

卒業論文概要書

2010年 2月提出

学科名	コンピュータ・ネット ワーク工学科	氏名	村上 真一	指導 教員	鷺崎 弘宜 准教授
		学籍番号	CD 1G06R196-4		
研究 題目	アスペクト指向を用いたリッチインターネットアプリケーションの 変更要求モジュール化に関する研究				

1. はじめに

リッチインターネットアプリケーション(Rich Internet Application : RIA)は, Ajax などの非同期通信の仕組みを利用し, ページ遷移と非同期に画面が更新されるリッチなユーザーインターフェースを有する. しかし一方で, RIA の設計は従来の Web アプリケーションより複雑になっており, 変更が必要な箇所はプラットフォームを越えて横断的に拡散することがあるため, 変更要求への対応は必ずしも容易ではない.

本研究では, RIA への変更要求に素早く正確に対応するために, 変更要求への対応に必要な実装がアーキテクチャを横断する場合において, 個々の実装内容の生成方法を定型処理化してまとめ, あらかじめパターンとして整理しておく方法を提案する.

2. RIA の登場背景と問題点

典型的な RIA のアーキテクチャを図 1. に示す. RIA のアーキテクチャは, 従来の Web アプリケーションのアーキテクチャと異なり, MVC の責務がサーバー内に集中しておらず, クライアントとサーバーのプログラムが協調して一つの機能を実現することが多い. 特に, 非同期通信を行う RIA の機能の実現は, 複数のモジュールに横断して存在する.

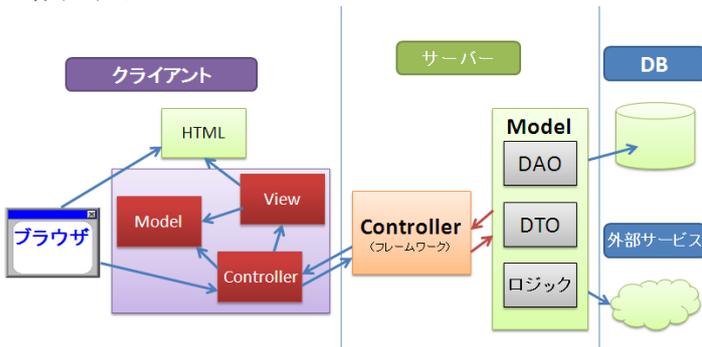


図 1. RIA のアーキテクチャ

RIA に変更要求が来た場合を考える. 変更要求には素早く正確に対応することが求められるが, 変更要求の対応に必要な変更箇所が, クライアントやサーバーといったプラットフォームを横断して存在する場合, 以下の 2 つの問題が発生することが考えられる.

P.1. 横断的に存在する変更箇所を, 開発者が一箇所ずつ巡回して変更を加えていく必要があると, 変更漏れが発生し, 欠陥となる恐れがある.

P.2. 変更箇所が横断的に必要となる変更要求において, 具体的に変更が必要となる箇所を, 開発者が全体の実装を確認しながら一箇所ずつ特定していく必要があると, 特定に抜けが生じ, 欠陥となる恐れがある.

本研究では, 上記 2 つの問題の解決を目指す.

3. 変更要求に対する解決のモジュール化とパターン化

P.1. の解決に貢献する技術として, アスペクト指向プログラミング (Aspect-Oriented Programming : AOP) [1] がある. ソフトウェアを開発する際, モデリング, 設計, 実装に登場する様々な観点を関心事と呼ぶ. 関心事には, 機能や非機能といった開発上の留意点が広く該当する. AOP は, 実装が設計に横断して必要となる関心事を一つのモジュールに局所化して実装することを可能にする技術である. AOP により, 本来モジュール群へと横断的に散らばる処理を局所化することができる.

しかし, AOP によるモジュール化はあくまで単一のプラットフォーム内に限定される. RIA のようなマルチプラットフォームによってはじめて実現されるアプリケーションにおいては, AOP のモジュール化は十分ではない.

実装レベルの技術である AOP に対して, 要求獲得, 分析, 設計を含むソフトウェア開発のあらゆる段階でアスペクトの視点から開発を行う手法を, アスペクト指向ソフトウェア開発 (Aspect-Oriented Software Development : AOSD) という.

本研究では, 変更要求に必要な実装を各プラットフォーム内でモジュール化した上で, AOSD レベルのアスペクトとしてまとめ (図 2.), 要求から実装までを一括して扱う手法を提案する.

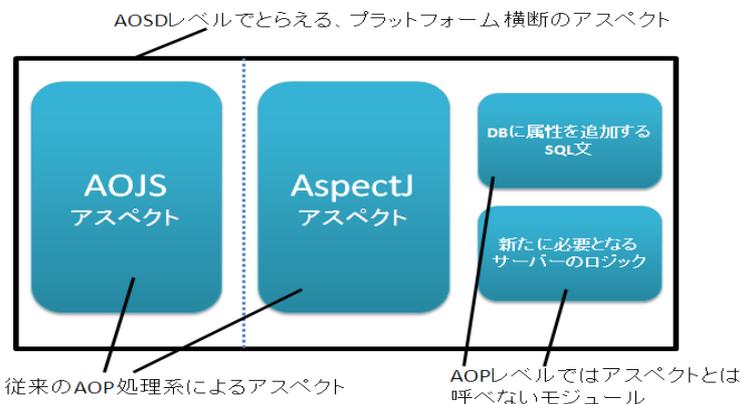


図 2. プラットフォーム横断の変更要求

AOP, AOSD はあくまでモジュール化の技術であり, 何をモジュール化すべきかという, 変更箇所の特定には貢献しない. P.2. を解決するために, 変更箇所特定の作業を定型処理化する方法が考えられる. 本研究では, 具体的な変更要求の種類を「問題」とし, その変更要求の実現のために必要な変更箇所を特定するためのルールを「解決」として, 問題と解決をパターンとして扱う方法を提案する. このパターンを適用して定型処理を行うことにより, 半完成のアスペクトを得ることができる. その流れを図 3. に示す.

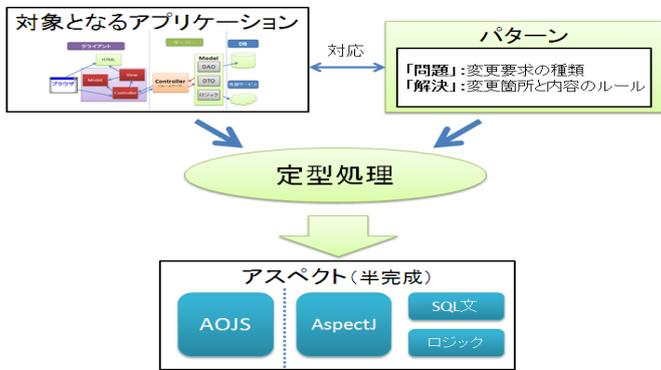


図 3. 本提案手法の流れ

すべての変更要求に対して、対応するパターンを用意することはできない。具体的には、処理のフローを追加する変更要求のように、処理のフローと密接に関係する変更要求は、パターン化が困難と考えられる。

4. データベースへの属性追加

本研究では、「RDBMS の CRUD 処理を行う RIA において、扱う属性を追加する」問題を対象に、必要な変更箇所と内容を、具体的なパターンの例としてまとめることに成功した。その内容を図 4. に示す。

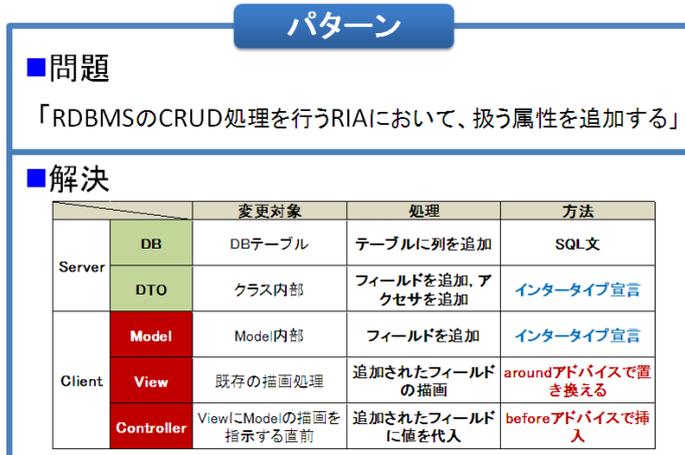


図 4. 本手法で提案するパターンの例

クライアントに JavaScript, サーバーに Java を用いる RIA を仮定する。クライアント内の Controller に対して、パターンに従い変更を適用するアスペクトを AOJS[2] を用いて記述した例を図 5. に示す。

```
<function functionname="/【対象とする Controller 関数】/updateModelView" pointcut="call">
  <before><![CDATA[
    【モデル名】.【追加のメンバ】=【適切な値, それを得るための処理】;
  ]]></before>
</function>
```

図 5. クライアント内 Controller の変更アスペクト

5. 評価

前章で示したパターンを、地図情報共有アプリケーションに適用する実験を行った。この地図情報共有アプリケーションは、Yahoo!地図サービス[3]を利用し、リッチなユーザーインターフェースを持った地図アプリケーションである。地図上の任意の地点を名前付きで DB に登録することができ、ユーザー間で地点の情報を共有することができる。この地図情報共有アプリケーションに、地点に名前の他にコメントとして情報を付加できるよう拡張を行う。(図 6.)

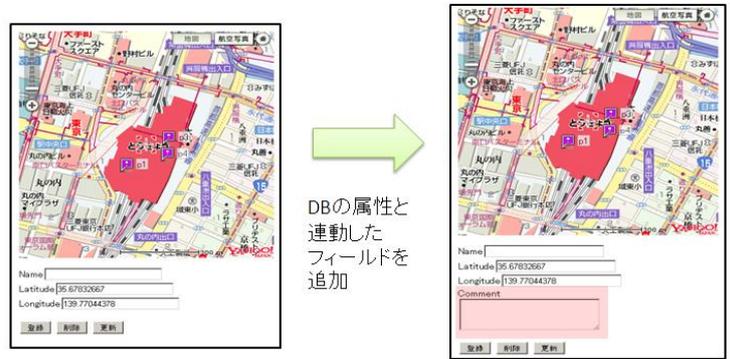


図 6. 地図情報共有アプリケーションの拡張実験

本変更実験に、本研究で提案したパターンを適用したところ、計 14 カ所に渡る変更箇所を変更要求のアスペクトの要素として特定することに成功した。その内容を図 7. に示す。また、必要な記述を補完することで拡張の機能が実現することを確認した。

本実験により、本研究によって提案したパターンの有効性が確認された。

		処理	抽出箇所数
Server	DB	テーブルに列を追加	1
	DTO	フィールドを追加、アクセスを追加	3
Client	Model	フィールドを追加	1
	View	追加されたフィールドの描画	1
	Controller	追加されたフィールドに値を代入	8

図 7. 地図情報共有アプリケーションへのパターン適用結果

6. 関連研究

Jacobson[4]らは、ユースケース単位で AOSD を行う手法を提案しているが、マルチプラットフォームに横断する場合について言及されていない。また、アスペクトの生成を支援する手法も提案していない。

7. おわりに

本研究では、マルチプラットフォームに横断する変更要求の変更箇所をパターンに従って特定し、アスペクトとしてまとめて扱うことに成功した。

今後の課題として、定型処理の自動化、すべての変更を同時に適用するように強制する MDA システムの導入、セキュリティなどの非機能要求のパターン化などがある。

参考文献

[1]Gregor Kiczales, John Lamping, Anurag Menhdhekar, Chris Maeda, Cristina Lopes, Jean-Marc Loingtier and John Irwin: Aspect-oriented programming, Proc. European Conference on Object-Oriented Programming (ECOOP), pp220--242, 1997.
 [2]Hironori Washizaki, et al "AOJS: Aspect-Oriented JavaScript Programming Framework", Proc. 6th Asian Workshop on Foundations of Software (AWFS 2009), 2009. (to appear)
 [3]Yahoo!JAPAN デベロッパー ネットワーク-地図. http://developer.yahoo.co.jp/webapi/map/
 [4]Ivar Jacobson, Pan-Wei Ng, "Aspect-Oriented Software Development With Use Cases", Addison-Wesley Professional, 2005