HITACHI
Inspire the Next

OSC2016 Tokyo/Fall

MongoDBで試すシャーディングの実力検証！

2016/11/05

株式会社 日立製作所
OSSソリューションセンタ

HITACHI
Inspire the Next

Contents

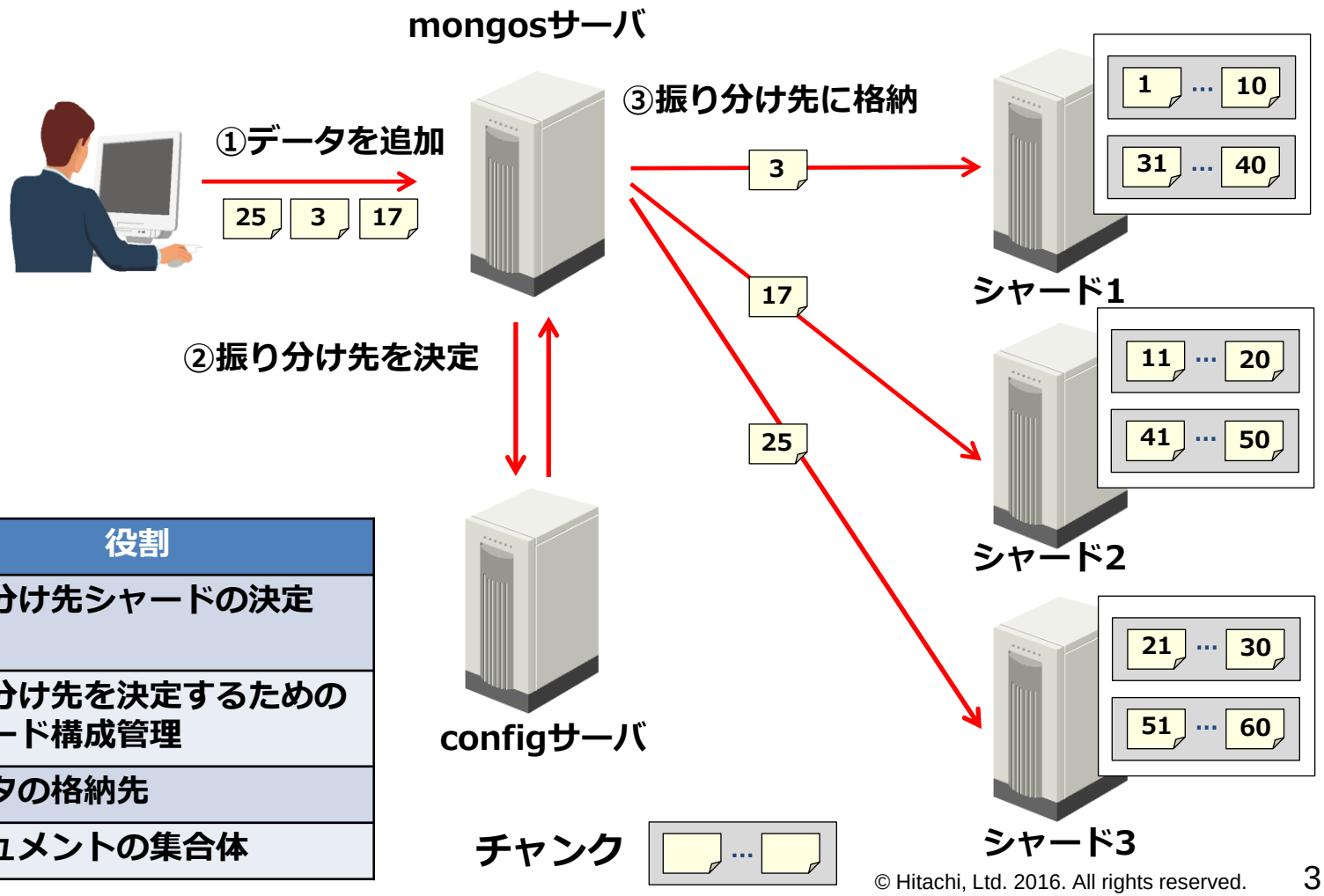
HITACHI
Inspire the Next

1．シャーディングとは？

1. シャーディングとは？

HITACHI
Inspire the Next

複数サーバにデータを配置することで負荷分散する機能です。
これにより大量データに対するスループットを向上します。



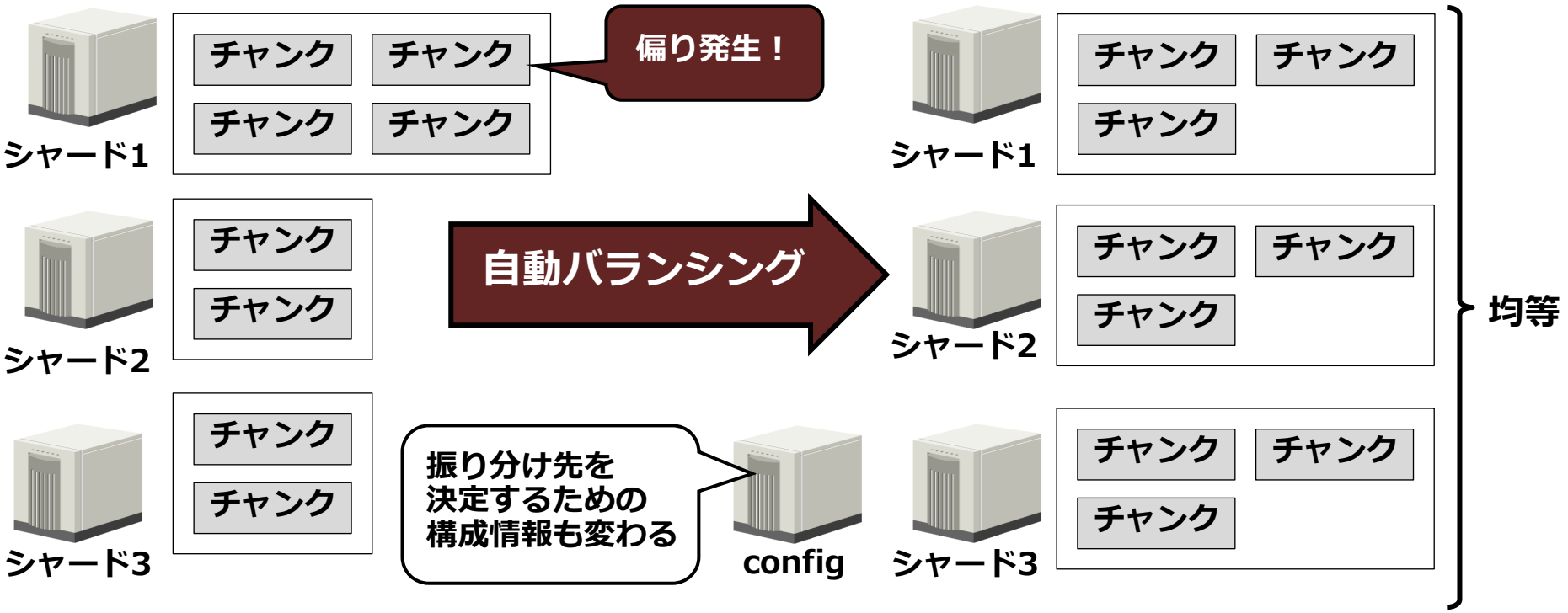
構成要素	役割
mongos サーバ	振り分け先シャードの決定
config サーバ	振り分け先を決定するための シャード構成管理
シャード	データの格納先
チャンク	ドキュメントの集合体

1. シャーディングとは？

HITACHI
Inspire the Next

【分散配置のルールと自動バランシングについて】

- 振り分け先の決定は値の範囲（レンジ）で決まります。その範囲のルールはデフォルトでは**MongoDBが自動的に決定**します。（設定も可能）
- データの配置に偏りが発生した場合はMongoDBの「自動バランシング」により自動的に均等に分配され、振り分け先決定ルールも暗黙的に変わります。
  ユーザが分散配置を調整する必要はありません。



HITACHI
Inspire the Next

2. 性能評価

2. 性能評価

HITACHI
Inspire the Next

シャーディングの実力を測るべく性能評価を実施しました。
評価項目は以下の通りです。

項番	測定内容	ねらいと実施項目
1	mongoimportによる データー括作成	シャーディングによる性能劣化は無いか？ 自動バランシングのオーバーヘッドはどの程度か？ 単体構成とシャーディング構成で比較
2	レスポンス スループット	負荷分散の効果は？ シャード数に応じて本当にレスポンス／スループットは向上するのか？ 単体構成と各種シャーディング構成で比較

検証環境

項番	項目	スペック
1	マシン	モデル：HA8000-bd/BD10 CPU：Core i7-610E 2.6GHz HDD：160GB メモリ：8GB
2	OS	Red Hat Enterprise Linux Server 7.2
3	MongoDB	バージョン ： 3.2.9 ストレージエンジン ： WiredTiger

2. 性能評価

HITACHI
Inspire the Next

【検証データ】
測定には、Webサーバのログを想定したサンプルデータを使用します。

```
"date","host","IdInfo","UserID","RequestDate","Request","StatusCode","size"
"20000107", "192.xxx.0.71" ,"-", "-", "[07/Jan/2000:10:45:36 +0900]", "GET /favicon.ico HTTP/1.1", ～
"20000107", "192.xxx.0.42" ,"-", "-", "[07/Jan/2000:10:45:36 +0900]", "GET /favicon.ico HTTP/1.1", ～
            :
"20081025", "192.xxx.0.43" ,"-", "-", "[08/Oct/2008:13:22:08 +0900]", "GET /favicon.ico HTTP/1.1", ～
            :
"20130716", "192.xxx.0.41" ,"-", "-", "[13/Jul/2013:09:10:25 +0900]", "GET /old_html/ HTTP/1.1", ～
            :
```

上記を入力した際に１ドキュメントの構造は以下のようになります。

```
{
    "_id" : ObjectId("5805ba3cf80ed40e2486a50b"),
    "date" : 20000107,
    "host" : "192.xxx.0.71",
    "IdInfo" : "-",
    "UserID" : "-",
    "RequestDate" : "[07/Jan/2000:10:45:36+0900]",
          :
}
```

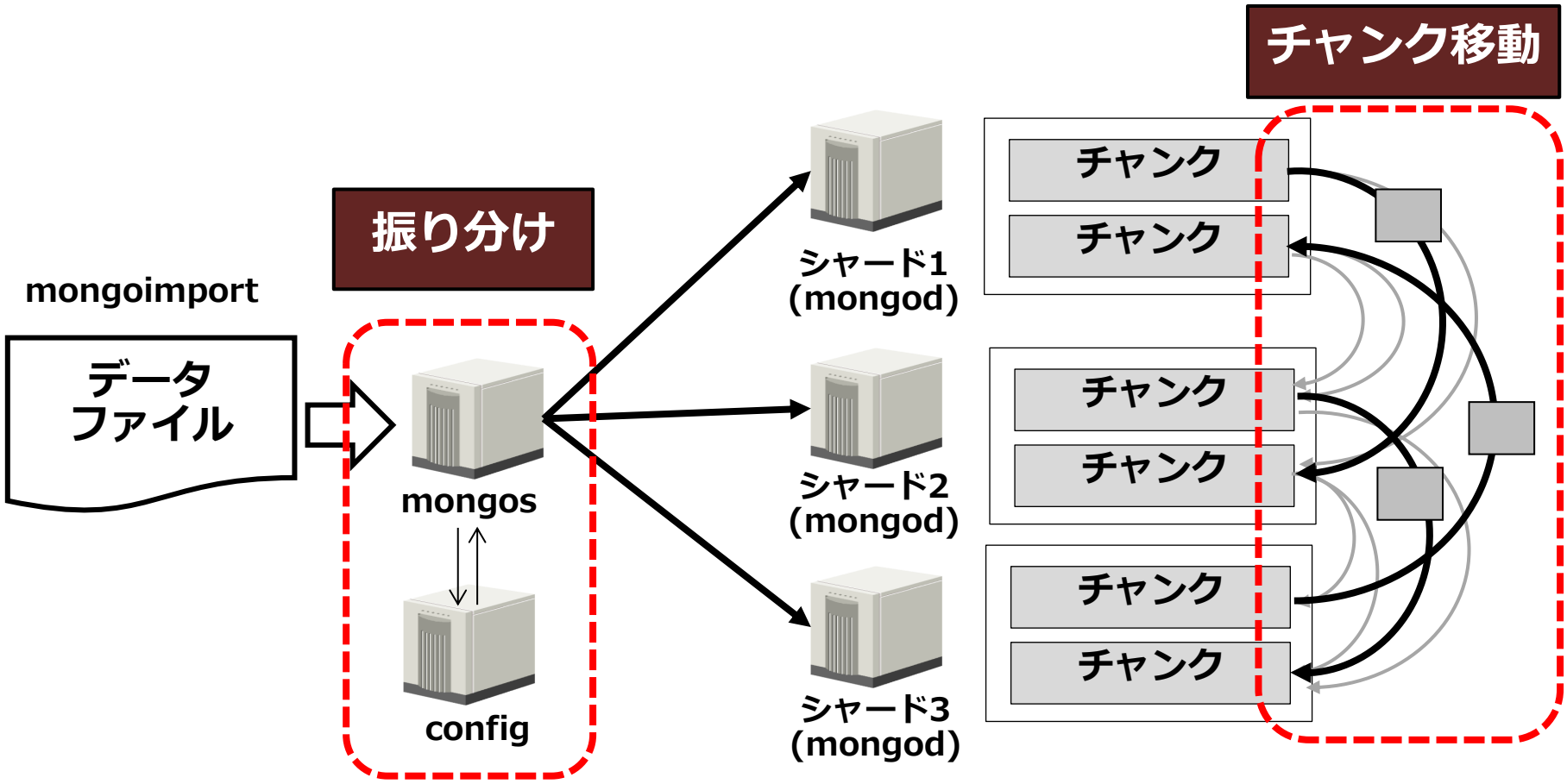
⇒自動付加フィールド
⇒年月フィールド
⇒アクセスユーザのIPアドレス

【シャードキーの設定】
カーディナリティを高めるため「"host"+"date"」の複合キーを設定します。

2.1 mongoimportによるデーター括作成

HITACHI
Inspire the Next

【データー括作成時の性能懸念事項】
シャーディング構成ではmongoimportによるデーター括作成時も
振り分け、および自動バランシングによるチャンク移動が発生します。
それらのオーバーヘッドがどの程度かは認識しておきたいところです。

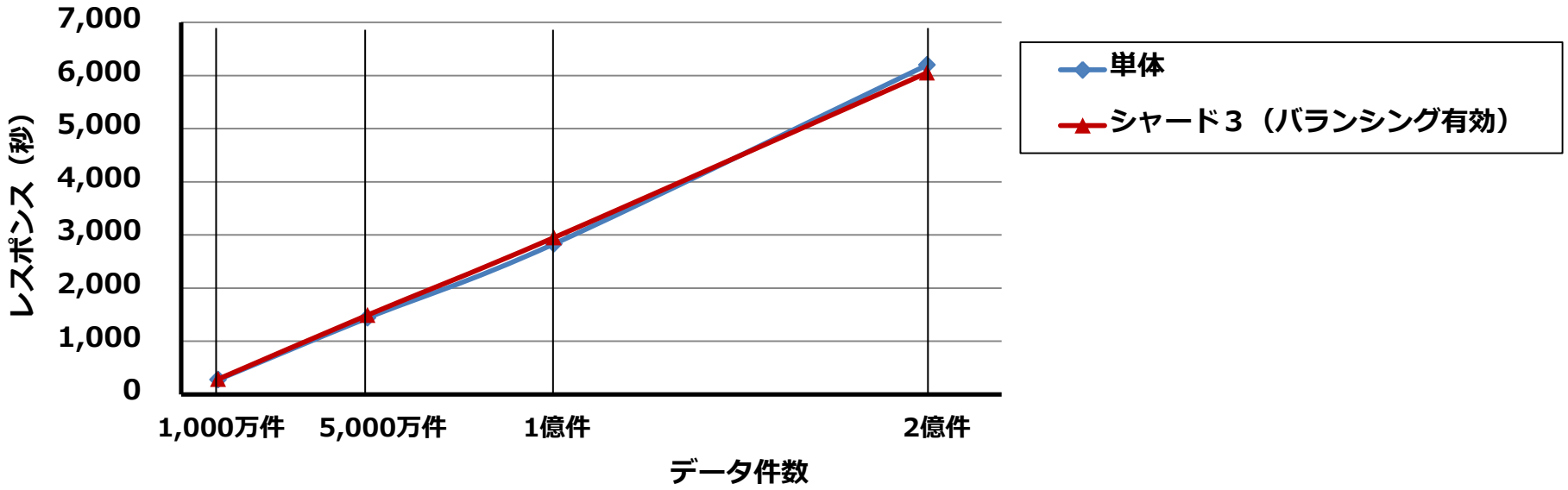


2.1 mongoimportによるデーター括作成

HITACHI
Inspire the Next

mongoimportによるデーター括作成の処理時間を測定

構成＼件数	1,000万件	5,000万件	1億件	2億件
単体（mongosなし）	278 秒	1,440 秒	2,832 秒	6,197 秒
シャード数3 （下段はチャンク移動回数）	288 秒	1,492 秒	2,951 秒	6,052 秒
	29回 （平均2秒）	90回 （平均4秒）	127回 （平均4秒）	313回 （平均5秒）



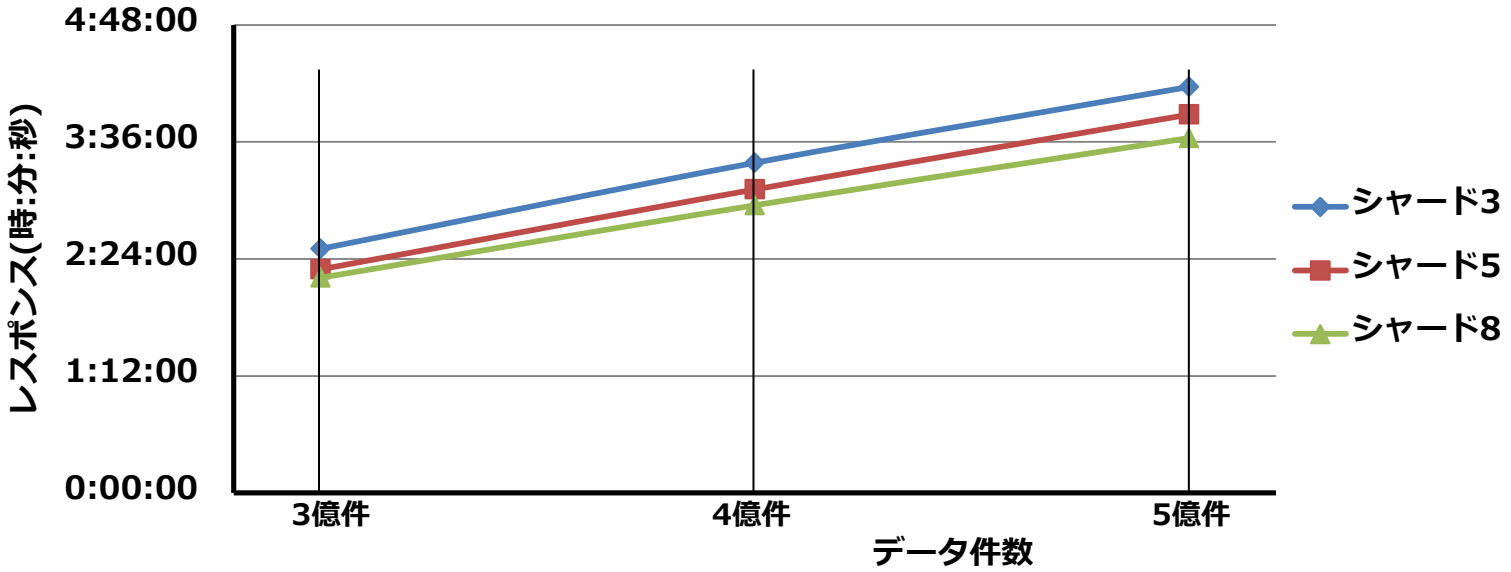
【結果】
シャード構成による性能影響は殆どありません。
件数に対する処理時間も線形に比例しており、自動バランシングに伴う
各種オーバーヘッドはあまり気にしなくて良いと考えます。

2.1 mongoimportによるデーター括作成

HITACHI
Inspire the Next

データー括作成処理時間とシャード数の関係を確認（データも増量）

構成＼件数	3億件	4億件	5億件
シャード数3	2:30:12	3:23:11	4:09:53
シャード数5	2:17:35	3:06:44	3:52:51
シャード数8	2:12:26	2:57:00	3:38:25



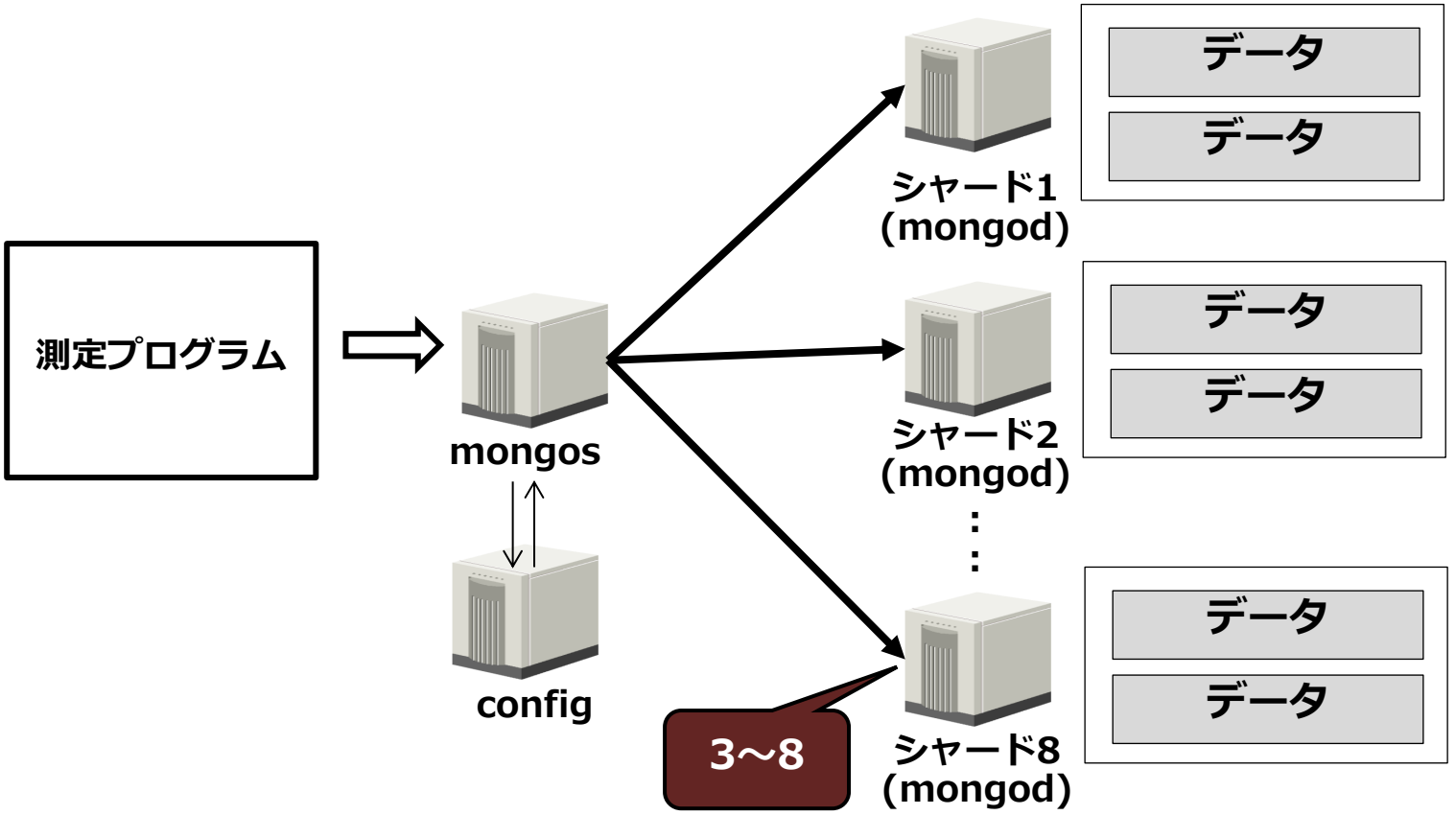
【結果】
データー括作成処理時間はシャード数に比例し短くなります。
先ほどと同様、自動バランシングに伴う各種オーバーヘッドはシャード数が増えてもデータ量が増えてもあまり気にしなくて良いと考えます。

2.2 シャード構成別検索性能

HITACHI
Inspire the Next

【仮説】
レスポンスはシャード数に比例して向上する。

【測定モデル】
- 全データ件数を3億～5億件
- 検索条件にhostを指定（ヒット率1%）
- シャード数は3,5,8、シャード5までI/Oあり、シャード8ですべてオンメモリ



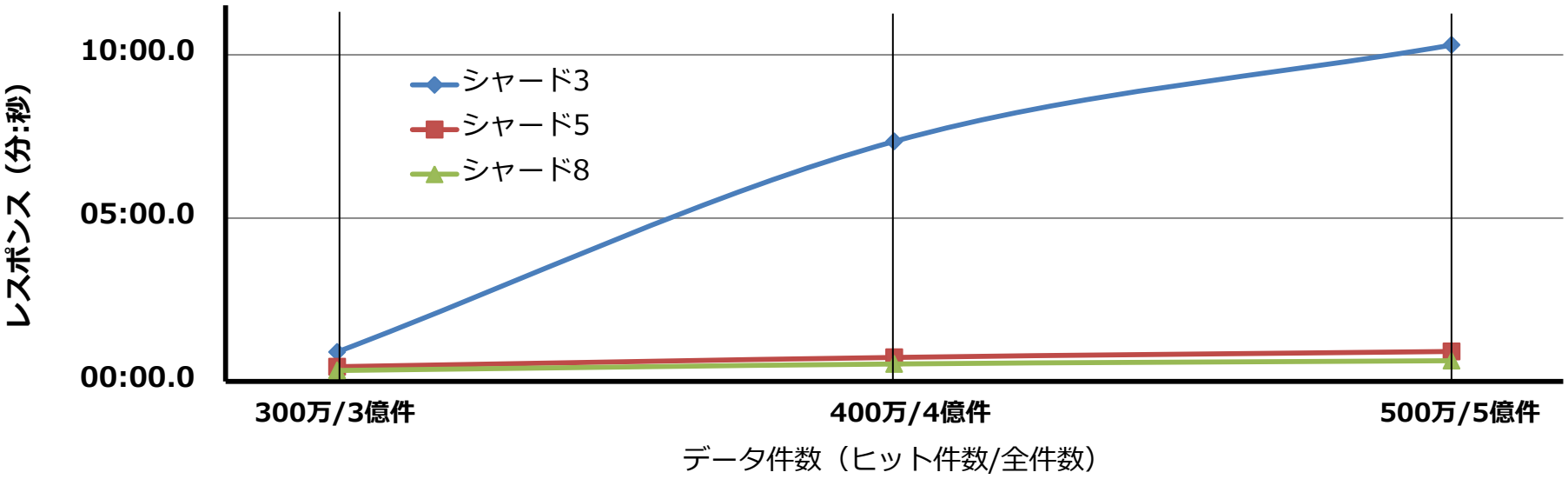
2.2 シャード構成別検索性能

HITACHI
Inspire the Next

シャード数3~8で条件検索

構成＼件数	3億件	4億件	5億件
シャード数3	00:54 （249回）	07:21 （171,000回）	10:18 （175,000回）
シャード数5	00:27 （0回）	00:44 （152回）	00:55 （456回）
シャード数8	00:20 （0回）	00:32 （0回）	00:38 （0回）

※（）内はディスク読込み回数。



【評価】
シャード数に応じてI/O量が低下し、レスポンスが向上することが確認できました。

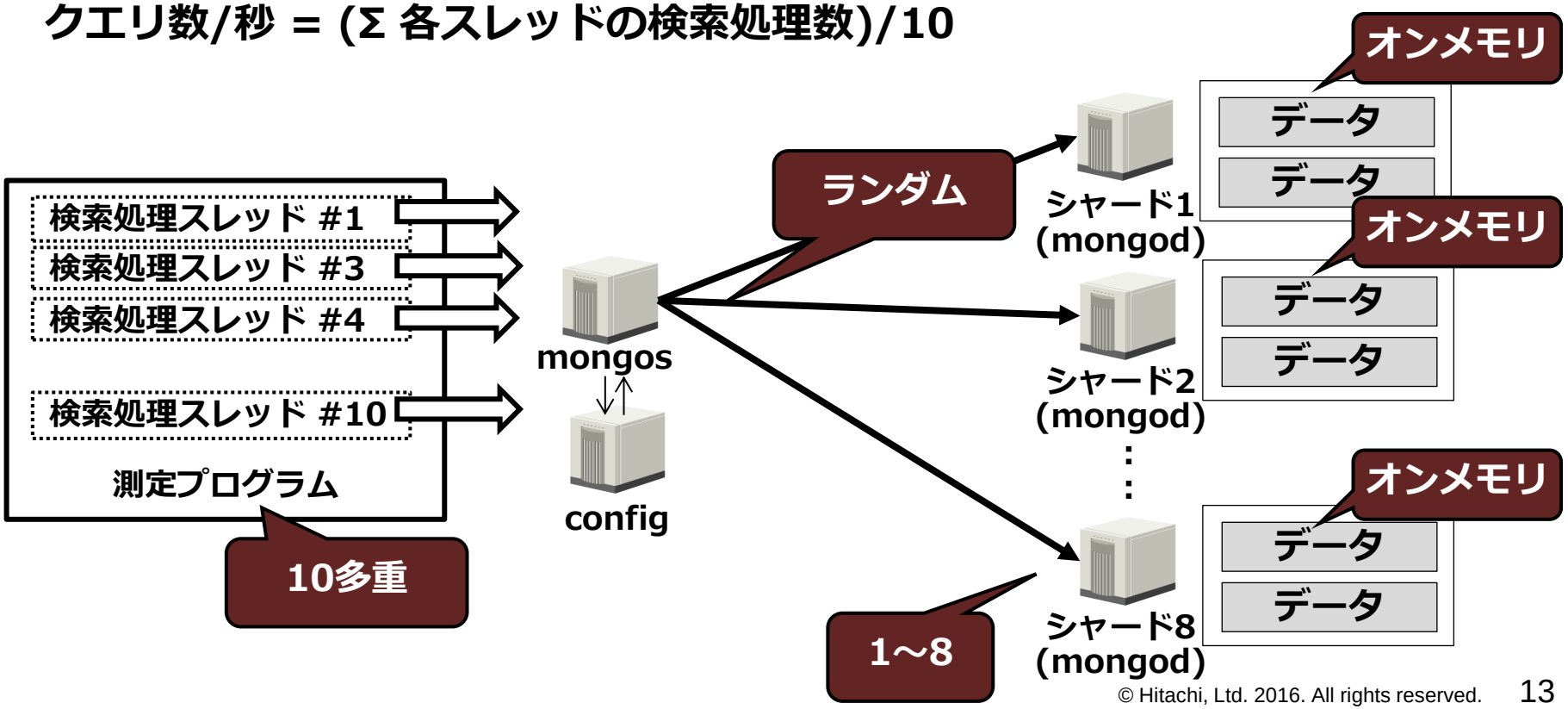
2.2 シャード構成別検索性能

HITACHI
Inspire the Next

すべてオンメモリの場合にスループットが向上するかを検証

【測定モデル】

- 測定プログラムから10多重で検索を実行
- 各シャードのデータはオンメモリ
  （全5000万件。純粋に負荷分散を評価したいのでI/Oなし）
- シャード数は1,3,5,8
- 検索条件はランダムで均等に。検索条件のhostもユニークなデータに変更。
- 検索処理スレッドはループで10秒検索を実行。スループットは下記で算出。
  クエリ数/秒 = (Σ 各スレッドの検索処理数)/10



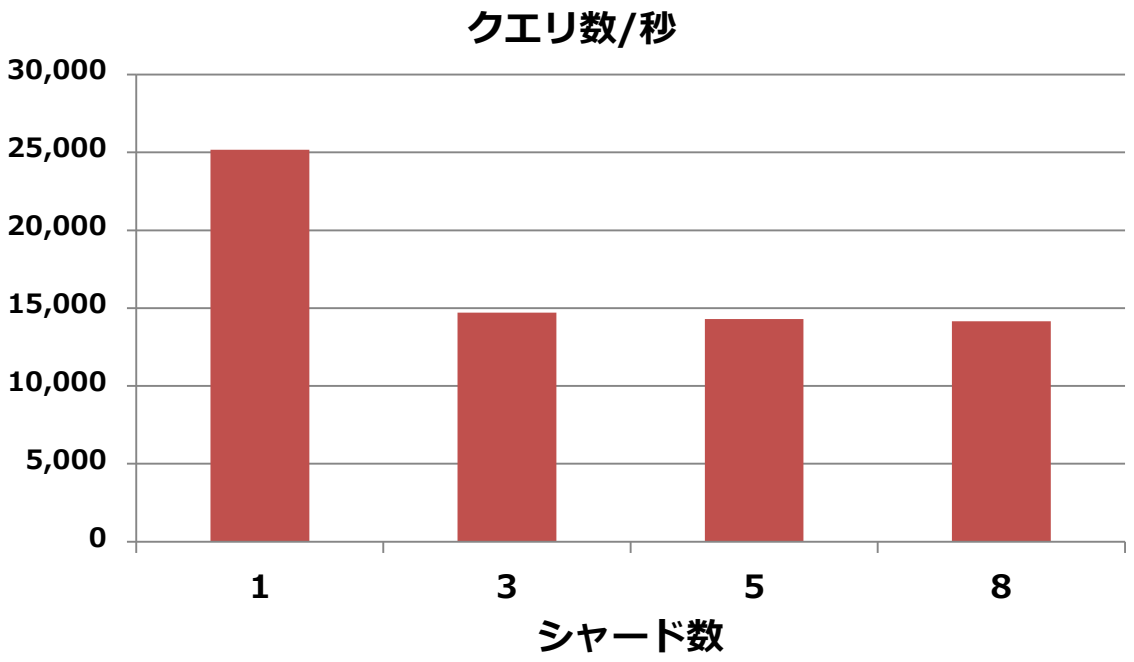
2.2 シャード構成別検索性能

HITACHI
Inspire the Next

シャード数1～8で条件検索

構成	クエリ数/秒
単体	25,161
シャード数3	14,716
シャード数5	14,289
シャード数8	14,147

単体はmongosなし



【結果】
シャード数3以上ではスループットが伸びない。

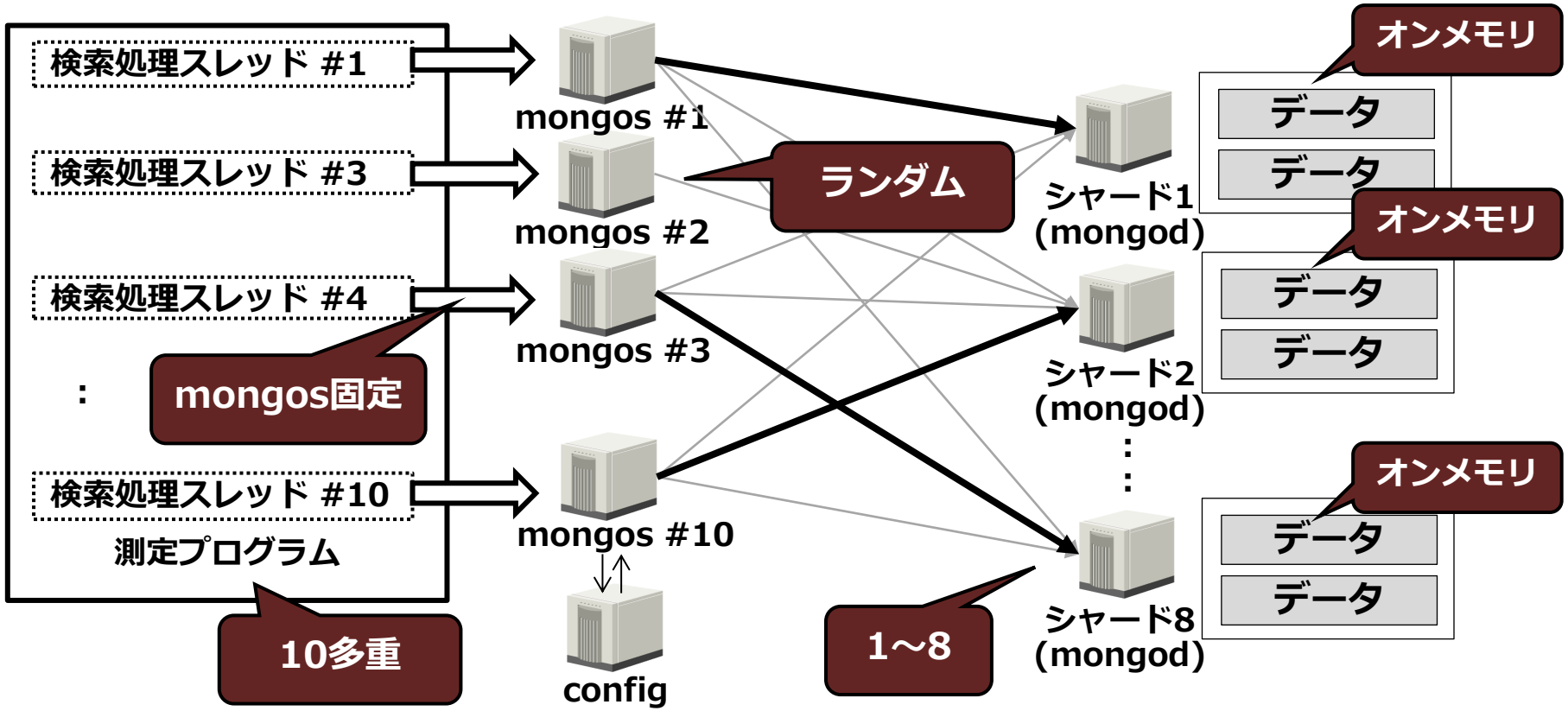
【考察】
mongosを1つしか起動しておらず、これがボトルネックと推定。
mongosを多重度と同じ10個起動して計測しなおし。

2.2 シャード構成別検索性能

HITACHI
Inspire the Next

(再)すべてオンメモリの場合にスループットが向上するかを検証

【測定モデル】
先ほどの条件に下記を追加。
- mongosを10個起動。測定プログラムの各多重で接続先mongosを固定。



2.2 シャード構成別検索性能

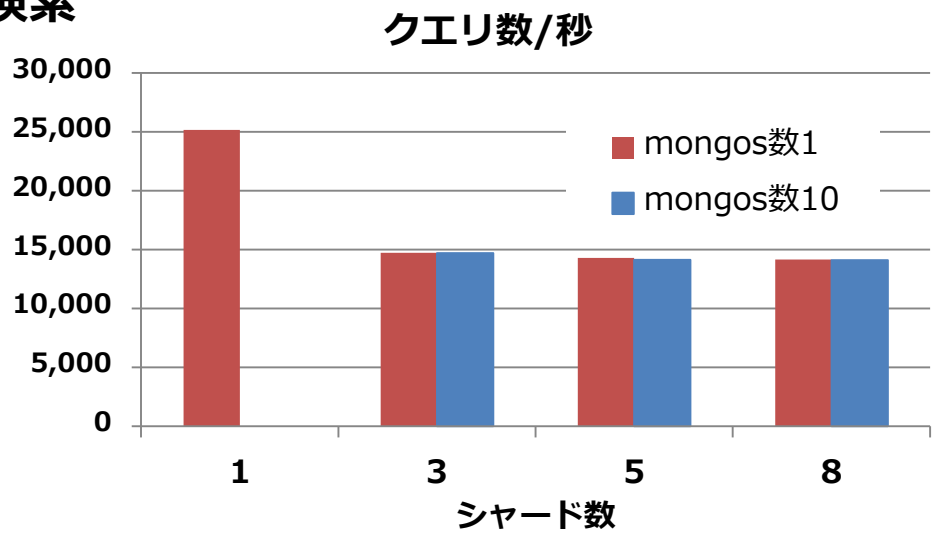
HITACHI
Inspire the Next

mongos数10、シャード数3～8で条件検索

構成	クエリ数/秒	
	mongos数1	**mongos数10**
単体	25,161	**-**
シャード数3	14,716	**14,705**
シャード数5	14,289	**14,142**
シャード数8	14,147	**14,109**

単体はmongosなし

変わらない



【結果】
mongosを増やしてもスループットが伸びない。

【考察】
各種確認の結果、1クエリの処理時間が単体とシャード構成で倍近い差があった。
これはmongosを経由するオーバーヘッドと考える。
スループット測定モデルは検索処理自体が非常に軽く、このオーバーヘッドが見えやすい。
この差がそのままスループットの差に出ている。
mongos経由の間延び分mongodには余裕があると思われ、負荷が足りてないかもしれない。

多重度を増やして再々測定します。

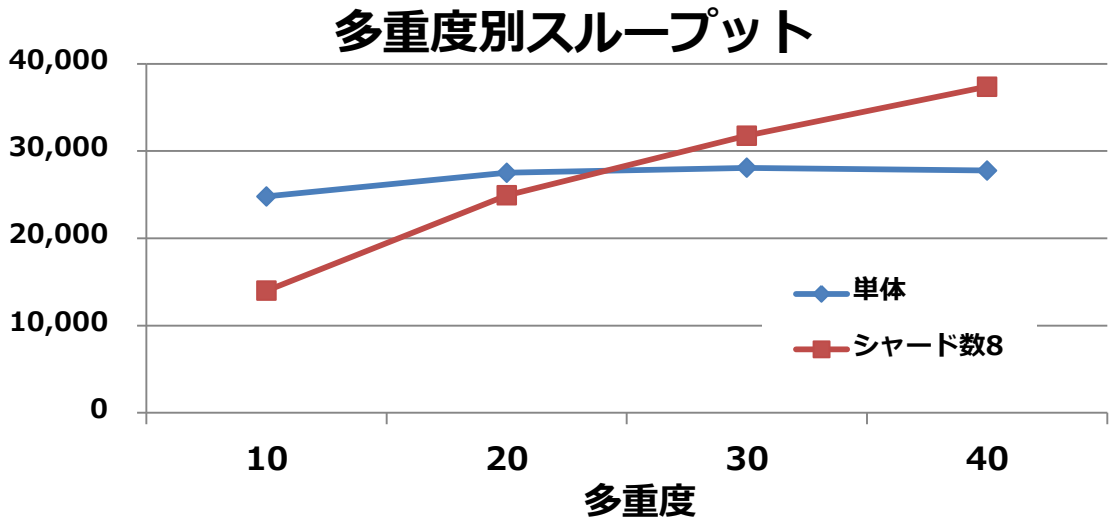
2.2 シャード構成別検索性能

HITACHI
Inspire the Next

mongos数1、シャード数8、多重度10～40で条件検索

多重度	単体	シャード数8
10	24,796	13,987
20	27,515	24,926
30	28,074	31,777
40	27,763	37,389

単体はmongosなし



【結果】
単体構成は30多重が限界、シャーディング構成では多重度に応じてスループットが向上。
やはり負荷不足であった。
mongos数10でも同様の傾向であった。

【結論】
高負荷な業務、あるいは検索処理単価が重いなど、業務パターンによっては
負荷分散としての効果がありそうである。

2. 性能評価

HITACHI
Inspire the Next

まとめ

項番	測定内容	結論
1	mongoimportによるデーター括作成	シャーディング構成での振り分け、自動バランシングのオーバーヘッドが性能面での懸念材料でしたが、シャード数やデータ量には依存せず、ほぼ無視できる程度であることが確認できました。
2	レスポンス スループット	I/Oが発生するケースではシャード数に応じてI/O量が分散され、レスポンスが向上することが確認できました。 すべてオンメモリで検索処理時間が短い場合、mongos経由のオーバーヘッドが大きくスループットが向上しないことが確認できました。 シャーディングを低負荷の業務や、I/O削減以外の負荷分散で使用する場合は注意が必要です。 低負荷で、全データが1台でオンメモリにできる場合はシャーディング構成にすべきではないと考えます。 一方で、負荷（クライアント多重度）が大きい場合には、シャーディング構成が有効です。

HITACHI
Inspire the Next

3. 運用管理

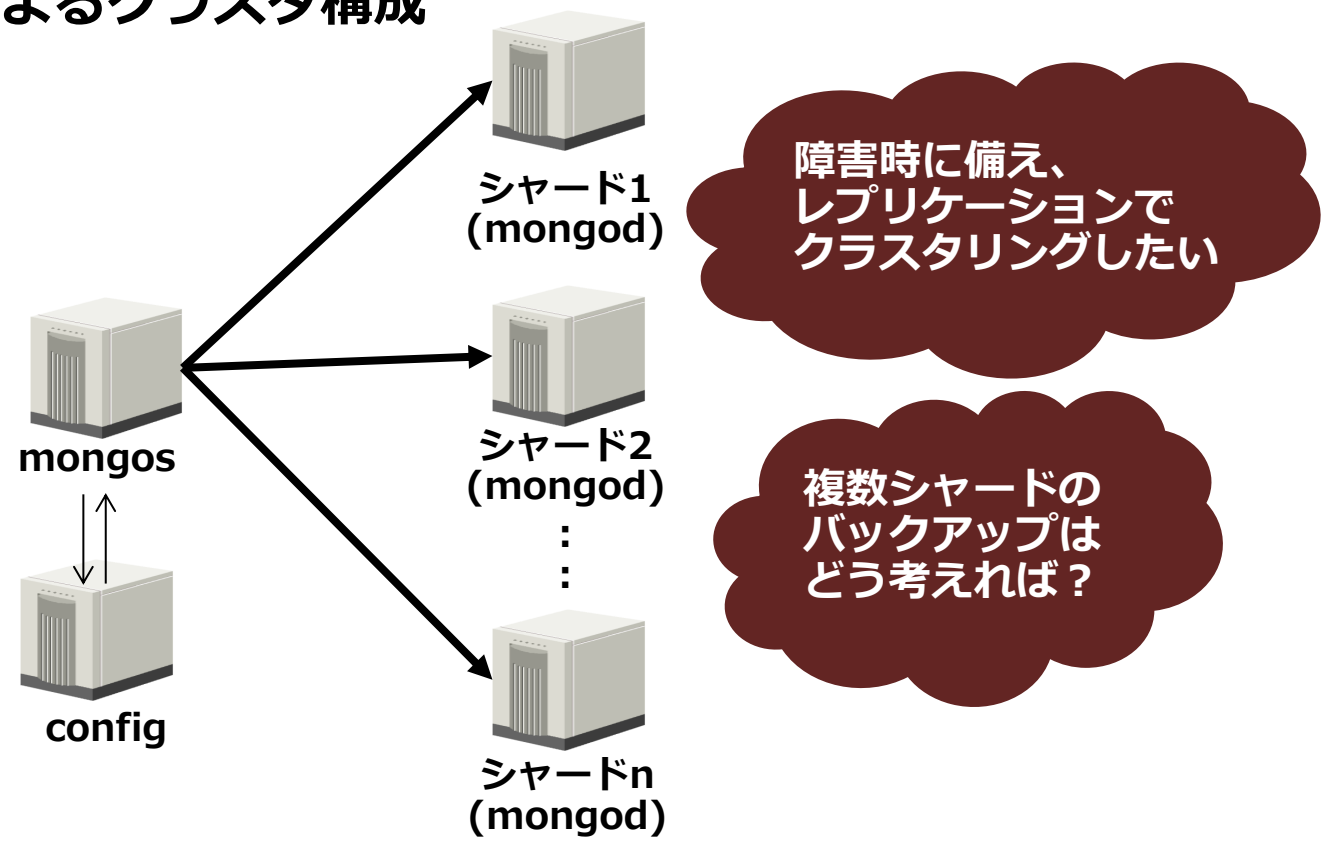
3. 運用管理

HITACHI
Inspire the Next

データー括作成や検索性能では期待する効果が見えましたが、シャーディング構成は複数マシンを運用管理する必要があるため操作も複雑になります。

下記運用管理のポイントをご紹介します。

- レプリケーションによるクラスタ構成
- バックアップと回復

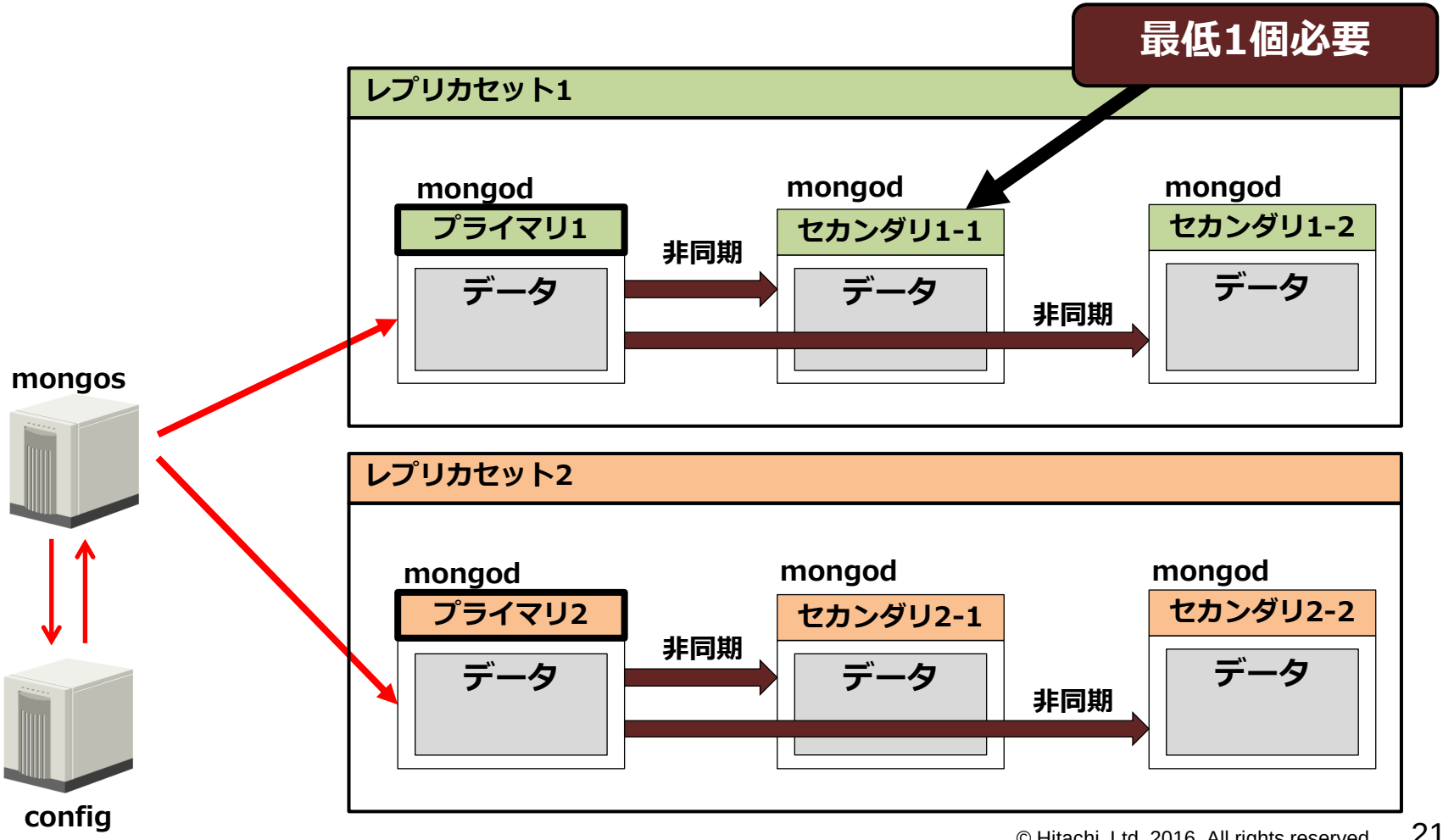


3.1 レプリケーションによるクラスタ構成

HITACHI
Inspire the Next

シャーディング構成ではシャード単位でレプリケーション

MongoDBではプライマリノード1個、セカンダリノード1個以上の「レプリカセット」という論理的なグループで構成します。
プライマリノードが書き込みを受け取り、セカンダリノードに非同期で反映します。

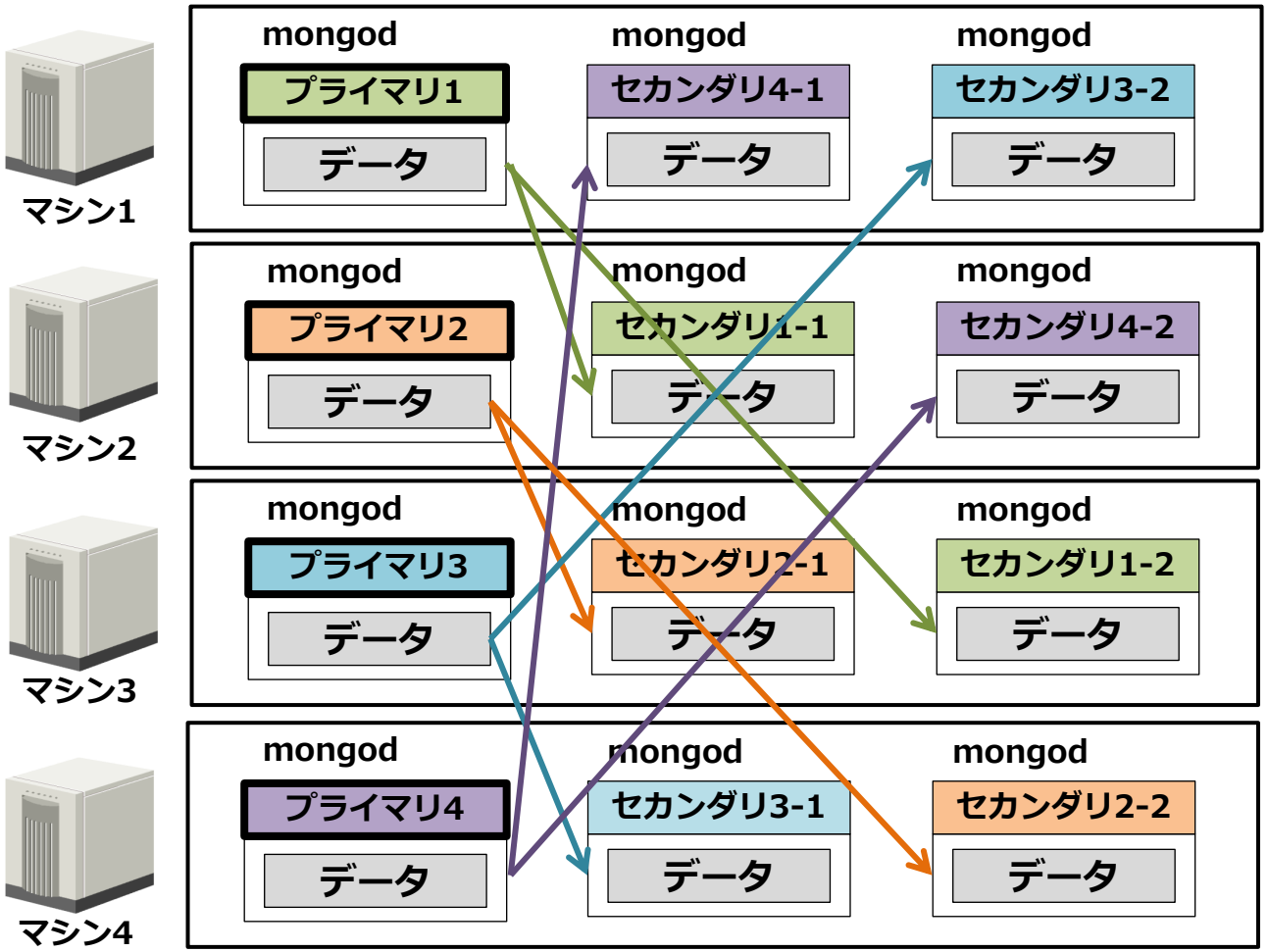


3.1 レプリケーションによるクラスタ構成

HITACHI
Inspire the Next

シャーディング構成を利用したレプリカセット

シャーディング構成での複数台という特性を利用し、マシン内に同一レプリカセットの
セカンダリノードが重複しない構成にして冗長性を確保します。
下記の例ではセカンダリを２個とした場合の例で同じ色が「レプリカセット」です。

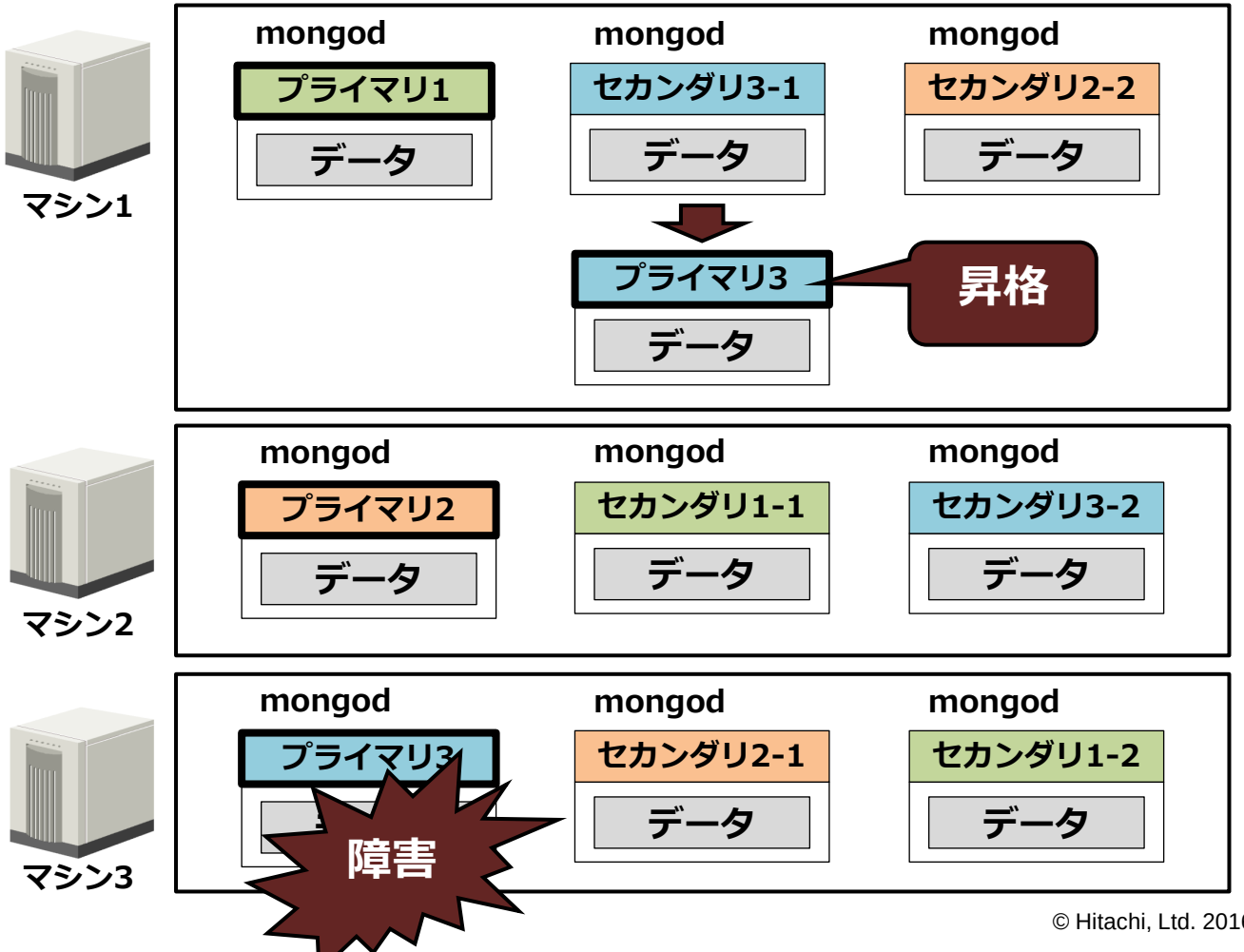


3.1 レプリケーションによるクラスタ構成

HITACHI
Inspire the Next

シャーディング構成では自動フェイルオーバ

レプリカセット内のノードはハートビート(2秒間隔でのping)で互いに生存確認を行います。
プライマリノードに障害が発生した場合、最も新しい状態のセカンダリノードが自動的に
プライマリノードに昇格します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

シャード構成ではデータが分散されていますが、バックアップ・リストアはどのようにすれば良いでしょうか？

Webでは「複雑で面倒」という記事を見かけますがどうなのでしょうか？

「百聞は一見に如かず」ということで実際にやってみました。

3.2 バックアップ・リストア

HITACHI
Inspire the Next

バックアップの方法は下記2つの方法が選択できます。

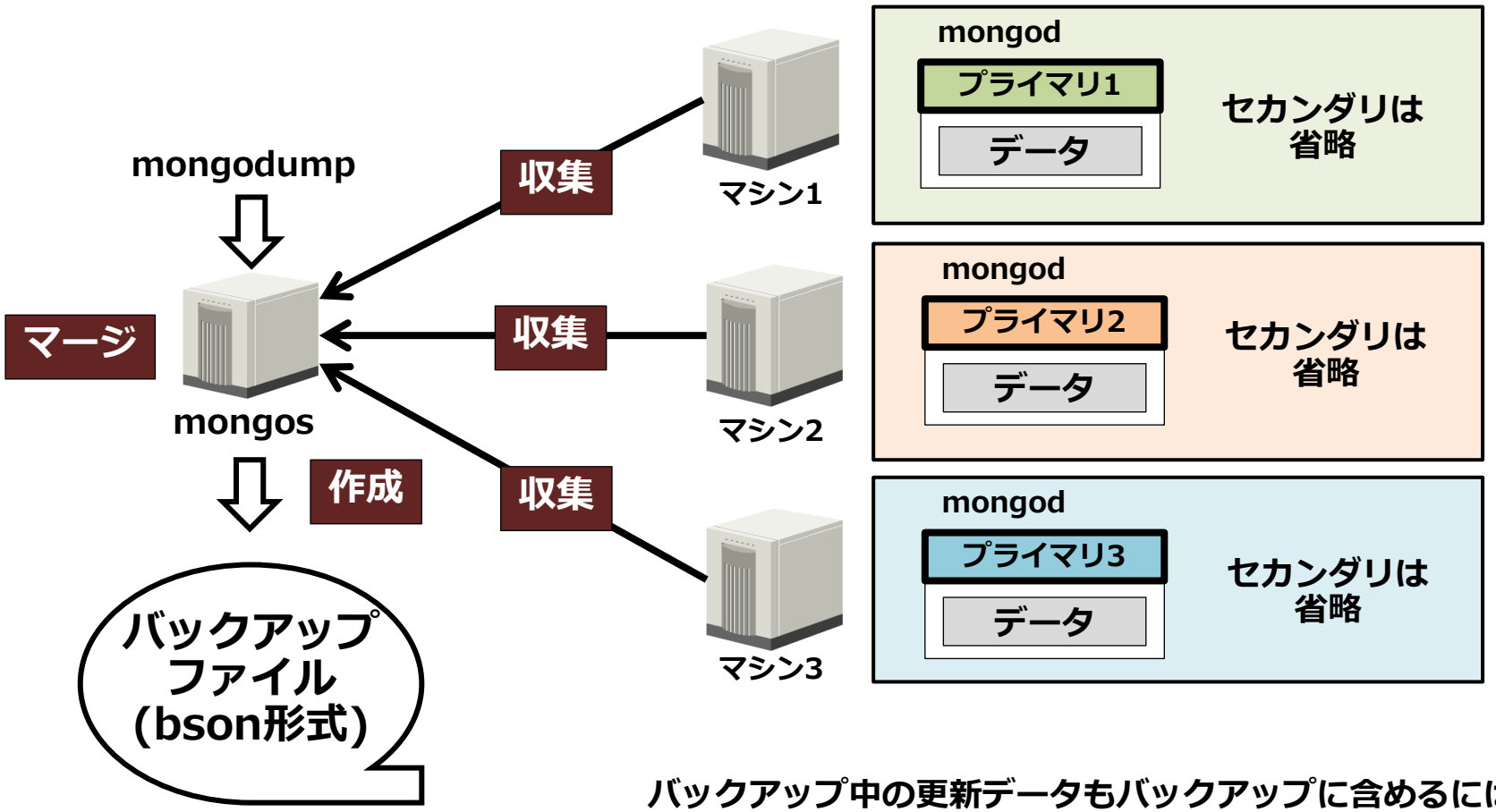
- mongodump/mongorestoreによる全体論理バックアップ・リストア
- シャード毎でファイルシステムでの物理バックアップ・リストア

マニュアルは手順が記載されており、それぞれの手順について説明します。

3.2 バックアップ・リストア

mongodumpによる全体論理バックアップ

mongodumpによるバックアップの手順はシンプルです。
mongodumpを実行するだけで、すべてのマシンからデータを収集し、
マージして単一ファイルを作成します。



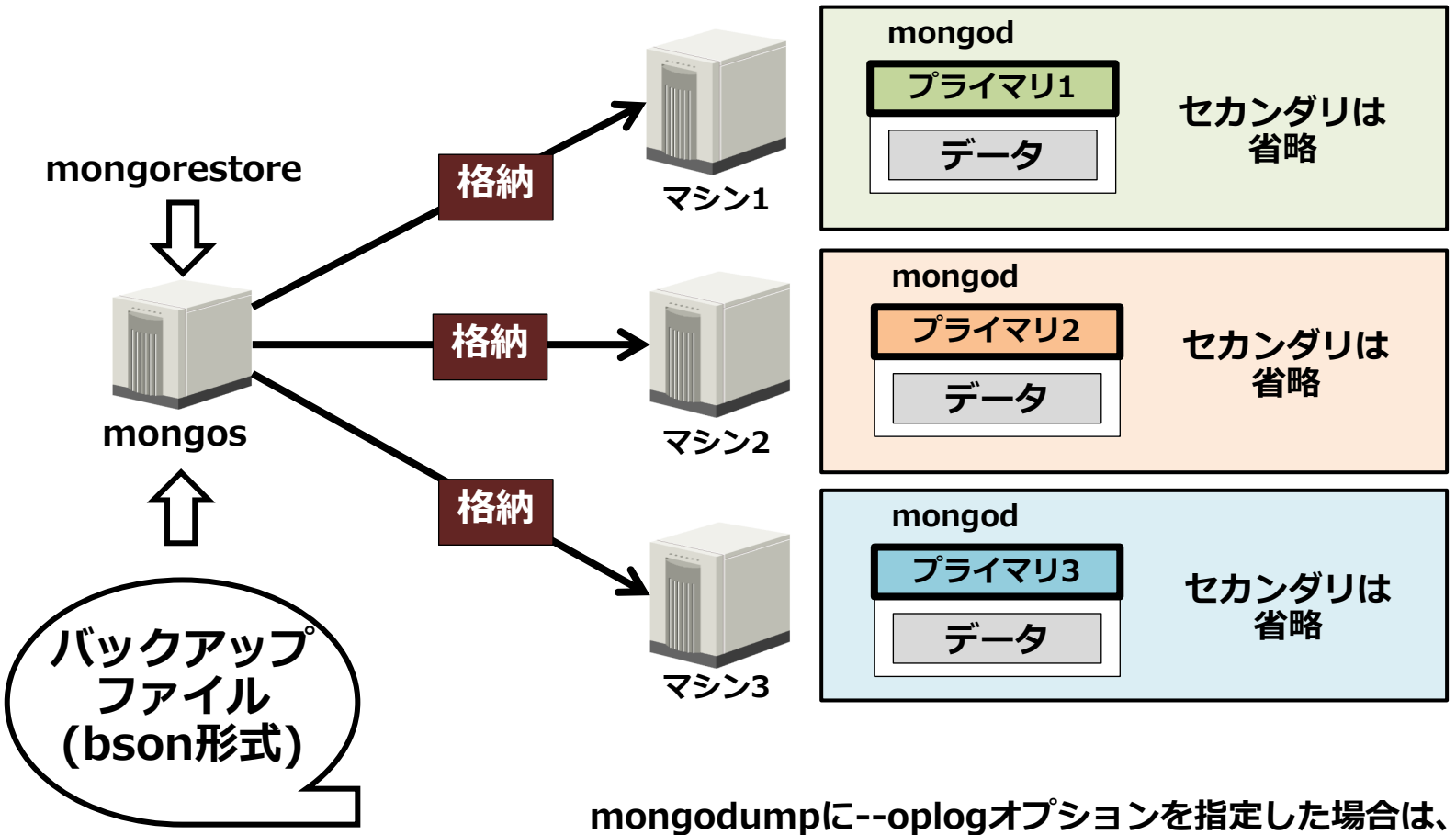
バックアップ中の更新データもバックアップに含めるには
mongodumpに--oplogオプションを指定します。

3.2 バックアップ・リストア

HITACHI
Inspire the Next

mongorestoreによる全体論理リストア

mongorestoreによるリストアの手順もシンプルです。
mongorestoreを実行するだけで、すべてのマシンにデータを分散格納します。



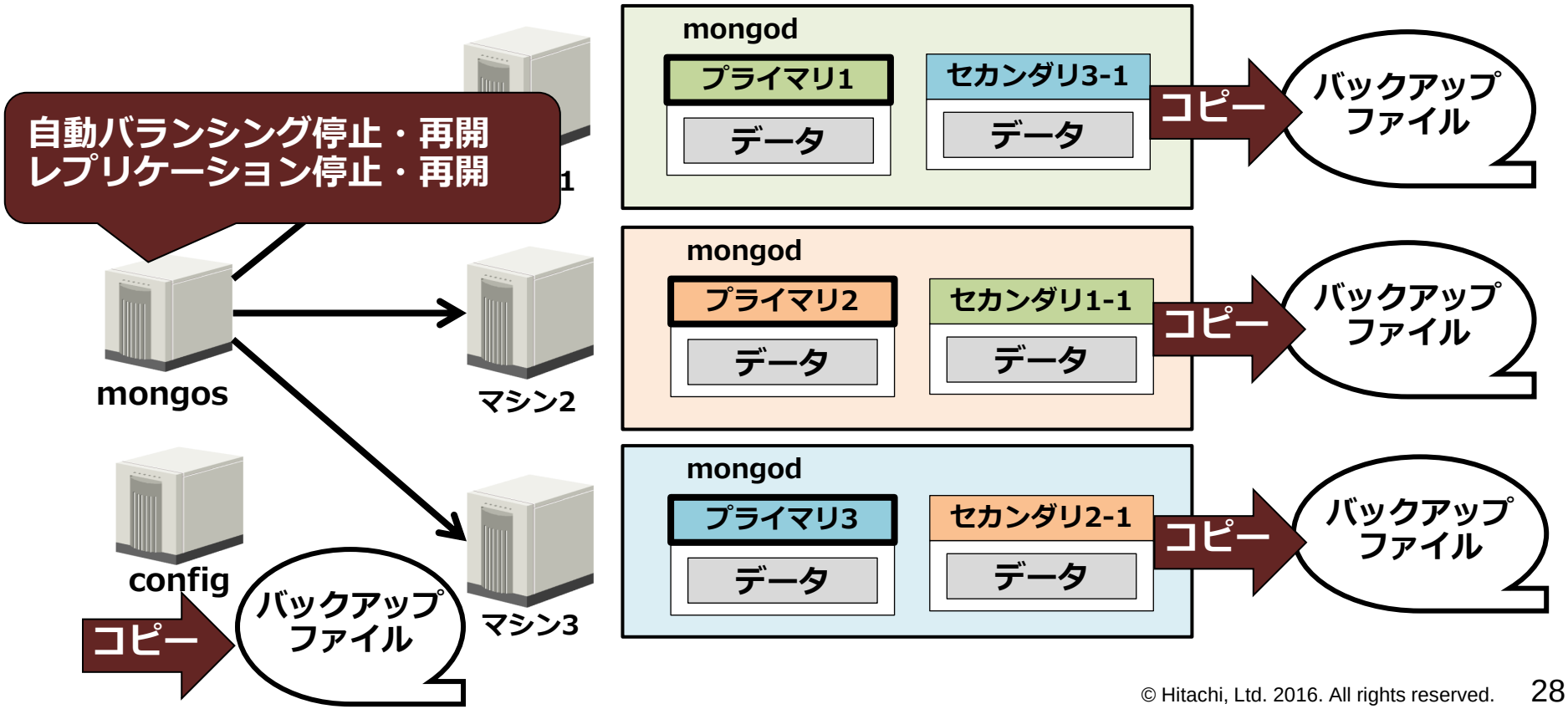
mongodumpに--oplogオプションを指定した場合は、
mongorestoreで--oplogReplayを指定します。

3.2 バックアップ・リストア

HITACHI
Inspire the Next

シャード毎でファイルシステムでの物理バックアップ

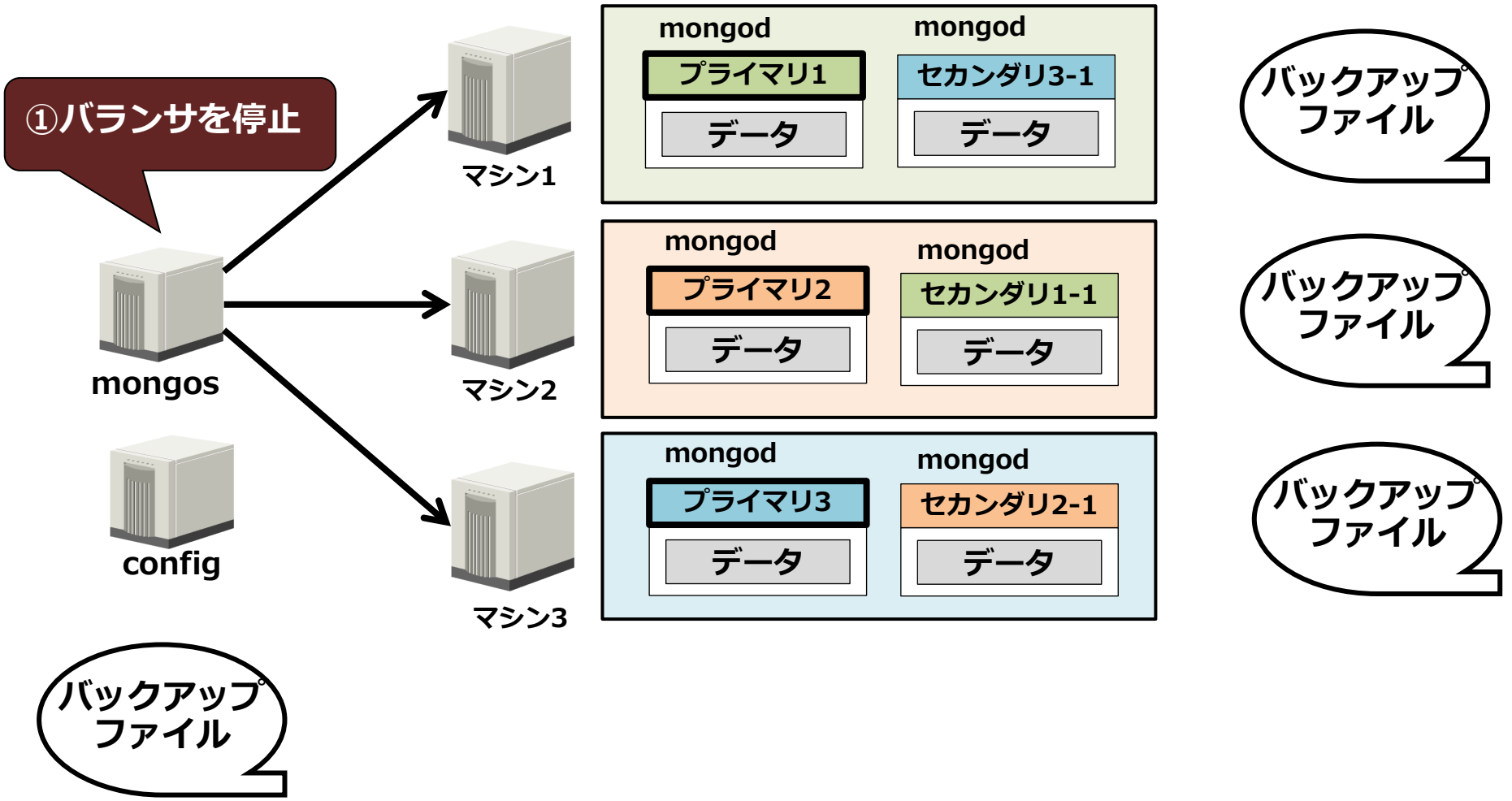
ファイルシステムによるバックアップの手順は煩雑です。
稼働中でバックアップするにはセカンダリを対象とします。
configサーバのバックアップも必要です。
また、稼働中のバックアップはバックアップデータは中途半端なものとなる可能性があるため、
自動バランシングやレプリケーションを停止し、静止化ポイント（データがFIXしている状態）
を設ける必要があります。
各マシンで必要な作業を以下に示します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

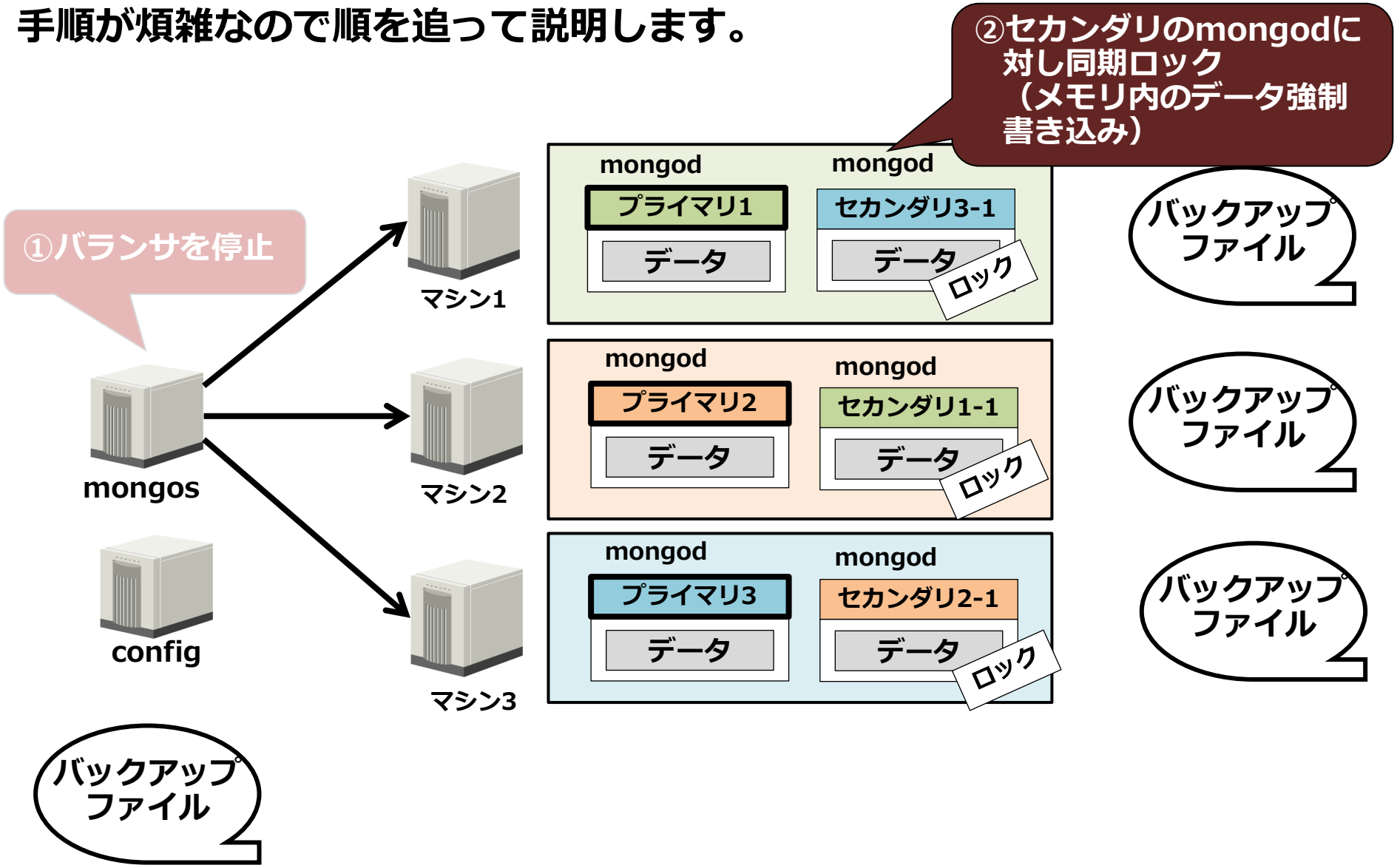
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

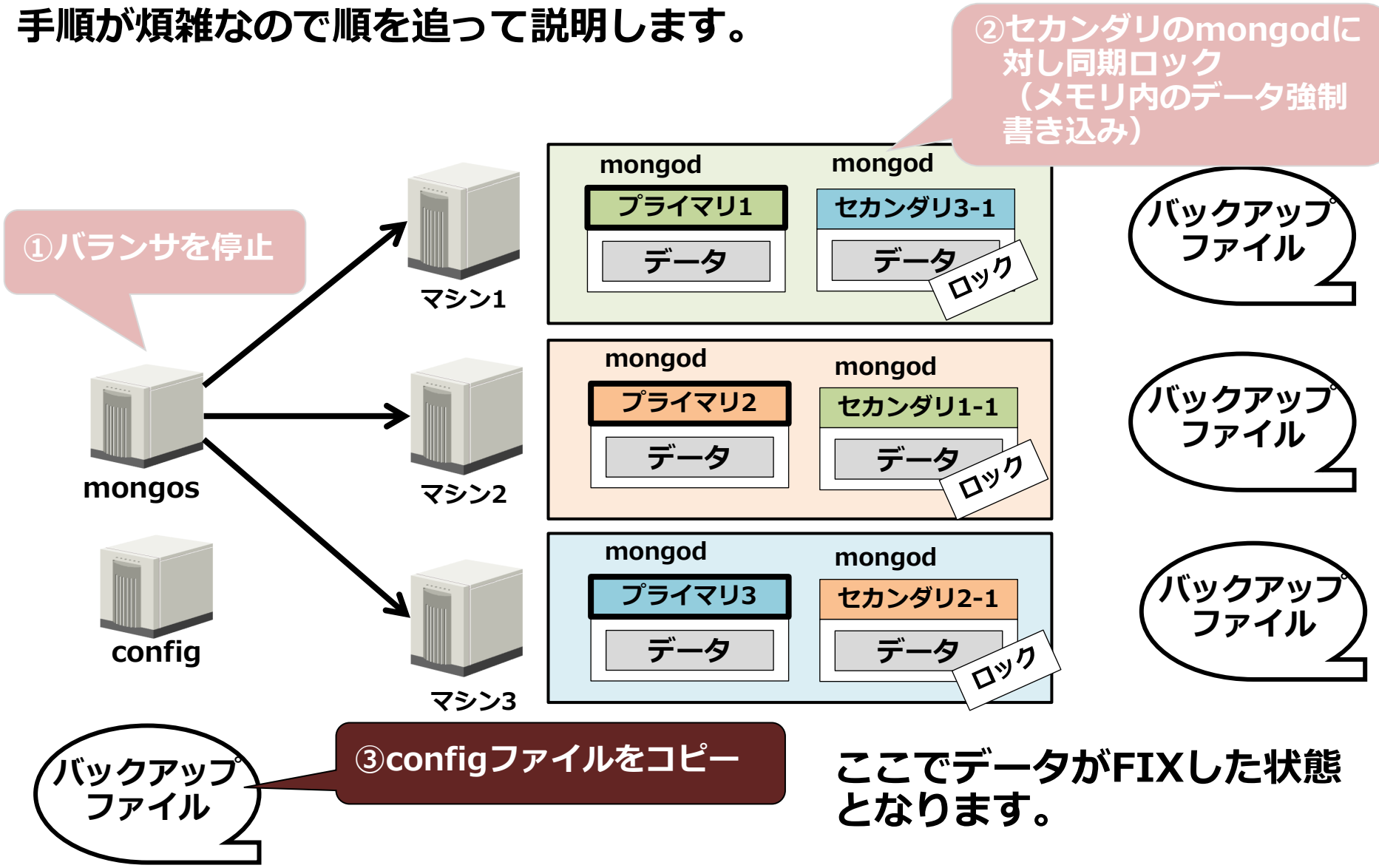
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

手順が煩雑なので順を追って説明します。

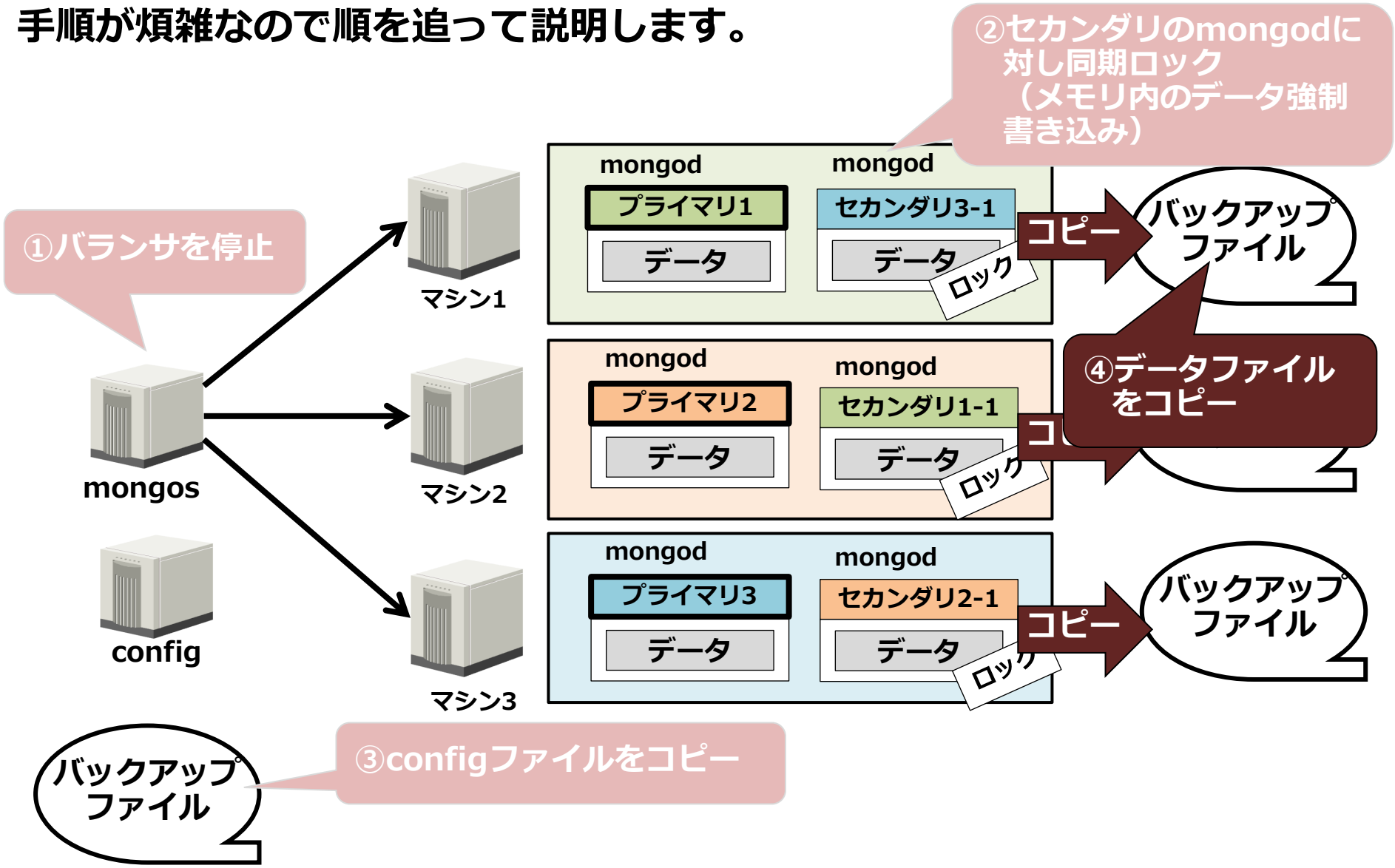


ここでデータがFIXした状態となります。

3.2 バックアップ・リストア

HITACHI
Inspire the Next

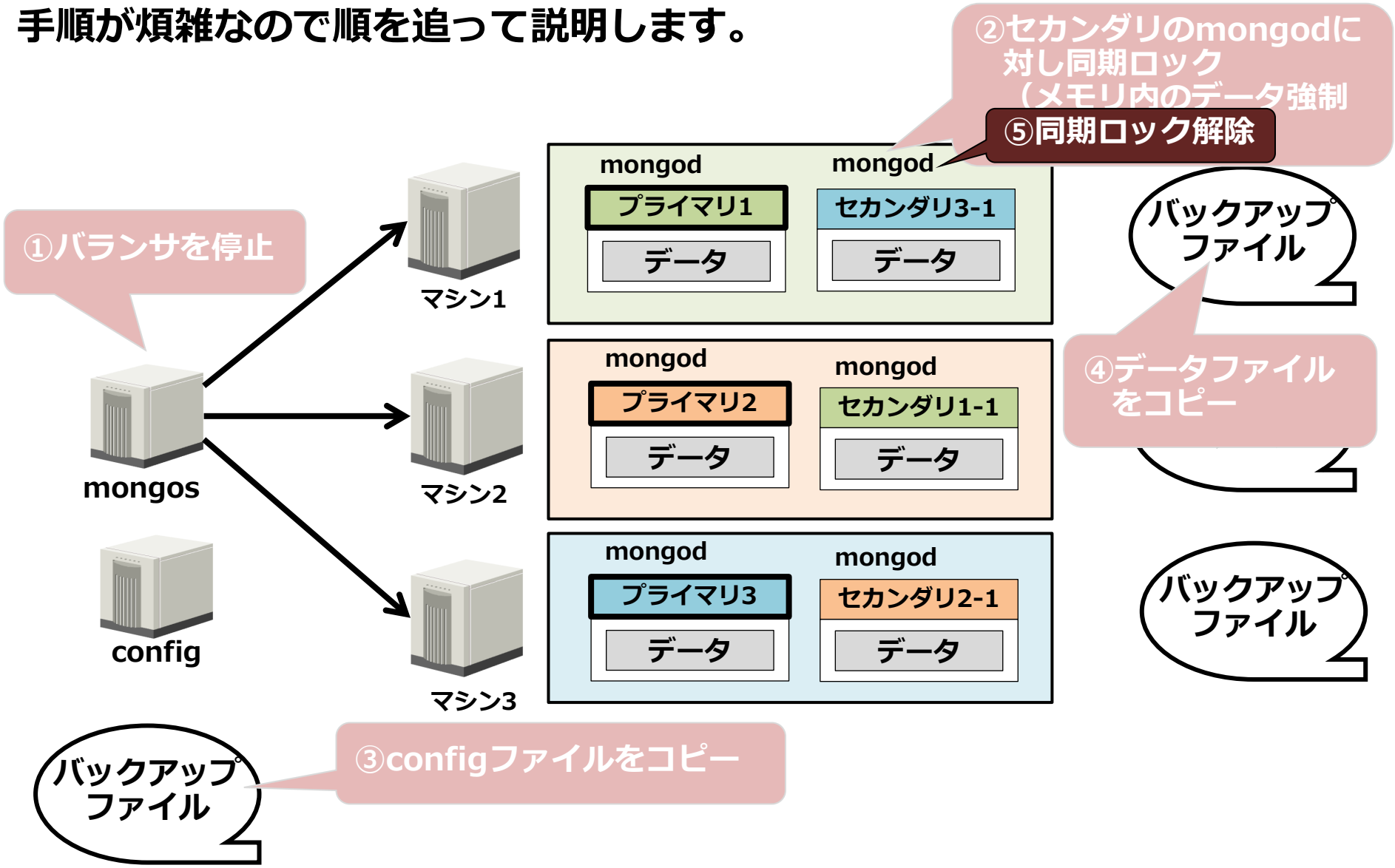
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

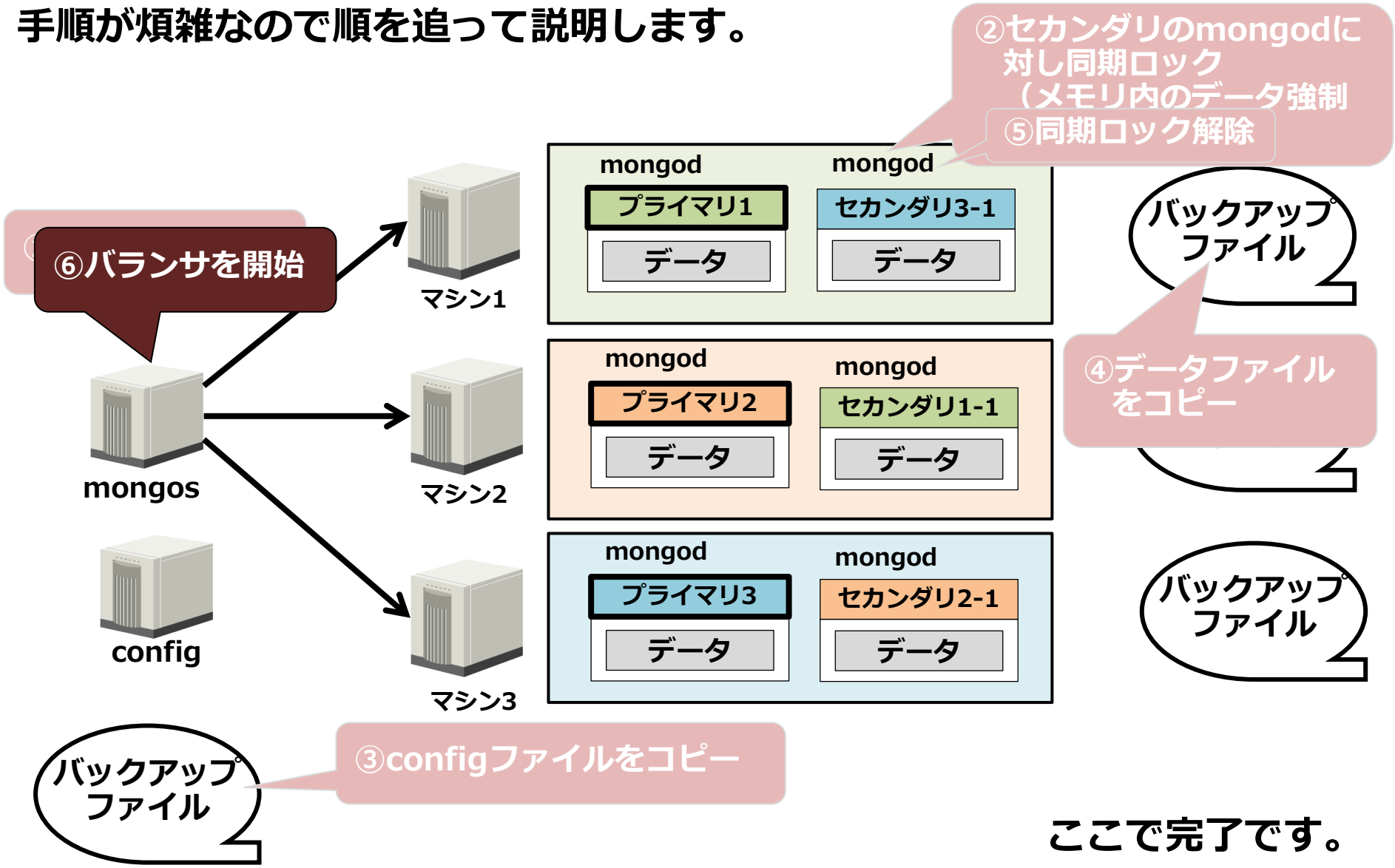
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

手順が煩雑なので順を追って説明します。



ここで完了です。

3.2 バックアップ・リストア

HITACHI
Inspire the Next

シャード毎でファイルシステムでの物理バックアップ

作業順序はまとめると以下の様になります。
②④⑤は各サーバでの実行する必要があります。

① バランサ停止
② セカンダリのmongodに対し同期ロック（メモリ内のデータ強制書き込み）

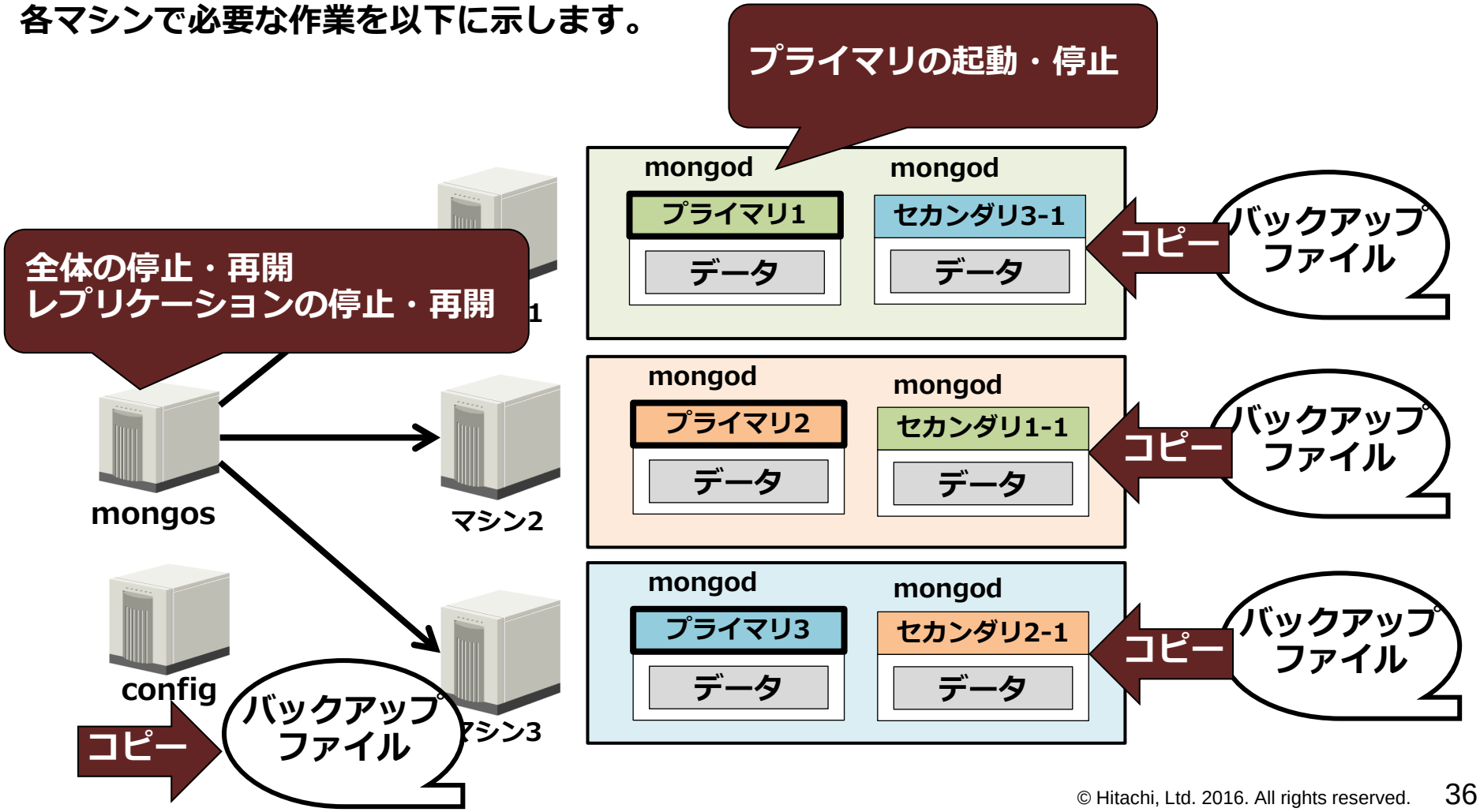
※重要
- ここで静止化ポイント（データがFIXした状態）ができます。

③ configファイルをコピー
④ データファイルコピー
⑤ レプリカのmongodに対し同期ロック解除
⑥ バランサ開始

3.2 バックアップ・リストア

シャード毎でファイルシステムでの物理リストア

ファイルシステムによるリストアの手順も煩雑です。
ファイルシステムでのリストアはデータベース作り直しを意味するためバックアップよりも更に多くの手順が必要です。
各マシンで必要な作業を以下に示します。



3.2 バックアップ・リストア

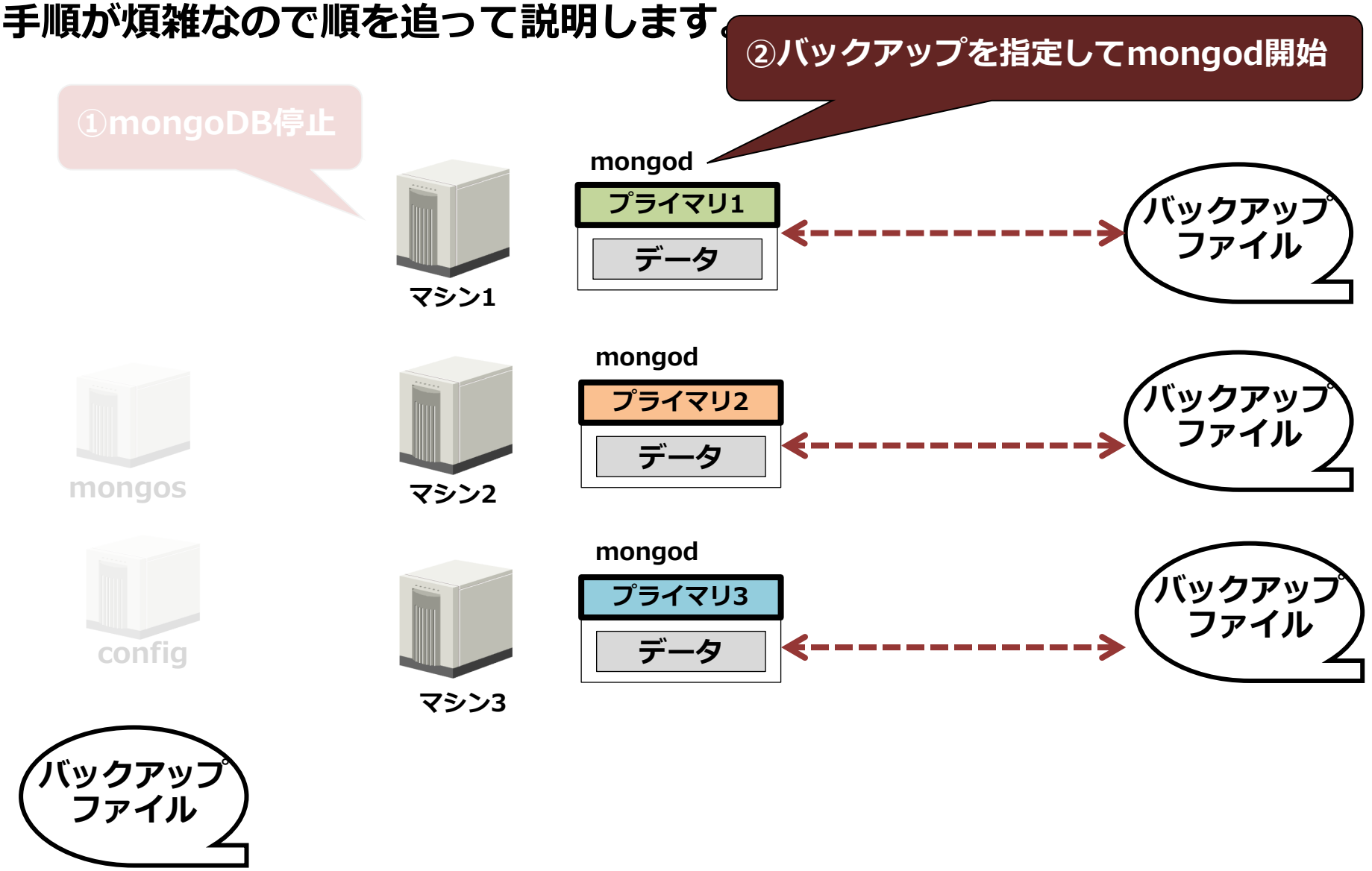
HITACHI
Inspire the Next

手順が煩雑なので順を追って説明します。

3.2 バックアップ・リストア

HITACHI
Inspire the Next

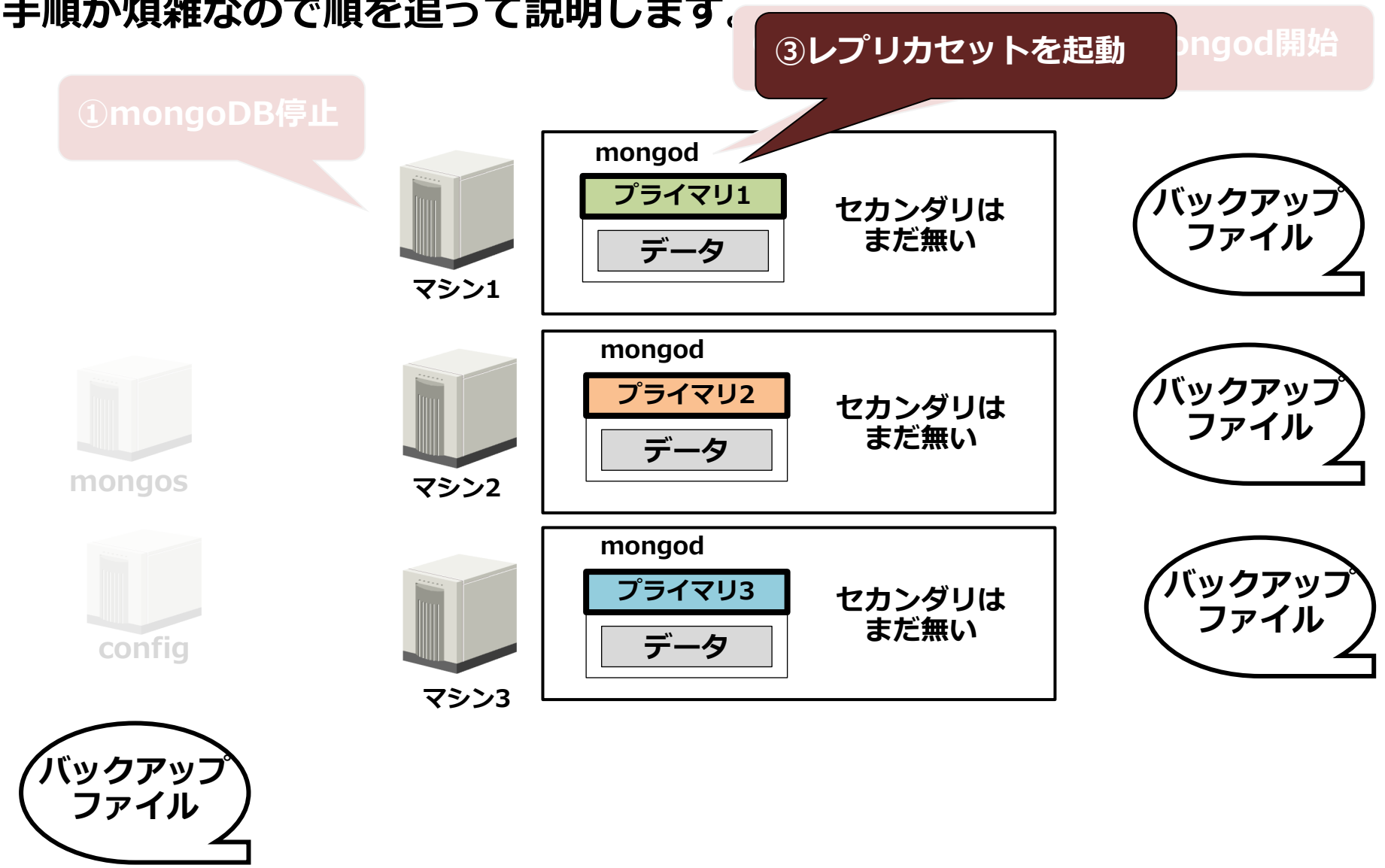
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

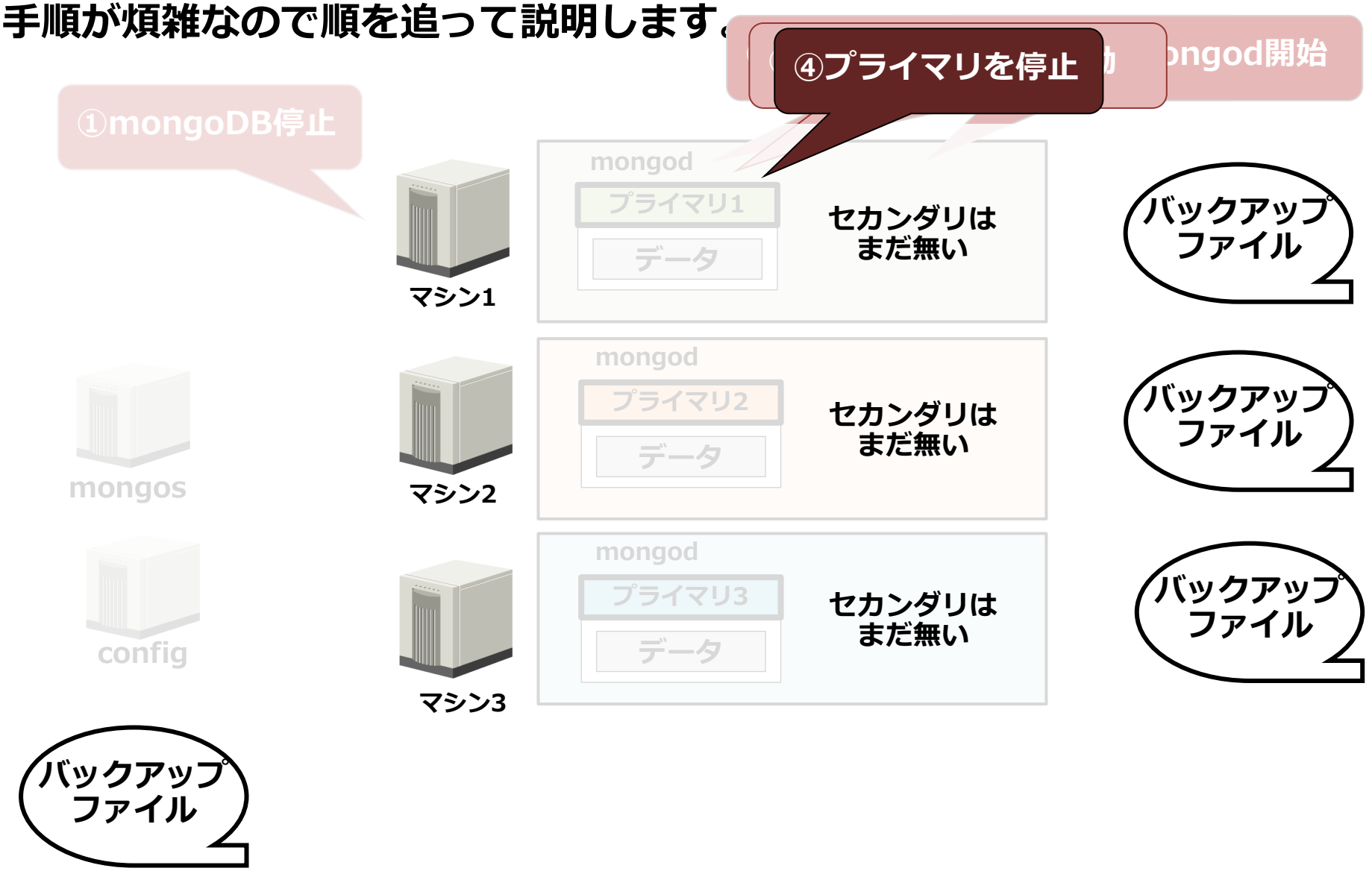
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

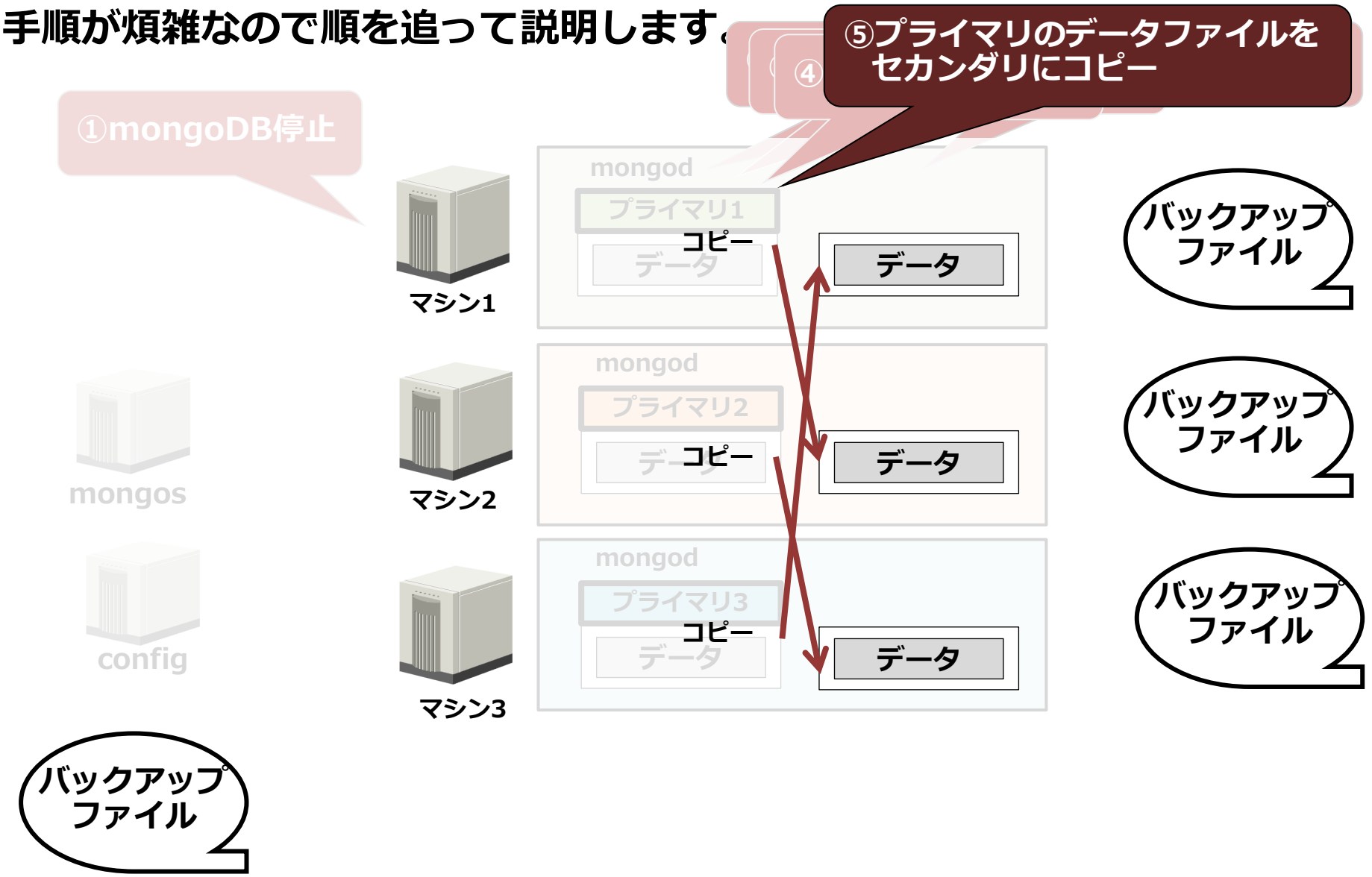
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

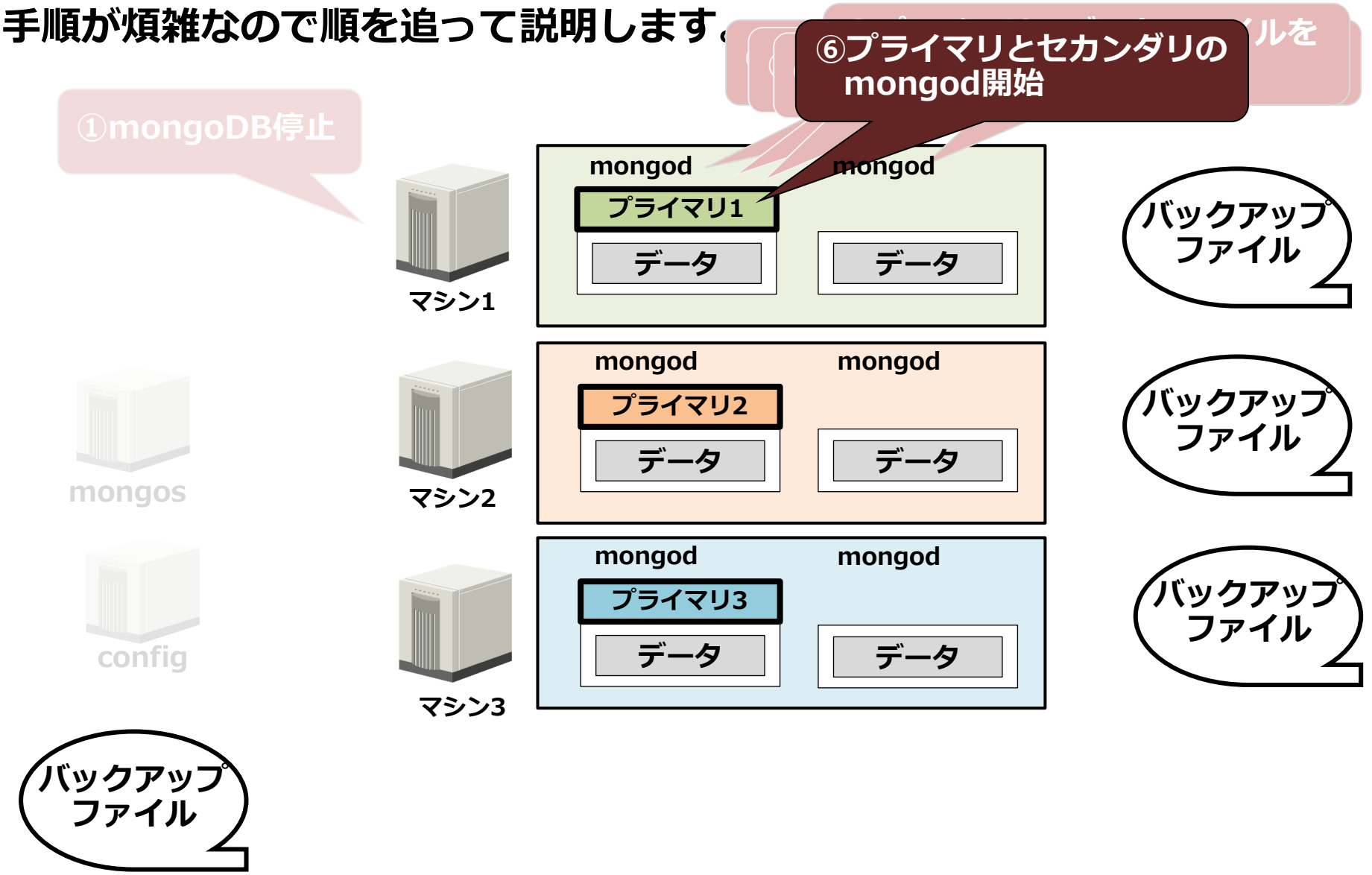
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

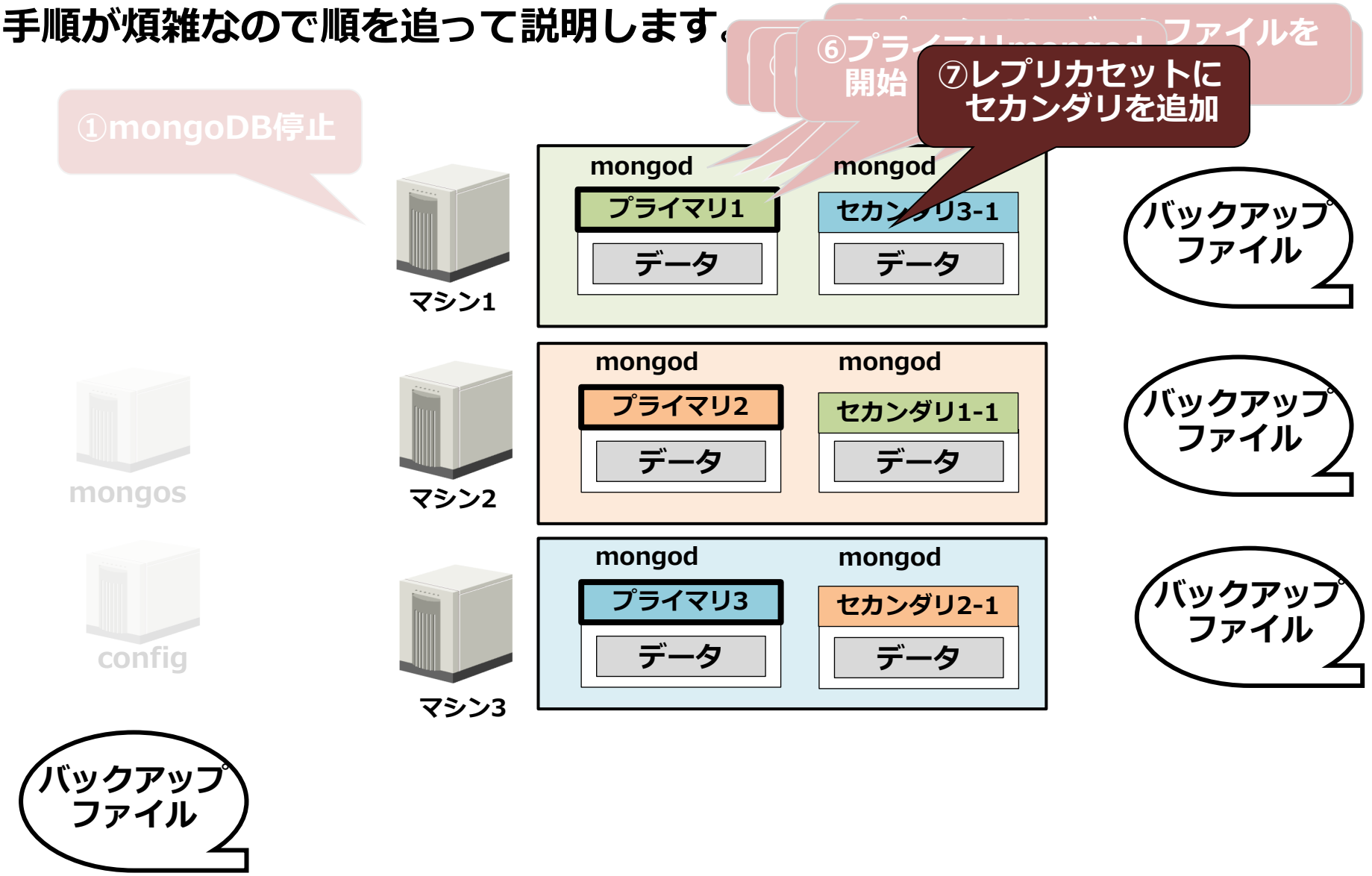
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

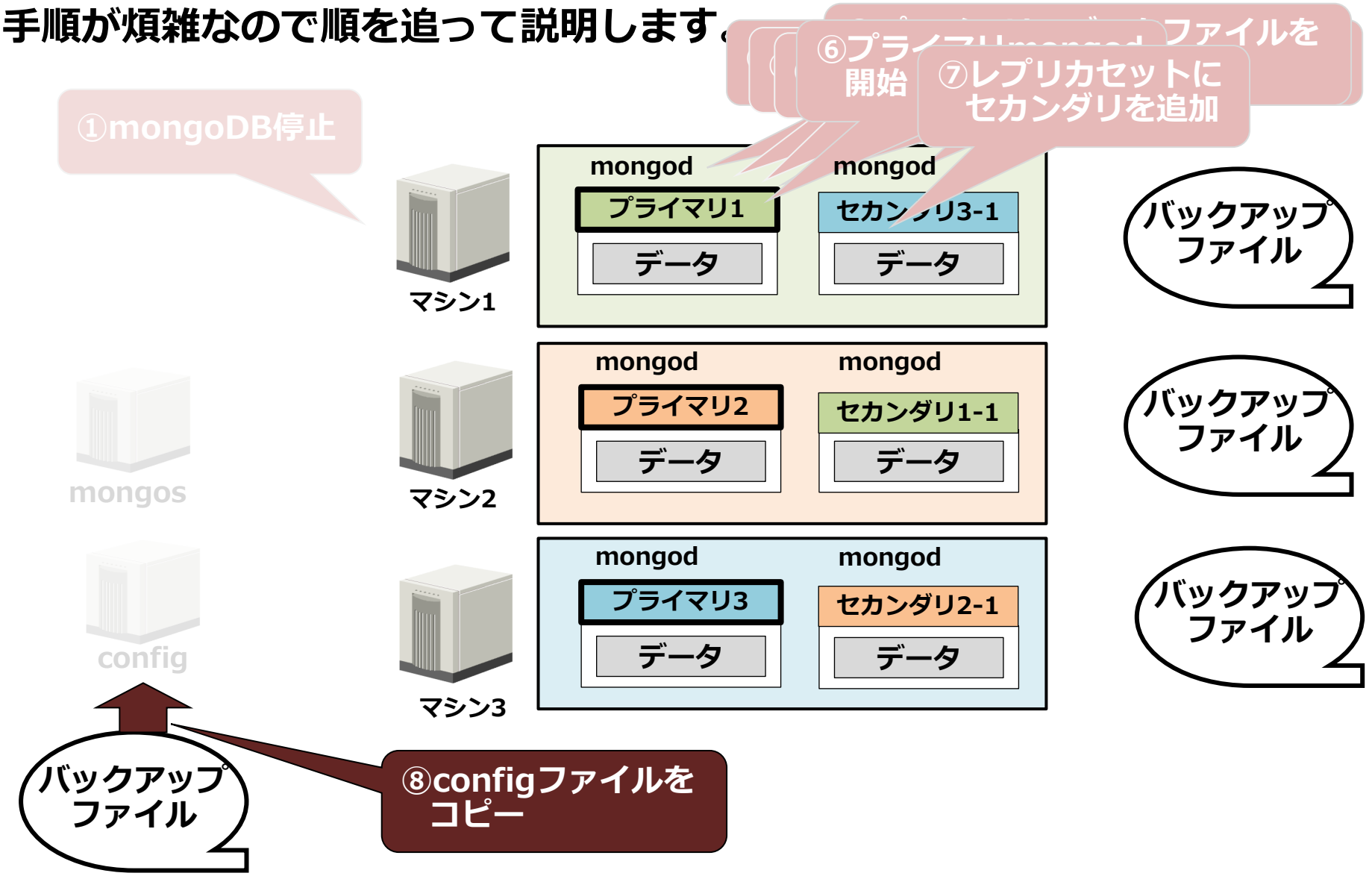
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

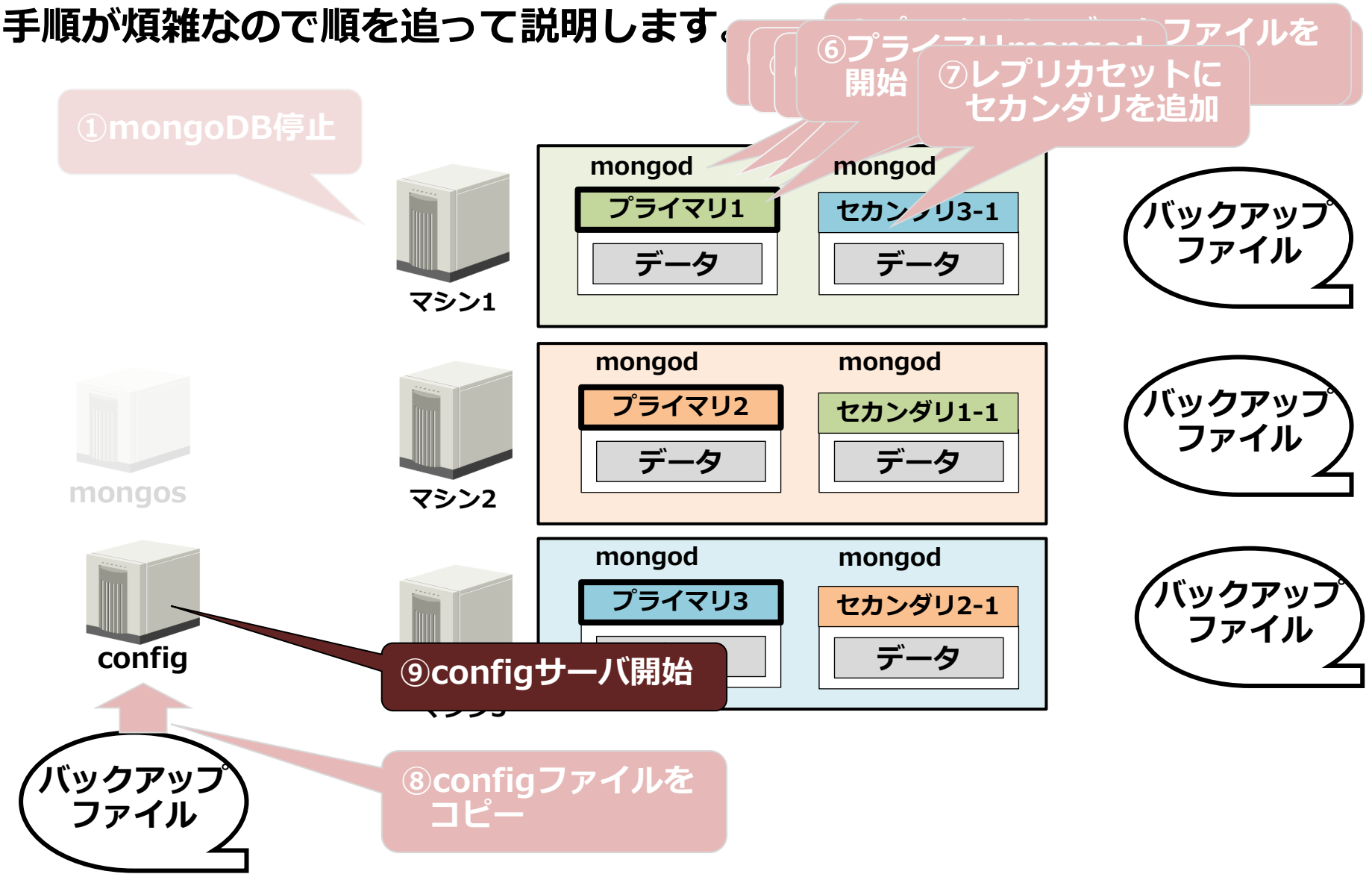
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

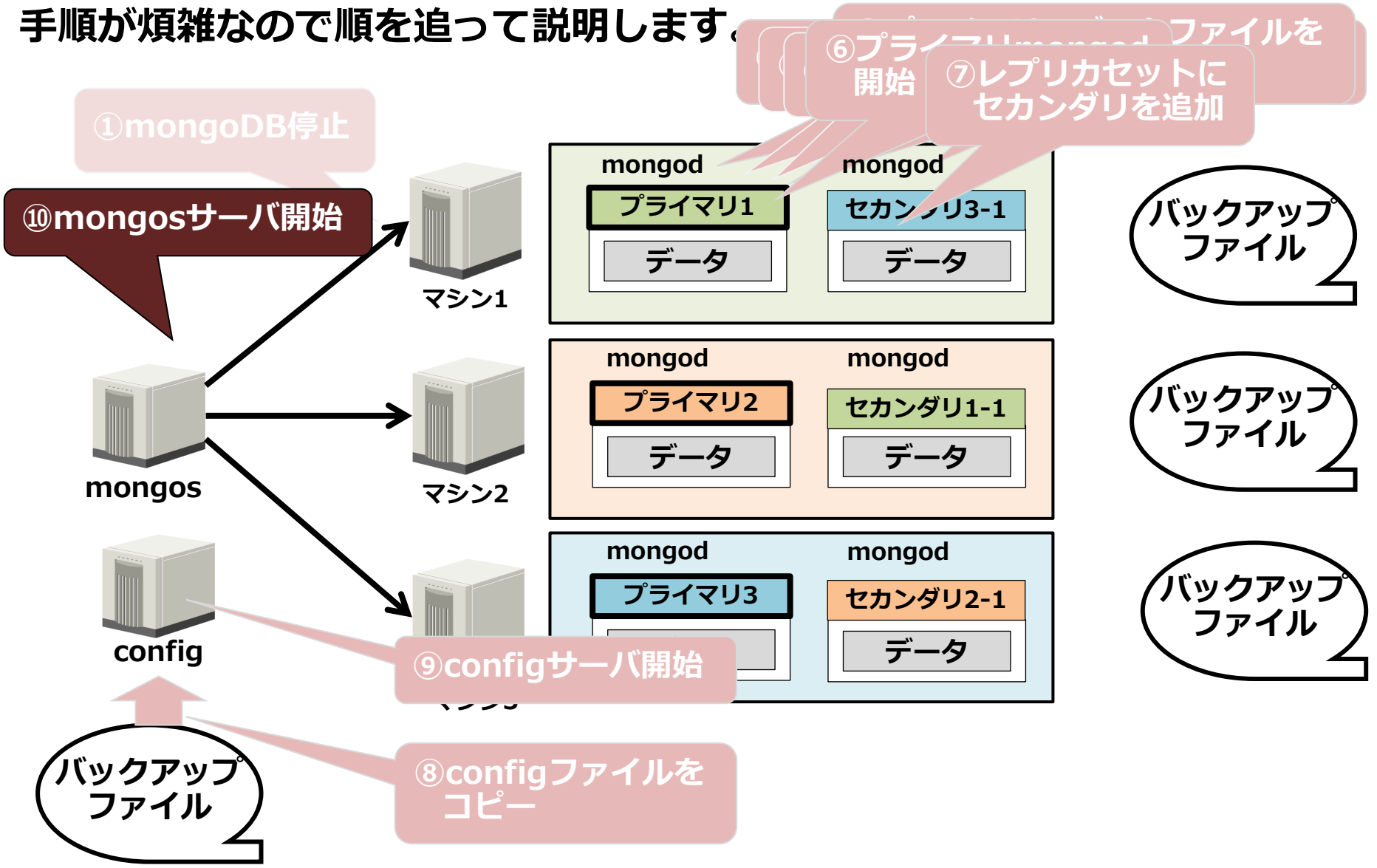
手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

手順が煩雑なので順を追って説明します。



3.2 バックアップ・リストア

HITACHI
Inspire the Next

シャード毎でファイルシステムでの物理リストア

作業順序は以下の様になります。
②～⑦は各サーバで実行する必要があります。

① mongoDB停止
② バックアップを指定してmongod開始
③ レプリカセットを起動
④ プライマリを停止
⑤ プライマリのデータファイルをセカンダリにコピー
⑥ プライマリとセカンダリのmongod開始
⑦ レプリカセットにセカンダリを追加
⑧ configファイルをコピー
⑨ configサーバ開始
⑩ mongosサーバ開始

3.2 バックアップ・リストア

HITACHI
Inspire the Next

バックアップ・リストアについて2つの手順を説明しました。
各手順をまとめると次のようになります。

シャード数：3
データ量：2億件

項目	論理バックアップ（mongodump mongorestore）	物理バックアップ（ファイルシステム）
運用容易性	簡単	煩雑
バックアップ処理時間	21分11秒	各シャードの合計 1分59秒（最大 56秒）
リストア処理時間	51分59秒	各シャードの合計 2分4秒（最大 1分4秒）
バックアップファイルサイズ	37.9GB	合計11.5GB
管理対象バックアップファイル数	1個	複数（シャード数＋config分）

論理バックアップ・リストアは手順が簡単ではあるものの下記の点で時間が掛ります。
- bson形式との変換があるため、その変換処理オーバーヘッド
- 単一ファイルへの書き込みであるため、処理がシリアライズされる

3. 運用管理

HITACHI
Inspire the Next

まとめ

項番	運用内容	結論
1	レプリケーション によるクラスタ構成	セカンダリの配置を間違えなければ、特に面倒なこともなくクラスタ構成が可能です。
2	バックアップ・リストア	手順を取るか、時間を取るかの選択が必要です。 例えば24時間ではない業務であり夜間に十分時間を確保できるのであれば、論理バックアップ・リストアも十分検討の余地があります。 業務要件から最適なバックアップ・リストアを選択する必要があります。

4. まとめ

HITACHI
Inspire the Next

データ量が多く1台のマシンではメモリに載せきれない場合は
確実にシャーディングの効果がありそうです。
そうではないケースでは業務パターンによります。
また、運用面の煩雑さも無視できません。

闇雲にシャーディングを選択せず、その特性を理解し、
業務特性、データ量、所有資産、運用にかけられる時間との
バランスで判断する必要があります。

HITACHI
Inspire the Next

END

MongoDBで試すシャーディングの実力検証！

2016/11/05

株式会社 日立製作所
OSSソリューションセンタ

付録. 他社商品名，商標等の引用に関する表示

HITACHI
Inspire the Next

- Linuxは，Linus Torvalds氏の日本およびその他の国における登録商標または商標です。
- Red Hatは，米国およびその他の国でRed Hat, Inc. の登録商標若しくは商標です。
- MongoDBは，MongoDB Inc.の登録商標です。

その他記載の会社名，製品名は，それぞれの会社の商標もしくは登録商標です。


HITACHI
Inspire the Next