Flashプラットフォーム用バイナリウィーバーの提案と実装

畔上剛† 福田浩章†
Takeshi Azegami† Hiroaki Fukuda†
† 芝浦工業大学　工学部　情報工学科
† Department of Information Engineering, Shibaura Institute of Technology

要旨

現在，多くの Web サイトで Flash の技術が用いられている．Flash を使用した開発では，ActionScript と呼ばれるプログラミング言語を使用する．しかし，ActionScript ではイベント駆動プログラミングによってソースコード上にイベントの処理が散在してしまうため，イベントのシーケンスが複雑化する．そのため，開発者はソースコード上のイベントの流れを把握しにくくなる．アスペクト指向はソースコード上に散在する処理を集約することができるそこで，アスペクト指向を用いてイベントのシーケンスをアスペクトに分離して以上の問題を解決する．分離されたアスペクトは最終的に元のプログラムと合成する必要があり，Flash は最終的にバイトコードになる．

そこで，本研究ではイベントのシーケンスをアスペクトに分離することを想定し，アスペクトと Flash のバイトコードを合成するバイトコード変換機構の提案と実装をする．

1. はじめに

Flash は，音声や動画，ベクターグラフィックスのアニメーションを組み合わせてインタラクティブな Web アプリケーションを開発する技術である. Flash で開発されたアプリケーション(以後 Flash アプリと呼ぶ)は FlashPlayer と呼ばれる実行環境上で実行される．現在，FlashPlayer は全ての PC に 99%インストールされている．また，Flash は Web だけではなく，デスクトップ，スマートフォンのアプリケーション開発も行うことができる. Flash アプリの開発では，主に無償のオープンソースフレームワーク Flex[1]を用いて開発する．Flex は Flash をベースにしたアプリケーション開発フレームワークであり，ActionScript3.0，MXML(Macromedia Flex Markup Lanuguage)と呼ばれる二種類の言語を使用する．ActionScript は Flash の開発に使用されるプログラミング言語で，アプリケーションのロジックを記述するために使用する．また，MXML は XML を基に開発された言語で，ボタンやテキストフィールドなどの GUI 部分をレイアウトするために記述するために使用する．Flash アプリの開発では，ボタンをクリックしたときなど，ユーザが何かの操作をしたときに実行したい処理を ActionScript で記述していく，イベント駆動型プログラミングが用いられる．そのため，複数のソースコード上にイベントの登録，送信などの処理が散在し，イベントのシーケンスが複雑化する．イベントのシーケンスの複雑化は，一つのソースコードにイベントのシーケンスが集約していないため，開発者はイベントの流れを把握することが困難になる．

一方，複数のソースコードに散在した処理を集約させる手法としてアスペクト指向プログラミング[2]が存在する．アスペクト指向では複数のソースコードに散らばる処理をアスペクトとしてモジュール化できる．アスペクトは最終的に元のプログラムに合成する必要がある．

本研究では，イベントのシーケンスをアスペクトと呼ばれる新たなモジュールに分離することを想定する．Flash は最終的に SWF[3]と呼ばれるバイトコードになるため，アスペクトと SWF の合成を行うバイトコード変換機構を提供する. 本稿ではアスペクトと SWF をバイトコードで合成を行うバイトコード変換機構について述べる．

2. 背景と問題点

本節では，ActionScript によるイベント駆動型プログラミングと，アスペクト指向プログラミングについて説明した後，Flash アプリ開発でのイベントのシーケンスの複雑化による問題点を指摘する．

2.1. ActionScript によるイベント駆動プログラミング

ActionScript は，イベント駆動型のプログラミング言語である．ボタンがクリックされた・マウスが動かされたといったユーザからの入力をイベントと呼ぶ．イベント駆動型プログラミングとは，イベントハンドラと呼ばれるイベント発生時に実行する処理を記述していくプログラミング手法である．ActionScript では，イベントハンドラをメソッドとして記述することができ，イベント発生時にイベントハンドラを呼び出すためには addEventListner メソッドを使用してイベントの登録を行う必要がある．また，ActionScript では dispatchEvent メソッドを使用しプログラムで明示的にイベントを発生させることが可能である．

2.2. アスペクト指向プログラミング

現在のソフトウェアの開発手法は，オブジェクト指向プログラミングが主流である．オブジェクト指向とは関連するデータとそのデータに対しての処理手続きの両方をクラスという単位にまとめ，モジュール化する技法である．しかし，ログ出力やアクセス制御といった処理は，オブジェクト指向を用いても複数のクラスに散在してしまう．このような複数のクラスに散在する特定の処理のことを横断的関心事と呼ぶ．横断的関心事は複数のソースコードに散在するため，一つのソースコードに集約して管理することはできない．

この問題に対して，アスペクト指向は横断的関心事をアスペクトと呼ばれるモジュールに分離することによって，横断的関心事を一つのソースコードに集約して管理することができ，プログラムの保守性が向上する．最終的に分離されたアスペクトはウィーバ(weaver)と呼ばれる処理系を用いて，コンパイルや実行時に元のモジュールに織り込まれる(weave).

2.3. イベントのシーケンスの複雑化

Flash はイベント駆動型であるため，ActionScript によるイベント駆動プログラミングが行われる．図 1 のように，イベント発生時にコンポーネント B から D，E のコンポーネントにデータを送信するには，コンポーネント間でイベントのやり取りを行う必要があり，ソースコード上にイベントの登録，送信などの処理が各コンポーネントに散らばる．イベント駆動型プログラミングではプログラムの流れが上から下に進むのではなく，次にどのイベントが起こるかによって処理の流れが決定する．したがって，イベントの登録，送信などの処理がソースコード上で散らばることにより，複数のコンポーネント間のイベントの送受信の順序が複雑になりソースコード上のイベントの流れがわかりにくくなる．そのため，イベントのシーケンスを制御する必要がある．

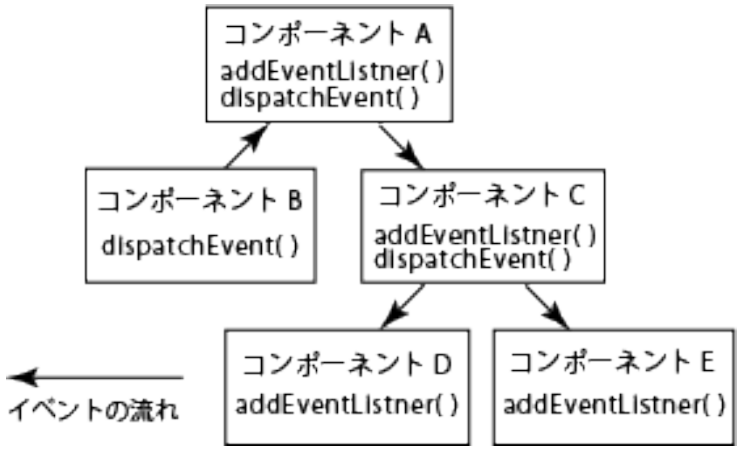

図 1　イベントのシーケンス

3. アプローチ

本研究では図 1 のようなイベントの流れを，図 2 のようにアスペクトに分離できていることを想定する．アスペクトとして分離したモジュールは，最終的に元のプログラムに weave しなければならない．Flash アプリは最終的に SWF と呼ばれるバイトコードになるため, 本研究では既存の SWF に対してアスペクトの weave を行うバイトコード変換機構の提案と実装を行い API として提供する．

既存の SWF にアスペクトを weave するためにはアスペクトに存在するクラス，メソッド，変数などを既存の SWF に追加する必要がある．したがって，提案する API では既存の SWF に対してバイトコードの書き換えによってクラス，メソッドの呼び出し，変数の追加を可能にする．図 3 のように SWF ファイルの構造は, ファイルのサイズやバージョンなどの情報を持ったヘッダから始まる. ヘッダの後には, タグの付いたデータブロックの列が続き，SWF の一部に doABC と呼ばれるタグが存在する．doABC タグには ABC[4]と呼ばれるバイトコードが埋め込まれており，ABC には図 4 のようにプログラムの命令コードおよびクラス，メソッド，変数などの情報が定義されている．

そのため，本研究では SWF に含まれる ABC に対してバイトコード変換を行う．

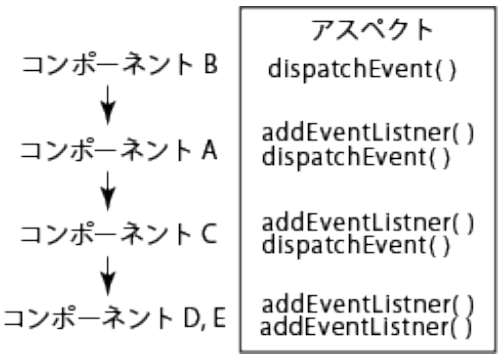

図 2　アスペクトによるイベントの記述

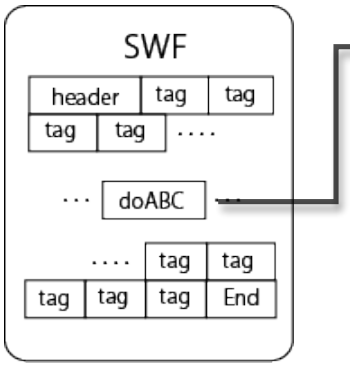

図 3　SWF の構造

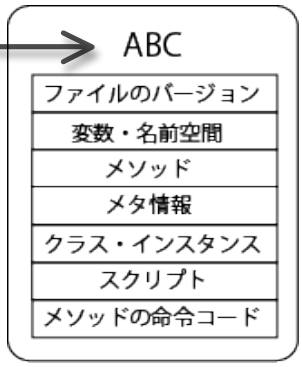

図 4　ABC の構造

3.1. 提案する API

提案する API として以下のようなクラスやメソッドの記述を想定する．

```
SWF original = new SWF(“Original.swf”);   //weave される SWF を読み込む
ABC abc = new ABC(original);              // weave される SWF の中の ABC を読み込む
ABC aspect = new ABC(“Aspect.abc”);       //weave する ABC を読み込む
abc.weave(asppect, file);                 //weave する
                                          //file：weave 場所が記述されたファイル
```

開発者は, SWF クラスを用いて機能が追加される SWF を読み込む. また, ABC クラスによって SWF に含まれる ABC の読み込みや weave する ABC を読み込む．weave する ABC は ASC コンパイラと呼ばれる ActionScript コンパイラを生成し，weave メソッドにより weave する．

3.2. バイトコード変換機構

バイトコード変換の手法として，まず図 5 のように SWF から doABC タグを発見し，ABC の部分を抽出する．次に抽出した ABC に対して解析を行うことによって，ABC の構造に基づいた構文木を生成する．その後，解析した構文木に対して変換処理を行うことによってクラス，変数，メソッドを追加する．そして，変換処理を行った構文木から ABC を再生成し，SWF に埋め込むことによってバイトコード変

換を実現する．

ABC の抽出，解析，生成，埋め込みはオープンソースである ax-compiler[5]の一部を使用する．本研究では構文木からの変換処理の部分を Java 言語で実装する．

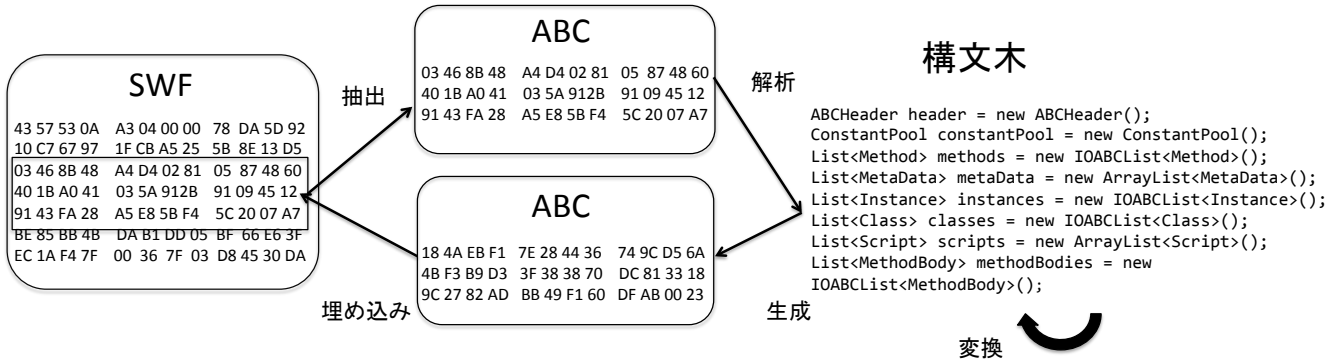

図 5　バイトコード変換の手法

4. 関連研究

バイトコード変換を用いた研究として以下のような研究が挙げられる．

4.1. Josh

Josh[6]は Java 言語によるアスペクトを Java ファイルに weave する処理系であり，クラスロード時にバイトコード変換によってクラスファイルをアスペクトに weave する．ソースコードで行うアスペクトの weave は weave した後にソースコードをコンパイルするため，プログラム変更の度にプログラム全体を再コンパイルしなければならない．

一方，バイトコード変換による weave ではアスペクトを変更した場合でも，アスペクトだけを再コンパイルすればよい．また，バイトコードによって weave するのでソースコードがないクラスファイルや jar ファイルに weave を行うことが可能である．そのため，本研究で提案するバイトコード変換機構の利点としてプログラム全体をコンパイルしなくてよいので開発効率の向上やソースコードがない場合にも機能を追加できる点が挙げられる．

5. まとめ

本稿では，既存の SWF ファイルに対してバイトコードの書き換えを行うバイトコード変換機構を述べた．現在バイトコード変換機構は実装中である．評価方法としては，ソースコードから ActionScript コンパイラを用いてプログラムを変更した場合と SWF からバイトコード変換機構を用いてプログラムの変更した場合のそれぞれの変換時間，ファイルサイズを比較し考察する．

参考文献

[1] Adobe Systems Incorporated, “Adobe Flex”, http://www.adobe.com/jp/products/flex.html, 2012
[2] G.Kiczales et al., “Aspect-oriented programming”. *In Proceedings of European Conference on Object-Oriented Programming*, 1997, pp. 220-242
[3] Adobe Systems Incorporated, “*SWF File Format Specification Version10*”, http://www.adobe.com/jp/devnet/swf.html, 2012.
[4] Adobe Systems Incorporated, “*Actionscript virtual machine 2 overview*”, http://www.adobe.com/jp/devnet/swf.html, 2012.
[5] Google Project Hosting, “ax-compiler”, http://code.google.com/p/axc/, 2012
[6] 中川清志，立掘道昭，千葉滋，”Josh:バイトコードレベルでの Java 用 Aspect Weaver，日本ソフトウェア学会 SPA’02，2002.