Available online at www.sciencedirect.com
SciVerse ScienceDirect
Procedia Technology 3 (2012) 205 – 213
Procedia Technology

The 2012 Iberoamerican Conference on Electronics Engineering and Computer Science

Developing Social Networks Mashups: An Overview of REST-Based APIs

Mario Andrés Paredes-Valverde[a]*, Giner Alor-Hernández[a],Alejandro Rodríguez-González[b], Gandhi Hernández-Chan[c]

[a]*Division of Research and Postgraduates Studies, Instituto Tecnológico de Orizaba, Oriente 9 No. 852, Emiliano Zapata, 94320 Orizaba, Mexico*
[b]*Universidad Carlos III de Madrid, Avenida Universidad No. 30 Escuela Politécnica Superior 28911 Leganés, Madrid, Spain*
[c]*Division of Information and Communication Technologies, Universidad Tecnológica Metropolitana,Calle 115 No. 404 Santa Rosa, 97279 Mérida, Mexico*

Abstract

The social networks have become in a powerful diffusion media in several fields such as communication, e-commerce and entertainment. However, the development of new applications that combine the functionality of different social networks with the purpose of providing added-value to users is not very common. In this context, a new kind of applications called mashups has emerged. A mashup is a web application that integrates data from multiple web sources in order to provide a unique service. Internal data sources, RSS/Atom feeds, Screen-Scraping and Web Services are some resources used by mashups. Nowadays, most of Web Services provided by social networks use the REST-based architectural style because it offers significant advantages in comparison with other technologies. The contribution of this paper is a review of REST-based APIs for the development of mashups that integrate well known social networks such as Youtube©, Picasa©, and Flickr©, among others. In addition, a set of 4 mashups were developed combining the APIs discussed. Also, this work provides a development guide to perform tasks such as extraction and combination from different data sources, as well as leads to the emergence of new ideas for developing web applications.

Keywords: Mashup; REST; Social Networks; Web Services

* Corresponding author. *E-mail address:* marioparedes@acm.org

doi:10.1016/j.protcy.2012.03.022

1. Introduction

Nowadays, Internet is one of the most used diffusion media in the world, this is largely due in part to social networks, such as Facebook©, Twitter©, Youtube©, among others. However, there are few applications that combine the functionality of different social networks. For example, although there are different social networks to share photos and images such as Flickr©, Photobucket© and Picasa©; it is necessary an application that combine the functionality of these social networks, providing faster results in comparison with doing these tasks in a separately way in each social network. In this sense, the need to have this kind of web applications has emerged. These web applications are called mashups. A mashup is a web application that integrates data from multiple web sources to provide a unique service [1]. Internal data sources, XML feeds, Screen-Scraping and Web Services are resources used by mashups. A web service is a software system designed to support interoperable machine-to-machine interaction over a network [2]. In early 2000, some organizations used SOAP for developing Web Services. Nowadays, REST has emerged as a new approach for developing Web Services. REST is an architectural style for distributed hypermedia systems [3] and is used to provide Web Services by a large number of social networks such as Flickr©, Twitter© or, Ebay©. REST describe 6 constraints applied to the architecture: (1) Client-server: an uniform interface separates clients from servers, (2) Stateless: no client context is stored on the server between requests, (3) Cacheable, clients can cache responses, (4) Layered system: a client cannot ordinarily tell whether it is connected directly to the end server, or to an intermediary along the way, (5) Code on demand: servers are able temporarily to extend or customize the functionality of a client by the transfer of executable code, and (6) Uniform interface: which is considered fundamental to the design of any REST service, in this interface, all resources are identified individually in the requests using URIs. In Fig.1 the REST architectural style is presented.

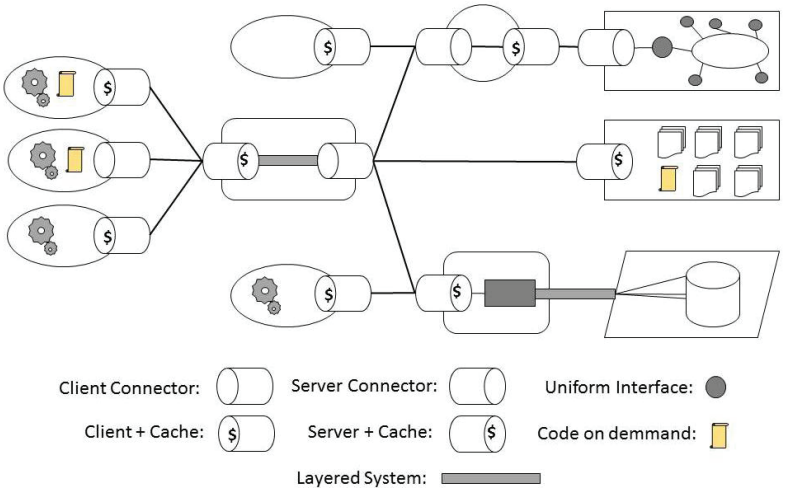

Figure 1 REST architectural Style

REST-based Web Services provide advantages such as: (1) requests and responses are shorter than SOAP, because it requires an enveloped XML format in every request and response, (2) the bandwidth required to transport requests and responses is lower than SOAP, and (3) to process requests and responses less memory and processing time is required than SOAP.

Nowadays, there are tools that allow developing mashups, such as: Yahoo PIPES© [4] or Dapper© [5] among others. However, these tools have limitations for building visually-appealing user interfaces, an issue of great importance in current web 2.0 applications. In order to have a greater number of mashups that integrates functionalities of social networks and delivers results of added-value to user, a review of REST-based APIs for social networks is mandatory. The main contribution of this paper is based on REST-based APIs for developing mashups that integrate well known social networks, such as Delicious©, Digg©, Myspace©, Vimeo©, Youtube©, Slideshare©, Scribd©, Flickr©, Picasa© and Photobucket©, which have a great number of users. Furthermore, as a case-study of these APIs, 4 mashups were developed integrating REST-based Web Services coming from the above social networks.

This paper is structured as follows. Section 2 presents the State-of-the-Art in mashup development. Section 3 presents a review of 10 REST-based APIs for social networks. Section 4 presents a set of 4 mashups that combine the APIs discussed in this paper, and finally, we present our conclusions and emphasize our contribution.

2. State of the Art in Mashup Development

Several initiatives are normally rely on the use of well-known tools and propose new languages for developing mashups, Other works discuss the importance of using mashups to build new application in cloud computing. Some of these works have obtained outstanding partial results. Bitzer et. al. [6] presents an analysis of mashups as architecture for knowledge management systems; also the integration potentials of mashups in a standardized architecture of knowledge management systems are examined. Belimpasakis et. al. [7], describes a web service platform for building augmented and mixed reality solutions. This platform, allows clients to create, retrieve and modify augmented reality enabled content using a unified interface based on standard web technologies like HTTP, REST and Linked Data. Fang et. al. [8] presents a lightweight 3D virtual world service platform using RESTful Web Services. It covers most of the key services now provided by 3D virtual world platforms. Antilla et. al. [9], describes a new approach to use distributed, lightweight Web Services for mobile collaboration. In this approach the collaboration between the people is direct without the need for centralized information storage. AlShahwan et. al. [10], proposes an implementation of two frameworks for providing Web Services from resource constraints devices. The first framework is implemented to provide SOAP Web Services and the second one is implemented to provide RESTful Web Services. Noimanee et. al. [11], describes an approach to implementing a conference system using web service technology fused with the concept of business process modeling to support the workflows of conference management tasks. Kopecky et. al. [12], proposes a microformat called hRESTS (HTML for RESTful Services) for machine-readable descriptions of web APIs. The hRESTS microformat describes main aspects of services, such as operations, inputs and outputs. Cha et. al. [13], presents a retrieval method using semantic technologies based on RESTful Web Services; this method provides more filtered and extended results than similarity based keyword searching methods. Zhao et. al. [14], proposes a formal model for describing individual RESTful Web Services and automating the composition. This work represents an initial effort towards the problem of automated RESTful web service composition. Mehmet et. al. [15], presents a lightweight enterprise data integration service called Damia, this service offers a framework and functionality for dynamic entity resolution, streaming and other value features particularly important in the enterprise domain.

Some others authors have focused their efforts on building tools for developing mashups, Abiteboul et. al. [16] presents MatchUp, a system that supports rapid, on-demand and intuitive development of mashups, based on a novel autocompletion mechanism. Tuchinda et. al. [1] presents Karma, a mashup builder that incorporates the concept of programming by demonstration. Wong et. al. [17], developed an end-user

programming tool called Marmite, which lets end-users create mashups that purpose and combine existing web content and services.

These initiatives suffer from several drawbacks such as: a) a vast amount of social networking are not included for developing mashups, and b) lack of usage for advanced capabilities of rich Internet application technologies. This work tries improving these aforementioned deficiencies providing a guide for developing REST-based social networks mashups using rich internet technologies in order to improve the user's experience.

3. REST-Based APIs for Developing Mashups

The social networks provide APIs to facilitate the applications development that integrate several elements and features of these social networks. A significant number of these APIs are REST-based. In Table 1, a review of 10 REST-based APIs for social networks is presented. These social networks were selected because represent different kinds of functionalities and features.

Table 1. Comparison of REST-Based APIs for Social Networks

API	Features	Response formats
Delicious©	It allows updating, adding, getting, and deleting bookmarks, getting, deleting, and renaming tags. This API is available at [18]. Delicious© also has data feeds which allow getting recent bookmarks by tag, popular bookmarks by tag, bookmarks for a specific user and recent bookmarks for a URL. These data feeds are available at [19].	XML, RSS, JSON
Digg©	It allows searching bookmarks, getting and posting comments, following a user, getting activity, comments, news and user's information, getting comments and stories information, getting, removing and saving stories. This API is available at [20].	JSON
Myspace©	It allows searching people, images, and videos, obtaining and uploading photos and videos, posting status updates and activities to the user's stream, obtaining user information and obtaining the activity stream of users and their friends. This API is available at [21].	JSON, XML
Vimeo©	It allows obtaining recent user's activity, creating, deleting and modifying albums, searching people and videos, modifying and subscribing to channels and groups, getting contact list, uploading videos. This API is available at [22].	JSON, JSONP, PHP, REST
Youtube©	It allows fetching videos feeds, comments, responses, and playlists, querying for videos that matching particular criteria, making authenticated requests to modify this information, uploading new video content to the web site. This API is available at [23].	Atom 1.0, RSS 2.0, Atom Publishing Protocol
Flickr©	It allows fetching, uploading, modifying, and deleting information such as photos, comments, tags, blogs, collections, and contact lists. Also it allows searching users by email or username. This API is available at [24].	REST, XML-RPC, SOAP, JSON, PHP
Photobucket©	It allows uploading images and videos, getting all recent media for: a user, all users or group albums, searching media matching a specific term, getting details associated with one piece of media, updating titles, descriptions, and tags. This API is available at [25].	XML, JSON, JSONP, PHP
Picasa©	It allows viewing, updating, creating, editing, or deleting albums, photos, and comments, querying for items that match particular criteria. This API is available at [26].	Atom 1.0, RSS 2.0, Atom Publishing Protocol
Scribd©	It allows uploading, converting, viewing, deleting and searching documents. This API is available at [27].	XML
Slideshare©	It allows uploading, editing, and deleting slideshows, retrieving slideshows information by user, tag, or group, retrieving, groups, tags, and contacts by user, searching slideshows. This API is available at [28].	XML

To illustrate the use of REST-based APIs for social networks reviewed in the table above, in the following section, 4 mashups are presented that integrate these APIs.

4. Developing Mashups Using REST-Based Web Services

The mashups developed integrate data from multiple web sources to provide a unique service, these applications are important because allow to deliver results of added-value to a user. Next, 4 mashups that combine Web Services of social networks discussed in the previous section are presented. These mashups have features that are not present in any other well-known application.

In Fig. 2(a), the mashup called Digglicious is presented. In this mashup, the Digg© API and Delicious© Feeds were integrated. Digglicious allows searching bookmarks that match particular keywords on Digg© and Delicious©.

In Fig. 3(a), the Digglicious workflow is presented. In order to start searching for bookmarks a keyword must be entered. Next, a request with the corresponding parameters is sent to each social network. Then, JSON documents are returned with a bookmark list matching the keyword. These documents are parsed and the results are displayed in a HTML format. The search on Digg© can be configured to get list order by promote date, submit date and digg count. Also, this search can be configured to get news, videos, images or all. For this case, digg count and all were selected. For the case of Delicious©, the data feed used was popular bookmarks by tag. To access some information on the social networks in question, API keys are needed. To get these API keys, an application must be register on Digg© and Delicious©. The information used in this mashup not requires any API key, because this information is public.

In Fig. 2(b), the mashup called VideoSpace is presented. In this mashup the Myspace© API, Vimeo© API and Youtube© API were integrated. VideoSpace is a metasearch engine because it allows searching videos that match particular keywords on Myspace©, Vimeo© and Youtube©.

These social networks were selected, because nowadays a great number of web users search videos browsing on different social networks. The goal of this mashup is provide results faster in comparison with doing this work in a separately way in each social network. In Fig. 3(b), the VideoSpace's workflow is presented. In order to start searching for videos a keyword must be entered. Next, a request with the corresponding parameters is sent to each social network. Then, for the case of Myspace© and Vimeo©, JSON documents are returned with a video list matching the keyword. For the case of Youtube©, the video list returned consists of RSS Feeds. These documents are parsed and the results are displayed in a HTML format. To view a video, it must be selected from the list.

The search on Myspace© can be configured by tag or text. Also, this search can be configured to get official or music videos. For this case, text and official videos were selected. The search on Vimeo© can be configured to get list order by relevance, most played, most commented or most liked. For this case, relevance was selected. The search on Youtube© can be configured to get videos of a specific category such as music, comedy, education, entertainment and sports. Also, this search can be configured to get the list order by relevance, rating, and view count. For this case, music and relevance were selected. As mentioned in Table 1, these social networks use different response formats. This feature gives the opportunity to select the format that meets the needs of the application. This represents a great advantage in comparison with SOAP-based Web Services because SOAP has only one response format. To access the information on the social networks in question, API keys are needed. To get these API keys, an application must be registered on Myspace©, Vimeo© and Google©. For searching videos on Youtube©, a Java-based library provided by Google© was used. This library allows performing all activities of Youtube© API described in Table 1.

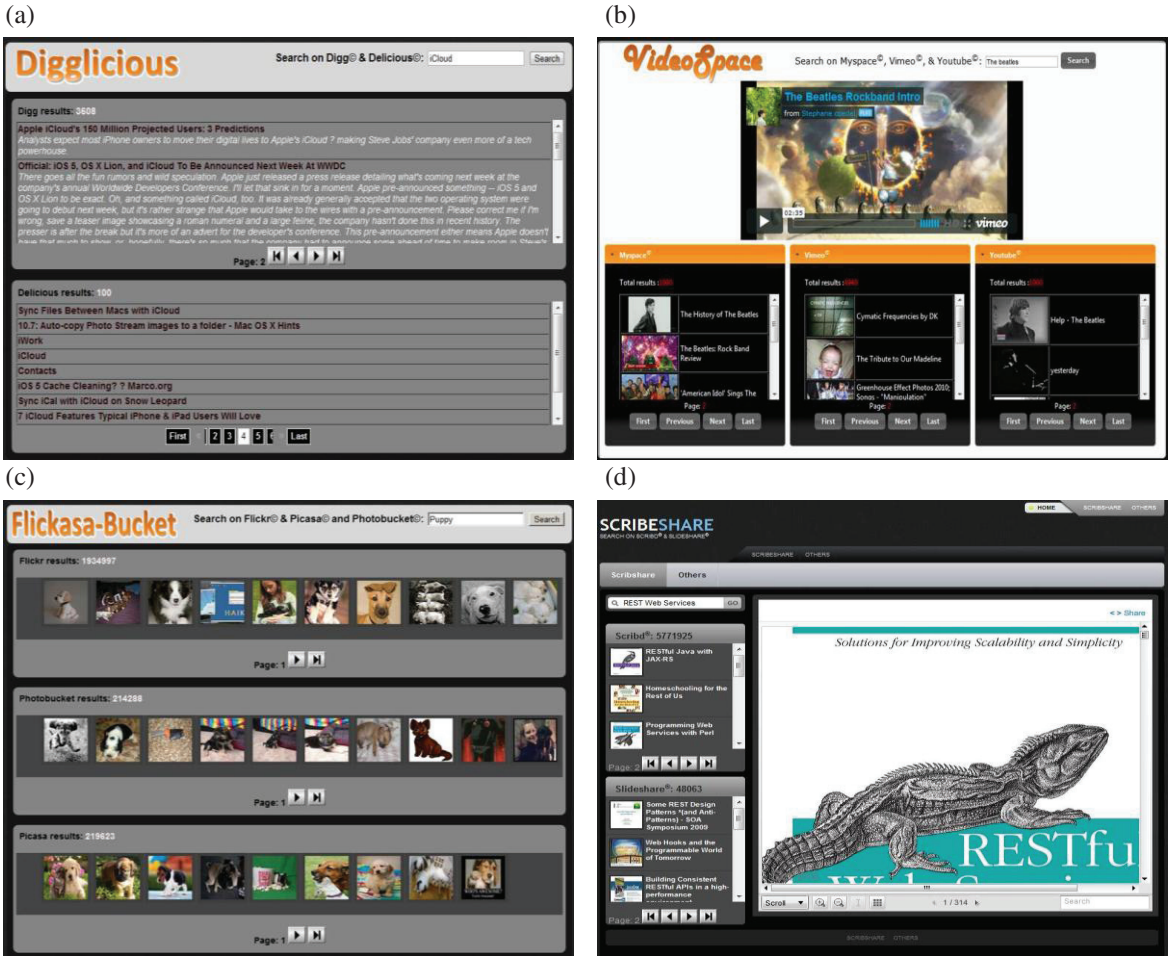

Fig. 2. (a) Mashup that integrates Digg© API and Delicious© API; (b) Mashup that integrates Myspace© API, Vimeo© API and Youtube© API; (c) Mashup that integrates Flickr© API, Photobucket© API and Picasa© API; (d) Mashup that integrates Slideshare© API and Scribd© API.

In Fig. 2(c), the mashup called Flickasa-Bucket is presented. In this mashup the Flickr© API, Photobucket© API and Picasa© API were integrated. Flickasa-Bucket is a metasearch engine because it allows searching pictures that match particular keywords in Picasa©, Flickr© and PhotoBucket©. The goal of this mashup is provide results faster in comparison with to do this work in a separately way in each social network.

In Fig. 3(c), the Flickasa-Bucket's workflow is presented. In order to start searching for pictures, a keyword must be entered. Next, a request with the corresponding parameters is sent to Flickr©, Photobucket© and Picasa©. Then a list of pictures that match the keyword is returned by each social network. As mentioned in Table 1, these social networks use different response formats, therefore it is necessary to analyze each response in order to display the results in a HTML format. The possibility of using JSON format in the responses represents a great advantage in comparison with SOAP-based Web Services, because sometimes the responses in JSON format are shorter than SOAP messages. This causes the decrease in the bandwidth required to transport every response. To view a picture in its original size, it must be selected from the list.

For accessing to the information on these social networks, API keys are needed. To get these API keys, an application must be registered on Flickr© and Photobucket©. For the case of Picasa©, the same API key that was used on Youtube© can be used, since both social networks are sponsored by Google©. For developing Flickasa-Bucket, a set of Java-based libraries were used that allow performing many actions described in Table 1.

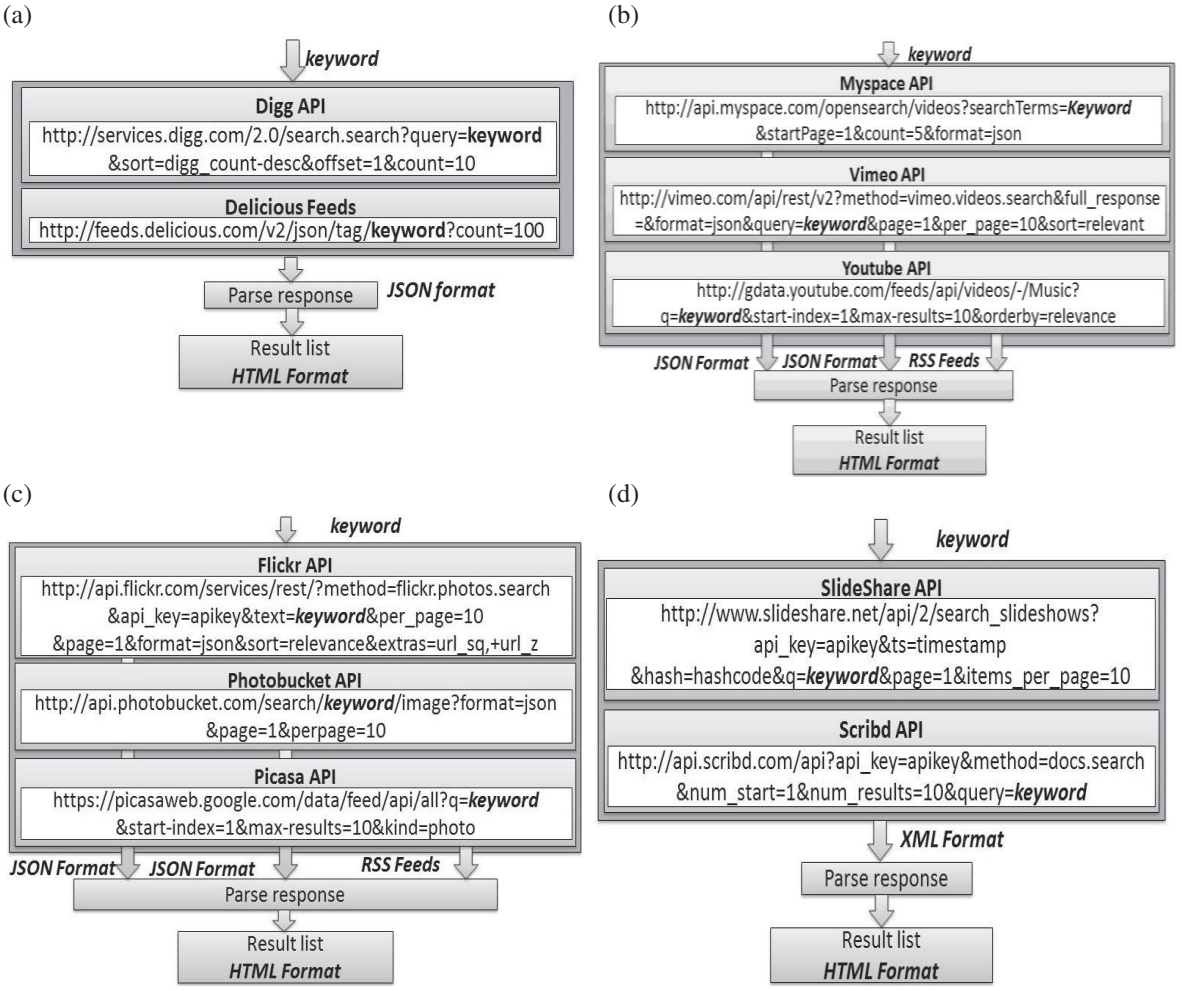

Fig. 3. (a) Digglicious workflow; (b) VideoSpace's workflow; (c) Flickasa-Bucket's workflow; (d) ScribeShare's workflow.

In Fig. 2(d), the mashup called ScribeShare, is presented. In this mashup the SlideShare© API and Scribd© API were integrated. ScribeShare is a metasearch engine [29] because it allows searching documents that match particular keywords in Scribd© and SlideShare©. The searching for documents is performed by a great number of web users, so with this mashup this work is faster in comparison with to do this work in a separately way in each social network.

In Fig. 3(d), the ScribShare's workflow is presented. In order to start searching for documents, a keyword must be entered. Next, a request with the corresponding parameters is sent to each social network. The search

on SlideShare© can be configured by tag or text; in this case the second option was selected. After sending the request, an XML-based document is returned with a list of documents that match the keyword. These lists are parsed and presented in a HTML format. To view a particular document, it must be selected from the list. To access the information on the social networks in question, API keys are needed. To get the API key for Scribd© an application must be registered on the corresponding web site.

For SlideShare©, the key is requested on the corresponding web site and it is provided via email. To search on SlideShare© and Scribd©, the Java-based libraries SlideShare4j and JavaScribd were used. These libraries allow performing functionalities described in Table 1.

The REST-based Web Services represent a great option to access the resources provided by different organizations. This kind of Web Services offers advantages such as: the use of different response formats such as JSON, XML, and Atom; the integration of these kinds of services is easy, and the access to resources is too intuitive, among others. Also, there are many more options for developing mashups, for example, a mashup that integrates the Netflix API and Rotten Tomatoes API for recommending movies.

5. Future Work

As future work, we can mention a more complete survey of REST-based APIs for social networks such as Facebook©, Twitter©, LinkedIn©, as well as the development of other mashups integrating at least three different REST-based API. In this sense, it is possible establishes an architecture that allows an integration scheme for this kind of application. This approach will offer a clear and general idea to perform tasks such as extraction and combination of information from different data sources, and the generation of new ideas for developing web applications.

6. Conclusions

Many social networks provide Web Services, which allow integrate features of these applications in any application; these services represent a great opportunity to developing innovative web applications. However, these services have not been adequately exploited or not exploited at all. In this work, we have presented a review of REST-based APIs for social networks as well as a set of developed mashups that integrate the APIs discussed. This kind of applications allow provide results faster in comparison with doing tasks in a separately way in each social network. To develop these mashups, we developed workflows knowing the general process for developing mashups that integrate REST-based APIs. In addition, we could work with different response formats, an important feature in REST-based Web Services, and which represents a great advantage for the developers, because it allows them to select the response format that best suits their needs. Finally, we believe this work presents a development guide to the emergence of new ideas for developing web 2.0 applications.

Acknowledgements

This work is supported by the General Council of Superior Technological Education of Mexico (DGEST). Additionally, this work is sponsored by the National Council of Science and Technology (CONACYT) and the Public Education Secretary (SEP) through PROMEP. Also, this work is supported by the Spanish Ministry of Industry, Tourism, and Commerce under the project TRAZAMED (IPT-090000-2010-007) and the Spanish Ministry of Science and Innovation under the project FLORA (TIN2011-27405).

References

[1] Rattapoom Tuchinda, Pedro Szekely, and Craig A. Knoblock. Building Mashups by example. In Proceedings of the 13th international conference on Intelligent user interfaces (IUI '08). ACM, New York, NY, USA, 2008, pp. 139-148.
[2] Workgroup W3C, (2004, February 11), "Web Services Architecture", [Online]. Available: http://www.w3.org/TR/ws-arch/.
[3] R.T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures", Ph.D. dissertation, California Univ., Irvine, Cal, 2000.
[4] Yahoo Pipes, Available: http://pipes.yahoo.com/pipes/ (2011, June 15).
[5] Yahoo Dapper, Available: http://open.dapper.net/ (2011, June 15).
[6] Ian Gorton, Chandrika Sivaramakrishnan, Gary Black, Signe White, Sumit Purohit, Michael Madison, and Karen Schuchardt. Velo: riding the knowledge management wave for simulation and modeling. In Proceeding of the 4th international workshop on Software engineering for computational science and engineering (SECSE '11). ACM, New York, NY, USA, 2011, pp. 32-40.
[7] P. Belimpasakis, P. Selonen, Y. You, "A Web Service Platform for Building Interoperable Augmented Reality Solutions". International Augmented Reality Standards Workshop, Seoul, Korea, October 2010. pp. 1-5.
[8] Zhi-Cong Fang and Hong Cai. Building Interoperable 3D Virtual World Platforms with RESTful Web Services. In Proceedings of the 2009 Congress on Services - I (SERVICES '09). IEEE Computer Society, Washington, DC, USA, 2009, pp. 70-77.
[9] Ville Antila and Jani Mantyjarvi. Distributed RESTful Web Services for Mobile Person-to-Person Collaboration. In Proceedings of the 2009 Third International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST '09). IEEE Computer Society, Washington, DC, USA, 2009, pp. 119-124.
[10] Feda AlShahwan and Klaus Moessner. Providing SOAP Web Services and RESTful Web Services from Mobile Hosts. In Proceedings of the 2010 Fifth International Conference on Internet and Web Applications and Services (ICIW '10). IEEE Computer Society, Washington, DC, USA, 2010. pp. 174-179.
[11] P. Noimanee, and Y. Limpiyakorn. Towards a RESTful process of conference management system. In Proccedings of the International Multiconference of Engineers and Computer Scientists, Hong Konk, 2009. pp. 996-1001.
[12] Jacek Kopecky, Karthik Gomadam, and Tomas Vitvar. hRESTS: An HTML Microformat for Describing RESTful Web Services. In Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01 (WI-IAT '08), Vol. 1. IEEE Computer Society, Washington, DC, USA, 2008. pp. 619-625.
[13] Seung-Jun Cha, Yun-Jeong Choi, and Kyu-Chul Lee. Development of Retrieval Methods for RESTful Web Services Using Semantic Technologies. In Proceedings of the 2010 IEEE/ACIS 9th International Conference on Computer and Information Science (ICIS '10). IEEE Computer Society, Washington, DC, USA, 2010. pp. 912-917.
[14] Haibo Zhao and Prashant Doshi. Towards Automated RESTful Web Service Composition. In Proceedings of the 2009 IEEE International Conference on Web Services (ICWS '09). IEEE Computer Society, Washington, DC, USA, 2009. pp. 189-196.
[15] Mehmet Altinel, Paul Brown, Susan Cline, Rajesh Kartha, Eric Louie, Volker Markl, Louis Mau, Yip-Hing Ng, David Simmen, and Ashutosh Singh. Damia: a data mashup fabric for intranet applications. In Proceedings of the 33rd international conference on Very large data bases (VLDB '07). VLDB Endowment. 2007. pp. 1370-1373.
[16] Ohad Greenshpan, Tova Milo, and Neoklis Polyzotis. Autocompletion for mashups. Proc. VLDB Endow. 2, 1 (August 2009), 2009. pp. 538-549.
[17] Jeffrey Wong and Jason I. Hong. Making mashups with marmite: towards end-user programming for the web. In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '07). ACM, New York, NY, USA, 2007. pp. 1435-1444.
[18] Delicious API, Available: http://www.delicious.com/help/api (2011, June 15).
[19] Delicious Feeds, Available: http://www.delicious.com/help/feeds (2011, June 15).
[20] Digg developers, Available: http://developers.digg.com/ (2011, June 15).
[21] Myspace Developer Platform, Available: http://developerwiki.myspace.com/index.php?title=Category:RESTful_API (2011, June 15).
[22] Vimeo Developer- Available: http://vimeo.com/api (2011, June 15).
[23] Youtube APIs and Tools. Available: http://code.google.com/intl/en-EN/apis/youtube/overview.html (2011, June 13).
[24] App Garden Flickr API. Available: http://www.flickr.com/services/api/ (2011, June 15).
[25] Photobucket Developers. Available: http://photobucket.com/developer (2011, June 28).
[26] Picasa Web Albums Data API. Available: http://code.google.com/intl/en-EN/apis/picasaweb/overview.html (2011, June 28).
[27] Scribd Developers Tools and APIS. Available: http://es.scribd.com/developers (2011, June 28).
[28] Slideshare Developers & API. Available: http://www.slideshare.net/developers (2011, June 28).
[29] Leonidas Akritidis, Dimitrios Katsaros, and Panayiotis Bozanis. Effective Ranking Fusion Methods for Personalized Metasearch Engines. In Proceedings of the 2008 Panhellenic Conference on Informatics (PCI '08). IEEE Computer Society, Washington, DC, USA, 2008. pp. 39-43.