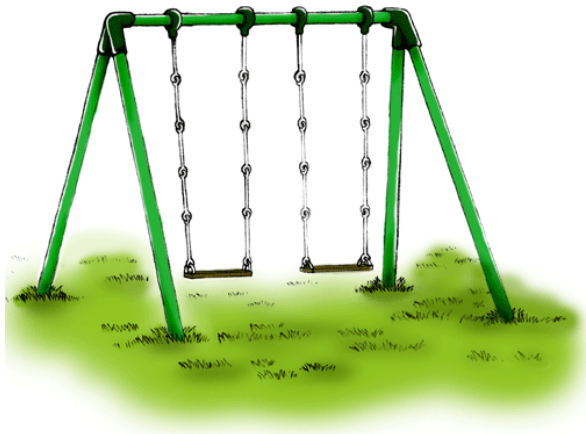
公開文書

ソースコードからソースコードを自動生成!?
進化した「blanco Framework」の正体とは

2012.09.07

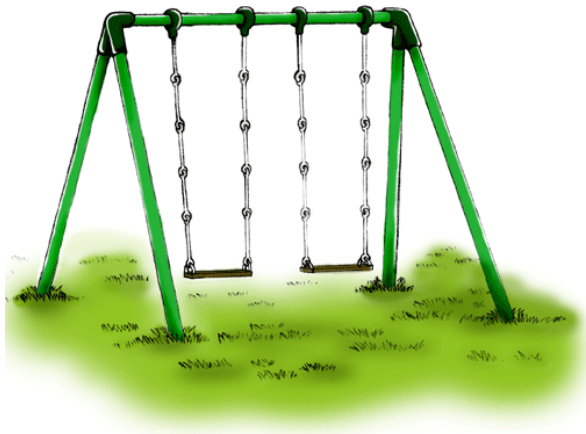
blanco Framework コミッタ
伊賀 敏樹 (いがぴょん)



目次

- 第1部
  - 第2世代 blanco Framework のご紹介
  - blanco Framework のご紹介
- 第2部
  - Eclipse 日本語化への取り組みについて

第2世代 blanco Framework のご紹介



おことわり

- これから紹介する第2世代 blanco Framework (Blanco2g) は、現状 Java 言語のみサポートです
  - 第1世代 blanco Framework (Blanco1g) は Java, C#.NET, JavaScript, VB.NET, PHP, Ruby, Python, Pascal に対応しています。
  - ※Blanco1g で 主にサポートされるのは Java, C#.NET です。

- 第1世代 blanco Framework (Blanco1g) も引き続きメンテナンスします
  - Blanco2g は Blanco1g と併用すると高効果を得られます。

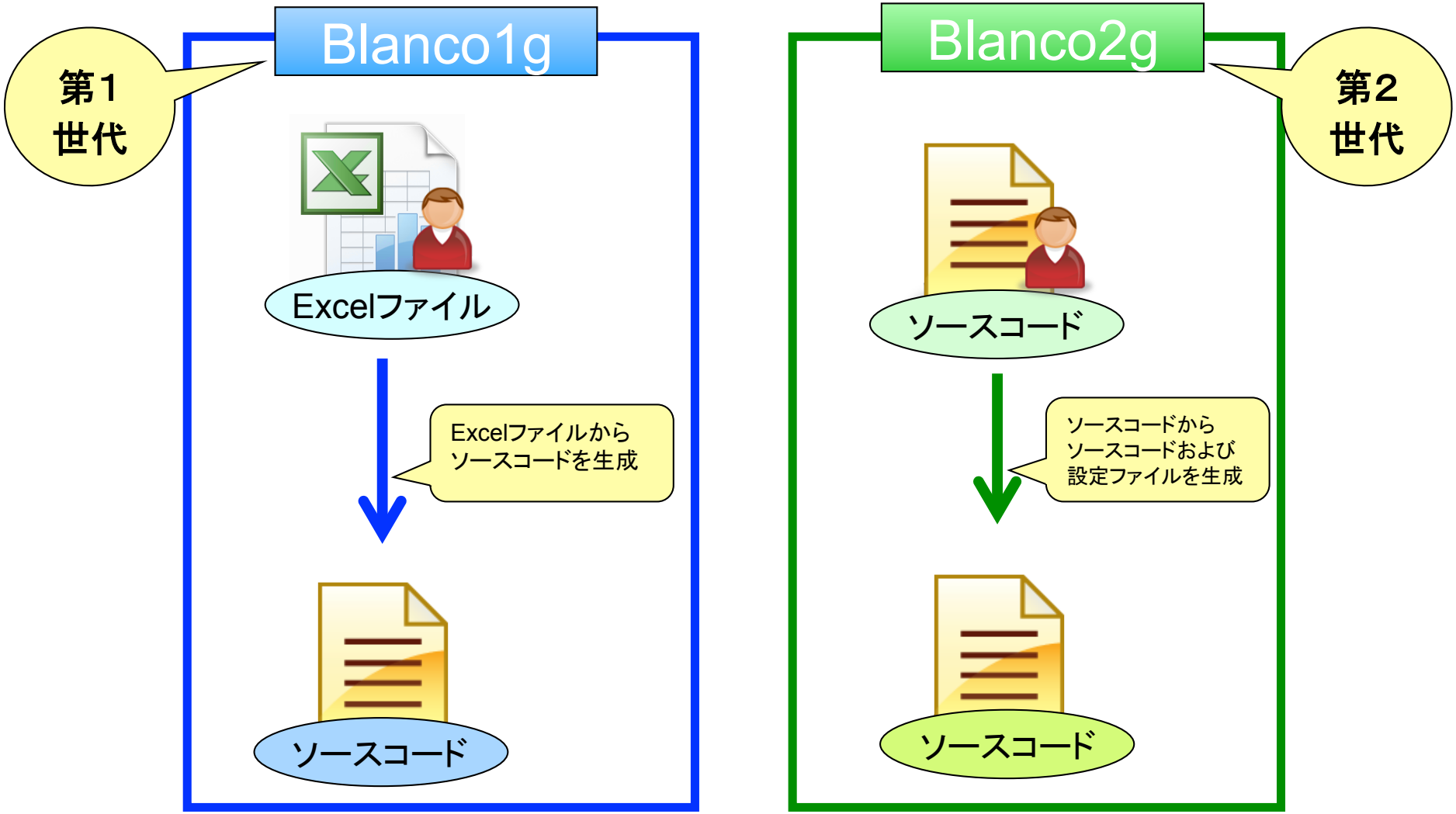
第2世代 blanco Framework がもたらす革新

- 古いアプリケーションサーバーや古い Java VM にも
- アノテーションベースによる最新の開発環境をもたらします。
  ☆Java SE 1.5.0 またはそれ以降が対象

- しかも、DI コンテナのような特殊なライブラリは不要です。

- ソースコード自動生成アプローチだからこそ実現できる革新

ソースコード自動生成は新世代へ

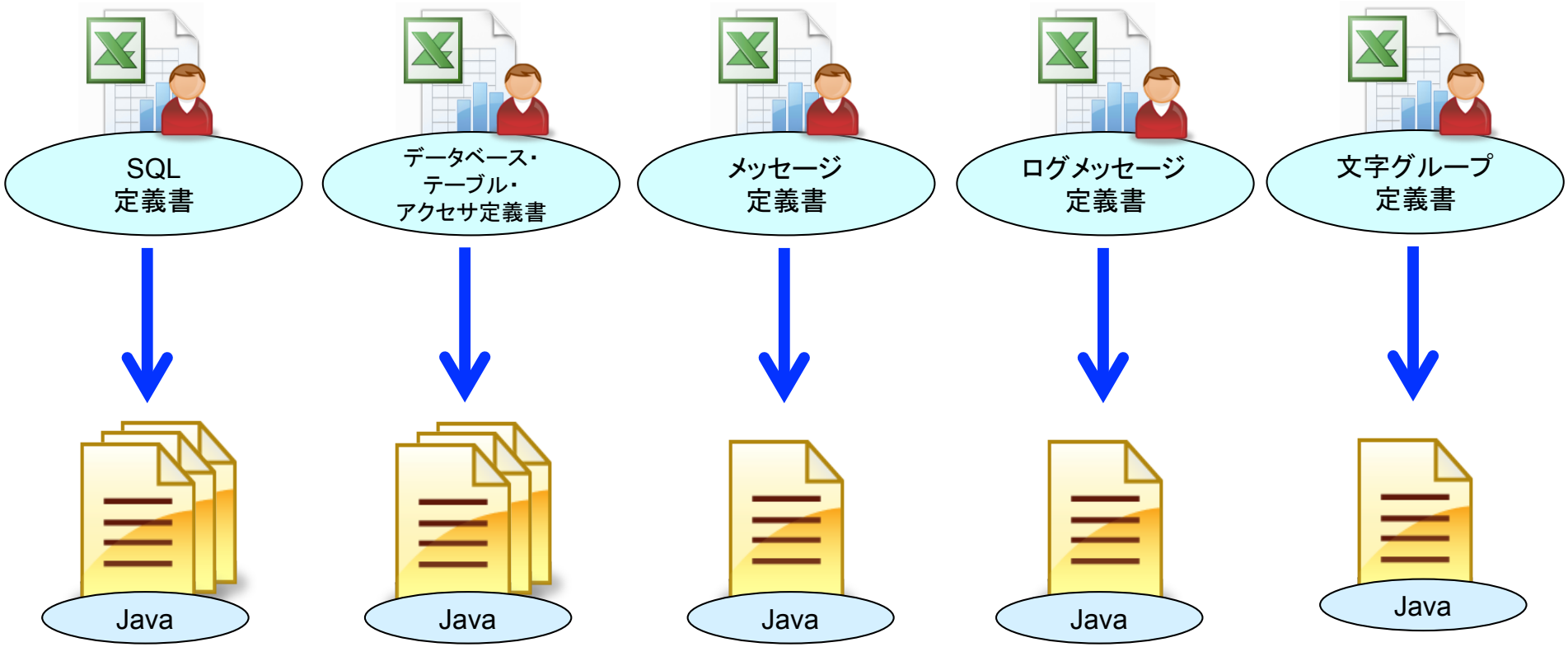


blanco Framework のソースコード自動生成は第2世代へと進みます

Blanco1g の典型的な利用例

Blanco1g

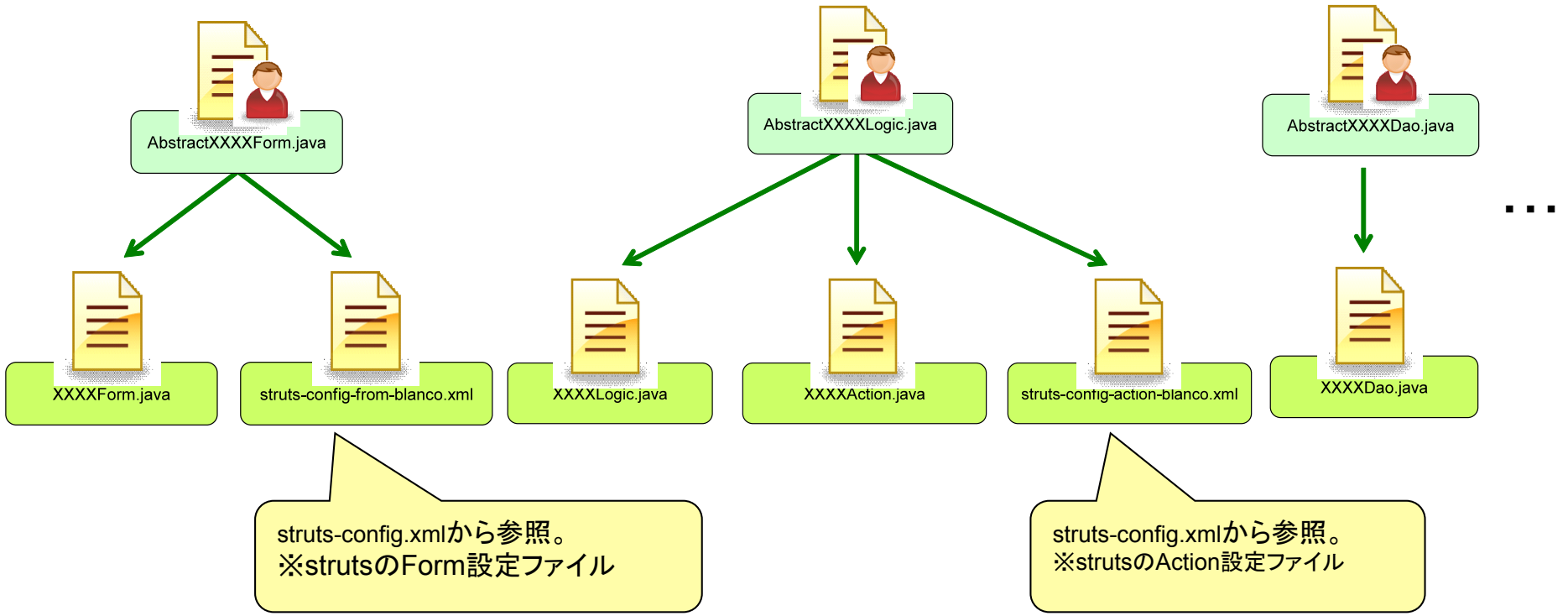
Excel ファイルからソースコードを生成



Blanco2g の典型的な利用例

Blanco2g

ソースコードからソースコードおよび設定ファイルを生成



Blanco2g の典型的なコード例 (1)

```
@BlancoStrutsLogic(path = "/run")
public abstract class AbstractSampleLogic {
    @BlancoStrutsForward(path = "/next.jsp")
    private static final String FORWARD_SUCCESS = "success";

    @BlancoStrutsForward(path = "/error.jsp")
    private static final String FORWARD_ERROR = "error";

    public String execute(ActionMapping mapping, SampleForm form, HttpServletRequest request,
            HttpServletResponse response, @BlancoInject Connection conn) throws Exception {
        // ここに実際のビジネスロジックを記述します。

         if (true) {
            return FORWARD_SUCCESS;
        } else {
            return FORWARD_ERROR;
        }
    }
}
```

設定ファイルをアノテーション記述で自動生成

データベース接続をアノテーション記述でインジェクション【トランザクション境界】

各種 XML 記述のほとんどが不要
アプリケーション実行時解決や「DI コンテナ」が不要

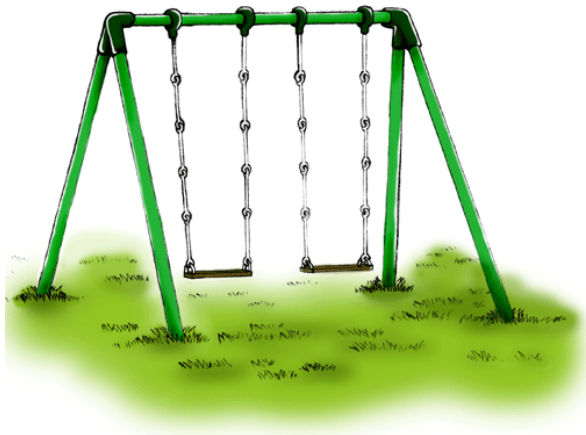
Blanco2g の典型的なコード例 (2)

```
@BlancoStrutsForm
public abstract class AbstractSampleForm extends ActionForm {
    private static final long serialVersionUID = 1L;

    @BlancoGetterSetter
    @BlancoStrutsReset
    @BlancoValidateMethodForStruts
    @BlancoValidateRequired
    @BlancoValidateLength(max = 5)
    protected String field1 = "";

    @BlancoGetterSetter
    protected int field2 = -1;
}
```

ゲッター・セッターを自動生成

バリデーション内容を対象フィールドの
至近距離に記述が可能

blanco Framework のご紹介



blanco Framework とは?

- オープンソースのソースコード自動生成型フレームワーク
  ライセンス：GNU LGPL
- 【Blanco1g】Excel(*.xls)ファイル形式の様式を中心としたソースコード自動生成
- 【Blanco2g】ソースコードを入力としてソースコード自動生成。Blanco1g と同時利用で高効果
- Apache Ant (Java) / Eclipse 上で動作
- 主たる適用事例は数十人月から数百人月規模のプロジェクト

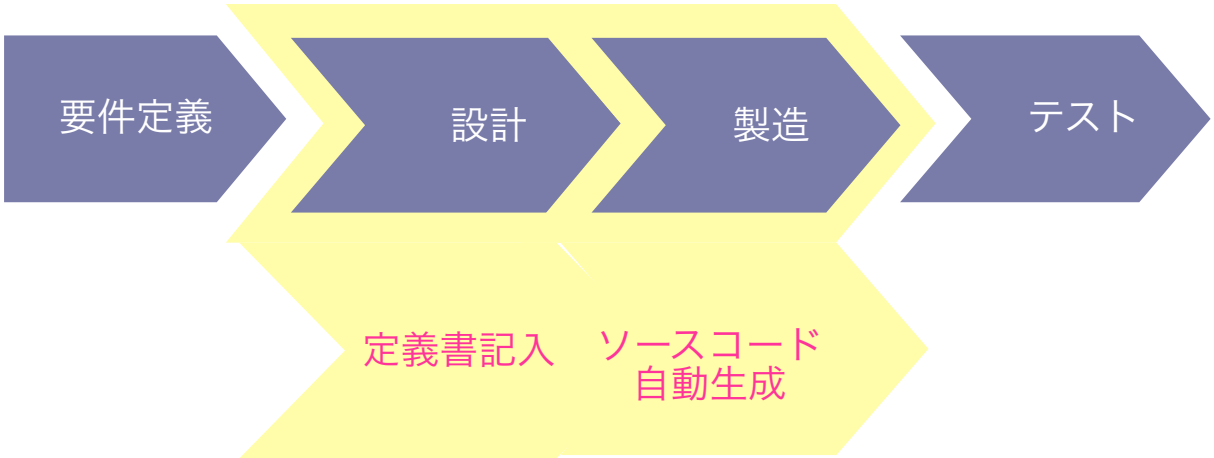
- SourceForge.JP において 2005/03/31 から活動

blanco Framework の特長

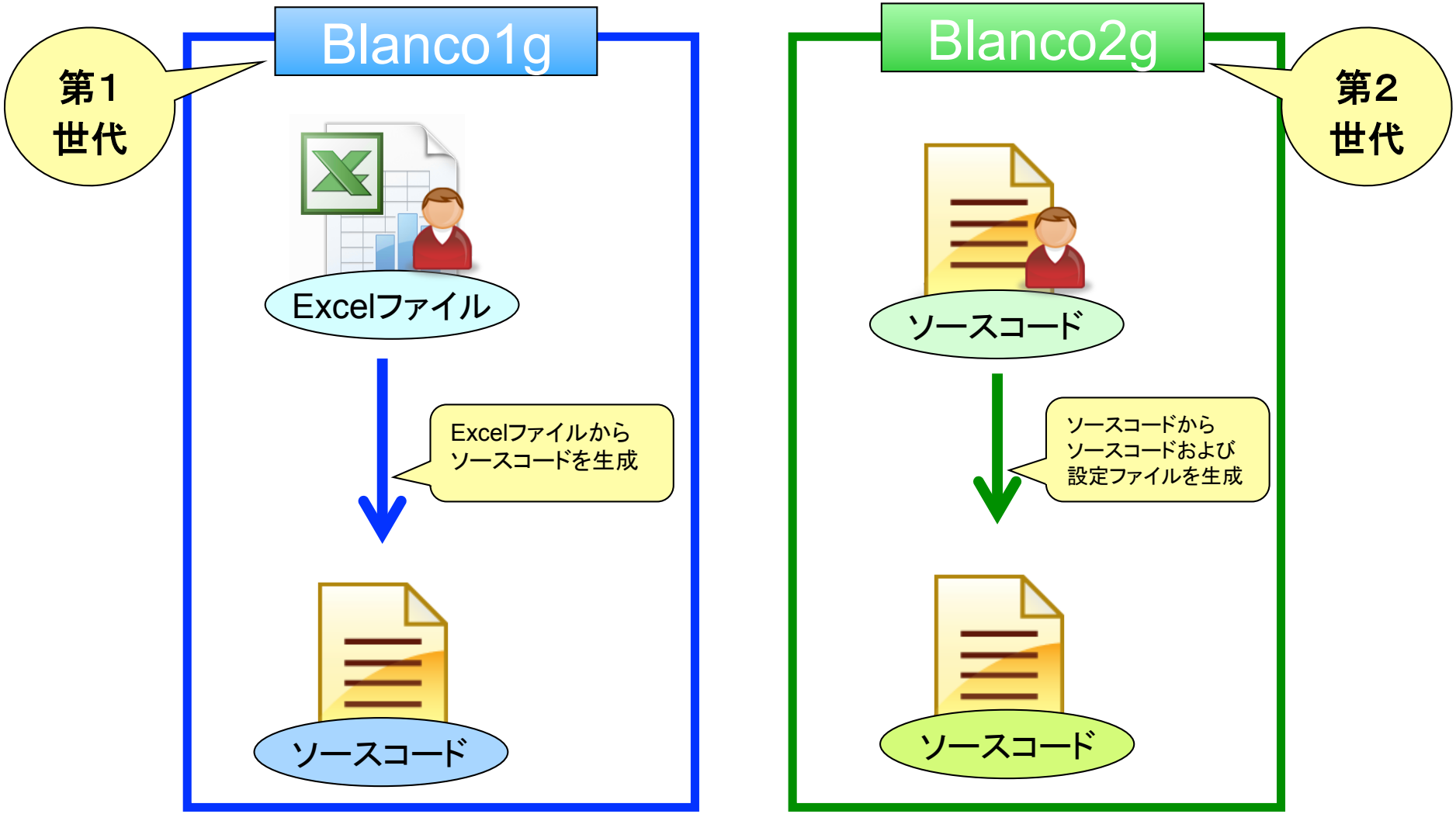
- 低コスト / 容易性
  - GNU LGPL
  - 単純な構造、導入が簡単
  - シンプルな操作性
- 複数言語対応【Blanco1g】
  - Java, C#.NET, JavaScript, PHP, VB.NET, Ruby, Python, Pascal （☆Blanco2g は Java のみ対応）
- 導入の柔軟性
  - 表計算ソフトで記入するだけ【Blanco1g】
  - 特殊な実行時ライブラリが不要
  - 部分的導入も可能
  - 他のフレームワークとも併用可能

適用工程

- 基本的に設計・製造工程を対象

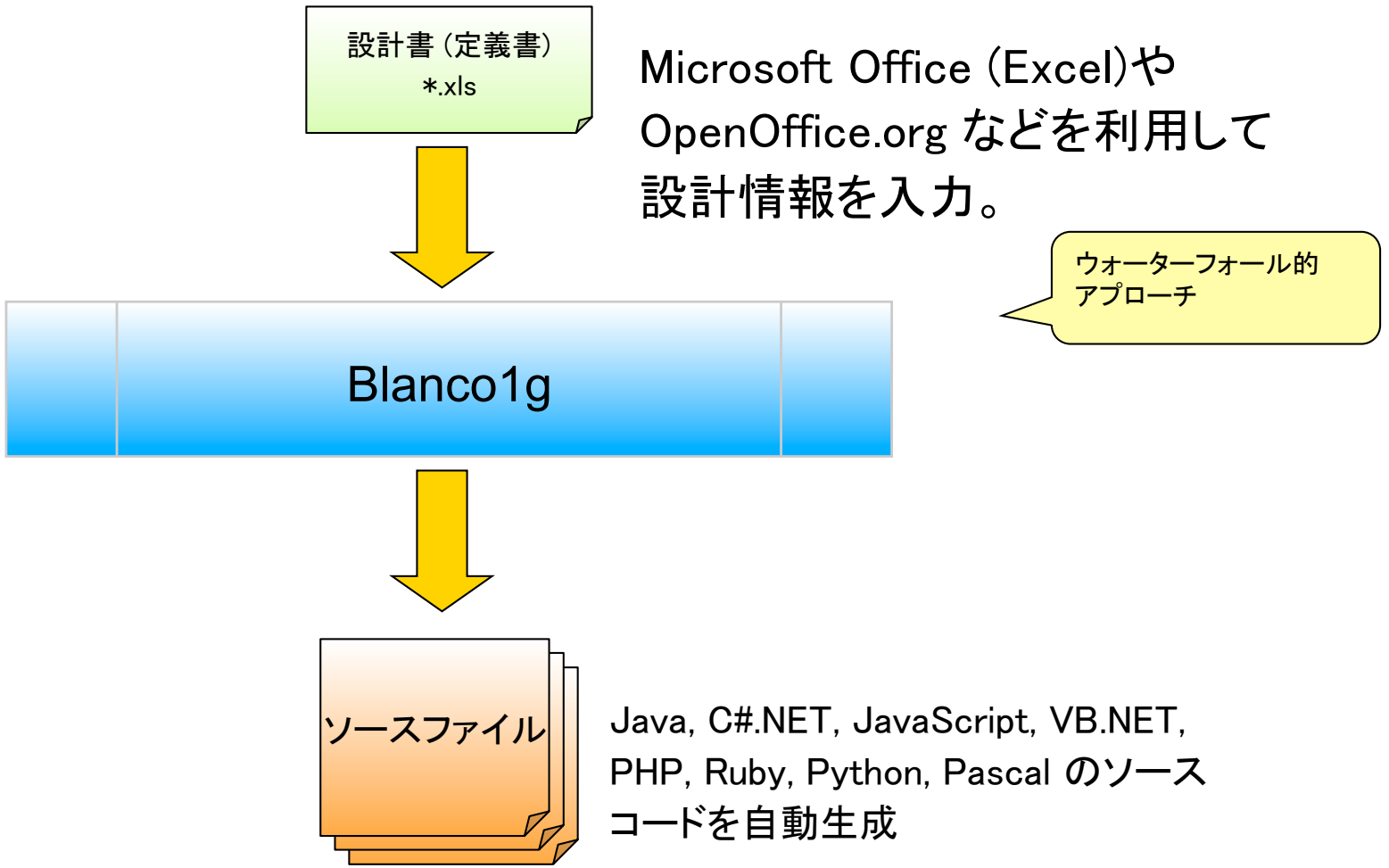


blanco Framework の自動生成



blanco Framework は2系統のソースコード自動生成を提供します

Blanco1g の概念図



Microsoft Office (Excel)や
OpenOffice.org などを利用して
設計情報を入力。

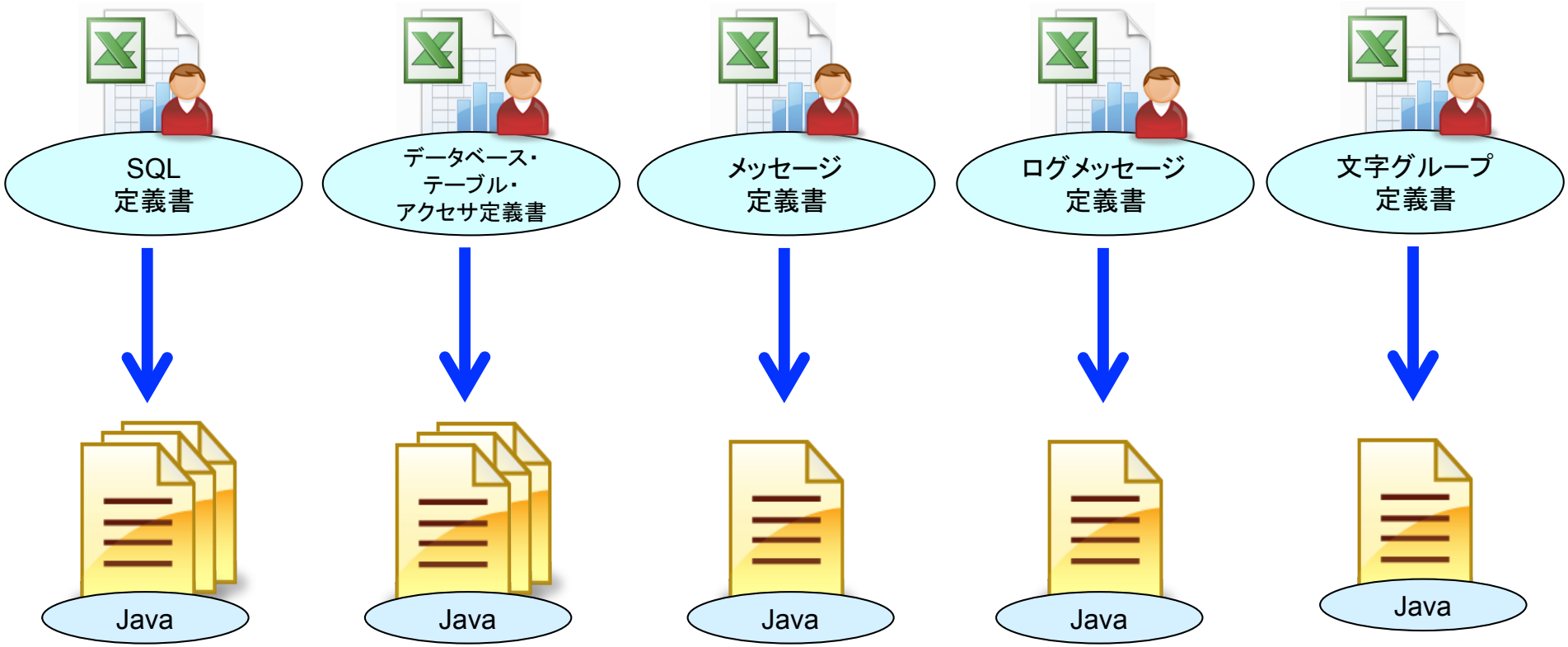
ウォーターフォール的
アプローチ

Java, C#.NET, JavaScript, VB.NET,
PHP, Ruby, Python, Pascal のソース
コードを自動生成

複数の自動生成の集合体

Blanco1g

小分けにされて複雑度の下がった複数の自動生成によって構成されます



設計書（定義書）
*.xls

ソースコード自動生成ツール

ソースファイル

反復的な

ソースコード自動生成

設計書とソースコードが常に一致。

仕様変更が楽になる。

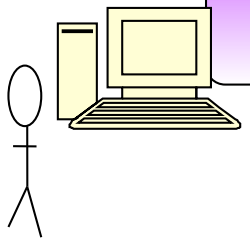
設計書どおりのソースコードが作られる

人為的ミスが入りにくい。

設計書とソースコードは機械的に一致する / 一致させる

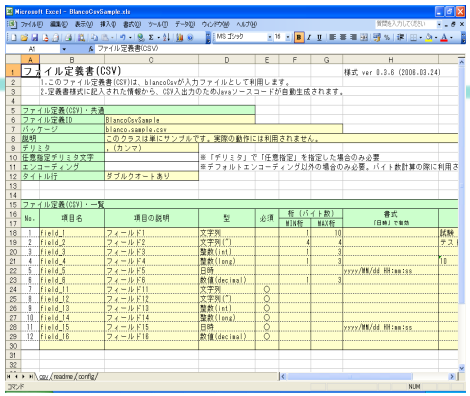
Microsoft Excel

OpenOffice.org



Microsoft Office (Excel)や
OpenOffice.org などを利用して
設計情報を入力。

記入



設計書(定義書) *.xls

- ファイル定義書
- SQL定義書
- メッセージ定義書
- 電文定義書
- バリューオブジェクト定義書

…など

Blanco1g

自動生成

ソースファイル

Java

ソースファイル

C#.NET

ソースファイル

JavaScript

ソースファイル

VB.NET

ソースファイル

PHP

ソースファイル

Ruby

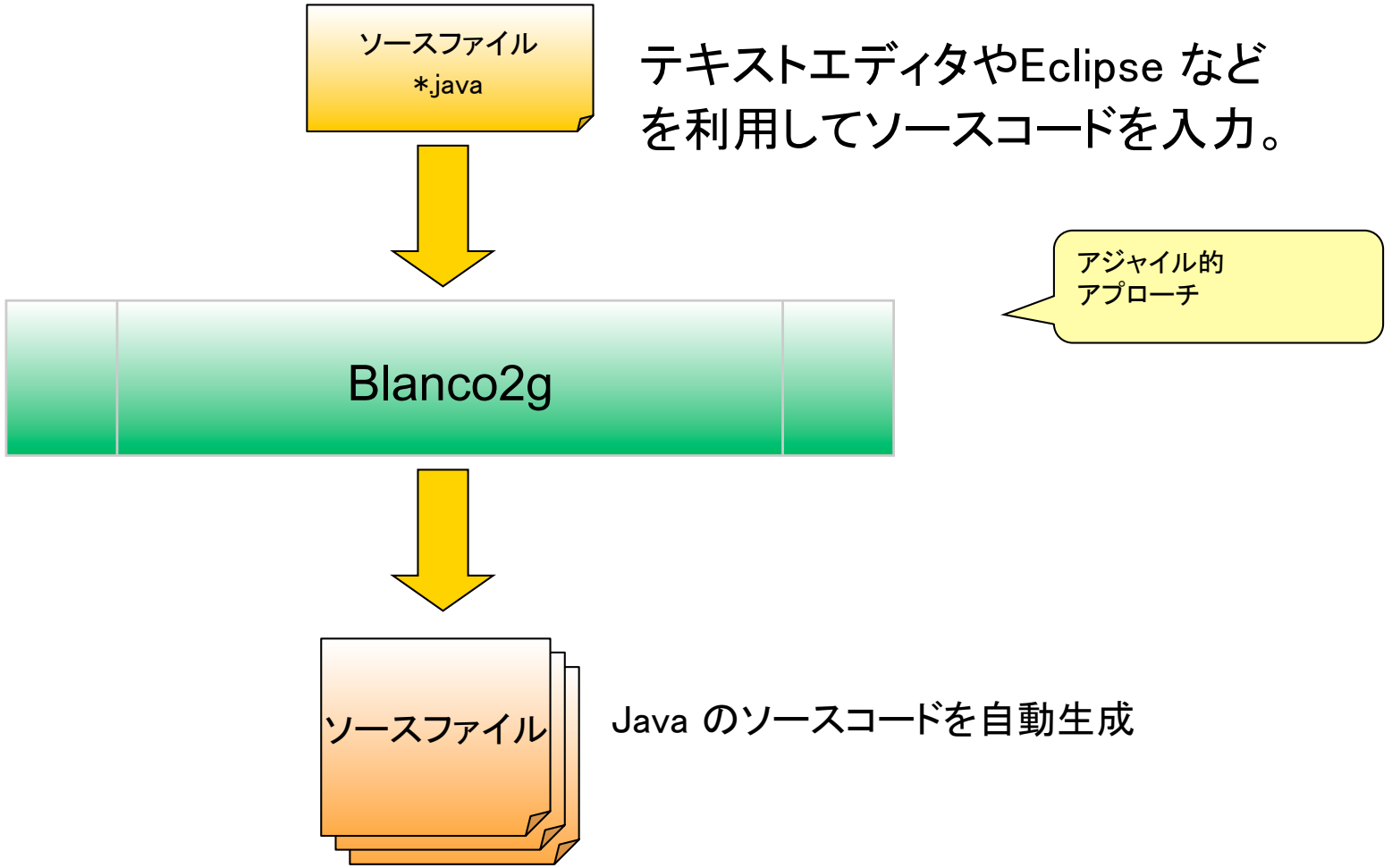
ソースファイル

Python

ソースファイル

Pascal

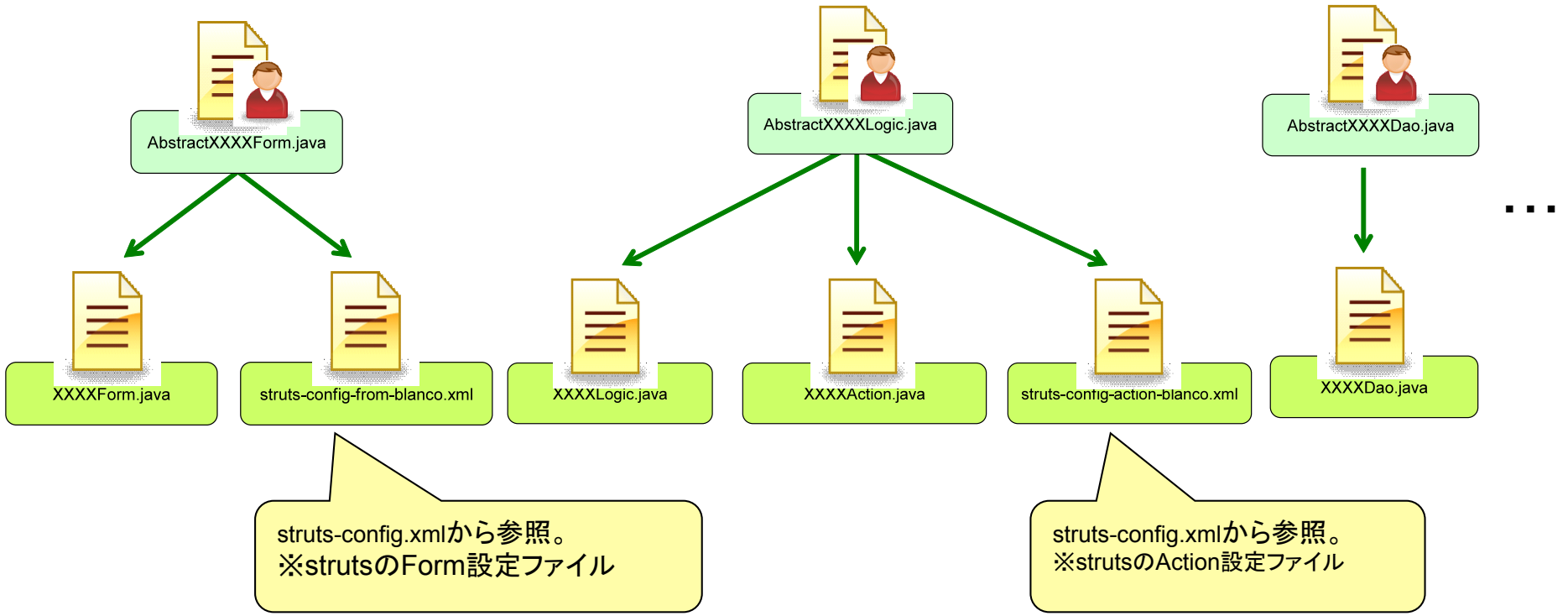
Blanco2g の概念図

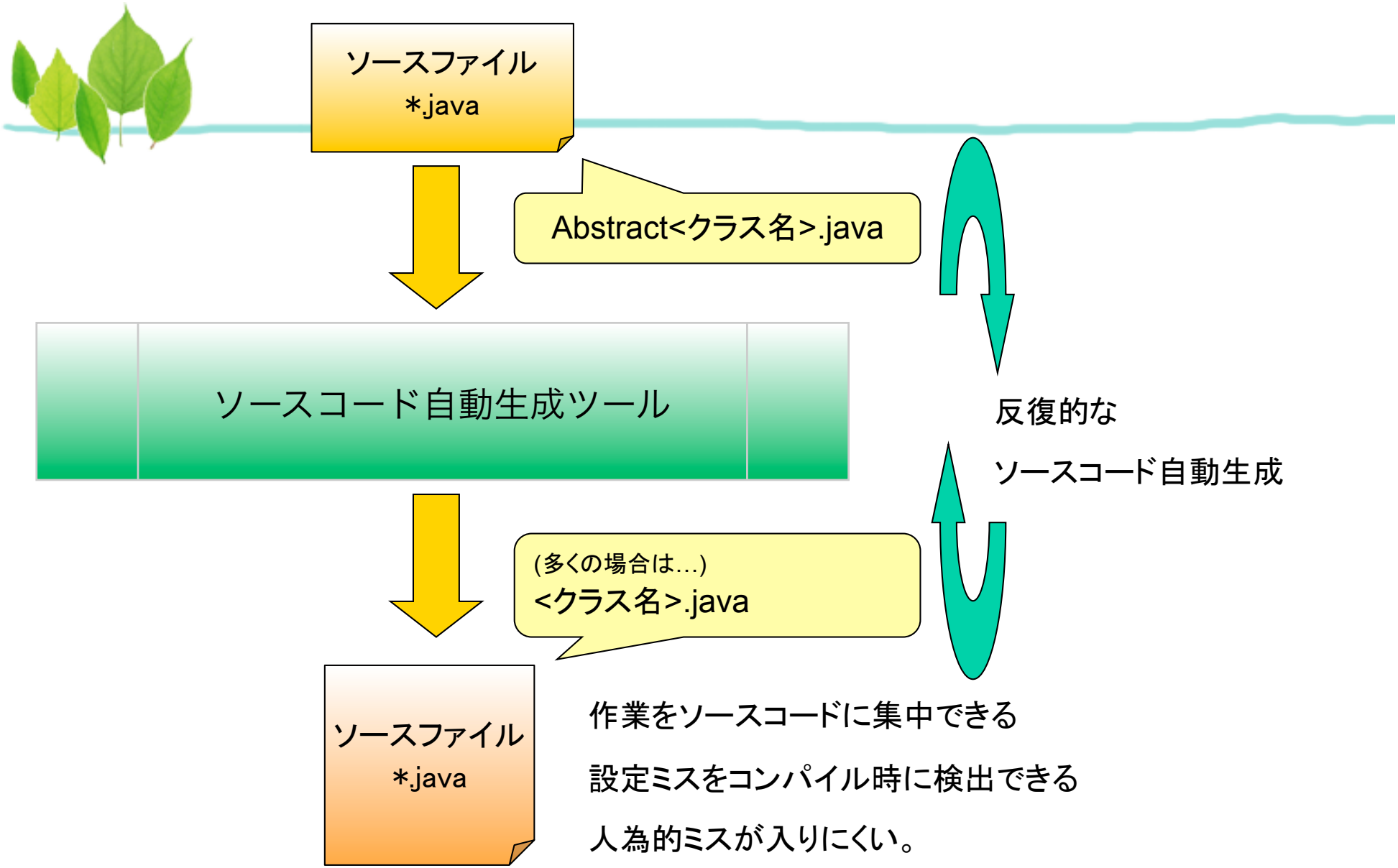


Blanco2g の典型的な利用例

Blanco2g

ソースコード上のアノテーション情報をもとに
ソースコードおよび設定ファイルを生成





設定情報をビルド時に伝播し、誤りはコンパイル時に検出する

blanco Framework 導入のメリット

- 生産性向上
  - 単純作業を自動化
  - 工数を削減
- 可読性向上
  - 均質なソースコードを自動生成
- 保守性向上
  - 【Blanco1g】定義書の変更が即座にソースコードに反映
  - 【Blanco2g】ソースコードの変更が他のソースコードに伝播
  - 仕様と実装が常に一致
- 品質向上
  - 自動化・均質化・変更の確実な実施などの結果、品質が向上

blanco Framework 導入のデメリット

- ソースコード規模増加
  - 自動生成されたソースコード規模の考え方の整理が必要
- ソースコード自動生成の時間
  - 必要に応じてプロジェクト分割などの工夫が必要

Blanco1g と Blanco2g の併用

- Excel ブックからの自動生成【Blanco1g】と
  ソースコードからの自動生成【Blanco2g】を
  適材適所に柔軟に使い分け

- Excel における表現が有用な自動生成については Blanco1g！
- ソースコード上のアノテーション表現が有用な自動生成については Blanco2g！

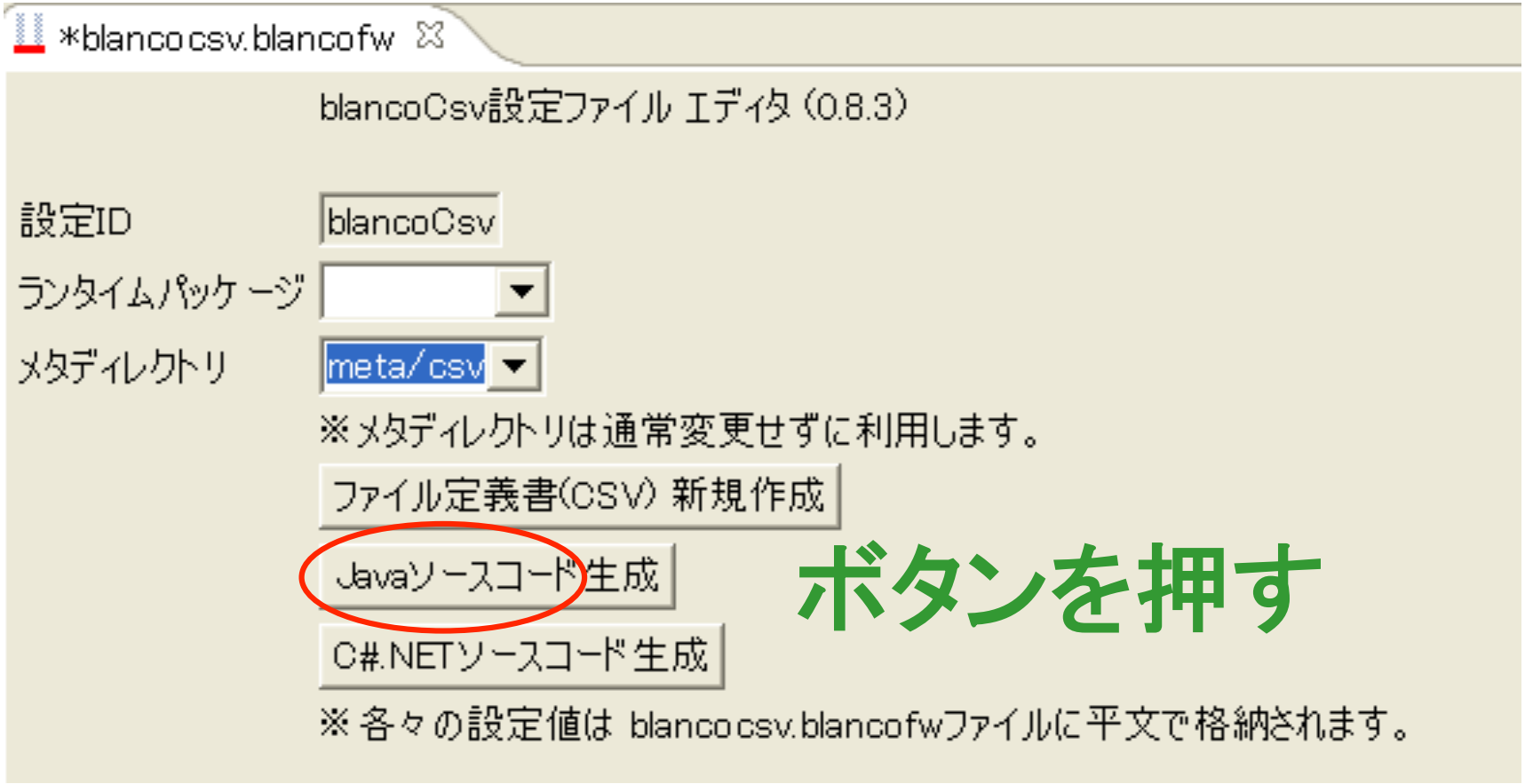
Blanco1g の例 (1)

- 所定の様式に必要事項を記入

	A	B	C	D	E	F	G	H	I
1	ファイル定義書(CSV)							様式 ver 0.3.6 (2006.03.24)	
2		1.このファイル定義書(CSV)は、blancoCsvが入力ファイルとして利用します。							
3		2.定義書様式に記入された情報から、CSV入出力のためJavaソースコードが自動生成されます。							
4									
5	ファイル定義(CSV)・共通								
6	ファイル定義ID	BlancoCsvSample							
7	パッケージ	blanco.sample.csv							
8	説明	このクラスは単にサンプルです。実際の動作には利用されません。							
9	デリミタ	,(カンマ)							
10	任意指定デリミタ文字		※「デリミタ」で「任意指定」を指定した場合のみ必要						
11	エンコーディング		※デフォルトエンコーディング以外の場合のみ必要。バイト数計算の際に利用されます						
12	タイトル行	ダブルクオートあり							
13									
14									
15	ファイル定義(CSV)・一覧								
16	No.	項目名	項目の説明	型	必須	桁（バイト数）		書式	規定値
17						MIN桁	MAX桁	「日時」で有効	
18	1	field_1	フィールド1	文字列		1	10		試験
19	2	field_2	フィールド2	文字列(")		4	4		テスト
20	3	field_3	フィールド3	整数(int)		1	3		
21	4	field_4	フィールド4	整数(long)		1	3		10
22	5	field_5	フィールド5	日時				yyyy/MM/dd HH:mm:ss	
23	6	field_6	フィールド6	数値(decimal)		1	3		
24		field_11	フィールド11	文字列	○				
25		field_12	フィールド12	文字列(")	○				
26		field_13	フィールド13	整数(int)	○				
27		field_14	フィールド14	整数(long)	○				
28		field_15	フィールド15	日時	○			yyyy/MM/dd HH:mm:ss	
29		field_16	フィールド16	数値(decimal)	○				
30									

Blanco1g の例 (2)

- **blanco〇〇プラグイン**



Blanco1g の例 (3)

- ソースコードが自動生成される

BlancoCsvSampleCsvReader.cs

```
blanco.sample.csv.io.BlancoCsvSampleCsvReader | ReadRecord()
            fReader = arg;
            fSimpleDateFormatField5 = "yyyy/MM/dd HH:mm:ss";
            fSimpleDateFormatField15 = "yyyy/MM/dd HH:mm:ss";
        }

        /// <summary>次の一行を読み込みます。</summary>
        /// <returns>行オブジェクトを返します。もはや行が無い場合には nullを返します。</returns>
        public BlancoCsvSampleCsvRecord ReadRecord()
        {
            string line = fReader.ReadLine();
```

BlancoCsvSampleCsvReader.java

```
    /**
     * 次の一行を読み込みます<br>
     *
     * @return 行オブジェクトを返します。もはや行が無い場合には nullを返し
     * @throws BlancoCsvIOException 入力データが不正な場合など。
     * @throws IOException 連結先リーダで異常が発生した場合。
     */
    public BlancoCsvSampleCsvRecord readRecord() throws BlancoCsvIOExcept
        final String line = fReader.readLine();
        if (line == null) {
            // ファイルの終端に到達しました。
            return null;
        }
        fLineCounter++;
        final BlancoCsvSampleCsvRecord record = new BlancoCsvSampleCsvRec
        final StringReader reader = new StringReader(line);
        String tokenString = null;

        // 項目番号[1]項目名[field_1/フィールド1]
        tokenString = BlancoCsvRuntimeUtil.readToken(reader, ',', false);
        if (tokenString == null) {
            throw new BlancoCsvIOException("入力" + fLineCounter + "行目
        }
        // 任意項目。
        if (tokenString.length() == 0) {
            // 必須項目ではない文字列項目
            record.setField1(tokenString);
        } else {
            if (tokenString.getBytes().le
                throw new BlancoCsvIOExce
            }
            if (tokenString.getBytes
                throw new BlancoCsvIO
            }
```

アウトライン

```
blanco.sample.csv.io
インポート宣言
BlancoCsvSampleC
fReader : Buffe
fLineCounter : ield_1/フィールド1]
fSimpleDateFor
fSimpleDateFor
BlancoCsvSam
readRecord()
getLineCounter
close()
```

C#.NET

Java

Blanco1g の例 (4)

SQL定義書

様式 ver 2.0 (2009.11.12)

1.このSQL定義書は、blanco Frameworkが入力ファイルとして利用します。
2.定義書様式に記入された情報から、データベース入出力のためJavaソースコードが自動生成

SQL 定義書【blancoDb】

SQL定義・共通	
SQL定義ID	BlancoDbSampleUpdateCategory
パッケージ	my.app
説明	カテゴリ名の更新
SQLタイプ	実行型
期待する処理件数	必ず１件処理
スクロール属性	FORWARD_ONLY
更新可能属性	なし
動的SQL	使用しない
パラメータのBean化	しない

※期待する実行結果件数を指定
※検索型の場合：カーソルのスクロール方向を指定
※検索型の場合：更新用検索を行うかどうかを指定

SQL インジェクションが
発生しにくい仕組み

SQL定義・入力パラメータ

No.	パラメータID	タイプ	パラメータ名	参考
1	cat_name	java.lang.String	カテゴリ名	
2	upd_user_cd	java.lang.String	更新ユーザー・コード	
3	upd_date	java.util.Date	更新日時	
4	cat_id	int	カテゴリID	
5	orig_upd_date	java.util.Date	変更前の更新日時	

SQL定義・出力パラメータ　※SQLタイプが「呼出型」の場合にのみ有効

No.	パラメータID	タイプ	パラメータ名

SQL定義・詳細説明

指定されたカテゴリID のカテゴリー名を更新します。

SQL定義・SQL文　補足説明

```
UPDATE
        t_cat
    SET
        cat_name = #cat_name
        ,upd_user_cd = #upd_user_cd
        ,upd_date = #upd_date
    WHERE
        cat_id = #cat_id
        AND upd_date = #orig_upd_date
```

ごく普通の SQL 文を記述
☆RDBMS の固有 SQL 文法も利用可能

Blanco1g の例 (5)

メッセージ定義書
【blancoMessage】

定義内容が
そのまま実装と連動

プレースホルダーの置換忘れが発生しない仕組み

メッセージ定義書　様式 ver 0.4.0 (2007.12.07)

1.このメッセージ定義書は、blancoMessageなどが入力ファイルとして利用します。
2.定義書様式に記入された情報から、メッセージを扱うためのソースコードが自動生成されま

メッセージ定義・共通	
メッセージ定義ID	BlancoLogProgram
パッケージ	blanco.log.resourcebundle
説明	blancoLogのメッセージクラス。プログラム中で利用されるメッセージを格
サフィックス	Message
ID埋め込み	○

メッセージ定義・一覧

No.	キー	レベル	文字列
1			【BlancoBinaryLog】
2	MBLGB001	ERROR	[MBLGB001] バイナリログファイルの出力先となる基準ディレクトリ[{0}]の作成に失敗しました。
3	MBLGB002	ERROR	[MBLGB0 (年月日つき)[{0}]の作成に失敗しました。
4	MBLGB003	ERROR	[MBLG 新規作成に失敗しました。
5	MBLGB004	ERROR	[MBLGBU 新規作成時に例外が発生しました。:{1}
6	MBLGB005	ERROR	[MBLGB0 新規作成時に例外が発生しました。:{1}
7	MBLGB006	ERROR	[MBLGB0 の書き込みに失敗しました。
8	MBLGB007	ERROR	[MBLGB0 の書き込み時に例外が発生しました。:{1}
9	MBLGB008	ERROR	[MBLGB008] バイナリログファイルを出力するファイル[{0}]への書き込み時に例外が発生しました。:{1}
10	MBLGB009	ERROR	[MBLGB009] バイナリログファイルを出力するファイル[{0}]のロック取得に失敗しました。:{1}
11	MBLGB010	ERROR	[MBLGB010] バイナリログファイルを出力するファイル[{0}]のロック解放に失敗しました。:{1}
12	MBLGB011	ERROR	[MBLGB011] バイナリログファイルを出力するファイル[{0}]のチャネルクローズに失敗しました。:{1}
13	MBLGB012	ERROR	[MBLGB012] バイナリログファイルを出力するファイル[{0}]のクローズに失敗しました。:{1}
14			
15			【blanco.log.logging】
16			「BlancoLogLog4jHandler」
17	MBLGLG01	ERROR	[MBLGLG01] Apache log4j クラスの読み込みに失敗しました。クラスパスに Apache log4j の jarファイルが適切に含まれていることを確認してください:{0}
18			
19			【blanco.log.custom】
20	MBLGCF01	ERROR	ThreadDeath によりログ展開できません。
21	MBLGCH01	ERROR	Java システムプロパティー[{0}]が指定されていま
22	MBLGCH02	ERROR	Java システムプロパティー[{0}]で指定されたディ
23	MBLGCH03	ERROR	予期せぬ入出力例外が発生:{0}
24	MBLGCH04	ERROR	ファイル[{0}]のロック取得できませんでした。
25	MBLGCH05	ERROR	指定のログ・ディレクトリー[{0}]は存在しません。
26	MBLGCH06	ERROR	指定のログ・ディレクトリー[{0}]は実際にはディ
27	MBLGCH07	ERROR	ログファイル[{0}]の作成に失敗しました。
28	MBLGCH08	ERROR	ログファイル[{0}]の作成に失敗しました。入出力例外が発生しました。:{1}
29	MBLGCH09	ERROR	予期せぬ入出力例外が発生:{0}

Blanco1g がもたらすメリット

- 設計情報と実装の一体化・一致化
- Excel や OpenOffice.org といった、一般的なツールをもちいて設計情報の投入が可能

Blanco1g の情報源

- @IT: Excel からプログラムを作る多言語対応オープンソース
  http://www.atmarkit.co.jp/fjava/special/blanco/blanco_1.html

- blanco Framework @ sourceforge.jp
  http://sourceforge.jp/projects/blancofw/wiki/blancofw

- プロダクト一覧
  http://www.igapyon.jp/blanco/blancoproductlist.html

Blanco2g の例 (1)

- データベース接続の注入

DI コンテナ不要、XML 記述不要の
データベース接続注入

```
public String execute(ActionMapping mapping, SampleForm form, HttpServletRequest request,
HttpServletResponse response, @BlancoInject Connection conn) throws Exception {
```

- メソッド呼び出しにデータベース・トランザクション境界を注入
  - メソッド呼び出し直前
    - データベース接続を取得
    - データベース・トランザクションの開始
  - メソッド呼び出し直後
    - データベース・トランザクションのロールバック
    - データベース・トランザクションの解放

Blanco2g の例 (2)

- ゲッター・セッターの注入

```
@BlancoGetterSetter
protected int field1 = -1;
```

DI コンテナなどが不要の
ゲッター・セッター・メソッド注入

- 特殊な機構抜きで ゲッター・セッター・メソッドを注入

Blanco2g の例 (3)

- フィールド・バリデーションの注入

DI コンテナなどが不要の
フィールド・バリデーション・メソッド注入

```
@BlancoGetterSetter
@BlancoStrutsReset
@BlancoValidateMethodForStruts
@BlancoValidateRequired
@BlancoValidateLength(max = 5)
protected String field1 = "";
```

- 特殊な機構抜きで フィールド・バリデーション・メソッドを注入

- ☆その他のアノテーションについては、次の URL を参照http://sourceforge.jp/projects/blancofw/wiki/Blanco2g_Annotation

Blanco2g がもたらすメリット (1)

- アノテーションベースのソースコード自動生成による生産性向上
  - 開発の容易さを得られます【EoD】
  - ほとんどの XML 記述を不要にします【EoD】
  - 実行時例外の多くをコンパイル時に検出できます【自動生成】

Blanco2g がもたらすメリット (2)

- DI コンテナの類が不要
  - 特殊なランタイム・ライブラリなしで、フレームワークを導入できます

    ☆Spring Core や CDI コンテナなどのコンテナ類が不要!
  - 組み込み開発でも効果を発揮します
  - しかも DI コンテナとの併用ですら可能です

Blanco2g がもたらすメリット (3)

- 古い Java であっても最新の開発技法が利用できる!
  - 古い Java 実行環境 (Java SE 1.5.0 以降) で利用可能
  - 古い Java アプリケーションサーバーで利用可能
  - 維持運用案件への導入が容易
  - 古い環境において最新の Java EE 6 が利用できてしまっているようにすら錯覚させられる
  - 古い本番環境や維持運用案件での生産性向上にも最適!

Blanco2g がサポートする UI

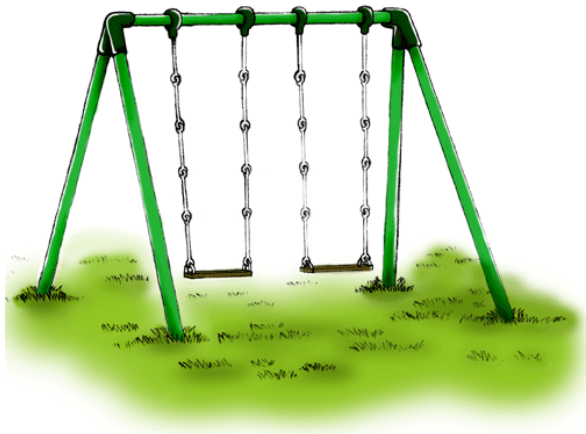
- 現時点で Blanco2g が対応する UI
  - Apache Struts 1.0
  - JSF 2.0 (含む Facelets)

blanco Framework を試してみてください

- ぜひ、blanco Framework を試してみてください。

- 試してみても、試さなくても、
  blog などで言及したり、
  Twitter でつぶやいたりしてください!

Eclipse 日本語化への取り組みについて



Eclipse の日本語化事情

- メニュー・メッセージについては、かなり進展
  ☆ヘルプなどについては、あまり進展せず

- Eclipse Babel Project
  http://www.eclipse.org/babel/

- Pleiades - Eclipse プラグイン日本語化プラグイン
  http://mergedoc.sourceforge.jp/pleiades.html

- Eclipse 日本語化言語パック (サードパーティ版)
  http://sourceforge.jp/projects/blancofw/wiki/nlpack.eclipse
  ☆blanco Framework が提供
- その他

Eclipse Babel Project (1)

- Eclipse 本家により各国語翻訳を提供するためのプロジェクト。
- 更新サイトおよび言語パック形式での翻訳リソース入手が可能
- ☆プロジェクトとしては Incubation Phase にあります。

- メニュー・メッセージの日本語翻訳ということでは、
  かなりの完成度 (翻訳率) に到達しました。
  ☆ボランティアの翻訳者の方々の努力に感謝!

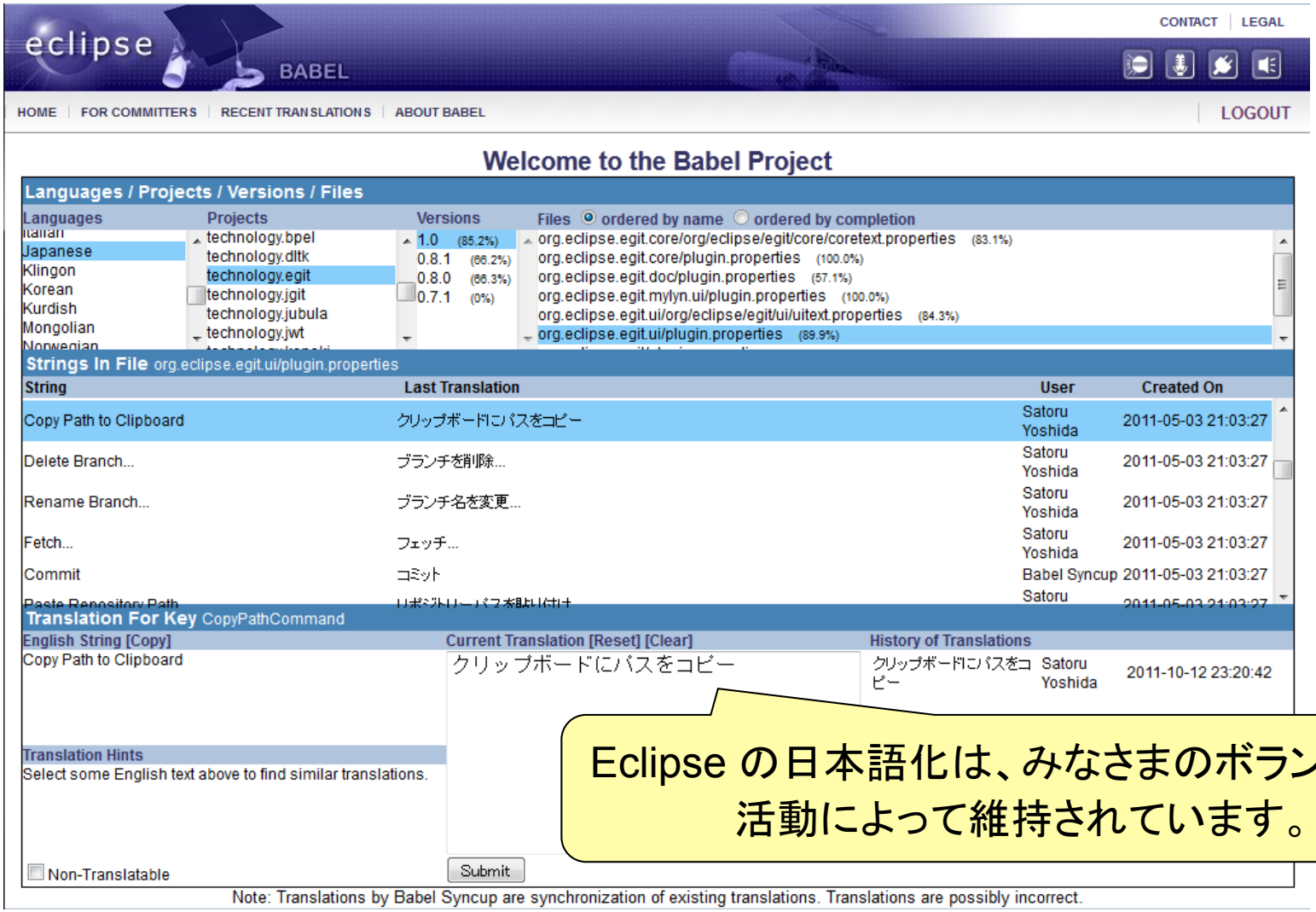
- 過去の経緯については、OSC 2008 Tokyo/Spring で NECソフトの森素樹さんの発表『Eclipse日本語言語パック　開発プロジェクト』を参照ください.
  http://www.ospn.jp/osc2008-spring/material/d1-b2fb-2-necsoft.pdf
  http://www.ospn.jp/osc2008-spring/modules/eguide/event.php?eid=3

Eclipse Babel Project (2)

- 翻訳に参加するには
- Eclipse の Bugzilla アカウントを取得してから Babel にログイン!

Eclipse Babel Project (3)

- Web ブラウザーを利用して翻訳を進めます



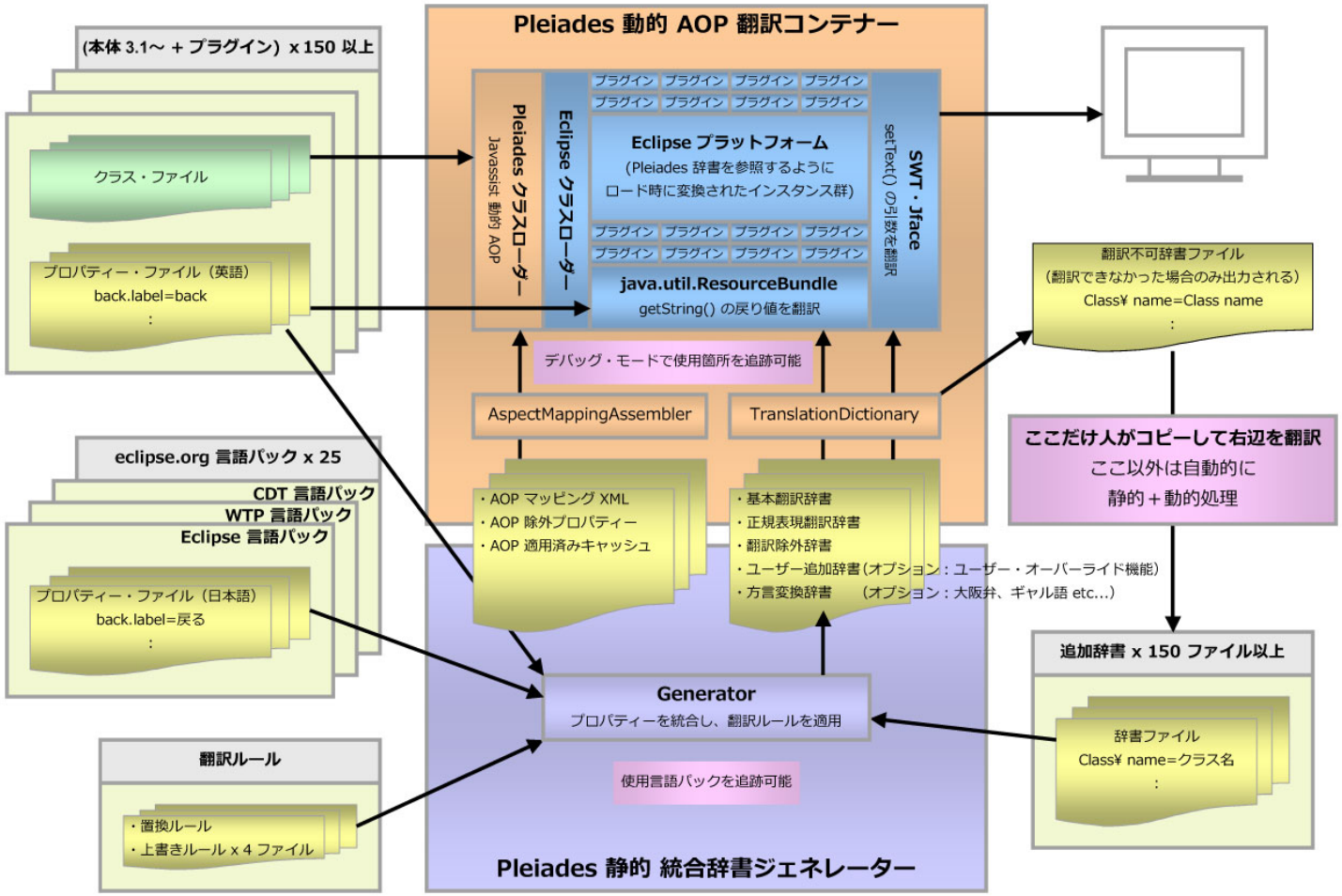
Eclipse の日本語化は、みなさまのボランティア活動によって維持されています。

Pleiades (1)

- 「Pleiades - Eclipse プラグイン日本語化プラグイン」
- Pleiades (プレアデス) は Java アプリケーションを AOP により動的翻訳 (実行時に翻訳) するためのツールです。

Pleiades (2)

- 実行時動的バイトコード変換というアーキテクチャを採用



Pleiades (3)

- Eclipse および関連プロジェクトのみならず、サードパーティ製 Eclipse プラグインや、果ては未知のプラグインまで日本語化を実現。
- すごく便利です。
- Eclipse 本家からの言語パック提供が Eclipse Babel 熟成までの長いあいだ途絶していたという経緯もあって、現在も Eclipse 日本語化において Pleiades はデファクトスタンダードの位置にあると思われます。

- ☆私も Pleiades の翻訳に微力ながら貢献しています。

Eclipse 日本語化言語パック (サードパーティ版) (1)

- Pleiades の翻訳機能および辞書をベースに、Eclipse 日本語化言語パックを作成
- blanco Framework の活動の一環として、この日本語化言語パックを提供
- 言語パック作成の過程でおこなわれた翻訳成果を Pleiades に随時フィードバック
  (☆Eclipse Babel に翻訳成果フィードバックの実績あり)

Eclipse 日本語化言語パック (サードパーティ版) (2)

- Eclipse Babel が軌道に乗ってきたため、「Eclipse 日本語化言語パック (サードパーティ版)」の存在意義は 若干薄れています (苦笑)
- 少人数の翻訳による、揺れの少ない翻訳成果を提供します。
- 最近、翻訳率が下降傾向にあります。

Eclipse の日本語化手法まとめ

- 翻訳作業モデルを『伽藍とバザール』にてらしあわせてみると,,,
  - 「Eclipse Babel Project」は典型的な「バザール」モデル
  - 「Pleiades - Eclipse プラグイン日本語化プラグイン」および「Eclipse 日本語化言語パック (サードパーティ版)」は「伽藍」モデル?
- ☆その他にも、Eclipse 日本語化の取り組みをおこなわれておられる方々がいらっしゃります。

- 日本語化手法は、用途や目的により使い分けられることでしょう。

- ☆メニュー・メッセージ以外の部分についての各国語化について、残念ながら、課題として引き続き残っています。

おわり