HITACHI
Inspire the Next

# PGConf.ASIA 2017

## PostgreSQLを使って変化に強いシステムへ！

2017/12/4

(株)日立製作所　システム＆サービスビジネス統括本部
OSSソリューションセンタ

© Hitachi, Ltd. 2017. All rights reserved.

HITACHI
Inspire the Next

# Contents

1. 日立の取り組み
2. PostgreSQLとIoT
3. 某DBからPostgreSQLへの移行事例
4. 弊社サービス紹介

© Hitachi, Ltd. 2017. All rights reserved.

# 自己紹介

HITACHI
Inspire the Next

●名前
稲垣　毅　(INAGAKI Tsuyoshi)

Facebook：　https://www.facebook.com/tsuyoshi.inagaki.3

●所属
・株式会社日立製作所　OSSソリューションセンタ
・PostgreSQL エンタープライズコンソーシアム
・OSSコンソーシアム　オープンCOBOLソリューション部会

●仕事
・PostgreSQL、MySQLのテクニカルサポート
・データベースの評価、検証
・外部団体の活動

●PostgreSQLの近況
・hookの面白い活用方法を考えている　（←あくまで趣味です。）

© Hitachi, Ltd. 2017. All rights reserved.

# １．日立の取り組み

© Hitachi, Ltd. 2017. All rights reserved.

# 1-1. デジタル化の進展

HITACHI
Inspire the Next

## 世界では産業・社会インフラのデジタル化が加速

**欧州**
Industrie4.0

**中国**
中国製造2025

**アジア**
スマートシティ

**日本**
Society5.0

**北米**
Industrial Internet

**パラダイムシフト：人、社会を中心に考えるサービス、技術の進化**

モノ → コト
占有 → シェア
クローズド → オープン
個別最適 → 全体最適

© Hitachi, Ltd. 2017. All rights reserved.

# 1-2. 社会イノベーション事業に貢献

HITACHI
Inspire the Next

高度道路交通システム

エネルギーバリューチェーン

新たなものづくりシステム

スマート・フードチェーンシステム

内閣府 第５期科学技術基本計画

**超スマート社会 Society5.0**

あらゆる人が活き活きと快適に暮らすことのできる社会

地域包括ケアシステム

スマート生産システム

インフラ維持管理・更新

総合型材料開発システム

地球環境情報プラットフォーム

おもてなしシステム

自然災害に対する強靭な社会

© Hitachi, Ltd. 2017. All rights reserved.

# 1-3. 日立がめざす姿

HITACHI
Inspire the Next

## IoT時代のイノベーションパートナー

**進化した社会イノベーション事業でお客さまとの協創を加速**

### 注力4事業分野

電力・エネルギー

産業・流通・水

アーバン

金融・公共
ヘルスケア

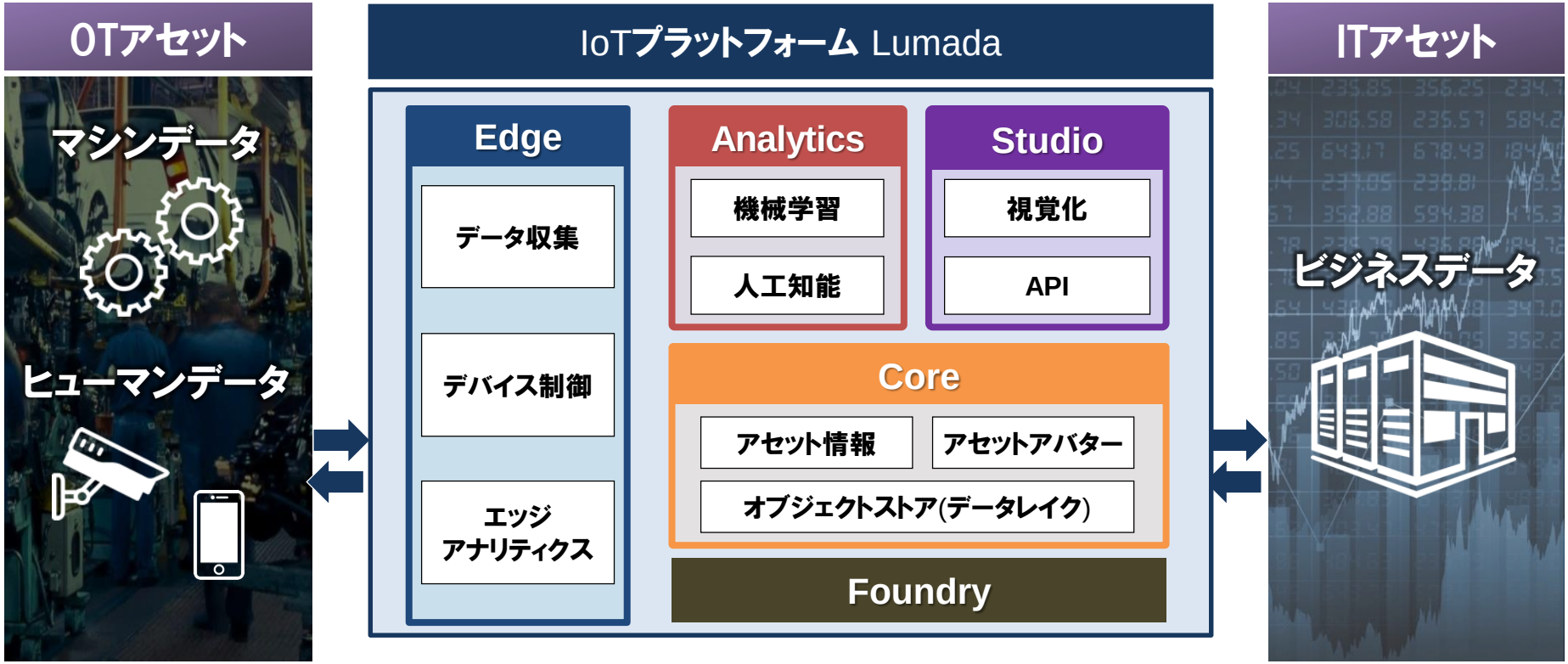
© Hitachi, Ltd. 2017. All rights reserved.

# 1-4. IoTプラットフォーム Lumada

HITACHI
Inspire the Next

**お客さまが保有する多様なアセットから生成されるデジタルデータを収集、人工知能等の最新デジタル技術により経営課題の解決を支援する**

フレキシブル　インテリジェント　コンポーザブル　セキュア




© Hitachi, Ltd. 2017. All rights reserved.

# 2. PostgreSQLとIoT

© Hitachi, Ltd. 2017. All rights reserved.

HITACHI
Inspire the Next

## ■ 概要

センサーデータ等のIoTデータは、多種多様(Variety)、 かつ大量データ(Volume)が高頻度(Velocity)の「3つのV」で発生するため、RDBではなくhadoopなどに蓄積されていることが多く、そこから新たな価値が創出されている。
最近では、さらに既存業務システムのデータとIoTデータを組み合わせて新たな価値を創出したいという需要がある。

しかし、企業内のシステムの多くが、目的ごとに独立してシステム設計・構築を行った結果、「他システムのデータを活用するのにコストや手間がかかる」という課題があり、連携するための施策を検討する必要がある。

そこで、この章では、既存システム（RDBはPostgreSQL前提）とhadoopなどIoTシステムとの連携・融合について検証結果を報告する。

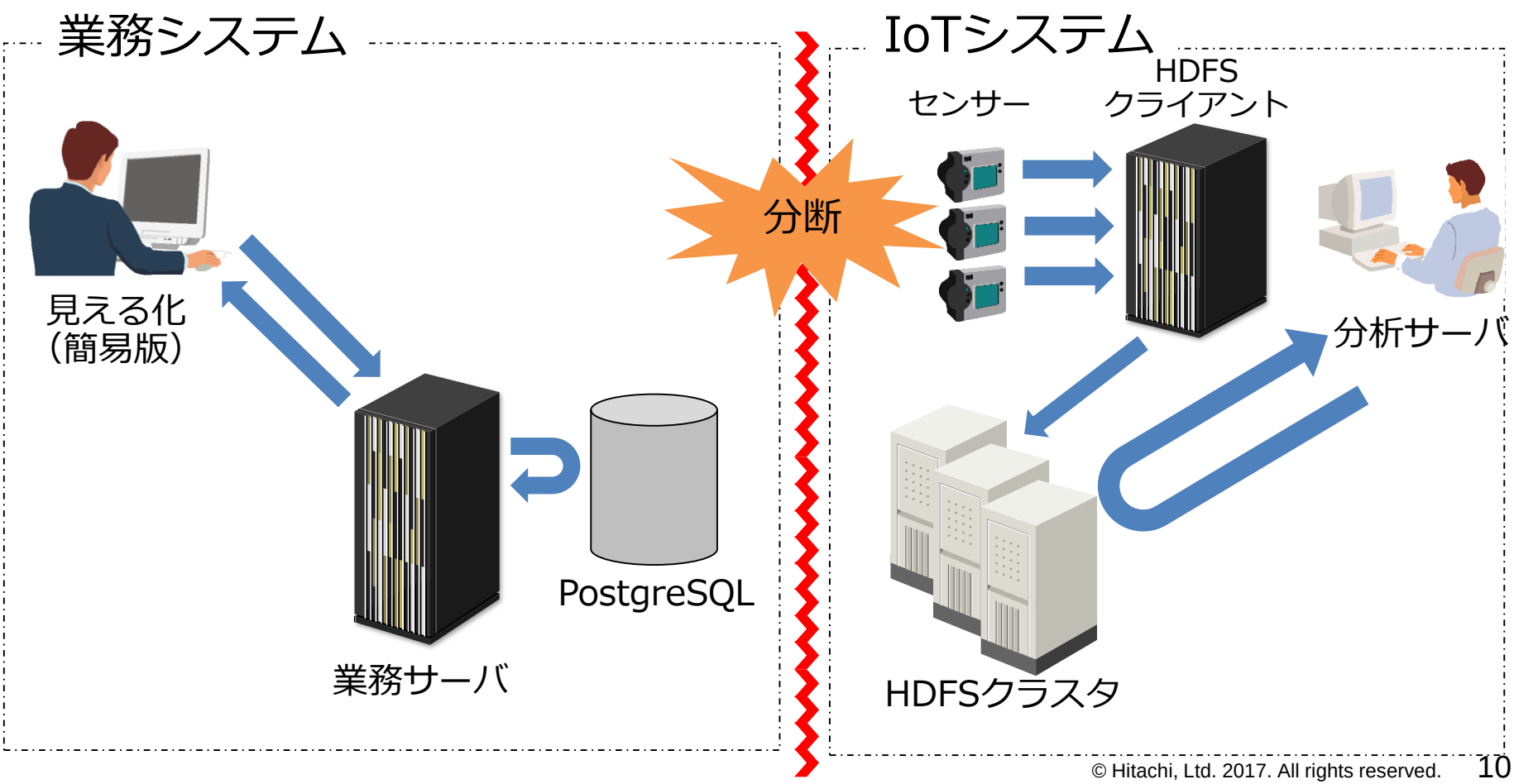
© Hitachi, Ltd. 2017. All rights reserved.

# 2-2. 現行システム

HITACHI
Inspire the Next

## ■ システムイメージ

今あるシステムは、目的別に独立して構築(サイロ化)されているため、
以下のように分断されていることが多いと思われる。



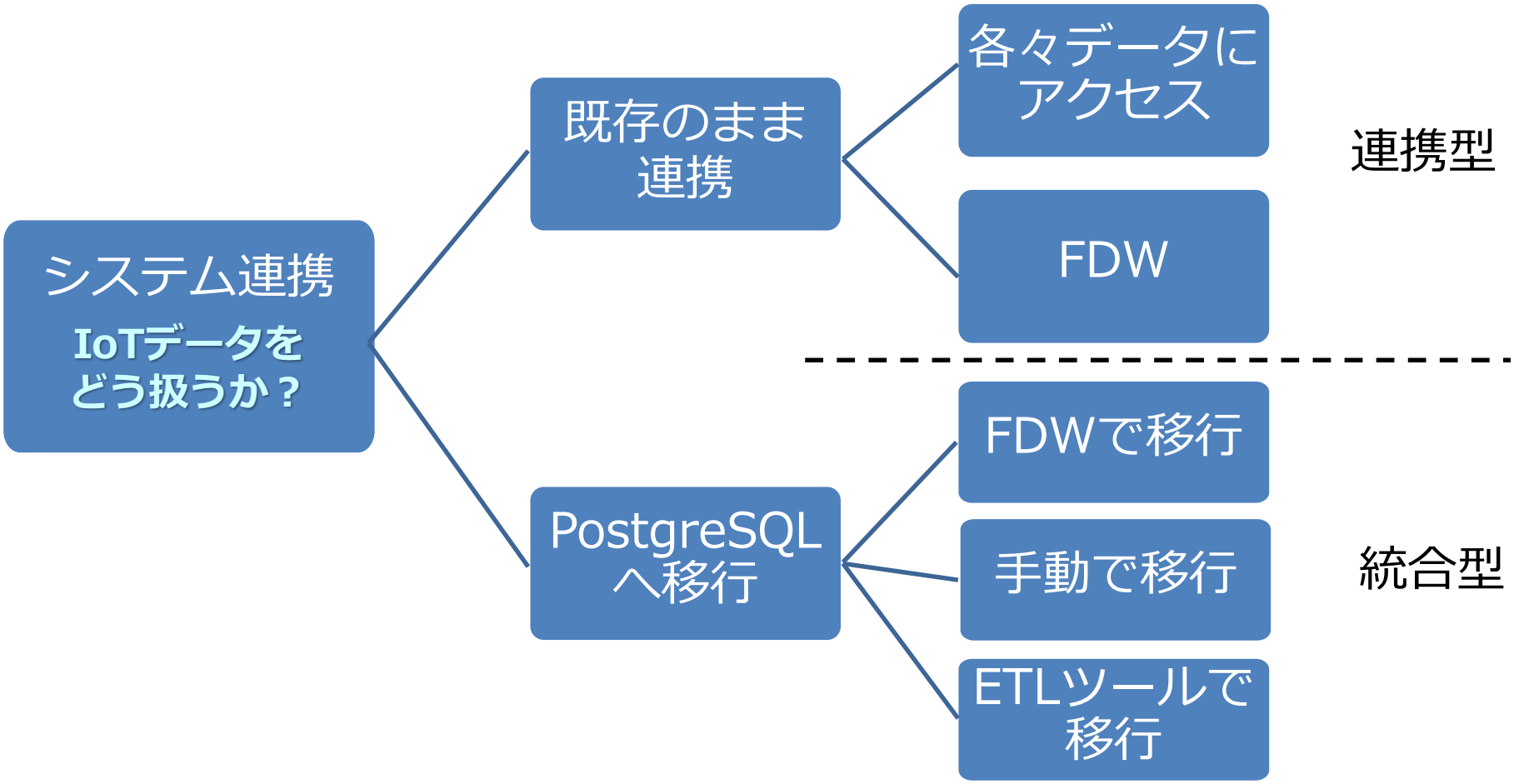

© Hitachi, Ltd. 2017. All rights reserved.

# 2-3. 手段について

HITACHI
Inspire the Next

## ■ 連携方法

業務システムとIoTシステムとの連携する上で、IoTデータの活用方法には以下の方法が考えられる。




© Hitachi, Ltd. 2017. All rights reserved.

HITACHI
Inspire the Next

## ■ FDWについて

FDW（Foreign Data Wrapper：外部データラッパー）とは、SQL/MEDの仕様を実装した、外部データにアクセスする仕組み。
PostgreSQLから、他のSQLデータベースやフラットファイルなど様々なリモートデータストアにアクセスを可能にするFDWが準備されている。

| DataStore | FDW | Owner |
|---|---|---|
| PostgreSQL | postgres_fdw | PGDG |
| Oracle | oracle_fdw | laurenz |
| MySQL | mysql_fdw | EnterpriseDB |
| Hadoop | pg-file-fdw-gds | wat4dog |
| | hadoop_fdw | openscg |
| | hdfs_fdw | EnterpriseDB |
| | hive-fdw-for-postgresql | youngwookim |
| | orc_fdw | gokhankici |

出展元：
https://wiki.postgresql.org/wiki/Foreign_data_wrappers

© Hitachi, Ltd. 2017. All rights reserved.

HITACHI
Inspire the Next

## ■ FDWについて

FDWの種類は様々あるが･･･、

**※ FDWのほとんどは、PGDG（PostgreSQLグローバル開発グループ）による正式サポートがされていないため、使用するに当たって注意する必要がある（オンラインマニュアルを参照）。**

弊社は、EnterpriseDB社とパートナー契約を結んでおり、EnterpriseDB社が開発しているFDWを使うことでサポート等の連携が可能。

| DataStore | FDW | Owner |
|---|---|---|
| Hadoop | pg-file-fdw-gds | wat4dog |
| | hadoop_fdw | openscg |
| | hdfs_fdw | EnterpriseDB |
| | hive-fdw-for-postgresql | youngwookim |
| | orc_fdw | gokhankici |

⇒本講演のFDWは、hdfs_fdwを使った評価結果を報告する。

出展元：
https://wiki.postgresql.org/wiki/Foreign_data_wrappers

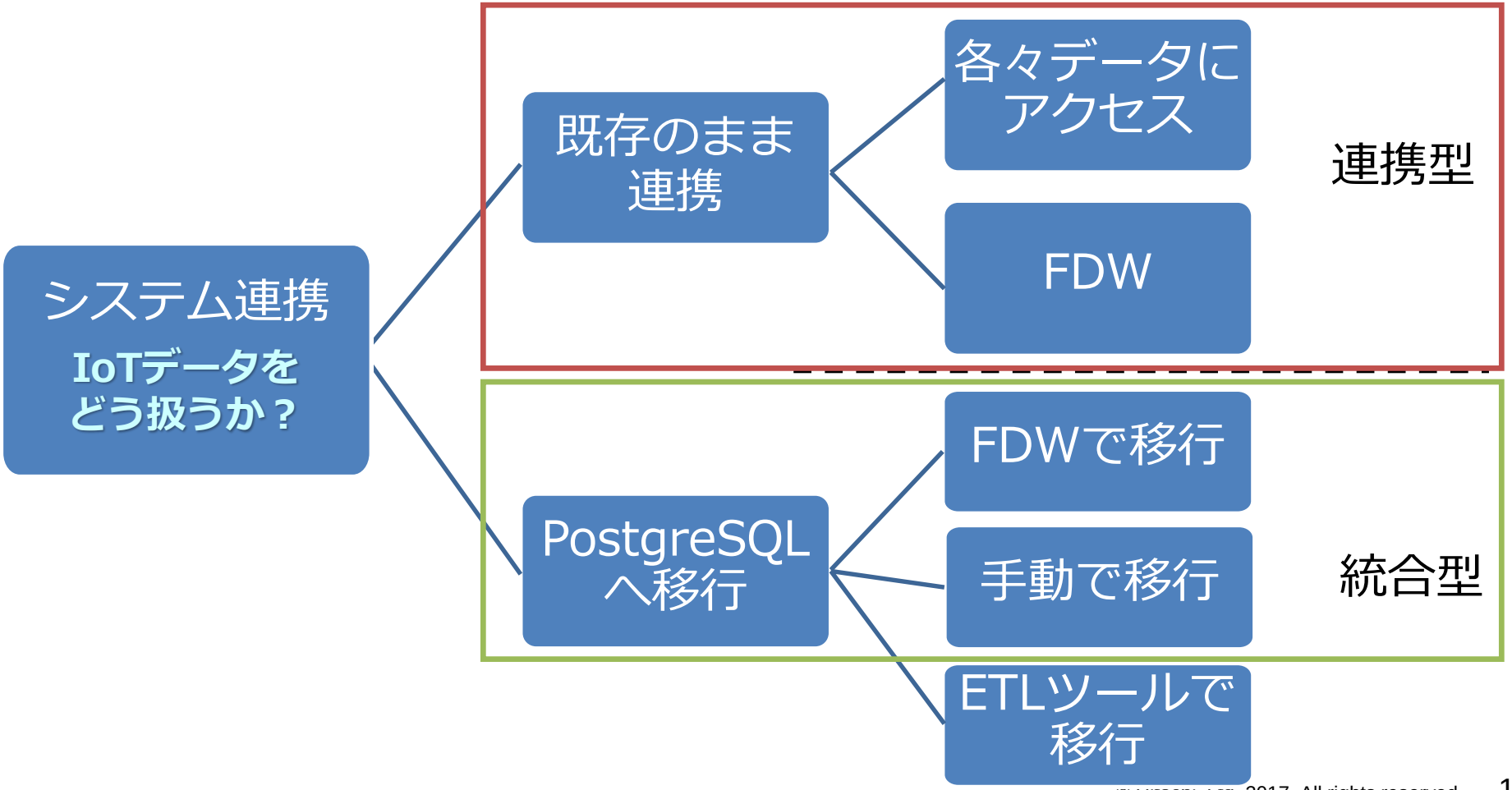
© Hitachi, Ltd. 2017. All rights reserved.

# 2-5. 評価の全体像

HITACHI
Inspire the Next

## ■ 評価

評価１：既存のまま連携する場合の各アクセス方式を評価
評価２：PostgreSQLに統合する場合の移行方式を評価

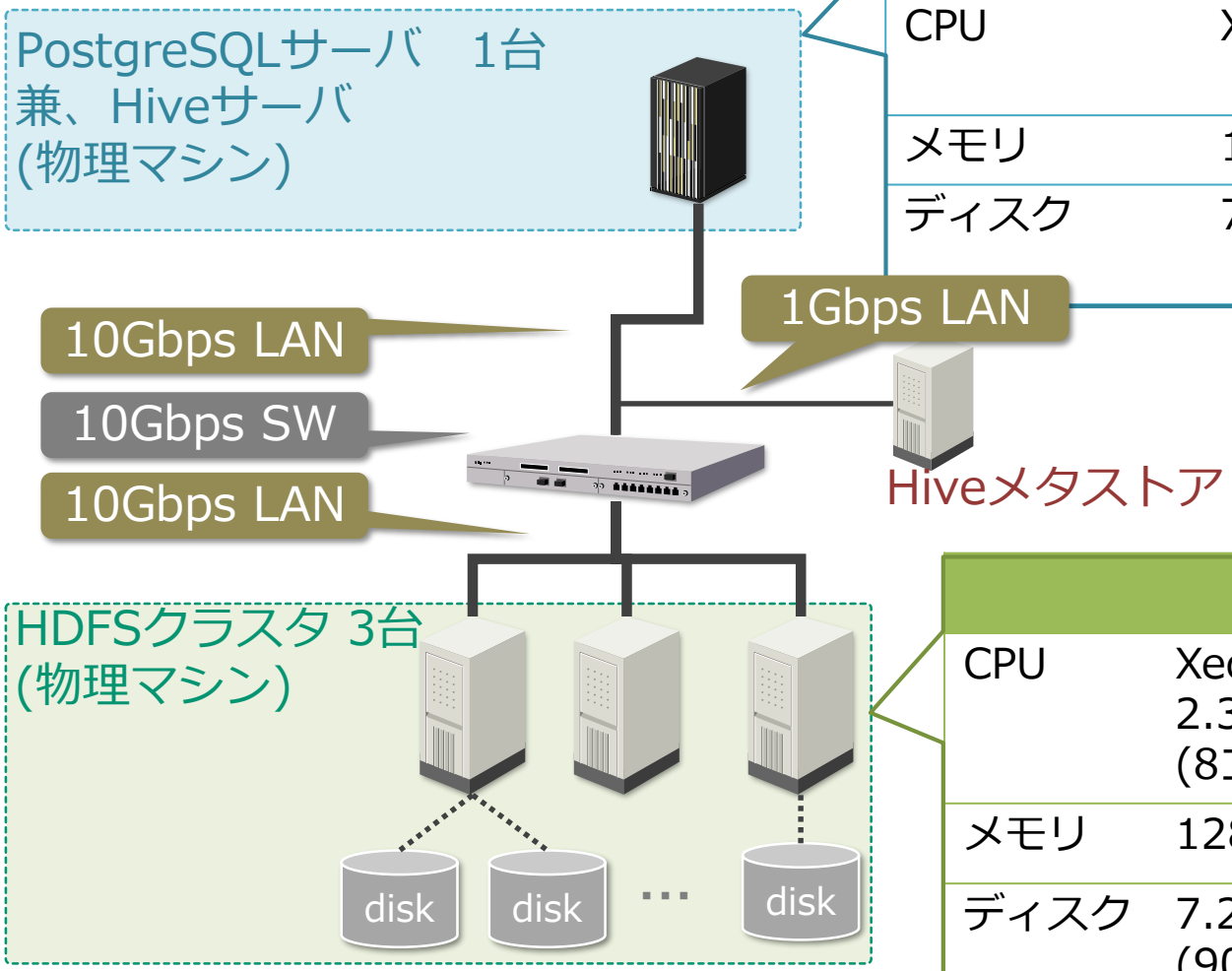



© Hitachi, Ltd. 2017. All rights reserved.

# 2-6. 評価環境

HITACHI
Inspire the Next

## ■ 物理構成




| | PostgreSQLサーバ 兼、Hiveサーバ |
|---|---|
| CPU | Xeon E5-2470 2.3GHz x 1 （8 コア、HT有効） |
| メモリ | 128 GB |
| ディスク | 7.2TB （900GB x 8台） |

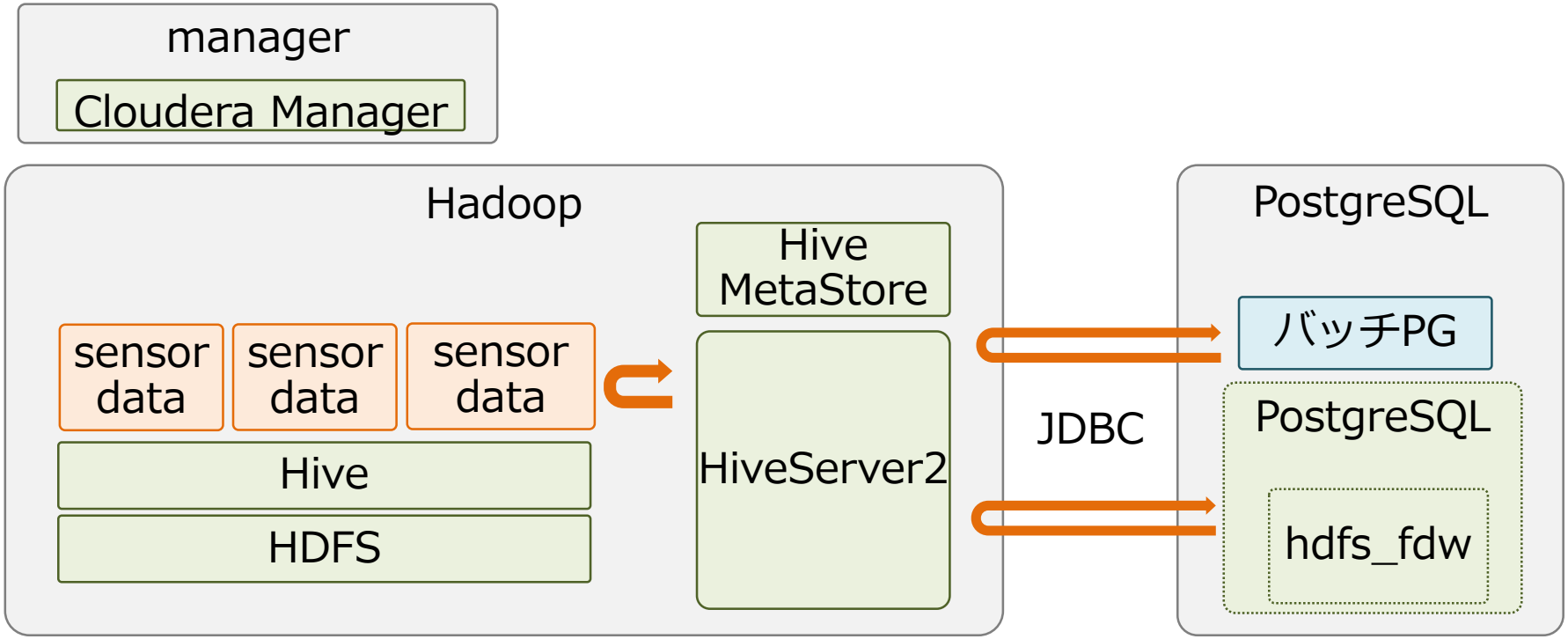
| | 1ノード | 3ノード合計 |
|---|---|---|
| CPU | Xeon E5-2470 2.3GHz x 1 (8コア、HT有効) | 48 コア |
| メモリ | 128 GB | 384 GB |
| ディスク | 7.2 TB (900GB x 8台) | 21.6 TB |

© Hitachi, Ltd. 2017. All rights reserved.

# 2-6. 評価環境

HITACHI
Inspire the Next

## ■ 論理構成




| ソフトウェア構成 |
| --- |
| RHEL 6.7 (x86_64) |
| CDH 5.11.1 |

| ソフトウェア構成 |
| --- |
| RHEL 6.7 (x86_64) |
| PostgreSQL 10.0 |
| hdfs_fdw 2.0.2 |

© Hitachi, Ltd. 2017. All rights reserved.

# 2-7. 評価1の概要

HITACHI
Inspire the Next

## ■ シナリオ

IoTデータの「見える化」サービスを行うためのバッチ処理作成。
今回は、具体的な例として、月初に前月分の工場内のセンサーデータ(IoT)を取得、時系列分析を行い、業務システムデータ(RDB)と組合せて「見える化」するための統計データ（見える化用データ）を作成する。

## ■ 評価観点

下記２種類のプログラム（以下、バッチPG）を作成し、以下の内容で比較評価する。

・バッチPGでPostgreSQLとhadoopにアクセスし、PG内で連携する
・バッチPGはPostgreSQLにアクセスし、FDWでHadoopに連携する

## ■ 評価内容

①コスト（設計、開発、構築、テスト、運用・保守等）面や業務変化対応の比較評価
②性能面の比較評価

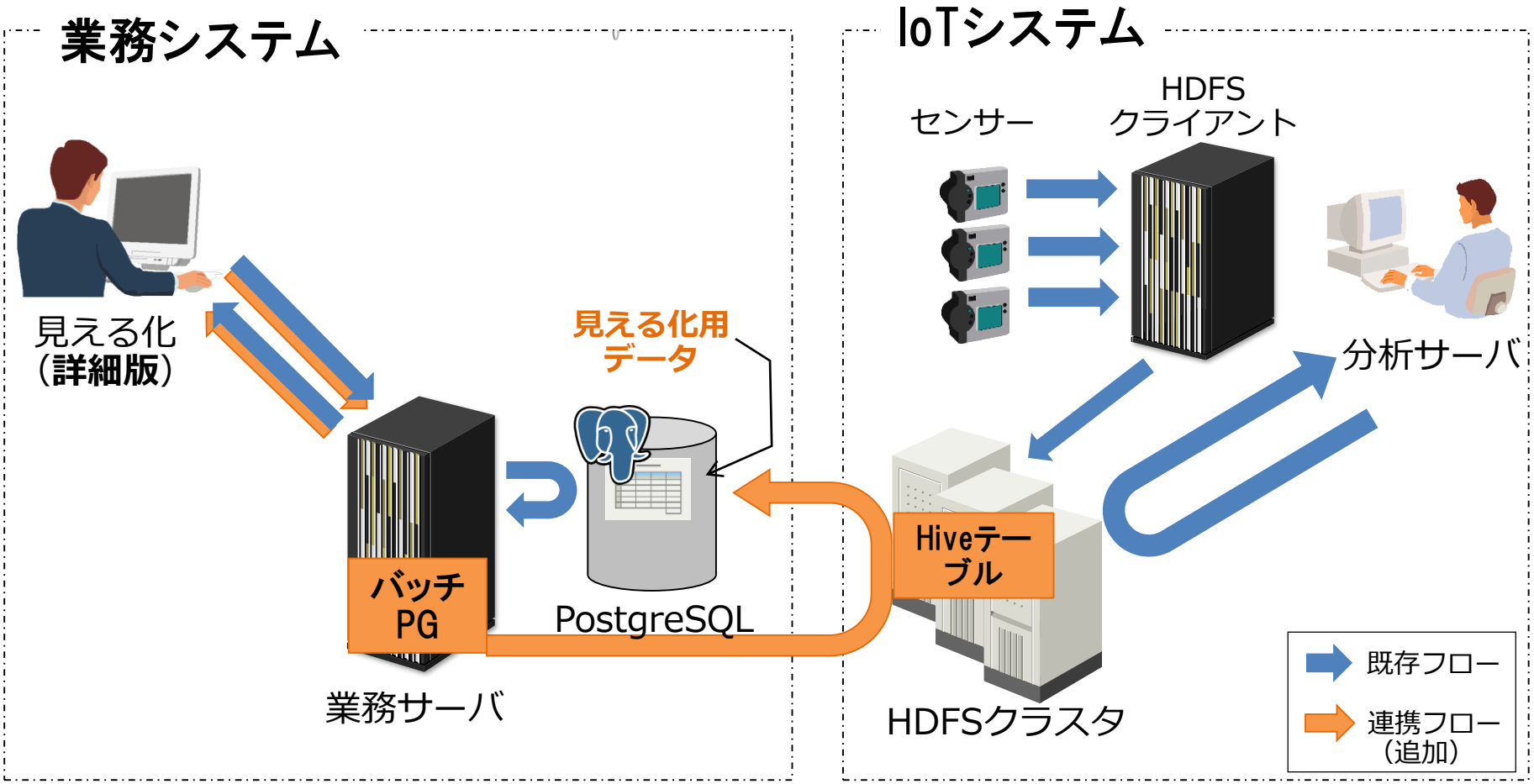
© Hitachi, Ltd. 2017. All rights reserved.

# 2-7. 連携後のシステム

HITACHI
Inspire the Next

## ■ アプリケーションで連携

アプリケーションで連携する場合のイメージを以下に示す。
（オレンジの部分が評価対象）



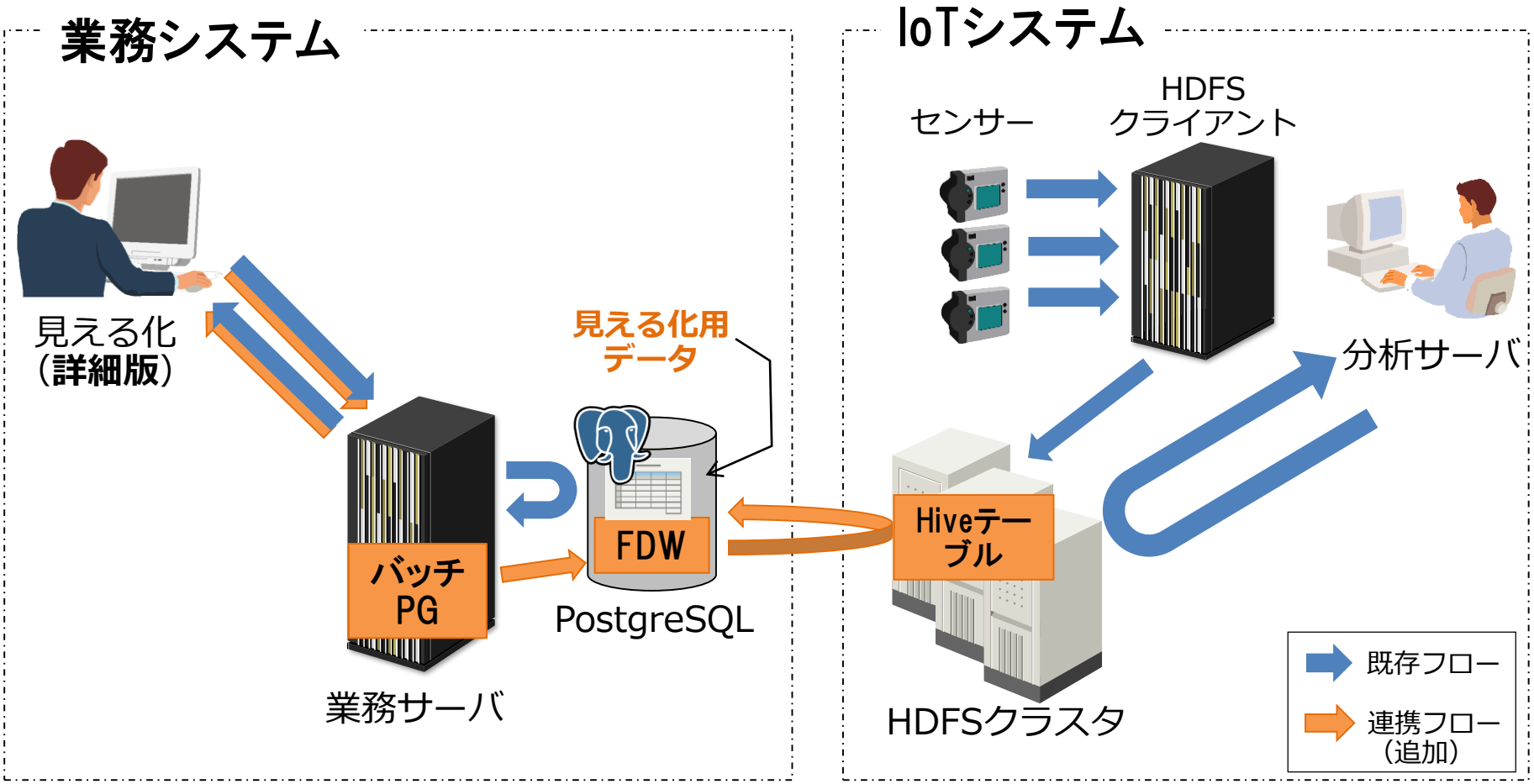

© Hitachi, Ltd. 2017. All rights reserved.

# 2-7. 連携後のシステム

HITACHI
Inspire the Next

## ■ FDWで連携

FDWで連携する場合のイメージを以下に示す。
（オレンジの部分が評価対象）

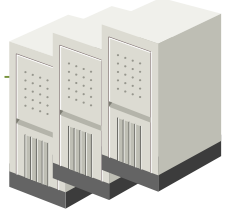



© Hitachi, Ltd. 2017. All rights reserved.

# 2-7. 評価イメージ

HITACHI
Inspire the Next

## ■ 処理概要



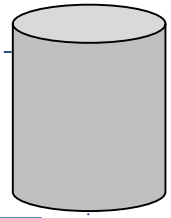
HDFSクラスタ
(Hiveテーブル)

**meter_dataテー**
**(センサーデータ)**

| Column | Type |
|---|---|
| meter_id | int |
| date_time | timestamp |
| usage | double |
| year | smallint |
| month | smallint |
| day | small |

[データ量]
10万センサ
1分毎送信
1年間分蓄積
合計：525.6億件

PostgreSQL



**metersテーブル**
**(対応データ)**

| Column | Type |
|---|---|
| fact_id | int |
| meter_id | i |

**factoryテーブル**
**(工場データ)**

| lumn | Type |
|---|---|
| _id | int |
| e | char |
| pref | char |
| city | char |
| weight | double |

**テーブル**
**データ)**

| | Type |
|---|---|
| fact_ | int |
| meter_id | int |
| month | int |

**タ生成**

**←テストを実施した結果、メモリ肥大化により、OOM-Killer発生！全てPostgreSQLで処理するのは現実的ではない**

© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 評価1改の概要

HITACHI
Inspire the Next

## ■ シナリオ

IoTデータの「見える化」サービスを行うためのバッチ処理作成。
今回は、具体的な例として、月初に前月分の工場内のセンサーデータ(IoT)を取得、時系列分析を行い、業務システムデータ(RDB)と組合せて「見える化」するための統計データ（見える化用データ）を作成する。

## ■ 評価観点

①Hadoop側で時系列分析を行ったデータを生成し、下記２種類のプログラム（以下、バッチPG）を作成し、以下の内容で比較評価する。
・バッチPGでPostgreSQLとhadoopにアクセスし、PG内で連携する
・バッチPGはPostgreSQLにアクセスし、FDWでHadoopに連携する
②Hadoop側で時系列分析を行ったデータを出力し、PostgreSQLにロードした上で、バッチPG内で連携する

## ■ 評価内容

①コスト（設計、開発、構築、テスト、運用・保守等）面や業務変化対応の比較評価
②性能面の比較評価（分析データの1か月分）

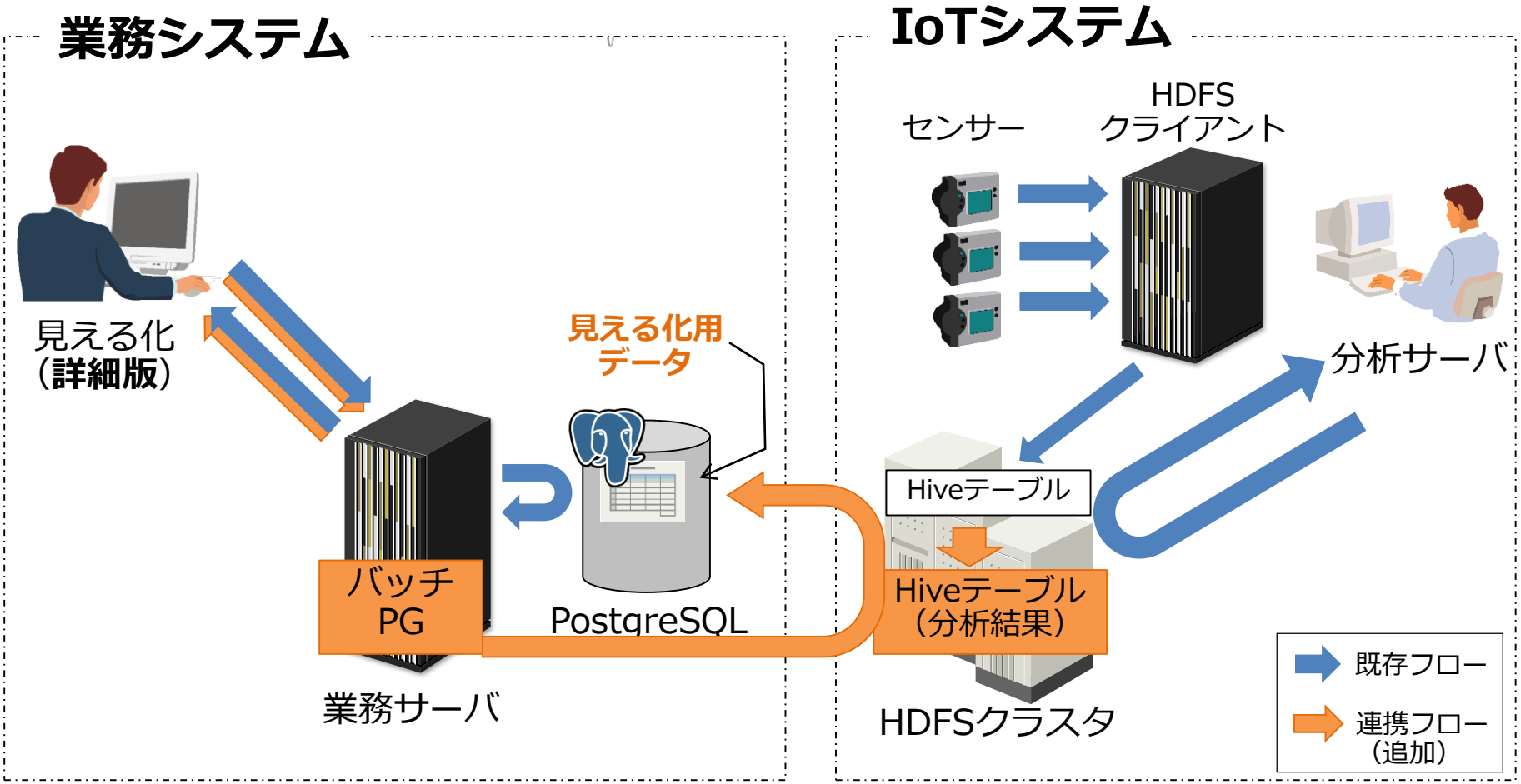
© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 連携後のシステム

HITACHI
Inspire the Next

## ■ アプリケーションで連携

分析データをHadoop側で作成し、アプリケーションで連携する場合のイメージを以下に示す。（オレンジの部分が評価対象）



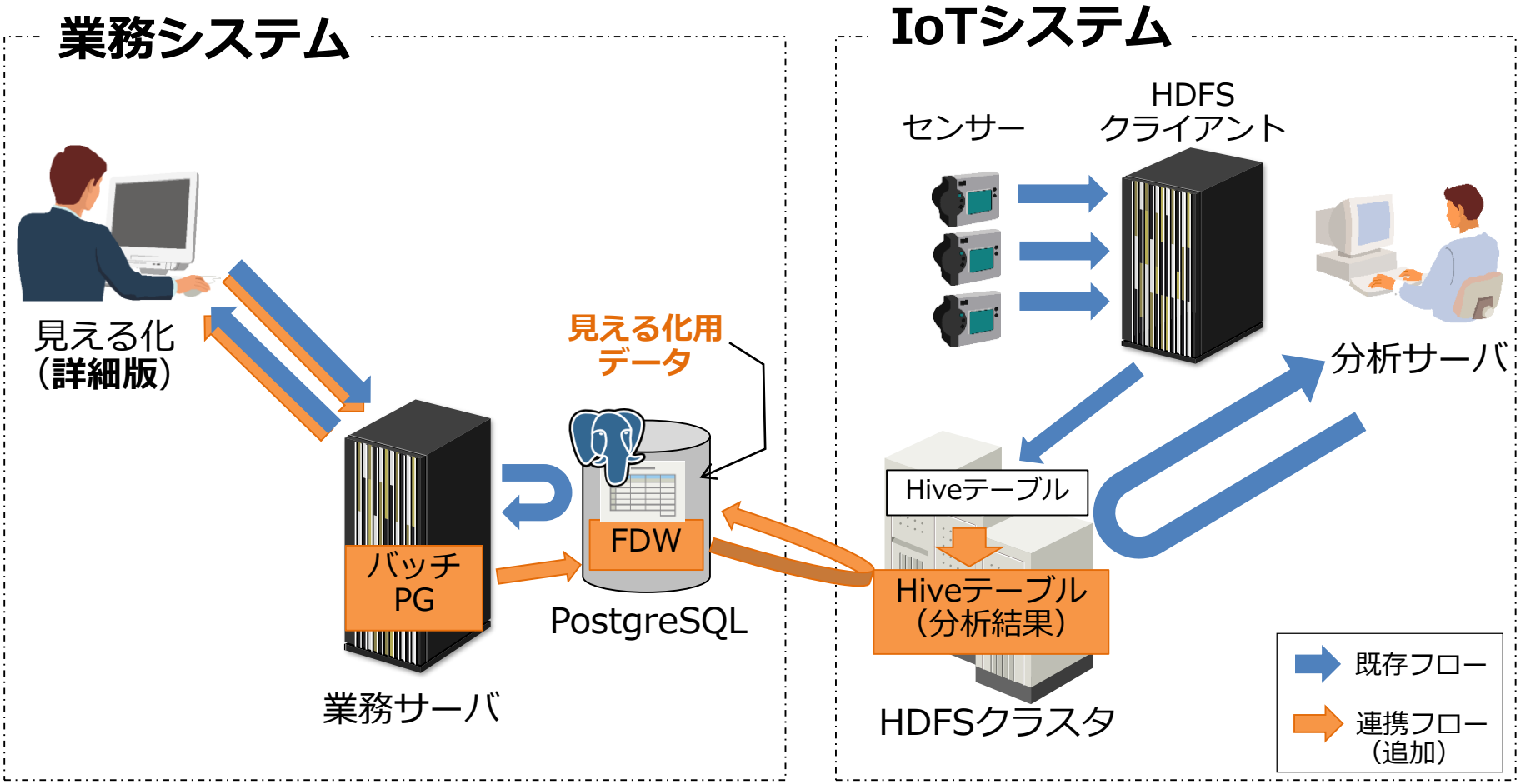

© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 連携後のシステム

HITACHI
Inspire the Next

## ■ FDWで連携

分析データをHadoop側で作成し、 FDWで連携する場合のイメージを以下に示す。（オレンジの部分が評価対象）



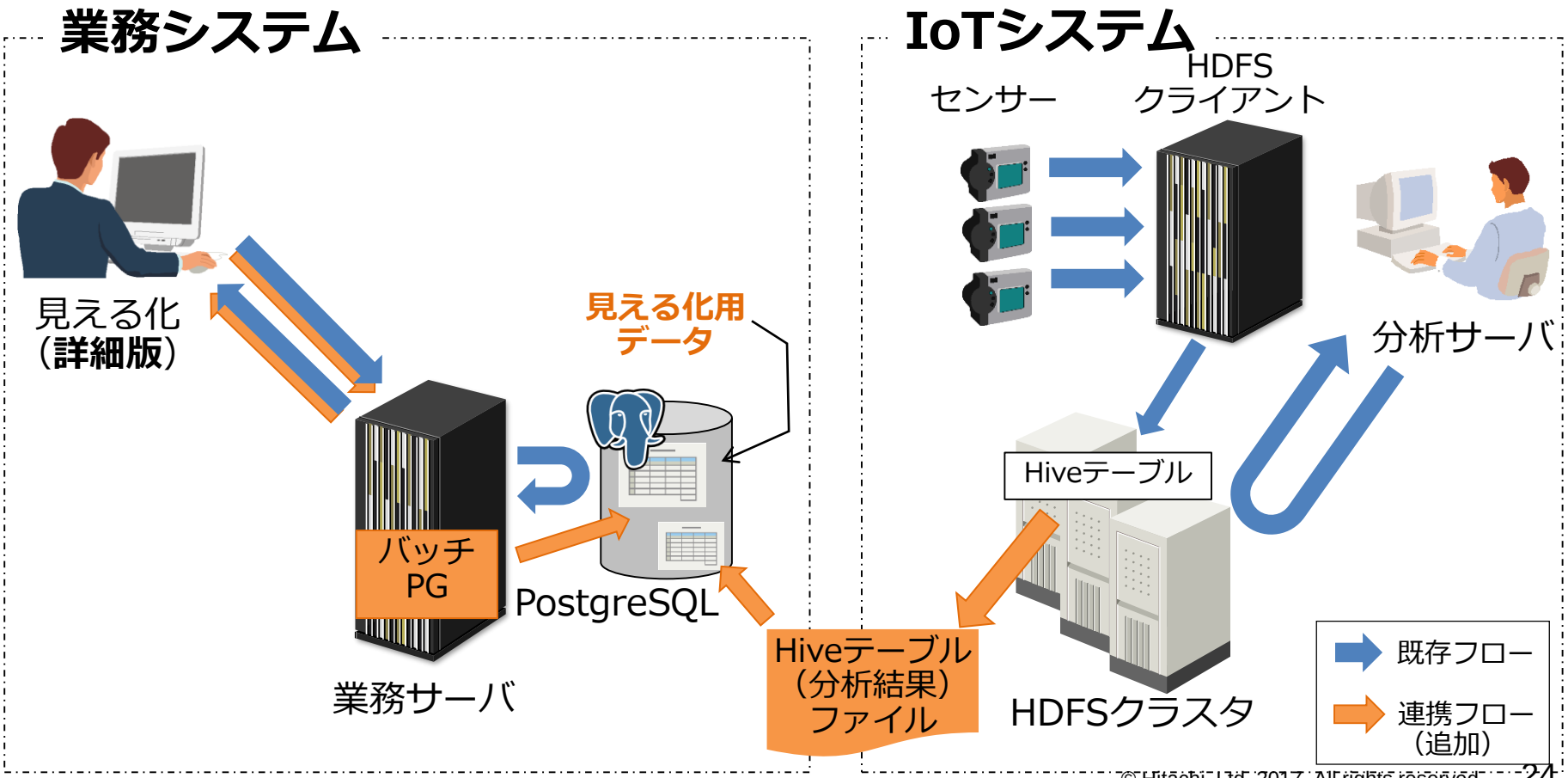

© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 連携後のシステム

HITACHI
Inspire the Next

## ■ PostgreSQLにロードして連携

Hadoop側で作成した分析データをファイル出力し、PostgreSQLにロード、PostgreSQLで連携する場合のイメージを以下に示す。（オレンジの部分が評価対象）

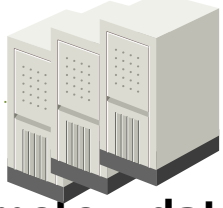



© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 評価イメージ

HITACHI
Inspire the Next

## ■ 処理概要



HDFSクラスタ
(Hiveテーブル)

**meter_dataテーブル**

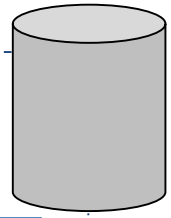
| Column | Type |
|---|---|
| meter_id | int |
| date_time | timestamp |
| usage | double |
| year | smallint |
| month | smallint |
| day | smallint |

**meter_analyzeテーブル**
**(集計結果テーブル)**

| Column | Type |
|---|---|
| meter_id | int |
| month | int |
| is_holiday | boolean |
| hour | int |
| usage | double |

データ生成

PostgreSQL



**metersテーブル**
**(対応データ)**

| Column | Type |
|---|---|
| fact_id | int |
| meter_id | int |

**factoryテーブル**
**(工場データ)**

| Column | Type |
|---|---|
| fact_id | int |
| type | char |
| pref | char |
| city | char |
| weight | double |

**Supportテーブル**
**(「見える化」データ)**

| Column | Type |
|---|---|
| fact_id | int |
| meter_id | int |
| month | int |
| is_holiday | boolean |
| hour | int |
| usages | double |

© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 評価設計

HITACHI
Inspire the Next

## ■ PostgreSQLパラメータ

PostgreSQLが運用中であることを想定したパラメータにしている。

| パラメータ | 値 | 目的 |
|---|---|---|
| shared_buffers | 8GB | DBリソースにあわせる |
| maintenance_work_mem | 2GB | VACUUMの高速化 |
| wal_level | replica | 運用中を想定 |
| checkpoint_timeout | 30min | 運用中を想定 |
| checkpoint_completion_target | 0.9 | CHECKPOINT処理の平準化 |
| archive_mode | on | 運用中を想定 |
| max_wal_senders | 2 | HA構成を想定 |
| hot_standby | on | 運用中を想定 |
| effective_cache_size | 16GB | DBリソースにあわせる |
| logging_collector | on | ログ出力 |
| autovacuum | on | 運用中を想定 |
| hdfs_fdw.jvmpath | （環境依存のため省略） | hdfs_fdwの設定 |
| hdfs_fdw.classpath | （環境依存のため省略） | |

© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 評価結果①

HITACHI
Inspire the Next

まず初めに、気になる設計・開発、運用・保守におけるコスト面、業務変化への対応の難易度について、机上評価を行った。

| 方式<br>フェーズ | アプリ連携 | FDW連携 |
|---|---|---|
| 設計・開発 | • やりたいことを細かいレベルで実現しやすい<br>• 開発コスト、テストコストが大きくなる（アプリで連携やSQLでできるような結合処理の実装が必要） | • 開発コストが少ない<br>• FDWを使うに当たっての選定や構築コストがかかる |
| 運用・保守 | • 障害発生時の切り分けが容易、内部だけで対応可能<br>• RDBやhadoopの接続はトラブルに陥りやすく、保守改修コストが大きい | • 活発なコミュニティであれば品質向上が図れる<br>• 障害時の切り分けコストが大きい（DBAとの連携が必要） |
| 要件追加・変更 | • 柔軟な対応ができる<br>• 小さな変更レベルでもアプリの改修が必要となるため、コストがかかる | • 容易に対応できる（SQLの修正のみ…等）<br>• FDWとしての制約がある（例、HadoopアクセスにはHiveが必要） |

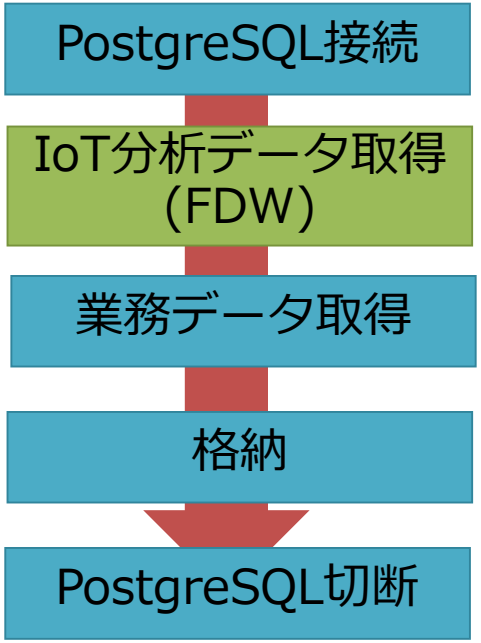
© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 評価結果①

HITACHI
Inspire the Next

バッチPGの処理概要から、IoT分析データ取得部分のコストを比較する。

## ■ FDW連携のバッチPG処理概要




```
pstmt = con.prepareStatement("select b.fact_id, a.meter_id,
            a.month, a.is_holiday, a.hour, a.usage from
            meter_data_custom a, meters b where ((a.meter_id
            = b.id) and (a.year_month = ?))");
pstmt.setInt(1, year_month);
res = pstmt.executeQuery();
while (res.next()) {
  // 1件ずつ取得
}
```

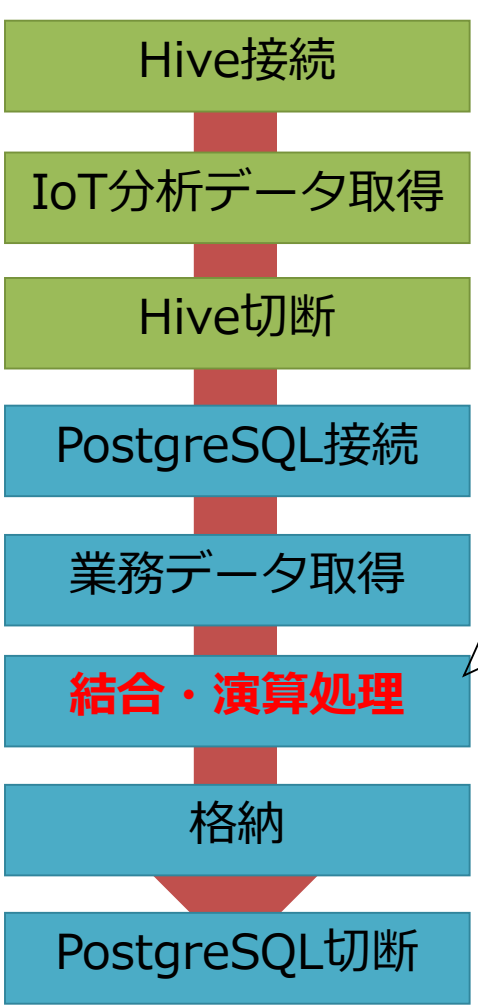
© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 評価結果①

HITACHI
Inspire the Next

バッチPGの処理概要から、IoT分析データ取得部分のコストを比較する。

## ■ アプリ連携のバッチPG処理概要




```
Class.forName(driverHive);
Connection con = null;
con = DriverManager.getConnection(urlHive, userHive,
            passwordHive);
pstmt = con.prepareStatement("select * from
            meter_data_avg_2017 where year_month = ?");
pstmt.setInt(1, year_month);
res = pstmt.executeQuery();
while (res.next()) {
  // 1件ずつ取得
}
```

```
                    :
 for (MeterObject meter:meterMap)
  {
    pstmt.setInt(1, meter_fact_map.get(meter.getMeter_id()));
    pstmt.setInt(2, meter.getMeter_id());
    pstmt.setInt(3, meter.getYear_month());
    pstmt.setBoolean(4, meter.getIs_holiday());
    pstmt.setInt(5, meter.getHour());
    pstmt.setDouble(6, meter.getUsage());
    pstmt.addBatch();
 }                  :
```

**アプリ連携では、接続処理やSQLレベルの結合演算処理も独自で開発する必要があり、複雑さの高い業務システムではコストが高い。運用・保守においても継続メンテが必要。**

© Hitachi, Ltd. 2017. All rights reserved.

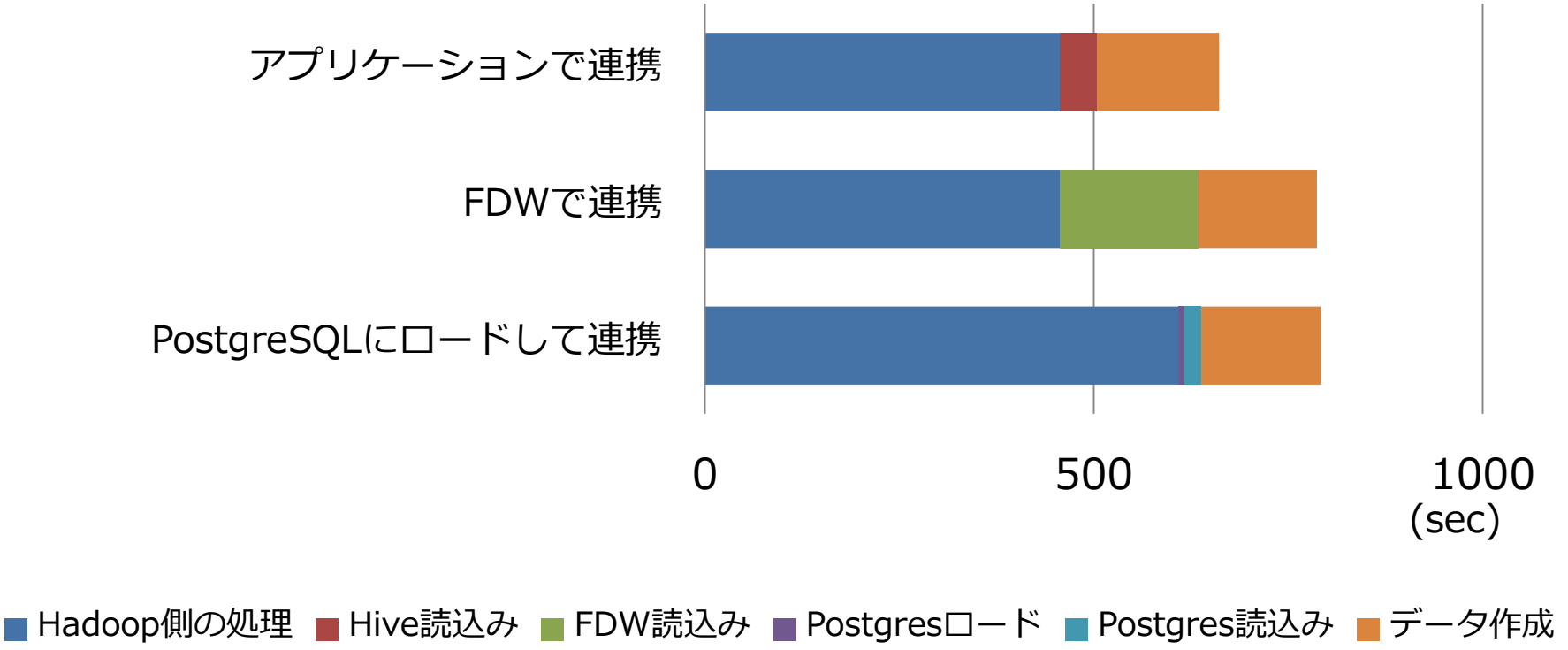
# 2-8. 評価結果②

HITACHI
Inspire the Next

次に、性能面の比較評価を行った。

## ■ 性能面の比較評価

①アプリケーション実行時間の比較評価




**連携方式による顕著な差異は見られなかったので、性能以外の面に着目すると良い。**

※hadoop, PostgreSQLのキャッシュは初期起動時に削除

© Hitachi, Ltd. 2017. All rights reserved.

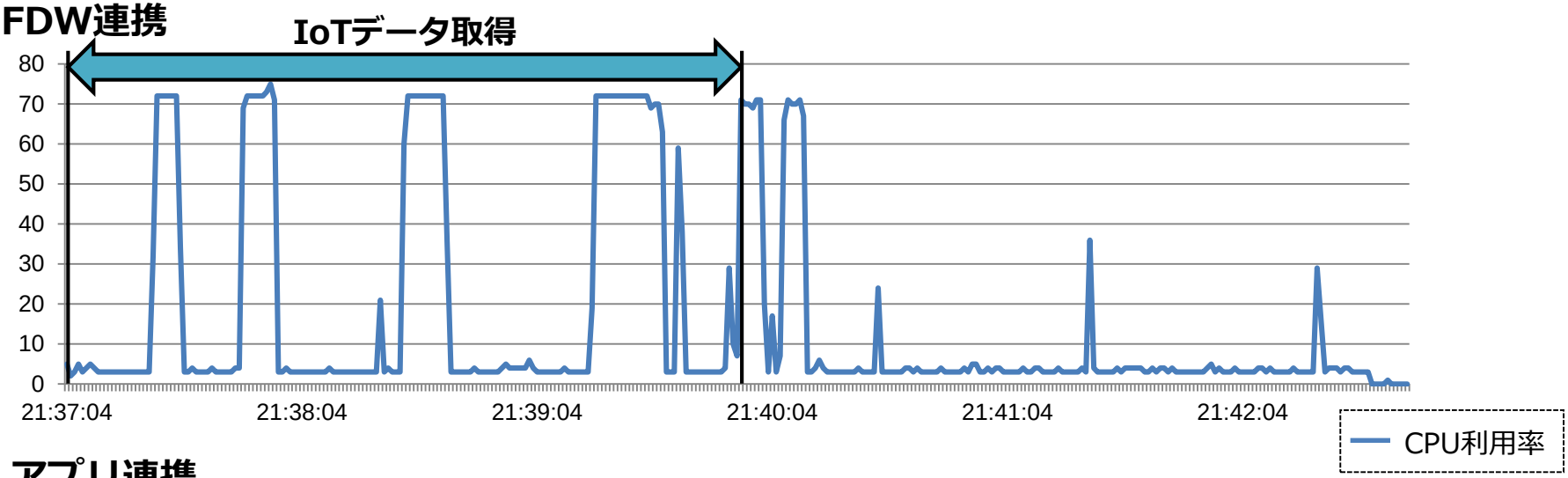
# 2-8. 評価結果②

HITACHI
Inspire the Next

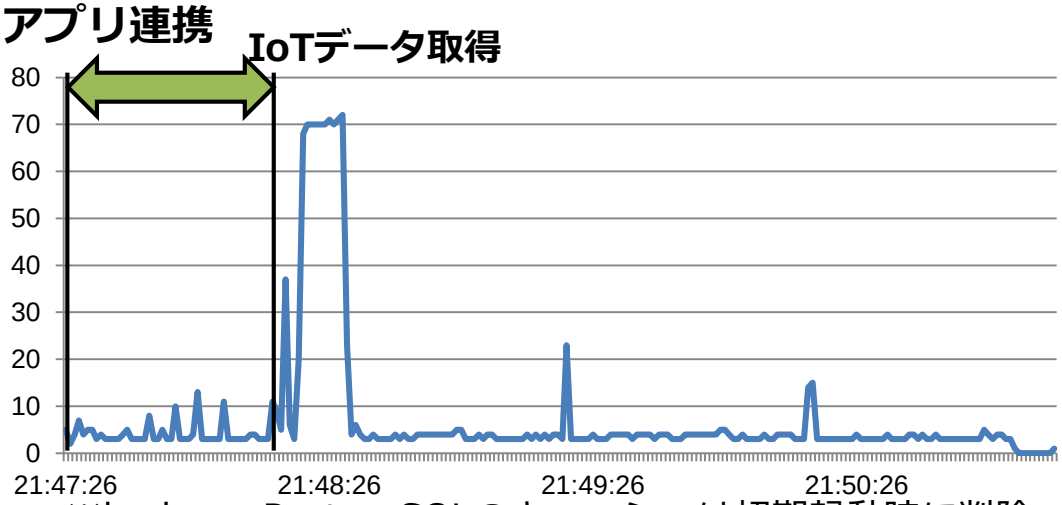
## ■ 性能面の比較評価

②システムへの影響の比較評価

- **FDW連携**




- **アプリ連携**




※hadoop, PostgreSQLのキャッシュは初期起動時に削除

**FDW連携時はCPU資料率が高騰する頻度が高かった。マスターデータは業務DBにあり、業務に影響が出る可能性があるため、事前の負荷検証は必ず必要と考える。負荷状況によってはCPU増設等ハード強化も検討要。**

**※なお、メモリの利用率については連携方式による差がなかったため、表記は省く。**

© Hitachi, Ltd. 2017. All rights reserved.

# 2-8. 評価結果②

HITACHI
Inspire the Next

## ■ 性能面の比較評価

③hdfs_fdwのクエリプランの評価

- **PostgreSQL**

```
hdfsfdw=# explain analyze select count(*) from meter_data_custom where year_month =
'201701';
                                                QUERY PLAN
-----------------------------------------------------------------------------------------
Aggregate  (cost=1100002.50..1100002.51 rows=1 width=8) (actual
time=39792.123..39792.124
 rows=1 loops=1)
   ->  Foreign Scan on meter_data_custom  (cost=100000.00..1100000.00 rows=1000
width=0) (actual time=306.846..38433.607 rows=4800000 loops=1)
 Planning time: 112.417 ms
 Execution time: 40187.927 ms
(4 rows)
```

- **Hiveログ**

```
Compiling command(queryId=hive_id): SELECT * FROM default.meter_data_avg_2017
WHERE ((year_month = '201701'))
Executing command(queryId=hive_id): SELECT * FROM default.meter_data_avg_2017
WHERE ((year_month = '201701'))
ORC pushdown predicate: leaf-0 = (EQUALS year_month 201701)
Reading ORC rows from
hdfs://pgmaster:8020/user/hive/warehouse/meter_data_avg_2017/year_month=201701/
000000_0 with {include: [true, true, true, true, true], offset: 0, length:
9223372036854775807, sarg: leaf-0 = (EQUALS year_month 201701)
expr = leaf-0, columns: ['null', 'meter_id', 'is_holiday', 'hour', 'usage']}
```

→ **Hadoop側での分析データは月単位で作成しているが、対象ファイルのみアクセス(pushdown)することでI/O削減による高速化ができていることがわかる。**

© Hitachi, Ltd. 2017. All rights reserved.

# 2-9. 評価2の概要

HITACHI
Inspire the Next

## ■ シナリオ

次に、hadoopに格納されているIoTデータをPostgreSQLに統合する場合のデータ移行を検討する。
今回は、各システムの業務特性が異なることからデータベースクラスタを新たに作成し、IoTデータは月ごとのパーティショニング構成を想定して、1か月分を子テーブルにデータを挿入した。

## ■ 評価観点

下記3つの方法で、移行の比較評価をする。
① FDWを使ってIoTデータをhadoopからPostgreSQLに移行する
(1) SQL(INSERT-SELECT文)で直接移行する方法　･･･(a)
(2) SQL(COPY文)で一度ファイルに出力し、そのファイルを使って移行する方法　･･･(b)
② hadoopからIoTデータをファイルで出力し、PostgreSQLに移行する（以降、「手動で移行」と略します。）　･･･(c)

## ■ 評価内容

各評価観点の1か月分のデータの移行時間を評価する

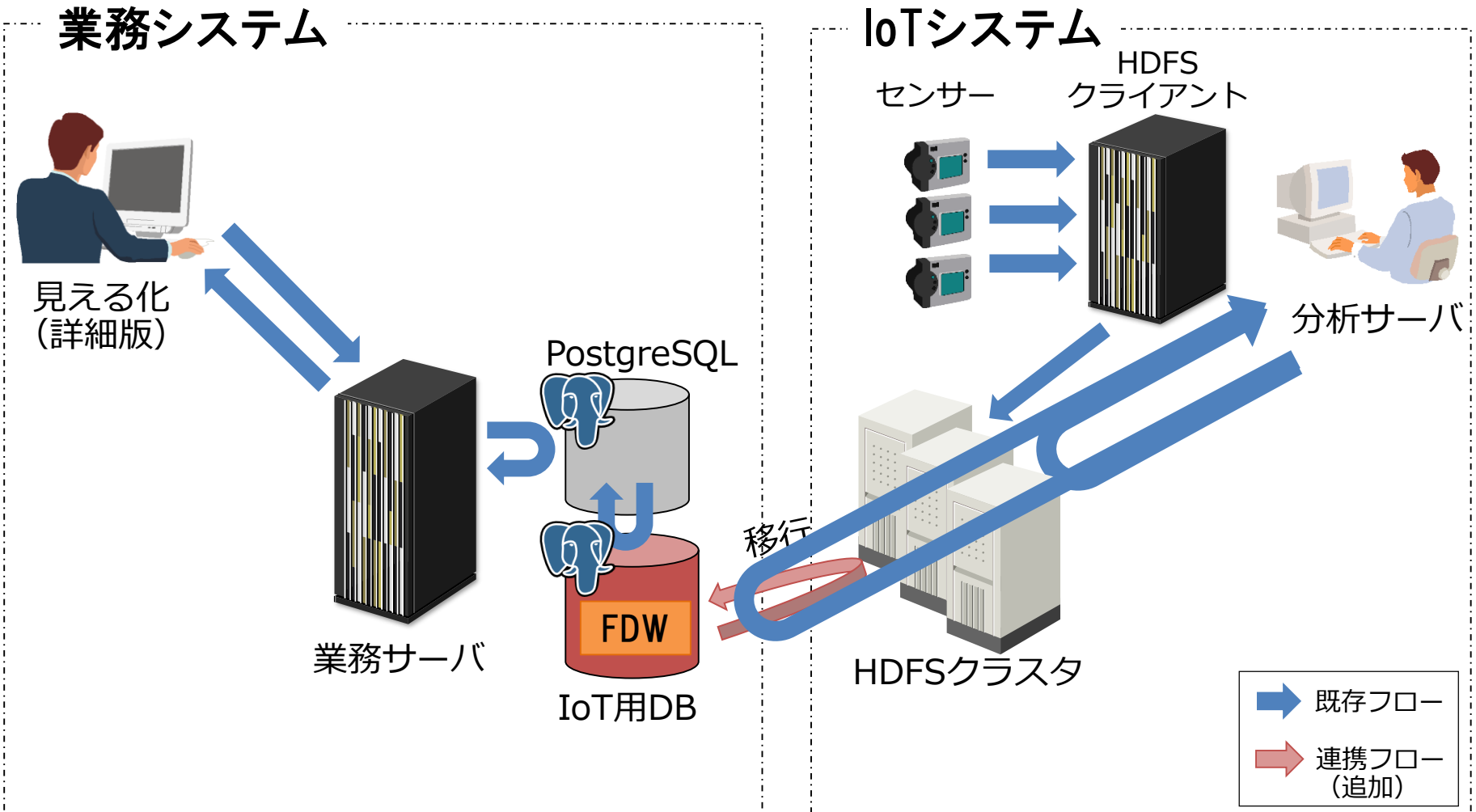
© Hitachi, Ltd. 2017. All rights reserved.

# 2-9. 移行後のシステム

HITACHI
Inspire the Next

## ■ FDWで移行

FDWで移行した場合のイメージを以下に示す。（赤の部分が評価対象）



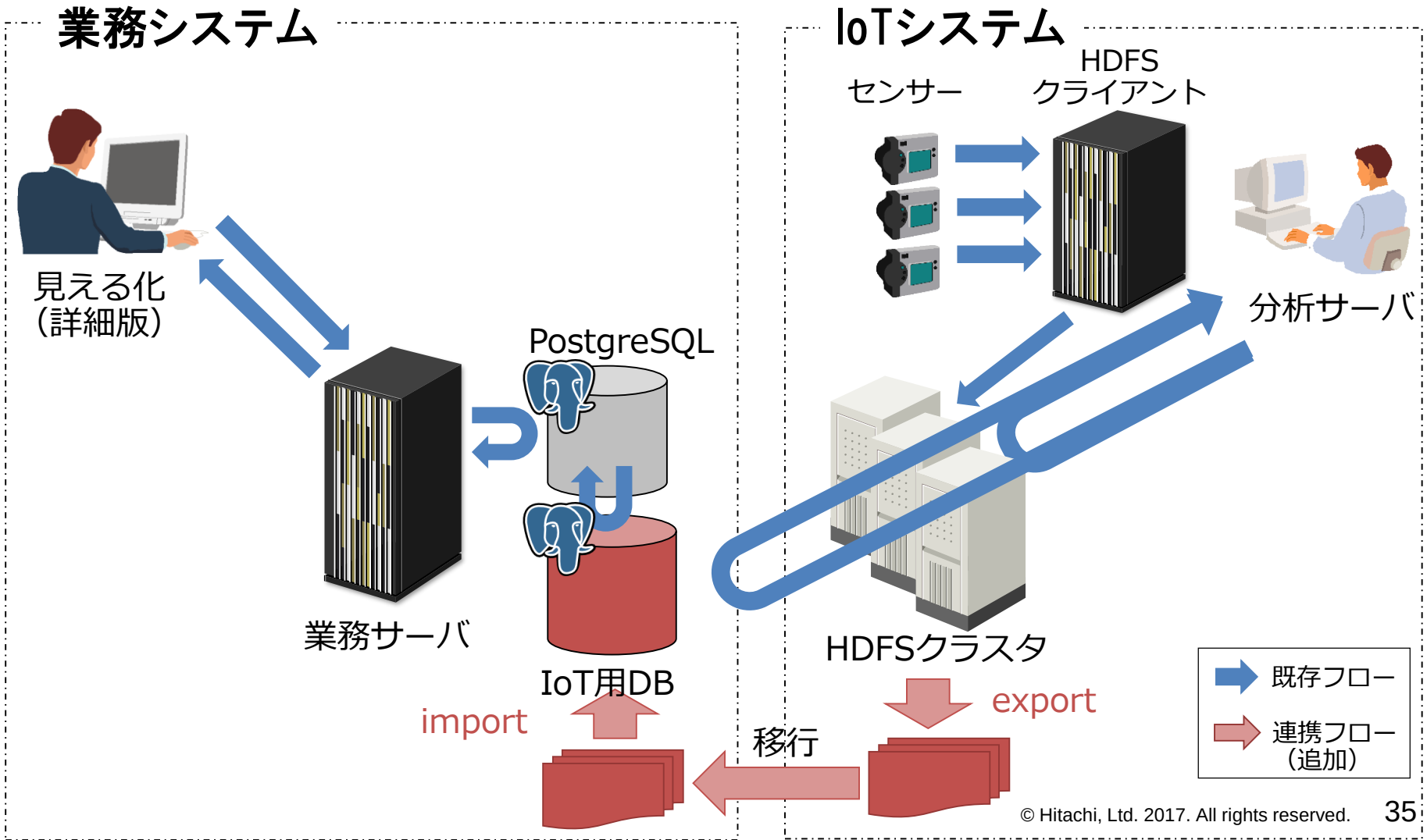

© Hitachi, Ltd. 2017. All rights reserved.

# 2-9. 移行後のシステム

HITACHI
Inspire the Next

## ■ 手動で移行

手動で移行した場合のイメージを以下に示す。（赤の部分が評価対象）




© Hitachi, Ltd. 2017. All rights reserved.

# 2-9. 評価設計

HITACHI
Inspire the Next

## ■ PostgreSQLパラメータ

新しくPostgreSQLを立てるため、データ移行に適した設定にした。

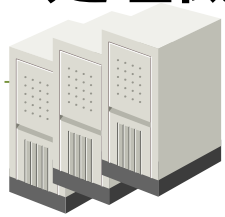
| パラメータ | 値 | 目的 |
|---|---|---|
| shared_buffers | 8GB | DBリソースにあわせる |
| maintenance_work_mem | 2GB | VACUUMの高速化 |
| wal_level | minimal | WAL書込みを極力減らす |
| synchronous_commit | off | 高速化（失敗すれば再実行すればよい） |
| checkpoint_timeout | 1d | CHECKPOINT実行抑止 |
| max_wal_size | 200GB | CHECKPOINT実行抑止 |
| checkpoint_completion_target | 0.9 | CHECKPOINT処理の平準化 |
| archive_mode | off | 移行のためアーカイブ不要 |
| effective_cache_size | 16GB | DBリソースにあわせる |
| logging_collector | on | ログ出力 |
| autovacuum | off | 移行中のVACUUM実行を防ぐ |
| hdfs_fdw.jvmpath | （環境依存のため省略） | hdfs_fdwの設定 |
| hdfs_fdw.classpath | （環境依存のため省略） | |

© Hitachi, Ltd. 2017. All rights reserved.

# 2-9. 評価イメージ

HITACHI
Inspire the Next

## ■ 処理概要



HDFSクラスタ
(hiveテーブル)

**meter_dataテーブル**
**(センサーデータ)**

| Column | Type |
|---|---|
| meter_id | int |
| date_time | timestamp |
| usage | double |
| year | smallint |
| month | smallint |
| day | smallint |

[データ量]
10万センサ
1分毎送信
1年間分蓄積
合計：525.6億件

データ移行

PostgreSQL

**metersテーブル**
**(対応データ)**

| Column | Type |
|---|---|
| fact_id | int |
| meter_id | int |

**factoryテーブル**
**(工場データ)**

| Column | Type |
|---|---|
| fact_id | int |
| type | char |
| pref | char |
| city | char |
| weight | double |

PostgreSQL

**meter_dataテーブル**
**(センサーデータ)**

| Column | Type |
|---|---|
| meter_id | int |
| date_time | timestamp |
| usage | double |

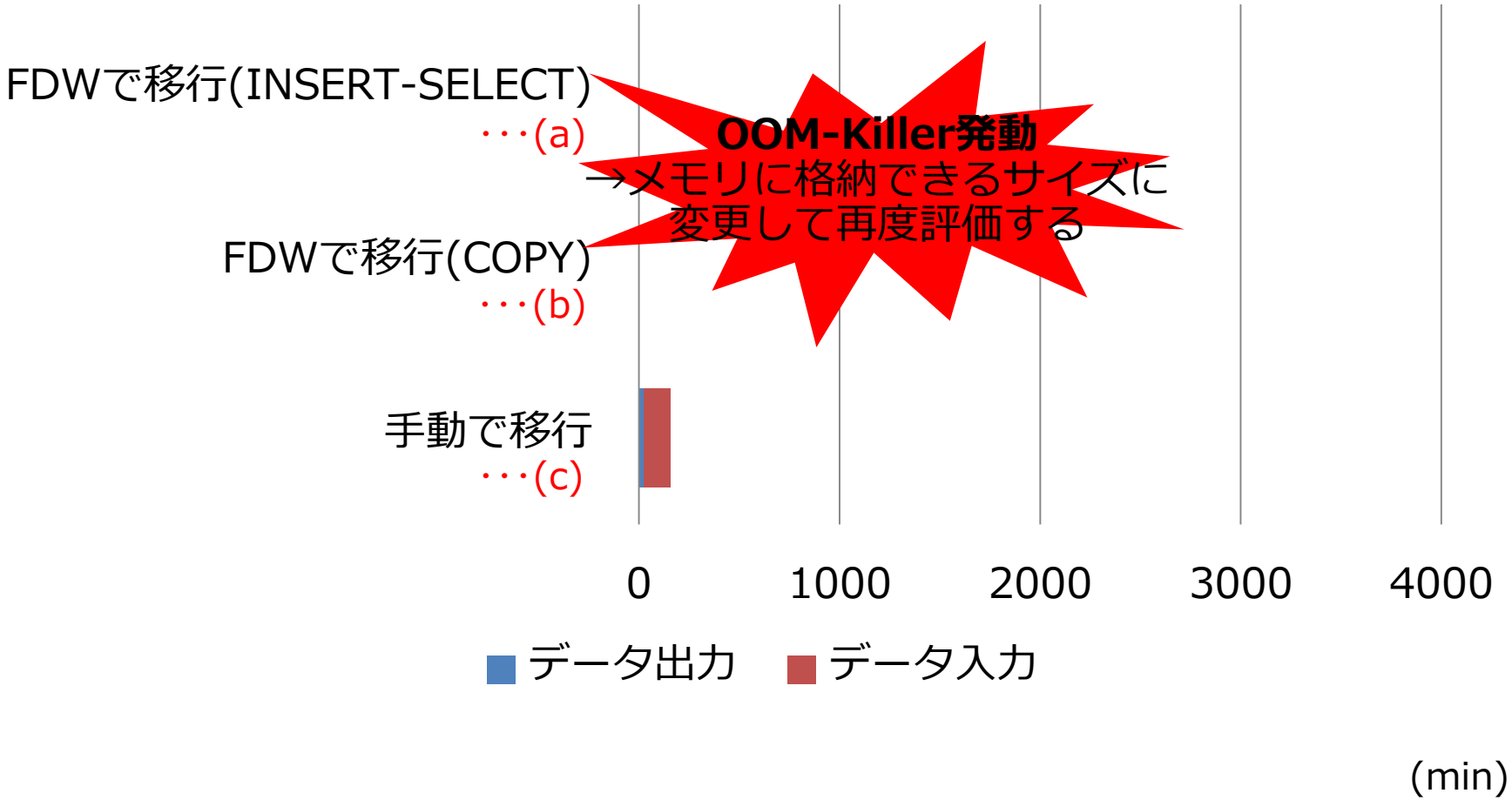
© Hitachi, Ltd. 2017. All rights reserved.

# 2-9. 評価結果

HITACHI
Inspire the Next

1か月分のデータを移行する場合のFDWを使った移行と、全て手動での移行との移行時間の比較評価を行った。

## ■ 移行時間の比較評価




※hadoop, PostgreSQLのキャッシュは初期起動時に削除

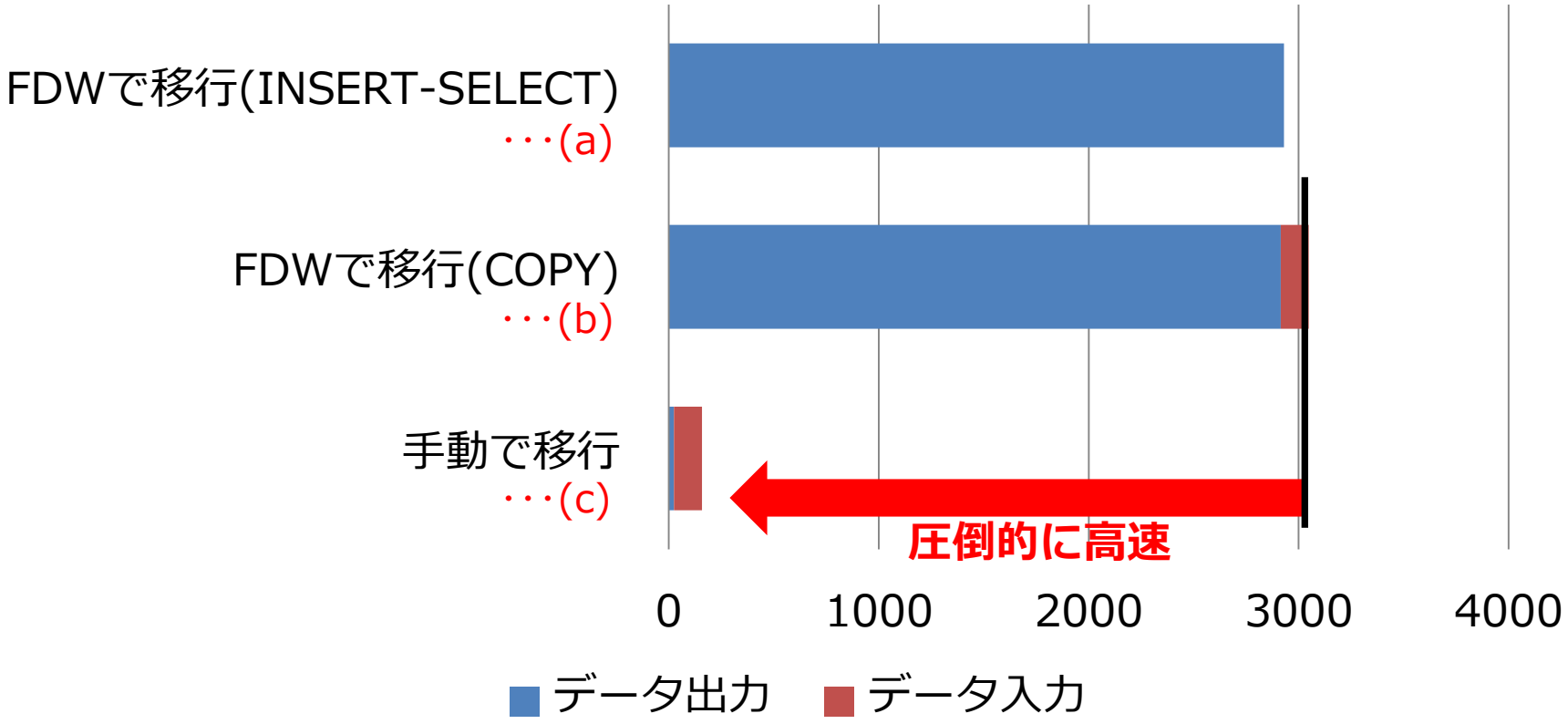
© Hitachi, Ltd. 2017. All rights reserved.

# 2-9. 評価結果

HITACHI
Inspire the Next

1か月分のデータを移行する場合のFDWを使った移行と、全て手動での移行との移行時間の比較評価を行った。

## ■ 移行時間の比較評価




(a),(b)は1日分ごとに移行を行なうSQLを1か月分処理した総計 (min)

※hadoop, PostgreSQLのキャッシュは初期起動時に削除

© Hitachi, Ltd. 2017. All rights reserved.

# 2-10. まとめ

HITACHI
Inspire the Next

IoT時代において、データから価値創出するにあたりシステム連携は必須であり、連携を実現するFDWは他の方法に比べて遜色なく有効であると感じた。

今回の検証でわかったように、何でもFDWに頼るのではなく、様々なコストに応じた対応を検討すべきである。

- 大量データ分析はHadoop側で行い、データ量を減らす。
- 移行は、既存の方法（手動で移行）のほうがコストが小さい。

© Hitachi, Ltd. 2017. All rights reserved.

# 3. 某DBからPostgreSQLへの移行事例
# ～TWX-21における移行事例～

© Hitachi, Ltd. 2017. All rights reserved.

# 3-1. 自己紹介

HITACHI
Inspire the Next

傳法谷　智　（でんぽうや　さとし）

・現職
　株式会社日立製作所
　IoT・クラウドサービス事業部
　アプリケーション第２部　統括主任技師

・職歴
　1993 年株式会社日立製作所入社。
　分散トランザクションマネージャ「uCosminexus OpenTP1」
　の開発に従事。
　1996 年から業務システムクラウドサービス「TWX-21」の開発
　及び運用業務を担当している。

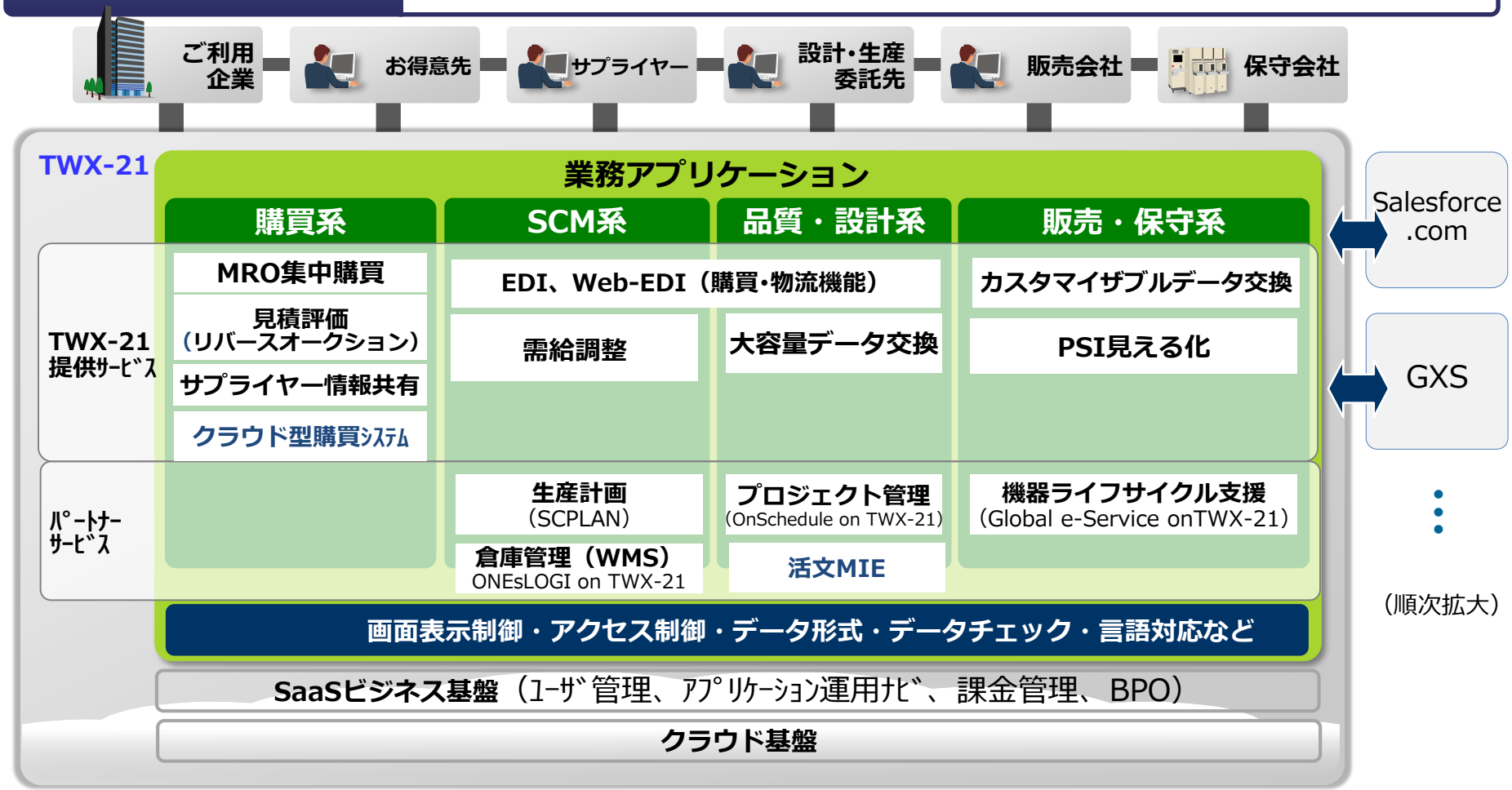
© Hitachi, Ltd. 2017. All rights reserved.

# 3-2. TWX-21とは

HITACHI
Inspire the Next

**業務システムクラウド**
**国内最大規模**

・企業向けの業務システムをＳａａＳ型で提供
・TWX-21会員約64,000社、約30カ国・地域、約31万ID




MRO：Maintenance, Repair and Operations　PSI：Production Sales Inventory
WMS：Warehouse Management System　BPO：Business Process Outsourcing　SCM：Supply Chain Management

© Hitachi, Ltd. 2017. All rights reserved.

# 3-3. TWX-21のグローバル展開

HITACHI
Inspire the Next

## 海外での主な利用地域

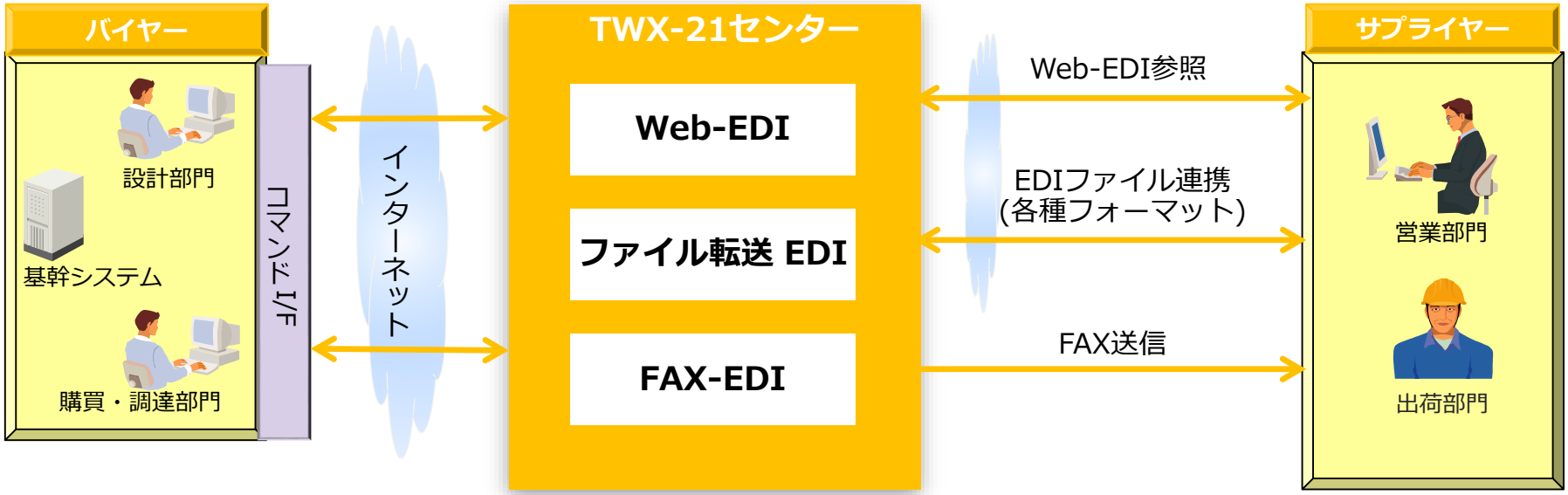
| データセンター | ヘルプデスク対応言語 | 現地契約可能な国 |
|---|---|---|
| 日本・中国 | 日本語・英語・中国語・<br>タイ語・インドネシア語・マレーシア語 | 日本、中国、タイ、<br>インドネシア |

© Hitachi, Ltd. 2017. All rights reserved.

# 3-4. プロジェクト概要

HITACHI
Inspire the Next

- サプライヤーが利用する形態に合わせて、TWX-21が伝送方式を変換することで
  バイヤー側のEDIを実現、建設業法の取引を含めてペーパーレス化を推進
- 注文書などの伝票データとともに、図面や仕様書などのデータも交換できるため業務効率が向上
- EDIシステムの管理コスト・運用負担を軽減でき、IT部門の要員を自社のコア業務にシフト可能




・販売終了製品のOS/ミドルウェア
　バージョンアップ対応不可
・機能エンハンス不可
・保守体制維持困難

アーキテクチャ含め
**リニューアル**
を実施することにした。

© Hitachi, Ltd. 2017. All rights reserved.

# 3-5. DBの選定

HITACHI
Inspire the Next

DBを選定するに当たり、8項目を得点制で比較検討実施。

| | 区分 | 重点比較項目 | | | | | | | 比較項目 | | | 評価結果 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | A | B | C | D | E | F | | G | H | | | 順位 |
| | | コスト | サポート | イベント駆動DBと業務DBを分離 | 性能 | 信頼性/拡張性 | 永続性 | 重点比較評価点小計 | 保守性 | 制限事項 | 評価点計 | 評価点合計 | |
| 1 | 製品A | 27 | 16 | -15 | 18 | 14 | 18 | 78 | 14 | 10 | 24 | 102 | 4 |
| 2 | 製品B | 12 | 12 | 0 | 20 | 20 | 20 | 84 | 20 | 12 | 32 | 116 | 2 |
| 3 | 製品C | 32 | 12 | 0 | 12 | 12 | 12 | 80 | 20 | 4 | 24 | 104 | 3 |
| ★ 4 | PostgreSQL | 40 | 20 | 0 | 12 | 12 | 12 | 96 | 20 | 4 | 24 | 120 | 1 |

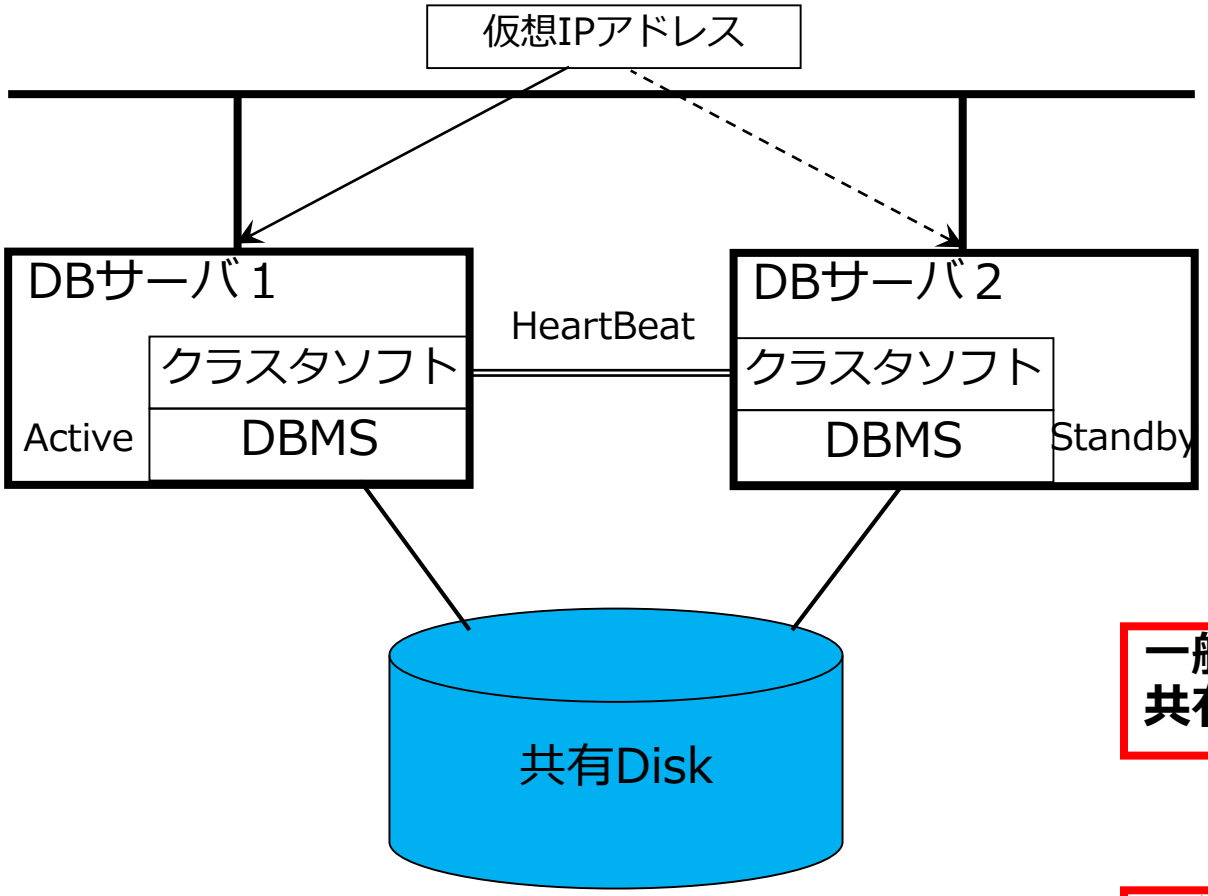
- PostgreSQLはコスト、サポート(弊社サポート部門)、実績、コミュニティ等で高得点
- 製品Aは、保守が困難。
- 製品Bは、仮想化上でのライセンスの懸案があり、極端なコスト増のため不可。
- 製品Cは、サポート、信頼性が低い。
- 他OSSでも実績多いものあるが、製品Bのベンダが所有していることから、将来性を鑑みて対象外とした。

© Hitachi, Ltd. 2017. All rights reserved.

# 3-6. 信頼性確保について

HITACHI
Inspire the Next

現行システム　　・・・クラスタソフトと共有Diskにより信頼性を確保




※サーバダウン、DBMSダウンした際クラスタソフトにより、HotStandbyで切替可能

**一般的なクラウド環境では、共有Diskは利用不可**

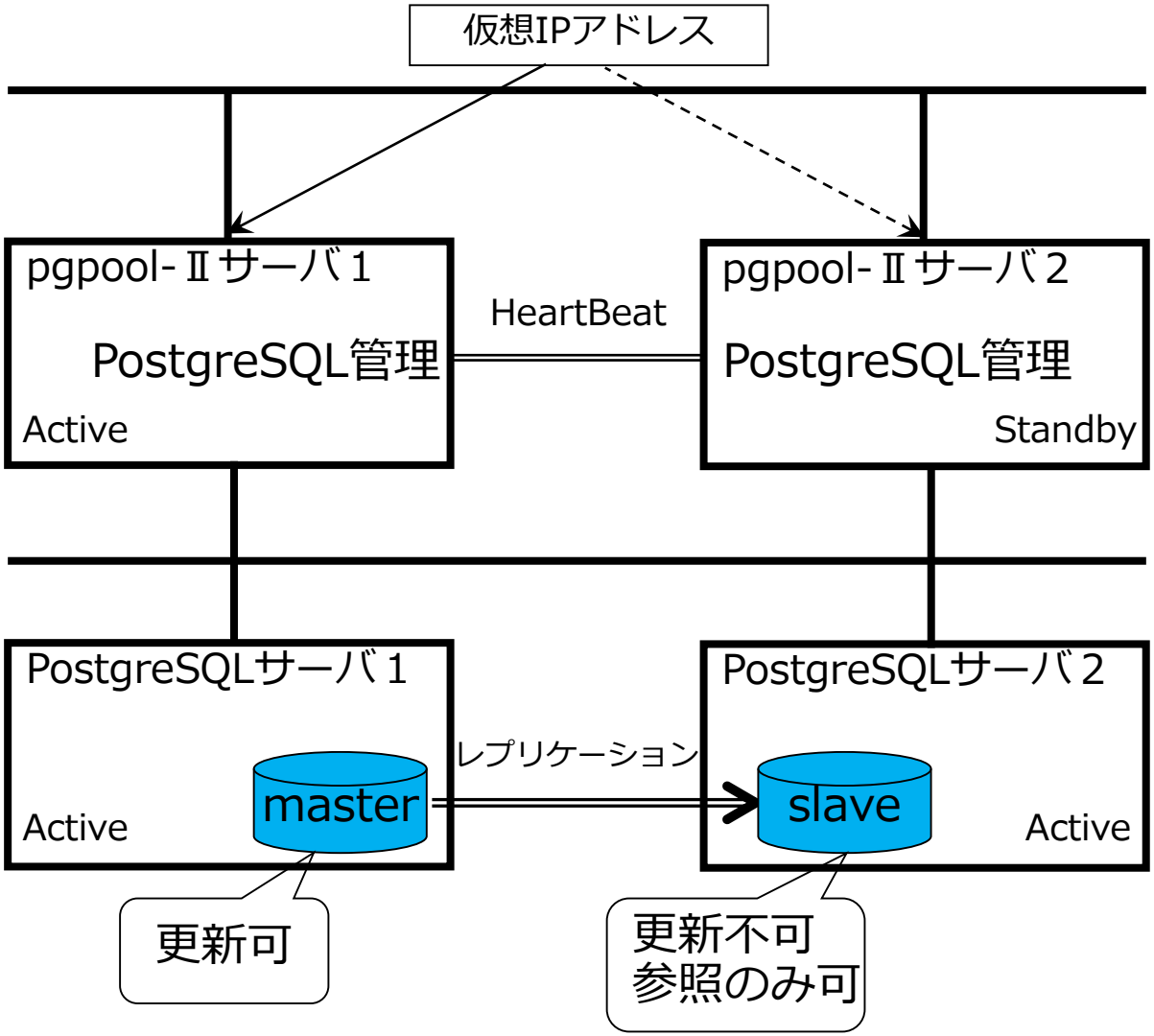
**現行システムから信頼性を落とさない方式検討が課題**

© Hitachi, Ltd. 2017. All rights reserved.

# 3-6. 信頼性確保について

HITACHI
Inspire the Next

新システム　・・・pgpool-Ⅱ(OSS)とストリーミングレプリケーションにより信頼性を確保




- 複数のPostgreSQLを管理
- master/slave環境で更新系SQLはmasterにのみ振分
- masterダウン時にはslave側をmasterへ昇格させる

- レプリケーション機能により、同一データを２台で保持
- masterは更新可、slaveは参照のみ
- masterダウン時のデータ欠損ないように、同期レプリケーションを選択
- 同期レプリケーションは性能検証が必要

© Hitachi, Ltd. 2017. All rights reserved.

HITACHI
Inspire the Next

# インシデントへの対応事例

■slave側への参照クエリーで検索できないことがある

①現象

masterに対して更新されたレコードを、その直後にslave側で参照した場合に、対象のレコードが検索できない場合がある。

②調査方法、調査結果

master、slaveのPostgreSQLのログより、対象のレコードが正しく検索できる場合とできない場合を整理し、正しく検索できない場合はslave側で参照クエリーが処理されていた。

同期モード時の負荷分散の仕様を再確認した結果、更新のクエリーはslaveへ送信されたWALが同期した時点でコミットが返る仕様であり（PostgreSQL 9.6 ではエンハンスされている）、slave側のテーブルに該当のレコードが書込まれた後ではなかった。

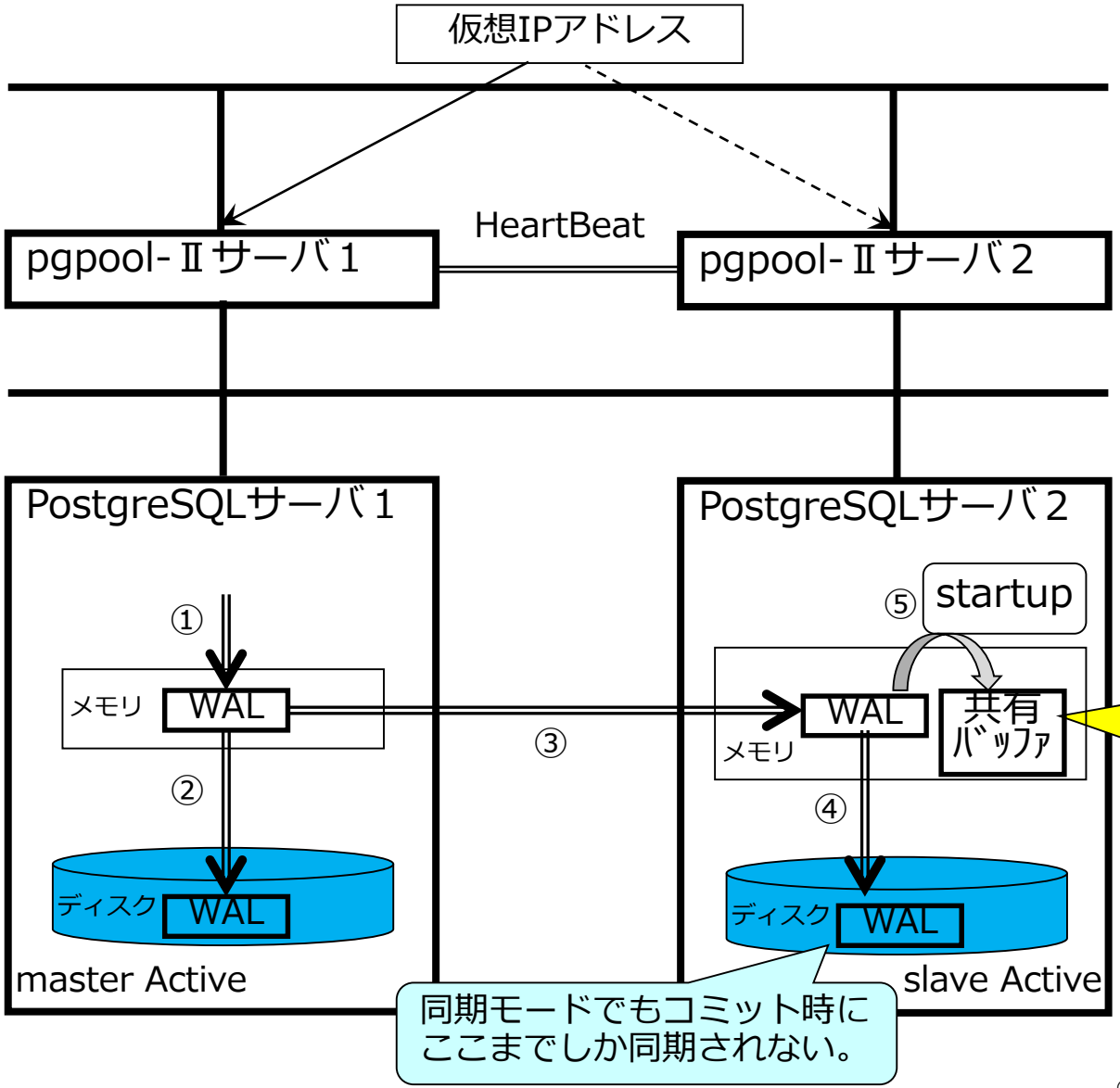
③対策

クライアントが発行するSELECT文の前に/*NO LOAD BALANCE*/コメントを付与することでクエリー毎に負荷分散しないまたは、するを宣言することができるがクライアントプログラムへの改修範囲が大きくなる。このためクライアントプログラム毎（サーバのJDBC/ODBCのドライバー単位）に識別できる特定の名称(アプリケーション名)を付与し、この名称毎に参照クエリーを負荷分散するまたは、しないかを区別する対策とした。

これにより、更新した後にすぐに参照が発行されるようなケースにおいても正しい結果が得られるようになった。

© Hitachi, Ltd. 2017. All rights reserved.

# 3-7. 課題・懸案対応




⑤startupプロセスは非同期に処理するため、テーブル参照時には更新されていなかった。

※9.6ではエンハンスされており、問題は発生しない。

© Hitachi, Ltd. 2017. All rights reserved.

# 3-7. 課題・懸案対応

HITACHI
Inspire the Next

| | 現象 | 原因 | 対策 |
|---|---|---|---|
| 1 | 検索結果が不特定に不正になる | JDBCにてPreparedStatementキャッシュが行われており、SQLが実行されなかったため。 | JDBCにprepareThreshold=0を設定した。 |
| 2 | 検索結果が不特定に不正になる | ODBCの設定で、カーソルを利用しないようになっていたため。 | ODBCの設定で、Declare-FetchをONにする。 |
| 3 | SQLがエラーになる場合がある | autovacuum中にslave側に対して検索したため。 | master側を検索するようにした。 |
| 4 | 検索が遅い時がある。 | テーブルの状態によって、シーケンススキャンされる場合があるため。 | set enable_seqscan = off;を追加した。 |
| 5 | SQLが接続エラーになる場合がある。 | コネクションプールを利用時に、FWで無通信タイムアウトにより、切断してしまっていたため。 | 切断のタイムアウト値を見直した。 |
| : | : | : | : |

インターネットで調査
弊社サポート部門と協力して調査・対策

© Hitachi, Ltd. 2017. All rights reserved.

# 3-8. まとめ

HITACHI
Inspire the Next

高信頼性のシステムを構築できた。

実績が豊富のため、インターネットで検索可能。

弊社サポート部門の協力により、
5月本番稼働からトラブルなし。

© Hitachi, Ltd. 2017. All rights reserved.

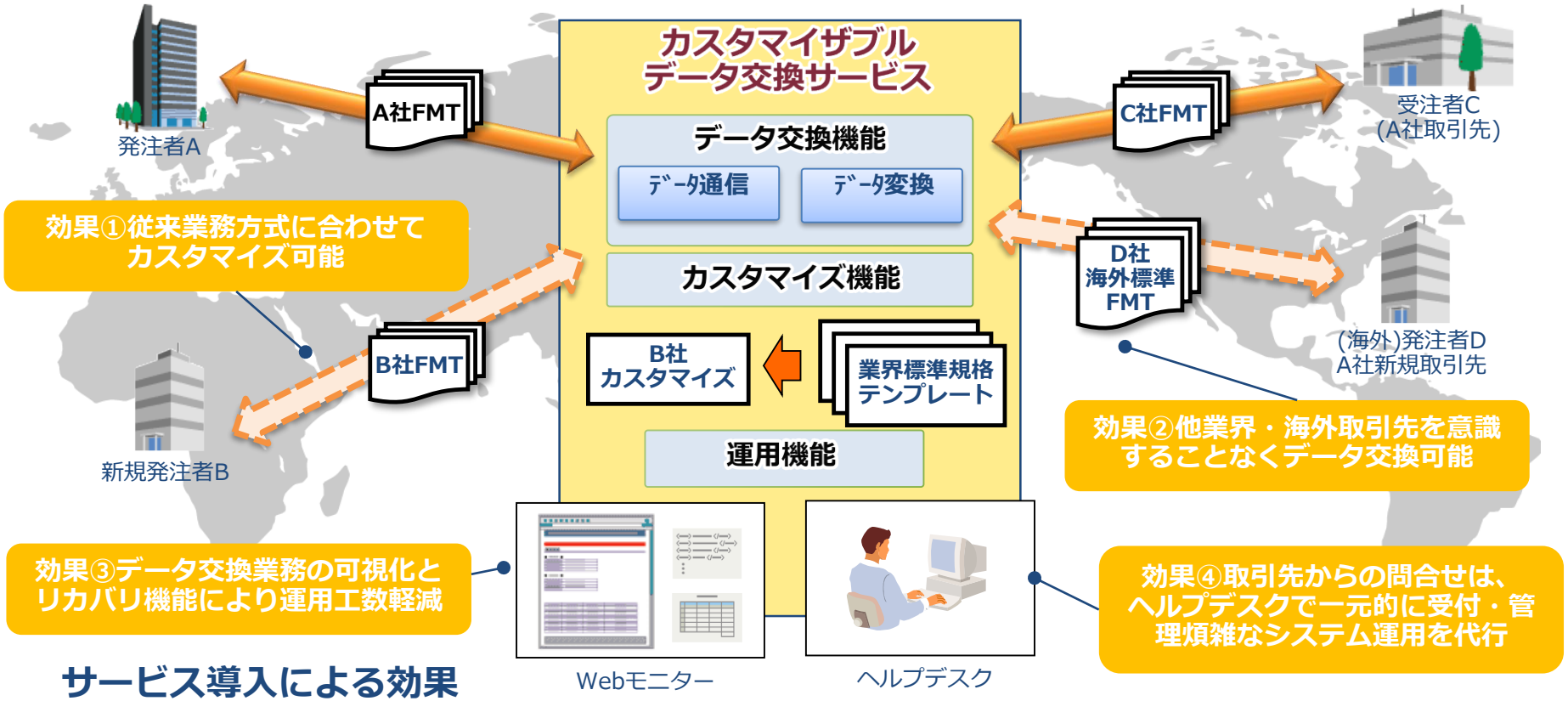
# 4. 弊社サービス紹介

© Hitachi, Ltd. 2017. All rights reserved.

# 4-1. カスタマイザブルデータ交換サービス

HITACHI
Inspire the Next

ビジネスパートナーとの多様なデータ交換を国・業界に
かかわらず容易に実現し、ビジネスの可能性を拡張




## サービス導入による効果

1．お客さまが管理する資産を増やさずにデータ交換が実現可能

2．お客さまの業務内容に合わせてスピーディーな導入を実現

3．Webモニターやヘルプデスクにより運用・保守工数を軽減

© Hitachi, Ltd. 2017. All rights reserved.

# 4-2. 日立のEDB Postgresサポートサービス

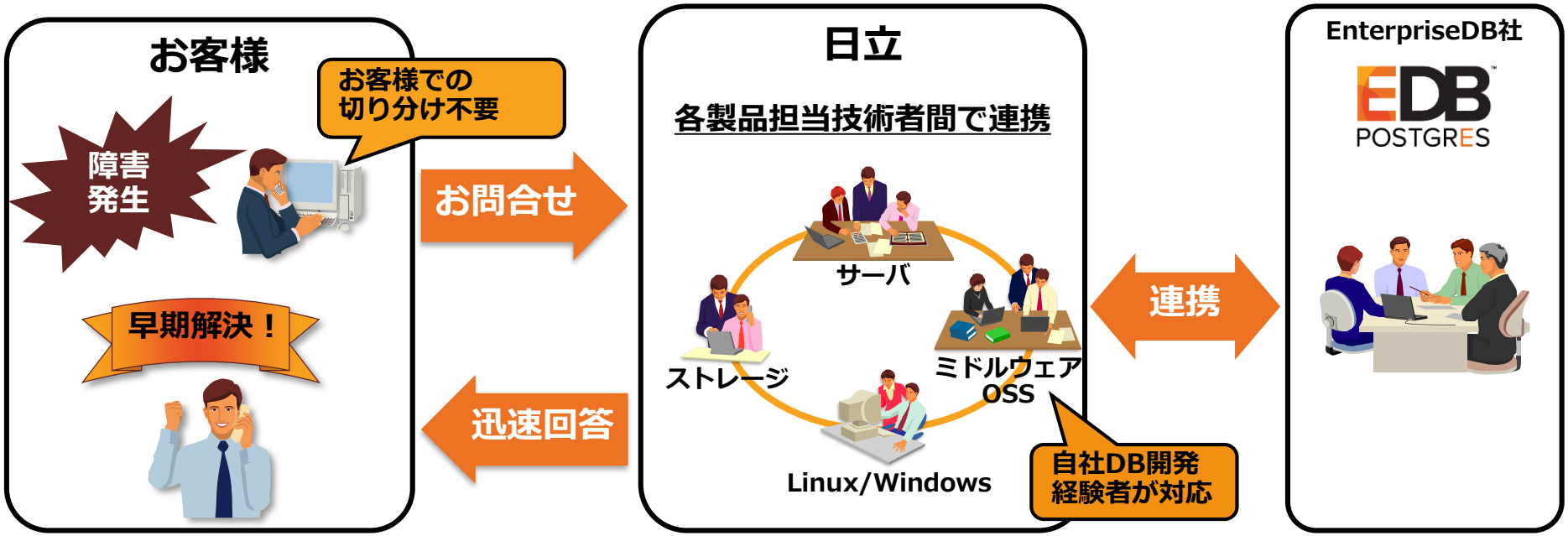
HITACHI
Inspire the Next

## ■ 日立が提供するEDB Postgresサポートサービスの特長

- 日立サポート360によるワンストップサポート
- 日立およびEnterprise DB社の技術者の連携による問合せ対応

### 日立サポート360ご利用のメリット

**日立の総合力によりシステム一体のサポートを提供します。障害発生時もお客さまは問題を切り分けることなくお問い合わせいただけます。**
**また、原因究明が難しい問題も各製品技術者、導入元ベンダとの連携により、多角的に問題解決にあたります。**




http://www.hitachi.co.jp/Prod/comp/soft1/support360/

© Hitachi, Ltd. 2017. All rights reserved.

# 4-2. 日立のEDB Postgresプロフェッショナルサービス

HITACHI
Inspire the Next

**安定稼働の実現を支援する、システムの設計から運用フェーズまでをサポートするサービスです。**

| 技術支援 | 概要 |
|---|---|
| **コンサルティング** | お客さまの困りごとに対して、コンサルティング支援を行います。下記の支援内容以外にも、例えば、テストや本番運用時に性能が出ない際の問題解決のためのノウハウ提供や支援を行います。 |
| **設計・構築支援** | 要件に応じた構築を行うために、構成検討やインストール設計、パラメタ設計、運用設計などのノウハウ支援、問題解決、作業支援を行います。また、立ち会いによる構築支援を行います。 |
| **クラスタ構築支援** | 高可用性、高信頼性を確保しなければならない基幹システムには、クラスタソフトウェアと組み合わせた構成で設計・構築を支援します。 |
| **マイグレーション支援** | 他の商用データベースからEDB Postgresに移行する際の先行調査、影響範囲を見極め、マイグレーションを支援します。移行の際の留意点やノウハウを提供します。 |
| **製品トレーニング** | 製品を初めてご利用になる開発技術者に向けたサービスメニューです。製品の概要説明や基本機能の使い方を習得できます。 |

© Hitachi, Ltd. 2017. All rights reserved.

# 4-3. 日立のサポート対象OSS

※下記以外のOSSにつきましては、個別にご相談ください。

HITACHI
Inspire the Next

| | | | |
|---|---|---|---|
| クラウド | • OpenStack | | |
| ビッグデータ | • Cloudera Enterprise<br>• Hadoop(コミュニティ版)<br>• Spark(コミュニティ版)<br>• Kafka(コミュニティ版) | • Hive(コミュニティ版)<br>• HBase(コミュニティ版)<br>• ZooKeeper(コミュニティ版) | • Hadoop(Cloudera版)<br>• Talend<br>• Asakusa |
| ビジネスルール | • JBoss BPM Suite | • JBoss BRMS | |
| Web・アプリケーションサーバ | • JBoss Enterprise Application Platform<br>• JBoss Web Server<br>• JBoss Fuse<br>• Apache HTTP Server | • Nginx<br>• Tomcat<br>• PHP<br>• mod_jk<br>• Nginx | • Apache Commons Lang<br>• Apache Commons Collection<br>• Apache Commons IO<br>• Apache Commons Net |
| データベース | • EDB Postgres<br>• 商用版MySQL<br>• PostgreSQL | • MySQL<br>• MongoDB<br>• PostGIS | • JBoss Data Grid<br>• JBoss Data Virtualization |
| クラスタソフト<br>※PostgreSQLとの組み合わせのみ | • pgpool-II | • pacemaker | • corosync |
| ネットワーク | • BIND | • Samba | |
| メール | • Postfix | • dovecot | |
| ファイル同期 | • rsync | | |
| 運用管理 | • Ansible | • MIRACLE ZBX | • JBoss Operations Network |
| 認証・セキュリティ | • OpenLDAP<br>• Spring Security OAuth | • OpenSSL | • GnuPG |
| 開発管理 | • Redmine | | |
| 開発言語・フレームワーク | • Ruby on Rails<br>• Apache Struts | • Spring Framework<br>• JBoss Developer Studio | • MyBatis |
| ライブラリ・API | • jQuery<br>• jQuery UI<br>• Log4j 2 | • SLF4J<br>• Logback | • JasperReports Library<br>• ApachePOI |
| OS・プラットフォーム | • Red Hat Enterprise Linux | • MIRACLE LINUX | |

© Hitachi, Ltd. 2017. All rights reserved.

HITACHI
Inspire the Next

# END

## PostgreSQLを使って変化に強いシステムへ！

2017/12/04

株式会社 日立製作所
OSSソリューションセンタ

© Hitachi, Ltd. 2017. All rights reserved.

# 付録．他社商品名，商標等の引用に関する表示

HITACHI
Inspire the Next

・HITACHIは、株式会社 日立製作所の商標または登録商標です。
・PostgreSQLは、PostgreSQL Community Association of Canadaのカナダにおける登録商標およびその他の国における商標です。
・EnterpriseDBは、EnterpriseDB Corporationの登録商標です。 EDBおよびEDB Postgresは、EnterpriseDB Corporationの商標です。
・MySQLは，Oracle Corporation 及びその子会社，関連会社の米国及びその他の国における登録商標です。
・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。
・Apache は、Apache Software Foundation の米国およびその他の国における登録商標または商標です。

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, Ltd. 2017. All rights reserved.

HITACHI

Inspire the Next