Stefan Deßloch, Albert Maier, Nelson Mattos, Dan Wolfson

Information Integration - Goals and Challenges

1 Introduction

Over the last decades, companies and organizations have gradually built up an IT environment for supporting their informational and operational needs, very often resulting in numerous islands of application processing and information systems. The fragmentation into system islands now poses a problem, and integration as well as interoperability of disparate systems becomes a primary goal in order to increase work productivity, streamline business processes, or allow information exchange and collaboration between departments, organizations and business partners.

The task of business integration, as it is often called, is dominated by the need to reuse existing, possibly autonomous systems, components, or information sources and has to overcome both distribution and heterogeneity of the application and information systems involved. Techniques and approaches to achieve integration can be divided into four major categories [JMP02].

- Portals are used to provide a single entry point to applications and information sources "on-the-glass" through an integrated end user interface that may be personalized and tailored to specific needs or usages. Often, a single sign-on capability is supported that permits the automatic propagation of identifying user information across different systems.
- Business process integration attempts to orchestrate application processing tasks by integrating them as activities into business processes that may span organizational and company boundaries. Workflow management technology and business process modeling are employed to achieve this goal, and XML as well as web services are becoming increasingly important as a means to access applications and exchange information between (sub-) flows in a uniform manner.
- Application integration focuses on wiring together applications or components that need to be aware of each other for various reasons because they work on similar or complementary aspects of the same problem. Middleware technologies for distributed object computing and message-oriented processing are heavily used in this space and application connectors or adapters serve as key concepts to expose the capabilities of (legacy) application systems and make them available to the middleware servers. Data exchange and data transformation are common tasks to be performed here, and XML is becoming more and more important in this context as well.
- Information integration aims at (logically or physically) bringing together various types of data from multiple sources such that it can be accessed, queried, processed and analyzed in an integrated and uniform manner. In a nutshell, the central goal of information integration is to support a holistic view of all the data within an enterprise and, to some extent, across enterprises.

Usually, business integration requires a combination of the above described approaches. As a consequence, providing a holistic view on integration technology across the various tiers becomes increasingly important. Supporting such a view requires common, integrated support for

- tooling that simplifies the building of an integrated IT environment and supports the development and the integration of new components,
- administration of the overall system,
- monitoring services across the different system layers.

The focus of our paper is on information integration (II). Both research and industry are putting significant efforts in providing II techniques and solutions (see [Hä02, IBM02, JMP02, MS02] for an overview of some of these efforts). In this paper, we describe the goals to be achieved by providing an II management system and some challenges that need to be overcome. Section 2 focuses on a description of II requirements. We then introduce a high-level architecture of an II management system that can fulfill these requirements in section 3. Subsequently, we describe different phases of developing a concrete II system, focusing primarily on the role of meta-data services during these phases.

2 Information Integration Requirements

Our goal is to provide a middleware system (II management system) that

- lets applications access the information required as if it were physically stored in a local, single database, regardless of the form and location requested and regardless of the quality of service needs (e.g. timeliness of information),
- offers sophisticated services for searching, transforming and analyzing the integrated information, and
- offers a comprehensive set of services enabling II systems to interact with other middleware systems (e.g. messaging systems and web services).

This goal is tremendously ambitious. Few organizations find that all of the information they need is readily available. Information is typically scattered throughout the enterprise in a wide variety of locations, data stores, and representations. Once you have access to the information, using combinations of data in meaningful ways is itself a challenge. The results may themselves need to be analyzed and/ or transformed into the desired final representation and delivered to the location specified.

The benefits, however, make this set of challenges worthwhile to overcome - across a wide variety of industries. For example, information integration can help in the following scenarios.

- Companies would like to combine real-time, operational and historical information to make better decisions.
- Medical researchers need to combine and correlate diagnosis and treatment data across multiple clinical informa-

tion systems.

- Manufacturers have to determine parts availability across a set of factories.
- Banks want to report total risk exposure across several lines of business

Achieving these benefits requires both an II management system and the development process to use it, which should be supported by an accompanying set of tools. A number of requirements need to be addressed in order to make information integration a reality [DLMWZ02]. These can be described best along three central dimensions that contribute to the complexity of information integration [JMP02, SL90].

- *Heterogeneity of data* characterizes the fact that various degrees of structure and structural variation of the data must be supported. Information integration needs to cover structured, semi-structured, and unstructured data. Such data is typically represented in different data models or formats, including relational data, XML (data-oriented or document oriented), or text documents and multi-media objects (e.g., image, video, audio data) in various formats. In order to achieve integration, flexible mapping and transformation functionality between these data representations is required and dependencies among data in different representations need to be represented. Moreover, approaches for managing such data, as well as manipulation and search/retrieval capabilities, which differ significantly for each kind of representation, need to be supported as well.
- *Federation* and *distribution* of data relates to the fact that information to be integrated is contained in an increasing number of different, possibly autonomous data sources throughout an enterprise. To perform integration, one can *consolidate* information in a single data store, usually resulting in a (mostly read-only) data store that is not connected back to the original sources containing the operational data. This approach may involve additional steps to transform and/or cleanse the data. An alternative approach is the use of *federated* query capabilities that allow to leave the data in the respective data sources and transfer data to the integration engine as needed. Obviously, both extremes have pros and cons, depending on application requirements such as performance, timeliness of data, etc., and both need to be available as alternative solutions. In both cases, the autonomy of existing data sources must not be disturbed and existing applications accessing these sources must not be impacted.
- An increasing desire to produce business *intelligence* from the data is one of the key drivers of information integration. Complex analysis, aggregation, and mining operations over increasingly heterogeneous data needs to be performed in order to harvest valuable information that helps drive business decisions or provides competitive advantage. Analysis can be either applied to single information items or to collections of information items. Operations like scoring or classification fall under the first category, operations like clustering and association mining under the second. Operations of the first category are often used in an *active analysis* context, i.e. they are triggered by new or changing data or other events, possibly incorporating current or historical information. This requires the tight integration of these analysis operations into the II management system. Other analysis scenarios benefit from flexible support of asynchronous processing paradigms, especially if the analysis cannot be performed at transactional speed or if it needs human interaction for example in order to deal with exceptions. Different analysis and mining techniques exist for different degrees of data heterogeneity, and information integration promises to bring these techniques together and enhance the overall outcome by being able to combine and correlate heterogeneous data and analysis results.

Additional requirements arise in the area of interoperability (e.g., with other middleware systems in an overall business integration architecture) and programming interfaces for accessing information integration services. An II management system needs to provide means for a *robust and flexible exchange and transformation of information*, based on *open, platform-independent standards*. In this context, powerful support of *XML* for storage and information exchange, as well as *web-services* is a must. Moreover, an II management system needs to be able to work within a larger context of *business processes and EAI* architectures.

3 Towards an Information Integration Architecture

Figure 1 illustrates a possible architecture to fulfill the requirements listed above. The overall system can be divided into three major component sets [RWKN02].

The *foundation tier* provides capabilities for *storage* of heterogeneous data in various formats. Persistent storage of relational data, which has recently been extended by complex object-relational structures and large objects [ISO99] is complemented by a native XML storage capability for storing XML data and documents. In a similar manner, *search and indexing* capabilities to support SQL are complemented by equivalent capabilities to support XML Query [CCF+02, Ch02] and to search over unstructured data like text or images [BR99]. In contrast to the precise SQL-type search, content-specific search for unstructured data is typically fuzzy and applications are often interested in “the best n” matching objects only [SCK02]. A flexible *crawler* support is available to get to the unstructured data to be indexed, regardless whether this data is stored in the II system, the web or an intranet. Basic *transformations* that map data from one representation into a different one have multiple usages, such as view support (in combination with federated access) or data warehousing and replication. Other types of transformations are used for realizing mappings between different data models. Through the definition of view and transformation capabilities, mappings between relational data and XML can be established which can then be exploited to, for example, query relational data using the XML Query language [SS+00,

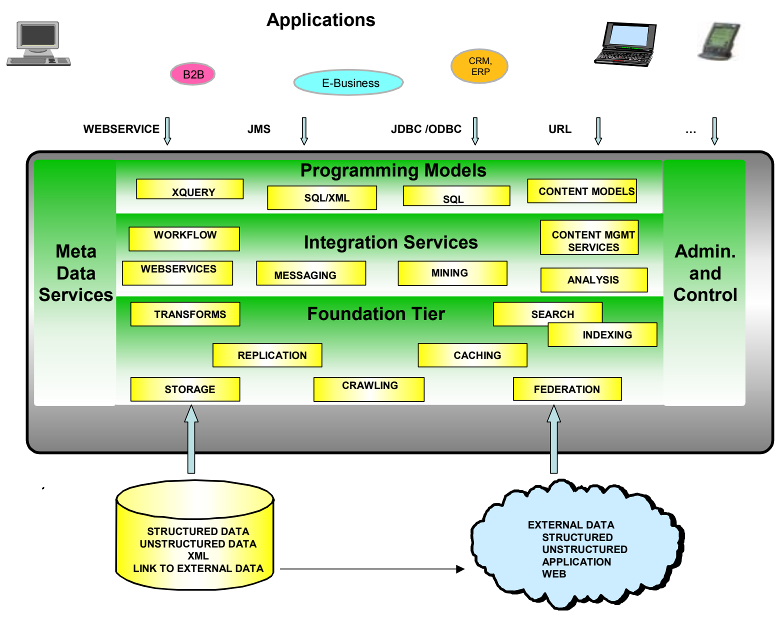

Figure 1: II Management System Architecture

SK+01, FKSSW02]. Access to foreign data sources is supported through *federated* query capabilities, which employ wrapper technology to pull data from foreign sources into the integration engine in an optimized manner [TS97, MKTZ99, HLR02, MM+01, MM+02, Wi93]. While federated search over structured data usually involves optimized processing of possibly complex queries with joins or aggregations over tables residing in multiple data stores, federated search over unstructured data is usually a union-style search ranging over a more loosely coupled federation of search engines, comparable to a free-text search over web content as provided by search engines like Google or Altavista [SCK02]. With respect to external data, the federation model is not the only one to be supported. A *URL-based linking scheme* is highly desirable to be able to point to external data, such as files or documents, without (logically or physically) introducing the content into the data store [RWKN02]. As one example, content management applications can benefit from this functionality by storing metadata or description data of as well as references to media objects in the data store, leaving the referenced content itself outside the data store on special media object servers, e.g. for streaming videos. Various degrees of control for linked data could be supported through such a mechanism without impacting URL-based application processing logic that works with the actual content. The concept of data links in SQL MED [MM+01, MM+02, ISO01] is one example of how links to external data can be supported in the above sense. Additional capabilities supported as part of the foundation tier include *replication* and *caching* capabilities [LKM+02]. Using a combination of federation, replication, and view materialization techniques, data from federated sources may be cached or materialized locally to improve access to the data. Different variations for implementing update operations on such data are possible. The support for data caching as defined above can therefore be seen as a "variable knob" that introduces a wider range of choices for data placement.

The *integration services* tier provides support for numerous capabilities related to producing "intelligence" from integrated information. Sophisticated *analysis* and *mining* technology, both for structured as well as semi-structured or unstructured data, can be employed to achieve this goal [BR99, FPSU96]. One example of this approach is defined in SQL MM, Part 6, Data Mining [ISO03], which describes the integration of data mining capabilities into the SQL engine from a language perspective. Realizations of this standard are already beginning to emerge in data base products [IBM01]. Additional functions are provided that are important in the *content management* area, such as a folder-based models, or support for document-oriented access control, check-in/check-out, and versioning mechanisms. Moreover, support for *workflow* and *messaging* is provided at the integration services layer. This support is twofold in the sense that these services are available inside the II management system to (1) initiate and coordinate information integration tasks and operations (e.g., data exchange and transformation steps) in a robust and reliable manner, but also (2) to ensure that information integration tasks can participate in more general business processes and can interoperate with applications and system components using reliable messaging services. Arbitrary *web services* can be called from within an II system (e.g., a web service returning the stock price of a company can be called from within SQL or XML Query) and arbitrary II operations can be called as a web service (e.g. an II query returning analysis results can be transformed into a web service).

Access to the services of an information integration system is possible through a heterogeneous set of *programming models*, *application programming interfaces*, and *query languages*. In addition to *SQL* as the standard for querying and manipulating structured relational data, support for XML is provided through extensions to SQL defined in the *SQL/XML* standard [ISO03b] as well as through native *XML query* support, which is currently being standardized as XML Query by the W3C [CCF+02]. Moreover, to better support applications working with unstructured data, content-specific programming models and query languages need to be offered as well. Queries, as well as other II service requests, may be submitted to the II management system through various interfaces, such as standard, *call-level APIs* such as ODBC [ODBC] or JDBC [JDBC], but also (either directly or indirectly) via *web services* or *asynchronous cli-*

ent capabilities based on messaging.

Across the three component sets, the II management system offers *administration* and *control services* as well as *tooling* that supports to build the concrete II systems and to add new integration services, e.g. new mining operations. In the same manner, *meta-data* will be discovered, stored, reused and exploited throughout the II management system across the different component sets, as explained in the following section. In addition, metadata services are highly important for upper layers in an overall integration architecture. For example, a business process integration layer needs this meta data to better understand where to get which data, how to combine it, and how to exploit the services offered.

Most of the individual components and technologies included in our II management system architecture have received a lot of attention from a research and product development perspective in the past and the results obtained can therefore be reused or generalized for our purposes. Especially the field of federated databases [SL90, LMR90, MKTZ99], and related areas such as data integration [Ha03, Hä02], schema integration [Co02], and schema matching [RB01] have received a lot of attention over the last years. (The references given here provide an overview of these areas; they are by no means exhaustive.)

However, some of the components and services outlined above still need additional research efforts, and the combined use of these technologies results in additional challenges. Especially areas such as native XML and unstructured document support require attention to arrive at a consistent and flexible data model and query language. Standards-related efforts such as XQuery or WebDAV [WebDAV] are only a beginning and a lot of open problems still need to be solved. For example, an explicit notion of collections of XML documents or fragments, as well as support for updates has been not been addressed in the XQuery specification yet, and research in this area is still fairly immature. Transactional semantics of operations across federated data sources, synchronized backup and recovery, a uniform privacy and security model across a multitude of systems, as well as general query and operational performance aspects in the presence of huge data volumes and increasing numbers of data sources are other areas that still require a lot of focus. Related to performance aspects, the notion of autonomic computing and self-tuning capabilities for II is an especially challenging one, due to the increased complexity of the overall II system.

4 Towards an Information Integration Model

Developing an II system that uses the capabilities described above requires support for various development phases, during which different types of meta-data are collected and created. This support should be provided through a set of tools that accompany the II management system. In the remainder of this section, we take a closer look at these phases and the kind of meta-data required.

There are three basic steps to develop an Information Integration system (see Figure 2): Discover, Design, and Deploy. Once a business problem has been defined, we begin our integration work by first discovering what is available and how it relates to solving our problem. The design phase focuses on designing all of the elements of the solution that need to be crafted yielding a logical design that is then deployed onto one or more physical topologies. Information developed during this process should be maintained as meta-data to facilitate the implementation of future projects.

Discovery is essentially collecting different kinds of metadata. To address the specific information integration needs, we need to find candidate data sources, the catalogs of schemas within those sources, libraries of transformations and pre-existing mappings that could be relevant. Information about data sources, the schemas they contain and other properties can sometimes be discovered automatically but will more often be entered manually. Some sources such as DB2 and Informix have discovery protocols that can simplify the job of locating relevant information. Some organizations maintain directories of data sources using LDAP or other technologies. Such directories can greatly facilitate the discovery of candidate data sources. Discovered metadata (or in some cases links to this metadata) is stored in a registry for sharing, lifecycle management, and analysis. This catalog of interesting metadata is then used during the design phase as then analyzed during the design phase.

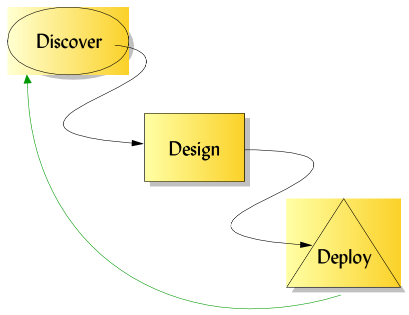

Figure 2: Developing an II System

The goal of the *design* phase is to develop the integration strategies and techniques that can satisfy our integration problem. During integration design, we derive the relationships among the data and their schemas. From an understanding of these relationships we can determine how to map heterogeneous sources into a common logical schema. This mapping may indicate the need for data transformations to convert data types, map values from one domain into another, or even to improve data quality through cleansing. The degree of transformation required is one of the many factors that guide an integration designer to choose to use a federation or movement integration strategy. Table 1 below lists a number of other characteristics. It is often desirable to combine federation and movement - one approach is through the use of caching techniques that use movement technology to maintain a local cache of data within the integration server and at the same time define nicknames for direct, federated access. Applications can dynamically choose to either use the cache or federated access depending upon their requirements. The design activity results in a logical deployment that may be realized on one or more integration servers.

The realization of a logical configuration onto a physical topology takes place during the *deployment* phase. During deployment, the logical configuration defi-

Characteristic	Federation	Movement
Computationally intensive transformations (cleansing, pivots, etc.)	-	+
Access to current data	+	-
A few common queries asked many times	-	+
Many varying or ad-hoc queries	+	-
Information update	+	-
... and many more		

Table 1: Federation and Movement - Design Choices

ned during design is applied to one or more Information Integration servers. Multi-server configurations accommodate large volumes of requests. The provisioning of the II environment includes configuring:

- data capture, transformation, and movement
- federated access to data sources
- caches over federated sources
- global views and functions
- monitoring and maintenance

Monitoring and maintenance procedures may depend heavily upon the particular data sources involved. For instance if there is heavy access to relational systems it will often be of value to periodically gather statistics for use by the Information Integration server.

Given the description of the three major II system design steps, we can now take a look at the meta-data needed to cover the activities in these steps. The model presented here can be seen as a refinement of the five-level schema architecture presented in [SL90]. Heterogeneity in information integration, as discussed above, clearly leads to the question of how the data model of an II management system would look like. Different types of applications need different “data model views” on the same data. For example, a relational or relational + XML is required for (extended) relational applications, an XML-only for applications focusing on data exchange and presentation, or a view containing documents plus XML or structured meta-data for content management applications. Independent of the actual II management system implementation aspects, a data model suitable for II needs to provide all of these aspects at the logical DM and API level, and this heterogeneity also has to be supported in the meta-data model of the II management system.

4.1 Data Source Information

An II meta-data system needs to provide information describing available data sources. These could be sources whose data model matches a model directly supported by the II management system (e.g., relational or XML), or sources that store data in a completely different format, as well as applications that provide operations over encapsulated data. Information about a source can be at various semantic levels. A generic *data source description*, e.g., would contain general information such as name, type, and location (or access model) of the data source, as well as a description that characterizes the data (content) and/or operations available from/on the data source. Because we want to use this information to find interesting data sources across a wide range of sources, the description needs to be at a representation level that can cover all sorts of data sources, regardless of the data model or format.

At a data model-specific level, an (optional) *data source schema* describes the artifacts (objects, operations) that are stored in the data source. This may be based on native meta-data in a format specific to the data store (e.g., a relational schema or an XML schema, or a description of operations and parameter objects in the form of a WSDL file). In addition, a *data source export schema* describes how data source schema objects are made visible in an II system, thereby forming the basis for integrating information of the data source in the II management system. The export schema needs to be specified in a representation that is specific to the data model supported by the intended data source connectivity interface. For example, this would be a relational schema, if the data source exposes the data as SQL/MED foreign tables.

All of the data source meta-data described above is created or collected during the discovery phase of II system development.

4.2 Logical Information Integration Schema

The first major task in the design phase of an II system is to create a *global information schema* that describes a consistent and integrated model of the data or entities coming from the various heterogeneous sources as well as *relationships* and *dependencies* among them. This task usually involves a range of schema integration and schema matching technologies, as well as the use of ontologies to resolve syntactic and semantic heterogeneity. The resulting dependencies are represented in the logical II schema as *mapping information*, which can be further divided into intra-data model (intra-DM) and inter-DM mapping information. *Intra-DM mapping information* describes how objects (probably from a DS export schema) map to a different representation within the same data model, whereas *Inter-DM mapping information* characterizes how the same “object” appears in the different data model. For example, user-defined (or default) mappings of relational tables into XML fall under this category. The mapping information may include additional enhancements, such as explicit rules, algorithms or transformation steps to support (bi-directional) updates of information.

While the discussion so far has focused on the structural aspects, behavioral aspects need to be covered as part of the logical schema as well. Operations supported by foreign data sources (e.g., through interfaces to legacy applications) need to be characterized, and mapping as well as relationship information needs to be represented as well. This information can then be used to provide federation support not just for queries and updates, but also for the invocation of user-defined routines defined in the II model. Further generalizations that support the combined use of foreign functions in flows are an interesting direction to exploit such capabilities [HH02, LR02].

With respect to relationships and dependencies represented at the logical II schema level, it is also important to emphasize that representation needs to abstract from various kinds of implementation techniques, such as view, triggers, etc. that can be used to maintain or implement the relationship.

4.3 Physical Information Integration Schema

The second major task during the design of an II system is to decide how to actually realize the logical II schema. This includes choices whether, for any given data object or representation, federated access or data movement strategies are employed. The same data representation may be available both as a federated data object and materialized/cached, however with different properties in terms of timeliness of the data. Depending on the chosen (combination of) strategies, the same mapping information stored in the logical II schema may be used to create nicknames and views on foreign data, or replication schedules to materialize copies of the data, or both.

In the same manner, update dependencies may be covered by view definitions and/or triggers of different type (e.g., instead-of vs. update triggers), and might even involve interaction with foreign application systems through function invocation or asynchronous messaging to reflect the update in the data source.

For each of the chosen alternatives, additional operational characteristics need to be provided, such as schedule information for replication, caching, or data warehousing operations.

All of that information is stored in the physical II schema so that it is available in the II system catalogs at run-time to support the operational components of the system, and can also be reused if similar schemas have to be deployed on other systems.

So far, our definition of II system development and related meta-data has focused mostly on integration aspects aiming at bridging the heterogeneity of data coming from different sources, or providing data in a different representation or format. An additional dimension now opens up when we start looking at the use of integration services such as analysis and mining techniques, or the use of asynchronous messaging and flow techniques inside the II system, which start to resemble fundamental aspects of an II application built on top of our II system. In principle, the development of such an application needs to follow the same steps outlined above for the II system. In other words, the discovery, development , and deployment now need to be performed to arrive at (potentially application-specific) processing steps or operations that are delegated to the II management system using the above techniques. In a similar manner, those processing steps need to be represented within our II meta-data repository using appropriate representation, which can again be divided into logical and physical aspects.

Despite existing approaches and techniques to represent and manage meta-data and bridge heterogeneity across schema information, the definition of a meta-data model and related meta-data services to address all of the above remains an unsolved problem. In addition to the development of a common meta-data model to cover all of the above aspects, the support of automated meta-data discovery is a major challenge that needs to be addressed to make II management systems successful. A range of tools is required that helps customers to understand which data is available and how to combine it. The use of existing, application-oriented meta-data (originating, for example, from integration systems at the EAI or BPI level) seems to be a promising direction here. Another aspect is the support of modeling and collecting performance- and quality-oriented meta-data to be used for supporting design decisions and performance tuning measures in the II system. Yet another dimension is related to supporting or reacting to schema evolution of data source and II management system schemas.

5 Summary and Conclusions

One of the major challenges in enterprises today is to develop applications that reach out across all the available data inside (and to some extent, outside) the company to combine, query, process and analyze that data, turning it into information assets that help to drive the company's business. We have analyzed the requirements and challenges appearing on the road to realizing information integration management systems that provide the middleware necessary to support such applications. A high-level component architecture for an II management system was presented, consisting of three major component sets: foundation, integration services, and programming model. We then took a look at the overall process to be followed during the development of an II system, using the II management system capabilities, with a special focus on the meta-data required and created in the steps of the development process. The ideas presented are partially reflected in an upcoming commercial system [IBM03], which addresses some of the challenges discussed in this paper and can provide the basis for building an II management system along the lines discussed above.

While a lot of technologies and research results exist to help in the overall task of information integration, there is still a tremendous amount of work to be done to make the II vision a reality. The approaches presented in this paper provide a good starting point to address the apparent challenges, and future research will focus on providing concrete solutions to the remaining open problems we identified.

6 References

[BR99]R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley, 1999.

[CCF+02]D. Chamberlin, J. Clark, D. Florescu, J. Robie, J. Simeon, and M. Stefanescu, XQuery 1.0: An XML Query Language, W3C Working Draft (April 2002). Available at http://www.w3.org/TR/xquery/.

[Ch02]D. Chamberlin, "XQuery: An XML Query Language," [IBM02], 597-615 (2002).

[Co02]S. Conrad: Schemaintegration: Integrationskonflikte, Lsungsanstze, aktuelle Herausforderungen, [Hä02], pp. 101-111, 2002.

[DLMWZ02]S. Deßloch, E. Lin, N. M. Mattos, D. Wolfson, K. Zeidenstein, "Towards an Integrated Data Management Platform for the Web", Datenbank-Spektrum 2(1): 5-13 (2002)

[FKSSW02]J. E. Funderburk, G. Kiernan, J. Shanmugasundaram, E. Shekita, and C. Wei, "XTABLES: Bridging Relational Technology and XML," [IBM02], 616-641 (2002).

[FPSU96]U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, Advances in Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, 1996.

[Ha03]A.Y. Halevy: Data Integration: A Status Report, Proc. BTW 2003, pp. 24-29, 2003.

[Hä02]T. Härder (ed.), Informatik Forschung und Entwicklung, Themenheft Datenintegration, vol. 17, no.3, Sept. 2002.

[HH02]T. Härder, K. Hergula, "Ankopplung heterogener Anwendungssysteme an Fderierte Datenbanksysteme durch Funktionsintegration", [Hä02], 135-148, 2002.

[HLR02]L. M. Haas, E. T. Lin, and M. A. Roth, "Data Integration Through Database Federation," [IBM02], 578-596 (2002).

[IBM01] Intelligent Miner Scoring: Administration and Programming for DB2. International Business Machines Corp, 2001. http://publib.boulder.ibm.com/epubs/pdf/idmr0a00.pdf

[IBM02]IBM Systems Journal 41, no.4, special issue on Information Integration, 2002.

[IBM03]IBM DB2 Information Integrator, see: http://www.ibm.com/software/data/integration/iipreview.html

[ISO99] ANSI/ISO/IEC 9075-2 Information Technology - Database Languages - SQL - Part 2: Foundation (SQL/Foundation), 1999

[ISO01]ISO/IEC 9075-9:2001, Information technology - Database language - SQL - Part 9: Management of External Data (SQL/MED), International Organization for Standardization, June 2001

[ISO03]SQL/MM Part 6: Data Mining.

[ISO03b] XML-Related Specifications (SQL/XML) - Working Draft SQL:200n Part 14, H2-2001-149, WG3:YYJ-012, Jim Melton (Editor)

[JDBC]See http://java.sun.com/products/jdbc/.

[JMP02]A. D. Jhingran, N. Mattos, H. Pirahesh, "Information Integration: A research agenda", [IBM02], 555-562, 2002.

[LKM+02]Q. Luo, S. Krishnamurthy, C. Mohan, H. Pirahesh, H. Woo, B. Lindsay, and J. Naughton, "Middle-Tier Database Caching for e-Business," Proceedings,ACMSIGMODInternational Conference on Management of Data, Madison, WI (June 3-6, 2002).

[LMR90]W. Litwin, L. Mark, N. Roussopoulos: Interoperability of Multiple Autonomous Databases, ACM Computing Surveys 22 (3), pp. 267-293, 1990.

[LR02]F. Leymann and D. Roller, "Using Flows in Information Integration," [IBM02], 732-742 (2002).

[MKTZ99] Nelson Mendona Mattos, Jim Kleewein, Mary Tork Roth, Kathy Zeidenstein: From Object-Relational to Federated Databases. BTW 1999: 185-209

[MM+01] Jim Melton, Jan-Eike Michels, Vanja Josifovski, Krishna G. Kulkarni, Peter M. Schwarz, Kathy Zeidenstein: SQL and Management of External Data. SIGMOD Record 30(1): 70-77 (2001)

[MM+02] Jim Melton, Jan-Eike Michels, Vanja Josifovski, Krishna G. Kulkarni, Peter M. Schwarz: SQL MED - a Status report.. SIGMOD Record 31(3), (2001)

[MS02]R. Meersmann, A.Sheth (eds.), ACM SIGMOD Record, vol.31, no.4, special section on Semantic Web and Data Management, Dec. 2002.

[ODBC]See http://www.microsoft.com/data/odbc/default.htm.

[RB01]E. Rahm, P. Bernstein: A Survey of Approaches to Automatic Schema Matching, VLDB Journal 10 (4), pp. 334-350, 2001.

[RWKN02] M. A. Roth, D. C. Wolfson, J. C. Kleewein, and C. J. Nelin, "Information Integration: A New Generation of Information Technology," [IBM02], 563-577 (2002).

[SCK02]A. Somani, D. Choy, and J. C. Kleewein, "Bringing Together Content and Data Management Systems: Challenges and Opportunities," [IBM02], 686-696 (2002).

[SL90]A.P. Sheth, J.A. Larson: Federated Database Systems for Managing Distributed, Heterogenous, and Autonomous Databases, ACM Computing Surveys 22 (3), pp. 183-236, 1990.

[SS+00]J.Shanmugasundaram, E.J. Shekita, R. Barr, M.J. Carey, B.G. Lindsay, H. Pirahesh, B. Reinwald: Efficiently Publishing Relational Data as XML Documents. VLDB 2000: 65-76

[SK+01]J. Shanmugasundaram, J. Kiernan, E.J. Shekita, C. Fan, J. Funderburk: Querying XML Views of Relational Data. VLDB 2001: 261-270

[TS97]M. Tork Roth and P. Schwarz, "Don't Scrap It, Wrap It! A Wrapper Architecture for Legacy Data Sources," Proceedings, 23rd Conference on Very Large Data Bases, Athens, Greece (August 26-29, 1997).

[WebDAV]See http://www.webdav.org

[Wi93]G. Wiederhold: Intelligent Integration of Information. SIGMOD Conference 1993: 434-437.