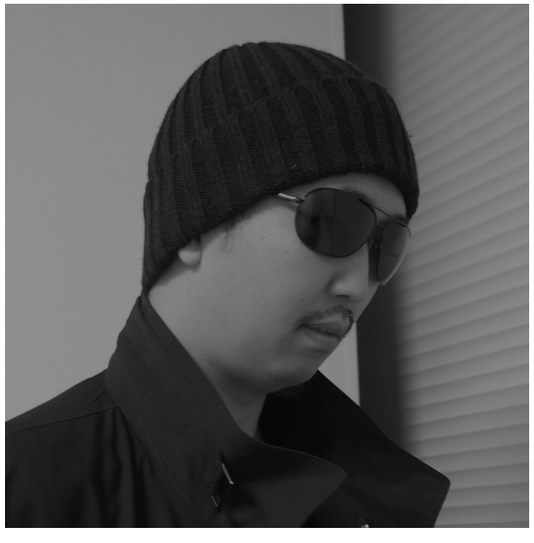
各社のTelemetry機能を触ってみて わかったこと

v1 Jan.26.2018

Atsushi Fujiwara
Global Operations Department.
Rakuten, Inc.

Rakuten

自己紹介



藤原 敦史

所属： 楽天株式会社
　　社内向けプラットフォームサービスの運用チーム

担当業務：
　　以前はネットワークエンジニア
　　現在は社内システムのパフォーマンス監視など

興味があるもの：
　　データ収集、可視化、自動化
　　InfluxDB、Grafana、Fluentd

今回発表するに至った経緯

Telemetry

- NW機器側からメトリクスデータをPUSH送信

- gRPC が用いられる(こともある)
  - HTTP/2 + Protocol Buffers

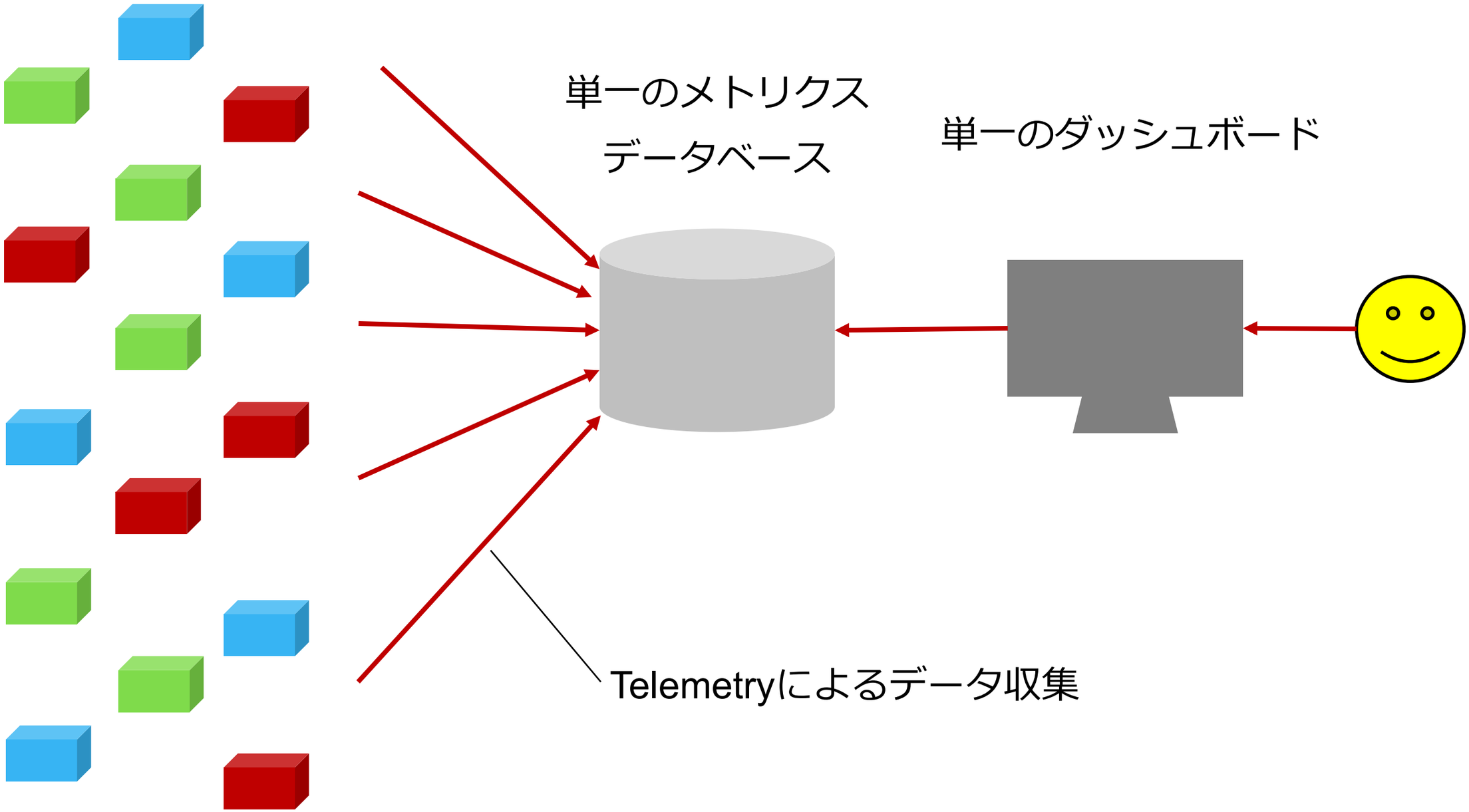
- 各社からOSSとの連携事例が紹介されている
  - Fluentd, Elasticsearch, InfluxDB, Kafka, Kibana, Grafana… など

ネットワークの担当から次期導入機器検証の協力要請を受ける

- 監視(Telemetry)機能部分の検証を共同で行うことに
- ベンダ非依存の監視システムが作れるのではないか？

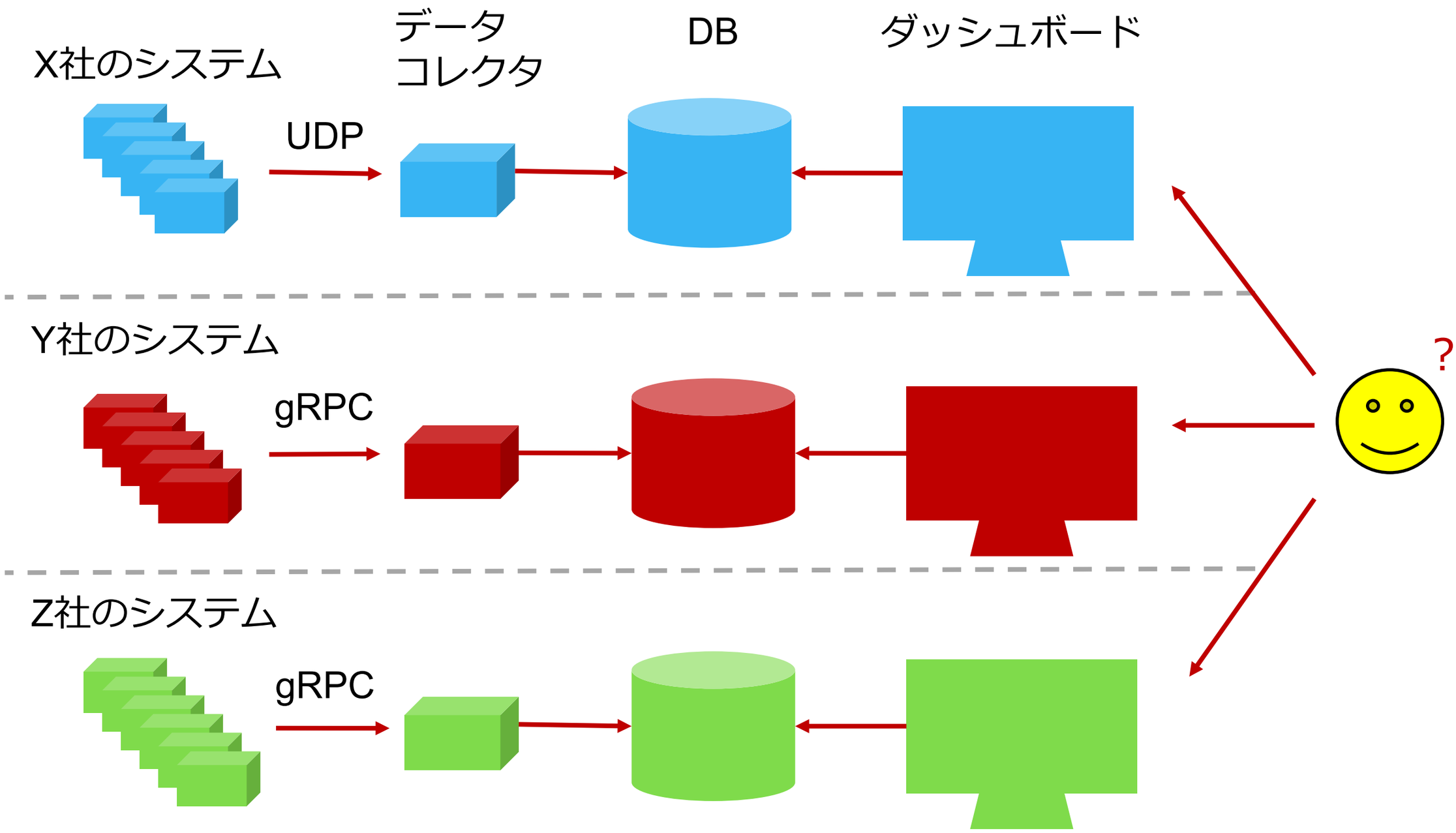
思い描いた理想のシステム

マルチベンダーの機器群

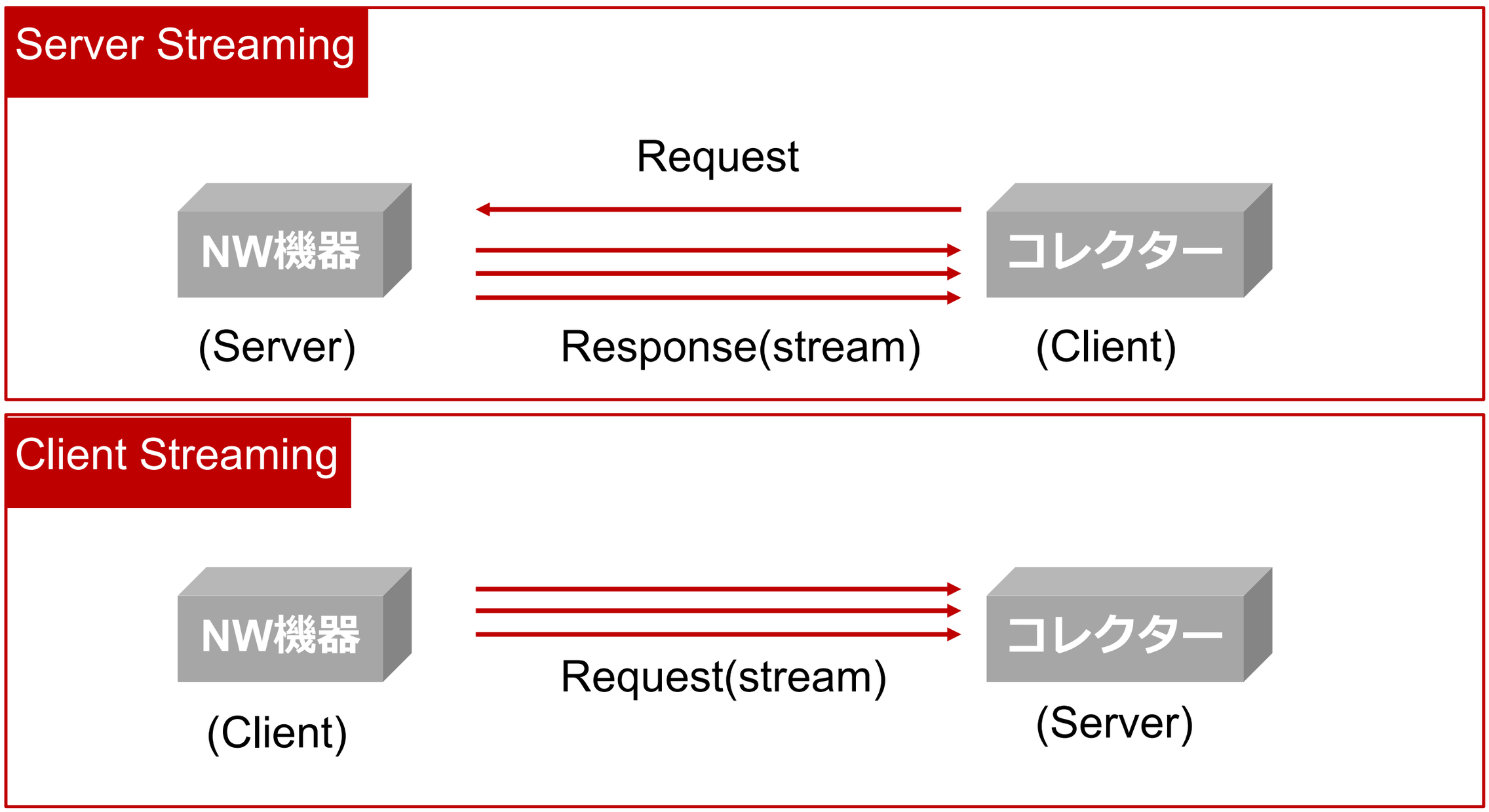


現実はこう

各社のOSSベースのソリューションを実装するとこのような形に



Server Streaming と Client Streaming



最終的にStreamingが行われるのは、どちらもNW機器 → コレクタの方向

サポートされるプロトコルとメッセージフォーマット

メーカー	トランスポート プロトコル	メッセージ フォーマット
X	UDP	Protocol Buffers
Y	HTTP	JSON
	TCP	-
	UDP	Protocol Buffers/JSON
	gRPC	Protocol Buffers
Z	gRPC	Protocol Buffers

gRPC は銀の弾丸ではない

gRPC の使い方

1. .proto ファイルを作成 ← ✅各社から提供

2. .proto ファイルをコンパイル(関数定義のみ)

3. 関数の処理を実装
   - Server の実装 ← ✅ NW機器内に実装済み
   - Client の実装 ← ⚠️ここはユーザが実装する必要がある

4. Server / Clientの実行

各社提供のツールや Docker image を使って簡単に動かす方法もある

- 使い方は限定的

Telemetry をどのように使っていくか

- gRPCは 使えるところだけ 使う
  - ベンダー提供のツールなどを利用できるところのみ

- その他は JSON over HTTP (or TCP)で十分
  - InfluxDB や Elasticsearch に入れるところでHTTPになるため

- PUSH通信 であるのは重要
  - 機器側に設定を入れてClient Streamingにするのが良さそう
  - 機器には多かれ少なかれ設定が必要

- ベンダー間の仕様の標準化には期待したい
  - ソリューション提供より標準インタフェースの実装を


Rakuten