SOAを実現するApache Camelとは？

日本Apache Camelユーザ会

古関　伸行

2011年11月20日

発表者

- 古関　伸行(こせき　のぶゆき)
- やっていること
  - 以前:
    EAI/BPM製品の展開、導入支援
    SOA関連情報の展開
  - 最近:
    CamelをEAI風に使って展開、導入支援

発表内容

この発表では、

- Apache Camelはどういったものか？
- Apache Camelをどのように使うか？

について説明します。

目次

- Apache Camelとは？
- Camel概要
- Camelの使い方
- 最後に

- Apache Camelとは？
- Camel概要
- Camelの使い方
- 最後に

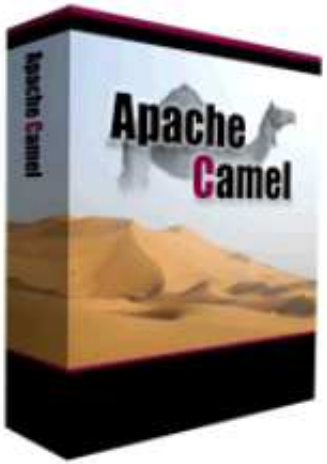
Apache Camelとは？

一言でいえば、

- ルーティングエンジン
- 多数の接続用コンポーネント

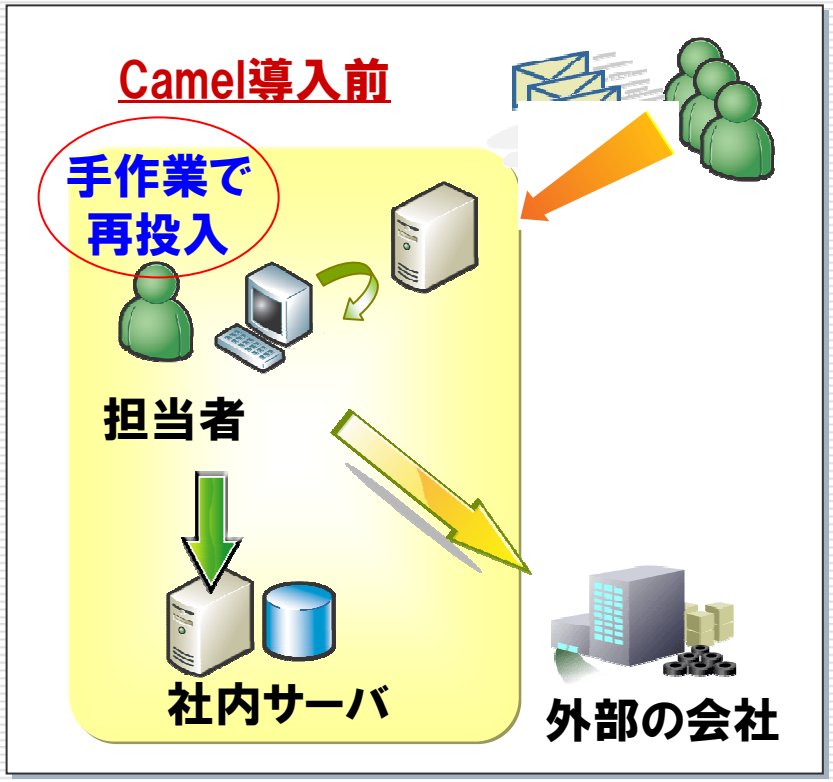
Apache Camelとは？ ～歴史～

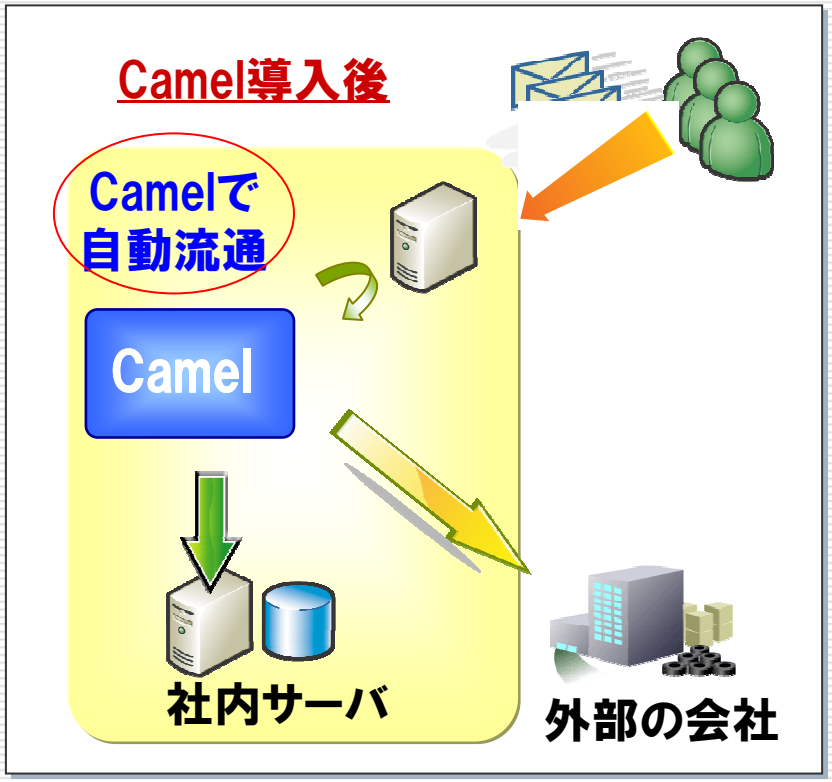
日本Apache Camelユーザ会

	2007年	2008年	2009年	2010年	2011年	2012年
1系	▲1.0 7月 ▲1.1 ▲1.2	▲1.3 ▲1.4 ▲1.5	▲1.6 Topプロジェクト	▲1.6.4		
2系		2系になってから頻繁にリリース（2～3ヶ月毎）	▲2.0 ▲2.1 ▲2.2 ▲2.3 ▲2.4 ▲2.5		▲2.6 ▲2.7 ▲2.8 最新は2.8.2 (11月現在) ▲2.9RC1	
3系					開発中	

Apache Camelとは？
～何ができるか？ その1～

□ 例えば、手作業を効率化



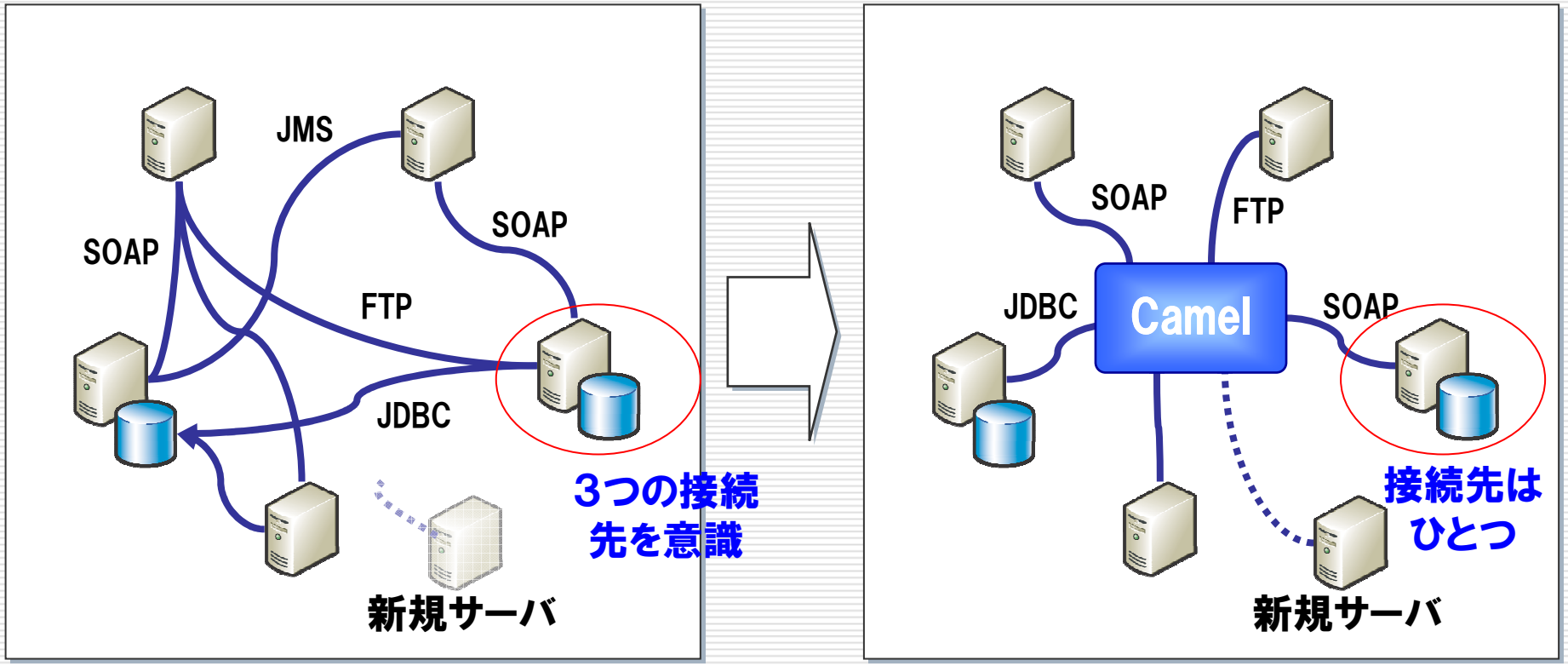


・担当者が**手作業で**実施

・Camelが**自動で**実施

Apache Camelとは？
～何ができるか？ その2～

- 例えば、スパゲティな連携を簡素化



- メンテナンスが煩雑
- 各システムが接続先を意識

- メンテナンスはCamelに集約
- Camelが接続先を振り分ける

Apache Camelとは？
～何ができるか？ その3～

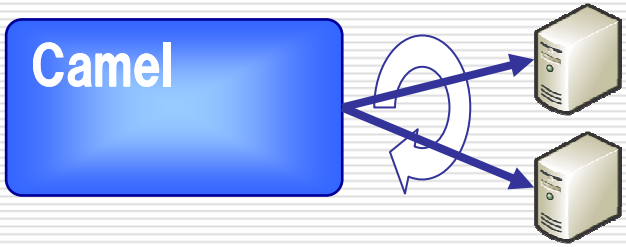
日本Apache Camelユーザ会

- 例えば、Camelが持っている機能を利用して…

一定間隔/決まった日時に
特定の処理を実行

シェルコマンド実行

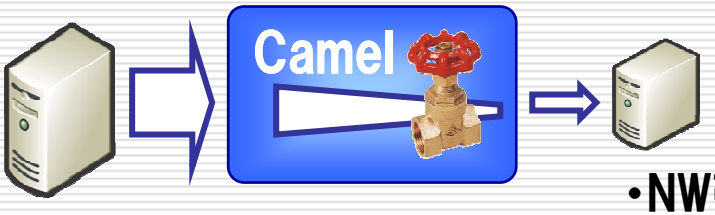
ラウンドロビンでの
負荷分散



サーバA

サーバB

流通データ量の制御



・NW帯域制限
・マシンリソース制限

Apache Camelとは？
～何ができるか？ その4～

□ Camelと他の機能を組み合わせると…

◆ Camelに機能追加

⇒ **ESB**

製品として実現
- Apache ServiceMix
- Talend

◆ Camelと別機能の組合せ

Camel ＋ BPM ⇒ **EAI/BPM**

Camel ＋ Rule ⇒ **CEP**

*ESB：Enterprise Service Bus
*EAI/BPM：Enterprise Application Integration / Business Process Management
*CEP：Complex Event Processing

- Apache Camelとは？
- Camel概要
- Camelの使い方
- 最後に

Camel概要
～特徴～

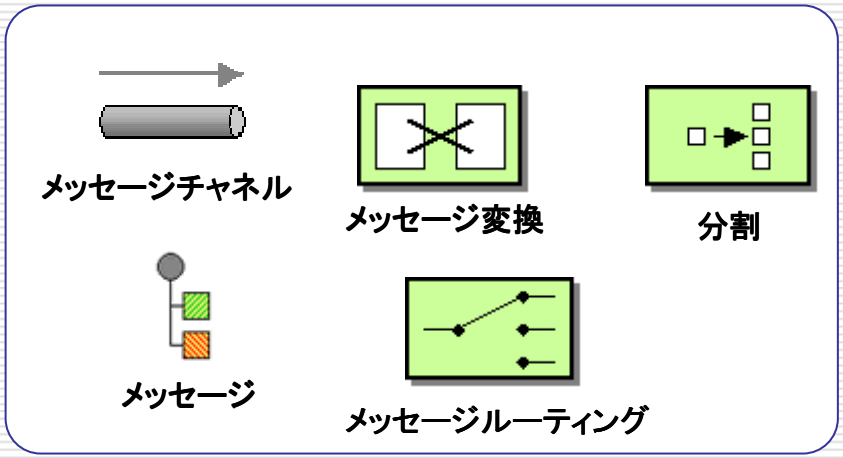
- Enterprise Integration Pattern (EIP)
- 優れた拡張性
- 多数のコンポーネント

Camel概要
～Enterprise Integration Pattern (EIP)～

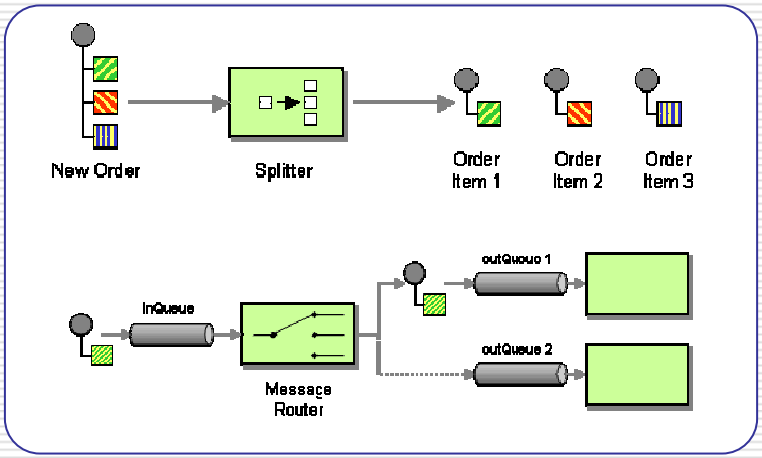
汎用的な設計パターンをGoFデザインパターンとして定義したように、エンタープライズ統合のパターンを定義

エンタープライズ統合のノウハウ → 汎用的なパターン

必要な機能を定義



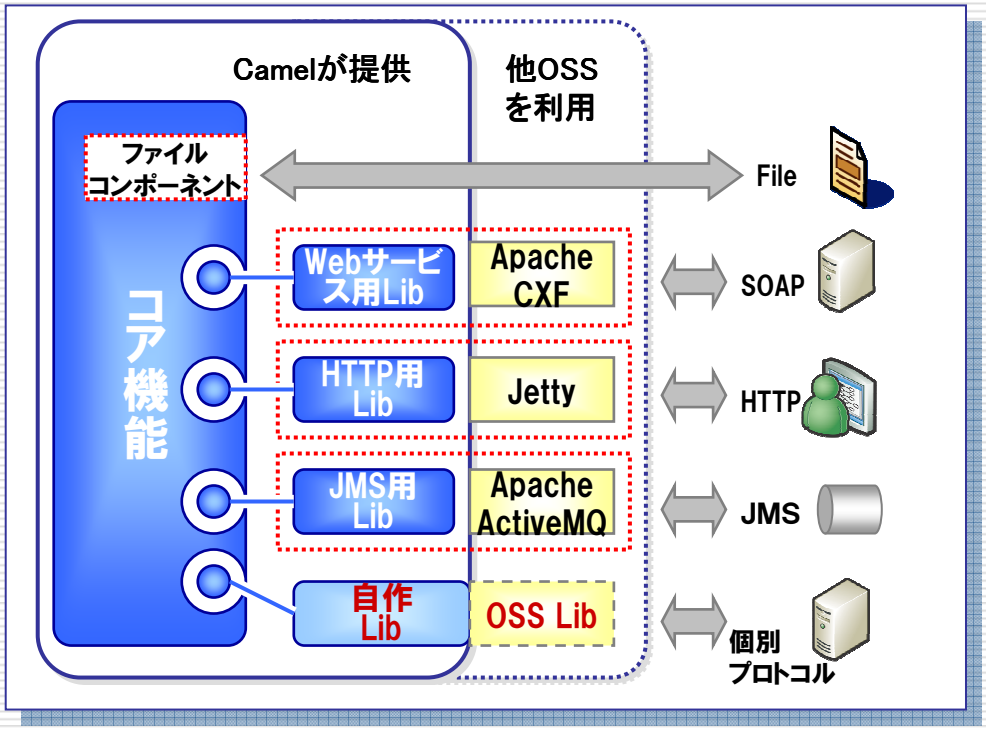
利用時の留意点を定義



⇒Camelでは多くのEIPのパターンが実現可能

Camel概要 ～優れた拡張性 1/2～

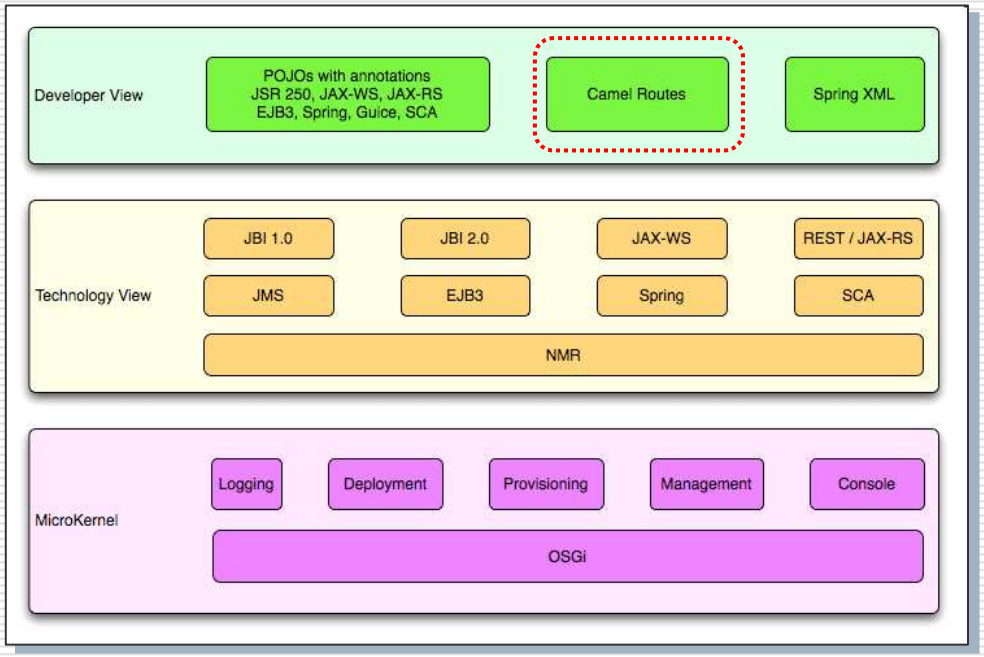
- 必要なコンポーネントのみ利用可能
  - 不要なコンポーネントは使わなくてよい
  - ない場合は自分で作成可能



Camel概要
～優れた拡張性 2/2 ～

- 単純なプロトコルスタック
  - コア機能は他ライブラリの依存性が低い
  - 他システムへの組込み/組合せが容易
    - ESB、CEPやEAI/BPMへの拡張が可能

オープンソースESBのServiceMixの場合

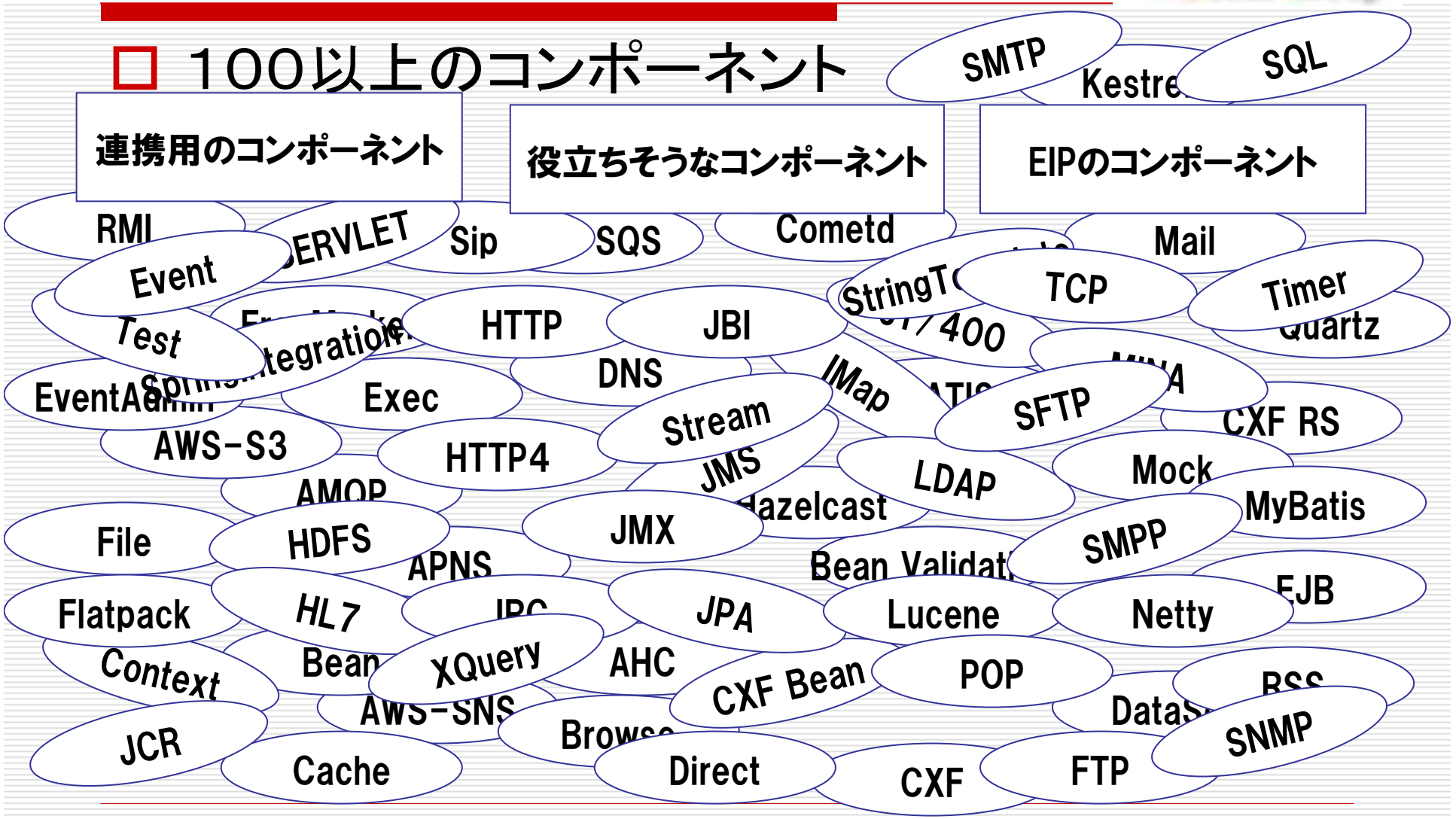


Camelと他OSSを
組み合わせて実現

http://servicemix.apache.org/architecture.htmlより引用

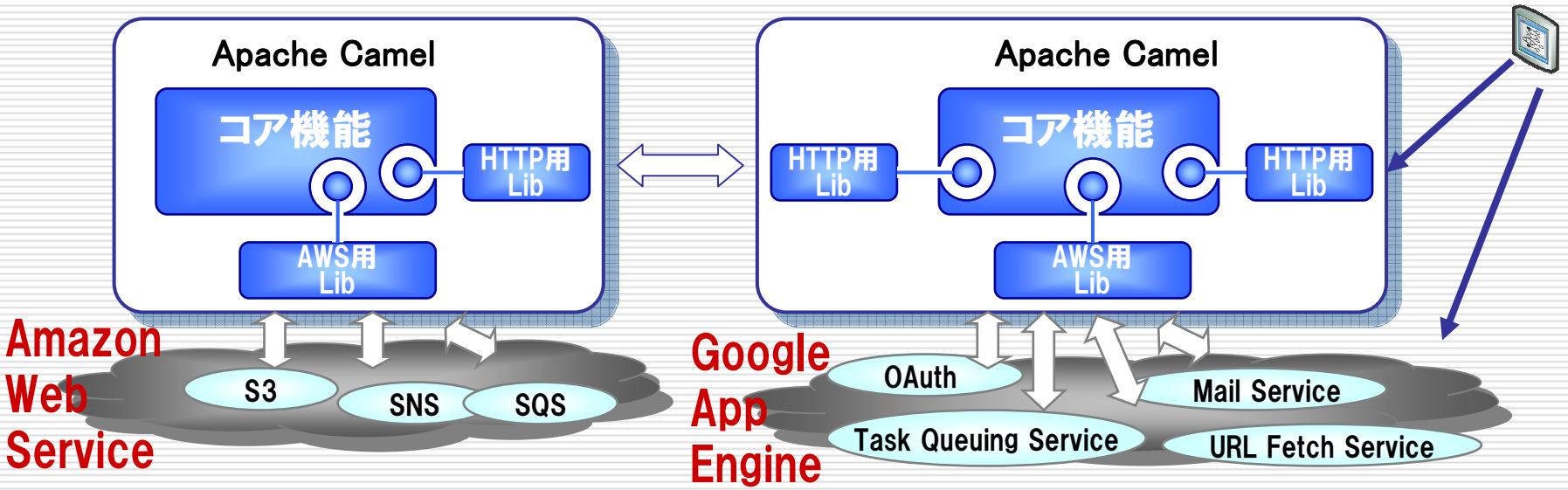
Camel概要
～多数のコンポーネント 1/2～

- 100以上のコンポーネント



Camel概要
～多数のコンポーネント　2/2～

- AWS,GAE等、注目のクラウド環境にも対応

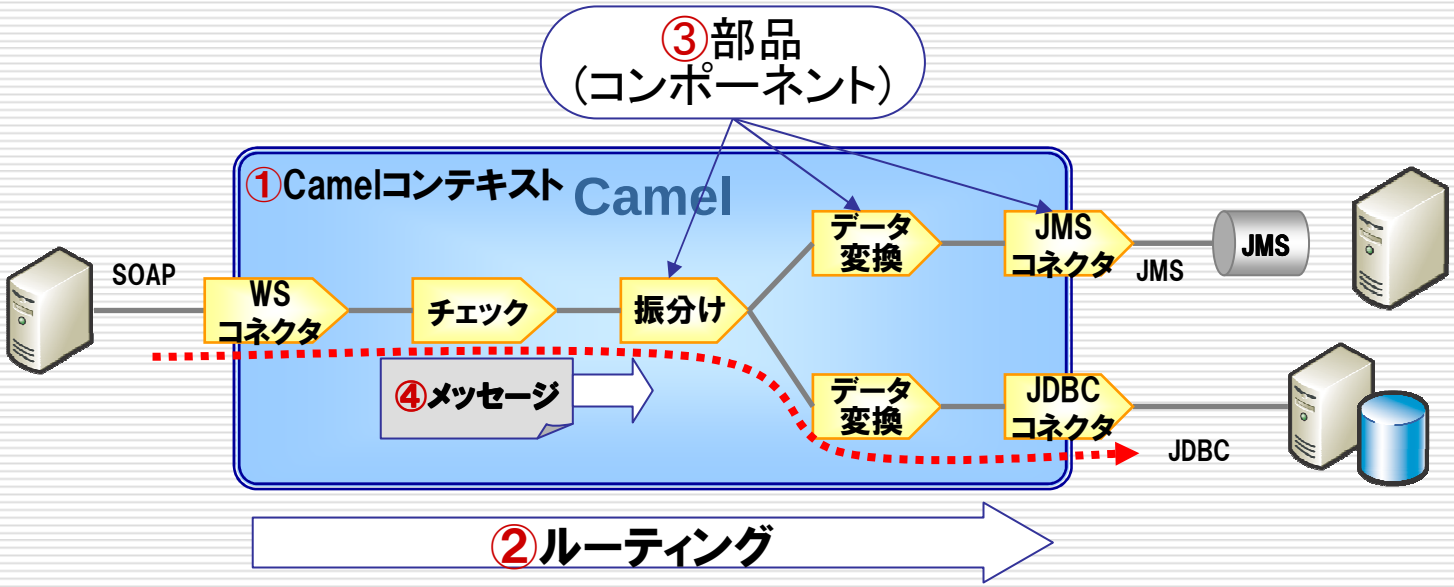


- Apache Camelとは？
- Camel概要
- Camelの使い方
- 最後に

Camelの使い方

使い方の概要

① Camelコンテキストを作成(おまじない)
② 全体の処理の流れをルーティングとして定義
③ ルーティング内の個々の処理をコンポーネント(部品)で実現
④ 実行時、定義済みのルーティングに従い、メッセージが流通



実際はJavaもしくはSpring用XMLでルーティングを定義

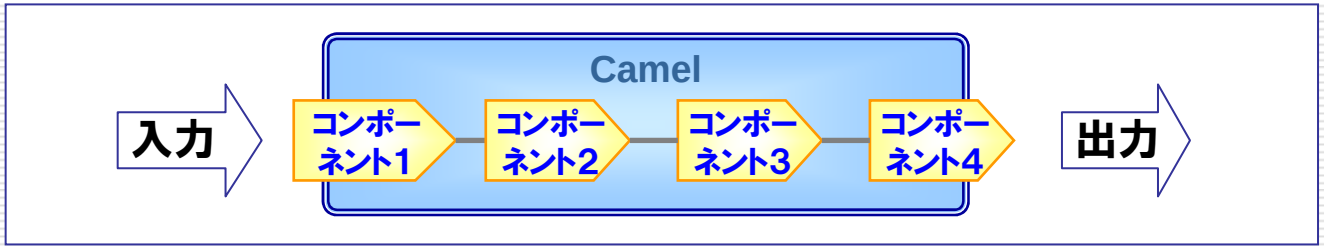
Camelの使い方 ～ルーティング～

- DSLでルーティングを定義
  - DSL：Domain Specific Language
    Camel固有のルーティング記述用の言語
- 複数のDSLでルーティング定義が可能
  - Java DSL
  - Spring DSL
  - Scala DSL

Camelの使い方 ～ルーティング～

□ ルーティングの基本

コンポーネント情報を順番に並べる。最初がfrom、それ以降はtoでつなげる



<Java DSLで記述>

```
from(“コンポーネント1情報”)
  .to(“コンポーネント2情報”)
  .to(“コンポーネント3情報”)
  .to(“コンポーネント4情報”);
```

処理の順番

<Spring DSLで記述>

ルーティングの範囲

```
<route>
  <from uri=“コンポーネント1情報” />
    <to uri=“コンポーネント2情報” />
    <to uri=“コンポーネント3情報” />
    <to uri=“コンポーネント4情報” />
</route>
```

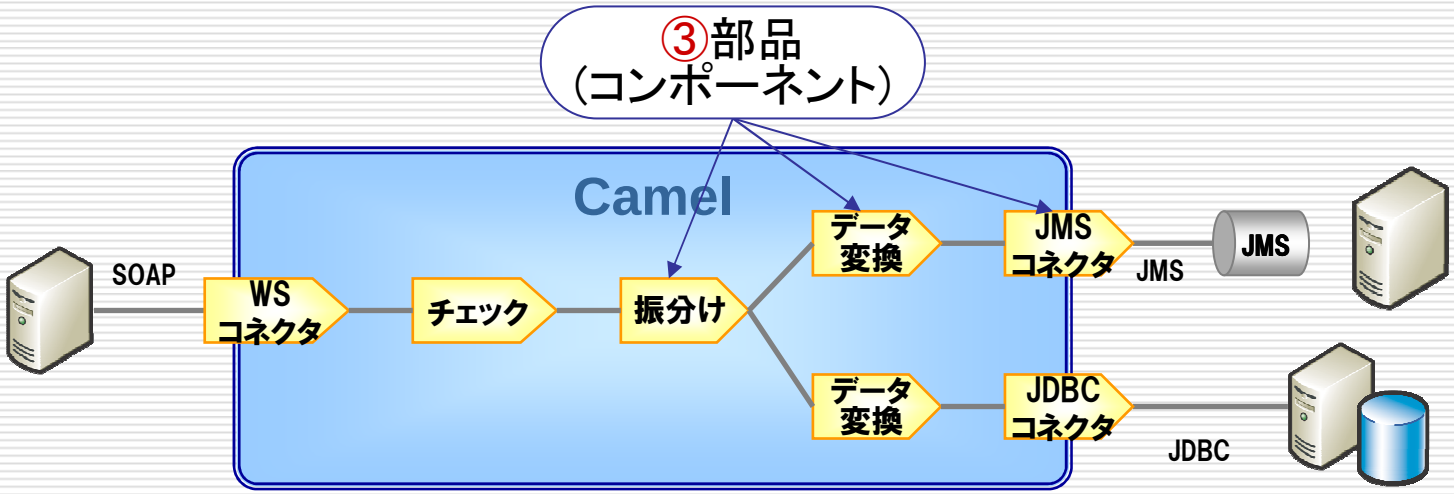
処理の順番

※：from、to以外にもCamelではDSLを定義している（choice, when, wireTap, etc.）

Camelの使い方

使い方の概要

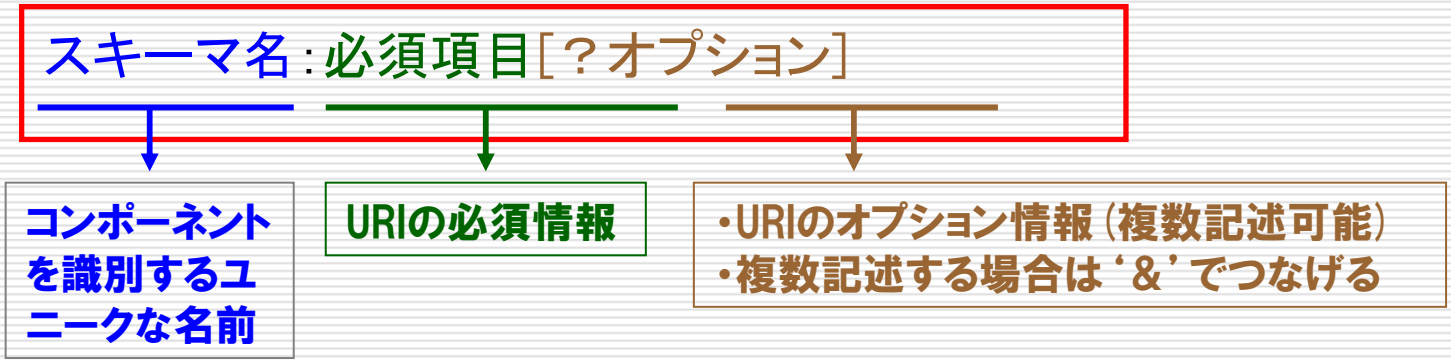
① Camelコンテキストを作成(おまじない)
② 全体の処理の流れをルーティングとして定義
③ ルーティング内の個々の処理をコンポーネント(部品)で実現
④ 実行時、定義済みのルーティングに従い、メッセージが流通



Camelの使い方 ～コンポーネント～

コンポーネントはURIで指定

〈URIの基本〉



/tmp/abcフォルダを5秒間隔でポーリング

file:/tmp/abc?delay=5000

Camelの使い方 ～コンポーネント～

- オプション指定で更に便利に！

新たな要件の追加

処理後、.doneディレクトリにファイルを移動したい

フォルダを再帰的にポーリングしたい

```
file:/tmp/abc?delay=5000&move=.done
```

```
file:/tmp/abc?delay=5000&recursive=true
```

Camelの使い方 ～コンポーネント～

日本Apache Camelユーザ会

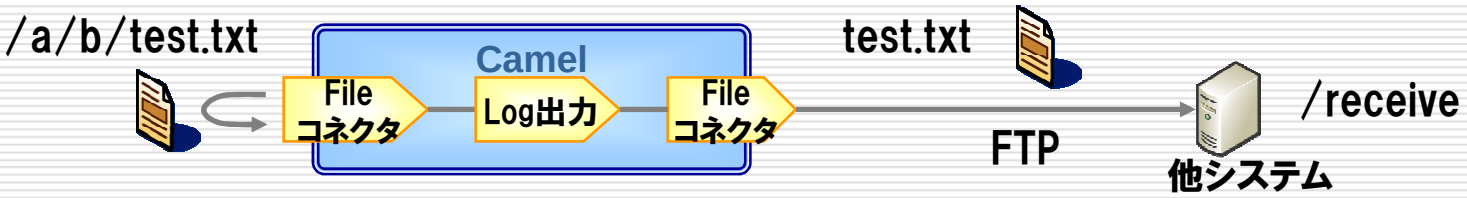
- 各種プロトコルに対応したコンポーネント
  - 書式はすべて「スキーマ名：必須項目？オプション」
  - プロトコル個別のお作法を学ぶ手間が低減

URIの記述例

コンポーネント	プロトコル	URI
CXF	Webサービス	cxf:address [?serviceClass=...]
File	File	file:fileOrDirectoryName [?options]
FTP/SFTP/FTPS	FTP/SFTP/FTPS	ftp:// [username@] host [:port] /directoryname [?options]
JDBC	JDBC	jdbc:dataSourceName [?options]
Jetty	HTTP(サーバ)	jetty:http://hostname [:port] [/resourceUri] [?options]
JMS	JMS	jms: [queue:|topic:] destinationName [?options]
AWS-S3	Amazon Simple Storage Service (S3)	aws-s3://bucketname [?options]
GHttp	URL fetch service (Google App Engine)	ghttp://hostname [:port] [/path] [?options] ghttp:///path [?options]
Log	-	log:loggingCategory [?options]

Camelの使い方 ～ルーティング例～

- ルーティングの記述例
  - 他システムへのファイル送信



<Java DSLで記述した場合>

```
from("file:/a/b")
  .to("log:testlog")
  .to("ftp:/receive");
```

<Spring DSLで記述した場合>

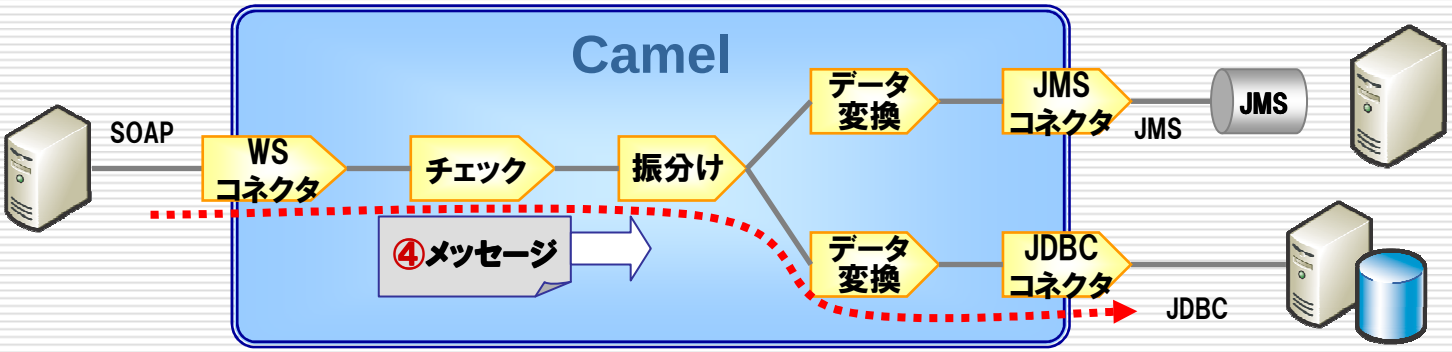
```
<route>
  <from uri="file:/a/b" />
  <to uri="log:testlog" />
  <to uri="ftp:/receive" />
</route>
```

※:ルーティング部分にフォーカスを当て、前後の処理は省略。以降も同様

Camelの使い方

使い方の概要

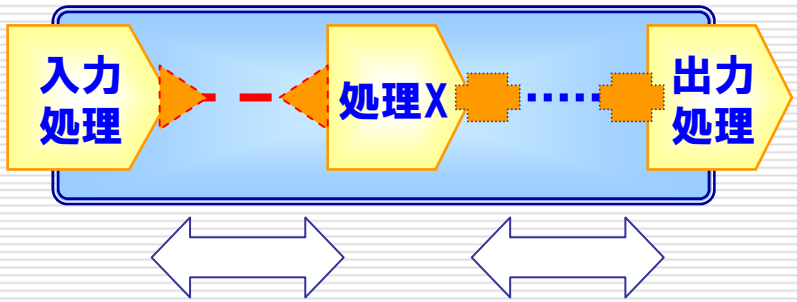
① Camelコンテキストを作成(おまじない)
② 全体の処理の流れをルーティングとして定義
③ ルーティング内の個々の処理をコンポーネント(部品)で実現
④ 実行時、定義済みのルーティングに従い、メッセージが流通



Camelの使い方
～内部で流通するメッセージ形式～

□ メッセージの話をする前に、少しだけ一般論

厳密に型を定義

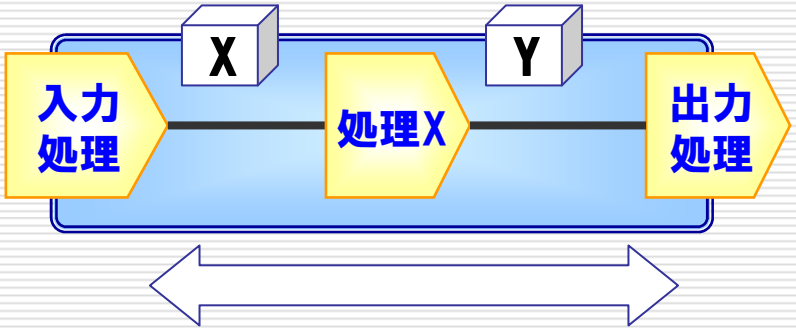


▲ の型のみ
流通を許容

■ の型のみ
流通を許容

- ユーザはデータへのアクセスが容易
- 全ての型定義を設計時に実施

利用者が型の内容を意識



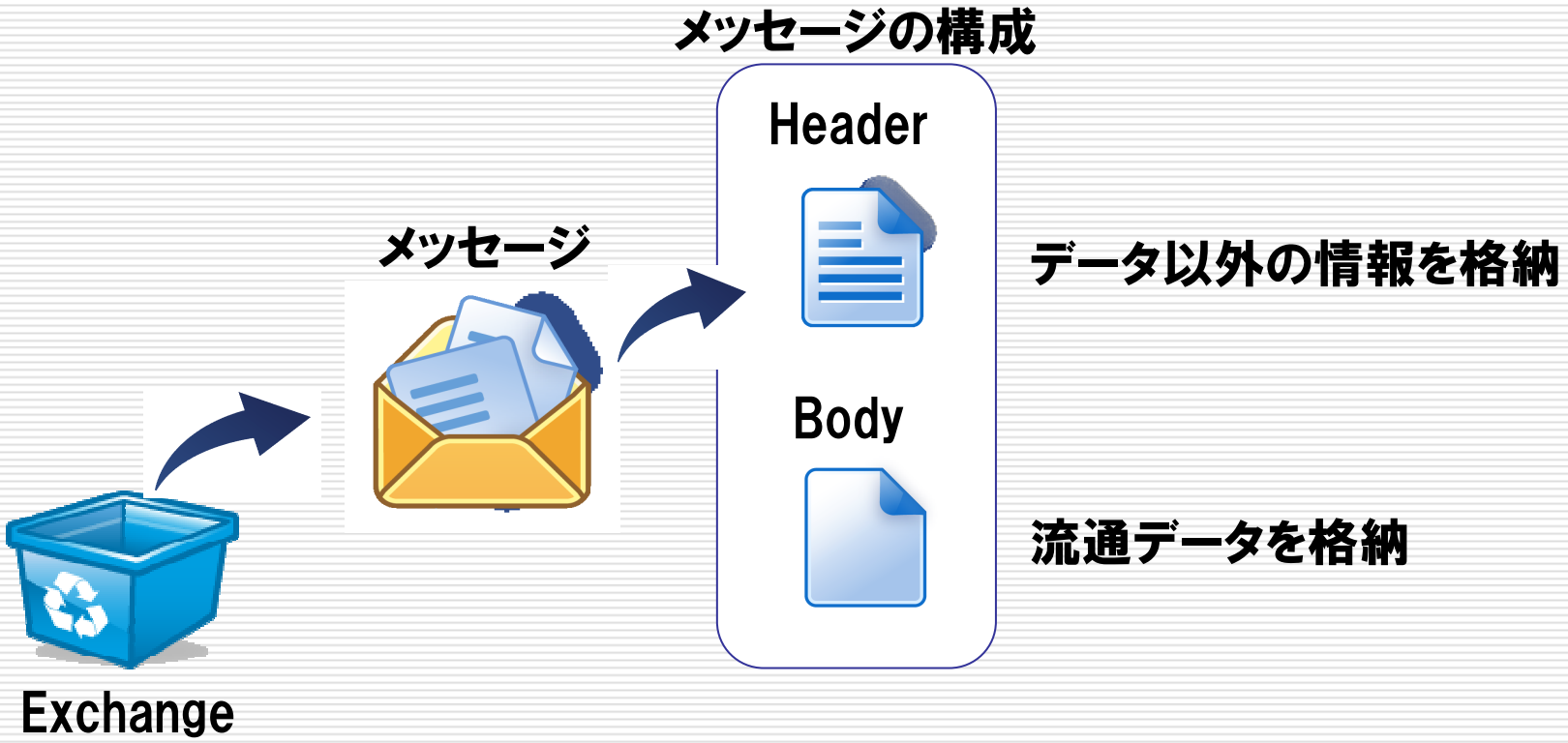
□ の中に何がどのように入っているか、ユーザが意識

- 設計時に型定義が不要
- データへのアクセスを各処理で実施

Camelはこの方法

Camelの使い方
～メッセージ形式～

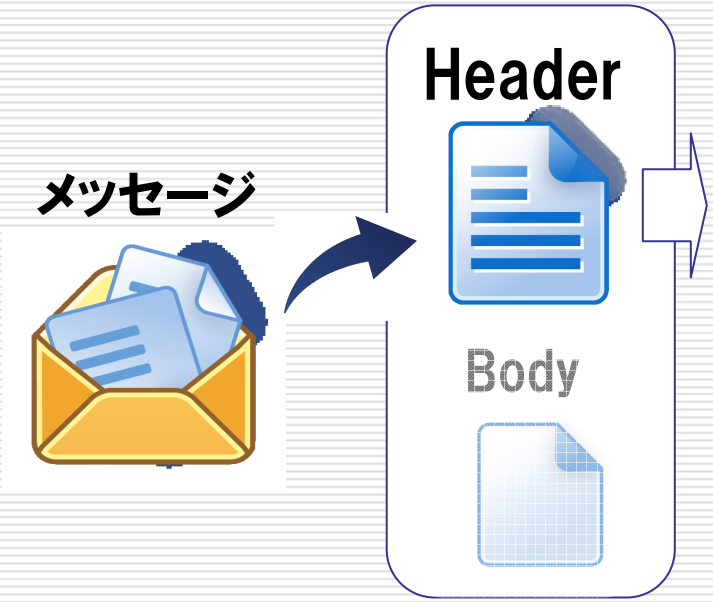
□ メッセージは“Header”と“Body”からなる



※：実際はExchange内に2つのメッセージがある（InMessage、OutMessage）

Camelの使い方
～メッセージ形式　Header～

- データ以外の情報がHeader部に詰まっている



- ◆例えば、ファイル名やディレクトリ名を格納
- ◆「キー＝値」の形式で格納
- ◆プロトコル毎に格納されるキーが異なる
- ◆ユーザが自由に追加/変更可能

ファイル受信時のヘッダ情報の例

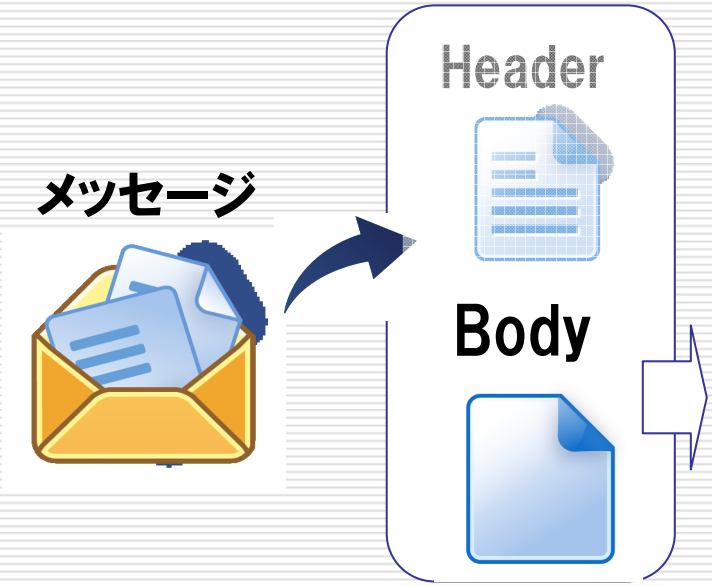
```
[CamelFilePath=C:¥work¥abc¥File.txt]
[CamelFileLength=320]
[CamelFileLastModified=yyyy/mm/dd]
…
```

Webサービス受信時のヘッダ情報の例

```
[Content-Type=text/xml;charset=UTF-8]
[operationNameSpace=http://ws.test.co.jp/]
[operationName=helloEcho]
…
```

Camelの使い方
～メッセージ形式 Body～

- Body部には流通データが詰まっている



◆流通するデータを格納
◆データの格納方法はプロトコルで異なる
◆データ以外の情報を格納しているプロトコルもある

ファイル受信時のBody部

◆クラス名は
org.apache.camel.component.file.GenericFile
◆処理対象のファイル情報を格納

Webサービス受信時のBody部

◆クラス名は
org.apache.cxf.message.MessageContentsList
◆受信データのリスト、引数のデータが順番に格納

Camelの使い方 ～データ変換～

□ 容易にBody部のデータ取得する方法を提供

データアクセス手段がないと…

・メッセージの中のデータへのアクセス手段を各処理で実装する必要がある

「処理X」でのデータアクセス手段

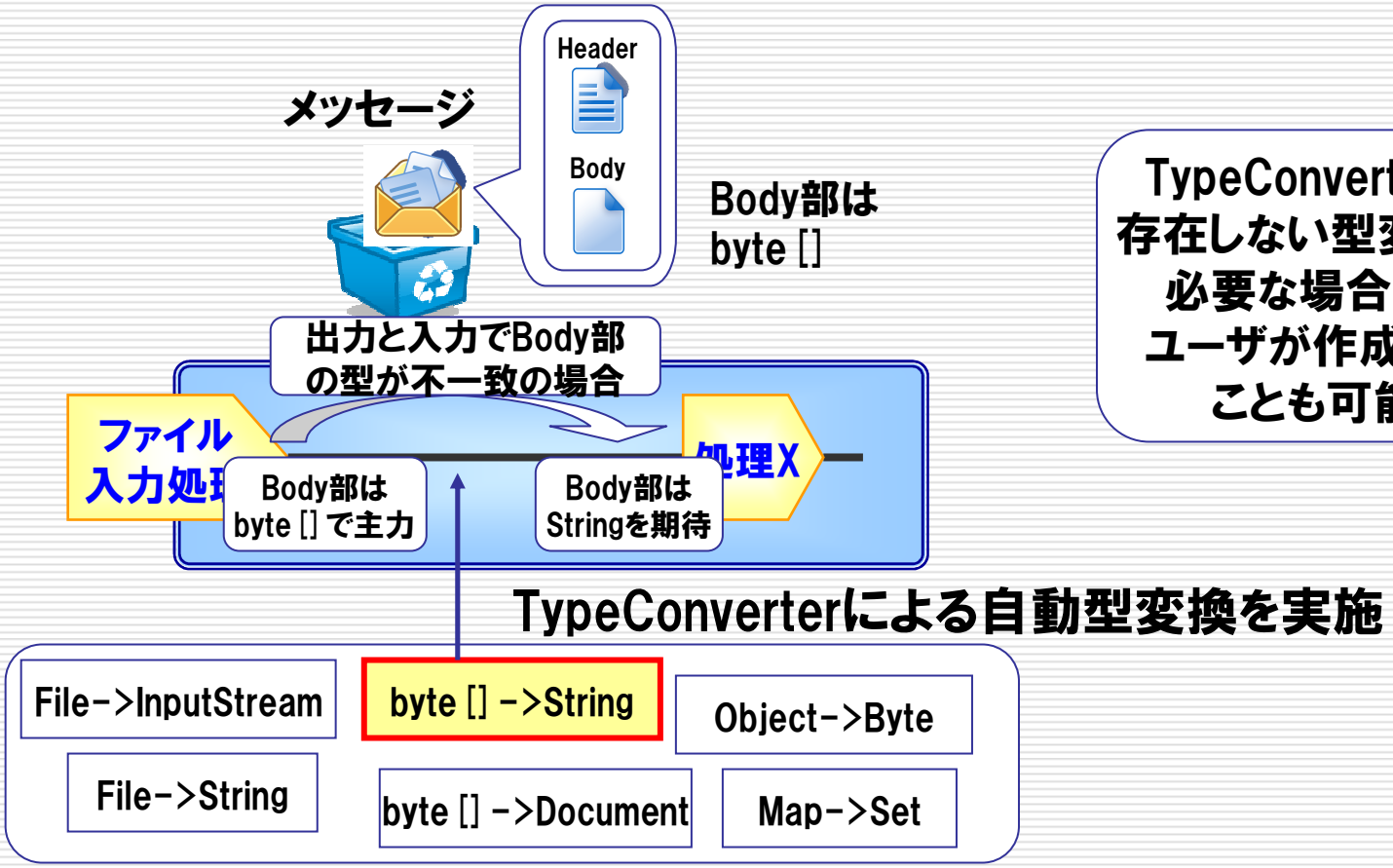
```
public class Process implements Processor {
  public void process(Exchange exchange) throws Exception {

    String data = exchange.getIn().getBody(String.class);
    …
```

処理Xはメッセージの Body部がStringかbyte [] かInputStreamか知らなくてもアクセスできる

Exchange | **InMessage** | **Body部をStringでくれ！**

Camelの使い方 ～データ変換～

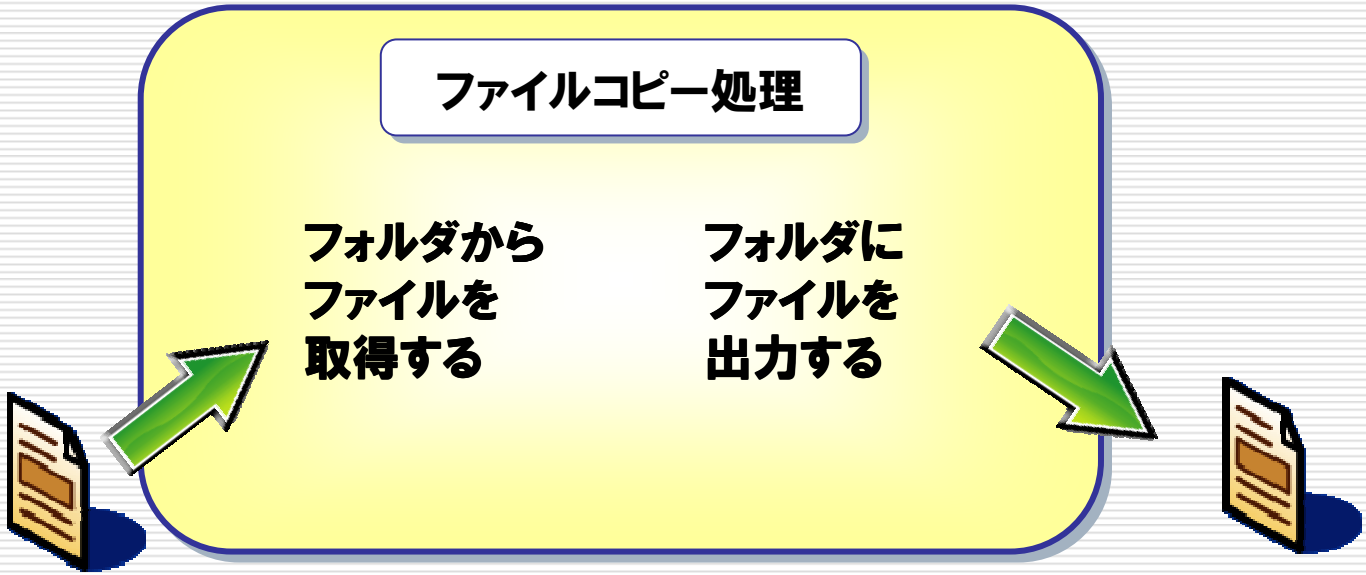
TypeConverterによる変換



TypeConverterに存在しない型変換が必要な場合は、ユーザが作成することも可能

Camelの使い方
～実際にやってみる～

- 例えば…
  ファイルをコピーする処理を考えてみる



Camelの使い方
～Javaで書いてみる～

日本Apache Camelユーザ会

- Javaでも簡単に記述可能

```
public static void main (String args [] ) throws Exception {
  File inboxDirectory = new File ("data/inbox") ;
  File outboxDirectory = new File ("data/outbox") ;
  outboxDirectory.mkdir () ;
  File [] files = inboxDirectory.listFiles () ;
  for (File source : files) {
    File dest = new File (outboxDirectory.getPath () + File.separator + source.getName () ) ;
    copyFile (source, dest) ;
  }
}
private static void copyFile (File source, File dest) throws IOException {
  OutputStream out = new FileOutputStream (dest) ;
  byte [] buffer = new byte [ (int) source.length () ] ;
  FileInputStream in = new FileInputStream (source) ;
  in.read (buffer) ;
  try {
    out.write (buffer) ;
  } finally {
    out.close () ;
    in.close () ;
  }
}
```

Camelの使い方
～Camelでやってみる～

Camelの場合(Java DSL)

```
public static void main (String args [] ) throws Exception {

  //1:最初にCamelContextを作成する
  CamelContext context = new DefaultCamelContext ();
  RouteBuilder routeBuilder = new FileToFileRoute ();
  context.addRoutes (routeBuilder);

  //2:作成したCamelContextを開始し、10秒後に終了する
  context.start ();
  Thread.sleep (10000);
  context.stop ();
}
```

```
public class FileToFileRoute extends RouteBuilder {

  @Override
  public void configure () throws Exception {
  from ("file:data/inbox?noop=true")
        .to ("file:data/outbox");
}
```

Camelの使い方 ～要件の考慮～

- ところが、普通はもっと要件が複雑…
  例えば、
  - 1分毎にファイルがあるかチェックすること
  - 出力ファイル名には日付を付与すること
  - .docの付いた拡張子は無視すること
  - 同一のファイル名がある場合は無視すること
  - サブフォルダも検索すること
  - コピーでなく移動に変更。バックアップも取って！
  - etc,…

Camelの使い方
～追加要件への対応～

- Camelの場合、オプション追加で対応可能

1分毎にファイルがあるかチェック

.docの付いた拡張子は無視

.doneフォルダにバックアップ

```
public class FileToFileRoute extends RouteBuilder {

  @Override
  public void configure () throws Exception {
  from ("file:data/inbox? delay=60000 & exclude=*.doc$ & move=.done &
                          noop=false & idempotent=true & recursive=true")
        .to ("file:data/outbox? fileName=${date:now:yyyy-MM-dd}_${file:name}");
}
```

サブフォルダも検索

コピーから移動に変更

出力ファイル名には日付を付与

同一のファイル名がある場合は無視

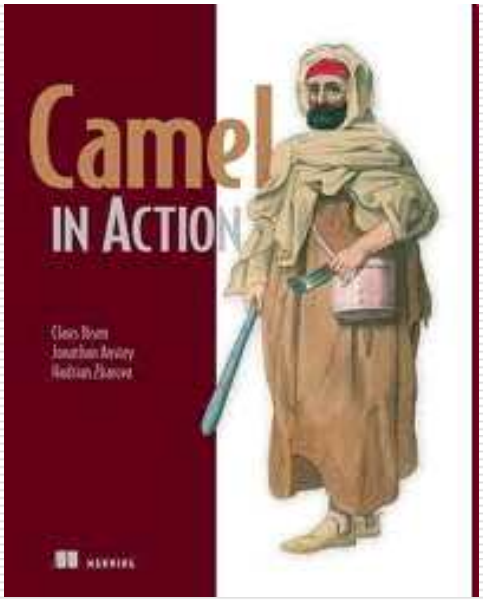
- Apache Camelとは？
- Camel概要
- Camelの使い方
- 最後に

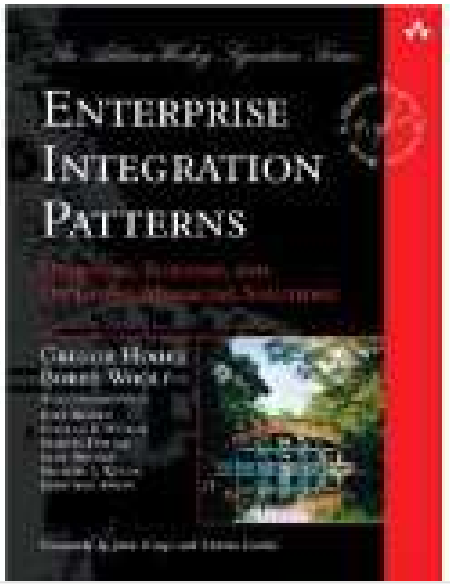
さいごに
～Camelの勉強～

日本Apache Camelユーザ会

Japan Apache Camel Users Group

- 英語の書籍ですが…





さいごに
～ユーザ会を立ち上げました～

- 皆さんにCamelを知って欲しい！
- 日本Apache Camelユーザ会
  http://sourceforge.jp/projects/cameluserjp/
- 主な活動内容
  - Camelのドキュメント翻訳
  - Camelのサンプル実装

  等をやりたいと思っています。

さいごに
～デモブース～

日本Apache Camelユーザ会のデモブースにて、

- Camel＋BPMでEAI/BPMのデモ
- Camel＋RuleでCEPのデモ

が見れます。

お気軽にお立ち寄りください

ご静聴
ありがとうございました