Mashup: Web application hybrid

Stephen Baden
Drew Lefebvre
Jamal Dar
Mathew Langner
Min-Tse Yu

Mashups - Introduction

What is a mashup?

The term mashup first appeared in music. It means mixing different types of music to create different sound effects. Then the term is used in Web 2.0. In Web 2.0, mashup can be a webpage or a web application that integrates contents such as data, presentation, or functionality from different sources to create a new service. The content can be obtained through a public interface, obtained from RSS/Atom, or provided by third party. More and more web applications have published their APIs to allow other users like software developers or webpage developers to use. Developers can use these APIs to integrate data or functionalities into their own service instead of building them from scratch by themselves. The new service depends on how the developer integrates the data or functionalities from APIs. Developers can mashup with multiple data sources like maps, photos, bookmarks into a new service they desire.

Types

There are numerous types of mashups: business mashups, consumer mashups, and data mashups. Business mashup are web applications that integrate their own content, often enhanced with external web services. It is used to allow collaborative action among businesses and developers. Business mashup are secure and usually visually rich web applications that expose actionable information from diverse internal and external information sources. Consumer mashups integrate different visualisations and data elements from multiple public sources to create a simple browser user interface that is more appealing consumption of information. Data mashup is opposites of consumer mashup. It integrates the same data level, whether it’s integrating files, database, external web service APIs from different multiple sources into a single representation. The most common type of mashup is the consumer mashup, aimed at the general public.

Mashups - History

Evolved from portals

To better understand mash-ups and how they evolved, it is necessary to understand the concept of a web portal. Web portals, being a feature of what is now called Web 1.0, were developed as a way to aggregate information from many resources into one unified format. Information such as news and stock-prices, were combined with services such as web search, web email and database access in order to provide users with a single point of access to Internet resources. Portals are developed under the traditional web server model which uses "portlets", a web-application with a standardized API, to observe and process information in real-time and generate a markup representation of that information. A portal will then aggregate the information typically by displaying several non-overlapping windows, each window being managed by a portlet. As expected this approach is somewhat limited. Mostly because it leads to a segregated data format, where the data is presented in a categorical style as it is difficult for strictly server side technology to be able to efficiently integrate that data according to the users individual needs. The non-existence of data sharing protocols also meant that broad access to the data was unavailable as data access was typically limited to that organization which generated it. For example, the data generated by a large institution such as a government or corporation was only available to that institution's web portal. This leads technology that can merely present small sets of data, mostly because of the limited ability to do anything useful with it.

Examples of portals include: iGoogle, MSN, and Yahoo!

The evolution of portals and the incorporation of mash-ups

As Internet technology evolved, web interfaces became more standardized and widespread access to data became more feasible; the development of powerful web-application hybrids that are able to access data provided by various web services and combine that data in new and interesting ways became more prevalent. These web-application hybrids became known as Mash-ups. The development of mash-ups as a means of integrating data from different sources was brought on by that emergence of powerful client side web software. The leading example of one such technology is AJAX (Asynchronous Javascript and XML). AJAX allows for a whole new paradigm in web software development. This is because it allows web applications to receive asynchronous data from web servers in the background, without interfering in the appearance of a web page. It makes possible web programs which respond more like desktop applications and rather than the click and wait server side oriented web applications of the past. Mash-ups take advantage of AJAX and the standardized web APIs provided by various content sources in order to obtain data, and that data is then aggregated mostly on the client site (although server side aggregation can occur). This has allowed for more integrated data representation because data can be manipulated by the user after it has been received from the server. Mash-ups provide the user with a much more customizable and interactive interface to data as they are not based on the read and update event models of portals which are defined through specific portlet APIs. These have instead been replaced by operations based on client side and web service based architectural principles. As such mash-ups can be integrated and combined into a new type of Internet portal capable of providing it's users with a much richer set of features.

Examples of Mash-ups include: Pubwalk.com, WeatherBonk.com, Parkingcarma.com

Mashups and beyond

The development and standardization of various web APIs and powerful client side data transfer and manipulation technologies, coupled with the increasing popularity and power of mobile devices and prevalence of powerful network infrastructure has allowed web sources to not only aggregate data but also share that data more easily.

The principles that allow for the advent of mash-ups can also be extended, in order to allow for web-applications that have the same functionality as many desktop applications. The ability to provide such applications over a web connection removes the need for the user to have individual programs, other than a browser, installed on their own computer. If the only program needed is a browser then it would make sense to integrate the browser and the operating system, essentially making the operating system a browser. This seems to be where the current trend is going, as Google's chrome OS follows this model. Any device, with Chrome OS installed is merely a web browser and most of the functionality that would be provided by desktop apps running on the the user's machine would instead by provided by web-applications. The end result of mash-ups and the ideas involved leads to a synthesis of all types of computer applications, allowing for some exciting possibilities.

Mashups - Architecture

Mashup architecture can be seen from a general perspective by looking at what needs to done to successfully design and develop a functioning, integrated application. Service oriented Architectures are general design principles that package functionalities into groups of services that inter operate. Mashup architecture design can use this as a basic starting to point to ensure that the process of integrating functionalities of two applications goes as smoothly as possible. The two requirements that an SOA must have are:

The multiple elements that are being combined must have some way in which they can inter operate in order to be properly integrated and must have some sort of protocol in which they can communicate.

It must be able to establish and maintain data flow to a autonomous database system or federated database system. This allows new functionality developed to reference a common business format for each data element.

From this, guiding principles are formed that emphasize that code and functionalities are reused and interoperable, that they obey some sort of standards compliance and that they are properly identified, categorized, monitored and tracked.

When comparing mashups to system design patterns we can see how it is structured similar to the facade pattern.   Multiple packages are included in the facade object and the clients then access the functionalities of the packages through the facade object. Through this, it provides a simplified interface to a much bigger group of code which in this case is the different API’s and feeds.

When designing a mashup, there are two different styles in which one is composed of. In Web-based mashups, the user’s Web-browser is used to manipulate and combine the data. This is a more front end based approach. With Server-based mashups, all of the data manipulation and combining is done on a remote server side and then sent to the users Web browser using tools such as HTTPRequest. This approach isolates the front end from the user more so than Web based . Both styles have their own advantages based on how isolated we want the user to be from the data manipulation and also based on what level of security we wish to have on the data maniupulation

There are three layers that mashups are composed of:

1) Presentation or User Interaction Layer: This is the user interface of the mashup where technologies we have seen such as HTML, CSS, AJAX, and XML are used to make the mashup as visually appealing and functionally efficient as possible.
2) Web Services: Used to connect to the products functionality with API services. Tools such as XMLHTTPRequest and SOAP are used to do this.
3) Data: How the sending, storing and receiving of data is done often XML is used.

Organization within mashup architecture is essential in ensuring that it will deliver the necessary functionalities quickly and efficiently and using the appropriate protocols and design principles will enable mashups to be designed quicker.

Mashups - Pros and Cons

While mashups are a useful tool, they come with a series of advantages and disadvantages. An advantage to using mashups is that it allows for the reuse of existing applications. So, instead of spending time developing a component for your website, you can instead use a preexisting one to implement the feature that you want on your website. This also leads to another advantage, since you are saving time not having to implement certain features, you are able to more rapidly develop applications. This allows you to got through several more prototypes in a shorter amount of time then if you were to develop all the components yourself. Another advantage is that the development of a mashup does not require the user to have extensive IT or programming skills. Mashups are designed so that most of the work is abstracted behind the scenes so that the user simply only needs to learn how to implement it. This allows users who normally could not implement such features on their own, to be able to incorporate these features. All of these previous advantages leads to another one, which is that the cost of application development is reduced greatly. Because there are all these preexisting applications which are easy to develop, money will not need to be spent on long development schedules as well as the training required to teach developers to implement the features they wanted from scratch. With all these advantages however also comes several disadvantages.

A disadvantage of mashups is that a user has no control over the quality or the features in the application component they are implementing. So the user is dependent on the other developers to make sure that the application does what it intends to and has no bugs. This dependence on the other developers also does not guarantee that this application component will be continuously supported. So if the service or API gets discontinued, the user will have no choice but to replace it. Another disadvantage is the problem of scalability. There is no guarantee that the service you are implementing will be able to accommodate the traffic your website will get if it grows bigger. Another disadvantage is that the contents used in the mashup service are not guaranteed to be secure. So if you are a large enterprise or have sensitive data, incorporating the mashup API may pose a security concern. Finally, a disadvantage of mashups is that there are no standards for the development or application of a mashup, so this poses yet another difficulty in designing and implementing security mechanisms to ensure the integrity of the data being used.

Mashups - Examples

Mashups are interactive web applications that take content form different sources and present them in entirely new and unique manner.
The web is continuously growing and becoming more social and open. Because of this continuous growth, many websites have made there API publically available to the programmers that allow them to get information and build interactive applications.

A good example is google maps, which is popular interface to many mashup applications.
Google provide maps API which are used by programmer to access maps.
The developer can then combine these maps with some other data source and create something new and unique.
For example if you want to create a web applications to find nearest stores. A mashup can help you to use google maps in combination with other data source (i.e. yelp) and present the user with easy to use interface by displaying the location on the map.

Because of their uniqueness, interactive interface and functionality the mashup have become quite popular in recent years.

Before we talk about example let us introduce different genres of the mashup first.

Mashup genres

1. Mapping mashups

Mapping mashup is powerful way to visually display data set that contains location data. One reason for the rapid success of the mashup was the introduction of Google maps API.

2. Video and photo mashups

The emergence of the photo sharing and social network sites like flick and picasa with API have led to a lot of exciting and entertaining mashups. Since the photos have metadata associated with them (like location, user and date) the developer can use these metadata to mash it with some other information related to the metadata.

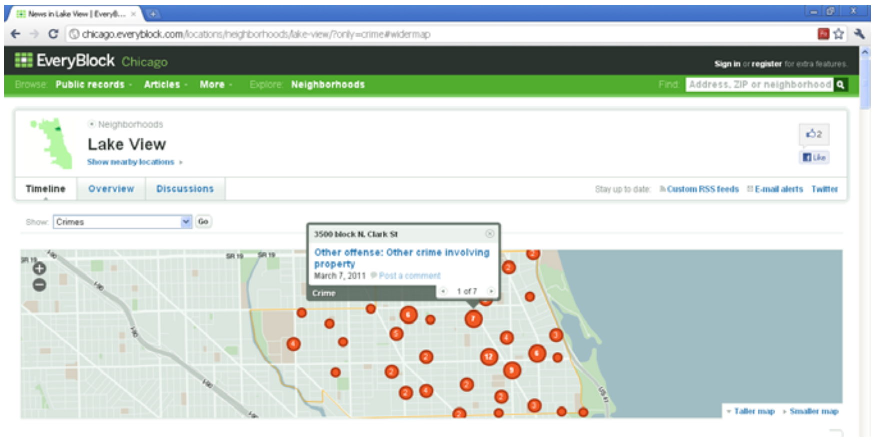
3. Search and Shopping mashups

A lot of ecommerce site like eBay and Amazon have provide API for accessing their contents.

Example of Mashup sites:

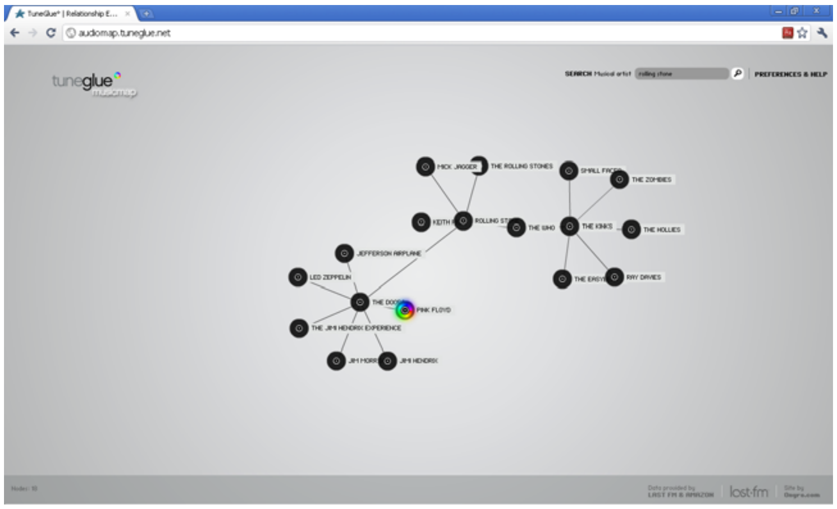
ChicagoCrime.org

The ChicagoCrime.org Web site is a great example of a mapping mashup, it mashes data from the Chicago Police Department’s online database with interface of google maps.
The site provide interactive interface that is both easy to use and visually powerful. The user can click on the pushpin displayed on the map and reveal the detail of the recent crime in the area.



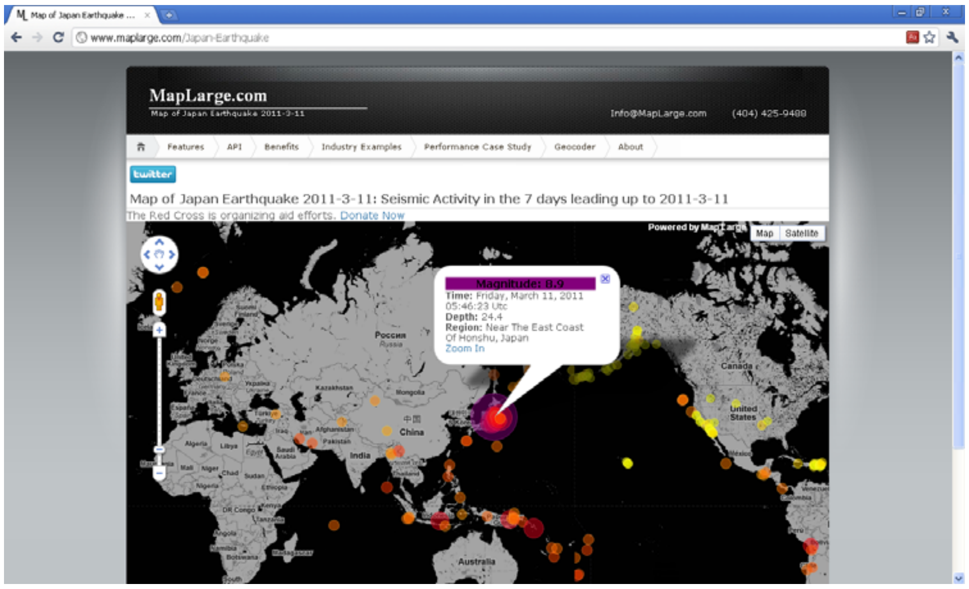
TuneGlue

Tuneglue is example of Search and Shopping mashup that uses data from Amazon e-commerce and last.fm. You can search music artist and view related artist and release by clicking on the node. The web application gets the data for related artist from last.fm and releases information from Amazon e-commerce if user likes the artist he can buy the album from Amazon. audiomap.tuneglue.net



MapLarge

The MapLarge uses google maps API and other data related to the location to display information related to location. For example it can display seismic activities related to the location on the map.
MapLarge.com


MapLarge.com
Map of Japan Earthquake 2011-3-11
Info@MapLarge.com (404) 425-9488
Features API Benefits Industry Examples Performance Case Study Geocoder About
Map of Japan Earthquake 2011-3-11: Seismic Activity in the 7 days leading up to 2011-3-11
The Red Cross is organizing aid efforts. Donate Now
Magnitude: 8.9
Time: Friday, March 11, 2011 05:46:23 Utc
Depth: 24.4
Region: Near The East Coast Of Honshu, Japan
Zoom In

Other Mashup website Example

http://www.langreiter.com/exec/yahoo-vs-google.html (google and yahoo)
http://www.maplarge.com/Japan-Earthquake (google maps and Maplarge)
http://sites.google.com/site/anthemsonmap/ (google maps and youtube)
http://www.programmableweb.com/mashups (Statistics on Mashup sites)

Resources

http://www.ibm.com/developerworks/xml/library/x-mashups.html
http://webtrends.about.com/od/webmashups/a/what-is-mashup.htm