システム開発における生産性の検証

Seasar

平成19年度　OISA「技術研究会」

JAVA 第2部会

2008.02.19

目次

1. 部員紹介

部員紹介

- 葛城啓之　（株式会社オーイーシー）
- 工藤寿彦　（九州東芝ｴﾝｼﾞﾆｱﾘﾝｸﾞ株式会社）
- 白石和稔　（大銀ｺﾝﾋﾟｭｰﾀｻｰﾋﾞｽ 株式会社）
- 川野芳樹　（株式会社オーイーシー ）
- 椎原健次　（大分大学大学院）
- 大谷謙二　（KCS大分情報専門学校）
- 阿部誠司　（株式会社シーエイシー）

２．生産性向上に向けて

生産性とは

生産性 ＝ 産出量 ／ 投入量

産出量

- 生産量
- 生産額
- 売上高
- 付加価値
- GDP

投入量

- 労働
- 資本
- 土地
- 原料
- 機械設備

システム開発における生産性

労働生産性

※労働者１人１時間あたりの生産性

技術的観点から生産性の向上を図る

生産性低下要因

- 依存性の高いプログラム設計
- 開発者レベル（Know How）の違い
- 企業の体制
- 情報検索時間の増加

解決策

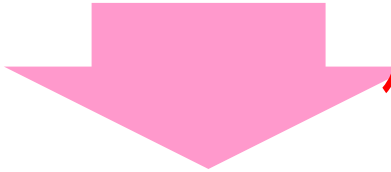
- インタフェースの活用
- 人材育成の強化
- 企業内の環境整備

生産性向上のための開発手法

- DIコンテナ
- AOP（アスペクト指向プログラミング）
- O/Rマッピング

DIコンテナ

インターフェースと実装を分離し、オブジェクトを外部から生成し設定する仕組み



依存関係を注入

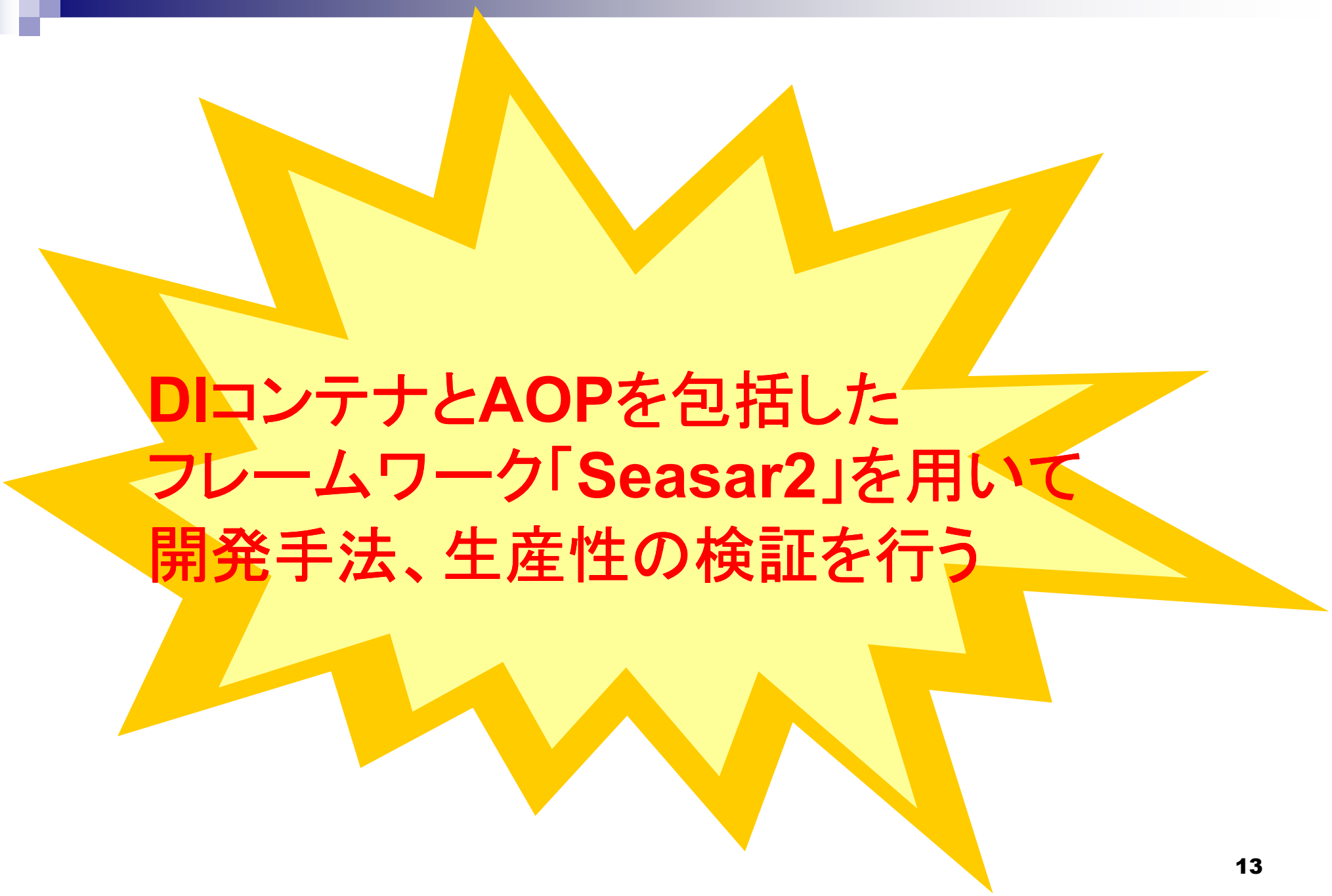
コンポーネント間の依存関係をソースコードから取り除く事が可能

AOP

プログラム本来の目的とは異なる処理を内部に埋め込まず、外から織り込むように作る

可読性のあるプログラム

把握・管理・変更が容易になる

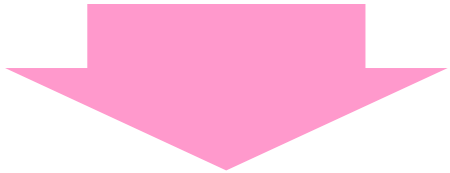

DIコンテナとAOPを包括した
フレームワーク「Seasar2」を用いて
開発手法、生産性の検証を行う

3. Seasar2

Seasar2

- DIとAOPをサポートした軽量コンテナ
- 設定ファイルの記述が少ない
- 日本語ドキュメントの充実

http://www.seasar.org/

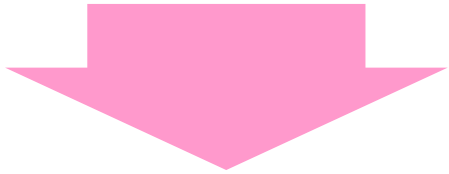


Eclipse等での導入が容易

Seasar2のメリット

DIとAOPのそれぞれの利点を生かせる

- コンポーネントがシンプル
- トランザクション制御の外部切り分け



- テスト容易性
- 拡張性
- 再利用性

向上が可能

定義ファイル

DIコンテナ作成には「diconファイル」を使用

diconファイルとは

コンポーネント生成情報をXML形式で記述

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
"http://www.seasar.org/dtd/components24.dtd">
<components>
        <component name="..." class="...">
                ...
        </component>
        <component name="..." class="...">
                ...
        </component>
</components>
```

省略不可

diconファイルの定義

コンポーネントの自動登録

```
<component class="org.seasar.framework.container.autoregister.FileSystemComponentAutoRegister">
        <property name="autoNaming">
                <component class="org.seasar.framework.container.autoregister.DefaultAutoNaming"/>
        </property>
        <initMethod name="addClassPattern">
                <arg>"addressBook.logic"</arg>
                <arg>".*LogicImpl"</arg>
        </initMethod>
</component>
```

パッケージ名

クラス名

クラス名には正規表現を使用可能

例）

AddressLogic.java（インタフェース）

AddressBookLogicImpl.java（実装）

の2つを指定パッケージに置けば自動登録される

慣例でこの名前にしている

ルート定義ファイル「app.dicon」

```
<?xml version="1.0" encoding="Shift_JIS"?>
<!DOCTYPE components PUBLIC "-//SEASAR2.1//DTD S2Container//EN"
"http://www.seasar.org/dtd/components21.dtd">
<components>
        <include path="s2struts.dicon"/>
        <include path="addressBook/dicon/addressBook.dicon"/>
        <include path="j2ee.dicon"/>
</components>
```

例）app.dicon

includeで指定したpathのdiconファイルを実行

「j2ee.dicon」

データベース設定を行うdicon

```
<component name="xaDataSource"  class="org.seasar.extension.db                    ">
        <property name="driverClassName">
                "com.mysql.jdbc.Driver"
        </property>
        <property name="URL">
                "jdbc:mysql://localhost/seasar"
        </property>
        <property name="user">"seasaruser"</property>
        <property name="password">"seasarpassword"</property>
</component>
```

mySqlのﾄﾞﾗｲﾊﾞ

"oracle.jdbc.driver.OracleDriver"

"jdbc:oracle:thin://localhost/seasar"

DB切替が容易

ﾕｰｻﾞ名・ﾊﾟｽﾜｰﾄﾞの指定

例）j2ee.dicon

- コネクションプール設定
- トランザクション設定

diconに記述可能

4. テストプログラムによる検証

概要

一般的なServletで作成したものと、Seasar2を使用し作成した物との比較を行う。

住所録

- 一覧表示機能
- 新規登録機能

開発環境

- 開発言語
  - Java バージョン 1.5.0
- アプリケーションサーバ
  - Tomcat バージョン 5.5.23
- データベース
  - MySQL バージョン 5.0.27

テストプログラム
デモ

相違点

- DB接続

Servlet

```
Class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection("jdbc:mysql://localhost/seasar?user=seasaruser&password=seasarpassword");
```

使用するたびに記述

2箇所

Seasar2

```
<component name="xaDataSource"
    class="org.seasar.extension.dbcp.impl.XADataSourceImpl">
    <property name="driverClassName">
        "com.mysql.jdbc.Driver"
    </property>
    <property name="URL">
        "jdbc:mysql://localhost/seasar"
    </property>
    <property name="user">"seasaruser"</property>
    <property name="password">"seasarpassword"</property>
</component>
```

設定ファイルに記述

1箇所

相違点

- インスタンス生成

Servlet

```
AddressBookLogic addressBookLogic = new AddressBookLogic();
```

使用するたびに記述

2箇所

Seasar2

```
<component
  class="org.seasar.framework.container.autoregister.FileSystemComponentAutoRegister">
  <property name="autoNaming">
    <component class="org.seasar.framework.container.autoregister.DefaultAutoNaming"/>
  </property>
  <initMethod name="addClassPattern">
    <arg>"addressBook.logic"</arg>
    <arg>".*LogicImpl"</arg>
  </initMethod>
</component>
```

設定ファイルに記述

1箇所

相違点

- 作成ファイル

Servlet

```
list.jsp
register.jsp
AddressBook.java
Servlet.java
web.xml
```

合計5ファイル

Seasar2

```
list.html
register.html
list.mayaa
register.mayaa
AddressBookDao.java
RegisterAddressBookDto.java
AddressBook.java
AddressBookLogic.java
AddressBookLogicImpl.java
ListAddressBookAction.java
ListAddressBookActionImpl.java
RegisterAddressBookAction.java
RegisterAddressBookActionImpl.java
RegisterAddressBookForm.java
addressBook.dao.dicon
addressBook.dicon
app.dicon
j2ee.dicon
struts-config.xml
validation.xml
validator-rules.xml
web.xml
```

JAVAファイル

10ファイル

設定ファイル

4ファイル

合計22ファイル

検証結果

メリット

- 誰が書いても同じ
- 設定ファイルなどを再利用できる
- 大規模プロジェクト向き

デメリット

- 命名規則に注意
- 設定ファイルの作成が大変
- 小規模プロジェクトには不向き

5. 考察

確認できなかったこと

- 直接的な生産性の向上

確認できたこと

- 設定ファイルの重要性
- 大規模開発向き

生産性・品質・技術者レベル

UP

生産性・品質・技術者レベルUP

- Java未経験者の導入が容易
- 排他制御などの実現が容易
- 画一的なコーディング方式
- 短期間での技術者育成

導入の際の留意点

- 規約はしっかり
- Seasar2のルールを押さえる

6. まとめ

自社の現状を把握する

- 各個人のスキル
- 保有するシステムのノウハウ

技術進歩は「秒進分歩」

自社だけでは対応できない

場合がある

複数の企業で協調する必要がある

- 社外研修への参加
- 他企業との交流

大分県の技術水準向上

大分県での受注増加

魅力的な職場

優秀な人材があつまる