Data Enrichment using Data Sources on the Web

Karthik Gomadam, Peter Z. Yeh, Kunal Verma
Accenture Technology Labs
50, W. San Fernando St,
San Jose, CA
{ karthik.gomadam, peter.z.yeh, k.verma} @accenture.com

INTRODUCTION

As enterprises become more data and analytics driven, many businesses are seeking to enrich their internal data records with data from data sources available on the Web. Consider the example, where the consumer database of a company might have the name and address of its consumers. Being able to use publicly available data sources such as LinkedIn, White Pages and Facebook to find information such as employment details and interests, can help the company collect additional features for tasks such as customer segmentation. Currently, this is done in a static fashion where a business buys data from one or two sources and statically integrates with their internal data. However, this approach has the following shortcomings:

1. **Inconsistencies in input data**: It is not uncommon to have different set of missing attributes across records in the input data. For example, for some consumers, the street address information might be missing and for others, information about the city might be missing. To address this issue, one must be able to select the data sources at a granular level, based on the input data that is available and the data sources that can be used.

2. **Quality of a data source may vary, depending on the input data**: Calling a data source with some missing values (even though they may not be mandatory), can result in poor quality results. In such a scenario, one must be able to find the missing values, before calling the source or use an alternative.

In addition to these, current approaches also suffer from software limitations. These include

1. A conventional client server approach to software development. Very few of the enterprise data tools are available as Web based solutions, necessitating complex installation and maintenance overheads. Further, in the scenario of using Web based data, where data providers give access keys and restrict data limits, the current approach also adds complexity in key management and ensuring that data limits are not exceeded.

2. Emphasis on coding as opposed to configuration for customization: Tradiational systems often require additional implementation for customization. In environments such as ours (a consulting firm), this often leads to significant additional man hours to stand up, deploy, and maintain the system.

3. Library driven approach: Often times, developer write custom libraries to access data sources (such as LinkedIn, Twitter) and these are used as binary libraries (JARs and DLLs). Given the hetereogenous platforms and architectures that are used in an enterprise, reuse of these libraries poses a significant challenge.

Add one line about how we are addressing. Shorten the shortcomings. :w

In this paper, we present an technical overview of the design and development methodology of the data enrichment framework which attempts to automate many sub-tasks of data enrichment[1]. In this paper we will articulate our design methodology and discuss how our framework helps overcome the above mentioned limitations. We leverage existing Web development standards such as REST (for communication) and Comet (for streaming updates), existing frameworks in Django (for Web application development) and Celery (for task distribution).

```
var http = new XMLHttpRequest();
var url = (URL not displayed)
var params = "name=!Contact.Name&
        city=!Contact.MailingCity&
        state=!Contact.MailingState";
http.open("POST", url, true);
var update_Contact = new sforce.SObject("Contact");
update_Contact.Id = "!Contact.Id" ;
var edatap= JSON.parse(http.responseText).data;
```

MOTIVATION

We motivate the need for data enrichment through a real-world example gathered from a large Fortune 500 company. We note here that the described system is currently being testing with the client.

Creating comprehensive customer models: Creating comprehensive customer models has become a holy grail in the consumer business, especially in verticals like retail and healthcare. The information that is collected impacts key decisions like inventory management, promotions and rewards, and for delivering a more personal experience. Often businesses engage customers to register for programs like reward cards or connect with the customers using social media. This limited initial engagement gives them the

[1]We request the reader to visit http://, for a detailed technical report on the enrichment algorithm

access to basic information about a customer, such as name, email, address, and social media handles. However, in a vast majority of the cases, such information is incomplete and the gaps are not uniform. For example, for a customer John Doe, a business might have the name, street address, and a phone number, whereas for Jane Doe, the available information will be name, email, and a Twitter handle. Leveraging the basic information and completing the gaps, also called as creating a 360 degree customer view has many applications, including:

1. **Personalized and targeted promotions**: The more information a business has about its customer, the better it can personalize deals and promotions. Further, business could also use aggregated information (such as interests of people living in a neighborhood) to manage inventory and run local deals and promotions.

2. **Better segmentation and analytics**: The provider may need more information about their customers, beyond what they have in the profile, for better segmentation and analytics. For example, a certain e-commerce site may know a personâĂŹs browsing and buying history on their site and have some limited information about the person such as their address and credit card information. However, understanding their professional activities and hobbies may help them get more features for customer segmentation that they can use for suggestions or promotions.

3. **Fraud detection**: The provider may need more information about their customers for detecting fraud. Providers typically create detailed customer profiles to predict their behaviors and detect anomalies. Having demographic and other attributes such as interests and hobbies helps building more accurate customer behavior profiles. Most e-commerce providers are typically under a lot of pressure to detect fraudulent activities on their site as early as possible, so that they can limit their exposure to lawsuits, compliance laws or even loss of reputation.

Current approaches to addressing this challenge largely revolve around subscribing to data sources like Experian. This approach has the following shortcomings:

1. The enrichment task is restricted to the attributes provided by the one or two data sources that they buy from. If they need some other attributes about the customers, it is hard to get them.

2. The selected data sources may have high quality information about attributes, but poor quality about some others. Even if the e-commerce provider knows about other sources, which have those attributes, it is hard to manually integrate more sources.

3. There is no good way to monitor if there is any degradation in the quality of data sources.

Using the enrichment framework in this context would allow the e-commerce provider to dynamically select the best set of sources for a particular attribute in particular data enrichment task. The proposed framework can switch sources across customer records, if the most preferred source does not have information about some attributes for a particular record. For low confidence values, the proposed system uses reconciliation across sources to increase the confidence of the value. The enrichment framework can also continuously monitor and downgrade sources, if there is any loss of quality.

SYSTEM ARCHITECTURE

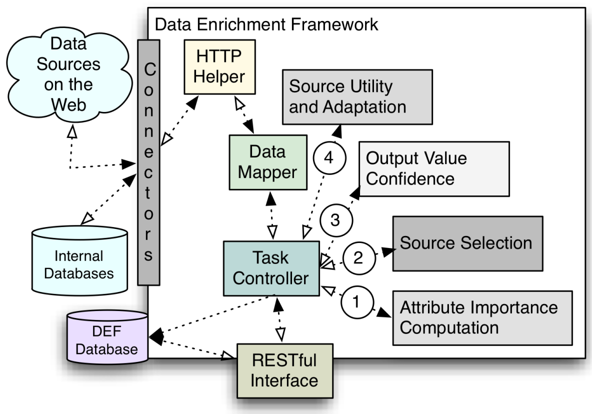


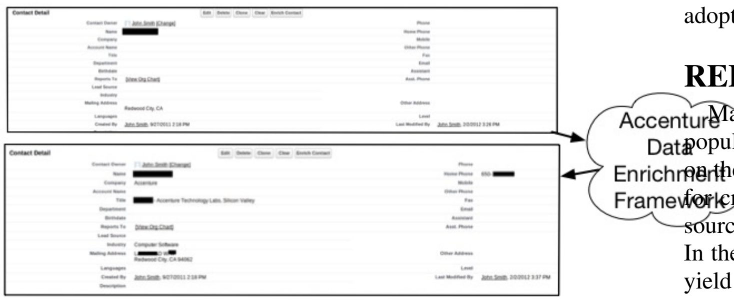
Figure 1: Overview of Enrichment Algorithm

Figure 1 illustrates the overview of the enrichment algorithm[2]. The task manager starts a new enrichment project by instantiates and executes the enrichment engine. The enrichment engine uses the attribute computation module to calculate the attribute relevance. The relevance scores are used in source selection. Using the HTTP helper module, the engine then invokes the connector for the selected data source. A connector is a proxy that communicates with the actual data source and is a RESTful Web service in itself. The enrichment framework requires every data source to have a connector and that each connector have two operations: 1) a return_schema GET operation that returns the input and output schema, and 2) a get_data POST operation that takes as input the input for the data source as POST parameters and returns the response as a JSON. For internal databases, we have special connectors that wrap queries as RESTful end points. Once a response is obtained from the connector, the enrichment engine computes the output value confidence, applies the necessary mapping rules, and integrates the response with the existing data object. In addition to this, the source utility is also computed. The mapping, invocation, confidence and source utility value computation steps are repeated until either all values for all attributes are computed or if all sources have been invoked. The result is then written into the enrichment database.

In designing the enrichment framework, we have adopted a service oriented approach, with the goal of exposing the enrichment framework as a “platform as a service”. The core tasks in the framework are exposed as RESTful end points. These include end points for creating data objects, importing datasets, adding data sources, and for starting an enrichment task. When the “start enrichment task” task resource is invoked and a task is successfully started, the framework responds with a JSON that has the enrichment task identifier. This identifier can then be used to GET the enriched data from the database. The framework supports both batch GET and streaming GET using the comet [5] pattern.

Data mapping is one of the key challenges in any data integration system. While extensive research literature exists for automated and semi-automated approaches for mapping and matching [4, 2] it is our observation that these techniques do not guarantee the high-level of accuracy required in enterprise solutions. So, we currently adopt a manual approach, aided by a graphical interface for data

[2]We do not describe the details in this paper. Please visit http:// to get additional information



Contact Detail

Contact Owner	John Smith [Change]	Phone	
Name	[redacted]	Home Phone	
Company		Mobile	
Account Name		Other Phone	
Title		Fax	
Department		Email	
Birthdate		Assistant	
Reports To	[View Org Chart]	Asst. Phone	
Lead Source			
Industry			
Mailing Address	Redwood City, CA	Other Address	
Languages		Level	
Created By	John Smith, 9/27/2011 2:18 PM	Last Modified By	John Smith, 2/2/2012 3:26 PM

Contact Detail

Contact Owner	John Smith [Change]	Phone	
Name	[redacted]	Home Phone	650-[redacted]
Company	Accenture	Mobile	
Account Name		Other Phone	
Title	[redacted], Accenture Technology Labs, Silicon Valley	Fax	
Department		Email	
Birthdate		Assistant	
Reports To	[View Org Chart]	Asst. Phone	
Lead Source			
Industry	Computer Software		
Mailing Address	[redacted] Redwood City, CA 94062	Other Address	
Languages		Level	
Created By	John Smith, 9/27/2011 2:18 PM	Last Modified By	John Smith, 2/2/2012 3:37 PM
Description			

Figure 2: Enrichment as a PAAS from Sales Force

mapping. The source and the target schemas are shown to the users as two trees, one to the left and one to the right. Users can select the attributes from the source schema and draw a line between them and attributes of the target schema. Currently, our mapping system supports assignment, merge, split, numerical operations, and unit conversions. When the user saves the mappings, the maps are stored as mapping rules. Each mapping rule is represented as a tuple containing the source attributes, target attributes, mapping operations, and conditions. Conditions include merge and split delimiters and conversion factors.

CHALLENGES AND EXPERIENCES

Data sources (exposed as RESTful services and/or Web APIs) are an integral part of the data enrichment framework. Even though data sources have been available for many years, their traction within the enterprise has been minimal. During the development of the enrichment framework, we got a few insights into the reasons for this lack of adoption. Rate limiting (especially in the number of calls that can be made within a short interval of time) and lack of syndication processes, make it hard to reliably use a data source, especially in client deployments. Further, many of the data sources do not offer explicit SLA driven contracts, thus making them unattractive. Poor API documentation is another reason. Often times, when a developer is using an API, they find that some of the capabilities that are present in the UI driven version of the service (such as LinkedIn API vs LinkedIn.com) are either absent or are not sufficiently documented. This makes it harder to understand the actual capabilities of an API. Further, the data formats for the input and output are often described in text (as opposed to having a JSON / XML snippet), adding to the complexity. API providers do not "push" API changes to the developer. In most cases, developers find out about the change after a failed call to the API. We feel that the above mentioned reasons play a significant role in impeding the adoption of data sources APIs in enterprise software development.

RELATED WORK

Mashups or Web application hybrids [7] have emerged as the popular approach for integrating data from different data sources on the Web. Plethora of tools such as [6, 3, 8] have been developed for creating mashups. However, in a traditional mashup, the data sources used and the order in which they are invoked are static. In the context of data enrichment, such a static approach may not yield desired results, since the gaps in a dataset are often not homogeneous. The granular selection and ordering of sources that we advocate in this paper is the main difference between our work and the standard mashup techniques. An evaluation of approaches for data mediation in mashup tools is presented in [1]. While this paper compares different approaches, it does not in itself advocate a novel method. [6] present a domain specific language for developing mashups. The paper proposes a manual mediation approach, similar to the approach discussed in this work. However, the DSL driven platform does not support run-time source selection based on available attributes and does not support source adaptation.

CONCLUSION

In this paper we present a framework for data enrichment using data sources on the Web. The salient features of our system include the ability to dynamically select the appropriate sequence of data sources to use, based on the available data. We also discuss approaches for automatically computing the utility of data sources and adapting to their usage. The framework is exposed (internally) as a "platform as a service", accessible via RESTful APIs. We are currently in the process of piloting the system in the energy and resources domain.

REFERENCES

[1] G. Di Lorenzo, H. Hacid, H. Paik, and B. Benatallah. Data integration in mashups. *ACM Sigmod Record*, 38(1):59–66, 2009.

[2] A. Doan, P. Domingos, and A. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *ACM Sigmod Record*, volume 30, pages 509–520. ACM, 2001.

[3] J. Fagan. Mashing up multiple web feeds using yahoo! pipes. *Computers in Libraries*, 27(10):8, 2007.

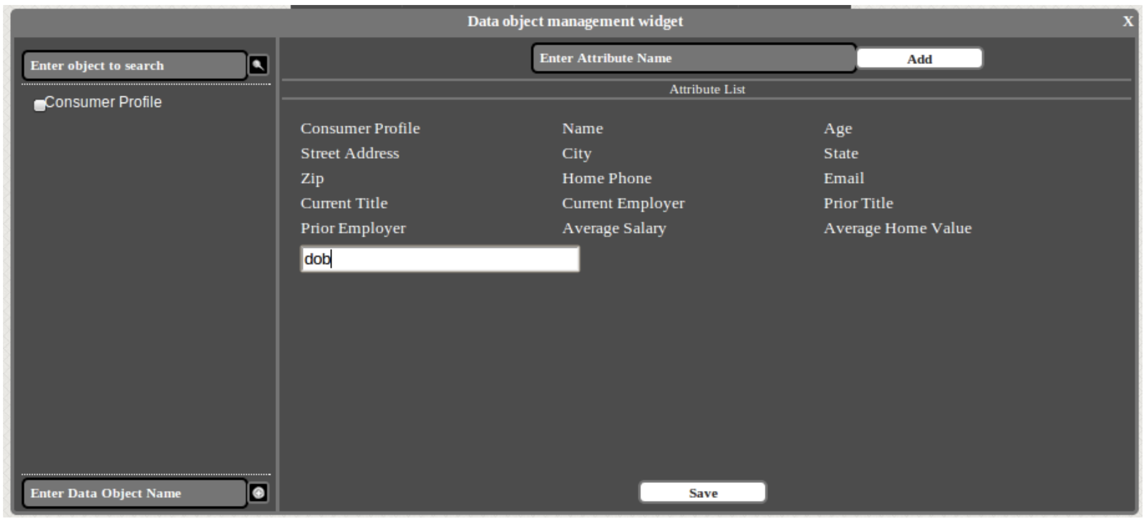
[4] J. Madhavan, P. A. Bernstein, K. Chen, A. Y. Halevy, and P. Shenoy. Corpus-based schema matching. In *IIWeb*, pages 59–63, 2003.

[5] M. Mahemoff. Design patterns for ajax. `http://ajaxpatterns.org/HTTP_Streaming`, 2006.

[6] E. Maximilien, H. Wilkinson, N. Desai, and S. Tai. A domain-specific language for web apis and services mashups. *Service-oriented computing–ICSOC 2007*, pages 13–26, 2007.

[7] D. Merrill. Mashups: The new breed of web app. *IBM Web Architecture Technical Library*, pages 1–13, 2006.

[8] K. Software". Open kapow. `http://kapowsoftware.com/`.



Data object management widget

X

Enter object to search

Consumer Profile

Enter Attribute Name

Add

Attribute List

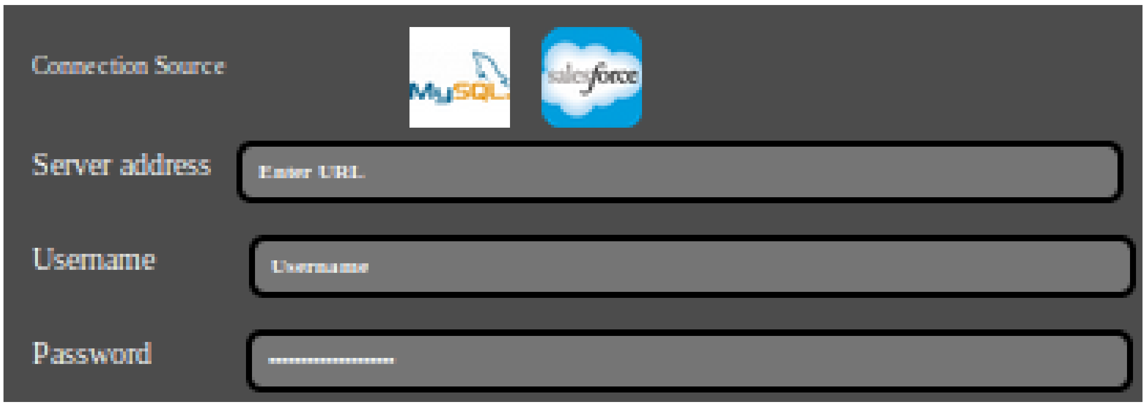
Consumer Profile	Name	Age
Street Address	City	State
Zip	Home Phone	Email
Current Title	Current Employer	Prior Title
Prior Employer	Average Salary	Average Home Value

dob

Enter Data Object Name

Save

Figure 3: Enrichment as a PAAS from Sales Force



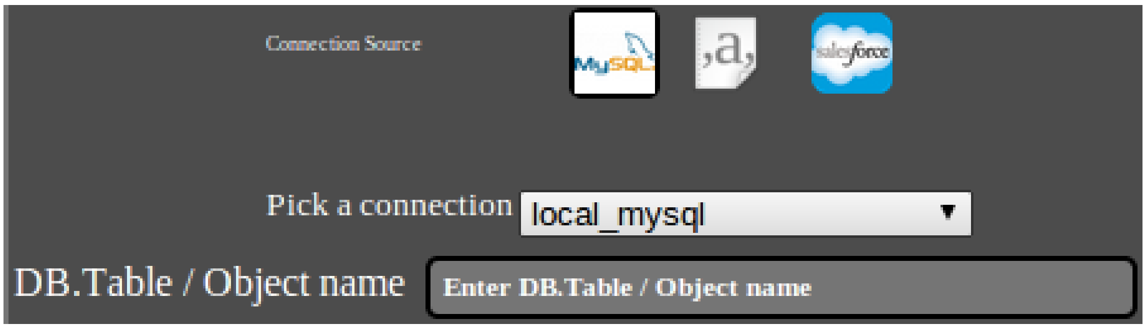
Connection Source

Server address: Enter URL

Username: Username

Password: ••••••••••••••••••••

Figure 4: Enrichment as a PAAS from Sales Force



Connection Source

Pick a connection: local_mysql

DB.Table / Object name: Enter DB.Table / Object name

Figure 5: Enrichment as a PAAS from Sales Force