特集論文

マルチテナント利用を実現する大規模データ分析共用基盤の構築

井ノ口裕也* 渡邊健太*
三屋誓志郎* 埋金進一**
福島愼一*

Construction of Large-scale Data Analysis Base to Realize Multi-tenant Use

Yuuya Inokuchi, Seishiro Mitsuya, Shinichi Fukushima, Kenta Watanabe, Shinichi Umegane

要　旨

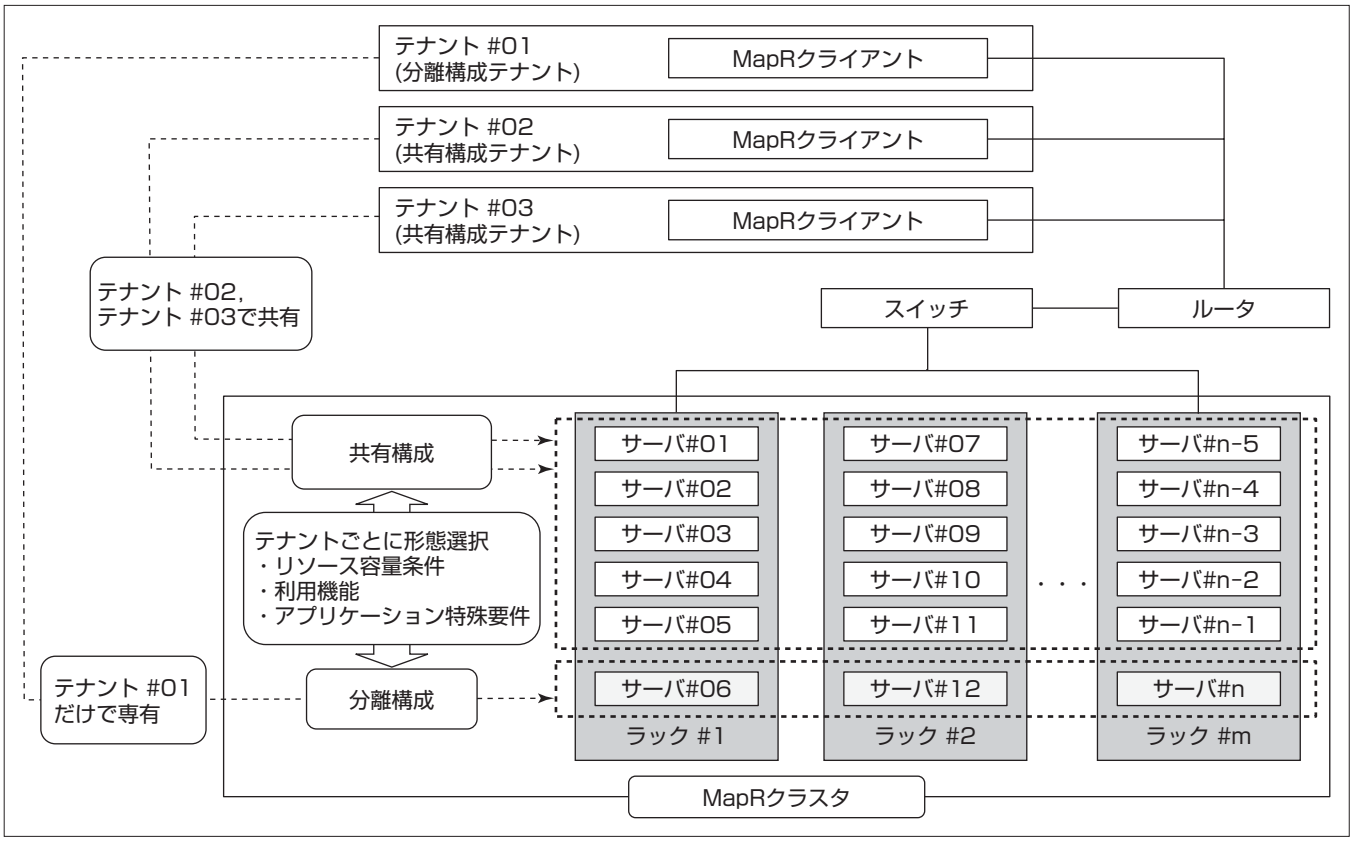
近年，コンピュータシステムが扱うデータ量は増加の一途をたどっており，膨大なデータを活用し，新たな価値を生み出すことが脚光を浴びてきている。

これらのデータを扱う手段の１つとしてJ. Dean等の論文[1]をベースにオープンソフトウェア（Open Source Software：OSS）として実装されたHadoop[注1]がある。Hadoopは複数マシンでの分散処理を実現でき，従来のデータベース，データウェアハウスでは取り扱うことができなかったデータ量を扱うことができる。さらに，分析の用途が増加するにつれ，Hadoopクラスタを共用するニーズが発生したが，既存のHadoopディストリビューション（Hadoop本体と利用に必要なアプリケーション等をまとめたもの）では用途ごとのリソース管理に幾つかの課題があった。

三菱電機インフォメーションシステムズ㈱（MDIS）と㈱ノーチラス・テクノロジーズは，高速化や高信頼性に加えてマルチテナント運用向け機能を持つHadoopディストリビューションであるMapR[注2][2]を用いて複数用途を共存させるマルチテナント利用を実現する大規模データ分析共用基盤の構築に取り組んだ。

専有サーバを割り当てる構成（分離構成），又はマルチテナントで共有する構成（共有構成）かを選択可能とするなど，テナント要件に沿って柔軟な構成を実現している。また，ログ運用，アプリケーションの制限ルールの設定，クラスタ拡張のための情報収集などの運用方針を整理した。MDISでは，この実績を基に，2015年６月から“MapRクラスタ構築サービス”の提供を開始している。

（注１） Hadoopは，The Apache Software Foundationの登録商標である。
（注２） MapRは，MapR Technologiesの登録商標である。



マルチテナント対応大規模データ分析共用基盤

大規模データ分析基盤をマルチテナント対応として構築したシステム構成図である。テナントごとにクラスタの一部を占有又はマルチテナントによるリソース共有といった，テナントの要件に沿った柔軟な構成をとることが可能なシステムとなっている。

*三菱電機インフォメーションシステムズ㈱
**㈱ノーチラス・テクノロジーズ

1. まえがき

近年，コンピュータシステムが扱うデータ量は増加の一途をたどっている。今までは処理しきれずに捨てていたデータを活用し，新たな価値を生み出すことが，ビッグデータの取組みとして脚光を浴びてきている。その基盤として複数マシンでの分散処理を実現するHadoopの構築を行う企業が増加している。

サーバ数が数十台，データ量が数百TB(TeraByte)の大規模データ分析基盤構築に際して，企業内の複数の部門が共用すること(マルチテナント運用)によるコスト削減やリソースの有効活用が求められるが，既存のHadoopディストリビューションでは複数アプリケーション間のリソースの競合を抑制する機能や，利用部門(テナント)ごとのディスク使用量を制限する機能が不足しており，マルチテナント運用を行うことが困難であった。

この問題を解決するため，MDISでは既存Hadoopと完全互換で，マルチテナント運用向け機能(以下“マルチテナント機能”という。)を持つHadoopディストリビューションであるMapRを用いてマルチテナント利用を実現する大規模データ分析共用基盤を構築した。

本稿では，従来のHadoopディストリビューションでの課題について述べ，その課題を解決するために実現すべきマルチテナント機能を定義した上で，その実現に向けたマルチテナント機能設計のポイントを述べる。さらに，運用上の留意事項についても述べる。

2. 従来の大規模データ分析基盤の課題

既存のHadoopディストリビューションでは，次に挙げる4つの課題があり，複数部門で大規模データ分析基盤を共用するマルチテナント運用を行うことが難しい状況にあった。

(1) テナントごとのディスク使用量制限機能の欠如

テナントごとのディスク使用量を制限できず，特定テナントによる大量データ利用が他テナントに影響を及ぼす可能性がある。

(2) テナント間でのリソースの競合による処理の遅延

クラスタを構成する全てのサーバに分散してデータを蓄積し，全てのサーバで処理を実行する構成のみの実装となっている。この方式は，全サーバのリソースを有効に活用できる点では有利であるが，複数処理の同時実行によってリソースの競合が発生する可能性があり処理時間の見積りが難しい。確実に規定時間内に処理を完了させるためには，リソースの競合が発生しないようにテナント間で利用サーバを分離する必要があるが，既存のHadoopディストリビューションでは，利用サーバを分離するために複数のHadoopクラスタを構築する必要があった。

(3) 耐障害性の不足

多くのテナントでの利用を展開するには，極力システム停止を伴わない運用が必要であり，耐障害性の向上は必須である。既存のHadoopディストリビューションの標準構成では単一障害点が存在するため，Hadoopディストリビューション以外のミドルウェアなどを利用して，独自に多重化構成を構築する必要があった。

(4) Hadoop固有APIのみの提供

既存のHadoopディストリビューションでは大規模データを扱うAPI(Application Programming Interface)としてHDFS(Hadoop Distributed File System)(注3)インタフェースしか提供されず，データアクセスの処理を新たに作りこむ必要があった。

(注3) Hadoopが利用している分散ファイルシステムである。OS(Operating System)のファイルシステムを代替するものではなく，その上に独自のファイル管理システムを構築するもので，アプリケーションからローカルファイルと同様のインタフェースでアクセスすることはできない。

3. 目標とするマルチテナント機能拡張要件

Hadoopの商用ディストリビューションであるMapRは，高速化や高信頼性に加えてマルチテナント機能を持っていることを特長としている(2)。MapRを用いて，次に挙げる3つのマルチテナント機能拡張要件を満たすことを目標として，従来の課題を解決することに取り組んだ。

(1) マルチテナントに対して，テナントごとにデータ管理機能を持たせて，かつ利用サーバを分離できるHadoopシステムを提供する。

(2) マルチテナント機能を，単一クラスタ上に構築する。

(3) データアクセスの分離とテナント間のリソースの利用をMapR及びLinux(注4)の標準機能で構築する。

(注4) Linuxは，Linus Torvalds氏の登録商標である。

4. マルチテナント機能設計のポイント

4.1 前提条件の設定

まずはマルチテナント機能を利用する範囲や利用するユーザーを定義した。システム設計・運用設計で要件の発散を排除するために重要なプロセスであるため，設計の初期段階で検討を行った。

(1) マルチテナント機能を利用する範囲の設定

今回構築したシステムは，社内利用であり，部門やプロジェクトごとに1テナントずつを割り当てる想定で設計した。社内ネットワークで接続され，ユーザーは提示されたルールを守ることを前提としたため，ルール無視のユーザーの誤使用を排除する対策等は実施していない。

(2) マルチテナント運用関係者の役割の明確化

マルチテナント運用関係者の役割(ロール)とそれぞれが利用できる機能を定義した。

具体的には設定ファイルの変更やリソースの払出しが可

能なインフラ担当者，クラスタの通常運用を行うMapR運用者，テナント側の接続環境を構築するテナントインフラ担当者，ファイルI／O(Input／Output)やアプリケーション実行が可能なテナント利用者である。

4.2　マルチテナント機能の実現方式

2章で述べた課題(1)～(4)を解決するために採用したマルチテナント機能実現のための具体的な方式を次に述べる。

4.2.1　テナントごとのデータアクセス制限の設定

テナントごとに利用可能なディレクトリを割り付け，アクセス権を適切に設定することによってテナント間のデータアクセスを分離した。既存のHadoopディストリビュージョンでもこのデータアクセスの分離までは実現可能であるが，クラスタ全体のストレージを1つのファイルシステムとして扱っているため，テナントごとのデータ使用量を制限することはできなかった。MapRでは“ボリューム”という概念でファイルシステムを分割管理しており，ボリュームごとにディスク使用量の上限値を設定できる[(2)]。これによって，ボリュームをテナントごとのディレクトリに割り付けることでテナントごとのデータ使用量を制限することを可能とした。また，MapRでは，ボリュームごとにスナップショットを生成する機能を持っており[(2)]，ユーザーがデータ処理で誤操作をした場合の復旧手段としてテナントごとにこの機能を提供することとした。

4.2.2　テナント間の利用サーバの分離

従来のHadoopでは，各テナントの利用サーバを分離するためには複数のクラスタに分割した構成にする必要があった。次に示すように，MapRの持つ機能を適切に組み合わせて活用することによって，単一のクラスタ構成で利用サーバ間の分離を柔軟に行うことを可能とした。

(1)　分離構成と共有構成の定義

今回構築したクラスタでは，リソースを割り当てる構成として次の2種類の構成を定義した。

①分離構成

1つのテナントに対して専有サーバを割り当てる構成である。他テナントとのリソース競合を回避できる構成であり，応答性能の要件が高い場合に選択する。

②共有構成

マルチテナントでサーバを共有する構成である。リソースを効果的に活用できるが，競合によって性能面の保証ができなくなる。応答性能の要件が高くない場合に選択する。

テナントごとにこれらを選択でき，かつ共存できるよう設計した。利用するアプリケーションの要件に応じてどちらを選択するかを決定する。

分離構成と共有構成の比較を**表1**に示し，それぞれのイメージを**図1**に示す。

(2)　分離構成の実現方式

分離構成を実現するには，テナント間で，データの分離と処理の分離を実現する必要がある。構築したシステムでは，MapRの“ボリュームトポロジー”機能と“ラベル”機能[(2)]を組み合わせることによって，分離構成を実現している。

表1．分離構成と共有構成の比較

項目	分離構成	共有構成
他テナントへの影響	ほとんどない	性能，容量面で影響がある
小規模なリソースの割当て	最低でも3台分のサーバリソースを割り当てる必要がある。小規模なリソース割り当ては不可	小規模なリソースを割当て可能
テナント追加	専有サーバを確保する必要があり，共有構成と比べて工数と時間が大きくかかる	既存リソースに余裕があれば，分離構成と比べて容易
クラスタ構成	1クラスタ(注5)	1クラスタ

(注5)　既存のHadoopで分離構成相当の機能を実現するには，テナント間のリソース競合を回避するためにテナントごとに別々のクラスタを構築する必要があった。この設計では1クラスタ構成でテナント間のリソース競合を回避できるため，既存のHadoopに比べコスト削減となる。

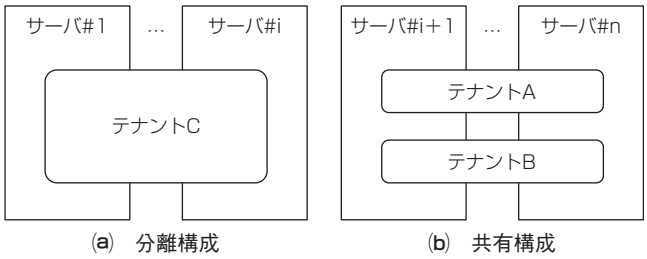


図1．分離構成と共有構成

①データの分離

“ボリュームトポロジー”機能を用いてデータの分離を行った。ボリュームトポロジーは，ボリュームのデータをどのサーバに格納するかを設定する機能である。各テナントのデータの保存位置を分離したサーバ内に限定することで，ディスクI／O，ネットワークI／Oの競合による性能劣化を防止する。

②処理の分離

“ラベル”機能を用いて処理の分離を行った。ラベルは，アプリケーションを実行するサーバをグループ化する機能である。テナントごとに，アプリケーションを実行するサーバを特定ラベルを付けたサーバに限定することでCPU(Central Processing Unit)，メモリの競合による性能劣化を防止する。

ボリュームトポロジーとラベルによるデータ・処理分離イメージを**図2**に示す。

先に述べたように，この設計での分離構成では，MapRの設定を変えることによってクラスタ内の物理的なサーバ台数を変化させずに，テナントが利用するサーバ台数を変化させることができる。この妥当性を確認するために，分離構成の設定によって利用可能サーバの台数を変化させた場合と，物理的にクラスタ内の接続サーバ台数を変化させた場合での

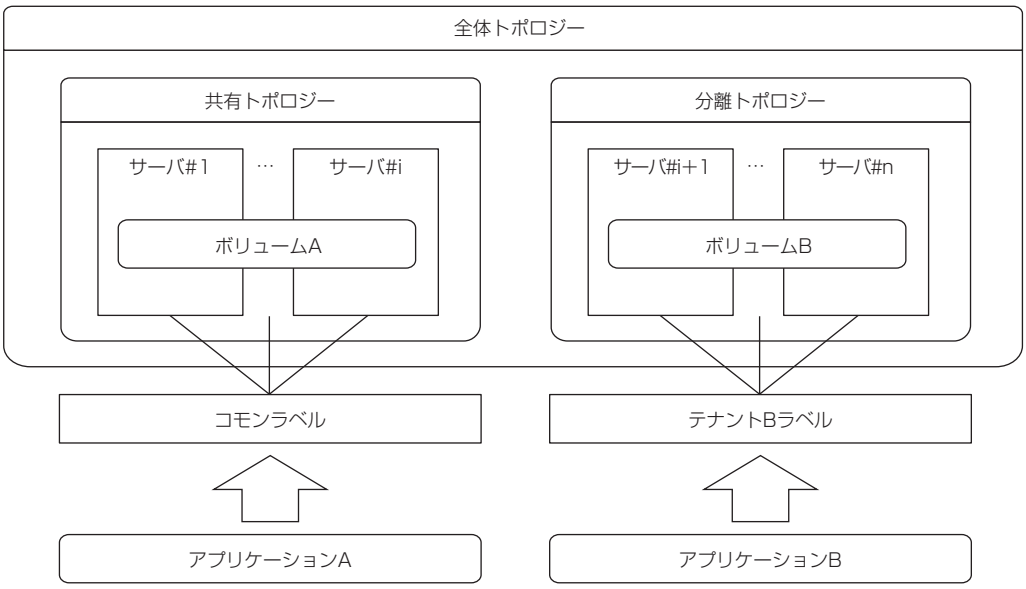


図2．ボリュームトポロジーとラベルによるデータ・処理分離

ベンチマークテストプログラムの応答性能の比較を行った。その結果，両者でほぼ同等の結果を得ることができ，分離構成によって有効にリソース分離ができていると考えられる。

4.2.3　耐障害性の向上

マルチテナントに対して安定した運用を提供するために，三重化冗長構成として二重までの障害に耐えうる構成としている。2台以下のサーバ，2台以下のハードディスク，2つ以下の管理サービスの同時障害が発生してもシステム停止が起こらないように設計した。これは，ファイルのレプリカ数(多重度)を“3”にすること，各管理サービスを三重化構成にすることによって実現している。前者は既存のHadoopから持つ機能，後者はMapRが提供する機能である[2]。また，管理サービスの配置設計では，各管理サービスを別ラックのサーバに分散させて1ラック全体障害発生時のシステム停止を回避するために，分離構成を構築するサーバでは管理サービスを動作させない構成として，分離構成利用テナントと共有構成利用テナント間で管理サービスに伴う影響を最低限にする等の構築上の工夫を行っている。

4.2.4　APIの拡張

この実現方式では，ファイルシステムに対し，従来のHDFSインタフェースに加えてNFS(Network File System)(注6)で接続する機能を提供している[2]。UNIX(注7)ファイルシステムと同様の操作でデータ入出力することができ，多くの既存アプリケーションをそのまま利用することが可能である。

NFSの構築では，HDFSインタフェースと同様にアクセス権を設定してテナント間のアクセス権の分離を実現している。また，NFSゲートウェイサービスを行うサーバをテナントごとに分散して割り付け，それぞれのマウントポイントに接続可能なクライアントマシンを制限することで負荷分散とセキュリティ向上を実現する設計とした。

(注6)　主にUNIX系OSで利用される分散ファイルシステムとそのプロトコルである。サーバのストレージをネットワーク経由でマウントすることによってローカルファイルと同様にアクセスすることができる。
(注7)　UNIXは，The Open Groupの登録商標である。

5.　マルチテナント運用上の注意点と対応策

5.1　ログの運用

各テナントのアプリケーション実行における障害発生時等の問題の解析・解決のためにテナントへログを提供する。ログ提供でもテナント間のアクセス権の分離が必要であるが，テナント間のアクセス権の分離ができないログもあり，ログの種類によって次の運用とした。

(1)　OSのログ，MapRデーモンのログ

テナントごとの分離が不可能なため，テナントにはそれらのログは公開せず，テナントの要請によってシステム管理者側でログ内容を確認する運用とした。

(2)　アプリケーション実行時のログ

他テナントがログをアクセスできないようにアクセス権を設定した上で各テナントから参照可能とした。なお，ログはクラスタ内で散在しており参照処理が煩雑であるため，MapRの持つ集中ロギング機能[2]を活用して，テナントへの便宜を図っている。

5.2　テナント運用形態の選択

新たにテナントを追加する際，テナント間のリソースの分離を適切に実現するために，分離構成とするか，共有構成とするかを決める必要がある。

追加テナントに対して，リソース容量条件，利用する機能，アプリケーションの特殊要件等のヒアリング項目を明確にし，どちらの利用形態が適しているかを判別できるようにした。基本的には他テナントへの，又は他テナントか

らの影響度によって判断する。

5.3 共有構成における運用方針

(1) クラスタのチューニング方針

共有構成ではどのようなアプリケーションが動作するかの予測が不可能であるため，パラメータは最も汎用性があるデフォルト値を採用した。なお，個別チューニングによる応答性能の向上が必要なテナントについては分離構成を選択することでパラメータのチューニングを可能としている。

(2) アプリケーションの制限ルールの設定

共有構成では，サーバのリソースを有効に利用できるメリットはあるが，サーバを共有するテナント間でのリソースの競合が発生する可能性が高くなる。通常は，Hadoopの機能でリソース利用はテナント間で適切に制御されるが，アプリケーションの利用方法によってはHadoopでは制御しきれないケースがある。そのため，他テナントに影響を与えないようにするアプリケーションの制限ルールを定めた。具体的には，Linux上の別アプリケーションを起動する場合の使い方の制限，一時ファイル等の出力先としてのLinuxローカルのファイルシステムの利用禁止等である。

5.4 アラートの監視

MapRでは問題を検知した際にアラートを発生させるが，そのアラートが特定テナントにのみ影響があるのか，又はクラスタ全体に影響があるのかを判別することが難しい。

発生する可能性のあるアラートをリスト化し，運用時に影響範囲を判断できるようにガイドを作成した。

5.5 クラスタの拡張方針の策定

マルチテナント運用を継続するには，運用中のテナントの処理量・データ量の増加や新規テナントの追加に備えてクラスタの拡張を検討する必要がある。拡張検討のために，日々の運用で，監視対象とするリソースと拡張基準を策定した。

主な監視対象は次の5点であり，定期的に監視することとした。なお，これらが拡張基準となるしきい値を超えた際には，クラスタの拡張を検討することとしている。

(1) CPU利用率(サーバごと)
(2) メモリ利用率(サーバごと)
(3) ディスクI／O(サーバごと)
(4) ネットワークI／O(サーバごと)
(5) ファイルシステム利用率
(全体，サーバ，ボリュームごと)

6. むすび

今回，MapRを利用して，データ用ディスク容量が数百TB規模のマルチテナント向け大規模データ分析共用基盤を設計・構築することができた。これまでは，この規模のシステムを単一テナントでしか利用できなかったが，このマルチテナント向け共用基盤を用い，ログの分析システムとIoT(Internet of Things)関連データ処理システムの2つの大容量データ処理テナントを同一プラットフォームで稼働させる運用を2015年4月から開始している。また，このノウハウを活用して2015年6月から“MapRクラスタ構築サービス”を開始している。

今後，今回開発した大規模データ分析共用基盤の技術をベースとして様々な適用分野を探っていくとともに，データ分析のためのアプリケーションの拡充を図っていく予定である。

参考文献

(1) Dean, J., et al.：MapReduce：Simplified Data Processing on Large Clusters, Proc. of Operating System Design and Implementation(OSDI), 137～150 (2004)
http://static.googleusercontent.com/media/research.google.com/ja//archive/mapreduce-osdi04.pdf
(2) MapR公式ドキュメント
http://doc.mapr.com/display/MapR3/Home