

TOSHIBA

Java Conference主催
「Javaに関する技術、応用、表現大賞'99」



技術部門大賞受賞

ユーザビリティの高い J2EEシステムを構築するには

FlyingServ

J-Frame Serverのご紹介



Java
COMPATIBLE

Collaborative
Innovation

東芝ソリューション株式会社

TOSHIBA SOLUTIONS CORPORATION

プラットフォームソリューション事業部

■ 背景

-J2EE環境におけるインタフェースの課題-

■ リッチクライアント採用の課題

■ GUIコンテナという考え方 (*FlyingServ* J-Frame Serverのご紹介)

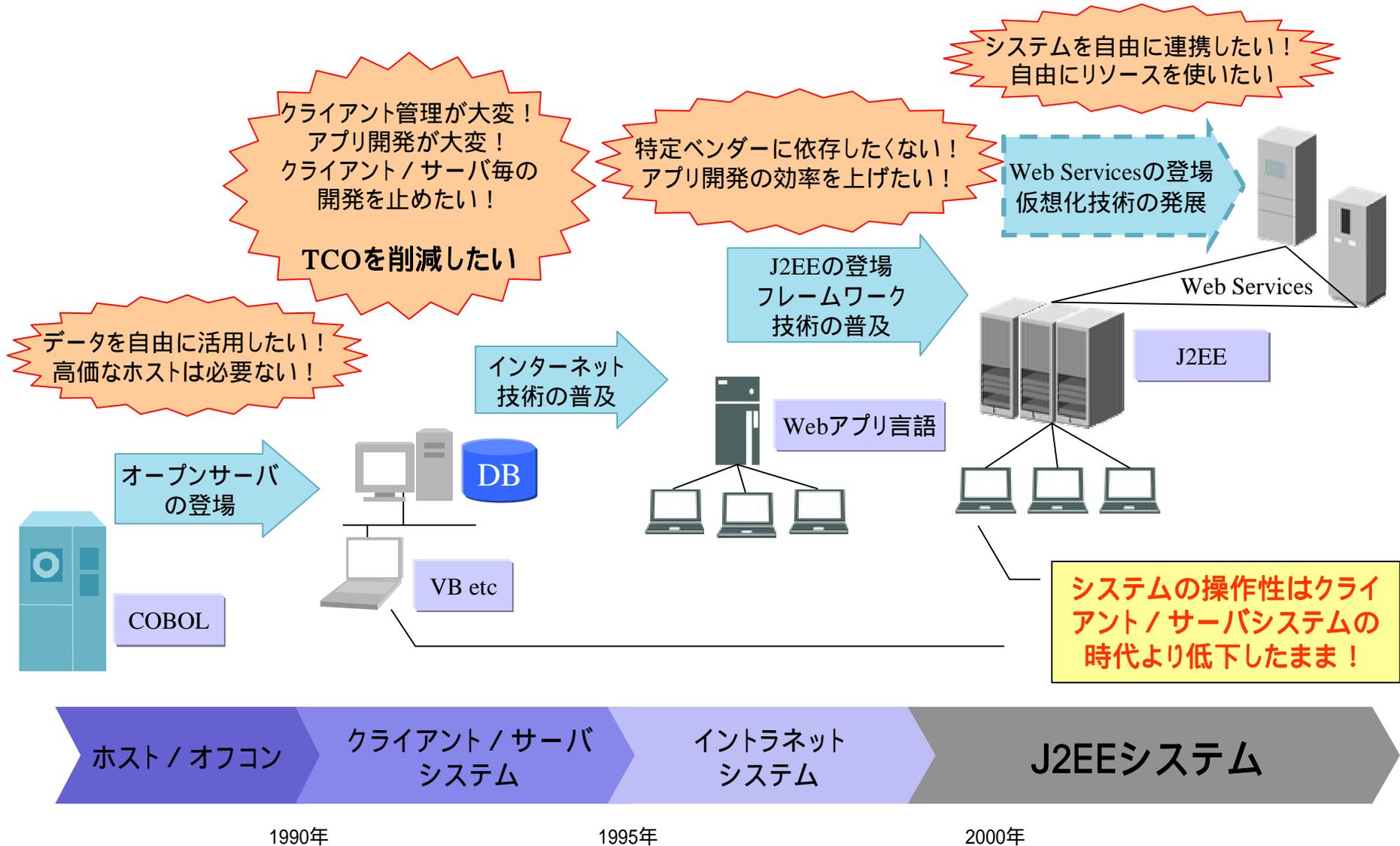
- 標準Java環境をサポートしたGUIコンテナ
- Appletがサーバで動く事の意味
- GUI部品によるアプリケーション開発
- J2EEアプリケーションとの連携
- J2EEフレームワークとの連携
- Java開発ツールを使ったGUI-J2EEアプリの一貫開発

■ 活用事例

背景

- J2EE環境におけるインタフェースの課題 -

標準プラットフォームの遷移



ユーザインタフェースの課題



- システムがWeb化されると、操作性は悪くなった！
(エントリ業務の効率低下！
 ➡ オフコンの方がまだ良かった！)

- システム開発者が利用者の要求を聞かなくなった！

- 設計した画面 ➡ Webブラウザでできるのか？？
 設計に時間がかかる！

- Webブラウザのスクリプトデバッグが大変！



業務システムにおけるユーザビリティの重要性(Webシステムとの両立の重要性)

Webシステム ユーザビリティの問題

エントリー業務にパート100名が従事
ところがWebシステムの操作性が悪いがために、
必要な量をこなせず、皆を一時間残業させた

事業的
な損害

どちらを優先したらいいか？

C/Sシステム 運用管理の問題

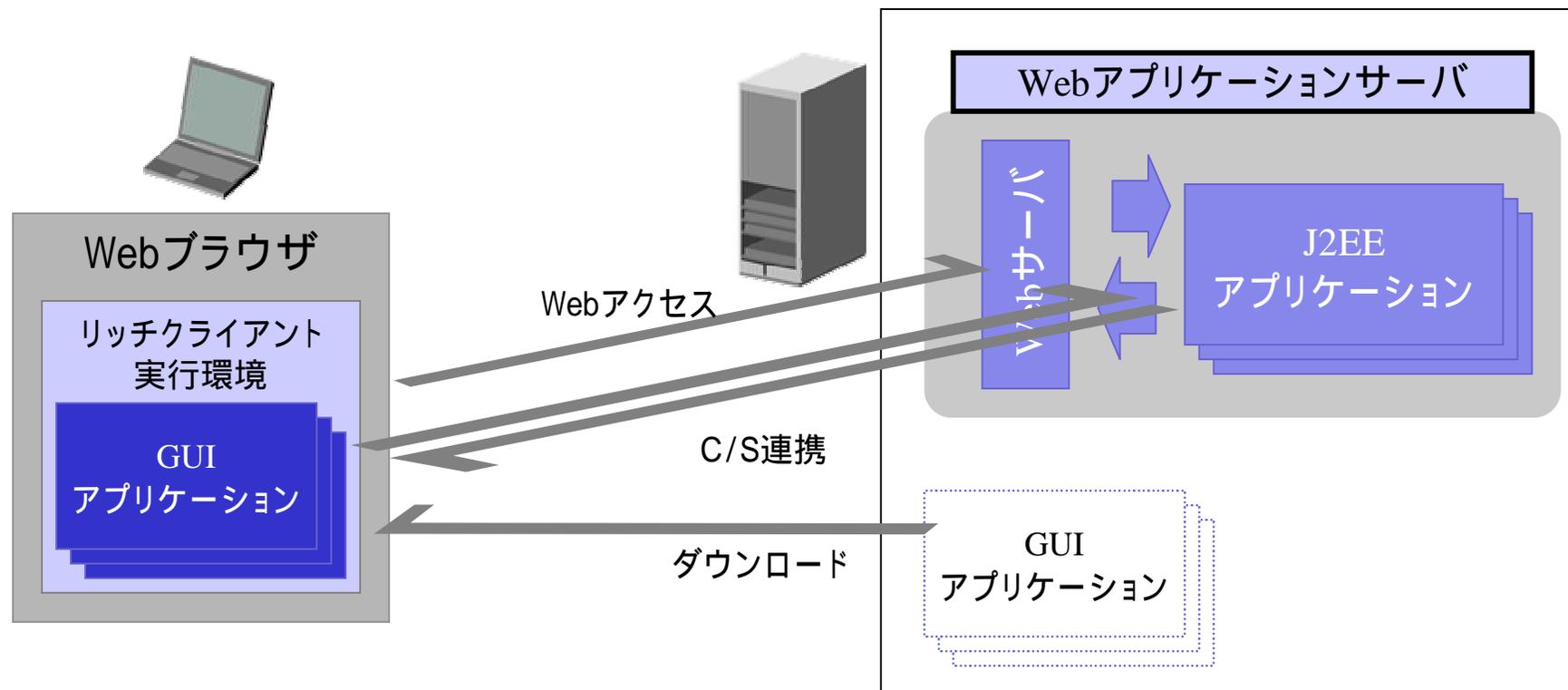
C/S型のエントリーシステムを稼動
多くのユーザからの問い合わせ対応のため、
クライアントVBプログラムとSQL*Netの設定
を行う専任サポート員を任命した

運用管理コスト
の増加

リッチクライアント採用の課題

一般的なリッチクライアントの仕組み

- リッチクライアントの実行環境を、予めWebブラウザにインストール
- GUIアプリケーション(またはGUI定義ファイル,スクリプト等)をWebの仕組みを使ってダウンロードし、ユーザインターフェース処理を実行
- サーバ側のデータはSOAP等を使ったC/S連携処理にて実現



リッチクライアント採用の課題

- クライアントに専用実行環境が必要となる

標準的なプラグイン Flash, Adobe Reader, Java Plugin これ以外は…

- クライアント開発に専用ツールが必要になる
- クライアント専門の開発チームができてしまう
- クライアントプログラムとサーバプログラムは、組み合わせフェーズにならないと連携確認ができない

まるで昔C / Sシステムを開発した時と同じ苦労が…

- サーバからデータを持ってくる時間を考えると、クライアントで一部ビジネスロジックを実装しないと、システム利用者の要求を満たさない…
といったことがある

J2EEではビジネスロジックは全てサーバに置けというが…

東芝ソリューションの考える

J2EEの為のリッチクライアントとは？



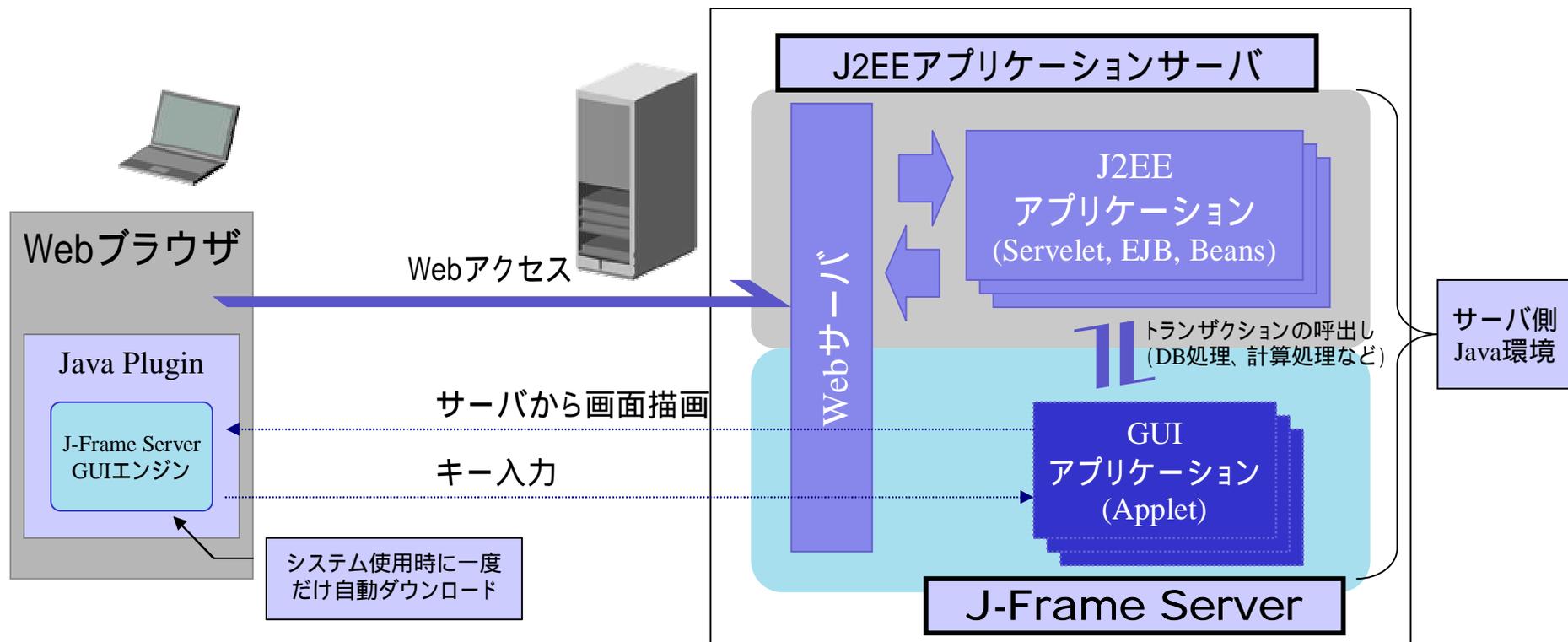
- クライアント環境に依存しないアーキテクチャでかつ、C/S型と同等の操作性を実現できること！
- オープンな標準技術で実現すること！
独自の開発言語、開発ツールで shouldn't be a thing!
- J2EEを補完し、J2EEの約束事を乱さないこと！！



GUIコンテナという考え方 (*FlyingServ* J-Frame Serverのご紹介)

FlyingServ J-Frame Serverとは

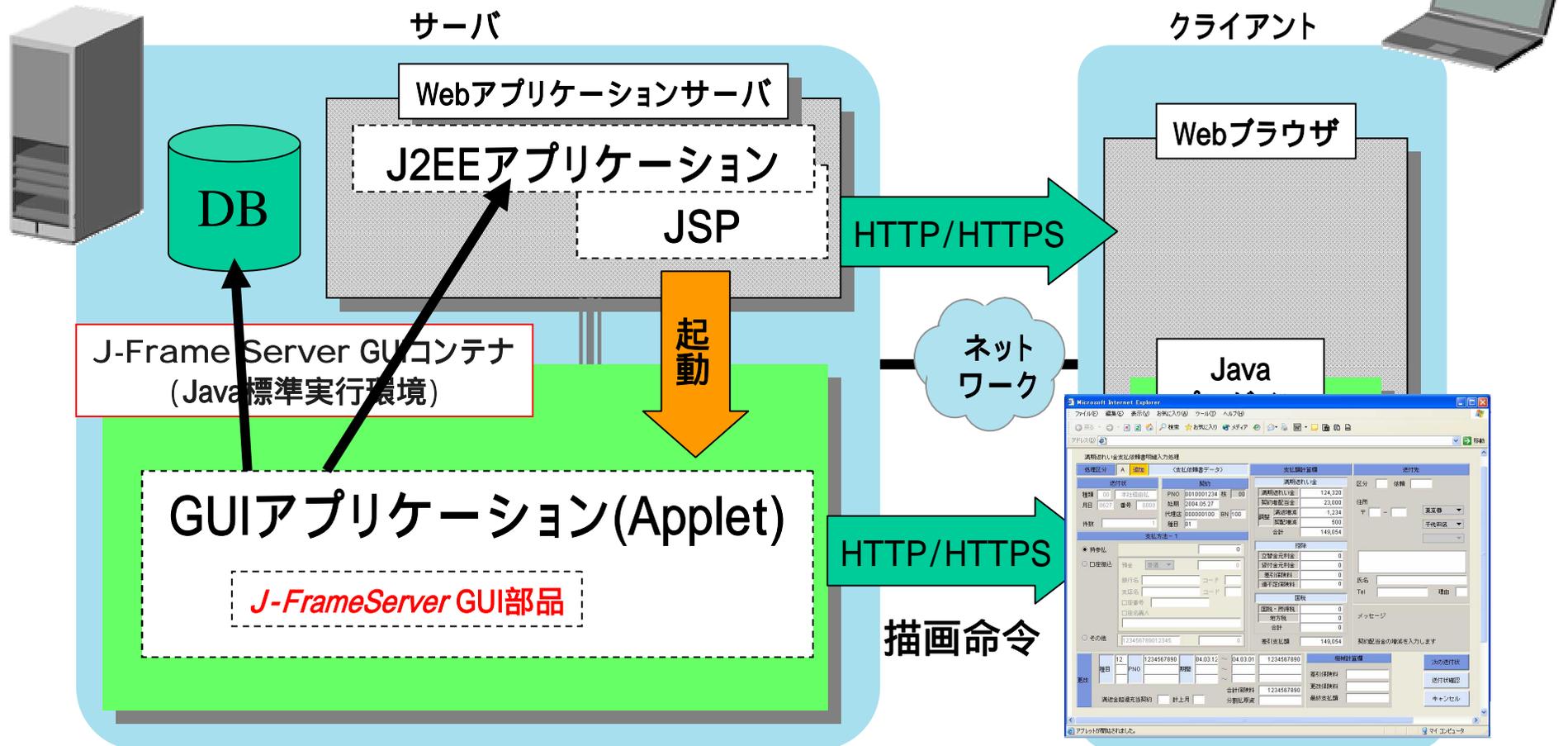
- サーバでGUIアプリケーションを動作させる環境を提供 (GUIコンテナ)
- サーバで動作するため、クライアント管理コストの増加を招かない
- サーバ側リソースを自由にアクセス、GUIアプリとJ2EEアプリは、Javaプログラムどうしの連携で一体となって動作
- GUIアプリはJSPより起動、Strutsなどのフレームワークで使用可能
- 標準Java仕様に準拠



FlyingServ J-Frame Serverの仕組み



GUI転送機能を付加したJava標準実行環境とビジネス用GUI部品



J-Frame Server GUIコンテナ・・・ Swingアプレットのライフサイクルを管理するJavaコンテナ
J-Frame Server GUI部品・・・ Swingで作成されたJavaBeans

標準Java環境をサポートしたGUIコンテナ



■ Java標準のGUI API (AWT , Swing) をサポート

- ◆ SUNの認証を受けたJava標準互換環境を提供。
- ◆ 特別なスクリプトやAPIの習得不要。独自APIに縛られないオープンな開発が可能。
- ◆ GUIビルダやサードベンダの提供する多彩な部品やツールを使用可能。

■ GUIアプリケーション (Applet) はJSPから呼び出し

- ◆ StrutsなどのJ2EEのフレームワークで使用可能。
- ◆ クラスのキャッシュ等、様々なGUIアプリケーション高速起動のしくみ。
(サーバアーキテクチャにしかできない)

■ HTTPによる通信

- ◆ J2EE環境における標準的な通信プロトコルである、HTTP / HTTPSをサポート。
- ◆ ファイアウォール越しのアクセス。
- ◆ HTTPSを使用して通信データ暗号化。



JSPおよび連携APIの例

GUIモジュールを起動するJSPの例

```
<%@page contentType="text/html"%>  
<%@taglib uri="/WEB-INF/fservts-fjlet.tld" prefix="fstst"%>  
<html>  
<head><title>Sample FJlet Page</title></head>  
<body>  
<fstst:fjlet code="sample.class" width="800" height="600" />  
</body>  
</html>
```

GUIコンテナ内で実行するモジュールを指定するため、カスタムタグとして"fstst"を定義。

sample.classをGUIモジュールとしてGUIコンテナ内で実行。表示される画面は、JSPから生成されるページ内に埋め込まれて表示。記述方法はAppletタグに準ずる。

GUIモジュールとBeansとの連携例 (GUIモジュール側のAPI呼び出し例)

GUIコンテナからJavaBeansへアクセスするための参照を取り出す。

```
manager = FJRemoteScope.application.getAttribute("manager");  
manager.setProperty("day", new Date());
```

BeansのsetDay()メソッドを呼び出して、日付をJavaBeansに設定

Appletがサーバで動く事の意味

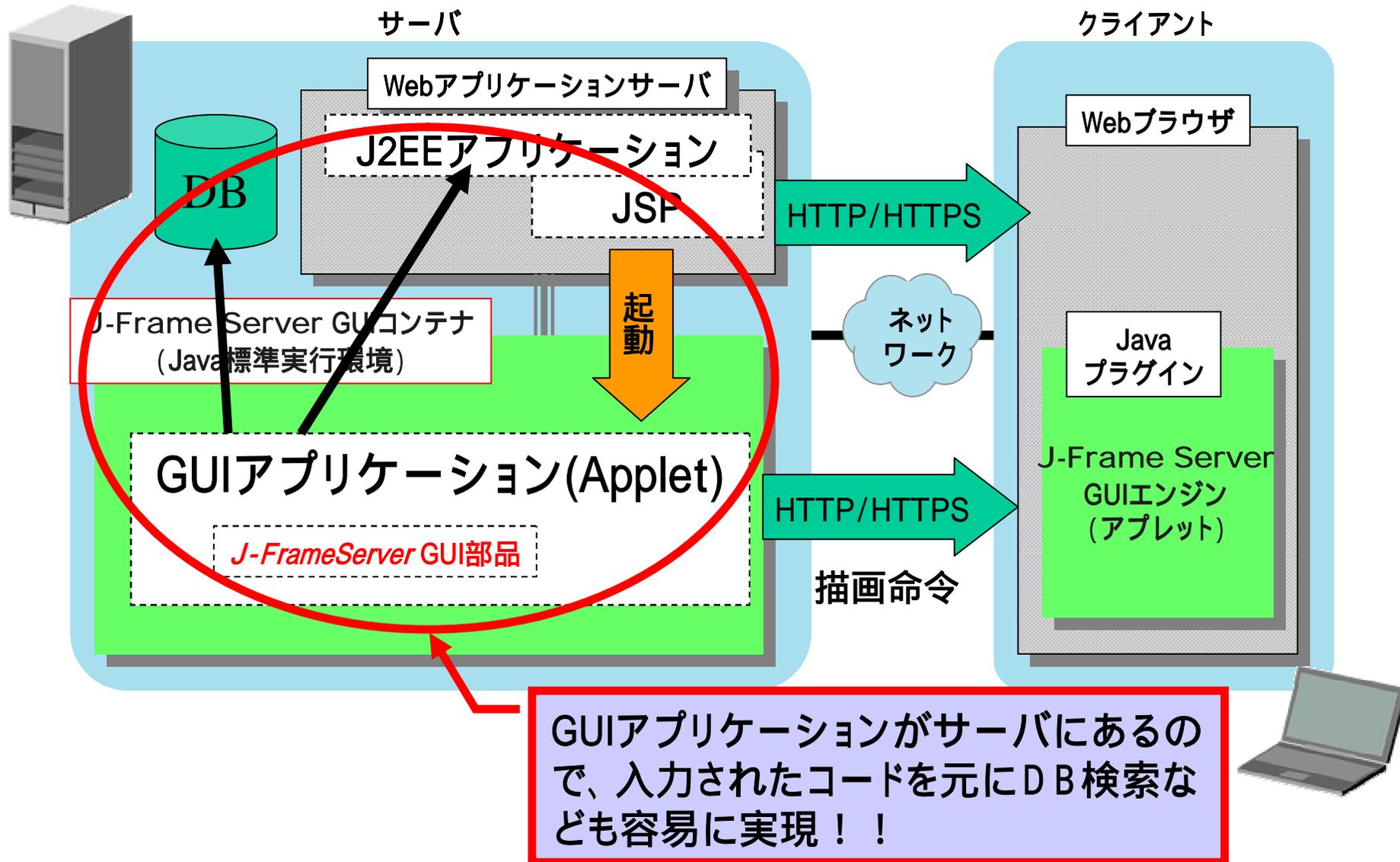


- J-Frame Serverを使えば、全てサーバプログラムとして開発できるので、C / S通信処理の設計は不要。
 - ◆ アプレットの制限なしに、サーバ側のリソース(DB,ファイルなど)をアクセス
 - ◆ GUIアプリケーションから、BeansやBeans経由でEJBにアクセス
 - ◆ Strutsなどのフレームワークとの連携も可能

- J-Frame Server を使えば、クライアントからの操作なしで、サーバ側からクライアント画面を更新可能。
 - ◆ サーバ情報のリアルタイム表示(株価情報など)
 - ◆ サーバ側の状態の監視(サーバ接続機器の監視など)

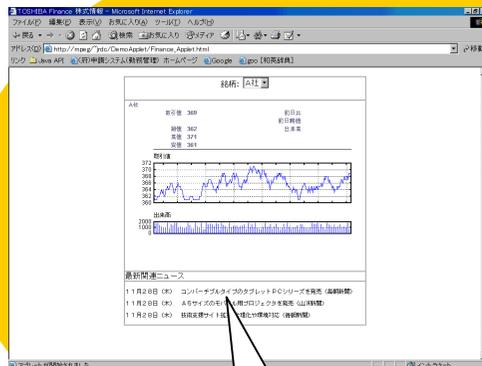
- J-Frame Server を使えば、キャッシュ技術によるAppletの高速起動が可能。
アプレットの1VMマルチアプリによる、効率的なメモリ管理を実現

サーバリソースのアクセス



サーバ側からクライアント画面を更新する例

Collaborative
Innovation



最新の状態に更新するためには、「更新」ボタンを押すなど、クライアント側からの操作が必要。

サーバ側の状態を即時にクライアント側に表示。グラフも刻々と変化

HTMLでは実現できないきめ細やかな操作性を実現！

- 表形式入力(HTML)
- 表形式入力
- 業務アプリ画面(HTML)
- 業務アプリ画面

サーバからクライアントの画面を更新

サーバ側からクライアントの画面を更新。刻々と変化するサーバのデータが、すぐに画面に反映されます。

- 株価表示
- 自動通知(クライアント)
- 自動通知(管理者)

汎用アプリ

最新関連ニュース

- 5月 6日(木) 準天頂衛星システムの事業化検討を行う新会社 (日系新聞)
- 5月 6日(木) 過去2位の好業績、02年9月中旬連続減済で《読売新聞》
- 5月 6日(木) 事業系パソコン再生事業を開始 (山洋新聞)

サーバ側で発生する事象をニュースとしてリアルタイムに通知。

GUI部品で簡単アプリケーション開発



- ビジネスで多用されるGUIを部品として提供。
- GUI部品はプロパティの設定だけのノンプログラミングで動作。
- オフコンやVB画面からWebへの移行が容易。
- フォーカス移動や入力化不可制御など、ストーリー性のあるユーザインタフェースをノンプログラミングで開発可能。
- J2EEとの連携機能をサポート。
- JavaBeansの仕様に準拠。市販のGUIビルダで使用可能。

JBuilderを使った GUI部品プログラミング例



The screenshot shows the JBuilder 9 IDE with a GUI form titled "満期遅れい金支払依頼書詳細入力処理" (Detailed Input Processing for Late Payment Request Form). The form contains various input fields and buttons. A red circle highlights the "OK" button in the form's toolbar. Another red circle highlights the "name" and "constraints" properties of the "FLD_P03_SYOKENNO" component in the Properties window. A third red circle highlights the "components" list in the Properties window, which includes "FLD_P03_SYOKENNO".

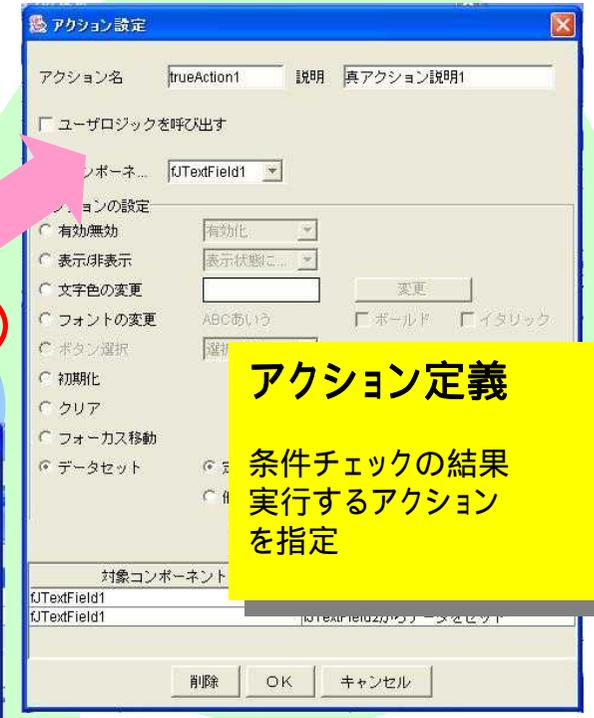
J-Frame Serverが提供するGUI部品

部品の動きを設定
かな漢字ON
編集タイプ
ガイダンス用文字列
フォーカス移動順序
etc

パネル設定によるイベントアクション定義



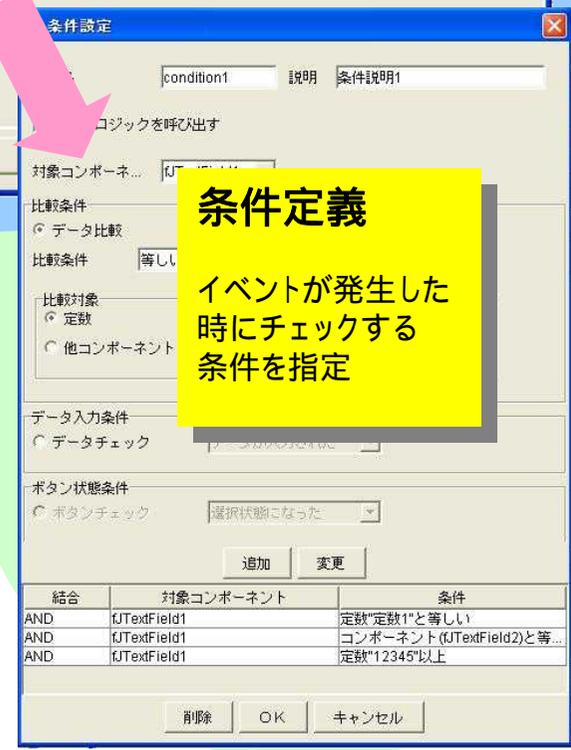
イベント定義
 イベントの種類と監視対象コンポーネントを指定



アクション定義
 条件チェックの結果実行するアクションを指定

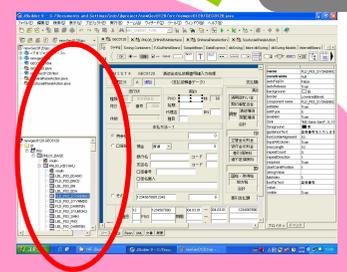
イベントアクションの設定

コンポーネント名	イベント	チェックする条件	真の場合に実行するアクション	偽の場合に実行するアクション
FLD_P10_GINKOCD	フォーカスを失う	GINKOCDCheck	GINKOCDSelected	GINKOCDInputAgain
FLD_P10_SHITENCD	フォーカスを失う	SHITENCDCheck	SHITENCDSelected	SHITENCDInputAgain
RBN_P09_JISAN	ボタンの状態が変化した	JISANChanged	JISANSelected	なし
RBN_P10_FURIKOMI	ボタンの状態が変化した	FURIKOMIChanged	FURIKOMISelected	なし
RBN_P11_SONOTA	ボタンの状態が変化した	SONOTAChecked	SONOTASelected	なし



条件定義
 イベントが発生した時にチェックする条件を指定

カスタマイザにより設定開始

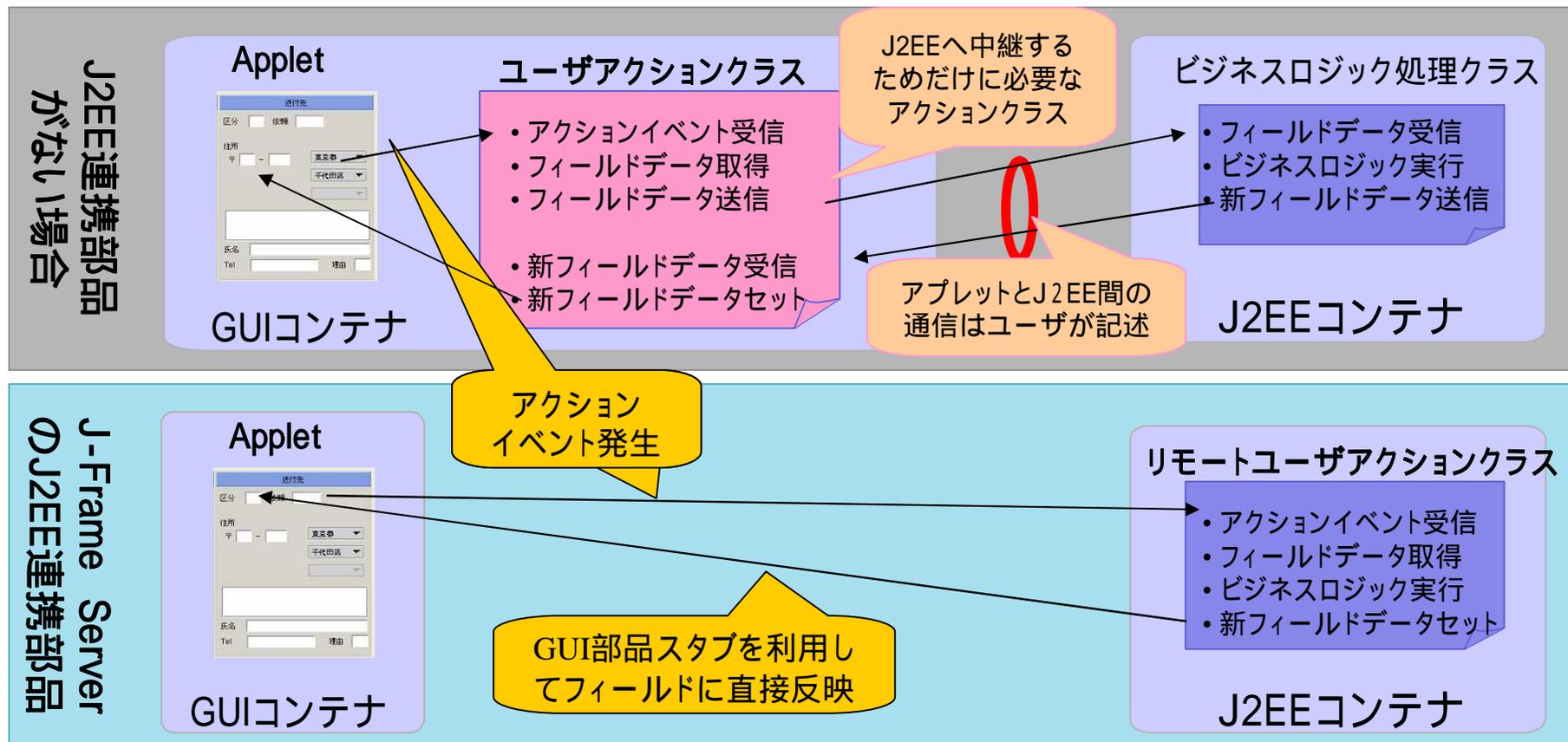


JBuilderのカスタマイザから表示される設定画面を使って、ノンプログラミングでアクションを定義

J2EEアプリケーションとの連携

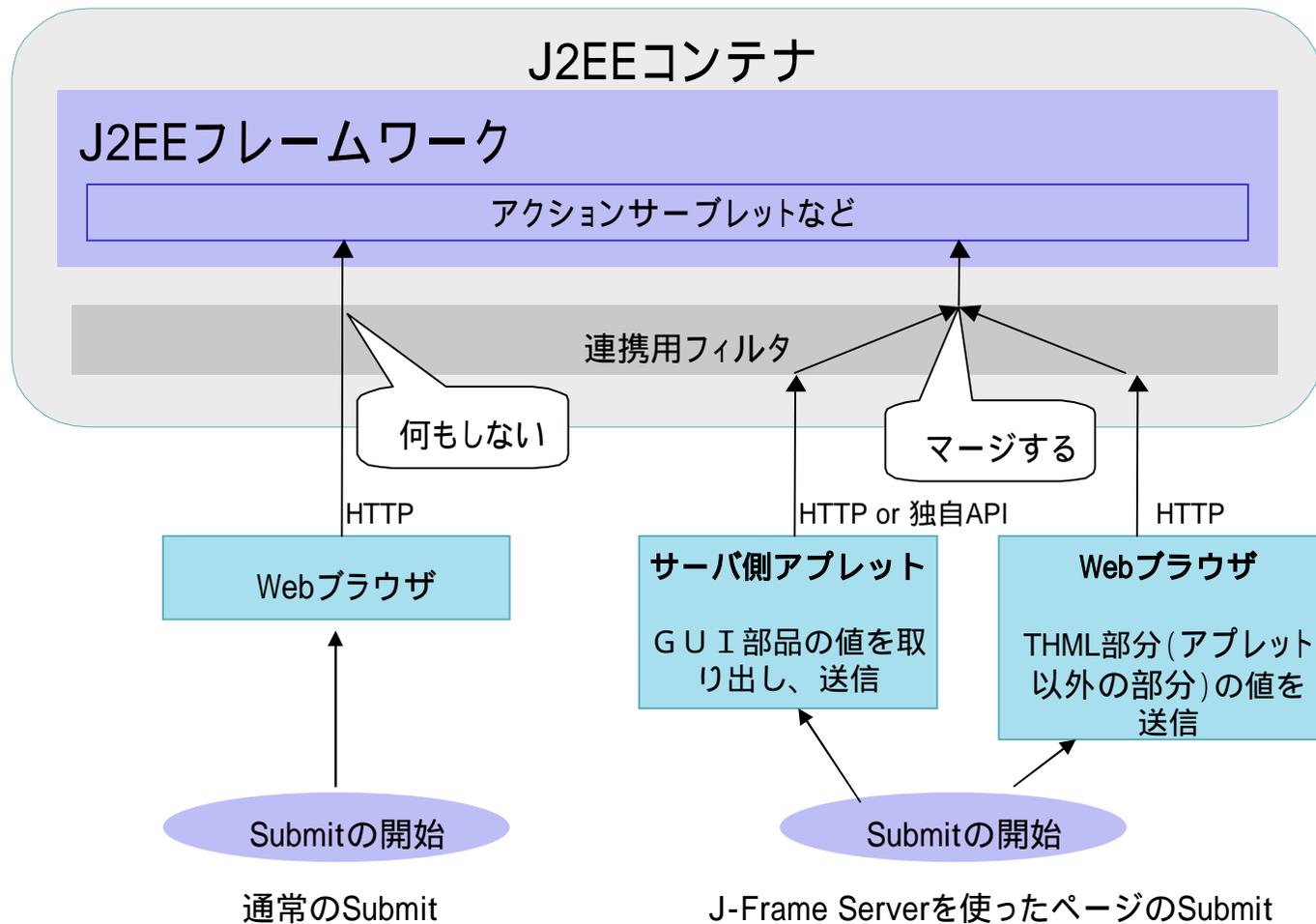
J2EE連携部品でJ2EEアプリケーションと簡単連携

- アプレットのイベントで、J2EE側のアクションを直接呼び出し
- アプレット - J2EE間の通信処理不要
- Applet側のデータをJ2EEコンテナで取り出し、J2EEアプリケーションからAppletへ処理結果をセット



J2EEフレームワークとの連携(1)

- フィールド情報マージ機能
 - HTMLとAppletが混在画面した画面データの一括Submit



J2EEフレームワークとの連携(2)

- ブラウザの「戻る」、「進む」のサポート
 - ページ遷移時、GUI部品データをJ2EEのセッション上への保存
 - 同じページを表示した際、セッションからデータをロード
 - アプレットとのinit()の最後にpanelのinit()、Appletのstop()にpanelのstop()を呼び出す。

```
public class ReloadApplet extends JApplet {  
    private FJSmartPanel panel;  
    public void init() {  
        panel.init();  
    }  
    public void stop() {  
        panel.stop();  
    }  
}
```

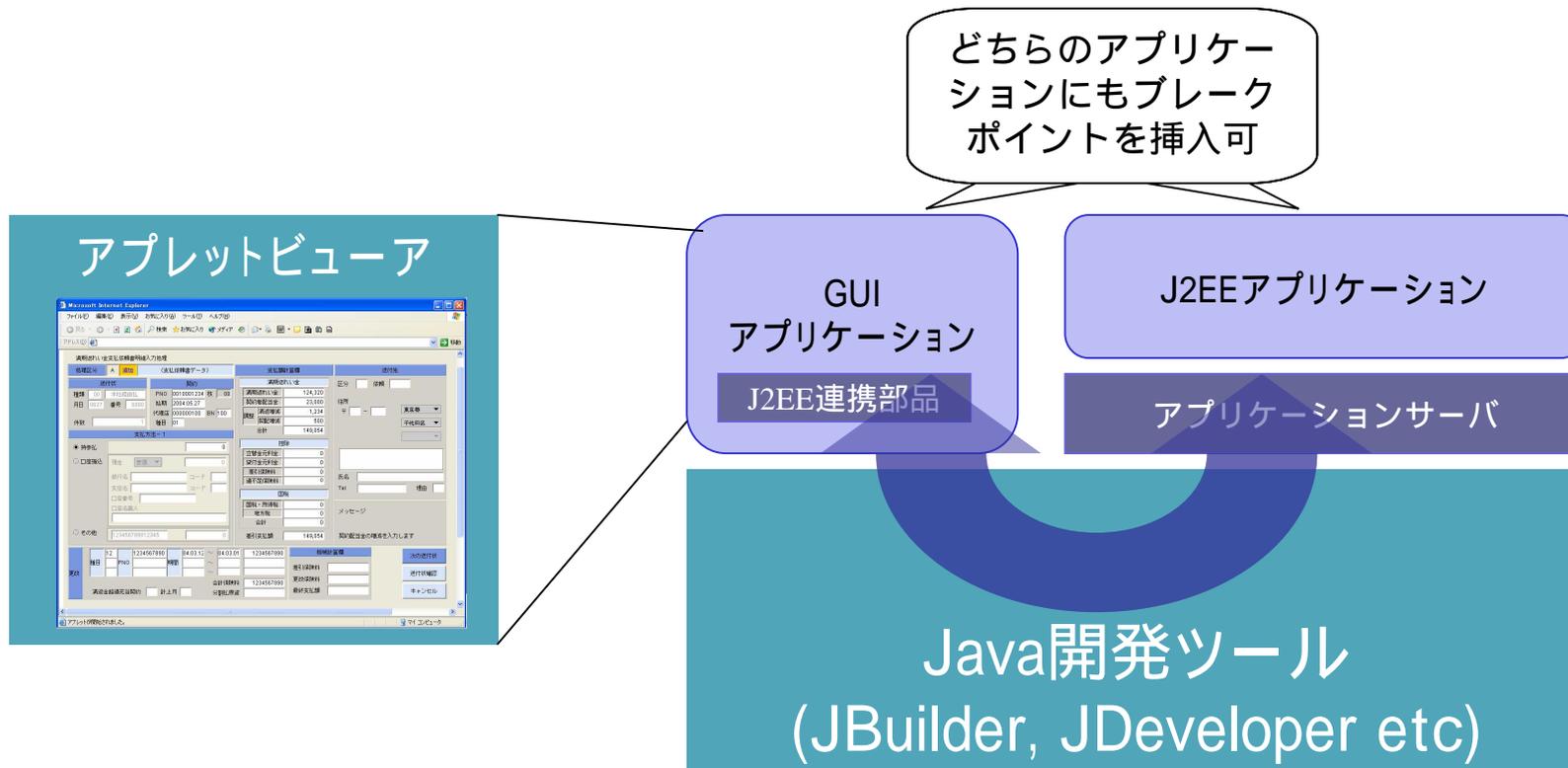
セッションからデータ
を取り出し、GUI部
品にセット

パネル上のGUI部品
のデータを取り出し、
セッションに保存

Java開発ツールを使ったGUI-J2EEの一貫開発



- 標準的なJava開発ツールを使ってGUIプログラム-J2EEプログラムの一貫開発を実現
- これはJ-Frame Server特有の機能ではなく、元々Javaが持っていた能力！



Eclipse+Tomcatとの組み合わせデバック画面



アプレットビューア: fjsample.SampleApp.class

アプレット

満期返れい金支払依頼書明細入力処理

処理区分 A 追加 (支払依頼書データ)

支払額計算欄

送付先

送付状

種類 00 本社経由払 PNO 0501080001 枚 00

月日 0627 番号 0000 開始日 2005.01.05

満期増減 0 区分 依頼

住所 東京都

代理店 N129 BN 21

種目 3

支払方法

持参払

口座振込

預金 普通 0

銀行名 1234

支店名 1234

口座番号

口座名義人

その他 123456789012345 0

立替金元利

貸付金元利

差引保険料

過不足保険料

国税・所得

地方税

合計

差引支払額

12 1234567890 04.03.12 ~ 04.03.01 1234567

満返金超過充当契約 計上月 合計保険料 1234567

分割払原資

アプレットが開始されました。

デバッグ - BankBean.java - Eclipse Platform

ファイル(E) 編集(E) ソース(S) リファクタリング(I) ナビゲート(N) 検索(A) プロジェクト(P) Tomcat 実行(R) ウィンドウ(W) ヘルプ(H)

スレッド [FJletManager-0] (実行中)

スレッド [FJletSessionManager] (実行中)

C:\jdk1.3.1_03\bin\javaw.exe (2005/01/06 17:31:46)

J2EE 1.4 [Java アプリケーション]

ローカル・ホスト上の sun.applet.AppletViewer:2484

スレッド [AWT-Shutdown] (実行中)

スレッド [AWT-Windows] (実行中)

スレッド [thread applet-fjsample.SampleApp.class] (実行中)

スレッド [Java2D Disposer] (実行中)

スレッド [AWT-EventQueue-0] (実行中)

スレッド [DestroyJavaVM] (実行中)

ブレークポイント

BankBean [行: 7] - getBankName(int)

BankBean [行: 9] - getBankName(int)

SoufusakiPanelAction [行: 116] - setTownCombo_performed(FJActionEvent)

変数 ブレークポイント

```
package fjsample.J2ee;

public class BankBean {
    public static String getBankName(int code) {
        System.out.println("銀行名の検索開始: code="+code);
        String name;
        switch (code) {
            case 1234:
                name = "東芝銀行";
                break;
            case 1111:
                name = "Flyingserv銀行";
                break;
            default:
                name = "";
        }
        System.out.println("銀行名の検索終了");
        return name;
    }

    public static String getBranchName(int branchcode) {
        System.out.println("支店名の検索開始");
        String name;
        switch (branchcode) {
            case 111:
                name = "";
        }
    }
}
```

コンソール [J2EE 1.4 [Java アプリケーション] C:\Program Files\Java\j2re1.4.2_05\bin\javaw.exe (2005/01/06 17:32:52)]

コンソール/タスク

銀行コード入力したところでストップ (J2EE側ロジック)



■サーバ(アプリケーション実行側)

対応Javaバージョン JDK 1.3.1、JDK 1.4.2
OS WindowsNT4.0、Windows2000
Windows2003,Solaris 8、9
Linux (RedHat LINUX 7.3J,MIRACLE LINUX 2.1)
必要メモリ 最小256MB以上 推奨 512MB以上
必要ディスク容量 40MB以上
対応アプリケーションサーバ

IBM WebSphere Application Server 5.0、
BEA WebLogic Server 8.1、
Sun Java System Application Server 7.0、
Oracle Application Server 10g v9.0.4、
Apache Tomcat 4.1

■クライアント(表示側)

OS Windows98、WindowsNT4.0、Windows2000、Windows XP
必要メモリ 最小32MB以上 推奨 64MB以上
必須ソフトウェア JRE1.3.1(JavaPlug-In 1.3.1) サーバ側がJDK1.3.1の場合のみ
JRE1.4.2(JavaPlug-In 1.4.2)
ブラウザ Microsoft Internet Explorer 5.5,6.0
Netscape Navigator 4.7,7.x

■GUIビルダ(GUI部品)

Borland JBuilder 9、X、2005
Oracle JDeveloper 10g
(Eclipse J2EE連携による開発のみ(Visual Builderは除く))

適用事例

本製品の適用事例

株式会社 TKC様における適用事例-



株式会社 TKC様における適用事例 「e-TASK財務会計マスター」

■ システムの概要

地方公共団体向けの電子決裁対応財務会計システム

■ 導入規模

2004年4月現在67サイト、3150クライアントライセンス

■ システムの要求

(a) 現行システムの問題点の解消

- 低速回線におけるクライアントソフトウェア配布時間大

(b) 現行システムのユーザインターフェースをキープ

- 実用に耐え得るアプリケーション起動時間
- 複雑な帳票系GUI

e-TASKは株式会社TKCの登録商標です。

本製品の適用事例

株式会社 TKC様における適用事例-



アプリケーション画面

メニュー画面

入力画面

各課メニュー (第2002.02版)

e-TASK 財務会計マスター

本日 平成14年 3月12日(火)

【歳出伝票入力】 負担行為兼支出命令 (単件)
平成13年度 総務部総務課: 鈴木 一郎

本日:平成14年 3月12日(火)
前回の管理番号

伝票入力・照会 予算編成・管理 決算 その他

総務部総務課:
鈴木 一郎

平成13年度

● 歳出伝票入力 ● 歳出照会・印刷

- 11 支出負担行為
- 12 支出負担行為変更
- 15 支出命令
- 16 支出負担行為(集合)
- 17 支出負担行為(集合)変更
- 18 支出命令 (集合)
- 19 負担行為兼支出命令(単件)
- 20 // (集合)
- 21 // (旅費)
- 22 // (給与)
- 23 戻入
- 24 資金前渡・概算払 (精算)
- 25 歳出更正振替伺
- 26 公金振替伺
- 41 予算執行状況(全)
- 42 予算執行状況(課)
- 43 科目別執行一覧
- 44 債権者別執行一!
- 45 精算状況
- 46 個別伝票
- 47 債権者
- 48 予算差引簿
- 49 予算執行状況明
- 50 債権者別執行一!
- 51 支出負担行為未
- 52 精算状況一覧表
- 53 歳出伝票一覧表
- 予算流用・充用
- 81 予算流用伺
- 82 予備費充用伺

管理番号 [] 簡略コード 000006 整理番号 325 専決区分 []

会計 001:一般会計 予算区分 1:現年

所属課 00 00 01 00 - 0000 議会事務局
 款項目 01 01 01 議会費
 事業 001 00 00 0 議会費
 節細節 01 01 議員報酬
 細々節 001 議員報酬

内容

請求書番号 [] 摘要 [] 計 金額 []

支出区分 0:通常払 課税区分 3:不課税

消費税 0
合計 0
控除額 []
差引額 0
配当済額 37,677,000
負担済額 24,011,209
配当残額 13,665,791
命令済額 23,962,341
予算残額 13,774,659
伝票印刷 0:有 印刷枚数 [] 枚

債権者 []

支払方法 1:口座振込 口座区分 1:第1口座

振込先 [] 通知有無 0:有 通知様式 0:ハガキ

添付資料 []

承認待ち: 76 件
差戻し: 0 件
供覧: 0 件 最新表示

お知らせ
 ■ペーパーレス化
 ○予算執行状況等を確認する場合は、プリンタに印刷し紙で確認することは用紙の無駄になります。
 ○画面上の電子帳票で確認しましょう。

Copyright (C)2000 TKC Corporation All Rights Reserved.

(存置項目: 全項目 科目 日付 内容 債権者等)

登録 伝票複写 摘要別録入力 債権者自由入力 摘要検索 イメージ読込 入力補助 終了 挿入

商品紹介HomePage:

[http:// flyingerv.toshiba-sol.co.jp](http://flyingerv.toshiba-sol.co.jp)

JavaおよびすべてのJava関連の商標およびロゴ及びSolarisは、米国および他の国における米国Sun Microsystems, Incの商標または登録商標です。

Microsoft、Windows、WindowsNTは、米国Microsoft Corporationの米国及びその他の国における登録商標です。

NetscapeおよびNetscape Navigatorは、Netscape Communications Corporationの商標または登録商標です。

OracleとOracleのロゴは、Oracle Corporationの登録商標です。Oracle10g、Oracle10g Application Serverは、Oracle Corporationの商標です。

IBMとWebSphereはIBM Corporationの商標です。

BEAおよびBEA WebLogicはBEA Systems, Inc.の登録商標です。BEA WebLogic ServerはBEA Systems, Inc.の商標です。

BorlandブランドおよびJbuilderは、Borland Software Corporation(旧社名Inprise Corporation)の米国またおよび他国における商標または登録商標です。

Macromedia およびFlashは、Macromedia, Inc. の米国およびその他の国における商標または登録商標です。

Adobe、Adobe ロゴ、Readerは、Adobe Systems Incorporated(アドビシステムズ社)の米国ならびに他の国における商標または登録商標です。

その他の社名、製品名は、それぞれ各社の商標または登録商標です。

本資料に掲載の商品名称は、それぞれ各社が商標として使用している場合があります。

本内容は変更する場合があります。詳細は、仕様書あるいは説明書をご覧ください

TOSHIBA



東芝ソリューション株式会社
TOSHIBA SOLUTIONS CORPORATION

プラットフォームソリューション事業部