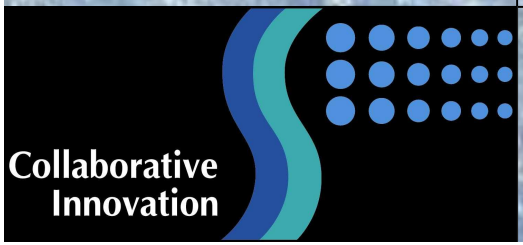
TOSHIBA

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



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

Agenda

Collaborative Innovation

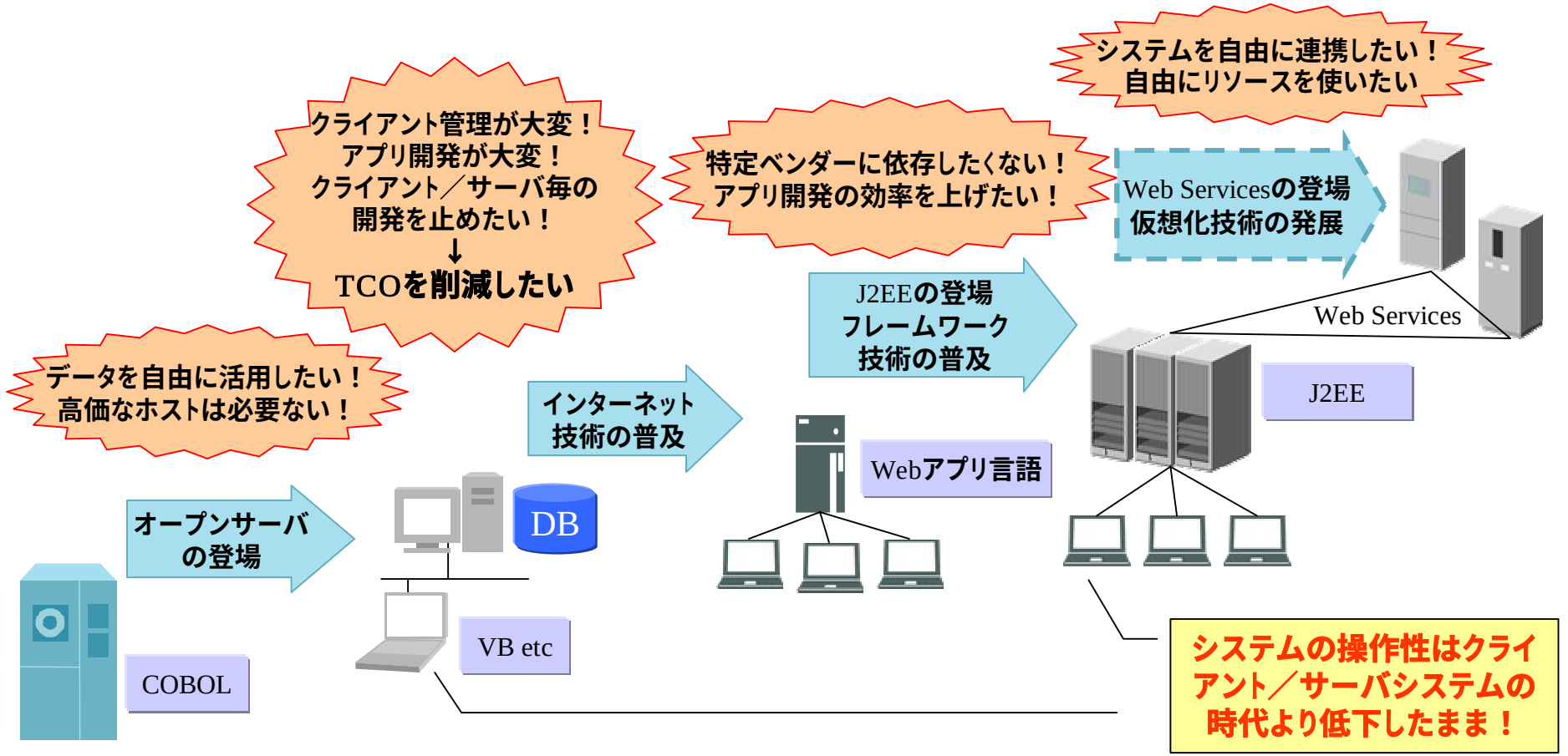
- 背景
  -J2EE環境におけるインタフェースの課題-
- リッチクライアント採用の課題
- GUIコンテナという考え方
  (*FlyingServ* J-Frame Serverのご紹介)
  - 標準Java環境をサポートしたGUIコンテナ
  - Appletがサーバで動く事の意味
  - GUI部品によるアプリケーション開発
  - J2EEアプリケーションとの連携
  - J2EEフレームワークとの連携
  - Java開発ツールを使ったGUI-J2EEアプリの一貫開発
- 活用事例

Collaborative Innovation

背景
-J2EE環境におけるインタフェースの課題-

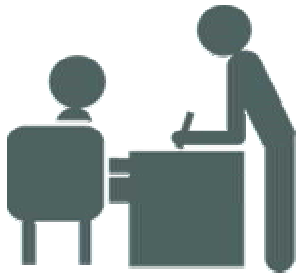
標準プラットフォームの遷移

Collaborative Innovation



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！
- 設計した画面は、⇒ 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)

Collaborative Innovation

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

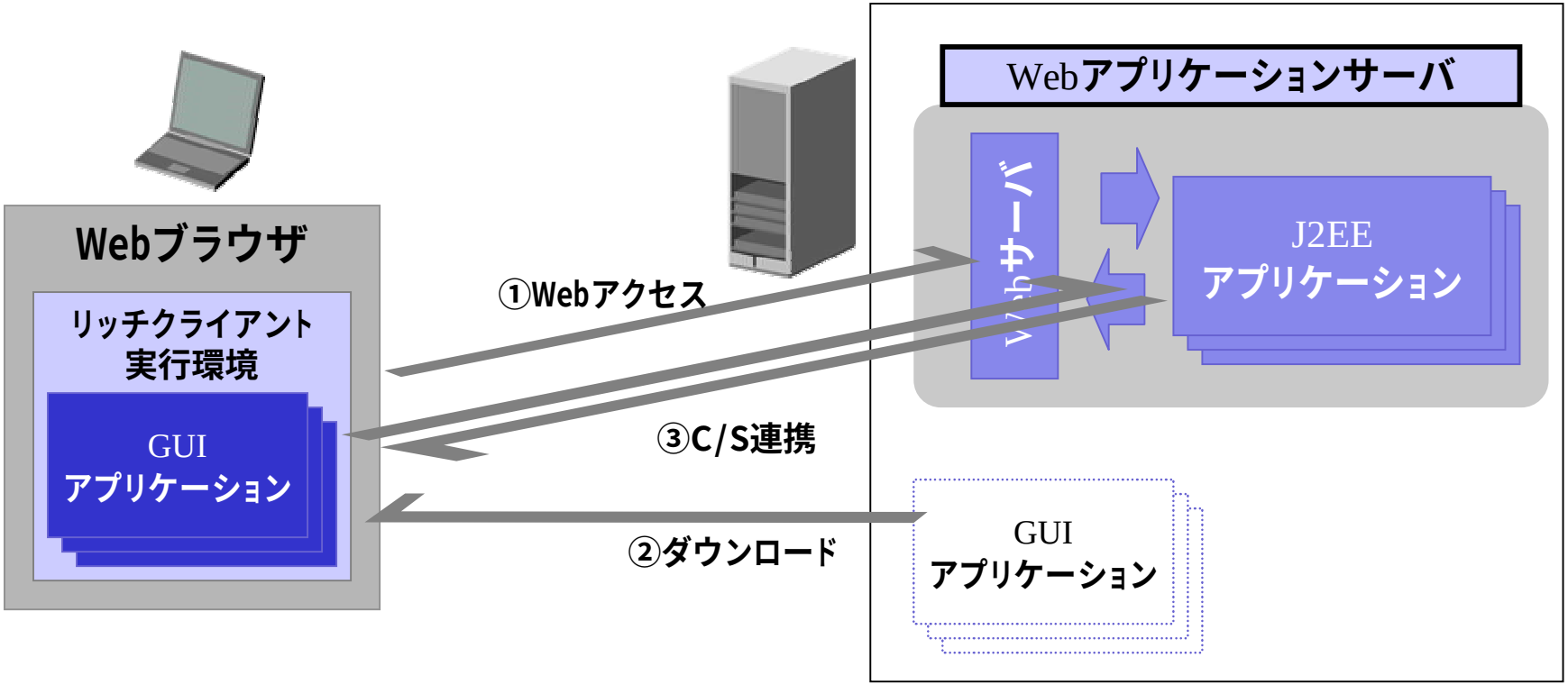
Collaborative Innovation

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

Collaborative Innovation

- 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

Collaborative Innovation

- 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・・

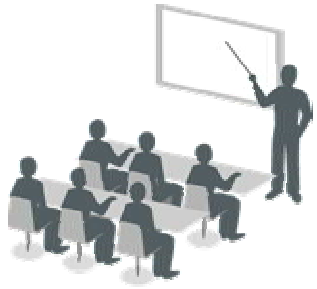
東芝ソリューションの考える
ＪＡＥＥの為のリッチクライアントとは？

Collaborative Innovation

- 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、開発ツールでしばらないこと！

- J2EEを補完し、J2EEの約束事を乱さないこと！！



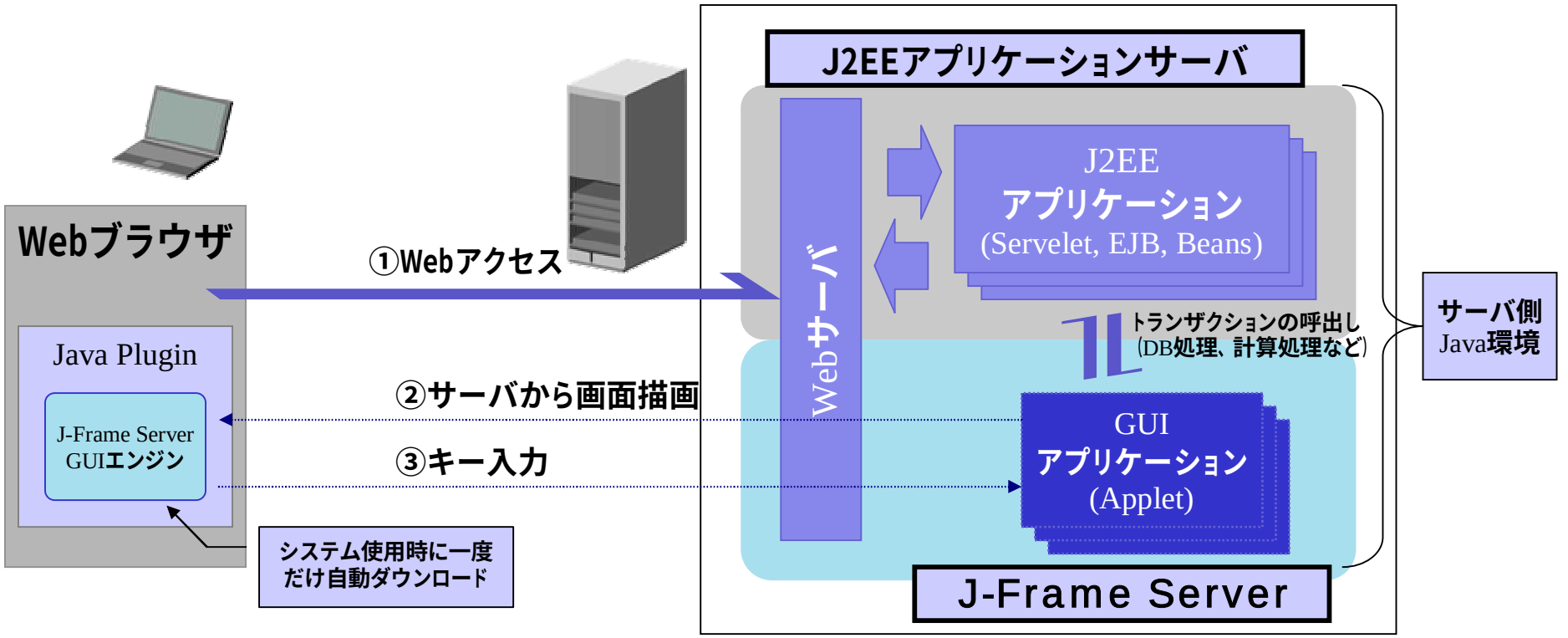
Collaborative Innovation

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

Collaborative Innovation

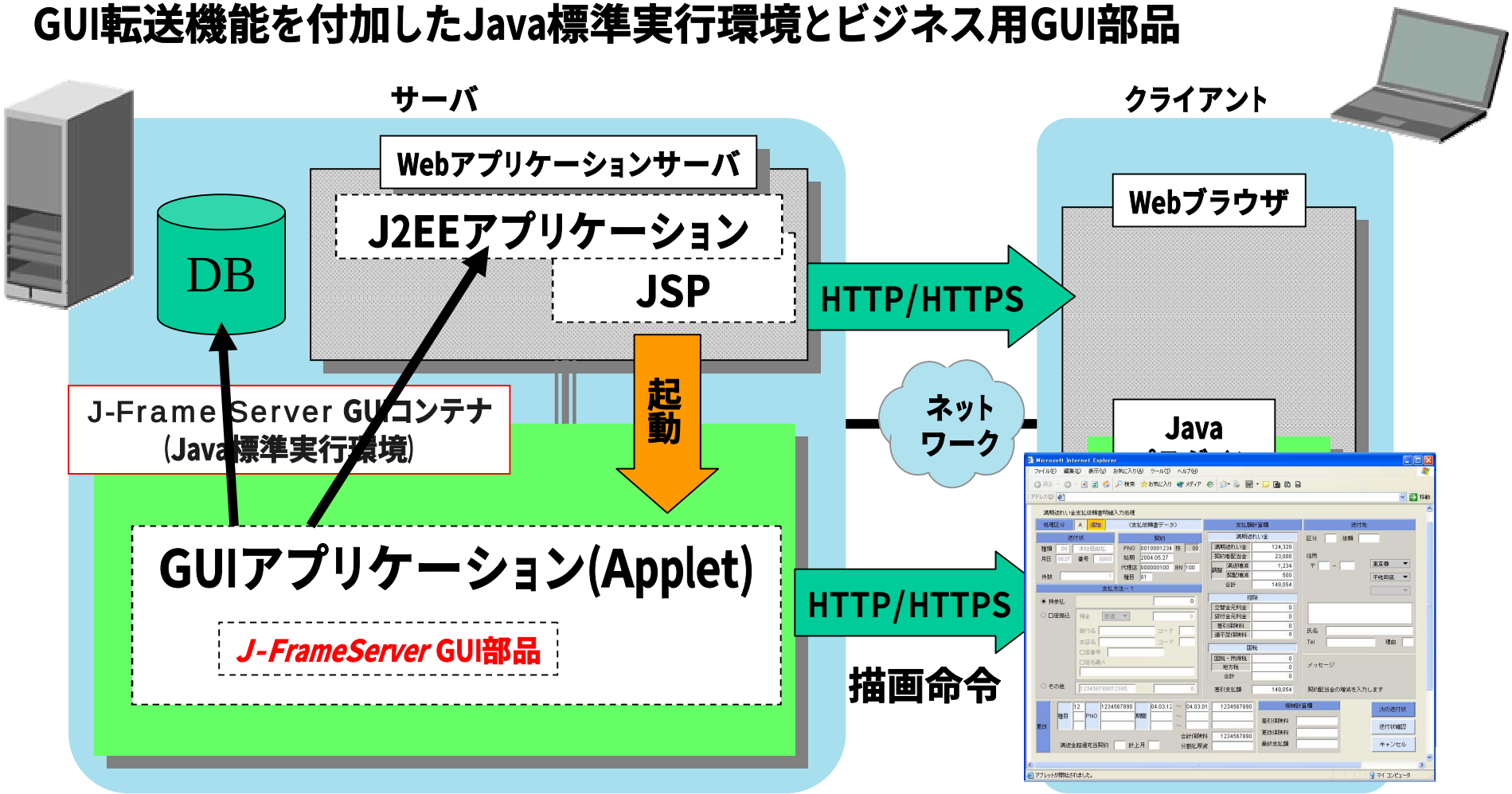
- サーバでGUIアプリケーションを動作させる環境を提供（GUIコンテナ）
- サーバで動作するため、クライアント管理コストの増加を招かない
- サーバ側リソースを自由にアクセス、GUIアプリとJ2EEアプリは、Javaプログラムどうしの連携で一体となって動作
- GUIアプリはJSPより起動、Strutsなどのフレームワークで使用可能
- 標準Java仕様に準拠



FlyingServ J-Frame Serverの仕組み

Collaborative Innovation

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

Collaborative Innovation

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。

Java COMPATIBLE

JSPおよび連携APIの例

Collaborative Innovation

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="fsts"%>
<html>
<head><title>Sample FJlet Page</title></head>
<body>
<fsts: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"fsts"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

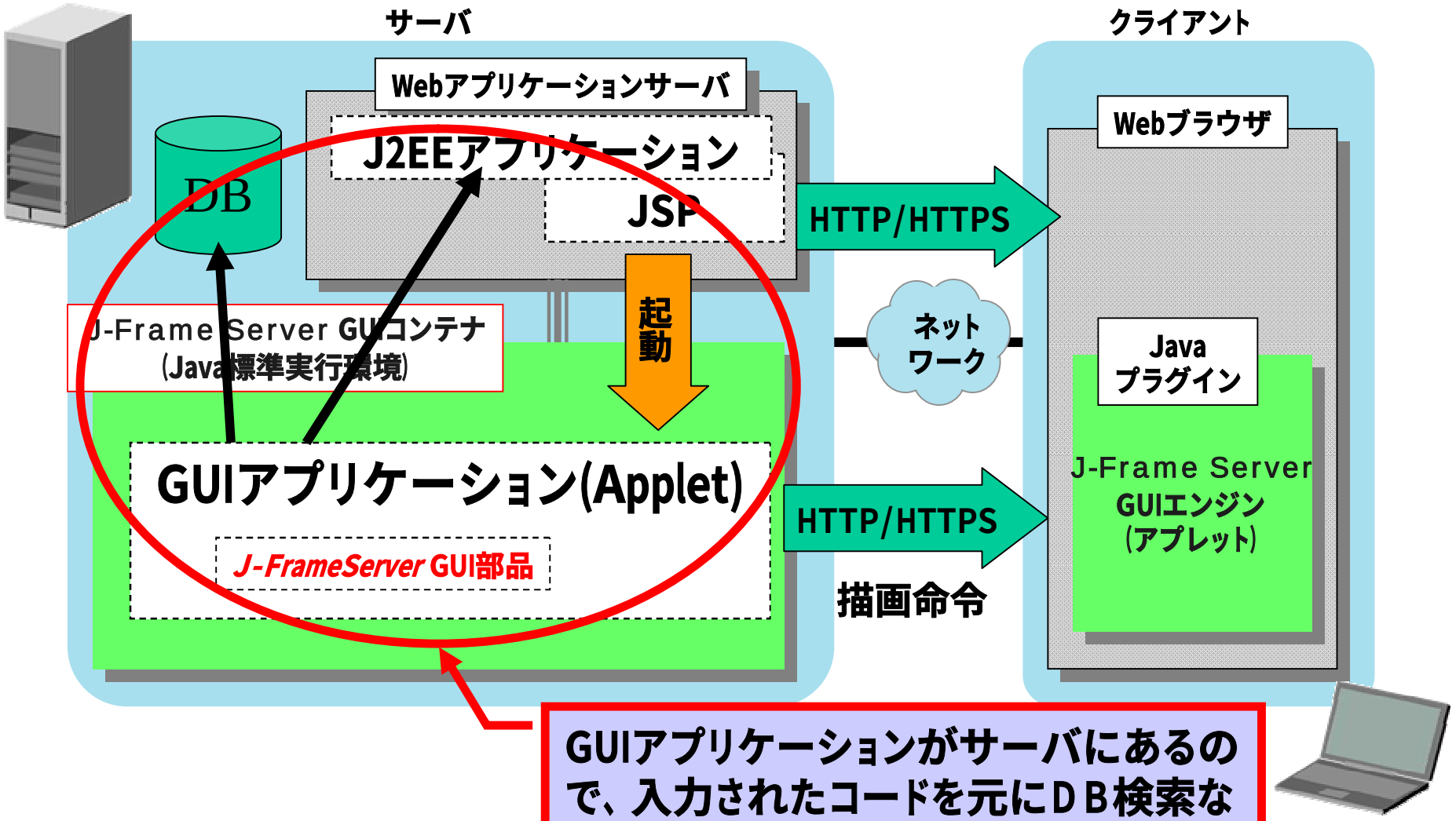
Collaborative Innovation

- 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

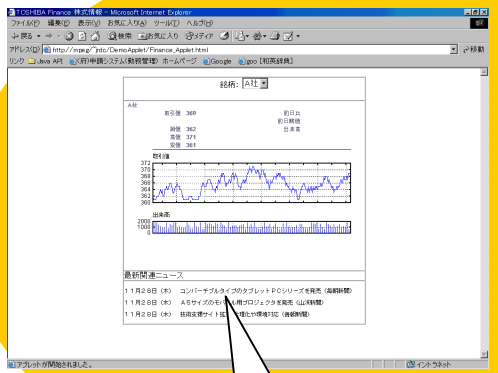
- J-Frame Server を使えば、クライアントからの操作なしで、サーバ側からクライアント画面を更新可能。
  - サーバ情報のリアルタイム表示（株価情報など）
  - サーバ側の状態の監視（サーバ接続機器の監視など）

- 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

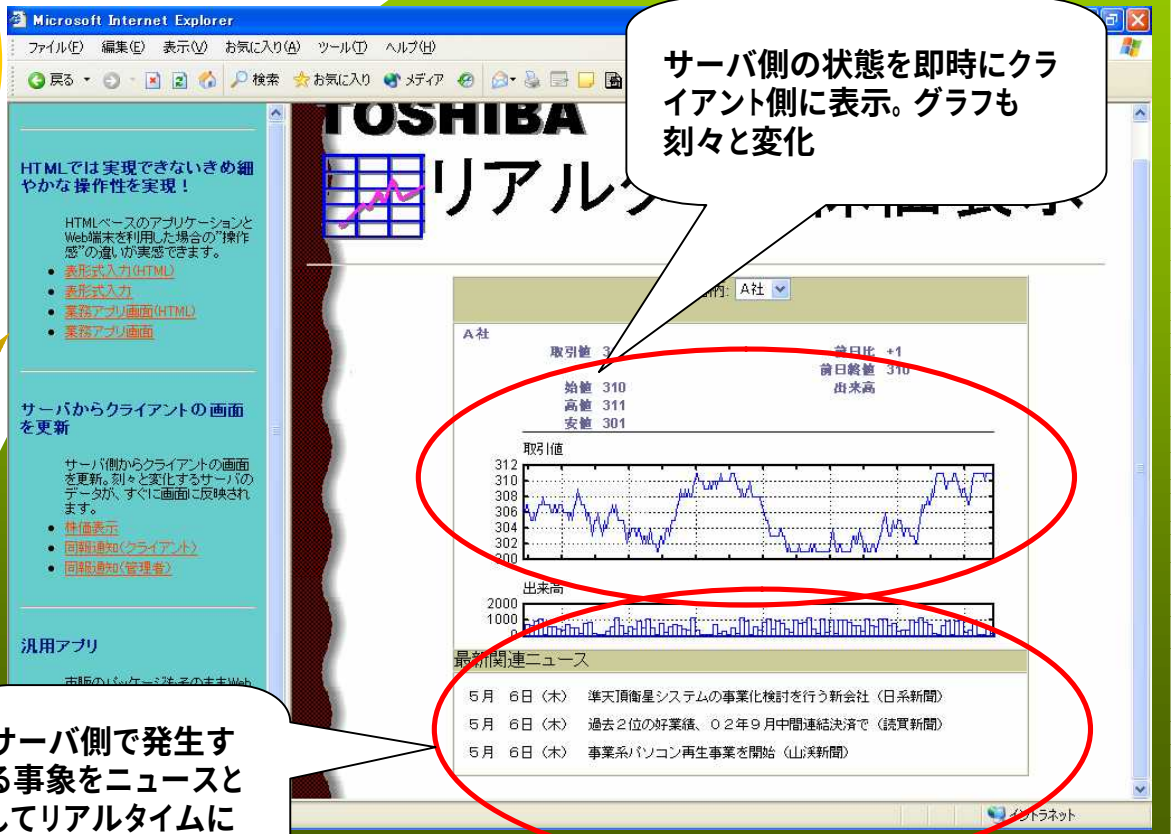
サーバリソースのアクセス



サーバ側からクライアント画面を更新する例



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

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。

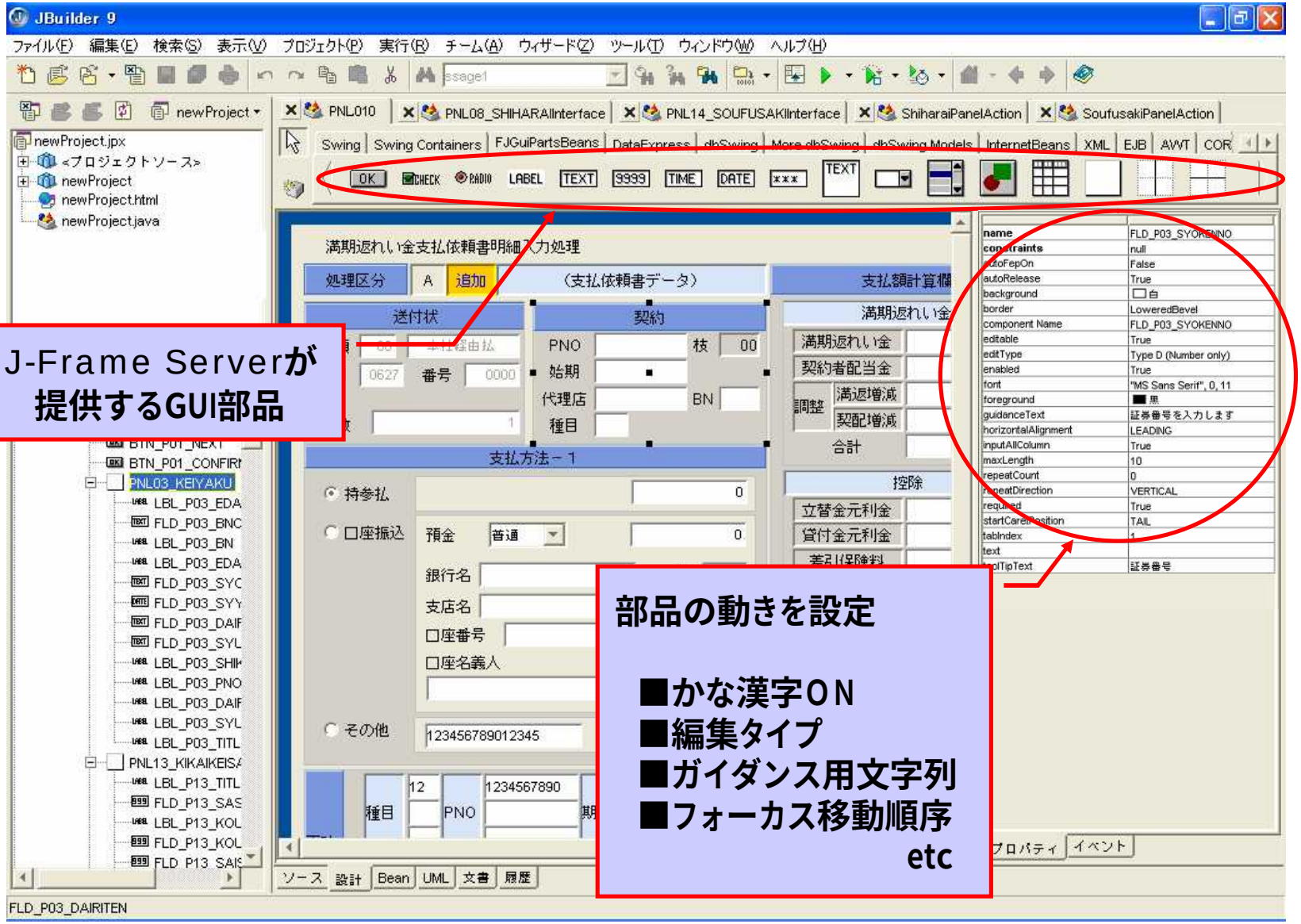
GUI部品で簡単アプリケーション開発

Collaborative Innovation

- 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

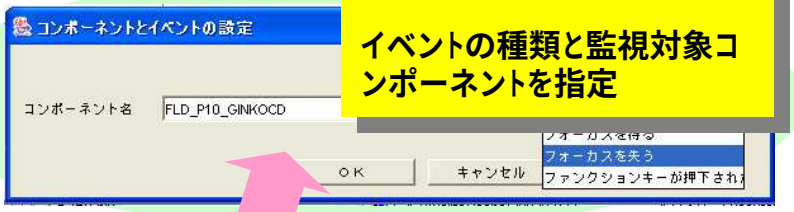
Collaborative Innovation

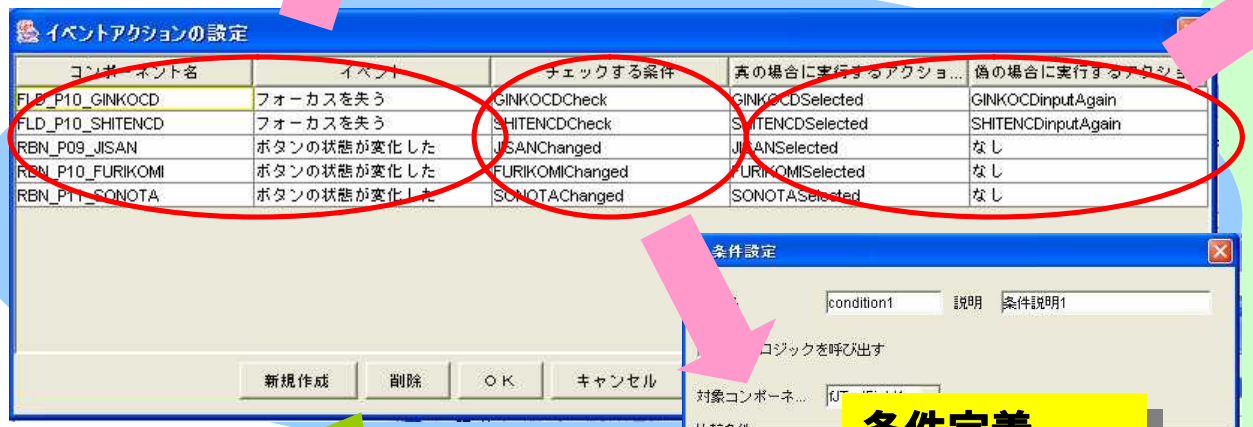


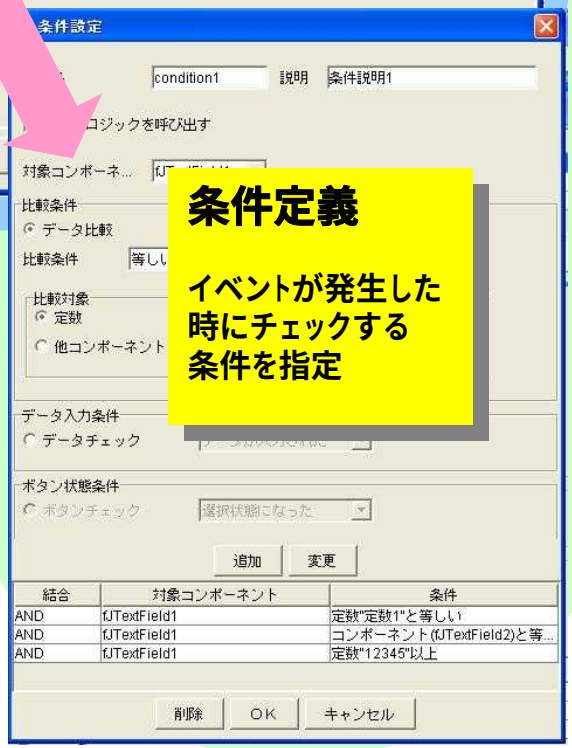
パネル設定によるイベントアクション定義

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

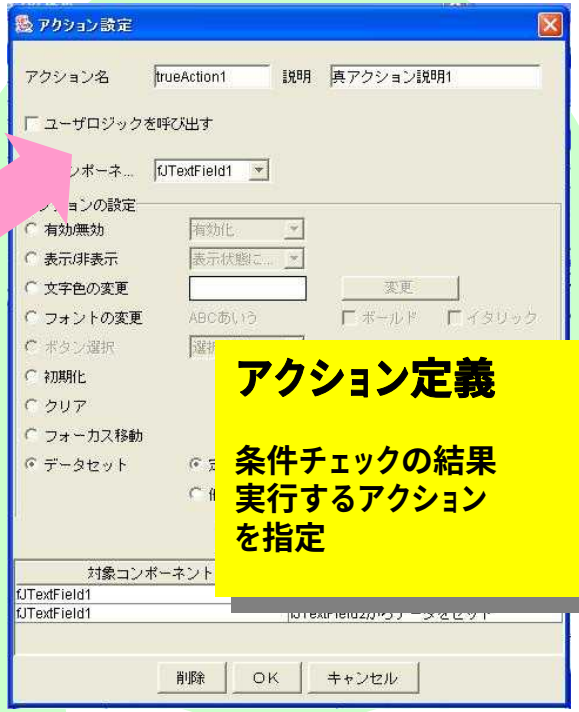






条件定義

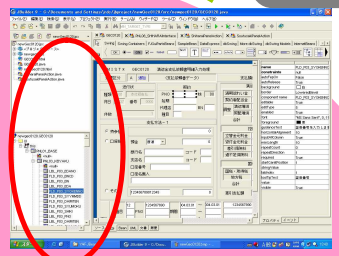
イベントが発生した時にチェックする条件を指定



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

カスタマイザにより設定開始

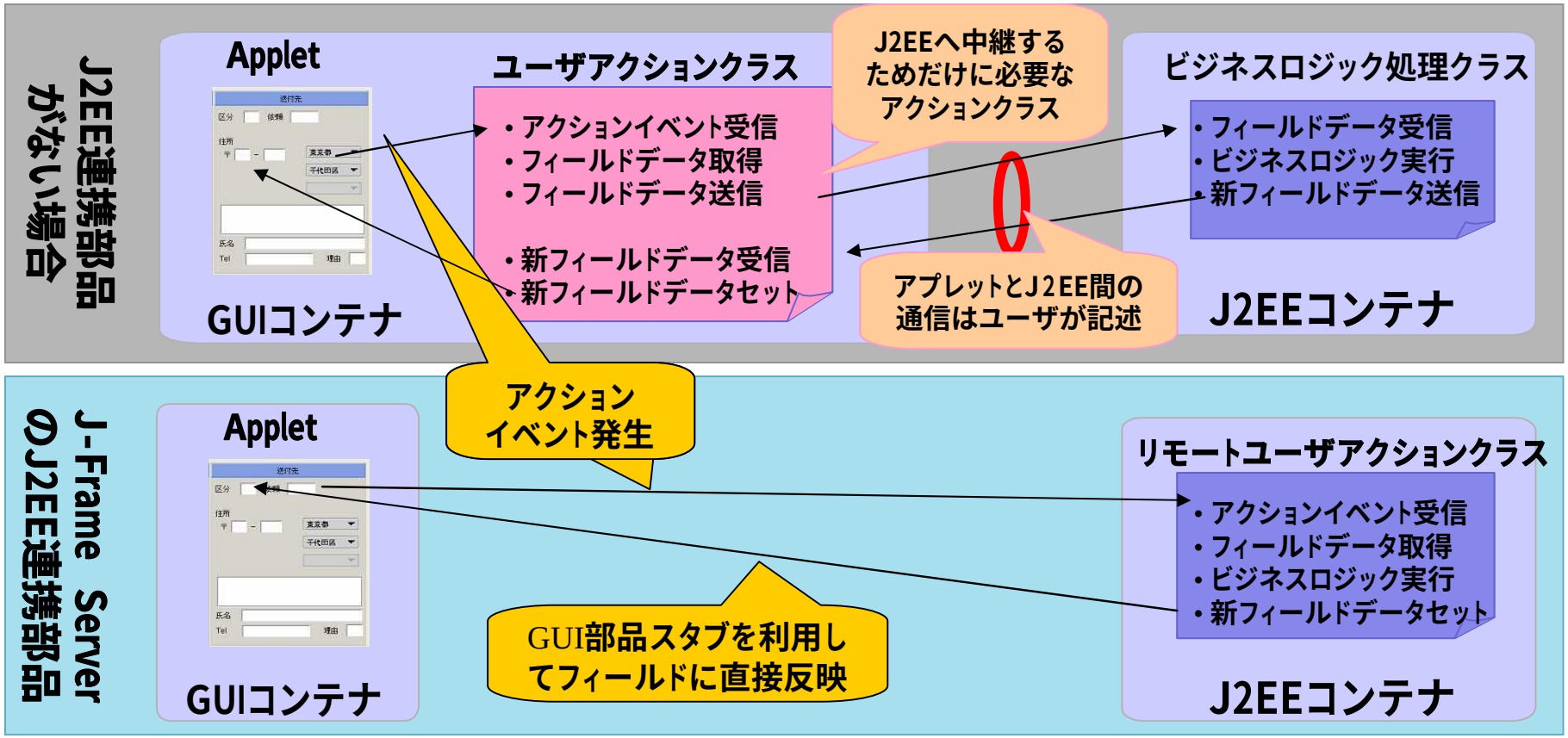


JBuilderのカスタマイザから表示される設定画面を使って、ノンプログラミングでアクションを定義

J2EEアプリケーションとの連携

Collaborative Innovation

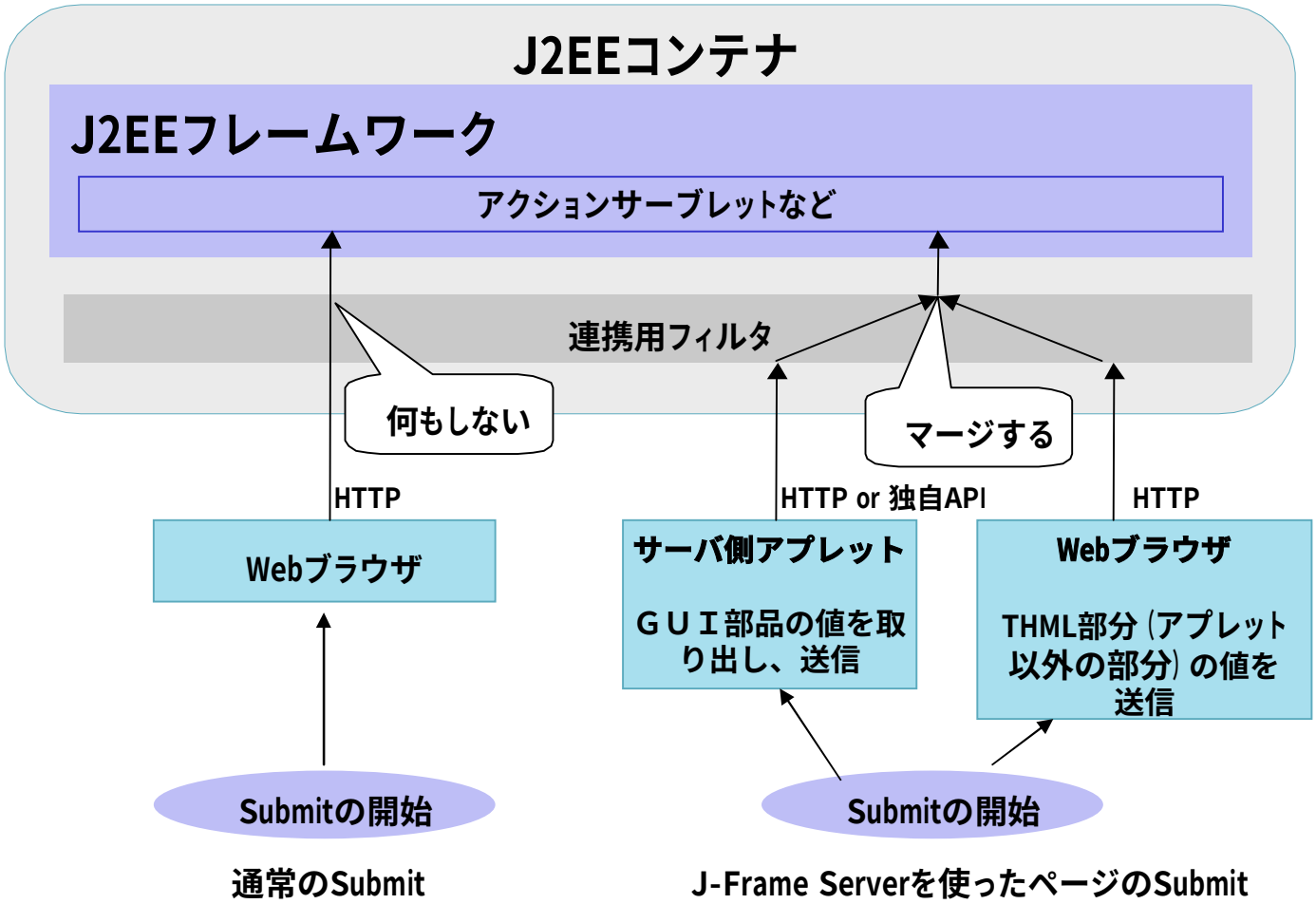
- J2EE連携部品でJ2EEアプリケーションと簡単連携
  - アプレットのイベントで、J2EE側のアクションを直接呼び出し
  - アプレットーJ2EE間の通信処理不要
  - Applet側のデータをJ2EEコンテナで取り出し、J2EEアプリケーションからAppletへ処理結果をセット



J2EEフレームワークとの連携(1)

Collaborative Innovation

- 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)

Collaborative Innovation

- ブラウザの「戻る」、「進む」のサポート
  - ページ遷移時、GUI部品のデータを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。

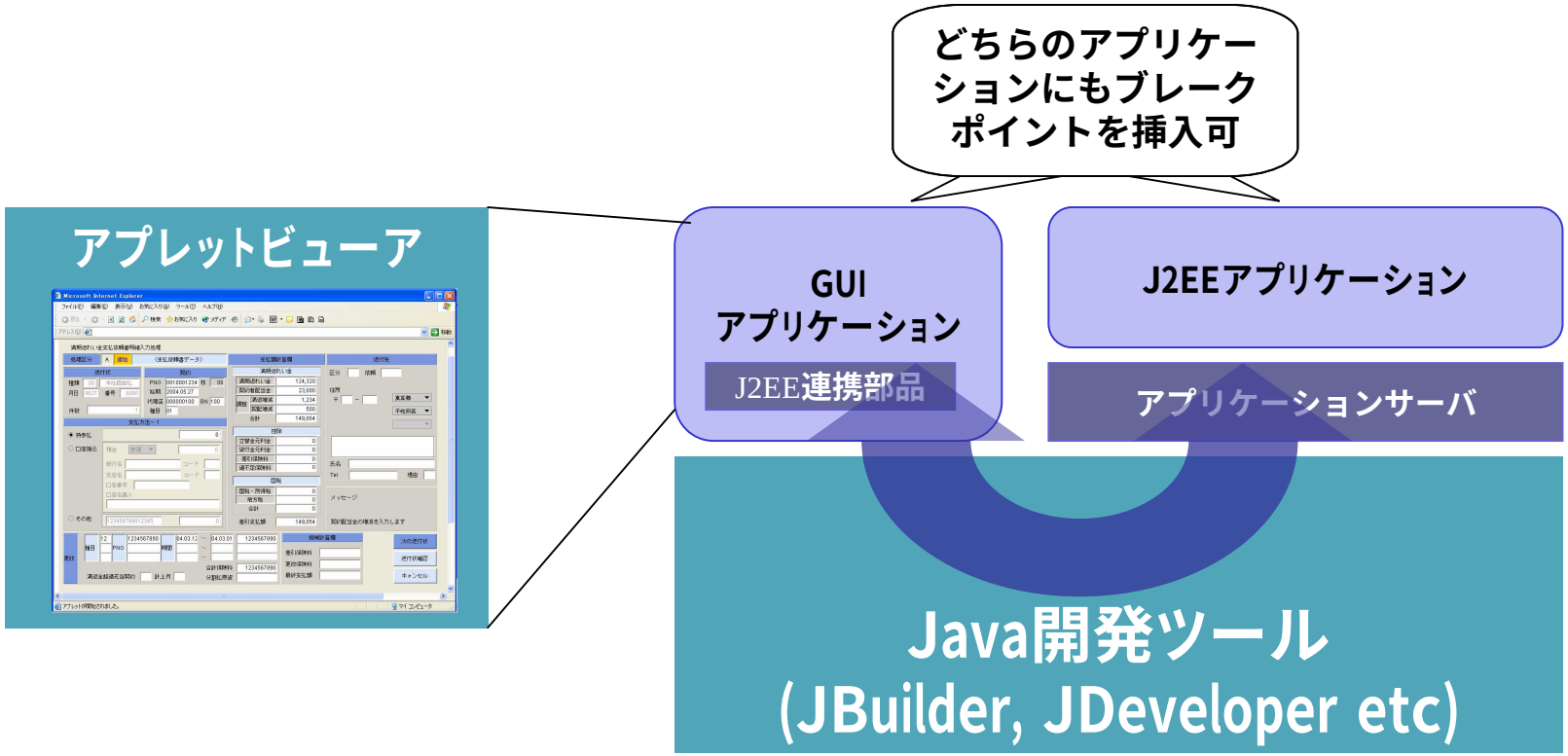
```
public class ReloadApplet extends JApplet {
  private FJSmartPanel panel;
  public void init() {
    panel.init();
  }
  public void stop() {
    panel.stop();
  }
}
```

セッションからデータを取り出し、ＧＵＩ部品にセット

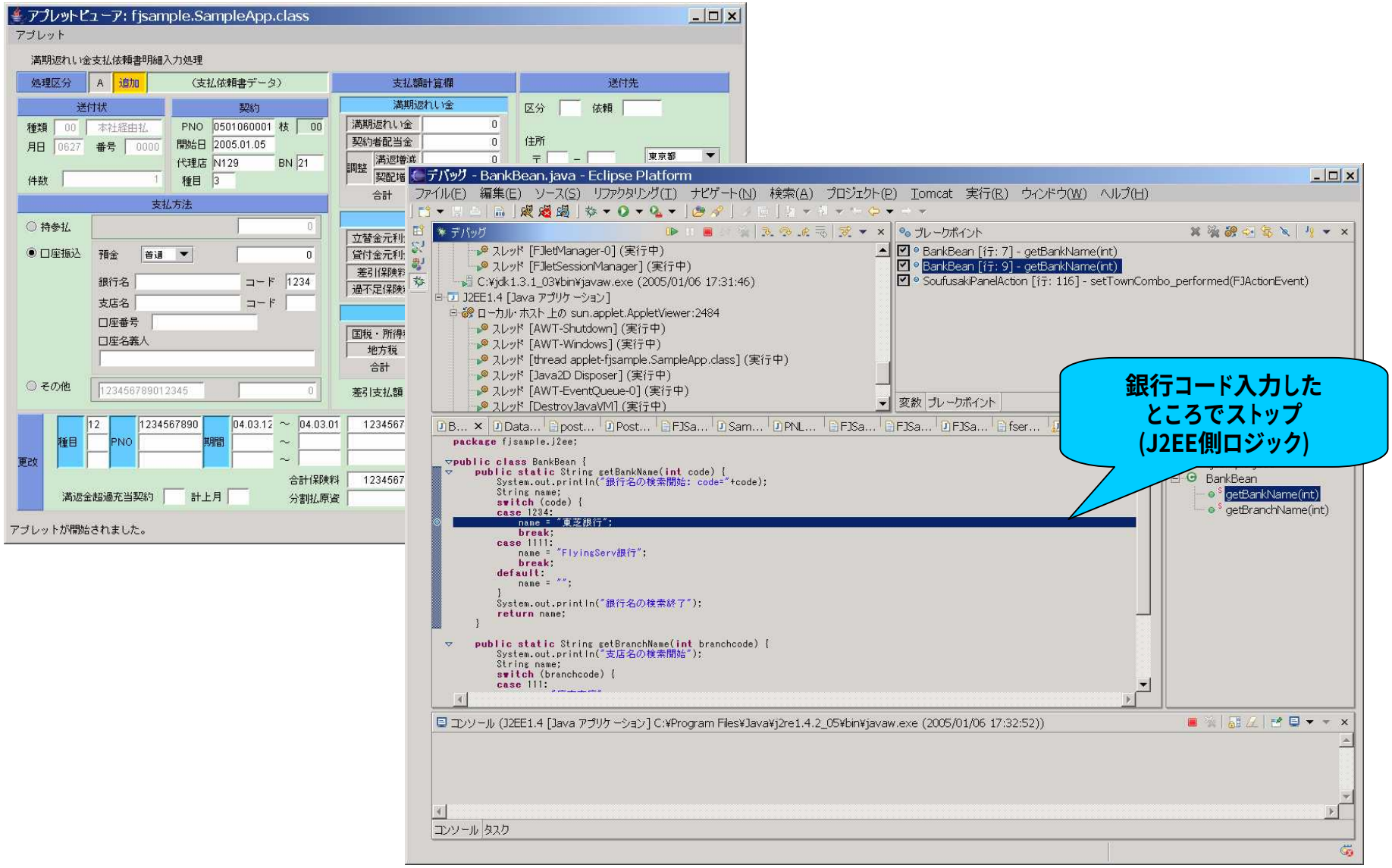
パネル上のＧＵＩ部品のデータを取り出し、セッションに保存

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



FlyingServ J-Frame Server 動作環境

Collaborative Innovation

■サーバ（アプリケーション実行側）

対応Javaバージョン	JDK1．3．1、JDK1．4．2
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)）

Collaborative Innovation

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-

Collaborative Innovation

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

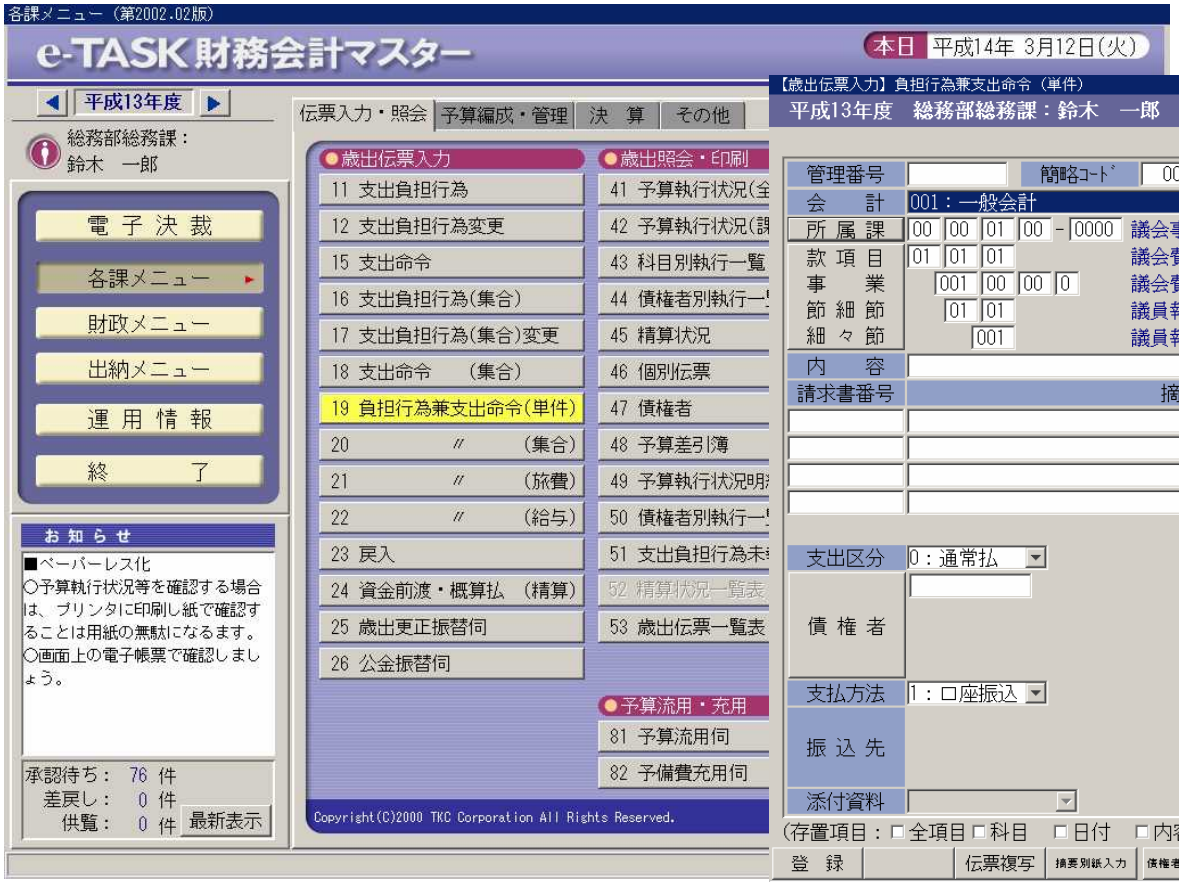
※ e-TASKは株式会社TKCの登録商標です。

本製品の適用事例
株式会社　TKC様における適用事例-

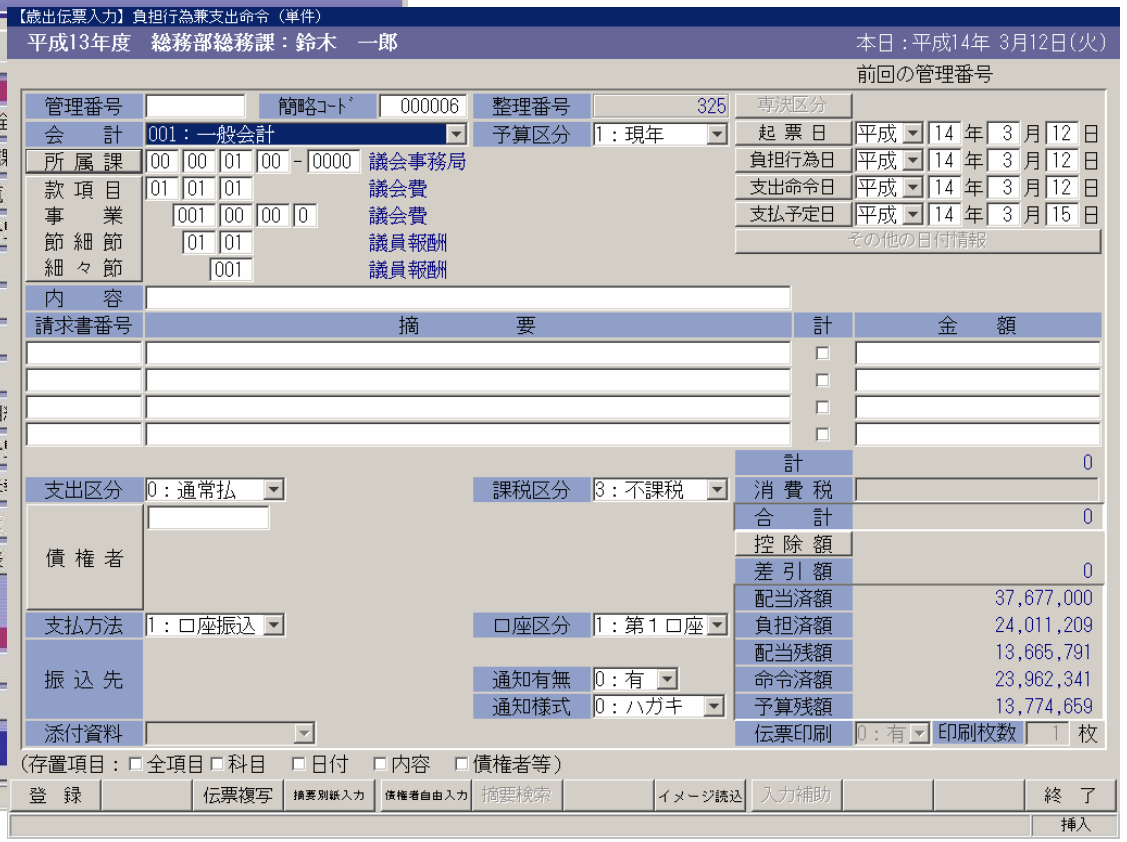
Collaborative Innovation

アプリケーション画面

■メニュー画面



■入力画面



URL

Collaborative Innovation

商品紹介HomePage：

http:// flyingserv.toshiba-sol.co.jp

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