Presentation Framework

that I want

一度は37Signals所属とか言ってみたい

なかやまひろゆき

Good Framework?

- なかなか納得いく *Presentation Framework* がない
- ではどんなものなら納得するの？

Point of View

- Architecture
- Configuration
- Function / Complexity
- User Interface

Architecture

- MVC
- Dialog/Document
- ほかに何かいいもの？

MVC

- 基本的にはPresentationはデータと表示と、それを結びつけるAction

- MVCで問題がない
- MVCモデルにロックインされているので他の設計は難しい

Architecture(TT)

- スクリプト自身がAction
- 処理する内容がデータ
- テンプレートを選ぶ
- 結びつけるのはモジュール

もっと簡単にArchitecture

- Request/ResponseのAPIの汚さ
- doPost/doGet/Service
- ServetContext,ServletConfig
- Wrapすべきものが多すぎる

Architecture

- APIのGood Wrapper
- POJO
- MVC
- 多少の不便さは対価として可能(設定とか)

Configuration

- Configurationは糊
- 構成に柔軟性を与える重要な役割

But...(Configuration)

- どのようなものにも対応できる
- Configurationできるものが増える
- なんでもConfiguration
- メンテナンス不能

Size(Configuration)

- Zeroは理想だが困難
- 柔軟は理想だが破綻

Size(Configuration)

- Zero
  - 可能であれば
- Less
  - 規約重要(from Rails)
- Large
  - WYSWIG（現実的であれば)

OCP(Configuration)

- 新しいActionを追加する
  - なぜかXMLも編集している
    - OCP(Open Close Principle)っぽくない
    - SRPにも反している
- やはりOCPであってほしい

Annotation(Configuration)

- OCP的に正しい
- 責務が明確
- 巨大なXMLを分割

Rule(Configuration)

- 暗黙の約束事

  余計なことはぐずぐず言わない

- Rule以外のことだけちゃんと言う

  - Conventions over Configurations

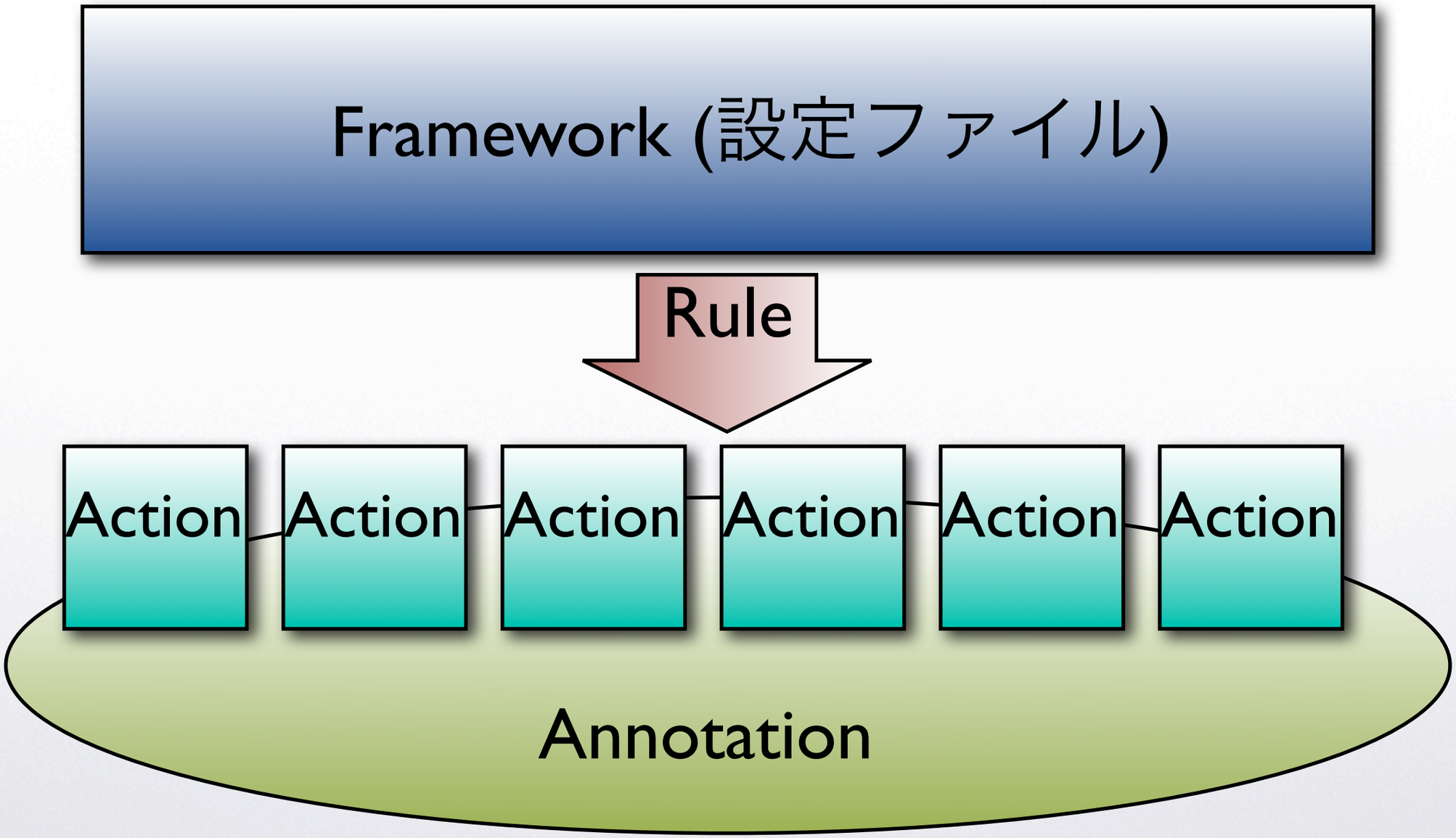
Rule(Configuration)

- そのルールは
  - 明快か？
  - 直感的か？
  - 覚えやすいか？
- 多すぎないか？（３つくらいまで）
- 効果的か？（多くをカバーしているか)

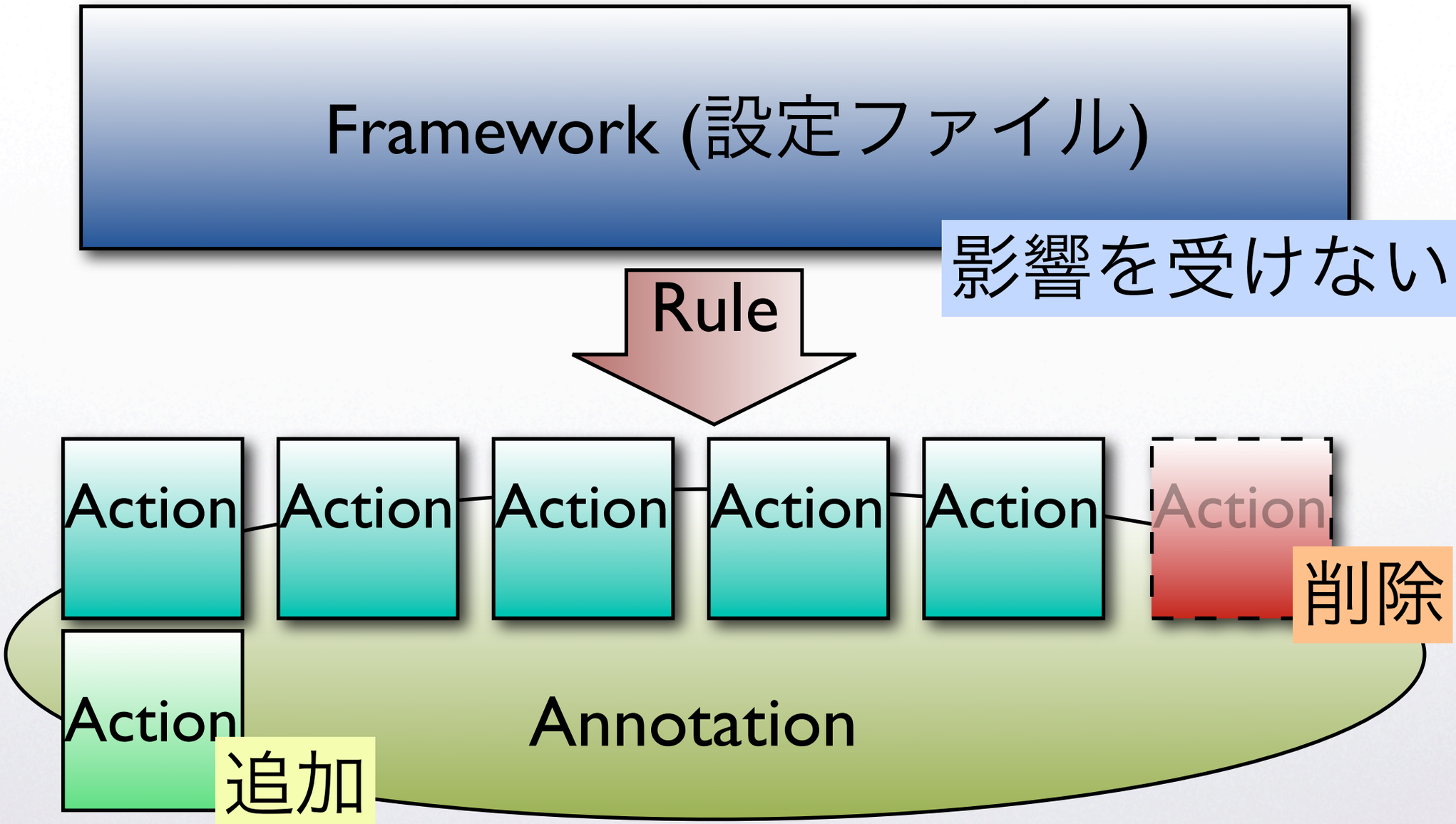
分割統治(Configuration)

- 設定リソースはメタリソース(設定ファイルの類)とAction(URIと結びつくソース)
- メタリソースはFramework全体を制御
- ActionはURIに結びつく情報の責任を追う(ex Dto,Template,,)
- ActionのDeploy/Undeployは感知しない

分割統治(Configuration)



分割統治(Configuration)



Function

- Frameworkに関心を持つときに調べるのは機能
- Framework比較は機能星取り表
- 「XXができないFrameworkイラネ」

Function/Complexity

- 「XXの機能を使うために、XXを設定してください」
- 等価交換の原則に陥りやすい
- 機能が増大する＝複雑さも増大しがち
- 多機能で使いこなせないFramework

Change the Rule

- 例の青い本
- その機能は顧客の利益に直結しているか？

Function/Complexity

- Frameworkにとっての顧客
  - Frameworkを利用する開発者

  - 開発者にとって「本当に」必要なものだけがあればよい
  - いらないものは欲しくない
  - 少しのことを上手に行う

Function/Complexity

- 自分のレイヤでやるべきことを行う
- 自分以外のレイヤでできることは、自分でやらない
- 他のプロダクトと協調できる

Function/Complexity

- Simpleとはいっても
  - 拡張性に閉じてはいられない
  - 柔軟性が０というのは使えない
  - pluginを増やすと設定が増える
  - Ruleを増やすと煩雑になる
  - どうやって設定を軽減するのか？

Mode(Function)

- 由来はデジカメの撮影モード
  - 撮影モードは小さな設定の集合
  - Frameworkでも細かい設定よりまとまったセット
  - Modeを使用すれば設定は０（ですむこともある
  - もちろん、マニュアルモードも用意

小さなまとめ

- 開発段階で増加するActionは
  Configurationから分離
- Configuration自体はFrameworkのみを制
  御
- Configurationそのものも他の手段で軽減

User Interface

- ProductにもUser Interfaceがある
- FrameworkにもUser Interfaceがある
  - Frameworkの顧客は利用者/開発者

User Interface

- User InterfaceとしてのFramework
  - 動かすまでの設定
  - 開発サイクルにおける煩雑さの軽減
  - 開発そのもののヘルパー

復習(User Interface)

- 動かすまでの設定
  - 究極的にはjarを置けば動くlevelに
- 開発サイクルの煩雑さの権限
  - ActionをOCPでデザイン可能に

Helper(User Interface)

- 2つのモードにきちんと対応
  - 開発モード
  - 運用モード
- 運用はともかくまずは開発モード

Helper(User Interface)

- Debug Screen
  - 500エラー画面からソースを捕捉
  - その場で編集
  - その場でDeploy
  - 同じパラメータで再実行可能

Error Support(User Interface)

- フレームワーク内でのエラー対応
  - OSSならソース嫁？
    - それはちょっとつらい
    - 時間もかかる
- エラー箇所に識別子を用意
- Debug Screenから識別子に従ったサポートページへのLinkを用意

Helper(User Interface)

- Visual Configuration
  - Configuration状況をHTML化
  - ActionのDeploy状況をHTML化
  - Enable/Disableもその場で操作
  - 履歴や状況をDumpし再現動作可能に

Testability(User Interface)

- Teeedaすごい。まじすごい。Respect
- 加えてSleipnir + WSHのWSH部分の生成
- Browserモジュールテストの充実

Trivial things?

- View Templateは絶対にPure HTML
- 単独でもPresentation機能が提供可能

最後に

- 皆さんのアイデアとかあればとても聞きたいのです
- この後、遠慮なく発表の内容はDisってください
- でもちょっとだけ容赦してください(笑)

感謝

- きっかけをくれたCraig Maclanahan
- 多くのアイデアをもらったRailsやS2Struts(DHH,キムキムほかの皆さん）
- 一部のアイデアはPlaggerから（宮川さん）
- よいProductは魔法のようであるという知恵をくれたDamian Coneway

Thanks

ご清聴
ありがとう
ございました