ENGINEERING AND R&D SERVICES

HCL

WHITE PAPER on Flex to HTML5 – The Migration Challenge

April 2014

e2 Engineering experiences

Acceleration | Value | Technology

TABLE OF CONTENTS

Abstract

Adobe Flex and HTML-JavaScript based development had been two strong options, when considering the front end development of a web based application. However, with the advent of HTML5 (the latest specification on HTML family by W3C), many companies and enterprises are considering to migrate their apps from Flex to HTML5. Though there are multiple pros and cons, the key reasons behind considering a migration are Adobe itself moving away support of Flex. HTML5 is coming up with rich features to enable it as a application development option thus enabling stronger adoption of HTML5 across multiple devices and platforms and many other such benefits.

Both Adobe Flex and HTML5 front end are web based and can be accessed through a browser (Google Chrome, IE, FireFox). Though both Flex and HTML5 are Web based technology used for writing the front end and relevant business logic at client side, they are quite different when you consider the linguistic nuances of each of them. Flex is quite an established and matured language with OOPs support; while HTML5 is a markup language with its associated technologies - JavaScript and CSS. Interestingly when a company thinks of migrating their app from Flex to HTML5, there is no direct or automated way!! The migration usually entails re-writing the application all over with HTML5 which is quite a cumbersome process. However there are right guidelines and methodologies and tools using which one can ensure a right way of migration and minimize both time and risk. This paper talks about the different methodologies and guidelines of migration and also some common issues / challenges that one faces during migration and some ways to counter these migration hassles.

Flex based application require the Flash player plug-in for the browser to render and display the application

HTML5 applications use HTML5 syntaxes in conjunction with JavaScript frameworks that render HTML5 content.

Flex to HTML5 Migration Overview

Flex Applications

Flex with its robust features had been a choice for development of presentation tier for many applications but with the advent of HTML5 and its evolving features, its reach across platforms and devices has forced many organizations to rethink and reconsider.

Flex is an open source application framework for building and maintaining expressive web applications that deploy consistently on all major browsers, desktops and devices. It provides a modern standards-based language and programming model that supports common design patterns suitable for developers from different backgrounds. Flex technology is used primarily to design and develop the presentation tier of complex web applications.

Flex based applications require the Flash player plug-in for the browser to render and display the application. The languages used for developing Flex applications are MXML and Actionscript.

HTML5 based Applications

HTML5 is slowly taking up as the technology of choice for any presentation tier.

HTML5 is the fifth edition of HTML, a markup language used for structuring and presenting content on the web and a core technology for the rapid growth of the internet. It aims to bring about improvements in language with support for the latest multimedia and graphical content, keeping it readable by humans and consistently understood by computers and

i. Understand the Approaches of Migration

ii. Choose a Client side MVC Framework to map with Flex MVC

iii. Transforming the communication Layer with the backend interface

devices. It includes detailed processing models to encourage more interoperable implementations. It extends, improves and introduces markup and application programming interfaces (APIs) for complex web applications.

Along with new syntactical refinements, HTML5 includes options to directly add video, audio, canvas elements as well as integration of scalable vector graphics content. These features have been designed to allow inclusion of multimedia and graphical content in the webpage without the requirement of proprietary plugins and APIs.

HTML5 applications use HTML5 syntaxes in conjunction with JavaScript frameworks that render HTML5 content.

Migration Solutions

The keys steps involved while going for a Flex to HTML5 migration are

1. **Define the Migration Approach**
2. **Choose a Client side MVC Framework to map with Flex MVC**
3. **Migrate Communication Layer used with backend interface**

Defining the Approach of Migration

Let's start with the approaches for migration of existing Flex based app to HTML5 based app. The approach to be adopted depends upon the time, budget and complexity of the code. So the approaches are:

a) Start from scratch and build a completely new app with HTML5 :

The most straightforward solution is to start from scratch and build a completely new HTML application. If the

current application is relatively simple or if there is unlimited time and budget, this approach offers the chance to create a fully optimized user experience (UX) in HTML. It will also be the easiest to maintain. In the end, one will have a full HTML application, so if time and budget will allow it, consider this approach.

Attempting a complete replacement does, however, carry certain risks, particularly around scope management, time to market and customer.

Consider this approach, when

i. Current application is relatively simple
ii. Restrictive time and budget are limited
iii. Its affordable to simply scrap existing app
iv. Rebuild, or manage two parallel apps

b) Gradual migration of existing application functionality to HTML :

If one cannot afford to simply scrap the existing app and rebuild, or manage two parallel apps, then consider a different strategy for the near-term. In most cases, one will want a strategy that focuses on gradual migration of the existing application functionality to HTML. In this case,an "outside-in" approach is recommended – especially for apps hosted on a Web browser. This means slowly changing parts of the application, starting with the most outermost, to HTML and continuing to use pieces of the Flex application within that new HTML until the Flex elements are all replaced.

The work typically begins by converting any application chrome and primary navigation elements to HTML. Then the core views or content areas can be converted one-by-

one as time and budget allow, until the final target of an HTML-only application is reached.

Consider this approach, when

i. application is hosted on the web browser
ii. application chrome and navigation is to be converted first
iii. conversion of core views depends on time and budget
iv. Gradual conversion to full HTML application is acceptable

c) Start with the HTML conversion of inner flex component workflows & slowly integrate application functionality :

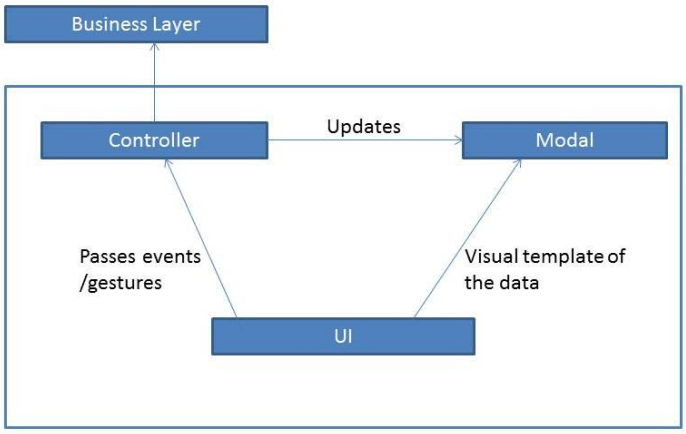
In other cases, it may make sense in the short term to focus on doing new development with HTML and continue using what was already done in Flex, instead of trying to port existing Flex work to HTML. While this is not the best approach for the majority of applications, it will be the right strategy for a few applications. In this case, we recommend an "inside-out" approach, in which HTML is initially used for new components or workflows accessed from within the Flex application then slowly expanded to cover more of the application functionality as desired. This approach introduces significant challenges for maintaining a seamless experience in some contexts. It can also be difficult to integrate it into a complete HTML application, so it only makes sense in cases where the existing Flex application will continue to live alongside new development on other platforms.

Consider this approach, when

i. HTML is initially used for new components or workflows accessed from within the Flex application, then slowly expanded to cover more of the application functionality as desired.
ii. existing Flex application will continue to live alongside new development on other platforms

Choose a Client side MVC Framework to map with Flex MVC

The Flex application can be seen as the view layer of the MVC. But the Flex application can itself be thought of as an MVC inside the web application MVC.



The View layer of the Flex MVC mainly consists of the user interface which, most of the time, is described as a hierarchy of MXML components. The application's root component represents the top of the UIComponents hierarchy. It can be

either an Application object (in the case of web Flex applications), or a WindowedApplication object (for AIR). In a Modular application, it can also be a Module object designed to be loaded in a container application.

Inside this hierarchy, communication is achieved through an implementation of the observer pattern: flash's Event mechanism. This allows communication while reducing dependencies, which is exactly the need of the hour. Moreover, Flex provides an additional mechanism to easily update shared data between the views, which also relies on Events : DataBinding.

This way, our views are as independent as possible from the rest of the application.

For example:

a) Consider a BookForm which lets a user update a Book object with a Flex form [Flex View component].

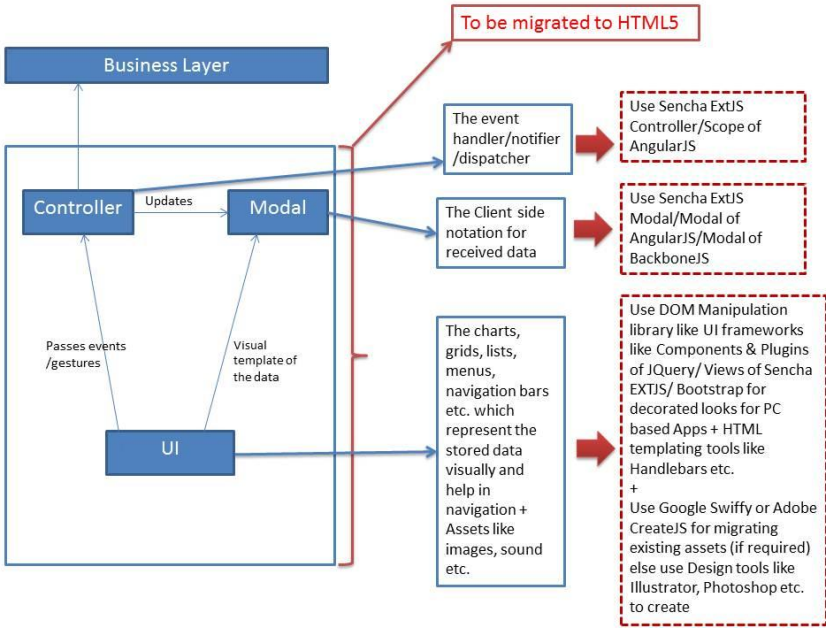
b) Our BookForm component will only have a reference to the currentlySelectedBook, a Book-typed property in our Flex Model (a ValueObject).

c) The user will click on a "Save" button, which will dispatch an UpdateBookEvent to Flex Controller.

d) The Flex Controller will listen to this event, do what it has to do, and update the Flex Model's currentlySelectedBook property accordingly.

e) Since this property is linked by DataBinding to our BookForm, it will immediately show the updated Book on the Flex View components present in the MXML.

HTML5 MVC framework for Point-to-Point mapping of Flex MVC pattern



Some solutions (like Sencha Touch / ExtJS) aim to provide an all-in-one framework. Sencha EXTJS provides the MVC framework for mapping the FLEX MVC architecture components. Let us map using the same Bookform example above:

a) Consider a BookForm, which lets a user update a Book object with a form [Sencha EXTJS View component]. This view can be decorated with CSS using BootStrap and can add dynamicity using jQuery.

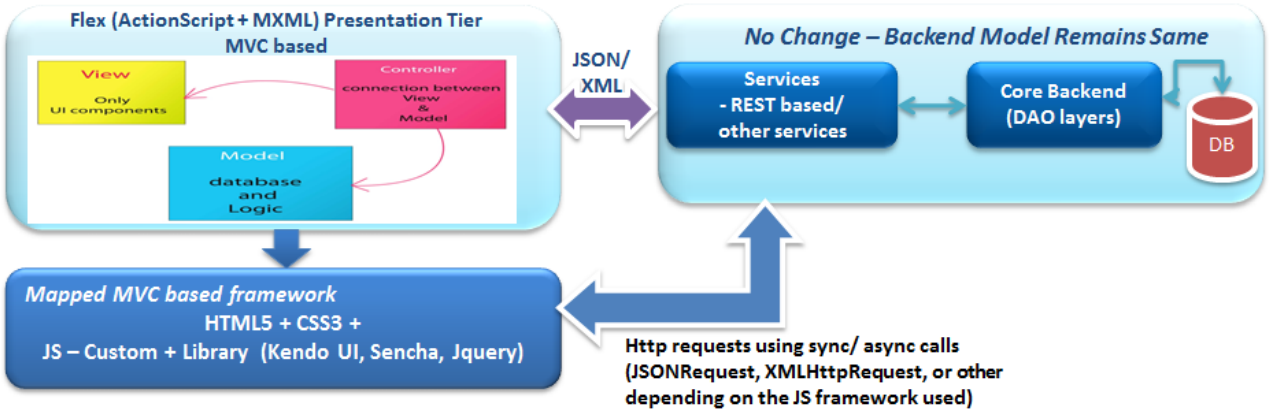
b) Our BookForm component will only have a reference to the currentlySelectedBook, a Book-typed property in our Sencha EXTJS Model called Storage (a ValueObject).

c) The user will click on a "Save" button, which will dispatch an UpdateBookEvent to Sencha EXTJS Controller.

d) The Sencha EXTJS Controller will listen to this event, do what it has to do, and update the storage's currentlySelectedBook property accordingly.

e) Since this property is linked by DataBinding to our BookForm, it will immediately show the updated Book on the Sencha View Component i.e. form.

A high level architechtural mapping shown below –



Migrating communication Layer with the backend interface:

The Http call mechanism for fetching backend data as used by Flex can be leveraged by using ajax callback mechanisms of used HTML5 framework.

Best Practises in Migration

SECURITY

The Client cannot be trusted in any web based application. It does not matter whether the client is plain old HTML, some kind of AJAX based DHTML, a complete ExtJS driven application, a flash movie or a native desktop application. Anyone using the client will have all the javascript on his/her local machine.

A solid authentication/authorization framework on the server can only be used to defend intrusion within code.

However, some guidelines can be followed to make the client less vulnerable to attacks.

ASSET MIGRATION

Animation assets of the Flex application can be migrated using automated tools like Google Swiffy/Adobe CreateJS.

SWFs can be converted to HTML5 [shown in a video representing the usage of a product]

Assets [like moving images/graphics] can be recreated to be used with HTML5 using timeline view of Flash in CreateJS.

1. There is no automation process for the migration of the Flex application to HTML5.
2. Meeting browser compatibility across different browsers will be point of concern
3. Defining the right approach for migration

Key Challenges in Migration

Few major challenges in the process of presentation tier migration from Flex to HTML5 –

1. There is no automation process for the migration of the Flex application to HTML5. Therefore the entire application in Flex has to be re-written in HTML5/JS using either a manual approach or by picking up a right set of tools/framework. Selection of the tool poses challenges in terms of:

    a. Cost

    The license cost of the tool/framework paired with other library or tool's cost for meeting significant requirement may go high. E.g. Sencha does not provide Pivotgrid which is a critical requirement for many of financial clients. In that case, mzPivotgrid has to be usedwhich has an additional cost associated to it. Thus using Sencha with mzPivotgrid has a cost addon challenge associated with it.

    b. Flexibility of adding UI components

    The view library of the chosen tool/framework may not suffice for certain requirement such as charts or grids. In that case, third party charts like AnyChart, Fusion Chart etc. have to be integrated with the chosen framework. The main challenge lies in the compatibility of the chosen framework to incorporate another plugin/UI component.

    c. Learning curve

    The learning curve of the tool chosen again should be assessed carefully and should comply with the skills of the stakeholders. E.g. Sencha with its high learning curve and

vast API poses significant challenge for short term projects with core HTML skill workers.

2. Defining the right approach for migration of a sophisticated system with complex logic applied for rendering the presentation tier remains challenging.

3. Also, the Flex application does not have much dependency on browsers but it requires the Flash Player plugin which is available at most of the systems and this makes reaching out to a wider audience possible using different browsers. But HTML5 is a comparatively new technology and requires the browser for rendering thus making browser compatibility across different browsers a point of concern.

4. Flex framework has a huge set of inbuilt features that can be reused and programmed to get rich user experience. Flex along with other AS3 libraries has been effectively used to get rich UI for the application. Thus, finding suitable HTML5/JS frameworks and libraries to get similar user experience poses significant challenge.

Conclusion

Migrating Flex based apps to HTML5 is a tough ask and hence defining the right approach, methodologies and tools holds a lot of significance at present. This paper has thrown light on these specific aspects to assist developers/development teams when they go for migration of a large Flex based frontend app. It is important to understand that it's not just translating code from Flex to HTML5/JS, but it requires adherence to maintain existing application design, pattern and architectural sanctity. One of the key criteria had been re-usability of existing framewoork and code, so as to minimize changes in migration. On the tools aspect, there is low/minimal automation that can be achieved with tools like Google Swify and Adobe CreateJS. On the other hand, Sencha had been a prime tool that provides adequate support on migration through its framework and components, especially when it comes to mapping front end MVC from Flex to HTML5.

However many times it will not be feasible (or quite difficult) to achieve the same look and feel like Flex on certain controls / components with HTML5, in those cases it will be upto management SPOCs to make a decision whether they can live with the closest match on UI / functionality that can be achieved with HTML5.

Reference

http://www.universalmind.com/blog/top-5-considerations-for-transitioning-from-flex-to-html5

http://www.effectiveui.com/downloads/publications/EffectiveUI_White_Paper_Moving_from_Flex_to_HTML.pdf

http://www.javascriptlint.com/

http://www.dehats.com/drupal/?q=node/35

Author Info

Kuntala Sinha is working with the ERS Practice team as a Technical Lead. She has around 6+ years of software experience in developing Gaming and Home Automation systems with expertise in Java, J2EE and Javascript technologies.

Akhilesh Gupta has been associated with HCL Technologies for over fourteen years. At present he is part of ERS Practice group working as Solutions Architect. Akhilesh has worked in-length in different technologies and domains right from embedded solutions to desktop, web and mobile based solutions. In ERS Practice he has been working in some specific areas like 'connected car solutions', 'HTML5 based solutions' and '3D printing'. His areas of interest include embedded segment development, technical solutions mapping to business requirements, defining technical/solution architecture from a techno-agnostic perspective.

Hello, I'm from HCL's Engineering and R&D Services. We enable technology led organizations to go to market with innovative products and solutions. We partner with our customers in building world class products and creating associated solution delivery ecosystems to help bring market leadership. We develop engineering products, solutions and platforms across Aerospace and Defense, Automotive, Consumer Electronics, Software, Online, Industrial Manufacturing, Medical Devices, Networking & Telecom, Office Automation, Semiconductor and Servers & Storage for our customers.

For more details contact**: ers.info@hcl.com**
Follow us on twitter: http://twitter.com/hclers
Our blog: http://www.hcltech.com/blogs/engineering-and-rd-services
Visit our website: http://www.hcltech.com/engineering-rd-services

HCL