

今話題のHadoop HBaseの 性能検証結果と Zabbixによる性能監視のご紹介



日本ヒューレット・パッカード株式会社
テクノロジーコンサルティング統括本部
データセンターソリューション第一本部コアテクノロジー部
石田精一郎

お話ししたい内容

- インフラの観点からのHBase
 - どのように信頼性が確保されているのか
 - スケールアウトやI/Oのアーキテクチャ
 - 性能監視のポイント

- 検証の観点とハマリどころ
 - プロダクトの基本的な特徴、性能特性の確認
 - 検証ノウハウの共有



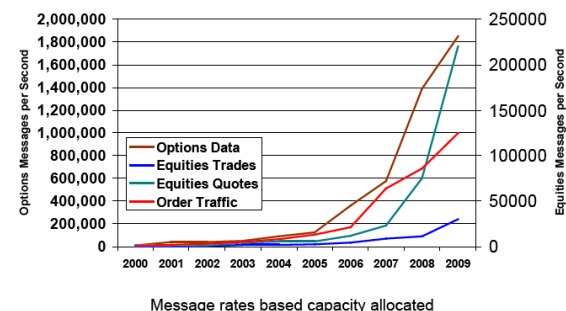
Hadoop概要



ビッグデータ対応における現在のシステムでの問題点

RDBMSの限界と非定形処理の増加

非定形処理が必要なデータの爆発的増加に対して、RDBMSを利用した従来の集中処理型のアーキテクチャは限界を迎えている。



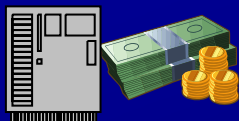
10 | NYSE Euronext



•非定形処理データの増大



•大量の処理要求によるシステム負荷の増大



•性能を上げるために高コストなハードウェアが必要

•スケールアップ型システムの限界



•ストレージの限界

•十分な保存領域を確保出来ない



Hadoop開発の歴史的経緯

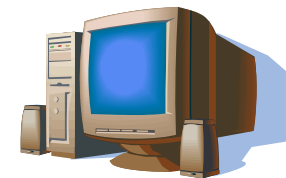
GoogleFileSystemやMapReduceなどと呼ばれる分散処理技術を独自に開発。
ソースコードではなく、その仕組みを論文として発表（2004年）

膨大なデータに対する検索処理で課題を抱えていたYahoo Inc.は、著名なエンジニアであるDoug Cutting氏を中心に上記論文を元に、Hadoopを開発。
Apache Hadoopプロジェクトとして公開され、オープンソースソフトウェアとして開発が進む。

米国を中心に利用が進み、
Hadoop自体も様々な改良
が加えられる。

日本でも導入事例が
増えつつある。

Hadoopシステム構成



クライアント/データソース



NameNode/JobTracker

役割：メタデータの保持、Data Nodeの状態管理
分散処理 (MapReduce) ジョブの管理

データの読み書き
処理の依頼

Hadoop クラスタ

MapReduce : 複数のサーバで分散処理することで高性能を実現

HDFS (Hadoop Distributed File System)

ファイルを分割して複数のサーバに複製して保持することで冗長性を担保



DataNode/TaskTracker

データの保持と分散処理を、スケールアウト構成で実現

HBase概要



HBaseとは

「HDFS上に構築された列指向データベース」

冗長化、永続性、データ一貫性、スケールアウト、複雑なデータの操作を実現したBigData用の高機能KVS (Key Value Store)

– 特徴

- データをHadoop 分散ファイルシステム上に保存することで冗長化と永続性を実現
- ノードを追加することでリニアにスケールアウト
- 行単位のロック操作でデータ一貫性を保証
- 複雑なデータを柔軟に操作するために、カラム・ファミリーを採用



HBaseの開発概要

HBaseはHDFSの欠点を埋めるための低レンテンシデータストアとして開発されました

開発概要

- Apacheソフトウェア財団のトッププロジェクト
- HBase開発の動機 — Hadoop HDFSとRDBMSの欠点を補完するデータストア
 - HDFSは、スケールアウト可能な分散データストアだが、大きなデータを一括で読み書きする処理に特化している。
 - RDBMSは、SQLでの高度なデータ処理とトランザクションをサポートしているがスケールアウトに限界がある。
 - 一方でデータ量は増加の一途をたどっており、それに対応したランダムライト／リードに対応したデータストアが必要。
- Google Bigtableを参考に設計された
- Facebookがメッセージング基盤に採用し、世界的に注目が高まっている。



HBaseとRDBMSの比較

HBaseとRDBMSの特性の違い(※1)

	RDBMS	HBase
データ構造 (※2)	行指向が多い (※3)	列指向
トランザクション	可能	行ロックのみ
データ操作	SQL (JDBC等で接続)	get/put/scan (※4) (独自APIで接続)
インデックス	任意の列	Row keyのみ
最大データサイズ	TBs	PB+
最大スループット	数1000クエリ/s	数100万クエリ/s

※1 Cloudera HBaseトレーニング資料を参考に作成。比較対象のHBaseのバージョンは「0.90」とした。

※2 「行指向」「列指向」は、データを扱う単位の違いを表わす。行指向データベースは、行ごとにデータを取り出すことに最適化されているが、列指向データベースは行をまたいで列ごとにデータを取り出すことに最適化されている。

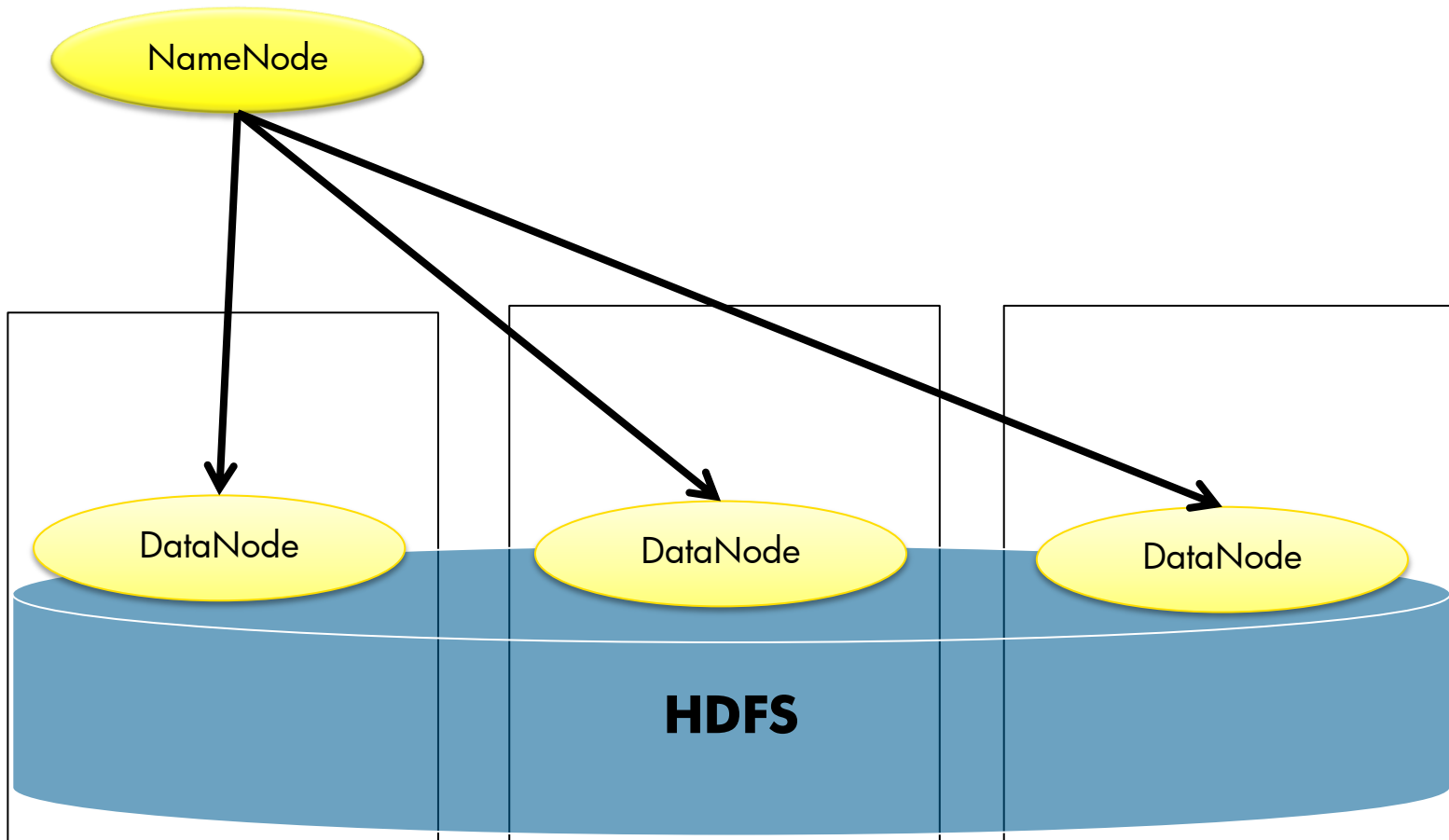
※3 たとえば、HP Verticalは列指向RDBMSで、列ごとの高速な読み出しに最適化されている。

※4 scanは、HBase独自の一定のキー範囲での読み込み(例: Key



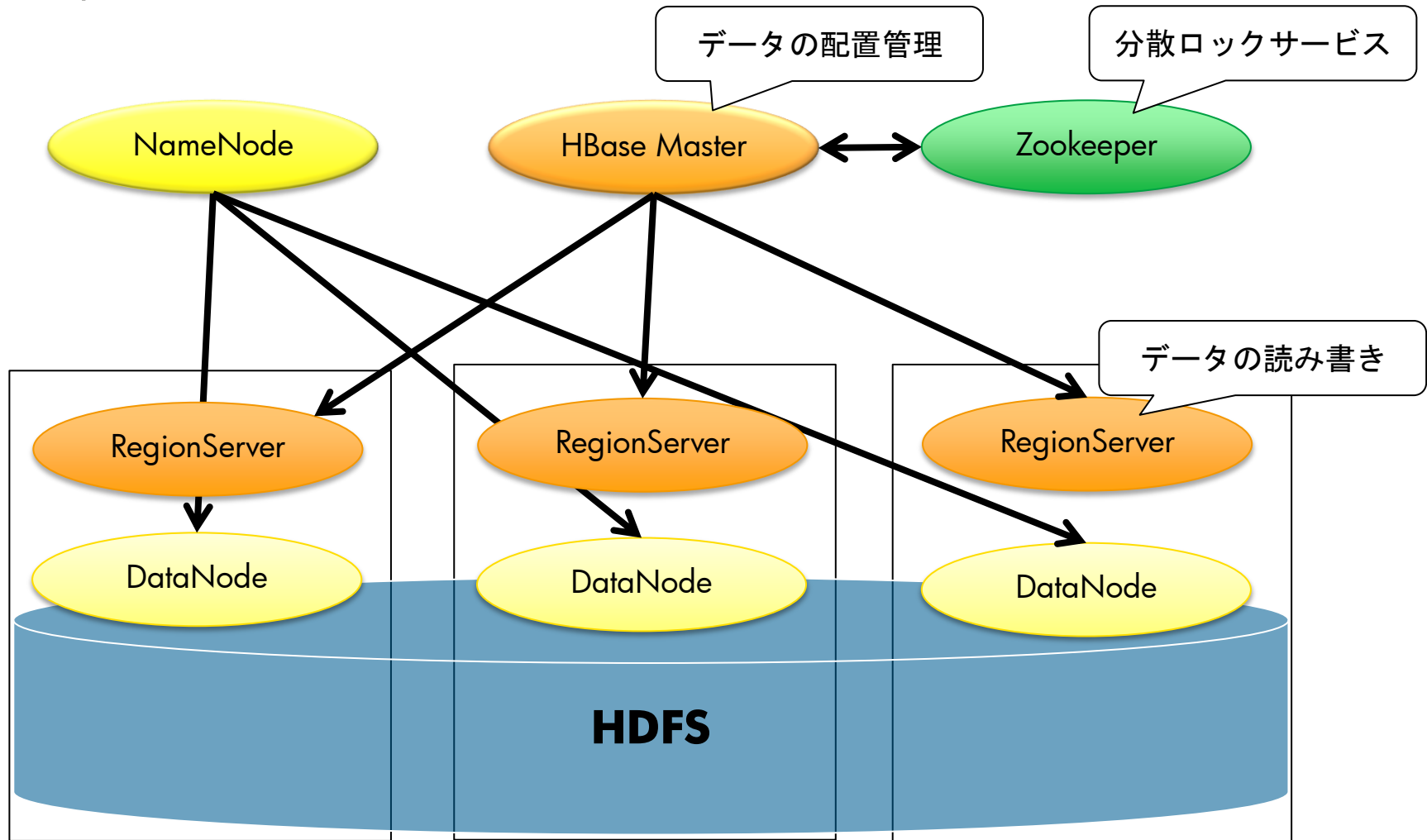
HBaseの基本構成

Hadoopの基本構成にHBase関連のサービスを追加



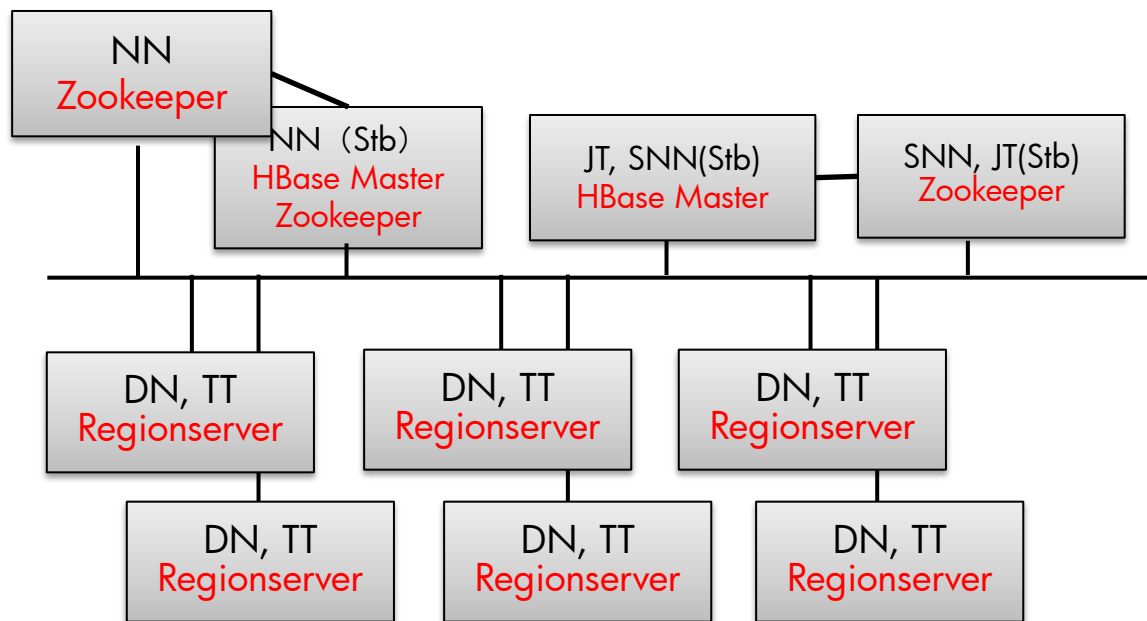
HBaseの基本構成

Hadoopの基本構成にHBase関連のサービスを追加



HBaseの基本構成例

HBaseの基本構成例



- Hadoopの基本構成にHBase関連のサービスを追加する。
- HBase Masterは2台のAct-Actで冗長化。
- ZookeeperはHBaseクラスタ管理を行うサービス。クォーラムを確保するため、ノードは3台以上の奇数で構成。
- HBaseのデータを保存するRegionserverは、Datanodeと同居させる。
- スレーブノードは最低3台から。

【略記・凡例】

Namenode: NN

Secondary Namenode: SNN

Jobtracker: JT

Datanode: DN

Tasktracker: TT

※赤字がHBase関連サービス



HBaseアーキテクチャ



テーブル表現方式

- ユニークなKeyで指定される行 (Row) を集めたテーブルとして管理
- 各行は、複数のカラムファミリーを持つ
- カラムファミリーには、複数の(label, data) の組や単一のデータなどを保持可能

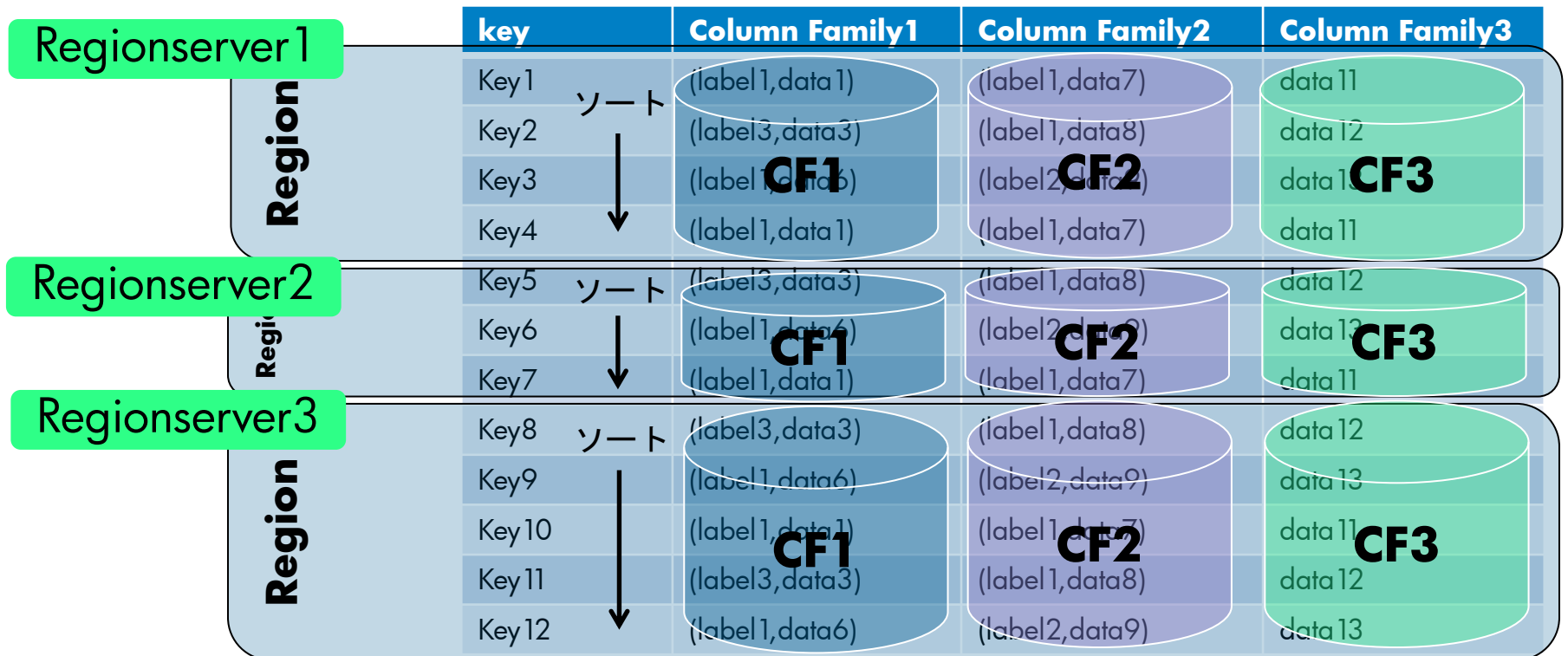
	Column Family1	Column Family2	Column Family3
Key1	(label1,data1) (label2,data2)	(label1,data7)	data11
Key2	(label3,data3) (label1,data4) (label4,data5)	(label1,data8)	data12
Key3	(label1,data6)	(label2,data9) (label3,data10)	data13

Diagram annotations:

- Callout: カラムファミリー名 (Column Family Name) pointing to the header row.
- Callout: Row pointing to the Key2 row.
- Callout: カラム (Column) pointing to the (label3,data3) entry.
- Callout: 単一データ (Single Data) pointing to the data12 entry.
- Callout: カラムファミリー (Column Family) pointing to the (label1,data6) entry.

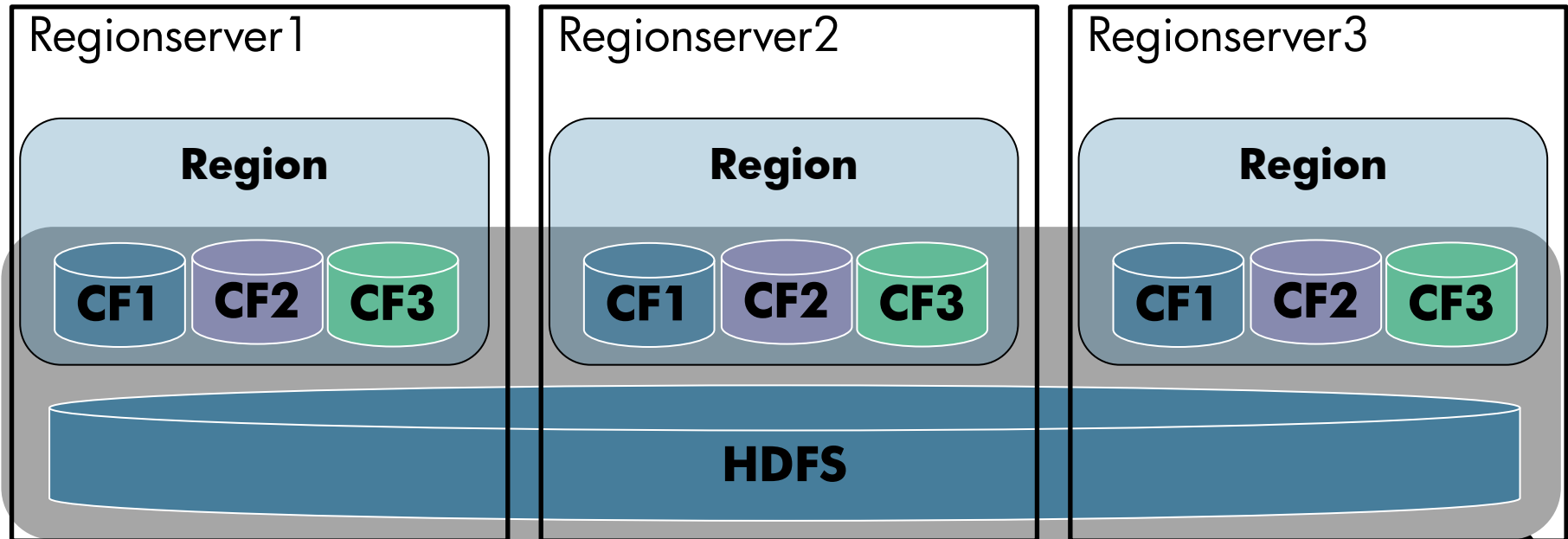
テーブルデータ管理方式

- テーブルは、カラムファミリー毎に管理（列指向データベース）
- Keyの範囲を分けてRegionに分割
- データはKeyでソートされて格納される



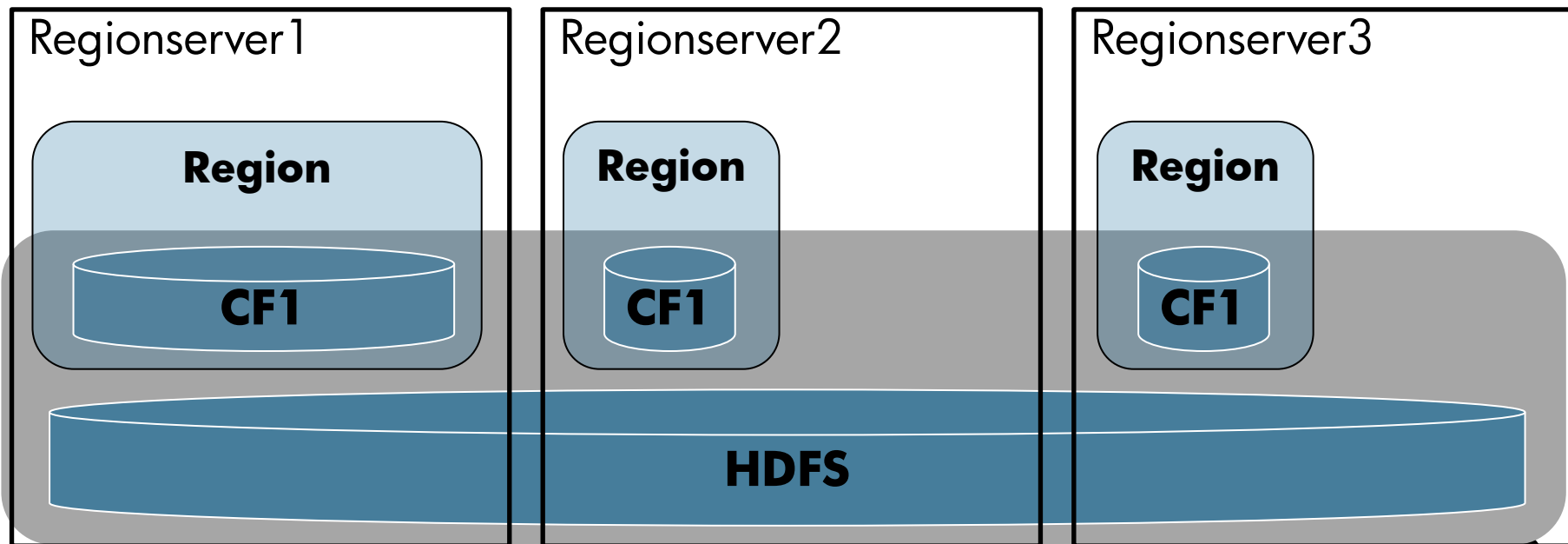
テーブルデータ格納方式

- カラムファミリーをRegionに分割することで、1つのテーブルを複数のサーバで処理可能となり、スケールアウトに対応
- データを定期的にHDFSに書き出すことで冗長化を実現



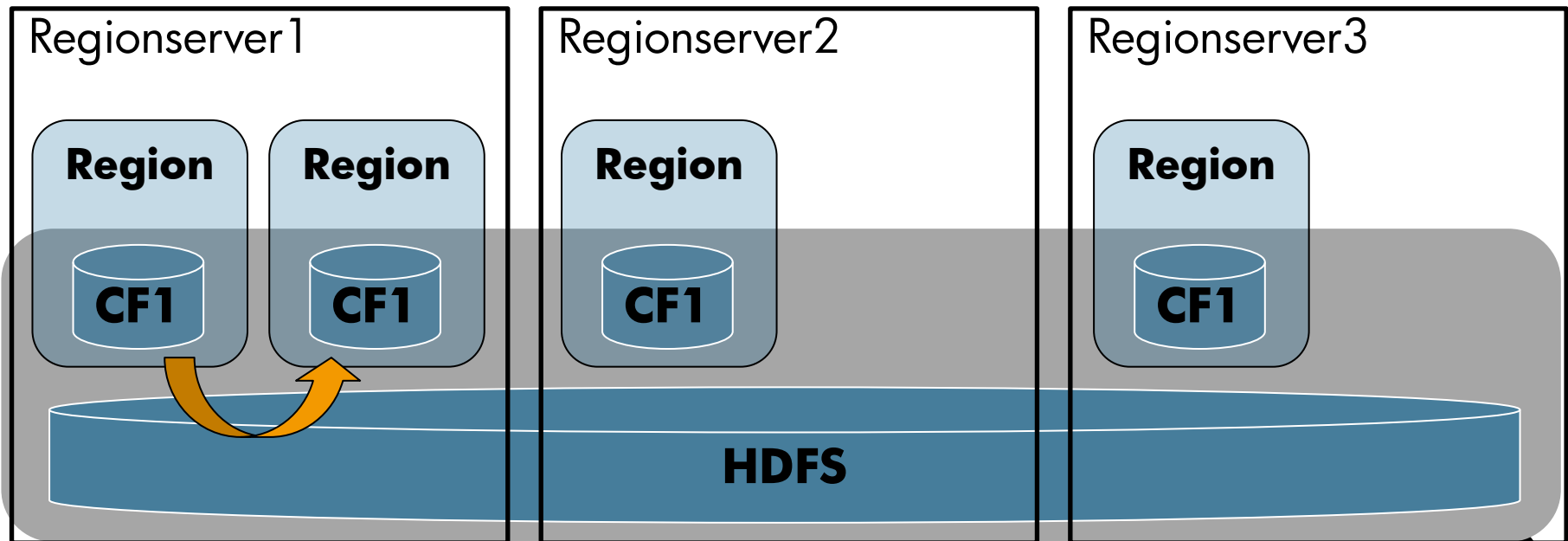
Region分割による負荷分散

- Regionのデータサイズが大きくなると、Regionを自動分割
- RegionserverごとのRegion数が均等になるように自動で再配置



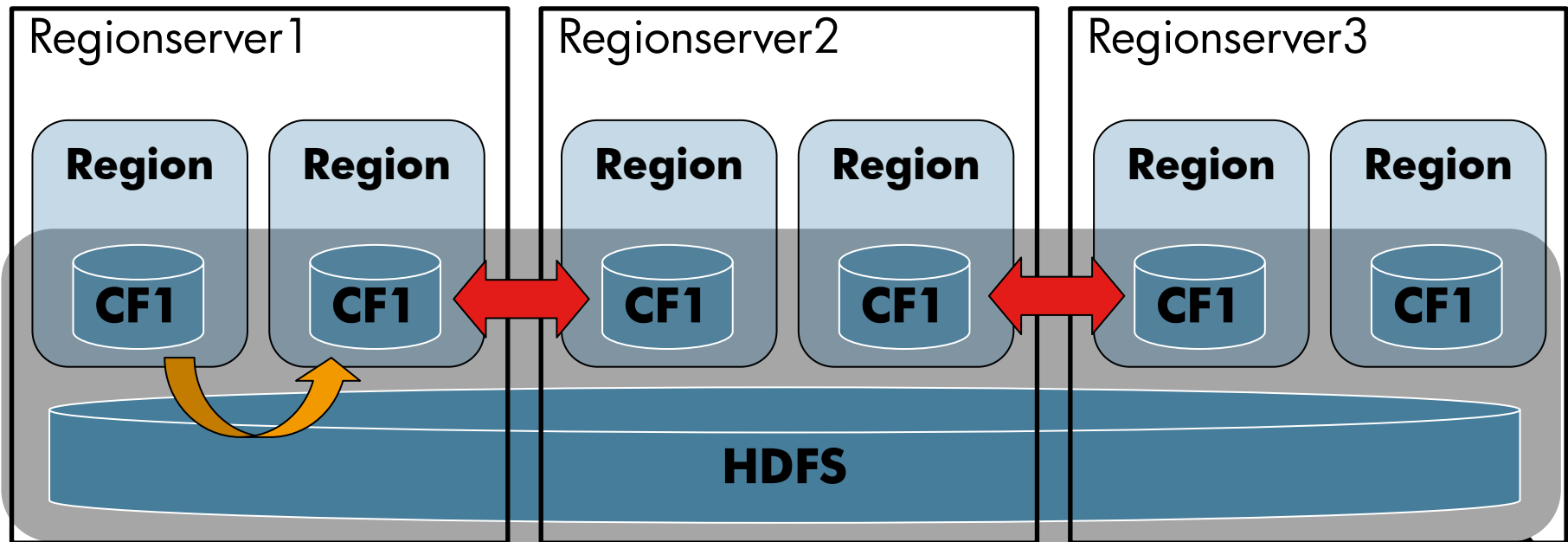
Region分割による負荷分散

- Regionのデータサイズが大きくなると、Regionを自動分割
- RegionserverごとのRegion数が均等になるように自動で再配置



Region分割による負荷分散

- Regionのデータサイズが大きくなると、Regionを自動分割
- RegionserverごとのRegion数が均等になるように自動で再配置



この節のまとめ

- HBaseは、「列指向データベース」
- キーごとにデータをRegionに自動分割、自動配置することでリニアにスケールアウト。
- データは、カラムファミリーごとに管理されていて、カラムファミリーはデータの物理配置に対応している。

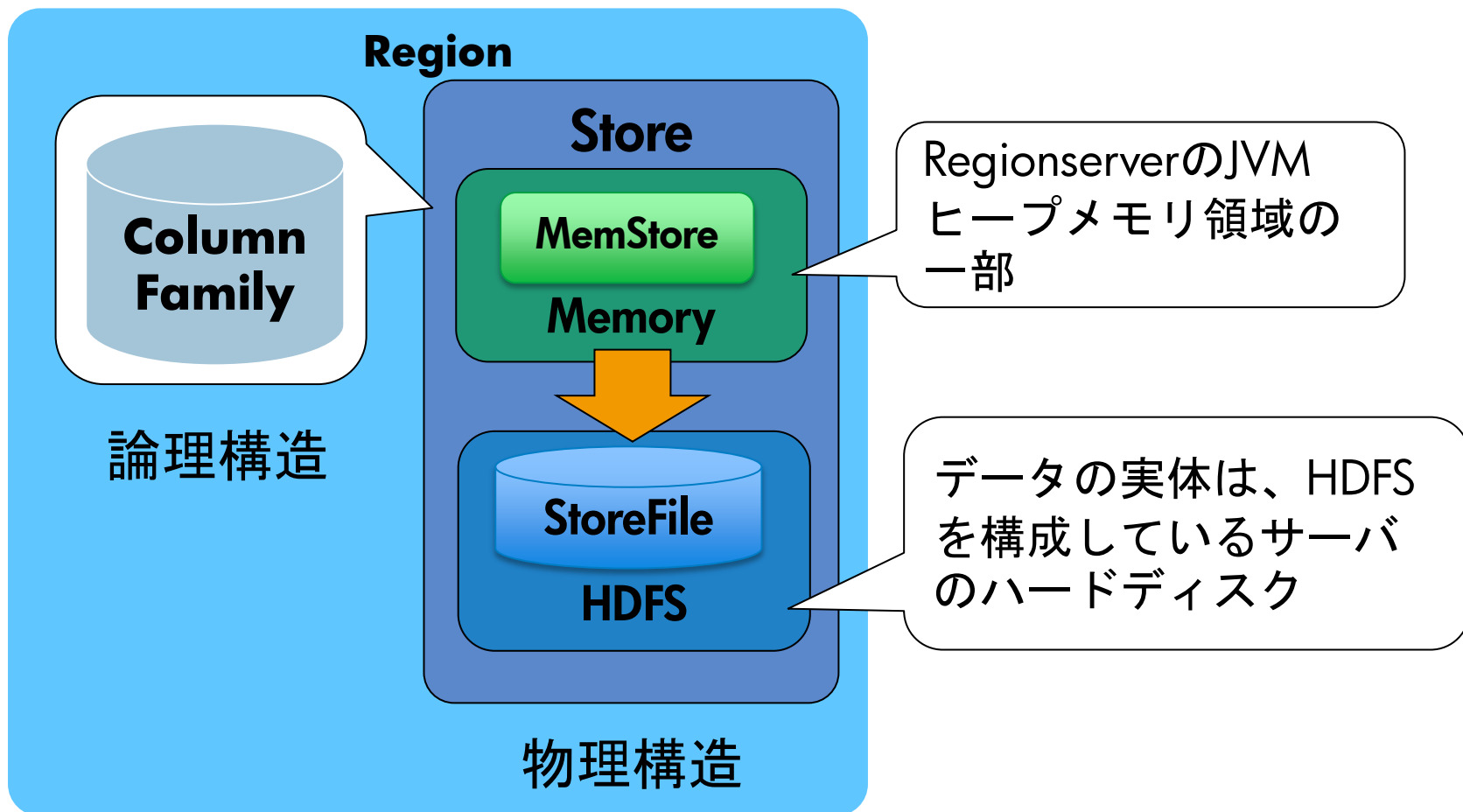


HBaseのデータ読み書き

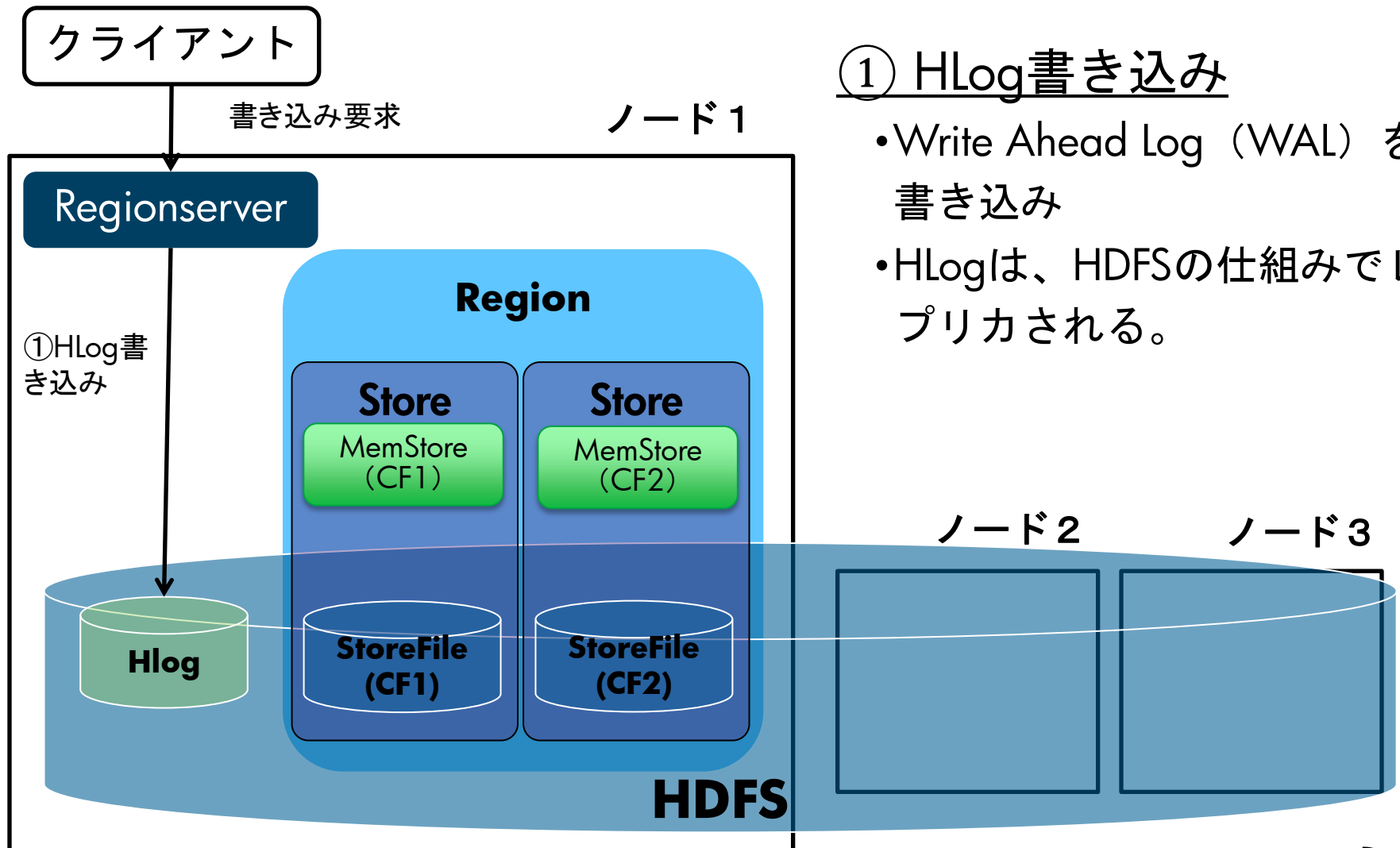


データの物理配置

- 各Regionのカラムファミリーは、「Store」と呼ばれる物理データ構造に1対1対応



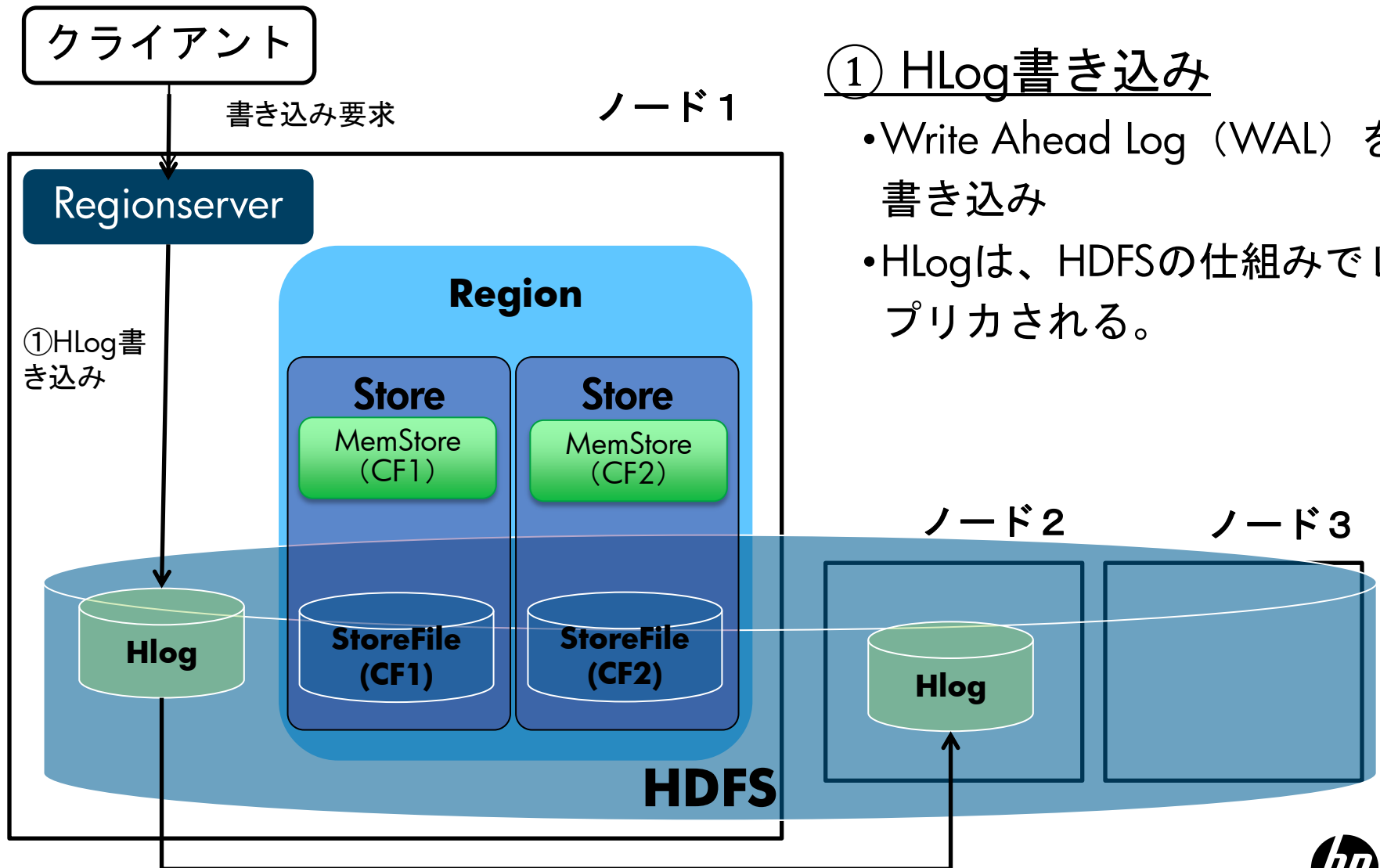
データ書き込みフロー



① Hlog書き込み

- Write Ahead Log (WAL) を書き込み
- Hlogは、HDFSの仕組みでレプリカされる。

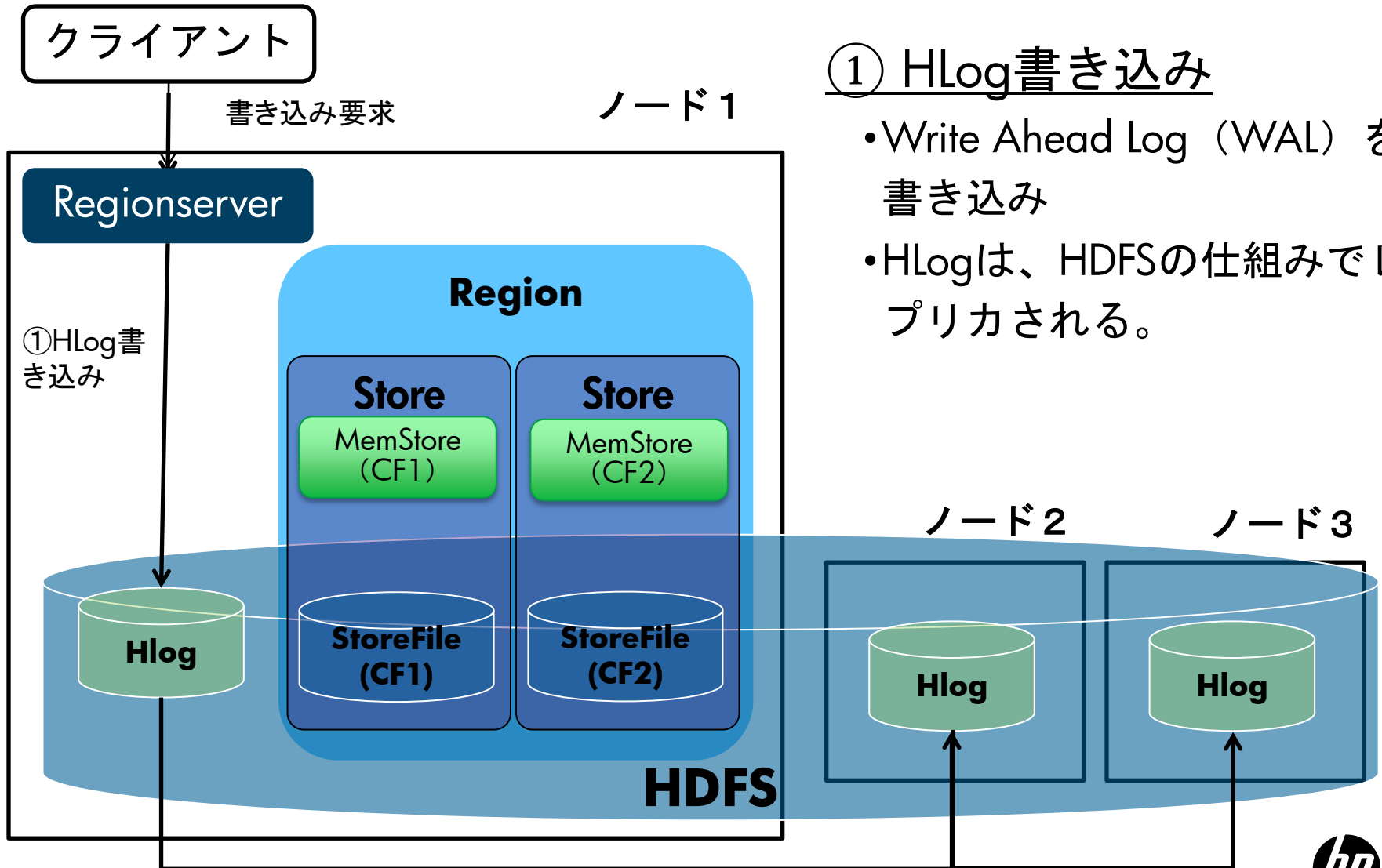
データ書き込みフロー



① HLog書き込み

- Write Ahead Log (WAL) を書き込み
- HLogは、HDFSの仕組みでレプリカされる。

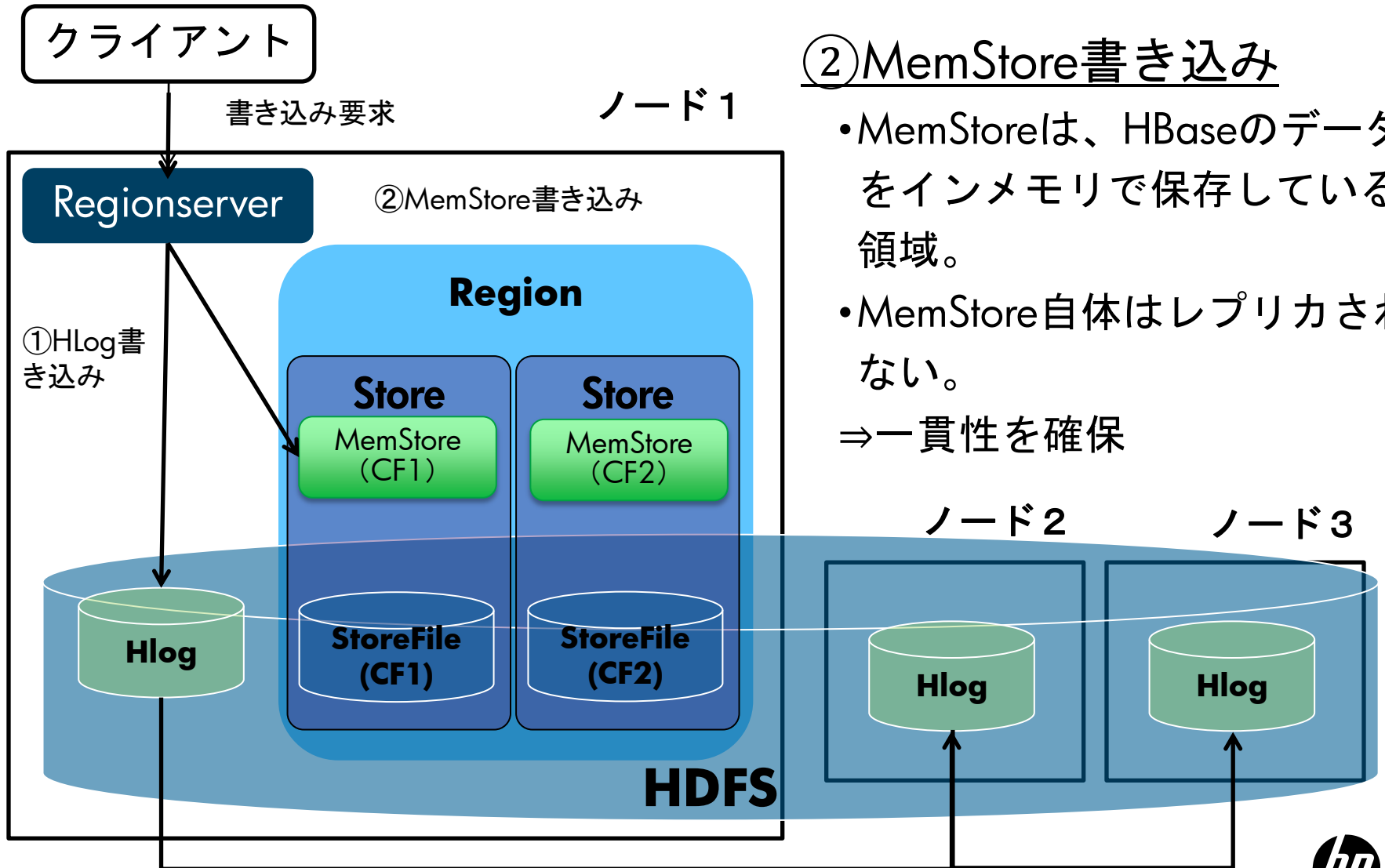
データ書き込みフロー



① Hlog書き込み

- Write Ahead Log (WAL) を書き込み
- Hlogは、HDFSの仕組みでレプリカされる。

データ書き込みフロー

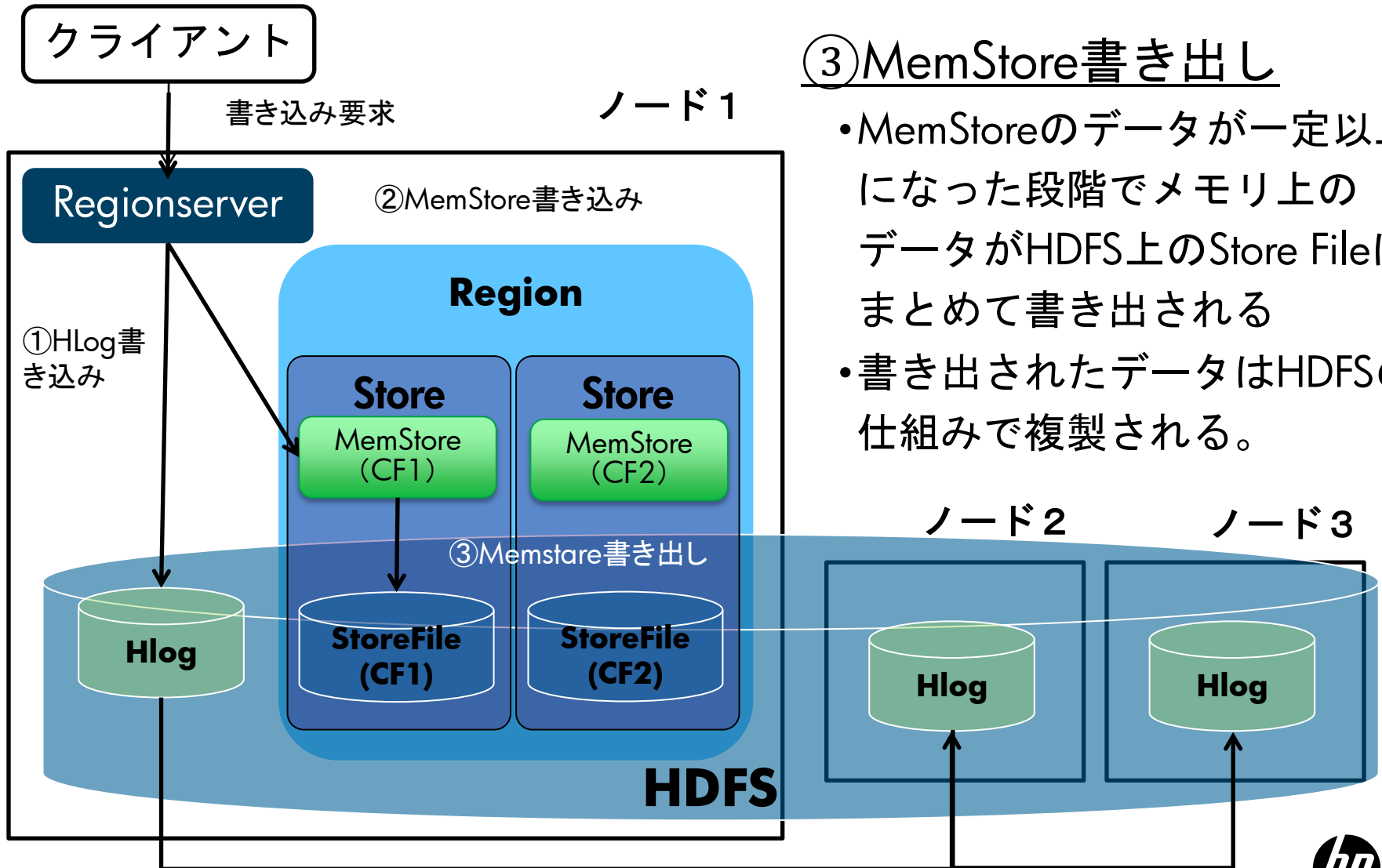


② MemStore書き込み

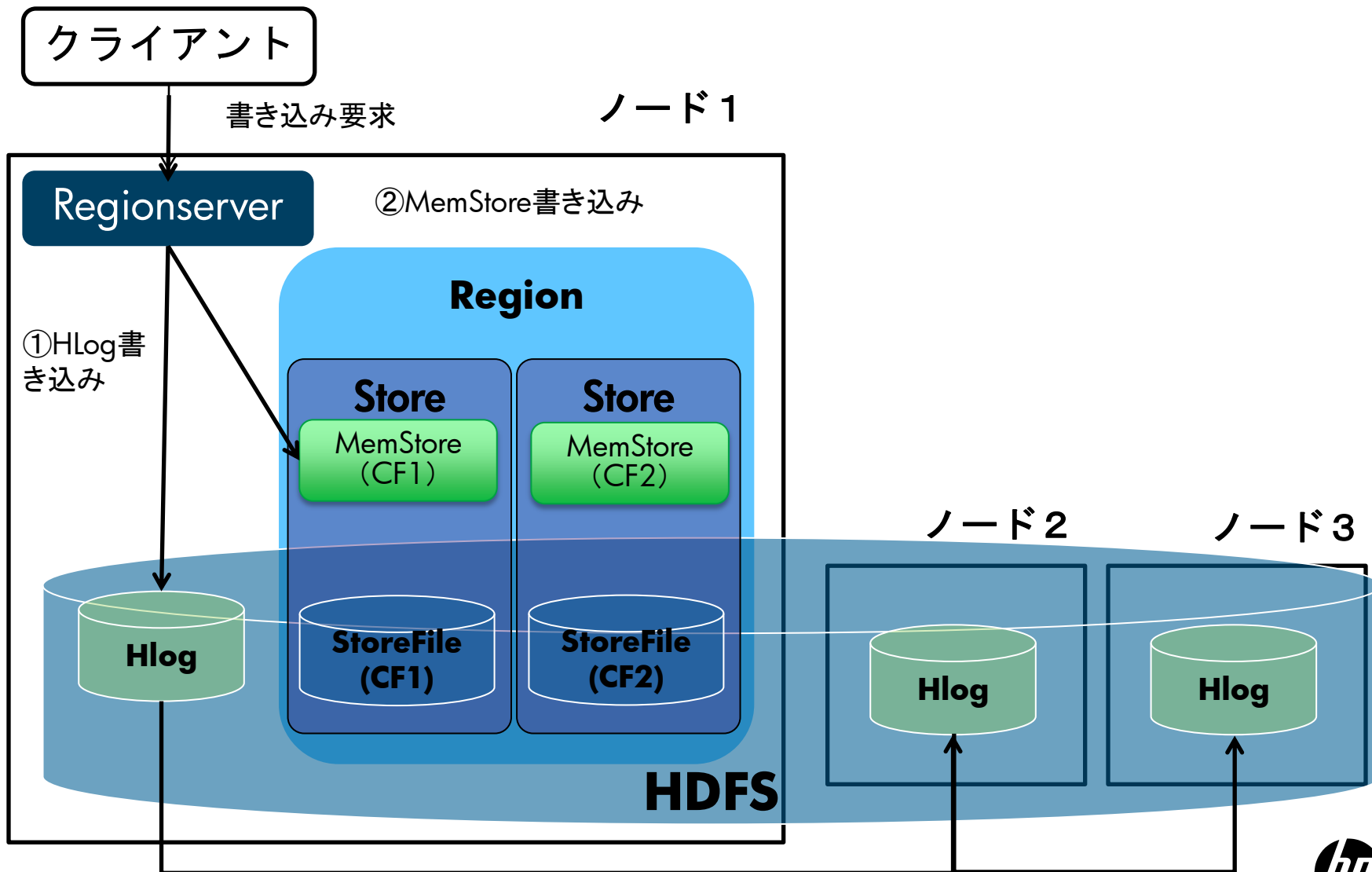
- MemStoreは、HBaseのデータをインメモリで保存している領域。
 - MemStore自体はレプリカされない。
- ⇒一貫性を確保



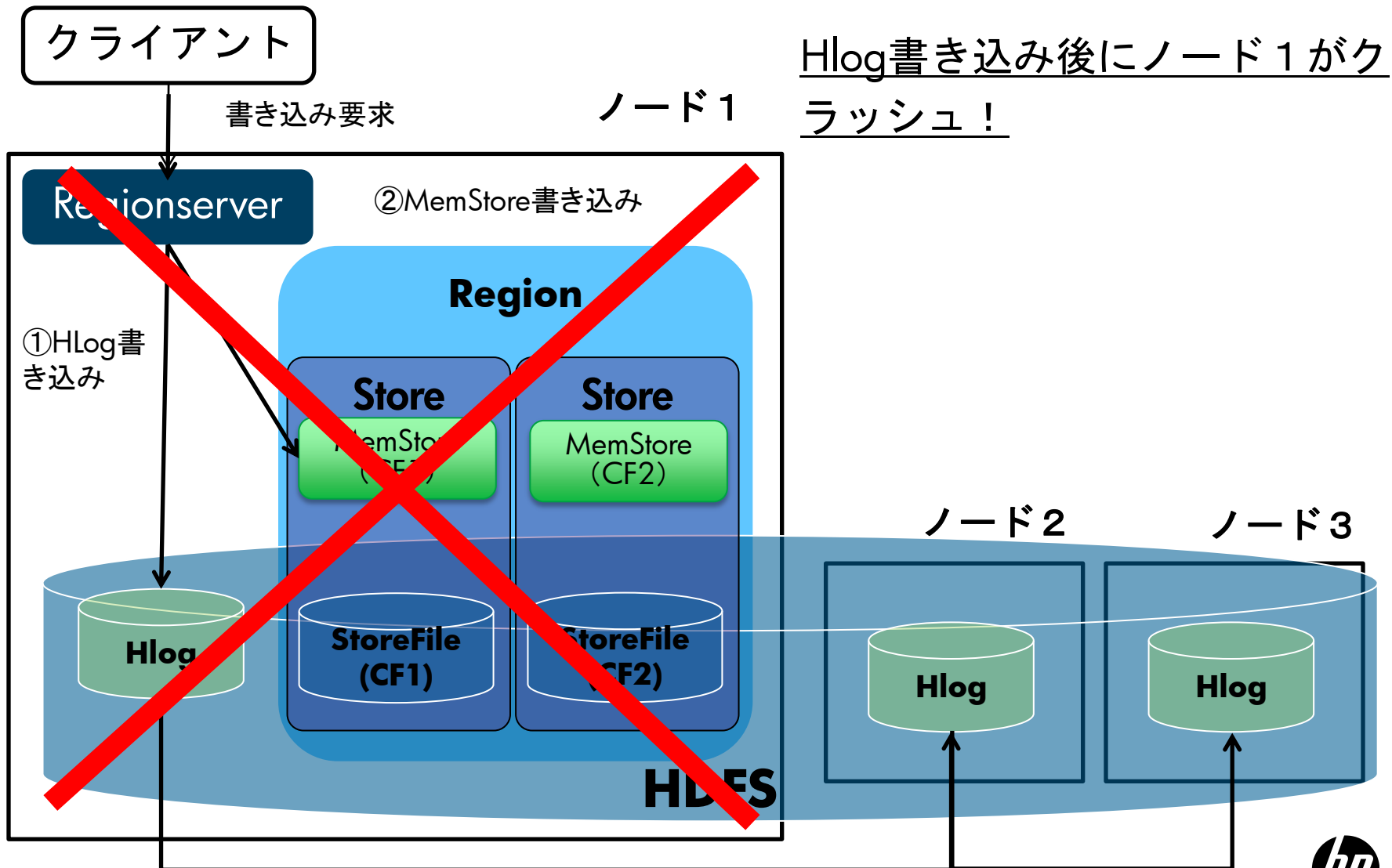
データ書き込みフロー



HBaseのサーバクラッシュ時の動作

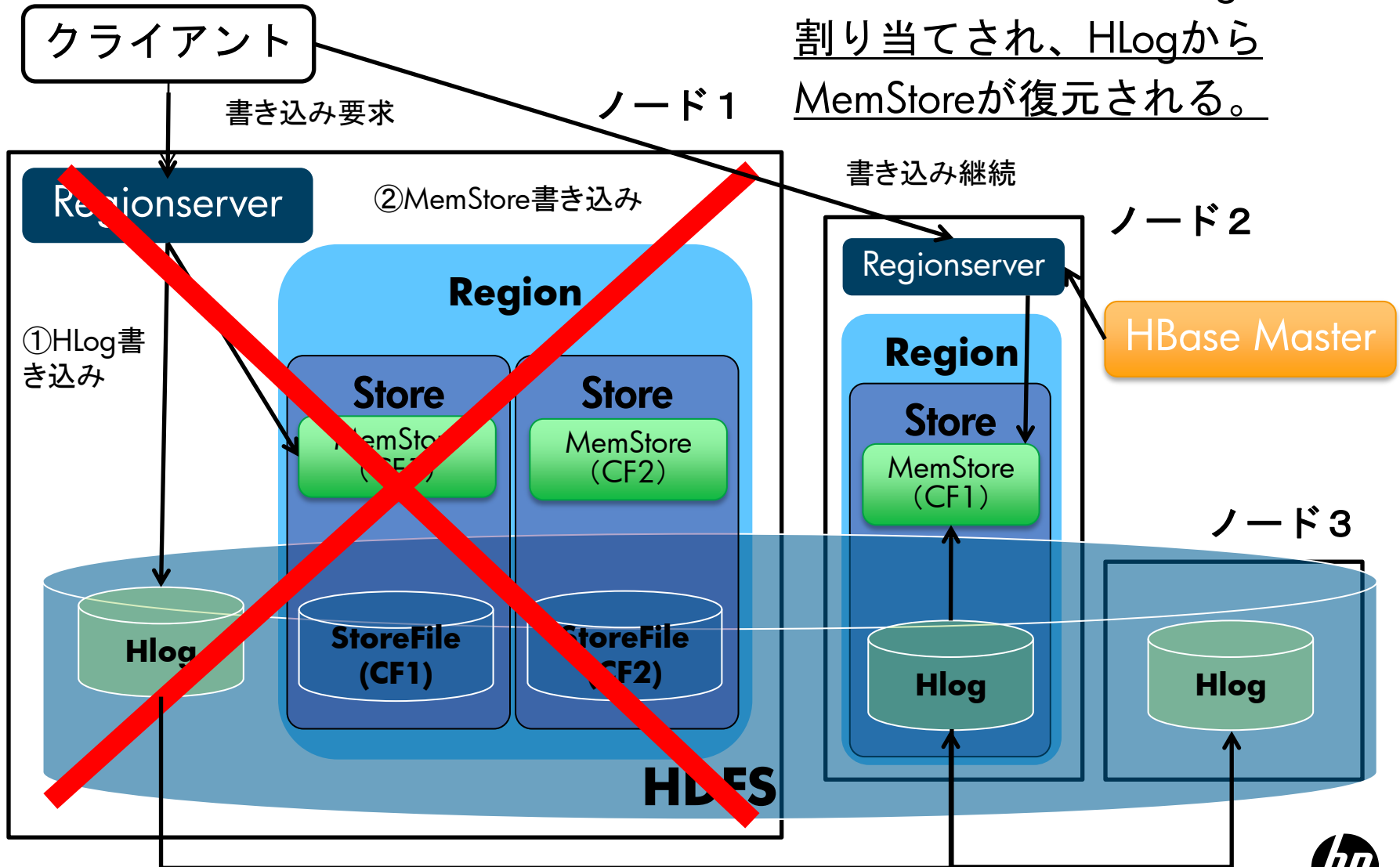


サーバクラッシュ時の動作

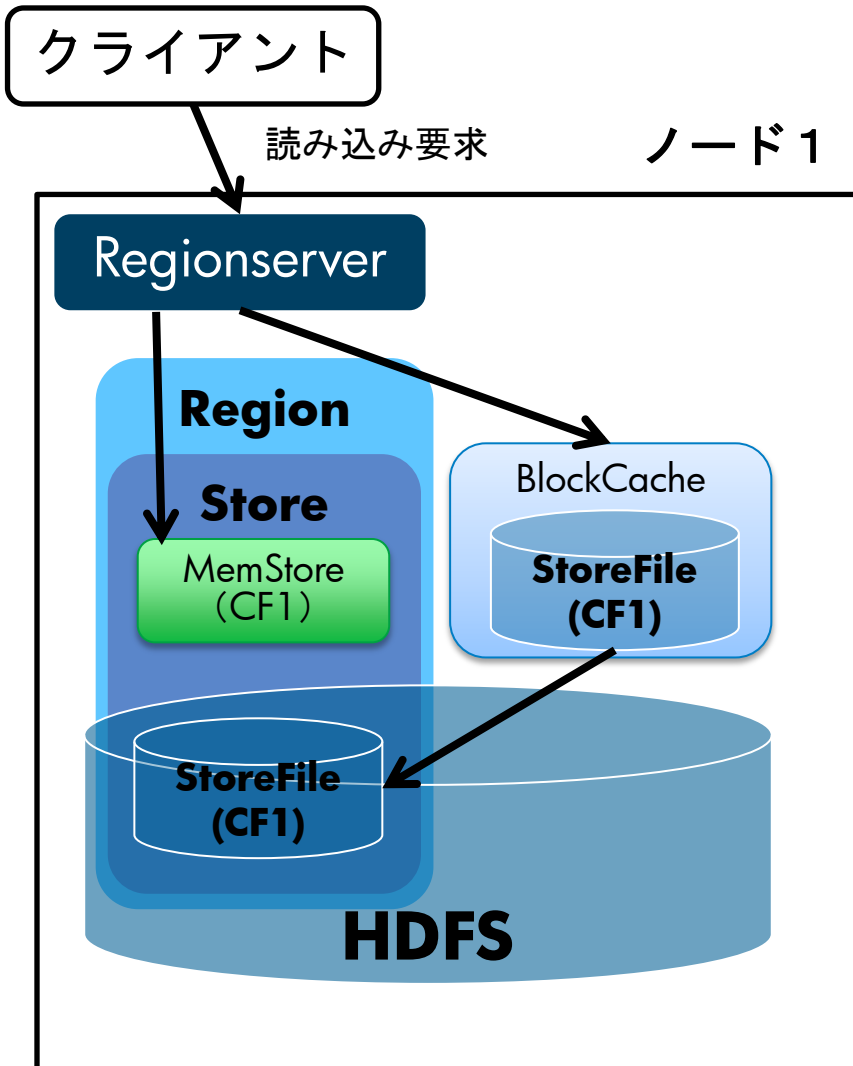


サーバクラッシュ時の動作

HBase MasterによりRegionが再割り当てされ、HLogからMemStoreが復元される。



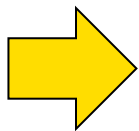
HBaseのデータ読み込み



- 読み込み時には、MemStoreとRegionserverのBlockCacheの両方にデータを探しに行く。
- BlockCache中に目的のデータがない場合は、HDFS上のStoreFileを読みだす

この節のまとめ

- HBaseはHLogによってデータ書き込みの信頼性を担保している。
- 1つのRegionをホストするノードはいつでも1つ。
- HBase MasterはRegion配置の管理のみを行っている。
- HBaseのIOは各読み込み、書き込みフェーズごとに最適化されている。
 - HBase
 - HDFS
 - 物理メモリ
 - 物理ディスク
- Regionserverのメモリ領域は以下の2種類がある。
 - 各Storeごとに1つのMemStore
 - Regionserverごとに1つのHDFSブロックキャッシュ
- HBaseはランダムライトとショートレンジスキャンに最適化されている。



これらを理解した上でモニタリングすることが重要
あとでZabbixを使った監視の手法をご紹介します。

HBase基礎性能検証



今回の検証の目的

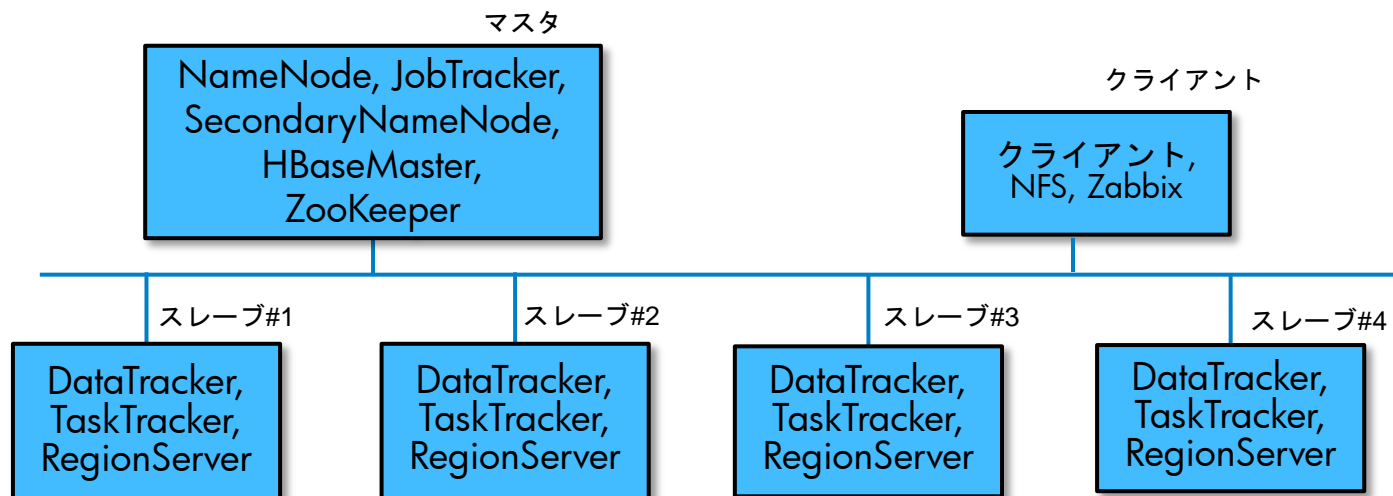
1. HBaseの基礎的な性能測定データの取得
 - 初期導入で想定される小規模構成における処理性能を把握
 2. HBaseの性能測定手法の確認
 - HBase性能測定の標準手法となっているYCSB(Yahoo! Cloud Serving Benchmark)の特性、利用にあたっての留意点
- 以下については今回の検証の範囲外となります
 - HBaseの導入方法
 - HBaseをスケールアウトさせた際の性能検証
 - スケールアウト検証は今後実施予定



試驗環境



試験環境構成

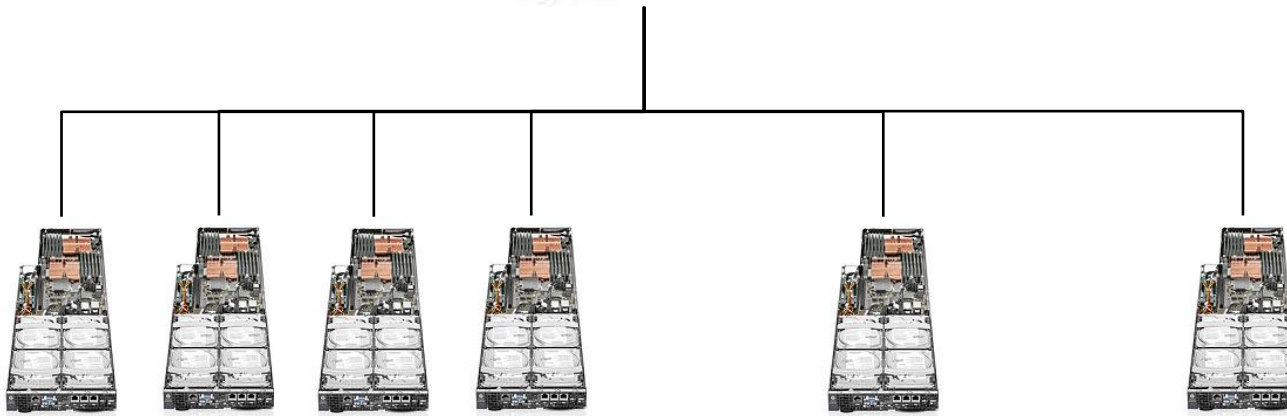


- マスタノードは1台構成
 - NameNodeのメタ情報の二次退避領域としてクライアントノード上にNFSサーバを構成
- スレーブノードは4台構成
 - HDFSのレプリカ数は3(default)
- 監視を行うためにクライアントノード上にZabbixサーバを構築
- 各ノード間をGigabit Ethernetスイッチにて接続



試験環境構成

Procurve2810-48G Gigabit Ethernet Switch



HBase RegionServer (Slave) x 4

HW: SL335sG7
OS: RHEL5.7(x86_64)
SW: CDH3u2, JDK1.6

HBase Master Server

HW: SL335sG7
OS: RHEL5.7(x86_64)
SW: CDH3u2, JDK1.6

YCSB Client

HW: SL335sG7
OS: RHEL5.7(x86_64)
SW: YCSB0.1, JDK1.6
Zabbix1.8



試験環境構成

	マスタノード	スレーブノード	クライアント
Hardware	HP ProLiant SL335s G7	同左	同左
CPU	AMD Opteron 4100(2.4GHz) 2P12C	同左	同左
Memory	32GB	同左	同左
HDD	SATA 1TB × 4 (RAID1+0)	SATA 1TB × 4 (RAIDなし)	SATA 1TB × 4 (RAID1+0)
OS	RHEL5.7 (x86_64)	同左	同左
NIC	Embedded HP NC362i DP Gb	同左	同左
Software	J2SDK 1.6.0_29 (64bit) CDH3u2 (64bit)	J2SDK 1.6.0_29 (64bit) CDH3u2 (64bit)	J2SDK 1.6.0_29 (64bit) YCSB 0.1.3 Zabbix1.8

- スレーブノードのHDDはRAID構成をとらず、4本のDiskにI/Oを分散させる
 - 冗長性はHDFSのレプリケーション機構で担保する
- YCSBはソースからのコンパイルを行いました



YCSBを用いた試験実施手順



試験シナリオ

- YCSBにより以下の2種類の負荷を生成
 - **Read** : 対象テーブルの全レコードから一様分布に従って選択されたrowkeyの全フィールド値を読みだす操作。(HBaseのAPIのget操作)
 - **Update** : 対象テーブルの全レコードから一様分布に従って選択されたrowkeyの全フィールド値をランダムな値で更新する操作 (HBaseのAPIのput操作)

- シナリオは以下の4種類を作成
 1. Readのみを実行
 2. Updateのみを実行
 3. ReadとUpdateを1:1の割合で実行
 4. ReadとUpdateを9:1の割合で実行



試験項目

- 試験項目

データサイズ	1レコード1KB × 3,000,000レコード = 3GB
シナリオ	シナリオ1～シナリオ4
スレッド数	10, 100, 250, 500, 1000の5種類 ※各操作と次の操作の間の待ち時間は0秒

- 試験結果の集計項目

- Read実行回数/秒
- Update実行回数/秒
- Read平均応答時間(ms)
- Update平均応答時間(ms)



主な設定値

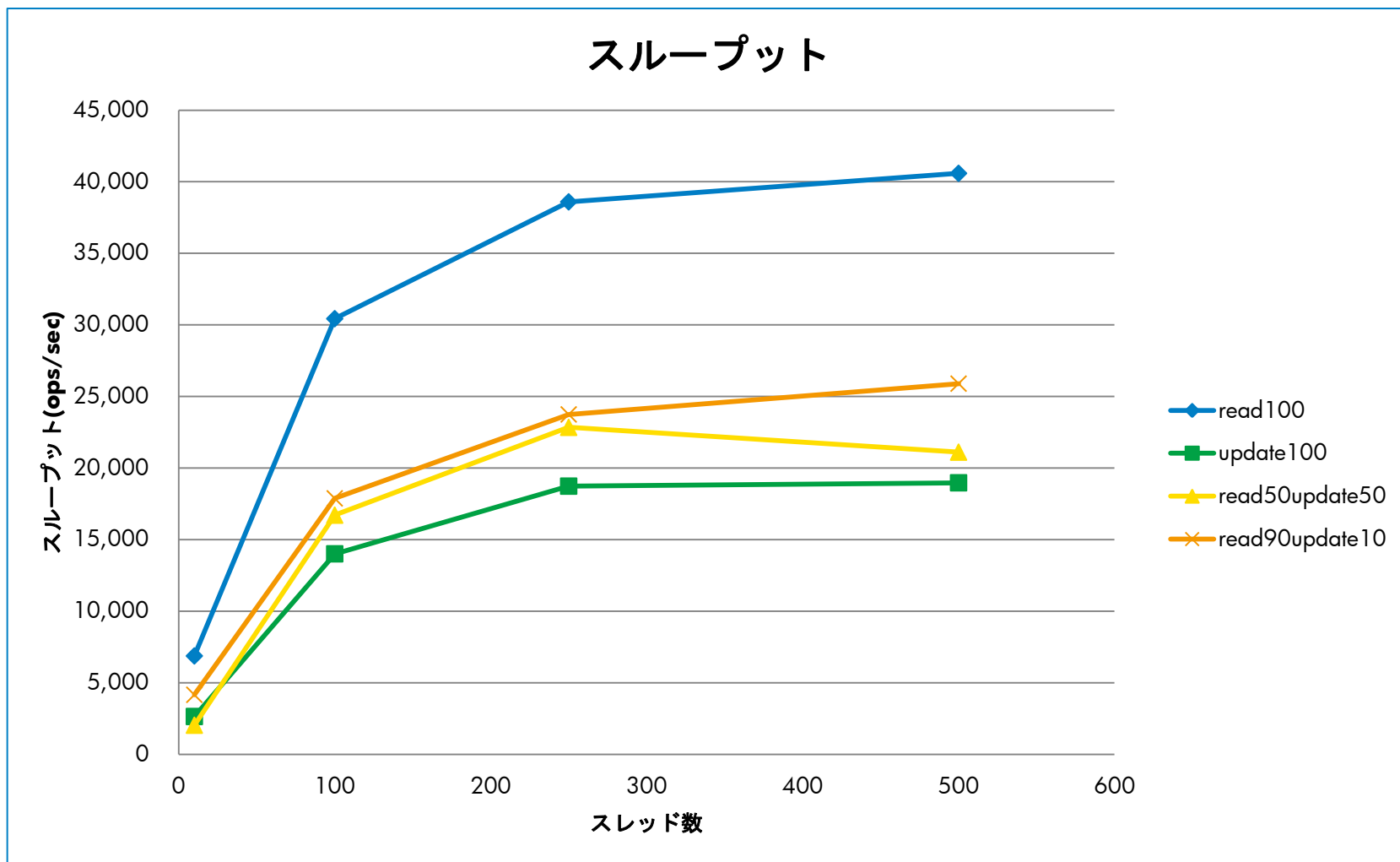
設定ファイル	項目	既定値	設定値	備考
core-site.xml	io.file.buffer.size	4096	16384	シーケンスファイルの読み書きで使用するバッファサイズ
hdfs-site.xml	dfs.block.size	67108864	33554432	新規ファイルのブロックサイズ
hbase-site.xml	hbase.hregion.memstore.mslab.enabled	false	true	書き込み負荷が高い状況下でヒープのフラグメンテーションを防ぐ
	hbase.hregion.memstore.mslab.chunksize	2097152	2097152	書き込み負荷が高い状況下でヒープのフラグメンテーションを防ぐ
	hfile.block.cache.size	0.2	0.3	Heap内のLRUキャッシュの割合
hbase-env.sh	HBASE_HEAPSIZE	1000	25000	region serverのヒープサイズ
	HBASE_OPTS	N/A	-ea - XX:+UseConcMarkSweepGC - XX:+CMSIncrementalMode	region serverのGC方式



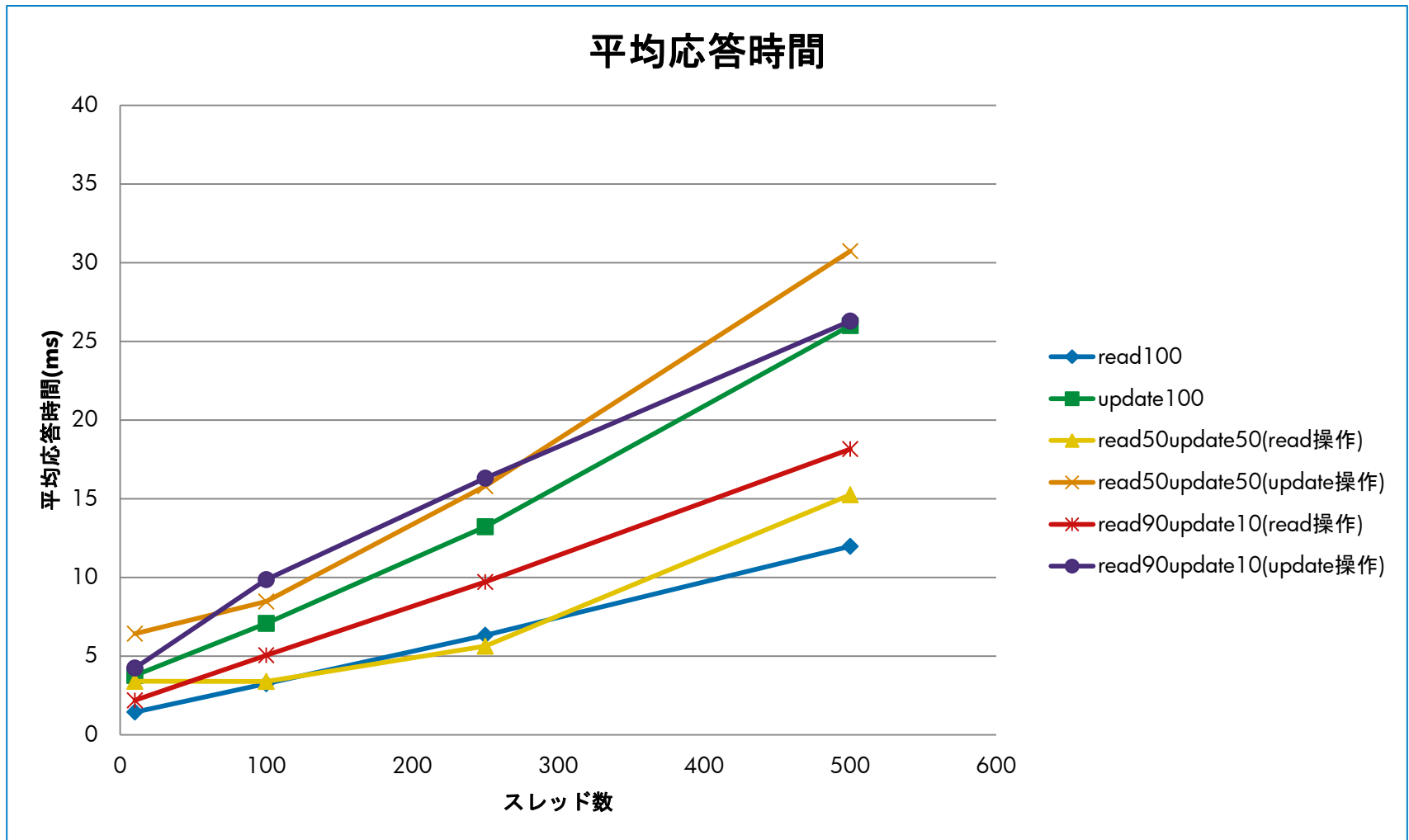
試驗結果



①スループット



②平均応答時間



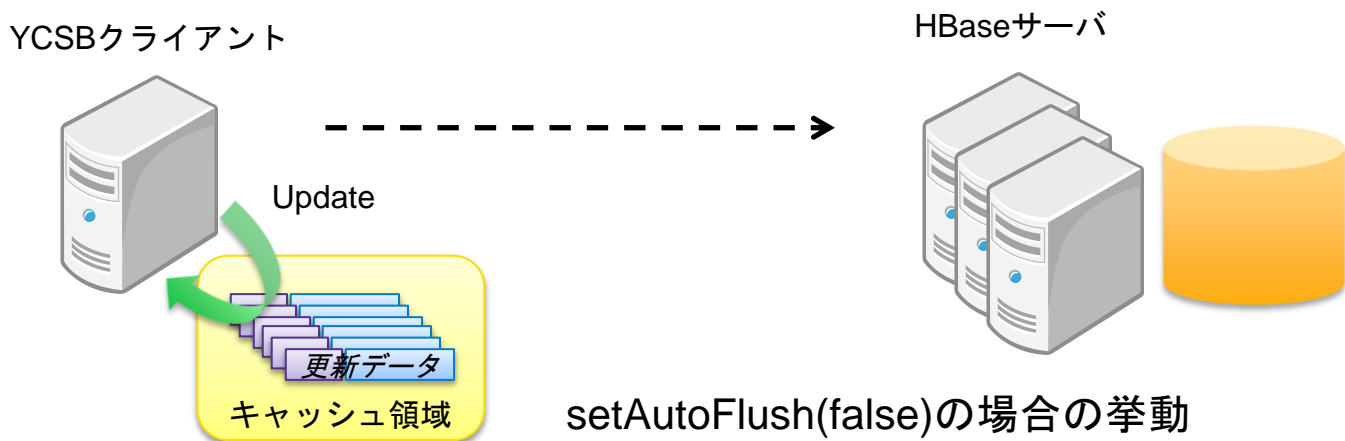
検証でハマったポイントと 今後の検証項目



検証でハマったポイント

その1 YCSBの書き込みが速すぎる！？

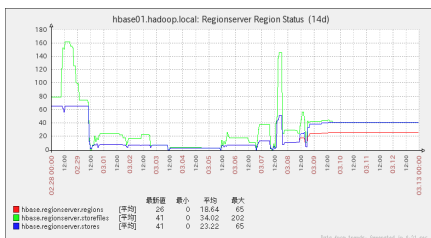
- YCSBをビルドしてupdateを行ったところ、マイクロ秒クラス（メモリの反応速度！）の応答時間！？
 - デフォルトではYCSBクライアントが更新レコードデータをサーバに送信せず、クライアント上でキャッシュする
 - Update操作の応答時間が非常に短く見える（40μsec程度）
 - クライアントキャッシュがフルになった時点でフラッシュするため、その時点で応答時間が極端に長時間になる
 - 対処：自動フラッシュを行うようにYCSBのソースコードを修正
 - `_hTable.setAutoFlush(false);` ← `true`に修正して自動フラッシュさせる



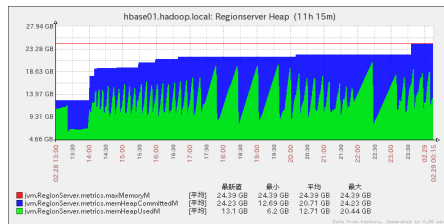
検証でハマったポイント

その2 書き込みの速度が落ちるタイミングがある

- Updateのベンチマークを続けていると時折書き込みが遅くなる。
- 原因はRegion分割？ フルガベージコレクション？
 - ZabbixとRegionserverのログから推移をトラッキング
 - Region分割時の書き込み停止の影響がわかったのでRegion数が適切な数に分割されたのち「hbase.hregion.max.filesize」を大きくして、ベンチマーク中のRegion分割を抑制
 - この変更によって断続的な速度劣化は解消。



Region数の推移



Regionserverヒープサイズの推移

※グラフは表示例です。

項目	設定値
hbase.hregion.memstore.mslab.enabled	true
hbase.hregion.max.filesize	100G
HBASE_OPTS	-ea -XX:+UseConcMarkSweepGC -XX:+CMSIncrementalMode

関連するHBase設定

<解析>

- Regionの分割は、各RegionserverのログおよびHBaseのメトリクスに記録されたRegion数の推移からトラッキングできる。
- YCSBのログに出力されている断続的なupdateの遅れとRegion分割のタイミングを突き合わせて原因を特定
- ガベージコレクションログを見たところ、フルガベージコレクションは発生していなかった。これはHBaseの新機能「MSLAB」を有効化したためだと考えられる。



検証でハマったポイント

その3 だんだん書き込み性能が劣化する

- Update試験を続けているとだんだんスループットが低下
 - HBaseのIOを見直して、HDFSのブロックサイズを32MBに変更し改善した
- 当初のHDFS設計
 - Hadoopクラスタで一般的な128MBに設定
 - この設定は一度に大量のデータを読み書きすることを前提としたもの
 - HBase向けではワークロードに応じた設定の見直しが必要

項目	設定値
dfs.block.size	33554432

関連するHDFS設定



今後の検証項目

- 今後検証を進めていきたい項目
 - スケールアウトによる性能向上
 - クライアントキャッシュの活用
 - 実運用時のワークロードを踏まえたScanの活用
 - BloomFilterを使ったRead性能の向上
 - データ圧縮によるディスクの節約とIOの削減
 - Namenode HA (Hadoop 0.23、CDH4b1～) を使ったHA検証

- 今後、引き続き検証を進め、成果を発表していきたい。



Zabbix向け Hadoop 監視プラグインのご紹介



Zabbixプラグイン開発の経緯

- 開発時の課題
 - Hadoopクラスタの監視は、GangliaとNagiosの組み合わせがデファクトスタンダードだが、日本国内での運用を考えた場合、運用性、アカウント管理やサポート等に課題
 - Ganglia/Nagiosのカスタマイズは、ソースコードを改変する必要がある。
 - Zabbixの選択理由
 - 日本国内で人気があり、コミュニティが活発なOSS統合監視ソフト
 - 性能監視と障害通知が1つのソフトで可能。
 - カスタマイズが容易。集めた情報をグルーピングして一覧表示可能
- ⇒社内検証用を開発し検証時のフィードバックを通じてブラッシュアップ



Zabbixプラグイン表示イメージ Hadoop Cluster Overview

Hadoop Cluster全体の負荷状況をモニタリング

- 各ノードの統計情報をグループ単位で集約（合計、平均、最小、最大）して表示
- Hadoopノードの死活やエラーを一画面で表示
- 集約の単位はカスタマイズ可能
- 表示範囲は、1時間～2年間まで分単位で選択可能。（※1）



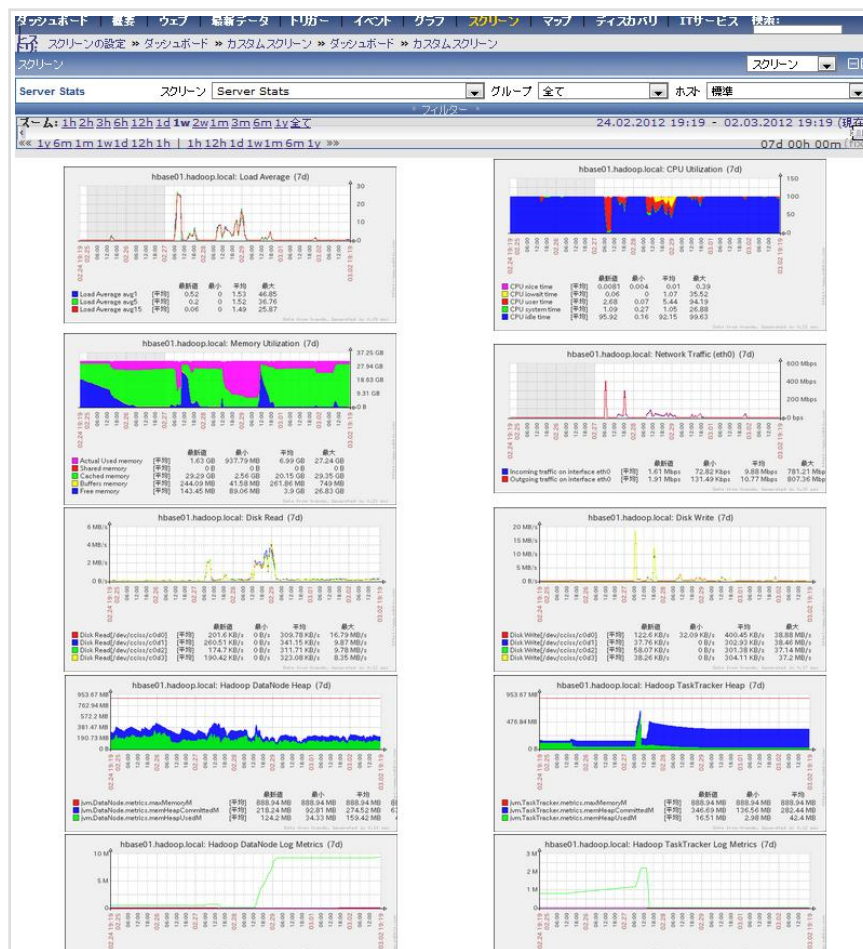
※1 表示できる期間は、データ保存期間に依存します。



Zabbixプラグイン表示イメージ 各ノードの負荷状況

各ノードの負荷状況をOSレイヤ、Hadoopレイヤを統合して表示

- 各ノードのOS統計情報とHadoop Metricsを一画面に表示
- 表示範囲は、1時間~2年間まで分単位で選択可能。(※1)



※1 表示できる期間は、データ保存期間に依存します。



Zabbixプラグイン表示イメージ HBase統計情報画面

HBaseの多様なメトリクスを一覧表示。任意の期間を指定できる。

- HBaseのノードごとに主要な統計情報を一覧表示
 - HBase MemStore（書込用メモリ領域）とブロックキャッシュ（ディスクキャッシュ領域）の使用量
 - キャッシュヒット率
 - Regionserverのヒープメモリ使用状況
 - ホストされているRegion（データ保存単位）数の推移
 - Regionserver内の操作の遅延時間
 - DatanodeのIO
- 表示範囲は、1時間～2年間まで分単位で選択可能。（※1）
- モニタリングで必要になることが多い主要なパラメータのグラフは最初から設定済み。
- 表示情報カスタマイズ可能
- 統計情報を横断的に把握することでボトルネック解析が容易になる



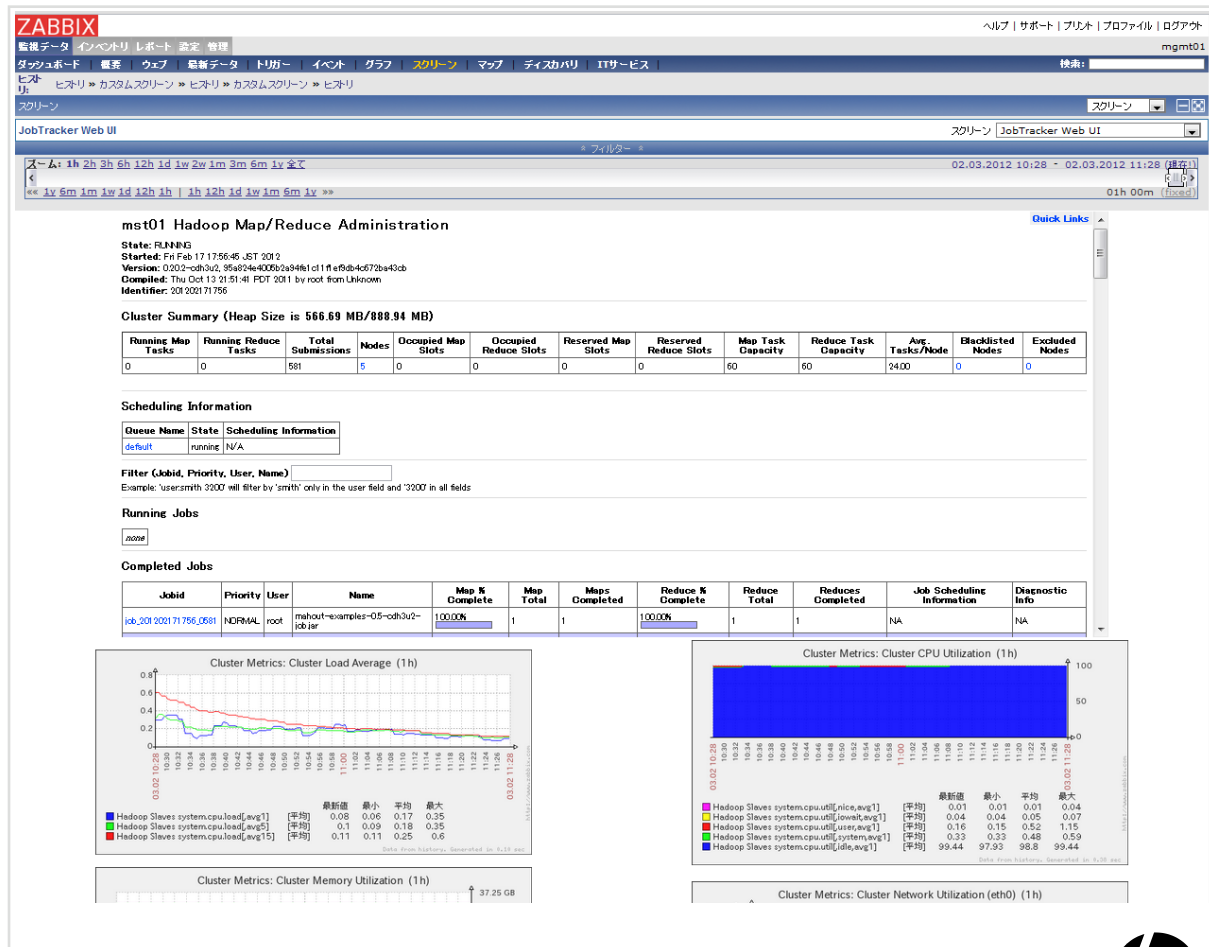
※1 表示できる期間は、データ保存期間に依存します。



Zabbixプラグイン表示イメージ カスタム画面例-MapReduceモニタリング

Hadoop JobTrackerとHadoopクラスタの負荷状況を一画面で閲覧

- Zabbix上に格納されている統計情報とHadoop JobTrackerの情報を一画面で表示する例
 - MapReduceジョブの進行状況とクラスタ全体の負荷状況を同時に確認可能
- このほかにも必要な情報をまとめたカスタム画面を随時追加可能
- Zabbixの「スクリーン」機能を利用することで、このように外部のURLも含めた統合を実現
- 定義済みスクリーンを組み合わせた画面表示も可能



そのほかの機能

Zabbixの機能をフル活用

監視対象の自動追加

- Zabbix Agentをインストールして起動するだけで、監視対象に自動追加（※1）

定義済みトリガー

- 基本的なトリガーがテンプレートに含まれており、通知方法を定義するだけですぐに監視ができます。



Hadoop関連 コンサルティングサービス



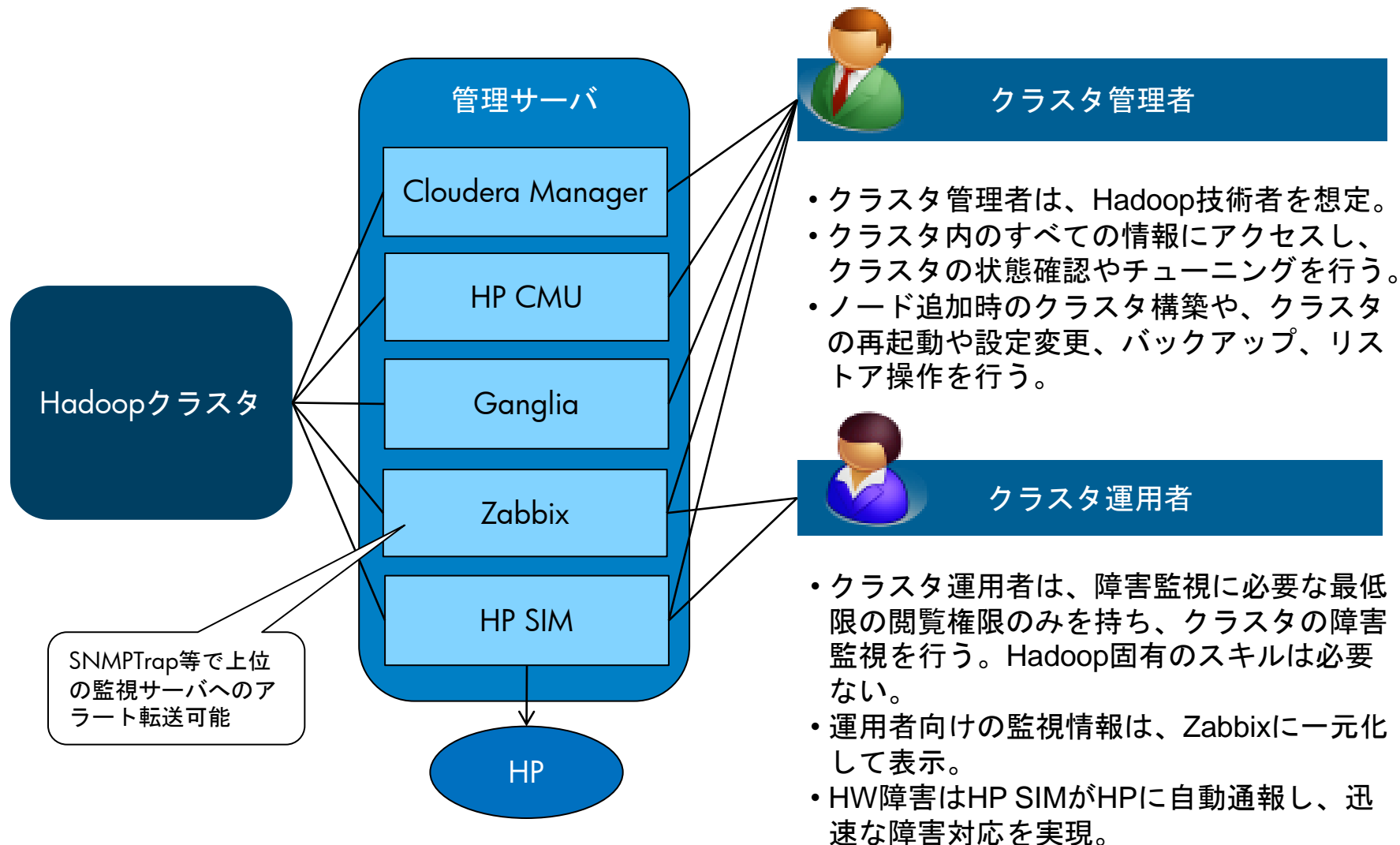
HPが提供するHadoop管理・監視ソリューションスタック

	対象	プロダクト
構築・管理	Hadoop (CDH)	<p>Cloudera Manager (※1)</p> <ul style="list-style-type: none"> Cloudera Managerは、CDH (※2) 専用の構築・管理ツール。OS上へのHadoop導入から設定管理を自動化。 GUI上からHadoopの設定の確認、変更、サービス再起動を実行可能。
	OS	<p>HP CMU</p> <ul style="list-style-type: none"> HP Cluster Management Utility(HP CMU)は、HPC分野で10年以上の実績を持つ、大規模システム向け管理ソフトウェア。国内でもTSUBAME2.0をはじめ、採用例多数。大規模システムでのOSデプロイ、バックアップ、リストア、システム全体に対するコマンド発行などを行うことができる。
	Hadoop (CDH)	<p>Ganglia</p> <ul style="list-style-type: none"> Gangliaは、大規模システムでの性能監視に特化したOSSプロダクト。 Hadoopには、Ganglia向けの設定が組み込まれており、それを使うことで効率的にHadoopシステムの性能情報収集と表示を行うことができる。
監視(性能監視・障害監視)	OS	<p>HP Hadoop プラグイン</p> <ul style="list-style-type: none"> 日本HPが作成のプラグインにより、Hadoopサービスの監視も統合 <p>Zabbix</p> <ul style="list-style-type: none"> Zabbixは、業界標準のOSSの統合管理ソフト。GUIから設定変更、サーバ状態の確認が可能な、使いやすいなインターフェースを持つ。 サーバ管理者と運用者などのサーバごとの監視権限設定が可能。
	HW	<p>HP SIM</p> <ul style="list-style-type: none"> HP Systems Insight Manager(HP SIM)は、HPが無償で提供しているサーバ監視ソフトウェア。専用Agentを用いて、HWのきめ細かい監視が可能。ノード障害の予兆を検知し、プロアクティブな対応を実現。HPへの自動通報機能を備える。

※1 Cloudera Managerには、50ノードまでの限定で無償で利用できるFree Edition版が用意されている。
 ※2 CDHは、Cloudera社が提供しているオープンソースのHadoopディストリビューションパッケージ。



管理・監視運用方式



★お客様の既存のサーバ運用方式とHadoop運用をシームレスに統合可能

Hadoop HBase サービス概要

- 大規模分散処理プラットフォーム「Hadoop」をベースとして、ビッグデータ活用に最適化したITインフラを構築するソリューション
- サーバー、データベース、運用ツールとあわせて、導入コンサルティングサービスも提供し、お客様のビッグデータ活用をワンストップで支援



Hadoop HBase コンサルティングサービス

- 適用領域、既存システムの連携についてのヒアリングを実施
- 必要に応じて、導入前検証をHPソリューションセンターで実施



Hadoop HBase導入支援サービス

- ヒアリング内容に基づくサーバ構成の検討・設計
- Hadoopに対応した全体アーキテクチャの設計・構築
- 性能測定・チューニング



Hadoop HBase運用支援サービス

- HP CMUおよびZabbix/Gangliaを使ったクラスタ運用基盤の構築
- Hadoopクラスタ運用支援

HP ビッグデータ分析 コンサルティングサービス



1. ディスカバリーワークショップ

- お客様の業務、システム改善を目的に、最新製品およびテクノロジーの紹介
- 統計分析によるテキスト/ログなどの事例紹介及びデモの実施



2. 分析支援コンサルティングサービス

- お客様の業務やシステムの全体像の把握、PDCAを回す全体設計
- 今後の対応策、予算規模、スケジュールなど上申書のもととなる情報を作成
- システムの将来像とロードマップの策定及び、全体の参考価格も算出



3. 分析システム導入支援サービス

- 策定したシステムの将来像に向けて、システム全体の要件定義、システム設計、システム構築サービスを提供
- 事前検証を実施



4. 分析システム運用支援サービス

- システム運用時の支援を実施

HP ビッグデータバッチ処理高速化 コンサルティングサービス



1. バッチ処理高速化コンサルティングサービス

- お客様にもアセスメントに参加して頂き、既存システムの問題点を的確に把握



2. バッチ処理高速化システム導入支援サービス

- アーキテクチャ、運用、監視などのシステム設計を実施
- 機能テストや、性能テスト、負荷テスト、耐久テストなどを実施
- 顧客毎のシステム要件にしたがいプログラミングを日本HPで行う

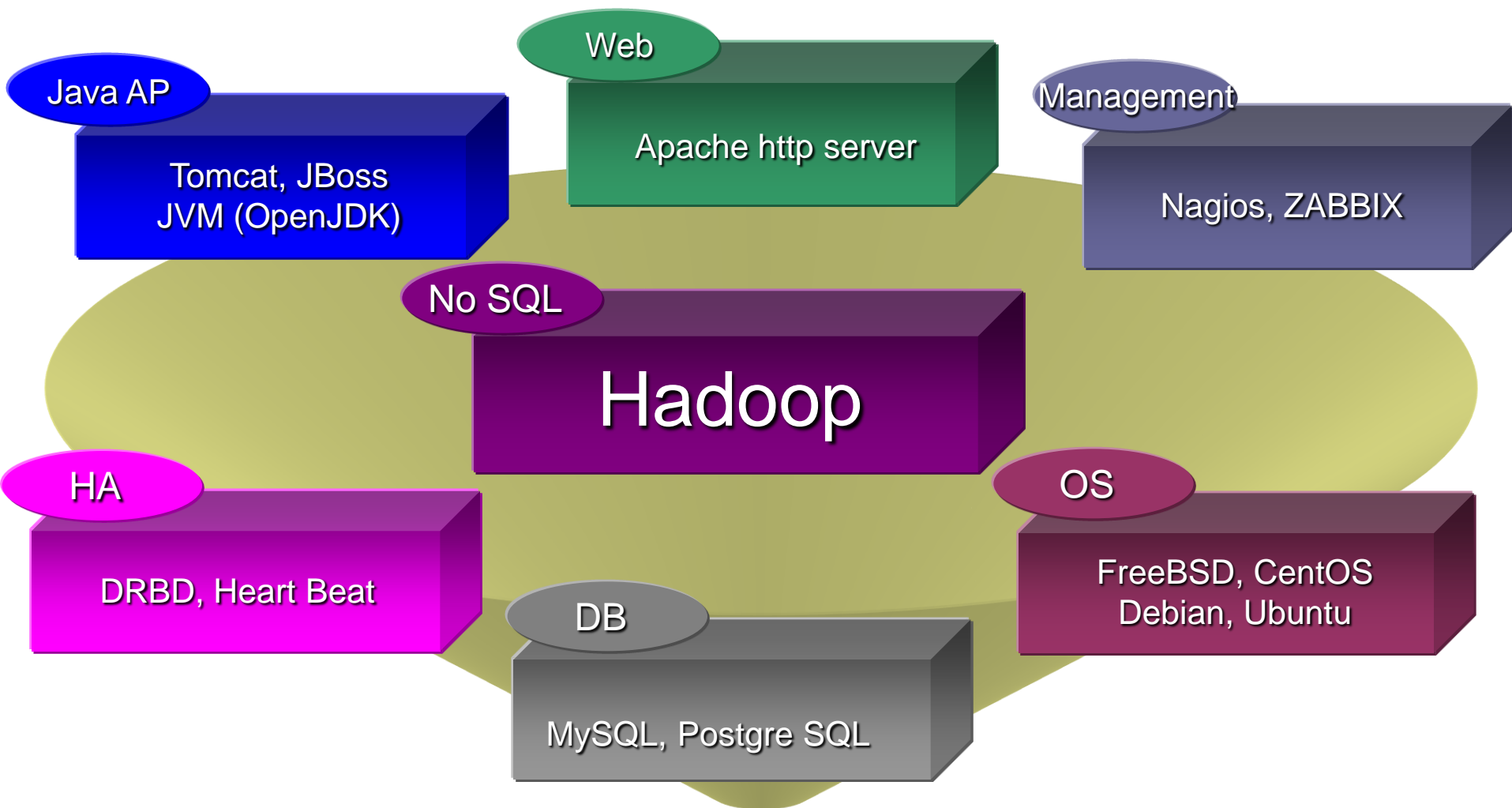


3. バッチ処理高速化システム運用支援サービス

- システム運用時の支援を実施

- 成果物：
- 既存システム問題解析結果報告書
 - システム設計書
 - システム運用手順書
 - システム監視手順書
 - テスト結果報告書
 - システム支援報告書

Hadoopだけでなく、関連するオープンソースソフトウェアを含めたトータルサポート



New!


- 最近の情報

- Clouderaの協力のもと、教育サービスを開始

- 次回開催: Hadoop開発者向けトレーニング 3/26-29

- <http://h50146.www5.hp.com/services/education/whatshot/newcourse.html>

ビッグデータ研修コース



»【新コース】ビッグデータのトレーニング開始！

- [Cloudera Apache Hadoop 開発者向けトレーニング\(4日間\)](#)
- [Cloudera Apache Hadoop 管理者向けトレーニング\(3日間\)](#)

(2012/3/2)

»スケジュールはこちら

- 「今さら聞けないHadoopとビッグデータ」をYoutubeで公開

- 前回のOSCで発表した古井が解説を行っております。

- <http://www.youtube.com/watch?v=Q4IZPDYB1y0>



今さら聞けない
Hadoopとビッグデータ

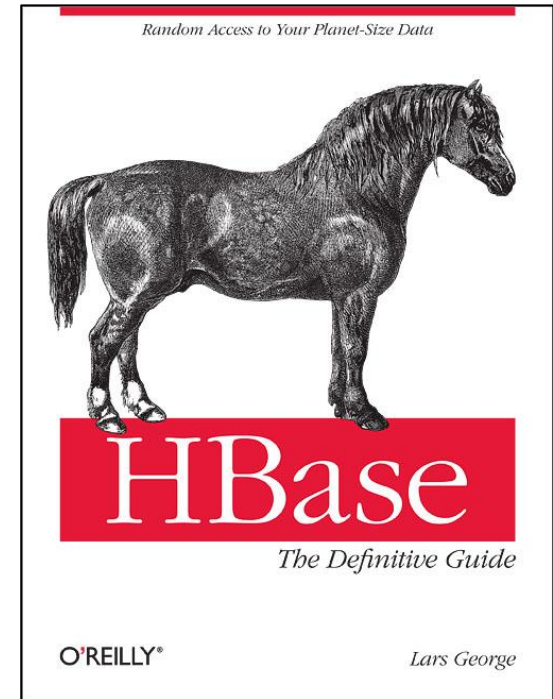
hp

日本ヒューレット・パッカード株式会社

それは料理好きの私にとっては興味深いですね
どういふ風に役立っているのが楽しみです

参考資料

- HBase: The Definitive Guide (O'REILLY)
 - Lars George 著
<http://ofps.oreilly.com/titles/9781449396107/>
- HBase Book (Web上で公開)
 - HBase プロジェクトの公式リファレンスガイド
<http://hbase.apache.org/book.html>



Q & A



Thank you

