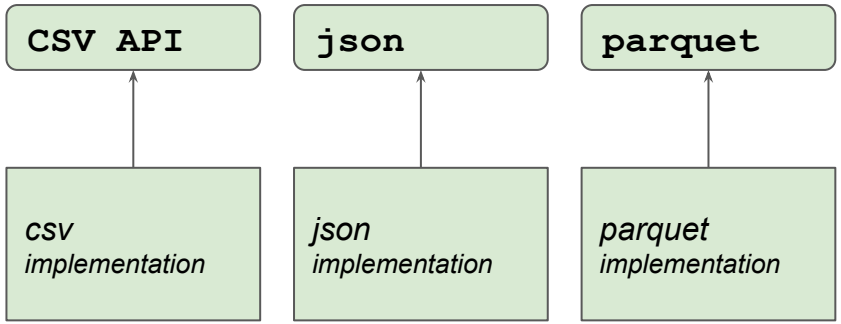
arrow2

# Proposed: Kernels

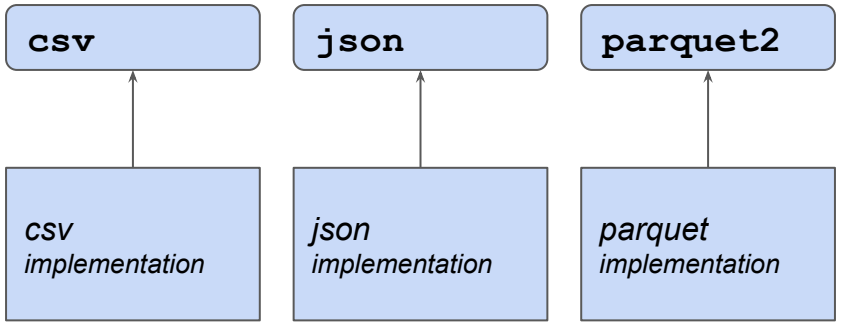



*Two separate apis with **one** implementation, using ArrayData to convert between one and the other*

# Current Structure: I/O



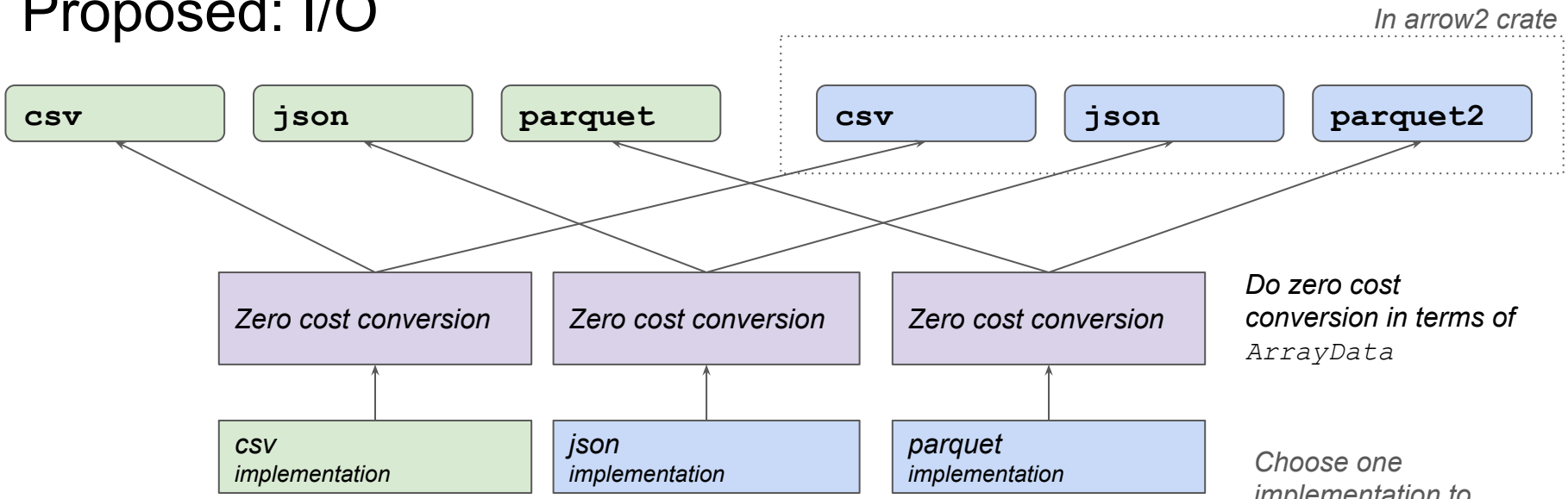

*Implementation in terms of arrow-rs::Arrays / ArrayData*




*Implementation in terms of arrow2::Arrays / Buffer<T>*

*Two separate apis with two separate implementations*

# Proposed: I/O




*Do zero cost conversion in terms of* `ArrayData`

*Choose one implementation to convert to ArrayData and maintain within arrow-rs (specific choice is example for illustration)*

*Two separate apis with **one** implementation, using ArrayData to convert between one and the other*

# Deprecation

- Some low-level APIs may not be possible to unify
- Deprecate and remove functionality where unification is not possible

# IP clearance

Plan IP clearance on the entire arrow2 repo (any others?)

Not clear how much would be copied in bulk or in parts

# Dangers

- Run out of ambition and the projects are left in half way state
- Frustrate downstream users with intrusive breaking changes