LARGE ENTERPRISES SPEND a great deal of time and money on “information integration”—combining information from different sources into a unified format. Frequently cited as the biggest and most expensive challenge that information-technology shops face, information integration is thought to consume about 40% of their budget.4,14,16,33,36 Market-intelligence firm IDC estimates that the market for data integration and access software (which includes the key enabling technology for information integration) was about $2.5 billion in 2007 and is expected to grow to $3.8 billion in 2012, for an average annual growth rate of 8.7%.20

Software purchases are only one part of the total expense. Integration activities cover any form of information reuse, such as moving data from one application’s database to another’s, translating messages for business-to-business e-commerce, and providing access to structured data and documents via a Web portal.
Beyond classical information-technology applications, information integration is also a large and growing part of science, engineering, and biomedical computing, as independent labs often need to use and combine each other’s data.

Software vendors offer numerous tools to reduce the effort, and hence the cost, of integration and to improve the quality. Moreover, because information integration is a complex and multifaceted task, many of these tools are highly specialized. The resulting profusion of tools can be confusing. In this article, we try to clear up any confusion by:

- Exploring an example of a typical integration problem
- Describing types of information integration tools used in practice
- Reviewing core technologies that lie at the heart of integration tools
- Identifying future trends.

An Example
Consider a large auto manufacturer’s support center that receives a flood of emails and service-call transcriptions every day. From any given text, company analysts can extract the type of car, the dealership, and whether the customer is pleased or annoyed with the service. But to truly understand the reasons for the customer’s sentiment, the company also needs to know something about the dealership and the transaction—information that is kept in a relational database.

Solving such an integration problem is an iterative process. The data must first be understood and then prepared for integration by means of “cleansing” and “standardization.” Next, specifications are needed regarding what data should be integrated and how they are related. Finally, an integration program is generated and executed by some type of integration engine. The results are examined, and any anomalies must be resolved, which often requires returning to step one and studying the data.

Many technologies are needed to support this process. We introduce a few here and then describe them in greater depth, along with others, in subsequent sections.

The first step toward integrating the text and the relational data is to understand what transactions and other information they contain and how to relate that information to each dealership. The manufacturer next needs to decide how to represent the integrated information. A simple schema—auto model, customer, dealership, date sold, price, size of dealership, date of problem, problem type—might suffice (See Figure 1). But how should each field be represented, and where will the data come from? Let’s assume that the auto model, customer, and dealership information will be extracted from the textual complaint, as will the date of problem and problem type. The relational database has tables about dealerships and transactions that can provide the rest of the information: size of dealership, date sold, and price.

Next, programs are needed to extract structured information from the email or transcription text. These programs’ outputs provide a schema for the text data—the “fields” that can be queried. Matching and mapping tools can be used to relate this derived schema to the target schema. Similarly, matching and mapping must be done for the relational schema. The dealership name extracted from the text can be connected to an entry in the dealership table, and the customer, auto model, and dealership to the transactions table, thus joining the two tables with the textual data.

Programs are needed to align data instances: because it is unlikely that the data formats of the extracted text are identical to those in the relational database, some data cleansing will be required. For example, dealership names may not exactly match. These data-integration programs must then be executed, often using a commercial integration product.

Types of Information-Integration Tools
A variety of architectural approaches can be used to solve problems like the

---

**Figure 1.** Annotators extract key information from email messages. This information is used to probe the relational source data to retrieve additional facts needed for the target schema.

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sirs:</td>
</tr>
<tr>
<td>My Galaxy’s brakes are squealing after only 6 months! I purchased this clunker at Billboy in Oshkosh…</td>
</tr>
<tr>
<td>Sincerely,</td>
</tr>
<tr>
<td>John J. Jutt</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Data and Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DEALERS</strong></td>
</tr>
<tr>
<td>Dealership</td>
</tr>
<tr>
<td>Oshkosh Billboy Ford</td>
</tr>
<tr>
<td><strong>TRANSACTIONS</strong></td>
</tr>
<tr>
<td>TransID</td>
</tr>
<tr>
<td>1234</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto Model</td>
</tr>
<tr>
<td>Galaxy</td>
</tr>
</tbody>
</table>
one above. We summarize these approaches in this section, along with the general types of products used. For information on specific products, we refer the interested reader to IT research companies that publish comprehensive comparisons and to Web search engines (using the product categories we define here as keyword queries).

Data Warehouse Loading. A data warehouse is a database that consolidates data from multiple sources. For example, it may combine sales information from subsidiaries to give a sales picture for the whole company. Because subsidiaries have overlapping sets of customers and may have inconsistent information about a particular customer, data must be cleansed to reconcile such differences. Moreover, each subsidiary may have a database schema (that is, data representation) that differs from the warehouse schema. So each subsidiary’s data has to be reshaped into the common warehouse schema.

Extract-Transform-Load (ETL) tools address this problem by simplifying the programming of scripts. An ETL tool typically includes a repertoire of cleansing operations (such as detection of approximate duplicates) and reshaping operations (such as Structured Query Language [SQL]-style operations to select, join, and sort data). The tool may also include scheduling functions to control periodic loading or refreshing of the data warehouse.

Some ETL tools are customized for master data management—that is, to produce a data warehouse that holds the master copy of critical enterprise reference data, such as information about customers or products. Master data is first integrated from multiple sources and then itself becomes the definitive source of that data for the enterprise. Master data-management tools sometimes include domain-specific functionality. For example, for customer or vendor information, they may have formats for name and address standardization and cleansing functions to validate and correct postal codes.

Virtual Data Integration. While warehouses materialize the integrated data, virtual data integration gives the illusion that data sources have been integrated without materializing the integrated view. Instead, it offers a mediated schema against which users can pose queries. The implementation, often called a query mediator or enterprise-information integration (EII) system, translates the user’s query into queries on the data sources and integrates the result of those queries so that it appears to have come from a single integrated database. EII is still an emerging technology, currently less popular than data warehousing.

Although the databases cover related subject matter, they are heterogeneous in that they may use different database systems and structure the data using different schemas. An EII system might be used, for example, by a financial firm to prepare for each customer a statement of portfolio positions that combines information about his or her holdings from the local customer database with stock prices retrieved from an external source.

To cope with this heterogeneity in EII, a designer creates a mediated schema that covers the desired subject matter in the data sources and maps the data source schemas to the new mediated schema. Data cleansing and reshaping problems appear in the EII context, too. But the solutions are somewhat different in EII because data must be transformed as part of query processing rather than via the periodic batch process associated with loading a data warehouse.

EII products vary, depending on the types of data sources to be integrated. For example, some products focus on integrating SQL databases, some on integrating Web services, and some on integrating bioinformatics databases.

Message Mapping. Message-oriented middleware helps integrate independently developed applications by moving messages between them. If messages pass through a broker, the product is usually called an enterprise-application integration (EAI) system. If a broker is avoided through all applications’ use of the same protocol (for example, Web services), then the product is called an enterprise service bus. If the focus is on defining and controlling the order in which each application is invoked (as part of a multistep service), then the product is called a workflow system.

In addition to the protocol-transla-
tion and flow-control services provided by these products, message-translation services—which constitute another form of information integration—are also needed.

A typical message-translation scenario in e-commerce enables a small vendor (say, Pico) to offer its products through a large retail Web site (Goliath). When a customer buys one of Pico’s products from Goliath, it sends an order message to Pico, which then has to translate that message into the format required by its order-processing system. A message-mapping tool can help Pico meet this challenge. Such a tool offers a graphical interface to define translation functions, which are then compiled into a program to perform the message translation. Similar mapping tools are used to help relate the schemas of the source databases to the target schema for ETL and EII and to generate the programs needed for data translation.

Object-to-Relational Mappers. Application programs today are typically written in an object-oriented language, but the data they access is usually stored in a relational database. While mapping applications to databases requires integration of the relational and application schemas, differences in schema constructs can make the mapping rather complicated. For example, there are many ways to map classes that are related by inheritance into relational tables. To simplify the problem, an object-to-relational mapper offers a high-level language in which to define mappings. The resulting mappings are then compiled into programs that translate queries and updates over the object-oriented interface into queries and updates on the relational database.

Document Management. Much of the information in an enterprise is contained in documents, such as text files, spreadsheets, and slide shows that contain interrelated information relevant to critical business functions—product designs, marketing plans, pricing, and development schedules, for example. To promote collaboration and avoid duplicated work in a large organization, this information needs to be integrated and published. Integration may simply involve making the documents available on a single Web page (such as a portal) or in a content-management system, possibly augmented with per-document annotations (on author and status, for example). Or integration may mean combining information from these documents into a new document, such as a financial analysis.

Whether or not the documents are collected in one store, they can be indexed to enable keyword search across the enterprise. In some applications, it is useful to extract structured information from documents, such as customer name and address from email messages received by the customer-support team. The ability to extract structured information of this kind may also allow businesses to integrate unstructured documents with preexisting structured data. In the example above, the auto manufacturer wanted to link transactional information about purchases with emails about these purchases in order to enable better analysis of problem reports.

Portal Management. One way to integrate related information is simply to present it all, side-by-side, on the same screen. A portal is an entire Web site built with this type of integration in mind. For example, the home page of a financial services Web site typically presents market prices, business news, and analyses of recent trends. The person viewing it does the actual integration of the information.

Portal design requires a mixture of content management (to deal with documents and databases) and user-interaction technology (to present the information in useful and attractive ways). Sometimes these technologies are packaged together into a product for portal design. But often they are selected piecemeal, based on the required functionality of the portal and the taste and experience of the developers who assemble it.

Core Technologies
Extensible Markup Language (XML). In any of the scenarios noted here, an integrated view of data from multiple sources must be created. Often any one of the sources will be incomplete with respect to that view, with each source missing some information that the others provide. In our example, the emails are unlikely to provide detailed information about the dealerships, while the relational data might not have the problem reports. In XML, a semi-structured format, each data element is tagged so that only elements whose values are known need to be included. This ability to handle variations in information content is driving EII systems to experiment with XML.

This flexibility makes XML an interesting format for integrating information across systems with differing representations of data. In some integration scenarios, it may not be

**Figure 2. Screenshot of a mapping tool.**
Information integration is a vibrant field powered not only by engineering innovation but also by evolution of the problem itself.

necessary to define a common schema—data from both sources can be merged into a single self-describing XML document—though in scenarios such as warehousing applications the transformation and fusing of the original data into a well-defined format is required. Still, its flexibility and the ubiquity of free parsers make XML attractive in scenarios with looser requirements, and it is increasingly being used for transferring data between systems and sometimes as a format for storing data as well.³

Schema Standards. It is easier to integrate data from different sources if they use the same schema. This consistency avoids the need to reformat the data before integrating it, and it also ensures that data from all of the sources have mutual meaning.

Even if sources do not conform to a common schema, each source may be able to relate its data to a common standard, either industry-wide or enterprise-specific. Thus two sources can be related by composing the two mappings that relate each of them to the standard. This approach only enables integration of information that appears in the standard, and because a standard is often a least common denominator, some information is lost in the composition.

There are many industry-wide schema standards.¹⁸, ²⁸, ²⁹ Some are oriented toward generic kinds of data, such as geographic information or software-engineering information. Others pertain to particular application domains such as computer-aided design, news stories, and medical billing.

When the schema standard is abstract and focuses on creating a taxonomy of terms, it is usually called an ontology. Ontologies are often used as controlled vocabularies—for example, in the biomedical domain—rather than as data formats.¹², ¹³

Data Cleansing. When the same or related information is described in multiple places (possibly within a single source), often some of the occurrences are inconsistent or just plain wrong—that is, “dirty.” They may be dirty because the data, such as inventory and purchase-order information about the same equipment, were independently obtained. Or they may simply have errors such as misspellings, be missing recent changes, or be in a form that is inappropriate for a new use that will be made of it.

A typical initial step in information integration is to inspect each of the data sources—perhaps with the aid of data-profiling tools—for the purpose of identifying problematic data. Then a data-cleansing tool may be used to transform the data into a common standardized representation. A typical data-cleansing step, for example, might correct misspellings of street names or put all addresses in a common format.¹⁰ Often, data-profiling and -cleansing tools are packaged together as part of an ETL tool set.

One important type of data cleansing is entity resolution, or deduplication, which identifies and merges information from multiple sources that refer to the same entity. Mailing lists are a well-known application; we have all received duplicate mail solicitations with different spellings of our names or addresses. On the other hand, sometimes seeming “duplicates” are perfectly valid, because there really are two different persons with very similar names (John T. Jutt and his son John J. Jutt) living at the same address.

Many data-cleansing tools exist, based on different approaches or applied at different levels or scales. For individual fields, a common technique is edit-distance; two values are duplicates if changing a small number of characters transforms one value into the other. For records, the values of all fields have to be considered; more sophisticated systems look at groups of records and accumulate evidence over time as new data appears in the system.

Schema Mapping. A fundamental operation for all information-integration systems is identifying how a source database schema relates to the target integrated schema. Schema-mapping tools, which tackle this challenge, typically display three vertical panes (see Figure 2). The left and right panes show the two schemas to be mapped; the center pane is where the designer defines the mapping, usually by drawing lines between the appropriate parts of the schemas and annotating the lines with the required transformations. Some tools offer design assistance in generating these transfor-
review articles

ations, which are often complex.26,30

Once the mapping is defined, most tools can generate a program to transform data conforming to the source schema into data conforming to the target schema.15 For an ETL engine, the tool might generate a script in the engine’s scripting language. For an EII system, it might generate a query in the query language, such as SQL. For an EAI system, it might transform XML documents from a source-message format to that of the target. For an object-to-relational mapping system, it might generate a view that transforms rows into objects.23

Schema Matching. Large schemas have several thousand elements, presenting a major problem for a schema-mapping tool. To map an element of Schema 1 into a plausible match in Schema 2, the designer may have to scroll through dozens of screens. To avoid this tedious process, the tool may offer a schema-matching algorithm,31 which uses heuristic or machine-learning techniques to find plausible matches based on whatever information it has available—for example, name similarity, data-type similarity, structure similarity, an externally supplied thesaurus, or a library of previously matched schemas. The human user must then validate the match.

Schema-matching algorithms do well at matching individual elements with somewhat similar names, such as Salary_of_Employee in Schema 1 and EmpSal in Schema 2, or when matching predefined synonyms, such as Salary and Wages. Some techniques leverage data values. For example, the algorithm might suggest a match between the element Salary of the source database and Stnd of the target if they both have values of the same type within a certain numerical range.

But matching algorithms are ineffective when there are no hints to exploit. They cannot map an element called PW (that is, person’s wages) to EmpSal when no data values are available; nor can they readily map combinations of elements, such as Total_Price in Schema 1 and Quantity Unit_Cost in Schema 2. That is, these algorithms are helpful for avoiding tedious activities but not for solving subtle matching problems.

Keyword Search. Keyword search is second nature to us all as a way to find information. A search engine accepts a user’s keywords as input and returns a rank-ordered list of documents that is generated using a pre-built index and other information, such as anchor text and click-throughs.3 A less familiar view of search is as a form of integration—for example, when a Web search on a keyword yields an integrated list of pages from multiple Web sites. In more sophisticated scenarios, the documents to be searched reside in multiple repositories such as digital libraries or content stores, where it is not possible to build a single index. In such cases, federated search can be used to explore each store individually and merge the results.24

While keyword search does integrate information, it does so “loosely.” The results are often imprecise, incomplete, or even irrelevant. By contrast, integration of structured data via an ETL tool or a query mediator can create new types of records by correlating and merging information from different sources. The integration request has a precise semantics and the answer normally includes all possible relevant matches from these data sets, assuming that the source data and entity resolution are correct (both of these are big assumptions). Both precise and loose integration techniques have merit for different scenarios. Keyword search may even be used against structured data to get a quick feel for what is available and set the stage for more precise integration.

Information Extraction. Information extraction is the broad term for a set of techniques that produce structured information from free-form text. Concepts of interest are extracted from document collections by employing a set of annotators, which may either be custom code or specially constructed extraction rules that are interpreted and executed by an information-extraction system. In some scenarios, when sufficient labeled training data is available, machine-learning techniques may also be employed.

Important tasks include named-entity recognition (to identify people, places, and companies, for example) and relationship extraction (such as customer’s phone number or customer’s address). When a text fragment is recognized as a concept, that fact can be recorded by surrounding it with XML tags that identify the concept, by adding an entry in an index, or by copying the values into a relational table. The result is better-structured information that can more easily be combined with other information, thus aiding integration.

Dynamic Web Technologies. When a portal is used to integrate data, it usually needs to be dynamically generated from files and databases that reside on backend servers. The evolution of Web technologies has made such data access easier. Particularly helpful has been the advent of Web services and Really Simple Syndication (RSS) feeds, along with many sites offering their data in XML. Development technology has been evolving too, with rapid improvement of languages, runtime libraries, and graphical development frameworks for dynamic generation of Web pages.

One popular way to integrate dynamic content is a “mashup,” which is a Web page that combines information and Web services. For example, because a service for displaying maps may offer two functions—one to display a map and another to add a glyph that marks a labeled position on the map—it could be used to create a mashup that displays a list of stores and their locations on the map. To reduce the programming effort of creating mashups, frameworks are now emerging that provide a layer of information integration analogous to EII systems, but which are tailored to the new “Web 2.0” environment.2
them but also by trying to simplify the process through better integration of the tools themselves. Information integration is currently a brittle process; changing the structure of just one data source can force an integration redesign. This problem of schema evolution\textsuperscript{32} has received much attention from researchers; but surprisingly few commercial tools that might reduce the cost of integration are available to address the problem. Another cause of brittleness, and another topic of research,\textsuperscript{9} arises from the complex rules for handling the inconsistencies and incompleteness of different sources. One possible approach, for example, is to offer a tool that suggests minimal changes to source data, thereby eliminating many of the unanticipated inconsistencies.

Most past work has focused on the problems of information-technology shops, where the goal of integration is usually known at the outset of a project. But some recent work addresses problems in other domains, notably science, engineering, and personal-information management. In these domains, information integration is often an exploratory activity in which a user integrates some information, evaluates the result, and consequently identifies additional information to integrate. In this scenario, called “dataspaces,”\textsuperscript{17} finding the right data sources is important, as is automated tracking of how the integrated data was derived, called its “provenance.”\textsuperscript{34}

Semantic technologies such as ontologies and logic-based reasoning engines may also help with the integration task.\textsuperscript{34}

Information integration is a vibrant field powered not only by engineering innovation but also by evolution of the problem itself. Initially, information integration was stimulated by the needs of enterprises; for the last decade, it has also been driven by the desire to integrate the vast collection of data available on the Web. Recent trends—the continual improvement of Web-based search, the proliferation of hosted applications, cloud storage, Web-based integration services, and open interfaces to Web applications (such as social networks), among others—present even more challenges to the field. Information integration will keep large numbers of software engineers and computer-science researchers busy for a long time to come.

Acknowledgments

We are grateful to Denise Draper, Alon Halevy, Mauricio Hernández, David Maier, Sergey Melnik, Srijan Raghavan, and the anonymous referees for many suggested improvements.

References


24. OASIS standards; www.oasis-open.org/specs/.


26. OASIS standards; www.oasis-open.org/specs/.


Philip A. Bernstein (philbe@microsoft.com) is a principal researcher in the database group of Microsoft Research in Redmond, WA.

Laura M. Haas (laura@almaden.ibm.com) is an IBM distinguished engineer and director of computer science at the IBM Almaden Research Center in San Jose, CA.

© 2008 ACM 0001-0780/08/0900 55.00