OSS Consortium

初歩から始める Asakusa Framework



第3回 2016 Asakusa Framework Day

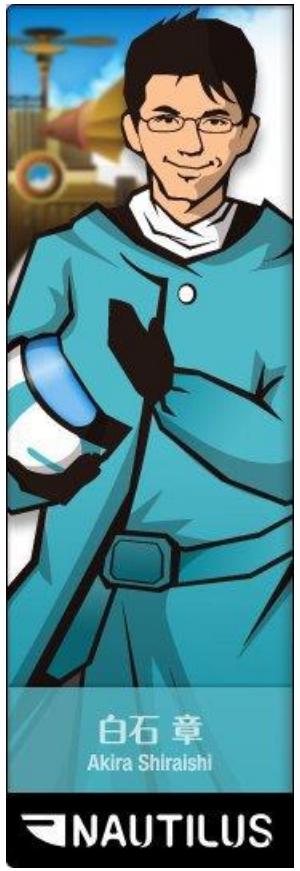
NAUTILUS

Akira shiraishi

Nautilus Technologies, Inc.

今日のテーマ

- Asakusaの概要と特徴
  - Asakusa Framework とは
  - Asakusaの3つの特徴
- Asakusaの新機能
  - Asakusa on M³BP(2016/4)
  - Version 0.9.0(2016/11)で予定されている追加機能



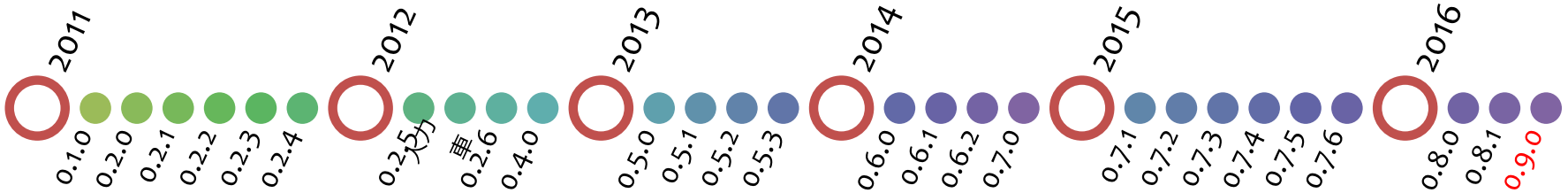
NAUTILUS

Asakusa Frameworkの基本

Asakusa Framework

Asakusa Frameworkの誕生

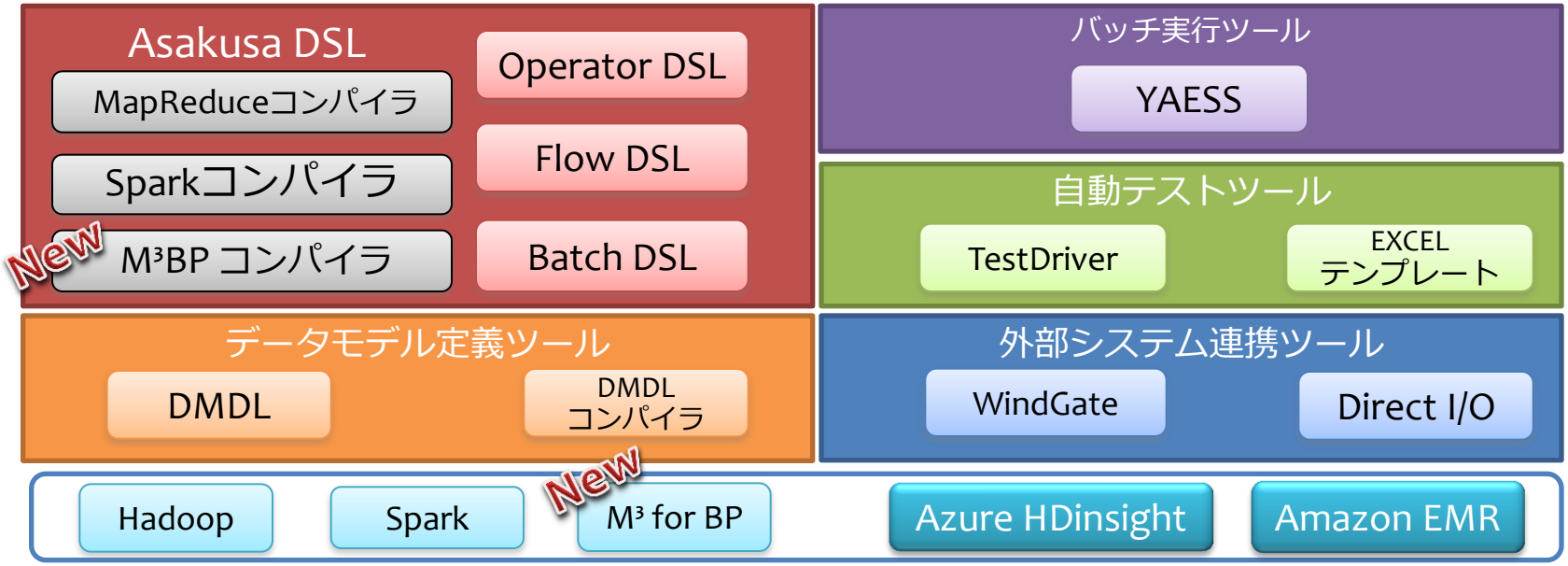
- 分散バッチ処理用フレームワーク
  - Asakusa Frameworkは分散処理技術を活かしてバッチ　アプリケーションを開発するフレームワーク
  - OSSとして2011/3に初版 (Version 0.1.0) リリース
    - おかげさまで、6年目になりました
  - 2016/11/末: 最新版 (Version 0.9.0)リリース予定
    - これまでに 計24回のリリース



Asakusa Frameworkとは

- **基幹業務向け分散バッチ開発フレームワーク**
  - Asakusa Frameworkは、基幹業務システムのバッチの高速処理を目的とした、Hadoopに対応した業界初のソフトウェア
  - 基幹バッチ開発に必要な、開発環境・実行環境・運用環境を用意
- **Hadoopの開発を容易にし、更に高速化**
  - MapReduce特有のコードを、Asakusaのコンパイラが自動生成
  - MapReduceのコードを最適化し、MapReduceの段数を削減させて高速化

Asakusa Frameworkコンポーネント



Asakusa Frameworkとは

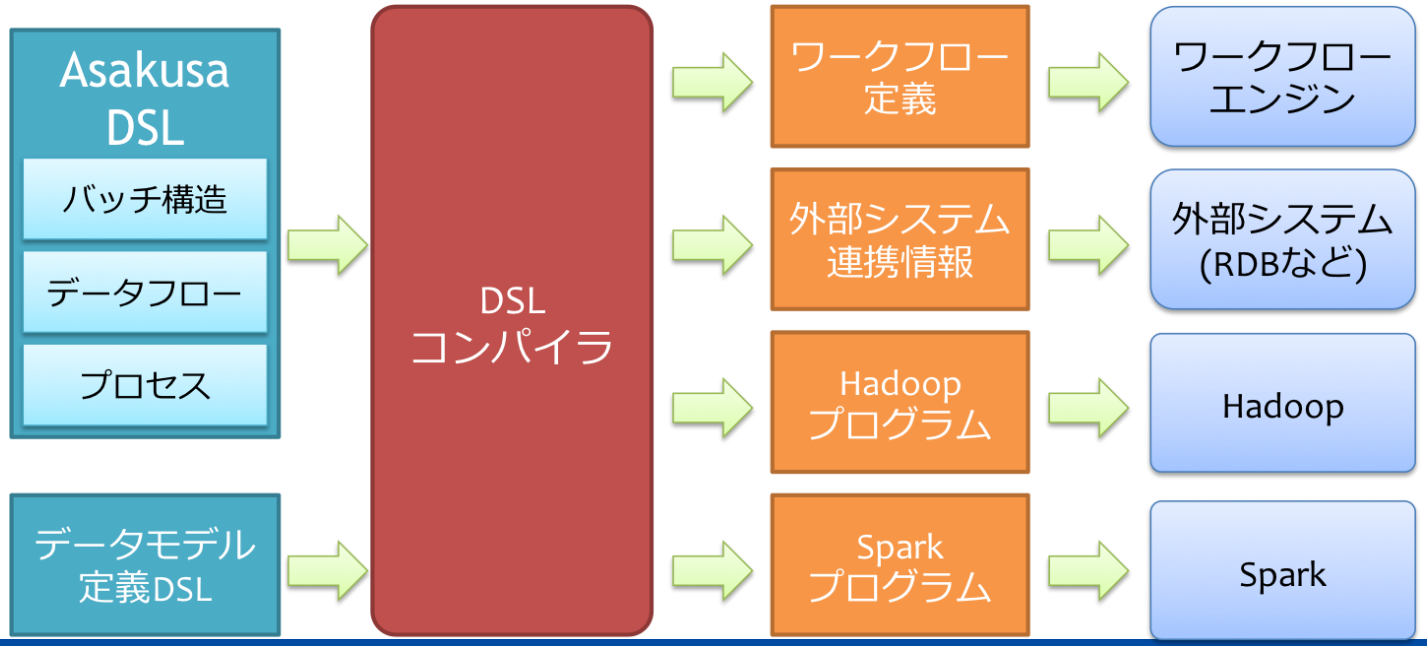


3つの特徴

開発容易性	高速処理	ポータビリティ
• HadoopやSparkの開発方法を覚えなくても開発可能 • データフローの設計をそのまま実装可能 • 開発時のテスト、チェック機能充実	• Asakusaコンパイラでアプリケーションの最適化 • Hadoop/Spark/M³BPのエンジンによる高速処理の実現 • ユースケースに応じ最適な実行基盤の選択が可能	• 一つのソースから複数エンジン向けの実行コードを生成する • Hadoop/Spark等のバージョンアップ時のアプリケーションの互換性を重視 • オンプレ・クラウドでも動作

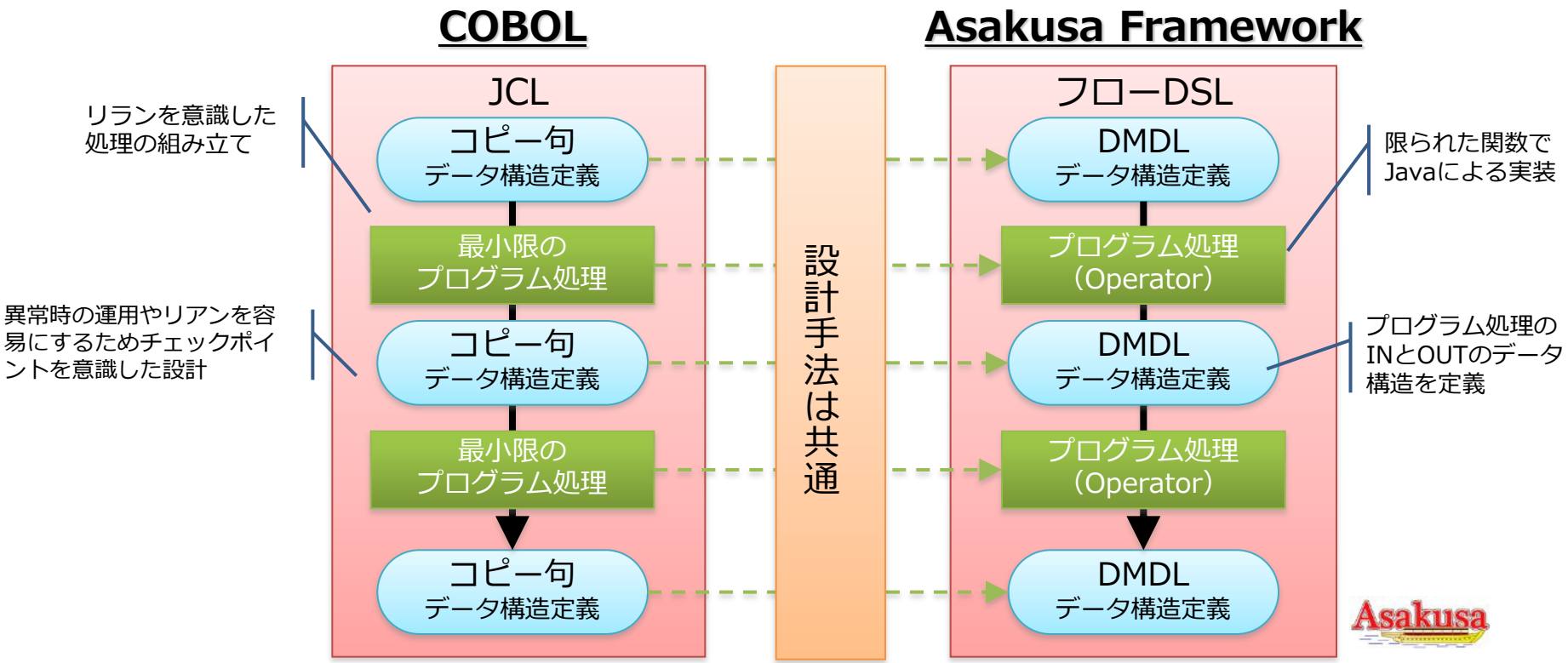
Asakusa Frameworkの開発容易性

- Asakusa DSLを中心としたフレームワーク
  - データフロー形式のバッチ処理を記述することに特化した言語
  - シンプルなJavaプログラムの組み合わせでアプリケーションを開発
- 分散処理基盤にHadoop/MapReduceを使用(Spark,M3BPにも対応)
  - MapReduce特有のコードを、Asakusaのコンパイラが自動生成
  - Asakusaのコンパイラーにて、MapReduceのコードを最適化し、MapReduceの段数を削減させて、更に高速化



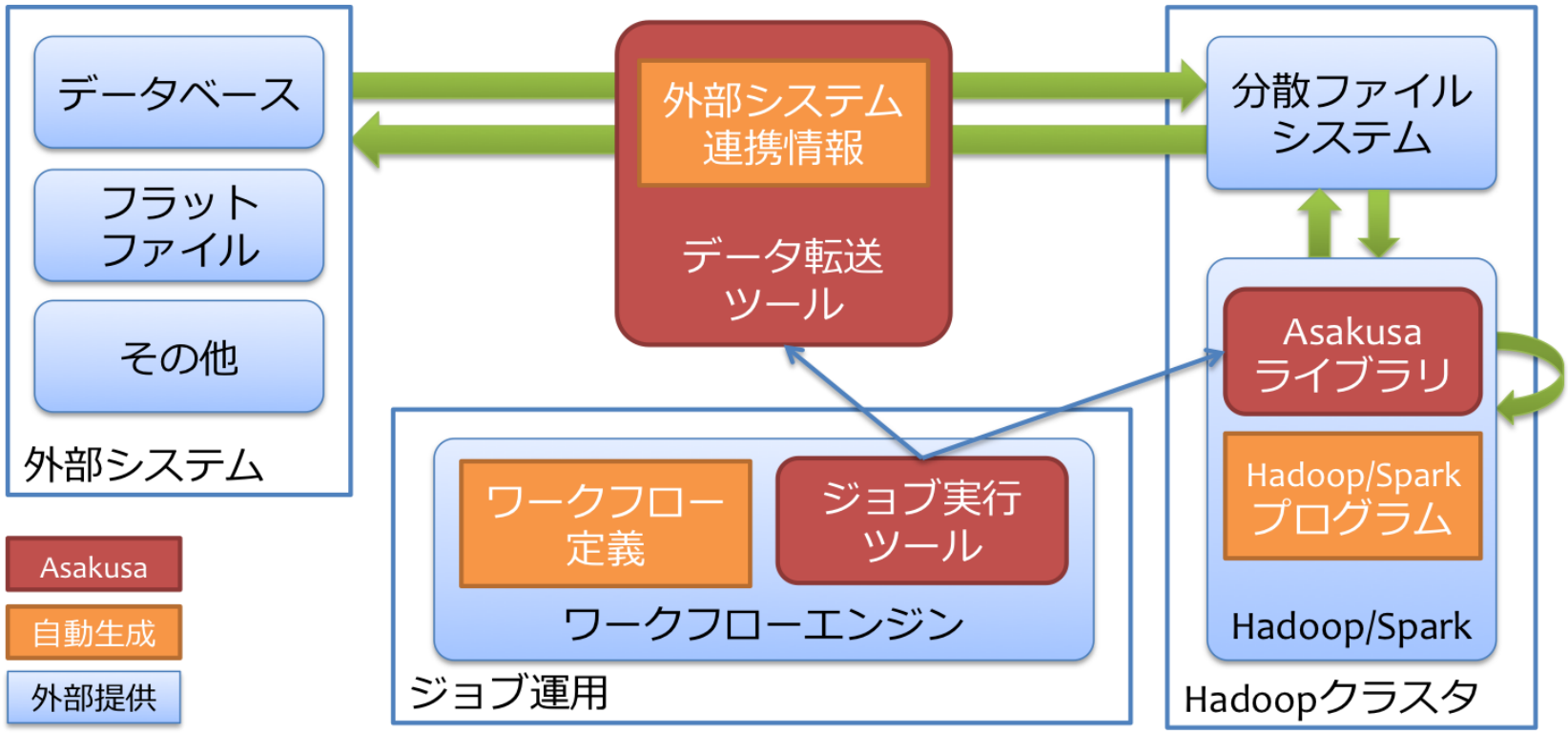
Asakusa Frameworkの開発容易性

- Asakusa Frameworkは、**構造化設計**に基づいたDSLを持っており、既存バッチの設計手法、業務分析成果物などが活かせる
- アプリケーション開発者は並列／分散処理やMapReduceのプログラミング、Sparkのプログラミング方法について意識する必要がない

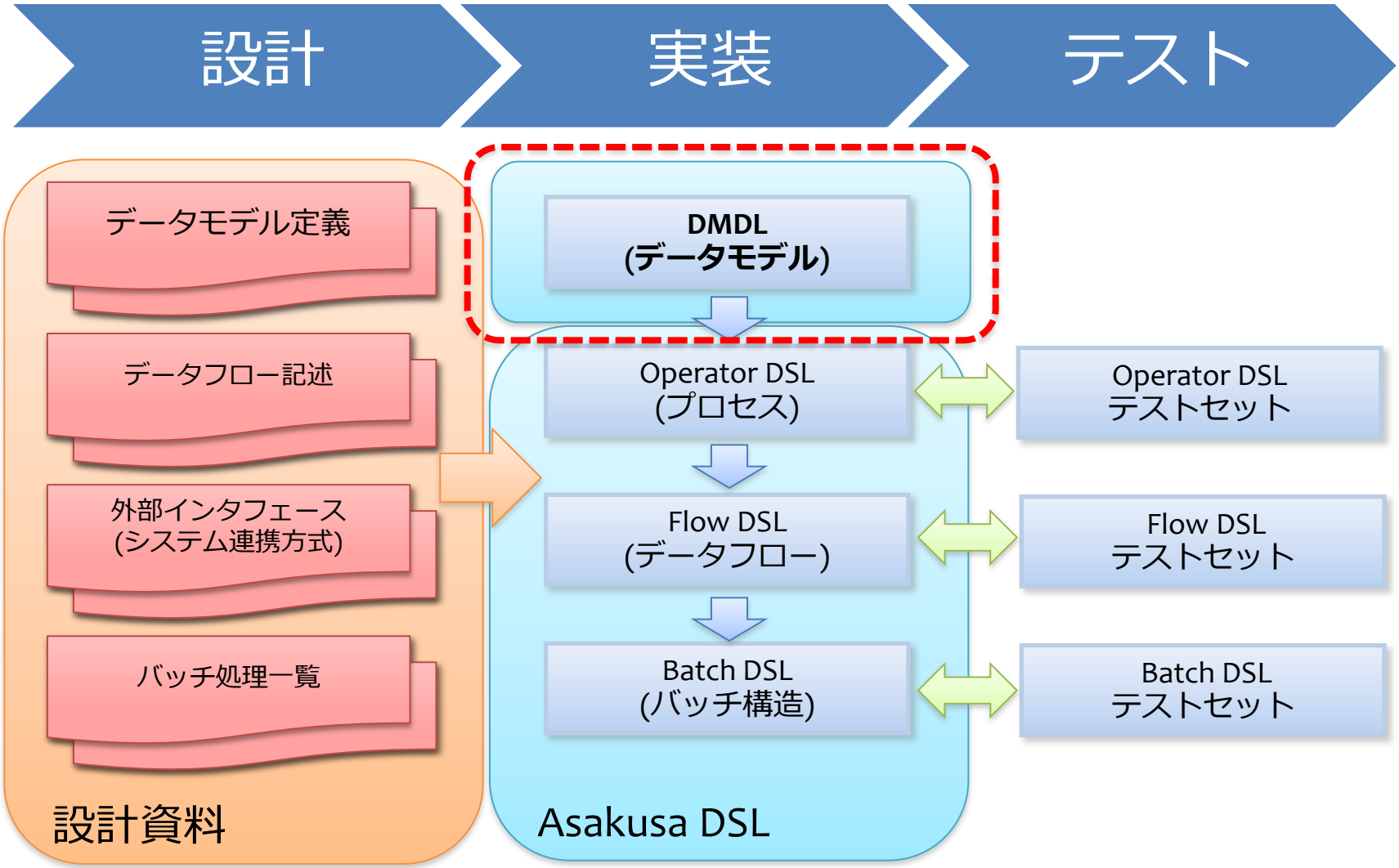


Asakusa Framework(ランタイム)

- 基幹システム開発向けに必要な周辺機能を提供
  - シンプルなデータモデル定義と結合や集計処理の自動生成
  - 様々な外部システムとのデータ連携モジュール
  - Asakusa DSLと連携したテストツール
  - システム構成に応じてカスタマイズ可能なバッチ実行ツール



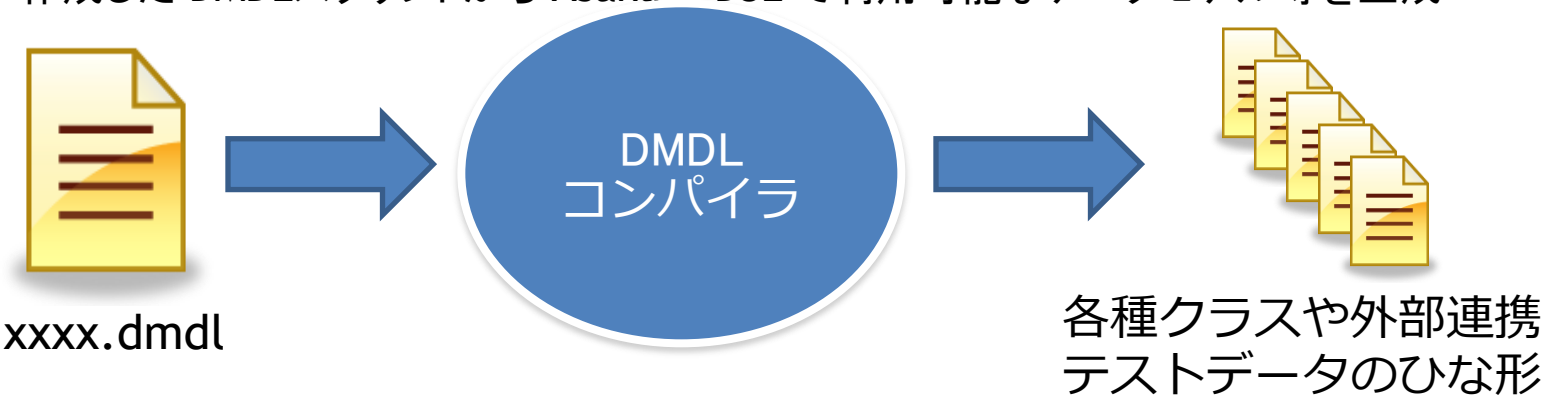
Asakusa Frameworkの開発の流れ



データモデルの定義:Asakusa DMDL

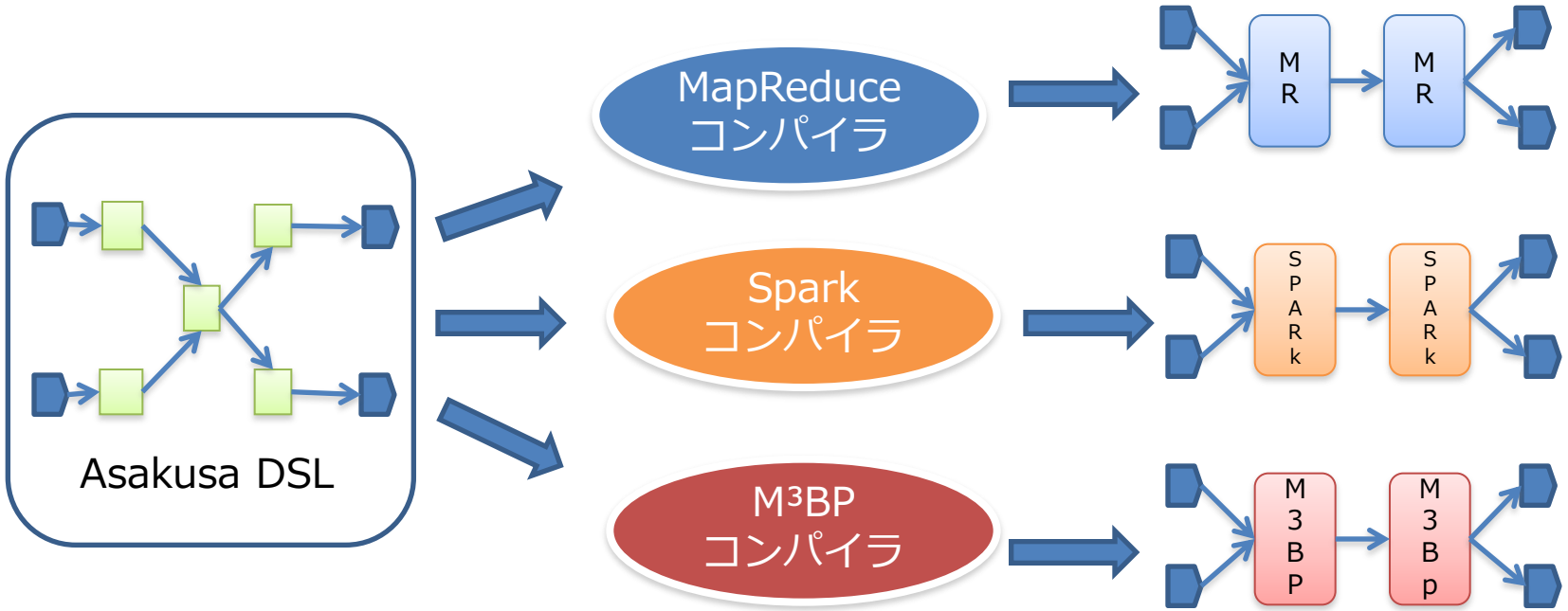
```
1  //DMDL 例
.
2
.  item = {
3      code : LONG;
       price : DECIMAL;
       name : TEXT
   };
```

- DMDL (Data-Model-Definition-Language)
  - Asakusa Framework で処理するデータ構造を定義
- DMDL の特徴
  - DMDL コンパイラにより Hadoop 特有の実装を自動生成
  - テストデータテンプレートの自動生成 (Excel ファイルを生成)
  - WindGate 、Direct I/O との連携情報を自動生成
- DMDLのコンパイル
  - 作成した DMDLスクリプトから Asakusa DSL で利用可能なデータモデル等を生成



Asakusa DSL

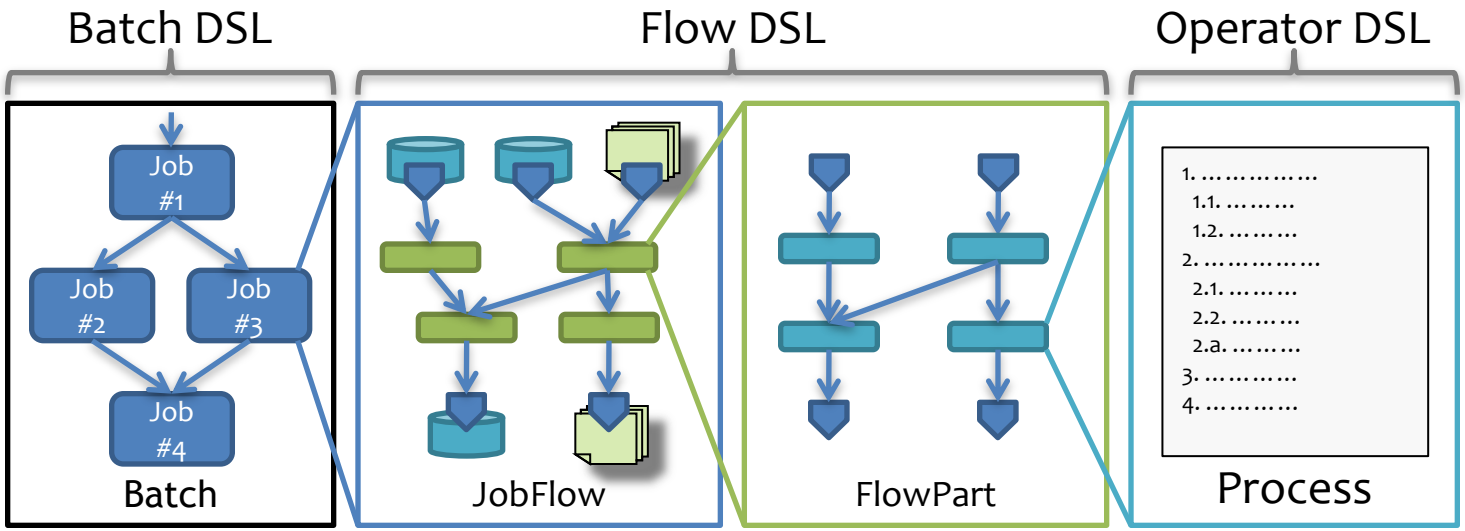
- **Asakusa DSL**を中心としたフレームワーク
  - 設計は、バッチ処理をデータフロー形式で記述します
  - 入力データ、処理内容、出力データを記述したデータフローの設計から、そのまま実装することが可能
  - ビジネスロジック実装にHadoopやSparkのAPI、実装方法を意識しなくて良い



Asakusa Framework のコンポーネント

Asakusa DSL

- 三層DSLによる**シンプルなJavaプログラム**の組合せ
  - 設計後の実装フェーズで演算子 (Operator DSL) から作成
  - 演算子を組み合わせジョブフロー (Flow DSL) を作成
  - ジョブフローの実行順序をバッチ処理 (Batch DSL) に記述します



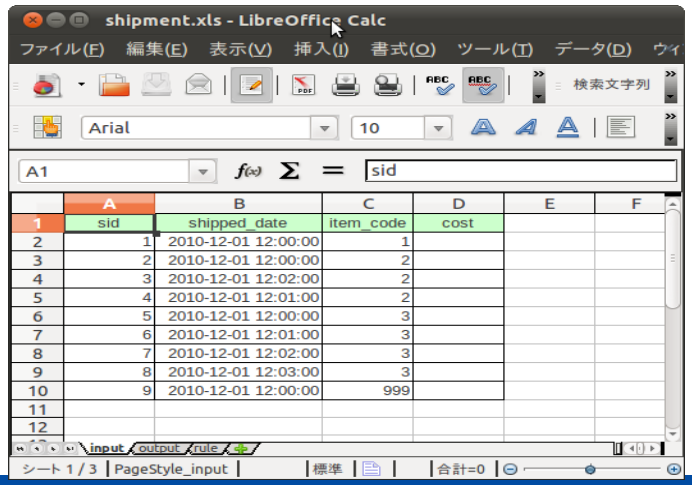
アプリケーションのテスト：TestDriver

- Asakusa Framework では、アプリケーションの開発テストのために、TestDriver や テストデータの雛形を提供
  - Hadoopでの大規模な開発の為の品質確保が可能
  - **OperaterDSL(演算子)でのテスト**
    - 演算子に対して通常の Java クラスと同様にJunitでテスト可能
  - **データフロー、バッチのテスト**
    - Hadoop や外部入出力と自動的に連携したテストが可能
    - 一連の処理を自動的に行う**『テストドライバ』**を提供
    - テストデータのひな形を Excel ファイル・JSONで定義

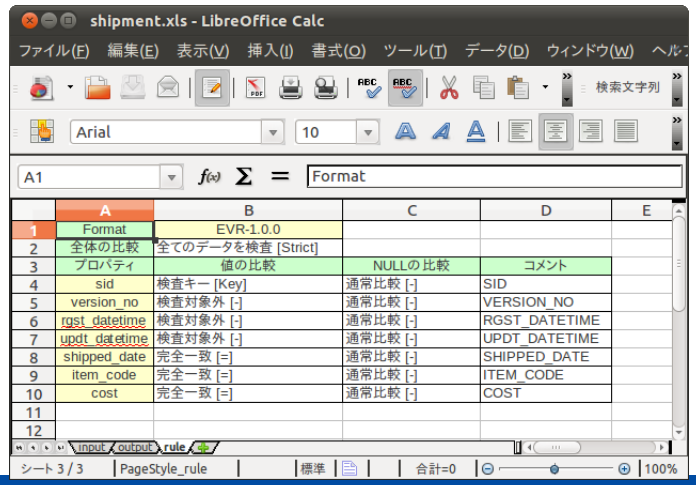
テストドライバの処理

1. 入力データを初期化する
2. 入力データを流し込む
3. 対象のプログラムをテスト実行する
4. 出力結果を取り込む
5. 出力結果と期待データを検証する

入力データ・期待データシート



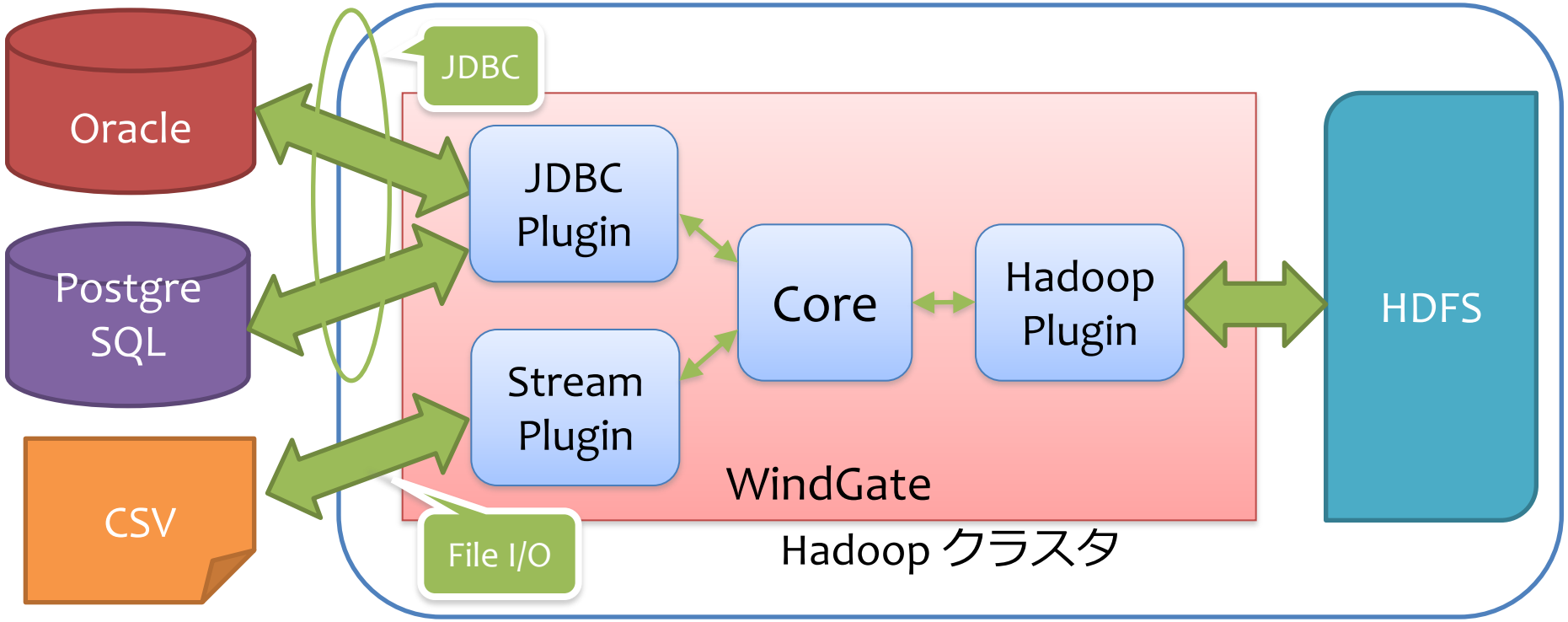
比較条件シート



Asakusa Framework のコンポーネント

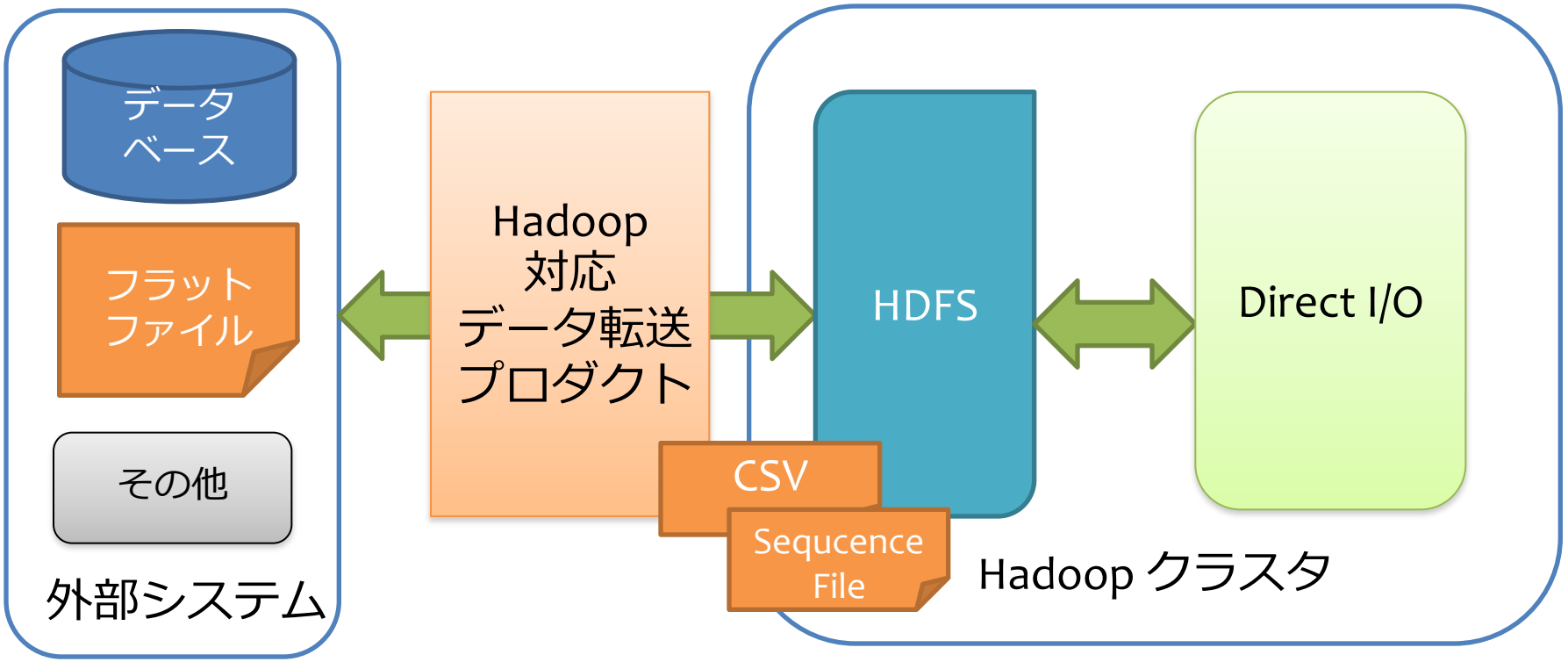
外部システム連携ツール：WindGate

- 様々な RDBMS や ローカルファイルとのデータ連携を行うツール
  - RDBMS には標準 SQL (JDBC) でアクセス
  - ローカルに配置した CSV ファイルの入出力も可能
  - プラグインを実装すればその他のリソースにも対応可能



外部システム連携ツール：Direct I/O

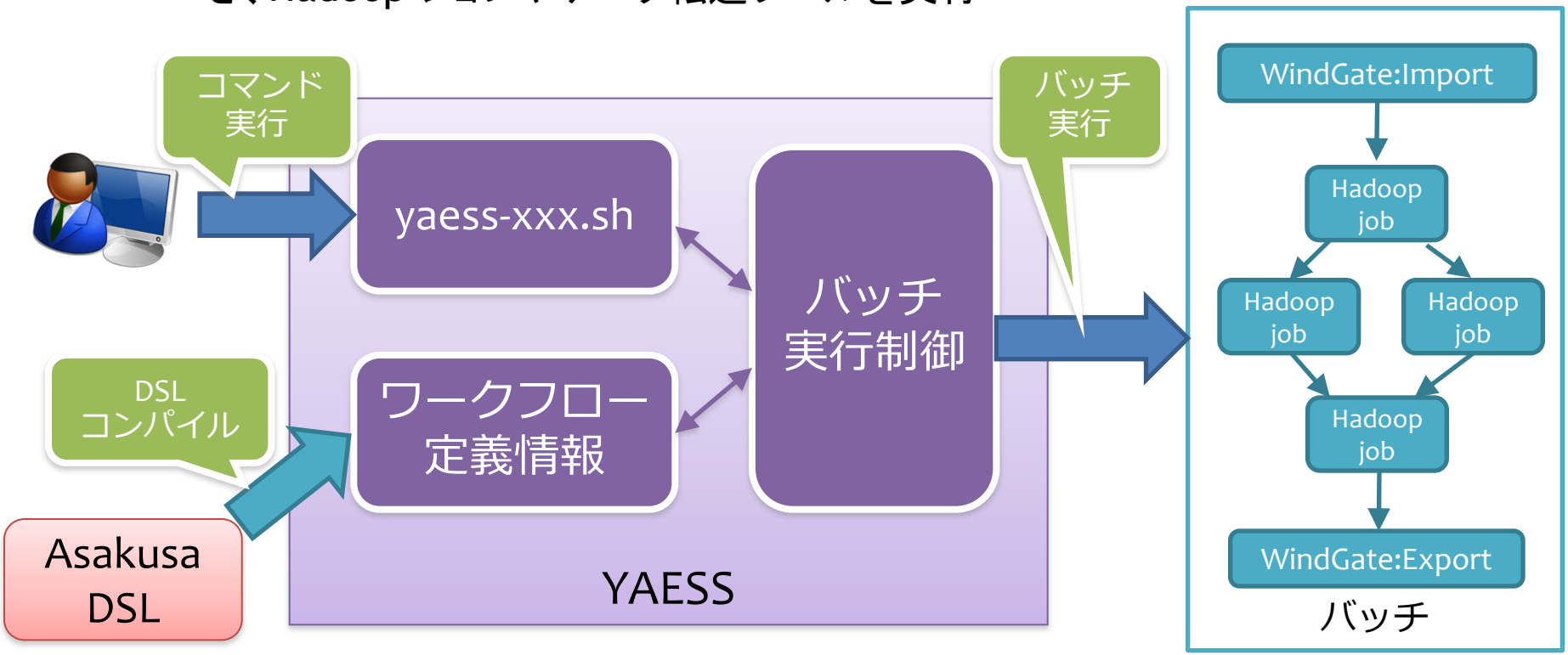
- HDFS 上のファイルを直接読み書きできる、高速な入出力機構
  - バッチアプリ開発と運用時に利用するデータソースや入出力の指定を分離して実行可能
  - HDFS 上の CSV ファイルやシーケンスファイルを扱える
  - Asakusa以外のプロダクト、Hive (Parquet,ORC file形式等)に連携可能



Asakusa Framework のコンポーネント

バッチ実行ツール：YAESS

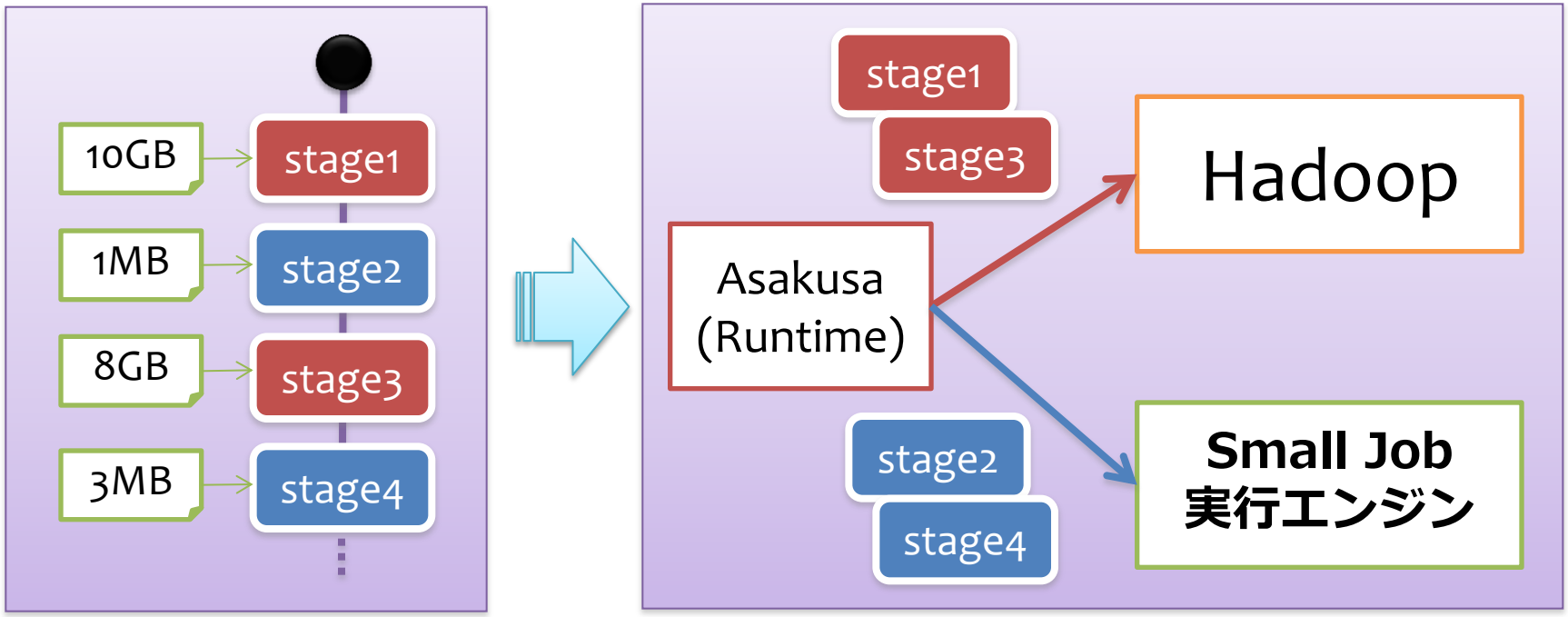
- バッチ処理を起動するツール (Shell Script)
  - バッチ実行用のコマンドラインインターフェースを提供
  - Asakusa DSL のコンパイル時に生成されるワークフロー定義情報に　基づき、Hadoop ジョブやデータ転送ツールを実行



Asakusa Framework のコンポーネント

Small Job実行エンジン

- スモールジョブを高速処理
  - 10MBの入力でHadoopの5~10倍高速にジョブを実行
- データサイズに応じて実行エンジンを自動選択
  - HadoopとSmall Job実行エンジンを選択可能



Asakusa Framework のコンポーネント

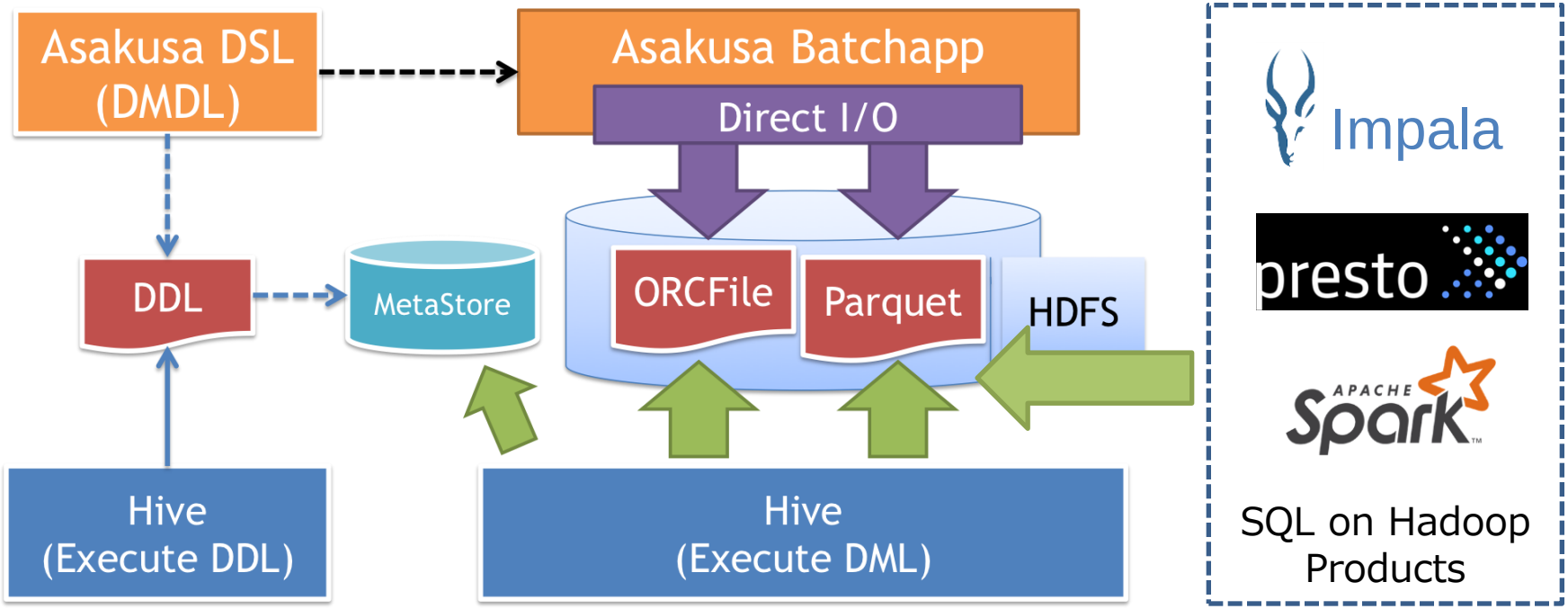
Asakusa on Spark

- **Asakusa on Spark**
  - Asakusa DSLをはじめとするAsakusa Framework開発基盤を利用し、**既存のAsakusa DSLを一切改修せず**にSpark処理基盤で利用可能な　機能セットを提供
  - Asakusa Framework0.8.0より、Developer previewから正式機能に
- **Sparkの優位性**
  - 業務系バッチにおける複雑な処理、数十GBレンジのデータボリュームにおいて、既存**MapReduceよりも高速処理**が可能
  - まったく同一のAsakusa DSLで、MapReduce、Sparkの両環境で動作可能のため、速度の比較検証も容易に可能で、バッチアプリケーションごとに**適切な分散処理基盤を選択可能**



SQL on Hadoop連携機能

- Hiveの対応するカラムナフォーマットの入出力をサポート
  - ORC File
  - Parquet
- Hiveのメタデータ管理方法と互換性がある、SQLベースのクエリーエンジンを持つ様々なプロダクトと連携することが可能に。



Asakusa Frameworkの周辺ツール

- 開発環境は、一括インストールツール『Jinrikisha』を用意
  - http://asakusafw.s3.amazonaws.com/documents/jinrikisha/ja/html/index.html



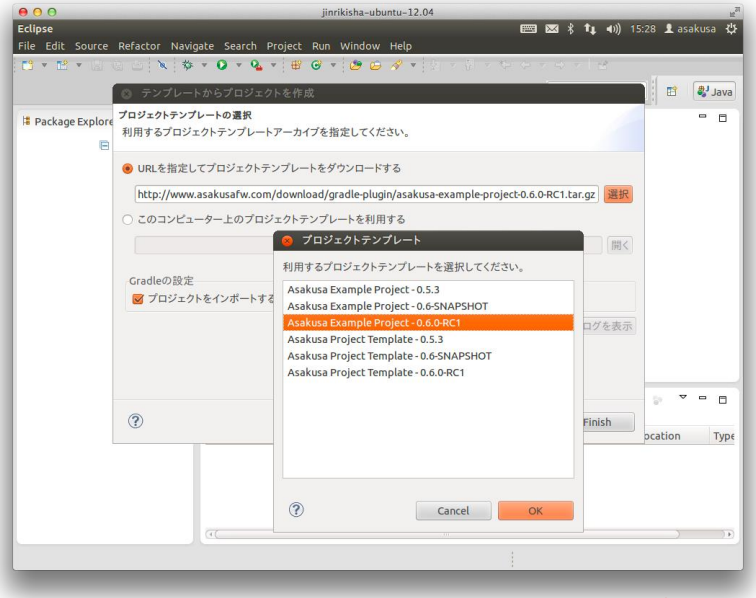
Hadoopや
Gradle/Eclipse
をAsakusaと一
括インストール











既存の他システムとの連携

RDBMS連携、HDFS連携モジュール提供
Asteria/DataSpider/Hulft等のEAI/ETL連携

ジョブ管理ツールとの結合

JP1やSystemWalker、千手、A-Auto等の
ジョブ管理製品との連携

Windows環境で開発・テスト・ビルド

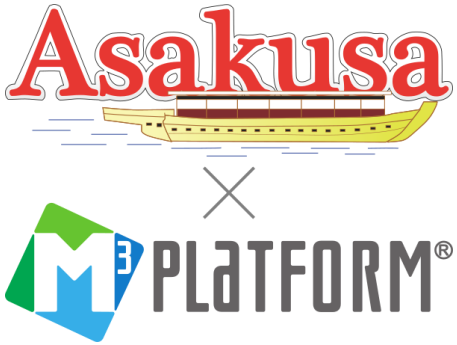
Eclipseプラグイン『Shafu』を用意
バッチテストランナーにてテスト時間を
短縮

Hadoop以外のエンジンも選択可能

Asakusaスモールジョブ実行エンジンや
Sparkも、処理時間に応じて選択可能

Asakusa Framework新機能

Asakusa on M³BP
（エムキューブドビーピー）



Asakusa on M³BP登場の背景

- Hadoop/Sparkなどの分散処理ソフトの進展
  - 数百GB、数TBの大量データを分析する例が少ない
  - 実際は、数GB、数十GBのデータだが、処理に時間がかかる方が多数
  - Hadoop/Sparkだと大掛りで、分散処理のオーバーヘッドが大きい
- RSA（Rack-Scale-Architecture）によるHPC化が進展していく
  - CPUのマルチコア、メニーコア化はさらに進む
    - CPUの性能アップが限界に近くなり、コア数を増やす方向に向かっている。
      - 現状は1台で22コアまで。近く1台で100CPUコア以上を搭載も可能に
  - Memoryの進化（不揮発性メモリーの登場）
    - 100GB以上、1TB以上の広大なメモリ空間が利用可能になってくる
    - 更に、電源オフ時でも消えない不揮発性メモリが登場
      - メモリーとHDD/SSDの区別がなくなってくる
        - 要は、ノード数を増やさなくても、単体で性能が出る
- 通常のプログラムはCPUコアを1つしか使えない
  - マルチコアCPUでも、並列分散処理にてアプリケーションを実行できないと高速処理ができない
  - データ量が単体マシンのスペック以上に大きい場合には、スケールアウトをさせたい

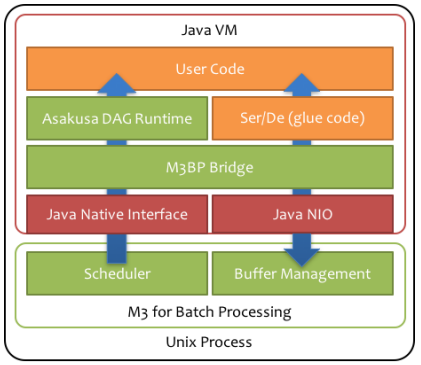
Asakusa Framework新機能

Asakusa on M³BP

- 分散処理エンジン(M³ for Batch Processing)をFixStars社と共同開発
  - マルチコア環境でDAG形式で並列処理をするインメモリエンジンを提供
    - 小規模データでの複雑な処理を、単一ノード上のマルチコア用に最適化
  - C++でネイティブアプリケーションで動作させて高速化
  - M³ for BPの実行環境には、Hadoop/Sparkは不要
- 単一ノード・マルチコア環境用のAsakusa on M³BP
  - Asakusa on M³BPにてCPUコアの制御やメモリ管理はC++でM³ for BPで処理し、アプリケーション部分はJVMに振分け高速化
  - Asakusa DSLには一切変更を加えることなくM³ for BPと、Hadoop/Sparkの処理　基盤を切り替えて使うことが可能

FIXSTARS Speed up your Business

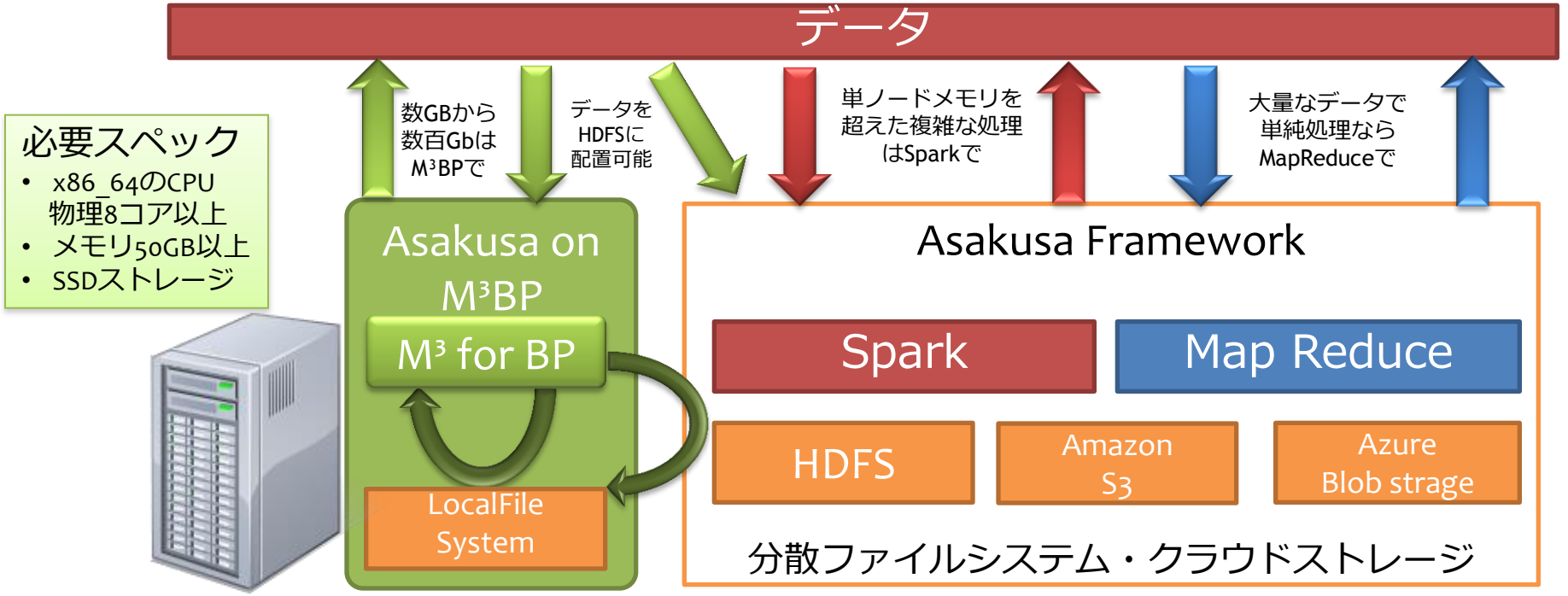
Asakusa × M3 PLATFORM®



単一ノード・マルチコア・大量メモリにて、小規模バッチ処理をSparkよりも高速処理するM³ for Batch ProcessingとAsakusa on M³BP

Asakusa on M³BPの動作環境

- ローカルと分散の両方にデータアクセス可能
  - 単ノードで、Linuxのローカルファイルシステムやマウント可能なストレージにアクセス
  - データだけ分散ファイルシステム上に配置し、M³BP単ノードで実行をすることも可能
    - クラウドも利用可能だが、性能面で物理コンピュータを推奨
    - Hadoop/Spark上で、Asakusa on M³BPを動作させるのは非推奨(リソースの干渉)
  - 数GBから～数百GBのデータについてM3 for BPを利用し、更にデータ量が増えたらアプリケーションを改変せずに、Hadoop/Sparkにエンジンを切り替え複数ノードにスケールアップ可能



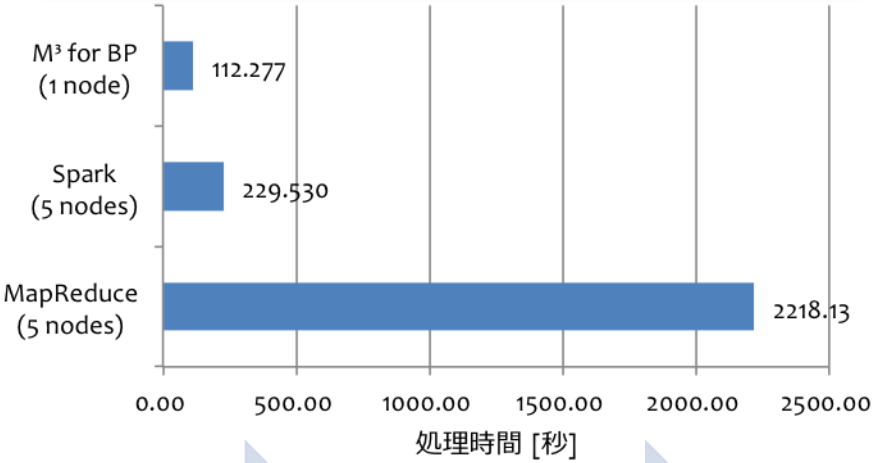
Asakusa on M³BPの性能

- Hadoop/Spark5ノードとM³ for BPにて性能比較
  - Microsoft Azureを利用し、実際の業務バッチで計測
  - 入力データサイズは、約5GB

表1　評価環境　　* 計算ノードとは別に管理用ノードが1台必要

	MapReduce	Spark	M³ for BP
バージョン	2.7.2	1.6.1	0.1.0
Java処理系	Java SE Development Kit 8 Update 74		
C++コンパイラ	N/A		GCC 4.8.5
OS	CentOS 7.1		
インスタンスタイプ	Microsoft Azure Virtual Machines D5 v2 16CPUコア、メモリ56GB		
計算用ノード数	5 (*)		1

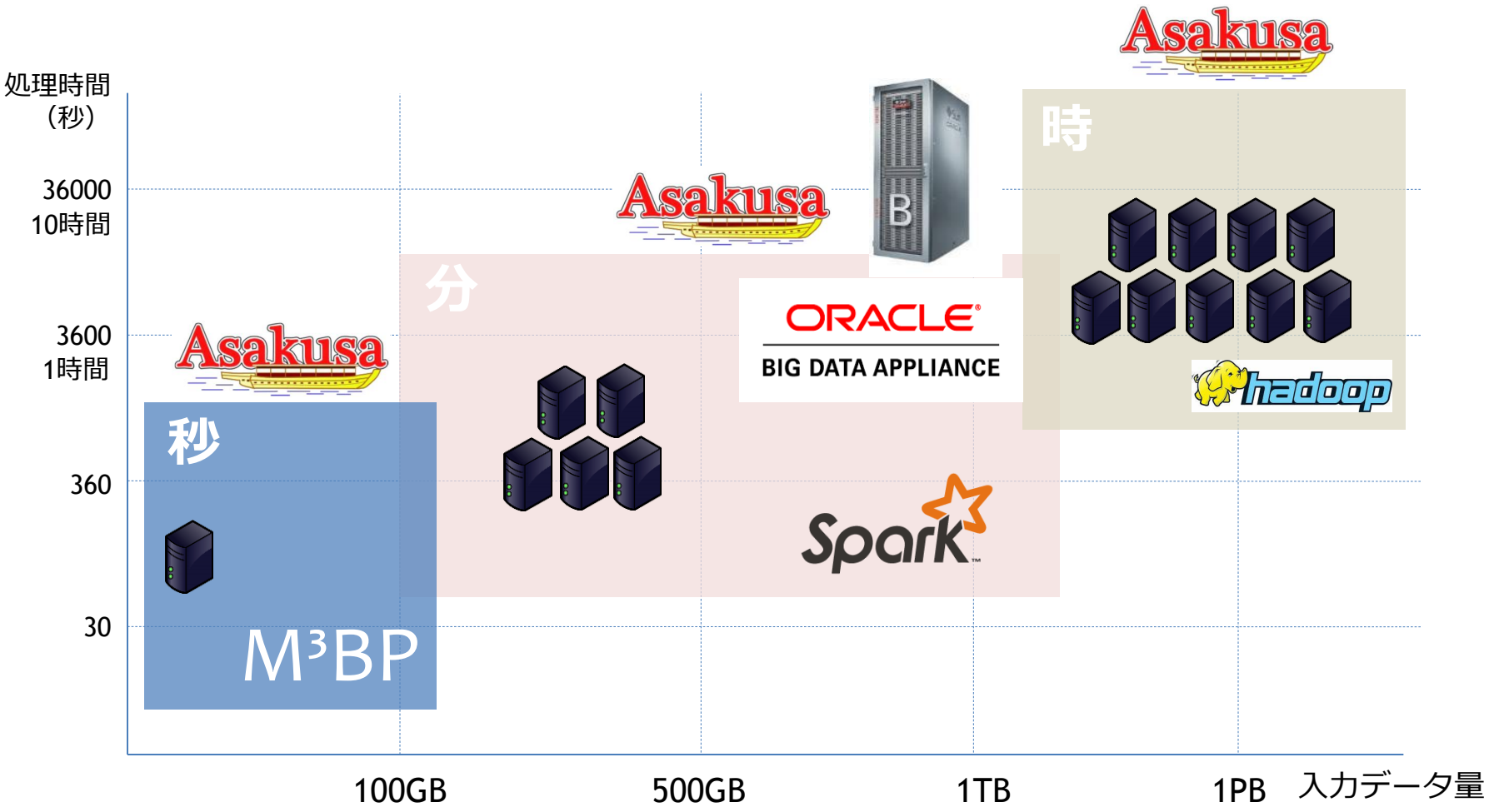
表2 評価結果
- Sparkの5ノードに対し、M³ for BP の1ノードで約2倍の性能



RDBMS 4時間以上 → Hadoop 20分まで短縮 → Spark 5分まで短縮 → M³BP 2分に短縮

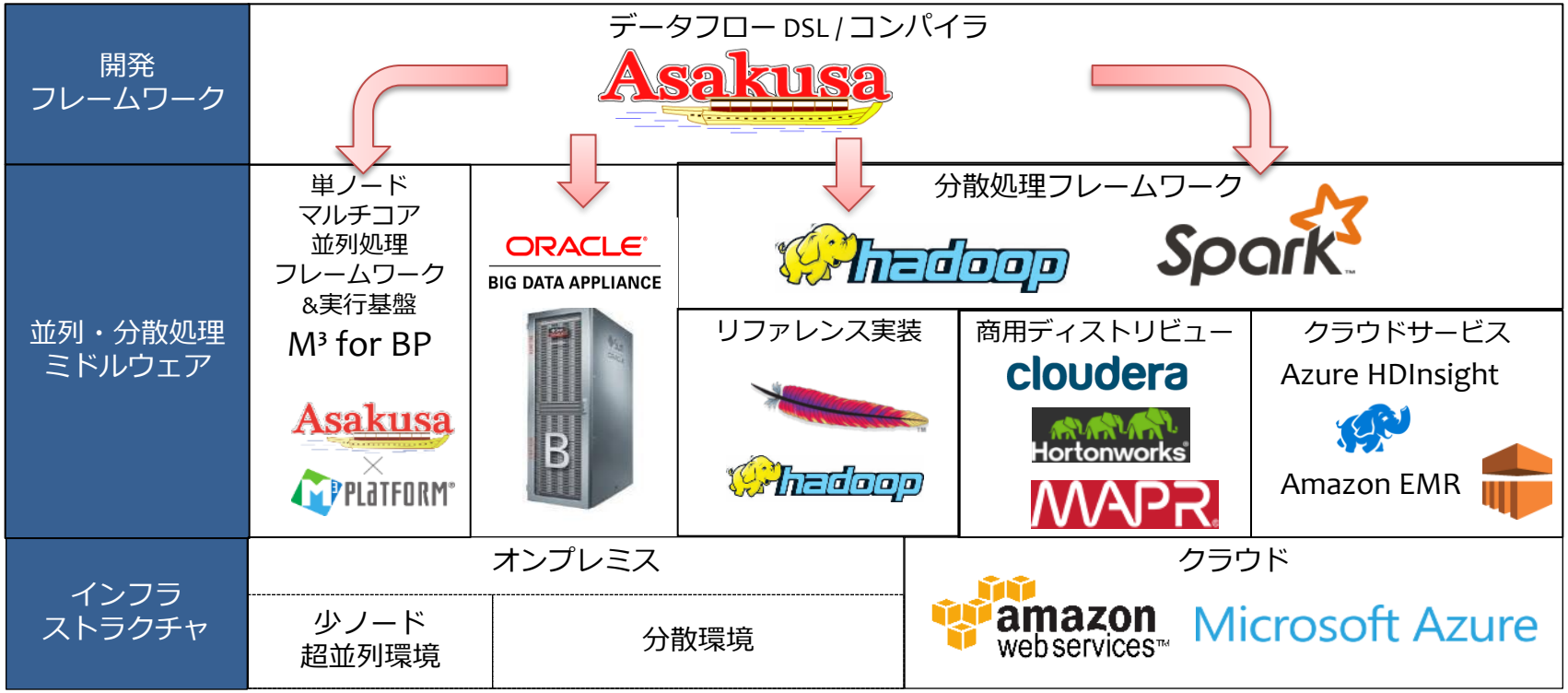
最適な環境を選択し、容易に移行可能

ユースケースに応じて、最適な実行環境で試行してみることができる



Asakusa Frameworkの特徴(ポータビリティ)

様々な分散・並列処理基盤をAsakusa Frameworkにより選択可能に



NAUTILUS

直近のAsakusa Frameworkの動向

Asakusa Framework 直近の動向

Oracle BDAにAsakusa Frameworkを搭載

分析処理に強みを持つBig Data Applianceと、業務バッチ処理の開発を容易にするAsakusa Frameworkを組合せ、エンタープライズのデータ処理ニーズに対応する

分散処理の開発が容易な
バッチ処理フレームワーク

Asakusa

NAUTILUS

RDBMSとの高い親和性

ORACLE

- Oracle Big Data Integrator
- Oracle Big Data SQL
- Oracle Big Data Connectors

豊富な分析ツール

ORACLE

- Oracle Big Data Discovery
- SQL Developer & SQL Tools
- Oracle R Advanced Analytics

可用性・拡張性の高い分散処理基盤

cloudera
Ask Bigger Questions

hadoop

HIVE

Spark

高性能・高信頼基盤

ORACLE

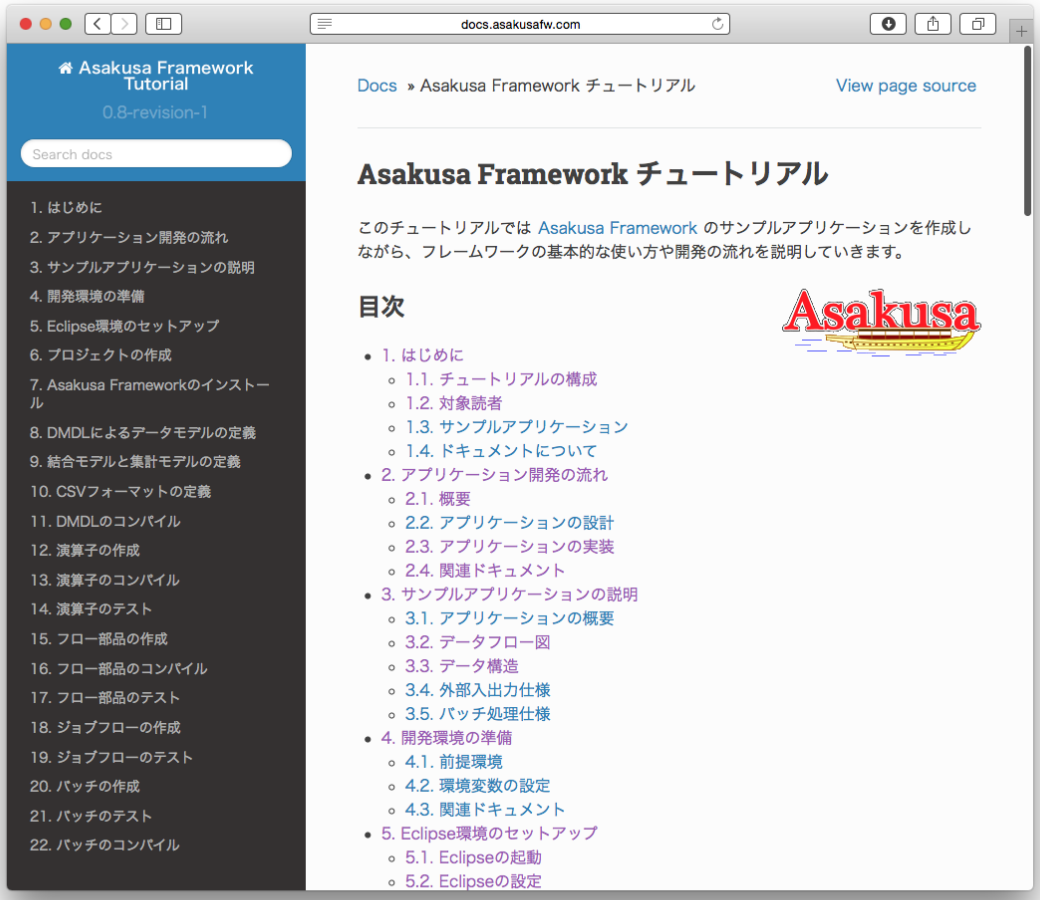
一元化された運用管理ツール
エンタープライズレベルサポート

事前構築・最適化された高性能ハードウエア

Asakusa Frameworkチュートリアル

NAUTILUS

- Asakusa　Frameworkチュートリアル
  - サンプルアプリケーション
    を作成しながらフレームワー
    クの基本的な使い方を学ぶ
    ことができる。
  - 実際の開発の流れに沿って
    構成されているので、
    初めてアプリケーションを
    開発する方でも学習を行う
    ことが可能



Asakusa Framework0.9.0最新版

- Asakusa Framework
  - リリース方式の変更
    - Asakusa Framework本体、Asakusa on Spark、Asakusa on M³BPのバージョンが乱立した問題に対処。組み合わせをまとめて提供
  - WIndGate Direct Path Insert
    - Oracleのダイレクト・パス・インサートを利用するクエリーを発行するためのオプションが追加
  - Direct I/O CSVフィールドのクォート方式の指定
    - Direct I/O CSVでフィールド値の内容に関わらず、常にクォート処理を行うモードが追加されました。
    - また、CSVの各フィールドで個別にモードを指定するためのDMDL記述が追加されました

Asakusa Framework0.9.0最新版

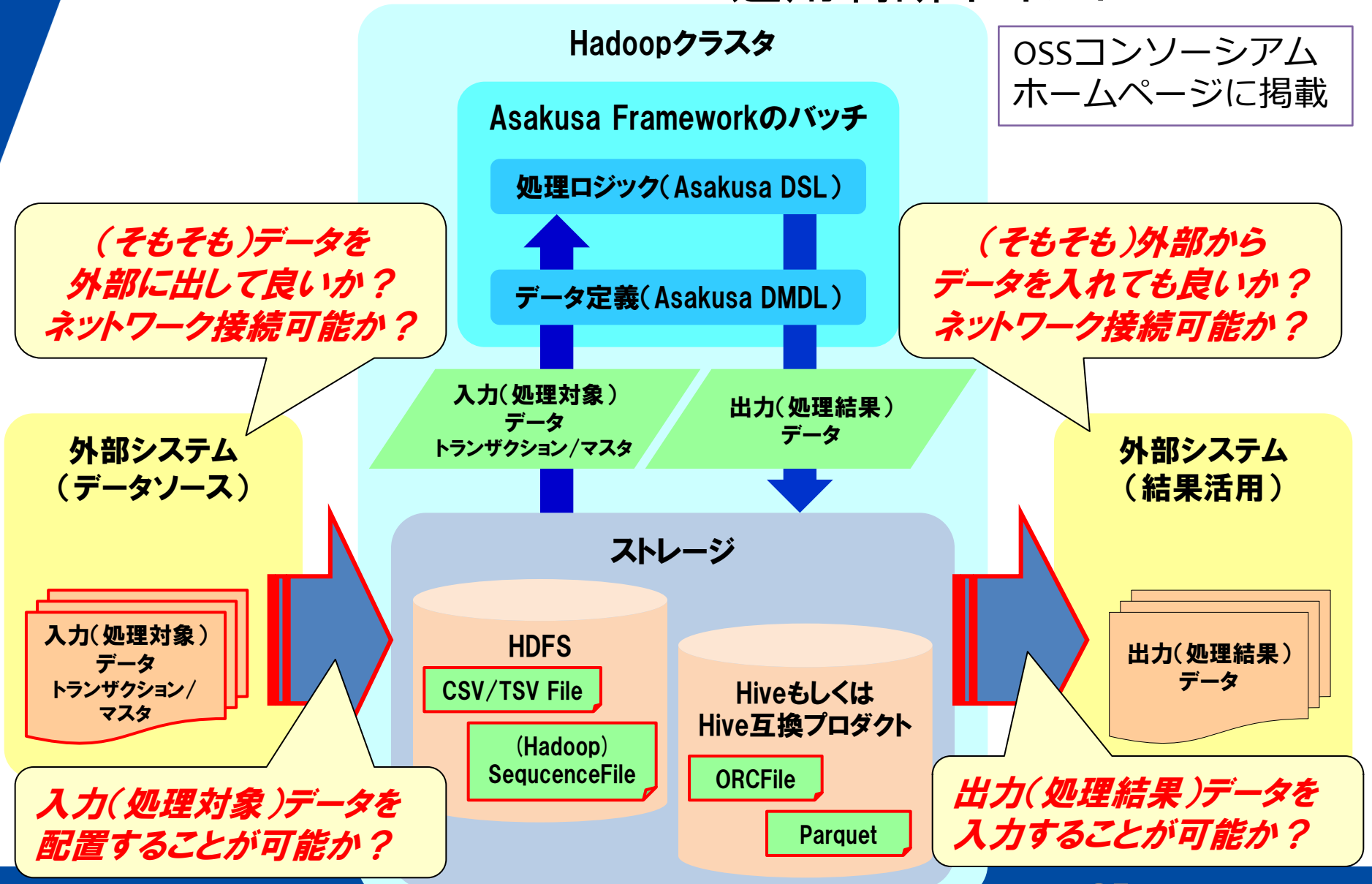
- Asakusa on Spark
  - Spark2.0対応
  - Direct I/Oの出力処理をSpark上で実行
- Asakusa Vanilla（試験的機能）
  - スモールジョブ実行エンジンの再実装
  - 単一ノード上で、JVM環境で、軽量、コンパクトに実行可能なエンジン
  - 将来的にエミュレーションモードで動作

  - 詳細は、後の神林の講演とAsakusa公式ドキュメントで

Asakusa Framework 適用領域・事例紹介

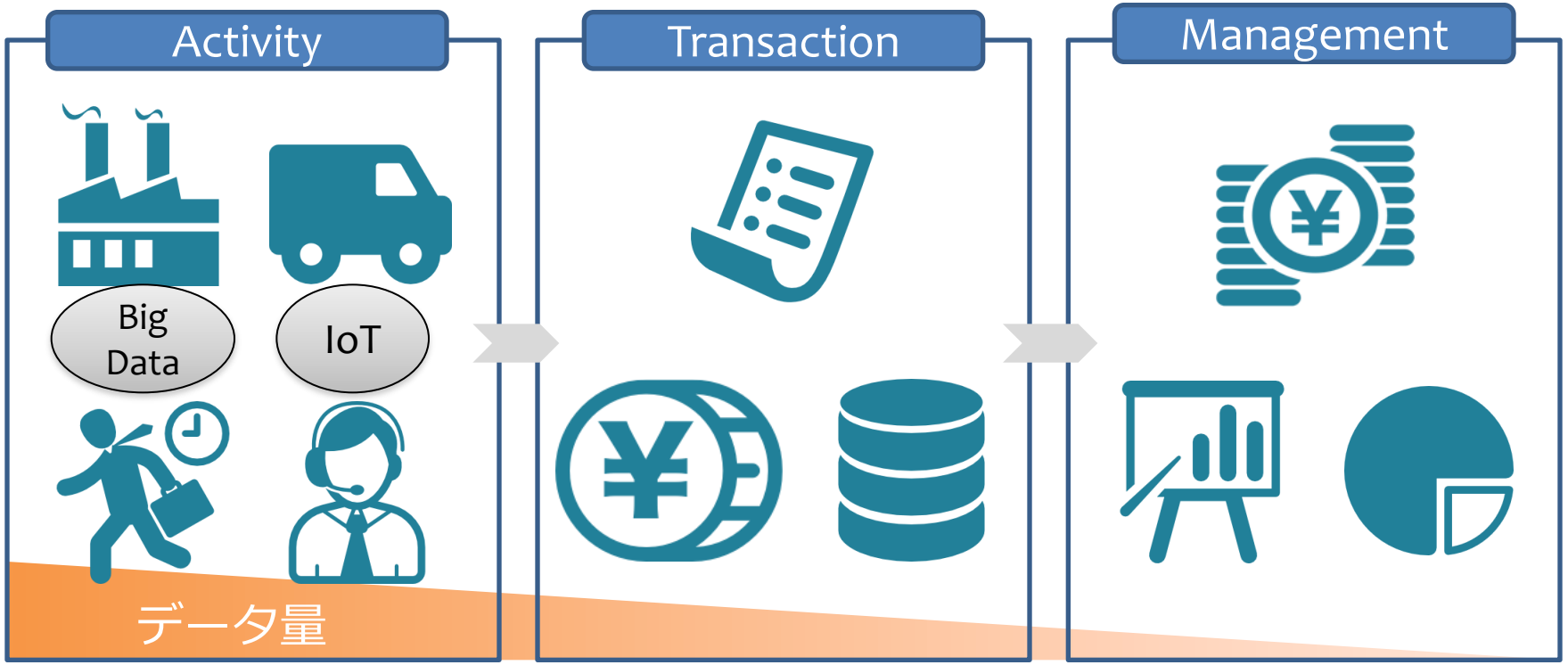
AsakusaFramework適用判断ポイント

NAUTILUS



OSSコンソーシアム ホームページに掲載

Asakusaの適用領域の変化



情報系業務
今後のAsakusaの適用領域

基幹系業務
現在のAsakusa適用領域

AsakusaFramework導入事例

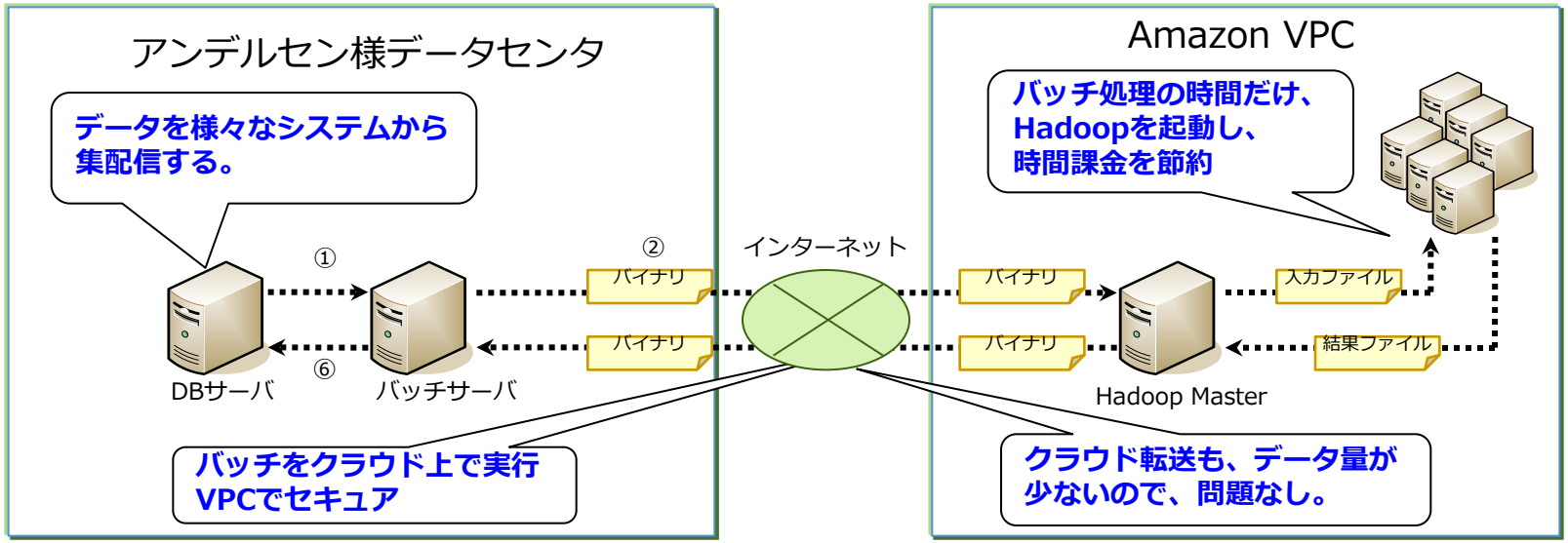
- 金融業、社会インフラ、製造業、流通業など様々な業界にご採用いただいております

にしてつストア 本部基幹・会計システム	SAKURA Internet データセンター原価計算処理	通信キャリア パケット料金計算システム
ANDERSEN 原価計算・生産管理システム	大手都市銀行 与信管理シミュレーション	設備サービス 基幹　年次バッチ処理
九州電力 設備巡視計画書作成システム	CTC Challenging Tomorrow's Changes クラウド高速データ処理基盤	ネット通販 原価計算システム

アンデルセン様　原価見える化事例

原材料の製品原価計算バッチが遅い

- 原材料からの製品原価計算で4時間のバッチを20分に短縮
  - 110万の原材料、3000品目の原価をツリー構造の積上げ計算を実施
  - BOMの展開・原価の４時間バッチで、週に２回実行するのが限界



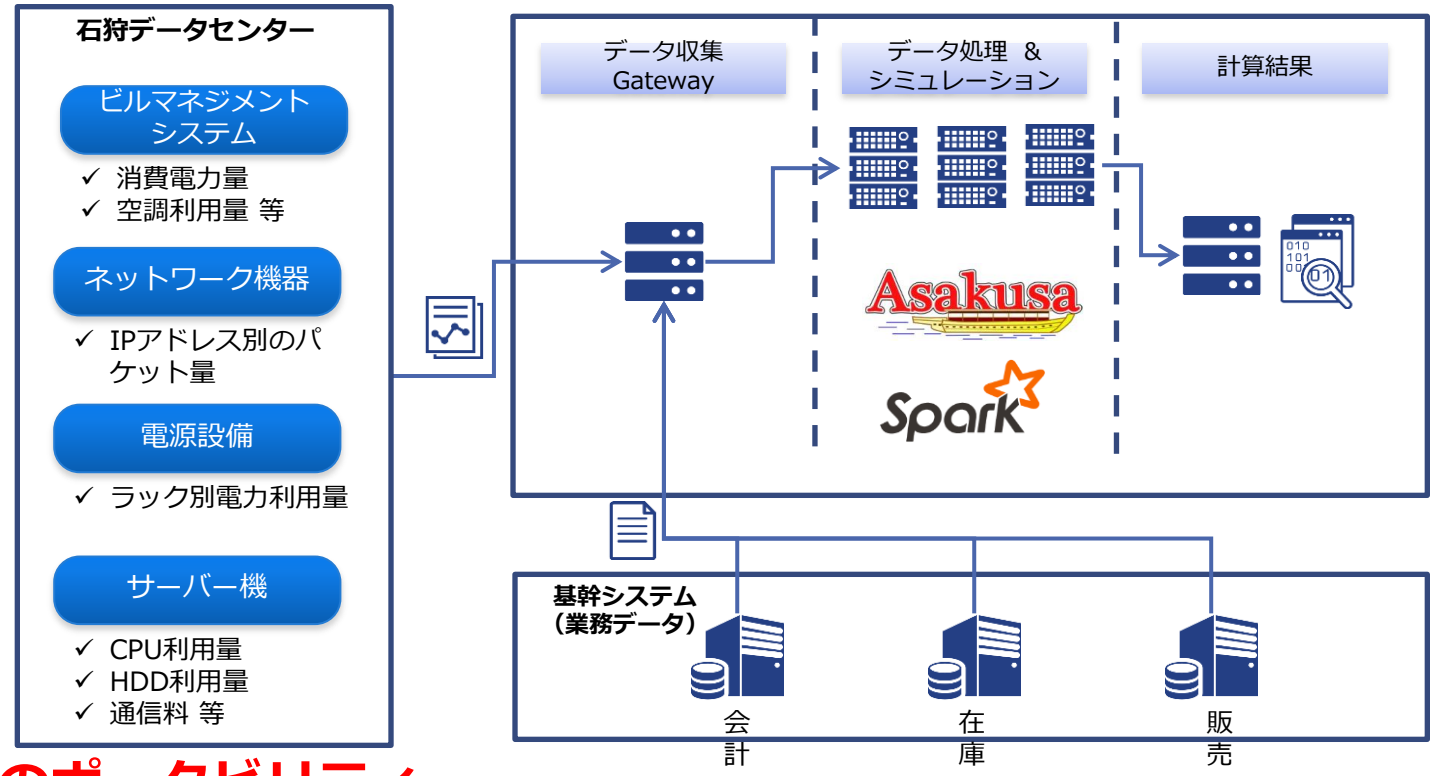
Asakusaのポータビリティ



さくらインターネット様

原価分析/IoTソリューション事例

- 電力量やトラフィック量等の膨大なログデータと、ユーザ毎の会計データを紐つけ原価を算出するシステムを構築し、石狩データセンターのVPSと専用サーバの原価計算を実施
- RDBMSで想定20時間かかるバッチをMapReduceで1時間に短縮。更にSparkで約10分に



Asakusaのポータビリティ

第１段階 MapReduce → 第２段階 Spark → 第３段階 M³BP

遅いバッチ処理は、まだまだある



バッチは遅いまま、手付かずで残っている

１、２時間のバッチ処理は、企業内に無数にある

- **バッチ処理が遅いのは現状の課題になっていない。**
  - 昔からできてないから、遅くても仕方ないと思われている
- **ビッグデータ、機械学習などの分析のブームでデータ量は更に増える**
  - データを活用するためには、既存のバッチ処理も早くする必要がでてくる。
  - せっかくの分析も、データの処理時間が遅いと業務の役に立たない
- **ただし、遅いバッチが本当に高速化できるかどうかわからない**
  - どのテクノロジーで、解決をするのが最適なのか？

データ分析だけでなく最新テクノロジーも仮説・検証

- **Hadoop/Spark/M³BP等のエンジン選択の前に**
  - まず、実現したい業務課題を発見することが先決
    - Hadoop/Sparkが流行っているから導入してみたいわけではない
  - 大事なのは、テクノロジーでなく、業務課題の解決です。
- **最新テクノロジー採用の仮説**
  - 今、この部分の処理時間がかかっているが、もし劇的に高速化したら、どんなメリットがでるだろう！？
  - この分野、領域に、この技術を活用をすると、こんなメリットがでるはず！

    という仮説が大事
- **データ分析も仮説・検証、最新テクノロジーも仮説・検証**
  - MapReduceが早いの？それともSparkの方が良いの？M³BPなの？
  - 本当に使えるのかどうか、やってみないとわからない
  - 更に新たな高速エンジンが出てくる可能性もある
    - では、全部試すのに、アプリケーションを都度開発をしますか？
- **アプリケーションへの投資が最も大事です。**

ぜひ、*Asakusa Framework*を試してみて下さい！



コミュニティによる勉強会や、OSS団体活動も

- ユーザーによる勉強会も開催されていま。トレーニングもあります。







- オープンソースビジネスを推進する団体であるOSSコンソーシアムにて、2014年7月よりAsakusa Framework部会の設立
  - **Asakusaパートナーと参加企業メンバー**
    - NTTコムウェア株式会社
    - アクセンチュア株式会社
    - 伊藤忠テクノソリューションズ株式会社
    - 株式会社日立ソリューションズ
    - 新日鉄住金ソリューションズ株式会社
    - 東芝ソリューション株式会社
    - シャープ株式会社
    - みずほ情報総研株式会社
    - 日本IBM株式会社

HITACHI Inspire the Next
TOSHIBA Leading Innovation
NS Solutions
MITSUBISHI ELECTRIC Changes for the Better
NTTコムウェア

Asakusa Framework Links

- Asakusa Framework ドキュメント
  - http://asakusafw.s3.amazonaws.com/documents/latest/release/ja/html/index.html
- Asakusa Framework チュートリアル
  - http://docs.asakusafw.com/basic-tutorial/0.8/release/ja/html/index.html
- Asakusa Framework コミュニティサイト
  - http://www.asakusafw.com/
- asakusafw.com 技術情報: バッチ設計と実装ガイド
  - http://www.asakusafw.com/techinfo/methodology.html
- asakusafw.com 技術情報: 参考資料
  - http://www.asakusafw.com/techinfo/reference.html