THREE DECADES OF DATA INTEGRATION — ALL PROBLEMS SOLVED?

Patrick Ziegler and Klaus R. Dittrich
Database Technology Research Group, Department of Informatics, University of Zurich
Winterthurerstrasse 190, CH-8057 Zürich, Switzerland
{pzieglerldittrich}@ifi.unizh.ch

Abstract Data integration is one of the older research fields in the database area and has emerged shortly after database systems were first introduced into the business world. In this paper, we briefly introduce the problem of integration and, based on an architectural perspective, give an overview of approaches to address the integration issue. We discuss the evolution from structural to semantic integration and provide a short outlook on our own research in the SIRUP (Semantic Integration Reflecting User-specific semantic Perspectives) approach.

Keywords: Data integration, integration approaches, semantic integration

1. Introduction

In today's business world, it is typical that enterprises run different but coexisting information systems. Employing these systems, enterprises struggle to realize business opportunities in highly competitive markets. In this setting, the integration of existing information systems is becoming more and more indispensable in order to dynamically meet business and customer needs while leveraging long-term investments in existing IT infrastructure.

In general, integration of multiple information systems aims at combining selected systems so that they form a unified new whole and give users the illusion of interacting with one single information system. The reason for integration is twofold: First, given a set of existing information systems, an integrated view can be created to facilitate information access and reuse through a single information access point. Second, given a certain information need, data from different complementing information systems is to be combined to gain a more comprehensive basis to satisfy the need.

There is a manifold of applications that benefit from integrated information. For instance, in the area of business intelligence (BI), integrated information can be used for querying and reporting on business activities, for statistical

analysis, online analytical processing (OLAP), and data mining in order to enable forecasting, decision making, enterprise-wide planning, and, in the end, to gain sustainable competitive advantages. For customer relationship management (CRM), integrated information on individual customers, business environment trends, and current sales can be used to improve customer services. Enterprise information portals (EIP) present integrated company information as personalized web sites and represent single information access points primarily for employees, but also for customers, business partners, and the public. Last, but not least, in the area of e-commerce and e-business, integrated information enables and facilitates business transactions and services over computer networks.

Similar to information, IT services and applications can be integrated, either to provide a single service access point or to provide more comprehensive services to meet business requirements. For instance, integrated workflow and document management systems can be used within enterprises to leverage intraorganizational collaboration. Based on the ideas of business process reengineering (BPR), integrated IT services and applications that support business processes can help to reduce time-to-market and to provide added-value products and services. That way, interconnecting building blocks from selected IT services and applications enables supply chain management within individual enterprises as well as cooperation beyond the boundaries of traditional enterprises, as in interorganizational cooperation, business process networks (BPN), and virtual organizations. Thus, it is possible to bypass intermediaries and to enable direct interaction between supply and demand, as in business-to-business (B2B), business-to-consumer (B2C), and business-to-employee (B2E) transactions.[1] These trends are fueled by XML that is becoming *the* industry standard for data exchange as well as by web services that provide interoperability between various software applications running on different platforms.

In the enterprise context, the integration problem is commonly referred to as enterprise integration (EI). Enterprise integration denotes the capability to integrate information and functionalities from a variety of information systems in an enterprise. This encompasses enterprise information integration (EII) that concerns integration on the data and information level and enterprise application integration (EAI) that considers integration on the level of application logic. In this paper, we focus on the integration of information and, in particular, highlight integration solutions that are provided by the database community. Our goal is to give, based on an architectural perspective, a database-centric overview of principal approaches to the integration problem and to il-

[1] Similarly, processes like government-to-government (G2G), government-to-citizen (G2C), and government-to-business (G2B) are used in e-government.

lustrate some frequently used approaches. Additionally, we provide an outlook to semantic integration that is needed in all integration examples given above and that will form a key factor for future integration solutions.

This paper is structured as follows: In the following Sect. 2, we sketch the problem of integration. Sect. 3 presents principal approaches to address the integration issue. In Sect. 4, the evolution from structural to current semantic integration approaches is discussed. Sect. 5 concludes the paper.

2. The Problem of Integration

Integration of multiple information systems generally aims at combining selected systems so that they form a unified new whole and give users the illusion of interacting with one single information system. Users are provided with a homogeneous logical view of data that is physically distributed over heterogeneous data sources. For this, all data has to be represented using the same abstraction principles (unified global data model and unified semantics). This task includes detection and resolution of schema and data conflicts regarding structure and semantics.

In general, information systems are not designed for integration. Thus, whenever integrated access to different source systems is desired, the sources and their data that do not fit together have to be coalesced by additional adaptation and reconciliation functionality. Note that there is not *the* one single integration problem. While the goal is always to provide a homogeneous, unified view on data from different sources, the particular integration task may depend on (1) the architectural view of an information system (see Fig. 1), (2) the content and functionality of the component systems, (3) the kind of information that is managed by component systems (alphanumeric data, multimedia data; structured, semi-structured, unstructured data), (4) requirements concerning autonomy of component systems, (5) intended use of the integrated information system (read-only or write access), (6) performance requirements, and (7) the available resources (time, money, human resources, know-how, etc.) [Dittrich and Jonscher, 1999].

Additionally, several kinds of heterogeneity typically have to be considered. These include differences in (1) hardware and operating systems, (2) data management software, (3) data models, schemas, and data semantics, (4) middleware, (5) user interfaces, and (6) business rules and integrity constraints.

3. Approaches to Integration

In this section, we apply an architectural perspective to give an overview of the different ways to address the integration problem. The presented classification is based on [Dittrich and Jonscher, 1999] and distinguishes integration approaches according to the level of abstraction where integration is performed.

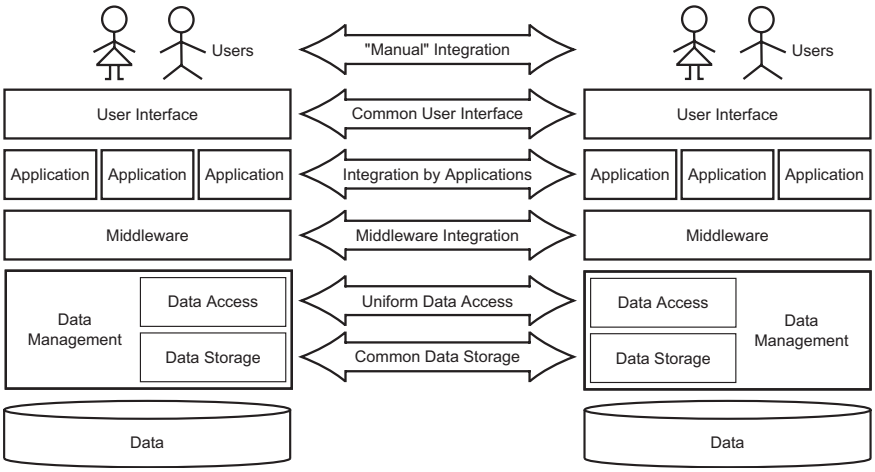

Figure 1. General Integration Approaches on Different Architectural Levels

Information systems can be described using a layered architecture, as shown in Fig. 1: On the topmost layer, users access data and services through various interfaces that run on top of different applications. Applications may use middleware — transaction processing (TP) monitors, message-oriented middleware (MOM), SQL-middleware, etc. — to access data via a data access layer. The data itself is managed by a data storage system. Usually, database management systems (DBMS) are used to combine the data access and storage layer.

Customarily, the integration problem can be addressed on each of the presented system layers. For this, the following general approaches — as illustrated in Fig. 1 — are available:

Manual Integration. Here, users directly interact with all relevant information systems and manually integrate selected data. That is, users have to deal with different user interfaces and query languages. Additionally, users need to have detailed knowledge on location, logical data representation, and data semantics.

Common User Interface. In this case, the user is supplied with a common user interface (e.g., a web browser) that provides a uniform look and feel. Data from relevant information systems is still separately presented so that homogenization and integration of data yet has to be done by the users (for instance, as in search engines).

Integration by Applications. This approach uses integration applications that access various data sources and return integrated results to the user. This solution is practical for a small number of component systems. However, applications become increasingly fat as the number of system interfaces and data formats to homogenize and integrate grows.

Integration by Middleware. Middleware provides reusable functionality that is generally used to solve dedicated aspects of the integration problem, e.g., as done by SQL-middleware. While applications are relieved from implementing common integration functionality, integration efforts are still needed in applications.[2] Additionally, different middleware tools usually have to be combined to build integrated systems.

Uniform Data Access. In this case, a logical integration of data is accomplished at the data access level. Global applications are provided with a unified global view of physically distributed data, though only virtual data is available on this level. However, global provision of physically integrated data can be time-consuming since data access, homogenization, and integration have to be done at runtime.

Common Data Storage. Here, physical data integration is performed by transferring data to a new data storage; local sources can either be retired or remain operational. In general, physical data integration provides fast data access. However, if local data sources are retired, applications that access them have to be migrated to the new data storage as well. In case local data sources remain operational, periodical refreshing of the common data storage needs to be considered.

In practice, concrete integration solutions are realized based on the presented six general integration approaches. Important examples include:

- *Mediated query systems* represent a uniform data access solution by providing a single point for read-only querying access to various data sources. A mediator [Wiederhold, 1992] that contains a global query processor is employed to send subqueries to local data sources; returned local query results are then combined.

- *Portals* as another form of uniform data access are personalized doorways to the internet or intranet where each user is provided with information tailored to his information needs. Usually, web mining is applied

[2]For instance, SQL-middleware provides a single access point to send SQL queries to all connected component systems. However, query results are not integrated into one single, homogeneous result set.

to determine user-profiles by click-stream analysis; that way, information the user might be interested in can be retrieved and presented.

- *Data warehouses* realize a common data storage approach to integration. Data from several operational sources (on-line transaction processing systems, OLTP) are extracted, transformed, and loaded (ETL) into a data warehouse. Then, analysis, such as online analytical processing (OLAP), can be performed on cubes of integrated and aggregated data.

- *Operational data stores* are a second example of a common data storage. Here, a "warehouse with fresh data" is built by immediately[3] propagating updates in local data sources to the data store. Thus, up-to-date integrated data is available for decision support. Unlike in data warehouses, data is neither cleansed nor aggregated nor are data histories supported.

- *Federated database systems (FDBMS)* achieve a uniform data access solution by logically integrating data from underlying local DBMS. Federated database systems are fully-fledged DBMS; that is, they implement their own data model, support global queries, global transactions, and global access control. Usually, the five-level reference architecture by [Sheth and Larson, 1990] is employed for building FDBMS.

- *Workflow management systems (WFMS)* allow to implement business processes where each single step is executed by a different application or user. Generally, WFMS support modeling, execution, and maintenance of processes that are comprised of interactions between applications and human users. WFMS represent an integration-by-application approach.

- *Integration by web services* performs integration through software components (i.e., web services) that support machine-to-machine interaction over a network by XML-based messages that are conveyed by internet protocols. Depending on their offered integration functionality, web services either represent a uniform data access approach or a common data access interface for later manual or application-based integration.

- *Peer-to-peer (P2P) integration* is a decentralized approach to integration between distributed, autonomous peers where data can be mutually shared and integrated. P2P integration constitutes, depending on the provided integration functionality, either a uniform data access approach or a data access interface for subsequent manual or application-based integration.

[3] That is, not within the same transaction but within a period of time that is reasonable according to the particular application requirements.

4. From Structural to Semantic Integration

Database technology was introduced in enterprises since the late 1960s to support (initially rather simple) business applications. As the number of applications and data repositories rapidly grew, the need for integrated data became apparent. As a consequence, first integration approaches in the form of multidatabase systems [Hurson and Bright, 1991] were developed around 1980 — e.g., MULTIBASE [Landers and Rosenberg, 1982]. This was a first cornerstone in a remarkable history of research in the area of data integration. The evolution continued over mediators (e.g., Garlic [Carey et al., 1995]) and agent systems (e.g., InfoSleuth [Bayardo et al., 1997]) to recent ontology-based (e.g., OBSERVER [Mena et al., 1996]), peer-to-peer (P2P) (e.g., Hyperion [Arenas et al., 2003]), and web service-based integration approaches (e.g., Active XML [Abiteboul et al., 2002]).

In general, early integration approaches were based on a relational or functional data model and realized rather tightly-coupled solutions by providing one single global schema. To overcome their limitations concerning the aspects of abstraction, classification, and taxonomies, object-oriented integration approaches [Bukhres and Elmagarmid, 1996] were adopted to perform structural homogenization and integration of data. With the advent of the internet and web technologies, the focus shifted from integrating purely well-structured data to also incorporating semi- and unstructured data while architecturally, loosely-coupled mediator and agent systems became popular.

However, integration is more than just a structural or technical problem. Technically, it is rather easy to connect different relational DBMS (e.g., via ODBC or JDBC). More demanding is to integrate data described by different data models; even worse are the problems caused by data with heterogeneous semantics. For instance, having only the name "loss" to denote a relation in an enterprise information system does not provide sufficient information to doubtlessly decide whether the represented loss is a book loss, a realized loss, or a future expected loss and whether the values of the tuples reflect only a roughly estimated loss or a precisely quantified loss. Integrating two "loss" relations with (implicit) heterogeneous semantics leads to erroneous results and completely senseless conclusions. Therefore, explicit and precise semantics of integratable data are essential for semantically correct and meaningful integration results. Note that none of the integration approaches in Sect. 3 helps to resolve semantic heterogeneity; neither is XML that only provides structural information a solution.

In the database area, semantics can be regarded as people's interpretation of data and schema items according to their understanding of the world in a certain context. In data integration, the type of semantics considered is generally real-world semantics that are concerned with the "mapping of objects in the model or computational world onto the real world [...] [and] the is-

sues that involve human interpretation, or meaning and use of data and information" [Ouksel and Sheth, 1999]. In this setting, semantic integration is the task of grouping, combining or completing data from different sources by taking into account explicit and precise data semantics in order to avoid that semantically incompatible data is structurally merged. That is, semantic integration has to ensure that only data related to the same or sufficiently[4] similar real-world entity or concept is merged. A prerequisite for this is to resolve semantic ambiguity concerning integratable data by explicit metadata to elicit all relevant implicit assumptions and underlying context information.

One idea to overcome semantic heterogeneity in the database area is to exhaustively specify the intended real-world semantics of all data and schema elements. Unfortunately, it is impossible to completely define what a data or schema element denotes or means in the database world [Sheth et al., 1993]. Therefore, database schemas do typically not provide enough explicit semantics to interpret data always consistently and unambiguously [Sheth and Larson, 1990]. These problems are further worsened by the fact that semantics may be embodied in data models, conceptual schemas, application programs, the data itself, and the minds of users. Moreover, there are no absolute semantics that are valid for all potential users; semantics are relative [García-Solaco et al., 1996]. These difficulties concerning semantics are the reason for many still open research challenges in the area of integration.

Ontologies — which can be defined as explicit, formal descriptions of concepts and their relationships that exist in a certain universe of discourse, together with a shared vocabulary to refer to these concepts — can contribute to solve the problem of semantic heterogeneity. Compared with other classification schemes, such as taxonomies, thesauri, or keywords, ontologies allow more complete and more precise domain models [Huhns and Singh, 1997]. With respect to an ontology a particular user group commits to, the semantics of data provided by data sources for integration can be made explicit. Based on this shared understanding, the danger of semantic heterogeneity can be reduced. For instance, ontologies can be applied in the area of the Semantic Web to explicitly connect information from web documents to its definition and context in machine-processable form; that way, semantic services, such as semantic document retrieval, can be provided.

In database research, *single* domain models and ontologies were first applied to overcome semantic heterogeneity. As in SIMS [Arens et al., 1993], a domain model is used as a single ontology to which the contents of data sources are mapped. That way, queries expressed in terms of the global ontology can be asked. In general, single-ontology approaches are useful for

[4] How much similarity is considered as sufficient depends on the particular information need and application area.

integration problems where all information sources to be integrated provide nearly the same view on a domain [Wache et al., 2001]. In case the domain views of the sources differ, finding a common view becomes difficult. To overcome this problem, multi-ontology approaches like OBSERVER [Mena et al., 1996] describe each data source with its own ontology; then, these local ontologies have to be mapped, either to a global ontology or between each other, to establish a common understanding.

Mapping all data to one single domain model forces users to adapt to one single conceptualization of the world. This contrasts to the fact that receivers of integrated data widely differ in their conceptual interpretation of and preference for data — they are generally situated in different real-world contexts and have different conceptual models of the world in mind [Goh et al., 1994]. COIN [Goh et al., 1994] was one of the first research projects to consider the different contexts data providers and data receivers are situated in.

In our own research, we continue the trend of taking into account user-specific aspects in the process of semantic integration. We address the problem how user-specific mental domain models and user-specific semantics of concepts (e.g., "loss") can be reflected in the data integration process. In the SIRUP (Semantic Integration Reflecting User-specific semantic Perspectives) approach, we investigate how data — equipped with explicit, queryable semantics — can be effectively pre-integrated on a conceptual level. That way, we aim at enabling users to perform declarative data integration by conceptual modeling of their individual ways to perceive a domain of interest.

5. Conclusions

In this paper, we gave an overview of issues and principal approaches in the area of integration seen from a database perspective. Even though data integration is one of the older research topics in the database area, there is yet no silver bullet solution and there is none to be expected in the near future. The most difficult integration problems are caused by semantic heterogeneity; they are being addressed in current research focusing on applying explicit, formalized data semantics to provide semantics-aware integration solutions. Despite this, considerable work remains to be done for the vision of truly user-specific semantic integration in form of efficient and scalable solutions to become true.

References

Abiteboul, Serge, Benjelloun, Omar, and Milo, Tova (2002). Web Services and Data Integration. In Ling, Tok Wang, Dayal, Umeshwar, Bertino, Elisa, Ng, Wee Keong, and Goh, Angela, editors, *Third International Conference on Web Information Systems Engineering (WISE 2002)*, pages 3–7, Singapore, December 12–14. IEEE Computer Society.

Arenas, Marcelo, et al. (2003). The Hyperion Project: From Data Integration to Data Coordination. *SIGMOD Record*, 32(3):53–58.

Arens, Yigal, et al. (1993). Retrieving and Integrating Data from Multiple Information Sources. *International Journal of Cooperative Information Systems (IJCIS)*, 2(2):127–158.

Bayardo, Roberto J., et al. (1997). InfoSleuth: Agent-Based Semantic Integration of Information in Open and Dynamic Environments. In *1997 ACM SIGMOD International Conference on Management of Data (SIGMOD 1997)*, pages 195–206, Tucson, Arizona, USA. ACM.

Bukhres, Omran A. and Elmagarmid, Ahmed K., editors (1996). *Object-Oriented Multidatabase Systems: A Solution for Advanced Applications*. Prentice-Hall.

Carey, M.J., et al. (1995). Towards Heterogeneous Multimedia Information Systems: The Garlic Approach. In *5th International Workshop on Research Issues in Data Engineering-Distributed Object Management (RIDE-DOM 1995)*, pages 124–131, Taipei, Taiwan, March 6–7.

Dittrich, Klaus R. and Jonscher, Dirk (1999). All Together Now — Towards Integrating the World's Information Systems. In Masunaga, Yoshifumi and Spaccapietra, Stefano, editors, *Advances in Multimedia and Databases for the New Century*, pages 109–123, Kyoto, Japan, November 30 – December 2. World Scientific Press.

García-Solaco, Manuel, Saltor, Fèlix, and Castellanos, Malú (1996). Semantic Heterogeneity in Multidatabase Systems. In Bukhres, Omran A. and Elmagarmid, Ahmed K., editors, *Object-Oriented Multidatabase Systems. A Solution for Advanced Applications*, pages 129–202. Prentice-Hall.

Goh, Cheng Hian, Madnick, Stuart E., and Siegel, Michael (1994). Context Interchange: Overcoming the Challenges of Large-Scale Interoperable Database Systems in a Dynamic Environment. In *Third International Conference on Information and Knowledge Management (CIKM 1994)*, pages 337–346, Gaithersburg, USA, November 29 – December 2. ACM.

Huhns, Michael N. and Singh, Munindar P. (1997). Agents on the Web: Ontologies for Agents. *IEEE Internet Computing*, 1(6):81–83.

Hurson, Ali R. and Bright, M. W. (1991). Multidatabase Systems: An Advanced Concept in Handling Distributed Data. *Advances in Computers*, 32:149–200.

Landers, Terry and Rosenberg, Ronni L. (1982). An Overview of MULTIBASE. In Schneider, Hans-Jochen, editor, *Second International Symposium on Distributed Data Bases (DDB 1982)*, pages 153–184, Berlin, Germany, September 1–3. North-Holland.

Mena, Eduardo, et al. (1996). OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. In *First IFCIS International Conference on Cooperative Information Systems (CoopIS 1996)*, pages 14–25, Brussels, Belgium, June 19–21. IEEE Computer Society.

Ouksel, Aris M. and Sheth, Amit P. (1999). Semantic Interoperability in Global Information Systems: A Brief Introduction to the Research Area and the Special Section. *SIGMOD Record*, 28(1):5–12.

Sheth, Amit P., Gala, Sunit K., and Navathe, Shamkant B. (1993). On Automatic Reasoning for Schema Integration. *International Journal of Intelligent and Cooperative Information Systems*, 2(1):23–50.

Sheth, Amit P. and Larson, James A. (1990). Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, 22(3):183–236.

Wache, H., et al. (2001). Ontology-Based Integration of Information — A Survey of Existing Approaches. In Stuckenschmidt, H., editor, *IJCAI-2001 Workshop on Ontologies and Information Sharing*, pages 108–117, Seattle, USA, April 4–5.

Wiederhold, Gio (1992). Mediators in the Architecture of Future Information Systems. *IEEE Computer*, 25(3):38–49.