Practice Makes Perfect

The Ashikunep Kotan 伍

The Ashikunep Kotan
計画の全貌

2007年5月27日 SeasarCon 2007春 @法政大学

株）グルージェント 代表取締役社長
特）Seasarファウンデーション 代表理事
The Ashikunep Kotan 運営委員
栗原 傑享（くりはら まさたか）

ポジションペーパー

Practice Makes Perfect
the Ashikunep Kotan 伍

- The Ashikunep Kotan
  - Seasarファウンデーション傘下
    - マルチブランド政策の適用第一号
  - 本格活動前夜
    - コミッタ8名（内、7名がグルージェント所属）
    - リリース物件1件、しかしα以前コンセプト版
    - コミッタ以外のユーザーはおそらく0名
  - コンセプト

すべての価値観において「漢」を最上とするOSS開発コミュニティの実現。

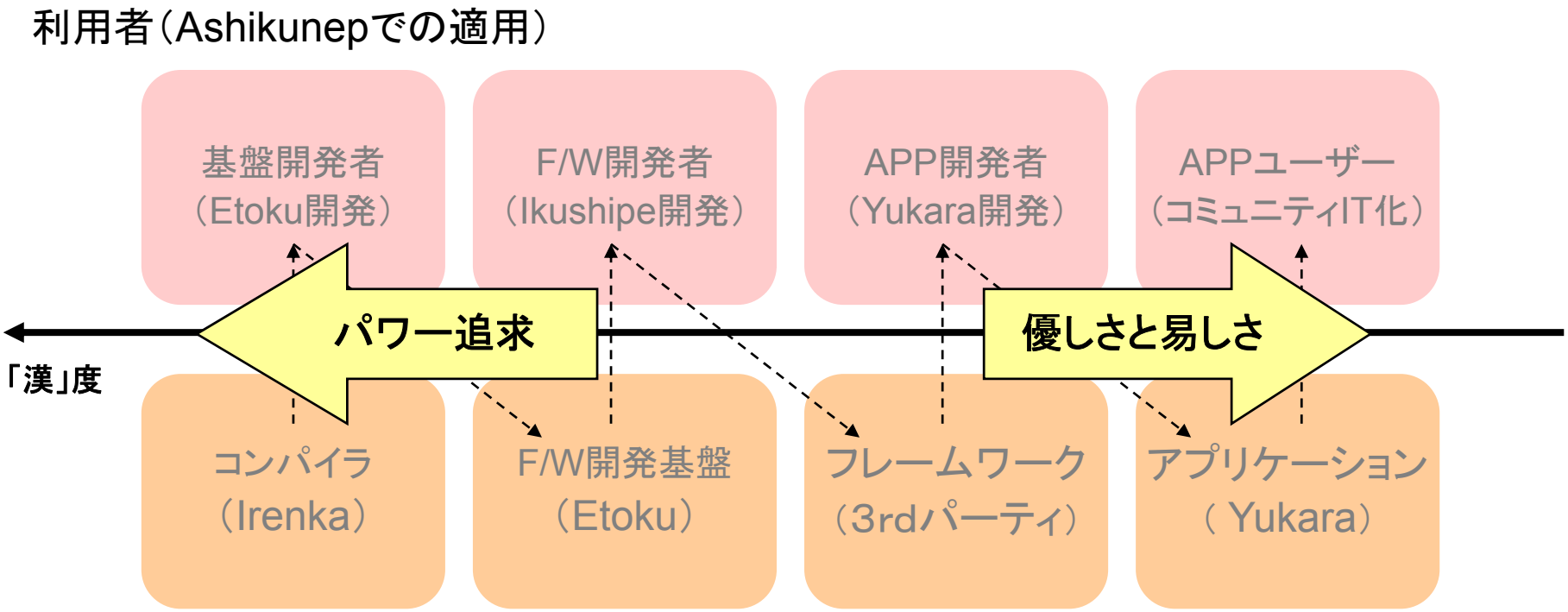
http://www.ashikunep.org/

Practice Makes Perfect the Ashikunep Kotan 伍

Ashikunepのプロダクト計画

Practice Makes Perfect the Ashikunep Kotan 伍

- 漢気あふれるプロダクトポートフォリオの実現
  - 利用者カテゴリの「漢」度に半比例して優しく易しく
  - 最も「漢」なレベルでは、最大のプログラミングパワー実現を目指す
  - しかし残念ながら漢には易しさは提供できない。JavaVMに優しくする



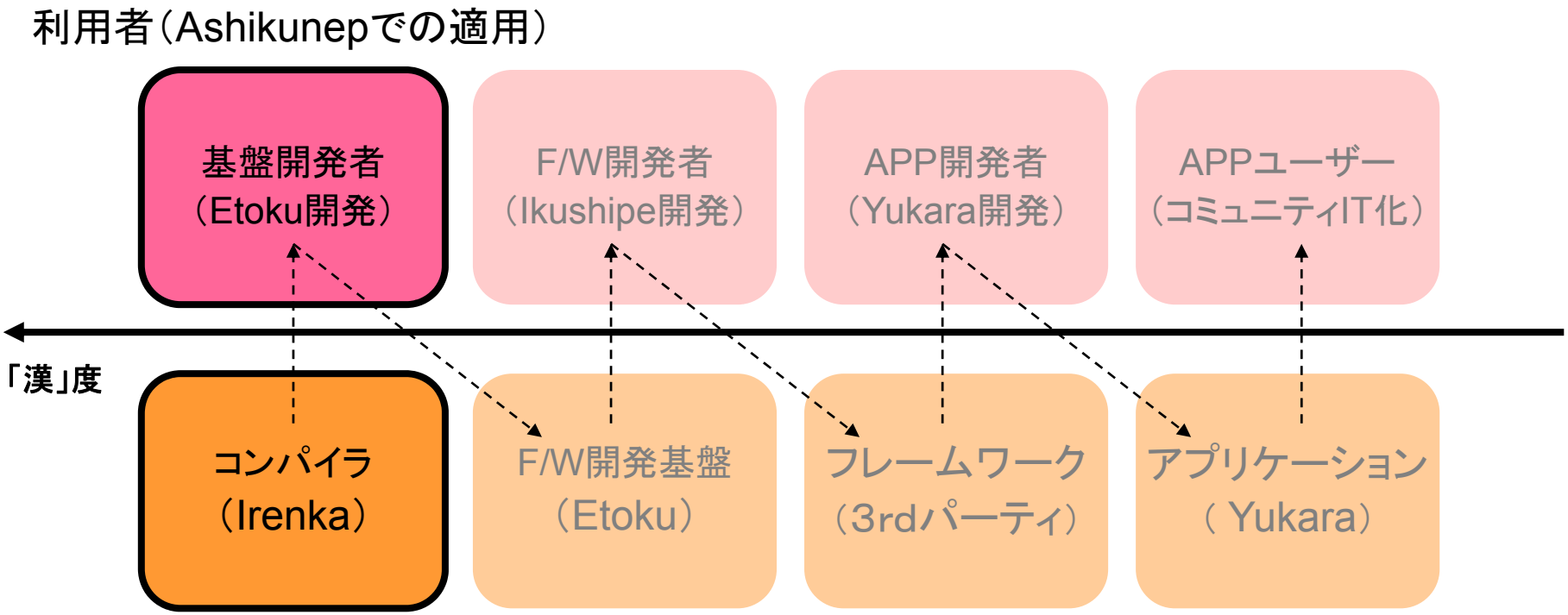
コンパイル時ソリューション

Practice Makes Perfect the Ashikunep Kotan 伍

- 「今日やれることは今日やる」
  - 昨日：F/W開発時
  - 今日：アプリケーション開発時
    - 開発時に解決しておくやり方
      - ソースコードジェネレート
      - コンパイル時バイトコードエンジニアリング
  - 明日：アプリケーション実行時
    - 実行時に解決を遅延するやり方
      - Javaリフレクション
      - 実行時バイトコードエンジニアリング
      - XML等コンフィギュレーション
- メタプログラミング
  - ソースコード上のヒントから、書かれている以上の内容のバイトコードを生成する
  - 人間が1行書いたら、10行～100行の働きを機械にさせる

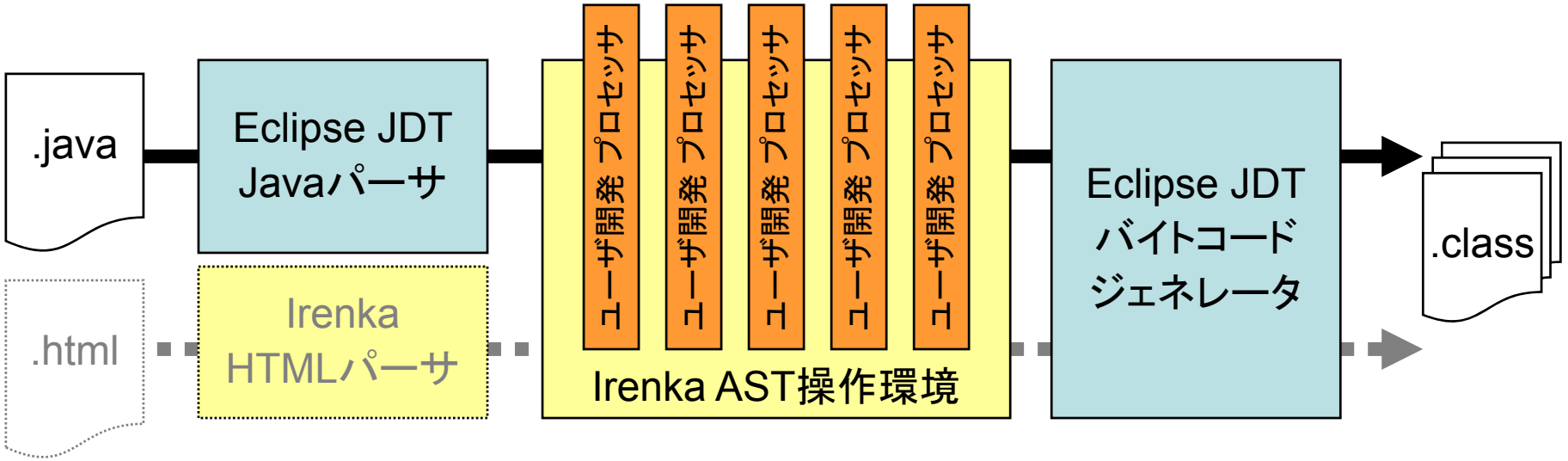
「コンパイラ」カテゴリ

- JavaVMに優しく作る
  - コンパイラで施策することにより、パフォーマンス上での不利を未然に予防する
- メタプログラミング



Irenka

- Irenka
  - コンパイル時に介入して、ASTを操作する
    - APTおよびJSR269に類似
      - よりリッチな操作セットで式レベルでの操作が可能
      - APTには無いコードジェネレート支援機能を搭載　　it's new!
      - JavaDoc記法を用いたASTへのクエリー言語を搭載　　it's new!
    - Eclipse JDT上に構築
      - JDTプログラミングの苦行を改善
      - Eclipse外での利用を前提、Eclipse内では便利に利用できる
  - 非JavaソースのAST化を将来機能として予定
    - HTMLやExcelシートなど



Irenkaサンプル

プロセッサ例

```
/**
 * NotNullアノテーション付きメソッドの返値にnullチェックを行うコードを埋め込む
 * @when  self in method.body                          // methodを定義
 *              notNull in method.annotations          // notNullを定義
 *              method = {@link CtMethod}              // methodの型制約
 *              template = {@link #checkNull(Object)}  // テンプレート取得
 */
public void processMethod(CtReturn self,
        CtAnnotationInstance<NotNull> notNull, CtMethod template) {
    // return文の式部分と、アノテーション値を取り出す
    CtExpression<?> returnExpr = self.getResult();
    CtExpression<?> value = notNull.getValueMap().get("value");
    // 取り出した値を引数としてロジックテンプレートメソッドの呼び出し
    CtMethodInvocation<?> checkInvocation =
            template.newInvocation(returnExpr, value);
    // メソッド呼び出しをインライン化し、元のreturn文の式部分と置き換える
    CtExpressionStatement<?> inlineExpr = checkInvocation.inline();
    returnExpr.substitute(inlineExpr);
}
```

(1)ノードクエリーの発行

Practice Makes Perfect the Ashikunep Kotan 伍

@whenタグの値がクエリ
複数行の場合、AND条件 ①

selfはメソッド引数から、CtReturn型。
つまりreturn文。かつ、「method」の
ボディに含まれる。「self」という名前
のものが、変更可能対象のノード。 ②

notNullはメソッド引数から、NotNullと
いうアノテーションのインスタンス型。
methodの修飾の中にこれを含む。 ③

methodは、CtMethod型。つまりメ
ソッドである。 ④

```
/*
 * @when  self in method.body
 *              notNull in method.annotations
 *              method = {@link CtMethod}
 */
public void processMethod(CtReturn self,
        CtAnnotationInstance<NotNull> notNull, … ) { … }
```

```
/**
 * ターゲットメソッド例
 */
@NotNull(“ユーザー名がnull”)
protected String getUserName() {
    return userName;
}
```

③より、抽出される「notNull」 ⑤

④より、抽出される「method」 ⑥

②より、抽出される「self」 ⑦

(2)テンプレート取得

templateの型はメソッド引数よりCtMethod、つまりメソッド 1

Javadocリンクより、メソッドへのリンク指定。これをノードクエリ中に書くと、該当ノードをインジェクトする。 2

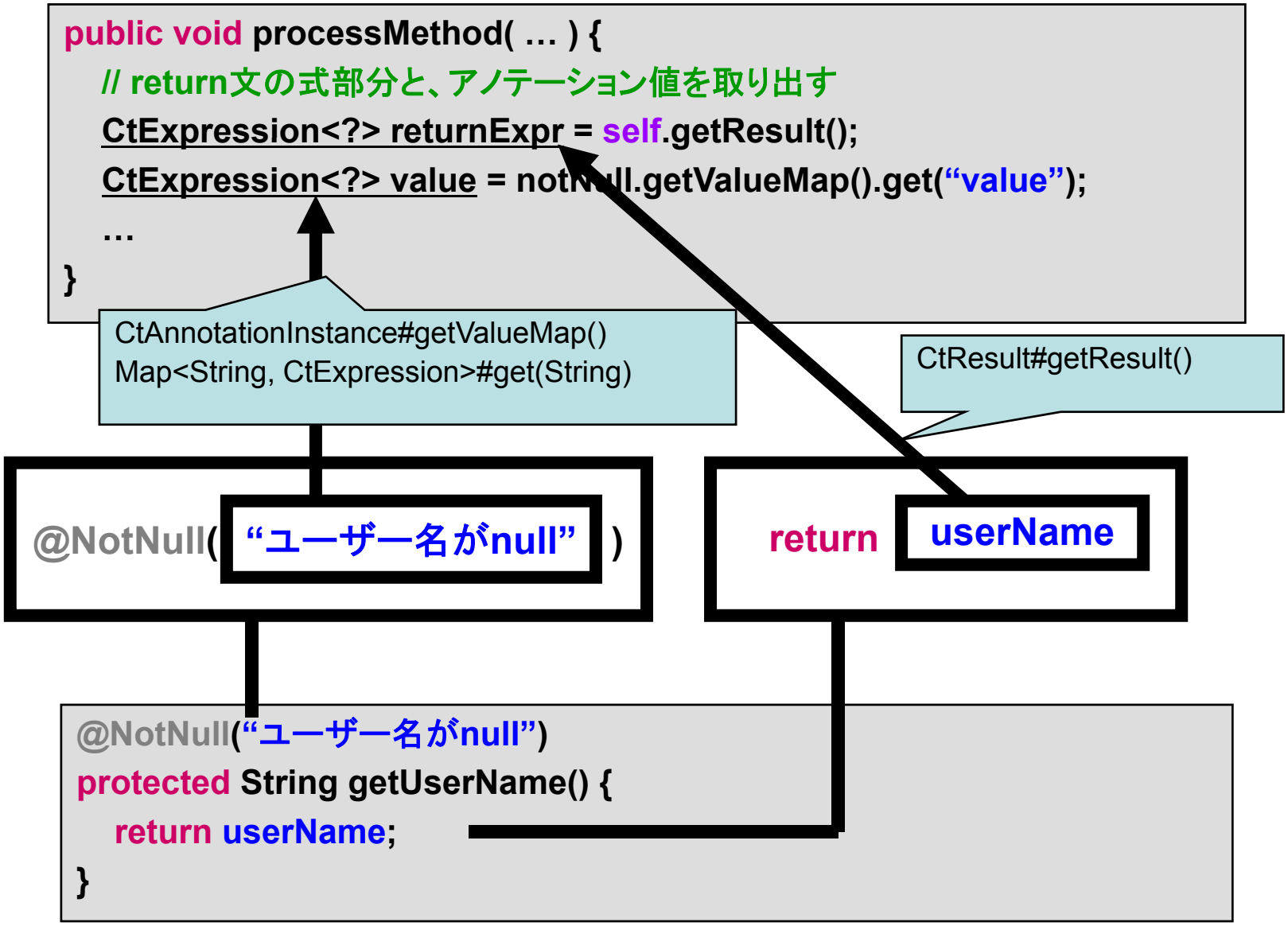
```
/**
 * @when
 *              template = {@link #checkNull(Object)}
 */
public void processMethod(..., CtMethod template) { ... }
```

```
/**
 * ロジックテンプレートメソッド。値がnullの時は例外を投げる。
 */
public <T> T checkNull(T test, String msg) {
    if(test == null) {
        throw new NullPointerException(msg);
    }
    return test;
}
```

②により、取得されるテンプレートメソッド。 3

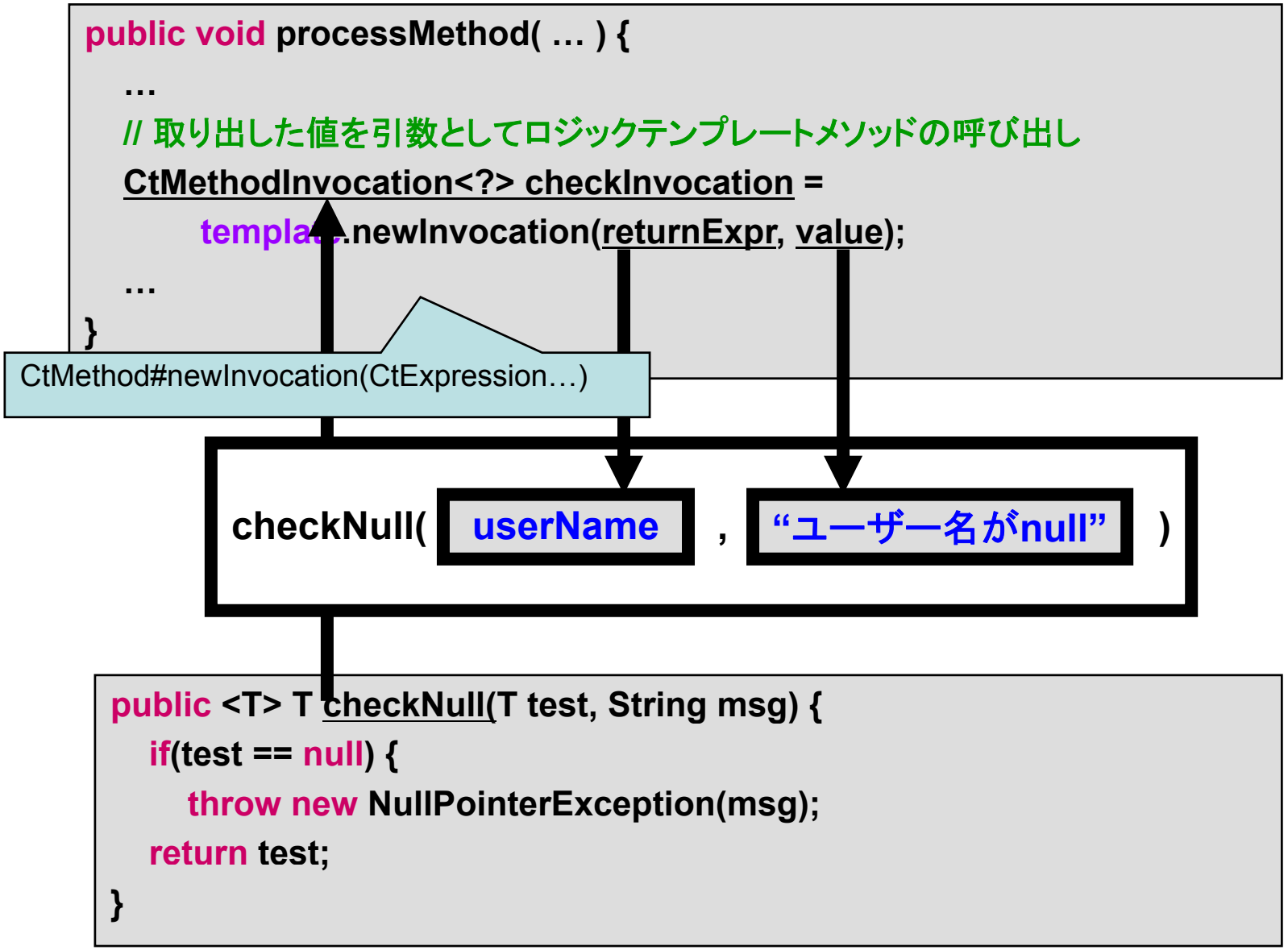
(3)ノードから値取得

Practice Makes Perfect the Ashikunep Kotan 伍



(4) テンプレート利用

Practice Makes Perfect the Ashikunep Kotan 伍



(5) インライン展開

Practice Makes Perfect the Ashikunep Kotan 伍

```
public void processMethod( … ) {
    …
    // メソッド呼び出しをインライン化し、元のreturn文の式部分と置き換える
    CtExpressionStatement<?> inlineExpr = checkInvocation.inline();
    …
}
```

```
public <T> T checkNull(T test, String msg) {
    if(test == null) {
        throw new NullPointerException(msg);
    }
    return test;
}
```

```
checkNull(userName , “ユーザー名がnull”)
```

```
String test = userName;
if(test == null) {
        throw new NullPointerException(“ユーザー名がnull”);
}
return test;
```

(6) 置き換え

Practice Makes Perfect the Ashikunep Kotan 伍

```
public void processMethod( … ) {
    …
    returnExpr.substitute(inlineExpr);
}
```

```
@NotNull(“ユーザー名がnull”)
protected String getUserName() {
    return userName;
}
```

```
String test = userName;
if(test == null) {
        throw new NullPointerException(“ユーザー名がnull”);
}
return test;
```

```
protected String getUserName() {
    String test = userName;
    if(test == null) {
        throw new NullPointerException(“ユーザー名がnull”);
    }
    return test;
}
```

変換結果

DEMO

量産ツールの必要性

Practice Makes Perfect the Ashikunep Kotan 伍

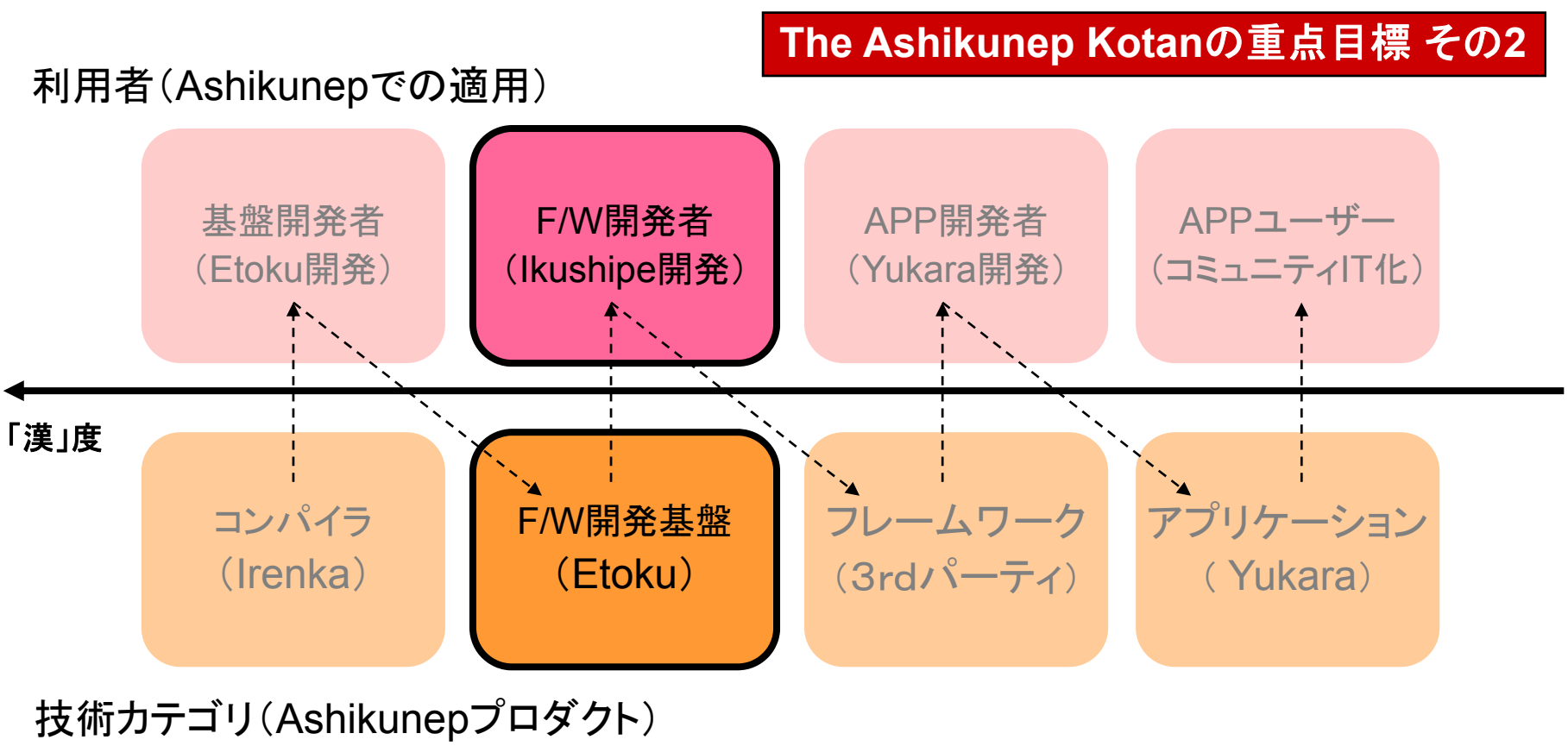
- コンパイル時ソリューションは難易度が高い
  - ASTクエリーおよび操作は概念がやや難しい
  - ASTの新規組み立ては手数が必要な操作
- コンパイル時に高度なことを行うF/W
  - ユーザーの書くソースコード上のヒントに反応
    - アノテーションに反応
    - 規約や簡易なAPIに反応
    - 行間の雰囲気に反応
  - ヒントから、大量のバイトコードを書き出す
    - 新規に大量のASTを構築する必要がある
    - 手組みでなく、あらかじめ用意したテンプレート利用
  - そういったF/Wを開発容易にするためのツールが期待されるのではないか？AshikunepはそこにEtokuを提案

「F/W開発基盤」カテゴリ

Practice Makes Perfect
the Ashikunep Kotan 伍

- 高度なフレームワークを量産する基盤提供
  - コンパイル時に施策する型のフレームワークを事情に応じて選択できるように、フレームワークの量産を助ける

The Ashikunep Kotanの重点目標 その2



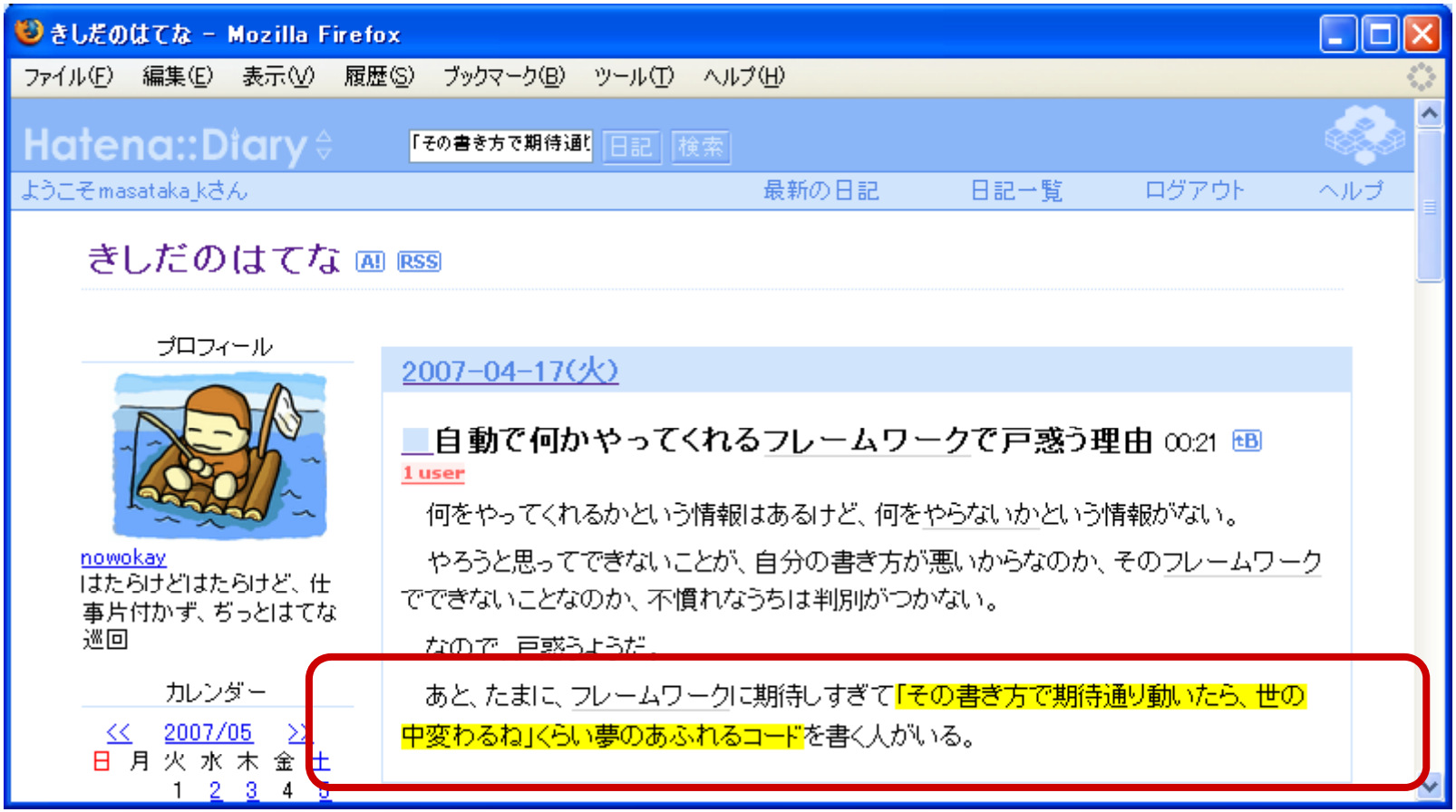
Etoku

- Javaテンプレートを用いてJavaクラスを自動生成するためのエンジン
  - Velocityライクな制御構文でJavaコードを加工
  - JavaDocによるジェネレートモデルの Javaコードのマークアップ

```
/**
 * @Template
 * @Refine servletClassName {@On  ServletTemplate}
 * @Refine className {@On SampleApplication}
 */
public class ServletTemplate extends HttpServlet implements SampleApplication {
      protected void service(HttpServletRequest req,HttpServletResponse res)
      throws IOException, ServletException {
            /**
             * @Refine "/" + context.trigger.welcomePage {@On "/index"}
             * @Refine context.trigger.welcomePage {@On index}
             */
            if (“/index”.equals(req.contextPath())) {
                  forward(index());
            }
            /**
             * @Foreach page {@In PageMethod}
             * @Refine "/" + page.pageName {@On "/login"}
             * @Refine page.method {@On login()}
             */
            if (“/login”.equals(req.contextPath())) {
                  forward(login());
            }
      }
   …
}
```

夢のあふれるコード

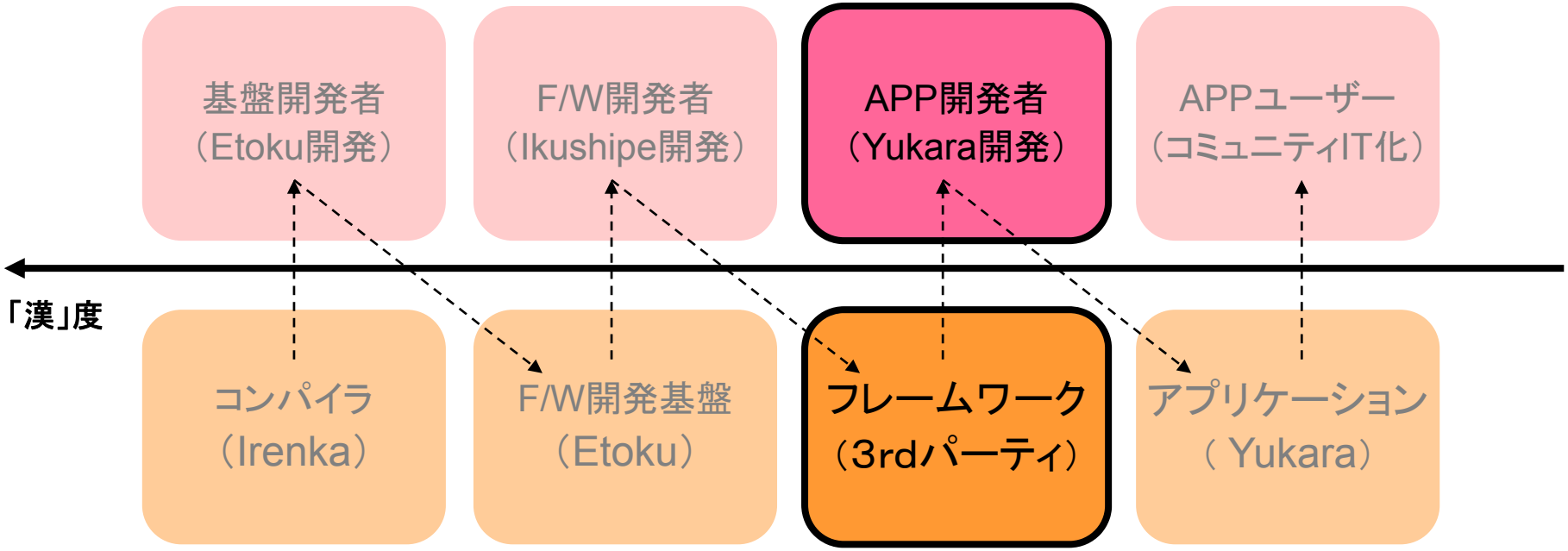
- それで世の中変わるなら、世の中を夢いっぱいにできないものだろうか



「フレームワーク」カテゴリ

Practice Makes Perfect
the Ashikunep Kotan 伍

- コンパイル時ソリューションの具体的事例
  - 利用者に、「夢のあふれるコード」を書かせられるような F/Wを具現化し、見せ、周囲に「夢」F/W量産を促す
  - 3rdパーティの参加を最も促したい領域



3rdパーティへの期待

Practice Makes Perfect the Ashikunep Kotan 伍

- IrenkaやEtokuにより、夢一杯のF/W量産
  - F/Wにより実現される夢は、人それぞれ
  - ぜひ、組織毎、もしくは案件毎にでもそういった、夢一杯F/Wを作って提供してあげてほしい
  - Seasarファウンデーション傘下コミュニティのコミッタ陣にもコンパイル時ソリューションの可能性について、ぜひ興味を持ってもらいたい
- Ashikunepは、基盤にまず注力して提供する
  - 応用サンプルとして、いくつかの典型機能例については、F/Wも提供していく（Ikushipe：WEBのページフロー制御）
  - 3rdパーティに応用F/Wを量産してもらいたい
    - Mayaa2：Seasarプロジェクトで開発されている、WEBテンプレートエンジンMayaaの新バージョン。HTMLをAST化して取り扱う

紺屋の白袴

Practice Makes Perfect
the Ashikunep Kotan 伍

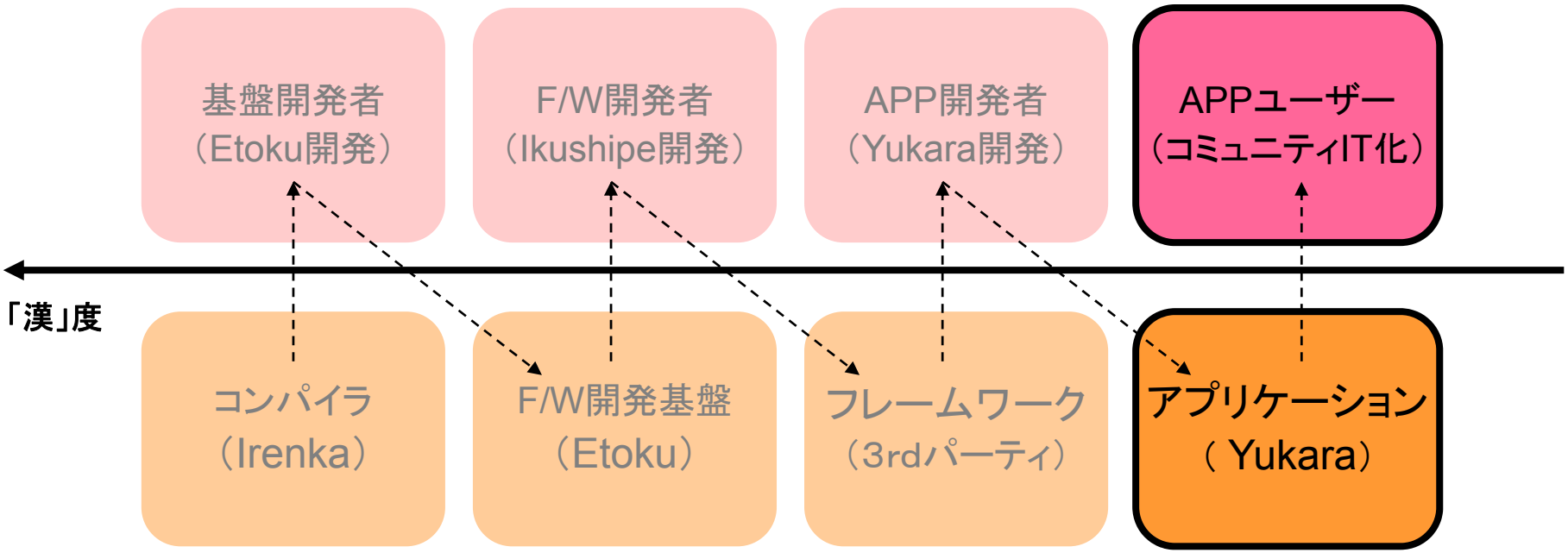
- OSS開発コミュニティなのに…
  - 企業システム構築に資するミドルウェア等を提供するにも関わらず、「自身」では使っていない
  - 運営の様々な事柄が手動にて、効率が低い
    - 会費納入事務
    - プロジェクトサーバ上での各種申請-設定
    - NPOのWEBサイトのコンテンツメンテナンス
    - その他、いっぱい
- 翻って見ると、SIerやISVでも似たようなもの
  - ソフトウェア開発の原価計算
  - ソフトウェア開発に関わる内部統制

「アプリケーション」カテゴリ

Practice Makes Perfect the Ashikunep Kotan 伍

- OSS開発コミュニティのIT化
  - Seasarファウンデーション傘下の各コミュニティをはじめとするOSS開発コミュニティの「紺屋の白袴」を解決する

The Ashikunep Kotanの重点目標 その1



プロトタイプコミュニティ建設

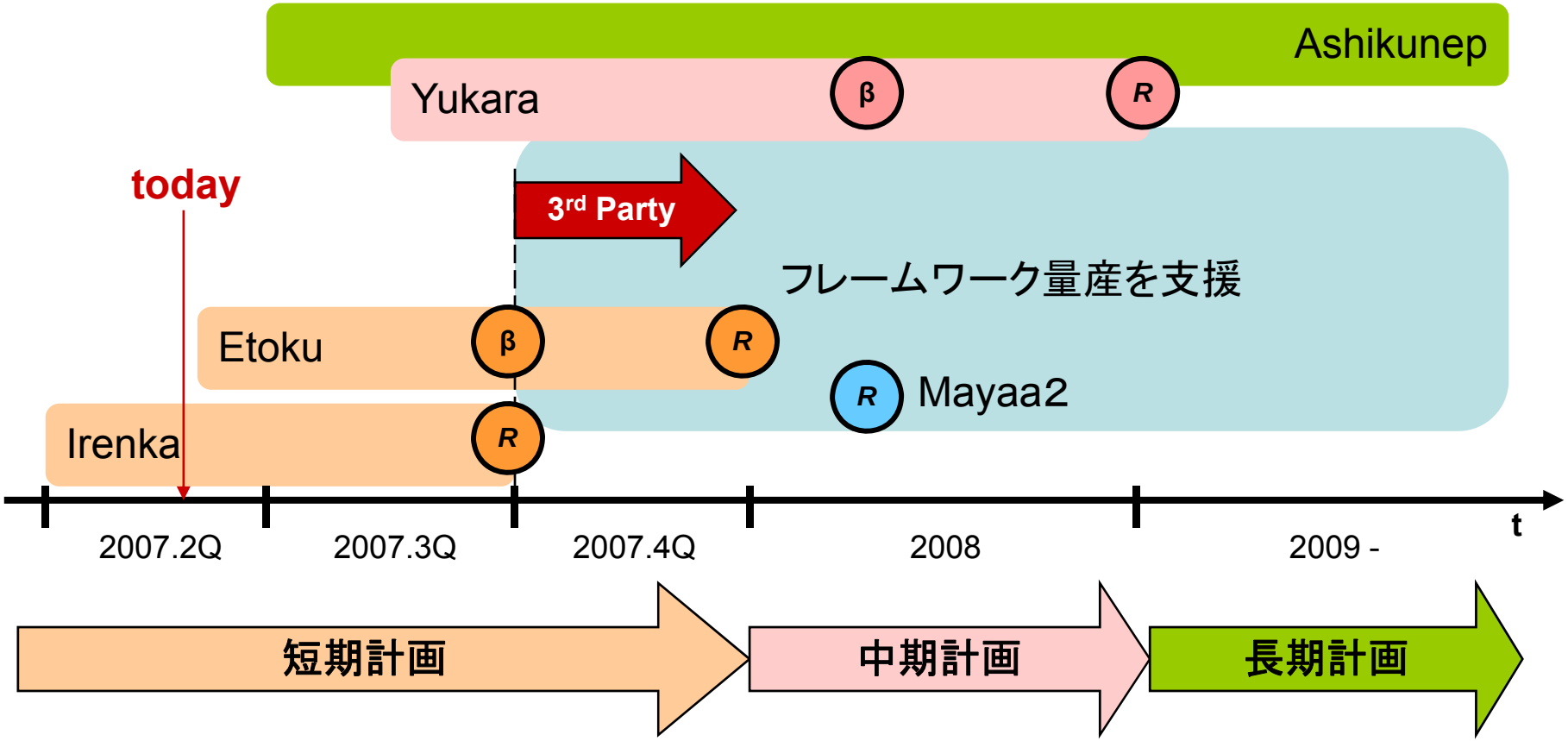
Practice Makes Perfect
the Ashikunep Kotan 伍

- OSS開発コミュニティのIT化
  - Yukara開発により、運営向上
    - コミュニティ運営に資するアプリケーション
    - 効率性向上、継続性向上、透明性の向上
- コミュニティプロトタイプの創造
  - Seasarファウンデーションはコミュニティの運営支援に徹し、傘下コミュニティが自主独立
    - コミュニティ内の規約等について事例が必要
  - 既存のプロダクトやコミッタのまだいないところで規約等の整備が行える
    - Ashikunepにて、検討、実験的実施
    - 有効なものについてはテンプレートとして知見提供

タイムライン

Practice Makes Perfect the Ashikunep Kotan 伍

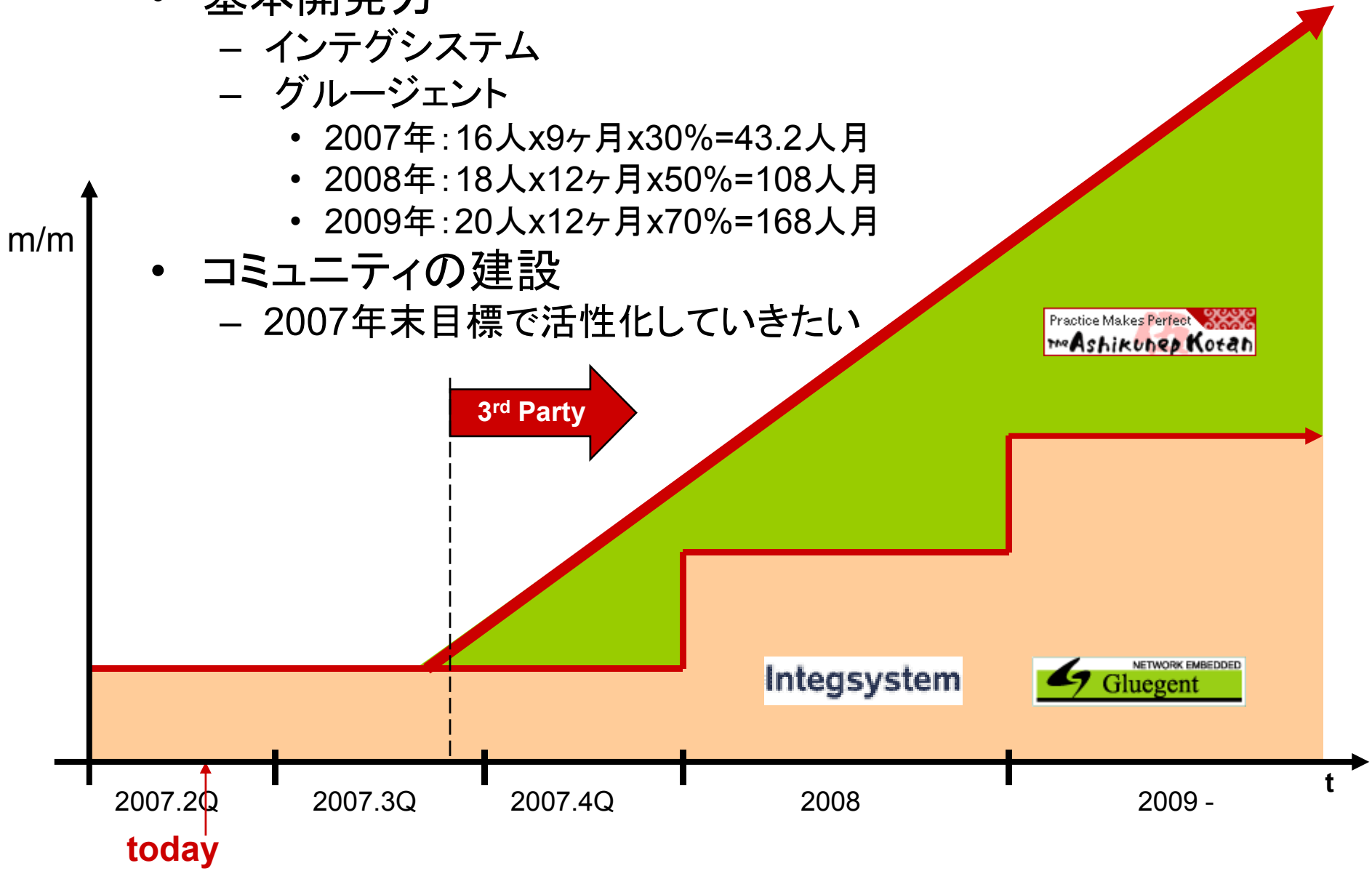
- 短期計画：コンパイル時ソリューション提供
- 中期計画：コミュニティIT化アプリケーション提供
- 長期計画：プロトタイプコミュニティの実現



開発リソース調達スキーム

Practice Makes Perfect
the Ashikunep Kotan 伍

- 基本開発力
  - インテグシステム
  - グルージェント
    - 2007年：16人x9ヶ月x30%=43.2人月
    - 2008年：18人x12ヶ月x50%=108人月
    - 2009年：20人x12ヶ月x70%=168人月
- コミュニティの建設
  - 2007年末目標で活性化していきたい



最後に

Practice Makes Perfect
the Ashikunep Kotan 伍

- Ashikunepは計画的に行動
  - OSSにも関わらず、中長期を含む計画を行う
  - OSSにも関わらず、昼間時間の開発者確保
- Ashikunepの楽観的な無計画行動
  - 企業リソースあるも、収益モデルは考えていない
  - 技術的難易度は高いが、短めに期間設定
  - 技術コンセプト有用度の検証は少ない
- しかし
  - ここで説明したことは、予定通りに実現する
  - 秋のSeasarCon（10月27日土曜日）に進捗確認

質疑応答

Practice Makes Perfect
the Ashikunep Kotan 伍

Q&A