ISSN: 0975-766X
CODEN: IJPTFI
*Available Online through* *Research Article*
www.ijptonline.com

# A STUDY AND DESIGN ON RICH INTERNET APPLICATION (RIA) TO ENHANCE WEB INFORMATION RETRIEVAL USING MASHUP

**V.Davina[1], Dr. RA. K. Saravanaguru[2]and Dr. K. Ramesh Babu[3]**
[1]Cognizant Technology Solutions, Kolkata, West Bengal.
[2,3]Associate Professor, School of Computer Science and Engineering, VIT University, India.
*Email: davinamichelle92@gmail.com*

*Received on: 18.10.2016* *Accepted on: 11.11.2016*

**Abstract**

Rich Internet Application (RIA) is a new technological revolution in the field of web application which gives a rich experience while using the user interface. One of the distinguishing technologies of web 2.0 is mashup technology. Mashup combines contents from different sources. In this paper, studies on various web information retrieval techniques are carried out and highlight the research gaps that are identified for incorporating the rich internet features. To address the open issues like dynamic content generation and maintenance, architecture is proposed with mashup support considering student learning as a case study. The performance of the proposed work is analysed through three popular mashup platforms namely JavaFx, Adobe Flex and Microsoft Silverlight.

**Keywords:** Rich Internet Application (RIA), Web information retrieval, Mashups, Student learning.

## Introduction

RIA gives a rich functionality for the user interface to retrieve dynamic web contents over the World Wide Web (www). In the first generation, web applications were page oriented. GUI logic and the application logic are present inside the same web page in the web server. In the second generation web applications the GUI and the application logic were separated. RIA is a third generation web application was the GUI logic is executed in the web browser whereas the application logic remains in the web server. Rich graphics and streaming multimedia are supported by RIA.

In the present days the application integration is data association, workflow [1], service composition [2], portal [3] and other integration technologies which are usually based on middleware server. These traditional combination technologies are performed in inflexible and time-consuming process to build and deploy. Also, the traditional technologies are

server-centric and do not fully exploit the computing power and storage capability of client. Mashup the newly emerging technology helps progress Web 2.0 into the enterprise[4]. This technology also reduces application development costs, perk up time-to-market and power.

Also researches have conducted qualitative and quantitative analyses and have suggested that mashup technology have improved the academic performance of the students. A mashup is created from several data sources and services, mashing up the sources and services to create something new or add value in some way. Mashup can be categorized as client presentation Mashup and client service Mashup.

Client presentation model is all about taking a package and widgets and adding them to apage with minimal integration. Whereas client service mashup is all about services like calling web services, pinging websites and feeds.[5]

**Literature Review**

There are three different ways in which an RIA can be deployed. The first is by way of plug-in. In this method the application is created in a dedicated platform and later deployed into an embedded or standalone application launched from the web browser. Adobe flex is an example of embedded approach and java web start is an example of standalone application. This type of deployment is simple to develop. The second type to deploy RIA is based on Ajax components. The interface is presented using HTML, CSS and java script is used for making asynchronous request. This approach does not need any pre-installed software and gives minimum footprint to the applications. The disadvantage of this system is that it encounters compatibility issues based on the platform and browser. The third way to deploy RIA is browser based. A user interface language built on XML is incorporated in this approach.[6] Mashup is an interactive web application. A number of organizations make their database available as APIs for mashup assembler to use. Some of them are multimedia database, search results blog post and others. Yi-Hsing Chang [7] in his research an adaptive learning system using the mashup technology is developed based on Felder and Silverman's Learning (FSL) style theory to gain knowledge of the students learning style and allow them to use a mashup search engine to get the related materials for their learning. The results suggest that this adaptive learning system has helped students in their academic performance and all of them had a positive evaluation for the learning system. Alexander Nussbaumer [8] in his work has suggested mashup based approach for personal learning environment. In this model all the components have open interface. Widgets and other services can use them. Also the mashup recommender back-end can be asked to get the

recommended widgets for ontology entities. Recently lightweight mashups are used by most of the web recommendation system as they have a lot of advantages. Content Similarity and Collaborative Filtering(CSCF) approach was proposed by Buqing [9] first content similarities where computed based on the user history of records and mashup services and secondly interest value is computed. Then user similarity model and service similarity model are calculated based on mashup QoS(Quality of Service) and a QoS predictive value is calculated. Finally the user interest value and the QoS predictive value is combined and the CSCF suggest the mashup rank and services to the user. The results showed that the CSCF method was better and effective for recommending services. Mashup technology is also used in the enhancement of business promotion for enterprise. Small and Medium Enterprises (SMEs) can be easily benefited by using the web technology and web application to establish their meta-application development methods. According Ahmed Patel[10] meta-applications are business application used to attract the customers. He also suggested using of mashup technology in such meta-application which consist of simple integrations without the involvement of IT department or a costly consultancy service for the business development. It is also necessary to provide a safe and secure environment for the mashup development as more companies are using this technology so that the enterprises can obtain a maximum profit. Wu He [11] has proposed a framework for developing a secure web mashup to avoid attacks or breaches. Mashup architecture consists of three layers: mashup layer, governance layer, service layer [12]. The top layer is the mashup layer. It aggregates and presents information supplied in the layers below it. This aggregation is normally done through rich internet applications (RIA) such as Microsoft Silverlight or Adobe Flex. The middle layer provides governance, business rules and workflow for the enterprise. Essentially, this layer makes sure that resources are used by appropriate people in an appropriate manner, and runs business processes for the company such as order management, supply notifications and risk triggers. The lowest layer consists of many data and functional services available to the enterprise. Wei Ye [13] in his work compares the traditional integrating technologies with the mashup technology. He also proposed a component and connector model for mashup technology and concluded that mashup is a better technique compared to other traditional methods such as portal and Service Oriented Architecture (SOA) compositions.

Xuanzhe Liu [14] in is work proposed a mashup architecture which consist of three components. The first component is the API/content provider. These are the providers of content where the data retrieval is done using web protocols such as SOAP and REST, web services and RSS/ATOMS. The second component is the hosting web browser. The client-side

code is directly embedded in the mashup's web page. It also contains the scripting API libraries and applets referenced by these Web pages.

This kind of mashup is called Rich Internet Application(RIA) which provides a very interactive user experience. The advantages of client-side mashing are: less overhead on behalf of the mashup server and a more seamless user-experience. This combination of both the server and the client side logic for data aggregation is by the mashup technology. The final component is the consumer web browser. The graphical rendering and the interaction with the user happens in the web browser. Duane Merrill [4] mentions that the required number of APIs is not available from content providers which force mashup developers to choose screen scraping. So that they could retrieve the information they seek to mash. The process of extracting semantic data structures of information which could further be used and manipulated using software tools for parsing and analyzing is known as scraping. Few of the mashups use screen scraping technology for data gaining. An example for mashup project that scrapes data is XMLTV [16]. It is a set of tools that aggregate TV listings from all over the world. It has two primary inherent drawbacks. The first disadvantage is that screen scraping has no precise programmatic convention between content-provider and content-consumer which is available in the APIs. The second and the final problem is that there are no specific tools for screen scrapping. The shortage of such APIs and toolkits is largely due to the tremendous application-specific needs of each individual scraping tool. Data retrieval and change is a huge problem with mashup. In fact, some researchers have begun to study how to use semantic modeling technologies to help ease the problem of automatic reasoning between different data sets. Another problem is how to solve potential danger brought by mashup applications. As mashup integrates services with different data sources and the browser often faces session invalidation, the exception handling is very important.

**Proposed Architecture with Ria: Student Learning Case Study**

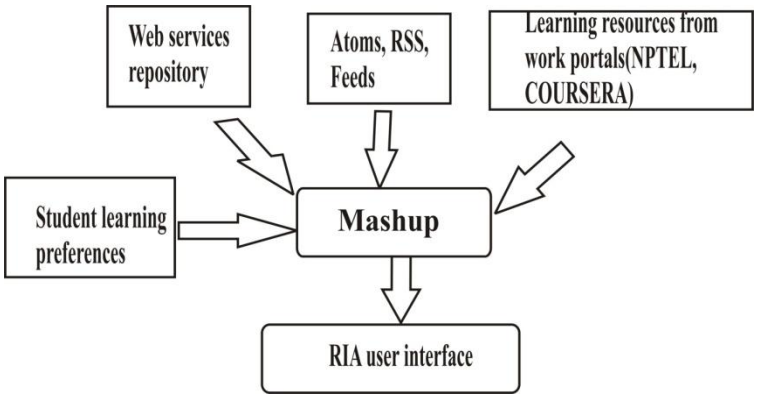


**Figure 1: Overall architecture.**

Student learning preferences: Contain the keywords or the preferred data the students want to search for. It may also contain the preferred API the student wants to connect to retrieve data. Web service repository: These are the web services available online. These are public domains which can be mashed up using the HTTP request. From these repositories services can be found and used for the user purpose. By simply entering the Web Service Description Language (WSDL) URL we can get the required web services from the public domain. The WSDL contains the interface of the web service and the implementation information. WSDL files are written in XML format.RSS: It stands for really simple syndications. It is a technology which describes of creating feeds. A feed is a frequently updated content published by a website. RSS enables users to get the updated version of the data from the websites and blogs. The atoms refer to a set of related standards. Both RSS and ATOM is used to create web feeds. Learning Portal: It brings data together in a uniform way from different sources. These portals enable users to search for contents. Also it supports the features such as e-mail service; news services etc., this portal provide e-learning service, online courses, video course from different institutions and organizations. The main aim of learning portal is to enhance the education system by providing free online data. Mashup: It is the technology which integrates different services to provide the aggregated data for the user to view different information in a single page. Mashups can be done between different types of data resources. Mashups have many advantages which provide better and effective performance for learning. RIA user interface: For displaying the mashup output we use RIA for the user interface. It gives rich look to the application and the user interaction is more when compared to the traditional way of deploying a web application. We use adobe flash/flex builder to deploy the student learning web application.

**Choosing the Right Deployment Platform for RIA**

The deployment platform for the proposed architecture is actually chosen as Adobe Flex/flash builder. The reason behind choosing this particular platform has been discussed by comparing the other two popular platforms from Oracle Java Fx and Microsoft Silverlight [15].

**Table 1: Comparison of RIA development tools.**

| **Features** | **JAVA FX** | **FLEX** | **SILVERLIGHT** |
|---|---|---|---|
| Operating System support | ✔ | ✔ | ✔ |

| | | | |
|---|---|---|---|
| Need Of Java JRE | ✔ | ✘ | ✘ |
| Cross Platform | ✔ | ✔ | ✔ |
| Integration And Workflow | ✔ | ✘ | ✔ |
| Flexibility | ✘ | ✔ | ✘ |
| Popularity | ✘ | ✔ | ✘ |
| Mobile Support | ✔ | ✔ | ✘ |
| Cross Browser Support | ✔ | ✔ | ✔ |
| User Experience | ✔ | ✔ | ✔ |
| Speed | Slow | Fast | Slow |

Table1 provides a summary of all three approaches and the selective features to support the development of RIA. From the observation Adobe Flex is chosen as a development platform.

### 1.1 Implementation workflow

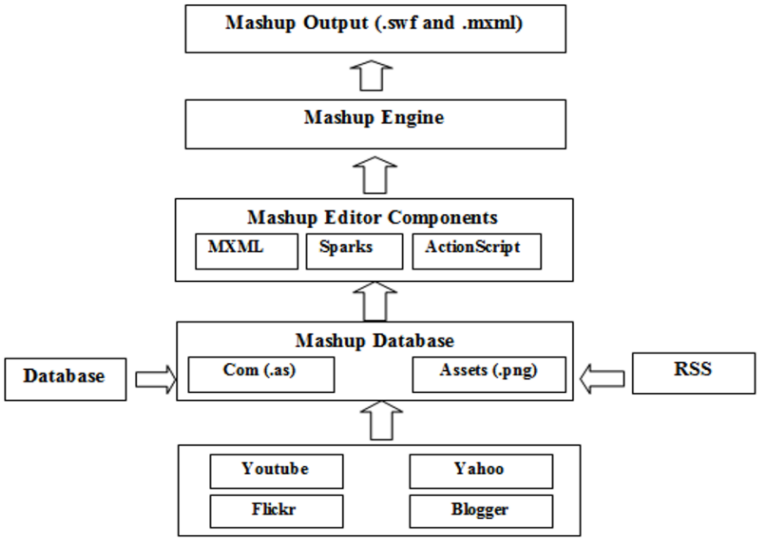


**Figure2: Workflow of the proposed method.**

MASHUP DATABASE: This is where the all the application programming interfaces (API), RSS feeds and the relational data were stored. In API and RSS, data are stored mainly in two folders: com and assets. These folders are stored in the source file.

MASHUP EDITOR COMPONENTS: This layer changes depending on the platform used by the users. In our application we have used flash builder. In this editor the components or classes used are MXML, SPARKS and action script.

MASHUP ENGINE: This is where the actual mashup happens. All the API and RSS data are mashed up and the output is displayed.

MASHUP OUTPUT: In flex builder the output is displayed using the flash which is an in-built plug in. In flex we also have an option of making the output as a desktop application using Adobe Integrated Runtime (AIR). The data grid component in adobe flex can be used to display the mashed up data in a table format.

**Deployment Scenario**

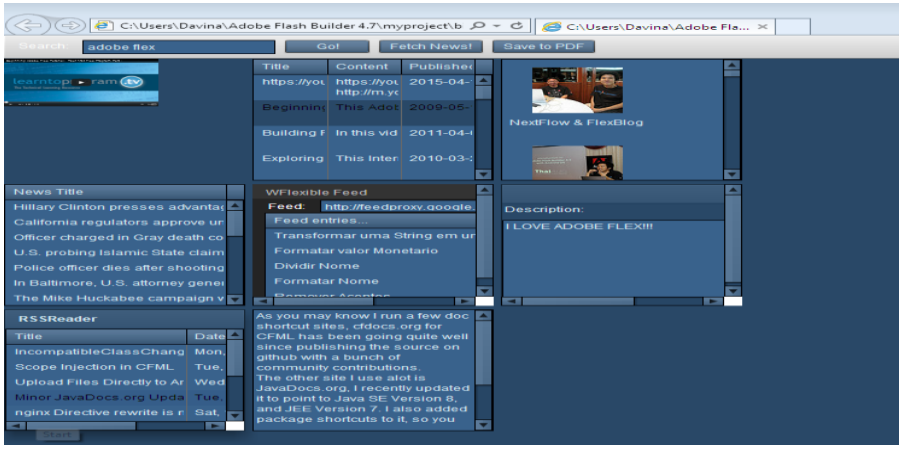


**Figure3: Snapshot of the proposed mashup RIA.**

The Figure 3 displays the final output of the mashup of API and other content providers. Flex can be easily built upon adobe flash builder software. In our application we try to create an environment where the users can search, browser for data and the information and also perform multiple tasks in a single web page through API. We use MXML language to create the application. To access the standard APIs one should register for the service and get the key with secret id to use those applications. Using the standard web protocols (SOAP, REST, JSON and XML) the corresponding APIs are invoked and the data are represented. In our application we provide APIs of YouTube, Flickr, Yahoo news and others. Then the data from the different APIs are being presented in the same page. This application can also be used as a desktop application by using the adobe AIR software.

**1.2 Sample Code Snippet of Adobe Flex to Develop Mashup**

The sample code as illustrated in Figure 4 and 5 shows the different components used in the mashup application (fx,mx).The different components are MXML component(mx), sparks(s) and flex(fx). These components help to build the mashup. Also <mx:HTTPService> connects to the desired web services as in this case to Yahoo API. The <mx:HTTPService> is always mentioned in the <fx:Declarations> tag.

```
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:mx="library://ns.adobe.com/flex/mx"
               minWidth="955" minHeight="600"
               creationComplete="init()" initialize="httpRSS.send()">
    <fx:Declarations>
        <!-- Place non-visual elements (e.g., services, value objects) here -->
        <mx:HTTPService id="YahooNewsService"
                        url="http://rss.news.yahoo.com/rss/mostemailed"
                        method="GET"
                        result="YahooNewsService_resultHandler(event)"
                        fault="YahooNewsService_faultHandler(event)"/>
```

**Figure 4: Components in Adobe Flex.**

The component used to display the mashup output is the data Grid component. When all the API's and web services are mashedup they are displayed in the user interface by this component. This also has other sub classes such as colums, Data Grid Colum and more.

```
<mx:DataGrid  id="dg" width="300" height="200" doubleClickEnabled="true"
              dataProvider="{dp}" doubleClick="displayYouTube(getURL())">
    <mx:columns>
        <mx:DataGridColumn dataField="title" labelFunction="dgLabelFunction"
                           headerText="Title" width="75"/>
        <mx:DataGridColumn dataField="content" labelFunction="dgLabelFunction"
                           headerText="Content" width="75" />
        <mx:DataGridColumn dataField="published" labelFunction="dgLabelFunction"
                           headerText="Published" width="75"/>
    </mx:columns>
</mx:DataGrid>
```

**Figure-5: Display component.**

## Conclusion and Future Work

Mashup technology is more lightweight in terms of computing power and storage capacity when compared to other traditional integrating systems. In this paper, a web based application using mashup is proposed and presented using RIA platform. The objective is to create a RIA based mashup application for student learning domain and it was successfully implemented using Adobe flash builder. It is also planned to include a customization icon as a future work which will enable the user to customize the APIs. Finally, a live forum is created for discussions and also to collect information required among online users.

## References

1. Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., &Trickovic, I. (2003). Business process execution language for web services.
2. Besson, F., Moura, P., Kon, F., &Milojicic, D. (2015). Bringing Test-Driven Development to web service choreographies. Journal of Systems and Software, 99, 135-154.

3. Nouri, A., & Daniel, F. (2016). Interactive, live mashup development through UI-oriented computing. In Rapid Mashup Development Tools (pp. 31-49). Springer Internationxal Publishing.
4. D. Merrill, Mashups: The New Breed of Web Application, www- 128.ibm.com/developer works/library/xmashups. html?ca=dgr-lnxw16MashupChallenges.
5. Daniel, F., &Pautasso, C. (Eds.). (2016). Rapid Mashup Development Tools: First International Rapid Mashup Challenge, RMC 2015, Rotterdam, The Netherlands, June 23, 2015, Revised Selected Papers (Vol. 591). Springer..
6. Farrell, J., &Nezlek, G. S. (2007, June). Rich internet applications the next stage of application development. In 2007 29th International Conference on Information Technology Interfaces (pp. 413-418). IEEE.
7. Chang, Y. H., & Chen, Y. Y. (2012, July). A mashup-based adaptive learning system. In 2012 International Conference on Machine Learning and Cybernetics (Vol. 5, pp. 1721-1726). IEEE.
8. Nussbaumer, A., Dahrendorf, D., Schmitz, H. C., Kravcik, M., Berthold, M., & Albert, D. (2014). Recommender and guidance strategies for creating personal mashup learning environments. Comput. Sci. Inf. Syst., 11(1), 321-342.
9. Buqing, C., Tang, M., & Huang, X. (2014). Cscf: A mashup service recommendation approach based on content similarity and collaborative filtering. International Journal of Grid & Distributed Computing, 7(2).
10. Patel, A., AlShourbaji, I., & Al-Janabi, S. (2014). Enhance business promotion for enterprises with mashup technology. Middle-East Journal of Scientific Research, 22(2), 291-299.
11. He, W. (2014, August). Enhancing secure web mashups development in enterprise environments. In Enterprise Systems Conference (ES), 2014 (pp. 279-283). IEEE.
12. Jin, H. J., & Lee, H. C. (2008, April). Web services development methodology using the mashup technology. In Smart Manufacturing Application, 2008. ICSMA 2008. International Conference on (pp. 559-562). IEEE.
13. Ye, W., Hu, W., Zhao, W., Gao, X., Zhang, S., & Wang, L. (2009, July). Towards lightweight application integration based on mashup. In 2009 Congress on Services-I (pp. 500-506). IEEE.
14. Liu, X., Hui, Y., Sun, W., & Liang, H. (2007, July). Towards service composition based on mashup. In Services, 2007 IEEE Congress on (pp. 332-339). IEEE.
15. Franz, K. (2015). Tests zurEffizienz. In Handbuchzum Testen von Web-und Mobile-Apps (pp. 215-236). Springer Berlin Heidelberg.