MZ Platform機能概要・リリース計画・MZ Platformを利用した社内IT化の推進

MZプラットフォーム研究会シンポジウム

平成17年7月1日

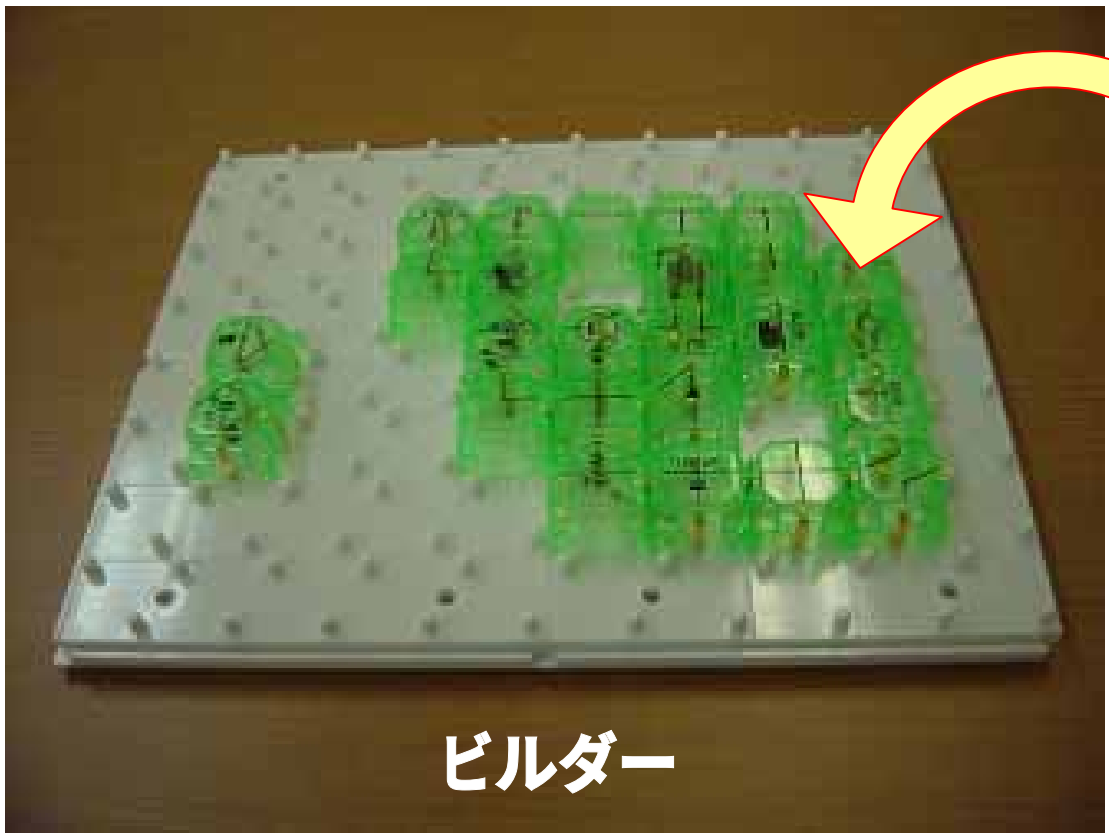
産業技術総合研究所 ものづくり先端技術研究センター
システム技術研究チーム
沢田浩之

内容

1. MZ Platform**概要**
2. **社内IT化推進のための効果的な利用方法**
3. **リリース計画**

MZ Platformの概念

コンポーネントの組み合わせによるアプリケーション開発

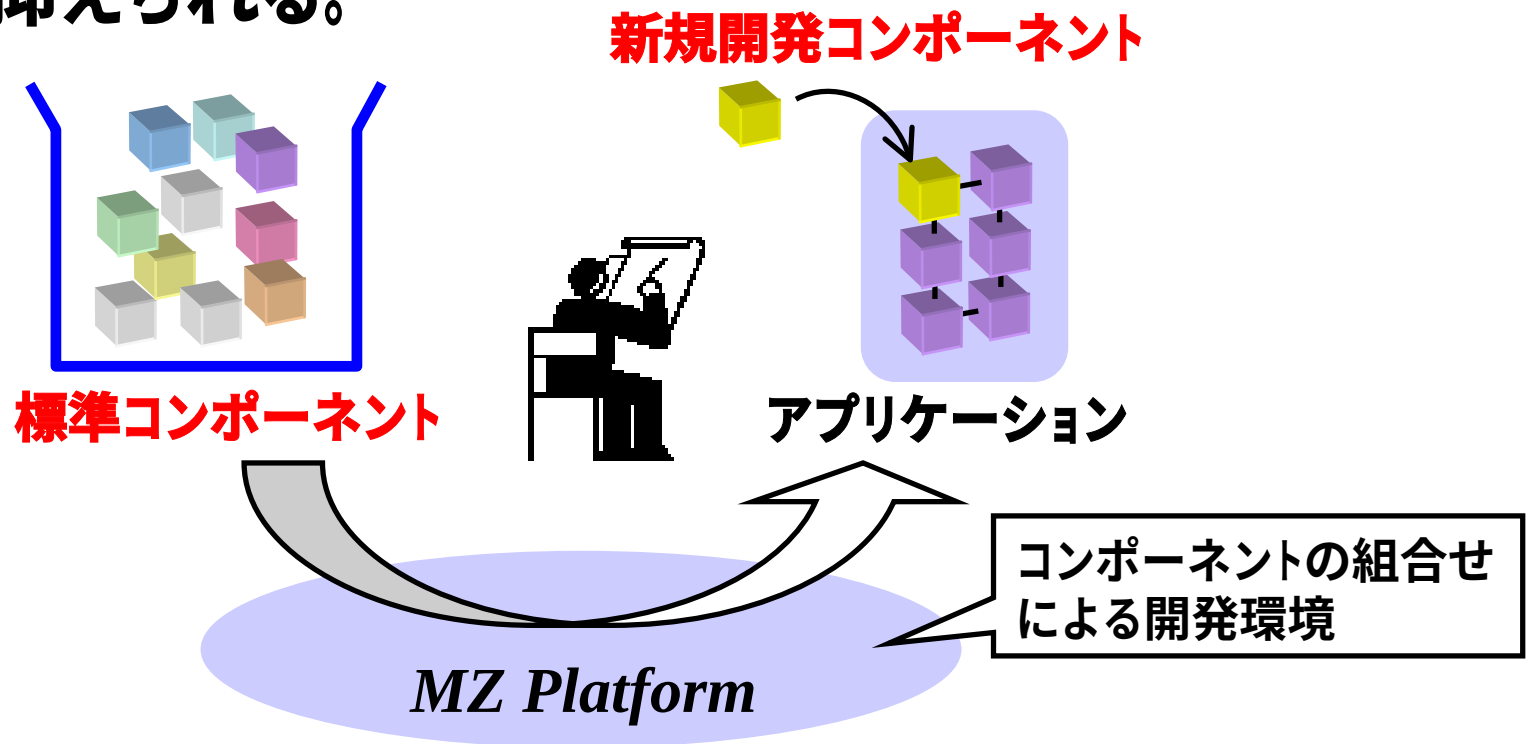


“コンポーネント”
(ソフトウェア部品)

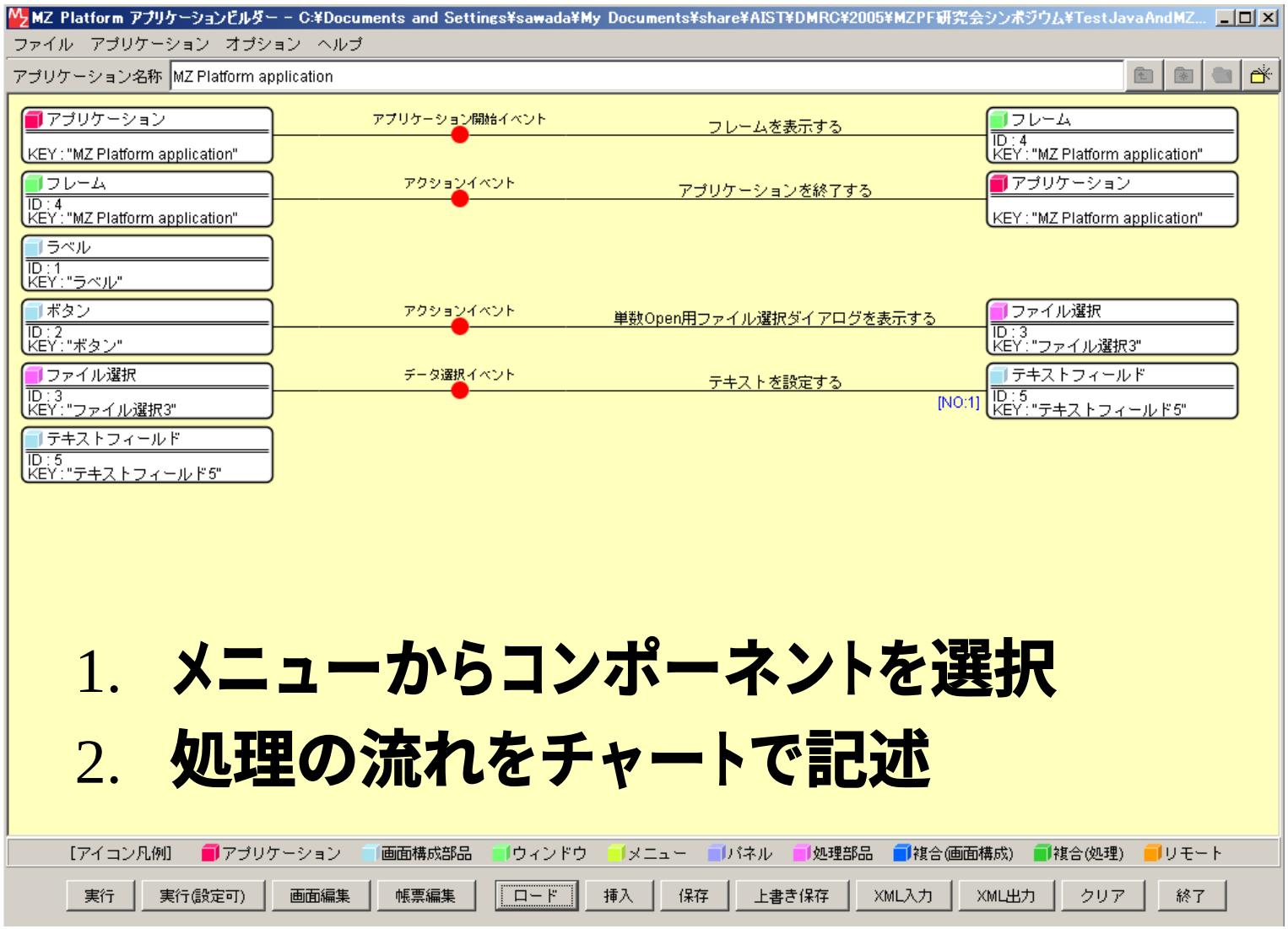
特定の機能を実現する
小規模プログラム

コンポーネント方式のメリット

- 標準コンポーネントを利用する限り、プログラムを書く必要がない。
- 新規機能開発（プログラミング）の対象を、最小限に抑えられる。



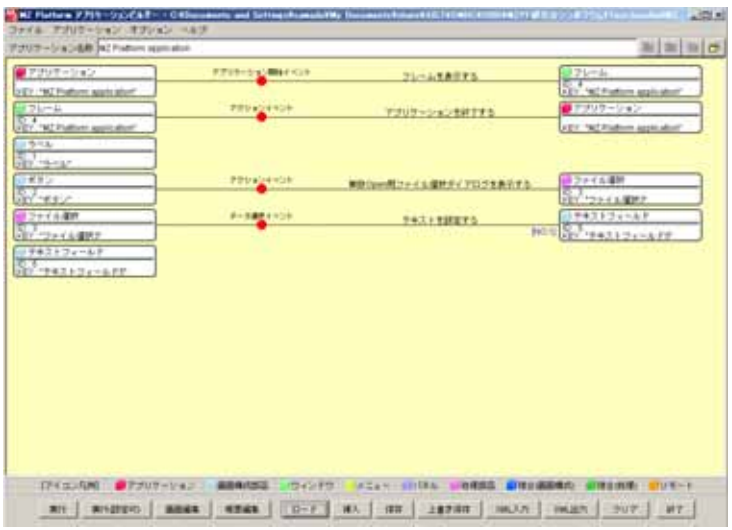
MZ Platform概観
ーアプリケーションビルダーー

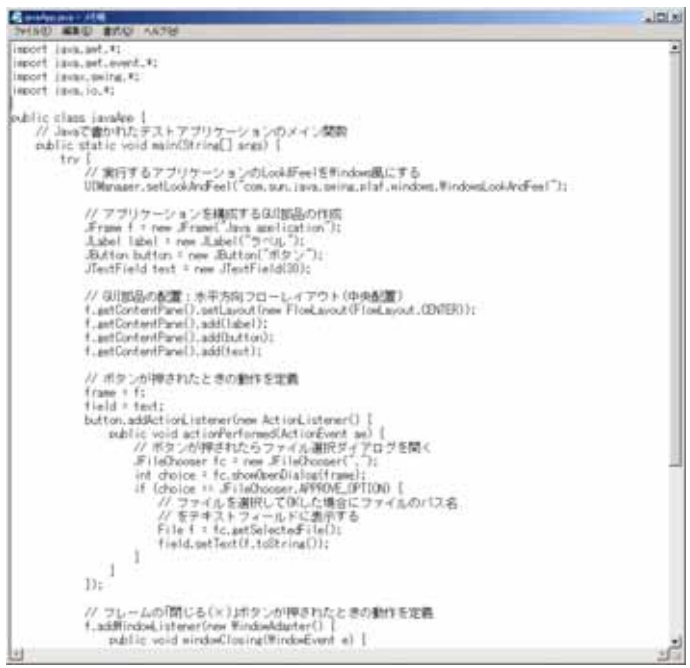


1. **メニューからコンポーネントを選択**
2. **処理の流れをチャートで記述**

MZ Platformのメリット (1)

処理の流れの直観的な記述＆把握



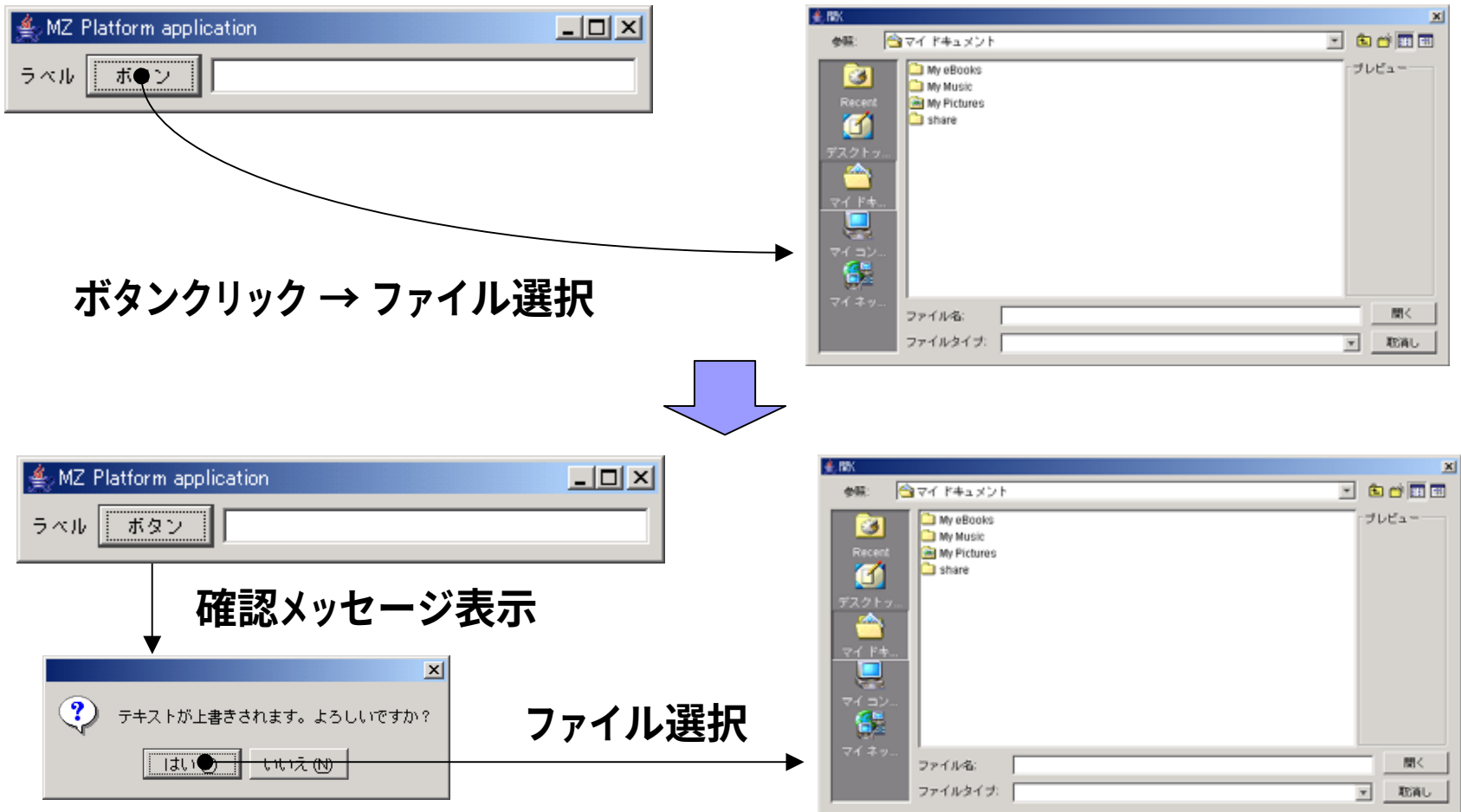


MZ Platform上の表現

Javaプログラム

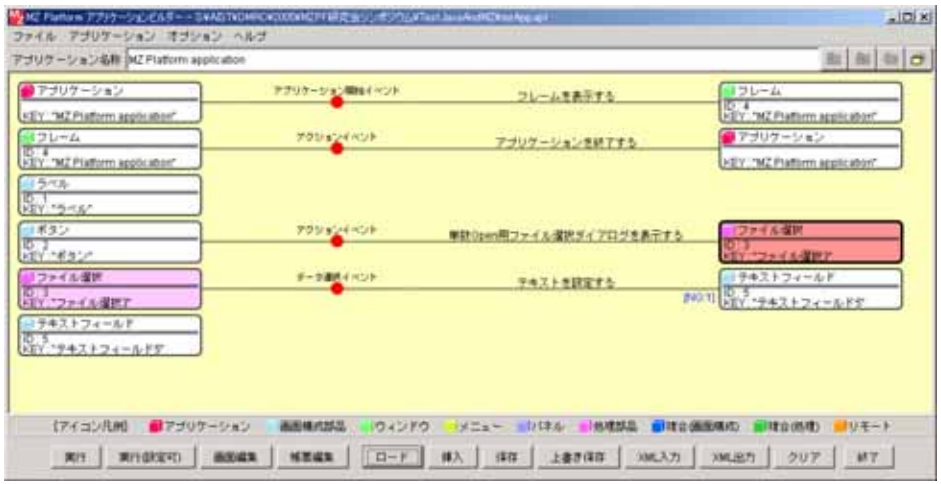
MZ Platformのメリット (2)

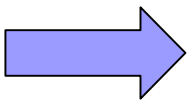
機能修正＆機能追加の容易さ

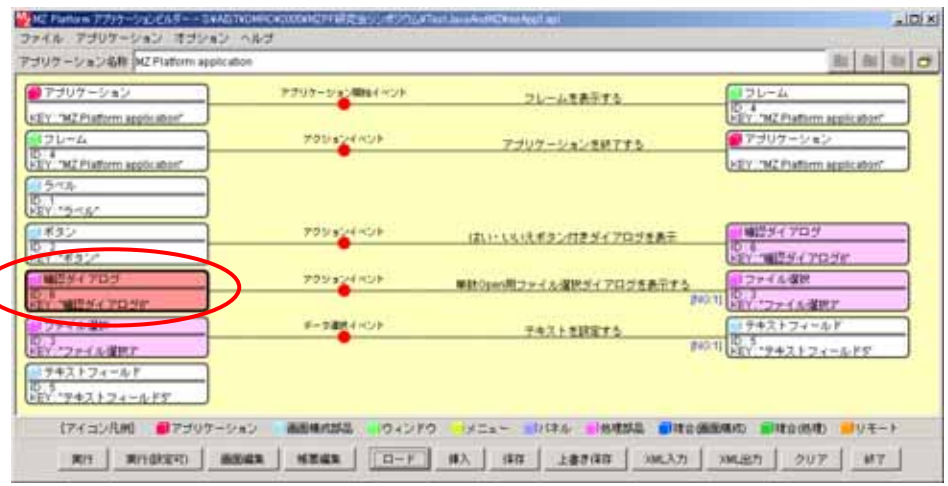


機能修正の例 (MZ Platform)

コンポーネントを追加して繋ぎ変え

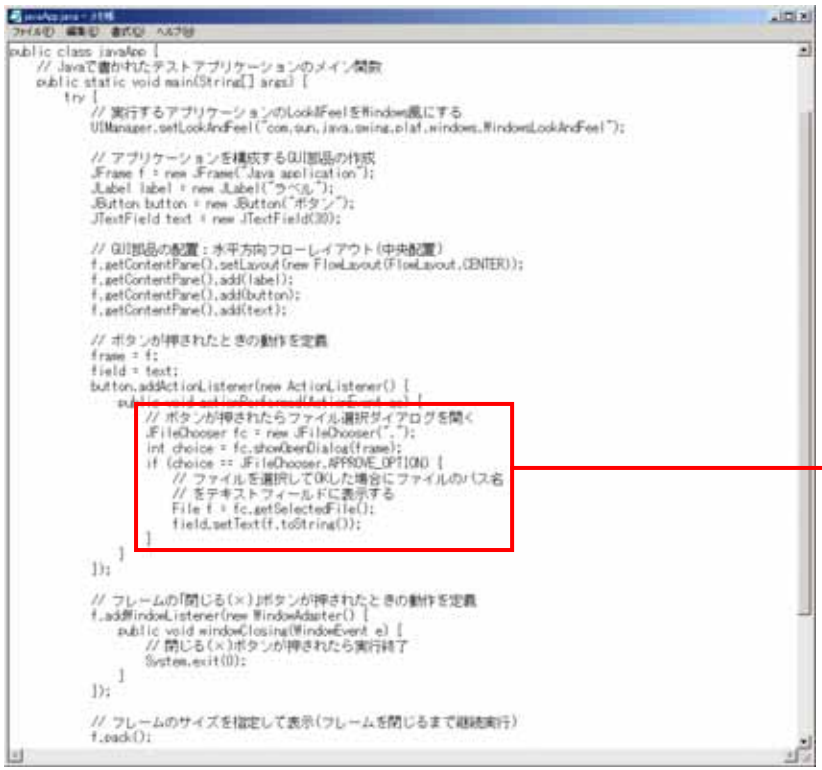




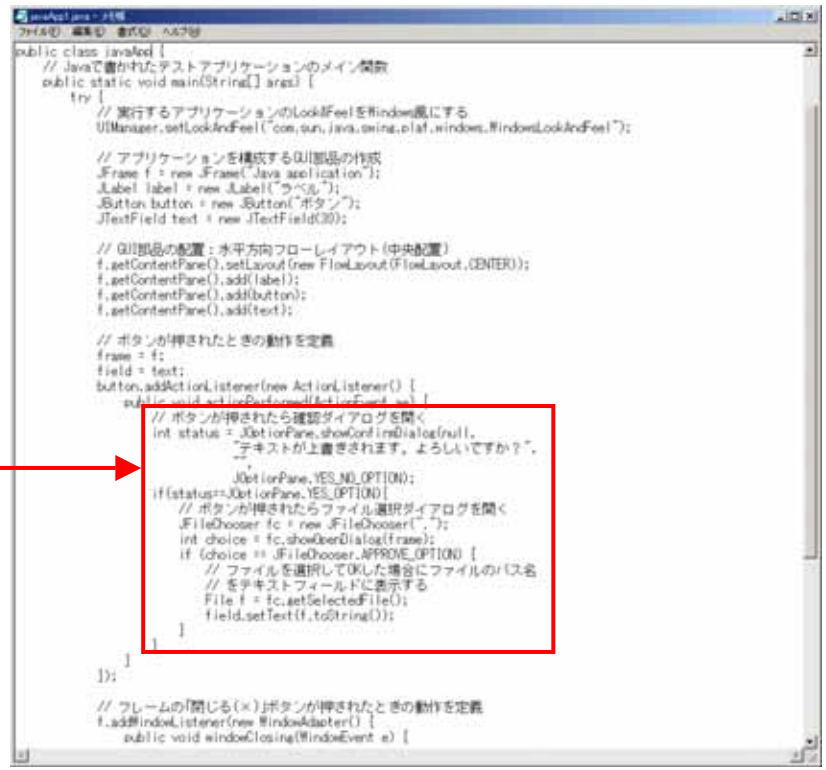


機能修正の例 (従来のプログラム)

ソースコードの編集



```
public class javaApp {
    // Javaで書かれたテストアプリケーションのメイン関数
    public static void main(String[] args) {
        try {
            // 実行するアプリケーションのLookAndFeelをWindows風にする
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");

            // アプリケーションを構成するGUI部品の作成
            JFrame f = new JFrame("Java application");
            JLabel label = new JLabel("ラベル");
            JButton button = new JButton("ボタン");
            JTextField text = new JTextField(30);

            // GUI部品の配置：水平方向フローレイアウト(中央配置)
            f.getContentPane().setLayout(new FlowLayout(FlowLayout.CENTER));
            f.getContentPane().add(label);
            f.getContentPane().add(button);
            f.getContentPane().add(text);

            // ボタンが押されたときの動作を定義
            frame = f;
            field = text;
            button.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                    // ボタンが押されたらファイル選択ダイアログを開く
                    JFileChooser fc = new JFileChooser(".");
                    int choice = fc.showOpenDialog(frame);
                    if (choice == JFileChooser.APPROVE_OPTION) {
                        // ファイルを選択してOKした場合にファイルのパス名
                        // をテキストフィールドに表示する
                        File f = fc.getSelectedFile();
                        field.setText(f.toString());
                    }
                }
            });

            // フレームの「閉じる(×)」ボタンが押されたときの動作を定義
            f.addWindowListener(new WindowAdapter() {
                public void windowClosing(WindowEvent e) {
                    // 閉じる(×)ボタンが押されたら実行終了
                    System.exit(0);
                }
            });

            // フレームのサイズを指定して表示(フレームを閉じるまで継続実行)
            f.pack();
```



```
public class javaApp {
    // Javaで書かれたテストアプリケーションのメイン関数
    public static void main(String[] args) {
        try {
            // 実行するアプリケーションのLookAndFeelをWindows風にする
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");

            // アプリケーションを構成するGUI部品の作成
            JFrame f = new JFrame("Java application");
            JLabel label = new JLabel("ラベル");
            JButton button = new JButton("ボタン");
            JTextField text = new JTextField(30);

            // GUI部品の配置：水平方向フローレイアウト(中央配置)
            f.getContentPane().setLayout(new FlowLayout(FlowLayout.CENTER));
            f.getContentPane().add(label);
            f.getContentPane().add(button);
            f.getContentPane().add(text);

            // ボタンが押されたときの動作を定義
            frame = f;
            field = text;
            button.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                    // ボタンが押されたら確認ダイアログを開く
                    int status = JOptionPane.showConfirmDialog(null,
                                 "テキストが上書きされます。よろしいですか？",
                                 "",
                                 JOptionPane.YES_NO_OPTION);
                    if(status==JOptionPane.YES_OPTION){
                        // ボタンが押されたらファイル選択ダイアログを開く
                        JFileChooser fc = new JFileChooser(".");
                        int choice = fc.showOpenDialog(frame);
                        if (choice == JFileChooser.APPROVE_OPTION) {
                            // ファイルを選択してOKした場合にファイルのパス名
                            // をテキストフィールドに表示する
                            File f = fc.getSelectedFile();
                            field.setText(f.toString());
                        }
                    }
                }
            });

            // フレームの「閉じる(×)」ボタンが押されたときの動作を定義
            f.addWindowListener(new WindowAdapter() {
                public void windowClosing(WindowEvent e) {
```

- 修正箇所の特定が困難
- 影響範囲の把握が困難

MZ Platformを利用したアプリケーション開発

MZ Platformの特長: 機能の修正・追加・削除が容易

⇒維持・管理・拡張に要する負担を抑えられる。（すべてコンポーネントの組み換えで行うことができる。）

標準コンポーネント: 160種類

アプリケーションで使われているコンポーネント数

- 工程管理システム簡易版: 41種類 267個
- MZ Checker: 31種類 451個
- SMART Process Management: 63種類 827個
- SMART Product Management: 66種類 2514個

内容

1. MZ Platform概要
2. **社内IT化推進のための効果的な利用方法**
3. リリース計画

社内IT化への取り組み (1)

社内IT化の位置付け: 業務改善の一環

- 業務の効率化
- コスト削減
- ⋮

⇒ IT化そのものが目的ではない (手段に過ぎない)

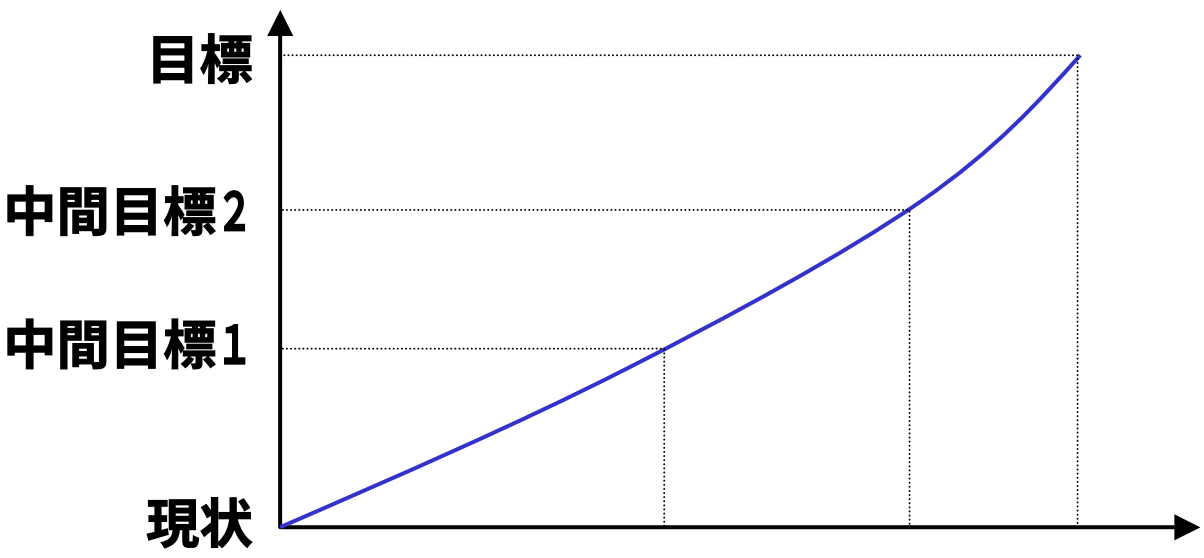
IT化の損得勘定: 目的達成のための検討事項

- 導入の負担 (コスト、時間、教育・研修、…)
- 運用の負担 (紙で運用する場合との比較)
- 業務改善の効果 (コスト、時間、利便性、…)

社内IT化への取り組み (2)

ロードマップの明確化

1. 現状分析 (問題点＆改善対象の明確化)
2. 具体的なゴールイメージ (あるべき姿) の設定
3. 具体的なマイルストーン (中間目標) の設定



- 経営層
- 現場作業者
- IT担当者

の意思共有が必須

アプリケーション開発における問題点

アプリケーション開発時に起こりがちな問題

⇒「思っていたのと違うシステムができあがった」

原因は？

- 要件定義が不明確だった（ソフト開発者の言い分）
- 業務を知らない奴が作った（エンドユーザの言い分）

⇓

プロトタイピングの必要性＆重要性

実物があれば、具体的な仕様を考えやすい

MZ Platformの効果的な利用方法

その場で直せるプロトタイピング ⇐ 修正が容易

- 開発期間の大幅な短縮＝開発コストの低減
- エンドユーザとソフト開発者間での具体的なイメージ共有

ピンポイントアプリケーション開発 ⇐ 機能追加が容易

- 業務改善効果の最も高い機能だけに絞った開発
- 周辺機能はあとから追加→高機能アプリケーションへ展開

内容

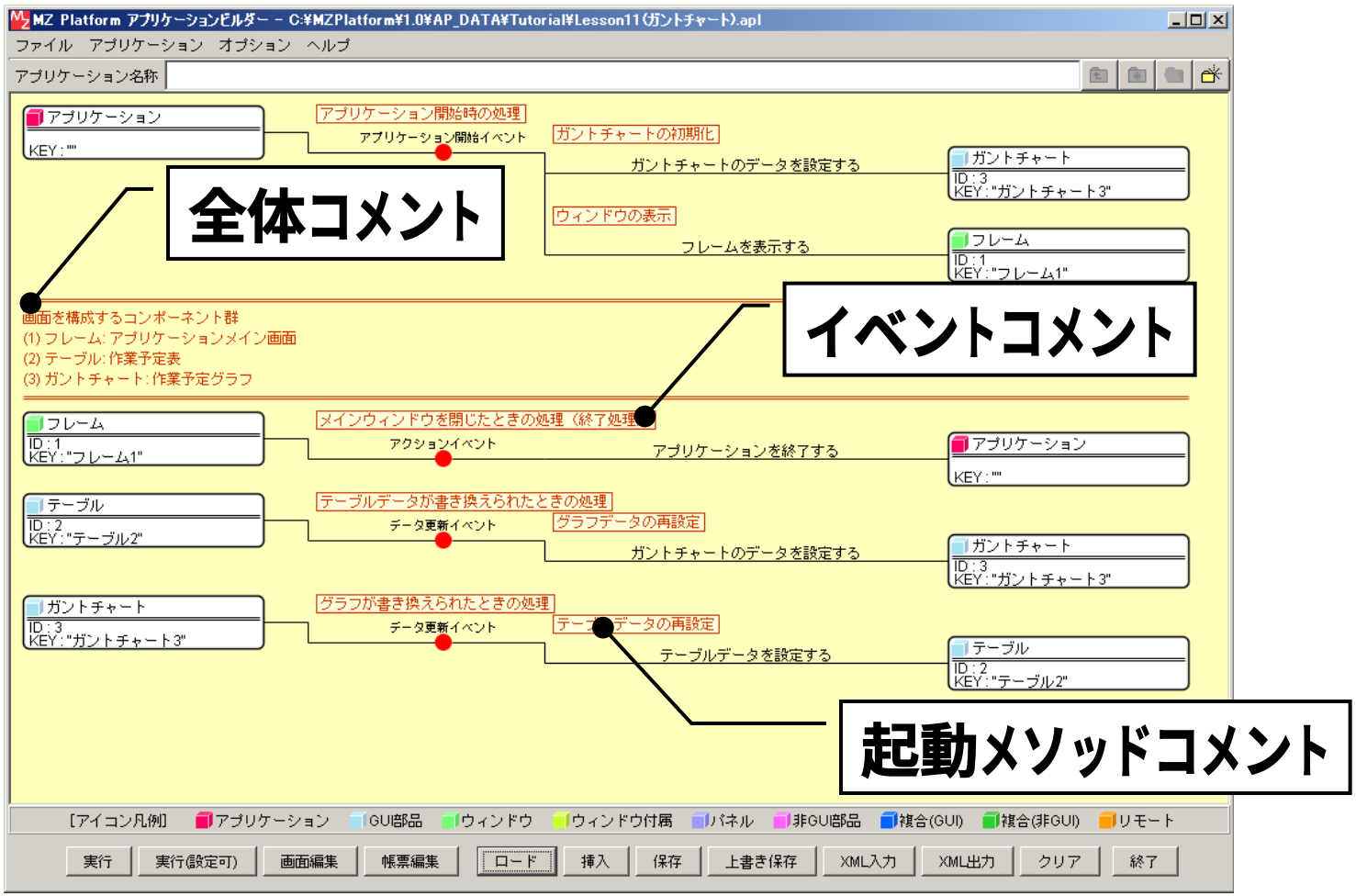
1. MZ Platform概要
2. 社内IT化推進のための効果的な利用方法
3. リリース計画

MZ Platform Version 1.2 リリース内容

- アプリケーションビルダー
  - コメント記述
  - 階層化（複合コンポーネント）とマルチウィンドウ
  - コンポーネントのコピー＆ペースト
- 標準コンポーネント
- マニュアルとサンプルアプリケーション

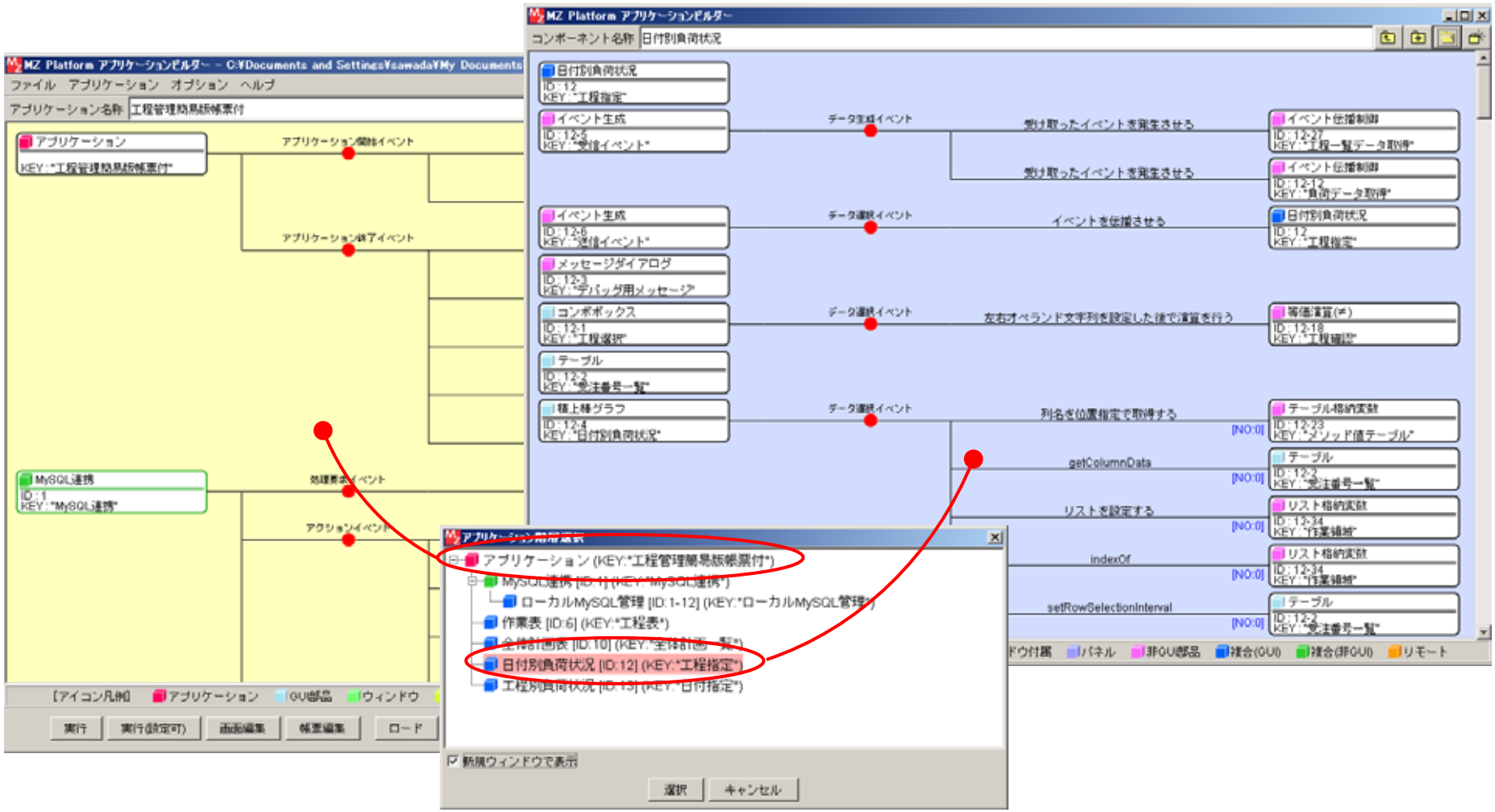
コメント記述

3種類のコメント記述

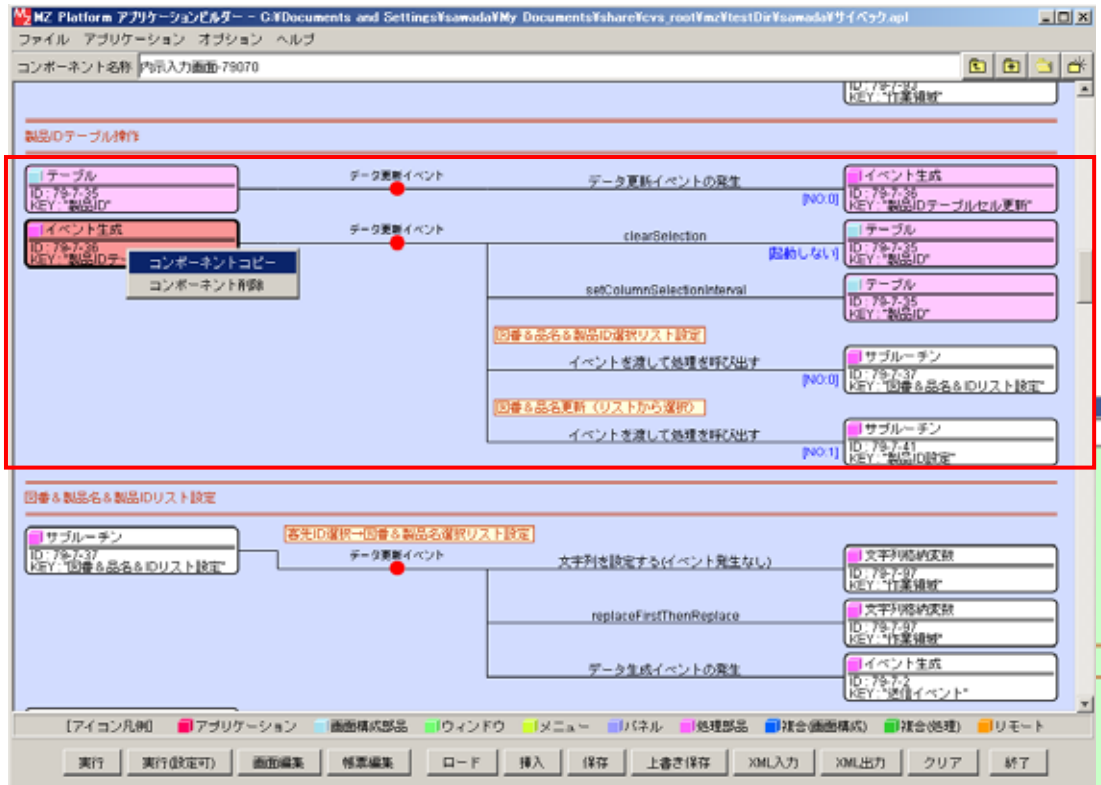


階層化（複合コンポーネント）とマルチウィンドウ

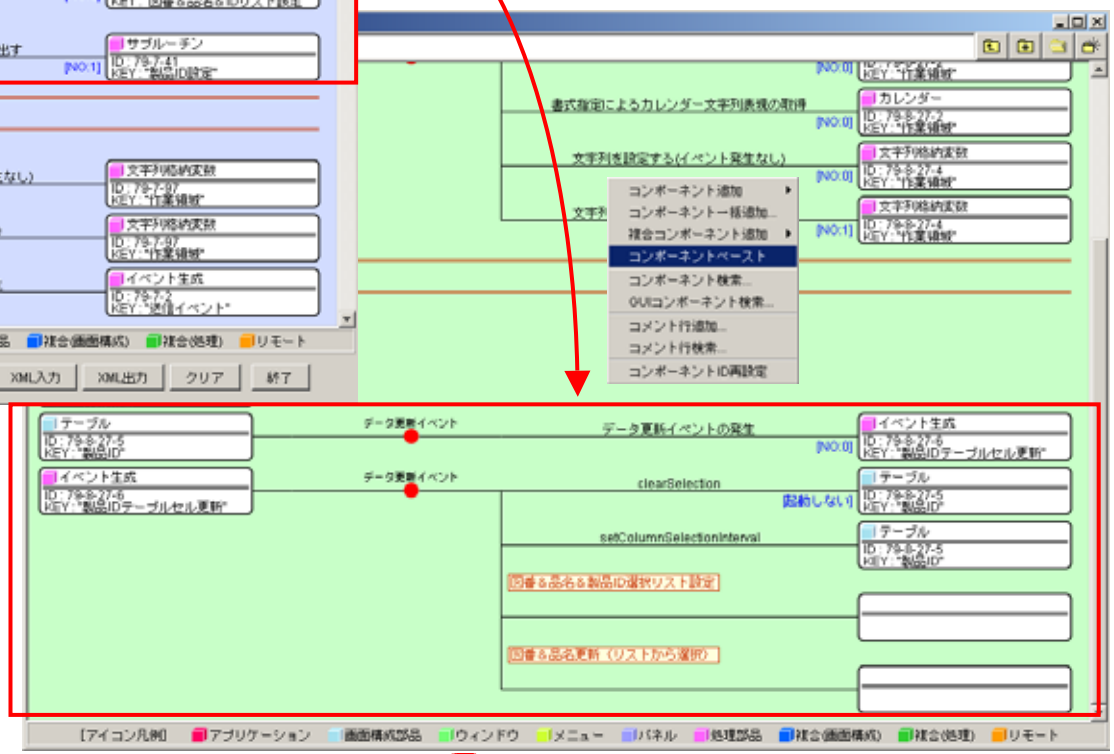
複数階層の同時参照・編集機能



コンポーネントのコピー＆ペースト



コピー範囲



ペースト結果

標準コンポーネント(Version 1.2)

標準コンポーネント: 160**種類**

- **画面構成部品** (59**種類**)

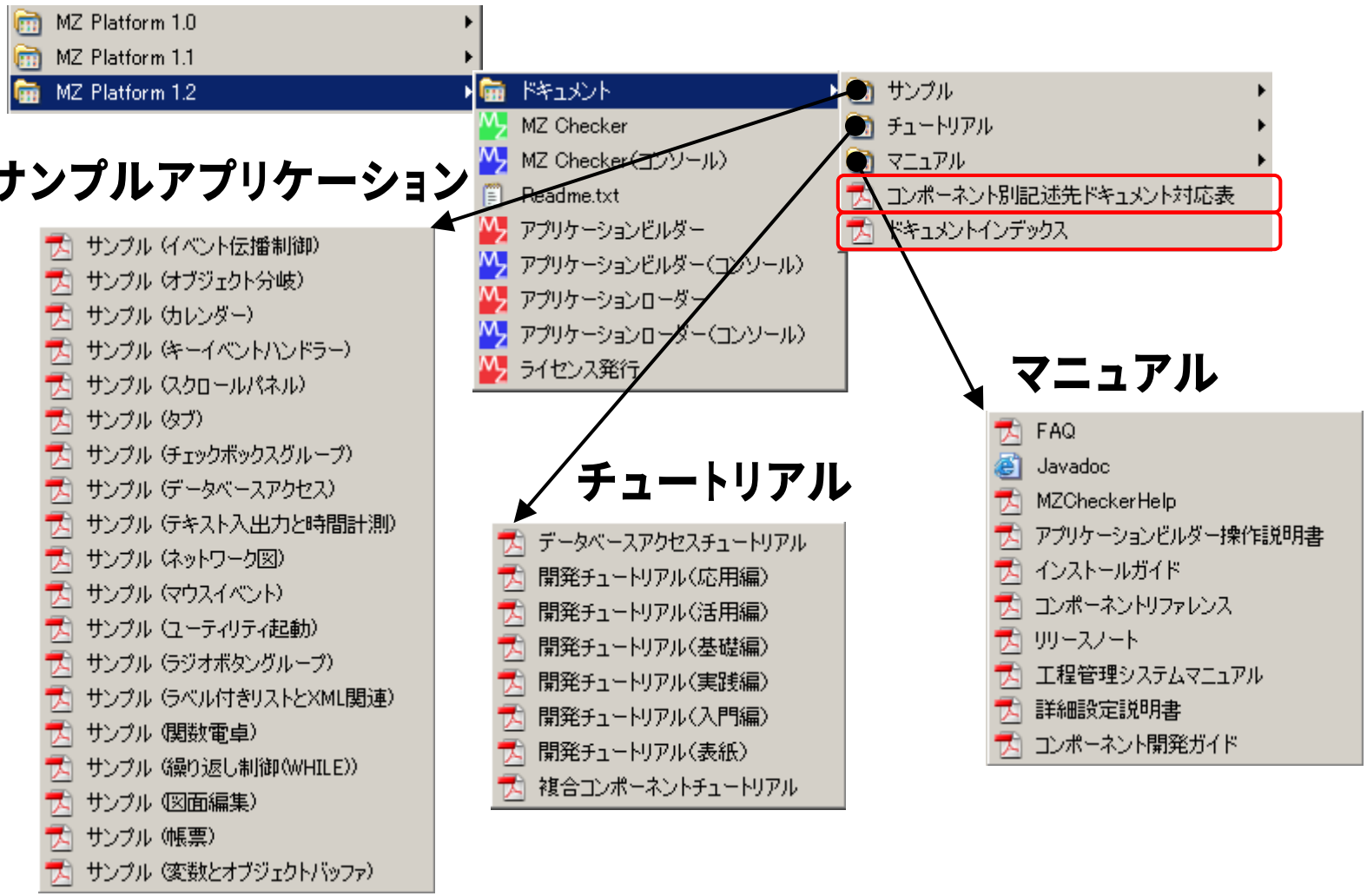
  ウィンドウ、メニュー、ボタン、グラフ他
- **処理部品** (83**種類**)

  条件制御、演算、統計、サブルーチン、バーコード他
- **入出力** (13**種類**)

  データベースアクセス、標準入出力，ファイル入出力、帳票
- **チュートリアル用サンプル** (5**種類**)

マニュアルとサンプルアプリケーション

スタートメニューから選択



ドキュメントインデックス

収録されているドキュメントの概要と用途を記した一覧表

No	ドキュメント名称	概要	用途
1	インストールガイド	インストール手順を説明	インストール手順を調べたい場合。
2	ReadMe.txt	インストール後の手順の概要を説明	インストール後の手順を手早く参照したい場合。
3	詳細設定説明書	MZ Platformを使用する際の高度な設定についての説明	MZ Platformの動作を高度に設定したい場合。
4	ドキュメントインデックス	各種ドキュメントに対するインデックス。	どういった場合に、どのドキュメントを参照すればいいかを調べる場合。
5	開発チュートリアル（入門編）	MZPlatformの目的や機能の説明	自習用。(MZPlatform初心者向け)
6	開発チュートリアル（基礎編）	MZPlatformの操作方法の説明	自習用。(MZPlatform初心者向け)
7	開発チュートリアル（応用編）	MZPlatformでのアプリケーション作成方法の説明	自習用。(MZPlatform中級者向け)
8	開発チュートリアル（実践編）	MZPlatformでのアプリケーション作成方法の説明	自習用。(MZPlatform上級者向け)
9	開発チュートリアル（活用編）	MZPlatformでのアプリケーション作成方法の説明	自習用。(MZPlatform上級者向け)
10	コンポーネントリファレンス	MZPlatformで使用する全コンポーネントのメソッドのリファレンス。	アプリケーション開発時に、使用するコンポーネントの公開メソッドを調べたい場合。
11	アプリケーションビルダー操作説明書	アプリケーションビルダーの操作方法の説明	アプリケーションビルダーの操作方法を調べたい場合。
12	コンポーネント開発ガイド	コンポーネントの実装方法の説明	コンポーネントを作成したい場合。
13	データ連携導入手順書	データ連携機能の導入方法を説明	データ連携機能を使用したい場合。
14	MZCheckerHelp	MZ Checkerの操作方法を説明	3次元CADデータ（STEP形式，IGES形式）のデータ品質をJAMA/JAPIAの定めるPDQガイドラインのチェック項目と指定した値とによりチェックしたいとき。
15	工程管理システムマニュアル	工程管理サンプルアプリケーションを説明	工程管理サンプルアプリケーションを参照する場合。
16	サンプル（変数とオブジェクトバッファ）	変数コンポーネントとオブジェクトバッファコンポーネントを使ったオブジェクト操作説	コンポーネント間でやり取りされるデータ（オブジェクト）の編集や、フィールド値の設定・
17	サンプル（ネットワーク図）	ネットワーク図コンポーネントを使ったグラフ構造データ操作説明	ノードとエッジから構成されるグラフ構造データの視覚的な操作

コンポーネント別記述先ドキュメント対応表

各コンポーネントについて、説明記述先を記した一覧表

（注）付属のJavadocには、すべてのコンポーネントの情報が記述されています。下表は、Javadoc以外の記述先ドキュメントを示したものです。

分類	グループ	コンポーネント名称	ドキュメント名称
画面構成部品	ウィンドウ	フレーム	開発チュートリアル(基礎編)
		ダイアログ	
	メニュー	メニュー	
		ポップアップメニュー	
		メニューアイテム	
		チェックボックスメニューアイテム	
		セパレータ	
		ツールバー	
		ツールバーセパレータ	
	パネル	パネル	開発チュートリアル(実践編)，開発チュートリアル(活用編)，データベースアクセスチュートリアル
		タブ	サンプル（タブ）
		分割パネル	サンプル（分割パネル)，サンプル（スクロールパネル）
		スクロールパネル	サンプル（スクロールパネル）
	テキスト	ラベル	開発チュートリアル(応用編)
		テキストフィールド	開発チュートリアル(活用編)，データベースアクセスチュートリアル
		数値入力フィールド	開発チュートリアル(基礎編)，開発チュートリアル(実践編)
		日付入力フィールド	
		マスク入力フィールド	
		パスワード入力フィールド	データベースアクセスチュートリアル
		テキストエリア	サンプル（テキスト入出力と時間計測）
		HTML表示パネル	
	ボタン	ボタン	開発チュートリアル(基礎編)
		トグルボタン	
	テーブル	テーブル	開発チュートリアル(応用編)，開発チュートリアル(実践編)，開発チュートリアル(活用編)
	ツリー	ツリー	開発チュートリアル(応用編)
	リスト	リスト	
	コンボボックス	コンボボックス	
	チェックボックス	チェックボックス	
		チェックボックスグループ	サンプル（チェックボックスグループ）

今後のリリース計画

Version 1.3 (2005年9月リリース予定)

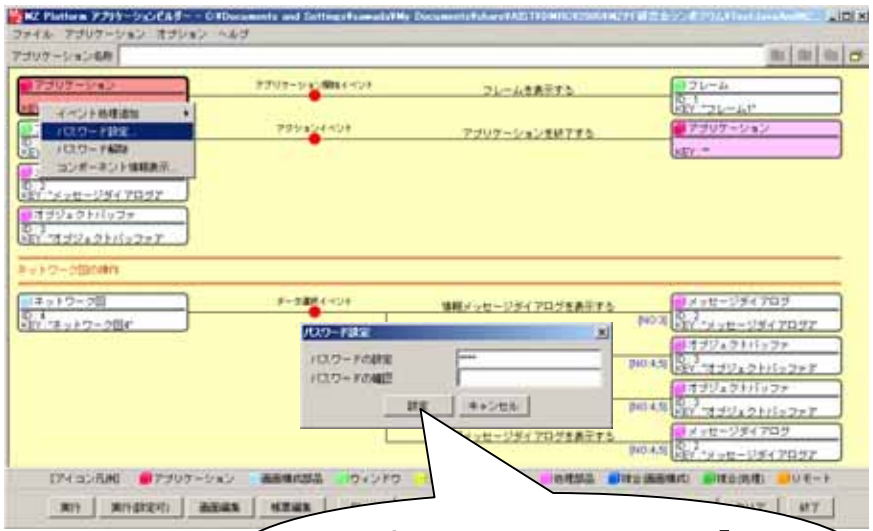
- アプリケーション・複合コンポーネントのパスワードロック機能
  - ローダーからの実行はパスワード不要
  - ビルダーからの参照＆編集はパスワード必要
- デバッガ機能
  - コンポーネントレベルでのアプリケーション実行のトレース

Version 1.4 (2006年2月リリース予定)

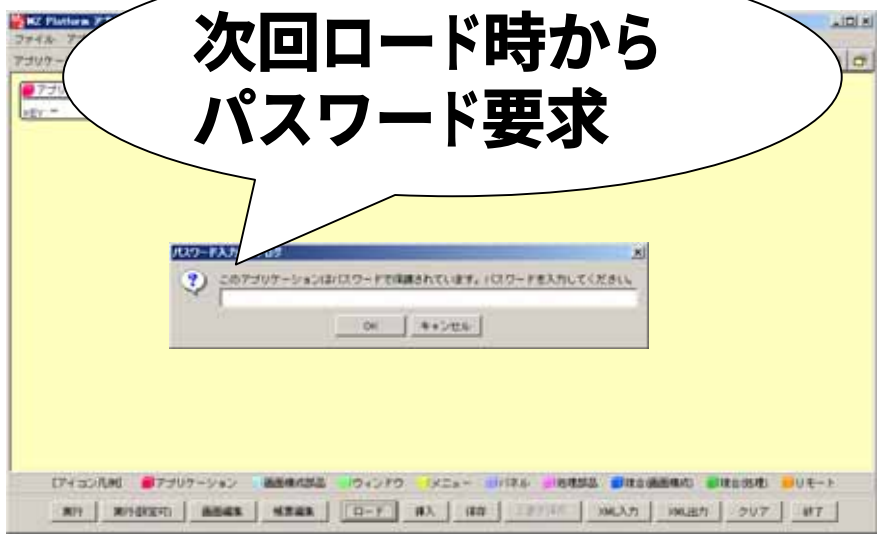
- アプリケーション・複合コンポーネントのXML入出力機能強化
  - Java、MZ Platformのバージョンアップに対する互換性確保

パスワードロック機能

参照＆編集の制限。実行は可能。



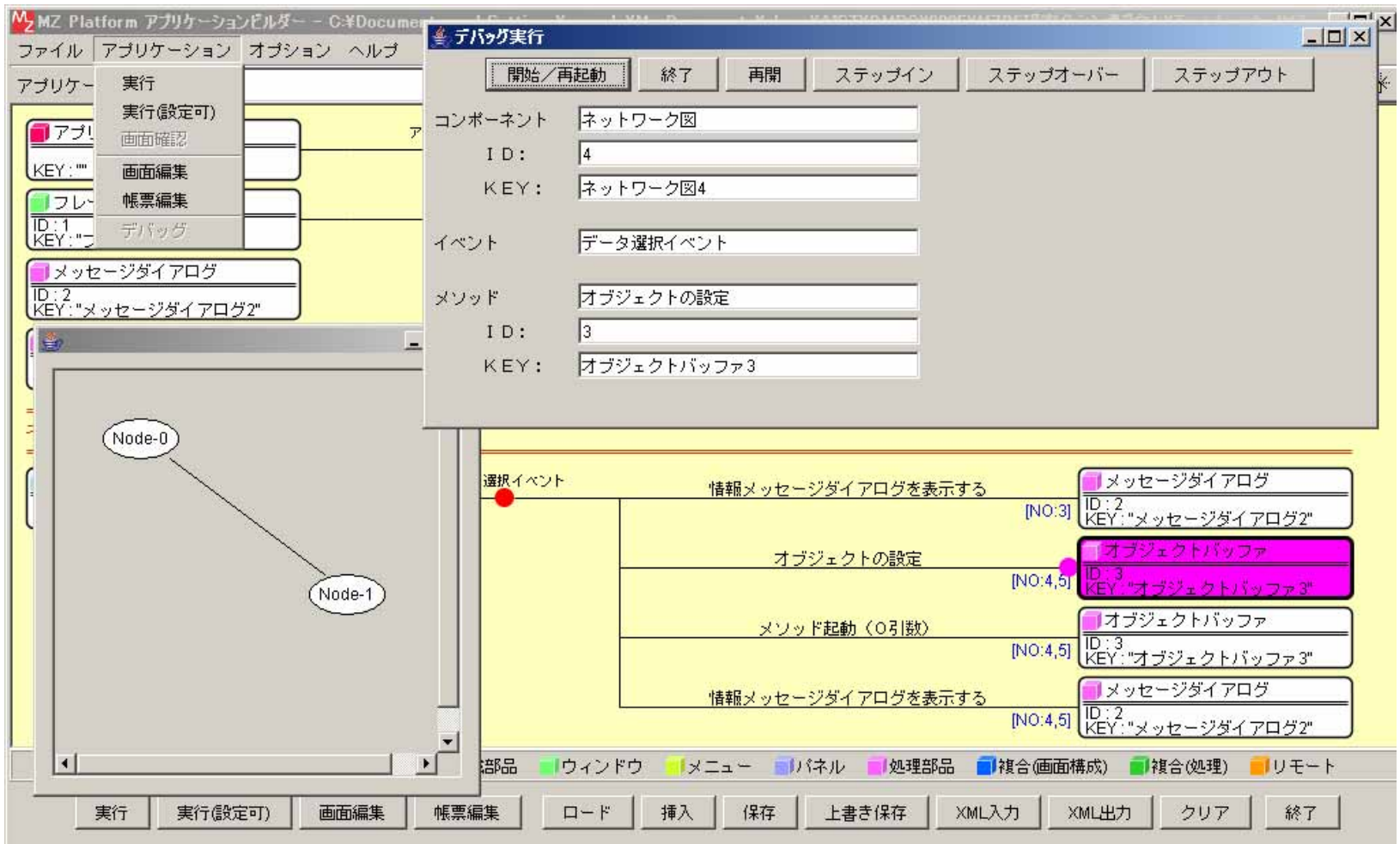
パスワード設定



次回ロード時から
パスワード要求

デバッグ機能

コンポーネントレベルでの実行トレース



サポート

- **講習会**
  - **つくばにて週2回開催**
  - **各地域にて不定期に開催 (これまでに栃木、長野、石川、広島、愛媛、高知で開催)**

- **ご質問、ご要望**
  - **MZプラットフォーム研究会へメールでお願いします。**
    pf-support@m.aist.go.jp

- **よくある質問 (FAQ)**
  - **研究会ホームページで随時更新。**
    http://unit.aist.go.jp/digital-mfg/mzpf/mz_faq.html