RIAでのデザイナと開発者の完全分業を実現するフレームワーク

A framework for realizing complete separation of developer's and designer's work in RIA

福田浩章† 山本喜一†
Hiroaki Fukuda† Yoshikazu Yamamoto†
†慶應義塾大学大学院理工学研究科
† Graduate School of Science and Technology, Keio Univ.

要旨

現在の情報化社会において，インターネットを利用したウェブアプリケーションは必要不可欠である．ウェブアプリケーションは一般にMVCモデルで構築され，ユーザが操作するView部分はデザイナが作成することが多い．しかし，一般にデザイナはロジックを記述できないため，開発者がViewにロジックを埋め込む必要がある．したがって，デザインに変更が生じる度に，開発者の手を煩わすことになる．

そこで本論文では,RIA開発ツールの1つであるFlexを用いて,デザイナと開発者の役割を完全に分離するフレームワークを提案する．

1. はじめに

インターネットの普及により，ウェブアプリケーションの使用はもはや一般的である．ウェブアプリケーションはHTTPとHTMLで構築されるため，デスクトップアプリケーションと比較するとインタフェースの表現力に乏しく，ネットワークのオーバーヘッドによる遅延などにより，操作性が良いとはいえない[1]．これらの問題点を解決するため，近年ではAjax (Asynchronous JavaScript and XML) を代表とした，いわゆるRIA (Rich Internet Application)が提案され，インタフェースのデザインやユーザの操作性を向上させる傾向にある[2]．そのため，ビジネスロジックを記述する開発者と，インタフェースを設計するデザイナの協業が必要不可欠である．ウェブアプリケーション開発では，一般にデザイナの作成するHTMLベースのインタフェースに，開発者がロジックを埋め込んでいく．したがって，システム開発中,または開発後にインタフェースの変更要求があると,デザイナの作業だけでなくロジックの記述，動作確認といった開発者の作業も必要になり，作業量が増加する．これはデザイナと開発者でインタフェース部分のHTMLを共有せざるを得ないという制約があるためである．

そこで本論文では，RIA開発ツールであるFlex[3]を利用し，デザイナと開発者の作業をソースレベルで完全に分離するフレームワーク，Camelを提案する．Flexでは，通常各コンポーネントで発生するイベントを処理するため，各イベントとそれを受け取るメソッドをインタフェース部分に直接記述する必要があるが，CamelではFlexのイベント処理を利用し，実行時にイベントリスナのロジックとイベントソースの依存性を確立する．そのため，デザイナが作成するインタフェース部分には一切ロジックを記述する必要がなく，デザイナと開発者の役割をソースレベルで完全に分離することが可能になる．

2. 開発プロセス

図1にウェブアプリケーション開発時におけるデザイナと開発者の関係を示す．図1(a)はHTMLベース，図1(b)はFlexをベースにしたウェブアプリケーション開発の概要である．HTMLベースの場合，デザイナはクライアントの要求を元に,Photoshopなどのツールを使用してウェブサイト全体をデザインする．そして，完成したデザインから画像を切り出し，HTMLとCSSを使用して静的なHTMLでウェブサイトを完成させる．次に，開発者はデザイナが作成した静的HTMLを元に，JavaやPHP，Javascriptなどを使用してビジネスロジックを実装していく．この時，ボタンをクリックするなど，ユーザの行動を契機にしたサーバへの送信データ確認，データ送信，条件による表示データの切り替えなどを行うため,開発者はデザイナが作成したHTMLにロジックを埋め込む必要がある．現在ではStruts[4]やJSTL[5]などのタグライブラリ，Smarty[6]を初めとするテンプレートエンジンの出現により，HTMLに直接プログラムを記述しない方式が考案され，実際に利用されている．しかし，デザインを重視する場合，デザイナが作成するHTMLは非常に複雑であり，その中に条件分岐や繰返し処理を埋め込むという作業は開

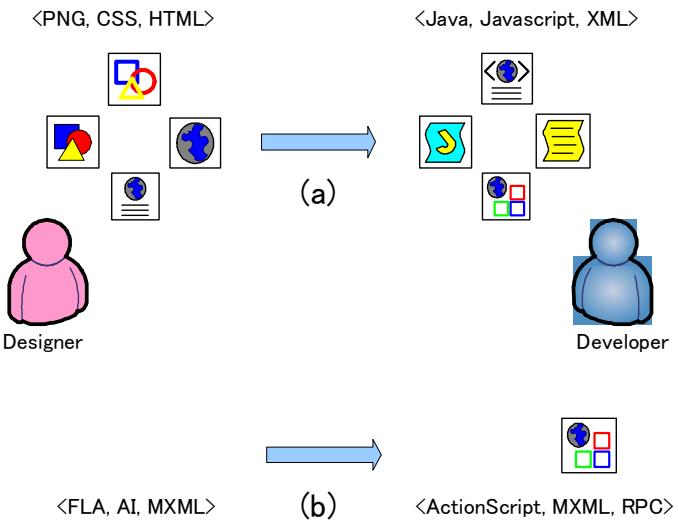

図 1 開発プロセス

発者にとって非常に困難な作業になる．また，ページのデザインのまま変更されないことはまれであるため，変更されるたびにデザイナだけでなく開発者の手を煩わすことになる．これは，本来文書の論理構造を記述するために作成された HTML がインターネットとともに普及し，ブラウザだけで利用できるウェブアプリケーションに発展したことが一つの原因である．

一方，Flex ではボタンやラベルなどのコンポーネントがあらかじめ用意されており，XML ベースの言語である MXML を利用してインタフェースを作成することができる．さらに，独自のコンポーネントも作成でき，それらをイラストレータや Flash で作成し，利用することも可能である．したがって，デザイナは普段使い慣れたツールで表現力の豊かなコンポーネントを作成し，インタフェースを作成することが可能である．また，Flex では ActionScript3.0 を利用してロジックを記述することができるため，開発者はデザイナが作成した MXML をベースにロジックを埋め込んでいく．図 2 に MXML の記述と生成されるインタフェースの例を示す．

図 2(b)に示すように，Flex ではコンポーネントをタグで表現し，インタフェースを記述する．そして，各コンポーネントで発生するイベントはタグの属性名として表現され，属性の値にイベントハンドラとなるメソッドが記述される．このように，Flex は HTML を使用する場合と比較してインタフェース部分におけるロジックの記述を削減することができるが，イベントハンドラの記述，およびハンドラを所有するオブジェクト生成の記述は依然として必要であり，デザイナと開発者の役割をソースレベルで完全に分離することはできていない．

(a)

EventName　EventHandler

```
<mx:DataGrid x="59" y="29">
    <mx:columns>
        <mx:DataGridColumn headerText="ID" dataField="id"/>
        <mx:DataGridColumn headerText="Name" dataField="name"/>
    </mx:columns>
</mx:DataGrid>
<mx:Button x="133" y="179" label="更新" id="updateButton" width="58" click="update();"/>
<mx:Button x="199" y="179" label="削除" id="removeButton" width="58" click="remove();"/>
<mx:Button x="67" y="179" label="新規" id="insertButton" width="58" click="create();"/>
```

(b)

図 2　Flex アプリケーションの例

3. アプローチ

Camel では，Conversion over Configuration の思想を受け，Struts や Spring[7]などで必要な設定ファイルを導入することなく，ソースレベルでデザイナと開発者の完全分離を実現する．これを実現するため，Camel では，(1)オブジェクトツリー，(2)イベント処理，(3)リフレクションを利用する．次にそれぞれについて述べた後，実行時に依存性を確立する方法について述べる．

3.1. オブジェクトツリー

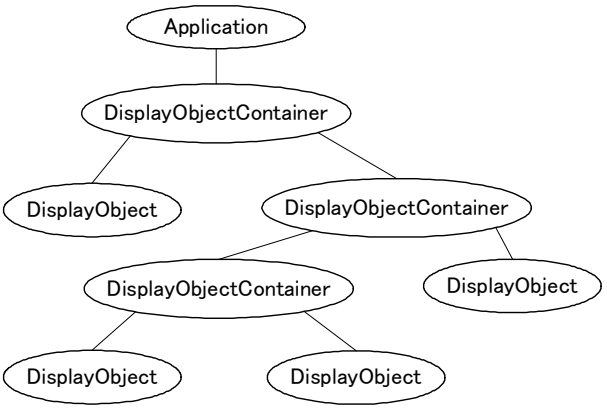

図 3 に示すように, Flex では Application クラスをルートにしたディスプレイリストと呼ばれるツリーに DisplayObject クラス, またはそれを継承したクラスを付加することによってインタフェースを表示している [8][9]．そして，Flex の各コンポーネントは DisplayObject クラスを継承した UIComponent を基底クラスにしている．この“付加する”とは，MXML のタグとしてコンポーネントを配置することであり，DisplayObject を継承していない非表示のクラスでも MXML のタグとして記述し，このツリーに付加することができる．したがって，このツリーを操作することによって，付加されたオブジェクトの参照を実行時にすべて取得することができる．

3.2. イベント処理

Flex はユーザの操作，あるいは状態の変化によってイベントが発生し，対応するイベントリスナがイベントを処理する．2 節でも述べたように，通常コンポーネントで発生するイベントと対応するイベントリスナは MXML タグの属性として記述されるが, EventDispatcher クラスに定義された addEventListener メソッドを利用することによって, イベントとイベントリスナを関連付けることができる. DisplayObject クラスは EventDispatcher クラスを継承しているため，それを継承した UIComponent クラスでも，addEventListener メソッドを利用してイベントとイベントリスナを関連付けることができる.

3.3. リフレクション

ActionScript には Java や Ruby と同様にリフレクション機能が存在し，XML で表現されたオブジェクト情報を EX4(ECMAScript for XML)[10]形式で操作することができる．また，型の制約も緩く，文字列と引数情報だけで容易にオブジェクトのメソッドを呼び出すことが可能である．そのため，オブジェクトの参照が取得可能であれば，プログラムに直接記述しなくても実行時に容易に任意のメソッドを呼び出すことができる.

3.4. 依存性の確立

Camel を利用する場合，デザイナと開発者は次の 1 から 3 に示す規則に従ってアプリケーション開発を行う必要がある．そして，Camel を使用して実行されたアプリケーションには，次の 4 から 6 の手順で実行時にイベントリスナの確立が行われる.

1. デザイナは MXML を用いてインタフェースを設計する．このとき，id 属性を使用し，重複のないよう各コンポーネントに名前をつける.
2. 開発者はデザイナが作成したインタフェース 1 つに対して，イベント処理のクラス (リスナークラス)を 1 つ作成する. リスナークラスにはインタフェースの各コンポーネントで発生するイベントを処理するメソッドを，次の命名規則で記述する.

   「コンポーネントの ID ＋ イベント名　＋　“Handler”」

   例えば, “calcButton”という id がつけられたコンポーネントのクリックイベントを処理するメソッ

ド名は“calcButtonClickHandler”となる．

3. デザイナが作成したインタフェースと，開発者が作成したリスナークラスを，同一の MXML ファイルにタグとして記述する．このとき，インタフェースには“View”，リスナークラスには“Listener”を接尾語にし，同一の名前をつける．
4. アプリケーションが開始されると，Camel では，各コンポーネントの生成完了イベントを契機に，”View”を接尾語としたコンポーネントの参照を取得し，保持する．
5. Camel は，View と同じ階層に付加されたリスナークラスの参照を取得する．View とリスナークラスの対応は，それぞれにつけられた名前と，接尾語をもとに判定する．
6. Camel は，View に付加されたコンポーネントの名前と，リスナークラスに定義されたメソッド名を元に，手順 2 で示した命名規則に合致した場合，addEventListener メソッドを利用してイベントとイベントリスナを関連付ける．

このように，Camel を利用することによって，インタフェースを記述する MXML と，イベントリスナを記述するクラスをソースレベルで完全に分離することが可能になる．そのため，インタフェースに変更がある場合でも，コンポーネントの名前に変更がなければ動作が保証される．

4. まとめと今後の方針

本論文では，RIA 開発ツールである Flex を利用し，ウェブアプリケーションのインタフェースを作成するデザイナと，ビジネスロジックを実装する開発者の役割を，ソースレベルで完全に分離するフレームワーク，Camel の提案と，具体的な手法について述べた．Camel を利用することにより，従来デザイナと開発者で共有せざるを得なかったインタフェースのソースを完全に分離することができるため，デザイン，またはロジックの変更が生じても，独立して作業をすることができ，開発効率の向上が期待できる．また，Camel ではコンポーネントの名前やメソッドの命名規則に制約はあるものの，それさえ守れば動作は保証されるため，メンテナンスやテストも容易に行うことができる．

本論文で述べたように，現在イベントとイベントリスナの依存性を実行時に確立は可能だが，複数のコンポーネントが存在する複雑なインタフェースの場合でもリスナークラスは 1 つであり，分割することはできない．そのため，コンポーネントの個数が増加すると，リスナークラスが肥大化してしまう．また，ビジネスロジックとインタフェースが密結合であり，複数のコンポーネントでビジネスロジックを共有することは難しい．そこで，今後 Flex のイベント処理を活用し，インタフェース部分の処理とビジネスロジック，ビジネスロジックとサーバサイドのロジックを疎結合にしたうえで，実行時に依存性を確立する仕組みを導入する予定である．

参考文献

[1] M. Driver, R. Valdes, and G. Phifer, “Rich Internet Applications Are the Next Evolution of the Web”, Technical report, Gartner, May 2005.
[2] L.D. Paulson, “Building rich web applications with Ajax”, Computer, IEEE, Vol. 38, No. 10, pp. 14-17. Oct. 2005.
[3] Adobe Systems Inc. “Adobe Flex2”, http://www.adobe.com/products/flex/whitepapers/.
[4] The Apache Software Foundation, “Apache Struts”, http://struts.apache.org/.
[5] Sun Microsystems Inc. “JavaServer Pages Standard Tag Library”, http://java.sun.com/products/jsp/jstl/reference/docs/index.html.
[6] Smarty, http://smarty.php.net.
[7] Spring Framework, http://www.springframework.org/documentation/.
[8] 布留川英一, “ActionScript3.0 ゲームプログラミング”, 株式会社毎日コミュニケーションズ, 2006.
[9] クジラ飛行机, “Adobe Flex2 プロフェッショナルガイド”，株式会社毎日コミュニケーションズ, 2007.
[10] Ecma International, “ECMA Script for XML Specification”, Dec. 2005.