Databases in the 3rd Millennium: Trends and Research Directions

Jaroslav Pokorny
Department of Software Engineering, MFF UK Praha
Malostranske nam. 25, 118 00 Praha
pokorny@ksi.mff.cuni.cz

Abstract. A database approach to data of arbitrary types has shown that the architecture of a universal database machine is restricting in a number of cases. There appear special-purpose data servers using now types of hardware devices. Data is moving toward the user, who exploits mobile devices equipped with database functionality. The goal of the paper is to present new trends in databases, particularly in their architectures and to show on cloud computing, data streams, mobile and embedded databases, and databases supporting Web 2.0 some new ideas and possibilities of solution of associated problems. The other goal of the paper is also to point out on actual problems associated with database research.

Key words: database, database architectures, cloud computing, data streams, mobile and embedded databases, Web 2.0

1. Introduction

Despite of a number of attempts with developing universal software for work with files in 60ies, the origins of database history go back to publication of the network model CODASYL from year 1969, subsequently standardized in year 1971. Also the notion of a Database Management System (DBMS) is dated from this time. The hegemony of relational databases, which followed and remains more or less till now, can establish an idea, that it is not possible to explore something better and more expressive in this area. This idea was reversed neither by object databases from 90ties, when their biggest advocates stood for, that relational databases, particularly the SQL standard, may not survive year 2000. However, database development behaved wisely. Relational databases not only survived but even went through certain metamorphosis in extension by some object features. Object-relational (OR) model including previous SQL tables as a special case, even allows a type extensibility which was shown as the step quite essential. If a table can be a set of arbitrary complex rows today, the database designer has an apparatus at disposal enabling to specify and process in principle arbitrary data in a database way. An important question remains, of course, but at what price. We may not forget that any database modelling takes place on a logical level, which is technologically certainly mastered in case of original SQL tables, nevertheless if, e.g., XML data occurs in a table column, it is not clear at first sight, how to process such data, let alone in association with the rest of relational data.

So what are actually trends in databases? Database specialists, who met in Lowell (Massachusetts) in year 2003, emphasized two driving forces of databases advancement [2]: Internet and particular scientific disciplines, such as physics, biology, medicine, astronomy, earth science, and engineering exploited computer processing scientific data, i.e., said in today’s terms, e-science.

While particular issues of e-science are rather remote from usual user needs, in the case of Internet the situation is else. On particular, the Web technology and XML repositories with a database support are in development. From here there is a step to advanced enterprise applications or „socially” focused entering the users into systems, known as Web 2.0 today. Behind the mainstream, ubiquitous computing comes into view, when “smart” home applications are managed by a computer or devices are triggered based on decisions obtained automatically from data gathered in real time. Thus, the notion of a database occurs more than otherwise in completely new connections and challenges.

Practically the same authors as in Lowell met once more in Claremont in 2008 and as starting motto of their report [4] they already stated that database research and industry associated with data management are at a turning point. This situation is influenced by several factors:

- increasing number of applications with big data,
- data analysis as a profit centre,
- ubiquity of structured and unstructured data,
- expanded developer demands,
- architectural shifts in computing (mobile and embedded databases).
In considerations about databases the matter of them is not only in traditional business applications or above mentioned e-science, but also in natural language processing, digital entertainment, analysis of social networks and other data-intensive applications. In this context there is an urgent and wide need for changes towards innovation of the notions of management or data management. Even often there is a need to propose data management systems in a less demanding way than current DBMSs, i.e. from simpler components. Displeasure over of some developers to develop applications „only“ via SQL is a motivation for emerging new programming models and new programming techniques.

Starting with traditional relational databases, an influence of database theoreticians in movement behind traditional databases and data management in enterprise is rather insufficient. There are two promising methodological approaches in development of new DBMS in situation awakened by factors formulated in Claremont: reformation and synthesis. Reformation means to utilize classical data centric approaches, i.e. their basic ideas, and accommodate them in new applications. The synthesis is intended to leverage a maximum from recent developed system architectures concerning, e.g., data integration, information extraction, data privacy and other results usable in new practices.

On the other hand, problems formulated in the past are still alive. They are taken into consideration both in research and practice [11]. Among such repeating challenges we can recognize:

- management of uncertain information,
- data privacy and security,
- data streams and networked data,
- self-tuning and adaptive systems,
- XML databases.

In connection with new applications the classical concept of DBMS architectures emerges too rigid. It is necessary to talk about data management more generally, particularly in connection with data driven application, as, e.g., processing streams of RFID data or data in distributed architectures, where data sources are not connected and a part of the data source is moved near to application, e.g., on a mobile device. Consequently, the notion of embedded database is again considered, with use either in mobile equipment or in an application server. Such embedded databases can lack a lot of functions occurring in commercial DBMS. In practice it can mean a return to technologies of file systems, e.g., to the method ISAM extended by possibilities of querying and synchronization with a backend database.

In the paper we focus on data streams and networked data as well, since they are closely related to mobile and embedded databases.

The paper is a follow-up of the works [21] and [22] from years 2006 and 2007, respectively. In Section 2 we start with sufficiently general database architecture and show its drawbacks in light of new requirements (Section 2.1). In Section 2.2 we justify a need for special-purpose servers for databases specialized on certain data type. New architectures are also influenced by recent advances in hardware, which we shortly mention in Section 2.3. Section 2.4 is devoted to the notion of data space, which generalizes the traditional database concept. As databases are always related to creating applications, they use them, we mention the trend of declarative programming in this context. In conclusion of Section 2 we present a rough enumeration of actual research issues in database architectures. In subsequent Sections 3-6 we focus on the types of database processing, which are coming from recent requirements of practice and offer even already in practice deployable solutions, nevertheless still provide broad possibilities of research. This concerns cloud computing, data streams, mobile and embedded databases, and databases in Web 2.0 environment. Section 7 mentions some practical problems of DBMSs. In conclusions, we summarize trends and research directions as well as some other issues, which are associated with research in the database field.

2. Database architectures

Basic changes and repeating challenges formulated in Section 1 are coming from practical observations. This justifies assertions that current market of relational databases suffers by the following limitations:

- the number of applications working with big data volumes grows considerably, that generates problems with poor price/performance,
- performance of decision-support systems (data warehouses, OLAP) does not work so effectively as transactional processing (OLTP),
- processing scientific data (e-science) is difficult,
application of database techniques in Web 2.0 is not clear, while intuitively eligible,

data life-cycle issues occur: provenance data, versioning, and schema evolution.

Since these limits are associated to a large extent with DBMS architecture, we will try to show, how new approaches deviate from classical universal architecture towards special-purpose servers on one side and towards a more general data space on the second side. We will mention also a role of hardware in new architectures and some requirements on programming applications.

In particular, C. Monash expresses a similar opinion for relational databases. Supposing three their essential elements – rows and columns, predicate logic, and fixed schemas, if any of these three elements is missing or inappropriate, then a traditional relational database management system may not be the best choice.

2.1 Universal architecture

Härder and Reuter published in year 1983 today well-known DBMS architecture (Table 1) based on a mapping model consisting from five abstraction layers [14].

<table>
<thead>
<tr>
<th>Level of abstraction</th>
<th>Objects</th>
<th>Auxiliary mapping data</th>
</tr>
</thead>
<tbody>
<tr>
<td>L5 non-procedural access</td>
<td>tables, views, rows</td>
<td>logical schema description</td>
</tr>
<tr>
<td>L4 record-oriented, navigational approach</td>
<td>records, sets, hierarchies, networks</td>
<td>logical and physical schema description</td>
</tr>
<tr>
<td>L3 records and access path management</td>
<td>physical records, access paths</td>
<td>free space tables, DB-key translation tables</td>
</tr>
<tr>
<td>L2 propagation control</td>
<td>segments, pages</td>
<td>buffers, page tables</td>
</tr>
<tr>
<td>L1 file management</td>
<td>files, blocks</td>
<td>directories</td>
</tr>
</tbody>
</table>

Table 1. The five-layered DBMS mapping hierarchy

In the most general version the architecture is encapsulated together with use of the SQL language in way given in Figure 1.

Layer L2 ensures dividing of linear address space of external memory into different types of pages. Among objects in layer L3 we can find data structures supporting indexing, e.g., B-trees for strings and numbers or R-trees for spatial data. Going upwards, the objects and associated operations become more complex, also additional integrity constraints can appear.

In the L5 layer we remind for a while a significant position of the OR model (SQL:1999, 2003) and its extendible type system. For example, nesting of data structures using constructs ARRAY and MULTISET supports processing scientific data (matrices, vectors, etc.). On the other hand, we are running into difficulties with realization of new, complex data types. In particular, it concerns

1. implementation of multimedia data types, such as VITA (video, image, text, audio), or XML,
2. integration of data types through a query in SQL.

The concept of multi-layered architecture considers its ideal implementation with a machine, which has \( k \) layers. In practice, the number of architecture layers is often reduced due to a more effective performance.

---

For VITA types it is possible to use common maximally the layer L1, the others have to be implemented for each type separately. Solutions with universal servers so popular in 90ties were based on adding loosely coupled additional modules (components) for each new data type (see Figure 2). Nevertheless, the problem of effectiveness of the resulted system was never resolved. It concerns issue (2) and, in narrower sense, a query optimization. An efficient solution of these problems results usually in modification of the DBMS kernel that is very expensive, time-consuming and tending to errors.

2.2 Special-purpose database servers

The database authority M. Stonebraker said that a universal architecture applicable to many different types of tasks poses an elephant, which is powerful, that is his main, unfortunately sole advantage. Or in other way: universal architecture of today’s databases – means one size fits all [25].

New DBMS architectures prefer rather separate database servers tailored to requirements of particular applications types. Not only functionality is considered, but also the way how data is stored. Candidates for special-purpose database servers can be found in areas as, for example:

- OLAP,
- data warehouses,
- native XML databases,
- text retrieval,
- data streams processing,
- wireless sensor networks,
- databases of scientific data.
In given areas there are produced large and complex datasets that require more advanced database support, than that one offered by current products. New mechanisms for information integration are needed. Newly emerging wireless sensor networks contain data obtained by collecting in real time and processed in real time (data streams). Data transmission poses a continuous streaming data, which is indexed in time dimension to be filtered to individual (possibly mobile) users. It seems that usual DBMSs are not effectively applicable in this area.

A special category is used for scientific data. Despite of the fact that several DBMS prototypes exist in this area from the past, only in year 2008 (in Asilomar, USA) requirements how should such DBMS look at were seriously formulated [26]. A new data model (multidimensional arrays) is needed and, particularly, a possibility to design own user operations. Authors of the subsequent proposal called the new DBMS SciDB and they are starting now with its development.

Other variant is also hybrid servers used, e.g., for SQL/XML implementation, when besides the relational server there is a native XML implementation. For example, IBM DB2 9 (for Linux, UNIX, and Windows) is a hybrid data server with optimized management both for XML and for relational data.

It seems, that universal and hybrid servers are effective especially when the data requirement can be simply decomposed into relatively independent parts evaluated separately in database kernel and in the module built for the given special data type.

But that is not all. Authors of [26] argue that the current architecture of relational DBMSs is not even appropriate for business data processing. They designed and implemented a new DBMS engine for OLTP applications built on so called H-store. Their experiments show that this implementation is a factor 82 faster on TPC-C benchmark.

Summarizing this subsection, the current approaches to DBMS architecture turn in towards

- improvements of present database architectures,
- development of special-purpose DBMSs (with native implementation of selected data types),
- radically new database architecture designs.

2.3 Progress in hardware

Not only requirements on processing ubiquitous data have an impact on development of database architectures, but also a progress in devices, i.e. in hardware. This involves mainly mobile and sensor devices and storages. Due to cheap and high-capacity flash storage (e.g., 32 GByte) as well as their low power consumption, flash disks are an ideal storage media for sensor devices and mobile phones. It is expected that the flash storage capacity will reach 128 GByte by year 2012. Also the cost of these storages is coming down (in year 2007, a 1 GByte flash chip cost 5$).

From the data processing view, flash storages are interesting by higher number of I/O operations/s, due to non-existence of disk seeks and rotational latency. For techniques of data processing on these storages it is new, that page write cost is typically higher than read.

Other changes concern exploiting many-core CPU and high capacity memories. Most significantly, the trends in storages reflect on development of storage technologies in current mobile and embedded systems (see Section 5). In every respect it means to rethink old data techniques and accommodate them to new facts.

Let pay also an attention to energy demands of database hardware. Authors of [17] indicate that energy cost is the third largest cost in a data centre (after server hardware, and power distribution and cooling costs). The servers and data centres in the US alone consumed about 61 billion kWh at a cost of $4.5 billion in year 2006. A consequence of these facts and obviously of current crisis scenarios is, that some new approaches take into account techniques of energy efficient query processing in DBMSs.
2.4 From universal database to a data space

It is observable that data appear rather in loosely coupled data sources, some of them are managed by a relational DBMS, some of them not. Thus, data sources are becoming a part of a dataspace. Such a new abstraction is described in [10]. Despite of traditional (enterprise) databases with a given schema the goal is to manage a rich collection of structured, semistructured, and unstructured data, spread in more enterprise repositories and on the Web. To control such data space of course does not mean other data integration approach. Data in data space rather coexists; semantic integration is not a necessity here, in order to operate parts of the system. Figure 3 adopted from [10] shows a categorization of current solution of data management in two dimensions. Administrative proximity indicates how close various data sources are in terms of administrative control. „Near“ means, that sources have the same or at least coordinated control. Semantic integration is a measure, how closely the schemas of different data sources match.

The development of corresponding software – Data Space Support Platform (DSSP) – is announced as the main item of the agenda of data management field, or as we also say - data engineering. DSSP does not suppose a complete control over data in data space. Instead, it allows the data to be managed by participant system, which belongs to. But it provides new services over aggregates of these systems. A part of DSSP is a catalogue containing descriptions of participants in data space and relationships between them. Participants are sources as relational databases, repositories of XML data, text databases, web services, sensors generating data, etc. Data space supposes to model any kind of relationship among participants, in the extreme case to specify a schema mapping between two participants. Or a simple dependence is concerned, when one source is only a version of another one. Integration in data space evolves, rather by demand, whereas data is still accessible, e.g., in the simplest way via key words.

Querying in data space includes both data and metadata, e.g., „Find emails sent me during the IDEAS 2010 conference days“, „Which sources has the attribute length?“ Approximate answers are supposed, or „best-afford“ results in the case of, e.g., unavailability of some sources. DSSP should enable come in if necessary to use of the query language of the data sources (if possible) or open up query mechanisms through a paid gate. In [13] authors present three examples of applications motivating the need for data space: Personal Information Management, scientific data management, structured queries and content on Web. An example of such structured content is GoogleBase. Then a challenge for research is to find sufficiently powerful searching mechanisms, that accept key words and chose relevant structured sources, and further to combine appropriately answers extracted from structured and unstructured data.

\[2 \text{http://www.google.com/base/}\]
It would be interesting to compare these proposals with current development of the Web, possibly the semantic one, which is based on ontologies and URLs, or its successors, e.g., Web 2.0. In a data space there are probably considered looser integrated data sources not necessarily the web ones.

Another vision is Web of Things, which absorbs embedded devices into Web by using usual web protocols. A more advanced solution is based on the success of Web 2.0 mashup applications [12].

2.5 Declarative programming of applications

The development of DBMS architectures influences also programming applications. Data centric approaches gave an impact on programming. Often the programmers are obtaining requirements on conceptually simple calculations applied on huge amount of data. A good example is calculation of the PageRank algorithm, which is working with a matrix and vectors of dimensions tens billions. Among existing declarative programming tools we include

- programming model Map-Reduce [8],
- new declarative languages, often grounded in Datalog,
- XQuery [28].

The use of declarative languages usually reduces code size by orders of magnitude, while also enabling distributed or parallel execution. For example, Map-Reduce, inspired by functional programming, enables to implement multiplication of sparse matrices by a vector in a natural way or to calculate multiple joins of tables simultaneously.

Another category of languages comes out from DATALOG. For example, in [16] the DATALOG is used to a program analysis. A special case of these languages is declarative, domain-oriented languages inspired by databases. Languages for programming sensor networks, specification of network protocols, events monitoring, program analysis, computer games, and languages for design of multi-layered web applications, e.g., Hilda [30], fall there.

The use of XQuery for programming is theoretically justified by its computational completeness [15]. A prominent advocate of this approach is Daniela Florescu (Oracle Corp.). Possibly not everybody will agree with her vision, since it highly emphasizes the significance of XML and XQuery for forthcoming future of approaches to applications development:

- information will stay in XML (i.e. no rows in tables or objects),
- imperative languages, as we know them today (Java, C, C++, C#), will be gone,
- applications will be programmed in an extension of the XQuery language or in any case in a declarative dataflow/workflow language specially designed for XML processing, SOA, and enabling universal information flow (similarly to the way as stored procedures do it by extending a simple SELECT-FROM-WHERE statement in the relational world).

While no standardized update mechanisms are at disposal for XQuery [29], prototypes XML programming languages are in development, see, e.g., [6].

2.6 Research problems

- Research problems concerning database architectures include [5]:
  - designing systems for clusters of many-core processors, which will exhibit limited and non-uniform access to off-chip memory;
  - exploiting remote RAM and flash memories as persistent media rather than only magnetic disks;
  - treating query optimization and physical data layout as a unified, adaptive, self-tuning task to be carried out continuously;
  - compressing and encrypting data at the storage layer, integrated with data layout and query optimization;
  - designing systems that embrace non-relational data models, rather than "shred" their data into tables;
  - trading off consistency and availability for better performance and scale out to thousands of machines;
  - designing power-aware DBMSs that limit energy costs without sacrificing scalability.

3 At conference CIDR 2003 (Conference on Innovative Data Systems) Florescu obtained an award for the "idea the world is least ready for".
3. Cloud computing

Though there is not yet an exact definition of cloud computing, most agree, that this way of computing provides services, where shared (external) computing sources call potentially diverse requirements given by users, whereas expectations of users are also very diverse. It is a case of a radical movement of computer processing, data storing as well as a deployment of software from end-users of local servers towards networks, data centres, shortly said to completely different infrastructures, which frees enterprises from large IT investment and, moreover, enables to gain powerful computing sources and services.

Two kinds of cloud infrastructures are distinguished:

- computing instance on demand (we add other computers),
- computing capacity on demand (we extend a memory, the number of computing tasks is increasing).

Both types of infrastructures are technically based on clusters of loosely coupled computers. A representative of the first kind is Elastic Compute Cloud\(^4\) (EC2) produced by Amazon, the computation model MapReduce is of the second kind. Based on this model, Google offers Google File System (GFS) and application called BigTable, on which Google operates its applications Google Earth and Google Finance. Techniques used by Google are also used in a more available open source system Hadoop\(^5\) (see its Hadoop File System). Other example is PNUTS\(^7\) – a massive parallel and geographically distributed database system for Yahoo!’s web applications. PNUTS provides a data repository organized as hashed or ordered tables with low latency appropriate for processing big amount of concurrent requirements including updating and querying. Behind these approaches we can find again the idea of declarative programming.

In work\(^1\) there are discussed possibilities of this way of computing for application data management. At the first sight it reminds paradigms of the application service provider (ASP) and database-as-and-service (DaaS). Indeed, today’s early cloud services offer much less than traditional DBMS. The have limited mechanisms for querying and data consistence control that pushes more programming burden on developers. A well-known example is Amazon’s SimpleDB\(^6\), providing a simple service for creating, storing, and querying datasets. Even no database schema is necessary to define in advance. On the other hand, in the case of using services of unrestricted SQL this approach solves the problem of non-predictable workload of SQL servers to some extent. An appropriate candidate from database area for cloud computing seems systems for data analysis (OLAP). Since data is mainly read in these applications, many problems appearing in usual transactional processing fall off.

4. Data streams

Data streams occur in many modern applications as, e.g., network monitoring, collecting records of transactions and their analysis, sensor networks, application exploiting RFID tags, call records, patients’ records, financial applications, Web logs, traffic data analysis, click-streams, etc. Special-purpose Stream Database Management Systems (SDMS) are in development – for a detailed overview see, e.g., \([3]\). These systems are not based primarily on loading data into a database, except transiently for the duration of certain operations. Consider, for a while, the most known data streams containing RFID tags generated from events obtained by RFID readers. Streams of RFID data are filtered, aggregated, transformed, and harmonized so that events associated with them can be monitored in real-time. A typical example is tracks carrying product parts labelled by RFID tags, which pass through a RFID gate. Events processing is data centric, it requires processing in memory based on rules and a query processing.

Although these systems proved to be an optimal solution for on the fly analysis of data streams, they cannot perform complex reasoning tasks. While reasoners are year after year scaling up in the classical domains of computing, reasoning upon rapidly changing information has been neglected or forgotten. Reasoning systems suppose static knowledge, and do not manage „changing worlds”, one can only update the ontological knowledge and then repeat the reasoning tasks. Authors of \([8]\) propose reasoning on data streams, what means an unexplored, yet high impact, research area. Moreover, the approach is multi-disciplinary, since it aims to development of tools integrating methods, models, and algorithms used in data streams and reasoning systems.

\(\text{http://aws.amazon.com/ec2/}\)
\(\text{http://hadoop.apache.org/}\)
\(\text{http://aws.amazon.com/simpledb/}\)
5. Mobile and embedded databases

A sufficient motivation for changes in approach to data management seems to be applications as healthcare, insurance or field services, which use mobile devices. In the case data is in backend databases, and/or generally somewhere on Web, or accessible, e.g., in form of cloud computing (see Section 3). In many applications databases are needed in a very restricted version directly on mobile devices, e.g., on sensors (sensor databases), for mobile CRM (Customer Relationship Management) or state meters of different sources (electricity, water, gas).

Embedded databases are a special case of embedded applications. Typically, they are single application databases that are not shared with other users, their management is automatic, and their functions are substantially limited. Their self-management includes at least backups, error recovery, or reorganization of tables and indices. Such applications run often on special devices, mostly mobile. We talk then about mobile and embedded DBMS.

Based on recent requirements, authors of [20] summarize characteristics of future database technology appropriate for mobile devices as follows:

- **embedding** – embedded DBMS components form an integral part of the application or the application infrastructure, often without any claims on management. Such database should be usable as a state-alone for multiple transactions and application.
- **minimization** – the size of embedded database software should not unnecessarily increase application software.
- **component approach** – a possibility to configure the database functionality according to application requirements and minimize thereby the size of application software.
- **in-memory database** – assertion of renewal of ideas of databases in main memory, even without requirements on any persistence. It means to use special query techniques and indexing methods.
- **database portability** – this feature is enforced by the requirement on simplicity of application installation. Portability should be not depreciated by complexity of database installation.
- **synchronization with back-end data source** – data cashing for such synchronization requires to include a mid-tier (application server) into resulted architecture, which ensures data cashing, operations on data, and synchronization with backend databases.

Microsoft proves that it is not only a vision, when it announces a free developer’s version of SQL Server 7.0 for embedding in mobile applications. So called Data Engine based on the technology of SQL Server 7.0 is intended for developers using Visual Studio tools for building mobile applications linked to central data repositories and for design of systems that can be “up-sized” to SQL Server, as data stores grow. Data Engine is not full-featured DBMS as SQL Server 7.0, it lacks tools for data management and other functions. Developers can embed this software into their applications and have not develop own database software.

Both Oracle and IBM announced the decision to develop new mobile databases, which will allow remote workers to communicate with enterprise information system via laptops and handheld PCs. It is appropriate for applications, where it is necessary to integrate, e.g., structured and unstructured data. Sybase iAnywhere shopping software for mobile and embedded databases (e.g., iAnywhere Mobile Office, RFID Anywhere) is considered the market leader in this area for several years.

6. Databases and Web 2.0

In year 2007 a panel discussion was held at conference VLDB 2007 [5] with goal to examine the relationship between data management and Web 2.0. Creating content and its sharing on the Web 2.0 offer a number of opportunities for use of database technologies. A new distinguished feature is the creation of user communities and their participation in adding value to applications, which they use in the framework of Web 2.0. There were considered the following issues:

**Accessibility.** It is associated with the growth of amount of various devices (cameras, sensors) producing data, which can be accessible via Web 2.0. Devices mediating the access to Web then enter more and more into every day life of users, into homes, when user can combine own produced data (clips, photos, object coordinates, texts) with data sources on the Web or share it trough communities with other users. One of the requirements, which naturally appears in this situation, is simplicity of user interfaces, which require, in contrast to previous databases, possibilities of combining data (primarily of VITA types).
Collaborative efforts. If data heterogeneity was solved in the past rather as an exception, for Web 2.0 it is a rule. For the database community it means to continue again this never-ending challenge, which is becoming the main theme now. From the view of database schemas we can recognize a total decentralization, when metadata arises ad hoc and is integrated on demand, without central authority, definitely with simple interface not more complex than spreadsheet.

Declarative mashups. One of the logical consequences of creating Web content is its combination through new types of services. This is now described by the notion of mashup. A typical example of mashup is internet maps, in which data from other applications are integrated. New methods are looked for how to build mashups particularly on the client side. It is clear now, that with the help of traditional programming languages (Java, JavaScript, PHP) is hardly to achieve such a goal and that a declarative approach is necessary. The traditional SQL, though declarative, is not the winner tool for this purpose. Combination of different mashups created by various technologies and with different interfaces is difficult itself. Thus, mashup specification must be supported by an appropriate infrastructure.

Data quality. The problems concerned lead to the most sceptical view on the Web 2.0 until now. On one hand, it is the world’s most comprehensive knowledge base, on the other hand it means a major data quality crisis. The main challenges in this problem area include trust analysis, authorization, authenticity, and other quality measures in social networks. Web 2.0 is not a Web of facts, but a Web of opinions [23]. This is documented by spamming in blogs, publishing contributions under someone else’s name, etc. Another example concerns vandalism on Wikipedia. Measurements presented in [24] document that 11% of Wikipedia’s articles had been vandalized at least once. Inaccuracy and distortion is inherited in building mashups and mashups over mashups, as they depend on the data quality of their underlying data sources.

Social relevance. Networks built by communities have a social content besides structural (graph-oriented) properties, which includes collaborative content tagging and reviewing. Users can view hotlists of popularity, list of the best books or songs. Consequently, it leads to modelling users, their interests and behaviour. S. Amer-Yaha remarks, that while information retrieval relies on the assumption of static content and dynamic user requirements, information filtering addresses static user requirements and dynamic content in social networks both content and user interest are dynamic. This presents a clear opportunity for rethinking the notions of information retrieval, query processing and content recommendation.

In summary, requirements on new data management need to be extended by adding a social dimension (reviews, tags, and critiques, explicit and implicit social ties) and incorporation of recommendation mechanisms.

Methodologies. A special issue is methodological aspects of database research in Web 2.0 environment. How to proceed the research? How to reach real social data in places, where it is stored? How to find fundamental Web 2.0 problems? An inspiring opinion for web data management is based on the following idea. Considering the Web as the largest database of unstructured data, a structure extraction becomes important. Such extraction, anyhow imperfect, could improve collaboration with the user. The users should obtain simple tools for extraction, integration, contribution of data to existing data, querying, visualization of data and services, and for networking in communities.

It is characteristic for Web 2.0, that its users adopt quickly new techniques and the associated research follows with a delay, often after finished, wide-spread tools (see already existing networks with Facebook, Linkeld, MySpace, etc). This reminds research in context of natural or social sciences, which is often delayed behind the evolving reality. Web 2.0 concerns not only technologies, but also social aspects (e.g., political relevance of blogs), for which database researchers or in general computer scientists are nowise prepared.

In conclusions there are formulated main areas, where data management technologies are relevant to Web 2.0:

- building scalable backend databases for Web 2.0 services,
- building platforms on which others can build other services,
- constructing new data-oriented services for Web 2.0 and studying user behaviour to improve services.
7. Situation in practice

In the world of commercial databases we expect two products, namely Microsoft SQL Server 2010 and Oracle 12g, which could be interesting from point of view of innovations.

We focus at least on the first one:

**Compression.** Remind that the version SQL Server from year 2008 brought a compression both for backup and data compression as a new feature. New version will build on data warehouse improvements occurring in version 2008.

**Self-managed services.** The main emphasise is put on managed services for DBMS, conceived as a self-tuning system now. These will influence the activities and duties of a database administrator.

**Policies.** Emphasis on policies is another bid change. Some polices have been already available in SQL Server 2005, but in SQL Server 2010 they will exist as default.

**Support of Web 2.0.** A better integration of email and Web 2.0 environment is also expected. SQL Server will enable, e.g., to use the Twitter.

An interesting question for a DBMS user is the choice of concrete product. One thing is observable always. A permanent competition among vendors leads to continual upgrades u users. For ITC people this invokes stress and puts a pressure on a budget as well. Database vendors add more functionality into their systems to extend the range of customers. Studies of some consulting companies (e.g., Monash Research [19]) document, that databases users should carefully consider, whether products from specialized vendors are not more appropriate for them. For example, the can be optimized for certain data area, or they are simpler and less expansive (e.g., Progress or Informix SE), or even open source (e.g., MySQL or PostgreSQL). Such systems are called **mid-range DBMSs**. They are often also much easier to administrate and for transactional databases are usually fairly sufficient. Vice versa, their compatibility with big databases can be worse in the case of portability for existing DBMS. An exclusion is, e.g., Postgres Plus Advanced Server of EnterpriseDB, which is strongly compatible with Oracle.

It does not always concern the fact that special-purpose databases have other set of functions, other types of indices, etc. A hardware platform can be crucial. Consider again an example of choice of database for data warehouses. For data warehouses ranged in hundreds of TByte the version Oracle on platform SMP (Symmetric Multi Processing) with architecture shared-eveything will probably not a good alternative, but rather cost-effective MPP (Massively Parallel Processing) with architecture shared-nothing, i.e., a technique with linear scalability. Except Oracle the product Medison of Microsoft’s company DATAlegr0 is interesting in this application category.

Oracle or SQL Server does not constitute also the most appropriate alternative for embedding a database in an application. Other examples have been mentioned in Section 2.

8. Conclusions

We have introduced some trends in databases, as they appear from the point of view of practice needs. It seems, that the research is often controlled directly from positions of big companies, as it is indicated by results of Google, Amazon, and Yahoo! How do reflect these new trends in database research in academic sphere? As a relevant source of information about situation of database research we can take the choice of topics proposed by prestigious scientific database conferences for year 2010. For example, conference EDBT (Extending Database Technology) invites submissions concerning models, algorithms, and architectures, e.g., in these areas:

- availability, reliability, and scalability,
- benchmarks and performance evaluation,
- biomedical databases,
- complex event processing,
- data duration, annotation and provenance,
- data streams and publish-subscribe systems,
- databases design and tuning,
- digital libraries, museums and archives,
- heterogeneous databases and semantic interoperability,

---

7 [http://www.datalegr0.com/](http://www.datalegr0.com/)
parallel, distributed, P2P, and grid data management,
privacy and security in trust databases,
user interfaces and data visualization,
web information and services.

Apparently, these topics are in accordance with challenges specified in this paper.

Acknowledgement: This research has been supported by the grant of GACR No. GA201/09/0990.

9. References


