acmqueue

Other People's Data

Companies have access to more types of external data than ever before. How can they integrate it most effectively?

Stephen Petschulat, SAP

Every organization bases some of its critical decisions on external data sources. In addition to traditional flat file data feeds, Web services and Web pages are playing an increasingly important role in data warehousing. The growth of Web services has made data feeds easily consumable at the departmental and even end-user levels. There are now more than 1,500 publicly available Web services and thousands of data mashups ranging from retail sales data to weather information to United States census data.[3] These mashups are evidence that when users need information, they will find a way to get it. An effective enterprise information management strategy needs to take into account both internal and external data.

TYPES OF EXTERNAL DATA

External data sources vary in their structure and methods of access. Some are well understood and have been a part of data-warehousing flows for many years: securities data, corporate information, credit risk data, and address/postal code lookup. These are typically formally structured, contain the "base" (most detailed) level of data, and are available through established data service providers in multiple formats. The most common access method is still flat files over FTP.

Web services are well understood from a software development perspective, but their relevance to enterprise data management is just emerging. Traditional data service providers such as Dunn & Bradstreet and Thomson Reuters have started offering most of their products via Web services. Hundreds of smaller Web service companies are providing data in the following areas: retail sales, Web trends, securities, currency, weather, government, medicine, current events, real estate, and competitive intelligence.

With Web services comes the ability to add value in the form of functional services. Instead of allowing the retrieval of a flat file of data—effectively a `fetch()` function—it is easy for a data provider to add services on top of the data: conversions, calculations, searching, and filtering. In fact, for most of the smaller start-ups the emphasis has been heavily on the functional services, which typically present a more highly processed subset of the overall data. This can save time and effort when the functions provided are what you need. The data is pre-cleansed, aggregated at the right level, and you don't have to implement your own search.

This can also lead to challenges, however, when the functional interfaces don't match your exact needs. EBay provides marketplace research data on the daily best-selling products in various categories, top vendors, and bid prices for products. This works well if these specific queries are what you want, but if you require a query that eBay has not thought of, you don't have access to the base data to create that custom query yourself.

Another important data source is the Web itself. A great deal of unstructured data exists in Web pages and behind search engines. The area of competitive intelligence has been driving the merging of unstructured and semistructured Web sources with the data warehouse. Competitive intelligence is also an area that is driving the shift from solely back-end data feeds to those all the way through the stack.[1] Structured Web services from Amazon and eBay are one source of specific market and sales information, while technologies from companies such as Kapow and Dapper allow users to turn any external Web page content into semistructured data feeds by mapping some of the visual fields on the page to data fields in the dynamic feed.

Although these tools are starting to make Web scraping easier, most end users still resort to cutting and pasting from competitors' Web sites into spreadsheets in order to gain the insights they need to do their jobs. This is a manually intensive and error-prone approach, but the alternative—sourcing market information, talking to IT, integrating the datasets into the core data-warehouse model, staging, testing, deploying—takes too long, particularly when sources may be changing on a weekly or monthly basis.

ARCHITECTURAL CONSIDERATIONS

External data should be considered and planned for differently from internal data. Much the way distributed computing architectures must account for latency and data failures, a robust data-warehousing plan must take into account the fact that external sources by definition are not in the sphere of control of the receiving organization. They are more prone to unpredictable delays, data anomalies, schema changes, and semantic data changes. That is not to say they are lower quality—plenty of internal sources have the same issues, and data-service companies are paid to provide top-quality data—rather, the communication channel and processing systems are *inter-* rather than *intra-*company, creating an additional source of issues and delays.

Competitive intelligence data (legally) scraped off of publicly available sites not only has to contend with cleanliness issues, but the schema can also change at any time and the publisher has no obligation to notify consumers of that change. If a scheduled report relies on this information, then it will often break, resulting in a blank or incomprehensible report. Even worse, it could result in incorrect numbers, leading to bad business decisions. Data reliability and accuracy need to be thought of as fundamental attributes throughout the data's flow in the organization.

FLEXIBILITY, QUALITY, AND COST

Not all data needs to go through the entire data-warehouse workflow. In information-intensive organizations the IT group can rarely accommodate every user's data needs in a timely manner. Tradeoffs must be made. These tradeoffs can be thought of along the dimensions of flexibility, quality, and cost. In most cases you get to pick two and have to trade off the third.

Flexibility refers to how easily you can purpose the data for the end users' needs. Getting base data from raw flat files maximizes your ability to massage the data further; however, the effort involved is much higher than getting a highly summarized feed from a Web service vendor. For example, the International Securities Exchange historical options daily ticker data for a single stock symbol has more than 80 fields.[2]

```
TRADE_DT,UNDLY,SEC_TYPE,SYM_ROOT,...70+ fields...,ISE_VOL,TOTAL_VOL
20090810,AAPL,1,QAA , ... ,2241,164.72,0.01,1,2
20090810,AAPL,1,QAA, ... ,2347,164.72,0.02,1,2
20090810,AAPL,1,QAA, ... ,3591,164.72,0.03,7,130
20090810,AAPL,1,APV, ... ,2714,164.72,40.7,10,15
...
```

Having all of the base data in CSV (comma-separated values) format provides maximum flexibility; you can derive any information you want from it. If all you want is the high-level summary information, however, you would be better off giving up that flexibility in exchange for a simpler API from a Web service provider such as StrikeIron or Xignite.

GET http://www.xignite.com/xquotes.asmx/GetSingleQuote?Symbol=AAPL

```
<QuickQuote>
  <Symbol>AAPL</Symbol>
  <Last>188.50</Last>
  <Change>7.85</Change>
  <Volume>25,094,395</Volume>
  <Time>4:01pm ET</Time>
</QuickQuote>
```

In the case of securities information, very few IT shops can afford to manage a raw stock feed directly from the exchanges. The monthly uncompressed data for the International Securities Exchange historical options ticker data is more than a terabyte, so even handling daily deltas can be multiple gigabytes on a full financial stream.[2]

Ticker information alone is not very useful without symbol lookup tables and other corporate data with which to cross-reference it. These may come from separate sources and may or may not handle data changes for you—someone has to recognize the change from SUNW to JAVA to ORCL and ensure it is handled meaningfully. Different feeds also come in at different rates. Data can change for technical, business, and regulatory reasons, so keeping it in sync with a usable data model is a full-time job.

Quality is a function of both the source of the data and the process by which it flows through the organization. If a complex formula shows up in hundreds of different reports authored by dozens of different people, the chance of introducing errors is almost certain. Adding up all of the invoices will produce a revenue number, but it may not take into account returns, refunds, volume discounts, and tax rebates. Calculations such as revenue and profit typically rely upon specific assumptions of the underlying data, and any formulas based on them need to know what these assumptions are:

- Do all of the formulas use the exact same algorithm?
- How do they deal with rounding errors?
- Have currency conversions been applied?
- Has the data been seasonally adjusted?
- Are nulls treated as zeros or a lack of data?

The more places a formula is managed, the more likely errors will be introduced.

Cost can be traded off against both quality and flexibility. With enough people hand-inspecting every report, it doesn't matter how many times things are duplicated—quality can be maintained through manual quality assurance. This is not the most efficient way to run a data warehouse, though. Cost and flexibility typically trade off based on the effort necessary to take raw data and turn it into useful data. Each level of processing takes effort and loses flexibility unless you are willing to invest even more effort to maintain both base and processed data.

Of course, you can always keep all of the base data around forever, but the cost of maintaining this can be prohibitive. Having everything in the core data warehouse at the lowest possible level of detail represents the extreme of maximizing flexibility and quality while trading off cost.

External Web services typically trade off flexibility in exchange for quality and cost. Highly summarized, targeted data services with built-in assumptions, calculations, and implicit filters are not as flexible, but they are often all that is needed to solve a specific problem. It doesn't matter that the daily weather average for a metropolitan area is actually sampled at the airport and the method of averaging isn't explicit. This loss of flexibility is a fair trade when all you care about is what the temperature was that day +/-3 degrees.

The following are questions to ask when determining which tradeoffs make sense for a given data source:

- What is the business impact of incorrect data?
- What is the cost of maintaining the data feed?
- How large are the datasets?
- How often does the data change?
- How often does the data schema change?
- How complex is the data?
- How complex and varied are the consumption scenarios?
- What is the quality of the data (how many errors expected, how often, magnitude of impact)?
- How critical is the data to decision making?
- What are the auditing and traceability requirements?
- Are there any regulatory concerns?
- Are there any privacy or confidentiality concerns?

ENTERPRISE DATA MASHUPS

Traditional warehouse life-cycle, topology, and modeling approaches are not well suited for external data integration. The data warehouse is often thought of as a central repository that is the single source of the truth. In reality, it can rarely keep up with the diversity of informational needs of the organization. This results in users resorting to sourcing their own external data, exporting to Excel, and doing their own data joins in spreadsheets.

Self-sourcing of data and user-driven data integration can alleviate some of the burden on IT, but it can also cause problems. It is not always optimal to have mission-critical decisions being made based on data mashups downloaded from the Internet. Few users understand the intricacies of data cleansing, relational joins, and managing slowly changing dimensions. Proper data modeling hasn't gotten any easier.

The range of approaches—from end-user-driven on-screen mashups to centralized data-management teams with detailed master data-management plans—all have their place in a modern data warehousing strategy. A decentralized data strategy has clear quality and consistency tradeoffs, but not all data is equal—sometimes faster, more flexible access to data is more important than getting it perfect. The decision should be made based on an awareness of the quality, flexibility, and cost tradeoffs. These are often linked to where the integration occurs.

INTEGRATING AT DIFFERENT LAYERS IN THE ARCHITECTURE

External data can be incorporated at any of the stages of the enterprise information flow: during ETL (extract-transform-load), in the core data warehouse, at departmental-level data marts, and during end-user consumption via reports, BI (business intelligence) tools, and applications. The two most common stages for data integration are in the initial ingestion or at the final consumption, but all stages merit consideration.

ETL

Enriching data on the way in is a common technique. All the major vendors provide ETL tools to cleanse, transform, and augment datasets on their way into the warehouse. Data problems (errors, omissions, and incompatibilities) are handled in an explicit manner at this layer, leading to high-quality data before it even hits the warehouse. Many of these tools now incorporate some level of Web service integration; however, ETL tools are typically geared toward large-volume batches, whereas most Web-service interfaces tend to assume single or small datasets per request. For this reason, Web services tend to play more prominently as the data gets closer to the user.

Some types of data lend themselves well to a purely enriching role and do not need to augment the existing data model. These are typically good candidates for ETL stage integration. This is a common role for address and geolocation-related data. During the ETL process addresses are compared against an external data source. Zip codes are corrected, street names are normalized, and latitude and longitude fields are added. The external source does not result in new tables in the warehouse with joins to the internal data. The data is just fixed up on the way in, perhaps adding a field here or there.

CORE DATA WAREHOUSE

Integrating at the core model, as is commonly done in the traditional flat-files-via-FTP model, ensures the greatest level of adherence to enterprise-data standards and data-management processes. Data cleanliness issues, interruptions in service, and semantic mismatches are all dealt with in the core model of the warehouse. The currency-conversion tables are consistent across the whole system, promoting the much sought-after “one version of the truth.” For financial information this is often essential.

In many cases, however, this is also the most costly approach to maintain. For rapidly changing business conditions, end users must engage in a dialog with IT and put up with data policies and procedures they may not understand or care about (even if they should). Changing your database schema is harder than changing your reports. If the information is important enough to the users and the system friction too great, then they will find a workaround.

DATA MART

Integrating external data at the mart layer reduces some of the friction typical of core data-warehouse-model integration. Although data-warehouse topologies vary considerably, for this purpose we consider the data mart to be an entity at least partially in the departmental sphere of control. Each mart—whether a separate database instance from the warehouse, an OLAP (online analytical processing) cube, or a completely different database technology—can source and integrate its own feeds, creating a perspective on the data that is relevant to that department or user base. The marketing department may need an external address database for cleansing before doing mass mailings, while the customer support organization may be more interested in geo-tagging for the purpose of analyzing case distribution patterns.

As integration moves closer to the end user, the key issue to be aware of is loss of data control. If each mart integrates the same data in a slightly different way, then the chances are greater that inconsistencies will be introduced. The sales group in North America may be adjusting for refunds to take into account high rotation in its box stores, while the Asia-Pacific group is not. A report that tries to aggregate the results from these two independent data marts may naively sum the numbers, resulting in an incorrect result. There may be good reasons for different rules at the data-mart level, but when values are combined or compared downstream, these differences can cause problems.

BI TOOLS

Most BI tool vendors have now incorporated data-mashup capabilities so end users can join external and internal data. This approach tends to be more end-user driven than doing it at the data-mart layer and often requires no administrative access to the database or formal data-modeling expertise. Vendors' tools vary in their approaches, but typically there are options to do lightweight business-user modeling, as well as on-screen combining of datasets via formula languages. This provides a great deal of flexibility while maintaining the ability to audit and trace usage.

The issues faced at this layer are similar to those of the data mart. Aspects of the data management and integration are still decentralized, which can result in redundant definitions of the same business concepts, increasing the risk of incorrect interpretations. Facilities for auditing and tracing, as well as common business-user-level semantic layers, help overcome some of the issues.

DESKTOP

This end of the spectrum represents the most common mashup scenario. Excel, flat files, macros, and Web app integration are easy to throw together and occasionally are even accurate. While database administrators and warehouse architects may cringe, the bottom line is that when people need information they will find a way to get it. From an information architecture point of view, planning for this eventuality allows you to decide which data belongs where and what the potential impacts will be in an informed way. For some data sources, where accuracy and traceability are not critical, this is a perfectly acceptable choice. As these data sources become more heavily used, though, it may make sense to push them further down the stack to ensure the data-integration work is done by only one person rather than many.

CONCLUSION

There is no "correct" layer at which to integrate external data into the enterprise information flow, rather a set of tradeoffs to consider. The characteristics of the data, its consumption scenarios, and the business context all need to be considered. These factors also need to be reevaluated periodically. As a data source becomes more widely used, economics may dictate centralizing and formalizing data acquisition. Choosing the right integration approach for an external data source can balance the variables of flexibility, quality, and cost while providing end users with timely answers to their business questions.

REFERENCES

1. Boncella, R. J. 2003. Competitive intelligence and the Web. *Communications of the Association for Information Systems* 12: 327-340; http://www.washburn.edu/faculty/boncella/COMPETITIVE-INTELLIGENCE.pdf.
2. International Securities Exchange; http://www.ise.com/.
3. Programmableweb.com; http://www.programmableweb.com/.

LOVE IT, HATE IT? LET US KNOW

feedback@queue.acm.org

STEPHEN PETSCHULAT is a senior product architect in the advanced analytics area of SAP Business Objects.