Seasar Conference
2007 Autumn

オープンソース ESB Mule
Mule と Seasar2 の連携プロダクト
S2Mule（仮称）

OGIS International, Inc.

藤倉成太（fujikura@ogis-international.com）

自己紹介

名前：藤倉 成太（ふじくら しげもと）

所属：OGIS International, Inc.

現在は「モデルベース SOA」を推進中

（S2Mule 開発プロジェクトリーダー）

バックグラウンド

UML

CORBA

OODB/XMLDB

Web サービス

BPEL/BPMN

SOA のアプローチ

（SOA の概念や目的の説明は省略）
実施手順について考えてみる:

- トップダウン（全社規模の業務視点）
  - 新規事業
  - ビジネスモデル改革
  - J-SOX
- ミドルアウト（全社規模の情報投資視点）
  - 情報基盤整備
  - 開発プロセス整備
- ボトムアップ（部署毎の業務視点）
  - 機能追加
  - 新規システム開発

ボトムアップアプローチ

- 全社規模の視点も重要だが判断が難しい
- 一方で今やらなければならないことは山積
- 短期的に効果が見えることが重要

ボトムアップによる SOA アプローチが有効

ボトムアップアプローチでは
現在の開発案件単体で効果を発揮しながら
徐々に SOA への移行準備をする

OSS を活用したアプローチ

- 実装するにはミドルウェアが必要
- しかし SOA 関連製品は高価なものが多い
- 導入コストがかかる技術は採用しにくい
- 導入するには中長期的な判断が求められる

OSS を上手く活用すれば,
ボトムアップアプローチも実現可能

Seasar2 + Mule

S2Mule（仮称）とは？

S2Mule とは Seasar2 アプリケーションと外部リソースの連携をサポートするためのプロダクトである.

分散処理をサポートする既存の S2 関連プロダクト:

- S2Dao(JDBC)
- S2Remoting(SOAP/RMI)
- S2JCA(JMS)

これら以外でも“連携”の必要がある

S2Mule

- 標準技術による分散処理は SOA への第一歩
- 次の一歩は，より柔軟で運用しやすいプラットフォームを利用すること
- SOA では ESB がその役割を担っている

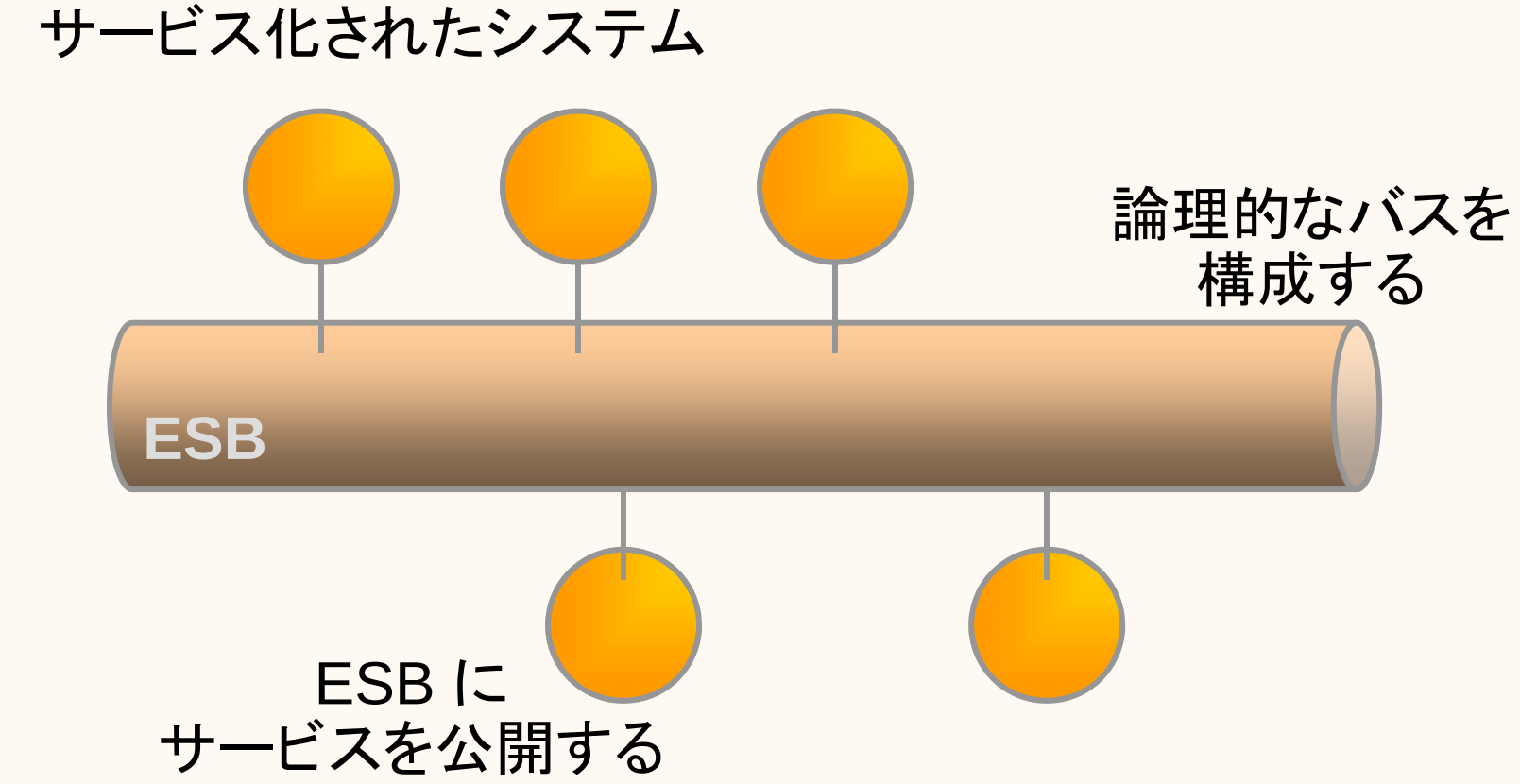
S2Mule

Mule ESB を利用しやすくする

ESB とは？

- ESB: Enterprise Service Bus
- SOA の基盤技術として注目されている
- システム連携のためのバス
- 標準技術を基本としている
- ただし ESB 自体には標準がない
  - JBI や SCA/SDO がそうなる可能性はあるが

ESB 適用イメージ



ESB の主な機能

ESB で統一的に管理・制御できるもの:

- セキュリティ
- 信頼性
- サービス管理
- メッセージ・ルーティング
- 複数トランスポートのサポート
- 同期／非同期のサポート
- データ変換

ESB の提供ベンダー（一部）

ベンダー	製品
IBM	WebSphere ESB
BEA	AquaLogic Service Bus
Oracle	Oracle ESB
TIBCO	TIBCO EMS
Sonic	Sonic ESB
富士通	Interstage Service Integrator
JBoss (OSS)	JBoss ESB
Apache (OSS)	ServiceMix
Mule (OSS)	Mule

ESB のメリット

- サービス間の接続性の確保
  - 多種多様な実装技術で構築されるサービス群の相互接続性を確保する.
  - 信頼性や安全性(セキュリティ)を単一の方法で設計，管理できる.
- ルーティング処理の分離
  - メッセージを他のサービスに送るためのロジックを分離する.
  - ルーティング機能を使ってビジネスロジックを構築しないよう注意する.
- サービス間の依存度を低減
  - メッセージ送信先の位置情報やメッセージ形式への依存度を低減(分離)する.

ESB の注意点

- 全体の通信基盤として一度に導入しようとすると費用と時間がかかってしまう.
- 導入効果は時間が経たなければ明確にならない.
- 標準仕様がないので，個別製品の機能に依存するとシステム連携全体がロックされてしまう.

適切な製品を適切なタイミングで導入すること

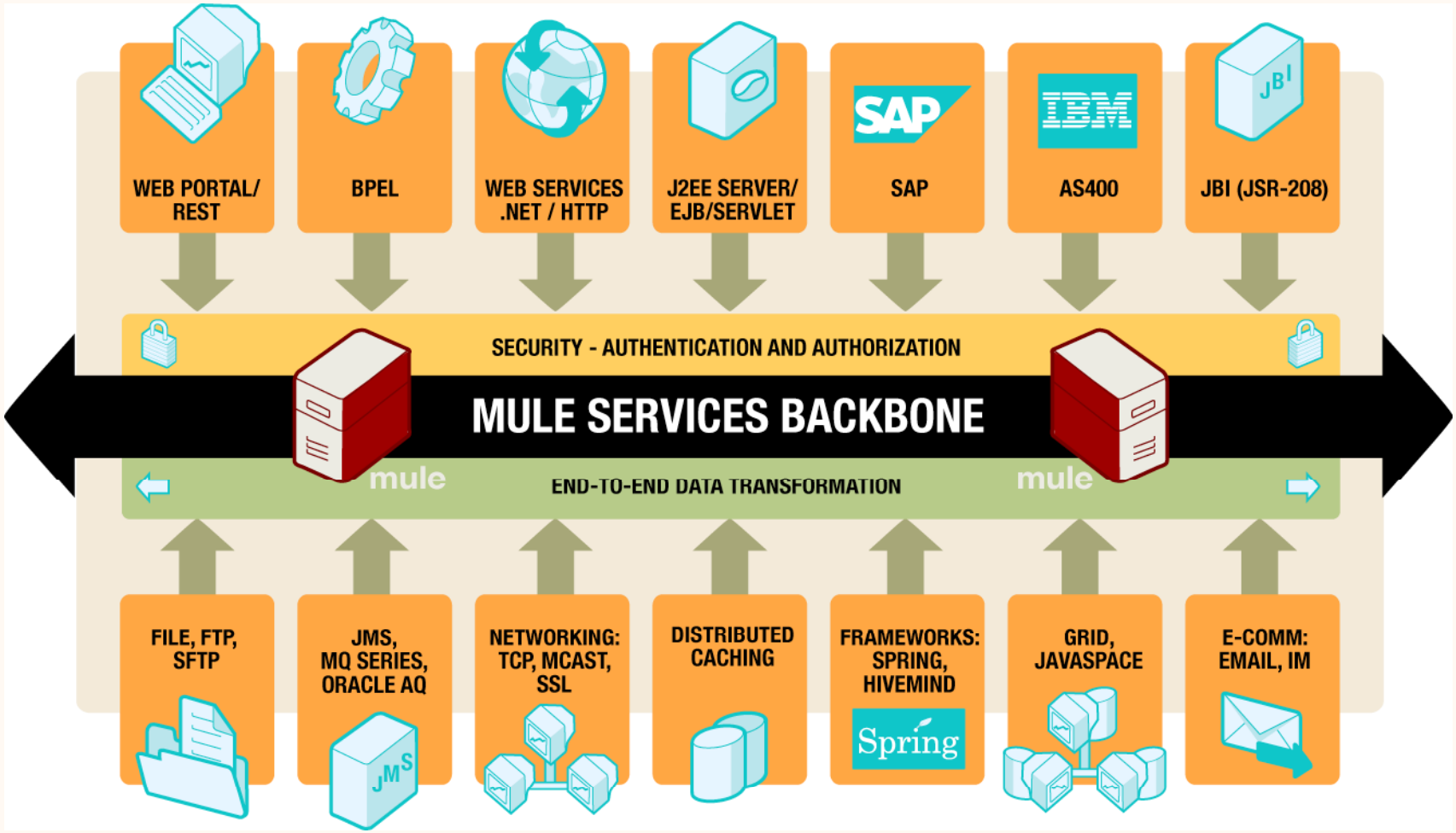
Mule とは？

[http://mule.mulesource.org/]

- オープンソースの ESB 実装
- アプリケーション統合とメッセージングプラットフォーム
- 2003 年に Ross Mason 氏によって開始
- バージョン 1.0 以来:
  - ダウンロード 500,000 回以上
  - 1000 人以上の開発者
  - 120 件以上の適用事例
- 最新バージョンは Mule 1.4
- Mule 2.0 は今年度中にリリース予定

Mule Overview

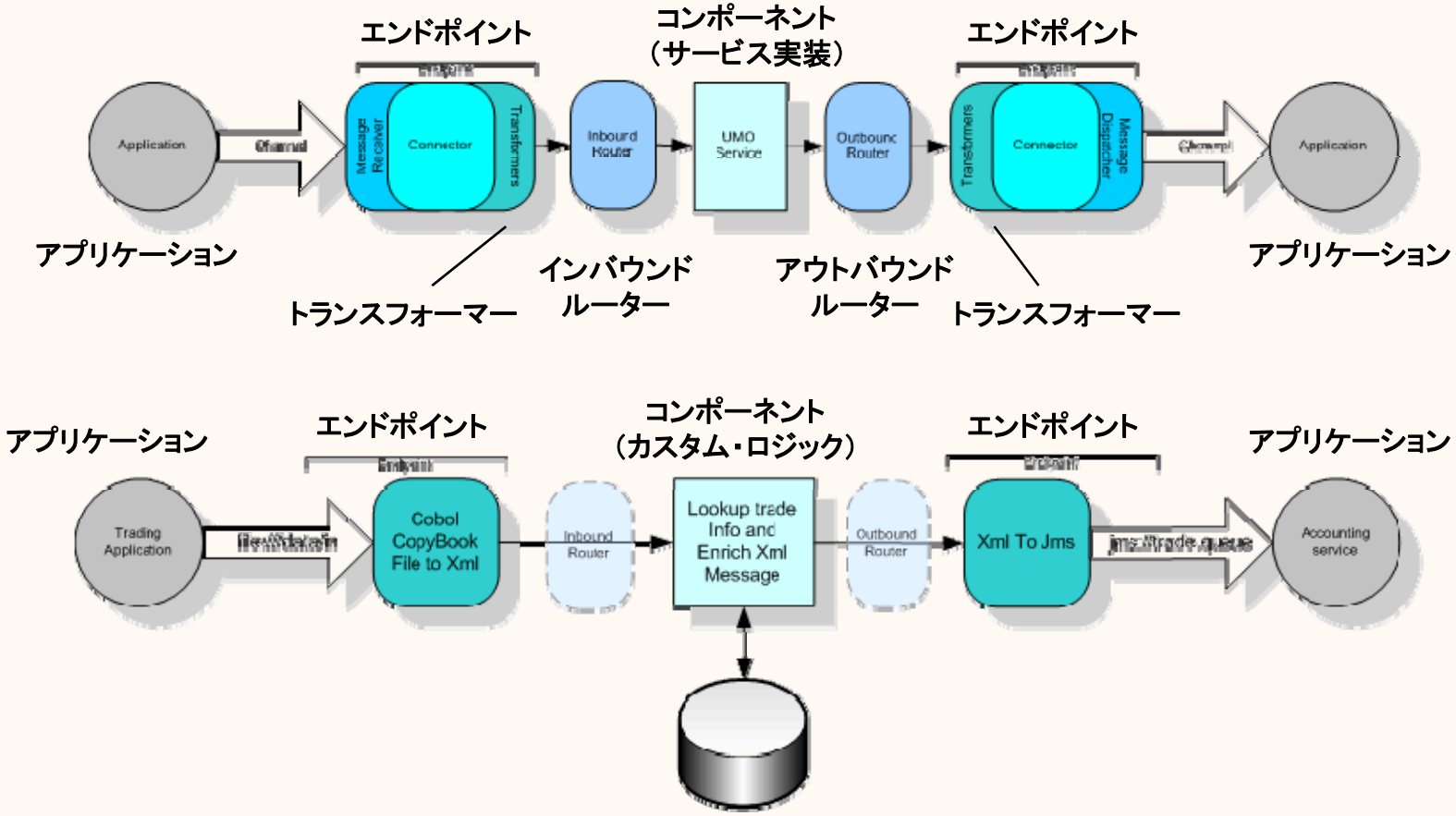
http://www.mulesource.com/ より引用



Mule の主な機能リスト

- メッセージ・ルーティング
- データ変換
- 多数のトランスポートをサポート
- 同期／非同期のサポート
- セキュリティ
- 分散トランザクション

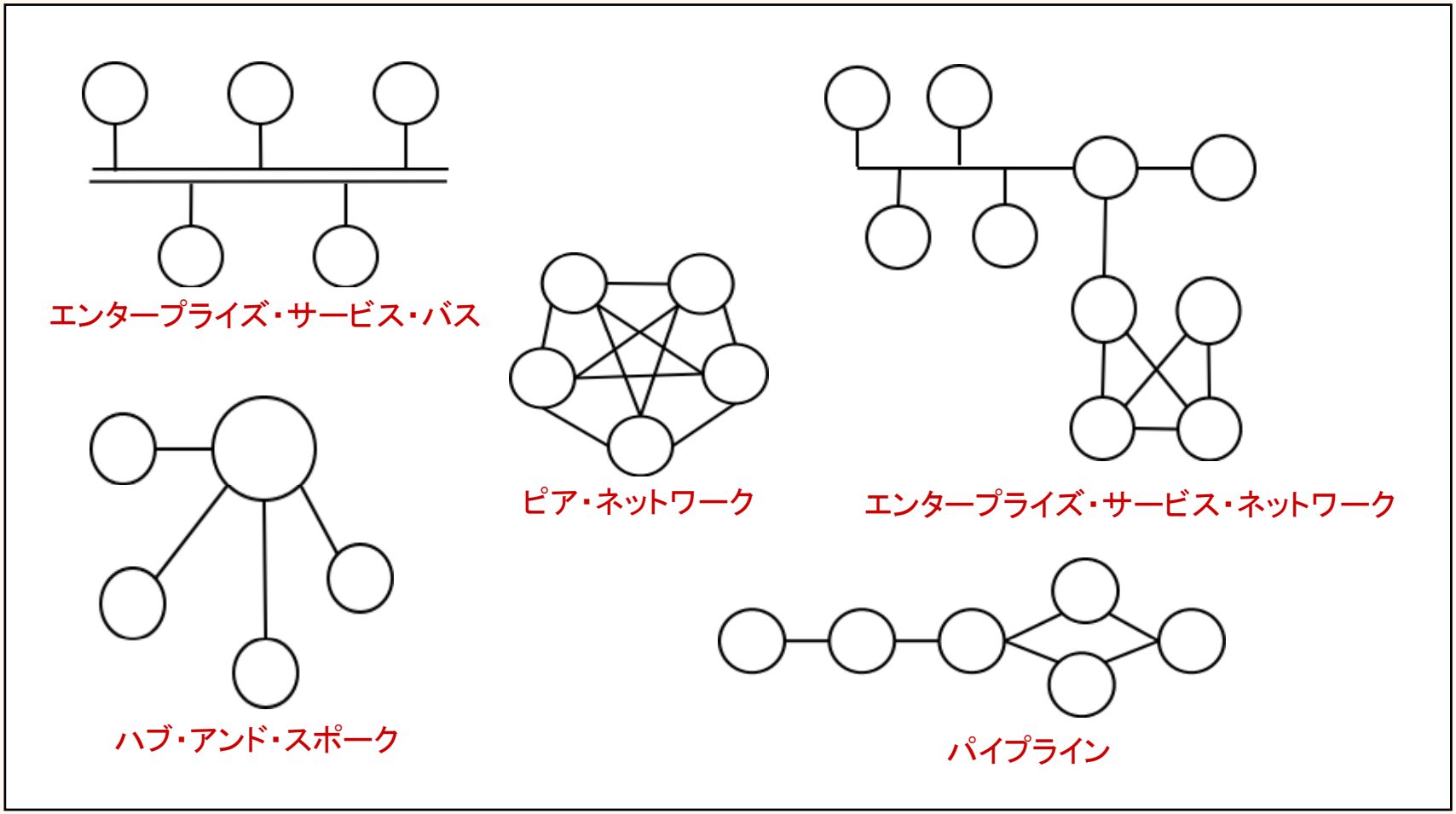
Mule のアーキテクチャ



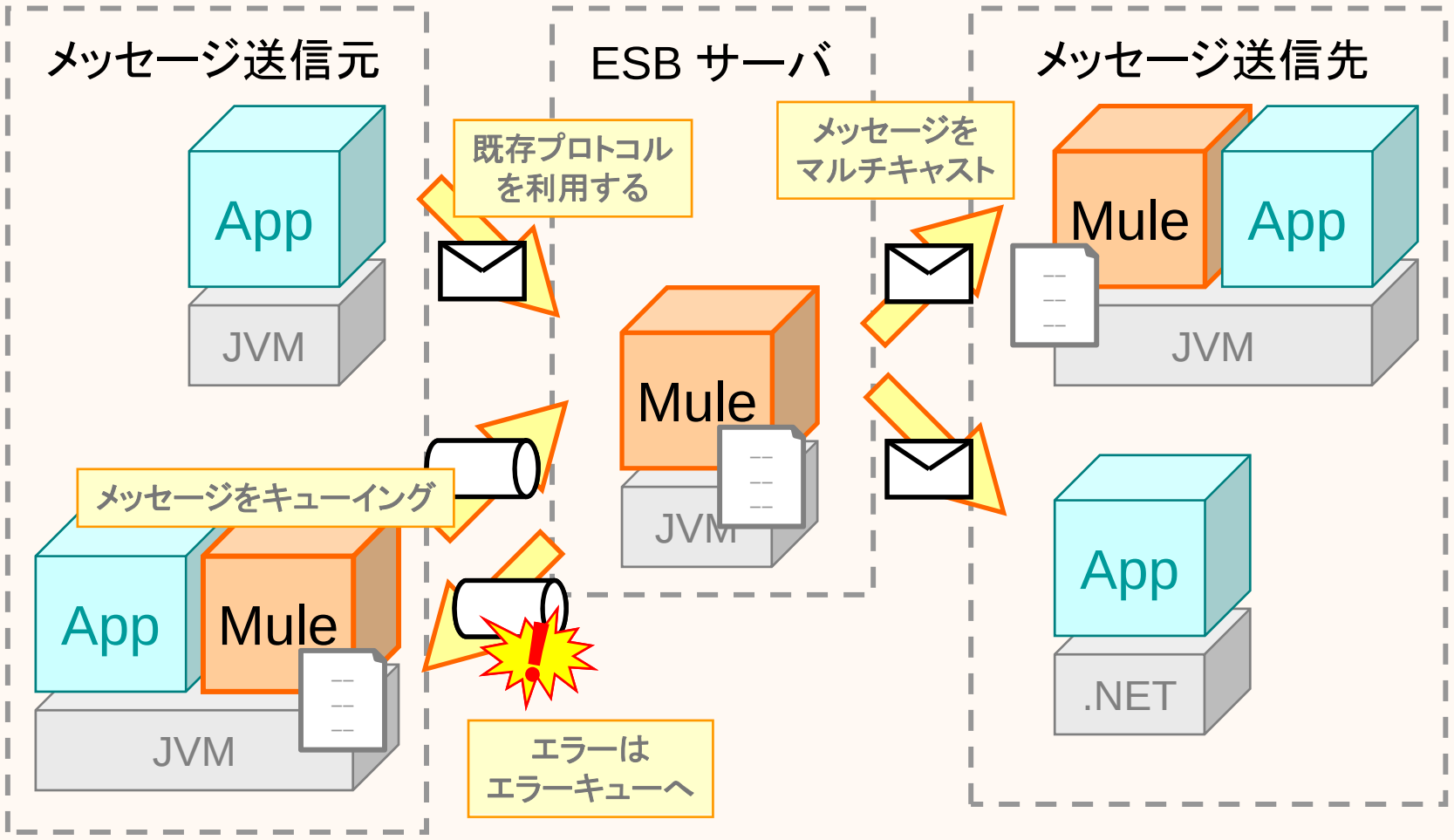
Mule のトランスポート（一部抜粋）

EJB	File	FTP
HTTP	IMAP	JDBC
JMS	POP3	RMI
SMTP	SOAP	SSL(TLS)
TCP	UDP	VM

Mule で実現できるトポロジの数々



Mule 使用例イメージ



Mule の基本的な設定内容

- Inbound
  - エンドポイント（メッセージを受信するための URI）
  - ルーター（提供されているクラスか独自の実装クラス）
  - トランスフォーマー（提供されているクラスか独自の実装クラス）
- コンポーネント
  - 実装クラス（提供されているクラスか独自の実装クラス）
  - その他プロパティ
- Outbound
  - エンドポイント（メッセージを送信する相手先の URI）
  - ルーター（提供されているクラスか独自の実装クラス）
  - トランスフォーマー（提供されているクラスか独自の実装クラス）

実装するもの

Mule を使用する場合にアプリケーション側で
実装しなければならないものは少ない.

- メッセージ送信元で，Mule にメッセージを渡すコード
- メッセージ送信先で，Mule を介して受信されたメッセージをアプリケーションに渡すコード

提供されている以外の特殊な処理をしたい場合

- ルーター（ルーティングルール）
- トランスフォーマー（メッセージ変換）
- コンポーネント

Mule まとめ

Mule は:

- 様々なトランスポートを使える
- メッセージ・ルーティングが設定できる
- メッセージ変換が設定できる
- セキュリティの設定ができる
- 分散トランザクションの設定ができる

Mule はシステムとシステムをつなぐ

S2Mule（仮称）とは？

S2Mule とは S2 アプリケーションから
Mule の持つ機能を簡単に利用できるようにするプロダクト
（2007.11 現在はフェーズ 1 が終了）

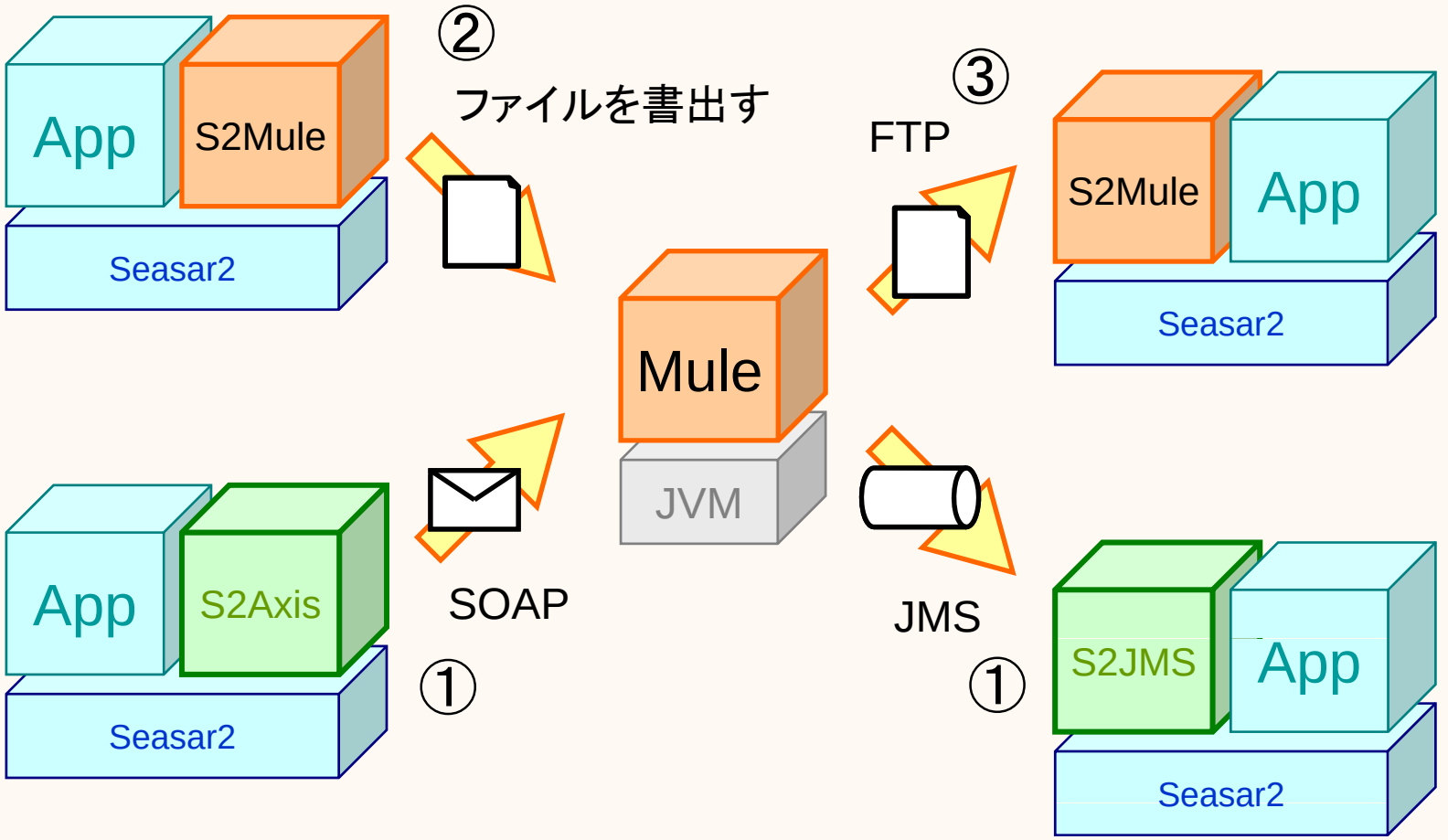
- S2Mule Client 機能
  - S2 アプリケーションからメッセージを送信する機能を提供
- S2Mule Server 機能
  - 外部から受信したメッセージを S2 アプリケーションで受け取る機能
- 特徴
  - dicon ファイルで設定可能
  - Mule がサポートする多数のトランスポートを利用可能
  - トランスフォーマーが利用可能
  - ルーターが利用可能

Mule, S2Mule の利用シナリオ

1. 既存の分散処理コンポーネント(S2Axis/S2JMS など)を使ってメッセージを送受信する.
2. S2Mule Client を使ってメッセージを送信する.
3. S2Mule Server を使ってメッセージを受信する.

S2Mule 使用例イメージ

※ 番号は前スライドのシナリオ



S2Mule Client API 案

- インターセプタを使って（S2Remoting のように），オブジェクト呼び出しを暗黙的にメッセージ送信に利用する．この方法はイベント指向型（EDA: Event Driven Architecture）のシステム連携には不向き．
- 特別なAPI を使って，明示的にメッセージ送信用コードを記述する．

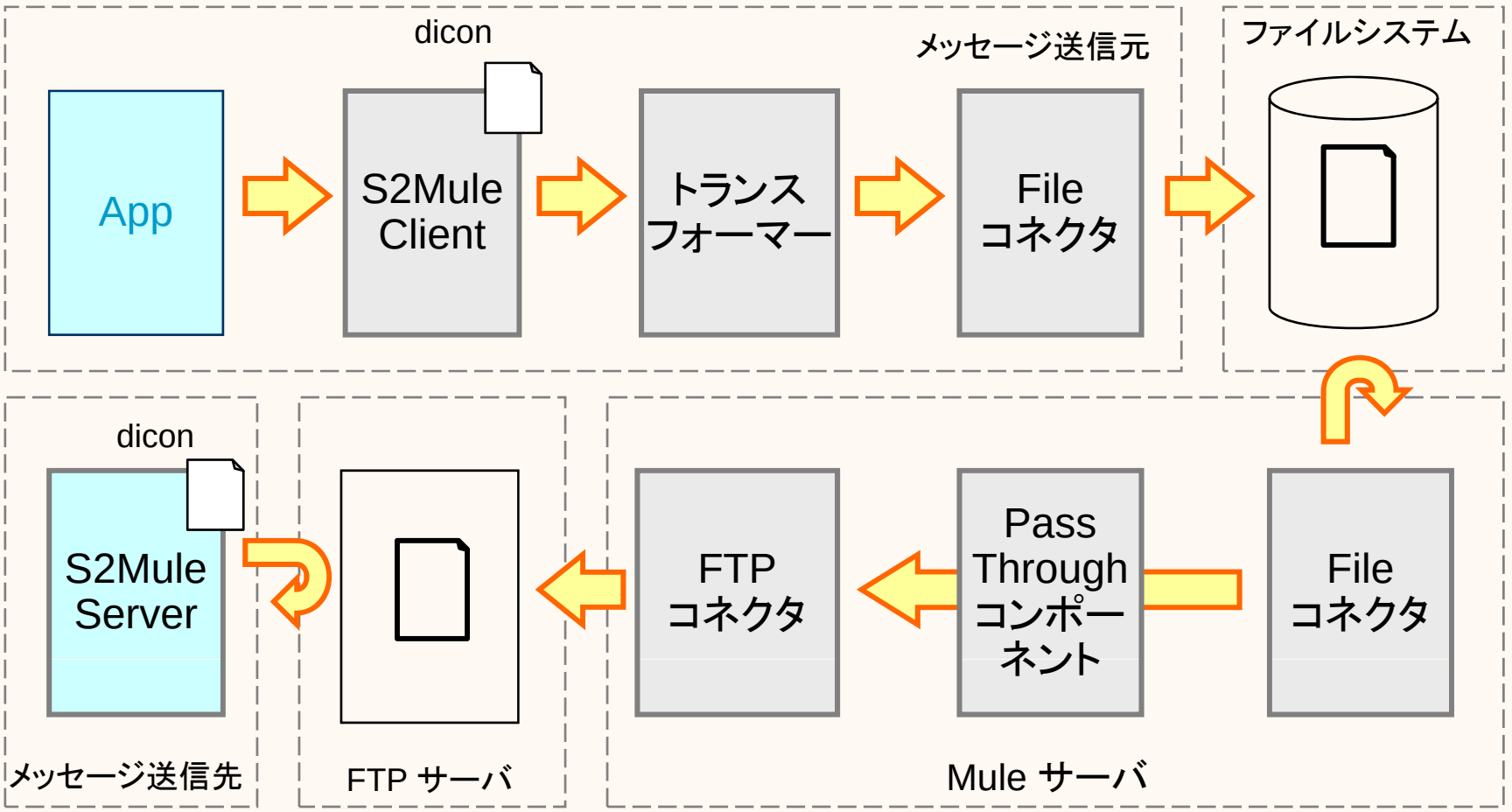
```
/* 非同期メッセージの送信 */
void dispatch(Object payload);
/* プロパティ付き非同期メッセージの送信 */
void dispatch(Object payload, Map messageProperties);
/* 同期メッセージの送信 */
UMOMessage send(Object payload);
/* プロパティ付き同期メッセージの送信 */
UMOMessage send(Object payload, Map messageProperties);
```

S2Mule Server dicon

```
<!-- mule-config.dicon -->
<component
        name="MyEchoUMO"
        class="jp.co.ogis_ri.s2mule.server.object.S2MuleConfiguration"
        autoBinding="none" >
        <property name="inboundEndpointUri">
                "stdio://System.in?promptMessage=Please In put&amp;amp;method=echoEcho“
        </property>
        <property name="outboundEndpointUri">"stdio://System.out"</property>
        <property name="umoImpl">MyEchoUMOImpl</property>
        <initMethod name="initialize" />
</component>

<!-- app.dicon -->
<component name="MyEchoUMOImpl" class="jp.co.ogis_ri.s2mule.test.MyEcho" />
```

S2Mule を使った処理例



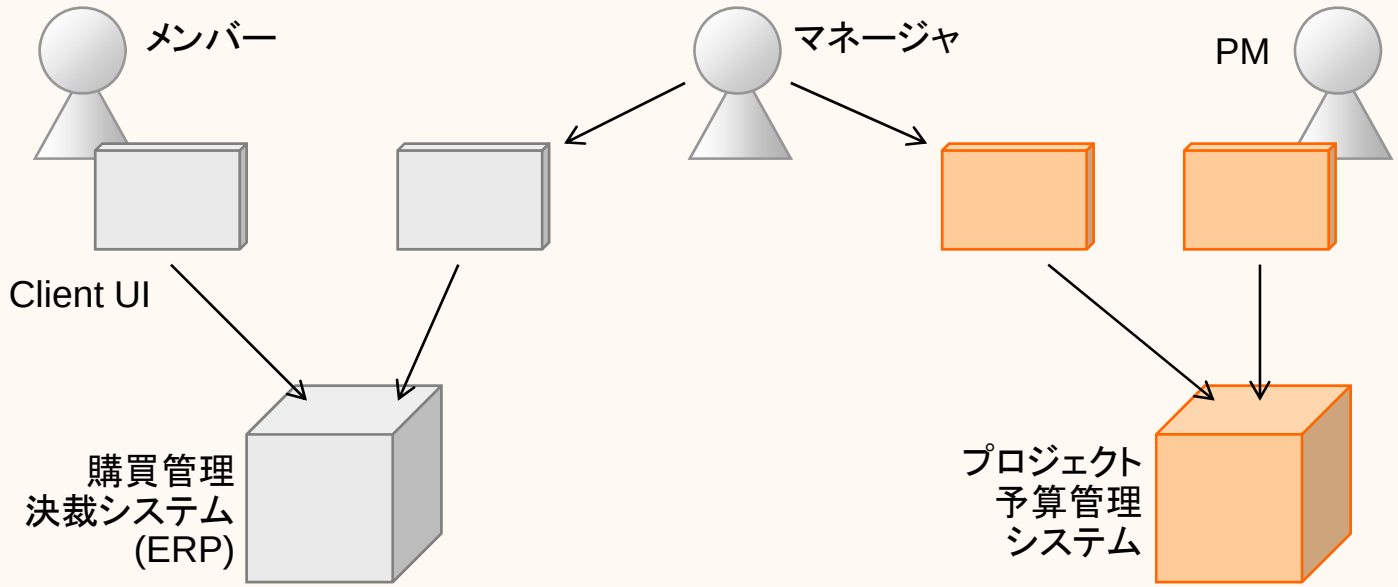
デモシナリオ

「メンバーがあげた購買申請を承認する」

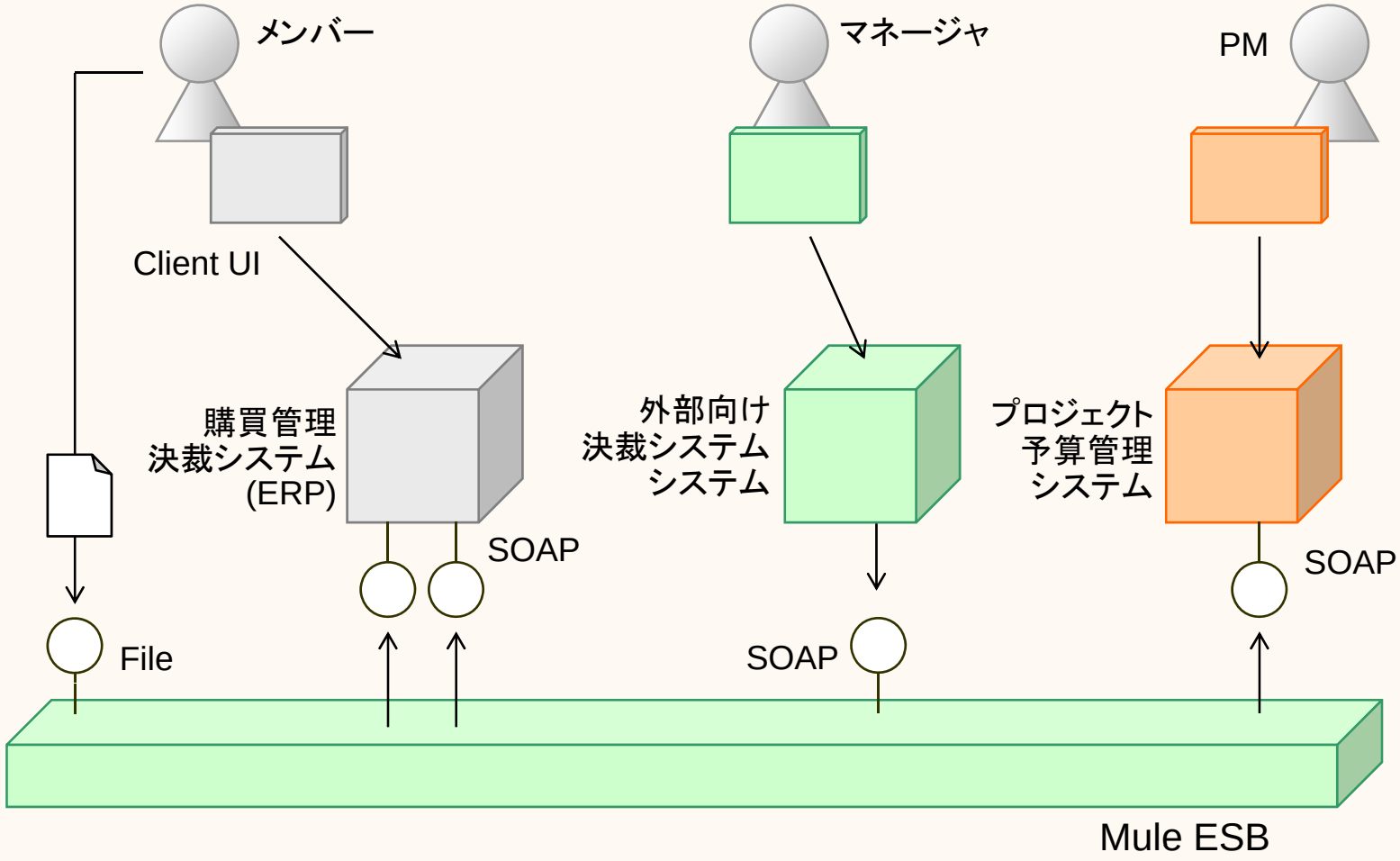
シナリオ:

- 購買管理用 ERP とプロジェクト管理システムがある.
- PM はプロジェクト管理システムを使ってプロジェクトを管理する.
- 所属のマネージャは購買管理用 ERP を使って部署の費用を管理する.
- ERP による資産管理とプロジェクト管理システムによる管理には情報の鮮度や粒度などに違いがある. まずは UI の変更によって使いやすいシステムが欲しい.
- ERP の購買管理はほとんど変更されないが, プロジェクト管理の仕組みは制度として変更しやすい. この変更タイミングのギャップを SOA で解決していきたい.

デモシステム構成 As-Is



デモシステム構成 To-Be



S2Mule の使いどころ

- S2Dao，S2Remoting，S2JCA を補完する.
  - 既に実績のあるプロダクト（RDBMS/SOAP/JMS など）を利用する.
  - それ以外のトランスポートは S2Mule で.
- まずは FTP や SMTP から使ってみる.
  - まずはありものの機能を便利に使うくらいで始める.

将来の SOA 時代に備えておく

SOA は小さく始めて大きく育てるのが良い

その他の開発内容

- Mule&Terracotta の組合せでの使用
  - Terracotta は Java のオブジェクトを複数 VM 間で透過的に共有する仕組みを提供するオープンソース製品
  - Mule&Terracotta の組合せで Mule の可用性向上を実現する
- 認証サービス等の共通サービス実装
  - ビジネス層よりは一段低レベルな層に位置づけられる共通機能サービスの参照実装