

分散型検索エンジンDroonga

有限会社未来検索ブラジル

Droongaとは何か

- スケーラブルな分散型のデータストア
- リアルタイム&ストリーム型検索エンジン

概要

- 分散型データストア
- レプリケーション&パーティショニング
- メッセージパッシング通信モデル
- 転置索引型 & ストリーム処理型全文検索
- 障害復旧,動的再構築,スキーマ進化

特徴

- 高速・スケーラブルな転置索引型全文検索
- 高速なリアルタイムインデクシング
- 更新イベントに対するストリーム処理型検索
- SPOFのない分散方式による高い可用性
- リアルタイム処理に適した通信モデル
- プラグインアーキテクチャによる高い拡張性

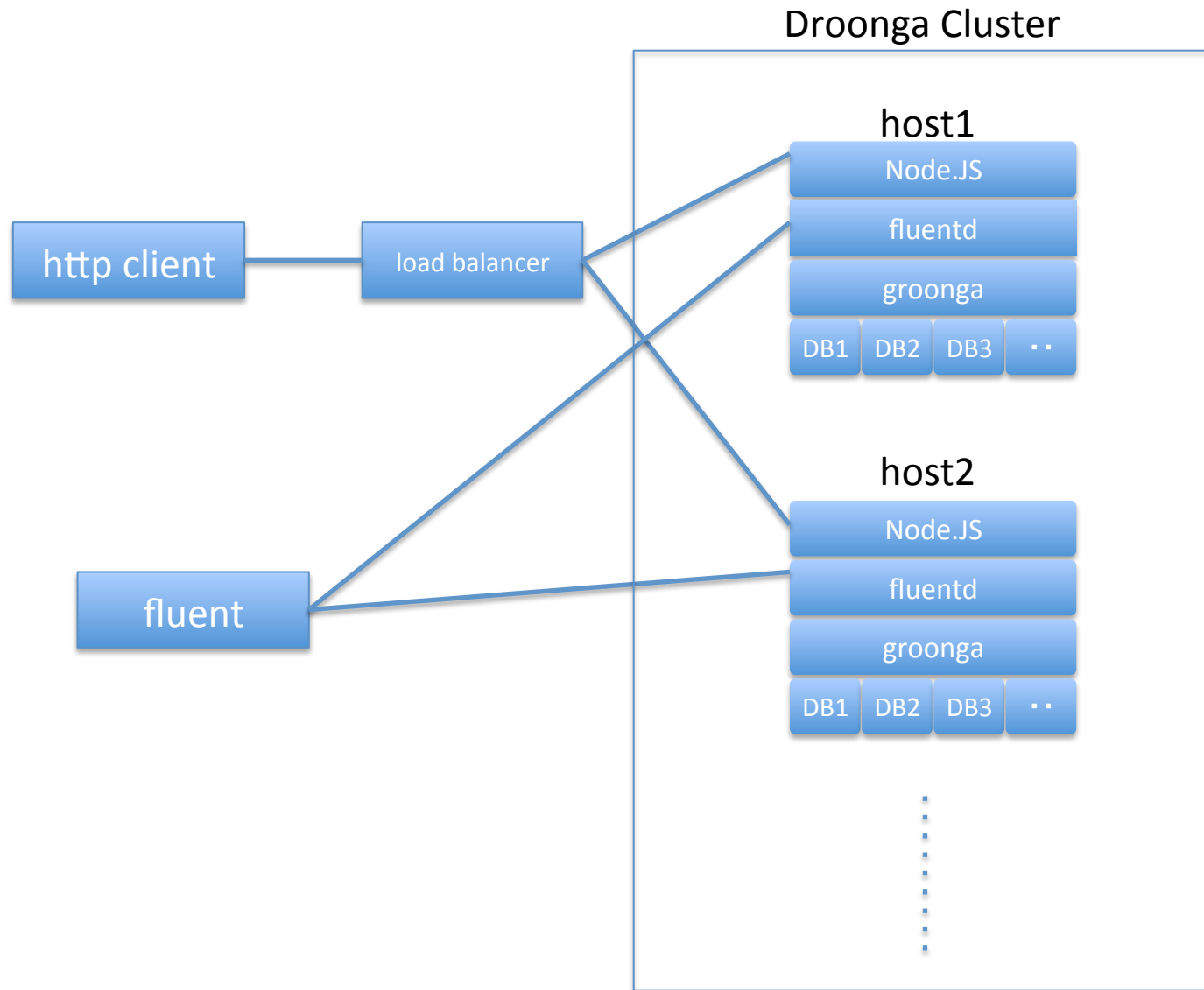
ソフトウェア構成

- groonga(全文検索・カラムストアライブラリ) C
- fluentd(メッセージ通信サーバ) Ruby
- Node.js(プロトコルアダプタ) JavaScript
 - (REST or Socket.IO APIが必要な場合)

システム構成

- 複数のホストに跨がって単一のDBを構築
- 各ホストでは一つ以上のfluentdを動かす
- システムカタログを全てのホストで共有する
- クライアントはどのfluentdに接続してもよい
 - (同じ結果が返される)

システム構成

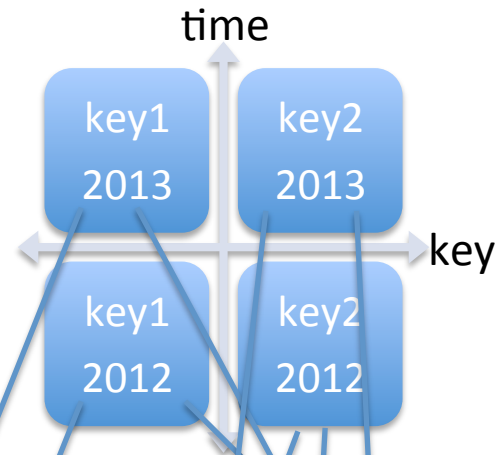


データ分散方式

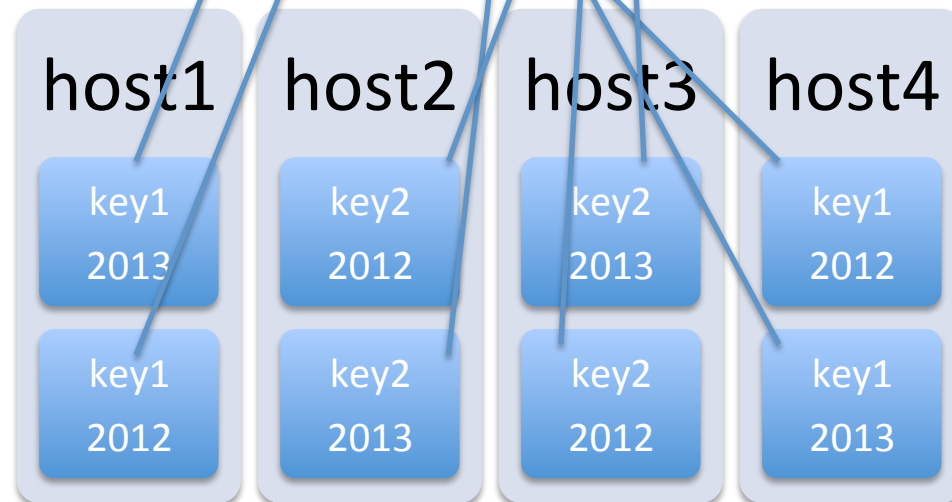
- ハッシュ値&世代によってパーティション化
- データセット毎にレプリケーション数を設定
- 1クラスタ内に以下の要素を任意数定義
 - ゾーン: ホスト資源とそのトポロジー
 - ファーム: ホスト毎のディスク資源
 - データセット: 操作単位となる複数のテーブルの組
- 上記より動的に資源を割り当ててDBを維持

データ分散方式

論理構造



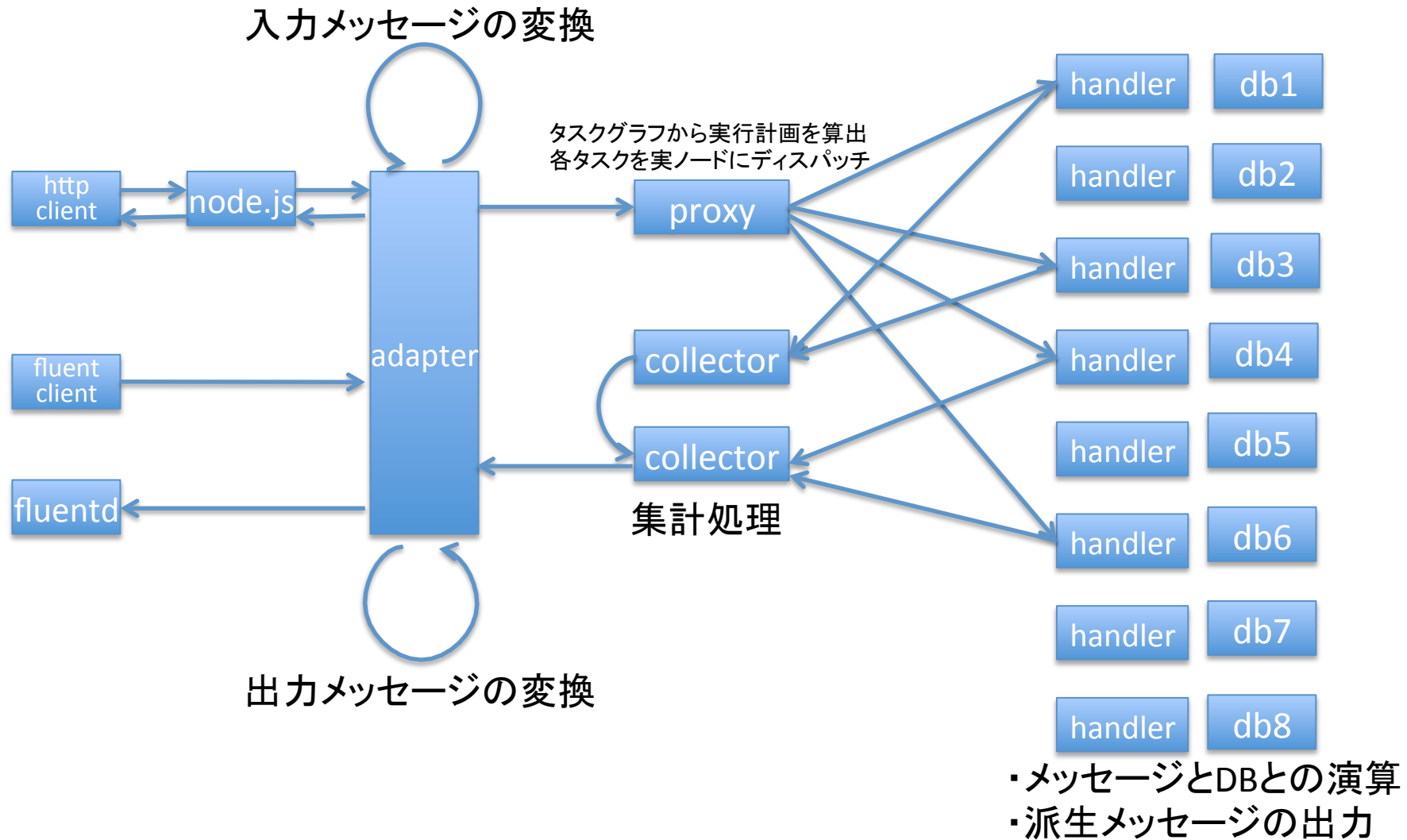
物理構造



通信方式

- 分散処理のための通信は全てfluentdを利用
- データもクエリも全てJSON(msgpack)で表現
- 全ての処理はメッセージパッシングで実現
- 通信パターン(同期・非同期・自律)はAPで選択
- 通常の検索処理はmap/reduceで実現
- 複数のタスクとその依存関係をjsonで定義
- バッチ処理や複雑な分析処理も自由に記述
- プランナが適宜(可能なタスクは並列に)実行

通信処理



アプリケーションインタフェース

- fluentdインタフェース
- Websocket(Socket.IO) API (要Node.JS)
- REST API (要Node.JS, C/S通信のみ)

- 3種類のプラグイン
- adapterプラグイン
 - 入出力メッセージを加工
 - アプリケーションとDroonga間の形式変換
- collectorプラグイン
 - reduce/gather/scatter等の通信処理を拡張
- handlerプラグイン
 - ストアドプロシージャに相当
 - 検索・更新処理の前後に任意の処理を追加
 - 自律メッセージ送出,cache/view更新,ログ集計等々..

リアルタイム・ストリーム型処理



スキーマ進化(予定)

- スキーマレスで任意のJSONデータを投入
- 既存データから推定スキーマ定義を支援
- 検索クエリから索引の要否等を推定
- 新たなスキーマに動的にマイグレーション
- スキーマと矛盾するデータはJSONのまま保持

検索性能

- ごく単純な全文検索クエリによる比較
 - 往復通信がネックとなるパターン
 - ()内はパーティション分割数
- groonga command(1): 2090qps
- groonga http server(1): 360qps
- droonga fluentd interface (1): 1600qps
- droonga REST API (1): 650qps
- droonga fluentd interface (4): 530qps
- droonga REST API (4): 350qps
 - パイプライン通信を行う分droongaが有利
 - 4分割すると集計処理の分だけdroongaが不利