Enterprise Information Mashups: Integrating Information, *Simply*

Anant Jhingran
CTO, Information Management
IBM Silicon Valley Laboratory
San Jose, CA

ABSTRACT

There is a fundamental transformation that is taking place on the web around information composition through *mashups.* We first describe this transformation and then assert that this will also affect enterprise architectures. Currently the state-of-the-art in enterprises around information composition is federation and other integration technologies. These scale well, and are well worth the upfront investment for enterprise class, long-lived applications. However, there are many information composition tasks that are not currently well served by these architectures. The needs of *Situational Applications* (i.e. applications that come together for solving some immediate business problems) are one such set of tasks. Augmenting structured data with unstructured information is another such task. Our hypothesis is that a new class of integration technologies will emerge to serve these tasks, and we call it an *enterprise information mashup fabric*. In the talk, we discuss the information management primitives that are needed for this fabric, the various options that exist for implementation, and pose several, currently unanswered, research questions.

1.INTRODUCTION

ProgrammableWeb.com is a compendium of the current state of mashups on the internet. A quintessential example of a mashup is *HousingMaps.com,* which displays the available houses in an area by combining a listing from *Craigslist.com* with a display map from *Google*. In effect, *HousingMaps.com* is a "join" of two web sites (using a common surrogate key, the "address"), but the join is often implicit (such as "display overlay"). These joins often happen on semantically known dimensions – location, time, keyword, UPC/ISBN primary keys etc. *Programmableweb.com* shows many other mechanisms exist for such combination (*union, implicit searching semantics, etc.).* They often happen alo

We would like to make a few other observations to motivate our work:

- The mashup applications (written, for example in PHP) resemble the same hodgepodge of data and application logic that the traditional enterprise applications of 1970's and 1980's did. Is there a way to build an *information mashup* layer that will allow a clean separation for these applications? This will help us build situational applications in the enterprise more quickly, complementing the more robust and scalable integration technologies that the enterprises have been investing in.

- There is an emerging trend called *situational applications* where applications are constructed "on the fly" for some transient need. Eventually, such an application either outlives its usefulness, or migrates to the IT department. These applications need data (such as presentations, emails, spreadsheets etc.) that are *either* not covered by traditional Enterprise Information Integration (EII) architectures, *or* it will take unacceptably long for the IT department to provide them. Our *information mashup* layer serves this need.

- Furthermore, because the information that is the basis for these applications is more personalized, it is often more incomplete and inaccurate compared to IT supported data. Therefore *information augmentation* through simple means is critical. For example, I might have a spreadsheet that lists only the first names of my employees. I understand what that column means – but how does one work with it to join it with some other information? In this case, the constructor of the application can easily specify the semantics of some attributes, and thus say that the full name can be obtained by joining with the IT provided employee directory on first names. We believe that such application specified augmentation is generally possible in these class of tasks.

- A critical task for mashups is *information standardization* – such as geocoding. There are many such services available on the web and sometimes the IT department provides these.

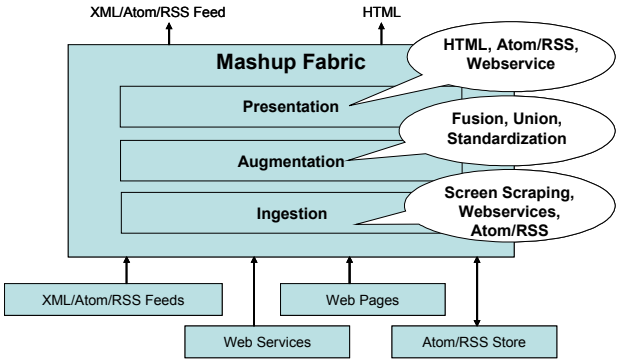
- Many enterprise information tasks benefit from external third party data sources. A great example of this would be the combination of machine-based tagging of documents (using techniques such as those available through IBM's Unstructured Information Management Architecture, or UIMA, (http://www.research.ibm.com/UIMA)) with external, social networking such as *del.icio.us*.

- While data providers have long provided SOAP and other WebServices interfaces to access them, a lot of the mashups are happening around simpler paradigms of REST (http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm) and RSS and ATOM feeds. A central thesis of ours is that the mashup fabric is best built around these emerging primitives.

VLDB '06, September 12–15, 2006, Seoul, Korea.

2.INFORMATION MASHUP FABRIC

The following figure represents our fabric's structure:



The mashup fabric consists of a framework that augments and combines information from various sources using various services to achieve the combination. There are similarities to ETL, but it is more light-weight and relies on a services model that is easily extensible via Web 2.0 techniques (specifically REST). The fabric has some built in services for standardization and augmentation (union, fuse/join, select, transform – all with intuitive semantics) and is extensible through web services.

While many of these primitives can be expressed in ETL, SQL or stream processing terms, the syntax is not suited for the tasks at hand. In fact, in our prorotype, we end up compiling our mashup language to an XQuery like syntax and execute it against our DB2 XQuery and federation engine; however, the end user paradigm is very different.

3.STRUCTURED + UNSTRUCTURED DATA

Enterprise Information architectures have so far not delivered the value of delivering *all* information (structured and unstructured) in a seamless way. Two approaches are emerging. The first extends *business intelligence* architectures with structured information and dimensions derived from unstructured data. The second extends *content and search techniques* by indexing all data in something like a text index. Each approach has its advantages and disadvantages and in the talk we discuss these in details. In the former, the ragged nature of dimensions and the uncertainty of implicit information from unstructured data gets in the way of traditional warehousing architectures. The latter has problems accessing enterprise applications with strict access controls, and beyond equality predicates, more sophisticated slicing and dicing is generally not possible. We assert that for a class of applications that need both data, *enterprise mashups* offer a third alternative. Let us give two examples.

A salesperson needs to make a call on a prospect. She would like to get a dossier built on the client that contains the following information: (i) How much did we sell to the customer in the last 5 quarters? (ii) What, if any, problems they have been having with our stuff? And (iii) Some personal information about their CIO so that she does not come across as overly formal. There are no explicit joins here – it is a matter of information assembly.

Consider a second example. A CFO has to meet his CEO the next day, and he knows that every one of his finance person has sent him a presentation containing their organization's financial picture in a spreadsheet, *but with a twist,* the spreadsheet is embedded in the presentation (that is the way meetings take place in large organizations!). So he has to extract the relevant mail messages, the presentations from those mail messages, the spreadsheets in them, the financial information in those spreadsheets, and then summarize it for his CEO.

4.RESEARCH TOPICS

We list a few here, doubtlessly there will be many more:

1. What are the standardization functions? Some will be domain specific, some will be generic.
2. What is the right join operator given that many of the attributes to join on are fuzzy, or happen on incomplete information? What can be reasoned about the output given this?
3. What is the right cost-based service selection in the case of use of external services?
4. Which style of structured/unstructured integration is best under what circumstances?
5. What part of the fabric can be pushed into database, XQuery, content and search primitives?
6. How does one detect enough structure in spreadsheets, presentations and documents to permit information extraction?
7. What is the most efficient way of building a persistence mechanism around RSS/Atom feeds?
8. What do REST/Atom interfaces on top of "databases" mean?
9. What is a robust way for expressing credentials for authentication and access control?

5.CONCLUSIONS

We have demonstrated the need to think of an information mashup fabric that builds up information primitives for situational applications, enabling light-weight information and service integration. The fabric does not replace traditional information architectures, but is likely to be used when the enterprise information architectures either do not cover the information that is needed, or when the information tasks that need to be done are not well supported by those architectures. We also established a philosophy that such a mashup fabric should be built around simple paradigms, and we have chosen RSS/Atom as the paradigm of choice, though of course, other choices are possible. We also showed that one of the most important information management task – combining structured and unstructured data – is a great candidate for this fabric.

6.ACKNOWLEDGMENTS

I would like to thank Volker Markl, Kevin Beyer, Mehmet Altinel, Florian Leybold, Susan Malaika, Hamid Pirahesh, Berthold Reinwald, David Simmen, Shivakumar Vaithyanathan, Laura Haas, Carol Jones and many other IBM colleagues.