FUJITSU Software INTARFRM Enterprise Edition ご紹介

２０２６年７月
富士通株式会社



© 2024 Fujitsu Limited

目次

FUJITSU

- １．ICT業界における一般的な課題と提案
- ２．フロントエンドの多様なニーズに対応
  （INTARFRM RIA Java Ajax）
- ３．保守性・拡張性の高いバックエンド開発を支援
  （INTARFRM Distributed Component Java）

  ※Distributed Component Javaは、富士通受託開発を前提とする製品です。
- ４．まとめ

【添付資料】
・開発の流れ
・機能一覧
・お試し版
・動作環境

本資料内では特別な記述が無い限り、INTARFRM Enterprise Editionを「INTARFRM」または「INTARFRM Ent. Edi.」と表記しています。

© 2024 Fujitsu Limited

1.ICT業界における一般的な課題と提案

© 2024 Fujitsu Limited

こんな課題でお悩みではありませんか

FUJITSU



経営層の課題

システム改修のたびにアプリケーション
開発費用・保守費用が高くなっていく

ビジネス(サービス)変化のスピードに
システムの整備が追いつかない

情報システム部門の課題

新たなフロントエンドの技術に
追随困難 (最新デバイス追加、利便性追求等…)

機能追加／変更のたびに資産が複雑化し、
保守性・拡張性が低下していく

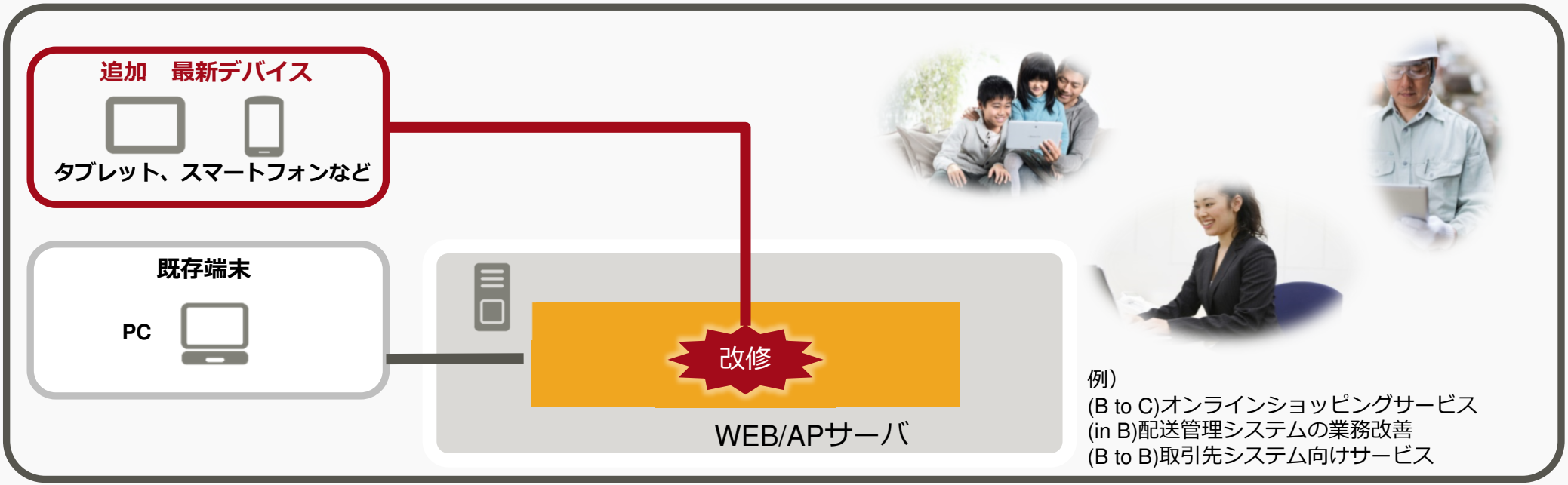
© 2024 Fujitsu Limited

例えば、最新デバイス追加対応

FUJITSU

課題　タブレットやスマートフォンなどの追加に伴い、システムの改修費用が多くかかる。

原因　資産の改修を繰り返していくうちに、システムのつくりが複雑になり、
新しい要件の追加が困難になっているため。



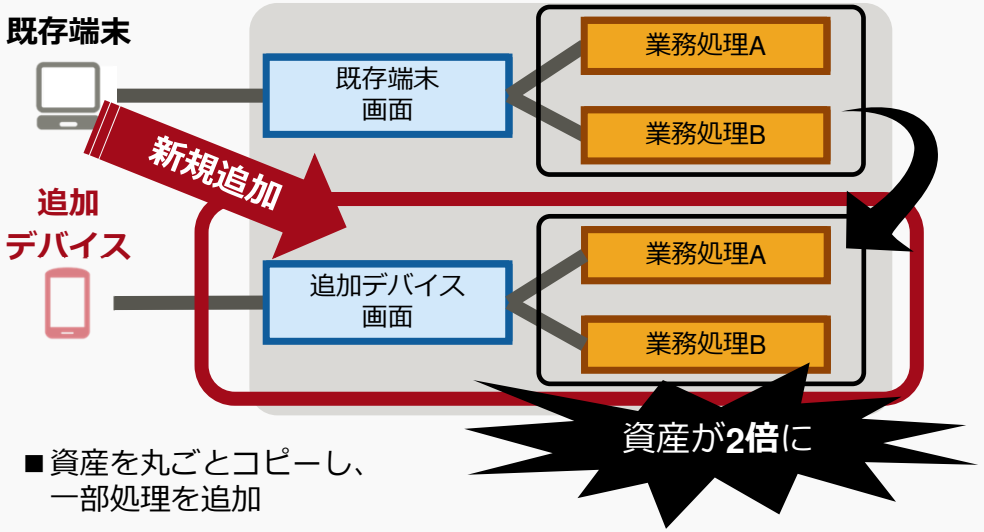
© 2024 Fujitsu Limited

システムのつくりが複雑になる原因と対策(例)

FUJITSU

● **原因**

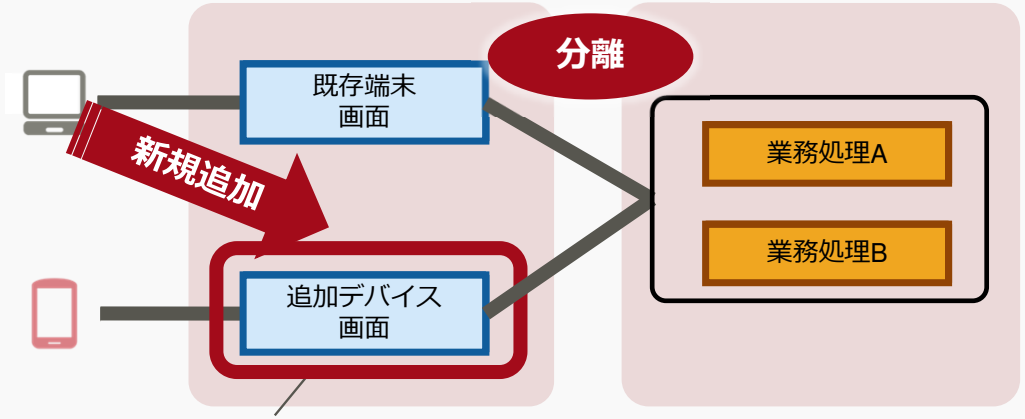
改修の際、既存資産を丸ごとコピーして流用するため。
資産が倍増し類似処理が増加することにより、
システムのつくりが複雑化する。
（例えば不要なロジックが多くなるなど）



■資産を丸ごとコピーし、
一部処理を追加

● **対策**

業務処理と画面処理を分離する。
改修の際、画面処理の追加のみで済むため
資産の増加を最小限に抑える。



■追加開発範囲が狭い
画面追加時、業務処理の追加は不要。
どちらの画面からも同じ業務処理を利用可能。

画面処理と業務処理を分離することにより、資産の増加を最小限に抑えることができます。

© 2024 Fujitsu Limited

フロントエンドとバックエンドの要件の違い

画面処理(フロントエンド)と業務処理(バックエンド)ではシステム要件が異なります。

■フロントエンドとバックエンドのシステム要件



フロントエンドの要件

- ■ 技術の進歩が速い
- ■ 利便性向上のニーズが多い
- ■ 流行などから改修や作り直しが多い



更改や全面作り変えの頻度が高い



バックエンドの要件

- ■ 基本技術は安定している
- ■ フロント更改に影響されたくない
- ■ 修正範囲を最小限に抑えたい



高い保守性を保ち長く維持し続けたい

画面処理と業務処理を分離した構造にすることで、
改修の影響範囲を最小限に抑えた開発、保守が可能です。

© 2024 Fujitsu Limited

INTARFRM Ent. Edi.の提案

INTARFRM Ent. Edi.は更改頻度が異なるフロントエンドとバックエンドを分離した形で開発が行えるので、
フロントエンドの頻繁な更改や全面作り変えに強く、高い保守性を保ち長く維持できます。
フロントエンド、バックエンドそれぞれのアプリケーション開発に最適な製品をご用意しています。



フロントエンド

フロントエンドに特化した開発向け製品

Fujitsu Software INTARFRM
RIA Java Ajax

画面部品やフレームワーク機能を提供し、
最新クライアント技術（最新デバイス追加、
利便性追求等…)に追随します。

頻繁な更改や全面作り変えに強い



バックエンド

バックエンドに特化した開発向け製品

Fujitsu Software INTARFRM
Distributed Component Java

業務機能を独立性の高い部品群として開発できる
環境を提供し、保守、拡張しやすいつくりにします。
また、Webサービスにも対応しています。

高い保守性を保ち長く維持できる

※Distributed Component Java（以降DC Javaと表記）

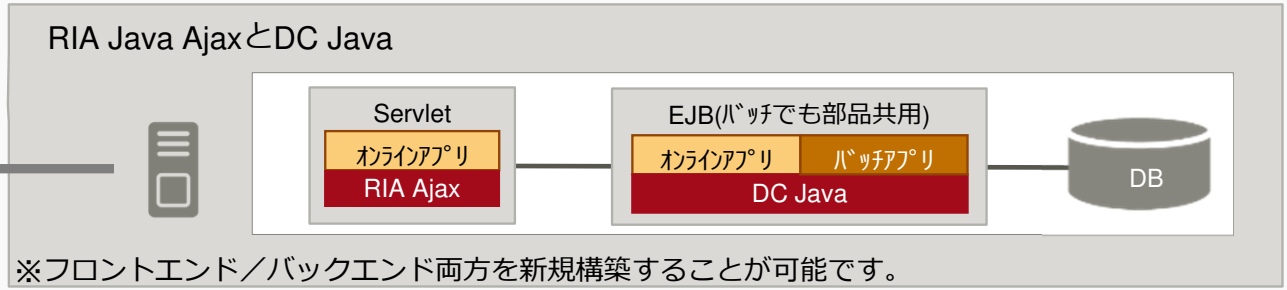
© 2024 Fujitsu Limited

【ご参考】 要件とシステム構成例

■新サービスの構築



例）オンラインショッピングサービス(B to C)

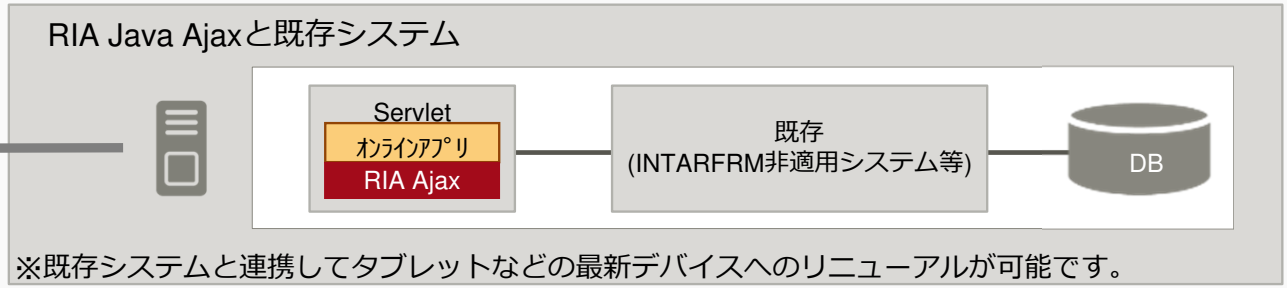


※フロントエンド／バックエンド両方を新規構築することが可能です。

■最新デバイスへのリニューアル



例) 配送管理システムの業務改善(in B)

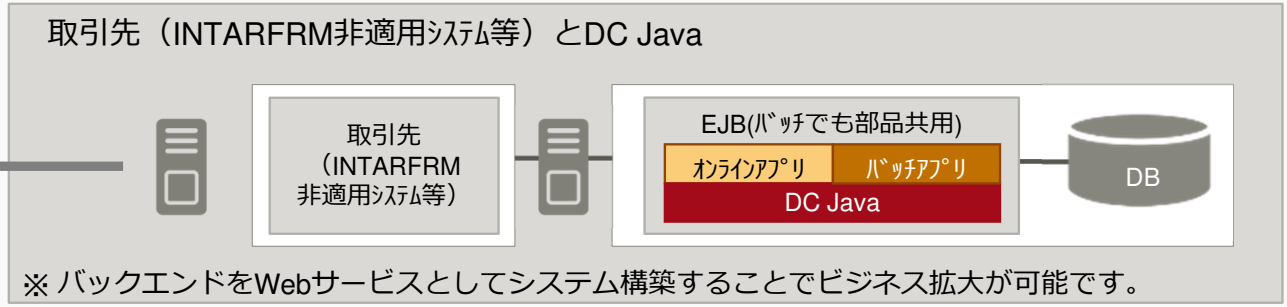


※既存システムと連携してタブレットなどの最新デバイスへのリニューアルが可能です。

■新Webサービス公開



例）取引先システム向けサービス(B to B)



※ バックエンドをWebサービスとしてシステム構築することでビジネス拡大が可能です。

© 2024 Fujitsu Limited

INTARFRM適用実績

FUJITSU

< 2015-2025年度販売・適用実績より >

■適用プロジェクト数　　390

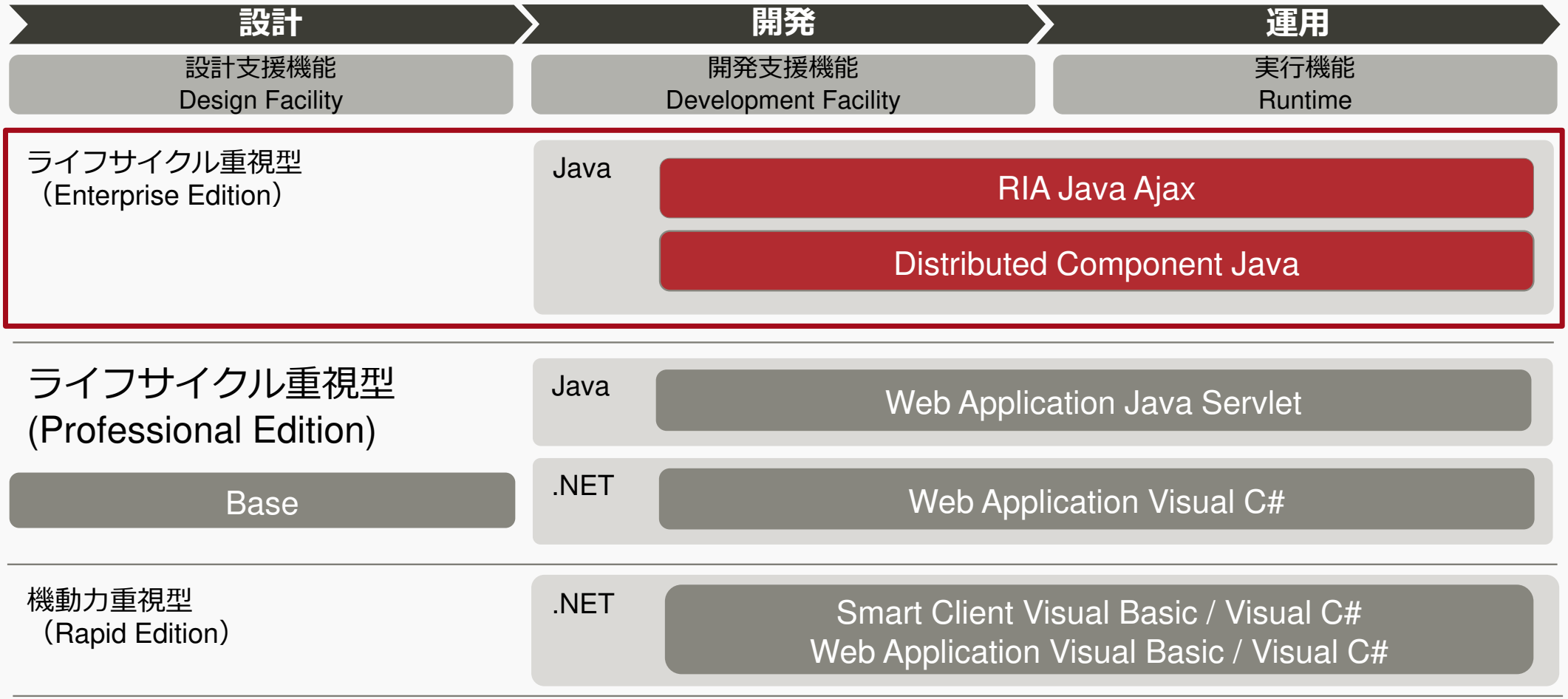
■パッケージ適用例

- FBC-eX （金融営業店向けパッケージ）
- iLisfiera（図書館パッケージ）
- K-12校務支援（小・中・高ソリューション）
- E3CIS（電力小売事業者向け顧客管理・料金計算パッケージ）

SIプロジェクトにおいても、官公庁や産業・流通など多くの業種で導入いただいております。

© 2024 Fujitsu Limited

【ご参考】INTARFRM全体の製品体系



© 2024 Fujitsu Limited

2.フロントエンドの多様なニーズに対応（INTARFRM RIA Java Ajax）

© 2024 Fujitsu Limited

RIA Java Ajaxとは

UXの高い**フロントエンド開発**
を支援するためのフレームワーク

FUJITSU Software INTARFRM
RIA Java Ajax

© 2024 Fujitsu Limited

フロントエンドシステムの動向

一般利用者向けシステムだけでなく、企業向けシステムにおいても

ユーザエクスペリエンス(UX) が求められる時代になりました。

▶ ニーズの変化とシステム形態の変遷

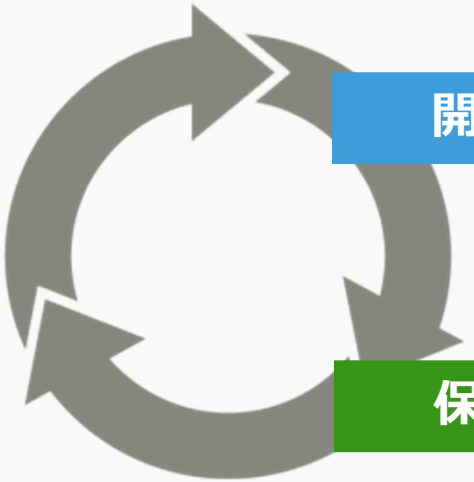
ニーズ	操作性重視	運用性重視 操作性は犠牲に	操作性・運用性 ともに重視	**利用者の総合的な価値体験を 重視したさまざまなニーズ**
キーワード			ユーザビリティ	**ユーザエクスペリエンス(UX)**
システム形態 (UIイメージ)	**クラサバ** 	**Webシステム** 	**RIA　(リッチクライアント)** 	

© 2024 Fujitsu Limited

UX開発の課題

ユーザエクスペリエンス(UX)を意識した
アプリケーションを実現しようとすると・・・

操作性・表現力を高めるための作り込みが必要



開発

- 開発量が増え、稼働までの時間がかかる
- 作り込みをするために高度なスキルが必要
- プログラムが煩雑になり品質が悪くなる

保守

- プログラムが煩雑になり保守の手間がかかる
- システム保守費用が高くなる
- 画面や機能の変更・追加を簡単にできない

© 2024 Fujitsu Limited

INTARFRM RIA Java Ajaxの提案

FUJITSU

「高い操作性・表現力」を実現します。

- 直感的に操作したい
- 表現力の高い・見た目を良くしたい
- 操作中のストレスを軽減したい

「スマート開発」を実現します。

- 費用対効果の高いシステムを導入したい
- ビジネススピードに合わせて、短期間で導入をしたい
- 高品質なシステムを使いたい

「らくらく保守」を実現します。

- 保守の手間をなるべく省きたい
- システム保守費用を抑えたい
- 画面や機能の変更・追加を簡単にしたい

© 2024 Fujitsu Limited

FUJITSU

高い操作性・表現力

「スマート」 開発

「らくらく」 保守

- 標準画面部品の活用
- jQuery一般公開プラグインが利用可能
- 入力補助機能
- 画面の部分更新／非同期通信
- 様々なデバイス・ブラウザで動作
- マルチデバイス開発時の考え方

© 2024 Fujitsu Limited

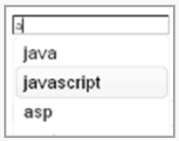
標準画面部品の活用(1/2)

■ ユーザエクスペリエンスの実現に必要な画面部品が利用可能（RIA Java Ajaxで標準提供）

アコーディオン

タブ

オートコンプリート



プログレスバー



アイコン



●画面例



※標準画面部品の詳細については、画面部品一覧のスライドで説明します。

表現力豊かな画面を実現できます。直感的な操作が可能になります。

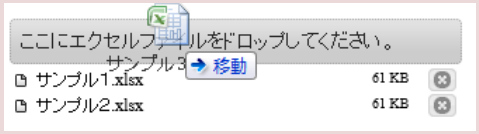
© 2024 Fujitsu Limited

標準画面部品の活用(2/2)

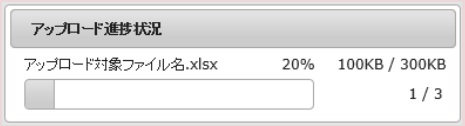
■ ユーザエクスペリエンスの実現に必要な画面部品が利用可能（RIA Java Ajaxで標準提供）

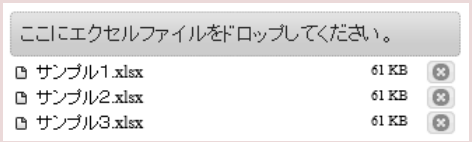
ファイルアップロード／ダウンロード

ドラッグ&ドロップでファイルを選択



プログレスバーで進捗を確認





タグイテレータ

●タグイテレータを利用したシステムの例



●タグイテレータを利用した明細表の例



表現力豊かな画面を実現できます。直感的な操作が可能になります。

© 2024 Fujitsu Limited

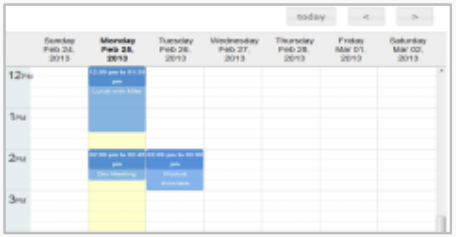
jQuery一般公開プラグインが利用可能

FUJITSU

■ jQuery一般公開プラグインとの連携機能を提供
■ 世界中に公開されているjQuery一般公開プラグインを利用することで、さらにUXの高い画面が実現可能

●jQuery一般公開プラグインの例 (※ jQuery一般公開プラグインのサポートについては各プラグインの提供元にご確認ください。)

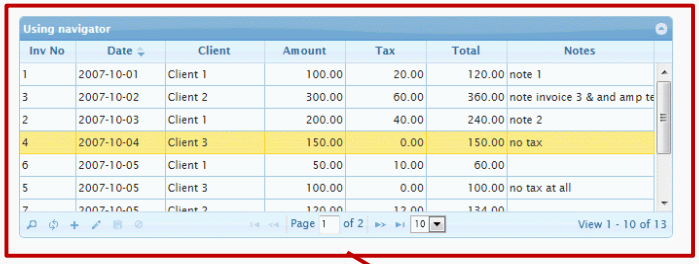
スケジュール



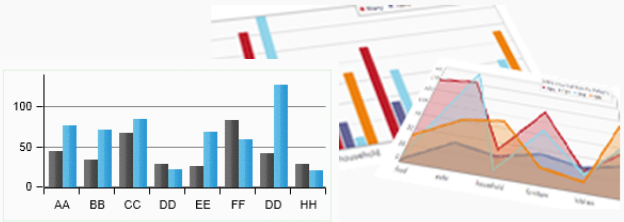
カレンダー



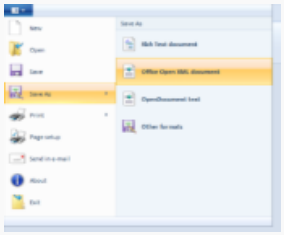
グリッド



グラフ



メニュー



イメージ



次ページに
事例を紹介しています

さらなる操作性・表現力の向上が期待できます。

© 2024 Fujitsu Limited

＜事例＞jQuery一般公開プラグインの組込例

■ jQuery一般公開プラグインでUXの高い画面を実現した事例です。
開発に手間のかかるJavaScriptの開発量を抑え、Javaによる開発が可能です。
これにより、開発効率の低下、品質のばらつき、ソースの可読性低下による保守性の低下等のリスクを軽減します。

●機能例



高品質で最先端の部品を利用することで、よりUXの高い画面が実現できます。

© 2024 Fujitsu Limited

入力補助機能

- 入力操作を助けるための入力補助機能を提供
- 入力補助機能を活用することにより、入力ミスの削減、高速入力が実現

●画面例





IME自動切替



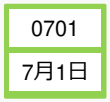
入力制限



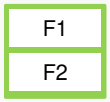
入力チェック



フォーカス自動遷移



フォーマット



ファンクションキー割当

タイピング数が減り、業務効率向上に貢献します。

© 2024 Fujitsu Limited

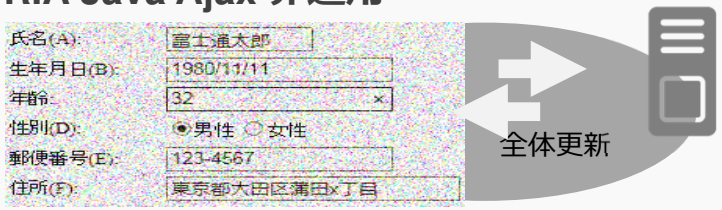
画面の部分更新／非同期通信

FUJITSU

画面の部分更新

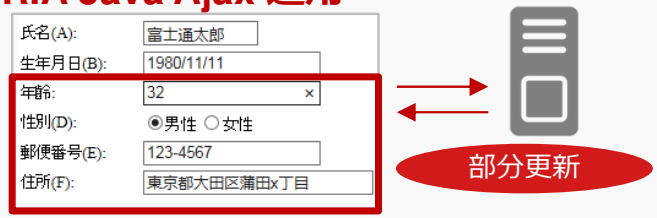
画面を更新する際、必要なコンテンツのみを更新するので**画面更新が速い**。

RIA Java Ajax 非適用



遅い　　チラつきあり

RIA Java Ajax 適用



速い　　チラツキなし

非同期通信

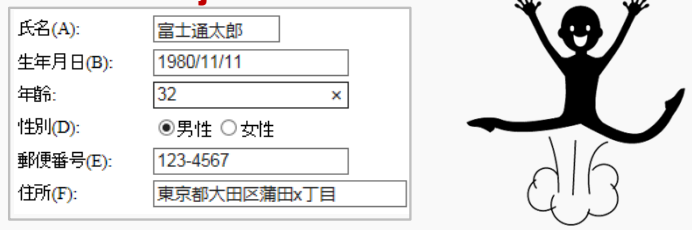
サーバに処理を依頼し、**結果を待たずにクライアントの操作が可能**。

RIA Java Ajax 非適用



待ち時間が多い

RIA Java Ajax 適用



待ち時間が少ない

画面のチラツキ・待ち時間が軽減され、操作中のストレスがなくなります。

© 2024 Fujitsu Limited

様々なデバイス・ブラウザで動作

■様々なデバイスやブラウザで動作、利用シーンが拡大

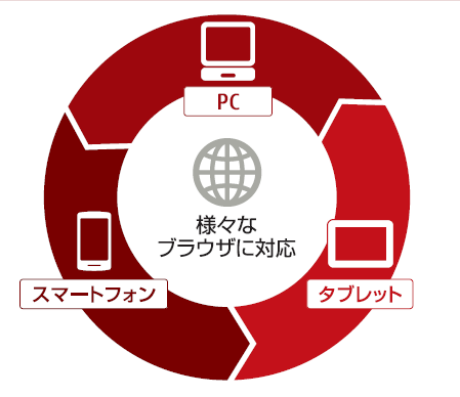
●動作実績

■対応デバイス

PC
タブレット
スマートフォン
(Windows, Android, iOS)

■対応ブラウザ

Microsoft Edge
Google Chrome
Mobile Safari



●活用例：　保険営業







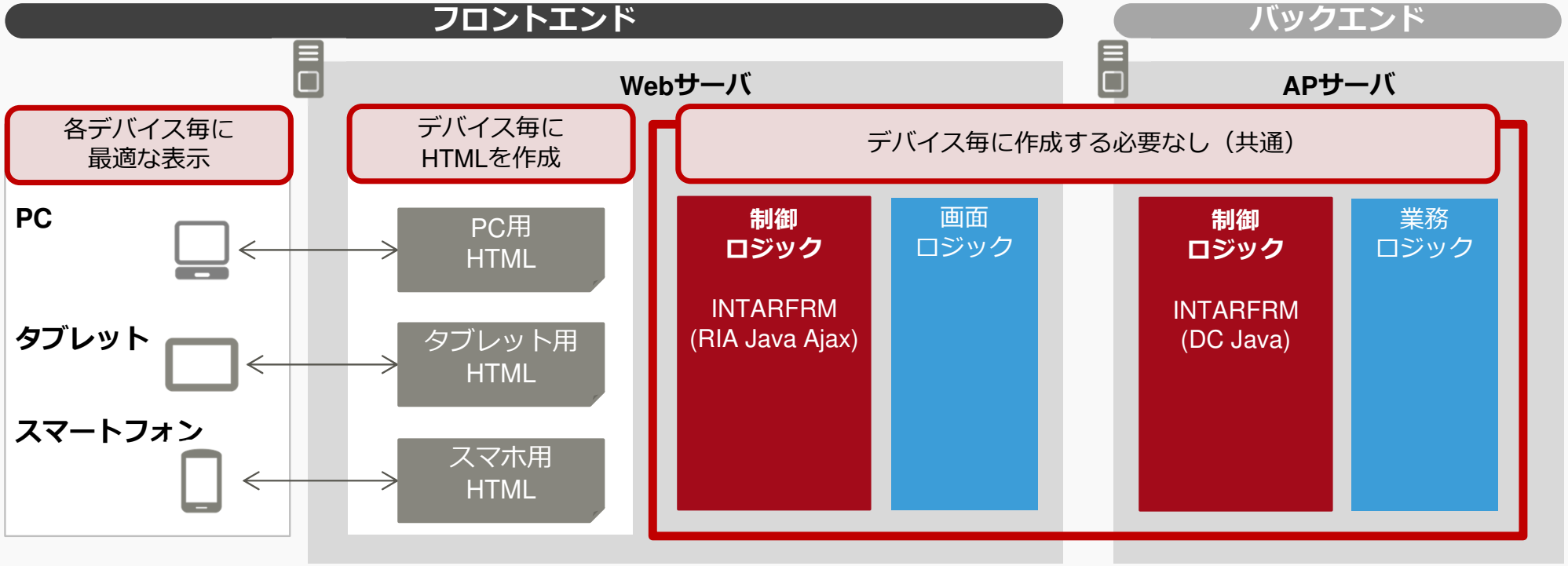
様々なデバイスやブラウザで動作するため、利用シーンが拡大します。

© 2024 Fujitsu Limited

マルチデバイス開発時の考え方

■タブレットやスマートフォンの画面サイズ・デザイン・レイアウトに対応する場合、各デバイス向けに調整したHTML(※)を作成するだけで、制御ロジック、画面ロジック、業務ロジックをデバイスごとに作る必要はありません。

（※）画面項目の並び変更等を想定



制御ロジック、画面ロジック、業務ロジックをデバイスごとに作る必要はありません。

© 2024 Fujitsu Limited

FUJITSU

高い操作性・表現力

「スマート」開発

「らくらく」 保守

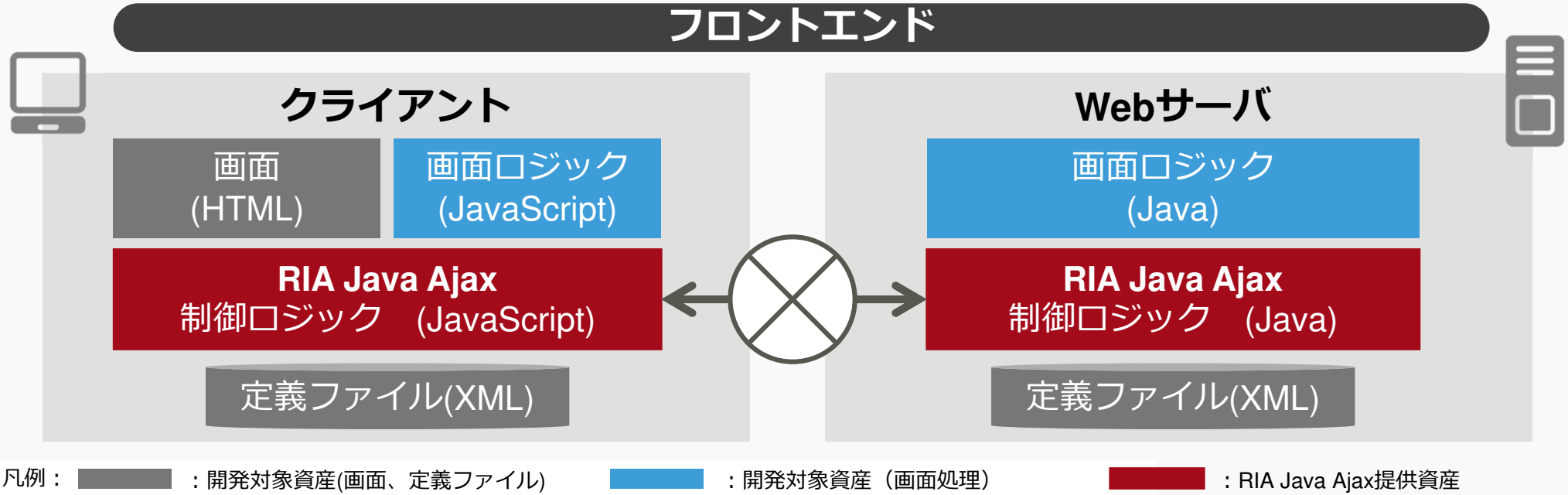
- RIA Java Ajaxの構成
- 制御ライブラリ
- 定義ファイル
- モックアップ機能で設計品質向上
- JavaScriptの開発量を削減、Javaで開発

© 2024 Fujitsu Limited

RIA Java Ajaxの構成

FUJITSU

- RIA Java Ajaxは画面ロジック(JavaScript, Java)、制御ロジックそれぞれを分離したつくりになっています。
  実装が難しい制御ロジックは、 RIA Java Ajax提供ライブラリと定義ファイル作成により実現します。
- 開発者は制御ロジックをコーディングレスで実現できます。
  「ロジックが整理される」、「開発対象が削減される」ことで開発効率・保守効率が向上します。

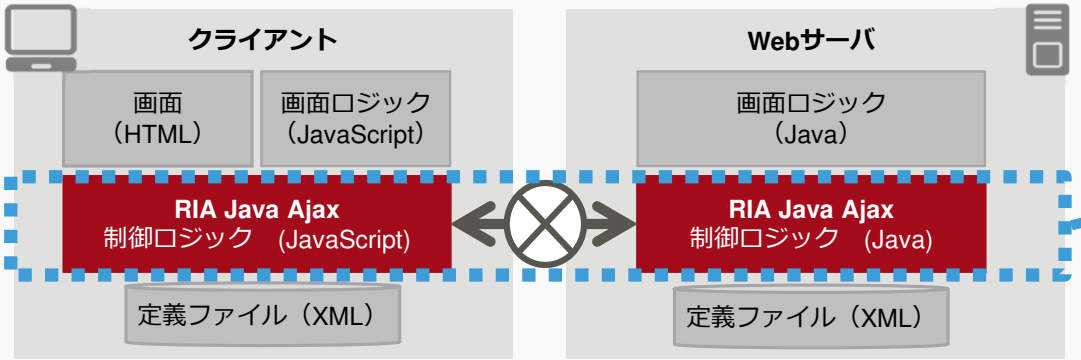


凡例：　：開発対象資産(画面、定義ファイル)　：開発対象資産（画面処理）　：RIA Java Ajax提供資産

システムの短期導入、高品質化、保守効率向上が実現できます。

© 2024 Fujitsu Limited

制御ライブラリ



品質を担保している制御ロジックを提供

制御ロジックはコーディングレス。

- 開発者は画面ロジック(JavaScript, Java)の開発に専念できます
- 開発に手間のかかるJavaScriptの開発量が減少します

●制御ライブラリの活用例

RIA Java Ajax 非適用

```
【HTML】
<form name="iform" action="abc.cgi" method="post">
   <input type="text" name=" userZipCode " size="30"
        onsubmit=" userZipCodeCheck();"><br>
</form>
【JavaScript】
function userZipCodeCheck(){
   var str = document.iform.userZipCode.value;
   if(str.match( /^[0-9]{3}-[0-9]{4}/)){
      alert("正しいフォーマットで入力してください。");  }  }
```

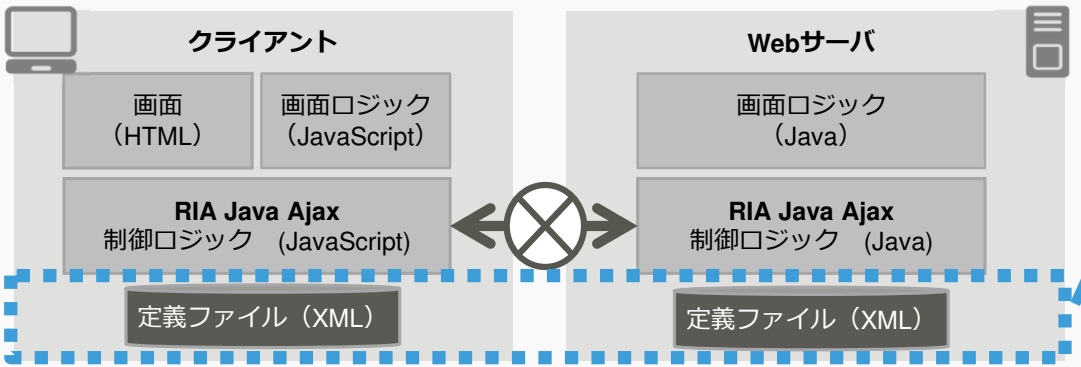
RIA Java Ajax 適用

```
【HTML】
<div rcf:type="CHFTextInput" rcf:itemId="userZipCode"
rcf:checker="chf.FormatChecker
('^[0-9]{3}-[0-9]{4}&#0036;', 'ZIP')"></div>
```

開発量が減り、「短期導入」が実現。開発効率、品質が向上します。

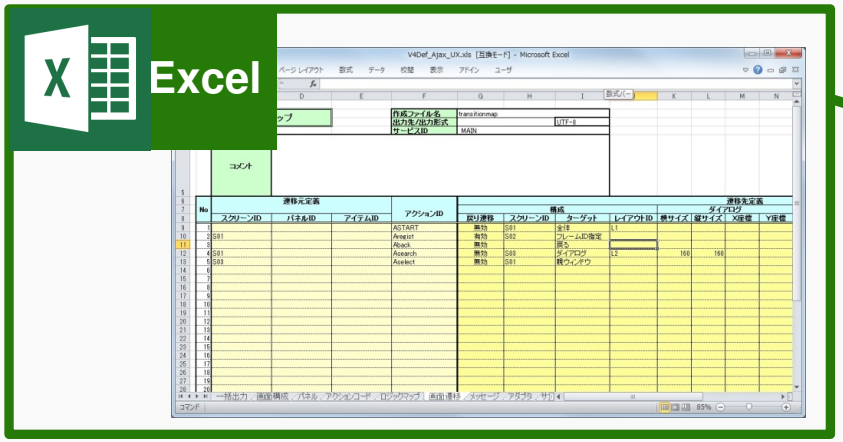
© 2024 Fujitsu Limited

定義ファイル



制御ロジックから制御部分を定義ファイルとして分離

- 専門的なスキルがなくても制御ロジックを利用できます
- 定義ファイルの設定情報の変更は、定義ファイルを入替え、アプリケーションを再起動することで実行環境に反映されます（コンパイルは必要なし）



Excelで編集できるツールを提供

- Excelで定義情報(※)を確認できるため、見える化が図れます

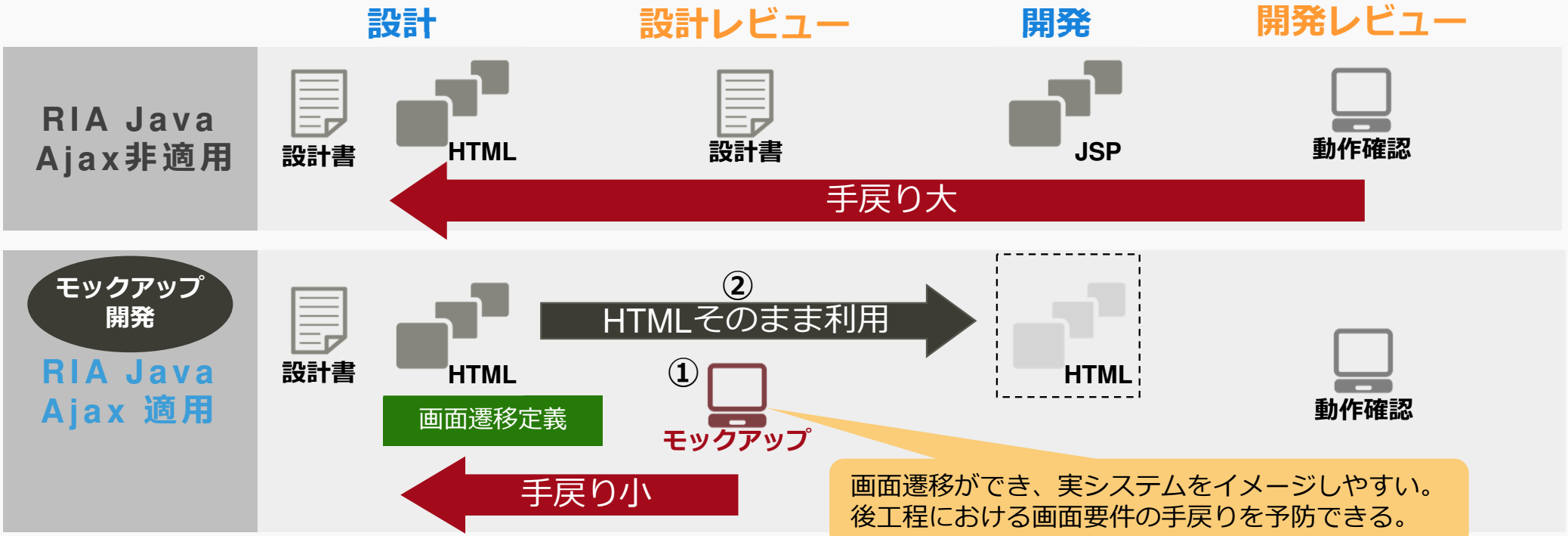
（※）定義情報：
画面構成、パネル、アクションコード（ボタンと動作の紐付け）、
ロジックマップ（アクションコードと呼び出す画面ロジックの紐付け）、
画面遷移、メッセージ、アダプタ、サブシステム

開発効率が向上します。仕様変更が簡単にでき保守効率が向上します。

© 2024 Fujitsu Limited

モックアップ機能で設計品質向上

①画面(HTML)と画面遷移の定義をするだけで、画面の見た目と画面遷移のレビューが可能
②モックアップで作成した画面HTMLは開発工程でそのまま利用可能

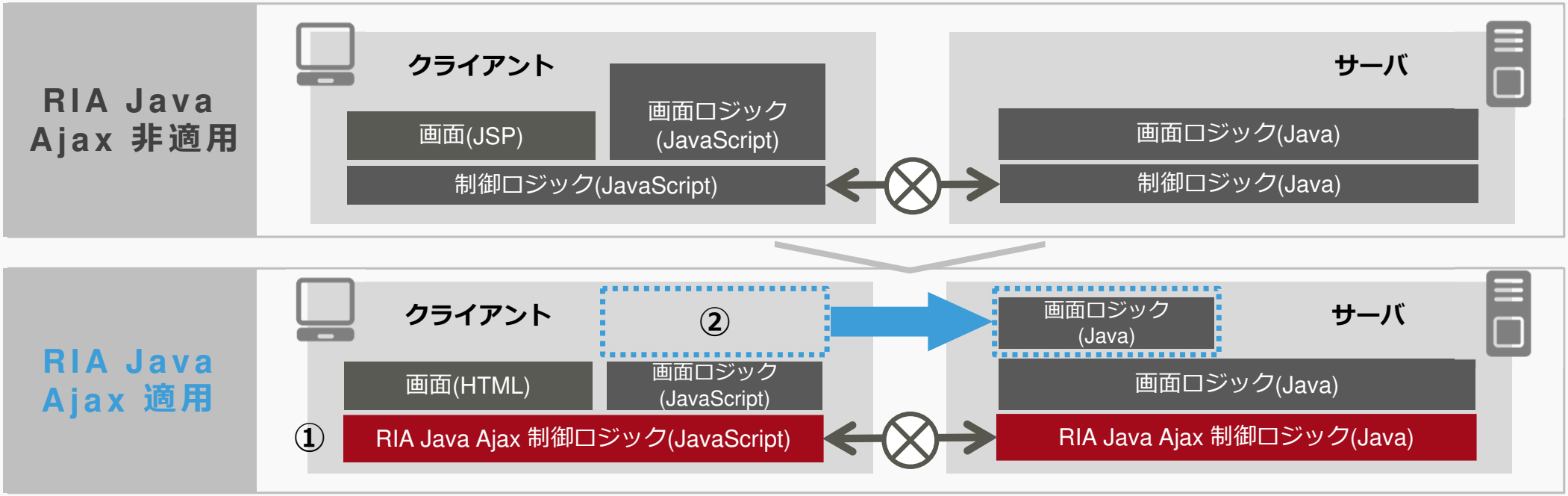


設計レビューで画面の動作が事前確認できるため、開発レビューに比べ手戻りが少なくなります。

© 2024 Fujitsu Limited

JavaScriptの開発量を削減、Javaで開発

①制御ロジックはライブラリとして提供されるため、開発者の実装量が減少
②クライアント側の入力チェックや画面項目の編集等画面ロジックをサーバ側のJavaで開発が可能



開発効率・品質が向上し、
短期間で高品質なアプリケーション開発を実現できます。

© 2024 Fujitsu Limited

課題の解決

FUJITSU

- 直感的に操作したい
- 表現力の高い・見た目を良くしたい
- 操作中のストレスを軽減したい

「高い操作性・表現力」を実現します。

- 費用対効果の高いシステムを導入したい
- ビジネススピードに合わせて、短期間で導入をしたい
- 高品質なシステムを使いたい

「スマート開発」を実現します。

- 保守の手間をなるべく省きたい
- システム保守費用を抑えたい
- 画面や機能の変更・追加を簡単にしたい

「らくらく保守」を実現します。

© 2024 Fujitsu Limited

3.保守性・拡張性の高いバックエンド開発を支援（INTARFRM Distributed Component Java）

© 2024 Fujitsu Limited

DC Javaとは

保守性・拡張性の高い**バックエンド開発**
を支援するフレームワーク

FUJITSU Software INTARFRM
Distributed Component Java



※Distributed Component Java(以降DC Javaと表記)**は、富士通受託開発を前提とする製品です。**

© 2024 Fujitsu Limited

DC Javaとは

DC Javaを導入することで下記の実現が可能です。

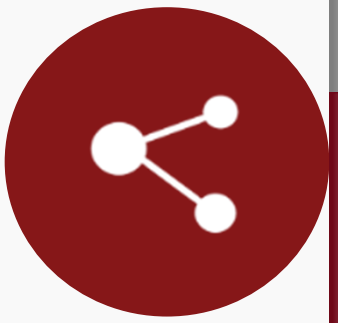


設計中心の開発

1.開発機能のインタフェース設計を起点とする型決め開発が可能です

2.設計書とインタフェース部品の整合性を保つことが可能です

3.インタフェース部品は不正な業務データの侵入を防ぐことが可能です



様々なサービスやデバイスと連携

1. 開発した機能をWebサービスからバッチまで様々な方式で呼び出せます

2.クラウド上の公開Webサービスをスピーディーに導入可能です

3.バックエンドの変更なしで多様なデバイスに対応可能です

© 2024 Fujitsu Limited

情報システム部門のよくある課題とDC Javaの提案

FUJITSU

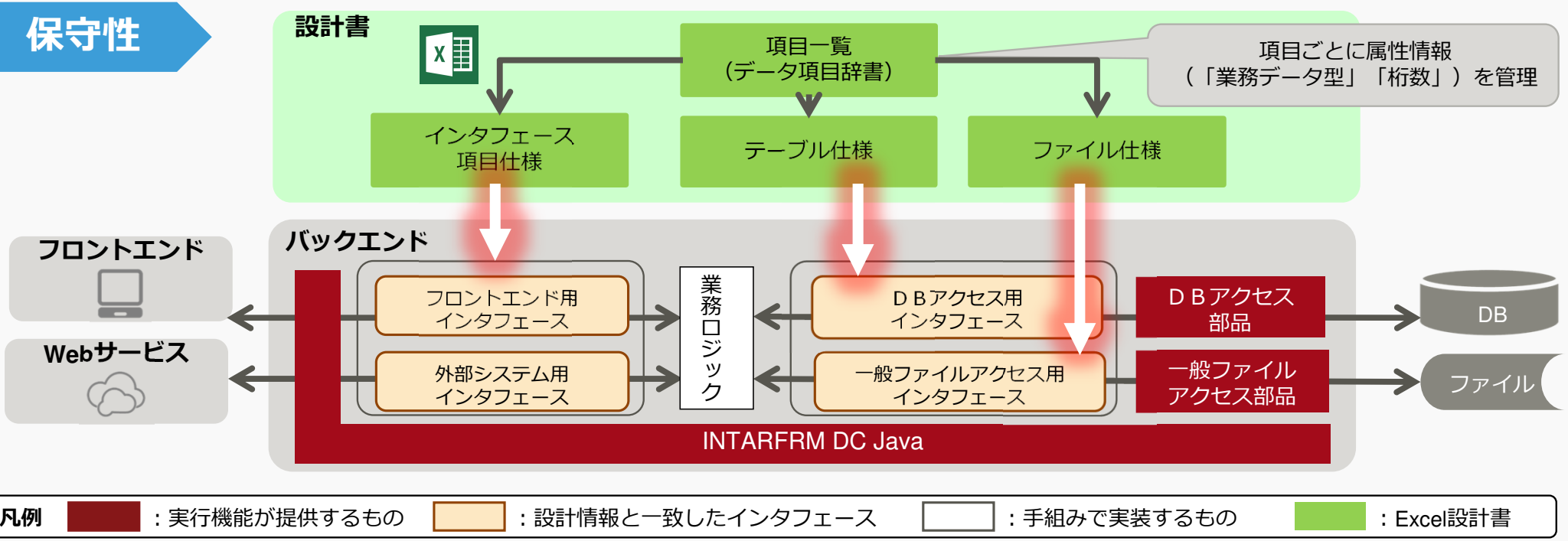
情報システム部門のよくある課題		DC Java の提案	
課題①	開発、テスト、保守の費用を抑え かつ、システム品質も良くしたい。	特長①	設計情報中心の開発プロセスを 実現します。
課題②	セキュリティの観点も含めてシステムに 不正なデータを取り込みたくない。	特長②	安心・安全なバックエンドシステムを 実現します。
課題③	フロントエンドのデバイスが変わる度に バックエンドの業務ロジックを改修したくない。	特長③	デバイスの選択肢を広げます。

© 2024 Fujitsu Limited

特長①　設計情報中心の開発プロセスを実現

■機能：　設計書の項目属性情報（業務データ型、桁数）をインタフェースに取り込むことにより
プログラムの属性情報の整合性を保つ。
■効果：　設計書とプログラムが乖離しないため、開発の全工程（設計／製造／テスト／保守）において
効率化および品質の向上を図りつつ、開発のライフサイクルマネジメントが実現できます。

保守性



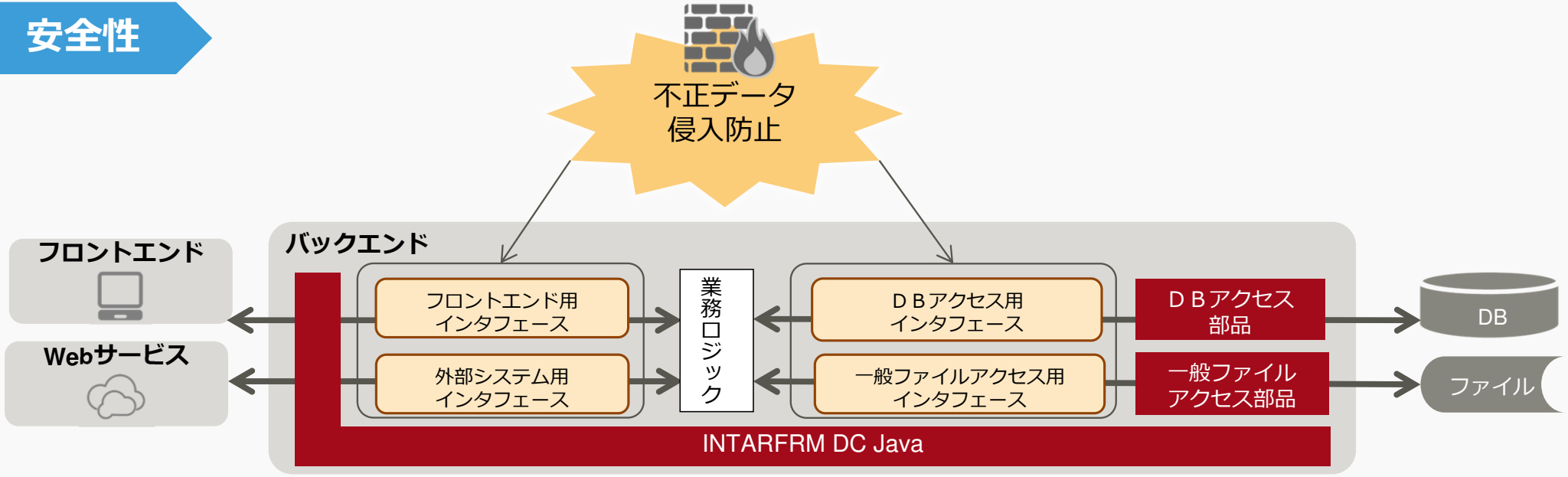
© 2024 Fujitsu Limited

特長②　安心・安全なバックエンドシステムを実現

■機能：　インタフェースの入口と出口でデータ項目（型・桁）の不正な業務データの侵入を防ぐ。

■効果：　安心・安全なバックエンドシステムを実現できます。 バックエンドシステム全体の実行品質が飛躍的に向上します。手組みする業務ロジックはシンプルになります。

安全性



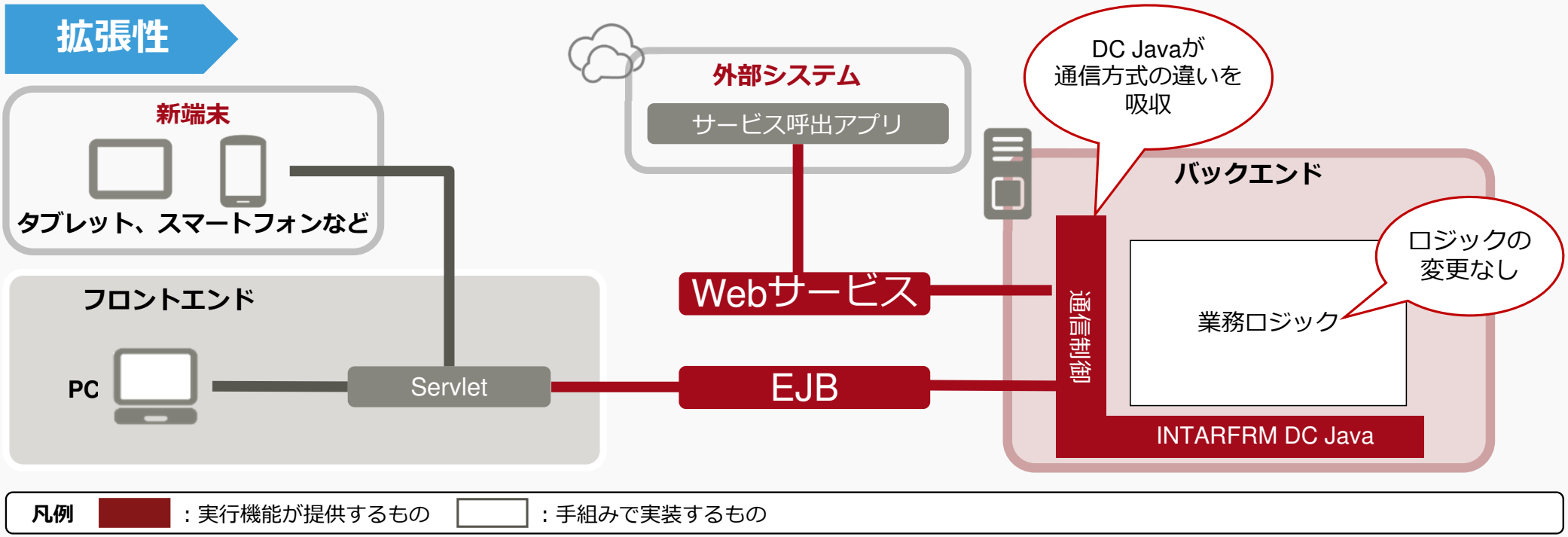
凡例　：実行機能が提供するもの　：設計情報と一致したインタフェース　：手組みで実装するもの

© 2024 Fujitsu Limited

特長③　デバイスの選択肢が広がる

■機能：　多彩な通信方式と連携

■効果：　業務ロジックが通信方式に影響を受けないためバックエンドの変更なしで、
多様なデバイスに対応可能です。　→システムの拡張性が向上します。



凡例　：実行機能が提供するもの　：手組みで実装するもの

© 2024 Fujitsu Limited

情報システム部門のよくある課題とその解決

FUJITSU

情報システム部門のよくある課題		DC Java の提案
課題① 開発、テスト、保守の費用を抑え かつ、システム品質も良くしたい。	▶	特長① ・設計情報から実装物が乖離しません。
課題② セキュリティの観点も含めてシステムに 不正なデータを取り込みたくない。	▶	特長② ・呼出元／呼出先（外部システム含む）からの 不正データがシステム内に混入しません。
課題③ フロントエンドのデバイスが変わる度に バックエンドの業務ロジックを改修したくない。	▶	特長③ ・デバイスの選択肢を広げます。 ・通信方式に影響を受けない業務ロジックを 実現します。

© 2024 Fujitsu Limited

4.まとめ

© 2024 Fujitsu Limited

課題を一気に解決

Fujitsu Software INTARFRM

INTARFRM Ent .Edi. 導入による課題解決

最新クライアント技術に追随します。

・画面部品の提供
・マルチデバイスに対応

保守、拡張しやすいつくりにします。

・設計中心の開発手法
・通信方式に影響されない業務処理実現

新たなフロントエンドの技術に
追随可能 (最新デバイス追加、利便性追求等…)



資産がまとまり、保守性・拡張性が向上

コスト削減

新システムへ投資



早期リリース

他社に先駆けサービス展開

© 2024 Fujitsu Limited

Thank you



© 2024 Fujitsu Limited

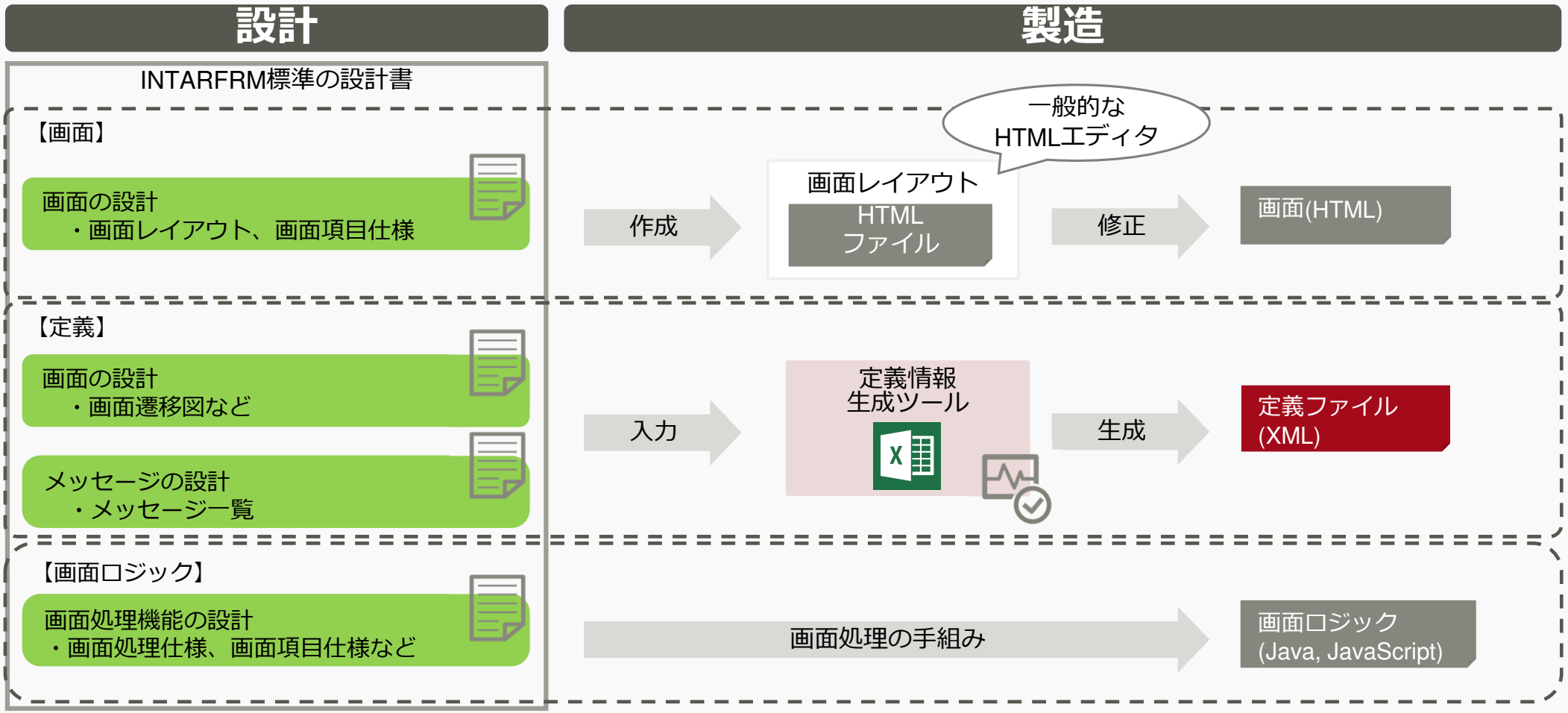
【添付資料】

© 2024 Fujitsu Limited

開発の流れ

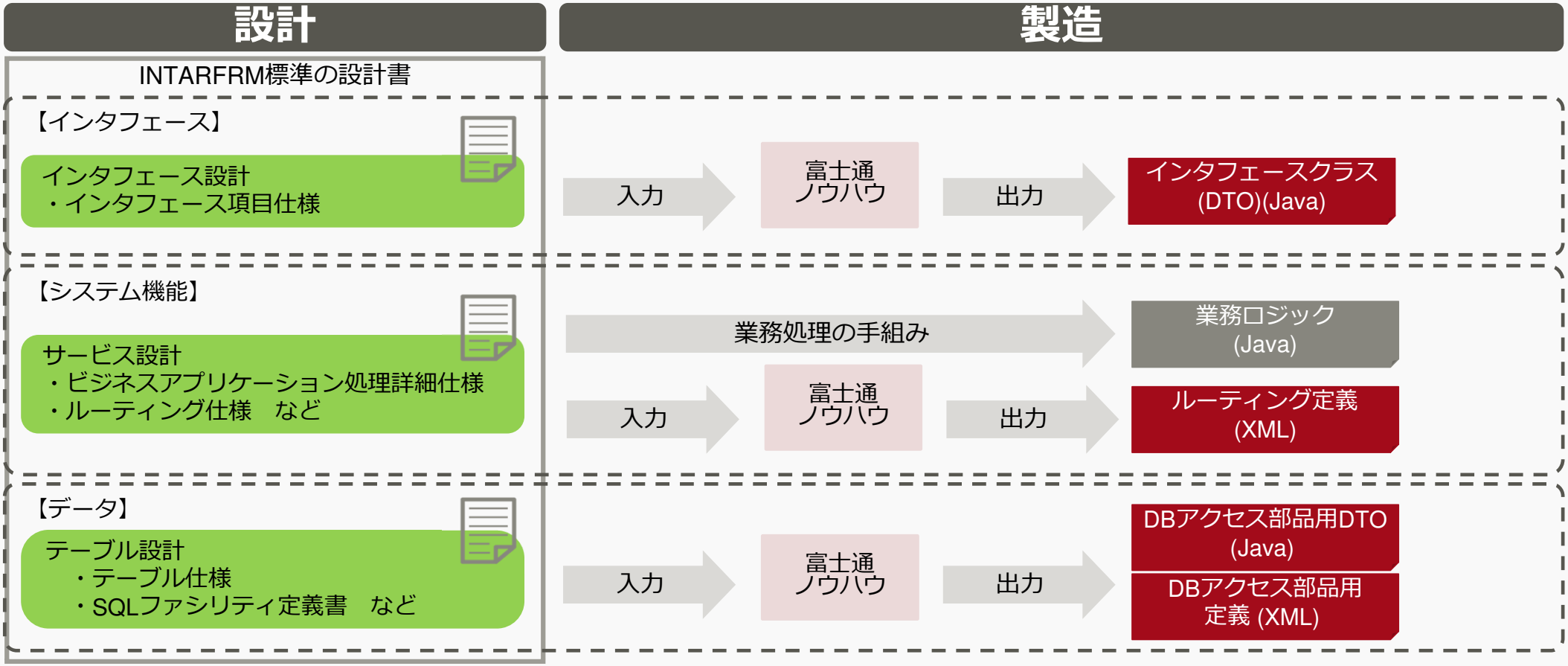
© 2024 Fujitsu Limited

開発の流れ



© 2024 Fujitsu Limited

開発の流れ



※Distributed Component Javaは、富士通受託開発を前提とする製品です。

© 2024 Fujitsu Limited

機能一覧

© 2024 Fujitsu Limited

機能一覧(1/2)

機能名			概要
画面制御	画面遷移制御	高速化表示	画面高速化技術（画面の事前読み込み、クライアント側キャッシュ）により、 高速な画面の切替えや表示が可能。
		パネル複数表示	1画面内にパネルを複数表示できる機能。
		マルチウィンドウ表示	業務画面を複数表示し、独立した複数の業務の実行が可能。
	データ入出力	カーソル移動（フォーカス遷移）	Tabキーのほか、任意キー（Enter等）押下によるフォーカス遷移機能。
		入力時データチェック	必須入力チェック等、入力項目をチェックする機能。
		入力時エラー処理	入力エラー発生時、必要な後続処理を呼び出す機能。
		入出力データ自動転記	画面間でのデータ引き継ぎを定義ファイルのみで可能とする機能。
		項目部品属性変更	サーバのロジックにて、画面項目の属性変更可能。
	画面部品制御	画面部品　(*1)	HTML標準で提供される画面部品に加えて、jQuery UIが提供する高い操作性と 表現力を持つ部品群。
		jQuery一般公開 プラグインとの連携 (画面部品拡張機能)	一般的に公開されているjQuery一般公開プラグインを活用する利用者側での 画面部品拡張を可能とするための仕組み。
開発支援	定義ファイル(XML) 生成	Excel定義からの自動生成	定義ファイルの開発支援等による定義ファイル作成の機能。
	画面エディタ	プレビュー機能	画面項目に値を設定した状態でのプレビューが可能となる機能。
		JavaScript入力補完	EclipseのJavaで実現されているような入力補完をJavaScriptでも実現するための機能。
	実行モード	テストモード・ プレゼンテーションモード(*2)	開発時向けに詳細ログを取得するテストモードや、画面ロジック(Java)を実装することなく 画面遷移を確認することができるプレゼンテーションモードを提供。

(*1) 各画面部品の詳細については、画面部品一覧をご参照ください。
(*2) テストモード　　：デバッグログが出ます。
　　運用モード　　：デバッグログが出ません。
　　プレゼンテーションモード：紙芝居をしたい場合に利用します。

© 2024 Fujitsu Limited

機能一覧(2/2)

機能名			概要
実行制御	システム制御	起動/終結処理	開始／終了手続きや、画面ロジック(Java)起動のための初期化・終了処理。
		例外処理（エラーハンドリング）	通信エラー等、通常のWebシステムでは制御できないエラーをハンドリングし、 処理継続を可能とする仕組み。
		動作環境(*3)動的切り替え	サブシステム(*4)の起動を動的に切り替えられる機能。
	イベント制御	イベント振り分け	画面イベントと画面ロジック(Java)の関係を容易に対応付けることを可能とする制御。 画面ロジック(Java)の再利用が可能。
		画面イベント制御	画面から通知されるイベントの一括制御機能。 ダブルクリックの抑止等、煩雑なイベント制御の実装が不要。
	メッセージ制御	メッセージダイアログ表示	メッセージダイアログの表示が可能となる機能。
		メッセージ文言メンテナンス	画面ロジック(Java)からメッセージ文言を独立させ、メッセージ文言のメンテナンスを 容易にする仕組み。
	ユーティリティ	オフライン機能	HTML5のオフライン機能を活用して、オフライン運用が可能なアプリケーションの 構築が実現可能となる機能。
		JavaScript処理ログのサーバへの出力	クライアントのログをサーバに出力する機能。
		JavaScript処理ログのクライアント出力	HTML5の機能を活用して、従来まで実現が困難であったクライアントへのログを 出力する機能及びログアップロード機能。
		画面サイズの可変化	ブラウザの表示サイズの変更に伴って、アプリケーション画面の表示サイズも 変更されるような画面の作成が可能となる機能。

(*3) 動作環境：定義情報生成ツールで定義する情報
(*4) サブシステム：別業務（子画面呼び出し）

© 2024 Fujitsu Limited

画面部品一覧

凡例：● jQuery UI部品群で提供　○ HTML部品群で提供

画面部品		提供形態
テキスト	任意の文字列を出力できる部品。	○
テキスト入力	1行の任意の文字列を入出力できる部品。	○
テキストエリア	複数行にわたり、任意の文字列を入出力できる部品。	○
チェックボックス	チェック形式によりON,OFFの状態を持つことができる部品。	○
チェックボックスグループ	チェックボックスをグループ化する部品。	○
ラジオボタン	ON,OFFの状態を持つボタン形式の部品。複数個組み合わせて、択一選択の入力を実現する。	○
ラジオボタングループ	ラジオボタンをグループ化する部品。	○
ボタン	押されたときにアクションを発生させる部品。	○
リンク	HTMLのリンク（Aタグ）を表示する部品。ボタンと同様に扱うことが可能。	○
選択リスト	あらかじめ用意された選択肢から選ぶことが可能な部品。	○
イメージ	ビットマップ等画像のデータを表示する部品。	○
カレンダー	カレンダーを表示する部品。	●
アコーディオン	コンテンツの折り畳み（開閉）が可能なメニュー表示部品。	●
タブ	複数の画面を1つのウィンドウ内で切り替えて使用するための部品。ボタン操作によりコンテンツの表示切替を行う。	●
タグイテレーター	画面部品を動的に繰り返し生成する部品。可変部品、繰り返し部品の実装負荷を軽減。	●
プログレスバー	処理の進捗状況を表現する部品。	●
オートコンプリート	過去の入力履歴を参照して次の入力内容を予想し、あらかじめ表示する部品。	●
スピナ	「△」や「▽」等のボタンで、入力欄の数値データを増減できる部品。	●
ツールチップ	カーソルが当てられた項目へのコメントを表示するための部品。	●
アイコン	画面に表示するアイコン画像のサンプル。	●
ファイルアップロード/ ダウンロード	HTML5機能を利用したファイルアップロード/ダウンロード部品。ドラッグ&ドロップによる直感的な操作が可能。 プログレスバーによるアップロード進捗状況の表示が可能。	●

© 2024 Fujitsu Limited

機能一覧(1/3)

No.	機能名称	機能説明
1	コンポーネントコンテナ ・オンライン用 ・バッチ用／センターカット用	分散コンポーネントを制御する基盤機能です。 オンライン処理方式／バッチ処理方式／センターカット方式(*1)に対応しています。
2	通信制御 連携クライアント	オンライン処理方式において、クライアントからコンポーネント呼び出しを行う際の 通信プロトコルとして複数の選択肢を提供します。 EJB, Servlet(HTTP), Webサービス(SOAP/REST), ローカルJava通信方式（センターカット含む）に対応しています。 また、クライアントアプリに対して通信プロトコルに依存しないコンポーネント呼出し機能を 併せて提供します。「サービス呼出定義」により通信形態や接続情報を外部定義します。
3	インタフェースクラス(DTO)	分散コンポーネントを呼出・応答するためのインタフェースオブジェクトを提供します。 属性バリデーション機能を備えたPOJO(*2)型オブジェクト（DTOと呼びます）を提供します。 DTOを利用するメリット ・POJO型メッセージであるため、外部との連携が容易です。 ・設計書の設計情報と実装は乖離しません。 ・設計書に記述された属性にもとづいた属性チェック機能を備えています。

(*1)　センターカット方式：バッチアプリからオンラインアプリを呼び出す方式。
(*2)　POJO：Plain Old Java Objectの略。Javaで記述されるオブジェクトのうち、特定のフレームワークやオブジェクトモデル、規約などに縛られず、
　　また、これらに依存しないごく普通のJavaオブジェクトの名称。

※Distributed Component Javaは、富士通受託開発を前提とする製品です。

© 2024 Fujitsu Limited

機能一覧(2/3)

No.	機能名称	機能説明
4	ルーティング制御	分散コンポーネント内ビジネスアプリケーションの処理振分け・順序制御を行います。 １度の分散コンポーネント呼出で、複数のビジネスアプリケーションを組み合わせて処理できます。 トランザクション制御、業務規制を行います。
5	共有データ管理	分散コンポーネント内業務ロジック間でデータを共有する機能です。 扱う共有データは、データベース、ファイル、Servletセションへ保存することができます。 アプリケーション環境（Servlet, EJBなど）に依存しない仕組みでステートフルなアプリケーションを 実現できます。
6	ログ出力制御 ・アプリ用／アプリ基盤用 ・バッチ／シェルスクリプト用	デバッグ用ログ（アプリケーション、フレームワーク）および運用ログの出力機能です。 スケルトン実装にてJava APIのログ出力（java.util.logging）と連携しています。
7	トランザクション制御	データベースアクセスを行うビジネスアプリケーションが容易かつ適切に データベースコネクションとコミット・ロールバックの制御を行うための機能です。 DBアクセス部品を利用する場合、アプリケーションにトランザクション制御処理を 実装する必要はありません。
8	例外・リトライ制御	例外発生時に処理の制御を行います。 またリトライ対象例外の場合はトランザクション処理の再実行を行います。
9	一般ファイルアクセス部品	レコード形式のインタフェースオブジェクトでCSV, 固定長ファイルの読み込み、 書き込みを行います。

※Distributed Component Javaは、富士通受託開発を前提とする製品です。

© 2024 Fujitsu Limited

機能一覧(3/3)

No.	機能名称	機能説明
10	DBアクセス部品	インタフェースクラス(DTO)を使用してDBの更新や検索などを行います。 O/Rマッピング機能(*3)を持っています。 単テーブルおよび複数テーブルアクセスが可能です。
11	スケジュールドバッチ（アプリ制御）	スケジュールドバッチ処理方式において、バッチアプリケーションの実行制御を行います。
12	リクエストバッチ（アプリ制御）	リクエストバッチ方式において、リクエストによるジョブの実行制御を行います。
13	業務規制	オンライン処理方式において、コンポーネントのサービス単位に利用可能時間の設定や 一時的な停止／開始の実行を制御します。
14	トランザクションログ（利用履歴）	コンポーネントのサービス処理の利用履歴をトランザクション単位にDB出力する機能です。

(*3)　O/Rマッピング機能：アプリケーションオブジェクト内で取り扱う項目（＝フィールド名）とテーブルに定義された項目（＝カラム名）との
マッピングロジック（相互の対応にもとづいた転記処理）を自動で行う機能。テーブル構造を意識せずプログラミングすることが可能。

※Distributed Component Javaは、富士通受託開発を前提とする製品です。

© 2024 Fujitsu Limited

動作環境

© 2024 Fujitsu Limited

動作環境

動作環境については、以下のINTARFRMサイトよりご確認ください。

富士通アプリケーションプラットフォーム情報サイト> INTARFRM > 製品資材一覧（PDF）
https://global.fujitsu/ja-jp/capabilities/software/platform

© 2024 Fujitsu Limited