分散 KVS の概要と okuyama の紹介

スクエア free セミナー 第9回

株式会社リンク
ディベロッパーサポート
文屋　宏

自己紹介

○氏名
　文屋　宏（ぶんや　ひろし）

○所属
　株式会社リンク　ディベロッパーサポート

○業務
　プロジェクトマネージャっぽいこと

○興味あること
　分散データベース，クラウドコンピューティング，ウェブ系の技術

○活動
　日本 Red5 ユーザー会メンバー

会社紹介（at+link とは）

株式会社リンクは
at+link の営業窓口

LINK, INC.
株式会社リンク
お問い合わせやお申し込みの受付窓口、営業担当による訪問、広報・宣伝などを担当しています。

ディベロッパーサポート
・開発者のためのサービス開発
・開発者の悩み相談

netforce
株式会社ネットフォース
データセンターに常駐し、ネットワーク管理とオンサイト保守を行っています。

A.T.WORKS
The Next Quality
株式会社エーティーワークス
ハードウェア製造・ソフトウェア開発のほか、サーバ運用に関する技術サポートを提供しています。

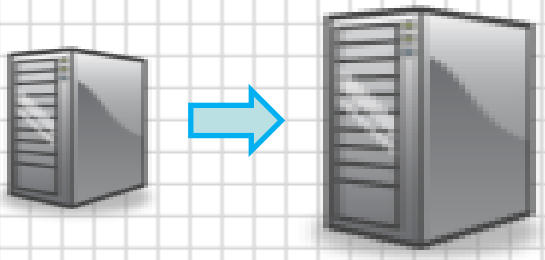
講演内容

- 自己紹介＆会社紹介
- NoSQL について
- KVS について
- memcached のデモ
- 様々な KVS
- okuyama について

分散処理 の必要性

サービスが成長してきたとき、どうする？

スケールアップ



- ハードウェア的な限界がある
- ネットワーク集中
- コストパフォーマンスが悪い
- 障害に弱い

or

スケールアウト

- ハードウェア的な限界がない
- ネットワーク分散
- コストパフォーマンスが高い
- 障害に強い（高可用性）

Web サーバだけでなく、DB サーバも分散させたい！

データベースソフトウェアの分類

RDBMS

・Oracle
・MySQL
・PostgreSQL
etc.

NoSQL

・カラム指向型
・ドキュメント指向型
・**キー・バリュー型**
etc.

一貫性重視
スケールアップ

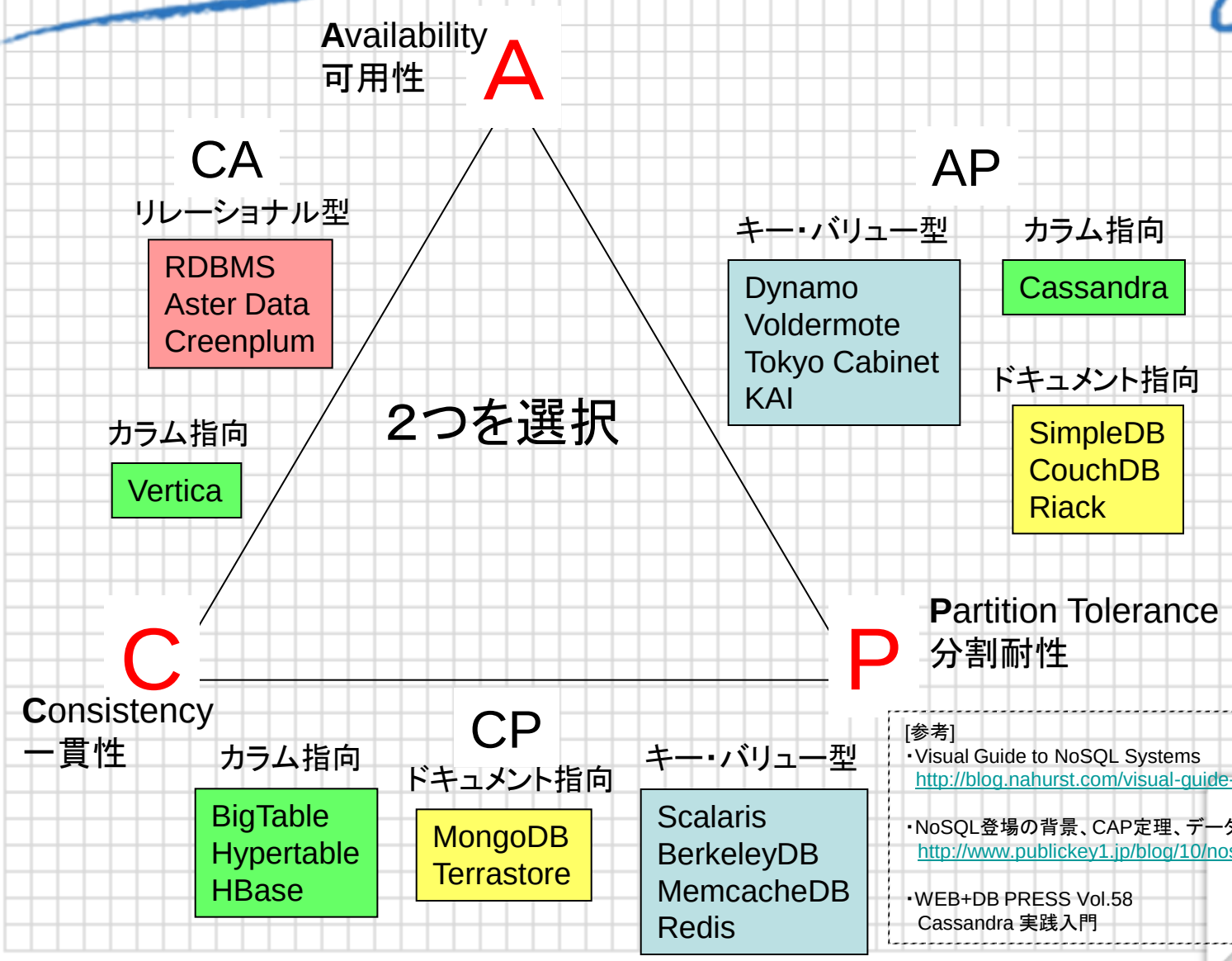
パフォーマンス重視
スケールアウト

KVS は NoSQL の一種
NoSQL = Not Only SQL

RDBMS と NoSQL は互いに補完し合う存在
どちらが優れている、ということはない

どんなシステムでも、何かを
犠牲にしなきゃいけない・・・

CAP 定理による NoSQL の分類



[参考]

・Visual Guide to NoSQL Systems
http://blog.nahurst.com/visual-guide-to-nosql-systems

・NoSQL登場の背景、CAP定理、データモデルの分類
http://www.publickey1.jp/blog/10/nosqlcap.html

・WEB+DB PRESS Vol.58
Cassandra 実践入門

Key-Value Store

Key-Value Store とは

・[Key, Value] のセットでデータ保存
・Key を指定して Value を取得
・処理を簡単にしてパフォーマンス向上

用途

・キャッシュ
・パフォーマンス重視のデータベース
・分散ファイルストレージ
・解析用のログ保存

KVS のデモ

キャッシュの代表 memcached
をさわってみよう

デモをご覧ください

memcached だけだと・・・

キャッシュサーバはすごく大事
でも・・・

〇あくまでもキャッシュに特化

⇒永続型 KVS も必要

〇分散多重保存できない

⇒分散型 KVS が必要

KVS には得意・不得意が

- **分散多重保存が得意**

  Flare, kumofs

- **データ永続化が得意**

  Tokyo Tyrant, Flare, kumofs

- **データの一貫性を保つのが得意**

  memcached, Tokyo Tyrant

「何でも得意」な KVS は無い！

分散アルゴリズム

データをどのサーバに保存するか

ハッシュ法（Mod 法）

MD5などでキーからハッシュ値を計算

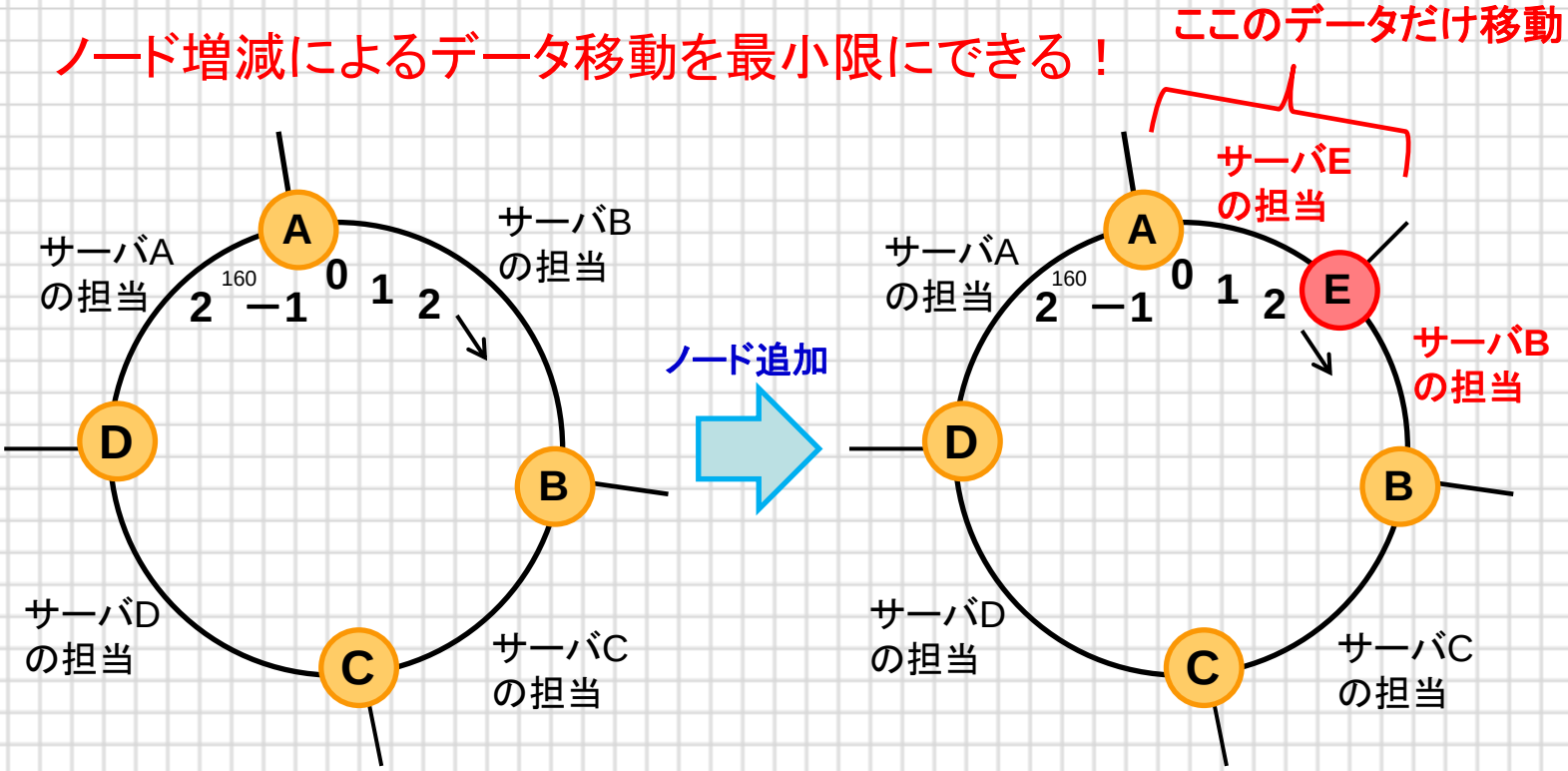
→得られたハッシュ値をサーバー台数で割った余り

サーバ追加時にデータ移動が大量発生

コンシステントハッシュ法

コンシステントハッシング

ノード増減によるデータ移動を最小限にできる！



- ノードやキーを仮想リング上に配置
- ノードの IP や、Key 値をハッシュ関数に通す
- それぞれ、リング上の該当位置に配置
- 時計回りで最初に遭遇するノードが担当

国産 KVS

国内だけでも、こんなにたくさんの開発者が

分散型

- ○Tokyo Tyrant　平林幹雄 氏（当時 mixi）
- ○kumofs　古橋貞之 氏（筑波大）
- ○Flare　藤本真樹 氏（GREE）
- ○ROMA　西澤無我 氏（楽天）
- ○okuyama　岩瀬高博 氏（神戸デジタル・ラボ）

素晴らしい！！

国産 KVS

分散型

- ○Tokyo Tyrant　平林幹雄 氏（当時 mixi）
- ○kumofs　古橋貞之 氏（筑波大）
- ○Flare　藤本真樹 氏（GREE）
- ○ROMA　西澤無我 氏（楽天）
- ○okuyama　岩瀬高博 氏（神戸デジタル・ラボ）

at+link でサービス化！

分散型 KVS「okuyama」

okuyama の特徴

■分散多重保存できるか

⇒ 得意！！

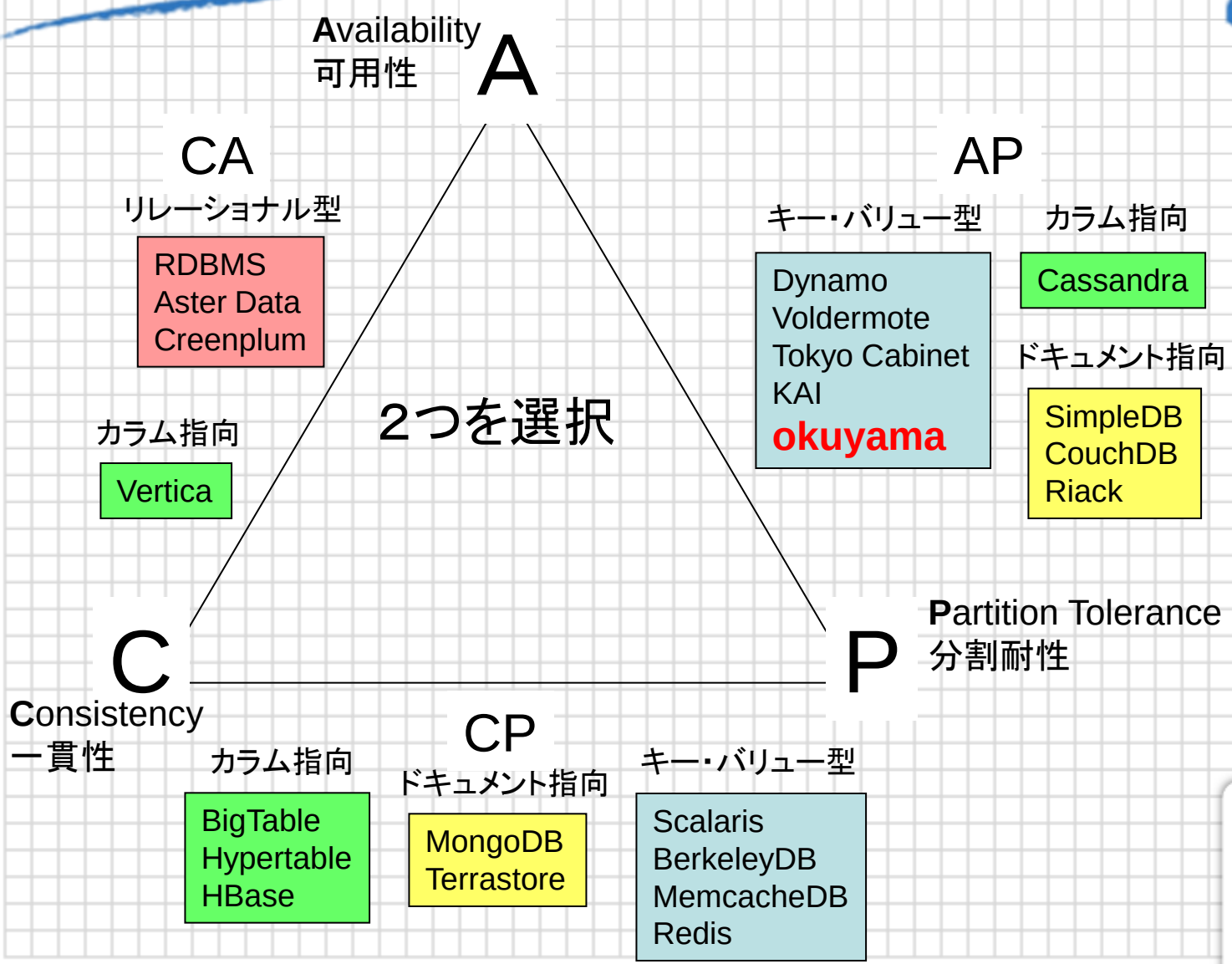
■データを永続化できるか

⇒ 4段階レベルを選択可能！！

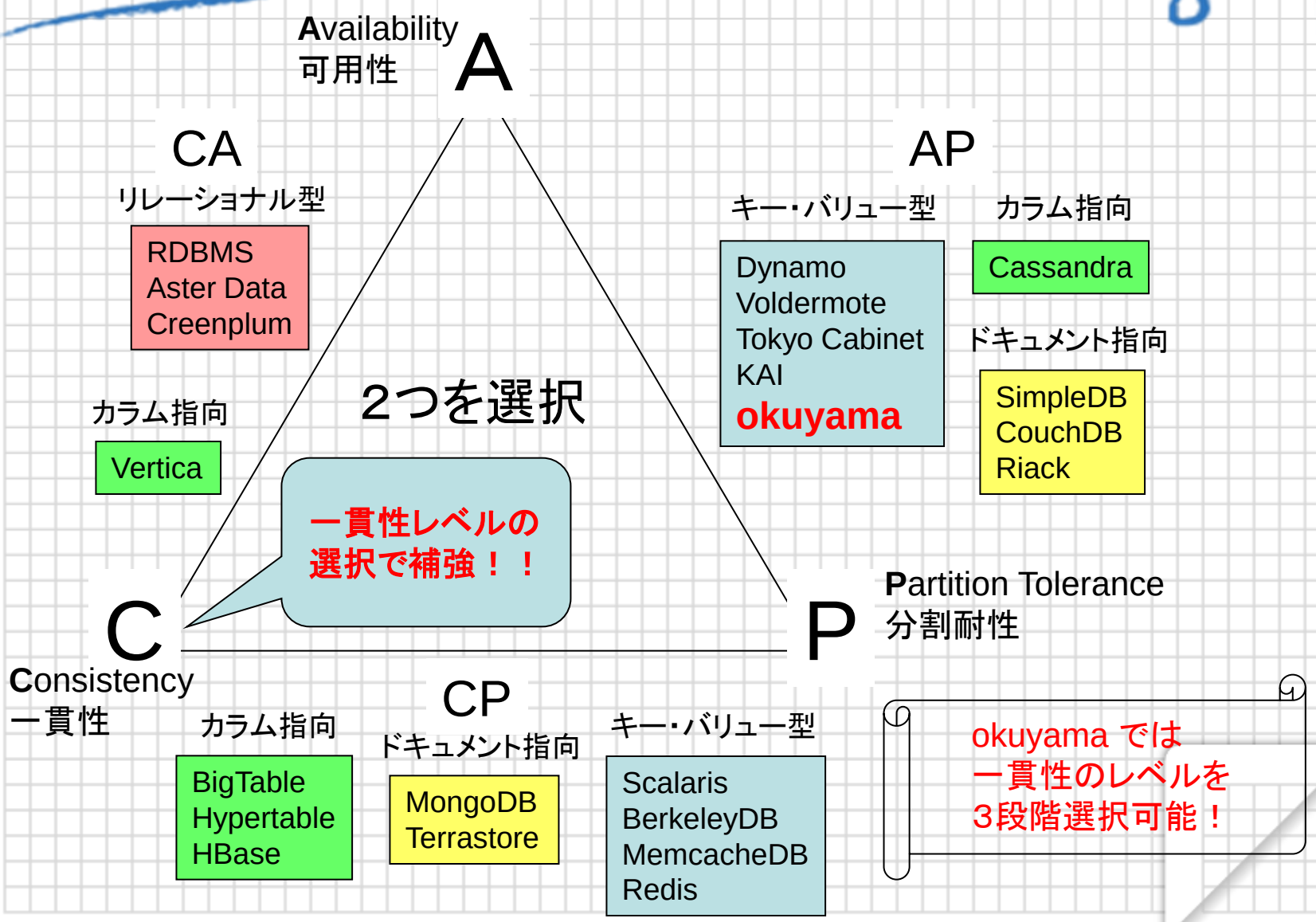
■データの一貫性を保てるか

⇒ 3段階レベルを選択可能！！

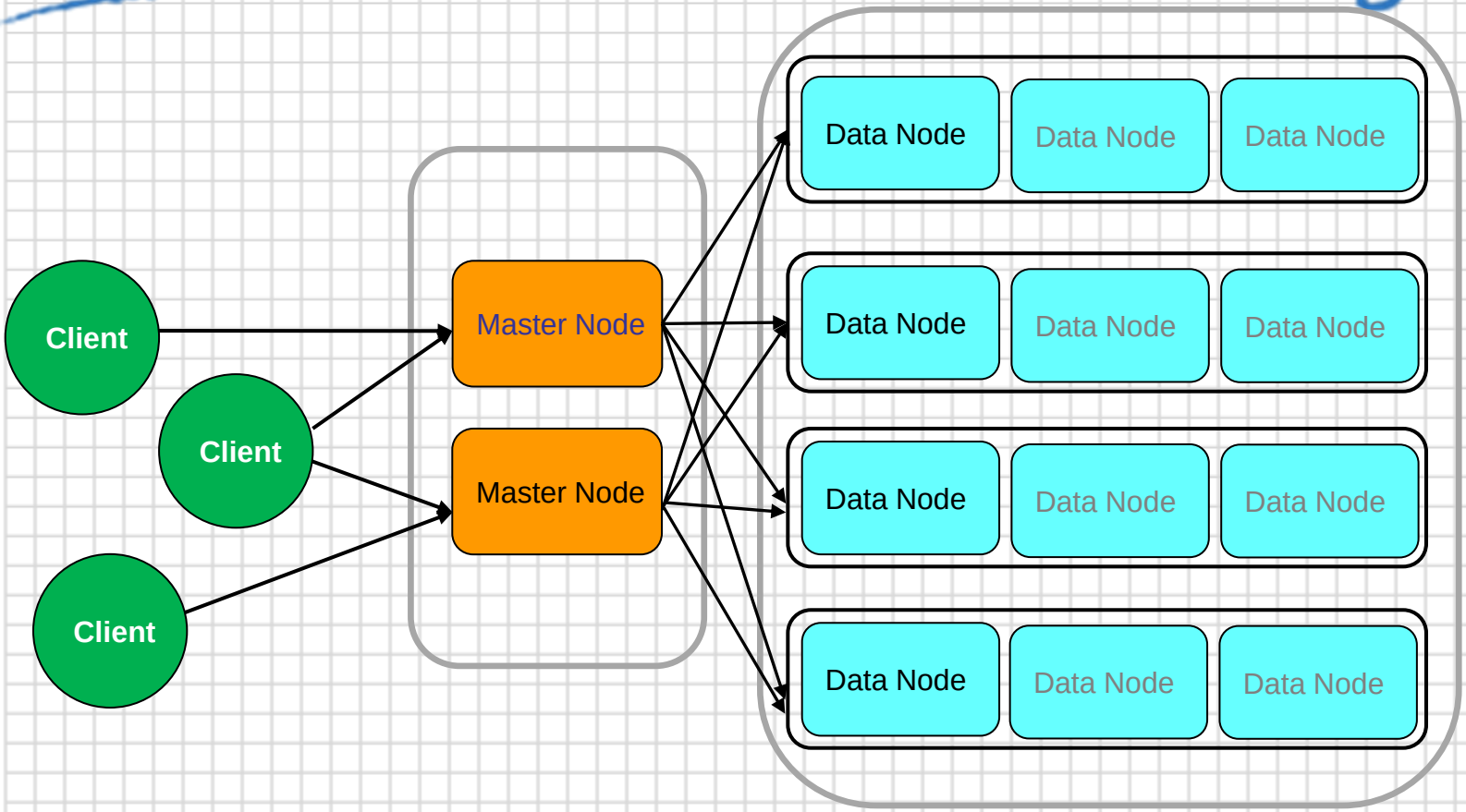
okuyama って、CAP 定理だとどこ？



okuyama って、CAP 定理だとどこ？



okuyama の構成イメージ



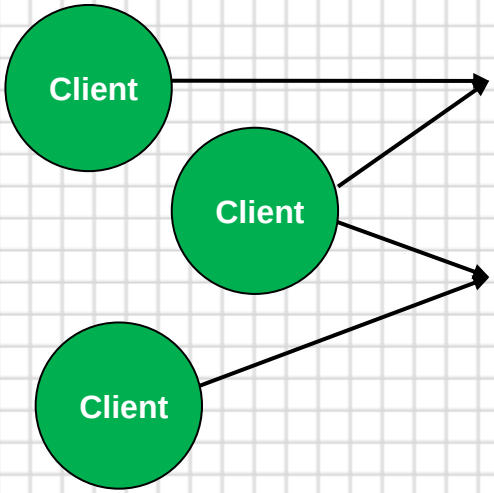
Client → Master Node → Data Node(×3)

（以降 okuyama 関連資料
神戸デジタル・ラボ 岩瀬高博 氏　提供）

okuyama クライアント

okuyamaへの問い合わせを実現

専用クライアントはJavaと、PHPを実装



okuyama マスターノード

・クライアントからのI/F

・サポートプロトコル

>オリジナル

>Memcached

・set ・get ・add

・delete

・gets ・cas

>HTTP

・GET

・データノード管理

>データ入出力

サポート分散アルゴリズム

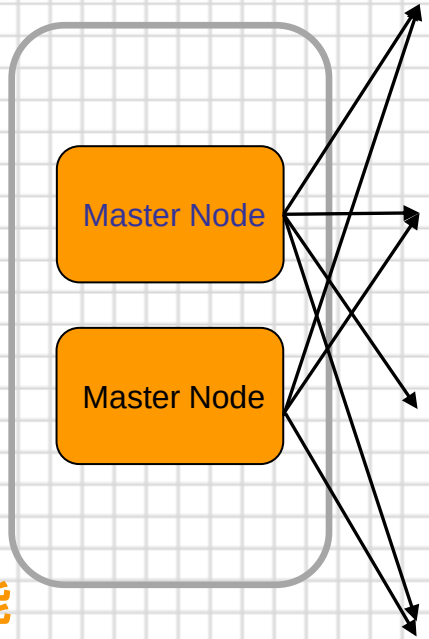
1. Mod

2. ConsistentHash

>生存監視

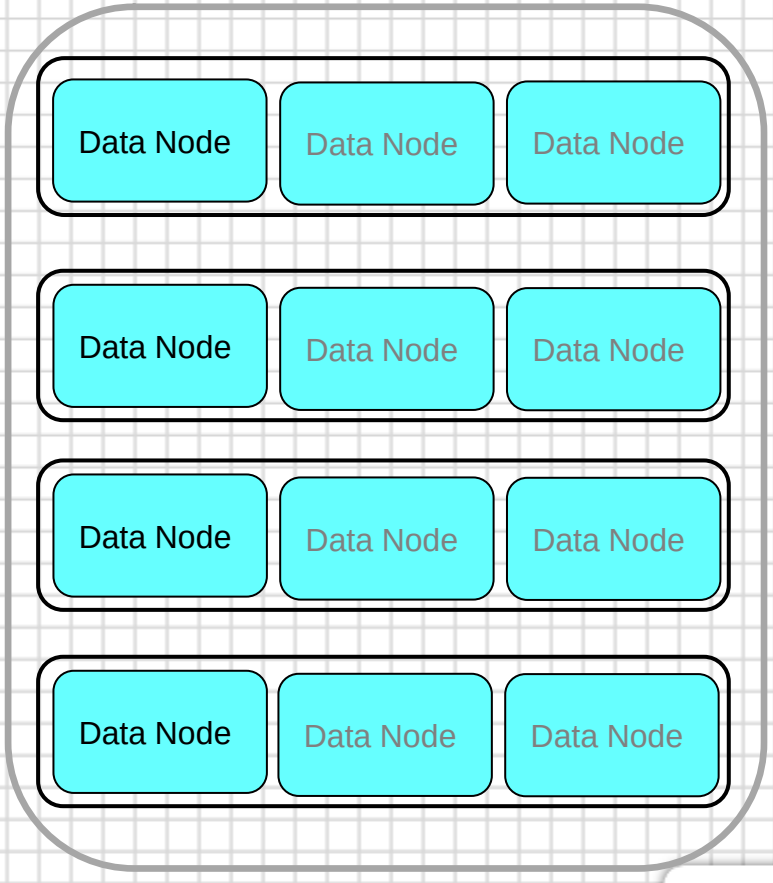
起動時のデータリカバリ

・制限台数なしに冗長化可能



okuyama データノード

- データの保存を実現
- データ保存方式を選択可能
- 最大3ノードにデータを保存
- アクセスバランシング
- 連鎖的ダウンの予防



データ保存方式の選択

1.全てのデータをメモリに保存

>非永続型（key=メモリ、value=メモリ）

2.データ操作履歴のみファイルに保存

>永続型（key=メモリ、value=メモリ）＋操作記録ファイル

3.データ本体をファイルに保存

>永続型（key=メモリ、value=ファイル）＋操作記録ファイル

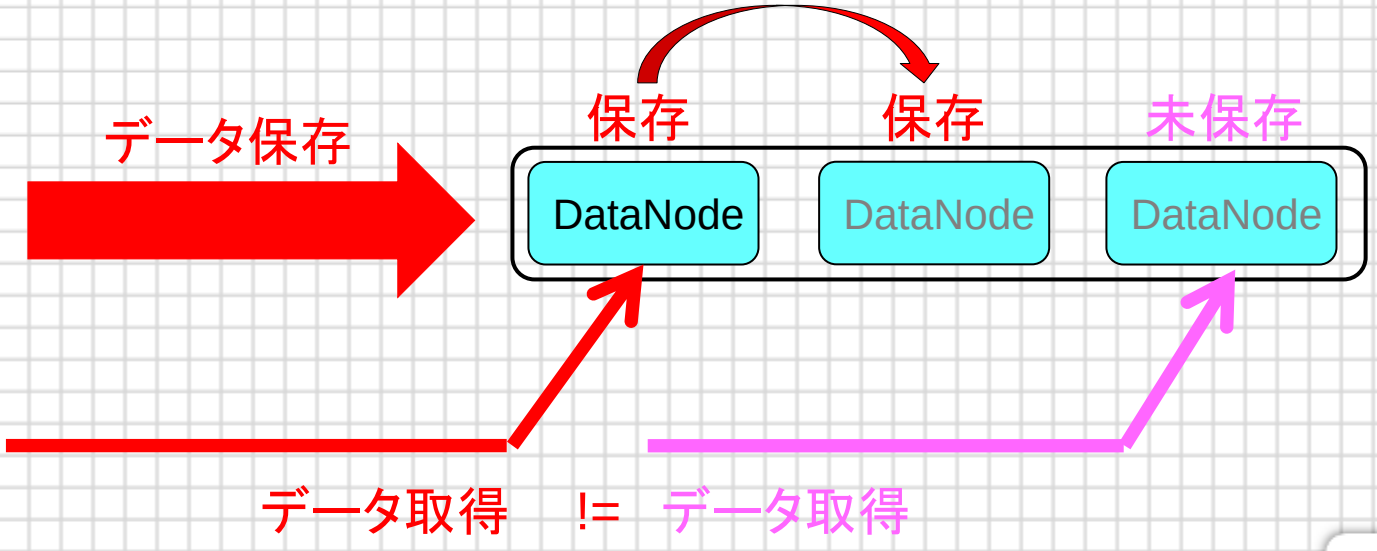
4.全てをファイルに保存

>永続型（key=ファイル、value=ファイル）＋操作記録ファイル

データの一貫性について

複数のノードに同一の値を保持していると、

データに異なる時間が発生する



データー貫性レベルの選択

1.弱ー貫性

>全てのデータノードにランダムにアクセス

2.中ー貫性

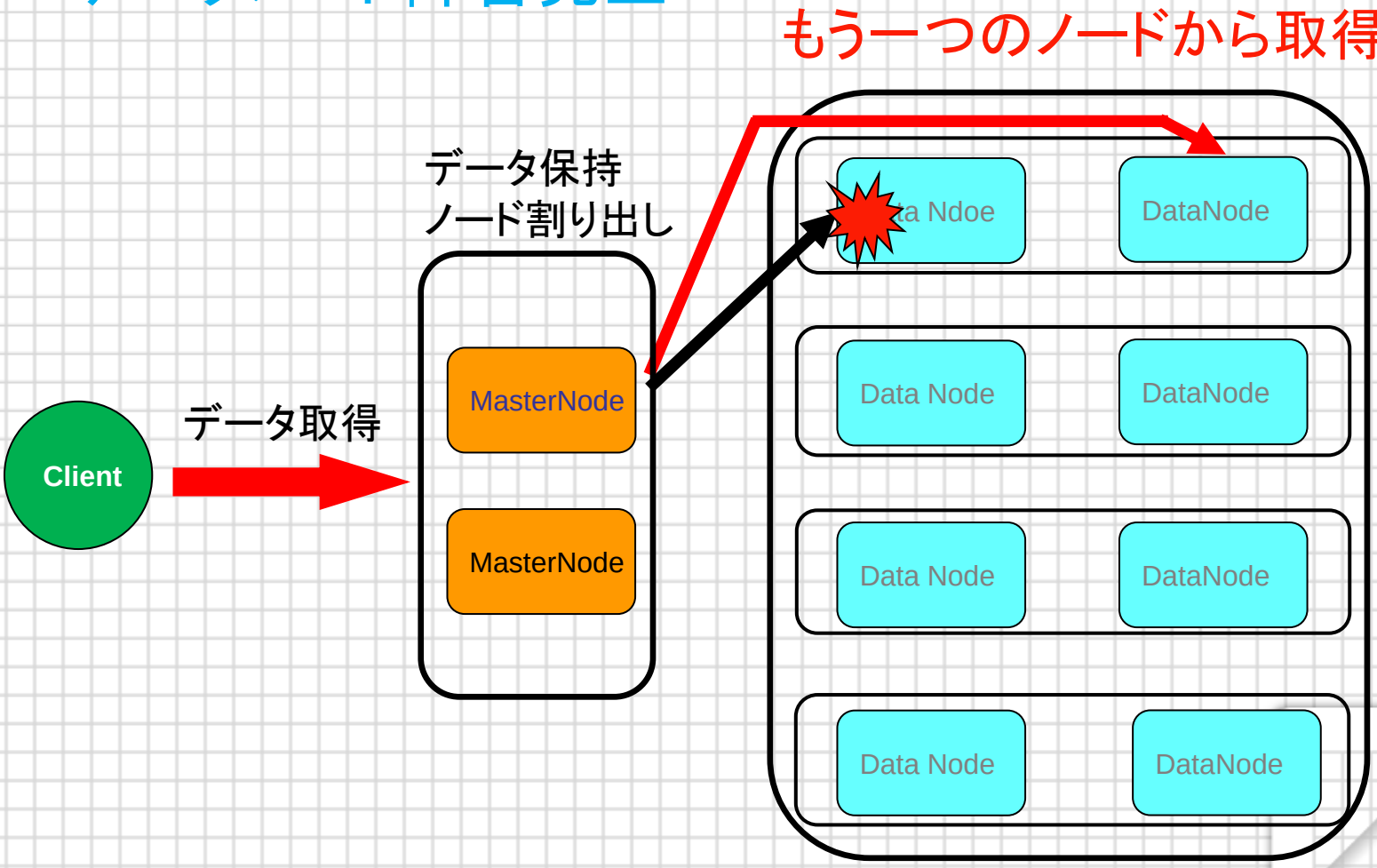
>常に最後に保存したデータノードからアクセス

3.強ー貫性

>データノードの値を検証

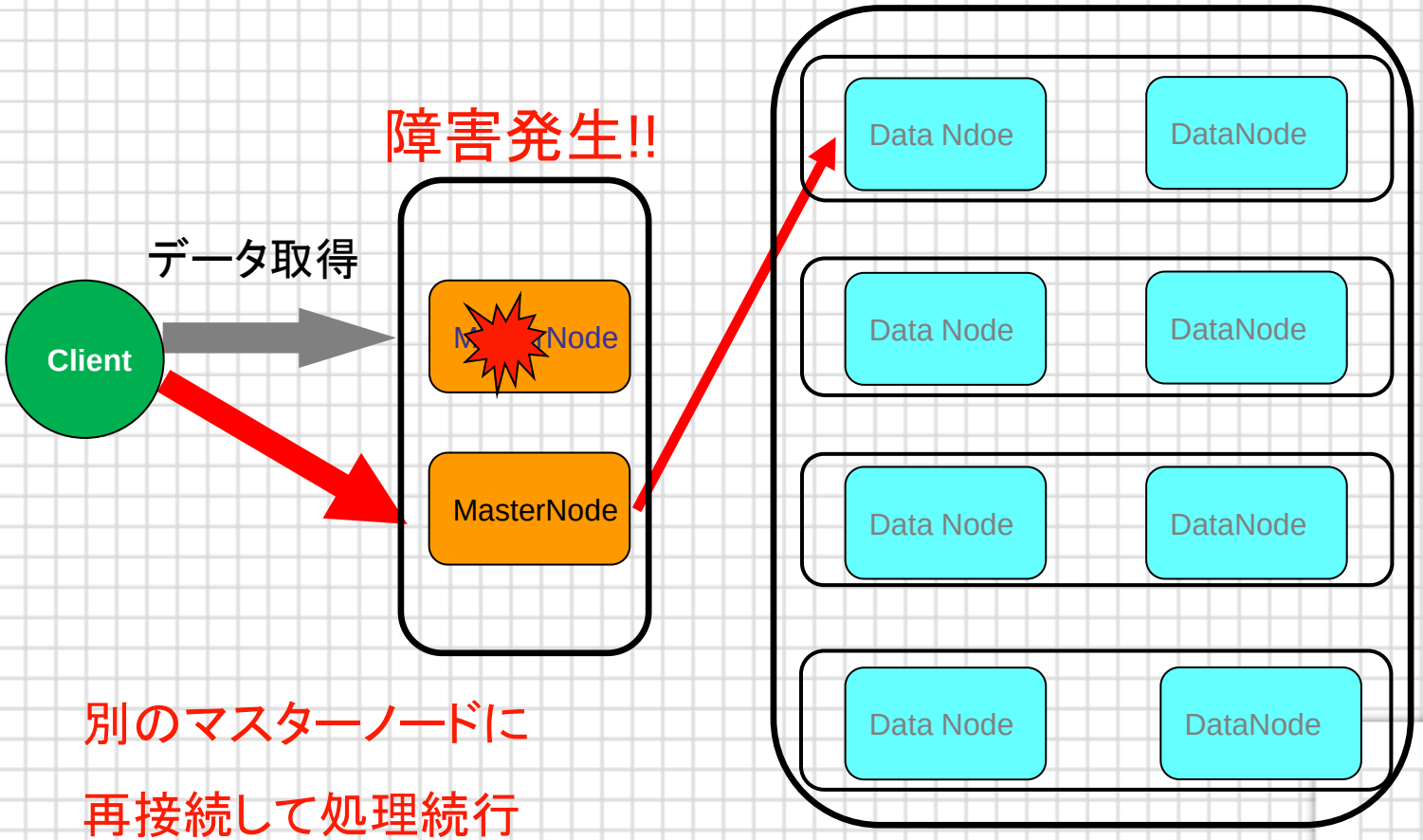
okuyama に単一障害点は無い！

データノード障害発生



okuyama に単一障害点は無い！

マスターノード障害発生



別のマスターノードに

再接続して処理続行

詳しくは、岩瀬氏の資料をご覧ください

〇Think IT 連載記事
http://thinkit.co.jp/story/2011/02/03/1990

〇講演資料
http://www.slideshare.net/okuyamaoo/20110517-okuyama-8035077

まとめ

・拡張性、可用性の面から KVS（NoSQL）

・KVS（NoSQL）も色々

・得意・不得意がある

・用途によって使い分けよう

・国内にも素晴らしい技術者がいっぱい

・okuyama オススメ

ご清聴ありがとうございました

では、KVS サービス化の話へ