Apollo

渡辺 章人，木浦 正博
Akihito Watanabe，Masahiro Kiura

1　はじめに

近年，web アプリケーションの動向として，Rich Internet Applicaton(RIA) が注目を浴びている．従来からの web アプリケーションの多くは，HTML，CSS，JavaScript によって構成されたドキュメントベースのアプリケーションである．これには，操作時に伴うページ遷移や通信待ち時間等の操作性や応答性の面でデスクトップアプリケーションに大きく劣る点がある．RIA は，こうのような問題を解決し，web アプリケーションの操作性を向上させるものである．現在，RIA により操作性を向上させたアプリケーションは，そのユーザービリティの高さにより，多くのユーザーに利用されている．そして現在，RIA の技術を利用した，新たな web アプリケーションの形として，Adobe 社によって“Apollo”が公開されている．

本報告では，Apollo が利用する RIA の技術に触れると共に，Apollo とはどのようなものであるかを解説し，今後について予測する．

2　RIA(Rich Internet Applicaton)

2.1　RIA とは

RIA とは，Ajax，Flash，Flex を利用した web アプリケーションのことである．データの処理をサーバ側ではなくクライアントで行うことにより従来の web アプリケーションではなしえなかった以下の事項を実現している．

- サーバ負荷の低減
- 通信待ち時間の短縮
- 動的なページ生成

これらを実現する技術について，2.2 節で述べる．

2.2　RIA の技術

- Ajax(Asynchronous JavaScript + XML)

  Ajax とは，JavaScript に実装された HTTP 通信を行うためのクラス XMLHttpRequest を利用し，HTTP プロトコルを用いた非同期通信による XML のやり取りを行う技術である．Fig. 1 にその概念図を示す．

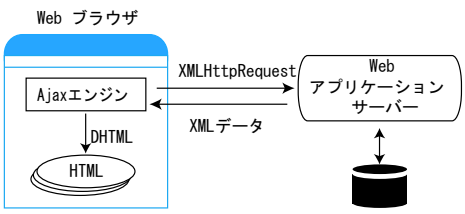

Fig.1　Ajax 概念図 (出展：[1] を参照)

XMLHttpRequest とは，JavaScript に実装された HTTP プロトコルによるサーバとの非同期通信を可能にするクラスである．Fig. 1 に示すように，XMLHttpRequest でサーバとデータを送受信を行う．受信した XML を JavaScript，CSS，DOM の連携である DHTML によって，web ページの部分的更新を可能にする．これにより通信待ち時間の短縮，ページ遷移を伴わない動的な web アプリケーションを実現している．

- Flash，Flex

  Flash は，スクリプト言語である ActionScript を使用し，web コンテンツ (swf ファイル) を作成する規格，及び開発環境の名称である．swf ファイルの実行ソフトとして Flash Player があり，インターネットに接続されている，様々なデジタルデバイスで動作するソフトウェアプラットフォームとなっている．また，近年，Flash に関連した技術として注目を浴びているのものに，Flex がある．

  Flex とは，ActionScript のフレームワークである．Flex を用いた開発では，XML で書かれた mxml に UI を，mxml に実装されるコンポーネントに対するアクションを ActionScript で記述する．このようなインターフェースとそれに対するアクションを別々に記述する方式は，多くの web アプリケーションのフレームワークに採用されている．Flash で作成されるアプリケーションはデザインを重視した web アプリケーションであった．しかし，Flex を用いた RIA 開発では，プログラムが更に重要視された．Flex を利用することにより，グラフィカルな UI を持つ, サーバサイドと連携したアプリケーションの作成を行い易くなった．

  Flash，Flex を用いたアプリケーションのサーバサイドとの連携を Fig. 2 に示す．

  Fig. 2 に示すように，Flash，Flex を用いたアプリケーションは HTTP/SOAP, 及び独自のプロトコルである AMF を利用してアプリケーションサーバとデータの送受信を行う．また，データの送受信を非同期に行い，取得したデータにより表示を動的に切り替えることが可能である．また，Ajax では異なるドメイン間での通信は不可能だが，Flash，Flex を用いたアプリケーションでは，ドメインをまたいで外部ファイルを読み込むことができる．

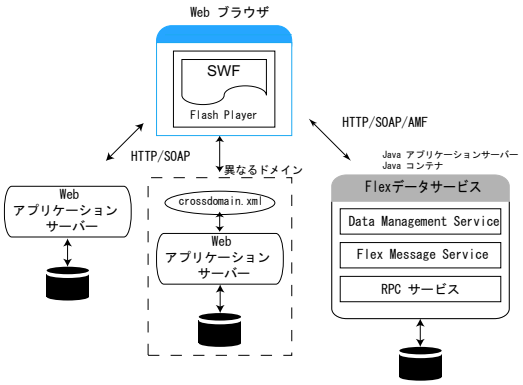

Fig.2　Flash，Flex アプリケーションのサーバサイドとの連携 (出展：[2] を参照)

2.3　RIA による変化と残された課題

RIA により，web アプリケーションの応答性，操作性は向上し，よりローカルなアプリケーションに近づいた．しかし，web アプリケーションとして以下の課題が残されている．

- オフライン時に利用不可能
  オフラインとは，ネットワークに接続していない状態を示す．オフライン時，クライアントはサーバサイドに保存されている情報の取得を行うことができない．
- Web ブラウザによる機能の制限
  Web ブラウザにはキーボード操作など制約が多い．そのため，RIA が高度なユーザーインターフェース (UI) の実装を行うことができない場合がある．また，Web ブラウザの UI とは異なる UI の実装が必要となる．

これらの課題を解決するソフトウェアとして“Apollo”が登場した．

3　Apollo

3.1　Apollo とは

Apollo とは，Adobe が公開したクロスプラットホームランタイムのコードネームである．Apollo は，デスクトップ上での web アプリケーションの開発，及び利用を可能にする．また，それらにはデスクトップアプリケーションが持つユーザビリティの実装を可能とする．Adobe は，このようなプラットフォームを提供することにより，オフライン時の対応を想定した web アプリケーションの促進を図ろうとしている．

3.2　Apollo アプリケーションの構成

インストール前の Apollo アプリケーションは，“.air”形式の AIR ファイルとして配布される．Apollo ランタイムがインストールされた環境において，Apollo アプリケーションのインストーラとして起動する.AIR ファイルの実体は，複数のファイルを ZIP でまとめたものである．開発環境である ApolloSDK のコマンドラインツールを使用して AIR ファイルを生成する．AIR ファイルの構成内容を以下に示す．

- ADF(Apollo Descriptor File)
  名前やバージョンやコピーライトといったアプリケーション情報を XML 形式で記述されている属性ファイルである．
- ルートコンテンツ (swf ファイル，html ファイル)
  Apollo アプリケーションのメインとなるコンテンツである．
- サブコンテンツ (swf ファイル，html ファイル，js ファイル)
  ルートとなるコンテンツ内で利用されるコンテンツである．
- アイコン (png 画像)
  アプリケーションのアイコンである．

Apollo ランタイムをインストールしたプラットフォーム上で，AIR ファイルの実行を行うと ADF に基づいてインストーラが起動し，プラットフォームに適した形式の実行ファイルに変換される．そのため，アプリケーション配布元はプラットフォームごとにアプリケーションを作成する必要がなく，異なるプラットフォームに対しても同じ AIR ファイルを作成することができる．

3.3　Apollo アプリケーションの実行環境

Apollo アプリケーションの実行環境を Fig. 3 に示し，それらの構成要素について以下で説明を行う．

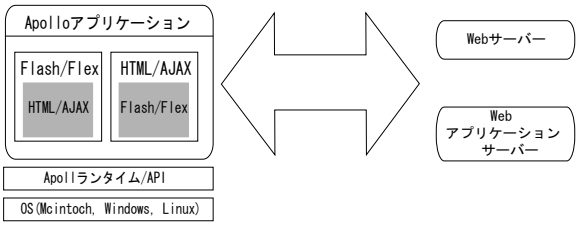

Fig.3　Apollo アプリケーションの実行環境 (出展：[3] を参照)

- Apollo ランタイム
  Fig. 3 に示した「Apollo ランタイム」が Apollo アプリケーションのクライアント側での実行環境である．Apollo ランタイムは HTML のレンダリングエンジンとして WebKit，また Flash のランタイムとして Flash Player が組み込まれている．WebKit は，アップルコンピュータによって開発，オープンソース化した HTML レンダリングエンジンである．その構成要素として，HTML のレンダリングを行う WebCore.framework と JavaScript の処理を担う JavaScriptCore.framework を含む．Apollo ランタイムに組み込んだ際に，ランタイムのファイルサイズが最小限であったこと，また WebKit 自体がモバイル機器にも搭載しやすいものであったため採用された．これらにより，web 技術のローカル環境での実行を可能にしている．Apollo ランタイムの構成を Fig. 4 に示す．

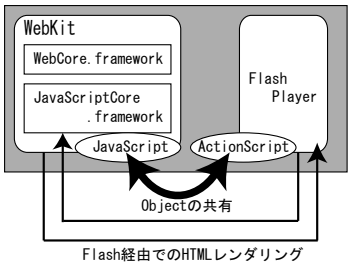

Fig.4　Apollo ランタイムの構成 (出展 : 自作)

Fig. 4 に示すように，WebKit と Flash Player はローレベルで連携している．この連携は以下を可能にしている．

- Flash 内 HTML
  Flash コンテンツ内で利用されている HTML を Flash Player 経由でレンダリングを行うことが可能となる．これにより，HTML コンテンツに Flash 内でビットマップに適用できる視覚効果を適用することができる．
- JavaScript, ActionScript 間での Object の共用
  ActionScript から JavaScript の，JavaScript から ActionScript コンテンツの Object 操作が可能となる．これにより，HTML と Flash を組み合わせたアプリケーションを容易に開発することができる．

- サーバサイドとの連携
  Fig. 3 に示すように，サーバサイドとの連携パターンは 2 つある．これらは，Ajax，Flash/Flex のような従来の RIA 技術を利用する．また，サーバサイドで動くアプリケーションは既存の web アプリケーションと同様の仕様となる．

3.4　デスクトップアプリケーションを実現する API

デスクトップアプリケーションの構築を実現するために，Apollo では次の機能の API が JavaScript，ActionScript に用意されている．

- ローカルファイルへの入出力
- ウィンドウの操作
- アプリケーションの更新
- ネットワーク検出機能

これらの機能は 2 章で述べた RIA 技術をシームレスにデスクトップで実行するための技術でもある．また，ネットワーク検出機能は,Apollo アプリケーションに対して，ネットワークの接続状態を伝える機能である．これは，オフライン対応型 web アプリケーションを構築するための重要な機能である．今後，ドラッグ&ドロップ，コピー&ペースト，Apollo アプリケーション同士の連携，PDF サポート等の API が用意される予定である．

3.5　利点

- オンライン/オフライン対応アプリケーションの作成
  3.4 節で示した「ローカルファイルへの入出力機能」と「ネットワーク検出機能」をうまく利用することで，オンライン時はリアルタイムに情報の更新，保存を行う．オフライン時はオンライン時に取得した情報の参照や編集が可能となる．
  例えば，オフライン時，Gmail でメールを作成，保存し，ネットワークに接続した時点でメールや添付データを送信することも可能となる．

- 既存技術，リソースの再利用
  Apollo アプリケーションは，web 技術を利用して作成することが可能である．オフライン時の実行への対応としては，既存の web アプリーションへの修正が必要となる．しかし，サーバサイド側について機能を変更する必要はない．また，RIA の技術を利用することにより，これまでのデスクトップアプリケーションを大きく拡張したアプリーションの作成が可能である．

- ブラウザからの独立
  web ブラウザでは，web アプリケーション間での連携を行えないことや，web ブラウザが持つ UI との重複が発生した．一方，Apollo アプリケーションは，ブラウザからの独立，API の利用によりこれらを解消する．また，ブラウザからの独立によりアプリケーションとしての明確性は保つことが可能となる．ユーザーに対して心理的側面から Apollo アプリケーションの利用を促す利点と考えられる．
  写真の一括アップロードなどデスクトップとの連携が必要なアプリケーションの操作性を高められる．

4　今後の展開

“Apollo”の登場により，ユーザーにネットワークとのつながりを感じさせないシームレスな web アプリケーションの作成，利用が可能となった．しかし，これを目指す団体は Adobe だけではない．Mozilla は Firefox の次期バージョンである Firefox3.0 に，“永続ストレージ”，“オフライン”機能への対応を行うことを明らかにしている．また，既存の Ruby on Rails アプリケーションをデスクトップ上で動作させる“Slingshot”というランタイムも近々リリースされる予定である．詳細は明らかにされていないが，“Apollo”と同じく，オフラインで動く web アプリケーションを目指している．今後，これらを用いた web アプリケーションは多数発表されることが予想される．

参考文献

1) [ThinkIT] 第 2 回 : Ajax アプリケーションとクラシック Web アプリケーションの違い
http://www.thinkit.co.jp/free/tech/33/2/
2) Adobe - Flex 2 LiveDocs
http://livedocs.adobe.com/flex/2_jp/
3) Adobe Labs - Apollo
http://labs.adobe.com/wiki/index.php/Apollo
4) Apollo ポケットガイド邦訳 Wiki
http://labs.anthill.jp/wiki/apollo/