Fast data processing
with Ruby and Apache Arrow

Sutou Kouhei

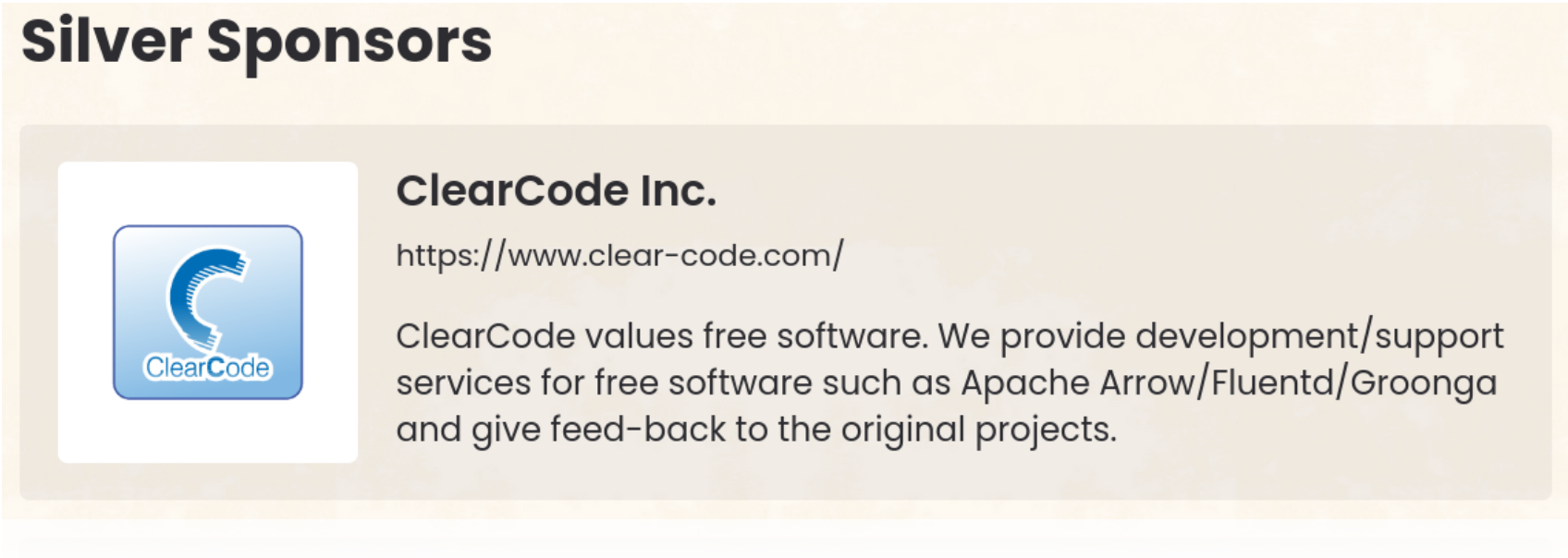
ClearCode Inc.

RubyKaigi 2022
2022-09-10

Sutou Kouhei
A president Ruby committer

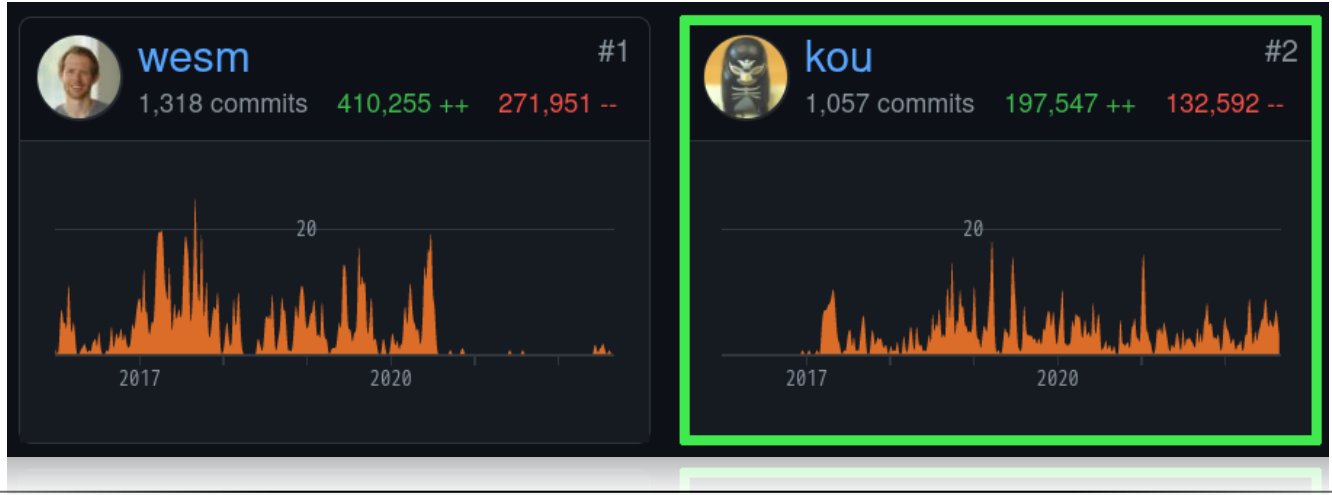
The president of ClearCode Inc.

クリアコードの社長



Sutou Kouhei
The Apache Arrow PMC chair

- ✓ PMC: Project Management Committee
  Apache Arrowのプロジェクト管理委員会のリーダー
- ✓ #2 commits（コミット数2位）



Sutou Kouhei
The pioneer in Ruby and Arrow

- ✓ A Ruby committer
  - ✓ Maintain some standard libraries/default gems
    標準ライブラリーとかデフォルトgemのメンテナンスをしている
- ✓ The author of Red Arrow
- ✓ Red Arrow:
  - ✓ The official Apache Arrow library for Ruby
    公式のRuby用のApache Arrowライブラリー

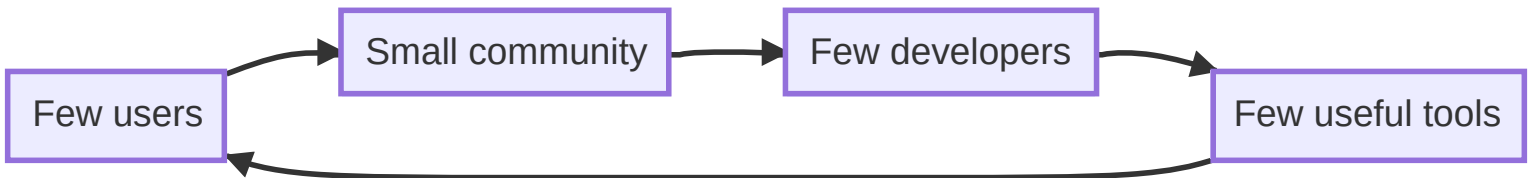
Why do I work on Red Arrow?
なぜRed Arrowの開発をしているか

- ✓ To use Ruby for data processing too!
  データ処理でもRubyを使いたい！
  - ✓ At least a part of data processing
    データ処理の全部と言わず一部だけでも
- ✓ Data processing is an important task
  データ処理は最近の重要なタスクの1つ
  - ✓ # of Rubyists will be increased by this
    データ処理にRubyを使えるようになるとRubyistが増えるはず

Current situation

Negative spiral

今は負のスパイラル

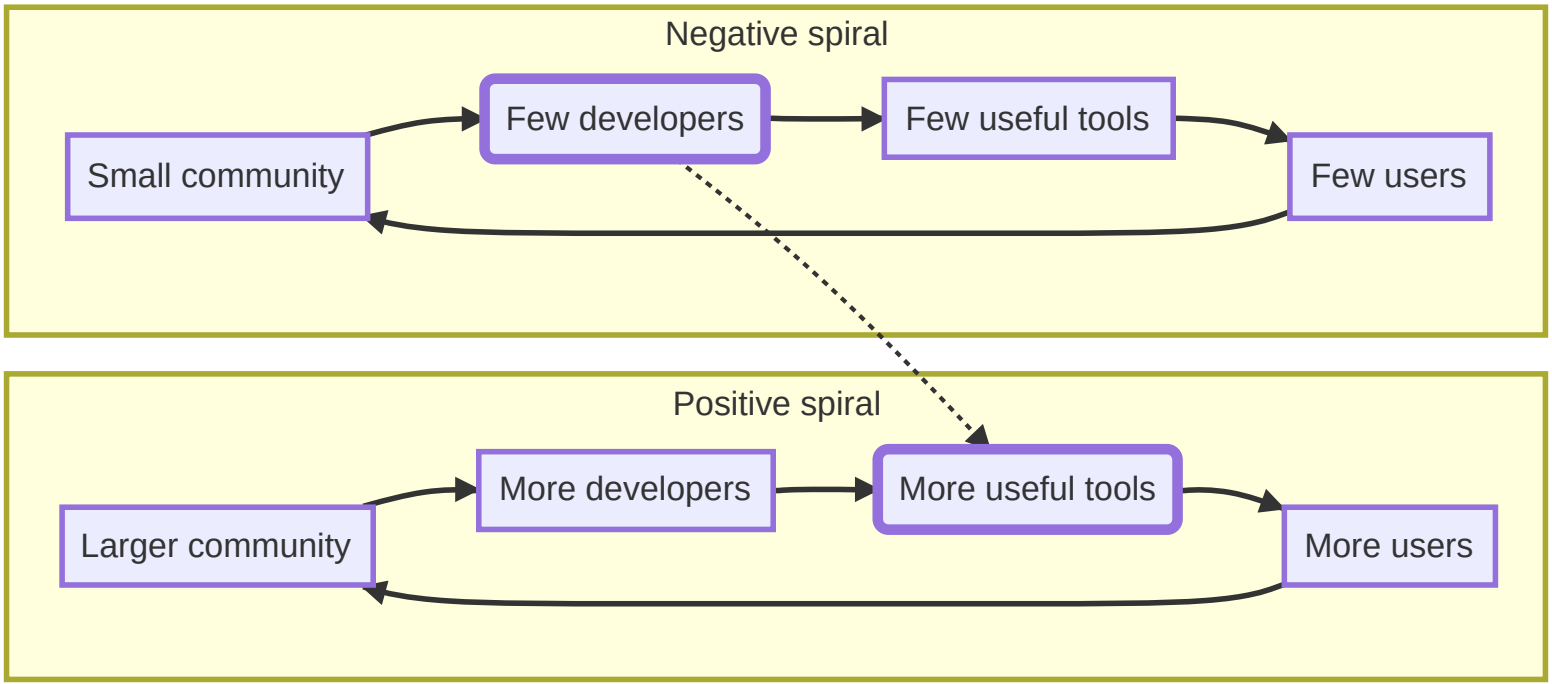


How to break the negative spiral?

どうやってこの負のスパイラルを打開する？

Expand useful tools with few developers

少人数で便利なツールを増やせればいいんじゃない？



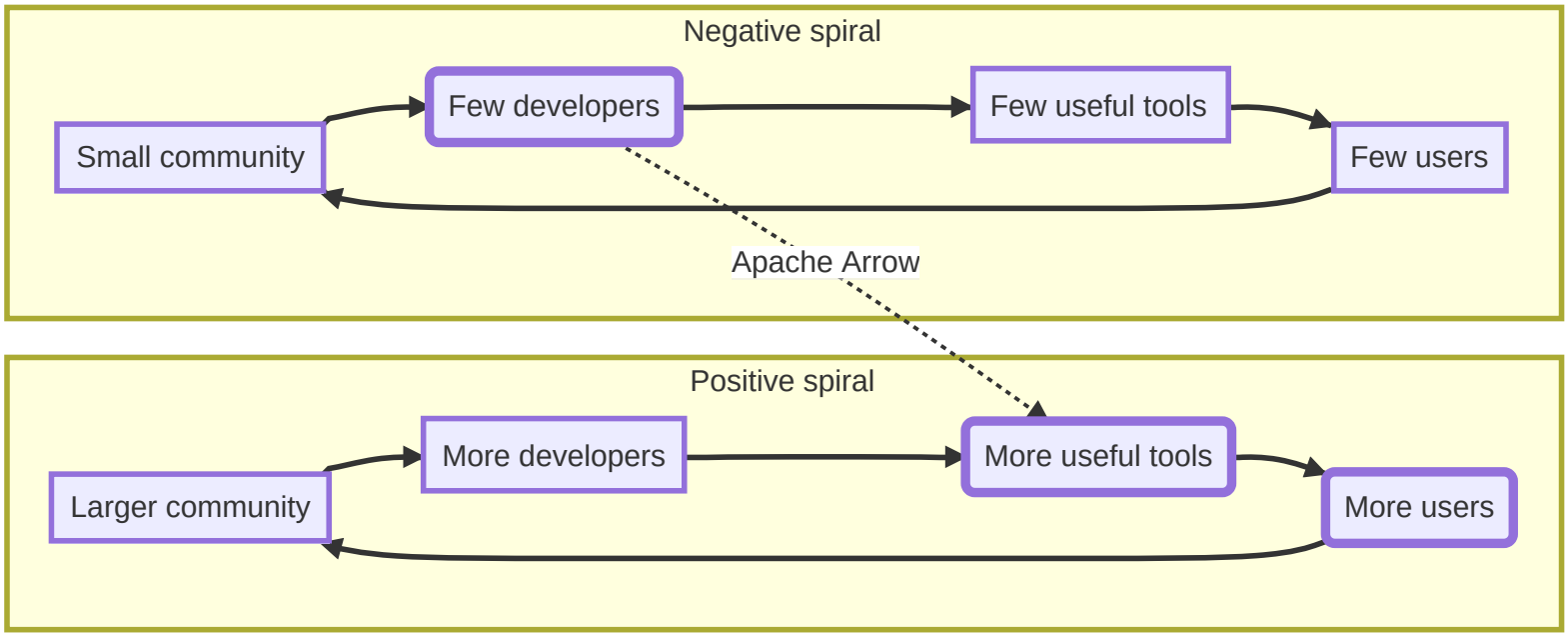
But how?
でもどうやって？

Apache Arrow

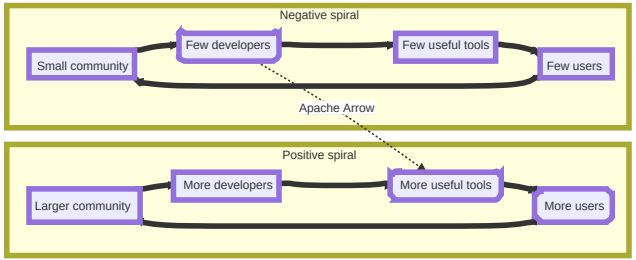
Apache Arrow

- ✓ **Cross-language** dev platform for data
  複数言語対応のデータ用の開発プラットフォーム
  - ✓ Ruby community doesn't need to dev everything
    Rubyコミュニティーがすべてを開発しなくてもよい
  - ✓ We can share common implementations
    共通の実装を言語を超えて共有できる
- ✓ Today's highlighted features
  今日注目する機能
  - ✓ Fast data processing（高速データ処理）
  - ✓ Fast data interchange（高速データ交換）

My approach
私のアプローチ



Goal of this talk
このトークのゴール



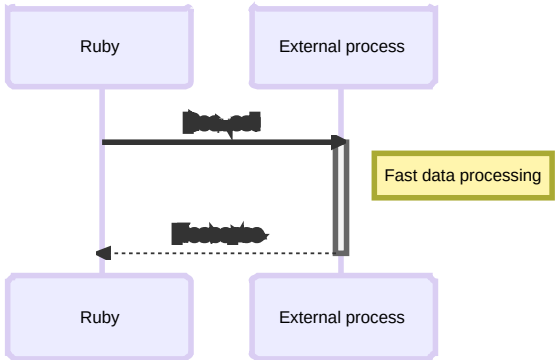
- ✓ You want to use Ruby for some data processings
  いくつかのデータ処理でRubyを使いたくなる
  - ✓ Especially, you want to implement a BI tool
    特にBIツールを作りたくなる
- ✓ You join Red Data Tools project
  Red Data Toolsプロジェクトに参加する
  - ✓ It provides data processing tools for Ruby
    Ruby用のデータ処理ツールを提供するプロジェクト
    https://red-data-tools.github.io/

Fast data processing
高速データ処理

- ✓ Ruby is slow to process data
  Rubyでデータを処理すると遅い
- ✓ Resolve in external process: (別プロセスで解決)
  Use case: Web app, batch process for Web app
  - ✓ Use fast data processing module (e.g.: DB)
    DBとか速いデータ処理モジュールを使う
- ✓ Resolve in the same process: (プロセス内で解決)
  Use cases: IRB, batch process for Web app
  - ✓ Implement core features in other fast lang
    他の速い言語でコアの機能を実装

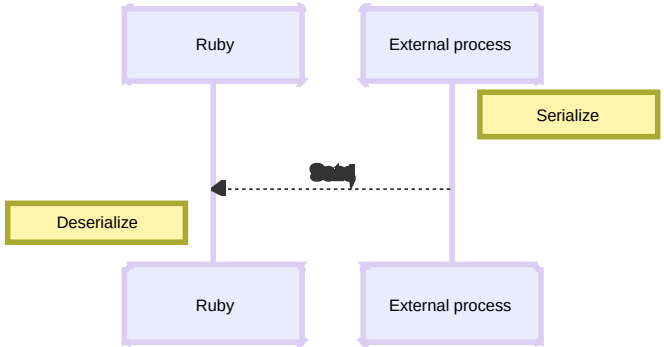
External process
別プロセス

- ✓ Popular case
  in current Ruby usage
  今のRubyの使われ方だとよくあるケース
- ✓ Small response: No problem
  レスポンスが小さい場合は問題ない
- ✓ Large response:
  レスポンスが大きい場合：
  - ✓ Sending/receiving response are slow
    レスポンスの送信・受信処理が遅い



Sending/receiving response
レスポンスの送受信

- ✓ Serialize/deserialize are slow
  シリアライズ・デシリアライズが遅い
- ✓ How to speed them up?
  どうやって高速化すればよいか
  - ✓ Apache Arrow format
  - ✓ Serialize/deserialize cost ≒ 0
    シリアライズ・デシリアライズコストがほぼ0



Why Apache Arrow format is fast

Apache Arrowフォーマットはなぜ速いのか

Apache Arrowフォーマットは なぜ速いのか

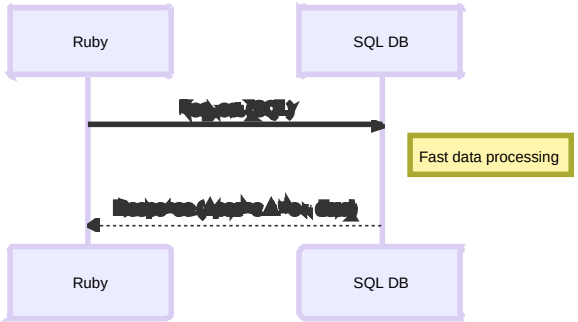
須藤功平

株式会社クリアコード

db tech showcase ONLINE 2020
2020-12-08

https://slide.rabbit-shocker.org/authors/kou/db-tech-showcase-online-2020/

Apache Arrow Flight SQL



- ✓ gRPC based protocol
  gRPCベースのプロトコル
  - ✓ NOTE: Other network libraries such as UCX can be used
    UCXなど他のネットワークライブラリーも使える
- ✓ Specialized to Apache Arrow format
  Apache Arrowフォーマットに特化
  - ✓ Serialize/deserialize cost ≒ 0
    シリアライズ・デシリアライズコストがほぼ0

Red Arrow Flight SQL

```
require "arrow-flight-sql"
location = "grpc://server:2929"
client = ArrowFlight::Client.new(location)
sql_client = ArrowFlightSQL::Client.new(client)
info = sql_client.execute("SELECT * FROM logs")
info.endpoints.each do |endpoint|
  reader = sql_client.do_get(endpoint.ticket)
  reader.read_all
end
```

Which SQL DBs support Apache Arrow Flight SQL?

SQL DB	Support?
MySQL	No
PostgreSQL	No
BigQuery	No
Trino	No
Dremio	Yes

Why don't most SQL DBs support it?
どうしてほとんどのSQL DBはサポートしていないの？

- ✓ Flight SQL is a new protocol
  Apache Arrow Flight SQLは新しいプロトコルだから
  - ✓ The first release: 2022-02（最初のリリース）
  - ✓ Still experimental（まだ実験的扱い）
- ✓ Tradition SQL DBs may not support
  MySQL・PostgreSQLとか昔からあるSQL DBはサポートしないかも
- ✓ New SQL DBs will support because...
  新しいSQL DBはサポートするはず。なぜなら…

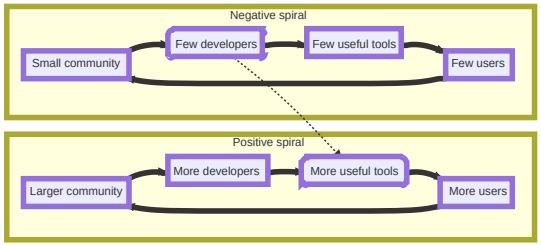
Compatibility is important
互換性が重要だから

- New SQL DBs often use major protocols
  新しいSQL DBは既存のメジャーなプロトコルを使うことが多い
  - To reuse existing client libraries
    ユーザーは既存のクライアントライブラリーで新しいSQL DBを使える
- For example:
  - MySQL protocol: TiDB, ...
  - PostgreSQL protocol: Cloud Spanner, CockroachDB, ...

Future
将来

- ✓ Flight SQL client libraries will be increased
  Flight SQLのクライアントライブラリーが充実するだろう
- ✓ New SQL DBs will support Flight SQL
  新しいSQL DBはFlight SQLをサポートするだろう
  - ✓ To reuse existing client libraries
    既存のクライアントライブラリーを再利用するため
- ✓ BI tools will support Flight SQL by default
  BIツールはデフォルトでFlight SQLをサポートするだろう

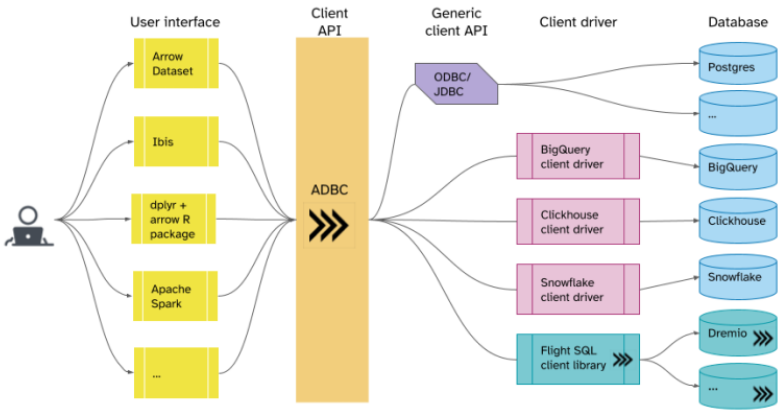
What should we do next?
私たちは次はなにをするべき？



- ✓ Implement an Active Record adapter for Flight SQL
  Flight SQL用のActive Recordアダプターを実装するといいんじゃないかな
  - ✓ For easy to use from Ruby on Rails apps
    Ruby on Railsアプリから使いやすくなるはず
- ✓ Join Red Data Tools!
  Red Data Toolsで開発しようぜ！
  https://red-data-tools.github.io/

But I'm using MySQL/PostgreSQL...

でも、MySQL/PostgreSQLを使っているし。。。

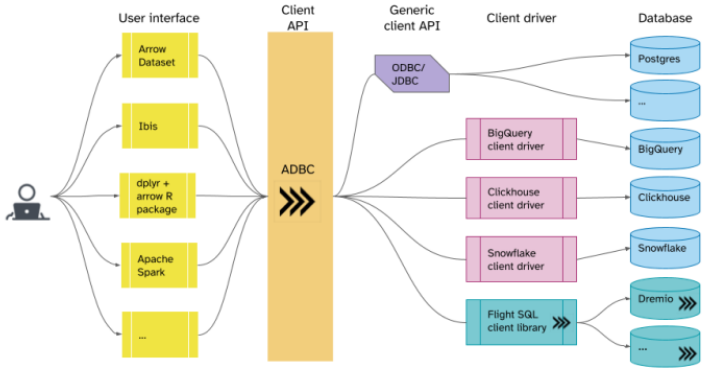


ADBC: Apache Arrow Database Connectivity

https://voltrondata.com/news/simplifying-database-connectivity-with-arrow-flight-sql-and-adbc/

ADBC

- ✓ Generic **fast** SQL DB client API
  任意のSQL DBに接続できる高速なAPI
  - ✓ We can use Flight SQL through ADBC
    ADBC経由でFlight SQLも使える



- ✓ Flight SQL is the most fast driver
  Flight SQLが最速のドライバー
  - ✓ But the same API for all SQL DBs is useful like Active Record for Rubyists
    なんだけど、すべてのSQL DBに同じAPIでアクセスできるのは便利
    Active Recordも便利でしょ？

ADBC and Ruby

- ✓ Implementing Ruby bindings
  Rubyバインディングを実装中
- ✓ Join Red Data Tools
  to implement an Active Record adapter
  for ADBC!
  Red Data ToolsでADBC用のActive Recordアダプターを開発しようぜ！
  https://red-data-tools.github.io/

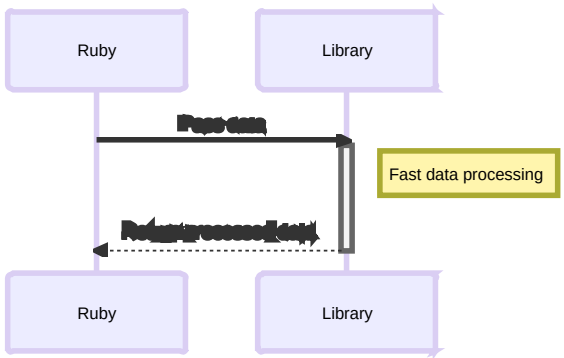
Red ADBC

```ruby
require "adbc"
options = {
  driver: "adbc_driver_sqlite",
  filename: ":memory:",
}
ADBC::Database.open(**options) do |database|
  database.connect do |connection|
    puts(connection.query("SELECT 1"))
  end
end
```

Wrap up: External process case
まとめ：別プロセスの場合

- ✓ Large response is slow
  レスポンスが大きいときに遅い
  - ✓ Bottle neck is serialize/deserialize
    ボトルネックはシリアライズ・デシリアライズ
- ✓ Apache Arrow Flight SQL/ADBC
  そこでApache Arrow Flight SQL/ADBCですよ！
- ✓ Let's implement AR adapters for them
  Active Recordのアダプターを実装しようぜ！

In-process
同一プロセス



- ✓ Not popular case in current Ruby usage
  今のRubyの使われ方だとあまりないケース
  - ✓ Use case: process data on local/remote storage
    IRB, batch process for Web app
    ユースケース：ローカル・リモートストレージにあるデータの処理

- ✓ Need a fast data processing library implemented in other fast language
  高速にデータ処理できる他の速い言語で実装されたライブラリーが必要

Fast language?
速い言語？

- ✓ C/C++
- ✓ Rust
- ✓ Julia
- ✓ ...

A C++ case: Apache Arrow

- ✓ Apache Arrow has a computation module
  Apache Arrowは計算モジュールも提供している
- ✓ Ruby bindings: Red Arrow/red-arrow
  RubyバインディングはRed Arrow/red-arrow
- ✓ Data frame based on Red Arrow
  Red Arrowベースのデータフレームもある
  - ✓ RedAmber/red_amber by @heronshoes
    Red Data Toolsメンバーでもある鈴木さんが開発
    https://mybinder.org/v2/gh/RubyData/docker-stacks/master?filepath=red-amber.ipynb

Red Arrow

```
require "datasets-arrow"
codes = Datasets::PostalCodeJapan.new.to_arrow
require "arrow"
sliced_codes = codes.slice do |slicer|
  slicer.prefecture == "東京都"
end
puts(sliced_codes)
```

Red Amber

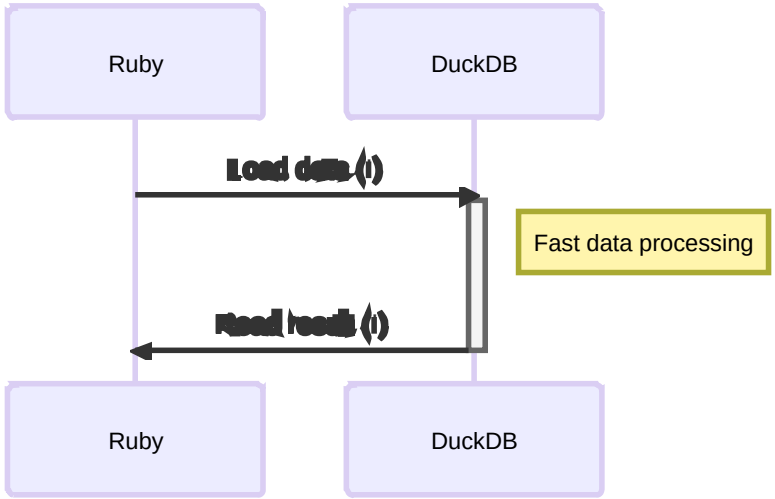
```
require "datasets-arrow"
codes = Datasets::PostalCodeJapan.new.to_arrow
require "red_amber"
data_frame = RedAmber::DataFrame.new(codes)
prefecture = data_frame[:prefecture]
puts(data_frame[prefecture == "東京都"])
```

A C++ case: DuckDB

- ✓ Similar to SQLite
  but for data analytics
  データ分析向けのSQLiteみたいなやつ
  - ✓ Fast aggregation/filter/sort
    高速な集計・フィルター・ソート
- ✓ Ruby bindings: ruby-duckdb by @suketa
  Rubyコミッターでもある助田さんがRubyバインディングを開発
  - ✓ We can impl. fast data processing with DuckDB
    DuckDBを使って高速データ処理を実現できる！
  - ✓ If we can interchange data w/ DuckDB fast...
    DuckDBと高速にデータ交換できればね…

Is fast data interchange important?

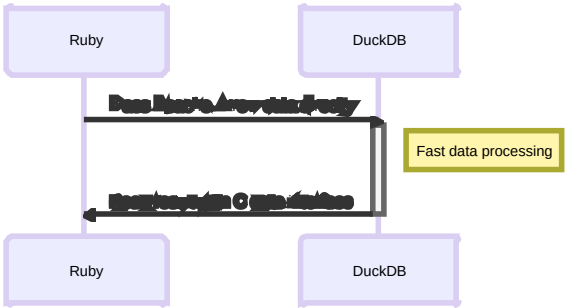
高速データ交換は重要なの？



If (!) are slow, total data processing is also slow

(!)が遅いと全体のデータ処理も遅くなる

Fast data interchange
高速なデータ交換



- ✓ Use data as-is: zero-copy
  データをそのまま使う：ゼロコピー
- ✓ Apache Arrow C data/stream interface
  - ✓ C ABI for fast data interchange
    高速にデータ交換するためのC ABI
  - ✓ FYI: C ABI in Ruby: MemoryView
    参考：RubyもMmeoryViewというC ABIを提供している

C data interface

```c
struct ArrowArray {
  // Array data description
  int64_t length;
  int64_t null_count;
  int64_t offset;
  int64_t n_buffers;
  int64_t n_children;
  const void** buffers;
  struct ArrowArray** children;
  struct ArrowArray* dictionary;
  // Release callback
  void (*release)(struct ArrowArray*);
  // Opaque producer-specific data
  void* private_data;
};
```

DuckDB with Apache Arrow

```ruby
require "datasets-arrow"
codes = Datasets::PostalCodeJapan.new.to_arrow
require "arrow-duckdb"
db = DuckDB::Database.open
c = db.connect
c.register("codes", codes) do # Use Apache Arrow data as-is
  c.query("SELECT * FROM codes WHERE prefecture = ?",
          "東京都", # Tokyo
          output: :arrow) # Output as Apache Arrow data
   .to_table # C data interface
end
```

C data interface on Web

Web上でもC data interface

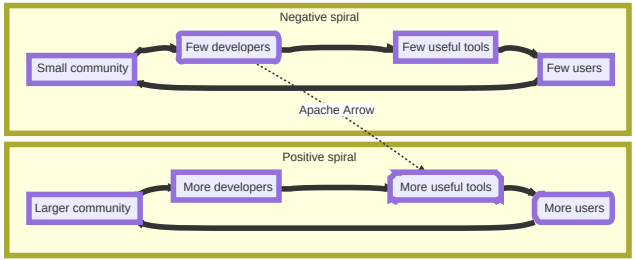
- ✓ Some unofficial WebAssembly ports exist
  非公式ながらいくつかWebAssembly対応のApache Arrowライブラリーがある
  - ✓ Rust based, Go based, ...
- ✓ WASM Ruby + C data I/F is useful?
  WebAssembly版のRubyとC data interfaceでなんかできるかも？
  - ✓ FYI: DuckDB supports WebAssembly too
    参考：DuckDBもWebAssemblyをサポートしている

A Rust case: Arrow DataFusion

- ✓ SQL query engine
  - ✓ Internal memory layout is Apache Arrow
    DuckDBと似ているが内部のメモリーレイアウトはApache Arrow
- ✓ Direct Ruby bindings: (直接のバインディング)
  - ✓ arrow-datafusion by @jychen7 with Magnus
- ✓ Ruby bindings via C API: (C API経由)
  - ✓ datafusion-c with cargo-c
  - ✓ Red DataFusion with datafusion-c

Ruby bindings via C API

C API経由のRubyバインディング



- ✓ To develop with devs from other langs
  - 他言語の開発者と一緒に開発するため
  - ✓ C API is useful for other languages too
    - C APIはJavaやGoなど他の言語のバインディング開発でも有用
- ✓ C API provides a normal C library
  - C APIは普通のCライブラリーを提供する
  - ✓ Headers: Generated by cbindgen automatically
  - ✓ Shared libraries: Built with cargo-c

Red DataFusion

```ruby
require "datasets-arrow"
codes = Datasets::PostalCodeJapan.new.to_arrow
require "datafusion"
context = DataFusion::SessionContext.new
context.register("codes", codes) # C data interface
data_frame = context.sql(<<-SQL)
SELECT * FROM codes WHERE prefecture = '東京都'
SQL
puts(data_frame.to_table) # C data interface
```

Remote data
リモートデータ

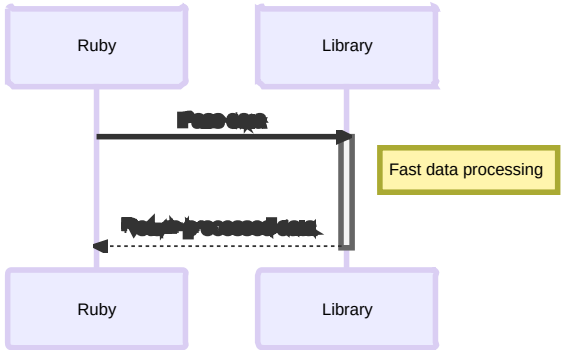
- ✓ Recent modules can read remote data
  最近のモジュールはリモートデータを読み込める
- ✓ At least these modules support it:
  少なくとも次のモジュールはできる
  - ✓ Apache Arrow
  - ✓ DuckDB
  - ✓ Apache Arrow DataFusion

Remote data example: DuckDB

```
SELECT COUNT(*)
  FROM parquet_scan('s3://ookla-open-data/parquet/performance/*/*/*/*.parquet',
                    HIVE_PARTITIONING=1);
--
-- ┌──────────────┐
-- │ count_star() │
-- ├──────────────┤
-- │  144567188   │
-- └──────────────┘
--
```

Wrap up: In-process case
まとめ：同一プロセスの場合

- To process data on local/remote storage
  ローカル・リモートストレージにあるデータの処理
- Low-level APIs (bindings) are preparing
  低レベルのAPIは整備できてきた
- Let's implement high-level API!
  高レベルのAPIを実装しようぜ！
  - e.g.: RedAmber, Active Record adapters, ...
    https://red-data-tools.github.io/



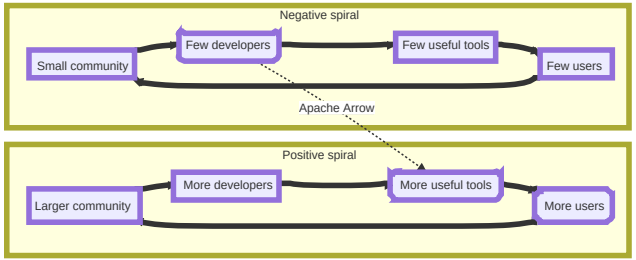
Acknowledgments
謝辞

- Voltron Data
  - Supports my Apache Arrow related work
    私のApache Arrow関係の作業にお金を払ってくれている
  - You can join Voltron Data or ClearCode to work on Apache Arrow as a job
    Voltron Dataさんかクリアコードに転職すると
    仕事でApache Arrowを開発できるよ！
    https://www.clear-code.com/recruitment/
- Red Data Tools members
  - They develop data processing tools for Ruby!
    Ruby用のデータ処理ツールを開発しているよ！

Wrap up
まとめ

- ✓ Can use Ruby for fast data processing with Apache Arrow in some cases
  Apache Arrowを使えば高速なデータ処理にRubyを使える…こともある
  - ✓ External process: Fast data interchange
    別プロセスの場合：高速データ交換機能を使う
  - ✓ In-process: Fast data processing/interchange
    同一プロセスの場合：高速データ処理・交換機能を使う
- ✓ But we still have missing pieces
  でも、まだ足りないところがある

Goal of this talk
このトークのゴール



- ✓ You want to use Ruby for some data processings
  いくつかのデータ処理でRubyを使いたくなる
  - ✓ Especially, you want to implement a BI tool
    特にBIツールを作りたくなる
- ✓ You join Red Data Tools project
  Red Data Toolsプロジェクトに参加する
  - ✓ It provides data processing tools for Ruby
    Ruby用のデータ処理ツールを提供するプロジェクト
    https://red-data-tools.github.io/