Business with Mashups

Manish Jalan[*], Sunil Joglekar[#]
Patni Computer Systems Ltd, Unit 18, SDF-VII, SEEPZ, Mumbai 400096.
[*]*manish.jalan@patni.com*, [#]*sunil.joglekar@patni.com*

Abstract

Mashups have been attributed of being ad-hoc and toy-like in nature. They are often thought of as being unreliable, difficult to scale and maintain. Yet, they have been immensely popular. They have helped people to quickly create applications which are good enough for solving the problem at hand at a given point of time by combining existing pieces of information.

Mashups are different than conventional web applications. They are lot more appropriate for situational applications. We have found Mashups to be quite useful in rapid prototyping and very effective in proving the viability of ideas. This has been useful while discussing requirements with our customers. Mashups have also proved very helpful in aggregating information from public sources.

In this paper we present some of our experiences with Mashups, solution blueprint that we derived from our experience and how we used some of the patterns for Mashups.

1. Introduction

HousingMaps.com was the first Mashup application on the web. It used real estate listings from craigslist and mashed it up with Google maps. One could now choose a location and see all the listings in and around that location using maps as the primary interface. What we have to understand here is that housingmaps.com has consolidated two seemingly isolated applications and come out with something so useful.

Sandwiches can be looked upon as Mashup of bread and vegetables. Do you need to be a chef to prepare the sandwich? Certainly not! Sandwiches can be made by almost all of us. One need not be a professional to create the Mashups. They are generally developed by users themselves and are ad hoc and easy to change. Mashups involve integration of disparate data in a meaningful way to produce something new and interesting that addresses a personal or an immediate business need. They are created by the user or business analyst in relatively short periods of time with little or no help from IT.

For years we have been getting data from other systems and combining data with the existing systems to produce a result. So why is all the hype about Mashups only now? There has been significant rise in end-users producing applications themselves. Today we have modern-day Mashup tools available and users no longer have to devise complex transformation and loading of data through scripts and other mechanisms. So yes, Mashups have been around for years. However, the concept of the end-user being able to easily ‘drag-drop’ and put together a application within minutes hasn’t been there earlier - and this is what it is all about.

Mashups vis-à-vis traditional web applications

Typical Web Applications are well thought out, the number of users who benefit is large, or application is mission-critical, mission-critical applications cannot rely on public data sources due to concerns about availability and data validity. Formal development and deployment models are used and generally, people with development and deployment skills are NOT the same as those with knowledge about the business domain. While Mashups generally address an immediate need, tend to be short-lived, join seemingly unrelated data in meaningful ways and have little or no formal agreements with the data providers.

Situational Applications - Mashups meeting the need of the hour

Organizations focus on their core business. The size of the IT department is generally a fraction of the organization’s overall strength supporting their business. IT might form the backbone for many organizations, it might enable their operations. Yet, IT department will take the role of a support function just like the HR or the Admin departments. IT department will be focusing on supporting the applications which are critical to the business and people of the organization at large.

There will be numerous pockets in the organization where people will have some custom IT requirement; may be in terms of reports or some data or some applications. With their limited capacity IT departments will face difficulty in fulfilling every such requirement. In fact, unless the requirements are business critical, most of them will be turned down. That's a huge gap of the Long Tail of situational application needs that is left unmet in most organizations. This has been depicted in the figure 1 below. Situational Applications address needs of users at the edges of the organization.

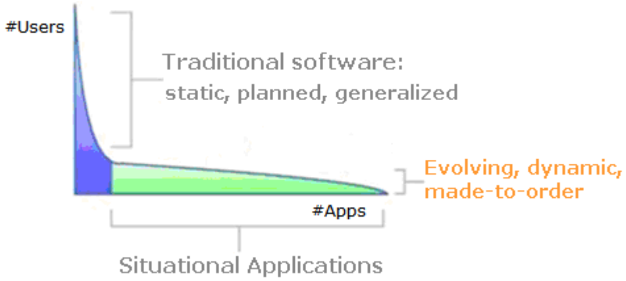

Figure 1: Situational Applications address needs of users at the edges of the organization

There are number of occasions when businesses need some quick solution to meet their situational requirements. They cannot wait for long IT development and budget cycles. They need immediate solutions to their business challenges. As a case in example, significant number of business collaborations tend to last less than 12 months. Average application integration work takes three to six months. There is little room for proving return on investment. Industry needs solutions that can be built rapidly & cost effectively to solve immediate & specific business problems. They need solutions that can be considered disposable when the market moves on.

Business users are always looking for more control in addressing their needs and also want to reduce their dependence on IT. On the other hand IT is always under pressure to deliver quickly and minimize the costs.

Mashups bridge the gap. They reduce the cost, the turnaround time and need less expensive skill sets. This has been summarized in the figure 2 below. Mashup has little emphasis on reliability, scalability, maintainability, and availability. Product-level features, performance, quality, traits are not expected and they are often short lived. They are usually tactical rather than strategic and are usually information-centric

2. Using Mashups for proving the viability of a solution

The problem

One of our customers was collecting web analytics data and in the process built a rich information repository. Lot of collective intelligence could be mined from this rich source and put to effective use in various contexts especially to provide information to users based on their roles and preferences. For example,

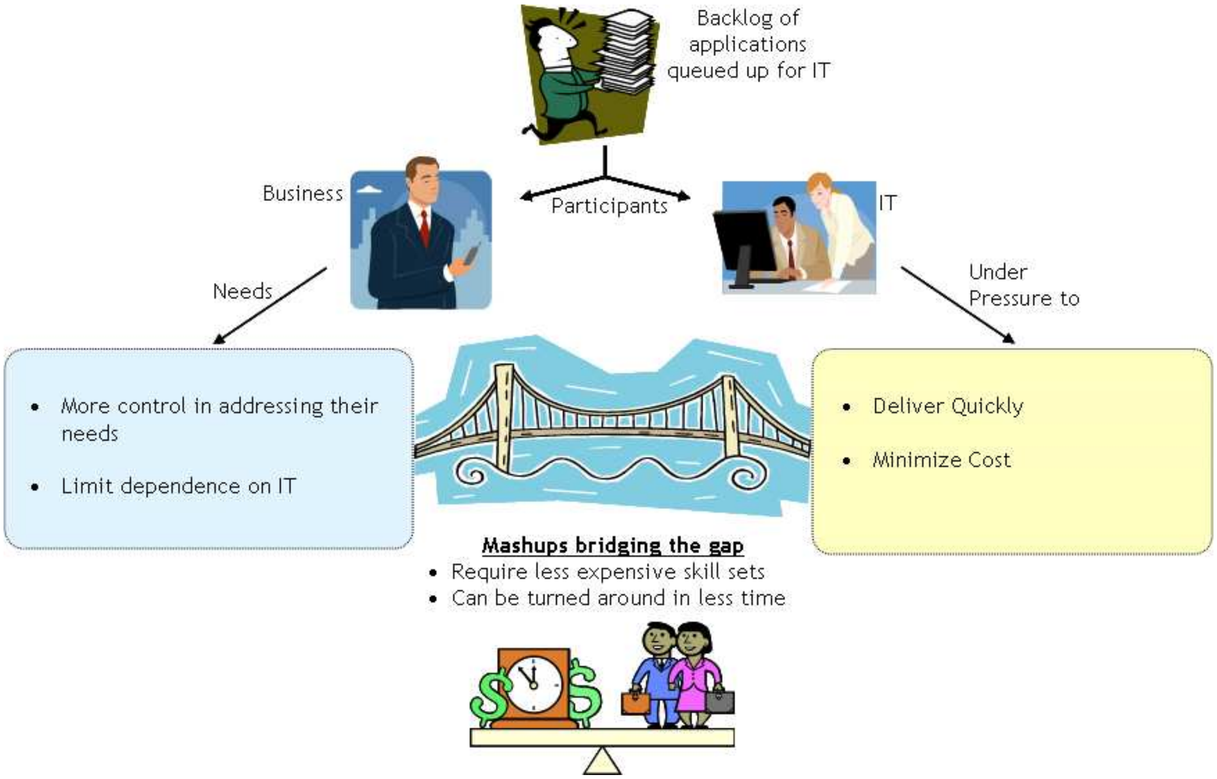
Backlog of applications queued up for IT
Business
Participants
IT
Needs
Under Pressure to
More control in addressing their needs
Limit dependence on IT
Deliver Quickly
Minimize Cost
Mashups bridging the gap
Require less expensive skill sets
Can be turned around in less time

Figure 2: Mashups bridge the gap between Business and IT

- Brand Manager would like to see information pertaining to their brands or franchises and their performance as compared to similar products manufactured by them
- Campaign managers would like to know more about their campaigns
- Technical people would like inputs about internet browsers visitors are using
- Regional Managers would like to see details about their regions

The information repository built through analytics could be used extensively to capture metrics and also for sending alerts for deviations from the goal or KPIs. However there were some gaps.

Gap 1: There were hundreds of parameters that are tracked in web analytics. In their raw form they made little sense to any of the stake holders. Users wanted to see these organized and mapped onto relevant categories and KPI/goals.

Gap 2: At the organization level, understanding what information from web analytics data will be really useful to each individual user on the field was difficult. Here came the gap between what the organization felt was useful for its users and what the users actually wanted. The reporting feature of the analytics system was only useful to establish an initial structure of top level parameters based on user roles and function. In addition to some top level categorization and KPI mapping, users needed granular control into what specific data points they could monitor.

The need: An ecosystem was needed that provided granular data points and tools for the users to build their own intelligence machinery.

We developed Mashups based solution and got the buy in from the customer for following reasons.

- Through the reporting function of the analytics system, the customer was able to serve the core functions. What was needed was empowering the users at the edges which no organization can identify accurately
- Check viability of this opportunity. It would make sense to invest in setting up formal intelligence machinery only if the users actually used them. You don't need to build super bikes if your users are going to continue walking. This went very well with the customer, especially due to the current economic situation where every penny was to be spent with caution.
- Mashups based solution could be built to be just good enough without taking too much,
  - Time => quicker time to value
  - money or efforts => solution could be disposed off if value if not realized

The Mashups solution

1. Making the web analytics system mashable: This involved the following.
   a. Identify the set of data points which users might need for building their dashboards
   b. Tag the data points according to anticipated usage so that the users could quickly search for them based on keywords
   c. Exposing RESTful APIs for the data points for enabling users to consume them in their Mashup application for creating their customized dashboards
2. Creating the application that will enable users to create their own Mashup applications from the data points captured in the system. This involved the following.
   a. A quick web application was developed using Ruby on Rails framework
   b. The application allowed the user to choose a set of parameters from the set of data points exposed at previous level. The user could also search and choose for parameters / data points based on tags
   c. For each data point of interest, the user could specify the monitoring interval.
   d. At a primitive level, the application allowed the user to combine data point values and specify thresholds for raising the alerts
   e. Per user choices were captured in the application's database
   f. The application would provide the customized feed URL for the user to subscribe for updates
3. An RSS application would generate customized RSS feeds for the individual users based on their choices captured in the application. Users would thus get to see a customized feed of alerts at a granularity and frequency specified by them. Thus putting them in control.

Learning: When a Mashup becomes popular and key to the business, the corporate IT needs to consider whether to invest in turning the Mashup, or at least elements of it, into a more robust performing application

3. Using Mashups for assisting employees with local information in new locations

The Problem

The customer's business required their staff to migrate to various cities. While it was not mandatory for the customer's business to ensure that their employees settle well in new locations, it was certainly a humanitarian responsibility on their behalf to make their employees comfortable in the new environment. This included helping them know the place, the options for commuting, communication, lodging, dining, education… Traditionally the customer used to prepare detailed handbook for each location. Every 2-3 years the customer would have to update the handbook to keep it relevant in the current times. Providing updated local information at a given point of time was bit of a challenge. Especially because their operations were spread across multiple cities. The customer was looking for some feasible solution to address this problem.

The Mashups solution

The customer needed a solution where they could get local updates and enable interaction between their people. A combination of wiki, Mashups and discussion board formed the solution that the customer was looking for. Wiki became the knowledge repository which evolved with inputs from the users over the period of time. The information on the wiki was useful to the users themselves. This fact motivated them to update the wiki with their experiences in turn. Crowd sourcing was seen at work. Discussion boards enabled employees / users to ask and answer questions and also advertise. This was especially useful for posting information on buying / renting of accommodation. Discussion board brought the buyers and sellers in one place. Besides it became a great information seeking place for eateries, etc.

Mashups filled the gap of providing updated local information. There were plenty of public sources which provided quite recent information about the locality. The task was left to identifying such fairly reliable sources for each location. For many places, Wikipedia provided good background information. For some locations, EveryBlock was useful. Some locations had their municipal offices provide official information through their websites. Mashups were created for each location which would aggregate inputs from multiple sources. Yahoo Pipes was very effective in aggregating the information. Scrapping, cleaning, aggregating, filtering was all performed using Yahoo Pipes. For some locations the information was also plotted on Google Maps by exposing the information in KML through Yahoo Pipes and then consuming the KML feed in a RESTful interface provided by Google Maps.

Every location required a separate Mashup solution as every location had to source information from different set of sources. Each location's coordinator could own the Mashup application as learning to build and maintain Mashups was much easier than facing a continuous stream of queries from the migrant employees on an ongoing basis.

4. Deriving a Solution Blueprint from our experience with Mashups

The toolset available for building Mashup solutions would generally address one of the Mashup needs, viz. Data sourcing, Data Transformation and Data Presentation. There are number of tools available in the market place for each of these needs. In most Mashup solutions we will see some combination. Based on our case studies, we arrived at the solution blueprint for solutions based on Mashups as shown in figure 3 below. The solutions blueprint is useful, as knowing the combination helps us choose the right set of tools. The problem context, the business domain will suggest the required combination to build the Mashups.

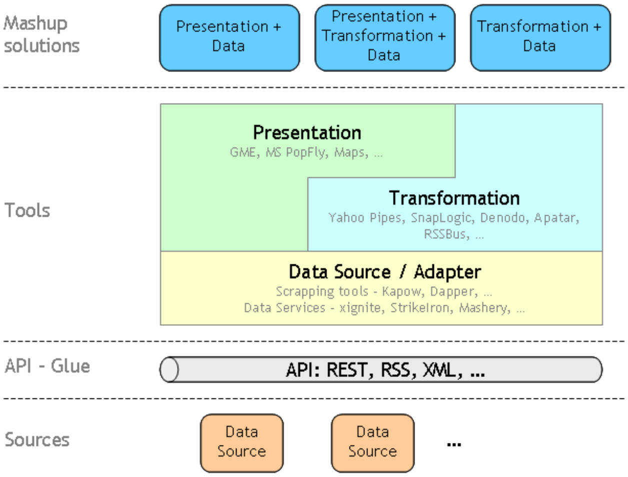
Mashup solutions
Presentation + Data
Presentation + Transformation + Data
Transformation + Data
Tools
Presentation
GME, MS PopFly, Maps, ...
Transformation
Yahoo Pipes, SnapLogic, Denodo, Apatar, RSSBus, ...
Data Source / Adapter
Scrapping tools - Kapow, Dapper, ...
Data Services - xignite, StrikeIron, Mashery, ...
API - Glue
API: REST, RSS, XML, ...
Sources
Data Source
Data Source
...

Figure 3: Solution Blueprint for Mashups

In the figure above, we have presented how Mashups solutions will be generally built. There will a string of data sources which will feed the Mashup. These data sources will expose certain API which could be REST, SOAP, or could be plain HTML i.e. without an API really. In all cases the data sources will have to be cleaned and brought in a uniform format before they could use for further aggregation. Whether it is for scrapping, cleansing or for unifying the formats, the data sources will be consumed via some source adapters. Some Mashups might need transformation and some may need simple presentation massage. Some might need both. Depending on what is needed in addition to the source adaptor the Mashup solution will usually comprise of

- Presentation + Data
  E.g. Google Reader
- Transformation + Data
  E.g. Yahoo Pipes + Google Reader
- Presentation + Transformation + Data
  E.g. HousingMaps.com (Google Maps + Craiglists)

5. Mashup Patterns seen at work

Patterns are a general form that helps us think about the structure of the solution. People have documented various Patterns seen in Mashup solutions [8, 9, 10]. The patterns were derived by examining how the Mashups were put to use: on existing websites, improved upon them, how data from multiple websites were combined, and what kinds of user tasks these Mashups might be suitable for. In our experience with Mashups we observed following patterns.

- Aggregation – This is a common feature of Mashups where multiple feeds / websites are aggregated.
  - Seen in Web analytics (combination multiple data points to form the dashboards feed) as well as Locals update Mashup case studies (combination of locals information from multiple sources)
- Personalization - Number of Mashups allow users to personalize the solutions based on their preferences or their personal information
  - Seen in Web analytics case study (users personalizing their dashboard feed data points, their granularity and frequency)
- Focused view of Data - This pattern is where the Mashup solutions exist to index or categorize a subset of another website's entire contents
  - Seen in Locals update Mashup case study (Information specific to the location getting collated)
- Real time monitoring - Not in the systems sense, but rather the Mashups that make a changing data set the focus of the design and are intended to make the user aware of changes
  - Seen in Web analytics (updated dashboard feeds) as well as Locals update Mashup case studies (Updated locals information)

6. Conclusion

Mashups are the means for creating ad hoc applications quickly without requiring too much of technical know-how. The Mashup solutions are ad-hoc. Yet, they are valuable for the business. We can deliver value quickly by applying Mashups.

One of the major challenges we face regularly is: to minimize time to market (and time to value!) for the conventional enterprise applications. Mashups help us tackle this problem successfully through the "situational applications". The reasons are simple: Mashups are driven and developed by the users themselves. Mashups need minimal investment in terms of skill, effort, and time thus making them highly indispensible for business users.

The solution blueprint that we derived from our Mashups work and the patterns that we observed will be useful in for our future work. This will help us identify and choose the right combination of tools for building the solution. Therefore, we strongly believe in using Mashups for ourselves and for developing solutions for our clients.

7. References

[1] Dion Hinchcliffe, 'Mashups: The next major new software development model?' Enterprise Web 2.0, URL: http://blogs.zdnet.com/Hinchcliffe/?p=106

[2] Charlton Barreto, 'Web 20-20: Architecture for the New Internet', *TPAC 2007*, URL: http://charltonb.typepad.com/talks/120407-cbb-web2020/Web2020TheNewInternet.pdf

[3] Jonathan Marsh, 'Mashup: Noun or Verb?', URL: http://wso2.org/repos/wso2/people/jonathan/Mashup%20Noun%20or%20Verb.pdf

[4] John Crupi and Chris Warner, 'Enterprise Mashups Part I: Bringing SOA to the People', SOA Magazine Issue XVIII: May 2008, URL: http://www.soamag.com/I18/0508-1.pdf

[5] Anthony Bradley, 'Reference Architecture for Enterprise 'Mashups'', Gartner Research, Sep 2007.

[6] Anthony Bradley, 'A Framework for Key Integration Patterns and Application Types for Enterprise Mashups', Gartner Research, Jan 2009.

[7] Gregor Hohpe, ‘Mashing Up with Google Mashup Editor and Yahoo! Pipes’, Colorado Software Summit, Oct 2007.

[8] Jeffrey Wong and Jason Hong, ‘What Do We "Mashup" When We Make Mashups?’, *Proceedings of the 4th international workshop on End-user software engineering,* International Conference on Software Engineering Pages 35-39, URL: http://mashup.pubs.dbs.uni-leipzig.de/files/Wong2008Whatdowemashupwhenwemakemashups.pdf, 2008.

[9] Holt Adams, ‘Mashup business scenarios and patterns’, IBM Developer Works, URL: http://www.ibm.com/developerworks/lotus/library/mashups-patterns-pt1/

[10] Michael Ogrinz, ‘Mashup Patterns: Designs and Examples for the Modern Enterprise’, Addison-Wesley Professional, March 18, 2009.

[11] Yahoo Pipes, http://pipes.yahoo.com/

8. About the Authors

Manish Jalan is a Sr. Specialist (Software) with the Product and Technology Initiative group at Patni Computer Systems Ltd. He has 8 years experience as a software professional working in global software teams. Manish specializes in emerging technology adoption, Web 2.0, Enterprise mobility, VoIP, IP telephony, Speech recognition, IVRS, Call center systems, Push To Talk. His achievements include protocol stack implementation, whitepapers and contribution to open source. His areas of interest are Mashups, Enterprise Mobility, Cloud computing et al.

Sunil Joglekar is a Sr. Technical Architect with the Product and Technology Initiative group, at Patni Computer Systems Ltd. He has 18 years experience as a Software Professional working in Global Software Teams. He has spent about 5 years at Sun Microsystems as a consultant. Sunil has extensive experience in the development of products / product suites / product lines and applying Agile Methods / Practices. He specializes in architecting testable solutions for complex enterprise software systems and their integrations such as EAI / B2Bi / SOA. His primary areas of interest are strategizing & architecting innovative solutions with Web 2.0, Cloud Computing, Autonomic Computing et al.